

Treatment of Multi-label and Multi-object Images in Deep Learning: A Focus on Image Annotation and Retrieval

(ディープラーニングにおける複数ラベルおよび複数オブジェクト
画像の処理：画像アノテーションと検索に焦点を当てた研究)



Jonathan Mojoo

Supervisor: Professor Takio Kurita

Department of Information Engineering

Hiroshima University

This dissertation is submitted for the degree of
Doctor of Philosophy in Information Engineering

Graduate School of Engineering

September 2021

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university.

This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Jonathan Mojoo

September 2021

Acknowledgements

Firstly, I would like to express my sincere gratitude to Professor Takio Kurita of The Graduate School of Advanced Science and Engineering at Hiroshima University, for supervising me throughout the research that culminated in this thesis. He introduced me to the field of machine learning, giving me the necessary foundation that helped me find direction in my research. His insightful guidance and encouragement throughout my study allowed me to advance in my research, and his comments on this thesis helped to improve its quality. Likewise, I would like offer special thanks to Professors Junichi Miyao, Bisser R. Raytchev, and Hiroaki Mukaidani of the Graduate School of Advanced Science and Engineering at Hiroshima University for their suggestions, which further served to polish up this manuscript. I am also grateful to my colleagues from the Pattern Recognition Lab. at Hiroshima University, with whom I shared research ideas and daily experiences that helped me to grow both as a researcher and as a person. I would also like to acknowledge the Ministry of Education, Culture, Sports, Science and Technology - Japan (MEXT) for awarding me the scholarship that financed my study and my stay in Japan.

I do not forget my family and friends, whose constant encouragement, emotional support and prayers kept me going when things got tough. I extend my sincere gratitude to them. I would like to offer my heartfelt thanks to my wife who has been a constant pillar of support and strength, and my motivation at various times throughout my study. I am truly thankful for her understanding in my times of occupation, her encouragement in my times of weakness, and her shared rejoicing in my times of success. Above all, I thank God Almighty for giving me purpose, granting me wisdom and blessing the work of my hands.

Abstract

Deep learning has seen an unprecedented boost in recent years, and has advanced artificial intelligence and its applications in many fields. One of the areas where research in deep learning has yielded amazing results is computer vision, which deals with the understanding of image data by computer-based algorithms. This is largely due to the emergence of convolutional neural networks (CNN), specialized networks that can learn and extract important features from images for various tasks. This is very significant, as digital images have become a part of our everyday life and continue to take up a large part of people's day-to-day activities, from social media to medicine and engineering applications. Our work is predicated on the fact that most real-world images tend to have metadata that includes multiple labels as opposed to a single label per image. This is because a single image database can be used by different individuals, in different usage scenarios, each possibly requiring different types of information and classification schemes. It is therefore important for deep learning algorithms to learn how to effectively learn various tasks effectively on multi-label image datasets. We argue that multi-label learning requires special treatment and poses unique challenges that are not encountered in single label learning. We propose improvements to standard multi-label deep learning algorithms in two computer vision tasks: multi-label image annotation and multi-label image retrieval. In multi-label annotation, we focus on the recurring problem of incomplete or incorrect labels in multi-label image datasets. This affects the ultimate performance of deep learning algorithms, since these models are greatly affected by the accuracy of the data they are trained on. We propose two separate solutions to solve this problem. We propose a regularization-based approach, that extends the loss function in-order to leverage label co-occurrence patterns inside the

training data, and label similarity outside the training data. Additionally, we propose using a separate graphical model (called a restricted Boltzmann machine) to learn label dependencies and reconstruct label vectors to supplement training images with extra labels. We evaluate our methods on 3 benchmark datasets and show that they lead to a more accurate multi-label annotation model than the baseline CNN model. On the image retrieval task, we propose a simple and intuitive function for calculating pairwise similarity of multi-label images, based on the intersection-over-union of their label vectors. This allows for a clear distinction between different similarity levels of image pairs in the training data, and leads to more accurate ranking of retrieval results at test time. We also tackle the idea of multi-object images, which are multi-label images with several instances of a particular concept appearing in a single image. We introduce corresponding changes in the loss function and neural network architecture to reflect the varying levels of similarity. We further improve upon this method by introducing label reconstruction using a variational auto-encoder, to correct errors in the similarity function introduced by missing labels. We show that our method achieves better results on both multi-label and multi-object retrieval tasks than the baseline, and a several other recently proposed methods.

Table of contents

List of figures	xiii
List of tables	xv
1 Introduction: Multi-label Images in Deep Learning and Neural Networks	1
1.1 Deep learning and computer vision	1
1.2 The significance and challenges of multi-label image data	2
1.3 Our approach to multi-label annotation	4
1.4 Our approach to image retrieval for multi-label and multi-object images	6
2 Multi-label Annotation with Missing Labels	9
2.1 Introduction	9
2.2 Related works	13
2.2.1 Multi-label learning with label relations	13
2.2.2 Approaches using specialized architectures	14
2.2.3 Regularization techniques in neural networks	14
2.2.4 Hayashi’s quantification method-type III	16
2.2.5 Laplacian eigenmaps	17
2.2.6 Word2Vec	18
2.3 Unified Graph Laplacian Regularization Approach	18
2.3.1 Training with missing labels	18
2.3.2 Internal distributions of label similarity	19
2.3.3 External distributions of label similarity	20
2.3.4 Regularization graph with unified co-occurrence distributions	20

2.3.5	Training with a combined objective function	21
2.4	Label Reconstruction by RBM Approach	22
2.4.1	Train Restricted Boltzmann Machine with missing labels	22
2.4.2	Reconstruction of labels by Pre-trained RBM model	24
2.4.3	Training the CNN using RBM reconstructed labels	25
2.5	Experiments	26
2.5.1	Datasets	26
2.5.2	Parameter Settings	27
2.5.3	Performance measures	27
2.5.4	Results and Discussion	28
2.6	Conclusion	33
3	Multi-label and Multi-object Deep Metric Learning for Image Retrieval	37
3.1	Introduction	37
3.2	Related Work	39
3.2.1	Deep Metric Learning	39
3.2.2	Deep Hashing and Quantization	40
3.2.3	Multi-label image retrieval	41
3.2.4	Relation networks	42
3.2.5	Triplet selection strategies for triplet networks	42
3.2.6	Losses for multi-label deep metric learning	44
3.2.7	Variational auto-encoders	46
3.3	Proposed Method I	48
3.3.1	Triplet generation scheme	48
3.3.2	Feature extractor	50
3.3.3	Feature aggregation	51
3.3.4	Triplet quantization	51
3.3.5	Triplet loss with an adaptive margin	53
3.4	Proposed Method II	55
3.4.1	Triplet generation scheme	55
3.4.2	Improved triplet loss	56
3.4.3	Label reconstruction using a variational auto-encoder	57

Table of contents	xi
3.5 Experiments	58
3.5.1 Datasets	58
3.5.2 Training	60
3.6 Results and Analysis	61
3.6.1 Method I	61
3.6.2 Method II	66
3.7 Conclusion	67
4 Conclusion	71
4.1 Summary	71
4.2 Findings	72
4.3 Limitations and Further Considerations	73
References	75
Appendix A Retrieval Examples	87

List of figures

1.1	Some of the tasks constituting computer vision. We focus on the highlighted tasks	2
1.2	An example showing how users may require different classification schemes for the same image collection	3
2.1	Binary relevance, implemented by a CNN	10
2.2	Example demonstrating the consistency of predicted label (arctic) with the given labels (bear, snow, polar, tundra), and its relevance to the image content	12
2.3	Illustration of the proposed approach	15
2.4	Illustration of a simple weighted graph with sample similarities . . .	17
2.5	The architecture of the proposed CNN+RBM model for multi-label annotation. Before input into the CNN, the missing labels are firstly reconstructed by the pre-trained restricted Boltzmann Machine. . . .	22
2.6	Restricted Boltzmann Machine	23
2.7	CNN architecture	27
2.8	of Corel5k dataset and predicted labels with probability $\geq 0.1\%$. Given labels: cat, tiger, tree Baseline CNN: cat, tiger, flowers, grass Proposed: cat, tree, grass, tiger, forest	31
2.9	Example image from NUS-WIDE-LITE dataset and predicted labels with probability $\geq 0.1\%$. Given labels: lake Baseline CNN: clouds, mountain Proposed: clouds, mountain, valley, sky	32

2.10	Example image from EspGame dataset and predicted labels with probability $\geq 0.1\%$. Given labels: guitar, woman, music, hair, sing Baseline CNN: guitar, man, music, light, hair, sing, singer Proposed: guitar, music, hair, sing, singer, man, light, band	33
2.11	Comparison of the micro-F1 score of the proposed method with that of the baseline CNN using different missing-label rates on the Corel5K dataset (probability $\geq 0.1\%$)	34
2.12	Comparison of the micro-F1 score of the proposed method with that of the baseline CNN using different missing-label rates on the Corel5K dataset (Top-3)	35
2.13	Comparison of the micro-F1 score of the proposed method with that of the baseline CNN using different missing-label rates on the Corel5K dataset (Top-5)	35
3.1	An illustration of hard, semi-hard, and easy triplets	44
3.2	An illustration of a basic auto-encoder	46
3.3	The architecture of a variational auto-encoder	48
3.4	An illustration of our architecture	49
3.5	An graphical illustration of the definition of the similarity function .	54
3.6	Examples of samples with missing labels in the NUSWIDE dataset .	58
3.7	Two sample images from the NUS-WIDE dataset	59
3.8	Two sample images from the COCO dataset (The number in brackets is the number of occurrences of each object in the target image) . . .	60
3.9	Retrieval examples for two variants of our method and the baseline method (using 32-bit quantized features)	65
3.10	Examples of label supplementation (text in bold indicates new labels suggested by the VAE)	67
3.10	More examples of label supplementation (text in bold indicates new labels suggested by the VAE). Originally dissimilar images become partially similar	68

List of tables

2.1	The example of cross tabulation.	16
2.2	Datasets and their corresponding labels	27
2.3	Comparison of the micro-F1 score on the Corel5k dataset	29
2.4	Comparison of the micro-F1 score on the NUS-WIDE-LITE dataset.	29
2.5	Comparison of the micro-F1 score on the EspGame dataset.	30
2.6	Model sensitivity to parameters α and β on the Corel5k dataset (micro-F1 score for top-3 labels %).	30
2.7	Comparison of the micro-F1 score on the Corel5k dataset	33
3.1	Mean average precision (MAP) for different number of bits on two benchmark datasets. The underlying architecture is AlexNet.	62
3.2	Mean average precision (MAP) for our method and several of its variants. The underlying architecture is AlexNet	62
3.3	Mean average precision for our method and the best competing method with VGG16 as the underlying architecture.	63
3.4	Mean average precision@0.50 (MAP@0.50) for our method and several of its variants on the NUS-WIDE dataset (AlexNet architecture).	63
3.5	Mean average precision (MAP) for method II compared to method I. The underlying architecture is AlexNet.	68
3.6	Effect of threshold parameter t on model performance (AlexNet).	69

List of Abbreviations

DNN	Deep Neural Network
CNN	Convolutional Neural Network
CBIR	Content Based Image Retrieval
RBM	Restricted Boltzmann Machine
RNN	Reccurent Neural Network
HQ-III	Hayashi's Quantification method-type III
RN	Relation Network
AE	Auto-Encoder
VAE	Variational Auto-Encoder
KL	Kullback Leibler
SGD	Stochastic Gradient Descent
FC	Fully Connected
MAP	Mean Average Precision

Chapter 1

Introduction: Multi-label Images in Deep Learning and Neural Networks

1.1 Deep learning and computer vision

Since its emergence, deep learning has steadily gained popularity in recent years because of the expressive power and vast applicability of deep neural networks (DNNs). The "deep" in deep neural networks refers to the hierarchical nature of these artificial neural networks, where layers of parameters learn to make decisions based on features learned from lower layers. This enables neural networks to learn and represent complex functions that can perform many pattern recognition and generation tasks. One particular type of DNN, called a convolutional neural network (CNN) [1–4] has given rise to a large family of algorithms categorized as *computer vision*. As illustrated by Figure 1.1, computer vision is a blanket term for algorithms aimed at such tasks as image classification [3, 5, 4, 6], image segmentation [7–10], object detection [11–16], activity recognition [17–20], image de-noising [21–24], super resolution [25–27], and image retrieval [28–31] just to mention a few. Some of these fields usually overlap or are extended to form a sub-field. For example, object detection includes some element of classification, and activity recognition could be considered as classification applied to video data.

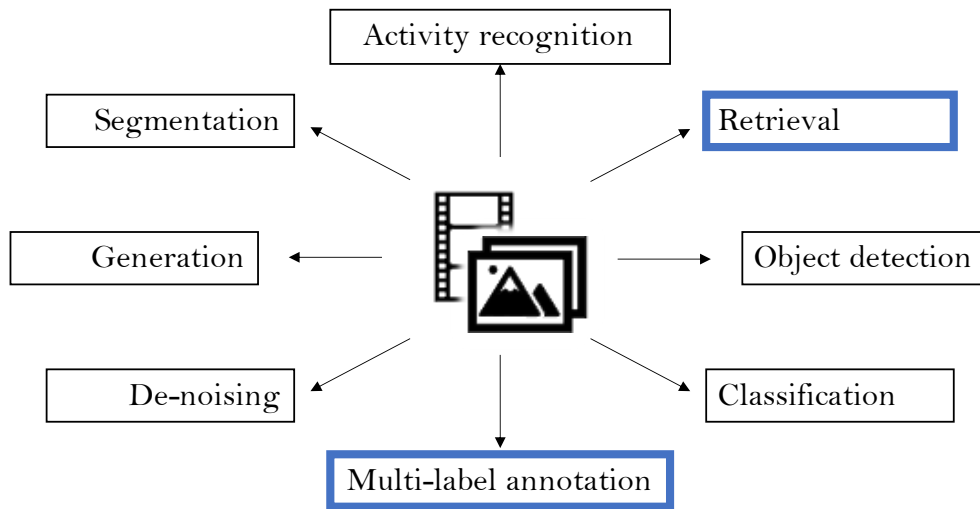


Fig. 1.1 Some of the tasks constituting computer vision. We focus on the highlighted tasks

CNNs have enabled neural networks to achieve near-human performance in computer vision tasks, and improved their real world applicability in fields such as autonomous driving, medical imaging, automatic surveillance, biometric imaging, visual media storage and retrieval, and many more. However, there is still on-going research to improve their applicability to various types and formats of visual data. One of the areas with room for improvement is the accuracy of CNNs on multi-label data.

1.2 The significance and challenges of multi-label image data

Most real life image collections, such as social media images and medical images cannot be easily labeled with a set of mutually exclusive labels such that each image has a single label. This is because a single image usually conveys multiple concepts or multiple types of information. Additionally, users may require different image categorization schemes depending on their needs. As an example, imagine a person searching a collection of photos for a particular type of event. This requires that the images be categorized according to event type (e.g. wedding, party, concert

etc.). However another person might want to search the same database for images featuring a particular person. This requires a second annotation scheme based on which people appear in the photos.






					
user1	animal	person	animal	animal	person
user2	reindeer		bird	shark	
user3	jungle	outdoor	sky	underwater	indoor

Fig. 1.2 An example showing how users may require different classification schemes for the same image collection

Being able to assign multiple labels to each image is usually a necessary feature of image databases. This flexibility is especially important for images that are created without an initial application purpose in mind e.g. social media images. In some cases, it is also possible to for a new categorization scheme to be added to a pre-existing image database if the usage scenario changes. Given the necessity of multi-label image data, it imperative for deep learning algorithms to be able to effectively learn on multi-label image datasets, to enable fast automated organization and retrieval of large image databases.

However, there are some challenges associated with multi-label annotation. Due the spread of digital imaging technology and the rise of social media, the difficulty and cost of producing and sharing digital images has decreased substantially. Therefore, more digital images are created by typical everyday users than professional users. Furthermore, multi-label image collections might be created by several users over an extended period of time, which means individual preference becomes a factor. This leads to inconsistent, inaccurate or incomplete labeling. Even when these datasets are created in a professional setting, it is difficult to ensure consistency and completeness, especially if the number of possible labels is large. Since the performance of a deep learning model is limited by the accuracy of the ground-truth

data it is trained on, there is need to address these labeling issues when training CNNs. This leads us to the following observations, with regards to deep learning for multi-label image data:

1. It is important to recognize the difference between single-label and multi-label image data, and find unique characteristics that can be leveraged when training deep learning models on multi-label images.
2. There is need to take into account the unique challenges that arise when training deep learning models on multi-label data

1.3 Our approach to multi-label annotation

Though multi-label annotation is in some cases considered an extension of the traditional classification problem, there are some fundamental differences. In traditional multi-class image classification we assume that given an example image, there is only one single correct label. Therefore, the assignment of a label to an image automatically assumes the exclusion of all other labels in the label set. In deep learning, this is usually implemented by a soft max activation function applied to units of the last layer of a CNN, as follows:

$$\sigma(\mathbf{y})_i = \frac{e^{y_i}}{\sum_{j=1}^N e^{y_j}} \quad (1.1)$$

where y_i is the output from the i -th unit in the output layer, and N is the number of labels. This can be interpreted as the relative strength of the i -th signal compared to the rest of the units. The objective would then be to maximize the output of the softmax function for the correct label, given by the ground-truth label, through a cross-entropy loss function. In a standard multi-label annotation formulation, a sigmoid function is applied to the output of each unit, as follows:

$$\sigma(\mathbf{y})_i = \frac{1}{1 + e^{-y_i}}, \quad (1.2)$$

Standard multi-class classifiers using soft-max activations are easy to train and evaluate. Additionally, there are numerous reliable datasets for developing and testing models. Multi-label annotation models on the other hand, have fewer datasets to work with and even these are have somewhat unreliable labeling. However, multi-label data has an added advantage because aside from image-to-label mappings, models can also learn from inter-label relations and co-occurrence dependency, hence improving prediction accuracy of each label. It is this property that we leverage in chapter 2 to circumvent the missing label problem of multi-label datasets described in section 1.2.

The standard cross entropy objective, coupled with the sigmoid activation shown in Equation 1.2, does not explicitly leverage label relations and co-occurrence patterns in the training data. We therefore introduce an explicit mechanism to enforce label relation-awareness in a multi-label annotation CNN. First of all, we adopt Hayashi’s quantification method type II, to derive a vector for each ground-truth label, such that the Euclidean distance of two vectors reflect the similarity of their corresponding labels. We then use this similarity in a Laplacian regularization term that enforces identical probability predictions for similar labels. We show that this alone can boost the prediction of relevant missing labels and test time accuracy in general. Using internal label co-occurrence patterns helps to smooth out the effect of incomplete and inconsistent labels, but it is ultimately still reliant on the goodness of the training data. Therefore, we introduce an additional Laplacian regularization term that incorporates label similarity based on Word2vec embeddings from an external corpus. Evaluated on three benchmark datasets, our method using regularization with internal similarity achieves an improvement of up to 1.4%, and our unified approach using internal and external label similarity achieves an improvement of up to 1.48% over the baseline.

Additionally, we present an approach that tries to solve the missing label problem in the pre-training stage. We train a generative graphical model called a restricted Boltzmann machine to capture label interactions and dependencies in the training data, and use it to predict additional labels for the training images. We show that the model is able to recover missing labels, and append additional relevant labels

to training samples. We then use these new labels to train a standard multi-label CNN and show that this approach yields comparable performance to the unified regularization approach, and performs slightly better than the "internal similarity only" regularization approach.

1.4 Our approach to image retrieval for multi-label and multi-object images

Content based image retrieval (CBIR) is a field that deals with the retrieval of digital images from a database, based on a supplied query image, using only image content as a basis for comparison. A family of algorithms collectively known as *deep metric learning* have made it possible for DNNs to be used in CBIR, enabling big improvements in this research area. Deep metric learning enables the compact representation of a digital image as a lower dimensional vector which in turn enables fast image comparison and retrieval. A sub-field called *deep hashing* aims to further encode these representations into binary vectors to enable even faster retrieval by comparing hamming distances. These models typically comprise a CNN backbone, followed by one or more fully connected layers. This pipeline extracts useful features from image pixel data, and reduces them to similarity preserving vectors, called embeddings. The similarity between a query and database image is usually defined as an inner product between their low dimensional embeddings.

There have been many methods proposed over the years to achieve state-of-the-art performance in deep metric learning. However, we note that most of these approaches have the following recurring issues:

1. They either ignore multi-label images or treat them more or less the way as single label images.
2. They do not tackle multi-object images, and how the definition of similarity might be affected in this case.

