Deep Binary Representation Learning for Single/Cross-Modal Data Retrieval

Yuming Shen

School of Computing Sciences

University of East Anglia

A thesis submitted for the degree of

Doctor of Philosophy

January 2018

Declaration

Parts of this thesis have been taken from certain published/to be published conference/journal papers. All of these papers were written primarily or partially by me, Yuming Shen, during and as a result of my PhD research. These papers are listed below.

- **Y. Shen**, L. Liu, L. Shao and J. Song. Deep Binaries: Encoding Semantic-Rich Cues for Efficient Textual-Visual Cross Retrieval. In *IEEE International Conference on Computer Vision (ICCV)*, 2017
- **Y. Shen**, L. Liu and L. Shao. Unsupervised Deep Generative Hashing. In *British Machine Vision Conference (BMVC)*, 2017
- L. Liu, F. Shen, **Y. Shen**, X. Liu and L. Shao. Deep Sketch Hashing: Fast Free-Hand Sketch-Based Image Retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017
- **Y. Shen**, L. Zhang and L. Shao. Semi-Supervised Vision-Language Mapping via Variational Learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017

Acknowledgements

I would like to extend my sincerest thanks to the following, who have all helped in the completion of this thesis.

First of all, many thanks to my family for their unconditional support during my PhD studies.

I would like to express my sincere appreciation to my supervisor Prof. Ling Shao for providing me with the opportunity of pursuing this PhD degree. We are allowed to have unprecedented research freedom under his supervision, and he is always willing to offer valuable advice and thorough suggestions on our research works.

I would like to thank Prof. Andy Day for his academic support.

I would like to thank the current and former research students, Postdoc researchers and visiting scholars in our lab, including Dr. Li Liu, Dr. Yang Long, Dr. Mengyang Yu, Dr. Heng Liu, Dr. Haofeng Zhang, Dr. Xiaoming Liu, Yi Zhou, Shidong Wang, Bingzhang Hu and Daniel Organisciak, for their helpful discussion and research ideas.

I would also like to thank my co-authors, Prof. Fumin Shen and Prof. Xianglong Liu, who have improved my research outcome from many aspects.

Abstract

Data similarity search is widely regarded as a classic topic in the realms of computer vision, machine learning and data mining. Providing a certain query, the retrieval model sorts out the related candidates in the database according to their similarities, where representation learning methods and nearest-neighbour search apply. As matching data features in Hamming space is computationally cheaper than in Euclidean space, learning to hash and binary representations are generally appreciated in modern retrieval models. Recent research seeks solutions in deep learning to formulate the hash functions, showing great potential in retrieval performance. In this thesis, we gradually extend our research topics and contributions from unsupervised single-modal deep hashing to supervised cross-modal hashing finally *zero-shot* hashing problems, addressing the following challenges in deep hashing.

First of all, existing unsupervised deep hashing works are still not attaining leading retrieval performance compared with the shallow ones. To improve this, a novel unsupervised single-modal hashing model is proposed in this thesis, named Deep Variational Binaries (DVB). We introduce the popular conditional variational auto-encoders to formulate the encoding function. By minimizing the reconstruction error of the latent variables, the proposed model produces compact binary codes without training supervision. Experiments on benchmarked datasets show that our model outperform existing unsupervised hashing methods.

The second problem is that current cross-modal hashing methods only consider holistic image representations and fail to model descriptive sentences, which is inappropriate to handle the rich semantics of informative cross-modal data for quality textual-visual search tasks. To handle this problem, we propose a supervised deep cross-modal hashing model called Textual-Visual Deep Binaries (TVDB). Region-based neural networks and recurrent neural networks are involved in the image encoding network in order to make effective use of visual information, while the text encoder is built using a convolutional neural network. We additionally introduce an efficient in-batch optimization routine to train the network parameters. The proposed mode successfully outperforms state-of-the-art methods on large-scale datasets.

Finally, existing hashing models fail when the categories of query data have never been seen during training. This scenario is further extended into a novel *zero-shot* cross-modal hashing task in this thesis, and a *Zero-shot* Sketch-Image Hashing (ZSIH) scheme is then proposed with graph convolution and stochastic neurons. Experiments show that the proposed ZSIH model significantly outperforms existing hashing algorithms in the *zero-shot* retrieval task.

Experiments suggest our proposed and novel hashing methods outperform state-of-the-art researches in single-modal and cross-modal data retrieval.

Contents

List of Figures

List of Tables

Glossary

AGH Anchor Graph Hashing

BRE Binary Reconstructive Embedding

CCA Canonical Correlation Analysis

CDQ Collective Deep Quantization

CMFH Collective Matrix Factorization Hashing

CM-NN Cross-Modal Neural Network

CMSSH Cross-Modal Similarity Sensitive Hashing

CMT Cross-Modal Transfer

CNN Convolutional Neural Network

CorrAE Correspondence Auto-Encoder

CVAE Conditional Variational Auto-Encoder

CVFL Cross-View Feature Learning

CVH Cross-View Hashing

DBM Deep Boltzmann Machine

DBN Deep Belief Network

DCDH Discriminative Coupled Dictionary Hashing

DCMH Deep Cross-Modal Hashing

DeViSE Deep Visual-Semantic Embedding

DGH Discrete Graph Hashing

DH Deep Hashing

DNH Deep Neural Hashing

DSH Deep Sketch Hashing

DVB Deep Variational Binaries

DVSH Deep Visual Semantic Hashing

GCN Graph Convolutional Network

GPU Graphics Processing Unit

HD2 Hamming Distance with radius 2

IAF Inverse Autoregressive Flow

IMH Inter-Media Hashing

IMVH Iterative Multi-View Hashing

ITQ Iterative Quantization

JLSE Joint Latent Similarity Embedding

K-L divergence Kullback-Leibler divergence

KLSH Kernelized Locality Sensitive Hashing

KNN K-Nearest-Neighbour

LCMH Linear Cross-Modal Hashing

LSH Locality Sensitive Hashing

LSPH Latent Structure Preserving Hashing

LSTM Long Short-Term Memory

mAP mean-Average Precision

MFB Multi-modal Factorized Bilinear pooling

MSAE Multi-modal Stacked Auto-Encoder

PCA Principal Component Analysis

PLSR Partial Least Square Regression

P-R curve Precision-Recall curve

Precision@100 Precision at top-100 retrieved candidates

Precision@200 Precision at top-200 retrieved candidates

Precision@5000 Precision at top-5000 retrieved candidates

QCH Quantized Correlation Hashing

RNN Recurrent Neural Network

RPN Region Proposal Network

RSH Ranking-based Supervised Hashing

SAE Semantic Auto-Encoder

SaN Sketch-a-Net

SBIR Sketch-Based Image Retrieval

SCM Semantic Correlation Maximization

SDH Supervised Discrete Hashing

SELVE Sparse Embedding and Least Variance Encoding

SePH Semantics-Preserving Hashing

SGD Stochastic Gradient Descent

SH Spectral Hashing

SitNet Similarity-transfer Network

SKLSH Shift-invariant Kernelized Locality Sensitive hashing

SM^{2}H Sparse Multi-Modal Hashing

SpH Spherical Hashing

SSE Semantic Similarity Embedding

SUBIC SUpervised structured BInary Code

SVM Support Vector Machine

t-SNE t-distributed Stochastic Neighbour Embedding

TVDB Textual-Visual Deep Binaries

UN-BDNN Unsupervised Binary Deep Neural Network

VAE Variational Auto-Encoder

ZSH Zero-Shot Hashing

ZSIH Zero-shot Sketch-Image Hashing

Chapter 1

Introduction

1.1 Research Background

Data similarity retrieval plays a role of importance in modern artificial intelligence systems. Providing a certain query, the retrieval model sorts out the related candidates in the database according to their similarities, where representation learning methods and approximate nearest-neighbour search apply. Embedding high-dimensional data to low dimensional binary codes, hashing algorithms arouse wide research attention in computer vision, machine learning and data mining. Considering the low computational cost of approximate nearest-neighbour search in the Hamming space [173, 111], hashing techniques deliver more effective and efficient large-scale data retrieval than real-valued embeddings, which is more appreciated and applicable in real cases.

As a classic but essential research topic, a number of works have been proposed throughout the years, employing a wide range of machine learning techniques, which can be either supervised or unsupervised. These methods typically involves an embedding function $f : \mathbb{R}^D \to \mathbb{R}^M$, where D and M respectively indicates the original and targeted representation dimensionality. Therefore, given a single data point $\mathbf{x} \in \mathbb{R}^D$, the resulting hash code can be obtained by applying an element-wise sign function to the encoder $f(\cdot)$.[1] By tuning the corresponding parameters of the embedding function, the hash model approaches the optimal representation that numerically describes the original data $\mathbf{x}$. A typical example of $f(\cdot)$ is the

[1]The hashed code $\mathbf{c}$ is computed by $\mathbf{c} = \mathtt{sign}(f(\mathbf{x};\theta)) \in \{0,1\}^M$, where $\mathtt{sign}(\cdot)$ is the element-wise sign function and θ refers to the parameters to learn.

linear projection function.[2] The encoder can also be based on other models, *e.g.* random projections, and a literature review on the way to build the hash functions is included following this chapter.

Providing a query, the retrieval model performs nearest-neighbour search [3] to find the most related candidate in the data collection as retrieval result. In a real scenario, finding the exact nearest neighbour as the retrieval candidate is usually infeasible as, in large-scale datasets, there can be more than one correct or relevant candidate to the query. Therefore, a conventional generalization of nearest-neighbour search yields K-Nearest-Neighbour (KNN) search, where a total number of K closest candidates are retrieved.

Existing researches in data hashing apply to several tasks. For instance, single-modal hashing learns the binary representations of standalone type of data, while cross-modal hashing tackles the problem of retrieving data between different modalities of data, *e.g.* images and text documents. Recently, deep learning technologies have been introduced in formulating the embedding function $f(\cdot)$, and produce promising encoding and retrieval performance, forming the term of deep hashing.

1.2 Objective

Dramatic progress has been achieved in the past few years, but challenges still remain in deep hashing from several aspects. The objective of this thesis is to solve the following problems by proposing novel hashing models:

1. Although deep learning has been proven to be successful in supervised hashing, existing deep learning based unsupervised hashing techniques still cannot produce leading performance compared with the non-deep methods, as it is hard to unveil the intrinsic structure of the whole sample space in the framework of mini-batch optimization.

[2]A typical linear projection model can be written as $\mathbf{c} = \mathbf{x}\mathbf{W}_\theta + \mathbf{b}_\theta$. Here $\mathbf{W}_\theta$ and $\mathbf{b}_\theta$ refer to the linear transformation parameters.

[3]Given a query $\mathbf{q}$ and the data collection $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_N\}$, nearest-neighbour search is defined by $NearestNeighbour(\mathbf{q}) = \text{argmin}_{\mathbf{x}\in\mathcal{X}}\ d\left(\texttt{sign}(f(\mathbf{q})), \texttt{sign}(f(\mathbf{x}))\right)$, where $d(\cdot)$ denotes the hamming distance computation.

2. Most of the traditional cross-modal binary encoding methods for textual-visual retrieval only consider holistic image representations and fail to model descriptive sentences. This renders existing methods inappropriate to handle the rich semantics of informative cross-modal data for quality textual-visual search tasks.

3. Providing training and test data subjected to a fixed set of pre-defined categories, the cutting-edge cross-modal hashing works obtain acceptable retrieval performance. However, most of the existing methods fail when the categories of query sketches have never been seen during training.

1.3 Contribution

In this thesis, We gradually rise the task difficulty throughout the chapters by extending our research topics from unsupervised single-modal deep hashing to supervised cross-modal hashing finally *zero-shot* cross-modal hashing. The contributions of this thesis include handling the above-mentioned challenges and, meanwhile, proposing novel deep hashing models with state-of-the-art performance in image retrieval and cross-modal data search, which can be summarized as follows:

- We propose a novel unsupervised image hashing algorithm that differs from existing deep-learning-based methods, extending the popular variational auto-encoders [76, 160] to tackle problem 1.

- Problem 2 is improved by designing novel neural network structures as encoders. The rich information carried by images and sentences is then efficiently encoded in a newly-proposed cross-modal hashing model.

- We finally define a novel *zero-shot* cross-modal retrieval task and accordingly design a deep hashing algorithm for problem 3. The proposed model manages to provide accurate retrieval candidates under the *zero-shot* setting, outperforming existing hashing works.

In the next section, how the contributions above are distributed into the chapters is elaborated.

1.4 Summary of Remaining Chapters

The rest of this thesis consists of three state-of-the-art models in learning deep binary representations addressing the aforementioned problems 1, 2 and 3. Gradually raising the task difficulty, we firstly present a deep unsupervised hashing model for single-modal data retrieval and then introduce a cross-modal deep hashing network. Finally, we extend our research topic to a novel *zero-shot* cross-modal retrieval scenario. The remaining chapters are summarized as follows:

Chapter 2: Literature Review. An overview of the state-of-the-art binary representation learning algorithms is given in this chapter, including both the traditional shallow models and the deep-learning-based ones.

Chapter 3: Unsupervised Deep Hashing for Image Retrieval. In this chapter, the above-mentioned problem 1 is considered. We propose a novel unsupervised deep hashing model, named Deep Variational Binaries (DVB). The conditional auto-encoding variational Bayesian networks are introduced in this work as the generative model to exploit the feature space structure of the training data using the latent variables. Integrating the probabilistic inference process with hashing objectives, the proposed DVB model estimates the statistics of data representations, and thus produces compact binary codes. Experimental results on three benchmark datasets, i.e. CIFAR-10, SUN-397 and NUS-WIDE, demonstrate that DVB outperforms state-of-the-art unsupervised hashing methods with significant margins.

Chapter 4: Supervised Deep Hashing for Image-Sentence Retrieval. We move from single-modal hashing to cross-modal hashing in this chapter, handling problem 2. Considering the factor that cross-modal data may contain semantic-rich cues, we develop a novel integrated deep architecture to effectively encode the detailed semantics of informative images and long descriptive sentences, named as Textual-Visual Deep Binaries (TVDB). In particular, region-based convolutional networks with long short-term memory units are introduced to fully explore image regional details while semantic cues of sentences are modeled by a text convolutional network. Additionally, we propose a stochastic batch-wise training routine, where high-quality binary codes and deep encoding functions are efficiently optimized in an alternating manner. Experiments are conducted on three multimedia datasets, i.e. Microsoft COCO, IAPR TC-12, and INRIA Web Queries, where the proposed TVDB model significantly outperforms state-of-the-art binary coding methods in the task of cross-modal retrieval.

Chapter 5: Deep Hashing for Zero-Shot Image-Sketch Retrieval. In this chapter, problem 3 is briefed as a novel but realistic *zero-shot* hashing task specified in category-level Sketch-Based Image Retrieval (SBIR). We elaborate the challenges of this special task and accordingly propose a zero-shot sketch-image hashing (ZSIH) model. An end-to-end three-network architecture is built, two of which are treated as the binary encoders. The third network mitigates the sketch-image heterogeneity and enhances the semantic relations among data by utilizing the Kronecker fusion layer and graph convolution, respectively. As an important part of ZSIH, we formulate a generative hashing scheme in reconstructing semantic knowledge representations for zero-shot retrieval. To the best of our knowledge, ZSIH is the first zero-shot hashing work suitable for SBIR and cross-modal search. Comprehensive experiments are conducted on two extended datasets, i.e. Sketchy and TU-Berlin with a novel zero-shot train-test split. The proposed model remarkably outperforms related works.

Chapter 6: Conclusion and Future Work. A brief summery of the contributions of this thesis is given in this chapter, followed by an outlook of future research interests.

Chapter 2

Literature Review

In this chapter, the articles and works related to this thesis are discussed. We firstly introduce the single-modal hashing methods in Section 2.1, which are associated with problem 1 mentioned in Chapter 1. Problem 2 and 3 are related to cross-modal hashing and retrieval tasks. Therefore, a review on existing cross-modal hashing methods is also provided in Section 2.2. Since this thesis focuses on learning deep binary representations, the state-of-the-at deep hashing models are also discussed in Section 2.3.

2.1 Single-Modal Hashing

Single-modal hashing learns the encoding function of a single modality. In general, it can be either supervised or unsupervised. As problem 1 and Chapter 3 are related to unsupervised hashing, in this section, we mainly focus on the unsupervised hashing algorithms and then have a quick review on the supervised ones.

Supervised hashing [155, 44, 86, 172, 132, 108, 106] utilizes data labels or pairwise similarities as supervision during parameter optimization. It attains relatively better retrieval performance than the unsupervised models as the conventional evaluation measurements of data retrieval are highly related to the labels. However, due to the cost of manual annotation and tagging, supervised hashing is not always appreciated and demanded. On the other hand, unsupervised hashing [43, 151, 178, 156, 82, 51, 111] learns the binary encoding function based on data representations and require no label information, which eases the task of data retrieval where human annotations are not available.

2.1.1 Unsupervised Hashing

Existing research interests on unsupervised hashing involve various strategies to formulate and optimize the encoding functions.

Iterative Quantization (ITQ) [43] aims at minimizing quantization error to produce binary representations. The computation of ITQ [43] consists of two stages. In the first stage, the model reduces data dimension using Principal Component Analysis (PCA) [70]. The second step is to find the target hash codes and an orthogonal rotation matrix that minimizes the difference between target codes and rotated data. The problem of ITQ [43] is that its optimization procedure only focuses on the second step above and does not cover the dimension reduction step. This means that though an optimal rotation matrix can be found, there is no guarantee that the produced codes best represent the original data.

Weiss *et al.* [178] propose Spectral Hashing (SH) to learn the hash function using the eigenfunctions. SH [178] aims at minimizing the weighted pairwise Euclidean distance where weights are determined by data similarities. In addition, the code balance condition is introduced during training, which means the number of data assigned to each code remains the same. Hash codes are obtained by thresholding the Laplacian eigenfunctions with the smallest eigenvalues at zero. The disadvantage of SH [178] comes from a strong assumption that data representations follow a uniform distribution, which does not always hold in a realistic scenario. As a result, it is not suitable for encoding and retrieving informative data such as images and texts.

Known as a typical random projection method for unsupervised hashing, Locality Sensitive Hashing (LSH) [21] successively applies thresholding mechanisms on projected data to producing binary codes, which has been further improved by Kernelized Locality Sensitive Hashing (KLSH) [87]. The aim of KLSH [87] is to build the locality sensitive hash functions with the pair-wise representation angle determined in the kernel space. Then the projection vector can be constructed from a Gaussian distribution based on the training data statistics. Random-projection-based encoding methods are not ideal for realistic applications since their performances drop dramatically with the decrease of code length. For example, KLSH [87] performs poor when the code length is less than 64, and requires a code length around 400 to obtain best retrieval performance. This is unappreciated as the state-of-the-art hashing algorithms generally produce reasonable retrieval performance with short code length, *e.g.* 16 or 32 bits.

Learning-based projection functions are also widely adopted in unsupervised hashing. Heoet *et al.* propose Spherical Hashing (SpH) [54]. Its hash function is composed of a set of spherical functions and pivots. Hashed codes are determined by thresholding the distance between the data representation and the pivots. SpH [54] does not directly penalize the similarity disagreement between the code space and data representation space, and thus is suboptimal for similarity search. Furthermore, the iterative optimization procedure in learning the pivots is time consuming.

Anchor Graph Hashing (AGH) [115] proposed by Liu *et al.* initially employs anchor graphs for hashing. An anchor graph represents the similarities between data pairs in the dataset using a small number of data anchors. In this way, the model is able to discover the neighborhood structure inherent in the training data in an efficient way. This design is further extended to Discrete Graph Hashing (DGH) [113] later on. Instead of thresholding the continuous variables for hashing in [115], DGH [113] directly produces discrete solutions for graph hashing. AGH [115] and DGH [113] successfully explore the data structure and therefore produce compact hash codes. However, their performance largely relies on the quality and quantity of selected anchor points. Bad choices of anchor points skews the data similarities, and thus results in less representative hash code. Finding the anchors that best suit each dataset is computationally expensive and sometimes unrealistic.

Mathematically profound as the above works are, their performance on similarity retrieval is still far from satisfying. Despite the disadvantages of these works discussed above, a key problem is that the shallow encoding functions, *e.g.* linear projections, in these works are not capable to handle complex data representations, and therefore the generated codes are suspected to be less informative. In this thesis, we utilize deep neural networks to tackle this problem.

2.1.2 Supervised Hashing

Supervised hashing generally attains better retrieval performance than unsupervised hashing on most open-sourced datasets.

One classic paradigm of supervised hashing is Binary Reconstructive Embedding (BRE) [86]. Kulis and Darrell [86] aim at minimizing the difference between the original representation distance in the Euclidean space and the reconstructed distance in the Hamming space. Hash codes are obtained using a kernel-based hash function with a projection matrix to learn. BRE [86] is hard to train as the learning objective is non-convex and it is infeasible to accelerate the optimization procedure by parallel computation like deep learning. In addition, its retrieval performance is still far from satisfactory.

Liu *et al.* [114] propose a kernel-based hashing scenario, namely Kernel-based Supervised Hashing (KSH). By utilizing pair-wise data similarities according to the category information, a learning objective is formulated where the inner-product of binary codes reflects Hamming distance. The hash function is then sequentially trained with low computational complexity. KSH [114] does not consider intra-class encoding distance during training, and thus fail to produce high-quality binary codes on datasets with large number of categories.

Kulis *et al.* [88] represent pairwise data relations using angle-based feature similarities. Then a positive-valued metric is learned, approximating the angle relations between data. The procedure of generating the angle similarities requires single-label data, and it is infeasible to compute the similarities between data with more than one categories according to [88]. As a result, this design is not working on multi-label datasets.

Recently, Shen *et al.* [155] propose Supervised Discrete Hashing (SDH) optimal for linear classification. The discrete constraints of a trace-like hashing objective is relaxed to continuous space and the targeted binary codes are treated as auxiliary variables employed in an alternating optimization procedure. SDH produces acceptable retrieval performance as the learned codes are usually conceptually distinguishable. The disadvantage of SDH [155] is that it requires training a Support Vector Machine (SVM) [24] to generate auxiliary binary variables for each training iteration, which is inefficient for large-scale datasets.

In addition to the label-based and pair-wise-based hashing, several works leverage stronger supervision to train the models, *e.g.* triplet similarities and ranking lists. Triplet-based hashing [98, 171] requires a set of positive-negative data pairs[1]. In [98], a column generation scheme is formulated as a convex problem suitable for large-margin learning. Codes are generated iteratively by selecting the best hash function at each iteration. On the other hand, Ranking-based Supervised Hashing (RSH) [174] maximizes the ranking quality of the encoded training samples, where the learning objective optimizing linear transformation parameters is solved by employing augmented Lagrangian multiplier method. The problem of ranking-based and triplet-based hashing techniques lie in labeling and organizing data relations. Such kind of strong supervision is expensive to obtain, requiring considerable manpower and time in model optimization on large-scale datasets. Therefore, these methods are not always appreciated.

2.2 Cross-Modal Hashing

Different from the works discussed in Section 2.1, cross-modal hashing handles the problem of large-scale data retrieval between different modalities, where more than one encoding function is formulated and trained for each modality. In particular, image-text retrieval has become a classic usage scenario. Traditional studies in cross-modal hashing also employ supervised and unsupervised learning schemes, though the unsupervised one generally results in inferior encoding quality.

2.2.1 Unsupervised Cross-Modal Hashing

Typical unsupervised hashing methods include Collective Matrix Factorization Hashing (CMFH) [28], Cross-View Hashing (CVH) [90], Inter-Media Hashing

[1]This dataset is typically denoted as $\mathcal{O} = \{(\mathbf{x}_i, \mathbf{x}_i^+, \mathbf{x}_i^-)\}_i$. Here $Similarity(\mathbf{x}_i, \mathbf{x}_i^+) > Similarity(\mathbf{x}_i, \mathbf{x}_i^-)$.

(IMH) [163], Linear Cross-Modal Hashing (LCMH) [215], Multi-modal Stacked Auto-Encoder (MSAE) [176], *etc.*

CVH [90] can be regarded as an extension of SH [178] on the cross-modal application, where hash codes are obtained by thresholding the eigenfunctions of the heterogeneous data similarities. Similar to SH [178], CVH [90] is computationally simple and efficient. However, it also assumes that all feature representations follow a uniform distribution, which no longer holds for cross-modal data.

CMFH [28] employs a collective matrix factorization procedure with a latent factor model preserving the data representation similarities. A key assumption of CMFH [28] is that all modalities of an instance generate identical or similar codes. This assumption works well when data carry simple concepts. However, it fails to handle information-rich data, as the multiple instances hidden in data confuse the learning objective.

LCMH [215] is advanced in optimization efficiency, where the training time complexity is linear to the size of the dataset. During training, data from each modality are partitioned into a set of clusters. Intra-modality code similarity is preserved by keeping code distances to the encoded clustering centers. Nevertheless, the inter-modality similarity is insufficiently considered in LCMH [215]. A simple common subspace transformation is not adequate in mitigating the gap between the heterogeneous representations.

MSAE [176] introduces several sets of auto-encoders to formulate the hash functions for heterogeneous data, where a two-phase training procedure is required for pre-training and fine-tuning. The pre-training phase preserves the intra-modality data similarities and the fine-tuning phase mitigates the cross-modal data heterogeneity. However, this two-phase optimization routine doubles the training time and the intra-modality data similarities are no longer preserved after fine-tuning.

Different from MSAE [176], IMH [163] simultaneously explores the intra-modal and inter-modal encoding consistency with two linear regression models. This is reached by combining the inter-media consistency and intra-media consistency together in the learning objective. The problem of IMH [163] is that the differences of the learned codes are measured in Euclidean distance, which does not fully reflect the Hamming distances, and therefore the encoding quality is limited.

2.2.2 Supervised Cross-Modal Hashing

Similar to single-modal hashing, supervised cross-modal hashing requires labels or pair-wise data relations to train the model and the retrieval performance is in general better than the unsupervised ones.

Cross-Modal Similarity Sensitive Hashing (CMSSH) [7] simply preserves the inter-modality correlation using label information and does not consider any other cross-modal problems. As a result, CMSSH is not producing outstanding retrieval performance.

Zhang and Li propose the Semantic Correlation Maximization (SCM) [205], which firstly establishes the cross-modal data affinity matrix according to their semantic labels and then, reconstructs this affinity using the hashed codes. By keeping the orthogonality constraint of the projection function, SCM produces compact binary codes for heterogeneous data retrieval. However, reconstructing the affinity matrix of the training set requires large memory, which is impractical for large-scale datasets.

Semantics-Preserving Hashing (SePH) [103] proposed by Lin *et al.* treats the data semantic affinity as a probability distribution model P. The approximated affinity Q is obtained by computing the code distance in Hamming space. By minimizing the Kullback-Leibler divergence (K-L divergence) between P and Q, the produced binary codes are forced to be semantic-aware. Training SePH [103] is time-consuming since computing the K-L divergence between the two distributions requires vast computational resources.

Quantized Correlation Hashing (QCH) [180] introduces the quantization loss across the modalities. It simultaneously optimizes inter-modality correlation and quantization error, but does not consider intra-modality data relation and semantic information. As a result, the produced codes are not ideal for similarity search.

Other iconic shallow supervised cross-modal hashing methods include Discriminative Coupled Dictionary Hashing (DCDH) [201], Sparse Multi-Modal Hashing (SM^{2}H) [181], Iterative Multi-View Hashing (IMVH) [62], *etc.* Most of these models are evaluated on image-text datasets. However, they are in lack of considering the modal-specific issues such as dominating objects in an image and how to understand combined words within a sentence. The retrieval performance can be further

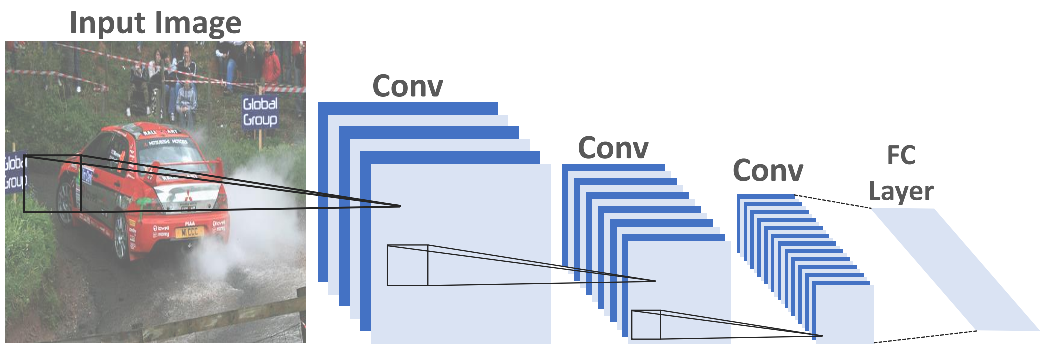


Figure 2.1: A typical structure of an image CNN. Here `Conv` refers to the convolutional layer and `FC layer` refers to the conventional fully-connected layer.

improved by proposing better ways to model image and text data, to which one feasible solution is to introduce deep learning techniques to formulate the hash functions. In the next section, we briefly introduce how deep learning currently is applied to data hashing and retrieval.

2.3 Deep Learning and Neural Networks

Known as a fast-growing research sub-branch of machine learning, deep learning techniques have been successfully adopted in several fields. Following the illuminating Deep Belief Networks (DBN) [55, 130] and Deep Boltzmann Machines (DBM) [150], a wide range of deep learning models have been proposed. Among them, deep neural networks, particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), attain state-of-the-art performance in computer vision and natural language processing tasks such as image recognition, video understanding, document classification, *etc.* Accelerated by Graphics Processing Unit (GPU) parallelization, current computational devices efficiently perform forward-propagation of a neural network and back-propagation with Stochastic Gradient Descent (SGD) on large-scale datasets for training.

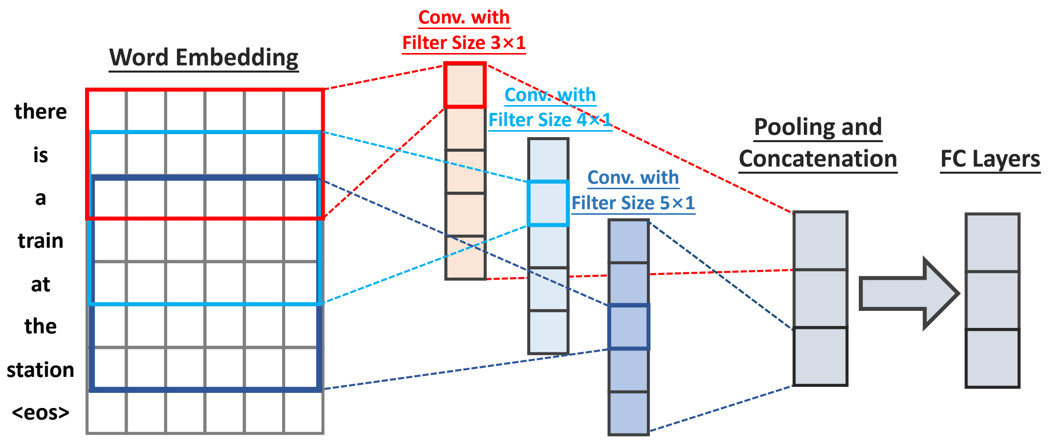

Word Embedding
there
is
a
train
at
the
station
<eos>
Conv. with Filter Size 3×1
Conv. with Filter Size 4×1
Conv. with Filter Size 5×1
Pooling and Concatenation
FC Layers

Figure 2.2: An illustration of how convolution is performed and computed along the text sequence. The structure is similar to the one described in [74]. The kernel numbers are omitted here for the simplicity of illustration.

2.3.1 Convolutional Neural Networks for Images and Texts

As a unique variant of conventional neural networks, CNNs [39, 94, 95] have been proven to be an efficient way in modelling images. The core of CNNs is the convolutional layers, where a sliding window of convolutional kernel filters the feature map obtained from the bottom layer. Convolutional layers are usually stacked with different filtering window sizes. In this way, CNNs conserve spatial information of an image from multiple scales. Combining convolutional layers with pooling [6, 153] and normalization [63, 119] mechanisms, CNNs learn compact hidden representations of the input data.

Figure 2.1 shows an illustrative example of how CNNs are applied to image understanding tasks. Popular CNN structures for images include AlexNet [85], VGG net [157], GoogLeNet [166] and ResNet [52]. The above-mentioned models obtain state-of-the-art classification performance on the benchmarked ImageNet challenge [147] and are partially adopted in many cutting-edge researches, *e.g.* detection [144, 145], segmentation [50] and super-resolution [96], with pre-trained network parameters publicly available.

In addition to the success in image-related tasks, CNNs are also able to model text [74, 23], where words are represented as embedding vectors and convolution is performed along the word sequence. Figure 2.2 illustrates this procedure.

2.3.2 Hashing with Deep Neural Networks

An early attempt of hashing with deep learning technologies is Semantic Hashing [151], employing a stack of DBMs to build the hash function. Recent years observe dramatic progressions in learning to hash with deep neural networks. Compared with traditional linear or random projection as hash functions, a deep neural network is significantly advanced in its non-linearity and the ability to catch complex information hidden in data. In general, deep hashing can be achieved by topping a shot fully-connected layer on a neural network and then performing thresholding or quantization on the activations of this layer. In image hashing, the corresponding hashing networks are basically prototyped using the aforementioned CNN structures.

A number of supervised deep hashing models [8, 10, 15, 64, 97, 183, 208] highly improve image retrieval performance, compared with the shallow ones. The recent SUpervised structured BInary Code (SUBIC) [64] employs the block-softmax nonlinearty on top of the encoding network. Its learning objective is formed by two entropy-based penalties, improving the one-hot block quality of the output codes. However, SUBIC does not fully utilize the code length as the block design does not allow multiple ones to exist in a certain segment of code, resulting in redundant code bins. Some other works seek alternating optimization solutions, iteratively updating the network parameters and targeted binary codebooks [10], while the training efficiency is not guaranteed. Alternating training is slow because the codebook updating step can be hardly accelerated by GPU or parallel computation and requires huge memory to load representations for large-scale datasets.

On the other hand, relatively fewer research attention is paid on unsupervised deep hashing and the retrieval performance is still far from satisfactory. Liong *et al.* [104] propose the Deep Hashing (DH) model by introducing quantization loss and bit decorrelation loss to train an encoding network. The encoder is composed of a stack of fully-connected layers and the input of DH [104] is heuristic feature. As the learning objective does not utilize similarity or semantic information of data, DH [104] is suboptimal for similarity search and data retrieval. Deepbit [101] produced by Lin *et al.* targets on learning compact codes with minimized quantization error, evenly distributed codes and uncorrelated bits. By augmenting training images with a set of rotation angles, the network output is claimed to be rotation-invariant to the input. Deepbit [101] does not employ any similarity-based learning objective, and therefore is suboptimal for similarity search. Do *et al.* [30] slightly improve the hashing quality with a complex multi-step optimization scheme, which is named as Unsupervised Binary Deep Neural Network (UN-BDNN). Training image features are firstly rendered to the encoding network to generate codes. The output codes are updated according to the original feature similarity and then stored as learning target for next training epoch. This design is also inefficient in optimization. Recently, deep variational models [76] are employed in deep hashing. Chaidaroon and Fang [18] quantize the latent variables of a Variational Auto-Encoder (VAE) [76] to hash text documents. However, simply quantizing the real-valued latent variables with a Gaussian prior is suboptimal in learning reliable codes. Dai *et al.* [25] propose the stochastic neurons to reparametrize the binary variables, allowing back-propagation through layers, but this simple encoder-decoder structure is not suitable for encoding high-dimensional data.

Deep hashing is also applied to cross-modal retrieval. Jiang *et al.* [66] propose an end-to-end cross-modal hashing network named as Deep Cross-Modal Hashing (DCMH), but they fail to design modality-specific structures in understanding images and texts. Text data are represented using simple word count vectors as text encoding network inputs, which is inadequate in leveraging information carried by sentences. Similar to [14], DCMH [66] involve alternating training procedure, which largely influences their training efficiency. On the other hand, though much effort has been devoted to optimizing hash functions, DCMH is still unable to solve problem 3 described in Section 1.1. When the categories of test data have not been seen during training phase, DCMH [66] does not produce compact codes for retrieval.

2.4 Summary

In this section, we discussed the state-of-the-art algorithms in single-modal hashing and cross-modal hashing. These models fail to solve problem 1 and 2 described in Chapter 1, leading to unsatisfactory retrieval performance. In addition, the above-mentioned models are not able to handle the train-test category exclusion problem for *zero-shot* retrieval. In the following chapters, we introduce three novel deep hashing models for similarity retrieval with significant performance improvement in these areas.

Chapter 3

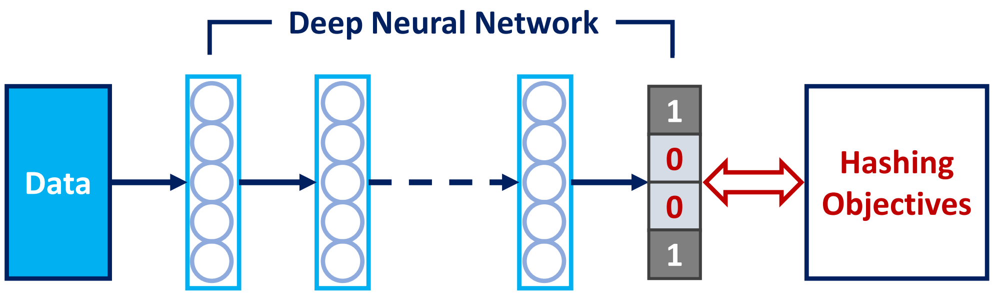
Unsupervised Deep Hashing for Image Retrieval

3.1 Introduction and Motivation

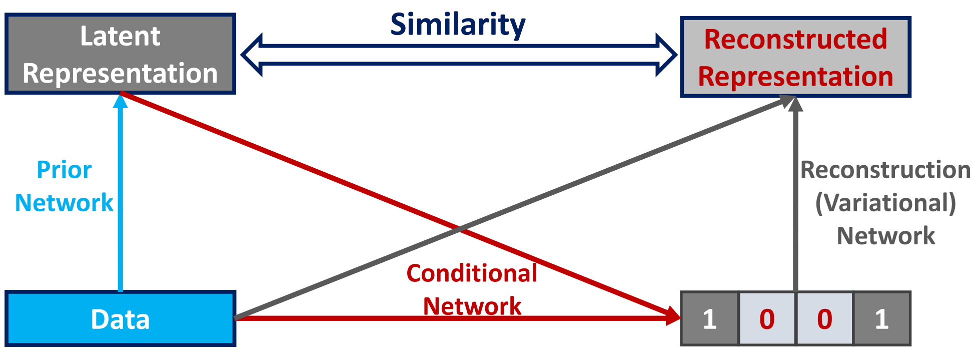
Embedding high-dimensional data representations to low dimensional binary codes, hashing algorithms arouse wide research attention in computer vision, machine learning and data mining. Considering the low computational cost of approximate nearest neighbour search in the Hamming space, hashing techniques deliver more effective and efficient large-scale data retrieval than real-valued embeddings. Hashing methods can be typically categorized as either supervised or unsupervised hashing, while this chapter focuses on the latter.

Supervised hashing [155, 44, 86, 172, 132, 108, 106] utilises data labels or pair-wise similarities as supervision during parameter optimization. It attains relatively better retrieval performance than the unsupervised models as the conventional evaluation measurements of data retrieval are highly related to the labels. However, due to the cost of manual annotation and tagging, supervised hashing is not always appreciated and demanded. On the other hand, unsupervised hashing [43, 151, 178, 47, 198, 163, 82, 51, 111, 112, 110, 107] learns the binary encoding function based on data representations and require no label information, which eases the task of data retrieval where human annotations are not available.

Existing research interests on unsupervised hashing involve various strategies to formulate the encoding functions. Mathematically profound as these works are, the performance of the shallow unsupervised hashing on similarity retrieval is



(a) The conventional unsupervised deep hashing.



(b) The proposed unsupervised deep hashing model.

Figure 3.1: Conventional unsupervised deep hashing models (a) only regularize the output binary codes. Contrarily, our proposed method (b) focus on reconstructing the latent representations from the encoded binaries and force it to be similar to the ones produced by the original data features.

still far from satisfying. This is possibly due to the fact that the simple encoding functions, *e.g.* linear projections, in these works are not capable to handle complex data representations, and therefore the generated codes are suspected to be less informative.

Recently, deep learning is introduced into image hashing, suggesting an alternative manner of formulating the binary encoding function. Although supervised deep hashing has been proven to be successful [14, 214, 91, 184, 109], existing works on unsupervised deep hashing [104, 101, 30, 17] are yet suboptimal. Different from the conventional shallow methods mentioned above [43, 115, 113], unsupervised deep hashing models mainly follow the mini-batch SGD routine for parameter optimization. Consequently, providing no label information, the intrinsic structure and similarities of the whole sample space can be skewed within training batches by these models.

Driven by the issues discussed above, a novel deep unsupervised hashing algorithm is proposed which utilises the structural statistics of the whole training data to produce reliable binary codes. The auto-encoding variational algorithms [76] have shown great potential in several applications [193, 89]. The recent Conditional Variational Auto-Encoding networks [160] provide an illustrative way to build a deep generative model for structured outputs, by which we are inspired to establish our deep hashing model, named as Deep Variational Binaries (DVB). In particular, the latent variables of the variational Bayesian networks [76] are leveraged to approximate the representation of the pre-computed pseudo clustering centre that each data point belongs to. Thus the binary codes can be learnt as informative as the input features by maximizing the conditional variational lower bound of our learning objective. It is worth noticing that we are not using the quantized latent variables as binary representations. Instead, the latent variables are treated as auxiliary data to generate the conditional outputs as hashed codes.

The main difference between DVB and most existing unsupervised deep hashing is shown in Figure 3.1. DVB employs a data reconstruction procedure to generate compact codes. The output binaries are used to reconstruct the latent representations (from the reconstruction network) which need to be close to the latent variables produced by the original data feature (from the prior network).

The contribution of this chapter can be summarized as:

- To the best of our knowledge, DVB is the first unsupervised deep hashing work in the framework of variational inference suitable for image retrieval.
- The proposed deep hashing functions are optimized efficiently, requiring no alternating training routine.
- DVB outperforms state-of-the-art unsupervised hashing methods by significant margins in image retrieval on three benchmarked datasets, i.e. CIFAR-10, SUN-397 and NUS-WIDE.

The rest of this chapter is organized as follows. A short literature review is provided in Sec. 3.2. We describe the learning framework of the proposed DVB model in Sec. 3.3. Extensive experiments and results are given in Sec. 3.4.

3.2 Related Work

3.2.1 Shallow Unsupervised Hashing

In general, traditional hashing models employ simple projection functions to encode data, *e.g.* linear projections and eigenfunctions.

Iterative Quantization (ITQ) [43] proposed by Gong *et al.* is a classic unsupervised hashing method. It aims at refining the initial projection matrix learned by PCA using an orthogonal rotation function. The hashing quality of ITQ [43] largely relies on the performance of PCA. As is discussed in Chapter 2, this is not a desired design for learning compact data representations because the features produced by PCA cannot be improved during training. Spectral Hashing (SH) developed by Weiss *et al.* [178] learns the hash function by preserving the balanced and uncorrelated constraints of the learnt codes. It assumes that data follow a uniform distribution, but this is not a realistic assumption for large-scale data retrieval.

