Learning Compact Representations for Efficient Whole Slide Image Search in Computational Pathology

by

Sobhan Hemati

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2022

© Sobhan Hemati 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

Supervisor:	Hamid R. Tizhoosh Professor, Artificial Intelligence and Informatics, Mayo Clinic, Rochester, MN, USA
Supervisor:	Shahryar Rahnamayan Professor, Department of Electrical, Computer, and Software Engineering (ECSE), Ontario Tech University
External Examiner:	Faisal Mahmood Assistant Professor, Harvard Medical School, University of Harvard
Internal-External Member:	Ali Ghodsi Professor, Dept. of Statistics and Actuarial Science, David R. Cheriton School of Computer Science, University of Waterloo
Internal-External Member:	Otman Basir Professor, Dept. of Electrical and Computer Engineering, University of Waterloo
Internal Member:	Apurva Narayan Assistant Professor UBC adjunct assistant professor, Systems Design Engineering, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contribution

All experimental results, figures, and text are generated from my own work during this Ph.D. research. The thesis is based on the following papers that I have published.

1. **S. Hemati**, et al. *A Non-alternating Graph Hashing Algorithm for Large-scale Image Search*, Computer Vision and Image Understanding, Volume 219, May 2022, 103415

2. **S. Hemati**, et al. *Beyond neighbourhood-preserving transformations for quantization-based unsupervised hashing*, Pattern Recognition Letters Volume 153, January 2022, Pages 44-50

3. **S. Hemati**, et al. *CNN and Deep Sets for End-to-End Whole Slide Image Representation Learning*, Proceedings of Machine Learning Research 143:301–311, 2021

4. **S. Hemati**, et al. *Learning Binary and Sparse Permutation-Invariant Representations for Fast and Memory Efficient Whole Slide Image Search*, under submission.

Abstract

Digital pathology has enabled us to capture, store, query and analyze scanned biopsy samples as digital images. The widespread adoption of digital pathology has spurred the digitization of tissue biopsy samples, known as whole slide images (WSIs). Content-based WSI retrieval and computational pathology are expected to reduce the physicians' workload, improve diagnostic performance, and facilitate the teaching and research in pathology. Recent advances in deep learning have the potential to contribute to computational pathology and more effective WSI search systems. Deep learning is a successful tool for image analysis, including various applications in the medical domain. However, considering the extremely large size of the multi-resolution images and lack of patch-level labelled data, deep networks are challenging to adapt for WSI analysis. More precisely, the gigantic size of WSIs imposes three main challenges to apply deep learning to represent pathology image data for efficient and accurate processing. First, it is not easy to obtain deep WSI embeddings in an end-to-end manner. Second, storing WSIs and patch embeddings in Euclidean space needs significant memory resources when operating in large repositories. Third, WSI and patch search using Euclidean embeddings in large image archives is infeasible. In order to address the above challenges, first, we propose Efficient Spectral Hashing (ESH), a method based on spectral hashing formulation with lower space and time complexities which leads to binary representations with an enhanced search performance compared to many recent hashing methods. We also proposed a novel quantization scheme, called non-rigid quantization (NRQ), where for the first time we proposed to employ non-rigid transformations for minimizing quantization loss. After studying standard hashing algorithms, the main challenge is modifying these methods so that they can be applied to WSIs. Due to the gigantic size of WSIs, the first step in processing WSIs is to replace them with a subset of their associated representative patches. Considering this multi-instance (bag of patches) representation per WSI, this is not clear how to apply the two proposed methods to learn binary WSI representations. To mitigate this challenge, we proposed we proposed CNN-Deep Sets (CNN-DS) to learn one permutation invariant vector representation per WSI in an end-to-end manner. Although using CNN-DS, we were able to obtain WSI embeddings, still there were two issues with this approach. First, the method faces high GPU memory usage during the training due to keeping multiple bags of patches in the memory. Second, the obtained embeddings were in Euclidean space which for the very large archives the search speed becomes very slow while they occupy significantly more storage. Further, applying ESH/NRQ on the extracted embeddings needs an additional learning step. To unify ideas from ESH/NRQ with CNN-DS that is learning compact (binary and sparse) permutation-invariant WSI representation for efficient search and also to bypass training time memory bottleneck we proposed a novel framework based on deep

generative modelling and the *Fisher Vector Theory*. We introduced new loss functions for learning sparse and binary permutation-invariant WSI representations that employ instance-based training achieving better memory efficiency. The learned WSI representations were validated on The Cancer Genomic Atlas (TCGA) and Liver-Kidney-Stomach (LKS) datasets. The proposed method outperforms *Yottixel* (a recent search engine for histopathology images) both in terms of retrieval accuracy and speed. Further, we achieve competitive performance against SOTA on the public benchmark LKS dataset for WSI classification. Finally, showed that learning sparse permutation-invariant WSI representations which in our framework is associated with encouraging sparsity on the gradients reduces the sharpness of the loss landscape and as a result improves the generalization of deep neural networks.

Acknowledgements

Writing this thesis has been fascinating and extremely rewarding. I'd like to thank many people who have contributed to my thesis in several ways.

First and foremost, thanks to my supervisor, Professor Hamid R. Tizhoosh for his constant support, encouragement, and patience throughout the PhD. I'm glad be part of his lab, Kimia Lab, where so many researchers are free to explore creative ideas. I'd like to thank Professor Shahryar Rahnamayan for his valuable comments that he gave me during thesis preparation and also constant emotional support.

I'd like to thank my co-authors Shivam, Hadi, Morteza, Cameron, Professor Chenouri, and Professor Ghodsi for all their collaboration in conducting the research. I would like to thank my friends, lab mates, colleagues and research team – Adnan, Abubakr, and Milad, for a cherished time spent together in the lab, and in social settings.

I owe a special thanks to Prof. Ali Ghodsi, Prof. Otmon Basir, and Prof. Apurva Narayan to be part of my thesis committee and for taking out their time to review my thesis and providing their valuable suggestions. I'd also like to thank Prof. Faisal Mahmood to be an external committee member for my thesis..

I'd like to thank my parents for teaching me to value humanity. I deeply appreciate all family and friends for their infinite support and guidance. Finally, I would like to thank God. He has given me strength and encouragement throughout all the challenging moments of completing this dissertation. I am truly grateful for His unconditional and endless love, mercy, and grace.

Dedication

To my beloved parents.