We argue that special treatment has to be observed when dealing with multi-label images, as opposed to single-label images. For example, care has to be taken when

defining ground-truth similarity, and choosing the objective for training deep metric learning models. We define *multi-object* images as those that not only have multiple labels per image, but also multiple instances of each label in a single image. In chapter 3, we introduce a new definition of ground-truth similarity for multi-label and multi-object images, and modify the objective function accordingly. We show that our model outperforms the baseline and some state-of-the-art approaches dealing with the same type of data. Later, we present a further improvement of our method, comprising an improved loss function, a new triplet selection strategy, and label reconstruction through an auto-encoder model, to deal with label incompleteness in multi-label data. We show that each of these contributions test time retrieval performance over both the baseline and our first proposed method.

Chapter 2

Multi-label Annotation with Missing Labels

2.1 Introduction

With the recent exponential growth of the web, and consequently multimedia, it has become necessary to be able store, organize and retrieve large quantities of images and videos. This usually requires efficient automatic image annotation or tagging. Multi-label image annotation involves specifying the most relevant labels for any given image that describe its visual content. During the past decade, automated annotation with multi-label learning has been widely researched [32, 33]. Multi-label image annotation has also achieved great progress in various sub-domains such as multi-object recognition [34, 35], scene recognition [36], facial action detection [37], and medical diagnostic prediction [38, 39]. It is difficult to prepare complete labels to ensure correct predictions for multi-label learning, which makes training of neural networks with missing labels an important problem in automated multi-label image annotation.

Most approaches to multi-label annotation begin by transforming the problem from one of multi-label annotation to one of single-label classification [40, 41]. Binary relevance [41, 42] is perhaps the most popular transformation of this kind and is considered the baseline for multi-label image annotation. This approach involves

learning several binary classifiers, one for each label in the label space. The idea of binary relevance has been extended to deep neural networks, where each node in the output layer predicts a score for a single label [43–45]. Figure 2.1 shows such a formulation implemented using a CNN.

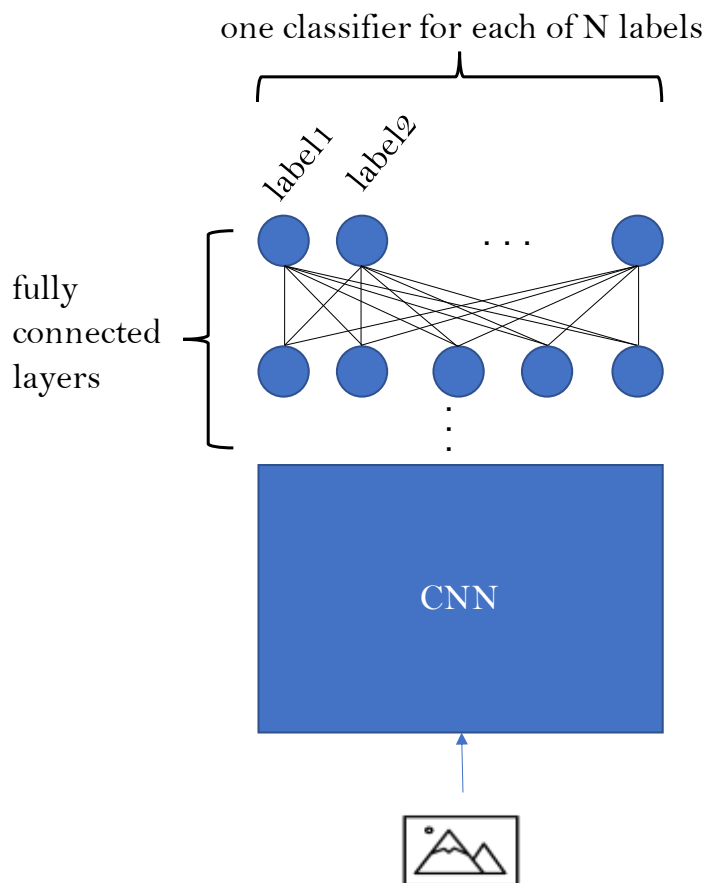


Fig. 2.1 Binary relevance, implemented by a CNN

While binary relevance is preferred among the various problem transformation techniques, it has been argued that it may lack the ability to model label co-occurrence relationships. However, several recently developed techniques based on deep neural networks have shown that label relations are important, especially when dealing with missing labels. Researchers have achieved acceptable multi-label annotation performance with this type of data by incorporating label relationship mechanisms. The relationships are derived using various strategies such as generating tree-structured graphs [46], using a structured inference neural network [47] and

co-occurrence dependency [48]. However, these methods only implement internal relations for multi-label learning. To better understand ambiguous examples and minimize false predictions, multi-label image annotation requires not only internal relations between labels but also strong relations that are external to the ground truth[47].

Incorporating both internal and external label relations in training a deep CNN could capture diverse contextual relations and specifically infer the remaining missing labels for a given set of labels for a particular image. For example, in Figure 2.2, the missing label “arctic” can be correctly recovered, which is meaningful and directly related to the given label subset as well as to the visual content of the image. In this chapter, we propose a regularization term that combines strong dependencies derived from both internal and external similarities between labels. Regularization enables superior and effective fine-tuning of the network layers, thus improving its performance. This could be understood from earlier studies showing that incorporating semantic regularization in a deep network improves accuracy and convergence speed [49–51].

Additionally, we propose an idea for multi-label learning with incomplete labels by combining a restricted Boltzmann machine (RBM) with a CNN. RBMs are primarily used to learn generative models[52], although they can be adapted to learn supervised discriminative models [53]. We use the RBM to generate additional labels (assumed to be missing) for images used to train a CNN for multi-label image annotation. Our reasoning is that by supplementing each training example with labels based on co-occurrence relationships learned from the entire set, we can improve the model’s capability to generalize to unseen data. This approach is similar in essence to the approach presented in [54], where RBMs are used for collaborative filtering for movie recommendation. In our case, each hidden unit learns to model dependency between different labels. The learned weights between visible and hidden layers hold co-occurrence information. This approach requires relatively less hyper-parameter tuning to build co-occurrence relationships between the input label vectors. In this study, we demonstrate that an RBM and CNN combination can more

accurately predict the relationship of input instances than the baseline multi-label CNN approach. We also compare its performance to the regularization approach.



Fig. 2.2 Example demonstrating the consistency of predicted label (**arctic**) with the given labels (bear, snow, polar, tundra), and its relevance to the image content

The contributions of this study can be summarized as follows: (1) A novel contextual regularization for CNN models is proposed for improved image annotation, which differs from conventional approaches, by introducing a unified internal and external graph Laplacian regularization term in the objective function of the CNN; (2) A contextual similarity metric between labels internally and externally is generated using Hayashi's quantification method-type III and the word2vec method, respectively; (3) A separate method is introduced that uses an RBM to pre-process training data to recover missing labels before training a CNN. Extensive evaluation on three different datasets is performed to confirm whether a unified internal-external label-relation regularization graph derived from co-occurrence data could produce better performance than individual regularization for an image annotation deep CNN, and whether test time accuracy is improved by our RBM-based label pre-processing

approach. Contributions (1) and (2) were published in [55] and contribution (3) was published in [56].

2.2 Related works

The problem of multi-label learning for data with missing-labels has been studied in a number of contexts. In this section we review related works on multi-label learning with a focus on label relations, followed by a discussion of regularization in neural networks.

2.2.1 Multi-label learning with label relations

Co-occurrence distributions learned from the internal label space can be used to compensate for missing labels [48]. For example, the multi-label local correlation approach encodes the local influence of label correlations using the feature representation of each instance [57]. Other researchers have proposed parametric models, which combine pair-wise correlations of class labels to solve the multi-label learning problem [58]. Several works on multi-label learning have suggested that external knowledge of label relations could improve label prediction [59, 60]. For example, [61] generated the co-occurrence of pairs of labels using external knowledge for multi-label annotation. Furthermore, to adequately address the problem of missing labels, an integrated framework can be used to learn the complex correlations between labels for multi-label classification with missing labels [62]. Lee et al. [63] proposed that label relations observed in the external space can be used to identify multiple unseen class labels for each input instance for performing multi-label classification.

The above-mentioned methods independently handled internal or external label relations in learning algorithms, while our proposed model incorporates co-occurrence distributions of both internal and external label relations. Note that inter-relations between different labels have already been exploited in recent techniques by using a mixed graph to encode a network of label dependencies [64], a unified correlative multi-label method to classify the labels [32], and quadratic energy

function graphs for constructing complete labels [65]. However, these algorithms are too complex and are not appropriate for large datasets.

2.2.2 Approaches using specialized architectures

Various techniques have been proposed to incorporate label co-occurrence information in training algorithms. Utilizing label inter-dependency information found in ground truth data can also solve the problem of inaccurate or missing labels. J. Wang et al. [66] proposed a framework that combines a CNN and a recurrent neural network (RNN), where the CNN acts as a feature extractor, and the RNN models image-label relationships and label dependency. Label dependency is incorporated by representing a multi-label annotation as an ordered path of label predictions. Another approach is presented in [43], which consists of a deep neural network-based Canonical-Correlated Auto-encoder that can exploit label inter-dependency during both label embedding and prediction processes. The authors also claim that it can easily be extended to the case of missing labels. Like one of the approaches presented in this paper, the approach presented in [67] also uses an RBM, with the main difference being that while they use a 'conditional RBM' network to both predict labels and fine tune those predicted labels, we train our RBM to reconstruct the training labels that are used to train a separate multi-label prediction CNN.

2.2.3 Regularization techniques in neural networks

The introduction of a label relation graph in the regularization term of a deep neural network model enables more efficient training and avoids over-fitting which in turn leads to better performance. Pengfei et al. [68] used semantic information to regularize the combination of two different neural network layers. Similarly, Yan et al. [51] implemented an attribute induced semantic regularization to tune the middle embedding layer. Several works have focused on the loss function of a neural network to solve the *multi-label learning with missing labels* problem with large-scale labels. Wu et al. [69] proposed a sub-modular objective function to handle the problem of large numbers of negative labels. Another study [70] exploited

the structure of a specific loss function for the annotation problem. Inspired by these research techniques, we measure the effectiveness of a unified label-relation regularization graph in training a deep CNN for multi-label image annotation.

In our proposed method, we combine two different matrices for regularization. One matrix is the graph Laplacian matrix of all the Word2Vec similarities between labels in the dataset [71]. The Word2Vec similarity is calculated from a model trained on the Wikipedia dumped data ¹. The second matrix is the graph Laplacian matrix with all the co-occurrence-based label similarities calculated by Hayashi's quantification method-type III (HQ-III) [72]. The combination of these two matrices is used as a regularization term in conjunction with the neural network's original objective function. Thus, the weights of the graph Laplacian matrix are calculated by using the similarities between vectors obtained from both the internal and external label spaces. Hence, the regularization term in the proposed objective function introduces correlation information between each pair of labels in the training process and increases the co-occurrence probability of labels with high observed co-occurrence frequency.

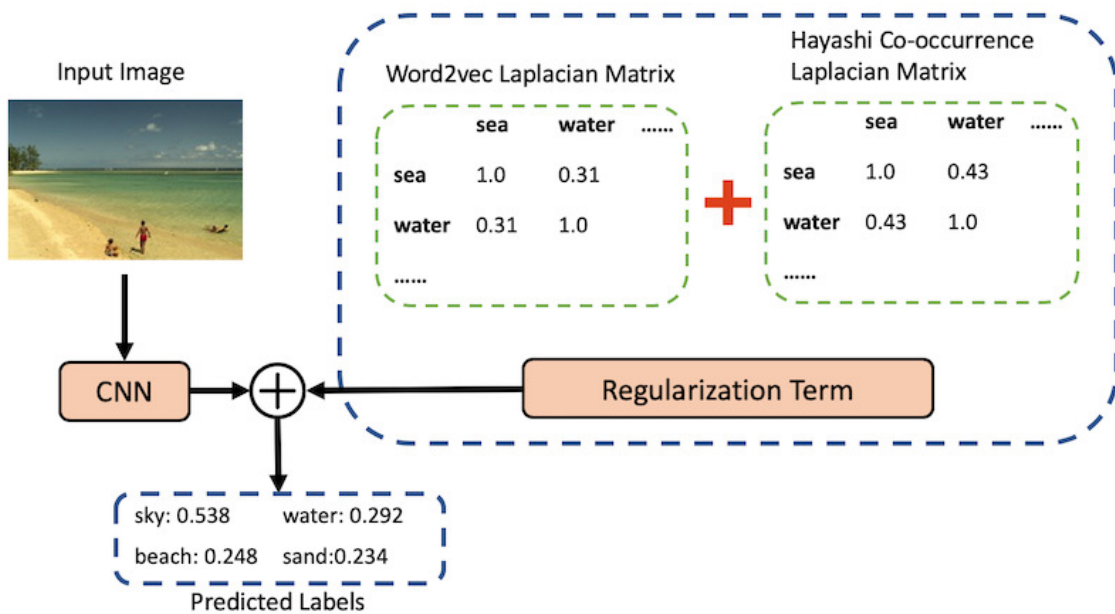


Fig. 2.3 Illustration of the proposed approach

¹at the time of writing, available at <https://dumps.wikimedia.org/enwiki/>

2.2.4 Hayashi's quantification method-type III

HQ-III is applied in the understanding of categorical data, including cross-tabulation or contingency tables [72]. It is used to calculate the vector representations of each row and column by utilizing the information of co-occurrences. Suppose the cross-tabulation that records label occurrence is represented by $T = [t_{ij}]$, where $i = 1, 2, \dots, M$ and $j = 1, 2, \dots, N$.

Table 2.1 The example of cross tabulation.

Examples(Ω)	Labels(Θ)				sum
	θ_1	θ_2	\dots	θ_N	
ω_1	t_{11}	t_{12}	\cdot	t_{1N}	$t_{1\cdot}$
ω_2	t_{21}	t_{22}	\dots	t_{2N}	$t_{2\cdot}$
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
ω_M	t_{M1}	t_{M2}	\dots	t_{MN}	$t_{M\cdot}$
sum	$t_{\cdot 1}$	$t_{\cdot 2}$	\dots	$t_{\cdot N}$	n

Using this table, we can extract the vector representations q_i and q_j of the i^{th} row and the j^{th} column respectively, by applying HQ-III. If we assign numerical scores u_i and v_j to each category and sample respectively, the weighted averages of u and v are defined as:

$$\bar{u} = \frac{t_{1\cdot}u_1 + t_{2\cdot}u_2 + \dots + t_{M\cdot}u_M}{n} \quad (2.1)$$

$$\bar{v} = \frac{t_{\cdot 1}v_1 + t_{\cdot 2}v_2 + \dots + t_{\cdot N}v_N}{n} \quad (2.2)$$

and the variance of u and v and their covariance, is defined as:

$$\sigma_u^2 = \frac{1}{n} \sum_i t_{i\cdot} (u_i - \bar{u})^2 \quad (2.3)$$

$$\sigma_v^2 = \frac{1}{n} \sum_j t_{\cdot j} (v_j - \bar{v})^2 \quad (2.4)$$

$$\sigma_{uv} = \frac{1}{n} \sum_i \sum_j t_{ij} (u_i - \bar{u})(v_j - \bar{v}) \quad (2.5)$$

The goal of HQ-III is to find values of u_i and v_i that maximize the correlation coefficient:

$$r_{uv} = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \quad . \quad (2.6)$$

This becomes an eigenvalue problem, with $N - 1$ possible solutions, we can use these vectors to get an $N - 1$ -dimensional vector for each label. Note that the distance between these vectors becomes small if the pattern of responses in the cross-tabulation is identical. In this study, HQ-III is used to calculate the distance between the vectors of each label, which can explain the internal similarity between each pair of labels.

2.2.5 Laplacian eigenmaps

This is a manifold learning algorithm based on spectral graph theory, proposed by Belkin et al. [73]. Given N samples, it constructs a vector representation for each sample based on sample similarities. Assuming the similarity between any two samples i and j is given by s_{ij} , they construct a weighted graph (G, E) where each node in G corresponds to a sample and the edge $(i, j) \in E$ connecting two nodes i and j represents the similarity s_{ij} between these two samples. Figure 2.4 illustrates this. Then the 1-dimensional representation y_i for each sample is obtained by minimizing

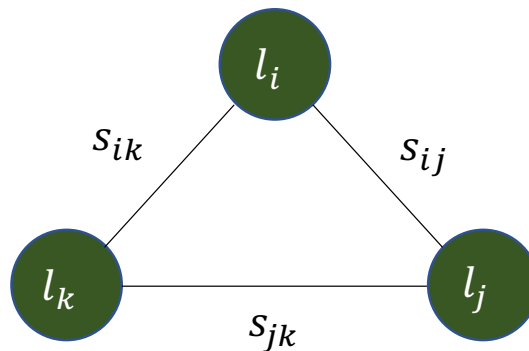


Fig. 2.4 Illustration of a simple weighted graph with sample similarities

the objective function

$$\frac{1}{2} \sum_{i,j}^N s_{ij} (y_i - y_j)^2 = \mathbf{y}^T \mathbf{L} \mathbf{y} \quad (2.7)$$

where $\mathbf{y} = [y_1 \ \cdots \ y_N]^T$. The matrix \mathbf{L} is Laplacian matrix defined as $\mathbf{L} = \mathbf{D} - \mathbf{S}$, where $\mathbf{S} = [s_{ij}]$ and \mathbf{D} is a diagonal matrix with diagonal elements $d_{ii} = \sum_{j=1}^N s_{ij}$. K. Watanabe et al. [74] introduced a regularization term based on this objective function to preserve locality of the input space in the outputs of multi-nominal logistic regression. In our study, we use this objective function as a CNN regularization term to constrain network outputs using ground-truth label similarities obtained from label co-occurrence information in the training data.

2.2.6 Word2Vec

The Word2Vec method was developed by Mikolov et al. [71] in 2013. The model is trained on a corpus of text (like Wikipedia or Google News) and then outputs the vector representations of all words in the text. Unlike the previous method, where a neural network learns the expression vectors of words, Word2Vec employs the Skip-gram model, which reduces the calculation of dense matrix multiplications. In our proposed method, we use the Word2Vec model to obtain the vector representations of labels in a dataset and then calculate the similarity between each pair of labels.

2.3 Unified Graph Laplacian Regularization Approach

2.3.1 Training with missing labels

Let $\{(x_i, t_i)\}_{i=1}^M = \{\mathbf{X}, \mathbf{T}\}$ be the set of training samples with missing labels, where

$$\mathbf{X} = [x_1 \ \cdots \ x_i \ \cdots \ x_M]^T \quad (2.8)$$

$$\mathbf{T} = [t_1 \ \cdots \ t_i \ \cdots \ t_M]^T \quad (2.9)$$

Here, x_i is the i^{th} image used as an input to the CNN and $t_i = [t_{i1} \ \cdots \ t_{iN}]$ represents the binary vector representation of the labels for the i^{th} image, where $t_{ij} = 1$ when the j^{th} label assigned to the i^{th} image; otherwise, $t_{ij} = 0$. M and N are the number of samples and labels, respectively.

Let $y = f(x; \theta)$ be the function of the CNN model for the input image x where θ represents the parameters of the CNN. AlexNet [3] is used as the baseline CNN architecture. The network has five convolutional layers and two fully connected layers. The acronym “BN” in the figure denotes batch normalization.

To estimate the posterior probability of each label, a sigmoid activation function is used after the output layer and the sum sigmoid cross entropies across all the labels used as the loss function. After training the parameters θ , we can estimate the labels \hat{y} by feeding the test input image x into the trained CNN model

$$\hat{y} = f(x; \theta) \quad (2.10)$$

The value of each component in the estimated vector \hat{y} is the probability of the corresponding label.

2.3.2 Internal distributions of label similarity

The internal similarity distributions of each label in the dataset is obtained by applying HQ-III on the frequency Table T . Let q_j , ($j = 1, \dots, N$) be the HQ-III vector representation of each label. We calculate the co-occurrence distance between a pair of labels using the vectors q_i and q_j in their internal label space as

$$d_{ij}^h = \|q_i - q_j\|^2 \quad (2.11)$$

The similarity using internal co-occurrence distributions is defined as

$$s_{ij}^h = \exp(-\delta \times d_{ij}^h) \quad (2.12)$$

where δ controls the influence of the distance.