Locality Sensitive Hashing (LSH) [21] successively applies thresholding mechanisms on projected data to producing binary codes. It is theoretically proven that as the code length increases, the value of Hamming distance between two codes will asymptotically approach the Euclidean distance between their original data representations. This design has been further improved by Shift-invariant Kernelized Locality Sensitive hashing (SKLSH) [141]. Random-projection-based encoding methods can be suboptimal for realistic applications since their performances drop dramatically with the decrease of code length. This is unappreciated as the state-of-the-art hashing algorithms generally produce reasonable retrieval performances with short code length.

Anchor Graph Hashing (AGH) [115] proposed by Liu *et al.* initially introduces the concept of anchor points for graph hashing, and is therefore extended to DGH [113] later. The key idea of graph hashing is to automatically discover the neighbourhood structure inhere data. In AGH [115], this neighbourhood structure is reflected in an anchor-based similarity matrix, where the anchors are obtained from the clustering centres of training data features. The difference between AGH [115] and DGH [113] is that DGH [113] additionally introduces the discrete constraints on output bits in the learning objective, making the produced codes more compact. The weakness of AGH [115] and DGH [113] is that bad setting of number of clustering centres influences the choice of anchor points and encoding quality as well. This setting varies from different datasets and requires additional training time to find the best value.

The recent Latent Structure Preserving Hashing (LSPH) [111] combines unsupervised hashing with nonnegative matrix factorization. The aim is to effectively preserve data probabilistic distribution and capture the locality structure from the high-dimensional data. Each bit of the encoded data represents a data-driven latent attribute. However, its multi-layered design with nonnegative matrix factorization makes the optimization slow and inefficient.

Despite the drawbacks discussed above, the shallow encoding functions in these works are not able to fully utilize the latent semantics and information hidden in data, because the quantities of learning parameters are limited [95, 104]. This influences the retrieval performance of these models. To this end, we seek solutions for unsupervised hashing in deep learning, where the latent information carried by data can be better leveraged.

3.2.2 Deep Unsupervised Hashing

Recently, supervised deep hashing methods have achieved impressive performances in both single-modal data retrieval [14, 214] and cross-modal retrieval [66, 109]. However, unsupervised deep hashing is still far from satisfactory.

Salakhutdinov *et al.* [151] introduce DBM into binary representation learning. A generative model is pre-trained and then fine-tuned to produce semantically representative codes on the top of the DBM. This early attempt in deep learning does not consider data similarities, and thus it underperforms the shallow methods [43, 111].

Deep Hashing (DH) [104] applies quantization loss on top of a neural network to regularize the output codes. A simple multi-layer perceptron is employed and it merely outperforms existing shallow hashing models with slight margins. The main idea of DH [104] is to quantize the network output and decorrelate the bits, but, similar to [151], it also fails to utilize data similarities and semantic information.

DeepBit [101] focuses on assigning the identical code to rotated images with different rotation angles. Images are firstly rotated and then rendered to a CNN for encoding. In addition to the hashing learning objective of DH [104], a penalty of code distances of an image with different rotation angles is introduced in DeepBit [101]. However, the code difference between two images is not considered. So the encoding quality is not satisfying.

Unsupervised Binary Deep Neural Network (UN-BDNN) [30] becomes one of the best-performing deep unsupervised methods. A simple multi-layer neural network is involved for encoding. UN-BDNN [30] preserves code similarities according to the feature distance in Euclidean space. However, its alternating optimization routine limits its training efficiency, since it requires additional computation which currently cannot be accelerated by GPUs.

3.2.3 Auto-Encoding Variational Networks

The variational Auto-Encoder (VAE) [76] proposed by Kingma *et al.* has become a popular generative model recently. Combining parametrized latent distributions and sampling methods for lower bound inference, VAE has been applied to many computer vision and machine learning aspects, *e.g.* domain adaptation [139], text modelling [196], tag-image generation [193], conditioned face reconstruction [89] and many other applications [189, 77, 36, 154]. Sohn *et al.* [160] extend the vanilla VAE to a conditional generative model to learning structured outputs, namely Conditional Variational Auto-Encoder (CVAE). In this work, the advances in CVAE are leveraged to formulate our theoretical basis.

By the time of writing, we are aware that Chaidaroon *et al.* [18] propose a variational binary encoder for text hashing. However, [18] is not suitable for image encoding for the following reasons.

- [18] takes discrete word count vectors as input, while images would have longer and more complex representations. Simply applying the vanilla VAE framework to image features as [18] may result in a lossy reconstructed likelihood distribution from short binary codes. It is hard to directly reconstructing high-dimensional representations from short features with a single decoder.
- In [18], the encoded binary representations are obtained by quantizing the latent variables, which means the output codes can be extremely intolerant to the randomness introduced by the Monte Carlo sampling procedure on the latent space of VAE.

3.3 Deep Variational Binaries

This work addresses the problem of data retrieval with an unsupervised hashing procedure. Given a data collection $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N} \in \mathbb{R}^{d \times N}$ consisting N data points with d-dimensional real-valued representations, the DVB model learns an encoding function $f(\cdot)$, parametrized by θ, so that each data point can be represented as

$$\mathbf{b}_i = \mathtt{sign}\left(f\left(\mathbf{x}_i; \theta\right)\right) \in \{-1, 1\}^m. \tag{3.1}$$

Here m indicates the encoding length and $\mathtt{sign}(\cdot)$ refers to the sign function for quantization. In the following description, index i will be omitted when it clearly refers to a single data point. In this section, we firstly explain the way to empirically exploit the intrinsic structure of the training set by introducing a set of latent variables $\mathbf{z}$ and then, the encoding function $f(\cdot)$ is formulated by a Monte Carlo sampling procedure for out-of-sample extension.

3.3.1 The Variational Model

As shown in Figure 3.2, the DVB framework involves three types of variables, i.e. the data representations $\mathbf{x} \in \mathbb{R}^d$, the output codes $\mathbf{b} \in \{-1, 1\}^m$ and the latent representations $\mathbf{z} \in \mathbb{R}^l$ as auxiliary variables, where l denotes the dimensionality of the latent space. The variables in DVB formulate three probabilistic models, i.e.

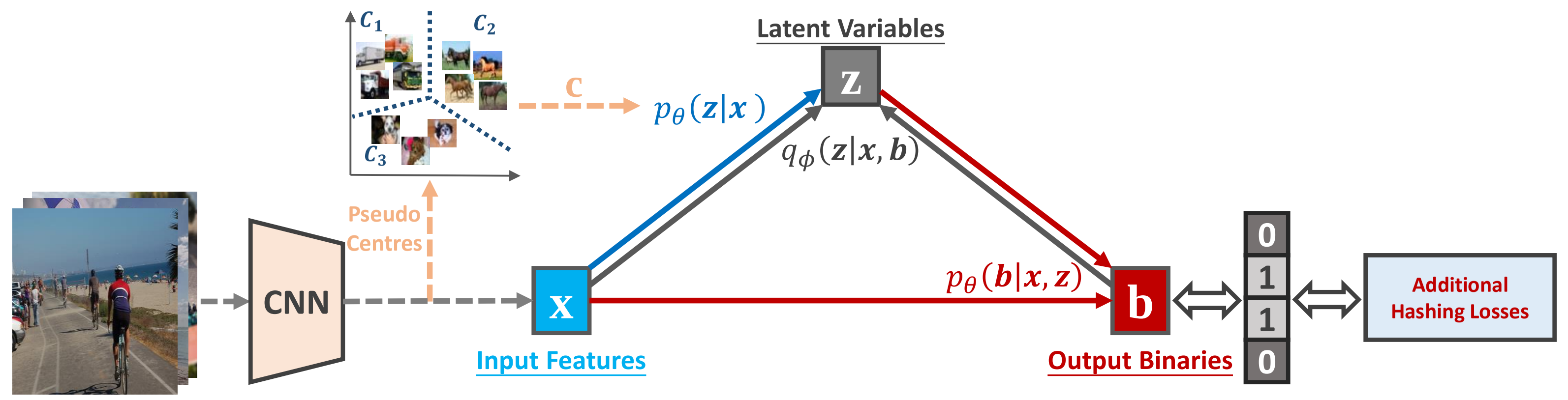


Figure 3.2: Illustration of DVB as a graphical model. The arrowed full lines with different colours indicate different probability models implemented with deep neural networks. In particular, $p_\theta(\mathbf{z}|\mathbf{x})$ in blue acts as the (conditional) prior of the latent variables $\mathbf{z}$; $p_\theta(\mathbf{b}|\mathbf{x},\mathbf{z})$ refers to the network producing $\mathbf{b}$; $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$ is the variational posterior of $\mathbf{z}$. The component computing data centres assigns a low-dimensional pseudo centre $\mathbf{c}$ to each data point $\mathbf{x}$ using some dimensionality reduction and clustering methods. Implementation details are given in Section 3.3.

the conditional prior $p_\theta(\mathbf{z}|\mathbf{x})$, the variational posterior $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$ and the generation network $p_\theta(\mathbf{b}|\mathbf{x},\mathbf{z})$. Following Kingma *et al.* [76], the probability models here are implemented using deep neural networks, parametrized by θ or ϕ. We consider the prototype of learning objective maximizing the log-likelihood $\log p_\theta(\mathbf{b}|\mathbf{x})$ for each training data point by approximating the true posterior $p_\theta(\mathbf{z}|\mathbf{x},\mathbf{b})$ using $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$. Starting with the K-L divergence between $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$ and $p_\theta(\mathbf{z}|\mathbf{x},\mathbf{b})$ according to [160]:

$$\begin{aligned}&KL(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) \parallel p_\theta(\mathbf{z}|\mathbf{x},\mathbf{b}))\\&= \int q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}{p_\theta(\mathbf{z},\mathbf{b}|\mathbf{x})} d\mathbf{z} + \log p_\theta(\mathbf{b}|\mathbf{x}),\end{aligned} \tag{3.2}$$

the likelihood of $\mathbf{b}$ can be written as [76]

$$\begin{aligned}\log p_\theta(\mathbf{b}|\mathbf{x}) =& KL(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) \parallel p_\theta(\mathbf{z}|\mathbf{x},\mathbf{b})) - \int q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}{p_\theta(\mathbf{z},\mathbf{b}|\mathbf{x})} d\mathbf{z}\\ \geq& \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}\left[\log p_\theta(\mathbf{b},\mathbf{z}|\mathbf{x}) - \log q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})\right].\end{aligned} \tag{3.3}$$

Here the expectation term $\mathbb{E}[\cdot]$ becomes the prototype of the learning objective of DVB. Considering the neural networks mentioned above, from [160], we factorize the lower bound and we obtain:

$$\begin{aligned}-\log p_\theta(\mathbf{b}|\mathbf{x}) \leq \mathcal{L}&\\&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}\left[\log q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) - \log p_\theta(\mathbf{b},\mathbf{z}|\mathbf{x})\right]\\&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}[\log q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) - \log p_\theta(\mathbf{z}|\mathbf{x}) - \log p_\theta(\mathbf{b}|\mathbf{x},\mathbf{z})]\\&= KL(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b}) || p_\theta(\mathbf{z}|\mathbf{x})) - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}\left[\log p_\theta(\mathbf{b}|\mathbf{x},\mathbf{z})\right].\end{aligned} \tag{3.4}$$

We denote $\mathcal{L}$ as the numerical **inverse** of the lower-bound to $\log p_\theta(\mathbf{b}|\mathbf{x})$ for the ease of description in the rest of this chapter. Therefore DVB performs SGD to **minimize** $\mathcal{L}$. It can be observed that the construction-reconstruction similarities of the latent variables are instantiated in the KL divergence between $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$ and $p_\theta(\mathbf{z}|\mathbf{x})$.

As image data are usually presented in high-dimensional representations, directly reconstructing $\mathbf{x}$ from $\mathbf{z}$ as [76, 18] is not optimal and could induce redundant noise to the training procedure. In DVB, $\mathbf{z}$ act as auxiliary variables encoding latent information through the conditional network $p_\theta(\mathbf{z}|\mathbf{x})$. By reducing the divergence between the posterior $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$ and $p_\theta(\mathbf{z}|\mathbf{x})$, the generated binaries $\mathbf{b}$

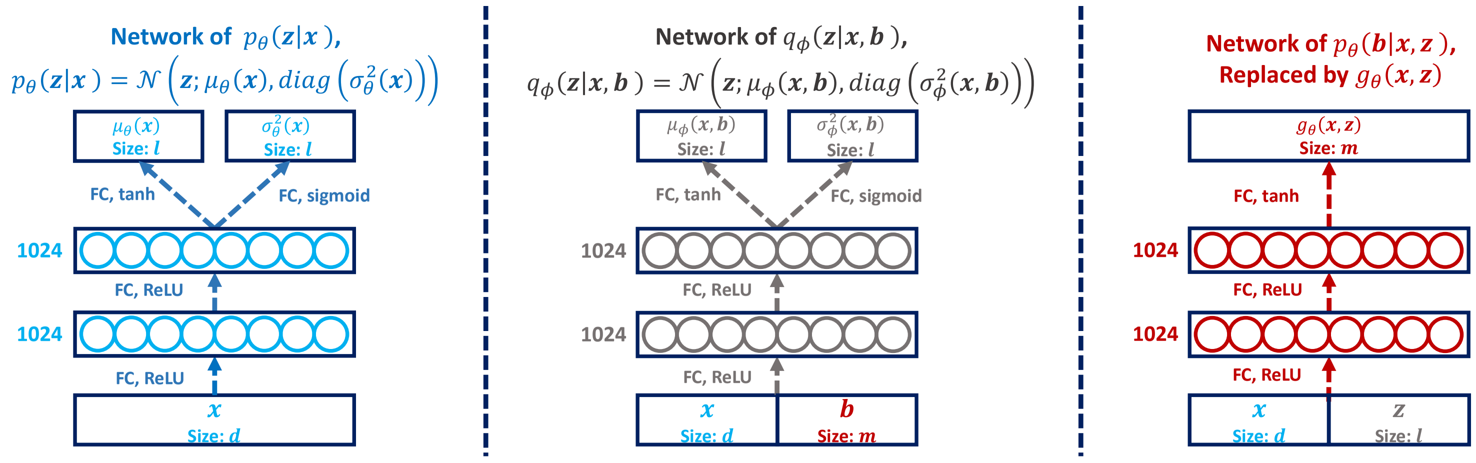

Network of $p_\theta(\boldsymbol{z}|\boldsymbol{x})$,
$p_\theta(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{z}; \mu_\theta(\boldsymbol{x}), diag\left(\sigma_\theta^2(\boldsymbol{x})\right)\right)$
$\mu_\theta(\boldsymbol{x})$ Size: l
$\sigma_\theta^2(\boldsymbol{x})$ Size: l
FC, tanh
FC, sigmoid
1024
FC, ReLU
1024
FC, ReLU
$\boldsymbol{x}$ Size: d
Network of $q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{b})$,
$q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{b}) = \mathcal{N}\left(\boldsymbol{z}; \mu_\phi(\boldsymbol{x},\boldsymbol{b}), diag\left(\sigma_\phi^2(\boldsymbol{x},\boldsymbol{b})\right)\right)$
$\mu_\phi(\boldsymbol{x},\boldsymbol{b})$ Size: l
$\sigma_\phi^2(\boldsymbol{x},\boldsymbol{b})$ Size: l
FC, tanh
FC, sigmoid
1024
FC, ReLU
1024
FC, ReLU
$\boldsymbol{x}$ Size: d
$\boldsymbol{b}$ Size: m
Network of $p_\theta(\boldsymbol{b}|\boldsymbol{x},\boldsymbol{z})$,
Replaced by $g_\theta(\boldsymbol{x},\boldsymbol{z})$
$g_\theta(\boldsymbol{x},\boldsymbol{z})$ Size: m
FC, tanh
1024
FC, ReLU
1024
FC, ReLU
$\boldsymbol{x}$ Size: d
$\boldsymbol{z}$ Size: l

Figure 3.3: Detailed settings of the three networks in DVB, i.e. the prior network $p_\theta(\mathbf{z}|\mathbf{x})$ (left),variational (reconstruction) network $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})$ (middle), and the conditional network $p_\theta(\mathbf{b}|\mathbf{x},\mathbf{z})$ (right).

are supposed to have similar semantics to the original feature $\mathbf{x}$ in reconstructing $\mathbf{z}$. To solve the intractability of the posterior $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{b})$, the inference network is built using the reparameterizaion trick in [76] with a Gaussian distribution so that

$$q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{b}) = \mathcal{N}\left(\mathbf{z};\ \mu_\phi(\mathbf{x}, \mathbf{b}), \mathtt{diag}\left(\sigma^2_\phi(\mathbf{x}, \mathbf{b})\right)\right). \tag{3.5}$$

Note that all $\mu_\cdot(\cdot)$ and $\sigma_\cdot(\cdot)$ can be implemented with multi-layer neural networks, which is provided in Figure 3.3. To obtain a certain set of sampled latent variables $\mathbf{z}^{(s)}$ from $q_\phi(\cdot)$, a Monte Carlo sampling process can be conducted by introducing a set of small-valued random values ϵ:

$$\mathbf{z}^{(s)} = \mu_\phi(\mathbf{x}, \mathbf{b}) + \epsilon \odot \sigma_\phi(\mathbf{x}, \mathbf{b}), \tag{3.6}$$

where $\odot$ refers to the element-wise multiplication product. A similar trick is also performed on $p_\theta(\mathbf{z}|\mathbf{x})$ as follows

$$p_\theta(\mathbf{z}|\mathbf{x}) = \mathcal{N}\left(\mathbf{z};\ \mu_\theta(\mathbf{x}), \mathtt{diag}\left(\sigma^2_\theta(\mathbf{x})\right)\right). \tag{3.7}$$

Although the continues distributions $p_\theta(\mathbf{z}|\mathbf{x})$ and $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{b})$ can be reparameterized, it is still hard to model $p_\theta(\mathbf{b}|\mathbf{x}, \mathbf{z})$ because $\mathbf{b}$ needs to be discrete and there is no additional supervision available. Hence, the log-likelihood $\log p_\theta(\mathbf{b}|\mathbf{x}, \mathbf{z})$ is replaced by a series of deep hashing learning objectives $\mathcal{H}(\cdot)$, i.e.

$$-\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}\left[\log p_\theta(\mathbf{b}|\mathbf{x}, \mathbf{z})\right] \longrightarrow \mathcal{H}\left(g_\theta(\mathbf{x}, \mathbf{z})\right). \tag{3.8}$$

Here $g_\theta(\cdot)$ refers to the deep neural network to generate $\mathbf{b}$ so that $\mathbf{b} = \mathtt{sign}\left(g_\theta(\mathbf{x}, \mathbf{z})\right)$.

3.3.2 Exploiting Intrinsic Data Structure.

In addition to the lower-bound mentioned above, we consider utilising the statistical information on the whole training set for better performance. Inspired by [115, 113], a small set of K anchor points $\{\mathbf{c}^j\}_{j=1}^K \in \mathbb{R}^{l\times K}$ are computed before the SGD training starts, which is also shown in Figure 3.2. Each anchor point refers to a pseudo clustering centre of the training data. Then each data point $\mathbf{x}_i$ is assigned with a clustering centre by nearest neighbour search, i.e. $\{\mathbf{x}_i, \mathbf{c}_i\}$. In practice, this is achieved by successively performing dimension reduction and clustering on the training set. Different from [115, 113], we are not building the anchor graph on the whole dataset since this is not practical for mini-batch SGD. Instead, the latent variable $\mathbf{z}$ is used to predict the representation of the corresponding anchor $\mathbf{c}$ of each $\mathbf{x}$. More precisely, the mean network $\mu_\theta(\mathbf{x})$ of $p_\theta(\mathbf{z}|\mathbf{x})$ is related to $\mathbf{c}$, formulating an additional $l2$ loss term, which particularly requires $\mathbf{z}$ have the same dimensionality as $\mathbf{c}$. This procedure intuitively endows the conditional network $p_\theta(\mathbf{z}|\mathbf{x})$ with more informative latent semantics. Therefore, the total learning objective of Equation (3.4) can be rewritten as

$$\widetilde{\mathcal{L}} = KL\left(q_\phi\left(\mathbf{z}|\mathbf{x},\mathbf{b}\right)||p_\theta\left(\mathbf{z}|\mathbf{x}\right)\right) + \mathcal{H}\left(g_\theta(\mathbf{x},\mathbf{z})\right) + \left\|\mu_\theta\left(\mathbf{x}\right) - \mathbf{c}\right\|^2. \tag{3.9}$$

Note that $\widetilde{\mathcal{L}}$ here can be no longer regarded as the exact lower-bound of $\log p_\theta(\mathbf{b}|\mathbf{x})$. This empirical learning objective partially represents the likelihood of $\mathbf{b}$ which requires further attention with regard to hashing. In the next subsection, the details of the hashing objective term $\mathcal{H}\left(g_\theta(\mathbf{x},\mathbf{z})\right)$ is discussed.

3.3.3 Hashing Objectives

The hashing objective $\mathcal{H}(\cdot)$ in Equation (3.9) in replacement of $-\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{b})}\left[\log p_\theta\left(\mathbf{b}|\mathbf{x},\mathbf{z}\right)\right]$ is formulated by several unsupervised loss components to regularise the output of the proposed hashing model. Since DVB is trained using mini-batch SGD, the losses need to be able to back-propagate. Based on several unsupervised deep hashing works [104, 101, 30], we formulate the following hashing losses to construct $\mathcal{H}(\cdot)$ within a batch of data points $\mathbf{X}_B = \{\mathbf{x}_i\}_{i=1}^{N_B}$ and sampled latent variables $\mathbf{Z}_B = \{\mathbf{z}_i\}_{i=1}^{N_B}$, where N_B is the batch size.

3.3.3.1 Quantization Penalty

As DVB produces binary codes, the output bits of $g_\theta(\cdot)$ need to be close to either 1 or -1. This minimizes the numerical gap between the network output and the quantized product of the $\texttt{sign}(\cdot)$ function. The quantization loss can thus be written with a Frobenius norm as follows

$$\mathcal{H}_1 = \left\|g_\theta\left(\mathbf{X}_B,\mathbf{Z}_B\right) - \texttt{sign}\left(g_\theta\left(\mathbf{X}_B,\mathbf{Z}_B\right)\right)\right\|_{\mathrm{F}}^2. \tag{3.10}$$

The quantization loss is widely adopted in several hashing works [104, 101, 14, 214] with different formulations. As is discussed in [104], the quantization loss helps the network output to be close to 1 or -1, and thus prevent the model from performance drop due to subsequent quantization with the $\texttt{sign}(\cdot)$ function. In our experiments, we find the Frobenius norm works best for DVB with a `tanh` activation on the top layer of $g_\theta(\cdot)$.

3.3.3.2 Bit Decorrelation

The encoded binaries in hashing algorithms are in general short in length. To make the produced code representative, it is necessary to decorrelate each bit and balance the quantity of 1 and -1 in a code vector. To this end, the second component of $\mathcal{H}(\cdot)$ is derived as

$$\mathcal{H}_2 = \| g_\theta(\mathbf{X}_B, \mathbf{Z}_B)^\intercal g_\theta(\mathbf{X}_B, \mathbf{Z}_B) - \mathbf{I} \|_{\mathrm{F}}^2, \tag{3.11}$$

where $\mathbf{I}$ refers to the identity matrix and both $\mathbf{X}_B$ and $\mathbf{Z}_B$ are row-ordered matrices. Equation (3.11) suggests an indirect way to enrich the information encoded in the binary codes by balancing the output bits.

3.3.3.3 In-Batch Similarity

For unsupervised hashing, it is usually in demand to closely encode data samples that have similar representations into the Hamming space. Inspired by [53, 5], the in-batch Laplacian graph is introduced to build the last term of $\mathcal{H}(\cdot)$. To do this, an in-batch Laplacian matrix is defined by $\mathbf{S} = \texttt{diag}(\mathbf{A}\mathbb{1}) - \mathbf{A}$. Here $\mathbf{A}$ is an $N_B \times N_B$ distance matrix of which each entrance $\mathbf{A}_{ij}$ is computed by

$$\mathbf{A}_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}, \tag{3.12}$$

where t is a small-valued hyper-parameter. A trace-like learning objective for in-batch similarity can be written as

$$\mathcal{H}_3 = -\texttt{trace}\left(g_\theta(\mathbf{X}_B, \mathbf{Z}_B)^\intercal \mathbf{S}\ g_\theta(\mathbf{X}_B, \mathbf{Z}_B) \right). \tag{3.13}$$

$\mathcal{H}_3$ functionally works similarly to the pre-computed low-dimensional clustering centres $\mathbf{c}$ in preserving the unlabelled data similarities. However, $\mathcal{H}_3$ focuses on regulating $\mathbf{b}$ within a batch while $\mathbf{c}$ provides support to form the latent space $\mathbf{z}$ on the whole training set.

Therefore, $\mathcal{H}$ in Equation (3.9) can be formulated by a weighted combination of $\mathcal{H}_1$, $\mathcal{H}_2$ and $\mathcal{H}_3$:

$$\mathcal{H}\left(g_\theta(\mathbf{x},\mathbf{z})\right) = \alpha_1\mathcal{H}_1 + \alpha_2\mathcal{H}_2 + \alpha_3\mathcal{H}_3, \tag{3.14}$$

where α_1, α_2 and α_3 are treated as hyper-parameters.

3.3.4 Network Setup

Three neural networks are built as the implementations of $p_\theta\left(\mathbf{z}|\mathbf{x}\right)$, $q_\phi\left(\mathbf{z}|\mathbf{x},\mathbf{b}\right)$ and $g_\theta\left(\mathbf{x},\mathbf{z}\right)$, shown in Figure 3.3. All the networks are composed of two fully-connected hidden layers, topped by the output layers representing the respective probabilistic models. We fix the length of the hidden layers to be 1024 with `ReLU` activations [131] for all networks. The output lengths of the networks are determined by the dimensionality of the latent space, notified as l. The `tanh` activations are applied to $g_\theta\left(\cdot\right)$ and the mean networks $\mu_\cdot\left(\cdot\right)$, while the `sigmoid` activations are empirically chosen for the variance networks $\sigma^2_\cdot\left(\cdot\right)$. Note that we are not building enormous networks as the size of the encoding network may heavily influence the coding efficiency, which is not appreciated in deep hashing.

3.3.5 Optimization

By introducing the hashing losses discussed in Subsection 3.3.3 into DVB, the overall learning objective $\widetilde{\mathcal{L}_B}$ on a mini-batch $\mathbf{X}_B$, after combining Equation (3.9) and Equation (3.14), can be written as follows

$$\begin{aligned}\widetilde{\mathcal{L}_B} =& \sum_{i=1}^{N_B}(KL\left(q_\phi\left(\mathbf{z}_i|\mathbf{x}_i,\mathbf{b}_i\right)||p_\theta\left(\mathbf{z}_i|\mathbf{x}_i\right)\right) \\ &+ \|\mu_\theta\left(\mathbf{x}_i\right) - \mathbf{c}_i\|^2) + \alpha_1\mathcal{H}_1 + \alpha_2\mathcal{H}_2 + \alpha_3\mathcal{H}_3.\end{aligned} \tag{3.15}$$

The SGD training procedure of DVB is illustrated in Algorithm 1. A flow diagram better illustrating all computation steps is additionally provided in Figure 3.4. For each data point $\mathbf{x}_i$ within a batch $\mathbf{X}_B$, a latent representation $\mathbf{z}_i$ is obtained by sampling the conditional distribution $p_\theta\left(\cdot|\mathbf{x}_i\right)$ and an estimated binary vector $\mathbf{b}_i$ can be calculated by $\mathbf{b}_i = \mathtt{sign}\left(g_\theta(\mathbf{x}_i,\mathbf{z}_i)\right)$ to further compute $\widetilde{\mathcal{L}_B}$. The parameters (θ,ϕ) are updated following the mini-batch SGD. $\Gamma\left(\cdot\right)$ here refers to an adaptive

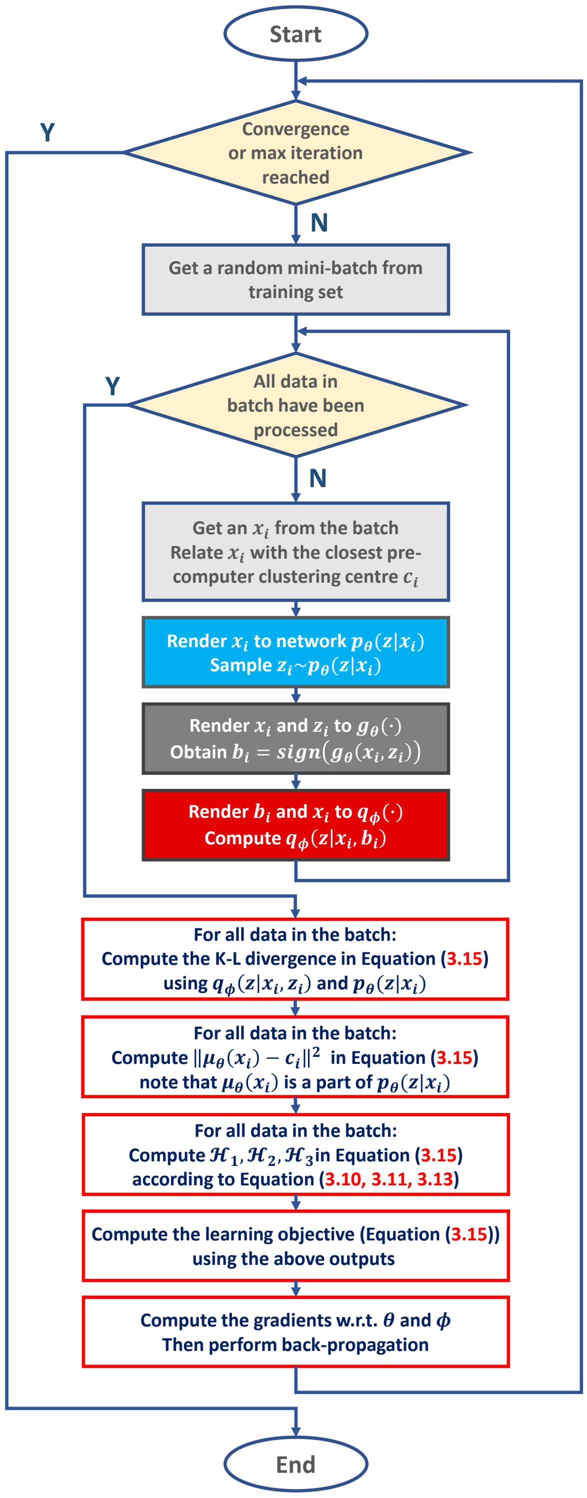

Start
Convergence or max iteration reached
Y
N
Get a random mini-batch from training set
All data in batch have been processed
Y
N
Get an x_i from the batch Relate x_i with the closest pre-computer clustering centre c_i
Render x_i to network $p_\theta(z|x_i)$ Sample $z_i \sim p_\theta(z|x_i)$
Render x_i and z_i to $g_\theta(\cdot)$ Obtain $b_i = sign(g_\theta(x_i, z_i))$
Render b_i and x_i to $q_\phi(\cdot)$ Compute $q_\phi(z|x_i, b_i)$
For all data in the batch: Compute the K-L divergence in Equation (3.15) using $q_\phi(z|x_i, z_i)$ and $p_\theta(z|x_i)$
For all data in the batch: Compute $\|\mu_\theta(x_i) - c_i\|^2$ in Equation (3.15) note that $\mu_\theta(x_i)$ is a part of $p_\theta(z|x_i)$
For all data in the batch: Compute $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ in Equation (3.15) according to Equation (3.10, 3.11, 3.13)
Compute the learning objective (Equation (3.15)) using the above outputs
Compute the gradients w.r.t. θ and ϕ Then perform back-propagation
End

Figure 3.4: Flow diagram of the detailed training procedure of DVB according to Algorithm 1.

Algorithm 1: Parameter Learning of DVB

```
Input: A dataset with representations $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N} \in \mathbb{R}^{d\times N}$ and max
  training iteration $T$
Output: network parameters $\theta$ and $\phi$
Perform dimension reduction and clustering on the dataset to have $\{\mathbf{c}\}$
repeat
    Get a random mini-batch $\mathbf{X}_B$ from $\mathbf{X}$
    for each $\mathbf{x}_i$ in $\mathbf{X}_B$ do
        Relate $\mathbf{x}_i$ with a closest clustering centre representation $\mathbf{c}_i$
        Sample $\mathbf{z}_i \sim p_\theta(\mathbf{z}|\mathbf{x}_i)$ following [76]
        $\mathbf{b}_i = \texttt{sign}(g_\theta(\mathbf{x}_i, \mathbf{z}_i))$
    end
    $\widetilde{\mathcal{L}_B} \leftarrow$ Equation (3.15)
    $(\theta, \phi)^{\text{new}} \leftarrow (\theta, \phi) - \Gamma\left(\nabla_\theta \widetilde{\mathcal{L}_B}, \nabla_\phi \widetilde{\mathcal{L}_B}\right)$ by back-propagation
until convergence or max training iter $T$ is reached;
```

gradient scaler, which is the Adam optimizer [75] in this chapter with a starting learning rate of 10^{-4}.

3.3.6 Out-of-Sample Extension

Once the set of parameters θ is trained, the proposed DVB model is able to encode data out of the training set. Given a query data $\mathbf{x}^q$, the corresponding binary code $\mathbf{b}^q$ can be obtained by a Monte Carlo sampling procedure defined as

$$\begin{aligned} f(\mathbf{x}^q; \theta) =& \frac{1}{L}\sum_{i=1}^{L} g_\theta\left(\mathbf{x}^q, \mathbf{z}_i^{(s)}\right); \quad \mathbf{z}_i^{(s)} \sim p_\theta(\mathbf{z}|\mathbf{x}^q), \\ \mathbf{b}^q =& \texttt{sign}\left(f(\mathbf{x}^q; \theta)\right), \end{aligned} \tag{3.16}$$

which simulates the sampling trick described in [76, 160]. In the experiments of this work, L is fixed to 10 for best performance via cross-validation. Since the output binaries of DVB are not obtained by directly quantizing the sampled latent representation $\mathbf{z}$, we mitigate the problem of uncontrollable output codes from [18] discussed in Sec. 3.2.3.

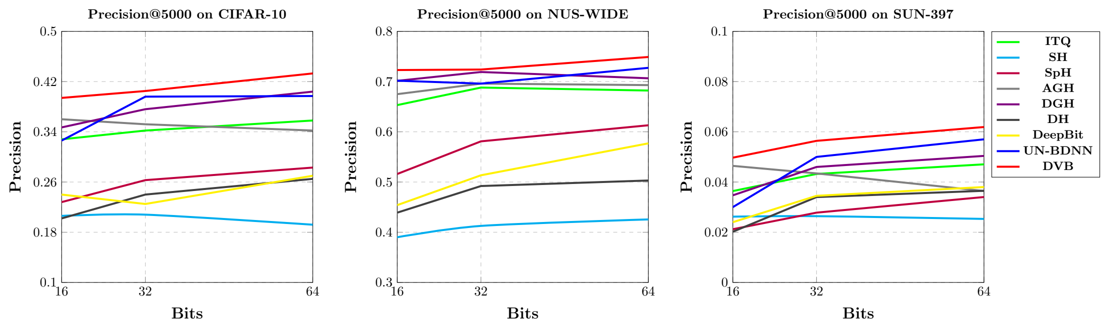


Figure 3.5: Precision@5000 curves for all bits of DVB and several existing methods with VGG-16 [157] features.

3.4 Experiments

The extensive experiments of DVB are conducted on three benchmark image datasets, i.e. CIFAR-10 [84], SUN-397 [187] and NUS-WIDE [22] for image retrieval. We firstly introduce the implementation details, and then the experimental results are provided according to the following themes:

- **Comparison with State-of-the-Art Methods.** Several existing unsupervised hashing models are introduced as baselines for retrieval performance comparison with DVB.
- **Ablation Study.** We assess the design of our proposed hashing model by introducing several variants of DVB as baselines.
- **Hyper-Parameters.** The retrieval performance of DVB *w.r.t.* different hyper-parameters are analysed.
- **Intuitive Results.** Some intuitive results are provided to have a better picture of the encoding quality of DVB.

3.4.1 Implementation Details

The DVB networks are implemented with the well-known deep learning library TensorFlow [1]. Before being rendered to the DVB networks, a 4096-dimensional deep feature vector of each training image is extracted using the output of the

Table 3.1: Image retrieval mean-Average Precision (mAP) on the three datasets with VGG-16 [157] features.

Method	**Deep Hashing**	**CIFAR-10**			**SUN-397**			**NUS-WIDE**		
		16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
ITQ [43]	✗	0.319	0.334	0.347	0.047	0.070	0.086	0.512	0.526	0.538
SH [178]	✗	0.218	0.198	0.181	0.021	0.033	0.048	0.346	0.358	0.365
SpH [54]	✗	0.229	0.253	0.283	0.032	0.039	0.043	0.418	0.456	0.474
LSH [21]	✗	0.163	0.182	0.232	0.006	0.007	0.011	0.410	0.416	0.439
SKLSH [141]	✗	0.103	0.112	0.114	0.005	0.006	0.008	0.377	0.379	0.388
SELVE [216]	✗	0.309	0.281	0.239	0.049	0.072	0.089	0.467	0.462	0.432
AGH [115]	✗	0.301	0.270	0.238	0.059	0.057	0.062	0.498	0.476	0.471
DGH [113]	✗	0.332	0.354	0.356	0.061	0.074	0.079	0.530	0.527	0.496
DH [104]	✓	0.172	0.176	0.179	0.035	0.047	0.056	0.404	0.467	0.427
DeepBit [101]	✓	0.193	0.216	0.219	0.029	0.058	0.061	0.452	0.463	0.496
UN-BDNN [30]	✓	0.301	0.309	0.312	0.062	0.073	0.088	0.513	0.517	0.547
DVB (proposed)	✓	**0.347**	**0.365**	**0.392**	**0.069**	**0.084**	**0.098**	**0.546**	**0.560**	**0.584**

Table 3.2: Image retrieval mean-Average Precision (mAP) and Precision@5000 on CIFAR-10 [84] with GIST features.

Method	**mAP**			**Precision@5000**		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
ITQ [43]	0.153	0.163	0.169	0.177	0.188	0.198
SH [178]	0.127	0.125	0.125	0.144	0.143	0.143
LSH [21]	0.129	0.138	0.149	0.144	0.143	0.169
AGH [115]	0.158	0.153	0.145	0.169	0.169	0.168
DGH [113]	0.160	0.160	0.156	0.187	0.183	0.179
DH [104]	0.162	0.166	0.170	0.182	0.184	0.193
DeepBit [101]	0.159	0.191	0.209	0.180	0.222	0.243
UN-BDNN [30]	0.157	0.161	0.169	0.181	0.189	0.197
DVB	**0.165**	**0.167**	**0.175**	**0.191**	**0.194**	**0.207**

`fc_7` layer of the VGG-16 network [157], pre-trained on ImageNet [147], i.e. $d = 4096$. We follow a similar way presented in [76, 160, 89] to build the deep neural networks $p_\theta(\mathbf{z}|\mathbf{x})$, $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{b})$ and $g_\theta(\mathbf{x}, \mathbf{z})$. The detail structures of these networks in DVB are provided Figure 3.2. The dimensionality of the latent space $\mathbf{z}$ is set to $l = 1024$ via cross-validation. To generate a set of pseudo data centres $\{\mathbf{c}\}$, PCA is performed on the $l2$ normalized training set $\mathbf{X}$ to reduce its dimensionality from 4096 to 1024, followed by a K-means clustering [123] procedure to obtain a set of $\mathbf{c}$. The number of clustering centres K is set according to different datasets. For the rest of the hyper-parameters, t, α_1, α_2 and α_3 are set to 10^{-3}, 0.5, 0.1 and 1 respectively. For all the experiments, the training batch size is fixed to $N_B = 256$.

3.4.2 Experimental Settings

3.4.2.1 CIFAR-10 [84]

This dataset consists of 60000 small-size images, subjected to 10 categories. We follow the setting in [113] to randomly select 100 images from each class as the test set, and use the rest 59000 images as the training set and retrieval gallery. K is set to 20 on this dataset by cross-validation.

3.4.2.2 SUN-397 [187]

A total number of 108754 images are in involved in this dataset with 397 exclusive class labels. For each class, 20 images are randomly selected to form the test set. The reset images are used as training and retrieval candidates. K is set to 300 on this dataset.

3.4.2.3 NUS-WIDE [22]

This is a multi-label dataset containing 269648 images. We use a subset of 195834 images from the 21 most frequent topics, from which 100 images for each topic are randomly picked for testing. K is set to 100 on this dataset.

3.4.3 Comparison with State-of-the-Art Methods

The performance of the proposed DVB model is evaluated by conducting image retrieval on the three datasets mentioned above. For experiments on CIFAR-10 [84] and SUN-397 [187], the retrieval candidates having the same label as the query image are marked as the ground-truth relevant data. Since NUS-WIDE [22] is a multi-label dataset, a relevant retrieval candidate is defined as sharing at least one label with the query image, which is a conventional setting in image hashing and retrieval. The code length m is chosen to be 16, 32 and 64.

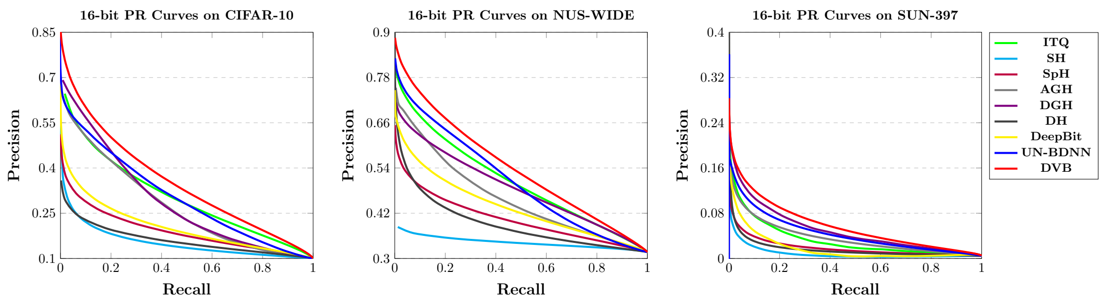
3.4.3.1 Baselines

Several benchmarked unsupervised hashing methods are involved in the experiments of this chapter, including ITQ [43], SH [178], SpH [54], LSH [21], SKLSH [141], Sparse Embedding and Least Variance Encoding (SELVE) [216], AGH [115] and DGH [113]. We employ these models as they are conventional baselines used in state-of-the-art unsupervised deep hashing articles [104, 101, 30] and these deep methods, i.e. DH [104], DeepBit [101] and UN-BDNN [30], are involved as well. To make a fair comparison between the shallow methods and the deep models, we utilize the VGG-16 [157] features as inputs for all baselines. As a result, the performance figures of the traditional hashing works reported here are slightly higher than those in their original papers but are still reasonable and illustrative. We directly use the source code of DeepBit [101] developed by the original authors for experiments, and implement DH [104] and UN-BDNN [30] on our own.