Table of Contents

List of Tables

List of Figures

List of Acronyms

k-NN k-nearest neighbors 64

AGH Anchor Graph Hashing 10, 12, 36, 37

AQ Angular Quantization 10, 11, 42, 44

BA Binary Autoencoder 10

BOW Bag of Visual Words 16–18, 74

C-Deep-BFV Conditioned Deep Binary Fisher Vector 76, 79, 83, 91

C-Deep-BFV Conditioned Deep Sparse Binary Fisher Vector 78, 79

C-Deep-FV Conditioned Deep Fisher Vector 79, 84, 90

C-Deep-SFV Conditioned Deep Sparse Fisher Vector 76, 79, 82–84, 90, 91

C-GMM-FV Conditional GMM-based Fisher Vector 79, 81

CAD Computer-assisted Diagnosis 94

CNN Convolutional Neural Network 7, 13, 14, 54, 58, 59, 61, 66, 70, 78, 83, 85

CNN-DS CNN-Deep Sets 59, 61, 63–66

CVAE Conditioned Variational Autoencoder 76–79, 82

DGH Discrete Graph Hashing 10, 12, 22, 25, 33, 37

DH Deep Hashing 10, 38, 54

DSH Discrete Spectral Hashing 12, 22, 25, 31–34, 37

DSO Double Soft Orthogonality 55

ESH Efficient Spectral Hashing xii, 6, 24, 27–34, 36–39, 41, 42, 96, 97

FIM Fisher Information Matrix 17, 73

GAN Generative Adversarial Network 21, 100

GMM Gaussian Mixture Model 17–21, 71, 74

GPU Graphics Processing Unit 4

IsoHash Isotropic Hashing 10, 11, 42

ITQ Iterative Quantization 10–12, 42–44, 47, 54, 56

KMH K-means Hashing 10, 31

KNNH k-Nearest Neighbors Hashing 10, 11, 32, 33, 51

LGHSR Large Graph Hashing with Spectral Rotation 10, 12, 22, 32–34, 37

LKS Liver-Kidney-Stomach 79, 81–83, 91

LSH Locality Sensitive Hashing 8, 33

LUAD Lung Adenocarcinoma 64

LUSC Lung Squamous Cell Carcinoma 64

mAP mean Average Precision 30–34, 49–51

MC Mutual Coherence 55

MIA Membership Inference Attack 85, 87, 98, 100

MIL Multi-Instance Learning 4, 5, 13–15, 20, 59, 68–71, 91, 99

NRQ Non-rigid Quantization 6, 47, 50–52, 54, 97

PCA Principal Component Analysis 11, 43, 44, 51, 55

RDSH Robust Discrete Hashing 12, 22, 25, 36, 37

SADH Similarity-Adaptive Deep Hashing 10, 38, 54

SCQ Simultaneous Compression and Quantization 10, 11, 42, 44, 51

SGH Scalable Graph Hashing 10, 33

SH Spectral Hashing 10, 12, 23

SNRQ Sequential Non-rigid Quantization 43, 47, 49–52, 54–56

SO Soft Orthogonality 55

SpH Spherical Hashing 10, 31

SVD Singular Value Decomposition 27, 47, 48

TCGA The Cancer Genome Atlas 59, 63, 64, 79, 81, 82, 91

UHBDNN Unsupervised Hashing with Binary Deep Neural Network 10, 38, 54

VAE Variational Autoencoder 21, 73–75, 78, 100

WSI Whole Slide Image 1, 7, 8, 12–17, 20, 21, 58–66, 68–71, 73–83, 91, 95–100

Chapter 1

Introduction

1.1 Motivation

The widespread use of the Internet and recent advances in the development of several fields such as smartphones, high-resolution microscopes, and cameras have massively contributed to creating large data repositories. The amount of data generated each year is accelerated more than before and this is just the beginning of the data revolution that will affect every business and many aspects of human life. This data explosion is observed in various fields, including business, finance, healthcare, and communication.

A healthcare-related example is digitizing histopathology glass slides of tissue samples. Histopathology is the examination of tissue under a microscope to study the relation between tissue morphology and different diseases [113, 53]. A digital histopathology image, called a Whole Slide Image (WSI), is captured via a special scanner where the tissue glass slide is digitized at different zoom levels. WSIs are gigapixel images generally with large dimensions, often larger than 100,000 by 100,000 pixels [19]. Pathologists are usually overworked, and their jobs are stressful and rigorous. It is expected that WSI content-based retrieval systems can be beneficial in different ways, including teleconsultation, workload efficiency, collaborations, improving diagnostic accuracy, virtual education, and research [113, 115]. More precisely, Pathologists examine biopsy tissues to detect the tumors and to investigate their characteristics in order to evaluate tumor aggressiveness which is s a complex task that needs many years of experience, and sub-speciality expertise [55]. However, having a fast and reliable retrieval system can act as additional knowledge that shares the experience of many other pathologists that investigated similar cases in the past. This will lead to a more reliable and objective diagnosis with less inter-observer variability. Please

note that although the phrase "retrieval" can refer to a more general notion compared with "search", in this thesis, we use "search" and "retrieval" interchangeably.

Considering the above mentioned potentials of digital pathology and WSI content-based retrieval systems, one expects that WSIs can be analyzed using the myriad of computer vision methods currently available for similar tasks in other fields. As such, the usage of deep learning for WSI analysis has become an active area of research [19]. Unfortunately, scientific progress with these data has been rather slow because of difficulties with the data itself. These obstacles include highly complex textures of different tissue types, colour variations caused by different stainings, rotationally invariant nature of WSIs, lack of labelled data, and most notably, the extremely large size of the images [19, 113]. Additionally, these images are multi-resolution, meaning that each WSI may contain images from different zooming levels, primarily 5x, 10x, 20x, and 40x magnification levels [19, 113, 53]. The largest obstacle that hinders developing a WSI content-based retrieval system and also the application of deep networks in computational pathology tasks is the sheer size of WSIs that makes the deployment of many solutions infeasible - or perhaps even impossible. Hence, techniques to deal with storage, search and analysis of giga- and terabyte datasets have become even more necessary than before.

The problem of search in big data is recognized as the nearest neighbour search in which the objective is to retrieve the nearest neighbour data point to the query data point among all data points in an archive. Because of data abundance, databases are generally very large (consisting of millions of data points). Further, in many applications, including computer vision, data points are high dimensional[1]. Considering both the gigantic size of databases and the high dimensionality of each data point, this is clear that nearest neighbour search is prohibitively time-consuming. Apart from the computational cost of search, significant memory resources are needed to store this massive data. These challenges are even more daunting for gigapixel images. This is because it is not trivial to have a single (global) representation for gigapixel images as this is not possible to simply feed these multi-magnification images into networks. Considering this difficulty, a general way for representing WSIs can be extracting some small patches from WSIs and simply use a deep pre-trained network to extract features from each patch. This approach provides us with a set of (or bag of) feature vectors for each WSI which makes the dataset significantly larger. This is because each WSI is itself a set of patches where the set size can generally be something between several hundreds to several thousands feature vectors. In this setting, where each WSI is presented by a set of patches, some challenges emerge. First, it is not clear to calculate the distance (dissimilarity) between two WSIs, i.e., two

[1]Today, the standard way of representing a regular image is using pre-trained networks that lead to the vector representation of the image with dimensionality in order of 1000-4000.

sets of patches (vectors). Second, the speed in patch-based search may become too low due to the high number of patches. Finally, significant memory resources are necessary for storing representations of these patches.