2.3.3 External distributions of label similarity

The external similarity of each label in the dataset is derived using the Word2Vec method. Let v_j be the vector of the j -th ($j = 1, \dots, N$) label. The word2vec distance between a pair of labels v_i and v_j is defined as

$$s_{ij}^w = \exp(-\epsilon \times d_{ij}^w) \quad (2.13)$$

where ϵ controls the influence of the Word2Vec distance.

2.3.4 Regularization graph with unified co-occurrence distributions

To control the similarities s_{ij} between estimated labels, we introduce the graph Laplacian regularization term based on Equation 2.7, defined as

$$G = \frac{1}{2} \sum_{i,j}^N (\hat{y}_i - \hat{y}_j)^2 s_{ij} = \hat{\mathbf{y}}^T \mathbf{L} \hat{\mathbf{y}} \quad (2.14)$$

where $\hat{\mathbf{y}} = [\hat{y}_1 \ \dots \ \hat{y}_N]^T$ and ($0 \leq \hat{y}_i \leq 1$) is the vector of network outputs representing estimated labels.

Since we have two similarities s_{ij}^h and s_{ij}^w representing both internal and external similarity, we can define two Laplacian regularization terms. The regularization term for the internal co-occurrence similarity graph is defined as

$$G_h = \frac{1}{2} \sum_{i,j}^N (\hat{y}_i - \hat{y}_j)^2 s_{ij}^h = \hat{\mathbf{y}}^T \mathbf{L}_h \hat{\mathbf{y}} \quad (2.15)$$

Similarly, the regularization term for the external co-occurrence similarity graph is defined as

$$G_w = \frac{1}{2} \sum_{i,j}^N (\hat{y}_i - \hat{y}_j)^2 s_{ij}^w = \hat{\mathbf{y}}^T \mathbf{L}_w \hat{\mathbf{y}} \quad (2.16)$$

where L_h and L_w are the co-occurrence dependency Laplacian matrices from the internal and external label spaces, respectively. For M training samples, we can define the average graph Laplacian regularization terms as

$$D_h = \frac{1}{M} \sum_{l=1}^M \left(\frac{1}{2} \sum_{i,j} s_{ij}^h (\hat{y}_{li} - \hat{y}_{lj})^2 \right) = \sum_{l=1}^M \hat{y}_l^T L_h \hat{y}_l \quad (2.17)$$

$$D_w = \frac{1}{M} \sum_{l=1}^M \left(\frac{1}{2} \sum_{i,j} s_{ij}^w (\hat{y}_{li} - \hat{y}_{lj})^2 \right) = \sum_{l=1}^M \hat{y}_l^T L_w \hat{y}_l \quad (2.18)$$

The values of D_h and D_w become small if the estimated labels are similar for a given pair of labels with similar vector representations implying high co-occurrence or similar meaning. Finally we combine the graph Laplacian regularization term from internal and external co-occurrence similarities as

$$D = \alpha D_h + (1 - \alpha) D_w = \sum_l \hat{y}_l^T (\alpha L_h + (1 - \alpha) L_w) \hat{y}_l \quad (2.19)$$

where α controls the contribution of each regularization term.

2.3.5 Training with a combined objective function

The original objective function for standard multi-label annotation over all the training samples is given by

$$E = \sum_{l=1}^M \sum_{k=1}^N \{-t_{lk} \log(\hat{y}_{lk}) - (1 - t_{lk}) \log(1 - \hat{y}_{lk})\}. \quad (2.20)$$

Here, the objective function is modified by combining the original objective function E with the graph Laplacian regularization term D , which gives

$$Q = E + \beta D. \quad (2.21)$$

The parameter β controls the effect of the regularization term.

2.4 Label Reconstruction by RBM Approach

In this section, we present an approach that uses an RBM to capture label dependencies in the training data, and reconstruct the label vectors with supplemental labels. The illustration of our proposed method is shown in Figure 2.5.

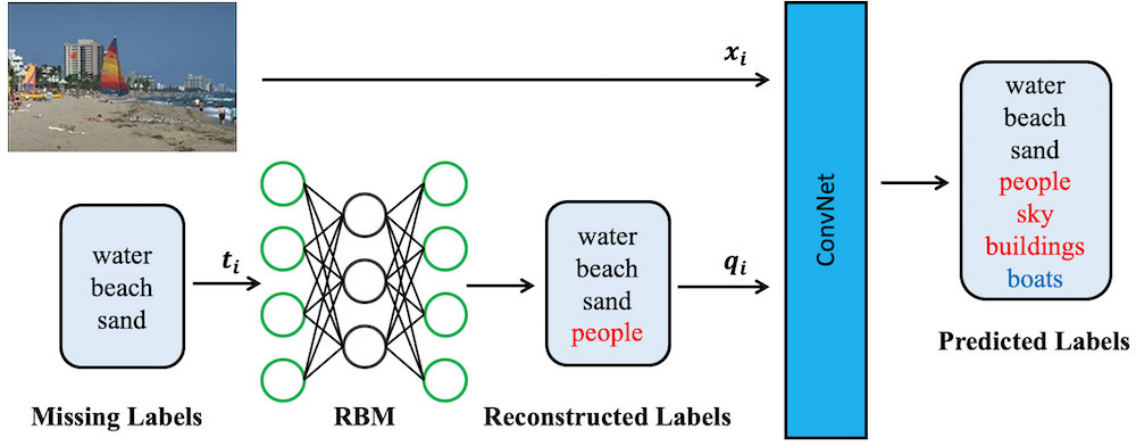


Fig. 2.5 The architecture of the proposed CNN+RBM model for multi-label annotation. Before input into the CNN, the missing labels are firstly reconstructed by the pre-trained restricted Boltzmann Machine.

2.4.1 Train Restricted Boltzmann Machine with missing labels

We use training samples with missing labels to train the RBM model. An RBM is a 2-layer neural network which contains visible units $v \in \{0, 1\}^N$ and hidden units $h \in \{0, 1\}^H$ where N and H are the numbers of visible and hidden units, respectively (Figure 2.6). It is an energy-based model and can be used to encode the dependencies between the visible units. In the learning process, we minimize the energy function of a certain state (v, h) , defined as

$$E(v, h) = -\mathbf{b}^T v - \mathbf{c}^T h - v^T W h \quad (2.22)$$

$$= -\sum_{i=1}^N b_i v_i - \sum_{j=1}^H c_j h_j - \sum_{i,j} v_i h_j w_{ij} \quad (2.23)$$

with the parameter $\theta = \{W, \mathbf{b}, \mathbf{c}\}$. W denotes the weights between the 2 layers, and \mathbf{b} and \mathbf{c} represent visible and hidden unit biases, respectively. If there are no

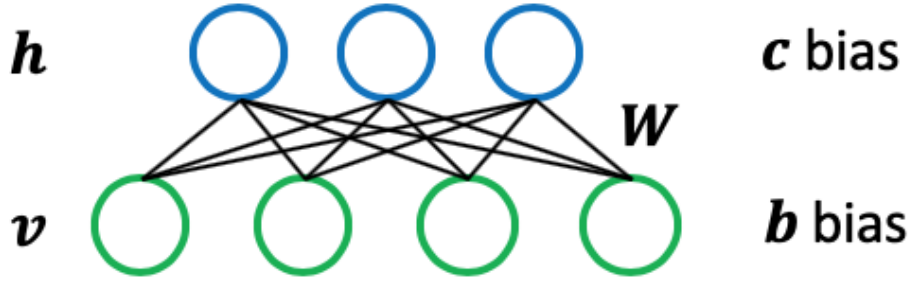


Fig. 2.6 Restricted Boltzmann Machine

connections between hidden units, then the probability distribution of v is given as

$$p(v) = \frac{1}{Z} \sum_h p(v, h) = \frac{1}{Z} \sum_h e^{-E(v, h)} \quad (2.24)$$

where Z is the normalization factor by the summation of over all possible pairs of visible and hidden vectors.

$$Z = \sum_{v, h} e^{-E(v, h)} \quad (2.25)$$

Because both visible and hidden units can only have state of 0 or 1, then the conditional probability of a visible and hidden unit value being set to 1 is defined as

$$p(h_i = 1 | v) = \sigma\left(\sum_{j=1}^M w_{ij} v_j + c_i\right) \quad (2.26)$$

$$p(v_j = 1 | h) = \sigma\left(\sum_{i=1}^N w_{ij} h_i + b_j\right) \quad (2.27)$$

where σ denotes the logistic sigmoid function. The log-likelihood for a single training example v is given as

$$\ln \mathcal{L}(\theta | v) = \ln p(v | \theta) = \ln \frac{1}{Z} \sum_h e^{-E(v, h)} \quad (2.28)$$

According to [75] the process of calculating the gradient function θ is

$$\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial w_{ij}} = p(h_i = 1|v)v_j - \sum_v p(v)p(h_i = 1|v)v_j \quad (2.29)$$

$$\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial b_j} = v_j - \sum_v p(v)v_j \quad (2.30)$$

$$\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial c_i} = p(h_i = 1|v) - \sum_v p(v)p(h_i = 1|v) \quad (2.31)$$

Computing the second term of equations 2.29, 2.30 and 2.31, one can approximate the expectation by samples obtained by Gibbs sampling. In the learning process, Gibbs chain runs for only k steps [76], which is called k -step contrastive divergence learning (CD- k).

2.4.2 Reconstruction of labels by Pre-trained RBM model

Let $D = \{(x_i, t_i)\}_{i=1}^M = \{X, T\}$ denote a set of training samples. Here x_i is the i -th sample used as input to the neural network and $t_i = [t_i^{(1)} \dots t_i^{(N)}]^T$ are the training labels (supposedly containing missing labels) which denotes the binary vector representation of the i -th sample. $t_i^{(j)} = 1$ when the j -th label is annotated to the i -th image, otherwise $t_i^{(j)} = 0$. M and N are the numbers of samples and labels respectively. According to the RBM, the number of hidden units H is less than the number of visible units N ($H < N$).

Suppose $\mathcal{R}(x)$ is the reconstruction function of the pre-trained RBM model for training label set T . The reconstructed label set q of the original label set t can be obtained using the equation $q = \mathcal{R}(t) = [q^{(1)} \dots q^{(N)}]^T$. The label reconstruction algorithm used in this study is presented in algorithm 1.

Algorithm 1: Label reconstruction algorithm \mathcal{R}

```

Given missing label set  $\mathbf{t} = [t^{(1)} \dots t^{(N)}]^T$ 
▷ Prop up
for  $i = 0, \dots, H$  do
  |  $h_i \leftarrow \sigma(\sum_{j=1}^N w_{ij}t^{(j)} + c_i)$ 
end
▷ Prop down
for  $j = 0, \dots, N$  do
  |  $v_j \leftarrow \sigma(\sum_{i=1}^H w_{ij}h_i + b_j)$ 
end
▷ Get binomial samples of the visible units based on their activation  $v$ 
for  $j = 0, \dots, N$  do
  |  $q^{(j)} \leftarrow \text{binomial}(v_j)$ 
end
▷ Keep all the elements in  $\mathbf{t}$  which are equal to 1
for  $j = 0, \dots, N$  do
  | if  $t^{(j)} = 1$  then
  | |  $q^{(j)} \leftarrow t^{(j)}$ 
  | end
end

```

2.4.3 Training the CNN using RBM reconstructed labels

Let $D_{new} = \{(x_i, q_i)\}_{i=1}^M = \{X, Q\}$ denote the newly reconstructed training samples. We use T to train the CNN. AlexNet [3] is used as the CNN architecture, which has 5 convolution layers and 2 fully connected layers. Because the true and predicted values are given by the state $\{0, 1\}$, we use sigmoid cross entropy to evaluate dissimilarity between them. The loss function used to train the CNN is

$$H(y, \hat{y}) = -y \log \hat{y} - (1 - y) \log (1 - \hat{y}) \quad (2.32)$$

where y and \hat{y} denotes true and predicted values, respectively. The original objective function of the CNN is

$$J = \sum_{i=1}^M \sum_{j=1}^N H(t_j^{(i)}, \hat{y}_j^{(i)}) \quad (2.33)$$

We used the reconstructed labels as a teacher signal and thus the objective function becomes

$$J' = \sum_{i=1}^M \sum_{j=1}^N H(q_j^{(i)}, \hat{y}_j^{(i)}) \quad (2.34)$$

Let $\mathcal{S}(x)$ be the function learned by the CNN for the input image x . Then the estimated labels \hat{y} can be obtained by feeding image x from the test data set to the trained CNN model as

$$\hat{y} = \mathcal{S}(x). \quad (2.35)$$

The value of the each element in the estimated vector \hat{y} can be considered as the probability of the corresponding label.

2.5 Experiments

To validate the proposed approach, we use three standard image datasets and compare the results to a baseline CNN without regularization, a CNN using internal similarity alone with HQ-III, and a CNN using external similarity alone by the Word2Vec method. In this section, we describe the datasets, model parameter settings, and experimental results.

2.5.1 Datasets

We use three benchmark image annotation datasets: Corel5k, NUS-WIDE-LITE, and ESPGame. To generate the training dataset with missing labels, we randomly remove some labels in each training sample such that the number of labels in each sample is more than two. Table 2.2 shows the datasets used in our experiments. The columns in Table 2.2 named average and missing average represent the average number of assigned labels in each training sample and the average number of processed labels respectively.

2.5.2 Parameter Settings

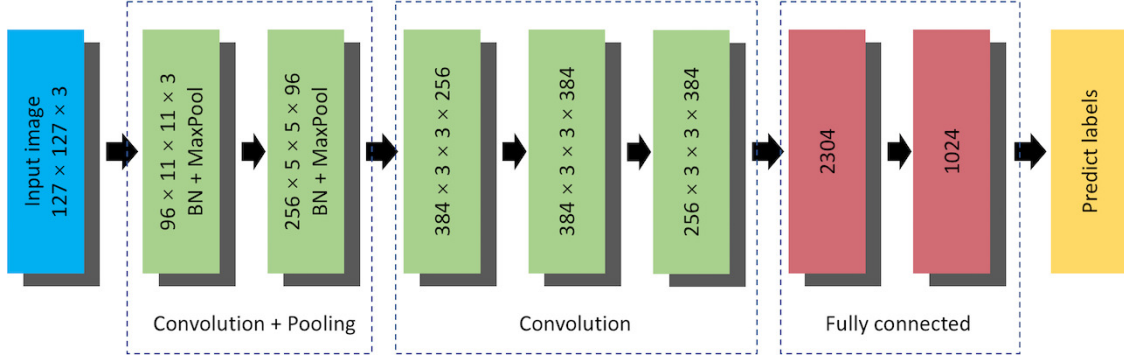


Fig. 2.7 CNN architecture

Before feeding the images into the CNN, we reshape all the original images to a size of 127×127 pixels. The parameter α in Eq.(2.19) is set to 0.5 and β in Eq.(2.21) is set to 0.1. We obtained these values by performing a validated parameter sweep in the ranges $[0, 1]$ and $[0.001, 10]$ for α and β respectively.

Table 2.2 Datasets and their corresponding labels

Dataset	Labels	#Training	#Test	Average	Missing Average
Corel5k	260	4500	499	3.4	2.5
NUS-WIDE-LITE	81	27807	27808	1.6	1.4
EspGame	268	18689	2081	3.7	3.0

2.5.3 Performance measures

The micro-F1 score is used to measure the performance of the proposed model for estimating multiple labels. This score measures the accuracy on the test datasets, considering both precision p and recall r , and is defined as

$$\text{Micro} - F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (2.36)$$

The performance on test datasets is calculated by

$$\text{Micro} - F_1 = \frac{1}{M} \sum_{i=1}^M \frac{2 \sum_{j=1}^N t_j^{(i)} y_j^{(i)}}{\sum_{j=1}^N t_j^{(i)} + \sum_{j=1}^N y_j^{(i)}}. \quad (2.37)$$

Here M and N are the number of test dataset samples and labels, respectively. The prediction is considered perfect when the score is 1 (perfect precision and recall) and worst when its value is 0. To select the most suitable labels, we use three kinds of label selection thresholds: labels whose predicted probability is above 0.1, the top-three ranked labels, and the top-five ranked labels. We compare the performance of the deep CNN model with unified regularization to those of a regularized deep CNN with internal label dependency alone, external label dependency alone, and a baseline CNN without any regularization term. Furthermore, to confirm the learning ability with incomplete labels, we remove some labels and observe the performance at different label removal rates (10%, 30%, and 50%) and compare the results in terms of the micro-F1-score against the baseline CNN.

2.5.4 Results and Discussion

Unified graph Laplacian regularization

We found that the model trained by the new combined objective function has better prediction ability than the one using the original objective function. To evaluate our results comparatively, we set up the experimental analyses according to previous works utilizing external [61] and internal [48] label similarity. Tables 2.3, 2.4 and 2.5 show the results evaluated by measuring the micro-F1 score at 30% label removal rates on different datasets based on the three kinds of label selection criteria. The proposed deep CNN model with unified regularization achieves higher F1-scores than the other methods, as highlighted in the tables. Furthermore, it can be observed that the deep CNN model with either internal similarity regularization or external similarity regularization produces better scores than the baseline CNN without a regularization term. On the Corel5k dataset, the micro-F1 score for labels with probability over 0.1% for the proposed method is higher than that of the baseline

CNN, by more than 0.64%. Similarly, the score of the proposed method is higher than the deep CNN model with a regularization term of red internal [48] and external [61] similarity by more than 0.30% and 0.6%, respectively. The approach presented in [77], using pairwise label correlations for multi-label classification achieved lower accuracy (12.1%) on the Corel5k dataset. Our unified approach modelling label relations on the same dataset achieves an acceptable improvement on label prediction. On the NUS-LITE, and EspGame datasets, the micro-F1 scores based on the labels with probability over 0.1% are higher (in the range of 5.37%-8.87%) for the unified regularization and lower (in the range of 4.22%-8.49%) for the baseline CNN, demonstrating the efficiency of the proposed unified regularization technique in deep CNN for predicting missing labels.

Table 2.3 Comparison of the micro-F1 score on the Corel5k dataset

Deep CNN methods	Top-3(%)	Top-5(%)	Probability \geq 0.1(%)
Without regularization	17.25	17.75	17.12
Regularized with internal similarity alone	18.14	18.23	17.30
Regularized with external similarity alone	17.88	18.13	17.16
Proposed unified regularization	18.52	17.99	17.76

Table 2.4 Comparison of the micro-F1 score on the NUS-WIDE-LITE dataset.

Deep CNN methods	Top-3(%)	Top-5(%)	Probability \geq 0.1(%)
Without regularization	5.95	6.31	4.22
Regularized with internal similarity alone	7.35	7.65	5.17
Regularized with external similarity alone	7.26	7.58	5.09
Proposed unified regularization	7.43	7.73	5.37

Furthermore, to evaluate the performance of the unified regularization model against the baseline CNN without regularization, we visualize and compare the predicted labels across three different dataset images used in this study, as shown in figs. 2.8, 2.9 and 2.10. The labels with predicted probability over 0.1 are considered

Table 2.5 Comparison of the micro-F1 score on the EspGame dataset.

Deep CNN methods	Top-3(%)	Top-5(%)	Probability ≥ 0.1 (%)
Without regularization	9.42	10.13	8.49
Regularized with internal similarity alone	9.46	10.40	8.54
Regularized with external similarity alone	9.44	10.39	8.52
Proposed unified regularization	9.68	10.74	8.87

Table 2.6 Model sensitivity to parameters α and β on the Corel5k dataset (micro-F1 score for top-3 labels %).

	$\alpha = 0$	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 1$
$\beta = 0.01$	17.88	17.74	17.59	17.39	17.12
$\beta = 0.1$	17.08	16.71	18.52	16.63	16.82
$\beta = 1$	17.64	17.74	17.09	18.02	18.14

for the comparison of the methods. The additional labels predicted by the proposed method are shown in bold. The results show that the proposed model can accurately capture labels that are related to the original given label set as well as to the visual content of the objects appearing in Figure 2.8. The proposed model notably predicts “tree” and “forest”, which are not predicted by the baseline CNN. It clearly demonstrates exemplars on which our proposed method improves the baseline predictions. Similarly, the proposed method detects the missing labels “valley” and “sky” for the given label “lake” in Figure 2.9. However, the baseline CNN without regularization predicted a limited number of missing labels. The proposed unified co-occurrence approach can effectively infer the labels “sky” and “valley” from “clouds” and “mountain”, respectively, because it can infer labels based on similarities in the internal space as well as from the external world of common sense. Though the micro-F1 score of the proposed approach is lower than that of the baseline CNN, the probability of predicting missing labels is more accurate, precise and relevant to the visual content of the objects appearing in Figure 2.9.