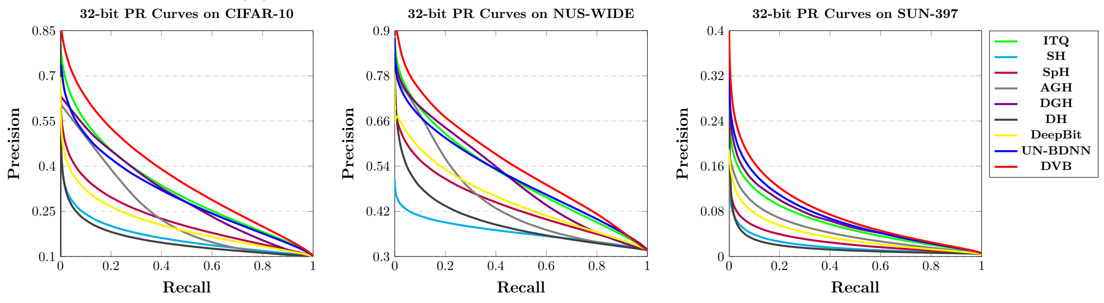
3.4.3.2 Quantitative Results with Deep Features

The image retrieval mean-Average Precision (mAP) [128] results are provided in Table 3.1, which gives a brief insight of binary encoding capability. A full definition of mAP is provided in Appendix A. In general, DVB outperforms all state-of-the-art shallow and deep unsupervised methods with evident margins in most cases. Particularly, the minimum mAP gaps yield 1.5%, 0.7% and 1.6% on the three datasets respectively between DVB and other methods. Note that the overall scores on the SUN-397 [187] are relatively low compared with those on the rest two datasets, as SUN-397 contains more categories. It is clear that some existing unsupervised deep hashing models [104, 101] are no longer leading the retrieval performance compared with the shallow ones with deep features. Although benefited from the compact encoding neural networks, these deep methods still struggle in handling unsupervised hashing. This is probably because the batch-wise SGD procedure only manages to preserve the in-batch data similarities and therefore skews the statistics of the whole training set, which is empirically compensated in DVB by introducing the latent variables $\mathbf{z}$. UN-BDNN [30] obtains most acceptable performance among the existing deep methods, while it involves a more sophisticated optimization procedure than DVB. The precision at top-5000 retrieved candidates (Precision@5000) [128] curves with all bits are plotted in Figure 3.5 to have a more comprehensive view on retrieval performance. This metric indicates the fraction of relevant candidates out of the first 5000 returned, of which the definition is also provided in Appendix A.

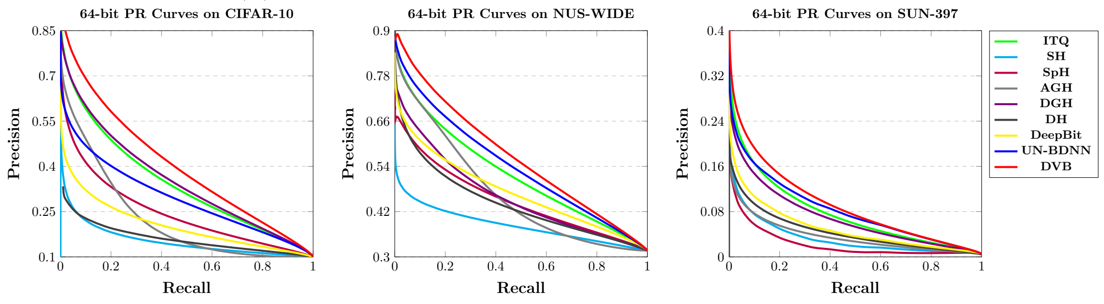
The Precision-Recall curves (P-R curves) [49, 149] for image retrieval are illustrated in Figure 3.6. The P-R curves represent the precision values corresponding to the recall values [149]. It can be clearly observed that DVB obtains a higher plot than the compared baselines. The plot suggests that DVB successfully keep the related candidates in top of the retrieved sequences, since it obtains higher precision scores at all recall values. The P-R cures agree the mAP performance comparison discussed above.



(a) 16-bit image retrieval Precision-Recall curves.



(b) 32-bit image retrieval Precision-Recall curves.



(c) 64-bit image retrieval Precision-Recall curves.

Figure 3.6: 16-bit (a), 32-bit (b) and 64-bit (c) image retrieval Precision-Recall curves (P-R curves) [49, 149] with VGG [157] features.

3.4.3.3 Quantitative Results with Hand-Crafted Features

In addition to the retrieval performances with deep features provided above, we also compare DVB with existing hashing algorithms using hand-crafted feature for a comprehensive view on encoding quality. Only the image retrieval performances with GIST features [134] on CIFAR-10 [84] are reported here to make the content concise. The proposed DVB model is slightly modified to fit the GIST features. PCA is no longer required in this case since GIST features are relatively short and easy to reconstruct. Therefore, k-means clustering [123] is performed directly on the input features in obtaining $\{\mathbf{c}\}$. As a result, the size of the latent space becomes exactly the same to the input feature length, while the rest part of DVB remains unchanged.

Table 3.2 shows the retrieval mAP and Precision@5000 scores of DVB and several state-of-the-art unsupervised hashing methods with GIST features on CIFAR-10 dataset. It is clear that DVB still outperforms the compared models except DeepBit [101]. The GIST feature [134] is based on orientation histograms, carrying rotation information. This can be fully utilized by the training objective of DeepBit [101], i.e. producing identical codes for one image with different rotation angles. As a result, DeepBit [101] outperforms DVB in this experiment. However, it is infeasible to require all types of descriptors carrying orientation information. On the other hand, DH [104] and UN-BDNN [30] just marginally reach the average performance level of the shallow methods, which agrees with their performance with deep features.

3.4.3.4 Efficiency

The training and encoding time of DVB with deep features are demonstrated in Table 3.3, where DH [104] and DeepBit [101] are included for comparison. All experiments are conducted on an Nvidia TitanX GPU. DVB requires less training time than the two listed deep models to reach the best retrieval performance, since it efficiently explores the intrinsic data structure. The test time of DVB is slightly longer than DH [104] and DeepBit [101]. This is because DVB involves a Monte Carlo sampling procedure with multiple sampled latent variables to encode test data.

Table 3.3: Comparison of training and encoding efficiency on CIFAR-10 [84] with some deep hashing methods.

Method	Bit	mAP	Training Time	Coding Time
DH [104]	16 bits	0.172	149 minutes	20.5ms
	32 bits	0.176	152 minutes	20.7ms
	64 bits	0.179	177 minutes	21.1ms
DeepBit [101]	16 bits	0.193	203 minutes	21.4ms
	32 bits	0.216	210 minutes	21.9ms
	64 bits	0.219	265 minutes	22.6ms
DVB	16 bits	0.347	127 minutes	28.9ms
	32 bits	0.365	127 minutes	29.4ms
	64 bits	0.392	127 minutes	31.7ms

Note that UN-BDNN [30] is not included in this experiment. An unstable training procedure is experienced. During optimization, the model unpredictably produces zeros as outputs for all data, and then it has to be retrained. Although the retrieval performance is recorded, its training time is not illustrative, since the model requires multiple attempts to reach the best performance. The cause of this phenomena is that the learning objective of UN-BDNN [30] does not involve bit decorrelation penalty. Therefore, it is possible that all output bits generate the same value.

3.4.4 Ablation Study

3.4.4.1 Baselines

We analysis the impact of different components of the learning objective in this subsection, by introducing the following variants of DVB for ablation study:

- **DVB-1.** The $l2$ loss on $\mu_\theta(\cdot)$ is omitted to form this baseline. Since the clustering centres are no longer required here, this baseline is able to be trained end-to-end together with the convolutional networks.

Table 3.4: Ablation study results (mAP) of DVB on CIFAR-10 [84] and NUS-WIDE [22] with some terms of the learning objective removed.

Method	**CIFAR-10**			**NUS-WIDE**		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
DVB-1 (Without $l2$ on $\mu_\theta(\cdot)$)	0.269	0.325	0.342	0.477	0.506	0.512
DVB-2 (Without $\mathcal{H}_2$)	0.286	0.344	0.349	0.483	0.489	0.530
DVB-3 (Without $\mathcal{H}_3$)	0.317	0.350	0.361	0.525	0.531	0.578
DVB-4 (Without $\mathcal{H}_2$ and $\mathcal{H}_3$)	0.275	0.293	0.303	0.466	0.490	0.487
DVB (Full)	**0.347**	**0.365**	**0.381**	**0.546**	**0.560**	**0.584**

- **DVB-2.** We build this baseline by removing the bit decorrelation term $\mathcal{H}_2$ in Equation (3.16).
- **DVB-3.** In this baseline the in-batch similarity regularization term $\mathcal{H}_3$ is omitted.
- **DVB-4.** This baseline combines the modification of DVB-2 and DVB-3.

The quantization penalty $\mathcal{H}_1$ is not involved in ablation study since it is usually compulsory in deep hashing.

3.4.4.2 Results and Analysis

The retrieval mAP results of the three baselines mentioned above are shown in Table 3.4. Ablation experiments are not conducted on SUN-397 [187] since the overall figures on this dataset are too low and the performance margins between different baselines can be narrow and hard to observe. We have experienced a significant mAP drop of 5% on average when omitting the $l2$ loss (DVB-1) on CIFAR-10 [84]. Although DVB-1 is trained end-to-end, it lacks the intrinsic structural information out of a single batch during training, which is compensated by regularizing $\mu_\theta(\cdot)$ using the pseudo data centres in this work. It also can be observed from the performances of DVB-2 and DVB-3 that $\mathcal{H}_2$ and $\mathcal{H}_3$ do have a positive impact on the final result of DVB. Interestingly, when simultaneously omitting $\mathcal{H}_2$ and $\mathcal{H}_3$ from the learning objective, the retrieval performance degrades remarkably. To

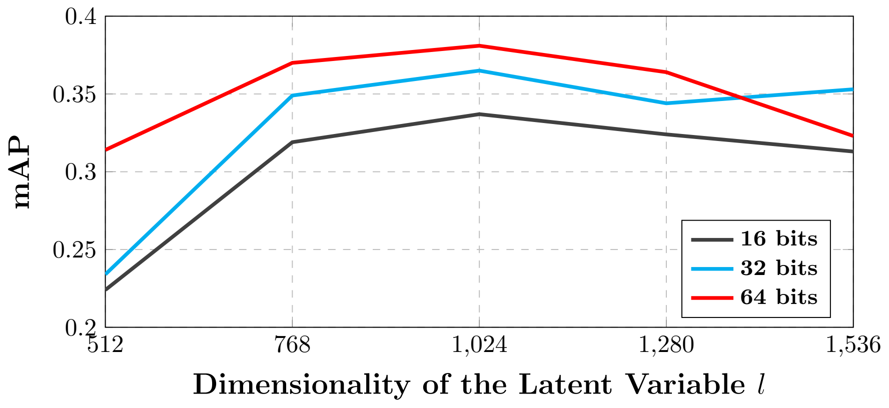


Figure 3.7: Retrieval performance on CIFAR-10 [84] *w.r.t.* different size of l.

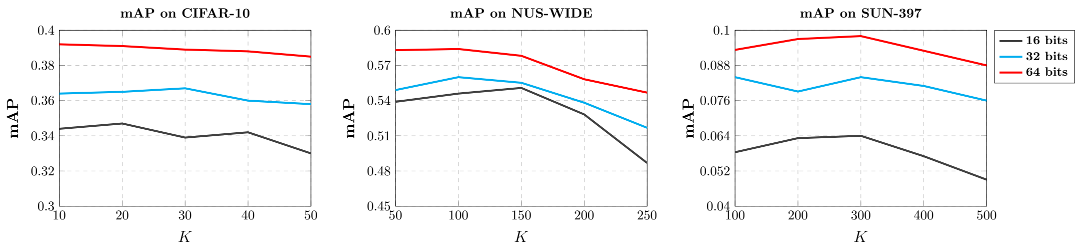


Figure 3.8: Retrieval performance on CIFAR-10 [84] (left), NUS-WIDE [22] (middle) and SUN-397 [187] (right) *w.r.t.* different settings of K, where the default values are 20, 100 and 300 respectively.

this end, it can be observed that the proposed model is successfully designed and all components are reasonably implemented.

3.4.5 Hyper-Parameters

The hyper-parameters dominant to the performance of DVB are analysed in this subsection. During our experiments, we figure out DVB tends to be not sensitive to t, α_1, α_2 and α_3. As long as they are established with a reasonable scale of values close to our default settings provided in Sec. 3.4.1, the model is still able to produce the aforementioned retrieval mAP scores with a performances difference less than 3% according to experiments. Therefore, we focus on the rest of the hyper-parameters, i.e. the number of clustering centres K and the dimensionality of latent space l.

3.4.5.1 Size of Latent Space

The retrieval performances of DVB on CIFAR-10 *w.r.t.* different sizes of l are shown in Figure 3.7. As l also determines the size of the pseudo data centres $\mathbf{c}$, a small value of l leads to lossy PCA based computation of results on the training of deep features for further clustering. Thus $\{\mathbf{c}\}$ can hardly provide informative semantics in regularizing the conditional network $p_\theta(\mathbf{z}|\mathbf{x})$. On the other hand, oversized l is neither appreciated since this hampers the reconstruction procedure of $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{b})$. 1024 becomes a reasonable setting for l in our experiments, as it is close the length of 95%-energy PCA results of VGG-16 features on the three datasets.

3.4.5.2 Number of Clustering Centres

Figure 3.8 illustrates the retrieval performances *w.r.t.* different values of K on the three datasets. It can be observed that the best-performing settings of K are generally close to the category numbers of the three datasets, while a slightly smaller value of K is still acceptable. Empirically, K reflects the number of semantic concepts hidden in the dataset, but it also influences the clustering quality to produce $\{\mathbf{c}\}$. Large values of K may lead to noisily computed $\{\mathbf{c}\}$ regarding the size of the training set. Consequently, the choice of K on SUN-397 is relatively smaller than the category size as this dataset consists of a large number of image classes.

3.4.6 Qualitative results

Qualitative analysis is also provided to empirically demonstrate the binary encoding performance of DVB. Some intuitive retrieval results on 32-bit CIFAR-10 are shown in Appendix B, which suggests DVB is able to provide relative candidates in top of the retrieval sequences. From Appendix B, it can be observed that the top-20 retrieved images are mainly subjected to the same topic as the query images. The t-distributed Stochastic Neighbour Embedding (t-SNE) [122] visualisation results on the test set of CIFAR-10 are illustrated in Figure 3.9. It can be observed that the produced codes are not widely scattered on the two-dimensional panel as

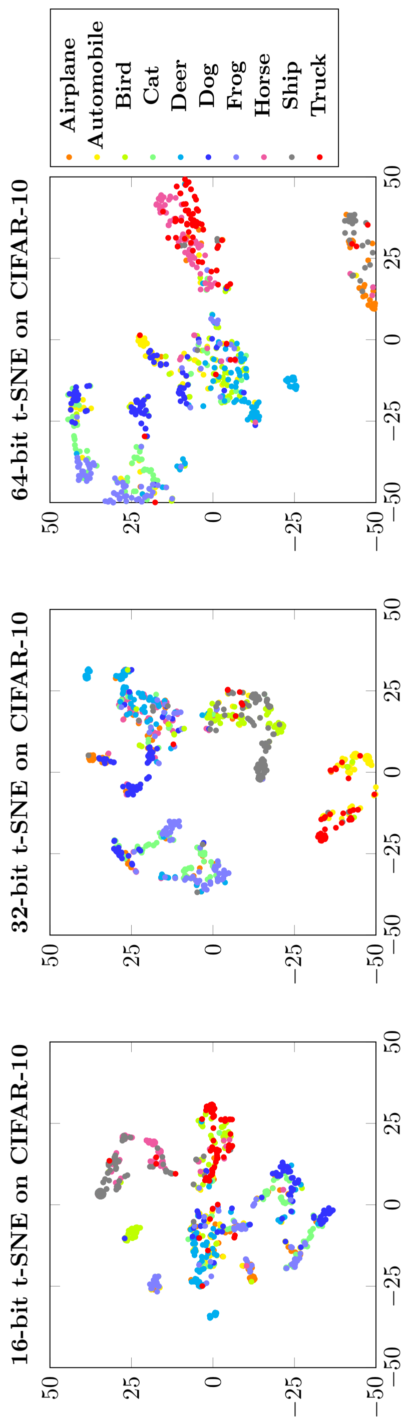


Figure 3.9: The t-SNE [122] visualization results of DVB on the CIFAR-10 [84] dataset.

no class information is provided during parameter training. However, most classes are clearly segregated, which means the produced binary codes are still compact and semantically informative to some extent.

3.5 Summary

In this chapter, a novel unsupervised deep hashing method DVB is proposed. The recent advances in deep variational Bayesian models have been leveraged to construct a generative model for binary coding. The latent variables in DVB approximate the pseudo data centres that each data point in the training set belongs to, by means of which DVB exploits the intrinsic structure of the dataset. By minimizing the gap between the constructed and reconstructed latent variables from data inputs and binary outputs respectively, the proposed model produces compact binary codes with no supervision. Experiments on three large-scale datasets suggest that DVB outperforms state-of-the-art unsupervised hashing methods with evident margins.

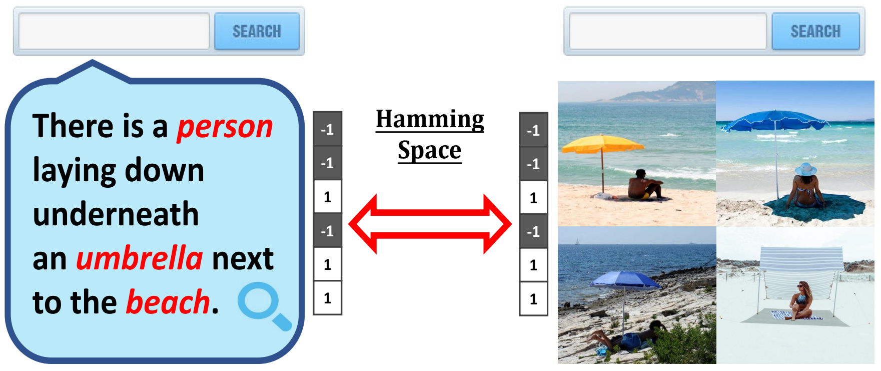
Chapter 4

Supervised Deep Hashing for Image-Sentence Cross Retrieval

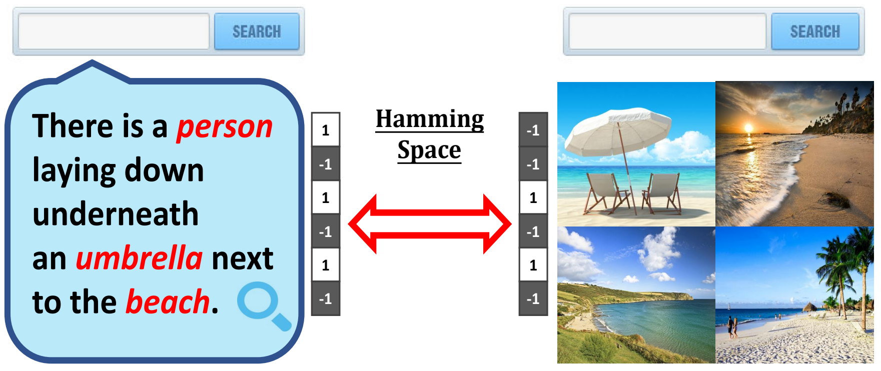
4.1 Introduction and Motivation

Learning the semantic relations between image and text modalities has aroused a lot of recent attention, becoming an overwhelming research topic in computer vision, natural language processing and machine learning. Dramatic progress has been achieved in visual question answering [124, 4, 40], caption generation [189, 31, 68] and textual-visual retrieval [191, 118, 73, 175, 38], with a variety of methods using real-valued shared space representations. Promising performances have been achieved by densely encoding image details and language semantics. Typically embedding different types of data with deep neural networks and trained with stochastic back-propagation, these methods achieve impressive fine-grained multi-modal mapping performance [118, 38, 73, 175, 67]. However, due to the high complexity of similarity computation, real-valued embeddings are usually infeasible for large-scale textual-visual retrieval.

Addressing this issue, cross-modal hashing schemes [28, 90, 205, 163, 103, 180, 11, 66, 62, 177, 204] have been proposed to encode heterogeneous data from a high-dimensional feature space into a shared low-dimensional Hamming space where an approximate nearest neighbor of a given query can be found efficiently. Traditionally, most existing research in cross-modal hashing only focus on image-tag mapping [28, 90, 205, 163, 7, 103, 180, 11, 66], where holistic image representations and semantic tags feed the shallow binary coding procedure.


SEARCH
SEARCH
There is a person laying down underneath an umbrella next to the beach.
-1 -1 1 -1 1 1
Hamming Space
-1 -1 1 -1 1 1

(a) TVDB (ours)


SEARCH
SEARCH
There is a person laying down underneath an umbrella next to the beach.
1 -1 1 -1 1 -1
Hamming Space
-1 -1 1 -1 1 -1

(b) Conventional Cross-Modal Hashing

Figure 4.1: The proposed model (a) aims at encoding all informative words of a sentence and all possible attractive regions in an image. Contrarily, conventional textual-visual binary encoding methods (b) only utilize simple representations of each modality and discard some information (*e.g.* the instances of *person* and *umbrella* are not well-encoded), resulting in low matching quality.

We argue that, first of all, it is insufficient to simply link images with tags, instead of real sentences, for multimedia searching problems nowadays. Image-tag mapping can be barely achieved or replaced by a multi-label image classification model, regarding text tags as labels. However, sentences usually enclose more knowledge and semantics than tags, which needs to be considered and modelled more precisely for cross-modal retrieval. Secondly, the detailed information of images is usually discarded by simply using holistic representations and features. Furthermore, the hashing functions for traditional methods are not usually compact. Most cross-modal hashing works mentioned above generally involve a linear projection as the encoding function, which is easy to be optimized but limited in the ability of encoding complex data into desired binary codes. Figure 4.1 gives an illustrative example. Traditional hashing methods is managed to link the query with retrieval candidates that share some semantic topics (*beach* in this case), but the detail information describing the *person under the umbrella* may be ignored. This is usually not desirable for cross-modal retrieval.

The recent deep cross-modal hashing methods [11, 37, 66, 127, 9] produce improved image-text retrieval performance. Utilizing deep neural networks as hashing functions, the non-linear cross-modal correlations between data points are captured effectively, which usually yields state-of-the-art retrieval results. However, similar to the traditional hashing methods, existing deep cross-modal hashing techniques but also suffer from several drawbacks. For instance, most of the deep models mentioned above still utilize coarse text representations, which is inappropriate for modeling long sentences and image data are likewise poorly modeled due to the lack of detailed regional information during encoding. Moreover, the network training efficiency can be further improved by more advanced code learning architectures.

Driven by the drawbacks of previous works, in this chapter, we consider a more challenging task to encode informative multi-modal data, i.e. semantic-rich images and descriptive sentences, into binary codes for supervised cross-modal search, termed as Textual-Visual Deep Binaries (TVDB). Particularly, the popular Region Proposal Network (RPN) [145] and Long Short-Term Memory (LSTM) units [203, 165] are introduced to formulate the image binary coding function, so that the regional semantic details in images can be well preserved from dominant to minor.

Meanwhile, the latest advances in text Convolutional Neural Network [74, 23, 57] is adopted to build the text binary encoding network, leveraging structural cues between the words in a sentence. The proposed deep architecture produces high-semantic-retentivity binary codes and achieves promising retrieval performance. The intuitive difference between the proposed method and the traditional ones are given in Figure 4.1. It can be seen that the proposed TVDB encodes as many details as possible from images and sentences, leading to more representative binary codes for matching.

In addition to the novel deep binary encoding networks of TVDB, a stochastic batch-wise code learning procedure is proposed, efficiently leveraging the label and similarity information as supervision. Inspired by Shen *et al.* [155], the binary codes in TVDB are discretely and alternately optimized during the batch-wise learning procedure. Batching data randomly and iteratively, the proposed training routine guarantees an effective learning objective convergence. TVDB is subjected to supervised cross-modal hashing as the its learning objective is built using the label information of training data. The contributions of this work can be summarized as follows:

- The TVDB model is proposed to effectively encode rich regional information of images as well as semantic dependencies and cues between words by exploiting two modal-specific deep binary embedding networks. In this way, the intrinsic semantic correlation between heterogeneous data can be quantitatively measured and captured, which leads to more satisfying cross-modal retrieval results (Figure 4.1).

- A novel stochastic batch-wise training strategy is adopted to optimize TVDB, in which reliable binary codes and the deep encoding functions are optimized in an alternating manner within every single batch.

- The evaluation results suggest the well-designed deep architecture to be crucial in image-sentence hashing and retrieval, and TVDB highly outperforms existing state-of-the-art binary coding methods in cross-modal retrieval on three semantically rich datasets.

The rest of this chapter is organized as follows. Several related works are introduced in Section 4.2. In Section 4.3, the architecture of the deep encoding networks of TVDB is demonstrated. The batch-wise alternating optimization procedure for code learning is discussed in Section 4.4. Extensive experimental results are provided and illustrated in Section 4.5. A short summary is given in Section 4.6.

4.2 Related Work

Several research areas are related to this work, including the deep learning algorithms, hashing techniques and various aspects related to vision and language.

4.2.1 Neural Networks for Image and Text

Several deep learning structures are involved in this chapter for the cross-modal retrieval task. Convolutional Neural Networks (CNNs) [95, 85, 157] have been successfully applied to many image-related areas. In addition to the classic classification task [147], CNNs are also widely used to extract image features. The hidden states from the top layers of a CNN contain multi-scale latent information [95], which is regarded as an efficient way to represent a simple image [72, 192, 126]. However, holistic image representations are not ideal for describing information-rich images. The multiple objects in image are not well-recognized and encoded. As a result, these objects cannot be directly related to the concepts carried by text data. Therefore, simply employing a CNN to encode image is suboptimal for image-sentence retrieval.

More recently, region-based convolutional architectures [42, 145] are proposed to leverage and recognize dominant image regions and components. RPNs [145] deterministically detect objects and instances in image and represent their positions and sizes using bounding boxes. The multi-anchor design for bounding box regression in RPN [145] provides a feasible way to recognize objects of different scales, which is desired for our task. In this chapter, we utilize this design as a part of our image encoder.

Recurrent Neural Networks (RNNs) are employed in text classification and sentiment prediction [159, 165, 136], suggesting a valid deep learning solution for text recognition. However, RNNs do not utilize structural information, *e.g.* the relations and dependencies between words, and is not suitable for learning text feature [23]. Alternatively, CNNs are also adopted in text understanding with competitive performance [74, 23, 57]. In general, CNNs employ a clearer design to leverage structural relations and information between the words in a sentence in which convolution is conducted along the word sequence. In this work, text-CNN is chosen for text binary encoding and it provides better performance in experiments.

4.2.2 Real-Valued Image-Text Embedding

The cutting-edge studies in vision and language achieve promising results in terms of visual question answering [124, 4, 40], caption generation [189, 31, 68], and real-valued cross-modal retrieval [164, 73, 175, 59, 69, 80, 72, 126, 121, 192]. The best-performing real-valued cross-modal retrieval models typically rely on densely annotated image-region and text pairs for embedding [69]. The utilization of deep neural networks, typically CNNs, enables these models to explore the heterogeneity between vision and language and thus builds a reliable shared representation space for data understanding and matching. For instance, in [72], images and sentences are firstly rendered to the corresponding neural networks for feature extraction. Then, the learning objective is constructed using the ranking loss, which requires a labelled ranking list as supervision for each image or sentence.

However, these methods are far from satisfactory for large-scale data retrieval due to the inefficient similarity computation of real-valued embeddings compared with binary representations. It is usually unnecessary to compare these methods with cross-modal hashing techniques because they are functionally different [111, 173, 103]. Learning to hash aims at efficient data search and quick matching [173], while real-valued embedding focuses on feature learning and do not consider search efficiency. Therefore, they are not involved in the experimental comparisons in this work.

4.2.3 Binary Image-Text Embedding

On the other hand, there exists several hashing methods [41, 178, 115, 54, 116, 43, 113, 21, 114, 87, 155] aiming at efficient retrieval. For textual-visual hashing, it has been a traditional and common solution to encode images and tags via shallow embedding functions with either unsupervised [90, 163, 28, 142, 215, 176, 169], pairwise based [7, 180, 62, 135, 177, 204, 117] or supervised [182, 13, 103, 205, 127] code learning methods. Among these traditional cross-modal binary encoding methods, Semantic-Preserving Hashing (SePH) [103] has become a successful proposal in reducing data heterogeneity by transforming semantic affinities into a probability distribution model. Its learning objective is based on the K-L divergence between the original feature distribution and code distribution, where the codes are obtained by two linear projection functions. However the shallow cross-modal hashing methods, including SePH [103], mainly suffers from the two following problems:

- The shallow encoding functions employed in these models, *e.g.* linear projections [103, 205, 127, 176] and eigenfunctions [215, 90], are not able to fully leverage the semantics carried by the input data features, resulting in less informative hash codes.
- The above-mentioned methods are designed for image-tag cross-modal retrieval, where language data are modelled using word count vectors as input features. This is not adequate to represent long sentences and word dependencies.

More recently, supervised deep hashing methods [14, 184, 217, 105, 30, 9] provide promising results in image recognition, which is also adopted in [11, 37, 66, 127, 9] for textual-visual retrieval.

Jiang *et al.* propose Deep Cross-Modal Hashing (DCMH) [66] for image-tag retrieval using a set of multi-layer neural networks which simply take deep holistic image features and word count vectors as input. DCMH can be regarded as an extension of conventional cross-modal hashing methods with a slight improvement regarding hash functions. As is discussed in Section 4.2.1, holistic features are suboptimal in representing information-rich images. The dominating image parts are not enhanced for encoding, leading to low-quality image codes. In addition, the text encoding network of DCMH [66] is not suitable for encoding sentences.

Collective Deep Quantization (CDQ) [9] combines data representation learning steps with quantization error controlling hash coding methods with deep neural networks. Nevertheless, it suffers from the same problem as DCMH [66] in network architecture. As a result, CDQ [9] does not produce satisfactory image-sentence retrieval performance.

Deep Visual Semantic Hashing (DVSH) [11] proposed by Cao *et al.* is a more feasible solution for image-sentence hashing as the sequential information of sentence data is better encoded by employing RNNs [203]. Although the RNN encoder preserves more semantic information than word count vectors used in [66, 9], it do not utilize the relations between words, and thus is not ideal for learning compact text features. Additionally, the structure of the image encoder in DVSH [11] is identical to [66, 9]. Therefore, the produced image codes do not efficiently represent the image content.

In the next section, we introduce the network structure of TVDB, which successfully handles the problems mentioned above.

4.3 Deep Encoding Networks for TVDB

This work addresses the problem of data retrieval between informative images and long sentences using deep binary codes. As shown in Figure 4.2, the proposed TVDB model is composed of two deep neural networks and a batch-wise code learning phase. The two deep neural networks play the role of binary encoding functions for images and sentences denoted as $f(\cdot)$ and $g(\cdot)$ respectively, of which the output is fixed to M bits in length. The batch-wise optimization allows using binary codes as supervision of $f(\cdot)$ and $g(\cdot)$ in a mini-batch during training.

Some preliminary notation is introduced here. We consider a multi-media data collection $\mathcal{O} = \{\mathbf{X}, \mathbf{Y}\}$ containing both image data $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}$ and sentence data $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{N}$, with N denoting the total number of data points of each modality in the dataset. The two deep binary encoding functions $f(\mathbf{X}; \mathit{\Theta})$ and $g(\mathbf{Y}; \mathit{\Phi})$ of TVDB are parameterized by $\mathit{\Theta}$ and $\mathit{\Phi}$. The sign function is applied to $f(\cdot)$ and

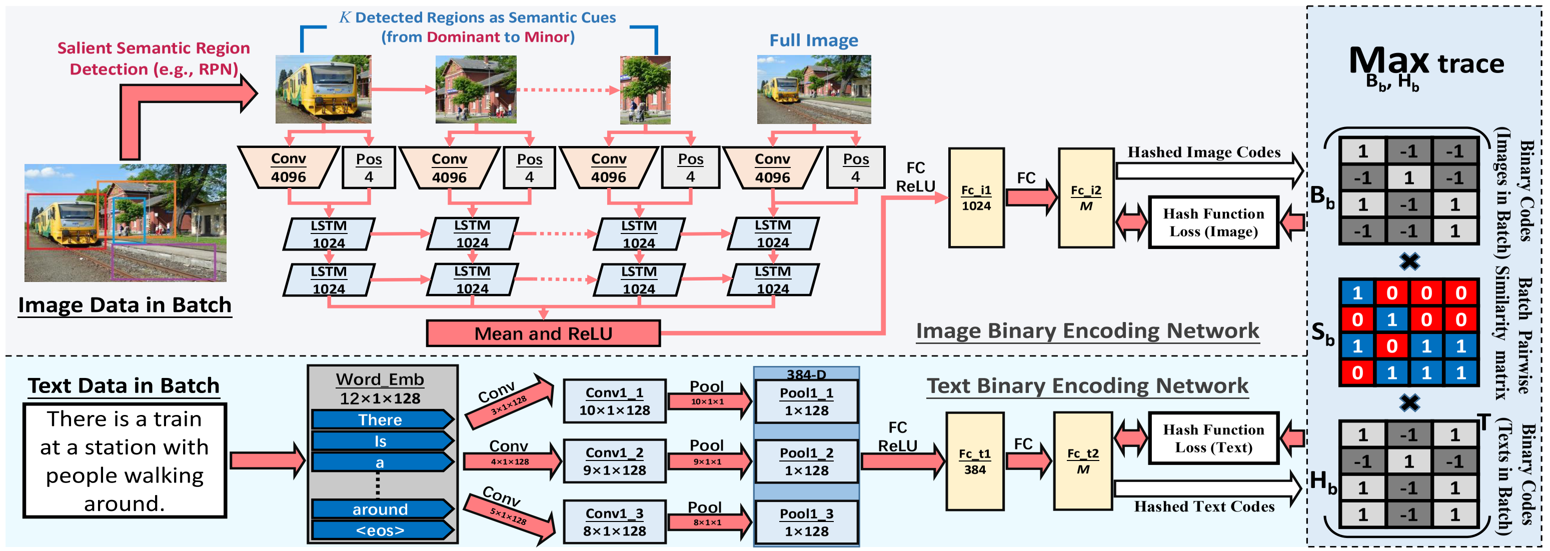


Figure 4.2: The network architecture of TVDB. RPN, CNN and LSTM are utilized to form the image encoding function $f(\cdot)$ (the upper network), encoding image regions from dominant to minor. The sentence encoding network $g(\cdot)$ is built with a text-CNN (the lower network). The rightmost component refers to a **batch-wise** coding procedure. Here the module `Pos` computes the region coordinates discussed in Subsection 4.3.1, while `Word_Emb` refers to the linear word embedding procedure introduced in Subsection 4.3.2. Note that during training iteration, a batch of images and descriptions are required to be simultaneously rendered to the network. During testing, a single image or sentence can be encoded using the corresponding network without acquiring a batch of data.

Table 4.1: Configurations of the binary coding networks.

Net	Layer	Kernel Size	Stride	Pad	Output Dim
Image Coding Net $f(\cdot)$	`LSTM_1`	-	-	-	$(K+1) \times 1024$
	`LSTM_2`	-	-	-	$(K+1) \times 1024$
	`Fc_i1`	1024	-	-	1024
	`Fc_i2`	1024	-	-	M
Text Coding Net $g(\cdot)$	`Word_Emb`	-	-	-	$12 \times 1 \times 128$
	`Conv1_1`	$3 \times 1 \times 128$	1	0	$10 \times 1 \times 128$
	`Pool1_1`	$10 \times 1 \times 1$	1	-	1×128
	`Conv1_2`	$4 \times 1 \times 128$	1	0	$9 \times 1 \times 128$
	`Pool1_2`	$9 \times 1 \times 1$	1	-	1×128
	`Conv1_3`	$5 \times 1 \times 128$	1	0	$8 \times 1 \times 128$
	`Pool1_3`	$8 \times 1 \times 1$	1	-	1×128
	`Fc_t1`	384	-	-	1024
	`Fc_t2`	1024	-	-	M

Note that in the text encoding network, as convolution is only performed along the word sequence, the second dimension of kernel size of the convolutional layers are always 1.

$g(\cdot)$ to produce binary representations:

$$\begin{aligned} \mathbf{B} &= \mathtt{sign}\left(f\left(\mathbf{X};\mathit{\Theta}\right)\right) \in \{-1,1\}^{M\times N}, \\ \mathbf{H} &= \mathtt{sign}\left(g\left(\mathbf{Y};\mathit{\Phi}\right)\right) \in \{-1,1\}^{M\times N}, \end{aligned} \tag{4.1}$$

where M refers to the target binary encoding length. In the following subsections, we introduce the setups of the deep binary encoding networks.

4.3.1 Image Binary Encoding Network

The architecture of the image encoding network $f(\cdot)$ is given in this subsection. As discussed previously, encoding the holistic image discards the informative patterns and produces poor coding quality, which is not desirable for our task. We consider a deep neural network architecture that embeds several salient regions of an image into a single binary vector to enrich the encoded semantic information. It is worthwhile to note that we are not directly linking every image region with a certain concept of a sentence as it is not feasible for cross-modal hashing problems and is contrary to the original intention of binary encoding in this work. Instead, regional semantic cues of images are leveraged here to improve the encoding quality.

4.3.1.1 Salient Semantic Region Selection

As shown in Figure 4.2, for each image in the mini-batch, TVDB firstly detects a number of regional proposals that possibly carry informative parts, *e.g.* recognizable or dominative objects in the image. Recent works in region-based CNN [42, 145] show great potential in detecting semantically meaningful areas of an image. We adopt the framework of the state-of-the-art RPN [145] as the proposal detection basis of TVDB. A total number of K semantic regions are sampled for further processing according to a simple heuristic attraction score a_k in descending order, that is:

$$a_k = (c_k + d_k)/2, \tag{4.2}$$

where $c_k \in (0,1)$ denotes the confidence score determined by the RPN and $d_k \in (0,1)$ refers to the normalized proportion of the k-th detected proposal in an image. We consider those regional proposals with high attraction score a_k semantically dominating to the whole image as they are usually the most recognizable image parts. This heuristic region selection solution highly fits the task of cross-modal binary encoding since it does not require any additional supervision or fine-grained region-sentence relations and provides good coding convergence.

4.3.1.2 Regional Representation and Augmentation

The selected image regions are fed into CNNs to extract vectorized representations. The benchmark CNN architecture AlexNet [85] is involved here, from which a feature representation of 4096-D is obtained. To make the most of structural information, the feature vector of each region is augmented with four additional digits indicating the normalized height, width, and center coordinates of the corresponding region bounding box, making the whole regional representation a 5000-D vector for each.

4.3.1.3 Recurrent Network for Encoding

For our task, it is desirable to use a method which capitalizes on information from the selected ordered regions so that dominating image parts contribute more to the final representation. It has been proven that human eyes sequentially browse image parts from dominant to minor [143]. Simulating this procedure, we sort the

K selected regional proposals according to their attraction scores in descending order and then the corresponding 5000-D representation for each proposal is sequentially fed into an RNN so that the dominant image parts can be well utilized. Additionally, the CNN feature of the holistic image is also appended to the end of the RNN input sequence, making a total of $K+1$ semantic regions for encoding. In particular, a two-level LSTM [56] is implemented as the RNN unit with 1024-D output length, following the popular structure described in [203]. Shown in Figure 4.2, the outputs of the LSTMs are averaged along the time sequence and appended with a `ReLU` [131] activation. Two fully-connected layers are applied to the top of the averaged LSTM outputs, with output dimension 1024 and M respectively. Thus the whole image can be encoded into an M-bit binary vector using the $\texttt{sign}(\cdot)$ function. We choose the `ReLU` [131] and identity function as the activations to the fully-connected layer for the convenience of code regression. The convolutional layers for images are built following AlexNet [85].

4.3.2 Sentence Binary Encoding Network

Although recurrent networks are widely adopted in textual-visual tasks [124, 31, 68, 146], it is still argued that RNNs such as LSTMs are not usually a superior choice for specific language tasks due to their non-structural designs [57, 23]. We aim at encoding the structural and contextual cues between the words in a sentence to ensure the produced binary codes have adequate information capacity. To this end, the text-CNN [74] is chosen as the text-side encoding network $g(\cdot)$, where each word in a descriptive sentence is firstly embedded into a word vector with a certain dimension and then convolution is performed along the word sequence.

We pre-process text data following the conventional manner where all sentences are appended with an *eos* token and padded or truncated to a certain length with all full stops removed. The sentence length after preprocessing is fixed to 12, which is about the mean length of text data in the datasets used in our experiments. Each word is embedded to a 128-D vector using a linear projection before being fed into the CNN. The text-CNN architecture in TVDB is similar to the one of Kim [74], with more fully-connected layers for coding. The full configuration of our text-CNN is given at the bottom of Figure 4.2. The `Word_Emb` layer in Figure 4.2

refers to the word embedding[1], the parameters of which are also involved in the back-propagation procedure. For the text convolution setups, the first and second digits of the kernel size denote the height and width of the convolutional kernels, with the third digit being the kernel number. Note that, as text convolution is only performed along the word sequence, the second dimension of kernel size of the convolutional layers are always 1. In this work, the number of kernels for all convolutional layers is set to 128, which follows the design in [74]. We also build two fully-connected layers here, where the first one `Fc_t1` takes inputs from all pooling layers followed by an `ReLU` activation, and for the second one, `Fc_t2`, the identity activation is applied.

4.4 Stochastic Batch-Wise Code Learning

The entire training procedure for $f(\cdot)$ and $g(\cdot)$ follows the mini-batch SGD since deep neural networks are utilized. We suggest the binary learning solution should provide reliable target codes as network supervision every time a mini-batch feeds.

4.4.1 Batch-wise alternating optimization

Let $\mathcal{O}_b = \{\mathbf{X}_b, \mathbf{Y}_b\}$ denote a mini-match stochastically taken from the data collection $\mathcal{O}$, where $\mathbf{X}_b = \{\mathbf{x}_i\}_{i=1}^{N_b}$ and $\mathbf{Y}_b = \{\mathbf{y}_i\}_{i=1}^{N_b}$ are image and sentence data in the mini-batch respectively. As the training process of TVDB is typically batch-based, we introduce an in-batch pair-wise similarity matrix $\mathbf{S}_b \in \{0,1\}^{N_b \times N_b}$ for target binary code learning, with N_b denoting the number of data points within a mini-batch for training. The entry $\mathbf{s}_{pq}$ of $\mathbf{S}_b$ is defined as follows:

$$\mathbf{s}_{pq} = \begin{cases} 1 & \mathbf{x}_p,\, \mathbf{y}_q \text{ share at least one semantic label,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

The aim is to learn a set of target binary codes $\mathbf{B}_b$ for images and $\mathbf{H}_b$ for sentences that best describe the in-batch samples. Fully utilizing the pairwise relations $\mathbf{S}_b$

[1]Word embedding is the technique that represents each word using a real-valued vector. It contains the semantic information of the encoded word. A typical example of word embedding is the skip-gram word vector [129].

as supervision, a trace-based prototypic learning objective for hashing is thus built as

$$\mathcal{L}\left(\mathbf{B}_b, \mathbf{H}_b, \mathbf{S}_b\right) = -\text{trace}\left(\mathbf{B}_b \mathbf{S}_b \mathbf{H}_b^{\top}\right). \tag{4.4}$$

Remark. The trace-like objectives are also adopted in several recent works such as Deep CCA [192]. We denote the main differences of TVDB from Deep CCA as follows: **1)** Deep CCA outputs real-valued representations for mapping while TVDB produces codes with binary constraints, which leads to different and more challenging ways to optimize the trace-like learning objectives; **2)** TVDB is a supervised hashing method, basing the similarity matrix $\mathbf{S}_b$ on label information, while Deep CCA computes the correlations according to the data representations.