In order to address these challenges, this thesis explores two computer vision research directions separately and then propose a framework for unifying these two research directions. The first idea is learning similarity preserving binary representations for patches. Note that as each WSI contains multiple patches, the number of patches may become too large (e.g., for large tissue samples), which may render the embedding in Euclidean space in the patch retrieval task infeasible. The second idea is learning one single representation per WSI that must capture the semantic information in the gigapixel image. Such representation is particularly useful in the WSI search problem both in terms of retrieval speed and memory usage. Finally, the possibility of proposing a framework for unifying binary representation learning and WSI representation learning tasks is explored.

For the patch search problem, where the number of WSI patches is prohibitively large, one can employ binary representations. Learning binary representations is known as "hashing" . Learning to hash is generally understood to be learning similarity-preserving binary representations and is one of the most effective techniques for the fast approximate nearest neighbour search due to its efficiency in storing big data, and computational speed [118]. That explains why there is a large body of works on different types of hashing for image-based applications where the high-dimensionality inherent to visual data poses unique challenges to any search and retrieval system. Hashing has been applied to different problems where the fast nearest neighbour search is necessary. Examples where hashing has been applied include image annotation [120], segmentation of videos [79], multimedia retrieval [29], [108], large-scale clustering [129] and audio search [116]. With recent advances in the area of approximate nearest neighbour search, the word *hashing* is being used in two different contexts. Traditionally, hashing refers to indexing algorithms that attempt to increase search speed to achieve better performance compared with exhaustive search (linear scan) [118]. These hashing algorithms increase the search speed using lookup tables. On the other hand, the phrase hashing is recently used to describe algorithms that learn similarity preserving binary codes from data in continuous spaces. In these algorithms, which are mainly referred to learning-based algorithm ("learning to hash"), the focus is on learning binary representations of images that preserve the neighbourhood structure of continuous data as much as possible. In learning-based methods, algorithms are evaluated based on their retrieval performance in exhaustive search (linear scan). This search strategy is known as *hash code ranking* [118] where the similarity is efficiently calculated using the Hamming distance.

For the WSI search problem, obtaining one single embedding per WSI that fully cap-

tures the semantic information of WSI can be a straightforward solution; having one feature vector per WSI improves both search speed and necessary memory resources to store WSIs. However, considering that the common practice in dealing with WSIs is patch selection, one has to deal with a set representation problem to obtain a vector that fully captures the semantic information in each WSI. Given this set of patches representing each WSI, one possible solution is Multi-Instance Learning (MIL) [51] where a label is assigned to a set of instances instead of one instance. However, as this is discussed in the next chapters, there are other alternatives based on the bag of visual words [17] that have been ignored by the computational pathology community and still have a great potential in the set representation problem and provide computational advantages compared with the common MIL scheme.

1.2 Problem Statement

The problem that is faced in designing a deep learning-based content-based WSI/patch retrieval is as following.

How can we obtain compact deep WSI/patch representations that can be used in WSI and patch search while training-time memory bottleneck is bypassed? To answer the above question, the problem is approached by developing the following methods:

1. Methods for learning high quality compact representations e.g., binary/sparse representations for patch retrieval problem.
2. Methods for learning high quality permutation invariant WSI representations while they can handle multi-magnifications and bypass the Graphics Processing Unit (GPU) memory bottleneck caused by gigantic size of WSIs.
3. A framework for unifying solutions for the two above problems to achieve ultimate efficiency, i.e., compact WSI representations.

In this thesis, efficiency is sought in multiple directions to propose a unified approach where these directions meet each other. First, for the patch search (and within WSI search) problem, it is desired to obtain binary representations (or any other form of compact representations like sparse representations) that lead to efficient image retrieval both in terms of retrieval speed and memory usage. Second, this is expected the proposed method obtains one vector embedding per WSI that fully captures the semantic information in the

WSI and, at the same time, handle multi-magnifications and enjoys a low computational and memory expenses during training. Furthermore, this is anticipated the developed deep neural net can obtain both compact representation (binary or sparse) for WSIs and patches and is trained in an end-to-end manner. This is desired because this neural net does not face memory bottleneck during training, i.e., it is trained on instances, not a bag of instances and employs information from all magnifications in the WSI.

For binary representation learning, the learning to hash literature is explored and this is tried to develop methods that suit the intrinsic nature of our data that is gigantic size. More precisely, this is attempted to adapt current methods such that they can easily bypass the memory bottleneck and reduce time complexity during binary representation learning of patches.

For representation learning of WSIs, i.e., obtaining one vector that models a WSI, a variety of ideas based on heuristic architectures, MIL-based approaches, and dictionary learning approach are studied. We propose learning schemes for WSI representation learning based on permutation invariant neural networks and also dictionary learning approach where the objective is to learn compact WSI representations from different magnifications in an end-to-end manner while the memory bottleneck is bypassed. As a matter of fact, although due to permutation invariance properties, the MIL scheme has become a standard approach for applying deep learning to WSIs, we argue that this approach may not be the best or the only possible option for WSI representation learning. The main reason for this argument is the fact that in MIL, multiple bags are fed to the neural net. This can be particularly problematic in the processing of WSIs as the number of patches per WSI (bag size) is around 200-300, which even with a batch size like 16 causes memory issues for end-to-end training on images. Considering this, our final proposed method scheme is not based on the common MIL scheme and instead we employ deep generative models as a dictionary learning-based approach to obtain compact WSI representations from different magnifications while the training is end-to-end and the memory bottleneck is bypassed.

In summary, we are interested in an approach that provides:

1. Permutation invariant representations for WSIs while the training is on instances instead of a bag of instances.
2. Representations for both WSIs and patches.
3. Compact representations (including binary or sparse).
4. Representations that are guided by available information, e.g., the primary site of the tissue.

5. Multi-magnification informed representations.

To the best of my knowledge, such a network does not exist in the literature of medical image processing, and its contribution can be very helpful.

1.3 Thesis Organization

In the last section, we briefly introduced problems that we may face to obtain high-quality, compact representation for patches and WSIs. We observed that some natural candidates for this difficult task are learning to hash and possible schemes for WSI representation learning. The rest of this thesis is organized as follows: the existing related literature on learning to hash and WSI representation learning will be briefly reviewed in Chapter 2. Subsequently, inspired by limitations in existing methods, in Chapters 3 and 4, we propose two efficient unsupervised hashing methods namely ESH and Non-rigid Quantization (NRQ) for learning high quality binary representations that can be easily applied to big data while enjoy lower time complexity. Then, in Chapter 5 we propose a learning scheme based on permutation invariant neural network Deep Sets that employs the hierarchical relation between primary site and primary diagnoses which helps the network to make better predictions. In Chapter 6 we propose a framework based on deep generative models for learning compact (binary/sparse) WSI permutation invariant representations, this framework enable us to to end-to-end training without training time memory bottleneck while it can handle multiple magnifications. Finally, in Chapter 7 the conclusion is presented.