Finally, we explain the results for Figure 2.10, which includes a large number of object classes. In terms of the micro-F1 score, the proposed approach outperforms the baseline CNN in terms of accurate and very relevant missing labels related to

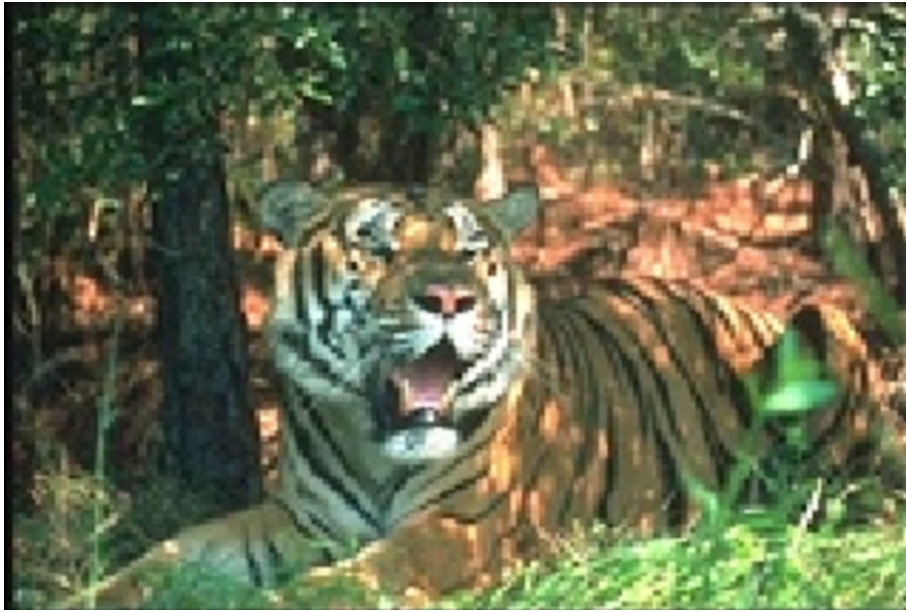


Fig. 2.8 of Corel5k dataset and predicted labels with probability $\geq 0.1\%$.

Given labels: cat, tiger, tree

Baseline CNN: cat, tiger, flowers, grass

Proposed: cat, **tree**, grass, **tiger**, **forest**

the visual content of the objects. The experimental results indicate that the internal and external distributions of label similarity are more appropriate for detecting missing labels when the number of training labels is small and the number of object classes is large. In addition, the learning ability of the proposed model with different missing-label rates, measured in terms of the micro-F1 score, compared to that of the baseline CNN on the Corel5K dataset is shown in Figure 2.11, Figure 2.12 and Figure 2.13. We observe that the proposed model outperforms the baseline CNN across different percentages of missing labels.

An analysis of the effect of parameters α and β on the performance of the model is shown in table 2.6. The results show that as the value of β increases from 0.01 to 1 exponentially, the optimal value of alpha changes from 0.25 to 1.0. This means that the two parameters interact with each other, with bigger values of β requiring bigger values of α . We reason that at larger values of β , increased contribution of the external label similarity term leads to too much contradiction with the ground truth.



Fig. 2.9 Example image from NUS-WIDE-LITE dataset and predicted labels with probability $\geq 0.1\%$.

Given labels: lake

Baseline CNN: clouds, mountain

Proposed: clouds, mountain, valley, sky

Label reconstruction by RBM

Table 2.7 shows the performance of the RBM-CNN combination, compared against the regularization-based methods. We note that the unified regularization approach achieves a minimal lead of 0.09% and 0.39% on the top-3 and probability ≥ 1 micro-F1 scores respectively. We interpret this to mean the unified regularization approach has a slightly better ability to capture label-dependencies and hence can recover missing labels accurately at test time. However, we note that the RBM-CNN combination still offers better performance than the baseline CNN, and more importantly, the internal similarity regularization approach. This is important because while the regularization approaches can only be used on outputs representing label probability predictions, label reconstruction approaches can be used on training data prior to any task. This makes this approach more versatile. We show in chapter 3 that label reconstruction using a graphical model can be applied to deep metric learning for image retrieval.



Fig. 2.10 Example image from EspGame dataset and predicted labels with probability $\geq 0.1\%$.

Given labels: guitar, woman, music, hair, sing

Baseline CNN: guitar, man, music, light, hair, sing, singer

Proposed: guitar, music, hair, sing, **singer, man, light, band**

Table 2.7 Comparison of the micro-F1 score on the Corel5k dataset

Deep CNN methods	Top-3(%)	Probability $\geq 0.1(\%)$
Without regularization	17.25	17.12
Regularized with internal similarity alone	18.14	17.30
Regularized with external similarity alone	17.88	17.16
Proposed unified regularization	18.52	17.76
Label reconstruction by RBM (hidden units=50)	18.43	17.37

2.6 Conclusion

In this chapter, we introduced a novel approach to address the multi-label image annotation problem with missing labels. Our study employed a unified approach, combining internal and external label dependencies. We implement this mechanism as a graph Laplacian regularization term in order to reliably recover missing labels from available labels and CNN inference. Due to this combined nature of label dependency representation, our proposed approach obtained distinctively more

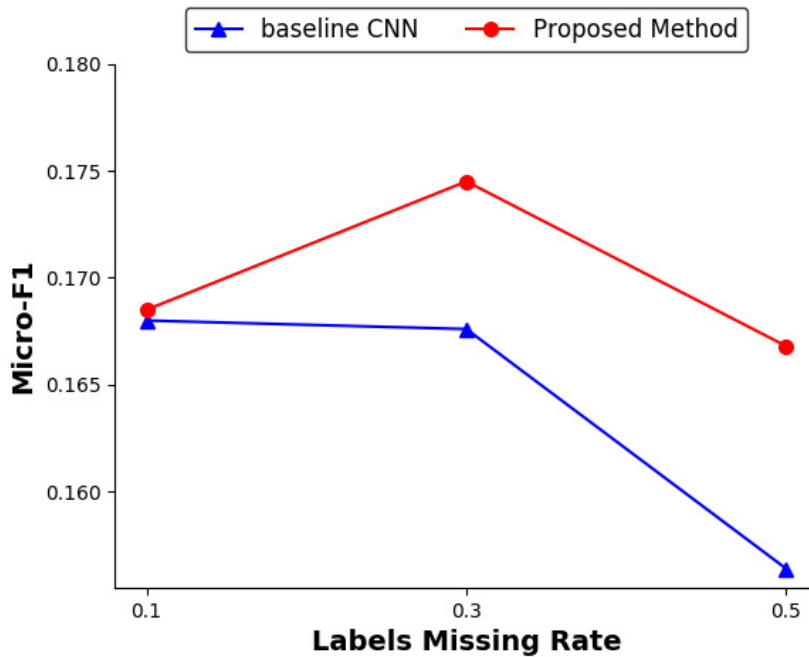


Fig. 2.11 Comparison of the micro-F1 score of the proposed method with that of the baseline CNN using different missing-label rates on the Corel5K dataset (probability $\geq 0.1\%$)

accurate results than other competing methods described in this study. Experimental analyses on three popular datasets revealed that our approach performs better than the baseline CNN without our proposed regularization. For future work, it would be interesting to test our approach with other neural network architectures. Moreover, we would like to conduct tests on other benchmark data sets to further evaluate its effectiveness for the recovery of missing labels.

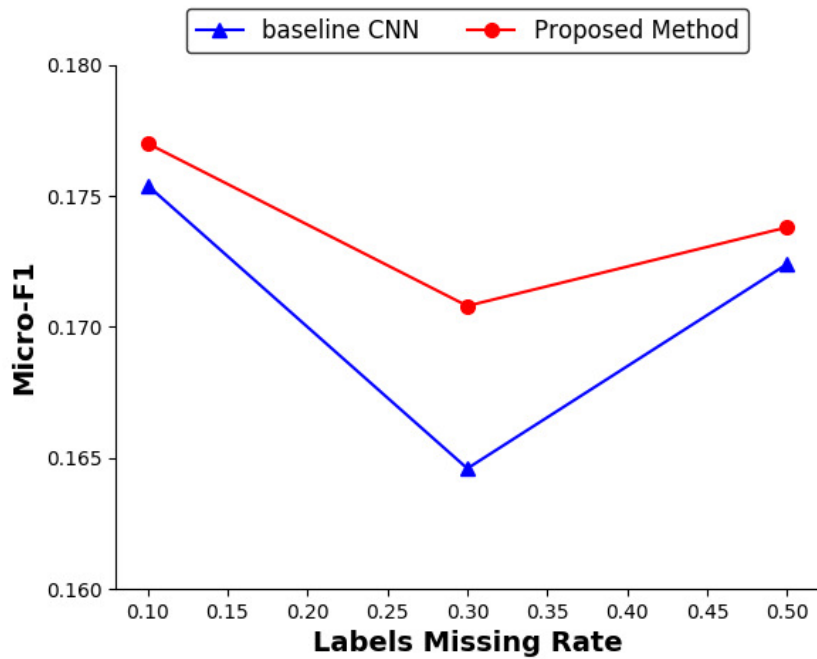


Fig. 2.12 Comparison of the micro-F1 score of the proposed method with that of the baseline CNN using different missing-label rates on the Corel5K dataset (Top-3)

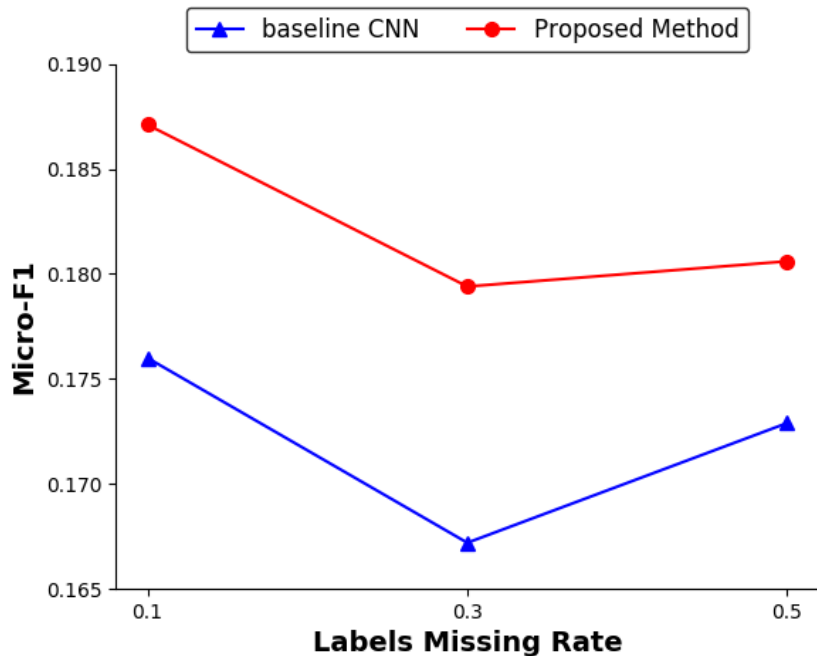


Fig. 2.13 Comparison of the micro-F1 score of the proposed method with that of the baseline CNN using different missing-label rates on the Corel5K dataset (Top-5)

Chapter 3

Multi-label and Multi-object Deep Metric Learning for Image Retrieval

3.1 Introduction

Image retrieval is one of the popular areas of interest in computer vision and pattern recognition today. Research in this area began in the early 1990's [78, 79] and recently, with the increase in the number of digital images collected and used in various domains, there is need for more accurate and faster algorithms for searching and retrieving images from large collections. Traditionally, real-world image retrieval systems use meta-data information associated with images to index and retrieve images. However, since textual information may not accurately or exhaustively describe visual content, CBIR is preferred [80]. CBIR is the search and retrieval of images from large-scale image databases using visual features such as texture, shape and color [81].

One of the challenges in CBIR remains the 'semantic gap' [82], which refers to the difficulty of describing high-level semantic concepts that humans are familiar with using low-level visual features [83–85]. Bridging this gap requires understanding and comparison of image content at a semantic level similar to that demonstrated by humans[83].

The ability to assess how closely one image matches another is an indispensable component of image retrieval systems, especially in search-by-example applications, where the aim is to retrieve images that are similar to a supplied query image. It has been shown that features extracted by convolutional neural networks can provide useful representations of image data [86, 87], which can in turn be used for various image processing and recognition tasks [88]. In deep metric learning [89], the goal is to learn such representations and their corresponding induced metrics by training the network to estimate similarity or relative similarity of image samples. The learned embeddings and their corresponding similarity metric can then be used directly in image retrieval, usually by performing a nearest neighbour search.

It has been argued that the ability to classify local image regions into semantic object or concept classes e.g water, rocks, person is key to achieving semantic scene understanding which is essential for better CBIR performance[90]. Furthermore, when considering semantic image retrieval, it is more desirable to represent and interpret images as scenes (that is, a collection of interacting objects and concepts) as opposed to collections of isolated objects [91].

In line with the above, our main contributions include the following:

1. A triplet metric learning architecture that combines the feature aggregation power of multi-layer perceptrons with the object-relation encoding and enumeration capabilities of relation networks, to learn embeddings that can be used to reliably compare image scenes.
2. A novel *soft-similarity* function for computing the similarity of multi-label image data. We show that our soft-similarity leads to better performance than a previously proposed approach, and works for both multi-label and multi-object images.
3. An adaptive margin triplet loss function that uses a similarity coefficient to adapt the margin for varying degrees of similarity.

Furthermore, we show that the embeddings learned by our architecture are quantizable, by employing a previously proposed quantization method that works on

top of a triplet metric learning network. Part of the contributions presented in this chapter were published in [92].

3.2 Related Work

3.2.1 Deep Metric Learning

In simplest terms, the metric learning problem is as follows: given an initial distance function $d(x, y)$ between two points x and y and a teacher signal representing the ground truth, construct a new distance function $d'(x, y)$ that is closer to the ground truth distance than the original function. Note that this definition does not strictly require a distance function i.e. it holds for some similarity function s . In deep metric learning, the problem becomes one of constructing a distance function $d'(f(x), f(y))$ where f is a deep neural network that extracts features $f(x)$ and $f(y)$ given inputs x and y respectively. There are two predominant models: *Siamese networks* [93–95] and *Triplet networks* [89, 96–98], which focus on contrastive embedding and triplet embedding respectively.

A basic *Siamese network* consists of a two-branch network that learns contrastive embedding from pairs of samples. The two parallel branches share parameters, and the model learns by minimizing a cost function that favors a small distance between pairs of samples labeled as similar and a large distance between pairs labeled as dissimilar. One shortfall of this type of model is that it requires training data with real-value precise pairwise distances which is not usually available [99]. It has also been observed that representations learned by these models provide sub-par results and for some types of data, the model almost completely fails to learn [89].

The *Triplet network* model is perhaps one of the most popular deep metric learning models [89, 96–98]. This model learns a triplet embedding by exploring the relative similarity of image samples. A triplet is made up of three samples: an *anchor*, a *positive* sample that is similar to the anchor and a *negative* sample that belongs to a different category from the anchor. The network learns by minimizing a cost function which favors an anchor-positive distance that is smaller than the anchor-negative distance.

There have been multiple variations to this function [100, 101], but primarily it is defined as:

$$L(\mathbf{a}, \mathbf{p}, \mathbf{n}) = \sum_{i=1}^N \max\{d(\mathbf{a}_i, \mathbf{p}_i) - d(\mathbf{a}_i, \mathbf{n}_i) + m, 0\} \quad (3.1)$$

where $d(\mathbf{x}_i, \mathbf{y}_i) = \|\mathbf{x}_i - \mathbf{y}_i\|_2^2$, $\mathbf{a}_i = f(\mathbf{I}_i)$, $\mathbf{p}_i = f(\mathbf{I}_i^+)$ and $\mathbf{n}_i = f(\mathbf{I}_i^-)$ are the outputs of a network f for inputs \mathbf{I}_i , \mathbf{I}_i^+ and \mathbf{I}_i^- respectively. \mathbf{I}_i , \mathbf{I}_i^+ and \mathbf{I}_i^- are the anchor, positive and negative example from the i -th triplet. m is a parameter that controls the gap between the distances of the two image pairs, and N is the sample size. The Triplet network model has been widely used in deep metric learning and has achieved promising results [96][102]. However, by using a fixed margin parameter, the cost function in Eq. (3.1) assumes equal ranking of all positive examples relative to the anchor. This function does not reflect the ground truth when images exhibit a non-binary similarity ranking, and may in fact disturb training. We propose a simple but effective version of this function that varies the margin parameter based on degree of similarity.

3.2.2 Deep Hashing and Quantization

Deep representations are effective for accurate image retrieval but for large-scale image datasets, converting the representations into compact binary codes allows more efficient storage and retrieval.

The goal of deep hashing is to learn deep image representations and their binary codes while preserving similarity. Xia et al. [103] proposed an approach that splits the problem into two stages: learning similarity-preserving hash codes for training samples using a scalable coordinate descent method, and learning an image deep hashing function that fits the learned hash codes. Lai et al. [104] presented an approach that jointly learns deep features and a hashing function, inside a triplet formulation to enforce ranking.

Quantization methods, which represent each sample by a code that points to the nearest center in a codebook, have been shown to be superior to hashing methods in approximate nearest neighbour search. One of such methods, proposed by Cao et al. [105], learns a similarity-preserving, quantizable deep image representation by jointly

minimizing a pairwise cosine loss and product quantization loss, respectively. Later, Cao et al. [106] adopted an approach that transforms deep image representations into a new space, where similarity with Word2Vec label embeddings is enforced using an adaptive margin loss, and quantization is achieved by minimizing quantization error of approximate inner-product search. An approach by Liu et al. [30] utilises a novel triplet selection approach for selecting hard triplets, called *Group Hard*, used in conjunction with triplet quantization with weak orthogonality to reduce codebook redundancy. We adopt DTQ’s quantization and triplet selection strategies in our approach.

3.2.3 Multi-label image retrieval

Most deep metric learning and deep hashing methods define pairwise similarity coarsely on a single-label basis. That is to say, two images are considered similar if any one label matches and dissimilar if no labels match. However this definition fails to capture the similarity ranking of multi-object images, and falls short of the fine-grained scene comparison capabilities displayed by humans. Recently, several approaches have been proposed to address this issue. DSRH [107] uses a weighted ranking loss to constrain deep hash codes to follow a ranking based on the number of common labels between data points. Another approach, presented in [108], first generates region proposals and then aggregates them into separate deep representations, which are combined and mapped to a single semantic hash code. IDHN [109] is another method targeted at multi-label images, that employs a soft-similarity based on the cosine distance of label vectors, and a mean square error function that constrains hash-codes to preserve this similarity.

To the best of our knowledge, all existing methods take into account the presence or absence of certain labels, but not their multiplicity. We propose a more generalized similarity function that is not only sensitive to the presence or absence of labels, but also the number of occurrences of their corresponding objects in the image. We also propose an architecture that is primed for extracting such information. For simplicity, we disregard examples where all labels are missing.

3.2.4 Relation networks

A relation network (RN) [110] is a neural network module that is geared towards relational reasoning. An RN is defined simply by the composite function

$$RN(O) = f_{\phi} \left(\sum_{i,j} g_{\theta}(o_i, o_j) \right) \quad (3.2)$$

where $O = \{o_1, o_2, \dots, o_n\}$ is a set of objects, o_i is the i^{th} object and f_{ϕ} and g_{θ} are functions with parameters ϕ and θ respectively [110]. The function g_{θ} infers the ways in which two objects are related and f_{ϕ} uses the sum of these relations to make a decision. The two functions g_{θ} and f_{ϕ} are feed forward neural networks and θ and ϕ are the weights of these networks. A CNN augmented with an RN was shown to offer superior performance in tasks involving object comparison and counting. A two-stage variant of the RN architecture was also used in [111] to extract relation-aware features and retrieve images with similar object relationships. In our work, we use an RN module in parallel with a fully connected layer to equip our model with the capability to individualize and count objects.

3.2.5 Triplet selection strategies for triplet networks

Triplet networks, introduced in Section 3.2.1, require good triplet selection strategies to speed up convergence and achieve higher quality image embeddings. To date, numerous such strategies have been proposed. These can be roughly categorized into online [112–117] and offline [118, 119] strategies. Online strategies generate all possible triplets or a number of useful triplets for training within each mini-batch. This is more efficient as it is faster to mine optimal triplets within a mini-batch and this can be done on the fly, during training. The downside of this approach, however, is that a mini-batch often does not provide an accurate reflection of the properties of the embedding space and it is possible to overlook some important triplets. Offline triplet generation involves generating all the triplets before training, and each mini-batch contains a fixed number of triplets. This strategy suffers from being computation-heavy, since we only have two options: generate all possible

triplets or search for optimal triplets over the entire training set. Additionally, as the model becomes better at embedding images, some of the originally selected triplets become ineffective for training. This can be counteracted by regenerating triplets using the latest model checkpoint at regular intervals.