A common learning procedure for cross-modal hashing can be formulated by solving the following problem:

$$\begin{gathered}\min_{\mathbf{B}_b, \mathbf{H}_b, \Theta, \Phi} \mathcal{L}(\mathbf{B}_b, \mathbf{H}_b, \mathbf{S}_b), \\ \text{s.t. } \mathbf{B}_b = \mathtt{sign}\left(f\left(\mathbf{X}_b; \Theta\right)\right), \mathbf{H}_b = \mathtt{sign}\left(g\left(\mathbf{Y}_b; \Phi\right)\right).\end{gathered} \tag{4.5}$$

Relaxing the binary constraints to be continuous, i.e. $\mathbf{B}_b = f\left(\mathbf{X}_b; \Theta\right), \mathbf{H}_b = g\left(\mathbf{X}_b; \Phi\right)$, results in a slow and difficult optimization process. Inspired by Shen *et al.*[155], we reformulate the problem of (4.5) by keeping the binary constraints and regarding $\mathbf{B}_b$ and $\mathbf{H}_b$ as auxiliary variables,

$$\begin{gathered}\min_{\mathbf{B}_b, \mathbf{H}_b, \Theta, \Phi} \mathcal{L}\left(\mathbf{B}_b, \mathbf{H}_b, \mathbf{S}_b\right) + \eta\Big(\left\|\mathbf{B}_b - f\left(\mathbf{X}_b; \Theta\right)\right\|_{\mathrm{F}}^2 \\ + \left\|\mathbf{H}_b - g\left(\mathbf{Y}_b; \Phi\right)\right\|_{\mathrm{F}}^2\Big), \\ \text{s.t. } \mathbf{B}_b \in \{-1, 1\}^{M \times N_b}, \mathbf{H}_b \in \{-1, 1\}^{M \times N_b},\end{gathered} \tag{4.6}$$

where η is a penalty hyper parameter and $\|\cdot\|_{\mathrm{F}}$ refers to the Frobenius norm. The two Frobenius norms here depict the quantization error between the binary codes $\mathbf{B}_b$, $\mathbf{H}_b$ and the binary coding function outputs $f\left(\cdot\right)$, $g\left(\cdot\right)$. It has been proven in [155] that with a sufficiently large value of η, Equation (4.6) becomes a close approximation to Equation (4.5), in which slight disparities between $\mathbf{B}_b$ and $f\left(\mathbf{X}_b; \Theta\right)$ or $\mathbf{H}_b$ and $g\left(\mathbf{Y}_b; \Phi\right)$ are tolerant to our binary learning problem.

Therefore, the comprehensive learning objective of TVDB is formulated. We provide optimization schemes below. It is observed that Equation (4.6) is a non-convex NP-hard problem due to the binary constraints. To better access it, an

Algorithm 2: The Training Process of TVDB

```
Input: Image-sentence dataset $\mathcal{O} = \{\mathbf{X}, \mathbf{Y}\}$, Max
       training iteration $T$
Output: Hash function parameters $\mathit{\Theta}$ and $\mathit{\Phi}$
Randomly initialize $\mathbf{B}$, $\mathbf{H} \in \{-1, 1\}^{M \times N}$
repeat
    Get a stochastic mini-batch $\mathcal{O}_b$ from $\mathcal{O}$
    Get $\mathbf{B}_b$, $\mathbf{H}_b$ from $\mathbf{B}$, $\mathbf{H}$ with respective indices
    Build $\mathbf{S}_b$ according to data relations and labels
    Update $\mathbf{B}_b \leftarrow$ Equation (4.8)
    Update $\mathbf{H}_b \leftarrow$ Equation (4.9)
    $(\mathrm{loss}_f, \mathrm{loss}_g) \leftarrow$ Equation (4.10)
    Update
      $(\mathit{\Theta}, \mathit{\Phi})^{new} \leftarrow (\mathit{\Theta}, \mathit{\Phi}) - \mathbf{\Gamma}\left(\nabla_{\mathit{\Theta}} \mathrm{loss}_f, \nabla_{\mathit{\Phi}} \mathrm{loss}_g\right)$
until convergence or max training iter $T$ is reached;
```

alternating solution based on coordinate descent is adopted to sequentially optimize $\mathbf{B}_b$, $\mathbf{H}_b$, $\mathit{\Theta}$ and $\mathit{\Phi}$ in every single batch as follows.

Updating $\mathbf{B}_b$. By fixing $\mathbf{H}_b$, $\mathit{\Theta}$ and $\mathit{\Phi}$, the subproblem of (4.6) *w.r.t.* $\mathbf{B}_b$ can be written as

$$\begin{aligned} &\min_{\mathbf{B}_b} \ \eta \|\mathbf{B}_b - f(\mathbf{X}_b; \mathit{\Theta})\|_{\mathrm{F}}^2 - \mathrm{trace}\left(\mathbf{B}_b \mathbf{S}_b \mathbf{H}_b^\top\right), \\ &\text{s.t.} \ \mathbf{B}_b \in \{-1, 1\}^{M \times N_b}, \end{aligned} \tag{4.7}$$

to which the closed-form optimal solution becomes

$$\mathbf{B}_b = \mathtt{sign}\left(2\eta f(\mathbf{X}_b; \mathit{\Theta}) + \mathbf{H}_b \mathbf{S}_b^\top\right). \tag{4.8}$$

Updating $\mathbf{H}_b$. Similar to $\mathbf{B}_b$, the solution to the subproblem of (4.6) *w.r.t.* $\mathbf{H}_b$ is

$$\mathbf{H}_b = \mathtt{sign}\left(2\eta g(\mathbf{Y}_b; \mathit{\Phi}) + \mathbf{B}_b \mathbf{S}_b\right). \tag{4.9}$$

Updating $\mathit{\Theta}$ and $\mathit{\Phi}$. The subproblem of (4.6) *w.r.t.* $\mathit{\Theta}$ and *w.r.t.* $\mathit{\Phi}$ can be respectively written as

$$\begin{aligned} &\min_{\mathit{\Theta}} \ \mathrm{loss}_f := \eta \|\mathbf{B}_b - f(\mathbf{X}_b; \mathit{\Theta})\|_{\mathrm{F}}^2, \\ &\min_{\mathit{\Phi}} \ \mathrm{loss}_g := \eta \|\mathbf{H}_b - g(\mathbf{Y}_b; \mathit{\Phi})\|_{\mathrm{F}}^2, \end{aligned} \tag{4.10}$$

when all other variables are fixed. These two subproblems are typically in the form of the $l2$ norm which are differentiable problems and thus $\varTheta$ or $\varPhi$ can be optimized in the framework of SGD using back-propagation. Here the updated auxiliary binary codes $\mathbf{B}_b$ and $\mathbf{H}_b$ act as the supervisions to the binary encoding networks $f(\cdot)$ and $g(\cdot)$.

4.4.2 Stochastic-batched training procedure.

In the above subsection, we have presented the binary code learning algorithm for each mini-batch of TVDB. However, how to apply the batch-wise learning objective to the whole dataset has not yet been discussed. In this subsection, the overall training procedure for mini-batch SGD is introduced. Note that simply keeping data in every mini-batch unaltered in each training epoch usually results in poorly-learned hash functions. This is because the cross-modal in-batch data are not able to interact with the data outside of the batch and thus the batch-wise similarity $\mathbf{S}_b$ skews the statistics of the whole dataset. To be more precise, we consider a data batching scheme to build every $\mathcal{O}_b$ and $\mathbf{S}_b$ that well explores the cross-modal semantic relationships across the entire dataset. To this end, a stochastic batching routine is designed. Each data mini-batch is randomly formed before being input into the training procedure. Therefore, $\mathbf{S}_b$ varies across each batch, which ensures the in-batch data diversity.

Combining the stochastic batching method with the alternating parameter updating schemes, the whole training procedure of TVDB is illustrated in Algorithm 2. The operator $\mathbf{\Gamma}(\cdot)$ in Algorithm 2 indicates the adaptive gradient scaler used for SGD, which is the Adam optimizer [75] in this work. Unlike some existing deep hashing methods [66] which update the target binary codes when a whole epoch is finished, TVDB updates $\mathbf{B}_b$ and $\mathbf{H}_b$ instantly when a mini-batch arrives. This training routine proves to achieve fast convergence for effectively learning the encoding networks $f(\cdot)$ and $g(\cdot)$ as shown in Figure 4.6. The code learning procedure of TVDB differs from those of recent deep hashing methods. For instance, DCMH [66] and CDQ [9] update codes only after each epoch of training.

4.4.3 Out-of-Sample Extension

Once the TVDB model is trained, given an image query $\mathbf{x}_q$ for example, we compute its binary code by rendering it to the respective encoding network

$$\mathbf{b}_q = \mathtt{sign}\left(f\left(\mathbf{x}_q; \mathit{\Theta}\right)\right). \tag{4.11}$$

For the retrieval database, the unified binary codes from each sentence is obtained via

$$\mathbf{H} = \mathtt{sign}\left(g\left(\mathbf{Y}; \mathit{\Phi}\right)\right). \tag{4.12}$$

A sentence query can be processed in the similar manner.

4.5 Experiments

4.5.1 Implementation Details

Experiments of TVDB on cross-modal retrieval are performed on three semantically fruitful sentence-vision datasets: Microsoft COCO [102], IAPR TC-12 [45] and INRIA Web Queries [83]. For implementation details, we utilize the RPN [145] for image proposal detection to pick $K = 20$ informative regions for further LSTM-based encoding, and the value of η in problem (4.6) is set to 10^{-4} via cross-validation. For the image-side CNNs, AlexNet [85] without its `fc_8` layer is adopted with the pre-trained parameters from ImageNet classification [147]. The TVDB framework is implemented using Tensorflow [1].

The evaluation results are reported according to the following themes:

- **Performance Comparison with Existing Methods.** We compare the proposed TVDB algorithm with some state-of-the-art image-text hashing methods to demonstrate the overall high-quality retrieval performance of it.

- **Deep encoding network ablation study.** To prove that the deep encoding networks in this chapter are designed reasonably, a set of variants of TVDB are built as baselines by modifying the deep encoding networks with some other architectures.

Table 4.2: Image-query-sentence mAP results of the proposed model compared with existing methods on the three datasets.

Task	**Method**	**Binary Code**	**Microsoft COCO**				**IAPR TC-12**				**INRIA Web Queries**			
			16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
Image Query Sentence	CVFL [188]	✗		0.488 (4096-D)				0.499 (4096-D)				0.137 (4096-D)		
	PLSR [179]	✗		0.528 (4096-D)				0.485 (4096-D)				0.258 (4096-D)		
	CCA [167]	✗	0.544	0.536	0.508	0.469	0.468	0.434	0.406	0.389	0.204	0.217	0.189	0.152
	CMFH [28]	✓	0.488	0.506	0.508	0.383	0.442	0.437	0.361	0.362	0.197	0.236	0.245	0.238
	CVH [90]	✓	0.489	0.474	0.436	0.402	0.422	0.401	0.384	0.374	0.204	0.200	0.187	0.163
	SCM [205]	✓	0.529	0.563	0.587	0.601	0.486	0.505	0.515	0.522	0.115	0.215	0.271	0.308
	IMH [163]	✓	0.615	0.650	0.657	0.677	0.463	0.490	0.510	0.521	0.233	0.250	0.277	0.295
	QCH [180]	✓	0.572	0.595	0.613	0.634	0.526	0.555	0.579	0.605	-	-	-	-
	CMSSH [7]	✓	0.405	0.489	0.441	0.448	0.345	0.337	0.348	0.374	0.151	0.162	0.155	0.151
	SePH [103]	✓	0.581	0.613	0.625	0.634	0.507	0.513	0.515	0.530	0.126	0.137	0.134	0.142
	CorrAE [37]	✓	0.550	0.556	0.570	0.580	0.495	0.525	0.558	0.589	-	-	-	-
	CM-NN [127]	✓	0.556	0.560	0.585	0.594	0.516	0.542	0.577	0.600	-	-	-	-
	DNH-C [92]	✓	0.535	0.556	0.569	0.582	0.480	0.509	0.526	0.535	-	-	-	-
	DCMH [66]	✓	0.562	0.597	0.609	0.646	0.443	0.491	0.559	0.556	0.241	0.268	0.301	0.384
	CDQ [9]	✓	0.559	0.590	0.623	0.672	0.497	0.508	0.576	0.593	0.249	0.312	0.338	0.394
	DVSH [11]	✓	0.587	0.713	0.739	0.755	0.570	0.632	0.696	0.724	-	-	-	-
	TVDB	✓	**0.702**	**0.781**	**0.797**	**0.818**	**0.629**	**0.697**	**0.731**	**0.772**	**0.368**	**0.405**	**0.419**	**0.446**

Table 4.3: Sentence-query-image mAP results of the proposed model compared with existing methods on the three datasets.

Task	**Method**	**Binary Code**	**Microsoft COCO**				**IAPR TC-12**				**INRIA Web Queries**			
			16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
	CVFL [188]	✗		0.561 (4096-D)				0.495 (4096-D)				0.291 (4096-D)		
	PLSR [179]	✗		0.538 (4096-D)				0.492 (4096-D)				0.257 (4096-D)		
	CCA [167]	✗	0.545	0.542	0.513	0.468	0.471	0.438	0.413	0.395	0.212	0.228	0.188	0.150
	CMFH [28]	✓	0.574	0.507	0.510	0.472	0.447	0.445	0.438	0.365	0.178	0.231	0.248	0.131
	CVH [90]	✓	0.486	0.470	0.434	0.401	0.424	0.403	0.386	0.376	0.204	0.200	0.179	0.164
	SCM [205]	✓	0.523	0.544	0.569	0.585	0.495	0.514	0.523	0.529	0.123	0.226	0.297	0.349
	IMH [163]	✓	0.610	0.679	0.728	0.740	0.516	0.526	0.534	0.527	0.251	0.275	0.306	0.284
Sentence	QCH [180]	✓	0.574	0.606	0.638	0.667	0.500	0.536	0.565	0.589	-	-	-	-
Query	CMSSH [7]	✓	0.375	0.384	0.340	0.360	0.363	0.377	0.365	0.348	0.154	0.153	0.156	0.147
Image	SePH [103]	✓	0.613	0.649	0.672	0.693	0.471	0.480	0.481	0.495	0.226	0.256	0.291	0.319
	CorrAE [37]	✓	0.559	0.581	0.611	0.626	0.498	0.520	0.533	0.550	-	-	-	-
	CM-NN [127]	✓	0.579	0.598	0.620	0.645	0.512	0.539	0.549	0.565	-	-	-	-
	DNH-C [92]	✓	0.525	0.559	0.590	0.634	0.469	0.484	0.491	0.505	-	-	-	-
	DCMH [66]	✓	0.595	0.601	0.633	0.658	0.486	0.487	0.499	0.541	0.227	0.305	0.322	0.380
	CDQ [9]	✓	0.551	0.584	0.615	0.646	0.504	0.499	0.543	0.547	0.233	0.295	0.320	0.371
	DVSH [11]	✓	0.591	0.737	0.758	0.767	0.604	0.640	0.680	0.675	-	-	-	-
	TVDB	✓	**0.713**	**0.779**	**0.787**	**0.810**	**0.674**	**0.678**	**0.704**	**0.721**	**0.353**	**0.462**	**0.464**	**0.470**

- **Training Efficiency.** The training efficiency and convergence of the proposed TVDB model are illustrated. Several baselines are built to show the feasibility of the designed optimization routine. Some existing deep cross-modal hashing methods are also involved for training time comparison.
- **Intuitive Results and Failure Cases.** Qualitative sentence-to-image retrieval results are given and discussed compared with several existing cross-modal benchmarks.
- **Hyper-Parameters** The settings of hyper-parameters η and K are also analyzed to produce the reported results.

4.5.2 Experimental settings

The experiments of sentence-vision retrieval are taken on three multimedia datasets. Following the conventional textual-visual retrieval measures [103, 11], the relevant instances for a query are defined by sharing at least one label.

4.5.2.1 Microsoft COCO [102]

The COCO dataset contains a training image set of $80,000$ samples with about $40,000$ validation images. Each image is assigned five sentence descriptions and labeled with 80 semantic topics. To be consistent with [11], we randomly select 5,000 images from the validation set and thus the retrieval gallery becomes around 85,000 images, from which we explicitly take 5,000 pairs as the query set and 50,000 images for training.

4.5.2.2 IAPR TC-12 [45]

It consists of 20,000 images. Each image is provided with 1.7 descriptive sentences on average. In addition, category annotations are given on all images with 275 concepts. Following the setting in [11], we use 18,000 image-sentence pairs that belong to the most frequent 22 topics as the retrieval gallery, from which we take 2,200 pairs as the query data and 5,000 as the training set.

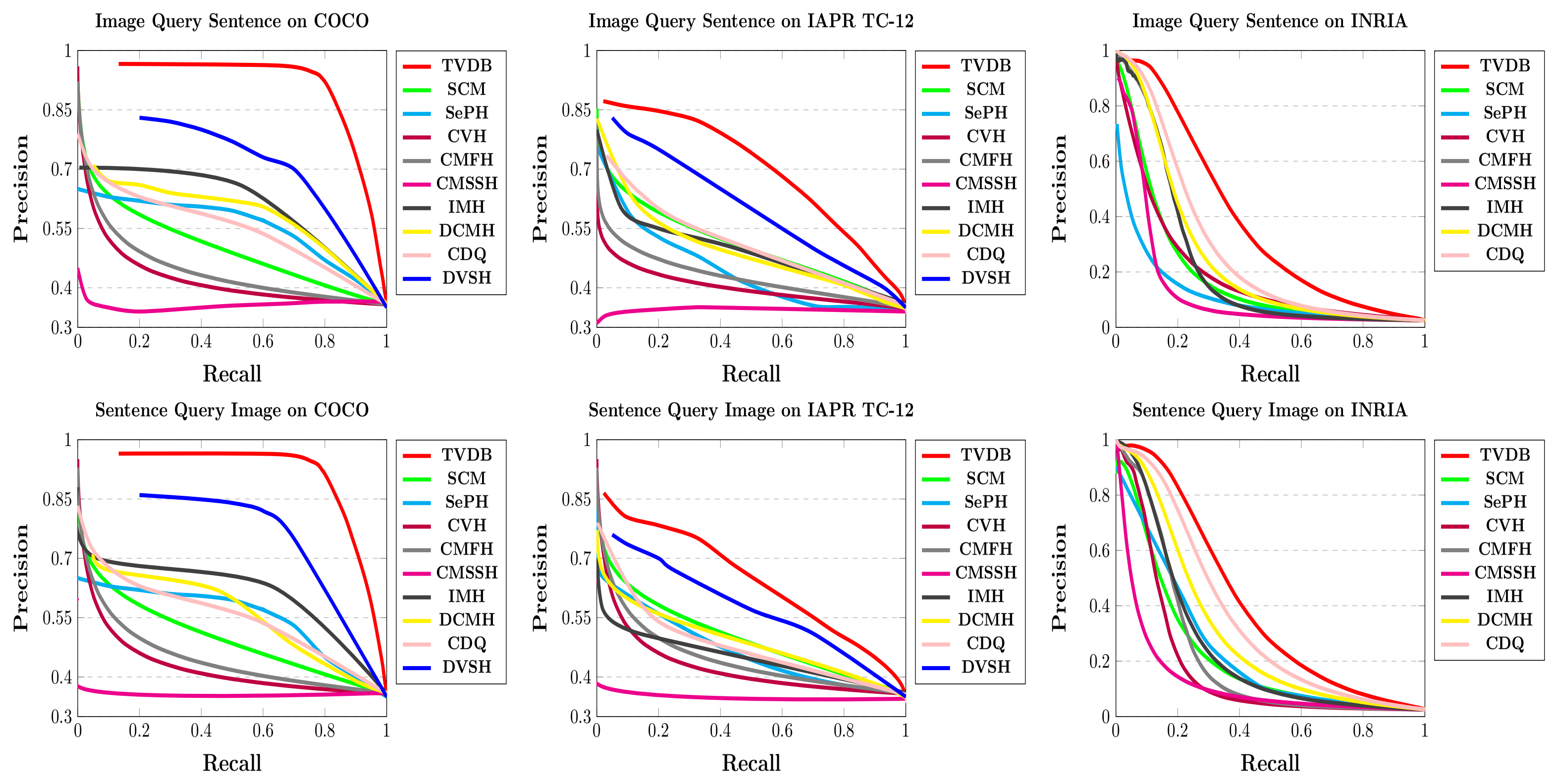


Figure 4.3: Cross-modal retrieval P-R curves of TVDB and some existing methods with code length $M = 32$.

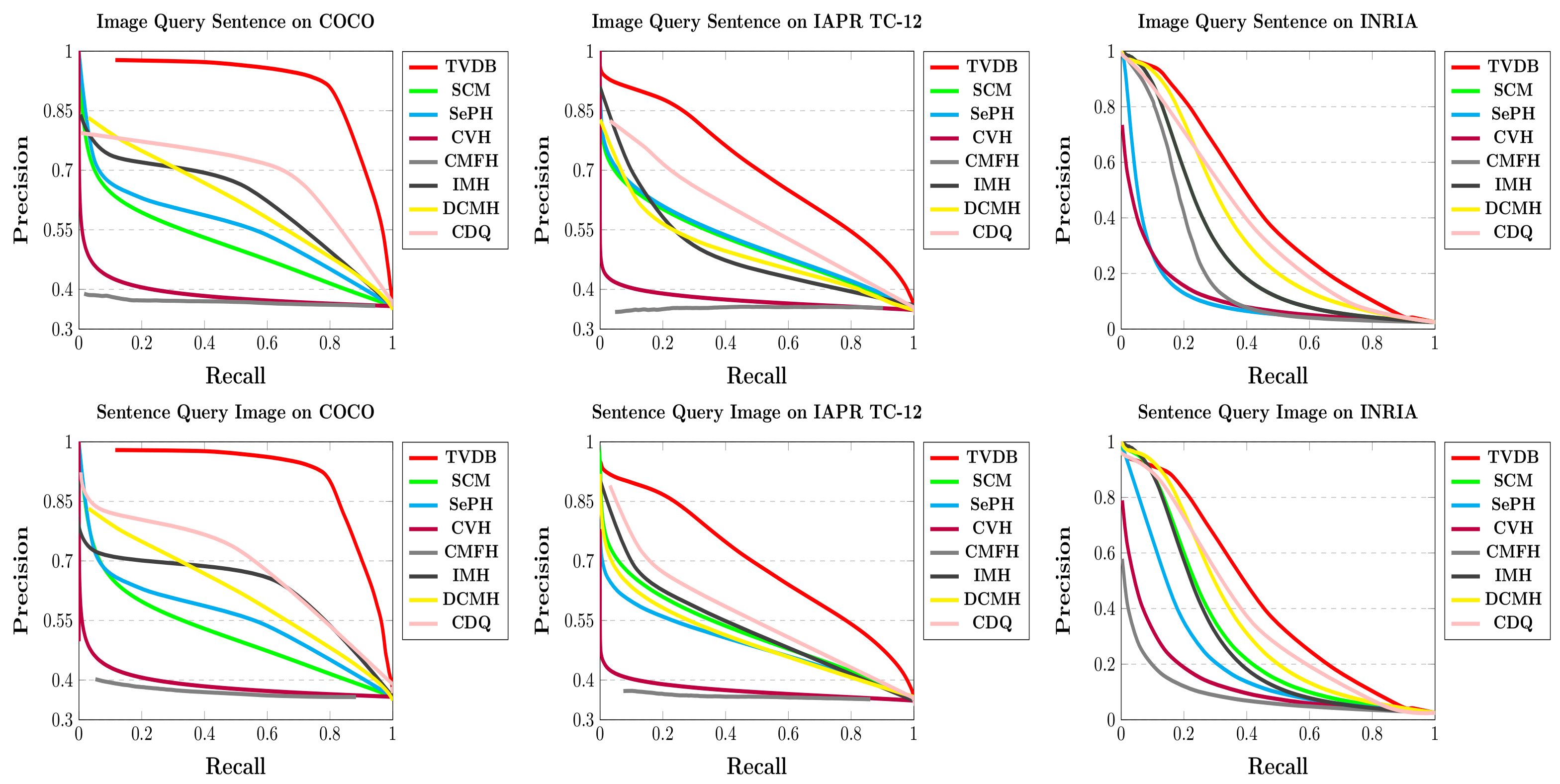


Figure 4.4: Cross-modal retrieval Precision-Recall curves of TVDB and some existing methods with code length $M = 128$. Note that for those methods of which the codes are not available and the Precision-Recall curves on 128 bits have never been reported are not listed here.

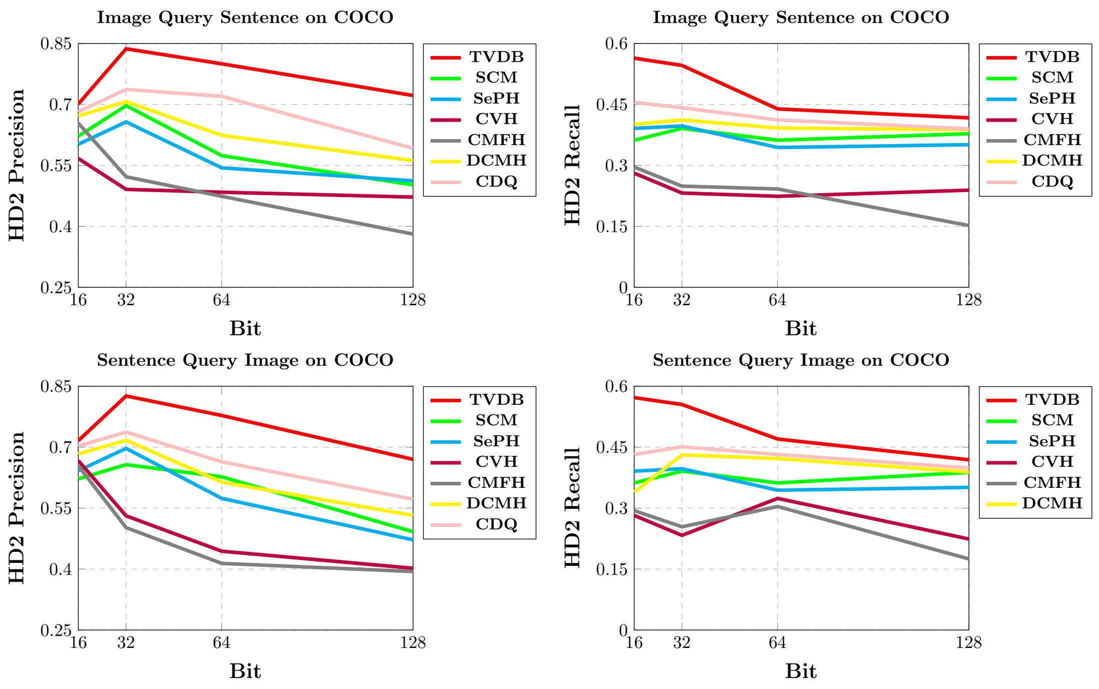


Figure 4.5: HD2 precision and recall of TVDB and some baselines on Microsoft COCO [102]

4.5.2.3 INRIA Web Queries [83]

This dataset contains about 70,000 images categorized into 353 conceptual labels. Sentence descriptions are provided to most of the images. We select images belonging to the 100 most frequent concepts, making the gallery 25,015 image-sentence pairs. For the query set, 10 images and sentences are randomly selected from each category and the rest are used as training data.

4.5.3 Comparison with Existing Methods

The quantitative retrieval performance of TVDB is analyzed and compared using mean-Average Precision (mAP) [128], precision at top-200 retrieved candidates (Precision@200), precision and recall of Hamming Distance with radius 2 (HD2). The detailed definitions of these metrics are elaborated in Appendix A. The Precision-Recall curves (P-R curves) [49, 149] are also given.

Table 4.4: Image-sentence cross-modal retrieval Precision@200 Performance.

Task	**Method**	**Microsoft COCO**				**IAPR TC-12**				**INRIA Web Queries**			
		16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
Image Query Sentence	CVFL [188]		0.519 (4096-D)				0.653 (4096-D)				0.130 (4096-D)		
	PLSR [179]		0.693 (4096-D)				0.596 (4096-D)				0.226 (4096-D)		
	CCA [167]	0.707	0.768	0.787	0.756	0.623	0.579	0.524	0.479	0.142	0.150	0.137	0.118
	CMFH [28]	0.683	0.733	0.750	0.491	0.557	0.555	0.384	0.385	0.193	0.215	0.221	0.214
	CVH [90]	0.695	0.745	0.730	0.651	0.546	0.504	0.463	0.431	0.204	0.200	0.187	0.163
	SCM [205]	0.647	0.727	0.770	0.792	0.608	0.648	0.667	0.679	0.127	0.157	0.181	0.201
	IMH [163]	0.601	0.661	0.667	0.691	0.493	0.531	0.565	0.584	0.202	0.239	0.256	0.279
	CMSSH [7]	0.361	0.445	0.413	0.429	0.335	0.348	0.362	0.396	0.143	0.177	0.169	0.166
	SePH [103]	0.622	0.623	0.625	0.634	0.469	0.478	0.458	0.459	0.125	0.128	0.116	0.152
	DCMH [66]	0.682	0.714	0.738	0.765	0.535	0.579	0.652	0.674	0.213	0.218	0.253	0.289
	CDQ [9]	0.714	0.745	0.786	0.821	0.630	0.671	0.705	0.746	0.224	0.250	0.274	0.326
	TVDB	**0.737**	**0.865**	**0.939**	**0.954**	**0.772**	**0.783**	**0.844**	**0.867**	**0.249**	**0.263**	**0.302**	**0.334**
Sentence Query Image	CVFL [188]		0.804 (4096-D)				0.622 (4096-D)				0.233 (4096-D)		
	PLSR [179]		0.727 (4096-D)				0.607 (4096-D)				0.249 (4096-D)		
	CCA [167]	0.718	0.831	0.876	0.836	0.599	0.566	0.524	0.483	0.139	0.145	0.163	0.119
	CMFH [28]	0.671	0.756	0.789	0.436	0.551	0.555	0.548	0.376	0.152	0.205	0.209	0.126
	CVH [90]	0.692	0.753	0.720	0.606	0.537	0.499	0.461	0.430	0.197	0.180	0.156	0.141
	SCM [205]	0.656	0.705	0.784	0.820	0.605	0.631	0.639	0.669	0.119	0.174	0.212	0.266
	IMH [163]	0.661	0.725	0.769	0.773	0.504	0.525	0.548	0.539	0.228	0.244	0.293	0.267
	CMSSH [7]	0.375	0.384	0.340	0.360	0.363	0.377	0.365	0.348	0.112	0.129	0.134	0.113
	SePH [103]	0.637	0.699	0.725	0.770	0.452	0.469	0.484	0.489	0.195	0.207	0.228	0.244
	DCMH [66]	0.705	0.731	0.770	0.839	0.557	0.563	0.582	0.625	0.194	0.238	0.284	0.306
	CDQ [9]	0.711	0.739	0.795	0.816	0.608	0.622	0.659	0.721	0.203	0.247	0.299	0.321
	TVDB	**0.809**	**0.889**	**0.891**	**0.903**	**0.772**	**0.784**	**0.881**	**0.888**	**0.261**	**0.276**	**0.352**	**0.377**

4.5.3.1 Baselines

Several baselines of traditional cross-modal hashing methods are adopted for comparison, including CMFH [28], CVH [90], SCM[205], IMH [163], QCH [180], CMSSH [7] and SePH [103], while Cross-View Feature Learning (CVFL) [188], Canonical Correlation Analysis (CCA) [167] and Partial Least Square Regression (PLSR) [179] are also involved as real-valued methods. The neural-network-based cross-modal hashing methods, i.e. Correspondence Auto-Encoder (CorrAE) [37], Cross-Modal Neural Network (CM-NN) [127], Deep Neural Hashing (DNH) [92], DCMH [66], CDQ [9] and DVSH [11], are also considered here. These models are selected in this chapter as they are widely regarded as conventional baselines in cross-modal hashing and they attain state-of-the-art performance in data retrieval.

Since the codes of some works mentioned above, *e.g.* DVSH and CorrAE, are not available and their performances have never been reported on the INRIA Web Queries dataset, we directly cite their results on COCO and IAPR TC-12 from [11] with the same settings and leave them blank on the INRIA Web Queries. For DNH [92], we cite the performances of its variants DNH-C provided in [11]. DCMH [66] is implemented by our own for comprehensive comparison before the authors release the official code.

To make fair comparisons, we utilize deep features for all traditional baseline methods mentioned above if the codes are available. For image features, we directly use the 4096-D AlexNet [85] pre-trained representations. For text features, a multi-label classification text-CNN, which shares most of its structure with our text encoding network $g(\cdot)$ excluding the last layer, is pre-trained on each dataset. Then a 384-D feature is extracted for each sentence from the pooling layers. In implementing DCMH [66] and testing CDQ [9], we build an identical text coding network to ours, enabling it to handle sentence data.

4.5.3.2 Results and Analysis

The mAP retrieval results on the three datasets are reported in Table 4.2 and Table 4.3. In general, the proposed TVDB model outperforms existing methods on the three datasets with large margins. Most traditional cross-modal hashing techniques are not usually designed specifically for images and sentences, which limits their performances on the three datasets. This suggests that image-tag hashing and retrieval are unrepresentative for vision-language tasks. The recent deep hashing model DVSH [11] hits the closest overall figures to ours as its modal-specific deep networks are able to explore the intrinsic semantics of images and sentences. TVDB provides even superior performance since both regional image information and relations between the words are well encoded, making the output binary codes more discriminative. Some results are left blank because the corresponding baseline codes are not available and the performances are never reported. Note that the overall mAP scores on INRIA Web Queries [83] are relatively lower than those on the other two. This is probably due to the relatively low image quality.

The Precision@200 scores given in Table 4.4 agree with the mAP performance. TVDB is leading all figures with large margins, barely followed by CDQ [9] and DCMH [66]. Precision@200 reflects the accuracy to the query within the top of the retrieved data sequence, which is essential to the empirical retrieval quality. As a result, we believe that TVDB retrieves intuitively better candidates to the query. This is supported by the intuitive analysis discussed in the following subsections. Figure 4.5 shows the HD2 precision and recall scores on the Microsoft COCO [102] dataset. It is clear that the HD2 precision and recall scores are not guaranteed to increase *w.r.t.* the encoding length M, because the overall hamming distances between data samples are expected to be larger with longer encoding length. However, TVDB is also able to outperform the listed baselines. This suggests that the proposed TVDB is able to produce descriptive binary codes employing semantic information for image-sentence retrieval. The corresponding precision-recall curves are given in Figure 4.3 (32 bits) and Figure 4.4 (128 bits).

We consider the proposed TVDB model benefits from both the precisely designed deep encoding networks and the high-efficiency alternating optimization routine, so it produces competitive image-sentence retrieval performance. The implications of the proposed deep binary encoding networks and the training routine are analyzed in the next two subsections.

Table 4.5: Ablation study: cross-modal retrieval mAP comparison of TVDB with several deep network baselines.

Task	**Method**	**Microsoft COCO**				**IAPR TC-12**			
		16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
	TVDB-I1 (full image only)	0.565	0.597	0.621	0.679	0.468	0.526	0.597	0.589
Image	TVDB-I2 (ave-pooled regions)	0.677	0.692	0.704	0.726	0.533	0.572	0.647	0.656
Query	TVDB-T1 (text bag-of-words)	0.553	0.565	0.591	0.631	0.436	0.508	0.503	0.548
Sentence	TVDB-T2 (text LSTM)	0.654	0.717	0.742	0.775	0.551	0.619	0,679	0.726
	TVDB (full model)	**0.702**	**0.781**	**0.797**	**0.818**	**0.629**	**0.678**	**0.731**	**0.772**
	TVDB-I1 (full image only)	0.573	0.634	0.653	0.657	0.462	0.514	0.516	0.568
Sentence	TVDB-I2 (ave-pooled regions)	0.633	0.645	0.687	0.717	0.541	0.568	0.615	0.629
Query	TVDB-T1 (text bag-of-words)	0.554	0.609	0.613	0.654	0.508	0.516	0.553	0.590
Image	TVDB-T2 (text LSTM)	0.642	0.734	0.761	0.782	0.613	0.606	0.635	0.696
	TVDB (full model)	**0.713**	**0.779**	**0.787**	**0.810**	**0.674**	**0.697**	**0.704**	**0.721**

4.5.4 Ablation Study of Deep Encoding Network

We demonstrate the impact of the deep encoding networks for TVDB in this subsection.

4.5.4.1 Baselines

Four variants of TVDB are built as baselines by modifying the deep encoding networks with some other architectures:

- **TVDB-I1** is built by replacing the region-based image encoding network of TVDB by a holistic AlexNet CNN [85].
- **TVDB-I2** mixes the image region features using average pooling instead of rendering them to the LSTM units.
- **TVDB-T1** takes text bag-of-words as sentence features with the original text-CNN removed.
- **TVDB-T2** is a variant of TVDB where the text-CNN is replaced by a two-layer LSTM structure, where images are encoded by a simple AlexNet CNN [85] and the text encoding network is LSTM-based.

4.5.4.2 Results and Analysis

Self-comparison mAP results on cross-modal retrieval are shown in Table 4.5. As we expected, the mAP scores drop dramatically with simple image CNN (TVDB-I1, TVDB-I2) and bag-of-words features (TVDB-T1), but are still acceptable compared to some existing methods. Image binary encoding without regional information is still far from satisfactory. TVDB-T2 with text-LSTM obtains reasonable performance and is in general superior to the state-of-the-art DVSH [11], since LSTM is also capable of modeling sentences. However, TVDB-T2 performs poorer than the original TVDB, suggesting that the proposed text-CNN architecture is a suitable choice for the cross-modal hashing task. To this end, it can be seen that our proposed binary encoding network is successfully designed and all components are reasonably implemented. It is worth noticing that TVDB-I1 uses the

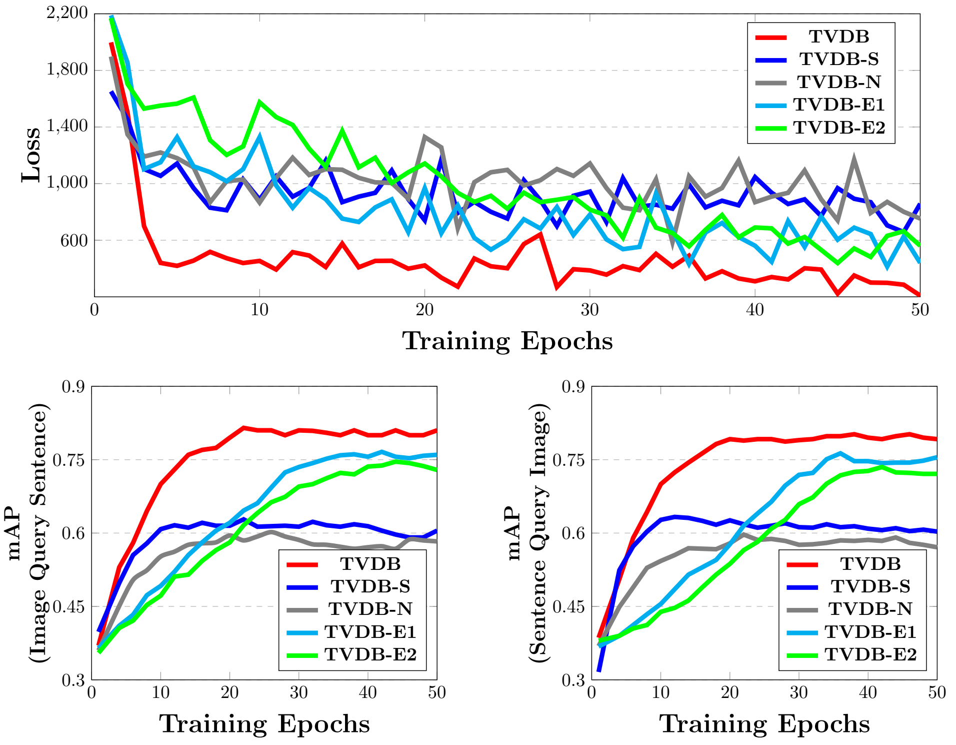


Figure 4.6: The 128-bit retrieval mAP of *Image Query Sentence* (bottom left) and *Sentence Query Image* (bottom right) *w.r.t.* training epochs on Microsoft COCO dataset are shown here. The corresponding losses are also given on the top.

same image and text representations as CDQ [9] and DCMH [66] during our experiments, and it performs slightly better than CDQ and DCMH. This may due to the well-designed training routine and learning objective evaluated in the next subsection.

4.5.5 Training Efficiency and Encoding Time

The training efficiency of the proposed learning objective and the *stochastic batch-wise training routine* are illustrated in this subsection. We firstly compare our optimization routine with several variant baselines of TVDB, while the networks are not modified for a fair comparison.

4.5.5.1 Baselines

- **TVDB-S** varies TVDB by keeping in-batch images and sentences unaltered with each epoch.
- **TVDB-N** is a variant of TVDB where $\mathbf{B}$ and $\mathbf{H}$ are initialized by $\max_{\mathbf{B},\mathbf{H}} \operatorname{trace}\left(\mathbf{B}\mathbf{S}\mathbf{H}^{\top}\right)$ and are not updated during SDG training. Here, $\mathbf{S}$ refers to the similarity matrix on the whole training set as in Equation (4.3).
- **TVDB-E1** is similar to the optimization of DCMH [66], performing epoch-wise binary code learning instead of batch-wise, i.e. updating both $\mathbf{B}, \mathbf{H}$ in a similar manner to Equation (4.8) and (4.9) on the whole training set after each epoch.
- **TVDB-E2** is similar to TVDB-E1, but code learning and updating is performed after every five epochs of training instead of each one.

4.5.5.2 Results and Analysis

Experiments of training efficiency are conducted on the Microsoft COCO [102] dataset with code length $M = 128$. We show results with $M = 128$ because the efficiency difference between TVDB and the above-mentioned baselines are most significant with this code length, and thus this makes the comparison illustrative. Less training iterations with superior performance are always in demand for cross-modal hashing and retrieval. The mAP scores and corresponding learning losses[2] *w.r.t.* training epochs are shown in Figure 4.6. It is obvious that TVDB converges quickly to an acceptable mAP score and then gradually hits the best performance at about the 20th epoch. TVDB is generally superior to the compared baselines both in terms of peak performance and training efficiency. It can be seen that TVDB-N obtains a similar rate of convergence to TVDB for the first five training epochs but ends up with significantly lower retrieval performance since the network parameters $\mathit{\Theta}$ and $\mathit{\Phi}$ are disjointly optimized with $\mathbf{B}$ and $\mathbf{H}$. TVDB-S follows a close path to TVDB-N with a slightly higher performance. It is clear that our code learning strategy with unaltered in-batch data is not appealing in exploring the

[2]The loss value here is the sum of Equation 4.10 for each training batch.

Table 4.6: Training Time Comparison with DCMH [66] (in Minutes). The first value of mAP here refers to the one of *Image Query Sentence* while the second refers to *Sentence Query Image*.