Chapter 2

Related Works

The increased acquisition of WSIs data has opened new opportunities in the quantitative analysis of tissue histopathology, e.g., supporting the diagnostic process by reducing the inter-and intra-observer variability among pathologists. As a result, there is a hope that employing machine learning algorithms in histopathology image analysis can provide more advantages from the digitization in the pathology domain. As such, the usage of deep learning for WSI analysis has become an active area of research. Unfortunately, scientific progress with these data has been slowed because of difficulties with the data itself. These difficulties include highly complex textures of different tissue types, colour variations caused by different stainings, rotationally invariant nature of WSIs, lack of labelled data and most notably, and the extremely large image size (often larger than 100,000 by 100,000 pixels). Additionally, these images are multi-resolution, meaning that each WSI may contain images at different zooming levels, primarily 5x, 10x, 20x, and 40x magnification [113].

One of the main usages of applying deep learning on WSIs is the representation learning of these images, which is challenging due to their large size. In practice, this hurdle is often bypassed by simply considering small 'patches' of the WSI, a set of which is meant to represent the entire WSI [27, 10]. Existing patching schemes allow us to split the WSI into tiles to be inputted to deep Convolutional Neural Network (CNN)s for WSI representation learning. However, such representations impose some new challenges.

1. Patch level labels generally are not available. In many cases, all we have is WSI level annotations which are not valid for all patches of that WSI.
2. Representing each WSI by a set of vector embeddings needs significant memory resources to store large archives of the patch and WSI representations.

3. Representing every WSI by a set of embeddings makes the downstream tasks, e.g., WSI classification and retrieval, non-trivial.

To address these challenges, three main ideas are explored in this thesis.

1. Learning high-quality binary representations of patch embeddings.
2. Developing a framework to learn one universal embedding per WSI. Ideally, we are interested in end-to-end methods that capture information from different magnifications and data modalities without facing the training time memory bottleneck.
3. Combining the first and second techniques to achieve compact (binary or any other efficient representation, e.g., sparse) WSI representations for ultimate efficiency.

2.1 Binary Representation Learning

Learning binary representations is one possible solution for efficient patch and WSI search problems. This is because of the two main reasons. First, binary representations significantly occupy less memory. Second, owing to the efficiency of the Hamming distance in calculating distance between binary embeddings, the retrieval speed for binary representations is significantly high. Given the advantages of binary representations for gigapixel WSI search, here the related literature is explored. Then in the next chapter, I present my approach for binary representation learning that bypasses the training time memory bottleneck inherent to big data (e.g., patches in our case) and reduces time complexity.

2.1.1 Categorization of Hashing Methods

Hashing algorithms can be categorized in many different ways. In the following, we briefly introduce these categorizations and explain each one separately.

Data-Independent and Data-Dependent

The first categorization classifies the hashing methods in two main sub-groups: data-independent and data-dependent algorithms [118]. The Locality Sensitive Hashing (LSH) [2] is one of the first representatives in the first group where hash functions are constructed using random projections. Although these methods are faster compared with

data-dependent algorithms, their performance is inferior. Furthermore, it is known that in the data-independent methods, due to ignoring the label information or the structure of data, the performance heavily depends on the length of binary codes, and as a result, they may not be efficient enough for big data.

Supervised and Unsupervised

To address shortcomings related to data-independent algorithms, many data-dependent (or learning-based) hashing algorithms have been proposed. The learning-based hashing methods are divided into supervised and unsupervised methods. As it is clear from their names, supervised hashing methods incorporate label information to learn similarity-preserving binary codes e.g., semi-supervised hashing [119], supervised hashing with kernels [77], and discrete hashing [104]. On the other hand, unsupervised methods only use the affinity information in the data [123].

Single-modal and Cross-modal

Another categorization of data-dependent hashing algorithms is based on the fact that they learn similarity-preserving binary codes from a single type of data or more than one data type, i.e., cross-modal hashing algorithms[118].

Deep Hashing

Recently deep learning has been applied to the supervised hashing problem [25]. This categorization is mainly referring to the type of hash functions. more precisely, when the hash function is a deep learning model, the authors refer the algorithm a deep hashing algorithm.

One-step and Two-step

Deep learning methods can be divided into two categories which are one-step and two-step methods. The one-step category refers to the methods where hash function and binary codes are learned simultaneously. As an example, authors in [70] proposed a scheme to jointly learn the hash function and image features with a triplet loss formulation. On the other hand, in two-step methods, first, binary codes are obtained, and then a deep network (deep hash function) is used to learn a mapping from continuous space to binary codes. The main challenge in these types of algorithms is the binary code inference step [7].

2.1.2 Challenges in Learning to Hash

Although deep hashing algorithms have improved the retrieval performance in supervised hashing problems, there are practical limitations in using these algorithms. The first limitation is the test-time computational cost that is obtaining binary code for the query (input) image. Note that one of the main reasons for employing hashing algorithms is their high computational speed in retrieval. However, in deep hashing methods, due to the large number of parameters in hash functions, they have slower test-time encoding compared with linear hash functions. Besides, deep hashing algorithms generally have been applied to supervised problems, while in practice, real-world problems are mainly unsupervised as labelling data can be prohibitively expensive.

2.1.3 Literature Review on Unsupervised Hashing

Because of the challenges related to the supervised hashing methods, unsupervised hashing methods have recently attracted more attention. Classic unsupervised hashing algorithms include Spectral Hashing (SH) [123], Hashing with Graphs [78], Spherical Hashing (SpH) [43], K-means Hashing (KMH) [36], Iterative Quantization (ITQ) [33], Anchor Graph Hashing (AGH) [78], and Discrete Graph Hashing (DGH) [76]. Furthermore, other more recent works on unsupervised hashing include Binary Autoencoder (BA) [8], Scalable Graph Hashing (SGH) [54], Large Graph Hashing with Spectral Rotation (LGHSR) [72], k-Nearest Neighbors Hashing (KNNH) [37], and Simultaneous Compression and Quantization (SCQ) [44]. Although deep learning has been mainly applied to the supervised hashing problem, some unsupervised deep hashing algorithms have recently been proposed. Some recent deep unsupervised hashing algorithms include Deep Hashing (DH) [25], Unsupervised Hashing with Binary Deep Neural Network (UHBDNN) [20], Deepbit [73] and Similarity-Adaptive Deep Hashing (SADH) [105].

Note that due to the discrete nature of the hashing problem, it leads to difficult optimization problems. For this reason, to simplify the problem, the above mentioned algorithms formulate the problem from different perspectives. For example, authors in ITQ [33], and Angular Quantization (AQ) [32] propose to minimize quantization loss using a neighbourhood preserving transformation. In Isotropic Hashing (IsoHash) [65], equalizing variance across projections was proposed. Authors in BA [8] suggested that if the binary codes are representative enough, one can reconstruct euclidean representation from them and, as a result, proposed to reduce reconstruction loss in an encoder-decoder architecture. Finally, authors in SH [123] proposed a formulation for direct binary code learning that preserves affinity in euclidean space.

In the following, we briefly review some seminal works and also recent advances on two main techniques out of the above algorithms for unsupervised hashing algorithms namely quantization-based and graph hashing algorithms.