In order to achieve faster convergence, it is important to select triplets that yield a non-zero loss, as these are the only triplets that the network can learn from. If we assume the standard triplet ranking loss, this can be written as:

$$\{\mathbf{a}, \mathbf{p}, \mathbf{n} : d(\mathbf{a}, \mathbf{p}) - d(\mathbf{a}, \mathbf{n}) + m > 0\} , \quad (3.3)$$

where d is the Euclidean distance function, and \mathbf{a} , \mathbf{p} and \mathbf{n} are the embeddings for the anchor, positive and negative examples respectively. m is a chosen margin parameter. Given an anchor, there are several ways of selecting the negative and positive examples. Batch-hard [114] selects the hardest positive and negative examples in a batch, that is to say:

$$\begin{aligned} \arg \max_{\mathbf{p}} d(\mathbf{a}, \mathbf{p}) \\ \arg \min_{\mathbf{n}} d(\mathbf{a}, \mathbf{n}) \end{aligned} \quad (3.4)$$

Batch-all [113] generates all valid triplets in a batch and averages the loss over non-zero loss triplets. Schroff et. al [112] argue that selecting the hardest negatives can cause the model to collapse, and instead adopt an approach that selects all possible (\mathbf{a}, \mathbf{p}) pairs in a batch and for each of these pairs, selects a semi-hard negative such that:

$$d(\mathbf{a}, \mathbf{p}) - d(\mathbf{a}, \mathbf{n}) + m > 0, \quad d(\mathbf{a}, \mathbf{n}) > d(\mathbf{a}, \mathbf{p}) . \quad (3.5)$$

Liu et. al [30] propose a similar approach but instead of selecting a semi-hard negative, they randomly pick a negative from all hard and semi-hard negative examples. We adopt this approach in method I (section 3.3) of this chapter, and a similar but slightly different approach in method II (section 3.4). Figure 3.1 illustrates

the idea of hard, semi-hard, and easy triplets. As a rule, easy triplets are disregarded as they do not contribute to training the model.

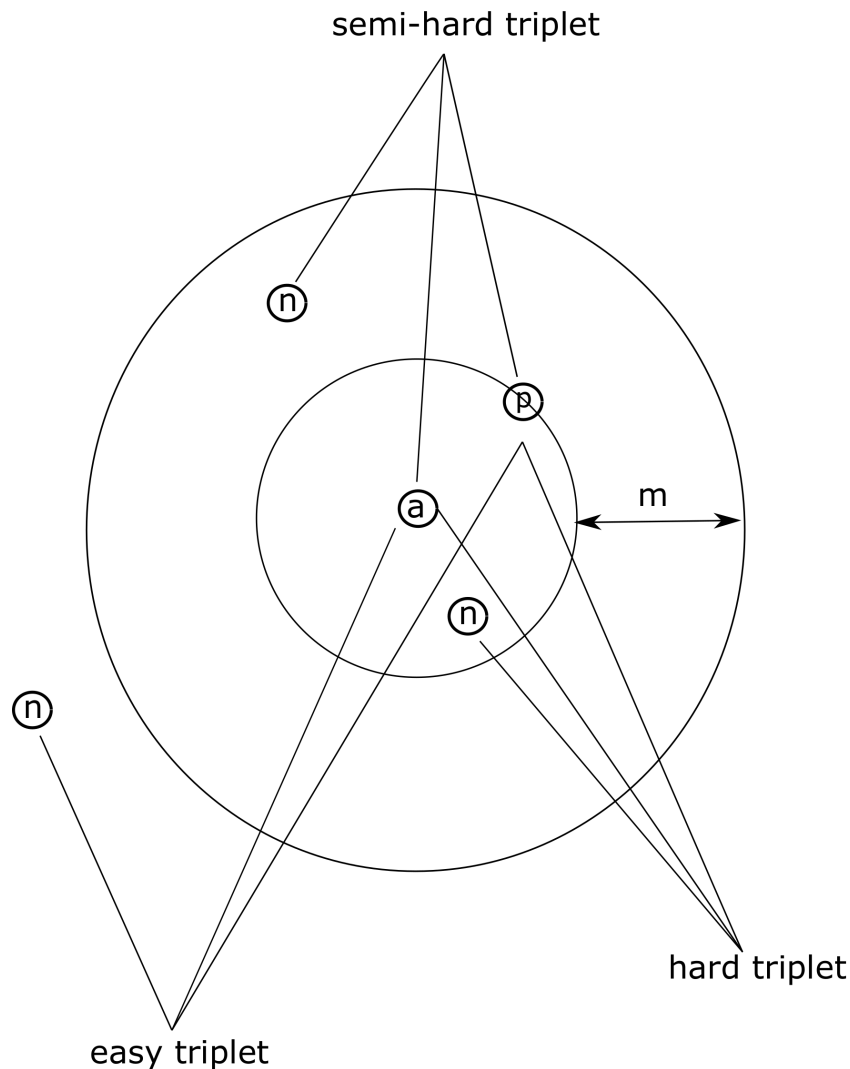


Fig. 3.1 An illustration of hard, semi-hard, and easy triplets

3.2.6 Losses for multi-label deep metric learning

As we mentioned in section 3.2.3, approaches that define pairwise similarity of multi-label images as a continuous variable are few. In this section, we briefly introduce the approaches that adopt this approach and how they modify the loss function to account for image similarity levels.

Zhao et al. [107] weight the entire triplet ranking loss with a coefficient based on the number of shared labels between a database point and a query. The weight term is defined as:

$$w = \frac{2^{W_{ap}} - 2^{W_{an}}}{Z} \quad (3.6)$$

where W_{ap} and W_{an} are the number of shared labels between the anchor, and the positive and negative samples respectively. Z is a constant of normalization. The loss then becomes:

$$L = w \max\{d(\mathbf{a}, \mathbf{p}) - d(\mathbf{a}, \mathbf{n}) + m, 0\} \quad (3.7)$$

The effect is that the larger the anchor-positive similarity is than the anchor-negative similarity, the bigger the gradients generated through the network.

Zhang et al. [109] propose a pairwise loss that constrains the inner product of a pair of binary embeddings to equal a value weighted by a similarity coefficient. The loss is defined as:

$$L = \left(\frac{\langle x_i, x_j \rangle + D}{2} - s_{ij}D \right)^2 \quad (3.8)$$

where $\langle x_i, x_j \rangle$ is the inner product between binary embeddings x_i and x_j , D is the dimensionality of the embeddings, and s_{ij} is defined as:

$$s_{ij} = \frac{\langle y_i, y_j \rangle}{\|y_i\| \|y_j\|} \quad (3.9)$$

which is the cosine similarity between label vectors y_i and y_j . The loss function in equation 3.8 forces the pairwise hamming distances of network-generated binary codes to preserve the ranking of their corresponding cosine similarities. In this chapter, we present two methods that involve weighting different parts of the triplet ranking loss with a similarity coefficient.

3.2.7 Variational auto-encoders

Auto-encoders (AE), introduced in the 1980s [120] have proven to be powerful unsupervised models for dimensionality reduction, image de-noising, compression, and feature extraction, among other tasks. The standard AE architecture consists of an encoder module terminating in a latent layer, followed by a decoder module that reconstructs the original input from the latent layer. An AE learns by mapping input to the latent space, and then mapping it back to the original space such that the reconstruction error between the input and output is reduced. Figure 3.2 illustrates the basic idea of an auto-encoder.

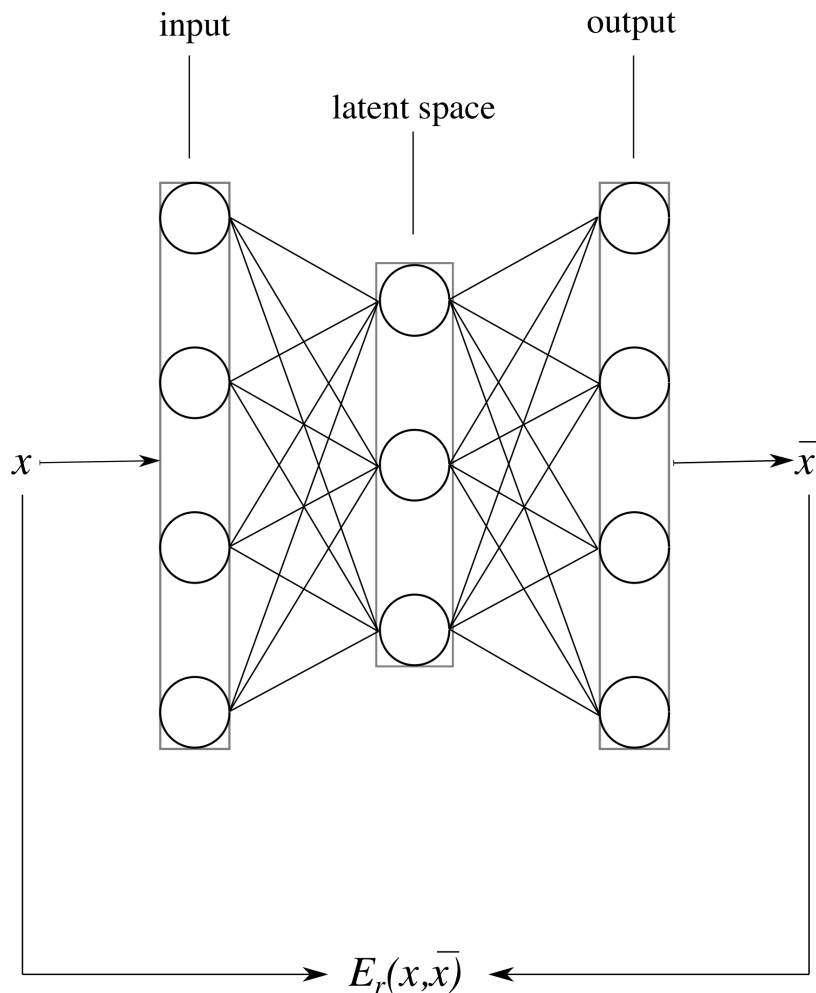


Fig. 3.2 An illustration of a basic auto-encoder

For an AE to perform any of the above mentioned tasks, a constraint must be placed on the latent layer. In de-noising and compression the latent layer is constrained to have a lower dimension than the input, forcing the encoder to learn a recoverable lower dimensional representation of the input [121, 122]. In some feature extraction AEs, a sparsity constraint is applied to the latent layer, creating an information bottleneck without reducing dimensionality of the original input [123, 124].

Variational auto-encoders (VAE), first introduced by Kingma et al. [50], have quickly gained popularity as generative models. Instead of directly assigning values to the latent variables, the encoder part of a VAE estimates the latent state as a probability distribution. The decoder then samples from this distribution and generates an example. The model typically learns by minimizing the following penalty:

$$L_{vae} = E_r(x, \bar{x}) + \sum_{i=1}^{|l_h|} KL(N(\mu, \sigma^2) || N(0, 1)) \quad (3.10)$$

where x is the input, \bar{x} is the reconstructed input, and $|l_h|$ is the size of the latent layer. The first term is a reconstruction error between the original input and reconstructed input, and the second term is the Kullback-Leibler (KL) divergence between the Gaussian distribution of the latent state, estimated by the encoder, and the standard Gaussian distribution. Figure 3.3 illustrates the idea of a VAE.

Graphical models have been used to capture inter-label relations in order to deal with incomplete labeling in multi-label datasets [64, 67]. We present one such approach in chapter 1, using a restricted Boltzmann machine (RBM). In recent years, VAEs have replaced RBMs in most applications because they are easier to optimize, versatile and extensible. We present a method in this chapter that employs a VAE to capture inter-label relations and complement label vectors with additional relevant labels. We show that training an embedding network on these new labels yields higher retrieval accuracy.

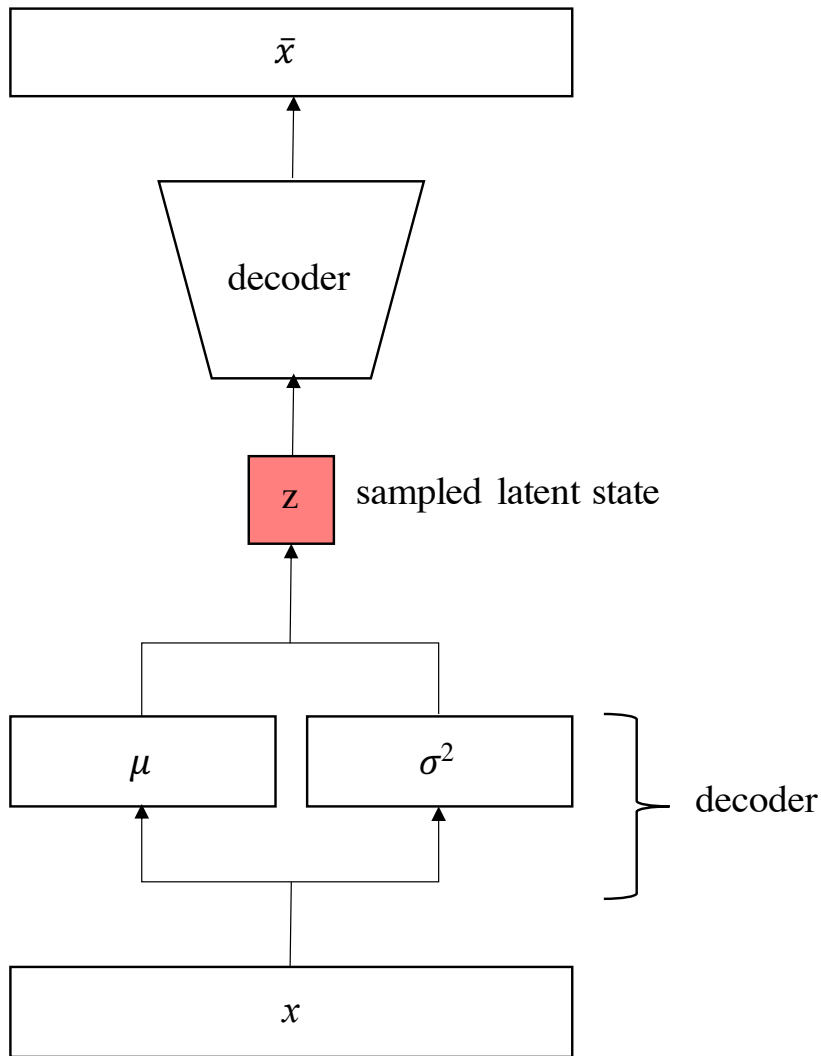


Fig. 3.3 The architecture of a variational auto-encoder

3.3 Proposed Method I

Our first proposed method is illustrated in Figure 3.4. In this section, we describe the various components that make up our model.

3.3.1 Triplet generation scheme

We adopt *Group-Hard*, a triplet selection scheme introduced in [30]. It involves dividing the training labels into P partitions, and randomly selecting a non-easy negative for each anchor-positive pair in each partition. Fresh triplets are generated

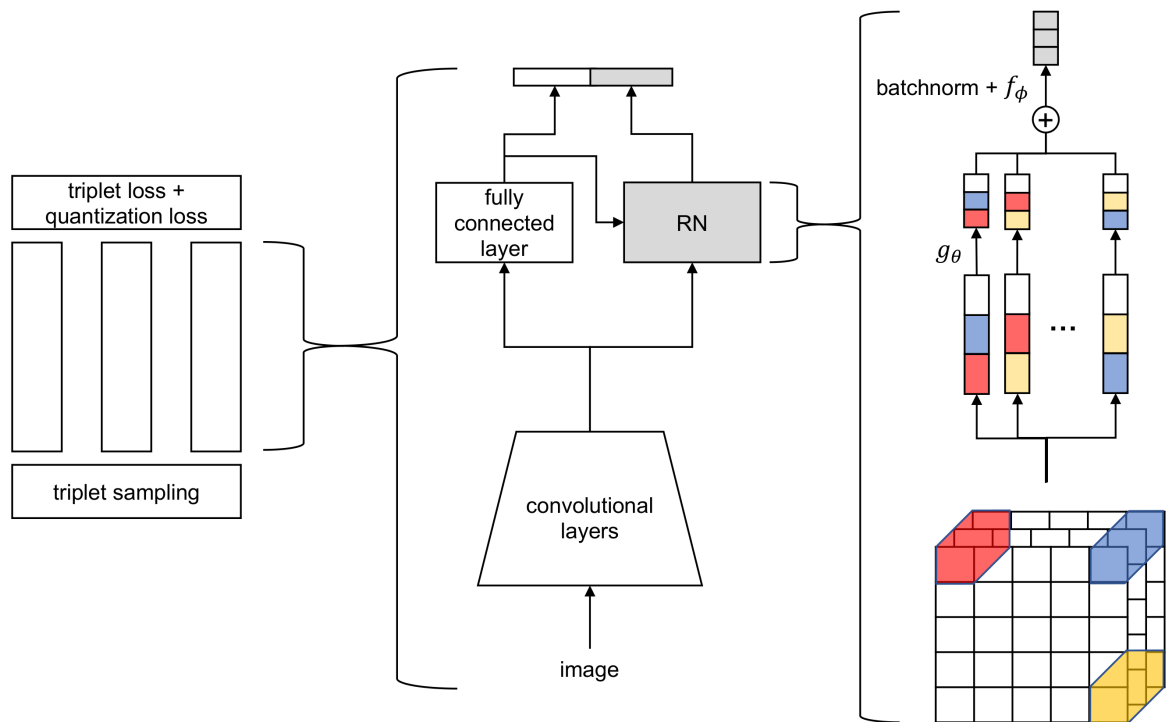


Fig. 3.4 An illustration of our architecture

at the beginning of each epoch, and once the number of triplets falls below a pre-determined threshold, the number of groups is halved to ensure that a sufficient number of triplets is available for training in each epoch. The strategy is summarized in Algorithm 2

Algorithm 2: The Group-hard triplet generation algorithm

Input: image embeddings $X = \{\mathbf{x}_i\}_{i=1}^N$
Input: image similarities $S = \{\mathbf{x}_i\}_{i=1, j=1}^N$
Randomly split X into P partitions of equal size
 $T \leftarrow \emptyset$
for $p=0$ to P **do**
 foreach $\{\mathbf{x}^a, \mathbf{x}^p \in G_p : s_{ap} > 0\}$ **do**
 $T_{ap} \leftarrow \emptyset$
 foreach $\{x^n \in G_p : s_{an} == 0\}$ **do**
 if $\|\mathbf{x}^a - \mathbf{x}^n\|_2^2 - \|\mathbf{x}^a - \mathbf{x}^p\|_2^2 < m$ **then**
 $\backslash\backslash$ \mathbf{x}^p and \mathbf{x}^n have the wrong distance-wise ranking w.r.t. \mathbf{x}^a
 $T_{ap} \leftarrow T_{ap} \cup \{ \langle \mathbf{x}^a, \mathbf{x}^p, \mathbf{x}^n \rangle \}$
 end
 end
 $T \leftarrow T \cup \text{rand}(T_{ap})$
end
end

3.3.2 Feature extractor

Without loss of generality, we follow [105, 106, 30, 109] and adopt AlexNet [3] as our base CNN architecture. The feature extractor consists of five convolutional layers with a ReLU non-linearity after each layer and max-pooling after the first, second, and fifth convolutions. Our features are the output of the fifth layer, consisting of 256 convolutional feature maps, max-pooled to a size of 6×6 . We use this architecture for simplicity and consistency with similar proposed methods, but other CNN architectures could just as easily be used with our approach. To prove this, we perform additional experiments using VGG16 [125] as the underlying architecture and show that our method outperforms the closest competing method with a larger margin.