Dataset	**Bit**	**TVDB**		**DCMH**	
		mAP	Time	mAP	Time
COCO	16	0.702/0.713	434	0.562/0.595	615
	32	0.781/0.779	479	0.597/0.601	684
	64	0.797/0.787	503	0.609/0.633	790
	128	0.818/0.810	527	0.646/0.658	954
IAPR	16	0.629/0.674	212	0.443/0.486	299
	32	0.697/0.678	240	0.491/0.487	342
	64	0.731/0.704	268	0.559/0.499	434
	128	0.772/0.721	285	0.556/0.541	513

Table 4.7: Query Encoding Time Comparison (in Milliseconds)

Modality	**Method**	**16 bits**	**32 bits**	**64 bits**	**128 bits**
Image	TVDB	52.2	53.7	54.2	54.9
	DCMH	16.5	17.3	17.9	18.7
Sentence	TVDB	6.4	6.5	6.9	7.4
	DCMH	6.4	6.5	6.9	7.4

generalized optima of $\mathbf{B}$ and $\mathbf{H}$ on the whole dataset. TVDB-E1 and TVDB-E2 carry out acceptable retrieval performances but are still outperformed by TVDB. Although TVDB-E1 and TVDB-E2 also perform alternating optimization, the overall training efficiencies are far from satisfactory. This demonstrates that each aspect of TVDB is necessary to obtain optimal performance. On the other hand, computing $\max_{\mathbf{B},\mathbf{H}} \operatorname{trace}\left(\mathbf{B}\mathbf{S}\mathbf{H}^{\top}\right)$ on the whole training set is extremely time consuming, taking 130 minutes for updating $\mathbf{B}$ and $\mathbf{H}$ each round with an Intel XEON CPU server. However, since Equations (4.8,4.9) are only dealing with in-batch data and can be assigned to GPUs for computation, it requires only about 300 milliseconds for each batch of TVDB to work out $\mathbf{B}_b$ and $\mathbf{H}_b$, which significantly saves the training time. In particular, for a 20-epoch training, TVDB-E2 requires approximately 2600 minutes additional time than TVDB to optimize the model.

For a comprehensive study on training and coding efficiency, we also consider comparing TVDB with the most recent deep cross-modal hashing method. Note that it is infeasible and unreasonable to compare the time efficiency of TVDB with the traditional cross-modal hashing methods as these methods usually involve simpler binary encoding functions and less training variables with inferior performance to TVDB. To make a fair comparison, the most recent cross-modal hashing method, i.e. DCMH [66], is selected here as the benchmark.

The training time (in minutes) of TVDB and DCMH on Microsoft COCO and IAPR TC-12 is given in Table 4.6. It is clear that TVDB is more efficient than DCMH for training. TVDB consumes at average 70% of the training time of DCMH. We are aware that DCMH consists of more simpler deep network architecture than TVDB, and it requires less time to forward and backward a single batch during training. However, DCMH performs code updating after each epoch, during which each data point for training has to be forwarded into the encoding network again. This procedure considerably slows down the training speed of DCMH for each epoch, while TVDB performs batch-wise code updating. Furthermore, it takes more training epochs for DCMH than TVDB to obtain an acceptable result.

The encoding time (in milliseconds) for each query image and sentence is given in Table 4.7. Since the image encoding network for TVDB is composed by more sophisticated deep architecture than DCMH, the query image encoding time for TVDB is relatively longer. As we use exactly the same text encoding network as that of TVDB to implement DCMH, the sentence query encoding time does not differ. It is worthwhile noticing that, for the version of Tensorflow [1] used for implementation, the word embedding procedure of the text-CNN we use can only be conducted on CPUs currently. It is possible to further improve the sentence encoding efficiency if the word embedding procedure can be assigned to GPUs.

4.5.6 Intuitive Results

More intuitive cross-modal retrieval results are provided in this section. We consider the task of *Sentence Query Image* to be more important and illustrative than *Image Query Sentence* and thus, some *Sentence Query Image* comparisons are given on the Microsoft COCO dataset with code length $M = 128$. We show results with $M = 128$ to make the comparison illustrative, because TVDB outperforms the compared baselines with largest margin using this code length.

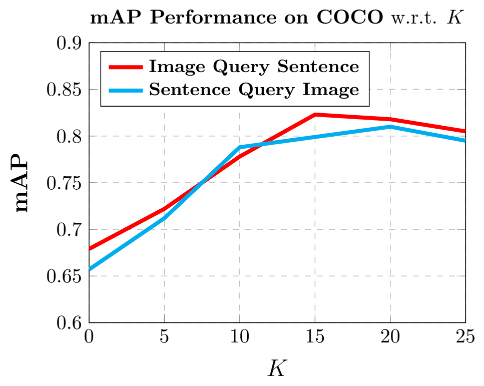
4.5.6.1 Positive Examples

Several selected query examples are listed and compared with some existing methods in Appendix C. We notice that these intuitive examples are the most illustrative and best-performing ones for TVDB. Again, TVDB provides closer retrieval candidates to the queries than the baselines. Most of the topics mentioned in the query sentences are correctly encoded and then retrieved by TVDB. On the other hand, the compared methods are generally able to select retrieval candidates sharing one or two semantic topics with the query but are empirically not providing satisfying matches. Taking the first query in Appendix C, i.e. Figure C.1, for instance, TVDB provides well-matched results with detailed information preserved (*e.g. people, bike, street*), while the compared methods are only able to give results around the topic of *people*.

4.5.6.2 Failure Cases

We provide two failure cases of TVDB in Appendix D, where almost all top-ten retrieved candidates are not relative. The first query in Appendix D consists of a long sentence of which the dominant object *zebra* appears at the end. The proposed text-CNN of TVDB is not able to handle this case. This can be improved by extending the input length of the text-CNN. In our experiments, we follow the design in [74] to set the input length of our text-CNN to be the average length of the training sentences, i.e. 12 words. Sentences longer than this length are trimmed using the first 12 words. As a result the information of an unusual long sentences is not fully encoded in our model. Extending the length of input sequence of the text-CNN allows the model to handle unusual long sentences. However, this requires more long descriptions as training data.

The second query of Appendix D illustrates a relatively rare situation. It is not very usual case that *carrots stand up* and *hold* something in the training set. The combination of these words results in a confusing semantic that our text-CNN can hardly process. To solve this problem, more training sentences with these unusual



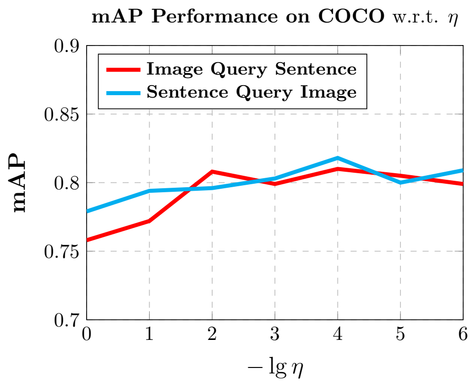


Figure 4.7: Hyper Parameter Analysis on Microsoft COCO [102] with coding length $M = 128$ for different K and η

descriptions are needed, and thus the network can recognize the actual semantic meanings of these text data.

4.5.7 Hyper-Parameters

Two hyper-parameters, i.e. K and η, are involved in the TVDB model, referring to the number of selected image regions and the quantization penalty respectively. We evaluate the retrieval performance *w.r.t.* K by fixing η and *vice versa*.

As is shown in Figure 4.7, the value of K plays an evident role in the overall performances of TVDB (the left-side figure). Reducing the value of K from 20 to 0 dramatically degrades the mAP score since the detailed image semantics are discarded, which agrees with the results of TVDB-I. On the other hand, we do not experience significant performance improvement when increasing K from 20 to 25. This is because it exceeds the encoding capability of our LSTM design.

The right-side graph of Figure 4.7 illustrates the retrieval performance of TVDB *w.r.t.* η. The overall performance difference *w.r.t.* η is within 5%. An extreme small value of η, i.e. 10^{-6}, does not produce best-performing retrieval scores as it scales down the loss function value of Equation 4.10, making it hard to train the network. Similarly, when setting η to be 1, the performance drops as well, because this results in large loss value of Equation 4.10, which is also not desired for neural network training.

4.6 Summary

In this chapter, we propose a deep binary encoding method termed as Textual-Visual Deep Binaries (TVDB) which is able to encode information-rich images and descriptive sentences. Two modal-specific binary encoding networks are built using LSTM and text-CNN, leveraging image regional information and semantics between the words to obtain high-quality binary representations. In addition, we propose a *stochastic batch-wise code learning routine* that performs effective and efficient training. Our experiments justify that both the proposed deep encoding networks and the training routine contribute greatly to the final outstanding cross-modal retrieval performance.

Chapter 5

Deep Hashing for Zero-Shot Image-Sketch Retrieval

5.1 Introduction and Motivation

Matching real images with hand-free sketches has recently aroused extensive research interest in computer vision, multimedia and machine learning, forming the term of Sketch-Based Image Retrieval (SBIR). Differing to the conventional text-image cross-modal retrieval, SBIR delivers a more applicable scenario where the targeted candidate images are conceptually unintelligible but visualizable to the user.. Several works have been proposed handling the SBIR task by learning real-valued representations [33, 34, 60, 61, 137, 138, 140, 148, 152, 161, 162, 170, 199, 200, 212, 213]. Cross-modal hashing techniques [7, 28, 205, 103, 90, 66, 9, 11] show great potential in retrieving heterogeneous data with high efficiency due to the computationally costless Hamming space matching, which is recently adopted to large-scale SBIR in [109] with impressive performance. Entering the era of big data, it is always feasible and appreciated to seek binary representation learning methods for fast SBIR.

However, the aforementioned works suffer from obvious drawbacks. Given a fixed set of categories of training and test data, these methods successfully manage to achieve sound SBIR performance, which is believed to be a relatively easy task as the visual knowledge from all concepts has been explored during parameter learning, while in a real-life scenario, there is no guarantee that the training data categories cover all concepts of potential retrieval queries and candidates in the

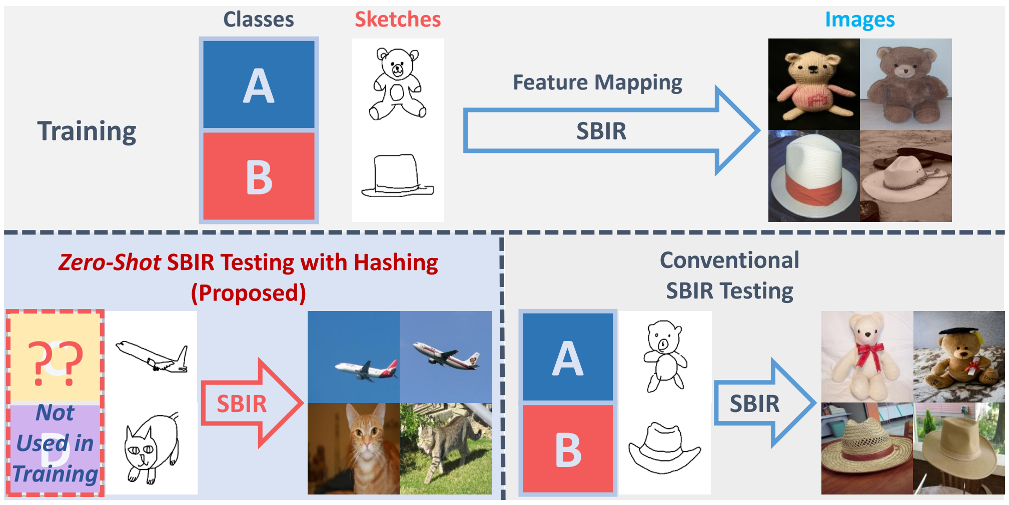


Figure 5.1: In conventional SBIR and cross-modal hashing (**bottom right**), the categories of training data include the ones of test data, marked as *'A'* and *'B'*. For our *zero-shot* task (**bottom left**), training data are still subjected to class *'A'* and *'B'*, but test sketches and images are coming from other categories, i.e. *'plane'* and *'cat'* in this case. Note that data labels are not used as test inputs and the test data categories shall be unknown to the learning system.

database. An extreme case occurs when test data are subjected to an absolutely different set of classes, excluding the trained categories. Unfortunately, experiments show that existing cross-modal hashing and SBIR works generally fail on this occasion as the trained retrieval model has no conceptual knowledge about what to find.

Considering both the **train-test category exclusion** and **retrieval efficiency**, a novel but realistic task yields *zero-shot* SBIR hashing. Figure 5.1 briefly illustrates the difference between our task and conventional SBIR task. In conventional SBIR and cross-modal hashing, the categories of training data include the ones of test data, marked as *'A'* and *'B'* in Figure 5.1. On the other hand, for the *zero-shot* task, though training data are still subjected to class *'A'* and *'B'*, test sketches and images are coming from other categories, i.e. *'plane'* and *'cat'* in this case. In the rest of this chapter, we denote the training and test categories as *seen* and *unseen* classes, since they are respectively known and unknown to the retrieval model.

Our *zero-shot* SBIR hashing setting is a special case of *zero-shot* learning in inferring knowledge out of the training samples. However, existing works basically focus on single-modal *zero-shot* recognition [158, 209, 210, 81], and are not suitable for efficient image retrieval. In [195], an inspiring *zero-shot* hashing scheme is proposed for large-scale data retrieval. Although [195] suggests a reasonable *zero-shot* train-test split close to Figure 5.1 for retrieval experiments, it is still not capable for cross-modal hashing and SBIR.

Regarding the drawbacks and the challenging task discussed above, a novel Zero-shot Sketch-Image Hashing (ZSIH) model is proposed in this chapter, simultaneously delivering (1) cross-modal hashing, (2) SBIR and (3) *zero-shot* learning. Leveraging state-of-the-art deep learning and generative hashing techniques, we formulate our deep network according to the following problems and themes:

1. Not all regions in an image or sketch are informative for cross-modal mapping.
2. The heterogeneity between image and sketch data needs to be mitigated during training to produce unified binary codes for matching.
3. Since visual knowledge alone is inadequate for *zero-shot* SBIR hashing, a back-propagatable deep hashing solution transferring semantic knowledge to the *unseen* classes is desirable.

The contributions of this work are summarized as follows:

- To the best of our knowledge, ZSIH is the first *zero-shot* hashing work for large-scale SBIR.
- We propose an end-to-end three-network structure for deep generative hashing, handling the train-test category exclusion and search efficiency with attention model, Kronecker fusion and graph convolution.
- The ZSIH model successfully produces reasonable retrieval performance under the *zero-shot* setting, while existing methods generally fail.

5.2 Related Work

ZSIH is proposed for a new task, i.e. *zero-shot* hashing for SBIR, which, to the best of our knowledge, has not been studied by existing works. Since ZSIH simultaneously handle (1) cross-modal hashing, (2) SBIR and (3) *zero-shot* learning, the related articles from these three aspects are discussed in this section. We also provide the reasons why these related works are not suitable for our task.

As is discussed in Chapter 4, general cross-modal binary representation learning methods [7, 28, 205, 90, 163, 127, 103, 66, 11, 180, 35, 211, 194, 9, 12, 125] target to map large-scale heterogeneous data with low computational cost. Although these works have shown potential in image-text retrieval, they are not suitable for our task for the following reasons:

- First of all, conventional cross-modal hashing methods do not employ compact encoding functions, and therefore fail to fully utilize the hidden information of images.
- Secondly, these models are not originally designed for SBIR. The structural similarities [109] between real images and hand-crafted sketches are not explored, resulting in unsatisfactory SBIR performance.
- Finally, existing cross-modal hashing methods do not consider the problem of train-test exclusion, and are not able to be applied to data of *unseen* categories.

SBIR, including fine-grained SBIR, aims at learning shared representations to specifically mitigate the expressional gap between hand-crafted sketches and real images [33, 34, 60, 61, 137, 138, 140, 148, 152, 161, 162, 170, 199, 200, 212, 213]. Sketch-a-Net (SaN) [200] is the first SBIR model that outperforms human on recognizing hand-crafted sketches. The utilization of the two-network pipeline in SaN largely improves the quality of learnt features. This network architecture is then extended in [140] with the siamese loss. The siamese loss [140] successfully mitigates the sketch-image heterogeneity. A similar network design is also employed in [199] focusing on retrieving shoe images using sketches. In [199], the triplet loss is involved to train the network in order to identify the similar shoe pictures from the dissimilar ones. However, conventional SBIR models are unable to handle our *zero-shot* cross-modal retrieval task for two reasons:

- Existing SBIR models aim at learning real-valued representations, which is inefficient for large-scale data retrieval.
- These works fail to handle the *zero-shot* tasks. The visual knowledge of *unseen* categories is not considered during training.

Zero-shot learning [38, 81, 209, 210, 158, 133, 19, 2, 185, 3, 93, 27, 20, 65, 207, 29, 99, 190, 71, 197] is also related to our work, though it does not originally focus on cross-modal retrieval. Traditionally, *zero-shot* learning focuses on recognizing and classifying data of *unseen* categories. To do this, auxiliary knowledge in addition to the visual information is introduced during training so that the category-wise semantic relations can be leveraged. Conventional semantic information for *zero-shot* learning includes category-level attributes [93] and word representation vectors [38, 129]. Existing works in this area mainly aim at learning a intermediate feature space between images and category-wise semantic representations for classification. In particular, two embedding functions, typically linear projection functions, project the images and the corresponding class-level semantic information, *e.g.* attributes or word vectors [129] of labels, to a common space, and then *zero-shot* classification is performed according to the distances between the image features and the category semantic features in this common space [38, 81, 209, 185]. Nevertheless, traditional *zero-shot* learning algorithms are not solving our task because of the following facts:

- The above-mentioned *zero-shot* learning methods only work on single-modal data, and the cross-modal data heterogeneity is not considered.
- These models are designed for classification tasks, and therefore are unable to directly produce compact binary codes for efficient similarity retrieval.

Among the existing research, Zero-Shot Hashing (ZSH) [195] and Deep Sketch Hashing (DSH) [109] are the two closest works to this chapter. DSH [109] considers fast SBIR with deep hashing technique. A three-encoder structure is defined, taking real images, hand-crafted sketches and image edge maps [100] as inputs. By introducing the edge maps [100], the geometric difference between sketches and real images is mitigated. The learning objective of DSH [109] is built to minimize the code differences of data within each training category, but the inter-category code distances are not considered. Similar to the traditional cross-modal hashing models, DSH [109] does not transfer the supervised label knowledge to *unseen* categories. As a result, it fails to handle the *zero-shot* setting. On the other hand, ZSH [195] is the first work that extends the traditional *zero-shot* learning model to a single-modal hashing scheme. The semantic information, *e.g.* attributes or word vectors [129] of labels, implicitly defines the category-wise code distances, which is utilized to formulate the learning objective of ZSH [195]. Therefore, the learnt code space is structured and capable to be generalized from the *seen* categories to the *unseen* ones. This design is followed by the Similarity-transfer Network (SitNet) [46]. Instead of the linear projection function for hashing in ZSH [195], SitNet [46] involves a CNN as the encoder. ZSH [195] and SitNet [46] are unable to simultaneously encode cross-modal data, and therefore are not suitable for the task of this chapter. In addition, a similar *remove-one-class* metric is evaluated in [120, 16], but these works are proposed for classification tasks instead of cross-modal hashing and SBIR.

5.3 The Proposed ZSIH Model

This work focuses on solving the problem of hand-free SBIR using deep binary codes under the *zero-shot* setting, where the image and sketch data belonging to the *seen* categories are only used for training. The proposed deep networks are expected to be capable for encoding and matching the *unseen* sketches with images, categories of which have never appeared during training. As this is a new task, in this section, we firstly introduce some preliminary notation with the task definition, and then the proposed model is discussed in detail.

We consider a multi-modal data collection $\mathcal{O}^c = \{\mathbf{X}^c, \mathbf{Y}^c\}$ from *seen* categories $\mathcal{C}^c$ covering both real images $\mathbf{X}^c = \{\mathbf{x}_i^c\}_{i=1}^N$ and sketch images $\mathbf{Y}^c = \{\mathbf{y}_i^c\}_{i=1}^N$ for training, where N indicates the set size. For the simplicity of presentation, it is assumed that image and sketch data with the same index i, i.e. $\mathbf{x}_i^c$ and $\mathbf{y}_i^c$ share the

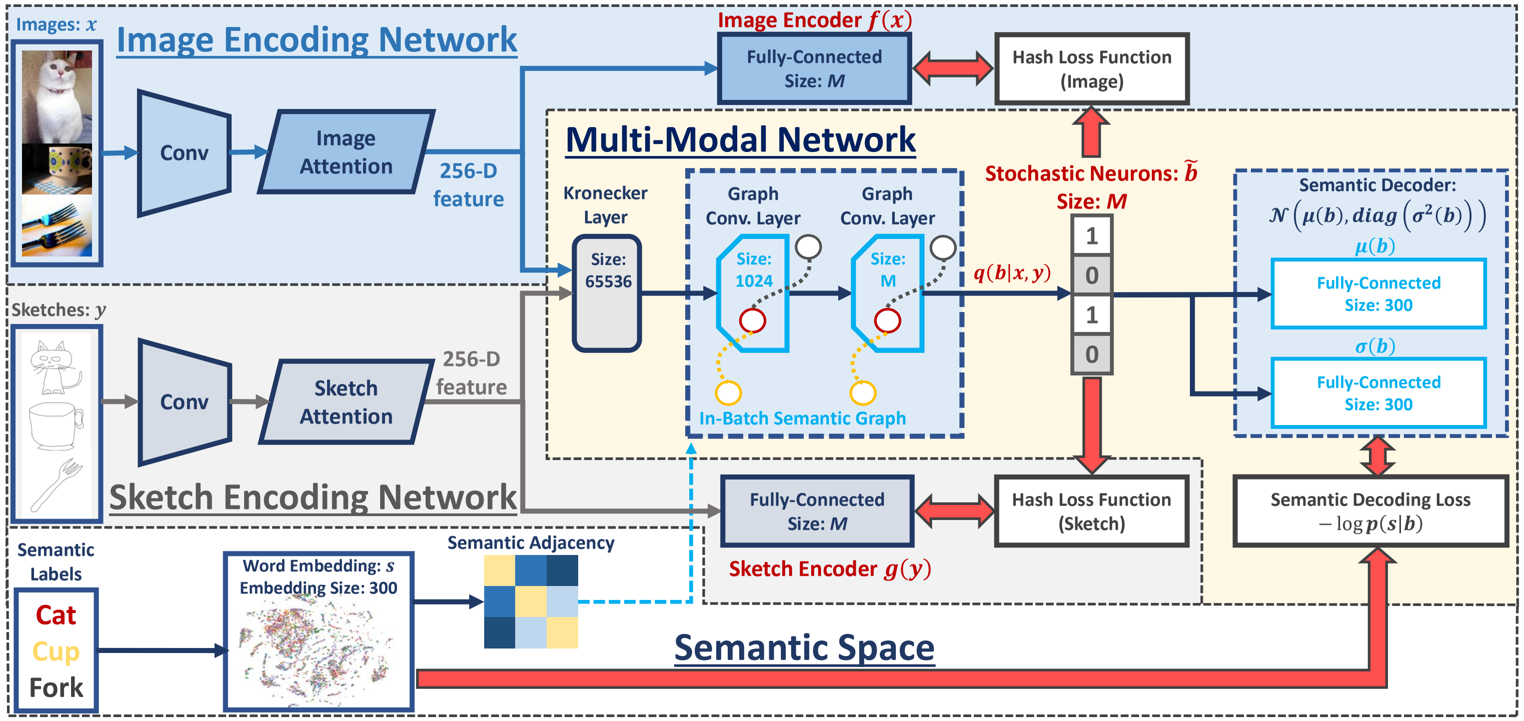


Figure 5.2: The deep network structure of ZSIH. The image (in light blue) and sketch encoding network (in grey) act as hash function for the respective modality with attention models [162]. The multi-modal network (in canary yellow) only functions during training. Sketch-image representations are fused by a Kronecker layer [58]. Graph convolution [79] and generative hashing techniques are leveraged to explore the semantic space for *zero-shot* SIBR hashing. **Network configurations** are also provided here.

same category label. Additionally, similar to many conventional *zero-shot* learning algorithms, our model requires a set of semantic representations $\mathbf{S}^c = \{\mathbf{s}_i^c\}_{i=1}^N$ in transferring supervised knowledge to the *unseen* data. The aim is to learn two deep hashing functions $f(\cdot)$ and $g(\cdot)$ for images and sketches respectively. Given a set of image-sketch data $\mathcal{O}^u = \{\mathbf{X}^u, \mathbf{Y}^u\}$ belonging to the *unseen* categories $\mathcal{C}^u$ for test, the proposed deep hashing functions encode these *unseen* data into binary codes, i.e. $f : \mathbb{R}^d \to \{0,1\}^M, g : \mathbb{R}^d \to \{0,1\}^M$, where d refers to the original data dimensionality and M is the targeted hash code length. The image and sketch data from the same category are supposed to be encoded closely in hamming space for cross-modal search. Concretely, as the proposed model handles SBIR under the *zero-shot* setting, there should be no intersection between the *seen* categories for training and the *unseen* classes for test, i.e. $\mathcal{C}^c \bigcap \mathcal{C}^u = \varnothing$. Note that since *zero-shot* learning conventionally targets at revealing category-level information of data points, it is feasible to define the relevant cross-modal data by sharing the coherent class label in our task.

5.3.1 Network Overview

The proposed ZSIH model is an end-to-end deep neural network for *zero-shot* sketch-image hashing. The architecture of ZSIH is illustrated in Figure 5.2, which is composed of three concatenated deep neural networks, i.e. the image/sketch encoders and the multi-modal network, to tackle the problems discussed above.

5.3.1.1 Image/Sketch Encoding Networks

As is shown in Figure 5.2, the networks with light blue and grey background refer to the binary encoders $f(\cdot)$ and $g(\cdot)$ for images and sketches respectively. An image or sketch is firstly rendered to a set of corresponding convolutional layers to produce a feature map, and then the attention model mixes informative parts into a single feature vector for further operation. The AlexNet [85] before the last pooling layer is built to obtain the feature map. We introduce the attention mechanism in solving issue 1 of Section 5.1, of which the structure is close to [162] with weighted pooling to produce a 256-D feature. Considering a convolutional feature map $\mathbf{h}^{conv} \in \mathbb{R}^{13\times13\times256}$ obtained from either the sketch or image CNNs, the attention score $\mathbf{s}_{i,j}$ of a certain entry (i, j) is computed as follows:

$$\mathbf{s}_{i,j} = \mathbf{h}_{i,j}^{conv}\mathbf{W}^{att}. \tag{5.1}$$

Note that the size of the convolutional feature map, i.e. $\mathbf{h}^{conv} \in \mathbb{R}^{13\times13\times256}$, is determined by the structure of the employed AlexNet CNNs [85]. We omit the bias here for the simplicity of representation. Therefore, the attention mask α is defined by

$$\alpha_{i,j} = \texttt{softmax}\left(\mathbf{s}_{i,j}\right). \tag{5.2}$$

Finally, the attended 256-D sketch/image representation $\mathbf{h}^{att}$ is obtained by

$$\mathbf{h}^{att} = \sum_{i,j} \alpha_{i,j}\mathbf{h}_{i,j}^{conv} \in \mathbb{R}^{256}. \tag{5.3}$$

Binary encoding is performed by a fully-connected layer taking input from the attention model with a `sigmoid` non-linearity. During training, $f(\cdot)$ and $g(\cdot)$ are regularized by the output of the multi-modal network, so these two encoders are supposed to be able to learn modal-free representations for *zero-shot* sketch-image matching.

5.3.1.2 Multi-Modal Network as Code Learner

The multi-modal network only functions during training. It learns the joint representations for sketch-image hashing, handling the problem 2 of modal heterogeneity. One possible solution for this is to introduce a fused representation layer taking inputs from both image and sketch modality for further encoding. Inspired by Hu *et al.* [58], we find the Kronecker product fusion layer suitable for our model, which is discussed in Section 5.3.2. Shown in Figure 5.2, the Kronecker layer takes inputs from the image and sketch attention model, and produces a single feature vector for each pair of data points. We index the training images and sketches in a coherent category order. Therefore the proposed network is able to learn compact codes for both images and sketches with clear categorical information.

However, simply mitigating the model heterogeneity does not fully solves the challenges in ZSIH. As is mentioned in problem 3 of Section 5.1, for *zero-shot* tasks, it is essential to leverage the semantic information of training data to generalize knowledge from the *seen* categories to the *unseen* ones. Suggested by many *zero-shot* learning works [81, 38, 195], the semantic representations, *e.g.* word vectors [129], implicitly determine the category-level relations between data points from different classes. Based on this, during the joint code learning process, we novelly enhance the hidden neural representations by the semantic relations within a batch of training data using the Graph Convolutional Networks (GCNs) [26, 79]. It can be observed in Figure 5.2 that two graph convolutional layers are built in the multi-modal network, successively following the Kronecker layer. In this way, the in-batch data points with strong latent semantic relations are entitled to interact during gradient computation. Note that the output length of the second graph convolutional layer for each data point is exactly the target hash code length, i.e. M. The formulation of the semantic graph convolution layer is given in Section 5.3.3.

To obtain binary codes as the supervision of $f(\cdot)$ and $g(\cdot)$, we introduce the stochastic generative model [25] for hashing. A back-propagatable structure of stochastic neurons is built on the top of the second graph convolutional layer, producing hash codes. Shown in Figure 5.2, a decoding model is topped on the stochastic neurons, reconstructing the semantic information. By maximizing the decoding likelihood with gradient-based methods, the whole network is able to learn semantic-aware hash codes, which also accords to our perspective of issue 3 for *zero-shot* sketch-image hashing. We elaborate on this design in Section 5.3.4 and 5.3.5.

5.3.2 Fusing Sketch and Image with Kronecker Layer

Sketch-image feature fusion plays an important role in our task as is addressed in problem 2 of Section 5.1. An information-rich fused neural representation is in demand for accurate encoding and decoding. To this end, we utilize the recent advances in Kronecker-product-based feature learning [58] as the fusion network. Denoting the attention model outputs of a sketch-image pair $\{\mathbf{y}, \mathbf{x}\}$ from the same category as $\mathbf{h}^{(sk)} \in \mathbb{R}^{256}$ and $\mathbf{h}^{(im)} \in \mathbb{R}^{256}$, the fused output of the Kronecker layer

$\mathbf{h}^{(kron)}$ in our model is derived as

$$\mathbf{h}^{(kron)} = \delta\big((\mathbf{h}^{(sk)}\mathbf{W}^{(sk)}) \otimes (\mathbf{h}^{(im)}\mathbf{W}^{(im)})\big), \tag{5.4}$$

resulting in a 65536-D feature vector. Here $\otimes$ is the Kronecker product operation between two tensors, and $\mathbf{W}^{(sk)}, \mathbf{W}^{(im)} \in \mathbb{R}^{256\times256}$ are matrices of trainable linear transformation parameters. $\delta(\cdot)$ refers to the activation function, which is the `ReLU` [131] non-linearity for this layer.

Kronecker layer [58] is supposed to be a better choice in feature fusion for ZSIH than many conventional methods such as layer concatenation or factorized model [202]. This is because the Kronecker layer largely expands the feature dimensionality of the hidden states with a limited number of parameters, and thus consequently stores more expressive structural relation between sketches and images.

5.3.3 Semantic-Relation-Enhanced Hidden Representation with Graph Convolution

In this subsection, we describe how the categorical semantic relations are enhanced in our ZSIH model using GCNs. Considering a batch of training data $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{s}_i\}_{i=1}^{N_B}$ consisting of N_B category-coherent sketch-image pairs with their semantic representations $\{\mathbf{s}_i\}$, we denote the hidden state of the l-th layer in the multi-modal network of this training batch as $\mathbf{H}^l$ to be rendered to a graph convolutional layer. As is mentioned in Section 5.3.1.2, for our graph convolutional layers, each training batch is regarded as an N_B-vertex graph. Therefore, a convolutional filter g_θ parameterized by θ can be applied to $\mathbf{H}^l$, producing the $(l+1)$-th hidden state $\mathbf{H}^{(l+1)} = g_\theta * \mathbf{H}^{(l)}$. Suggested by [79], this can be approached by a layer-wise propagation rule, i.e.

$$\mathbf{H}^{(l+1)} = \delta\big(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}_\theta\big), \tag{5.5}$$

using the first-order approximation of the localized graph filter [26, 48]. Again, here $\delta(\cdot)$ is the activation function and $\mathbf{W}_\theta$ refers to the matrix of linear transformation parameters. $\mathbf{A}$ is an $N_B \times N_B$ self-connected in-batch adjacency and $\mathbf{D}$ can be defined by $\mathbf{D} = \texttt{diag}(\mathbf{A}\mathbb{1})$. It can be seen in Figure 5.2 that the in-batch adjacency

$\mathbf{A}$ is determined by the semantic representations $\{\mathbf{s}_i\}$, of which each entry $\mathbf{A}_{j,k}$ is computed by

$$\mathbf{A}_{j,k} = e^{-\frac{\|\mathbf{s}_j - \mathbf{s}_k\|^2}{t}}, \tag{5.6}$$

where t is a hyper-parameter included in our ablation study. In the proposed ZSIH model, two graph convolutional layers are built, with output feature dimensions of $N_B \times 1024$ and $N_B \times M$ for a whole batch. We choose the `ReLU` nonlinearty [131] for the first layer and the `sigmoid` function for the second one to restrict the output values between 0 and 1.

Intuitively, the graph convolutional layer proposed by [79] can be construed as performing elementary row transformation on a batch of data from a fully-connected layer before activation according to the graph Laplacian of $\mathbf{A}$. In this way, the semantic relations between different data points are intensified within the network hidden states, benefiting our *zero-shot* hashing model in exploring the semantic knowledge. Traditionally, correlating different deep representations can be tackled by adding a trace-like regularization term in the learning objective. However, this introduces additional hyper parameters to balance the loss terms and the hidden states in the network of different data points are still isolated, leading to inferior performance to GCNs.

5.3.4 Stochastic Neurons and Decoding Network

The encoder-decoder model for ZSIH is introduced in this subsection. Inspired by [25], a set of latent probability variables $\mathbf{b} \in (0,1)^M$ are obtained from the second graph convolutional layer output respective to $\{\mathbf{x}, \mathbf{y}\}$ corresponding to the hash code for a sketch-image pair $\{\mathbf{x}, \mathbf{y}\}$ with the semantic feature $\mathbf{s}$. The stochastic neurons [25] are imposed to $\mathbf{b}$ to produce binary codes $\widetilde{\mathbf{b}} \in \{0,1\}^M$ through a sampling procedure:

$$\widetilde{\mathbf{b}}^{(m)} = \begin{cases} 1 & \mathbf{b}^{(m)} \geqslant \epsilon^{(m)}, \\ 0 & \mathbf{b}^{(m)} < \epsilon^{(m)}, \end{cases} \quad \text{for } m = 1 \ldots M, \tag{5.7}$$

where $\epsilon^{(m)} \sim \mathcal{U}([0,1])$ are random variables. As has been proven in [25], this structure can be differentiable, allowing error back-propagation from the decoder

Algorithm 3: The Training Procedure of ZSIH

```
Input: Sketch-image dataset $\mathcal{O} = \{\mathbf{X}, \mathbf{Y}\}$, semantic representations $\mathbf{S}$
  and max training iteration $T$
Output: Network parameters $\Theta$
repeat
    Get a random mini-batch $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{s}_i\}_{i=1}^{N_B}$, assuring $\mathbf{x}_i, \mathbf{y}_i$ belong to the
      same class
    Build $\mathbf{A}$ according to semantic distances
    for $i = 1 \ldots N_B$ do
        Sample a set of $\epsilon^{(m)} \sim \mathcal{U}([0,1])$
        Sample a set of $\widetilde{\mathbf{b}} \sim q(\mathbf{b}|\mathbf{x}_i, \mathbf{y}_i)$
    end
    $\mathcal{L} \leftarrow$ Equation (5.10)
    $\Theta^{new} \leftarrow \Theta - \mathbf{\Gamma}(\nabla_\Theta \mathcal{L})$ according to Equation (5.11)
until convergence or max training iter $T$ is reached;
```

to the previous layers. Therefore, the posterior of $\mathbf{b}$, i.e. $p(\mathbf{b}|\mathbf{x}, \mathbf{y})$, is approximated by a Multinoulli distribution:

$$q(\widetilde{\mathbf{b}}|\mathbf{x}, \mathbf{y}) = \prod_{m=1}^{M} (\mathbf{b}^{(m)})^{\widetilde{\mathbf{b}}^{(m)}} (1 - \mathbf{b}^{(m)})^{1-\widetilde{\mathbf{b}}^{(m)}}. \tag{5.8}$$

We follow the idea of generative hashing to build a decoder on the top of the stochastic neurons. During optimization of ZSIH, this decoder is regularized by the semantic representations $\mathbf{s}$ using the following Gaussian likelihood with the reparametrization trick [76], i.e.

$$p(\mathbf{s}|\mathbf{b}) = \mathcal{N}\big(\mathbf{s}|\mu(\mathbf{b}), \mathtt{diag}(\sigma^2(\mathbf{b}))\big), \tag{5.9}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ are implemented by fully-connected layers with identity activations. To this end, the whole network can be trained en-to-end. The learning objective is given in the next subsection.

5.3.5 Learning Objective and Optimization

The learning objective of the whole network for a batch of sketch and image data is defined as follows [25]:

$$\begin{aligned}\mathcal{L} = \sum_{i=1}^{N_B} \mathbb{E}_{q(\mathbf{b}|\mathbf{x}_i,\mathbf{y}_i)} \big[&\log q(\mathbf{b}|\mathbf{x}_i, \mathbf{y}_i) - \log p(\mathbf{s}_i|\mathbf{b}) \\ &+ \frac{1}{2M}\big(\|f(\mathbf{x}_i) - \mathbf{b}\|^2 + \|g(\mathbf{y}_i) - \mathbf{b}\|^2\big)\big],\end{aligned} \tag{5.10}$$

which is composed of an expectation term $\mathbb{E}[\cdot]$ and two $l2$ terms. Concretely, the expectation term $\mathbb{E}[\cdot]$ in Equation (5.10) simulates the variational-like learning objectives [76, 25] as a generative model. However, we are not exactly lower-bounding any data prior distribution since it is generally not feasible for our ZSIH network. $\mathbb{E}[\cdot]$ here is an empirically-built loss, simultaneously maximizing the output code entropy via $\mathbb{E}_{q(\mathbf{b}|\mathbf{x},\mathbf{y})}[\log q(\mathbf{b}|\mathbf{x}, \mathbf{y})]$ and preserving the semantic knowledge for the *zero-shot* task by $\mathbb{E}_{q(\mathbf{b}|\mathbf{x},\mathbf{y})}[-\log p(\mathbf{s}|\mathbf{b})]$. The single-model encoding functions $f(\cdot)$ and $g(\cdot)$ are trained by the stochastic neurons outputs of the multi-modal network using $l2$ losses. The sketch-image similarities can be reflected in assigning related sketches and images with the sample code. To this end, $f(\cdot)$ and $g(\cdot)$ are able to encode out-of-sample data without additional category information, as the imposed training codes are semantic-knowledge-aware. The gradient of our learning objective *w.r.t.* the network parameter Θ can be estimated by a Monte Carlo process in sampling $\widetilde{\mathbf{b}}$ using the small random signal ϵ according to Equation (5.7), which can be derived as

$$\begin{aligned}\nabla_\Theta \mathcal{L} \simeq \sum_{i=1}^{N_B} \mathbb{E}_{\epsilon}\Big[\nabla_\Theta \Big(&\log q(\widetilde{\mathbf{b}}|\mathbf{x}_i, \mathbf{y}_i) - \log p(\mathbf{s}_i|\widetilde{\mathbf{b}}) \\ &+ \frac{1}{2M}\big(\|f(\mathbf{x}_i) - \widetilde{\mathbf{b}}\|^2 + \|g(\mathbf{y}_i) - \widetilde{\mathbf{b}}\|^2\big)\Big)\Big].\end{aligned} \tag{5.11}$$

As $\log q(\cdot)$ forms up into an inverse cross-entropy loss and $\log p(\cdot)$ is reparametrized, this estimated gradient can be easily computed. Note that we are not propagating the $l2$ errors back to the stochastic neurons. Algorithm 3 illustrates the whole training process of the proposed ZSIH model, where the operator $\mathbf{\Gamma}(\cdot)$ refers to the Adam optimizer [75] for adaptive gradient scaling. Different from many existing deep cross-modal and *zero-shot* hashing models [9, 109, 195, 66] which require alternating optimization procedures, ZSIH can be efficiently and conveniently trained end-to-end with SGD.

5.3.6 Out-of-Sample Extension

When the network of ZSIH is trained, it is able to hash image and sketch data from the *unseen* classes $\mathcal{C}^u$ for matching. The codes can be obtained as follows:

$$\begin{aligned}\mathbf{B}^{im} &= (\texttt{sign}(f(\mathbf{X}^u - 0.5)) + 1)/2 \in \{0,1\}^{N^u \times M},\\ \mathbf{B}^{sk} &= (\texttt{sign}(g(\mathbf{Y}^u - 0.5)) + 1)/2 \in \{0,1\}^{N^u \times M},\end{aligned} \tag{5.12}$$

where N^u is the size of test data. As is shown in Figure 5.2, the encoding networks $f(\cdot)$ and $g(\cdot)$ are standing on their own. Semantic representations of test data are not required and there is no need to render data to the multi-modal network. Thus, encoding test data is non-trivial and can be efficient.

As is suggested by existing *zero-shot* research [195, 46], the auxiliary semantic representations during training implicitly demonstrates the category-wise similarities of data. By preserving the category-wise semantic relations between encoded data, the learnt feature space is capable to be applied to *unseen* visual concepts [195]. Therefore the trained model handles the*zero-shot* retrieval task. In this chapter, this is achieved by introducing the GCNs [79] and the semantic decoder into the multi-modal network. The GCNs [79] mix data hidden representations with semantically related concepts. The semantic decoding network introduces semantic knowledge to the stochastic neurons [25] to produce compact codes. In this way, ZSIH recognizes *unseen* visual concepts for the *zero-shot* SBIR task.

5.4 Experiments

Extensive experiments are elaborated in this section. We firstly introduce our implementation details and the *zero-shot* experimental settings, and then results are provided according to the following themes:

- Comparison with existing methods

- Conventional category-level SBIR
- Ablation study
- Training efficiency and overfitting
- Qualitative results and failure cases

5.4.1 Implementation Details

The proposed ZSIH model is implemented with the popular deep learning toolbox Tensorflow [1]. We utilize the settings of AlexNet [85] pre-trained on ImageNet [147] before the last pooling layer to build our image and sketch CNNs. The attention mechanism is inspired by Song *et al.* [162] without the shortcut connection. The attended 256-D feature is obtained by a weighted pooling operation according to the attention map. All configurations of our network are provided in Figure 5.2. We obtain the semantic representation of each data point using the 300-D word vector [129] according to the class name. When the class name is not included in the word vector dictionary, it is replaced by a synonym. For all of our experiments, the hyper-parameter t is set to $t = 0.1$ with a training batch size of 250. Our network is able to be trained end-to-end.

5.4.2 Zero-shot experimental settings

To perform SBIR with binary codes under the novelly-defined *zero-shot* cross-modal setting, the experiments of this work are taken on two large-scale sketch datasets, i.e. Sketchy [152] and TU-Berlin [32], with extended images obtained from [109]. We follow the SBIR evaluation metrics in [109] where sketch queries and image retrieval candidates with the same label are marked as relevant, while our retrieval performances are reported based on nearest neighbour search in the hamming space. Note that the *zero-shot* retrieval experiments reported in this section are strictly evaluated on unseen images and sketches.