Quantization-based Hashing

One line of research on unsupervised hashing are quantization-based methods where they attempt to reduce quantization error. In this point of view, the corruption in the neighbourhood structure of data is due to quantization error. Considering this, in order to learn similarity preserving binary codes, one may attempt to reduce quantization error. One common step in conventional unsupervised quantization-based hashing algorithms that attempts to minimize the quantization loss is applying Principal Component Analysis (PCA), a transformation that reduces the redundancy among the features and also projects the data into a rotation-invariant space. This approach is used in many hashing algorithms including in the seminal algorithm ITQ [33], AQ [32], and IsoHash [65] where the rotation is calculated in a way such that the quantization loss of obtaining binary codes is minimized. The ITQ is still among algorithms with the highest performance for unsupervised hashing as it captures the affinity information in the original space leading to binary codes with good quality. However, there are two obvious limitations in this algorithm. First, the process of projecting data to a rotation invariant space and minimizing the quantization loss occurs in two separate stages. Second, the quantization error is reduced merely by a rotation matrix which may not be powerful enough to reduce quantization error to the ultimate limit. To overcome the first limitation of ITQ, which is projecting data and minimizing quantization loss in two independent steps, authors have recently attempted to combine these two steps. For example, in SCQ [44], authors also propose to compress and quantize data using a single matrix simultaneously. More precisely, this algorithm is similar to the ITQ, where the orthogonal matrix for minimizing quantization loss is no longer a square matrix. The KNNH [37] is also a new development based on the ITQ skeleton where a heuristic approach was suggested to minimize conditional entropy of deriving binary codes. In practice, in KNNH, is exactly equivalent to the ITQ algorithm except that a simple heuristic that is replacing each data point by the average of k nearest data points is used to reduce conditional entropy. This step is called shrinkage as pull data points closer to each other (and probably far away from Hamming vertices!).

Graph Hashing

The first graph hashing algorithm is SH [123]. In this work, the authors developed a loss function where pairwise similarities between data points in Euclidean space are used as penalty terms such that the more similar data points are mapped to closer binary codes in the Hamming space. This loss function, along with bit balance and bit uncorrelation constraints, leads to a formulation known as *spectral hashing* (SH). The SH is a natural approach as it incorporates the affinity matrix directly into deriving affinity-preserving binary codes. However, the mentioned SH method has three main problems. First, calculating the pairwise affinity between data points is computationally expensive, that is, $O(n^2 \times d)$ where n is the number of data points and d data dimensionality. Second, even for one bit, the optimization problem in SH is equivalent to balanced graph partitioning, which is a hard problem. Third, the problem size grows linearly with the number of data points which makes it interactable to be used for large archives. As a result, different attempts have been made to address these limitations. To reduce the computational complexity of calculating affinity matrix, authors in hashing with graphs paper AGH [78] suggested employing neighbourhood graph to derive a low-rank approximation of the affinity matrix. The complexity of this approximation is $O(n)$. To deal with the computational complexity of the discrete optimization faced in spectral hashing, initially, continuous relaxation was proposed. However, in this simplified approach, the binarization step destroys the learned manifold structure and degrades the quality of binary codes. To mitigate this issue, some authors proposed to incorporate a quantization loss. For example, spectral rotation (LGHSR) [72] a two-step approach similar to the idea in ITQ, was recently introduced, which finds the optimal rotation that minimizes the quantization loss. In discrete graph hashing (DGH) [76] authors attempted to solve the relaxed problem and minimize the distance between continuous and the binary set simultaneously. In Robust Discrete Hashing (RDSH) [126], an objective function for obtaining the relaxed solution along with both optimal rotation for reducing quantization loss and hash function was developed. Finally, authors of Discrete Spectral Hashing (DSH) [46] recently proposed another approach to joint optimization over relaxed variables with minimum quantization. They showed that their solution achieves better performance while reducing training complexity.

2.2 WSI Representation Learning

Although binary representations seem to be a good solution for the WSI search problem, considering the bag of patches representation for WSIs, there is no established way to

calculate the distance between two sets of vectors. One solution was proposed by Kalra et al. in [58] where authors resorted to the heuristic approach of taking the median of minimums to calculate the total distance between two WSIs. Although they were able to show that their approach achieved satisfactory performance, due to the computational complexity inherent to the median of minimums method, when the number of patches is high, even using binary representations, this approach can be computationally expensive. Further, there is no theoretical guarantee behind the median of minimum method.

On the other hand, representing a WSI using one global vector representation not only removes the necessity of resorting to decision fusion methods in WSI classification but also considerably simplifies the WSI search problem. Considering the gigapixel nature of WSIs, there is a large body of work on producing WSI representations suitable for different quantitative tasks. In the following, the related literature on WSI representation learning is reviewed. For better organization, papers are categorized into three main groups, i.e., heuristic architectures, MIL based methods, and dictionary learning approaches which are explained in the following subsections.

2.2.1 Heuristic Architectures

One line of research for WSI representation learning is based on the methods that split the WSI learning problem into multiple steps to simplify the problem. More precisely, in these methods, first, there is an instance-based training step where instances are smaller parts of WSIs, typically patches, and then another network is trained to obtain WSI embedding while capturing spatial relationship among the patches. As some examples, authors in [15] trained an Inception-V3 model on patches extracted from 20x and 5x magnifications for lung cancer subtype classification. To predict a label for a WSI from patch label predictions, they employed a simple heuristic based on the proportion of the patches assigned to each category. Tellez et al. [112] proposed a two-step method to employ CNNs for WSI classification. To this end, in the first stage, they compress image patches using unsupervised learning. Then compressed patches are placed together (such that their spatial position is kept), and they are fed to another CNN for final prediction. Bejnordi et al. [4] developed a context-aware stacked CNN, which consisted of two networks to capture information in both fine-grained and low-resolution domains. In their model, the first CNN was trained on high-resolution patches to learn cellular-level information. Then, the output of this network is connected to the second CNN to build the new model. In the second round of the training, this new model is trained on the larger patches, and during the training, the weights of the first sub-network are frozen. In Spatio-Net [64] patches

are first processed by a CNN, then the embedded patches with each neighbour are fed into 2D-LSTM layers to capture the spatial information.

2.2.2 MIL-based Approach

Representing each WSI as a bag of image patches makes permutation invariant networks and MIL-based schemes [18, 57, 51] a natural approach for WSI representation learning [93]. The two main types of MIL methods are instance-based and embedding-based methods. In the instance-based approach, the neural net is trained on instances, and for every instance, a probability between 0 and 1 is calculated (in binary classification setting). Then, a MIL-based symmetric function removes the set nature of data and calculates the label of the bag. The advantage of the instance-based method is its ability to detect the key instances (instances that contribute the most to recognizing the label of the bag). This explainability is particularly important in the medical imaging domain. On the other hand, the disadvantage of instance-based methods compared with embedding-based MIL has inferior performance in bag classification. The embedding-based methods are pretty similar to the instance-based MIL. The main difference is that the MIL-based symmetric function is applied on instance embeddings instead of instance scores. The main limitation in embedding-based methods is that there is no way to retrieve key instances.