3.3.3 Feature aggregation

The features from the last convolutional layer are fed into a feature aggregation module that consists of a fully connected section and an RN in parallel. The fully connected layers combine all the locations from the incoming feature maps into a global, compact representation, whereas the RN encodes object relationships, and more importantly, instances. The fully connected section is made up of two layers from the AlexNet architecture, *fc-6* and *fc-7*, and terminates in a linear layer that outputs a 64-dimensional vector, which we will denote \mathbf{x}^{FC} . The RN takes as its input O , the set of point-wise features from the last convolutional layer. It is a slight modification of equation (3.2), and is defined as:

$$\mathbf{x}^{RN} = RN(O) = f_{\phi} \left(\sum_{i,j} g_{\theta}(\mathbf{x}^{FC}, \mathbf{o}_i, \mathbf{o}_j) \right) \quad (3.11)$$

where \mathbf{x}^{FC} is appended to add global context to the processing of each pair. g_{θ} and f_{ϕ} consist of three fully connected layers each. We apply a ReLU activation after each layer except the last layer, which outputs a 64-dimensional vector, \mathbf{x}^{RN} . We apply a batch normalization layer before f_{ϕ} , as we found that the summation in equation (3.11) leads to large gradients, which cause the model to diverge. The final compact deep representation is a concatenation of the outputs of the two parallel modules:

$$\mathbf{x} = [\mathbf{x}^{FC}, \mathbf{x}^{RN}] \quad (3.12)$$

where $[\cdot, \cdot]$ denotes concatenation.

3.3.4 Triplet quantization

Out of the quantization schemes presented in section 3.2.2, the one introduced in [30] most fits our approach, since it is targeted at the triplet embedding case. It also includes an orthogonality constraint, which prevents codeword duplication across codebooks. Furthermore, it was proven to yield better results than the product quantization presented in [105]. Therefore, we adopt the triplet quantization scheme presented in [30] with a slight modification. It involves learning a set of M codebooks

$C = [C_1, \dots, C_M]$, with each codebook containing K D -dimensional cluster-centroid vectors $C_m = [C_{m1}, \dots, C_{mK}]$. Then for each deep representation, we learn a binary assignment vector \mathbf{b}_i , made up of M indicator vectors where each indicator vector \mathbf{b}_{mi} selects 1 of the K codewords in the m -th codebook to approximate the i -th deep representation \mathbf{x}_i . The set of codebooks C is shared across all triplets to enable knowledge sharing. To reduce redundancy, an orthogonality penalty is applied across the M codebooks. The entire quantization objective is defined as:

$$Q = \sum_{i=1}^N \left\| \mathbf{x}_i - \sum_{m=1}^M C_m \mathbf{b}_{mi} \right\|_2^2 + \gamma \sum_{m=1}^M \sum_{n=1}^M \|C_m^\top C_n - I\|_2^2 \quad (3.13)$$

where the second term, weighted by γ , is the orthogonality penalty. N is the number of samples in a mini-batch, including anchors, positives, and negatives. Thus far, this is similar to the approach presented in [30]. In our case, we found that it was better to split the quantization of \mathbf{x}^{FC} and \mathbf{x}^{RN} , as these branches learn different deep representations. We split the set of codebooks into two, where each part has $M/2$ codebooks and each codebook has K $D/2$ -dimensional centroid vectors. Before splitting, the number of bits used for each code is $M \log_2 K$. Assigning $M/2$ codebooks to each part of the output vector means the total number of bits remains $2(M/2) \log_2 K = M \log_2 K$. This allows us to preserve the number of bits used for each code, but requires the total number of codebooks M to be even. Hence equation (3.13) becomes

$$Q = \sum_{* \in (FC, RN)} \left(\sum_{i=1}^N \left\| \mathbf{x}_i - \sum_{m=1}^{M/2} C_m^* \mathbf{b}_{mi}^* \right\|_2^2 + \gamma \sum_{m=1}^{M/2} \sum_{n=1}^{M/2} \|C_m^{*\top} C_n^* - I\|_2^2 \right) \quad (3.14)$$

During retrieval, we follow [105, 106, 30] and compute the Asymmetric Quantizer Distance (AQD), which is defined as the inner product similarity between the embedding of a given query and the vector obtained by reconstructing a database

point \mathbf{x}_n from its binary code. In our case, taking into account the splitting of the codebooks, it becomes:

$$\text{AQD}(\mathbf{q}, \mathbf{x}_n) = \sum_{* \in (FC, RN)} \mathbf{z}_q^{*\top} \left(\sum_{m=1}^{M/2} C_m^* \mathbf{b}_{mn}^* \right) \quad (3.15)$$

where \mathbf{z}_q is the embedding of query \mathbf{q} . With AQD, we can pre-compute all possible values of the inner product $\mathbf{z}_q^{*\top} C_m^* \mathbf{b}_{mn}^*$ and store them in a query specific $M \times K$ lookup table from which we calculate the AQD between the query and each database point. This only requires M table lookups and additions for each database point, which leads to a reduced computational cost of retrieval.

3.3.5 Triplet loss with an adaptive margin

Given two label vectors \mathbf{y}_i and \mathbf{y}_j , we compute a similarity measure based on the Jaccard similarity coefficient [126]. The general idea is illustrated in Figure 3.5 . Using label vectors, we formally define this metric as:

$$s_{ij} = \frac{\sum_{n=1}^{|S|} \min(y_{in}, y_{jn})}{\sum_{n=1}^{|S|} \max(y_{in}, y_{jn})}. \quad (3.16)$$

where $|S|$ is the size of the label space under consideration and $y_{in} \geq 0$ is the number of occurrences of the n -th label for the i -th example. To put it simply, this is the intersection over union (IOU) between two label vectors, which takes the number of occurrences of each object into account. This similarity measure has the nice property of being in the range $[0, 1]$, is robust against sparsity in label vectors, and treats both binary (for multi-label) and non-binary (for multi-object) label vectors. We modify the triplet loss function in Equation 3.1 to adjust the margin based on the similarity of the positive pair as follows

$$L = \sum_{i=1}^N \max\{d(\mathbf{x}_i^a, \mathbf{x}_i^p) - d(\mathbf{x}_i^a, \mathbf{x}_i^n) + s_{a,p}^i m, 0\} \quad (3.17)$$

where $s_{a,p}^i$ is the similarity between the anchor and positive from the i -th triplet. This is slightly similar to the function presented in [106]. However, there are some

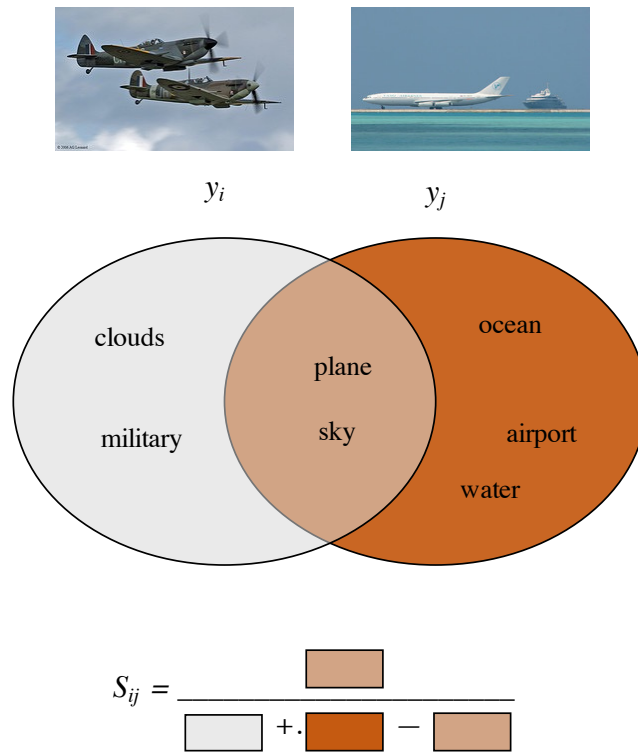


Fig. 3.5 An graphical illustration of the definition of the similarity function

key differences; the loss presented in [106] is a function of cosine similarities with no constant part, whereas ours uses our proposed similarity measure to scale a constant margin, allowing us to use large margin values. Additionally, they consider cosine similarities between deep representations and word embeddings whereas we consider relative distances between deep representations. The final loss function is then defined as

$$E = L + \lambda Q, \quad (3.18)$$

where λ is a parameter that controls the strength of the quantization loss term.

3.4 Proposed Method II

In this section, we present several incremental improvements over the method proposed in section 3.3. These include a new triplet generation scheme, a corresponding modification to the loss function, and a strategy for reconstructing labels under an assumption of incompleteness.

3.4.1 Triplet generation scheme

In the first method, we utilize a triplet selection scheme adopted from [30]. Based on that, we introduce a simple but effective triplet selection scheme we developed in order to explicitly enforce the correct ranking of retrieved results based on the continuous similarity level of database points. Whereas the approach presented in the authors' paper assumes that relative to a chosen anchor image, an example can be either positive or negative, our approach only considers relative similarity. Relative to a query, the goal is then to find two examples that have incorrect distance-wise ranking, given their ground-truth similarity to the query. We select a small m as the minimum margin with which to separate $d(x^a, x^i)$ and $d(x^a, x^j)$, for any triplet of embeddings x^a , x^i and x^j such that $s_{ai} \neq s_{aj}$. We still divide the training data into a number of P partitions, but unlike in Algorithm 2, this number remains fixed throughout training. This is because the number of triplets generated by our algorithm does not decrease substantially. The triplet selection algorithm is summarized in algorithm 3.

Algorithm 3: Our triplet generation algorithm

Input: image embeddings $X = \{\mathbf{x}_i\}_{i=1}^N$
Input: image similarities $S = \{s_{ij}\}_{i=1,j=1}^N$
Randomly split X into P partitions of equal size
 $T \leftarrow \emptyset$
for $p=0$ to P **do**
 foreach $\{\mathbf{x}^a, \mathbf{x}^i \in G_p : s_{ai} > 0\}$ **do**
 $T_{ai} \leftarrow \emptyset$
 foreach $x^j \in G_p$ **do**
 if $s_{aj} < s_{ai}$ **and** $\|\mathbf{x}^a - \mathbf{x}^j\|_2^2 - \|\mathbf{x}^a - \mathbf{x}^i\|_2^2 < m$ **then**
 $\backslash\backslash$ \mathbf{x}^i and \mathbf{x}^j have the wrong distance-wise ranking w.r.t. \mathbf{x}^a
 $T_{ai} \leftarrow T_{ai} \cup \{ \langle \mathbf{x}^a, \mathbf{x}^i, \mathbf{x}^j \rangle \}$
 end
 end
 $T \leftarrow T \cup \text{rand}(T_{ai})$
 end
end

3.4.2 Improved triplet loss

Since the objective is simply to encourage the correct ranking between a pair of examples relative to the anchor, we no longer implement an adaptive margin. Since it is possible to have a triplet such that $s_{ai} > s_{aj} > 0$, we can directly control the ordering of positive results. There is hence no need to vary the margin to account for the different levels of similarity that a "positive" example might have. Instead, we introduce the similarity coefficient into the loss function as follows:

$$L = \sum_{i=n}^N \max\{w_n \cdot [d(\mathbf{x}_n^a, \mathbf{x}_n^i) - d(\mathbf{x}_n^j, \mathbf{x}_n^i) + m], 0\} \quad , \quad (3.19)$$

$$w_n = W_{ai}^n \cdot (s_{ai}^n - s_{aj}^n) \quad (3.20)$$

where W_{ai}^n is the number of shared labels between the anchor and sample i . Note here that $s_{ai}^n > s_{aj}^n$, therefore w_n is always positive. The weight term has the following effect:

- It yields big gradients for triplets with a large difference in relative similarity, as opposed to the adaptive margin implemented in section 3.3.5, which does not propagate back into the network.
- It attributes more importance to triplets that involve large numbers of shared labels

3.4.3 Label reconstruction using a variational auto-encoder

We introduced VAEs in section 3.2.7. Assuming the existence of examples with missing labels in the training data, we use a VAE to learn label correlations in the label space and add supplemental labels to the training samples. To teach the model to recover missing labels, we introduce noise in the input by zeroing out a percentage of the labels. We train the model using the following function:

$$L_{vae} = E_r(\mathbf{y}, \bar{\mathbf{y}}) + \gamma \cdot \sum_{i=1}^{|\mathcal{H}|} KL(N(\mu, \delta^2) || N(0, 1)) \quad (3.21)$$

where γ is a parameter that controls the strength of the KL divergence regularization term. \mathbf{y} is the label vector (before label removal) and $\bar{\mathbf{y}}$ is the label vector reconstructed from the latent state by the decoder, in form of probabilities. At test time, we calculate the reconstructed label vector as follows:

$$\hat{\mathbf{y}} = \left\lfloor \max(y_i, \bar{y}_i) + 1 - t \right\rfloor_{i=1}^{|\mathcal{S}|} \quad (3.22)$$

where t is a threshold parameter. Essentially, this function just adds any new labels that are predicted with probability $\geq t$ to the original label vector. We later analyze the effect of this parameter on model performance.

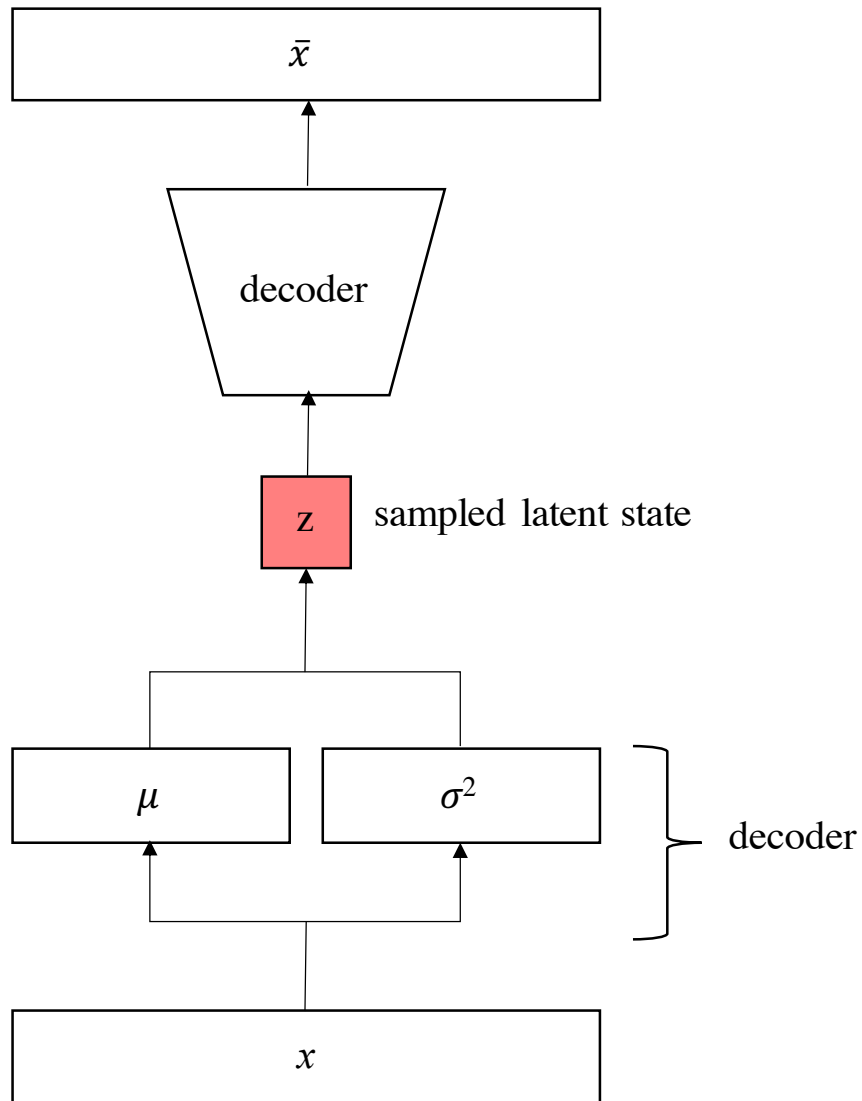


Fig. 3.6 Examples of samples with missing labels in the NUSWIDE dataset

3.5 Experiments

3.5.1 Datasets

We train our model on two benchmark datasets, NUS-WIDE [127], and MS-COCO [34].

NUS-WIDE contains 269,648 images, where each image is labeled with a subset of 81 concepts. We follow [105, 106, 30] and randomly sample 5000 images as queries,

10,000 images for training, and use the rest as database data. This dataset is an example of a *multi-label* image dataset. Two sample images are shown in Figure 3.7.



(a) sun, sunset, sky, clouds



(b) buildings, tower, nighttime, sky, clouds

Fig. 3.7 Two sample images from the NUS-WIDE dataset

MS-COCO contains 82,783 training images and 40,504 validation images, where objects in each image fall in one of 80 categories. We follow the procedure in [30] and randomly sample 5000 images as queries, 10,000 images for training, and use

the rest as database data. This dataset is an example of a *multi-object* image dataset. We show two sample images from this dataset in Figure 3.8.



(a) person (2), frisbee (1)



(b) cow (8)

Fig. 3.8 Two sample images from the COCO dataset (The number in brackets is the number of occurrences of each object in the target image)

3.5.2 Training

For method I, we set the number of partitions for triplet generation $P=20$ for MS-COCO and $P=50$ for NUS-WIDE. Following [106, 30], we fix the number of

codewords as $K = 256$. Thus for M codebooks, the binary code for each sample is $8M$ -bit long. We quantize the *RN* and *FC* components of the output vector separately, and assign an equal number of codebooks to each, so that the final number of bits becomes $2 \times 8M$.

During training, we fine-tune convolutional layers from *conv-3* to *conv-5*, and fully connected layers *fc-6* and *fc-7* of AlexNet. On VGG16, we only train the fully connected layers *fc-6* and *fc-7*. The last fully connected layer, and the entire RN portion of the architecture are trained starting from scratch, using a random Gaussian initialization. We use stochastic gradient descent (SGD) with a momentum of 0.9 as our optimizer, and start with a learning rate of 10^{-5} , later scaling it down by 10^{-1} . We determined the rest of the parameters by doing parameter searches within reasonable ranges. We set $m = 15$ by doing a parameter search in the range $[5, 30]$, and $\lambda = 0.001$ by doing a parameter search in the range $[0, 1]$. We initially held back 1000 samples from the training set for validation and hyper-parameter tuning, then used the entire training set to train the final model.

For method II, we use a VAE with layers $\{(in:81, out:128), (in:128, out:256), \mathbf{(in:256, out:256)}, (in:256, out:256), (in:256, out:128), (in:128, out:81)\}$, where the layer in bold is the latent layer. We set the KL divergence regularizer strength as $\gamma = 1e - 05$. We optimize the model using SGD with momentum, with an initial learning rate of 0.1, which is decayed to 0.01 at 50000 iterations. For the metric learning model, we set P as 25, and the margin as $m = 8$. We noticed that the contribution of the RN part of our architecture is minimal, so we drop it for method II, to focus more on the contribution of the loss function and triplet selection scheme.

3.6 Results and Analysis

3.6.1 Method I

Table 3.1 shows the MAP results of our method compared to some state-of-the-art methods, and it shows that our method outperforms the comparison methods, with margins as large as 5% on NUS-WIDE and 12.2% on MS-COCO. Specifically, our

Table 3.1 Mean average precision (MAP) for different number of bits on two benchmark datasets. The underlying architecture is AlexNet.

Method	NUS-WIDE		MS-COCO	
	16 bits	32 bits	16 bits	32 bits
DQN [105]	0.735	0.752	0.653	0.685
DVSQ [106]	0.790	0.797	0.712	0.720
DTQ [30]	0.798	0.801	0.760	0.767
MLMO1 (Ours)	0.803	0.805	0.806	0.807

Table 3.2 Mean average precision (MAP) for our method and several of its variants. The underlying architecture is AlexNet

Method	NUS-WIDE			MS-COCO		
	deep feat	16 bits	32 bits	deep feat	16 bits	32 bits
MLMO1-RN	0.799	0.789	0.763	0.775	0.773	0.774
MLMO1-FC	0.799	0.797	0.799	0.799	0.794	0.795
MLMO1-cosine	0.800	0.796	0.800	0.773	0.774	0.775
MLMO1	0.805	0.803	0.805	0.806	0.806	0.807

method shows gains of 0.4% and 4% over DTQ, a method which shares the most similarities with our method. This MAP is calculated using the coarse definition of similarity, that is, two data points are considered similar if they share at least 1 label, and dissimilar otherwise. This shows that for multi-label data, using the coarse definition of image similarity for triplet sampling hurts performance, even if we maintain the same definition during evaluation. We also note that our method outperforms the competing methods by a larger margin on the MS-COCO dataset than on the NUS-WIDE dataset. This proves that while our method offers competitive performance for multi-label data, there is marked improvement over previous methods when dealing with multi-object data.

Table 3.3 compares the mean average precision of our method against the closest competing method, with VGG16 as the underlying architecture. We note that our

Table 3.3 Mean average precision for our method and the best competing method with VGG16 as the underlying architecture.