5.4.2.1 Sketchy Dataset [152] (Extended)

This dataset originally consists of 75,471 hand-drawn sketches and 12,500 corresponding images from 125 categories. With the extended 60,502 real images provided by Liu *et al.* [109], the total size of the whole image set yields 73,002. We randomly pick 25 classes of sketches and images as the *unseen* test set for SBIR, and data from the rest 100 *seen* classes are used for training. During the test phase, the sketches from the *unseen* classes are taken as retrieval queries, while the retrieval gallery is built using all the images from the *unseen* categories. Note that the test classes are not presenting during training for *zero-shot* retrieval.

5.4.2.2 TU-Berlin Dataset [32] (Extended)

The TU-Berlin dataset contains 20,000 sketches subjected to 250 categories. We also utilize the extended nature images provided in [109, 206] with a total size of 204,489. 30 classes of images and sketches are randomly selected to form the retrieval gallery and query set respectively. The rest of the data are used for training. Since the quantities of real images from different classes are extremely imbalanced, we additionally require each test category have at least 400 images when picking the test set.

5.4.3 Comparison with Existing Methods

As cross-modal hashing for SBIR under the *zero-shot* setting has never been proposed before to the best of our knowledge, the quantity of potential related existing baselines is limited. Our task can be regarded as a combination of conventional cross-modal hashing, SBIR and *zero-shot* learning. Therefore, we adopt existing methods according to these themes for retrieval performance evaluation. We use the *seen*-*unseen* splits identical to ours for training and testing the selected baselines. The deep-learning-based baselines are retrained end-to-end using the *zero-shot* setting mentioned above. For the non-deep baselines, we extract the respective AlexNet [85] `fc_7` features pre-trained on the *seen* sketches and images as model training inputs for a fair comparison with our deep model.

Table 5.1: *zero-shot* SBIR mAP comparison between ZSIH and some cross-modal hashing baselines.

Method	Cross Modal	Binary Code	Zero Shot	Sketchy (Extended)			TU-Berlin (Extended)		
				32 bits	64 bits	128 bits	32 bits	64 bits	128 bits
ZSH [195]		✓	✓	0.146	0.165	0.168	0.132	0.139	0.153
SitNet [46]		✓	✓	0.139	0.170	0.153	0.118	0.145	0.149
CCA [167]	✓			0.092	0.089	0.084	0.083	0.074	0.062
CMSSH [7]	✓	✓		0.094	0.096	0.111	0.073	0.077	0.080
CMFH [28]	✓	✓		0.115	0.116	0.125	0.114	0.118	0.135
SCM-Orth [205]	✓	✓		0.105	0.107	0.093	0.089	0.092	0.095
SCM-Seq [205]	✓	✓		0.092	0.100	0.084	0.084	0.087	0.072
CVH [90]	✓	✓		0.076	0.075	0.072	0.065	0.061	0.055
SePH-Rand [103]	✓	✓		0.108	0.097	0.094	0.071	0.065	0.070
SePH-KM [103]	✓	✓		0.069	0.066	0.071	0.067	0.068	0.065
DSH [109]	✓	✓		0.137	0.164	0.165	0.119	0.122	0.146
ZSIH	✓	✓	✓	**0.232**	**0.254**	**0.259**	**0.201**	**0.220**	**0.234**

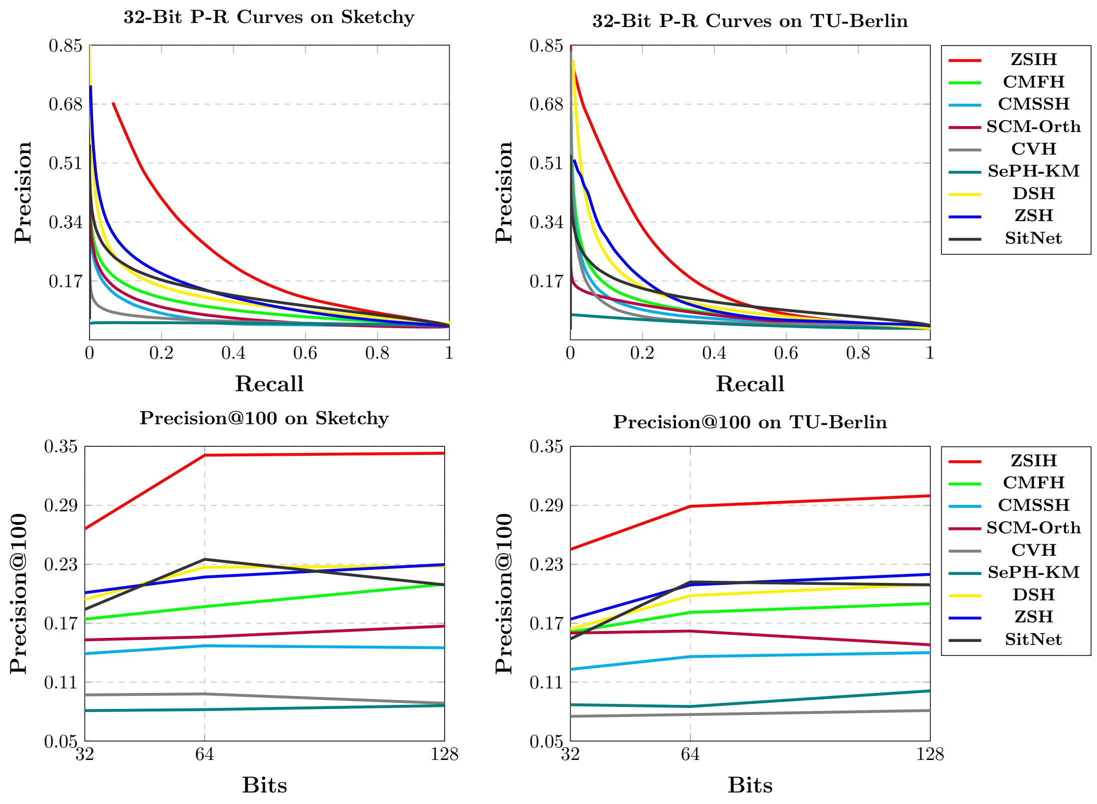


Figure 5.3: P-R curves and Precision@100 results of ZSIH and several hashing baselines are shown above. To keep the content concise, only 32-bit precision-recall curves are illustrated here.

5.4.3.1 Cross-Modal Hashing Baselines

Several state-of-the-art cross-modal hashing works are introduced including CMSSH [7], CMFH [28], SCM [205], CVH [90], SePH [103] and DSH [109], where DSH [109] can also be subjected to an SBIR model and thus is closely related to our work. In addition, CCA [167] is considered as a conventional cross-modal baseline, though it learns real-valued joint representations. These models are widely used as baselines in cross-modal hashing research [109].

5.4.3.2 *Zero-Shot* Baselines

Existing *zero-shot* learning works are not originally designed for cross-modal search. We select a set of state-of-the-art *zero-shot* learning algorithms as benchmarks, including Cross-Modal Transfer (CMT) [158], Deep Visual-Semantic Embedding (DeViSE) [38], Semantic Similarity Embedding (SSE) [209], Joint Latent Similarity Embedding (JLSE) [210] and Semantic Auto-Encoder (SAE) [81] because they are the best-performing *zero-shot* learning models in classification tasks [186]. In addition, ZSH [195] and SitNet [46] are introduced for comparison since , to the best of our knowledge, they are the only two existing *zero-shot* hashing models.

For CMT [158], DeViSE [38] and SAE [158], two sets of 300-D embedding functions are trained for sketches as images with the word vectors [129] as the semantic information for nearest neighbour retrieval, and the classifiers used in these works are ignored. SSE [209] and JLSE [210] are based on *seen-unseen* class mapping, so the output embedding sizes are set to 100 and 220 for Sketchy [152] and TU-Berlin [32] dataset respectively. We train two modal-specific encoders of ZSH [195] and SitNet [46] simultaneously for our task.

5.4.3.3 Sketch-Image Mapping Baselines

Siamese CNN [140], SaN [200], GN Triplet [152], 3D Shape [170] and DSH [109] are involved as SBIR baselines. These models are originally designed for category-level SBIR and attain state-of-the-art performance [109]. As a result, they are suitable baselines for our experiments. We follow the instructions of the original papers to build and train the networks under the *zero-shot* setting. A softmax baseline

Table 5.2: *Zero-shot* sketch-image retrieval performance comparison of ZSIH with existing SBIR and *zero-shot* learning methods.

Type	**Method**	**Sketchy (Extended)**				**TU-Berlin (Extended)**			
		mAP	**Precision @100**	**Feature Dimension**	**Retrieval Time (s)**	**mAP**	**Precision @100**	**Feature Dimension**	**Retrieval Time (s)**
SBIR	Softmax Baseline	0.099	0.176	4096	3.9×10^{-1}	0.083	0.139	4096	4.7×10^{-1}
	Siamese CNN [140]	0.143	0.183	64	5.2×10^{-3}	0.122	0.153	64	6.3×10^{-3}
	SaN [200]	0.104	0.129	512	4.4×10^{-2}	0.096	0.112	512	5.1×10^{-2}
	GN Triplet [152]	0.211	0.310	1024	8.9×10^{-2}	0.189	0.241	1024	1.4×10^{-1}
	3D Shape [170]	0.062	0.070	64	5.6×10^{-3}	0.057	0.063	64	7.0×10^{-3}
	DSH (64 bits) [109]	0.164	0.227	64 (binary)	6.3×10^{-5}	0.122	0.198	64 (binary)	7.5×10^{-5}
Zero-Shot	CMT [158]	0.084	0.096	300	3.1×10^{-2}	0.065	0.082	300	3.7×10^{-2}
	DeViSE [38]	0.071	0.078	300	3.2×10^{-2}	0.067	0.075	300	3.7×10^{-2}
	SSE [209]	0.108	0.154	100	1.1×10^{-2}	0.096	0.133	220	1.3×10^{-2}
	JLSE [210]	0.126	0.178	100	1.1×10^{-2}	0.107	0.165	220	1.3×10^{-2}
	SAE [81]	0.210	0.302	300	3.1×10^{-2}	0.161	0.210	300	3.7×10^{-2}
	ZSH (64 bits) [195]	0.165	0.217	64 (binary)	6.3×10^{-5}	0.139	0.174	64 (binary)	7.5×10^{-5}
	SitNet (64 bits) [46]	0.170	0.235	64 (binary)	6.3×10^{-5}	0.145	0.192	64 (binary)	7.5×10^{-5}
Proposed	**ZSIH (64 bits)**	**0.254**	**0.340**	64 (binary)	6.5×10^{-5}	**0.220**	**0.291**	64 (binary)	7.9×10^{-5}

is additionally introduced, which is based on computing the 4096-D AlexNet [85] feature distances pre-trained on the *seen* classes for nearest neighbour search.

5.4.3.4 Results and Analysis

The *zero-shot* cross-modal retrieval mean-Average Precision (mAP) of ZSIH and several hashing baselines are given in Table 5.1, while the corresponding Precision-Recall curves (P-R curves) [128] and precision at top-100 retrieved candidates (Precision@100) scores are illustrated in Figure 5.3. The definitions of these metrics are given in Appendix A. The performance margins between ZSIH and the selected baselines are significant, suggesting the existing cross-modal hashing methods fail to handle our *zero-shot* task. ZSH [195] and SitNet [46] turn out to be the only two well-known *zero-shot* hashing model and they attain relatively better results than other baselines. However, it is originally designed for single-modal data retrieval. DSH [109] leads the SBIR performance under the conventional cross-modal hashing setting, but we observe a dramatic performance drop when extending it to the *unseen* categories. An illustrative comparison of ZSIH and DSH [109] are provided in Appendix E to demonstrate the difference of their *zero-shot* retrieval performance. ZSIH attains higher accuracy than DSH [109] in the top-10 retrieved candidates, which agrees with the results reported in Table 5.1. Figure 5.4 shows the 32-bit t-SNE [122] results of ZSIH on the training set and test set, where a clearly scattered map on the *unseen* classes can be observed. We also illustrate the retrieval performance *w.r.t.* the number of *seen* classes in Figure 5.4. It can be seen that ZSIH is able to produce acceptable retrieval performance as long as an adequate number of *seen* classes is provided to explore the semantic space.

The comparisons with SBIR and *zero-shot* baselines are shown in Table 5.2, where an akin performance margin to the one of Table 5.1 can be observed. To some extent, the SBIR baselines based on positive-negative samples, *e.g.* Siamese CNN [140] and GN Triplet [152], have the ability to generalize the learned representations to *unseen* classes. SAE [81] produces closest performance to ZSIH among the *zero-shot* learning baselines. Similar to ZSH [195], these *zero-shot* baselines suffer from the problem of mitigating the modality heterogeneity. Furthermore, most of the methods in Table 5.2 learn real-valued representations, which leads

Table 5.3: Conventional (non-*zero-shot*) SBIR mAP comparison between ZSIH and some cross-modal hashing baselines. The experimental settings are coherent to the ones proposed in [109].

Method	**Sketchy (Extended)**			**TU-Berlin (Extended)**		
	32 bits	64 bits	128 bits	32 bits	64 bits	128 bits
CMFH [28]	0.320	0.490	0.190	0.149	0.202	0.180
CMSSH [7]	0.206	0.211	0.211	0.121	0.183	0.175
SCM-Seq [205]	0.306	0.417	0.671	0.211	0.276	0.332
SCM-Orth [205]	0.346	0.536	0.616	0.217	0.301	0.263
CVH [90]	0.325	0.525	0.624	0.214	0.294	0.318
SePH[103]	0.534	0.607	0.640	0.198	0.270	0.282
DCMH [66]	0.560	0.622	0.656	0.274	0.382	0.425
DSH [109]	0.653	0.711	**0.783**	0.358	0.521	0.570
ZSIH	**0.689**	**0.730**	0.776	**0.415**	**0.562**	**0.597**

to poor retrieval efficiency when performing nearest neighbour search in the high-dimensional continuous space.

5.4.4 Conventional Fully-Supervised Category-Level SBIR

In addition to the *zero-shot* retrieval setting discussed above, we believe the conventional fully-supervised category-level SBIR setting is also essential in justifying our model plausibility. We follow the experimental settings discussed in [109] and report the mAP score on the two extended datasets of Sketchy [152] and TU-Berlin [32] in Table 5.3. The baselines for this experiment include CMFH [28], CMSSH [7], SCM [205], SePH [103], DCMH [66] and DSH [109].

It can be clearly seen from Table 5.3 that, even under the fully-supervised setting, ZSIH is able to produce state-of-the-art results. The mAP scores of ZSIH are generally higher than the ones of the best-performing DSH [109]. This suggests that ZSIH manages to recognize category-level information using the semantic decoder. These results agree with the t-SNE [122] diagram where data from different categories are clearly scattered. Although ZSIH consists of a simpler encoder structure than DSH [109], it still outperforms DSH [109] in most of our experiments.

Table 5.4: Ablation study. 64-bit mAP results of several baselines are reported.

Description	**Sketchy**	**TU**
Kron. layer $\rightarrow$ concatenation	0.228	0.207
Kron. layer $\rightarrow$ MFB [202]	0.236	0.211
Stochastic neuron $\rightarrow$ bit regularization	0.187	0.158
Decoder $\rightarrow$ classifier	0.162	0.133
Without GCNs	0.233	0.171
GCNs $\rightarrow$ word vector fusion	0.219	0.176
$t = 1$ for GCNs	0.062	0.055
$t = 10^{-6}$ for GCNs	0.241	0.202
ZSIH (full model)	**0.254**	**0.220**

5.4.5 Ablation Study

Some ablation study results are reported in this subsection to justify the plausibility of our proposed model.

5.4.5.1 Baselines

The baselines in this subsection are built by modifying some parts of the original ZSIH model. To demonstrate the effectiveness of the Kronecker layer for data fusion, we introduce two baselines by replacing the Kronecker layer [58] with the conventional feature concatenation and the Multi-modal Factorized Bilinear pooling (MFB) layer [202]. Regularizing the output bits with quantization error and bit decorrelation loss identical to [104] is also considered as a baseline in replacing the stochastic neurons [25]. The impact of the semantic-aware encoding-decoding design is evaluated by substituting a classifier for the semantic decoder. We introduce another baseline by replacing the graph convolutional layers [79] with conventional fully connected layers. Fusing the word embedding to the multi-modal network is also tested in replacement of graph convolution. The hyper-parameter t of Equation (5.6) is also analysed here. Two baselines are proposed with $t = 1$ and $t = 10^{-6}$ respectively.

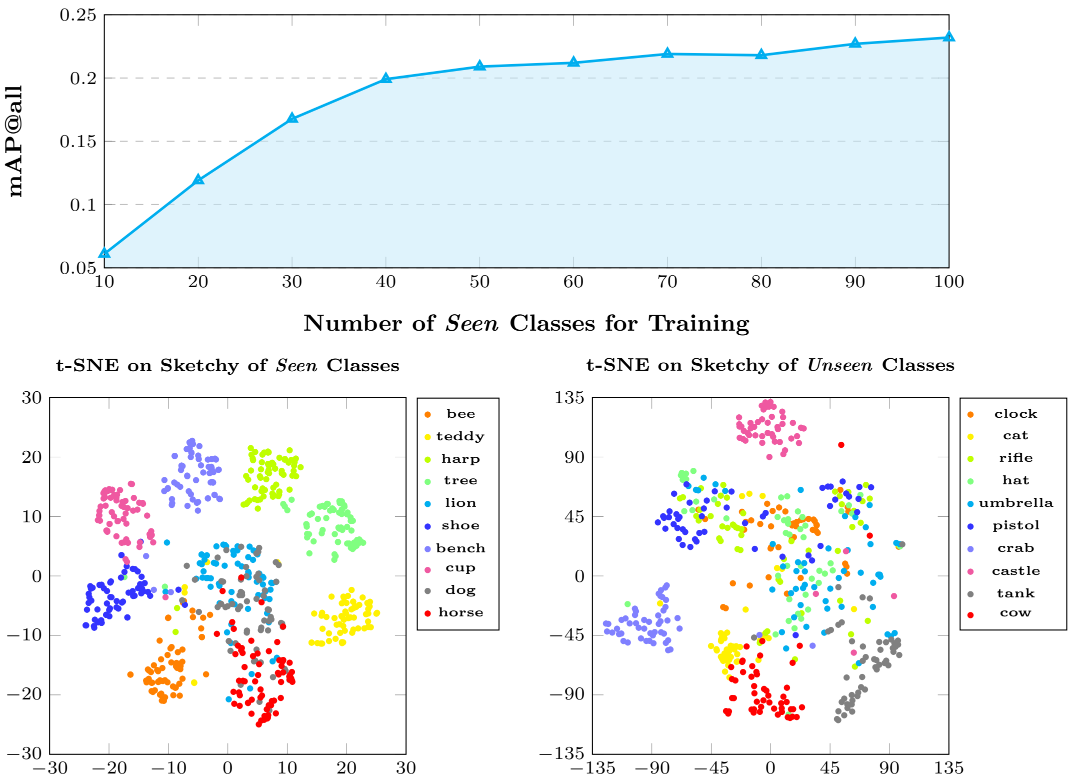


Figure 5.4: First row: 32-bit ZSIH retrieval performance on Sketchy according to different numbers of *seen* classes used during training. Second row: 32-bit t-SNE [122] scattering results on the Sketchy dataset of the *seen* and *unseen* classes.

5.4.5.2 Results and Analysis

The ablation study results are demonstrated in Table 5.4. We only report the 64-bit mAP on the two datasets for comparison in order to ensure the chapter content to be concise. It can be seen that the reported baselines typically underperform the proposed model. Both feature concatenation and MFB [202] produce reasonable retrieval performances, but the figures are still clearly lower than our original design. This is because the Kronecker layer [58] considerably expands the hidden state dimensionality, i.e. from 256-D features to 65536-D features, and thus the network produces compact hidden representations, carrying more information for cross-modal hashing. When testing the baseline of bit regularization similar to [104], we experience an unstable training procedure easily leading to overfitting. The quantization error and bit decorrelation loss introduce additional hyper-parameters to the model, making the training procedure hard. Replacing the semantic decoder with a classifier results in a dramatic performance fall as the classifier basically provides no semantic information and fails to generalize knowledge from the *seen* classes to the *unseen* ones. Graph convolutional layer [79] also plays an important role in our model. The mAP drops by about 4% when removing it. Graph convolution enhances hidden representations and knowledge within the neural network by correlating the data points that are semantically close, benefiting our *zero-shot* task. As to the hyper-parameters, a large value of t, *e.g.* $t = 1$, generally leads to a tightly-related graph adjacency, making data points from different categories hard to be recognized. On the contrary, an extreme small value t, *e.g.* $t = 10^{-6}$, suggests a sparsely-connected graph with binary-like edges, where only data points from the same category are linked. This is also suboptimal in exploring the semantic relation for *zero-shot* tasks.

5.4.6 Training Efficiency and Overfitting

We compare the training efficiency of ZSIH with some baselines in this subsection.

5.4.6.1 Overfitting

We pick three baselines here from our ablation study. The first baseline is formed by replacing the stochastic neurons with bit regularization similar to [104]. In the second baseline, the semantic decoder is replaced by a classification layer. In the last baseline, the hyper-parameter t is set to $t = 10^{-6}$.

The retrieval performance *w.r.t.* the training epochs is illustrated in Fig. 5.5. Note that, only in this experiment, we fix the CNN parameters for a clearer insight into the training efficiency trend and therefore, the figures are slightly lower than the ones reported above.

It can be observed that all selected baselines show clear performance decay immediately after hitting the best figures, while the original ZSIH model persists reasonable mAP scores along the training epochs. The proposed ZSIH model successfully learns the semantic relations between encoded images and sketches due to the utilization of the semantic decoder and GCNs [79] simultaneously. When

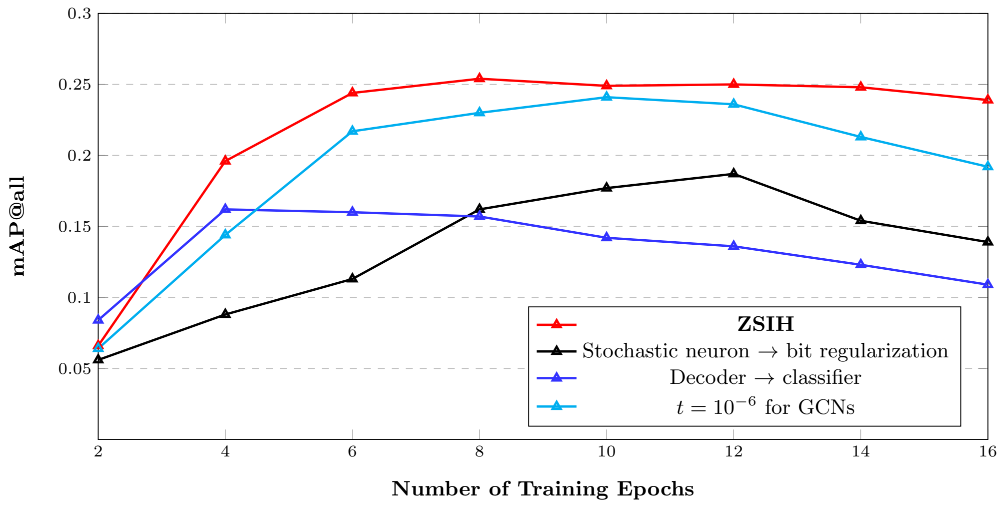


Figure 5.5: The 64-bit retrieval performance *w.r.t.* the training epochs on the Sketchy dataset [152]. Note that, only in this experiment, we fix the CNN parameters for a clearer insight into the training efficiency trend. The baselines in Table 5.4 which are not included here do not have any effect on training epochs.

replacing the semantic decoder with a classification layer, i.e. the dark blue baseline in Figure 5.5, the codes are generated only according to the label information. Therefore, the semantic similarity structure of the code space [195] is no longer preserved. As a result, this baseline fails to transfer supervised knowledge to *unseen* categories for *zero-shot* tasks. It is also clear that, with the CNNs fixed, our model attains reasonable training efficiency, where only 8 epochs are enough to reach the best retrieval performance. On the other hand, some baselines here show slower convergence speed. This phenomena also applies when the CNN parameters are set trainable.

5.4.6.2 Training Efficiency

Fig. 5.6 gives a rough illustration on the training procedure of ZSIH and most existing deep hashing models. Existing supervised [109, 66, 11, 9] or *zero-shot* hashing [195, 46] generally require alternating optimization to meet the discrete constraint. This ultimately influences the training efficiency as updating the discrete variables usually cannot be accelerated by GPU computation and may require

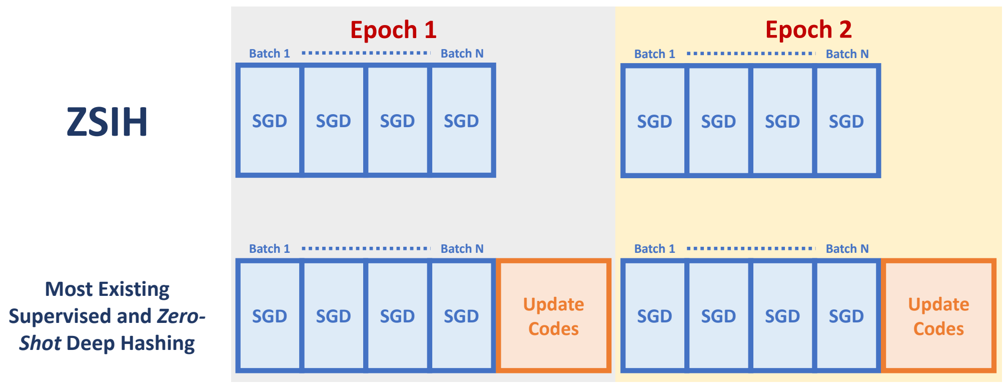


Figure 5.6: The proposed ZSIH model does not involve a trivial alternating training procedure, while existing supervised [9, 11, 66, 109] or *zero-shot* hashing [195, 46] generally require alternating optimization. Here the `Update Codes` phase in the second line refers to a computation step where the binary codes of all data are optimized on the whole training set and then are stored as target codes for the next training epoch [66, 109].

large size of memory to store them. On the other hand, due to the three-network design and the utilization of stochastic neurons, ZSIH only needs to perform SGD to obtain trained parameters, which ensure better training efficiency.

5.4.7 Qualitative Results and Failure Cases

We show more illustrative *zero-shot* SBIR results in Appendix F. As is shown in Appendix F, ZSIH attain high accuracy among the first ten retrieved candidates, which accords to the quantitative results discussed in Section 5.4.3 that ZSIH produces high-quality *zero-shot* SBIR results. On the other hand, we do experience failure cases during our experiments. Recalling the t-SNE diagram on the *unseen* classes, though most of the data are clearly scattered, some categories of data are still unintelligible from the others. This is because some sketches are drawn with bad quality and therefore, are hard to be recognized by the network. This phenomena is shown in Appendix G. Another cause is that some sketch category names are semantically close to some other concepts existing in the train/test classes, which puzzles the model, but we do not observe a severe problem caused by this.

5.5 Summary

In this chapter, a novel but realistic task of efficient large-scale *zero-shot* SBIR hashing is studied and successfully tackled by the proposed *Zero-shot* Sketch-Image Hashing (ZSIH) model. We design an end-to-end three-network deep architecture to learn shared binary representations and encode sketch/image data. Modality heterogeneity between sketches and images is mitigated by a Kronecker layer with attended features. Semantic knowledge is introduced in assistance of visual information by graph convolutions and a generative hashing scheme. Experiments suggest the proposed ZSIH model significantly outperforms existing methods in our *zero-shot* SBIR hashing task.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis mainly focuses on learning binary representations for efficient image and image-text similarity retrieval using the popular deep learning techniques. We explore three challenging cases in learning to hash, i.e. unsupervised deep hashing (Chapter 3), cross-modal hashing (Chapter 4) and *zero-shot* cross-modal hashing (Chapter 5), and respectively propose novel algorithms that outperforms the state-of-the-art.

6.1.1 Unsupervised Deep Image Hashing

In Chapter 3, we addressed the problem of unsupervised deep hashing where the existing research and models are not producing satisfying image retrieval performance. Unlike the most common solution that simply regularizes the output bits of a neural network [104, 101, 17], our Deep Variational Binaries (DVB) model is based on the popular conditional variational auto-encoders, where the latent variables are learnt in addition to the output codes. A three-network architecture is built and this approach significantly improves the retrieval performance, compared with existing models on three widely-recognized datasets.

6.1.2 Image-Sentence Cross-Modal Hashing

Conventional cross-modal hashing does not have model-specified designs in learning compact representations for informative images and sentences and thus, are not ideal for large visual-language retrieval. This problem is observed and addressed in Chapter 4 by proposing the deep cross-modal hashing namely Textual-Visual Deep Binaries (TVDB). TVDB employs an RPN-LSTM structure to model images of multi topics and a text CNN encoding long sentences, capturing informative patterns hidden in images and texts. The model is trained using a alternating optimization scheme with acceptable efficiency. Extensive experiments agree that the proposed TVDB model outperforms existing cross-modal algorithms by large margins.

6.1.3 Zero-Shot Cross-Modal Hashing

In Chapter 5, a novel problem of *zero-shot* cross-modal hashing is introduced. Existing supervised models fail to handle this situation where the test categories exclude the training ones. This problem is studied in a special category-level SBIR case. We propose the *Zero-shot* Sketch-Image Hashing (ZSIH) model, utilizing several cutting-edge deep learning structures. A shared representation of images and sketches are learnt using the Kronecker layer [58]. The GCNs [79] and generative hashing framework [25] ensure the network to be semantic-aware, suitable for *zero-shot* tasks. Experiments conducted on two datasets agree with our insight into this problem and ZSIH clearly leads *zero-shot* SBIR performance.

6.2 Open Questions and Future Work

6.2.1 Binary Representations for Video Sequences

In this thesis, we studied the problem of learning to hash with images and sentences. However, there exist other data modalities, *e.g.* video, audio, *etc.* A large number of video hashing algorithms have been proposed to retrieve video data within standard datasets, which simply focus on learning the holistic representations of the whole video. In industrial-level applications, recent years have witnessed the challenge to pick a certain temporal part or segment from a long video sequence correlated to a relatively short video query with high efficiency. This task can be considered as a combination of representation learning and video abstraction, where a temporal attention model and learning to hash could apply. We are planning to have a thorough study on this problem in the near future.

6.2.2 Improved Reparametrization Tricks for Discrete Variables

In Chapter 5, the stochastic neurons [25] for hashing are priored and reparametrized with a Multinoulli distribution. This can be typically regarded as a discrete extension of the Gaussian latent variables in the vanilla VAEs [76]. However, we observe unpredictable loss perturbation at the first two training epochs when optimizing ZSIH. This is caused by the incompact feature sampled from the Multinoulli posterior when the network is not well tuned. Although a better pre-training strategy can be of help here, it is also expected that an in-depth statistical analysis and advanced reparametrization model will largely improve the aforementioned phenomena and further promote the hashing quality. A typical example is Inverse Autoregressive Flow (IAF) [78] for continuous variables, where a discrete extension is reasonable and appreciated. We are about to have a closer insight into this in the near future.

6.2.3 Binary Representations for Fine-Grained Retrieval

In this thesis, the proposed novel models mainly focus on category-level data retrieval. This is because (a) category-level tasks are ultimately suitable for large-scale retrieval; (b) this has been a classic and conventional evaluation metric for hashing in most related literatures. However, the demand in *fine-grained* retrieval is growing, where an exact retrieval candidate is expected according to a query. For instance, *fine-grained* SBIR is now a welcomed application [161, 162] for its potential in web shopping and many other areas.

Learning *fine-grained* hash codes is challenging. *Fine-grained* retrieval basically requires highly descriptive features to distinguish one data point from the other with reasonable distances, but binary representations carry relatively less information than the continuous ones of the same length. This claims the full entropy utilization of each bit.

We are planning to study an even harder case in *fine-grained* hashing, which is *zero-shot fine-grained* SBIR. The difficulties here lie in simultaneously solving the above-mentioned challenges and the ones described in Chapter 5. Hard as the task is, we believe it can be a promising research topic.

Appendix A

Quantitative Retrieval Evaluation Measures

In this appendix, the quantitative evaluation matrices used in this thesis are elaborated.

A.1 Precision@K

Precision at top-K retrieved candidates (Precision@K) is a widely used performance metric in large-scale data retrieval. In this thesis, we respectively use Precision@5000, Precision@200 and Precision@100 in Chapter 3, 4 and 5, which can be computed by

$$Precision@K = \frac{|\{relevent\ data\} \cap \{\text{top-}K\ retrieved\ data\}|}{K}, \tag{A.1}$$

where $|\cdot|$ indicates the set size. In our experiments, we report the mean value of this score among all test queries.

A.2 Mean-Average Precision

In data retrieval, mean-Average Precision (mAP) is a key performance measure. To obtain the mAP score, the first step is to compute the retrieval average precision of each query. Given a query $\mathbf{q}$, the average precision of the retrieval result is defined by

$$AveragePrecision(\mathbf{q}) = \frac{\sum_{k=1}^{N_c}(Precision@k \times Rel(k)}{|\{relevent\ data\}|}. \tag{A.2}$$

Here N_c is the number of retrieved candidates. $Rel(k)$ is an indicator function which equals 1 if the retrieved item at rank k is relevant to the query $\mathbf{q}$, zero otherwise [168]. Therefore, the mAP of a set of queries is computed by

$$\text{mAP} = \frac{\sum_{i=1}^{N_q} AveragePrecision(\mathbf{q}_i)}{N_q}, \tag{A.3}$$

where N_q refers to the number of queries.

A.3 HD2 Precision and Recall

The precision and recall of Hamming Distance with radius 2 (HD2) also illustrate the code similarity of relevant data [109]. For each query, the HD2 precision and recall are defined as follows:

$$\begin{aligned} \text{HD2}Precision &= \frac{|\{relevent\ data\} \cap \{data\ within\ Hamming\ distance\ of\ 2\}|}{|\{data\ within\ Hamming\ distance\ of\ 2\}|}, \\ \text{HD2}Recall &= \frac{|\{relevent\ data\} \cap \{data\ within\ Hamming\ distance\ of\ 2\}|}{|\{relevent\ data\}|}. \end{aligned} \tag{A.4}$$

In our experiments, we report the mean values of these scores among all test queries.

Appendix B

Qualitative Results of DVB

In this appendix, five intuitive image retrieval cases of DVB introduced in Chapter 3 on CIFAR-10 [84] are provided. Note that the CIFAR-10 [84] dataset contains low-resolution images, i.e. 32×32. It can be clearly seen that the top-20 retrieved images are mainly subjected to the same topic as the query images.

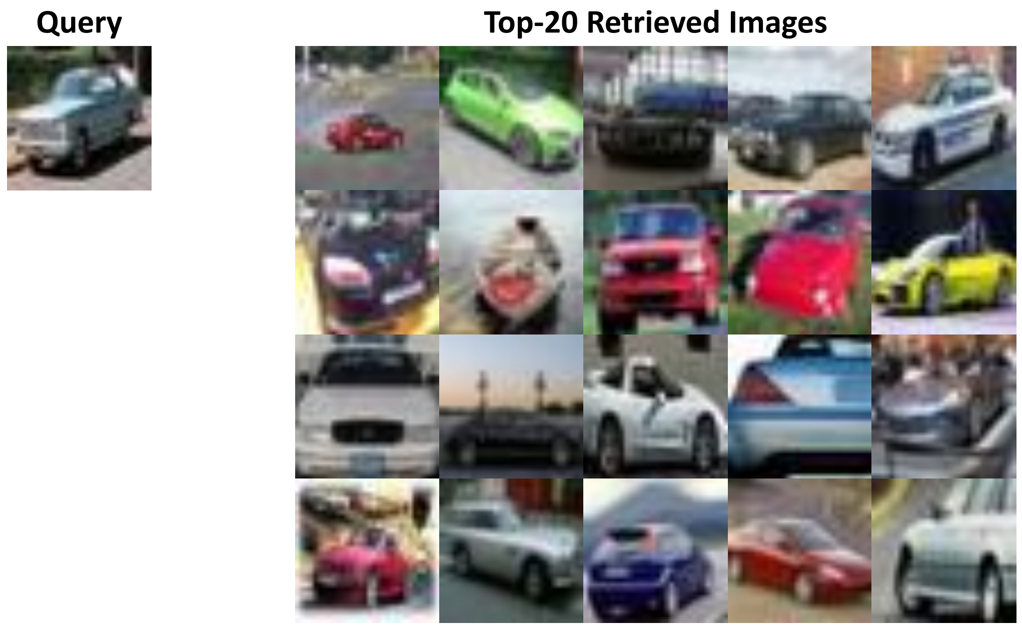


Figure B.1: Examples of top-20 32-bit image retrieval results of DVB on CIFAR-10 [84] dataset.

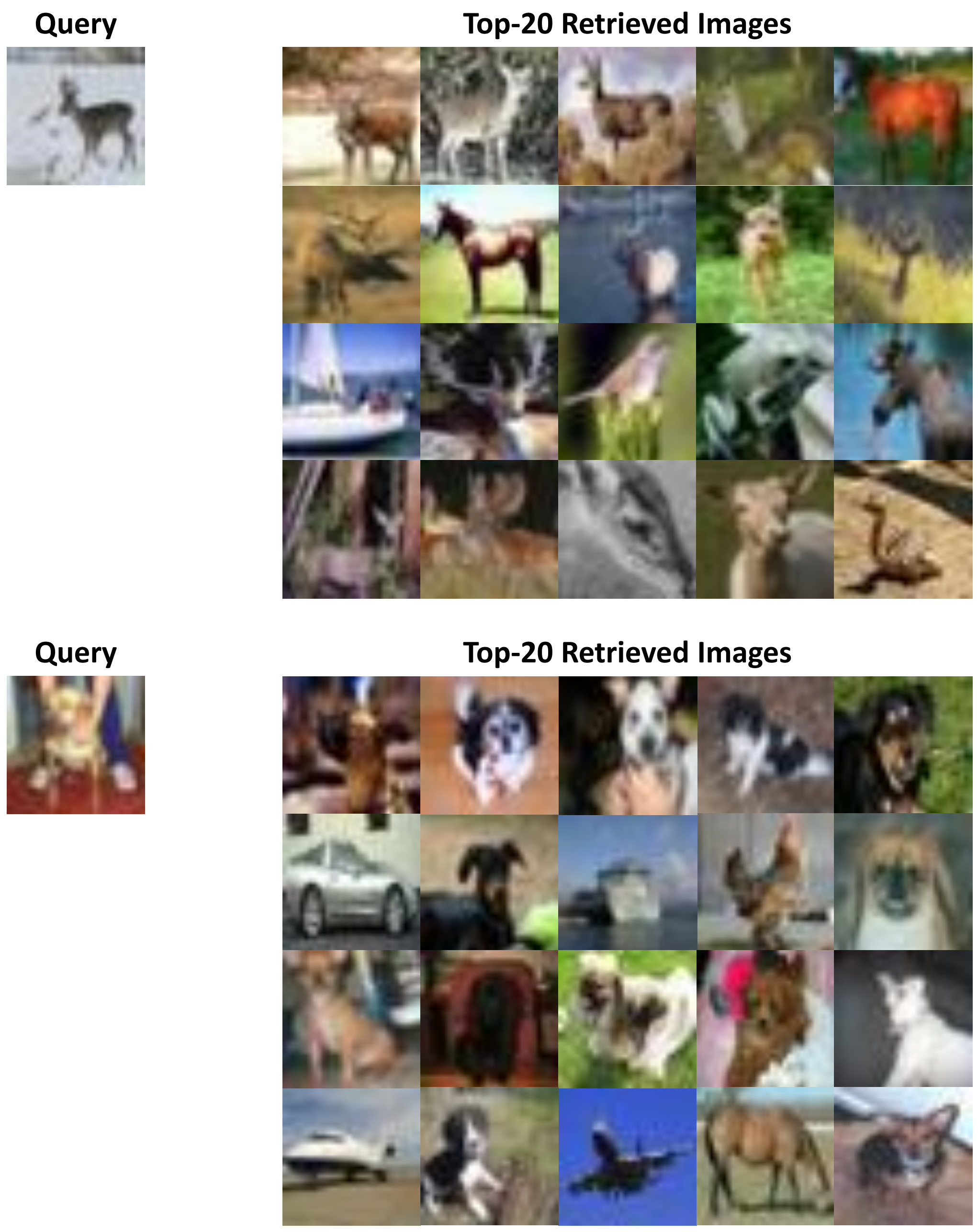
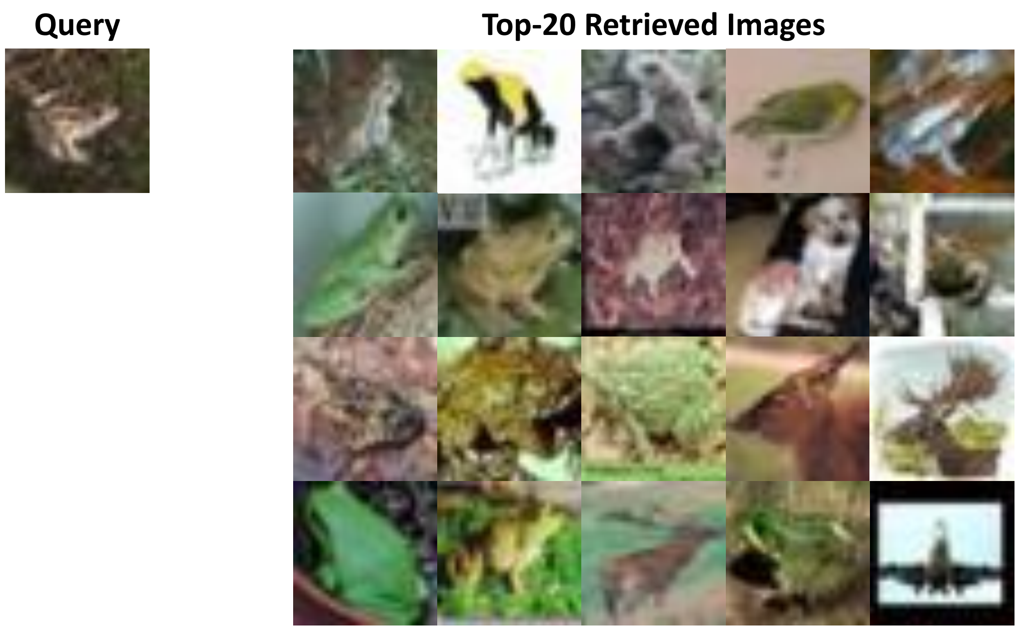


Figure B.2: Examples of top-20 32-bit image retrieval results of DVB on CIFAR-10 [84] dataset (continued).