One of the initial efforts in MIL-based WSI classification was conducted by Hou et al. [45]. They proposed a patch-level classifier for WSI classification. In order to combine their patch-level classifier, they proposed a decision fusion model. By considering the spatial relationship among the patches, they utilized an expectation-maximization method to obtain the set of distinct patches from each WSI. More precisely, the mentioned patch-based CNN by Hou et al. [45] can be seen as a two-step instance-based MIL method where an algorithm is presented to determine instance classes. The issue with Hou et al. [45] is the fact that this approach is not end-to-end, meaning the two-stage neural networks and EM approach appeared to perform sub-optimally. Motivated by this limitation, employing end-to-end MIL algorithms has been an area of research for better WSI representation learning. Deep Sets is one recent work on permutation invariant networks [130] where the authors specified a permutation-invariant function and proposed to employ universal set function approximators in the neural network. They showed that despite its simplicity, their proposed permutation-invariant architecture could achieve promising performance in a variety of tasks, including point cloud classification. Another permutation-invariant neural network similar to Deep Sets is PointNet [91] which has also been shown to be effective, showing promising results on Point Cloud classification and segmentation tasks and can be applied to WSI representation task. For further improvement over traditional

pooling layers, authors in [49] introduced attention-based MIL. Although the attention-based MIL approach showed promising results compared with traditional pooling layers, they are more prone to overfitting when the number of WSIs is small. [11] proposed an end-to-end MIL-based method for simultaneous patch and WSI representation learning in a single framework where a center loss is introduced to map patch embeddings from the same WSI to a single centroid and reduce the intra-class variations. They showed that their approach achieves promising results compared with other MIL-based methods, especially two-stage MIL methods.

These and many other papers all used patch-level training with decision fusion methods to achieve WSI-level labels. Although this can be a helpful approach for classification, for WSI search, it leads to a set of vector representations that have to be used to calculate distances between WSIs. There is no established way how to calculate the distance between two sets of vectors. For example, authors in [58, 96] resorted to the heuristic approach of taking the median of minimums to calculate the total distance between two WSIs. Although they were able to show that their approach achieved satisfactory performance, due to the computational complexity inherent to the median of minimums method, the retrieval time can be considerably high. On the other hand, representing a WSI using one global vector representation not only removes the necessity of resorting to decision fusion methods in WSI classification but also considerably simplifies the WSI search problem.

2.2.3 Dictionary Learning Approach

This is clear that due to the permutation invariance property of the MIL scheme, it has become a standard approach for applying deep learning to WSIs. However, I argue that this approach may not be the best or the only possible option for WSI representation learning. The main reason for this argument is that multiple bags are fed to the neural net in the MIL scheme. This can be particularly problematic in processing WSIs as the number of patches per WSI (bag size) is around 300, which even with a batch size like 16 leads to a batch of data with a size of $16 \times 300 \times 1000 \times 1000 \times 3$. This causes memory issues for end-to-end training on WSIs. Furthermore, there is no interpretability in embedding-based MIL, which is suitable for obtaining one representation per WSI.

Another approach that can be used for the WSI representation problem (or equivalently, the set representation problem) has some roots in old-school computer vision. As the proposed ideas in this thesis (chapter 6) are deeply related to this approach, the related literature and ideas are reviewed in more depth.

Bag of Visual Words for Set Encoding

Before the invention of deep learning, local image descriptors were being used for image representation. For histopathology image representation, the problem is fairly similar, except that instead of local descriptors, deep networks are employed to represent patches. As a result, in order to obtain a universal WSI representation, patch representations can be combined using methods similar to the Bag of Visual Words (BOW) [17] and other dictionary earning based methods. In BOW method, a clustering algorithm with K cluster is run on all local descriptors (deep patch embedding) from all images to build a dictionary. Then, for each WSI, we initialize a zero vector with K elements associated with k clusters and fill the vector by counting the number of patches that belong to each cluster. By the help of this method, one can easily get rid of the bag of patch deep embedding representation of data.

Fisher Kernel

Authors initially introduce the idea of employing the gradient of the log-likelihood of the generative model with respect to parameters in [52]. They proposed *Fisher Kernel* for exploiting generative models in discrimination tasks. More precisely, the key idea behind Fisher Kernel is to derive a kernel function from a generative probability model. Initially, the main motivation for deriving such a kernel was bridging the gap between generative and discriminative models. The Fisher Kernel achieves this by capturing the generative process in a metric between examples (or set of examples, i.e., $\mathbf{X} = \{\mathbf{x}_t, t = 1, \dots T\}$ where T is the number of examples in the set). In fact, in the classification task, the objective is to find the differences in the posterior probabilities for the labels L. In comparison, here we are interested in the difference between the generative process of two examples, i.e., x_i and x_j or two sets of examples, i.e. $\mathbf{X}_i$ and $\mathbf{X}_j$ (we shall see how the difference between two sets of examples can be handled using this kernel). This characteristic of encoding sets particularly makes Fisher Kernel a natural candidate for WSI representation learning. To capture the generative process in a metric between two sets of examples, Tommi S. Jaakkola and David Haussler [52] proposed to employ the gradient space of the generative model. Based on [52], "the gradient of the log-likelihood with respect to a parameter describes how that parameter contributes to the process of generating a particular example". To formulate this idea more formally consider a class of probability models $P(\mathbf{X} \mid \theta)$ where θ is a parameter vector and $\theta \in \mathbf{\Theta}$ and $\mathbf{X}$ is set of examples i.e, $\mathbf{X} = \{\mathbf{x}_t, t = 1, \dots T\}$. In this case, the Fisher Score is defined as

$$U_{\mathbf{X}} = \nabla_{\theta} log P(\mathbf{X} \mid \theta) \tag{2.1}$$

Note that the $U_{\mathbf{X}} \in \mathbb{R}^d$ where d is the dimensionality of the Fisher Score which is equal to the number of parameters in the generative model $P(\mathbf{X} \mid \theta)$ and is independent of the number of data points in the set T.

Then, the Fisher Information Matrix (FIM) is calculated as follows:

$$\mathbf{I} = E_{x \sim P(x|\theta)}\{U_{\mathbf{X}} U_{\mathbf{X}}^T\} \tag{2.2}$$

Based on [52], the class of probability models $P(\mathbf{X} \mid \theta)$ defines Riemannian manifold M_{Θ} and the Fisher Score $U_{\mathbf{X}} = \nabla_{\theta} log P(\mathbf{X} \mid \theta)$ transform an example (or set of examples) into a feature vector that is a point in the gradient space of the manifold M_{Θ}. This gradient $U_{\mathbf{X}}$ defines the steepest ascent that increases the log-likelihood of the generative model. In other words, fisher kernel theory transforms data points from data space to (generative) model space. Having the Fisher information matrix $\mathbf{I}$, [52] proposed to measure the similarity between two sets of examples, i.e., $\mathbf{X}$ and $\mathbf{Y}$ using the Fisher Kernel

$$K(\mathbf{X}, \mathbf{Y}) = U_{\mathbf{X}}^T \mathbf{I}^{-1} U_{\mathbf{Y}} \tag{2.3}$$

GMM-based Fisher Vector for Image Classification

Although BOW encoding scheme can be helpful for set representation, one can argue that it only captures the first-order statistics of data to obtain WSI representation. This statement can be validated if we further look into the encoding process. Let us say we apply k-means as a clustering step. Then, each patch is encoded in WSI representation based on its distance to the cluster centers, which is the average of data points in the clusters. More precisely, BOW only employs membership of each data point which is merely specified based on the distance to the cluster centers (first-order statistics).