Method	NUS-WIDE			MS-COCO		
	deep feat	16 bits	32 bits	deep feat	16 bits	32 bits
DTQ [30]	0.806	0.801	0.803	0.803	0.801	0.802
MLMO1	0.835	0.828	0.833	0.856	0.859	0.859

Table 3.4 Mean average precision@0.50 (MAP@0.50) for our method and several of its variants on the NUS-WIDE dataset (AlexNet architecture).

Method	NUS-WIDE			MS-COCO		
	deep feat	16 bits	32 bits	deep feat	16 bits	32 bits
DTQ [30]	0.398	0.392	0.400	0.121	0.117	0.116
MLMO1- RN	0.454	0.366	0.354	0.123	0.119	0.170
MLMO1- FC	0.451	0.449	0.447	0.173	0.170	0.170
MLMO1- cosine	0.455	0.444	0.447	0.148	0.145	0.146
MLMO1	0.454	0.449	0.451	0.173	0.168	0.170

method outperforms DTQ with a larger margin (about 3% on NUS-WIDE and 5% on MS-COCO) than when using the AlexNet architecture. This shows that our method is more able to utilize the increased network capacity to yield a corresponding boost in performance than other methods. At the very least, it shows that our method maintains its lead in performance despite the underlying architecture.

Table 3.2 shows the MAP results for ablative experiments on our architecture. The results justify some architectural choices we made for our final model. MLMO-RN is a variant of our method using only the RN module after the feature extractor, whereas MLMO-FC is a variant using only the fully connected module. MLMO-cosine is a variant that uses cosine distance to compute similarity between label vectors, like some other works propose [106, 109]. Our final proposed model has the highest MAP for all numbers of bits on both datasets. It is worth noting that MLMO-FC, a version of our model with the RN module removed, still performs better than MLMO-cosine in multiple configurations (all configurations on MS-COCO, 16-bits

on NUS-WIDE) and offers competitive performance in the rest of the configurations. This points to the importance of our IoU-based similarity measure.

Since the deep feature can also be directly used in nearest neighbour search, we include this result to show the information loss incurred due to the quantization step. We note that this is something other works on the same topic fail to include. Our results show no drop in MAP between the deep features and the 32-bit quantized features. This proves that our learned deep feature can be quantized while preserving similarity between images.

Table 3.4 shows MAP@0.50 results for the same ablative experiments shown in table 3.2. This is similar to the standard MAP, but we only consider a retrieved image as positive if its similarity s_{qi} with the query, defined in equation 3.16, is greater than 0.50. We also include results for the baseline, DTQ. We include this metric to investigate how well our model learns the objective based on the similarity measure we proposed, and how well it performs in multi-label and multi-object retrieval. Under each configuration, a variant of our method outperforms the baseline, which proves the superiority of our method on both multi-label and multi-object image retrieval. Except for the deep feature-based nearest neighbour search on NUS-WIDE, variants of our method using the proposed similarity metric give the best MAP@0.50. It is also worth noting that on the NUS-WIDE dataset, where label vectors are binary and there is no supervised information of object multiplicity, the margin between the models using cosine similarity and Jaccard index is small (0.4 under 32 bits%). On MS-COCO dataset, however, the margin is large (2.4%), emphasizing the superiority of our proposed similarity metric for multi-object retrieval.

Figure 3.9 shows the top-10 retrieved results for our method, its cosine variant, and the baseline method. The IOU between the label vectors of the retrieved image and the query is shown against each result. All the examples use 32-bit quantized features. For the NUS-WIDE example, the cosine and Jaccard similarity variants of our model offer comparative performance, with our approach (MLMO) returning slightly more results with an IOU over 0.5. Both variants perform better than the baseline. The MS-COCO example shows that our method is more able to factor in

object multiplicity during retrieval than both the cosine similarity-based variant and the baseline.















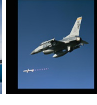



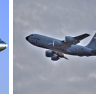














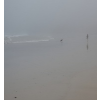



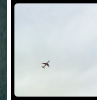
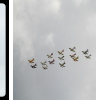




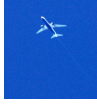
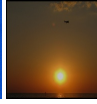

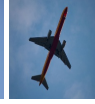









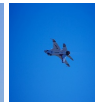




Query	Top 10 retrieved images (s_{qi})										
NUS-WIDE 											MLMO1
	(0.6)	(0.8)	(0.8)	(0.8)	(0.0)	(0.67)	(0.5)	(0.8)	(0.5)	(0.67)	
	military vehicle sky clouds plane										
(0.6)		(0.6)	(0.6)	(0.6)	(1.0)	(0.8)	(0.4)	(0.8)	(0.4)	(0.8)	
DTQ											
	(0.5)	(0.5)	(0.3)	(0.5)	(0.5)	(0.6)	(0.57)	(0.4)	(0.29)	(0.5)	
	MS-COCO 										
(0.75)		(0.17)	(0.0)	(0.67)	(0.17)	(0.0)	(0.17)	(0.43)	(0.67)	(0.17)	
airplane (x6)											
	(0.83)	(0.17)	(0.17)	(0.17)	(0.17)	(0.17)	(0.17)	(0.33)	(0.17)	(0.17)	
	DTQ										
(0.0)		(0.33)	(0.17)	(0.33)	(0.17)	(0.17)	(0.17)	(0.33)	(0.17)	(0.17)	

Fig. 3.9 Retrieval examples for two variants of our method and the baseline method (using 32-bit quantized features)

3.6.2 Method II

Effectiveness of label reconstruction

We present some examples to demonstrate how the VAE adds relevant missing labels to some examples in the NUSWIDE dataset. Figures 3.10 and 3.10 demonstrates the ability of our trained VAE to suggest supplemental labels that are relevant to the images. From Figure 3.10, we observe that label reconstruction increases the similarity s for training images that were originally dissimilar. This encourages the correct semantic ranking of retrieved results. In this specific case, the similarity coefficient of the examples shown in Figures 3.10e and 3.10f changes from $s = 0$ to $s = 0.5$.

Retrieval performance

In this section, present the quantitative retrieval performance of method II in terms of MAP. We achieve an improvement of 0.4% over method I on the NUSWIDE dataset, and 0.4% over the closest competing method. More notable is the improvement in MAP@0.5, which requires that a database point have $s \geq 0.5$ to be a positive result. On this metric, method II offers an improvement of 2.1% over the first method, and 35% over the closest competitor. This shows that the improvements introduce in method II are necessary for superior performance in multi-label image retrieval. Table 3.6 shows the effect of the threshold parameter t on the retrieval performance of the model. Note that $t = 1$ is the case where the original labels are maintained. We note that there is still an improvement in MAP@0 (single label similarity) in this case, which means that our triplet generation strategy and loss function alone lead to a performance improvement. However, a bigger improvement is observed with label reconstruction on the MAP@0.5 metric (multi-label similarity) than without. This shows that incomplete labeling affects the accuracy of s in describing the true similarity of multi-label images in the training data. We also observe that for the most part, 16-bit codes yield better results than 32-bit codes. This means that the feature learned by method II can be quantized with fewer centroid vectors, which could lead to better retrieval speed.

(a) military, sky, plane, **clouds**(b) coral, fish, water, **animal**(c) lake, water, clouds, **sky**, **ocean**(d) garden, clouds, grass, **sky**

Fig. 3.10 Examples of label supplementation (text in bold indicates new labels suggested by the VAE)

3.7 Conclusion

In this chapter, we introduced a deep metric learning approach focused on scene comparison between multi-label and multi-object images. We define a simple and

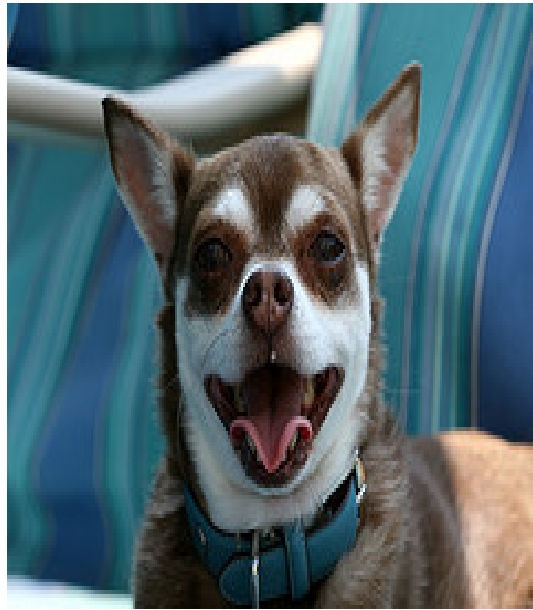
(e) bird, **animal**(f) dog, **animal**

Fig. 3.10 More examples of label supplementation (text in bold indicates new labels suggested by the VAE). Originally dissimilar images become partially similar

Table 3.5 Mean average precision (MAP) for method II compared to method I. The underlying architecture is AlexNet.

Method	NUS-WIDE (MAP@0)			NUS-WIDE (MAP@0.5)		
	deep feat	16 bits	32 bits	deep feat	16 bits	32 bits
MLMO1	0.805	0.803	0.805	0.454	0.449	0.451
MLMO2	0.810	0.810	0.809	0.247	0.475	0.472

effective similarity measure that enables our model to learn similarity preserving deep representations for multi-label and multi-object images. We show that our method offers better performance than some state-of-the-art methods on two benchmark datasets. Our work represents an important step towards fine-grained scene comparison in CBIR. A possible extension of this work would be to introduce a similarity measure that also takes into account the relations between objects in an image, and an architecture that reliably preserves the similarity of entire scene graphs.

Table 3.6 Effect of threshold parameter t on model performance (AlexNet).

t	NUS-WIDE (MAP@0)			NUS-WIDE (MAP@0.5)		
	deep feat	16 bits	32 bits	deep feat	16 bits	32 bits
0.4	0.810	0.810	0.807	0.241	0.465	0.460
0.5	0.810	0.810	0.809	0.247	0.475	0.472
1.0	0.808	0.807	0.807	0.248	0.452	0.454

Chapter 4

Conclusion

4.1 Summary

In this study, we developed techniques for dealing with multi-label and multi-object image data in deep learning algorithms. In chapter 1, we introduced multi-label image data and why it is important in real world applications. We presented the unique challenges that emerge when training deep learning algorithms on multi-label data, and established that there is need to pay special attention in such cases. In chapter 2, we tackled the task of multi-label image annotation using a convolution neural network (CNN), and focused on how to deal with missing labels, which is usually an inevitable problem in multi-label image datasets. We presented two possible solutions, both of which showed promise in smoothing out the effect of incomplete labeling in training data. In chapter 3, we changed the target task to image retrieval through deep metric learning, and looked at how we can adapt the loss function, architecture and sample selection strategy to the case of multi-label data. We proved on popular benchmarks that our approaches are effective in increasing the performance and quality of retrieval offered by CNNs.

4.2 Findings

We presented two approaches in chapter 2. One involved using a regularization term that leverages label co-occurrence and similarity distributions inside and outside the training data. Either method, when used alone, still offers a performance improvement over the baseline, but their combined effect is better. Using an RBM to reconstruct the training labels offers competitive, albeit slightly lower performance than the unified regularization approach. We think that all approaches that utilize internal label co-occurrence distributions will ultimately be limited by the training data itself. However, using a separate model like the RBM to pre-process label data in the pre-training stage holds an advantage in that it can be applied to other tasks apart from image annotation, since it is not couple with the target model itself. The disadvantage is that RBMs are not straightforward to train and the approach employing label reconstruction adds an extra stage to the whole algorithm.

We also presented two approaches in chapter 3, with the second one being an extension of the first. Both approaches are better than the baseline in quantitative and qualitative retrieval performance, with the second method being the best. The second method's major contributions are an improved triplet selection scheme, a modified loss function, and label reconstruction using a VAE. We observe that using an RN to aggregate convolutional features in order to increase the model's ability to tackle multi-object data did not yield substantial improvements than using a simple feed forward network. However, we speculate that a specialized model to separate instances of an object in an image is still necessary and with further architecture adjustments, an RN could still offer the required boost. In keeping with current trends, we included a quantization step in our model pipeline, to enable low storage and computation costs. Whereas other researchers report some performance variability with different numbers of bits used in hashing or quantization models, we found that our model is invariable to the number of bits for the most part. We reason that since the embedding and quantization stages of the architecture are separate, the quality of the quantized feature is directly dependent on the quality of the embedding. Above a certain number of bits, the retrieval performance of the

quantized vector will saturate, unless the embedding itself becomes better. This is a phenomenon that is more observable in quantization models rather than hashing models.

4.3 Limitations and Further Considerations

One limitation common to the approaches presented in both chapter 2 and 3 is the backbone network. To enable easy troubleshooting and reduce training time, we resorted to using the simple AlexNet CNN architecture, but it is quite possible that the architecture itself puts a limit on how much performance improvements we are able to get from our proposed contributions. At a certain point, the performance saturates because of network capacity regardless of improvements made in other parts of the algorithm. Using more recent, deeper architectures [125, 128, 4, 6, 129], we might see a bigger margin between the competing approaches and our method.

As mentioned above, as an additional solution to the missing label problem, we used an RBM to encode label dependencies and generate supplemental labels for the training data in chapter 2. However, it is important to note that recently, VAEs have begun to replace RBMs as generative models, since they are easier to train and in most circumstances offer better performance. Based on this, we made a change in chapter 2 and switched to a VAE as the model used to learn label dependencies in the training data. In future, it might be pertinent to switch out the RBM with a VAE in the method presented in Chapter 2 as well.

A drawback of both approaches proposed in Chapter 3 is that they use offline triplet mining to generate training examples. This means that fresh triplets have to be generated at the beginning of every epoch to account for the change in model parameters and structure of the embedding space. This slows down training noticeably. We tried moving the triplet generation online, by computing similarities batch-by-batch, but failed to achieve comparable performance. One reason for this could be because online triplet generation causes the network to learn on triplets containing a lot of repeated examples, hence reducing variability within a batch.

One pitfall of learning on multi-label data is that the missing label problem persists through to the evaluation stage. Though we notice a performance improvement in both multi-label annotation and multi-label retrieval after taking measures to deal with incomplete labeling, the quantitative score could still be negatively affected by missing labels in the test data. To reliably evaluate the ability of a method to learn a good model regardless of missing labels in training data, the test data itself needs to be labeled perfectly. This is a tedious and time-consuming process, especially for large-scale datasets. Therefore, in addition to quantitative evaluation, it is necessary to analyze the performance qualitatively, by inspecting several annotation or retrieval examples.

References

- [1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.
- [2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.
- [6] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2017.
- [7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Proc. International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.
- [8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *Proc. Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.
- [9] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

- [10] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-scnn: Gated shape cnns for semantic segmentation," *Proc. Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5229–5238, 2019.
- [11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.
- [12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," *Proc. European conference on computer vision*, pp. 21–37, Springer, 2016.
- [13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.
- [14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *Proc. Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.
- [15] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, "M2det: A single-shot object detector based on multi-level feature pyramid network," *Proc. Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 9259–9266, 2019.
- [16] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *Proc. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10781–10790, 2020.
- [17] Y. Zhao, R. Yang, G. Chevalier, X. Xu, and Z. Zhang, "Deep residual bidir- lstm for human activity recognition using wearable sensors," *Mathematical Problems in Engineering*, vol. 2018, pp. 1–13, 2018.
- [18] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," *Proc. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12046–12055, 2019.
- [19] N. Yudistira and T. Kurita, "Deep packet flow: Action recognition via multiresolution deep wavelet packet of local dense optical flows," *Journal of Signal Processing Systems*, vol. 91, no. 6, pp. 609–625, 2019.
- [20] M. S. Ryoo, A. Piergiovanni, J. Kangaspunta, and A. Angelova, "Assemblenet++: Assembling modality representations via attention connections," *Proc. European Conference on Computer Vision*, pp. 654–671, Springer, 2020.
- [21] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3929–3938, 2017.

- [22] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.
- [23] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron, "Unprocessing images for learned raw denoising," *Proc. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11036–11045, 2019.
- [24] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M. H. Yang, and L. Shao, "Learning enriched features for real image restoration and enhancement," *Proc. 16th European Conference on Computer Vision, ECCV 2020*, pp. 492–511, Springer Science and Business Media Deutschland GmbH, 2020.
- [25] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.
- [26] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., "Photo-realistic single image super-resolution using a generative adversarial network," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.
- [27] J. Mojoo, M. Sabri, and T. Kurita, "Video super resolution with estimation of motion information by using higher resolution images obtained by single image super resolution," *Proc. 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2019.
- [28] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," *Proc. European conference on computer vision*, pp. 241–257, Springer, 2016.
- [29] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," *Proc. Proceedings of the IEEE international conference on computer vision*, pp. 3456–3465, 2017.
- [30] B. Liu, Y. Cao, M. Long, J. Wang, and J. Wang, "Deep triplet quantization," *Proc. Proceedings of the 26th ACM international conference on Multimedia*, pp. 755–763, 2018.
- [31] B. Cao, A. Araujo, and J. Sim, "Unifying deep local and global features for image search," *Proc. European Conference on Computer Vision*, pp. 726–743, Springer, 2020.
- [32] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang, "Correlative multi-label video annotation," *Proc. 15th ACM International Conference on Multimedia*, pp. 17–26, 2007.
- [33] C. Wang, S. Yan, L. Zhang, and H.-J. Zhang, "Multi-label sparse coding for automatic image annotation," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1643–1650, IEEE, 2009.

- [34] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," Proc. European conference on computer vision, pp. 740–755, Springer, 2014.
- [35] K. Kang, W. Ouyang, H. Li, and X. Wang, "Object detection from video tubelets with convolutional neural networks," Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 817–825, IEEE Computer Society, 2016.
- [36] J. Shao, K. Kang, C. C. Loy, and X. Wang, "Deeply learned attributes for crowded scene understanding," Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4657–4666, IEEE Computer Society, 2015.
- [37] W.-S. Chu, F. De la Torre, and J. F. Cohn, "Learning spatial and temporal cues for multi-label facial action unit detection," Proc. 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), pp. 25–32, IEEE, 2017.
- [38] Z. Xue, S. Antani, L. R. Long, and G. R. Thoma, "Automatic multi-label annotation of abdominal ct images using cbir," Proc. Medical Imaging 2017: Imaging Informatics for Healthcare, Research, and Applications, vol. 10138, p. 1013807, International Society for Optics and Photonics, 2017.
- [39] Y. Zhang, R. Henao, Z. Gan, Y. Li, and L. Carin, "Multi-label learning from medical plain text with convolutional residual models," Proc. Machine Learning for Healthcare Conference, pp. 280–294, PMLR, 2018.
- [40] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," Proc. Data mining and knowledge discovery handbook, pp. 667–685, Springer, 2009.
- [41] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," Pattern recognition, vol. 37, no. 9, pp. 1757–1771, 2004.
- [42] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, and X. Geng, "Binary relevance for multi-label learning: an overview," Frontiers of Computer Science, pp. 1–12, 2018.
- [43] C.-K. Yeh, W.-C. Wu, W.-J. Ko, and Y.-C. F. Wang, "Learning deep latent spaces for multi-label classification," Proc. 31st AAAI Conference on Artificial Intelligence, pp. 2838–2844, 2017.
- [44] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe, "Deep convolutional ranking for multilabel image annotation," arXiv preprint arXiv:1312.4894, 2013.
- [45] J. Nam, J. Kim, E. L. Mencía, I. Gurevych, and J. Fürnkranz, "Large-scale multi-label text classification—revisiting neural networks," Proc. Joint european conference on machine learning and knowledge discovery in databases, pp. 437–452, Springer, 2014.
- [46] J. Deng, S. Satheesh, A. C. Berg, and L. Fei-Fei, "Fast and balanced: efficient label tree learning for large scale object recognition," Proc. 24th International Conference on Neural Information Processing Systems, pp. 567–575, 2011.