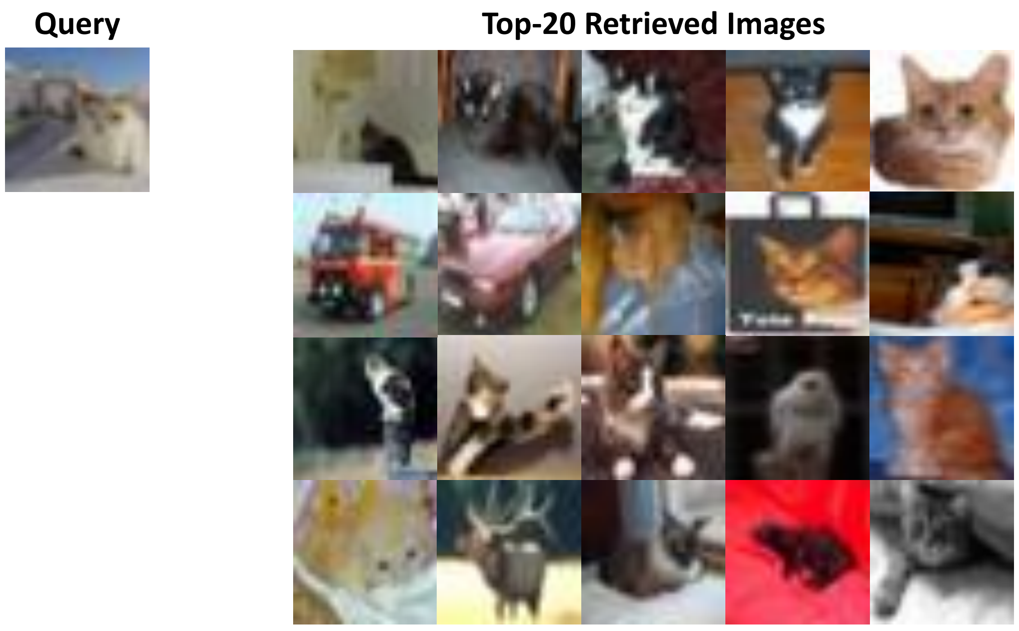


Figure B.3: Examples of top-20 32-bit image retrieval results of DVB on CIFAR-10 [84] dataset (continued).

Appendix C

Qualitative Results of TVDB

In this appendix, eight intuitive *Sentence Query Image* cases of TVDB introduced in Chapter 4 on Microsoft COCO [102] are provided.

We show results with $M = 128$ to make the comparison illustrative, because TVDB outperforms the compared baselines with largest margin using this code length. As is shown in Figure C.1,C.2,C.3,C.4,C.5,C.6,C.7,C.8, TVDB retrieves the best matching candidates, where most of the objects mentioned in the query are included. This agrees the quantitative analysis provided in Chapter 4 that TVDB produces compact binary codes for cross-modal retrieval.

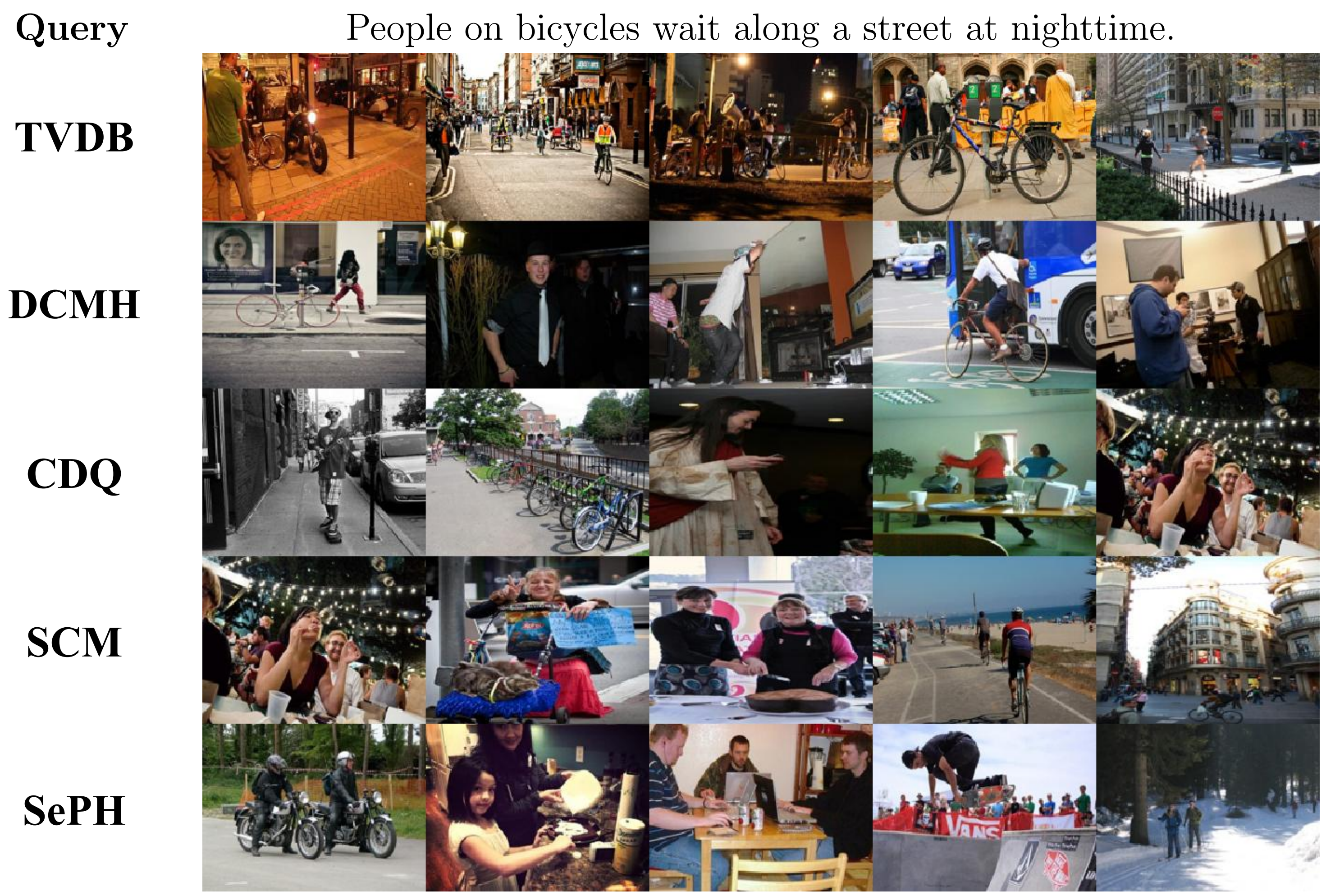


Figure C.1: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102].

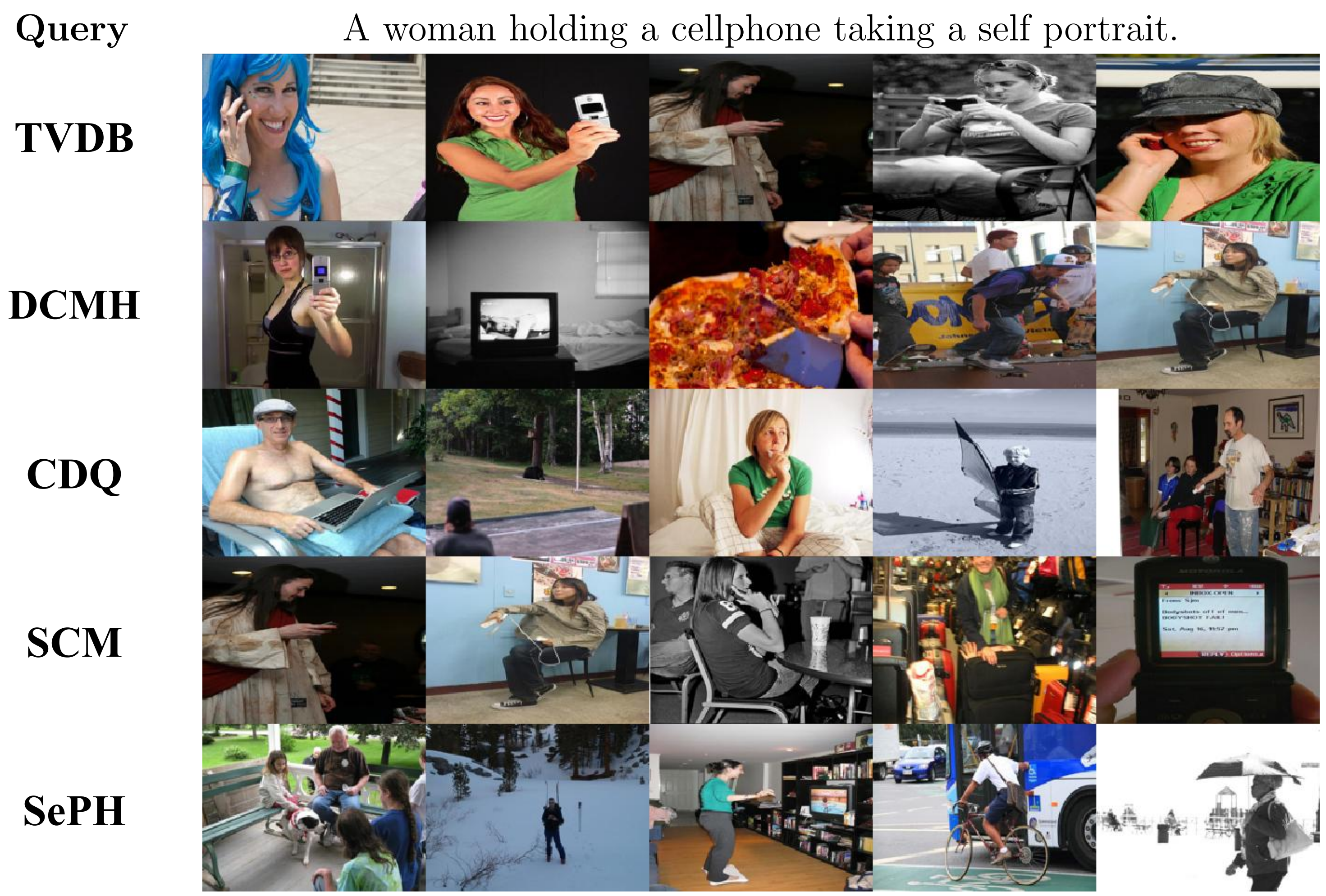


Figure C.2: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

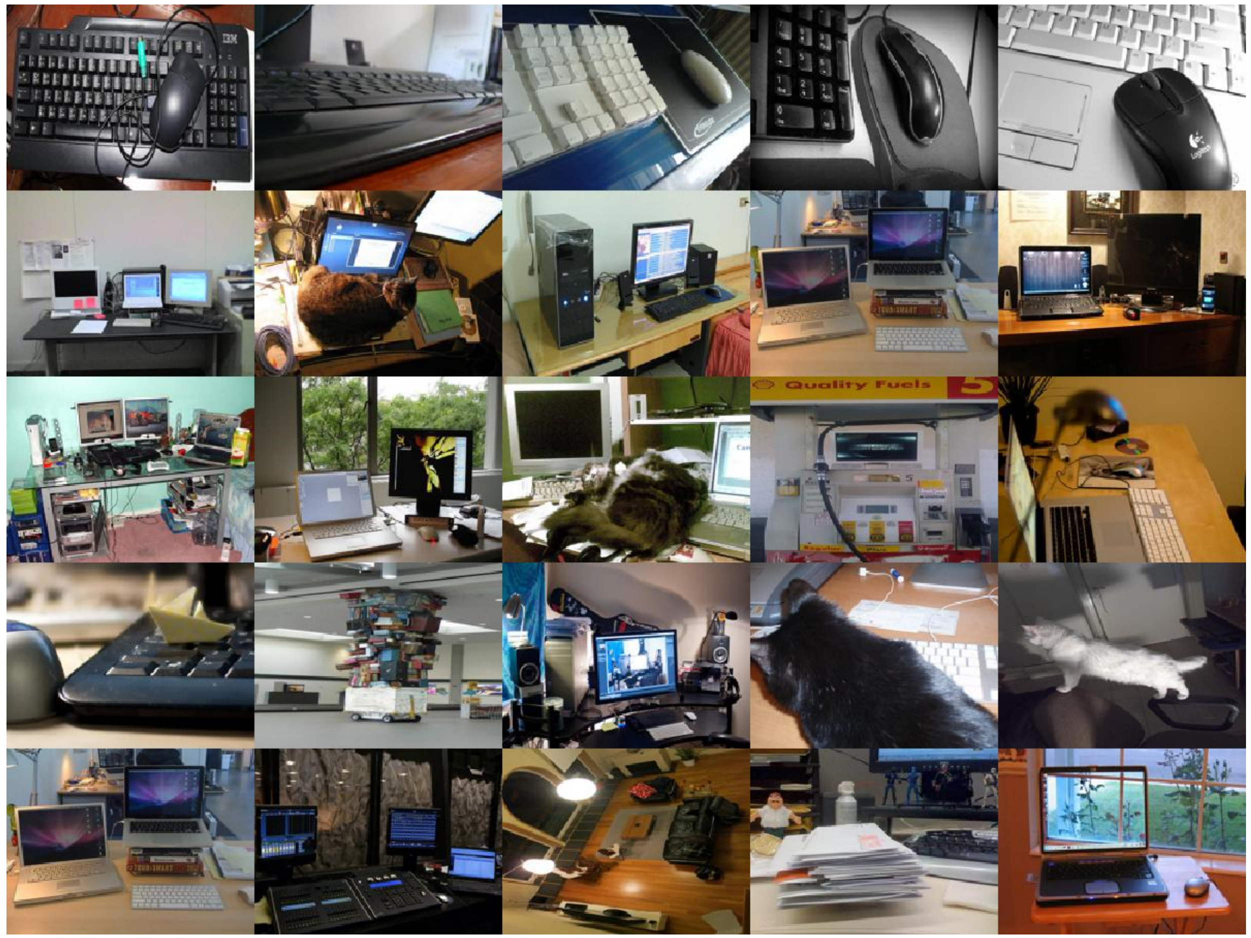


Figure C.3: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

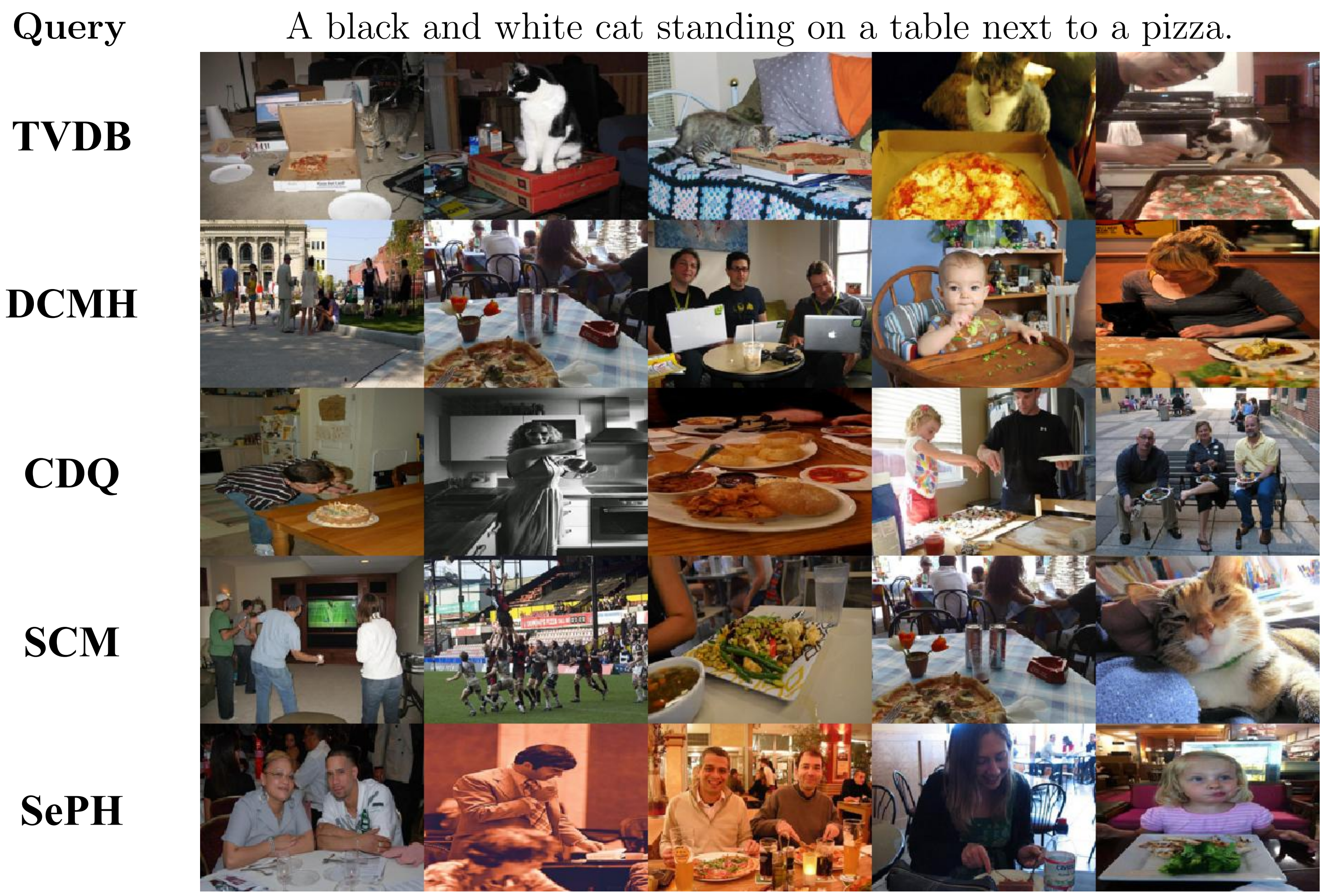


Figure C.4: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

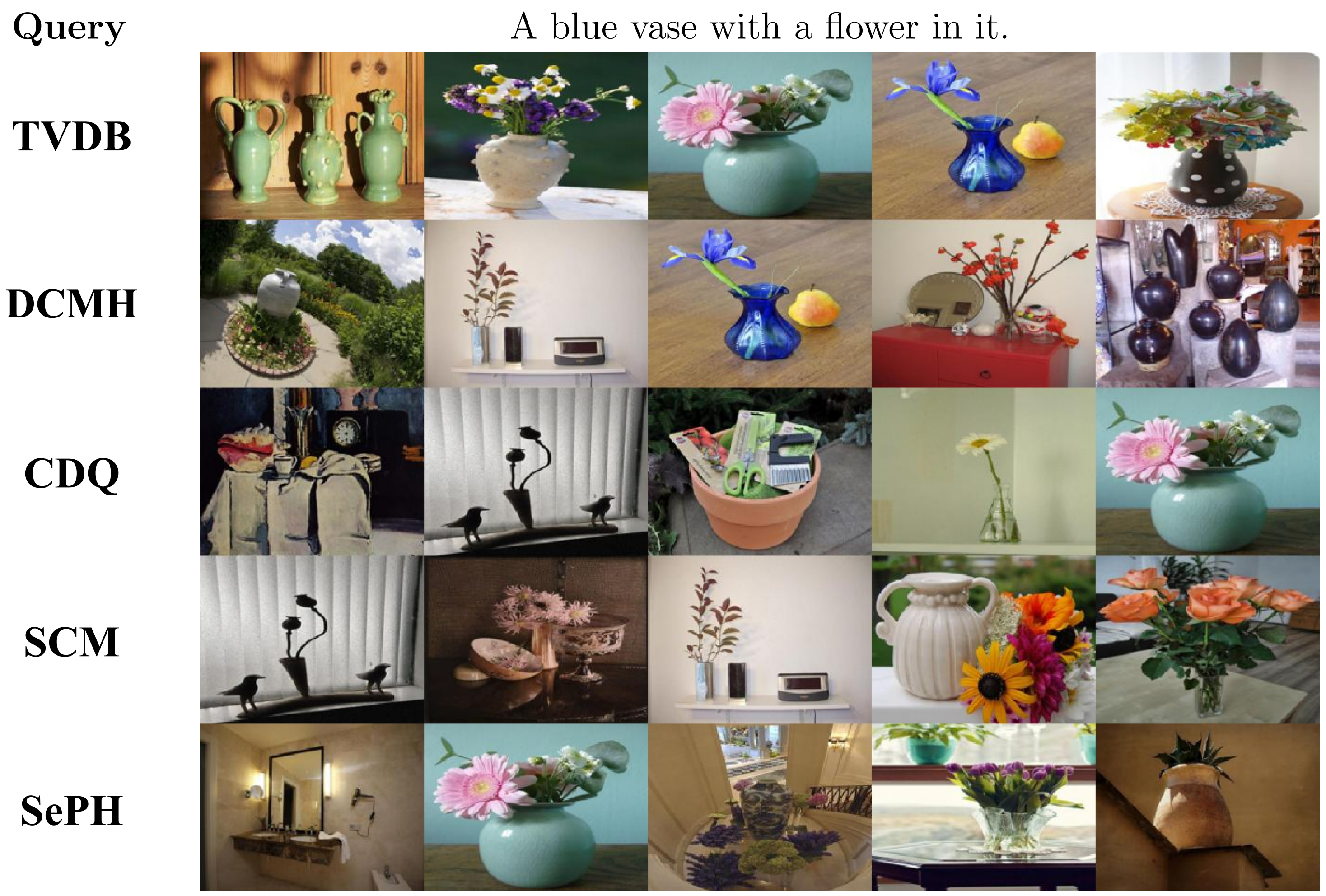


Figure C.5: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

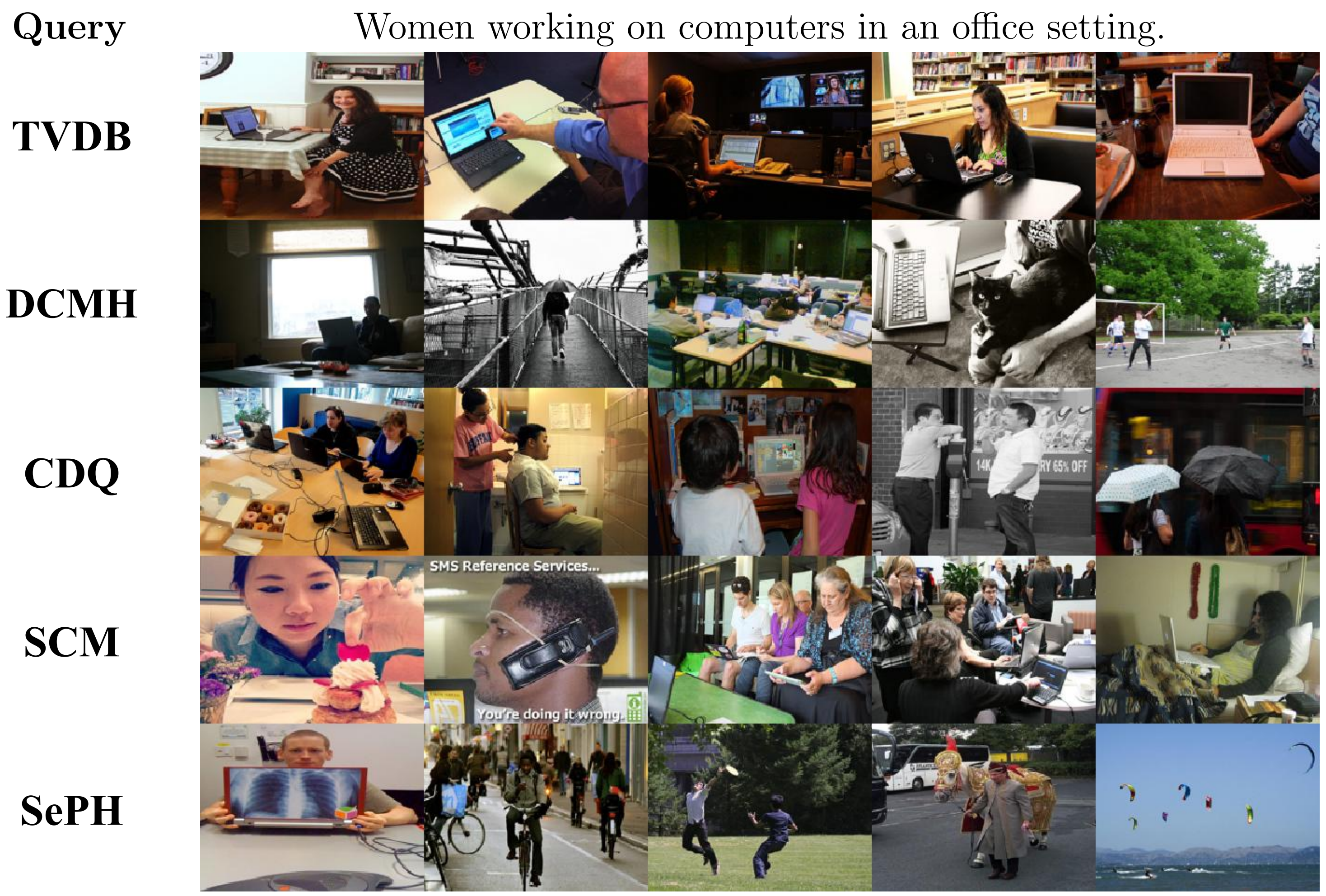


Figure C.6: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

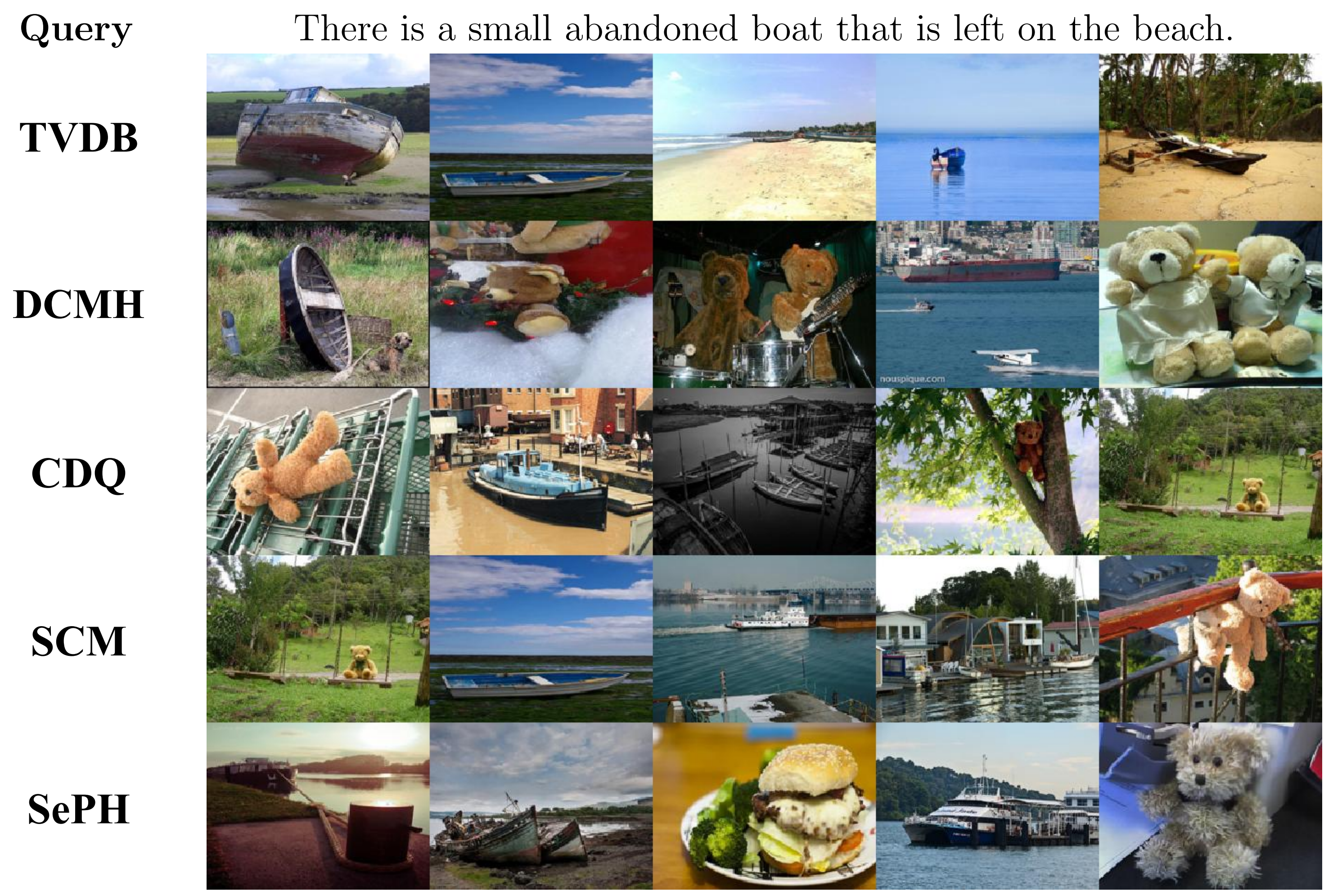


Figure C.7: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

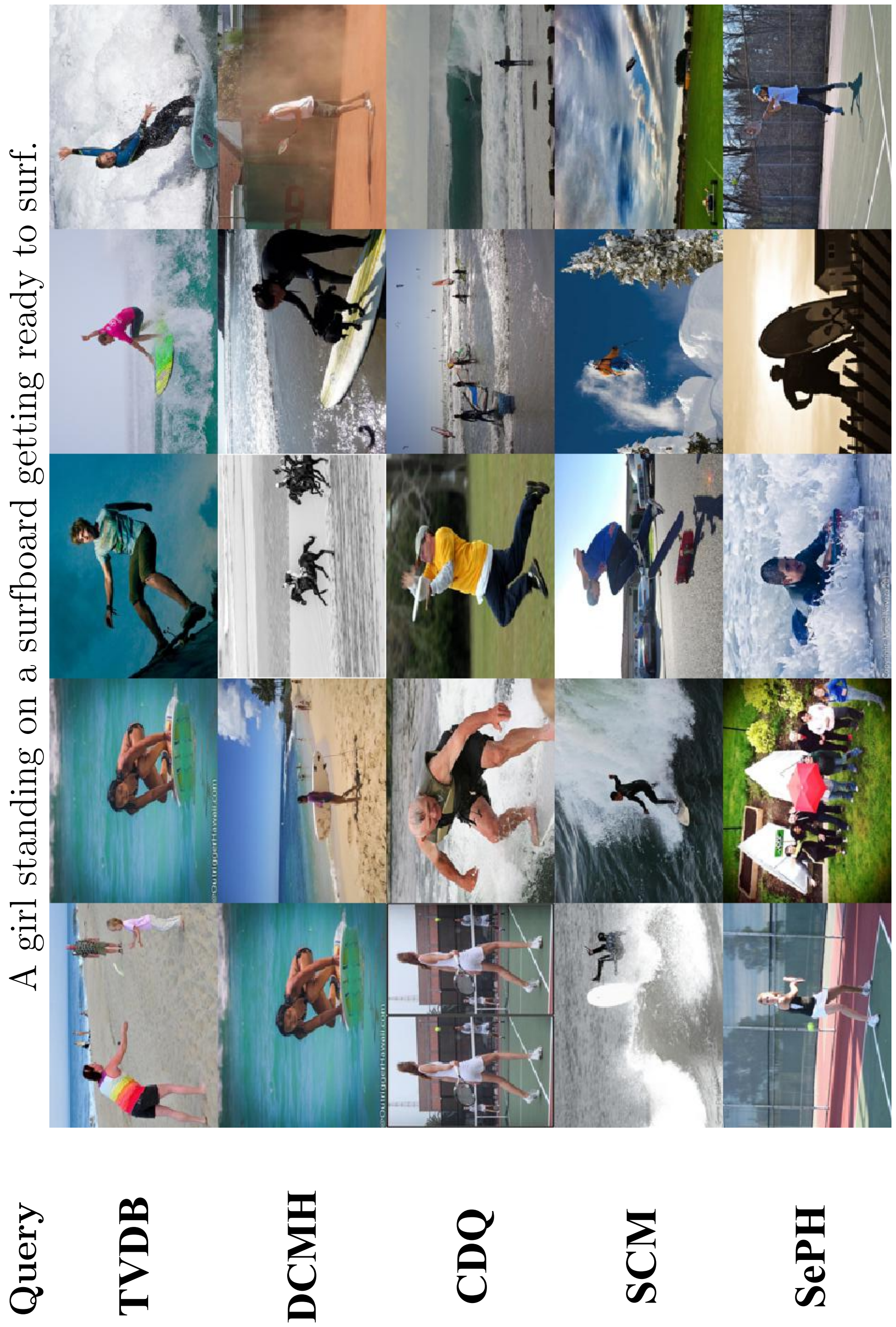
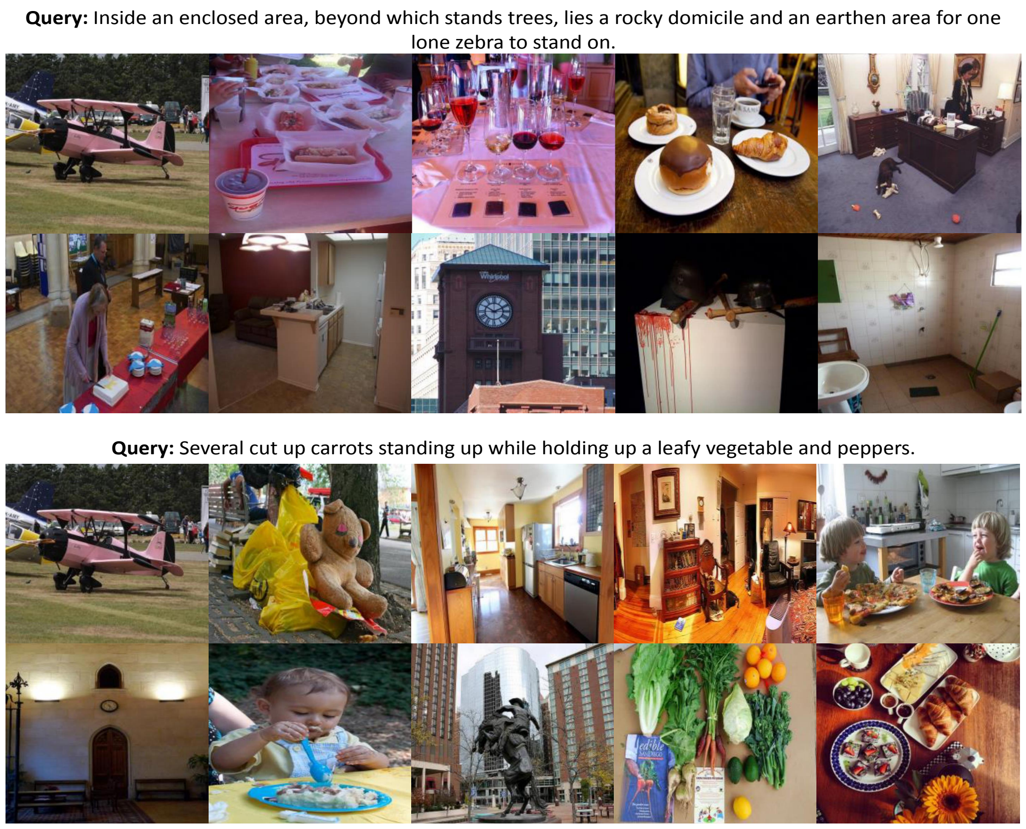


Figure C.8: Intuitive 128-bit top-5 *Sentence Query Image* results (ranked from left to right) on Microsoft COCO [102] (continued).

Appendix D

Failure Cases of TVDB

In this appendix, two failure retrieval cases of TVDB discussed in Chapter 4 are shown.



Query: Inside an enclosed area, beyond which stands trees, lies a rocky domicile and an earthen area for one lone zebra to stand on.

Query: Several cut up carrots standing up while holding up a leafy vegetable and peppers.

Figure D.1: Two failure cases of TVDB with top-10 retrieved candidates on Microsoft COCO [102].

Appendix E

Qualitative Comparison between ZSIH and DSH

In this appendix, Three intuitive *zero-shot* SBIR retrieval cases of ZSIH introduced in Chapter 5 are reported and compared with DSH [109].

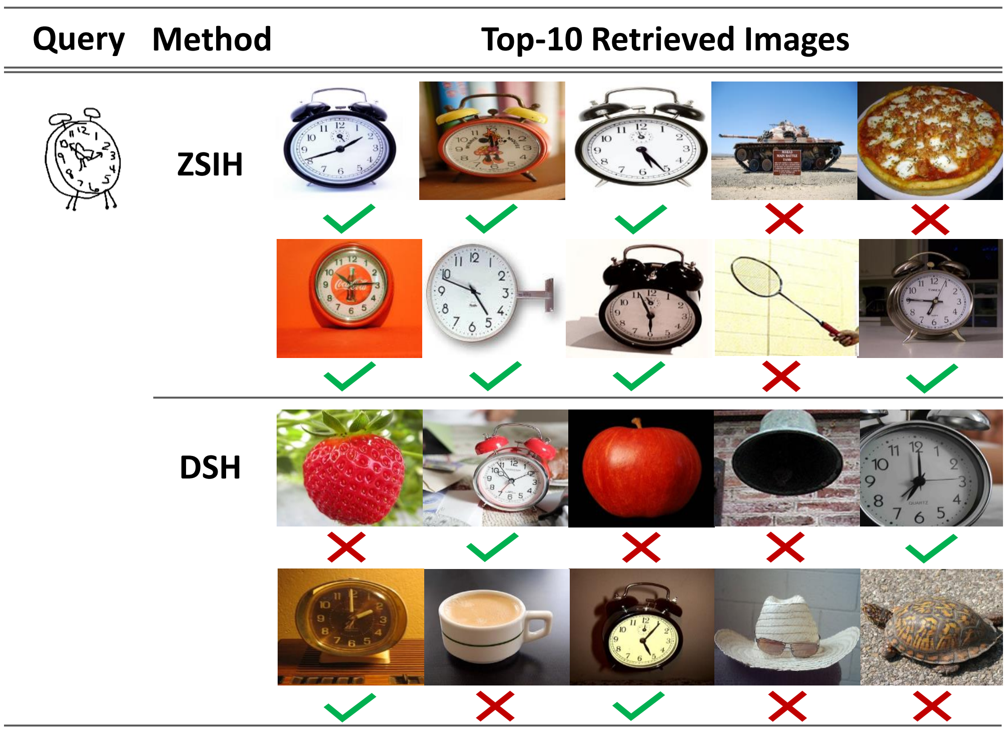
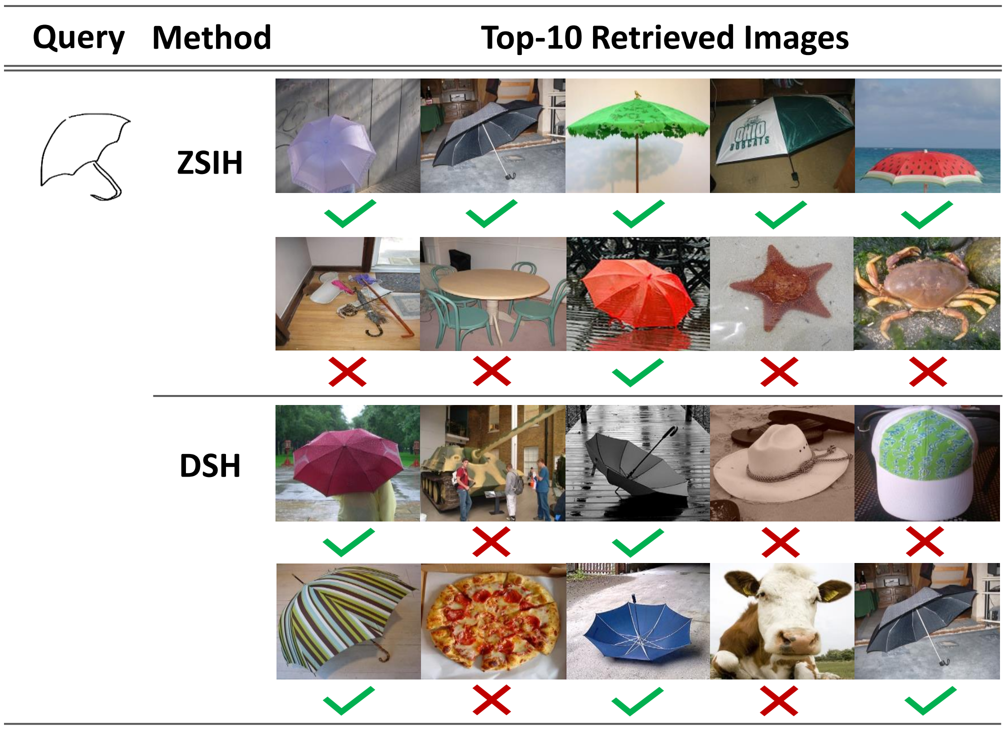


Figure E.1: Top-10 *zero-shot* SBIR results of ZSIH and DSH [109] are shown here according to the hamming distances, where the green ticks indicate correct retrieval candidates and red crosses indicate the wrong ones.



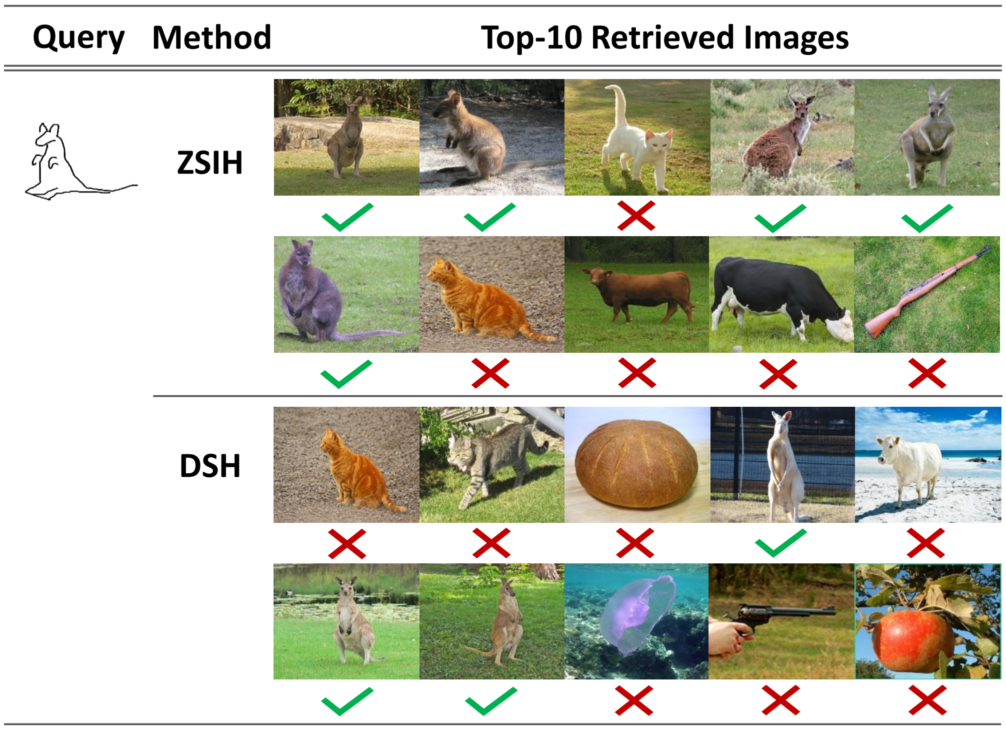


Figure E.2: Top-10 *zero-shot* SBIR results of ZSIH and DSH [109] are shown here according to the hamming distances, where the green ticks indicate correct retrieval candidates and red crosses indicate the wrong ones (continued).

Appendix F

More Qualitative Results of ZSIH

In this appendix, More successful *zero-shot* SBIR retrieval cases of ZSIH introduced in Chapter 5 are demonstrated with top-10 retrieved candidates on the Sketchy dataset [152].