Inspired by this limitation, authors in [89] replaced k-means with Gaussian Mixture Model (GMM). They introduced GMM-based Fisher Vector, which can be calculated using the gradient of the log-likelihood of the GMM with respect to parameters, i.e., mixing coefficients, means, and variances, given some observations. Unlike BOW, the Fisher vector

representation can capture higher-order statistics information while obtaining representation for a set of local descriptors. In fact, [100] showed that BOW is a special case of GMM-based Fisher vector when the gradient is only calculated with respect to the weight parameters of a GMM. Authors in [89] employed these derivations to encode each set of local descriptors to one global image representation. To this end, first, note that the Fisher information matrix is symmetric and positive definite [52] and subsequently the $\mathbf{I^{-1}}$ matrix has the same characteristics. Considering this, if we replace $\mathbf{I}^{-1}$ with its Cholesky decomposition $\mathbf{I}^{-1} = \mathbf{L}^T\mathbf{L}$ the Fisher Kernel in Eq. 2.3 can be written as

$$K(\mathbf{X}, \mathbf{Y}) = U_{\mathbf{X}}^T \mathbf{L}^T \mathbf{L} U_{\mathbf{Y}} \tag{2.4}$$

Then, the Fisher Vector for a set of data points $\mathbf{X}$ is defined as normalized gradient of the log-likelihood function of generative model [100]:

$$\textit{Fisher Vector} = \mathbf{L}U_{\mathbf{X}} = \mathbf{L}\nabla_\theta log P(\mathbf{X} \mid \theta) \tag{2.5}$$

In practice, to remove the dependency of Fisher Vector on the sample size, it is normalized to the sample size T when this is being calculated for a set of data points $\mathbf{X} = \{\mathbf{x}_t, t = 1, \dots T\}$,:

$$\textit{Fisher Vector} = \frac{1}{T}\mathbf{L}\nabla_\theta log P(\mathbf{X} \mid \theta) = \mathbf{L}\frac{1}{T}\sum_{t=1}^{T} \nabla_\theta log P(\mathbf{x}_t \mid \theta) \tag{2.6}$$

This enables us to derive a representation for a set of data points in a metric space. This is due to the fact given a set of independent data points, the gradient of the log-likelihood function of the generative model is the summation of gradients of the generative model given each data point. This is notably similar to the counting procedure in the BOW method. However, instead of merely employing first-order statistics information (captured using the distance of data points from k-means cluster centers), Fisher Vector uses higher-order statistics by the help taking the gradient of the generative model with respect to its parameters. Another interesting property of Fisher Vector is its robustness to background image information [90].

For GMM as a generative model, the formulas for the gradient with respect to the

GMM parameters, i.e., mixing weights, means, and variances, are presented in the [100]. Based on the Eq. 2.6, after having the gradients, the only remaining element to calculate the Fisher Vector for a set of given observations is $\mathbf{L}$ which is the square-root of the inverse of the FIM. In the GMM case, the authors in [100] found a diagonal approximation of FIM, which reduces the computational load of calculating the normalizing matrix, i.e., the square root of the inverse of the FIM.

Improving GMM-based Fisher Vector

To improve the GMM-based Fisher Vector, several heuristics have been proposed in the literature. In the following, some of these proposed improvements are briefly reviewed.

L2 Normalization: According to the [90], the Fisher Vector automatically nearly ignore the background (image independent) information. However, based on the [90], Fisher vector still depends on the proportion of foreground image-specific information. As a result, two images containing the same object but a different proportion of background information (e.g. same object at different resolutions) will have different Fisher Vectors. This can be problematic in images that a high proportion of the image is background. To remove the dependency, the authors proposed to L2 normalize the gradient vector. Their experiments showed that this L2 normalization step could improve the performance of the Fisher Vector.

Power Normalization: Another improvement proposed by [90] is power normalization. The empirical observation inspires this normalization step that as the number of Gaussians increases, Fisher Vectors become more sparse. This means that for any given dimension, the distribution of features experiences its peak around zero. The main issue with this sparsity is the fact that this degrades the performance of L2 distance. To deal with this limitation, the authors propose to either employ a more suitable distance for sparse vectors like L1 distance or "unsparsify" the vectors in a way the L2 similarity is preserved. For achieving the latter, the authors [90] proposed to apply the function

$$f(z) = \text{sign}(z)|z|^{\alpha}, \tag{2.7}$$

where $0 \leq \alpha \leq 1$ is a normalization parameter. The authors showed that this normalization step improves the performance of the Fisher Vector. To combine power and L2 normalizations, authors in [90] proposed first to apply power and L_2 normalization steps.

Supervision in GMM-based Fisher Vector: Although Fisher Vector provides many advantages, most importantly a natural way for set representation, one disadvantage

of this approach is the fact that there is no clear way to incorporate labels (weak labels) or available information in this framework. In fact, according to the Tommi S. Jaakkola and David Haussler, [52]: "A kernel classifier employing the Fisher kernel derived from a model that contains the label as the latent variable is, asymptotically, at least as good a classifier as the maximum a posteriori labelling based on the model". Motivated by this, some attempts have been made to incorporate label information in GMM. For example, in [26], the authors proposed a heuristic approach to train one specific GMM per class. Then for test time encoding, they proposed to fold representation obtained from all GMMs together to obtain on global representation. Although this approach showed significant improvement in the performance compared with unsupervised Fisher Vector, this method leads to intractable high dimensional representations for problems with a large number of categories as the dimension of the Fisher Vector increases linearly with the number of categories. To mitigate this limitation, other heuristics have been proposed [89]. However, there is no straightforward method to incorporate the label information in the training of GMMs. As we will see in the next sections, this limitation can be easily solved with the help of semi-supervised deep generative models.