- [47] H. Hu, G.-T. Zhou, Z. Deng, Z. Liao, and G. Mori, "Learning structured inference neural networks with label relations," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2960–2968, 2016.
- [48] J. Mojoo, K. Kurosawa, and T. Kurita, "Deep cnn with graph laplacian regularization for multi-label image annotation," *Proc. International Conference Image Analysis and Recognition*, pp. 19–26, Springer, 2017.
- [49] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *Proc. International Conference on Machine Learning*, pp. 1060–1069, PMLR, 2016.
- [50] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
- [51] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2image: Conditional image generation from visual attributes," *Proc. European Conference on Computer Vision*, pp. 776–791, Springer, 2016.
- [52] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," *Proc. Artificial intelligence and statistics*, pp. 448–455, PMLR, 2009.
- [53] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio, "Learning algorithms for the classification restricted boltzmann machine," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 643–669, 2012.
- [54] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," *Proc. 24th international conference on Machine learning*, pp. 791–798, 2007.
- [55] J. Mojoo, Y. Zhao, M. S. Kavitha, J. Miyao, and T. Kurita, "Completion of missing labels for multi-label annotation by a unified graph laplacian regularization," *IEICE Transactions on Information and Systems*, vol. 103, no. 10, pp. 2154–2161, 2020.
- [56] J. Mojoo, Y. Zhao, M. Kavitha, J. Miyao, and T. Kurita, "Learning with incomplete labels for multi-label image annotation using cnn and restricted boltzmann machines," *Proc. International Conference on Neural Information Processing*, pp. 286–298, Springer, 2019.
- [57] Y. Zhu, J. T. Kwok, and Z.-H. Zhou, "Multi-label learning with global and local label correlation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1081–1094, 2017.
- [58] N. Ueda and K. Saito, "Parametric mixture models for multi-labeled text," *Proc. 15th International Conference on Neural Information Processing Systems*, pp. 737–744, 2002.
- [59] V. Ordonez, J. Deng, Y. Choi, A. C. Berg, and T. L. Berg, "From large scale image categorization to entry-level categories," *Proc. IEEE International Conference on Computer Vision*, pp. 2768–2775, 2013.

- [60] J. Johnson, L. Ballan, and L. Fei-Fei, "Love thy neighbors: Image annotation by exploiting image metadata," Proc. 2015 IEEE International Conference on Computer Vision, pp. 4624–4632, IEEE, 2015.
- [61] Y. Zhao, J. Miyao, and T. Kurita, "Multi-label image annotation via cnn with graph laplacian regularization based on word2vec," Proc. International Workshop on Frontiers of Computer Vision, 2018.
- [62] L. Xu, Z. Wang, Z. Shen, Y. Wang, and E. Chen, "Learning low-rank label correlations for multi-label classification with missing labels," Proc. 2014 IEEE International Conference on Data Mining, pp. 1067–1072, IEEE, 2014.
- [63] C.-W. Lee, W. Fang, C.-K. Yeh, and Y.-C. F. Wang, "Multi-label zero-shot learning with structured knowledge graphs," Proc. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1576–1585, IEEE, 2018.
- [64] B. Wu, S. Lyu, and B. Ghanem, "Ml-mg: Multi-label learning with missing labels using a mixed graph," Proc. 2015 IEEE International Conference on Computer Vision, pp. 4157–4165, IEEE, 2016.
- [65] Z.-J. Zha, T. Mei, J. Wang, Z. Wang, and X.-S. Hua, "Graph-based semi-supervised learning with multiple labels," Journal of Visual Communication and Image Representation, vol. 20, no. 2, pp. 97–103, 2009.
- [66] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," Proc. IEEE conference on computer vision and pattern recognition, pp. 2285–2294, 2016.
- [67] X. Li, F. Zhao, and Y. Guo, "Conditional restricted boltzmann machines for multi-label learning with incomplete labels," Journal of Machine Learning Research, vol. 38, pp. 635–643, 2015.
- [68] P. Liu, X. Qiu, J. Chen, and X.-J. Huang, "Deep fusion lstms for text semantic matching," Proc. 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1034–1043, 2016.
- [69] B. Wu, S. Lyu, and B. Ghanem, "Constrained submodular minimization for missing labels and class imbalance in multi-label learning," Proc. 30th AAAI Conference on Artificial Intelligence, pp. 2229–2236, 2016.
- [70] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon, "Large-scale multi-label learning with missing labels," Proc. International Conference on Machine Learning, pp. 593–601, PMLR, 2014.
- [71] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," Proc. 26th International Conference on Neural Information Processing Systems-Volume 2, pp. 3111–3119, 2013.
- [72] C. Hayashi, "Quantification method iii or correspondence analysis in medical science," Annals of Cancer Research and Therapy, vol. 1, no. 1, pp. 17–21, 1992.

- [73] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering." *Proc. Nips*, vol. 14, pp. 585–591, 2001.
- [74] K. Watanabe and T. Kurita, "Locality preserving multi-nominal logistic regression," *Proc. 19th International Conference on Pattern Recognition*, pp. 1–4, IEEE, 2008.
- [75] A. Fischer and C. Igel, "An introduction to restricted boltzmann machines," *Proc. Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pp. 14–36, Springer, 2012.
- [76] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.
- [77] J. Huang, G. Li, S. Wang, Z. Xue, and Q. Huang, "Multi-label classification by exploiting local positive and negative pairwise label correlation," *Neurocomputing*, vol. 257, pp. 164–174, 2017.
- [78] T. Kato, T. Kurita, and H. Shimogaki, "Multimedia interaction with image database systems," *ACM SIGCHI Bulletin*, vol. 22, no. 1, pp. 52–54, 1990.
- [79] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra, "Relevance feedback: a power tool for interactive content-based image retrieval," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 5, pp. 644–655, 1998.
- [80] T. Takahashi and T. Kurita, "Mixture of subspaces image representation and compact coding for large-scale image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1469–1479, 2015.
- [81] A. K. Alhassan and A. A. Alfaki, "Color and texture fusion-based method for content-based image retrieval," *Proc. 2017 International Conference on Communication, Control, Computing and Electronics Engineering*, pp. 1–6, IEEE, 2017.
- [82] P. P. Mane and N. G. Bawane, "An effective technique for the content based image retrieval to reduce the semantic gap based on an optimal classifier technique," *Pattern Recognition and Image Analysis*, vol. 26, no. 3, pp. 597–607, 2016.
- [83] A. Alzu'bi, A. Amira, and N. Ramzan, "Semantic content-based image retrieval: A comprehensive study," *Journal of Visual Communication and Image Representation*, vol. 32, no. Supplement C, pp. 20–54, 2015.
- [84] T. Kurita and T. Kato, "Learning of personal visual impression for image database systems," *Proc. 2nd International Conference on Document Analysis and Recognition*, pp. 547–552, 1993.
- [85] X. Li, T. Uricchio, L. Ballan, M. Bertini, C. G. M. Snoek, and A. D. Bimbo, "Socializing the semantic gap: A comparative survey on image tag assignment, refinement, and retrieval," *ACM Comput. Surv.*, vol. 49, no. 1, pp. 14:1–14:39, 2016.

- [86] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," arXiv preprint arXiv:1301.3557, 2013.
- [87] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," arXiv preprint arXiv:1312.6229, 2013.
- [88] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," Proc. IEEE conference on computer vision and pattern recognition workshops, pp. 806–813, 2014.
- [89] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," Proc. International Workshop on Similarity-Based Pattern Recognition, pp. 84–92, Springer, 2015.
- [90] J. Vogel and B. Schiele, "Semantic modeling of natural scenes for content-based image retrieval," International Journal of Computer Vision, vol. 72, no. 2, pp. 133–157, 2007.
- [91] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F. fei Li, "Image retrieval using scene graphs," Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3668–3678, 2015.
- [92] J. MOJOO and T. KURITA, "Deep metric learning for multi-label and multi-object image retrieval," IEICE Transactions on Information and Systems, vol. 104, no. 6, pp. 873–880, 2021.
- [93] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "Matchnet: Unifying feature and metric learning for patch-based matching," Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3279–3286, 2015.
- [94] J. Masci, D. Migliore, M. M. Bronstein, and J. Schmidhuber, "Descriptor learning for omnidirectional image matching," Proc. Registration and Recognition in Images and Videos, pp. 49–62, Springer, 2014.
- [95] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4353–4361, 2015.
- [96] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1386–1393, 2014.
- [97] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," Proc. IEEE conference on computer vision and pattern recognition, pp. 815–823, 2015.
- [98] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 3109–3118, 2015.

- [99] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," Proc. IEEE International Conference on Computer Vision, pp. 2593–2601, 2017.
- [100] S. Zhang, Y. Gong, and J. Wang, "Deep metric learning with improved triplet loss for face clustering in videos," Proc. Pacific Rim Conference on Multimedia, pp. 497–508, Springer, 2016.
- [101] A. Hermans, L. Beyer, and B. Leibe, "In Defense of the Triplet Loss for Person Re-Identification," arXiv preprint arXiv:1703.07737, 2017.
- [102] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large scale online learning of image similarity through ranking," Journal of Machine Learning Research, vol. 11, pp. 1109–1135, 2010.
- [103] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning," Proc. 28th AAAI Conference on Artificial Intelligence, pp. 2156–2162, AAAI Press, 2014.
- [104] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," Proc. IEEE conference on computer vision and pattern recognition, pp. 3270–3278, 2015.
- [105] Y. Cao, M. Long, J. Wang, H. Zhu, and Q. Wen, "Deep quantization network for efficient image retrieval," Proc. 13th AAAI Conference on Artificial Intelligence, 2016.
- [106] Y. Cao, M. Long, J. Wang, and S. Liu, "Deep visual-semantic quantization for efficient image retrieval," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1328–1337, 2017.
- [107] F. Zhao, Y. Huang, L. Wang, and T. Tan, "Deep semantic ranking based hashing for multi-label image retrieval," Proc. IEEE conference on computer vision and pattern recognition, pp. 1556–1564, 2015.
- [108] H. Lai, P. Yan, X. Shu, Y. Wei, and S. Yan, "Instance-aware hashing for multi-label image retrieval," IEEE Transactions on Image Processing, vol. 25, no. 6, pp. 2469–2479, 2016.
- [109] Z. Zhang, Q. Zou, Y. Lin, L. Chen, and S. Wang, "Improved deep hashing with soft pairwise similarity for multi-label image retrieval," IEEE Transactions on Multimedia, vol. 22, no. 2, pp. 540–553, 2020.
- [110] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," Proc. Conference on neural information processing systems, pp. 4967–4976, 2017.
- [111] N. Messina, G. Amato, F. Carrara, F. Falchi, and C. Gennaro, "Learning relationship-aware visual features," Proc. European Conference on Computer Vision, pp. 486–501, 2018.



- [112] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.
- [113] S. Ding, L. Lin, G. Wang, and H. Chao, "Deep feature learning with relative distance comparison for person re-identification," *Pattern Recognition*, vol. 48, no. 10, pp. 2993–3003, 2015.
- [114] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.
- [115] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh, "No fuss distance metric learning using proxies," *Proc. Proceedings of the IEEE International Conference on Computer Vision*, pp. 360–368, 2017.
- [116] H. Xuan, A. Stylianou, and R. Pless, "Improved embeddings with easy positive triplet mining," *Proc. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2474–2482, 2020.
- [117] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4004–4012, 2016.
- [118] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1386–1393, 2014.
- [119] B. Amos, B. Ludwiczuk, M. Satyanarayanan, et al., "Openface: A general-purpose face recognition library with mobile applications," *CMU School of Computer Science*, vol. 6, no. 2, 2016.
- [120] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation*, p. 318–362, MIT Press, Cambridge, MA, USA, 1986.
- [121] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," *Proc. Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, 2008.
- [122] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *arXiv preprint arXiv:1703.00395*, 2017.
- [123] M. Ranzato, C. Poultney, S. Chopra, Y. LeCun, et al., "Efficient learning of sparse representations with an energy-based model," *Advances in neural information processing systems*, vol. 19, p. 1137, 2007.
- [124] A. Makhzani and B. Frey, "K-sparse autoencoders," *arXiv preprint arXiv:1312.5663*, 2013.
- [125] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.


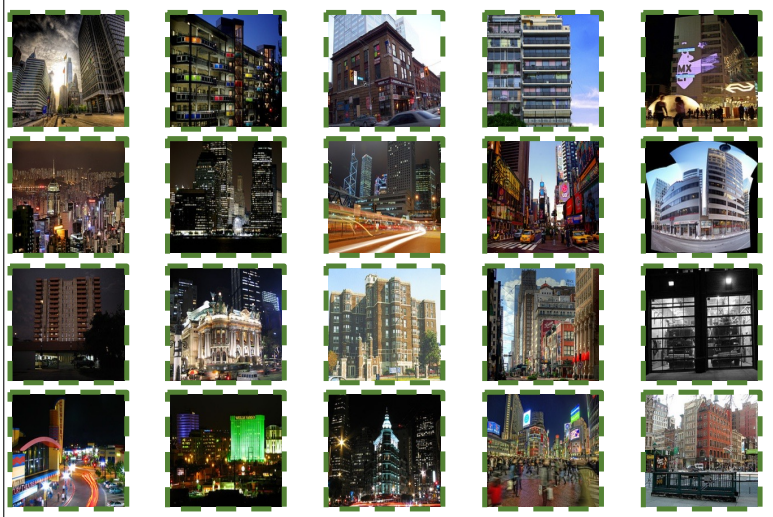

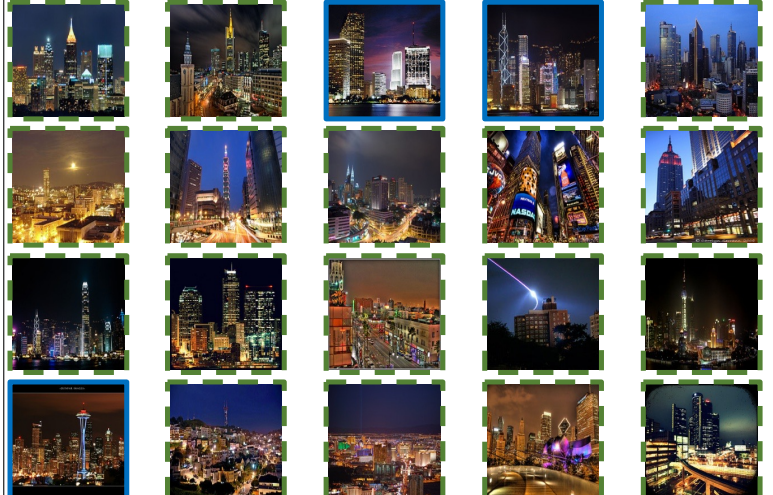
-
- [126] P. Jaccard, "The distribution of the flora in the alpine zone. 1," *New phytologist*, vol. 11, no. 2, pp. 37–50, 1912.
- [127] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "Nus-wide: a real-world web image database from national university of singapore," *Proc. ACM international conference on image and video retrieval*, pp. 1–9, 2009.
- [128] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.
- [129] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Proc. Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.


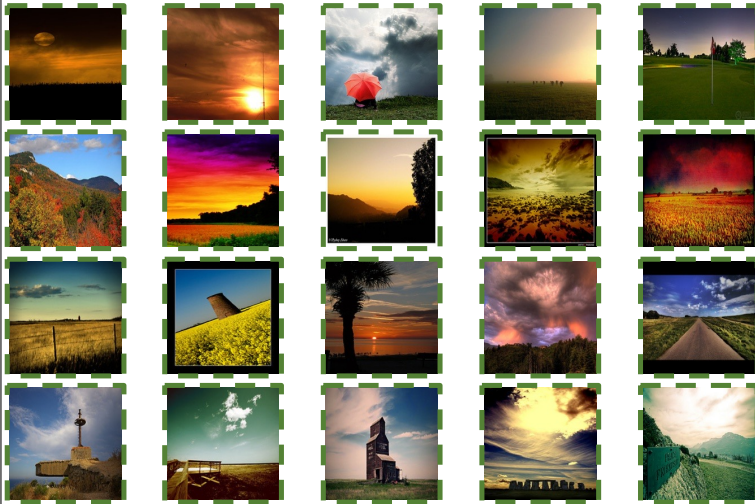
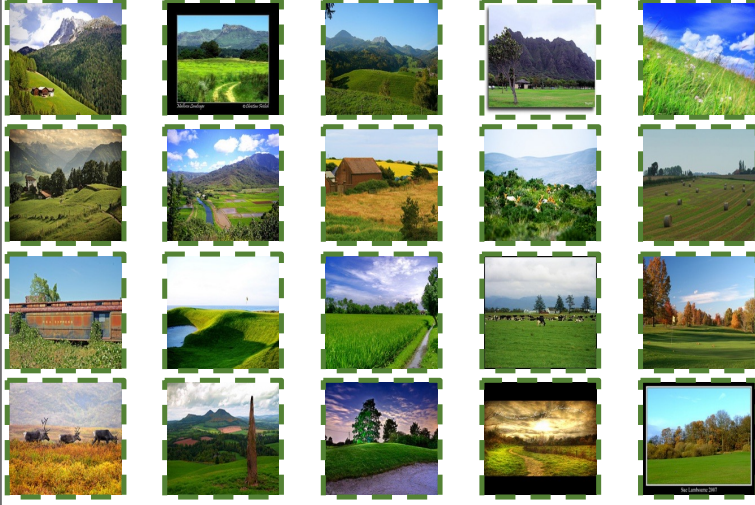
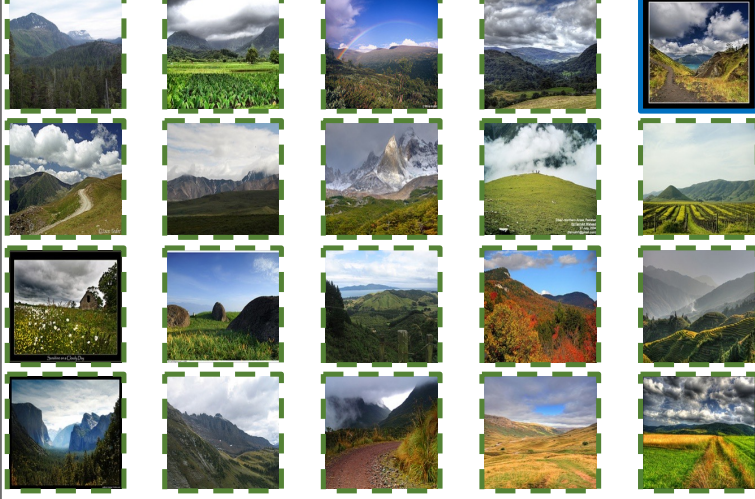
Appendix A


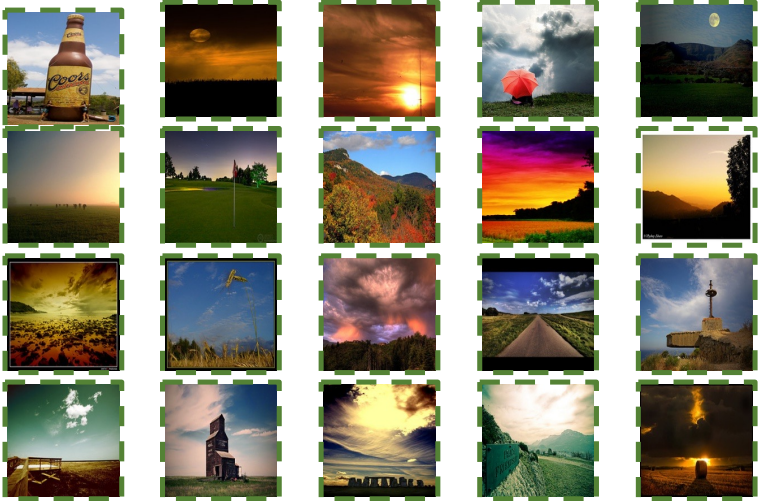

Retrieval Examples

This appendix illustrates qualitatively the retrieval performance of our methods in chapter 3, compared with the closest competing method. The labels given in the dataset for each query are indicated next to it. We added relevant labels where applicable to provide more context and demonstrate the models' performance assuming that all labels are available.

Query						plants, animal, flowers
						<p>— partially similar</p> <p>— fully similar</p>
Method	Top 20 retrieved images					
DTQ [30]						
MLMO1						
MLMO2						

Query	 <p data-bbox="813 336 1283 414">buildings, tower, nighttime, sky, clouds[, water, reflection]</p> <p data-bbox="893 425 1189 504"> — partially similar — fully similar </p>				
Method	Top 20 retrieved images				
DTQ [30]					
MLMO1					
MLMO2					

Query						<p>sky, valley, rocks, sky, water, clouds, mountain, grass</p>
	<p>— partially similar — fully similar</p>					
Method	Top 20 retrieved images					
DTQ [30]						
MLMO1						
MLMO2						

<p>Query</p>	 <p>flowers, sky[, plants]</p> <p> — partially similar — fully similar </p>				
<p>Method</p>	<p>Top 20 retrieved images</p>				
<p>DTQ [30]</p>					
<p>MLMO1</p>					
<p>MLMO2</p>	