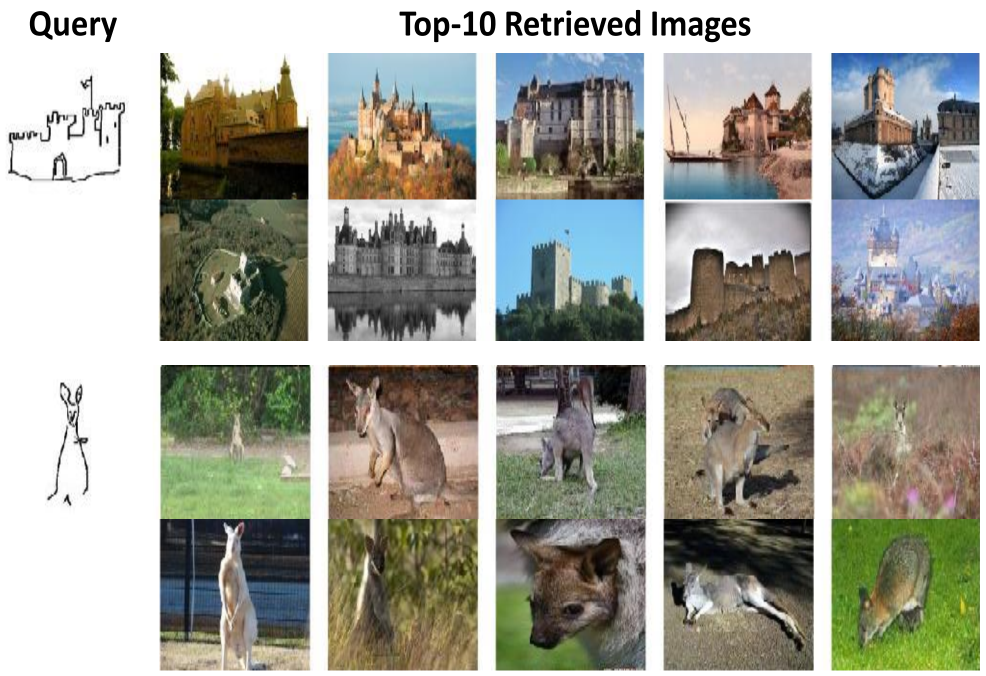


Figure F.1: Successful *zero-shot* retrieval results on the Sketchy dataset [152] of ZSIH.

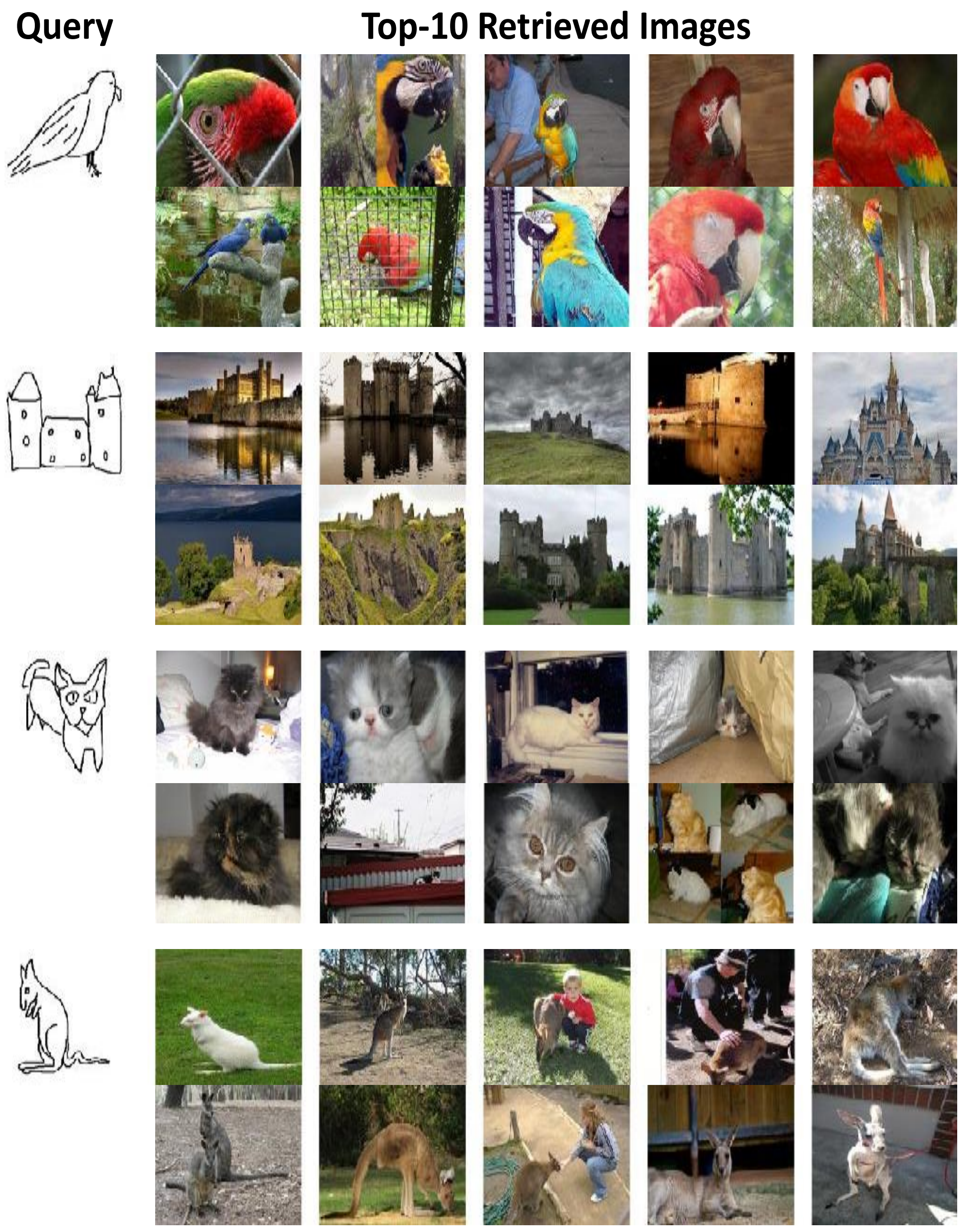


Figure F.2: Successful *zero-shot* retrieval results on the Sketchy dataset [152] of ZSIH (continued).

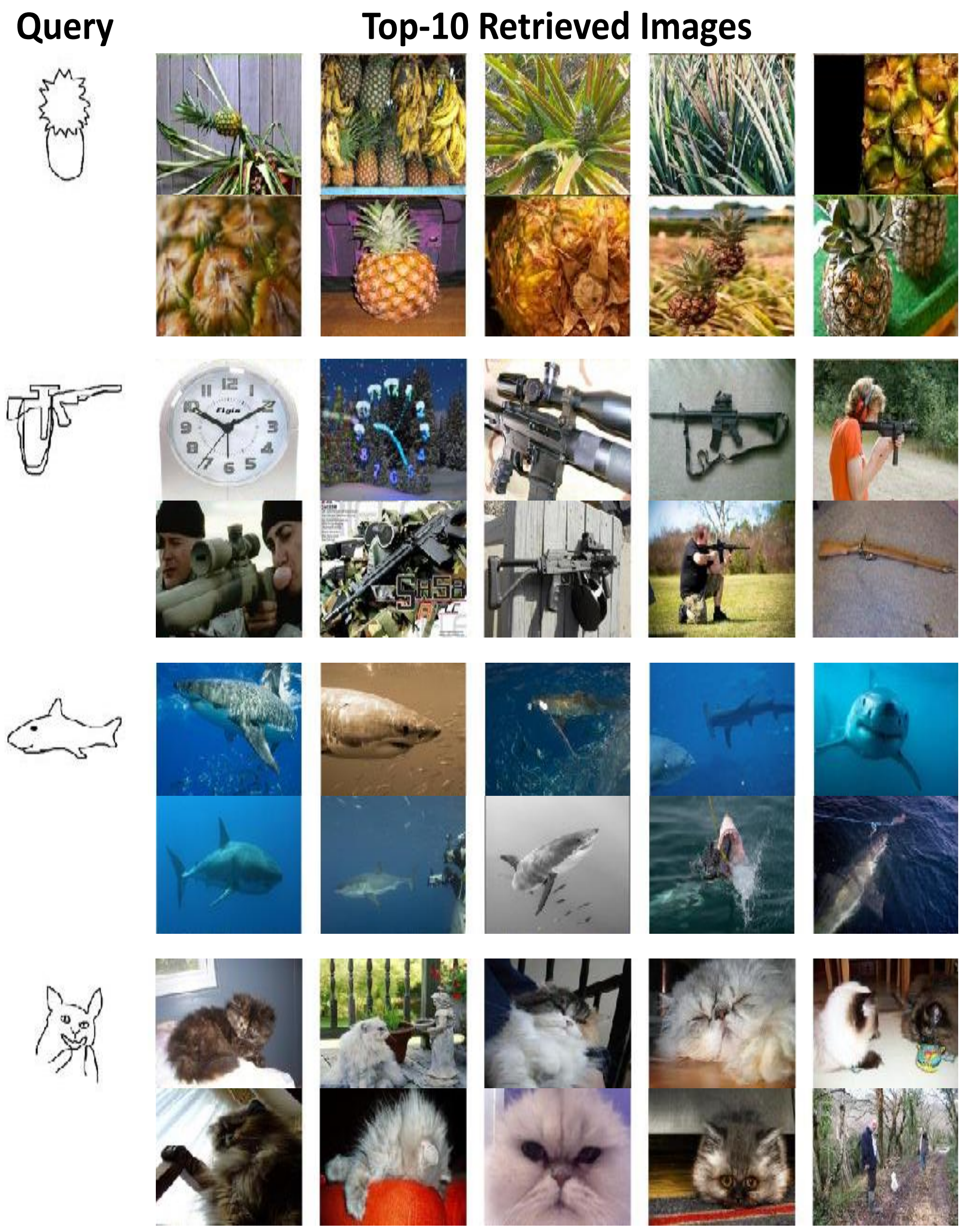


Figure F.3: Successful *zero-shot* retrieval results on the Sketchy dataset [152] of ZSIH (continued).

Appendix G

Failure Cases of ZSIH

In this appendix, three failure *zero-shot* SBIR cases of ZSIH on the Sketchy dataset [152] discussed in Chapter 5 are illustrated.

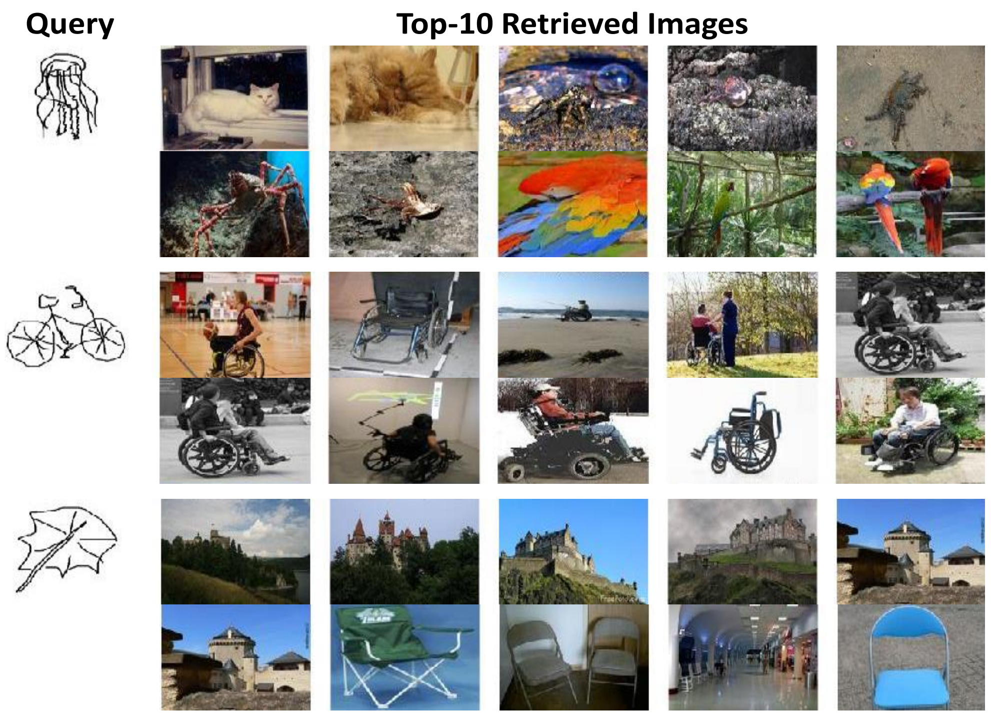


Figure G.1: Failure cases of ZSIH on the Sketchy dataset [152].

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016. 34, 63, 78, 97

[2] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 86

[3] Z. Al-Halah, M. Tapaswi, and R. Stiefelhagen, "Recovering the missing link: Predicting class-attribute associations for unsupervised zero-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 86

[4] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "Vqa: Visual question answering," in *IEEE International Conference on Computer Vision(ICCV)*, 2015. 47, 52

[5] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems (NIPS)*, 2001. 30

[6] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *International Conference on Machine Learning (ICML)*, 2010. 14

[7] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing,"

in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 12, 47, 53, 64, 65, 70, 71, 82, 85, 99, 101, 104

[8] F. Cakir, K. He, S. Adel Bargal, and S. Sclaroff, "Mihash: Online hashing with mutual information," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 15

[9] Y. Cao, M. Long, J. Wang, and S. Liu, "Collective deep quantization for efficient cross-modal retrieval." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017. ix, 49, 53, 54, 62, 64, 65, 70, 71, 72, 75, 82, 85, 96, 108, 109

[10] ——, "Deep visual-semantic quantization for efficient image retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 15

[11] Y. Cao, M. Long, J. Wang, Q. Yang, and P. S. Yu, "Deep visual-semantic hashing for cross-modal retrieval," in *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2016. ix, 47, 49, 53, 54, 64, 65, 66, 71, 72, 74, 82, 85, 108, 109

[12] Y. Cao, M. Long, J. Wang, and P. S. Yu, "Correlation hashing network for efficient cross-modal retrieval," in *British Machine Vision Conference (BMVC)*, 2017. 85

[13] Y. Cao, M. Long, J. Wang, and H. Zhu, "Correlation autoencoder hashing for supervised cross-modal search," in *ACM International Conference on Multimedia Retrieval (ICMR)*, 2016. 53

[14] Y. Cao, M. Long, J. Wang, H. Zhu, and Q. Wen, "Deep quantization network for efficient image retrieval." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2016. 16, 19, 22, 29, 53

[15] Z. Cao, M. Long, J. Wang, and P. S. Yu, "Hashnet: Deep learning to hash by continuation," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 15

[16] ——, "Hashnet: Deep learning to hash by continuation," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 87

[17] M. A. Carreira-Perpinán and R. Raziperchikolaei, "Hashing with binary autoencoders," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 19, 111

[18] S. Chaidaroon and Y. Fang, "Variational deep semantic hashing for text documents," in *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2017. 16, 23, 24, 26, 33

[19] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 86

[20] S. Changpinyo, W.-L. Chao, and F. Sha, "Predicting visual exemplars of unseen classes for zero-shot learning," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 86

[21] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *ACM Symposium on Theory of Computing (STOC)*, 2002. 7, 21, 35, 36, 37, 53

[22] 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," in *ACM International Conference on Image and Video Retrieval (CIVR)*, 2009. ii, vii, xi, 34, 37, 42, 43

[23] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for natural language processing," *arXiv preprint arXiv:1606.01781*, 2016. 14, 50, 52, 58

[24] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995. 10

[25] B. Dai, R. Guo, S. Kumar, N. He, and L. Song, "Stochastic generative hashing," in *International Conference on Machine Learning (ICML)*, 2017. 16, 91, 93, 95, 96, 105, 112, 113

[26] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems (NIPS)*, 2016. 91, 92

[27] B. Demirel, R. G. Cinbis, and N. I. Cinbis, "Attributes2classname: A discriminative model for attribute-based unsupervised zero-shot learning," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 86

[28] G. Ding, Y. Guo, and J. Zhou, "Collective matrix factorization hashing for multimodal data," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 10, 11, 47, 53, 64, 65, 70, 71, 82, 85, 99, 101, 104

[29] Z. Ding, M. Shao, and Y. Fu, "Low-rank embedded ensemble semantic dictionary for zero-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 86

[30] T.-T. Do, A.-D. Doan, and N.-M. Cheung, "Learning to hash with binary deep neural network," in *European Conference on Computer Vision (ECCV)*, 2016. 16, 19, 23, 29, 35, 36, 37, 38, 40, 41, 53

[31] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 47, 52, 58

[32] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *ACM Transactions on Graphics*, vol. 31, no. 4, pp. 44–1, 2012. iv, 97, 98, 101, 104

[33] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa, "An evaluation of descriptors for large-scale image retrieval from sketched feature lines," *Computers & Graphics*, vol. 34, no. 5, pp. 482–498, 2010. 82, 85

[34] ——, “Sketch-based image retrieval: Benchmark and bag-of-features descriptors,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 11, pp. 1624–1636, 2011. 82, 85

[35] V. Erin Liong, J. Lu, Y.-P. Tan, and J. Zhou, “Cross-modal deep variational hashing,” in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 85

[36] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton *et al.*, “Attend, infer, repeat: Fast scene understanding with generative models,” in *Advances in Neural Information Processing Systems (NIPS)*, 2016. 23

[37] F. Feng, X. Wang, and R. Li, “Cross-modal retrieval with correspondence autoencoder,” in *ACM international conference on Multimedia (MM)*, 2014. 49, 53, 64, 65, 71

[38] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov *et al.*, “DeViSE: A deep visual-semantic embedding model,” in *Advances in Neural Information Processing Systems (NIPS)*, 2013. 47, 86, 91, 101, 102

[39] K. Fukushima, “Neocognitron: A hierarchical neural network capable of visual pattern recognition,” *Neural networks*, vol. 1, no. 2, pp. 119–130, 1988. 14

[40] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu, “Are you talking to a machine? dataset and methods for multilingual image question,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015. 47, 52

[41] A. Gionis, P. Indyk, R. Motwani *et al.*, “Similarity search in high dimensions via hashing,” in *nternational Conference on Very Large Data Bases (VLDB)*, 1999. 53

[42] R. Girshick, “Fast R-CNN,” in *IEEE International Conference on Computer Vision(ICCV)*, 2015. 51, 57

[43] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, “Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013. 6, 7, 18, 19, 21, 22, 35, 36, 37, 53

[44] ——, “Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013. 6, 18

[45] M. Grubinger, P. Clough, H. Müller, and T. Deselaers, “The iapr tc-12 benchmark: A new evaluation resource for visual information systems,” in *International Workshop OntoImage*, 2006. iii, 63, 66

[46] Y. Guo, G. Ding, J. Han, and Y. Gao, “SitNet: Discrete similarity transfer network for zero-shot hashing,” in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. ix, 87, 96, 99, 101, 102, 103, 108, 109

[47] Y. Guo, G. Ding, L. Liu, J. Han, and L. Shao, “Learning to hash with optimized anchor embedding for scalable retrieval,” *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1344–1354, 2017. 18

[48] D. K. Hammond, P. Vandergheynst, and R. Gribonval, “Wavelets on graphs via spectral graph theory,” *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011. 92

[49] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009. vii, 38, 39, 69

[50] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask r-cnn,” in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 14

[51] K. He, F. Wen, and J. Sun, “K-means hashing: An affinity-preserving quantization method for learning binary compact codes,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 6, 18

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 14

[53] X. He and P. Niyogi, "Locality preserving projections," in *Advances in Neural Information Processing Systems (NIPS)*, 2003. 30

[54] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 8, 35, 37, 53

[55] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006. 13

[56] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 58

[57] B. Hu, Z. Lu, H. Li, and Q. Chen, "Convolutional neural network architectures for matching natural language sentences," in *Advances in Neural Information Processing Systems (NIPS)*, 2014. 50, 52, 58

[58] G. Hu, Y. Hua, Y. Yuan, Z. Zhang, Z. Lu, S. S. Mukherjee, T. M. Hospedales, N. M. Robertson, and Y. Yang, "Attribute-enhanced face recognition with neural tensor fusion networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. viii, 88, 90, 91, 92, 105, 106, 112

[59] R. Hu, H. Xu, M. Rohrbach, J. Feng, K. Saenko, and T. Darrell, "Natural language object retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 52

[60] R. Hu and J. Collomosse, "A performance evaluation of gradient field hog descriptor for sketch based image retrieval," *Computer Vision & Image Understanding*, vol. 117, no. 7, pp. 790–806, 2013. 82, 85

[61] R. Hu, T. Wang, and J. Collomosse, "A bag-of-regions approach to sketch-based image retrieval," in *IEEE International Conference on Image Processing (ICIP)*, 2011. 82, 85

[62] Y. Hu, Z. Jin, H. Ren, D. Cai, and X. He, "Iterative multi-view hashing for cross media indexing," in *ACM international conference on Multimedia (MM)*, 2014. 12, 47, 53

[63] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456. 14

[64] H. Jain, J. Zepeda, P. Perez, and R. Gribonval, "Subic: A supervised, structured binary code for image search," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 15

[65] H. Jiang, R. Wang, S. Shan, Y. Yang, and X. Chen, "Learning discriminative latent attributes for zero-shot classification," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 86

[66] Q.-Y. Jiang and W.-J. Li, "Deep cross-modal hashing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. ix, xi, 16, 22, 47, 49, 53, 54, 62, 64, 65, 70, 71, 72, 75, 76, 77, 78, 82, 85, 96, 104, 108, 109

[67] X. Jiang, F. Wu, X. Li, Z. Zhao, W. Lu, S. Tang, and Y. Zhuang, "Deep compositional cross-modal learning to rank via local-global alignment," in *ACM international conference on Multimedia (MM)*, 2015. 47

[68] J. Johnson, A. Karpathy, and L. Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 47, 52, 58

[69] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. A. Shamma, M. S. Bernstein, and L. Fei-Fei, "Image retrieval using scene graphs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 52

[70] I. T. Jolliffe, “Principal component analysis and factor analysis,” in *Principal component analysis*. Springer, 1986, pp. 115–128. 7

[71] N. Karessli, Z. Akata, B. Schiele, and A. Bulling, “Gaze embeddings for zero-shot image classification,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 86

[72] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 51, 52

[73] A. Karpathy, A. Joulin, and F. F. Li, “Deep fragment embeddings for bidirectional image sentence mapping,” in *Advances in Neural Information Processing Systems (NIPS)*, 2014. 47, 52

[74] Y. Kim, “Convolutional neural networks for sentence classification,” in *Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2014. vi, 14, 50, 52, 58, 59, 79

[75] D. Kingma and J. Ba, “Adam: A method for acm symposium on theory of computing (stoc)hastic optimization,” in *International Conference on Learning Representations (ICLR)*, 2015. 33, 62, 95

[76] D. Kingma and M. Welling, “Auto-encoding variational bayes,” in *International Conference on Learning Representations (ICLR)*, 2014. 3, 16, 20, 23, 26, 28, 33, 36, 94, 95, 113

[77] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, “Semi-supervised learning with deep generative models,” in *Advances in Neural Information Processing Systems (NIPS)*, 2014. 23

[78] D. P. Kingma, T. Salimans, and M. Welling, “Improving variational inference with inverse autoregressive flow,” in *International Conference on Learning Representations (ICLR)*, 2017. 113

[79] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017. viii, 88, 91, 92, 93, 96, 105, 107, 112

[80] B. Klein, G. Lev, G. Sadeh, and L. Wolf, "Fisher vectors derived from hybrid gaussian-laplacian mixture models for image annotation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 52

[81] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 84, 86, 91, 101, 102, 103

[82] W. Kong and W.-J. Li, "Isotropic hashing," in *Advances in Neural Information Processing Systems (NIPS)*, 2012. 6, 18

[83] J. Krapac, M. Allan, J. Verbeek, and F. Juried, "Improving web image search results using query-relative classifiers," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. iii, 63, 69, 72

[84] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009. ii, vii, ix, xi, 34, 36, 37, 40, 41, 42, 43, 45, 117, 118, 119

[85] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105. 14, 51, 57, 58, 63, 71, 74, 89, 90, 97, 98, 103

[86] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in Neural Information Processing Systems (NIPS)*, 2009. 6, 9, 18

[87] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *IEEE International Conference on Computer Vision (ICCV)*, 2009. 7, 8, 53

[88] B. Kulis, P. Jain, and K. Grauman, "Fast similarity search for learned metrics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2143–2157, 2009. 9

[89] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, "Deep convolutional inverse graphics network," in *Advances in Neural Information Processing Systems (NIPS)*, 2015. 20, 23, 36

[90] S. Kumar and R. Udupa, "Learning hash functions for cross-view similarity search," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011. 10, 11, 47, 53, 64, 65, 70, 71, 82, 85, 99, 101, 104

[91] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 19

[92] ——, "Simultaneous feature learning and hash coding with deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 64, 65, 71

[93] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 453–465, 2014. 86

[94] 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. 14

[95] 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. 14, 22, 51

[96] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 14

[97] W.-J. Li, S. Wang, and W.-C. Kang, "Feature learning based deep supervised hashing with pairwise labels," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. 15

[98] X. Li, G. Lin, C. Shen, A. Hengel, and A. Dick, "Learning hash functions using column generation," in *International Conference on Machine Learning (ICML)*, 2013. 10

[99] Y. Li, D. Wang, H. Hu, Y. Lin, and Y. Zhuang, "Zero-shot recognition using dual visual-semantic mapping paths," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 86

[100] J. J. Lim, C. L. Zitnick, and P. Dollar, "Sketch tokens: A learned mid-level representation for contour and object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 87

[101] K. Lin, J. Lu, C.-S. Chen, and J. Zhou, "Learning compact binary descriptors with unsupervised deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 16, 19, 23, 29, 35, 36, 37, 38, 40, 41, 111

[102] 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," in *European Conference on Computer Vision (ECCV)*, 2014. iii, viii, ix, x, 63, 66, 69, 72, 76, 80, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129

[103] Z. Lin, G. Ding, M. Hu, and J. Wang, "Semantics-preserving hashing for cross-view retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 12, 47, 52, 53, 64, 65, 66, 70, 71, 82, 85, 99, 101, 104

[104] V. E. Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, "Deep hashing for compact binary codes learning," in *IEEE Conference on Computer Vision*

and Pattern Recognition (CVPR), 2015. 15, 16, 19, 22, 23, 29, 35, 36, 37, 38, 40, 41, 105, 106, 107, 111

[105] H. Liu, R. Wang, S. Shan, and I. Reid, "deep supervised hashing for fast image retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 53

[106] L. Liu, Z. Lin, L. Shao, F. Shen, G. Ding, and J. Han, "Sequential discrete hashing for scalable cross-modality similarity retrieval," *IEEE Transactions on Image Processing*, vol. 26, no. 1, pp. 107–118, 2017. 6, 18

[107] L. Liu and L. Shao, "Sequential compact code learning for unsupervised image hashing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2526–2536, 2016. 18

[108] L. Liu, L. Shao, F. Shen, and M. Yu, "Discretely coding semantic rank orders for supervised image hashing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6, 18

[109] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao, "Deep sketch hashing: Fast free-hand sketch-based image retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. ix, x, xii, 19, 22, 82, 85, 86, 87, 96, 97, 98, 99, 101, 102, 103, 104, 108, 109, 116, 130, 131

[110] L. Liu, M. Yu, and L. Shao, "Unsupervised local feature hashing for image similarity search," *IEEE Transactions on Cybernetics*, vol. 46, no. 11, pp. 2548–2558, 2016. 18

[111] ——, "Latent structure preserving hashing," *International Journal of Computer Vision*, vol. 122, no. 3, pp. 439–457, 2017. 1, 6, 18, 22, 52

[112] ——, "Learning short binary codes for large-scale image retrieval," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1289–1299, 2017. 18

[113] W. Liu, C. Mu, S. Kumar, and S.-F. Chang, "Discrete graph hashing," in *Advances in Neural Information Processing Systems (NIPS)*, 2014. 8, 19, 21, 22, 28, 29, 35, 36, 37, 53

[114] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 9, 53

[115] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *International Conference on Machine Learning (ICML)*, 2011. 8, 19, 21, 22, 28, 29, 35, 36, 37, 53

[116] X. Liu, X. Fan, C. Deng, Z. Li, H. Su, and D. Tao, "Multilinear hyperplane hashing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 53

[117] M. Long, Y. Cao, J. Wang, and P. S. Yu, "Composite correlation quantization for efficient multimodal retrieval," in *ACM SIGIR Conference on Research and development in information retrieval (SIGIR)*, 2016. 53

[118] X. Lu, F. Wu, X. Li, Y. Zhang, W. Lu, D. Wang, and Y. Zhuang, "Learning multimodal neural network with ranking examples," in *ACM international conference on Multimedia (MM)*, 2014. 47

[119] S. Lyu and E. P. Simoncelli, "Nonlinear image representation using divisive normalization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 14

[120] C. Ma, I. W. Tsang, F. Peng, and C. Liu, "Partial hash update via hamming subspace learning," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1939–1951, 2017. 87

[121] L. Ma, Z. Lu, L. Shang, and H. Li, "Multimodal convolutional neural networks for matching image and sentence," in *IEEE International Conference on Computer Vision(ICCV)*, 2015. 52

[122] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008. vii, ix, 44, 45, 103, 104, 106

[123] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, 1967, pp. 281–297. 36, 40

[124] M. Malinowski, M. Rohrbach, and M. Fritz, "Ask your neurons: A neural-based approach to answering questions about images," in *IEEE International Conference on Computer Vision(ICCV)*, 2015. 47, 52, 58

[125] D. Mandal, K. N. Chaudhury, and S. Biswas, "Generalized semantic preserving hashing for n-label cross-modal retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 85

[126] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille, "Deep captioning with multimodal recurrent neural networks (m-rnn)," in *International Conference on Learning Representations (ICLR)*, 2015. 51, 52

[127] J. Masci, M. M. Bronstein, A. M. Bronstein, and J. Schmidhuber, "Multimodal similarity-preserving hashing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 4, pp. 824–830, 2014. 49, 53, 64, 65, 71, 85

[128] B. McFee and G. R. Lanckriet, "Metric learning to rank," in *International Conference on Machine Learning (ICML)*, 2010. 38, 69, 103

[129] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *International Conference on Learning Representations (ICLR) Workshop*, 2013. 59, 86, 87, 91, 97, 101

[130] A.-r. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012. 13

[131] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning (ICML)*, 2010. 31, 58, 92, 93

[132] M. Norouzi and D. M. Blei, “Minimal loss hashing for compact binary codes,” in *International Conference on Machine Learning (ICML)*, 2011. 6, 18

[133] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean, “Zero-shot learning by convex combination of semantic embeddings,” in *International Conference on Learning Representations (ICLR)*, 2014. 86

[134] A. Oliva and A. Torralba, “Modeling the shape of the scene: A holistic representation of the spatial envelope,” *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001. 40

[135] M. Ou, P. Cui, F. Wang, J. Wang, W. Zhu, and S. Yang, “Comparing apples to oranges: a scalable solution with heterogeneous hashing,” in *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2013. 53

[136] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward, “Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval,” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 4, pp. 694–707, 2016. 51

[137] K. Pang, Y.-Z. Song, T. Xiang, and T. Hospedales, “Cross-domain generative learning for fine-grained sketch-based image retrieval,” in *British Machine Vision Conference (BMVC)*, 2017. 82, 85

[138] S. Parui and A. Mittal, “Similarity-invariant sketch-based image retrieval in large databases,” in *European Conference on Computer Vision (ECCV)*, 2014. 82, 85

[139] S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu, “Variational recurrent adversarial deep domain adaptation,” in *International Conference on Learning Representations (ICLR)*, 2017. 23

[140] Y. Qi, Y.-Z. Song, H. Zhang, and J. Liu, “Sketch-based image retrieval via siamese convolutional neural network,” in *IEEE International Conference on Image Processing (ICIP)*, 2016. 82, 85, 101, 102, 103

[141] M. Raginsky and S. Lazebnik, “Locality-sensitive binary codes from shift-invariant kernels,” in *Advances in Neural Information Processing Systems (NIPS)*, 2009. 21, 35, 37

[142] M. Rastegari, J. Choi, S. Fakhraei, H. Daumé III, and L. S. Davis, “Predictable dual-view hashing.” in *International Conference on Machine Learning (ICML)*, 2013. 53

[143] K. Rayner, “Eye movements and attention in reading, scene perception, and visual search,” *The Quarterly Journal of Experimental Psychology*, vol. 62, no. 8, pp. 1457–1506, 2009. 57

[144] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” *arXiv preprint arXiv:1612.08242*, 2016. 14

[145] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015. 14, 49, 51, 57, 63

[146] A. Rohrbach, M. Rohrbach, R. Hu, T. Darrell, and B. Schiele, “Grounding of textual phrases in images by reconstruction,” in *European Conference on Computer Vision (ECCV)*, 2016. 58

[147] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, “Imagenet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. 14, 36, 51, 63, 97

[148] J. M. Saavedra, “Sketch based image retrieval using a soft computation of the histogram of edge local orientations (s-helo),” in *IEEE International Conference on Image Processing (ICIP)*, 2014. 82, 85

[149] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets,” *PloS one*, vol. 10, no. 3, p. e0118432, 2015. vii, 38, 39, 69

[150] R. Salakhutdinov and G. Hinton, “Deep boltzmann machines,” in *Artificial Intelligence and Statistics*, 2009, pp. 448–455. 13

[151] ——, “Semantic hashing,” *International Journal of Approximate Reasoning*, vol. 50, no. 7, 2009. 6, 15, 18, 22, 23

[152] P. Sangkloy, N. Burnell, C. Ham, and J. Hays, “The sketchy database: learning to retrieve badly drawn bunnies,” *ACM Transactions on Graphics*, vol. 35, no. 4, p. 119, 2016. iv, ix, x, 82, 85, 97, 98, 101, 102, 103, 104, 108, 132, 133, 134, 135

[153] D. Scherer, A. Müller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in *International Conference on Artificial Neural Networks (ICANN)*, 2010. 14

[154] I. V. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. C. Courville, and Y. Bengio, “A hierarchical latent variable encoder-decoder model for generating dialogues.” in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017. 23

[155] F. Shen, C. Shen, W. Liu, and H. Tao Shen, “Supervised discrete hashing,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 6, 9, 10, 18, 50, 53, 60

[156] F. Shen, C. Shen, Q. Shi, A. Van Den Hengel, and Z. Tang, “Inductive hashing on manifolds,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 6

[157] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *International Conference in Learning Representations (ICLR)*, 2015. vii, xi, 14, 34, 35, 36, 37, 39, 51

[158] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, “Zero-shot learning through cross-modal transfer,” in *Advances in Neural Information Processing Systems (NIPS)*, 2013. 84, 86, 101, 102

[159] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts *et al.*, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2013. 51

[160] K. Sohn, H. Lee, and X. Yan, “Learning structured output representation using deep conditional generative models,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015. 3, 20, 23, 26, 33, 36

[161] J. Song, Y.-Z. Song, T. Xiang, and T. Hospedales, “Fine-grained image retrieval: the text/sketch input dilemma,” in *BMVC*, 2017. 82, 85, 113

[162] J. Song, Q. Yu, Y.-Z. Song, T. Xiang, and T. M. Hospedales, “Deep spatial-semantic attention for fine-grained sketch-based image retrieval,” in *IEEE International Conference on Computer Vision (ICCV)*, 2017. viii, 82, 85, 88, 89, 97, 113

[163] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen, “Inter-media hashing for large-scale retrieval from heterogeneous data sources,” in *ACM International Conference on Management of Data (SIGMOD)*, 2013. 11, 18, 47, 53, 64, 65, 70, 71, 85

[164] N. Srivastava and R. Salakhutdinov, “Multimodal learning with deep boltzmann machines,” *Journal of Machine Learning Research*, vol. 15, pp. 2949–2980, 2014. 52

[165] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2014. 49, 51

[166] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 14

[167] B. Thompson, “Canonical correlation analysis,” *Encyclopedia of statistics in behavioral science*, 2005. 64, 65, 70, 71, 99, 101

[168] A. Turpin and F. Scholer, "User performance versus precision measures for simple search tasks," in *ACM Conference on Research and Development in Information Retrieval (SIGIR), year=2006*. 116

[169] D. Wang, P. Cui, M. Ou, and W. Zhu, "Learning compact hash codes for multimodal representations using orthogonal deep structure," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1404–1416, 2015. 53

[170] F. Wang, L. Kang, and Y. Li, "Sketch-based 3d shape retrieval using convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 82, 85, 101, 102

[171] J. Wang, J. Wang, N. Yu, and S. Li, "Order preserving hashing for approximate nearest neighbor search," in *ACM International Conference on Multimedia (MM)*, 2013. 10

[172] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large-scale search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2393–2406, 2012. 6, 18

[173] J. Wang, W. Liu, S. Kumar, and S.-F. Chang, "Learning to hash for indexing big dataa survey," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 34–57, 2016. 1, 52

[174] J. Wang, W. Liu, A. X. Sun, and Y.-G. Jiang, "Learning hash codes with listwise supervision," in *IEEE International Conference on Computer Vision (ICCV)*, 2013. 10

[175] L. Wang, Y. Li, and S. Lazebnik, "Learning deep structure-preserving image-text embeddings," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 47, 52

[176] W. Wang, B. C. Ooi, X. Yang, D. Zhang, and Y. Zhuang, "Effective multi-modal retrieval based on stacked auto-encoders," in *International Conference on Very Large Data Bases (VLDB)*, 2014. 11, 53

[177] Y. Wei, Y. Song, Y. Zhen, B. Liu, and Q. Yang, "Scalable heterogeneous translated hashing," in *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2014. 47, 53

[178] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in Neural Information Processing Systems (NIPS)*, 2009. 6, 7, 11, 18, 21, 35, 36, 37, 53

[179] H. Wold, "Partial least squares," *Encyclopedia of statistical sciences*, 1985. 64, 65, 70, 71

[180] B. Wu, Q. Yang, W.-S. Zheng, Y. Wang, and J. Wang, "Quantized correlation hashing for fast cross-modal search," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. 12, 47, 53, 64, 65, 71, 85

[181] F. Wu, Z. Yu, Y. Yang, S. Tang, Y. Zhang, and Y. Zhuang, "Sparse multi-modal hashing," *IEEE Transactions on Multimedia*, vol. 16, no. 2, pp. 427–439, 2014. 12

[182] ——, "Sparse multi-modal hashing," *IEEE Transactions on Multimedia*, vol. 16, no. 2, pp. 427–439, 2014. 53

[183] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning." in *AAAI Conference on Artificial Intelligence (AAAI*, 2014. 15

[184] ——, "Supervised hashing for image retrieval via image representation learning," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2014. 19, 53

[185] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele, "Latent embeddings for zero-shot classification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 86

[186] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning - the good, the bad and the ugly," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 101

[187] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. ii, vii, 34, 37, 38, 42, 43

[188] W. Xie, Y. Peng, and J. Xiao, "Cross-view feature learning for scalable social image analysis." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2014. 64, 65, 70, 71

[189] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning (ICML)*, 2015. 23, 47, 52

[190] X. Xu, F. Shen, Y. Yang, D. Zhang, H. Tao Shen, and J. Song, "Matrix tri-factorization with manifold regularizations for zero-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 86

[191] X. Xu, Y. Yang, A. Shimada, R.-i. Taniguchi, and L. He, "Semi-supervised coupled dictionary learning for cross-modal retrieval in internet images and texts," in *ACM international conference on Multimedia (MM)*, 2015. 47

[192] F. Yan and K. Mikolajczyk, "Deep correlation for matching images and text," in *IEEE International Conference on Computer Vision(ICCV)*, 2015. 51, 52, 60

[193] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2image: Conditional image generation from visual attributes," in *European Conference on Computer Vision (ECCV)*, 2016. 20, 23

[194] E. Yang, C. Deng, W. Liu, X. Liu, D. Tao, and X. Gao, "Pairwise relationship guided deep hashing for cross-modal retrieval." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017. 85

[195] Y. Yang, Y. Luo, W. Chen, F. Shen, J. Shao, and H. T. Shen, "Zero-shot hashing via transferring supervised knowledge," in *ACM international conference on Multimedia (MM)*, 2016. ix, 84, 86, 87, 91, 96, 99, 101, 102, 103, 108, 109

[196] Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick, "Improved variational autoencoders for text modeling using dilated convolutions," in *arXiv preprint arXiv:1702.08139*, 2017. 23

[197] M. Ye and Y. Guo, "Zero-shot classification with discriminative semantic representation learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 86

[198] M. Yu, L. Liu, and L. Shao, "Structure-preserving binary representations for rgb-d action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 8, pp. 1651–1664, 2016. 18

[199] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C.-C. Loy, "Sketch me that shoe," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 82, 85

[200] Q. Yu, Y. Yang, Y.-Z. Song, T. Xiang, and T. Hospedales, "Sketch-a-net that beats humans," in *British Machine Vision Conference (BMVC)*, 2015. 82, 85, 101, 102

[201] Z. Yu, F. Wu, Y. Yang, Q. Tian, J. Luo, and Y. Zhuang, "Discriminative coupled dictionary hashing for fast cross-media retrieval," in *ACM SIGIR Conference on Research and development in information retrieval (SIGIR)*, 2014. 12

[202] Z. Yu, J. Yu, J. Fan, and D. Tao, "Multi-modal factorized bilinear pooling with co-attention learning for visual question answering," in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 92, 105, 106

[203] W. Zaremba and I. Sutskever, "Learning to execute," *arXiv preprint arXiv:1410.4615*, 2014. 49, 54, 58

[204] D. Zhai, H. Chang, Y. Zhen, X. Liu, X. Chen, and W. Gao, "Parametric local multimodal hashing for cross-view similarity search." in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2013. 47, 53

[205] D. Zhang and W.-J. Li, "Large-scale supervised multimodal hashing with semantic correlation maximization." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2014. 12, 47, 53, 64, 65, 70, 71, 82, 85, 99, 101, 104

[206] H. Zhang, S. Liu, C. Zhang, W. Ren, R. Wang, and X. Cao, "Sketchnet: Sketch classification with web images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 98

[207] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 86

[208] Z. Zhang, Y. Chen, and V. Saligrama, "Efficient training of very deep neural networks for supervised hashing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 15

[209] Z. Zhang and V. Saligrama, "Zero-shot learning via semantic similarity embedding," in *IEEE International Conference on Computer Vision (ICCV)*, 2015. 84, 86, 101, 102

[210] ——, "Zero-shot learning via joint latent similarity embedding," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 84, 86, 101, 102

[211] X. Zhao, G. Ding, Y. Guo, J. Han, and Y. Gao, "Tuch: Turning cross-view hashing into single-view hashing via generative adversarial nets," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. 85

[212] Y. Zhen and D.-Y. Yeung, "Co-regularized hashing for multimodal data," in *Advances in Neural Information Processing Systems (NIPS)*, 2012. 82, 85

[213] R. Zhou, L. Chen, and L. Zhang, "Sketch-based image retrieval on a large scale database," in *ACM international conference on Multimedia (MM)*, 2012. 82, 85

[214] H. Zhu, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2016. 19, 22, 29

[215] X. Zhu, Z. Huang, H. T. Shen, and X. Zhao, "Linear cross-modal hashing for efficient multimedia search," in *ACM International Conference on Multimedia (MM)*, 2013. 11, 53

[216] X. Zhu, L. Zhang, and Z. Huang, "A sparse embedding and least variance encoding approach to hashing," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 3737–3750, 2014. 35, 37

[217] B. Zhuang, G. Lin, C. Shen, and I. Reid, "Fast training of triplet-based deep binary embedding networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 53