Fisher Vector for WSI Representation Learning

Having all discussed characteristics of Fisher Vector in mind, it can be seen as an alternative for MIL in the whole slide representation task. Remarkably, unlike the MIL scheme, the training step in the Fisher Vector is instance-based which significantly bypasses the memory bottleneck. Although the Fisher Kernel theory seems to be a suitable approach for WSI representation, there are very limited works that have employed this theory for the WSI representation task. One reason for this can be the fact that fisher vector theory is formulated in a fully unsupervised manner while considering the challenges inherent to pathology images (e.g., challenging textures, colour variations, etc.) employing available information, i.e., primary site or primary diagnosis of the WSI in obtaining an efficient global representation is necessary. To the Best of my knowledge, the only papers that have employed Fisher Vector for WSI representation are [109, 110]. Unfortunately, both of these papers employ GMM-based Fisher Vector in an unsupervised manner which is not enough for a difficult task like WSI representation learning. To mitigate this limitation, both [109, 110] employed a supervised mapping as a separate second step to refine the fisher vector representations. However, there are still some limitations in their approach. First, the supervision has not been incorporated in the learning process of generative model parameters. Second, they employed GMM as the generative model, which is known to be sub-optimal as this cannot be applied in an end-to-end manner to the images. Besides,

GMM is not able to fully capture the natural clustering of the patch descriptors. This is due to the inefficient training scheme in GMM and the fact that by employing the GMM, no more than second-order statistics of data are captured using Fisher Vector. To train the generative model in an end-to-end manner, incorporate label information and enable the Fisher Vector to capture higher-order statistics, recently, Fisher Vectors based on the deep generative models have been introduced. More precisely, Adversarial Fisher Vector [132], and Fisher Vector-Variational Autoencoder (VAE) [92] based on the Generative Adversarial Network (GAN), and VAE are the latest works on deep generative model-based Fisher Vectors. Although recently GAN and VAE-based Fisher Vectors have been introduced in the general computer vision community, they have not been used for ?? representation. Deep generative models along with Fisher Vector theory can be a very good candidate for learning compact WSIs representations without any training bottleneck.

This thesis develops a framework based on deep generative models that can achieve:

1. Permutation-invariant WSI representation trained on instances.
2. Representations for both WSIs and patches.
3. Representations guided by available information, e.g., primary site and diagnosis.
4. Compact representations (including binary or sparse).
5. Multi-magnification informed representations.

Chapter 3

ESH-A Non-alternating Graph Hashing Algorithm for Large-scale Image Search

3.1 Prologue

The content of this Chapter is based on the following paper published during the Ph.D. research:

1. **S. Hemati**, et al. *A Non-alternating Graph Hashing Algorithm for Large-scale Image Search* Computer Vision and Image Understanding, Volume 219, May 2022, 103415

3.2 Introduction

Here, we propose improving the current enhancements for spectral hashing formulation in terms of training-time memory usage, time complexity, and retrieval performance. Although different methods have been proposed to achieve better-relaxed solutions while reducing complexity, we should note that all of the proposed methods, including LGHSR, DGH, RDSH, and DSH, employ an alternating approach between two or more optimization variables and keep the binary decision variable within the problem. However, we argue that this increases the runtime complexity and is unnecessary. In this section, we propose an efficient graph hashing methods built on top of spectral hashing formulation, i.e.,

direct optimization of binary codes. More precisely, we develop a new formulation that allows us to convert the optimization problem raised in SH [123] with $n \times k$ parameters to a problem with $d \times k$ decision variables (d dimensionality and k number of bits), where $d \ll n$. In addition, in the proposed formulation, the optimization problem is a function of one decision variable, and as a result, the optimization is non-alternating, which significantly reduces the computational complexity. Two optimization algorithms were employed to obtain a solution for the suggested optimization problem. The results on four public datasets show that the proposed formulation outperforms the state-of-the-art approaches and, at the same time, is more efficient than the recent algorithms proposed for the graph Laplacian hashing problem.

3.3 Spectral Hashing

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ denote a mean zero and unit variance data matrix with rows representing training data points, each with a dimensionality of d. The original spectral hashing formulation [123] is as follows:

$$\begin{aligned} &\text{argmin} \sum_{i,j} A_{i,j} \|\mathbf{b}_i - \mathbf{b}_j\|^2 \\ &\text{s.t.} \sum_i \mathbf{b}_i = \mathbf{0}, \quad \frac{1}{n} \sum_i \mathbf{b}_i \mathbf{b}_i^T = \mathbf{I}_k, \quad \mathbf{b}_i \in \{-1, 1\}^k, \end{aligned} \tag{3.1}$$

where $\mathbf{b}_i$ and $\mathbf{b}_j$ are k-bit binary codes corresponding to the i-th and j-th data points, respectively, $\mathbf{I}_k$ is a k by k identity matrix, and $A_{i,j}$ is the $(i,\ j)$ entry of the affinity matrix $\mathbf{A}$, which measures the similarity between the i-th and j-th data points. Let us denote the k-bit binary representation of the data matrix $\mathbf{X}$ by $\mathbf{B} \in \{-1, 1\}^{n \times k}$. Suppose $\mathbf{D}$ is an $n \times n$ diagonal matrix whose diagonal element i is given by $D_{i,i} = \sum_j A_{i,j}$. The matrix form of the optimization problem can then be written as follows:

$$\begin{aligned} \underset{\mathbf{B}}{\text{argmin}} \quad & Tr\{\mathbf{B}^T(\mathbf{D} - \mathbf{A})\mathbf{B}\} \\ \text{s.t.} \quad & \mathbf{B}^T\mathbf{B} = n\mathbf{I}_k, \quad \mathbf{B}^T\mathbf{1}_{n \times 1} = \mathbf{0}, \quad \mathbf{B} \in \{-1, 1\}^{n \times k}., \end{aligned} \tag{3.2}$$

where Tr represents the trace operation, $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the graph Laplacian matrix [75], and $\mathbf{1}_{n \times 1}$ is a vector of length n, where all elements are 1.

To reduce the computational complexity in calculating the affinity matrix $\mathbf{A}$ from $O(n^2d)$ to $O(n)$, we employ the low-rank approximation of the hashing with graphs paper [78]. This approximation is based on a small number (m) of data points, that is, anchors, with $m \ll n$. In general, the m cluster centers obtained from the K-means algorithm, after running a few iterations, are considered as m anchors u_j for $j = 1, \ldots, m$. These anchors are used to construct an $n \times m$ affinity matrix $\mathbf{Z}$ using a Gaussian kernel of pairwise distances between the n observed data points and the m anchors. That is, the (i, j) entry of the matrix $\mathbf{Z}$ is given by $Z_{i,j} = K(x_i, u_j)/N_0$, where x_i is the i-th data point, $K(x_i, u_j) = \exp(\|x_i - u_j\|_2^2/\sigma^2)$, and σ is a hyperparameter. To impose sparsity on $\mathbf{Z}$, only distances from the $s \ll m$ nearest neighbor anchors are kept, and the rest are set to zero. Furthermore, the normalization factor N_0 is calculated as $\sum_{j \in <i>} K(x_i, u_j)$, and the normalized low-rank approximation of the affinity matrix (which is also sparse) can be calculated as $\mathbf{A} = \mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{Z}^T$ with $\mathbf{\Lambda} = diag(\mathbf{Z}^T\mathbf{1})$. Finally, note that after using this low-rank approximation, we have $\mathbf{D} = \mathbf{I}$.

3.4 Efficient Spectral Hashing

To develop the Efficient Spectral Hashing (ESH) algorithm, first recall that the problem in Eq. 3.2 can be written as follows:

$$\begin{aligned} \underset{\mathbf{B}}{\text{argmin}} \quad & -Tr\{\mathbf{B}^T\mathbf{A}\mathbf{B}\} \\ \text{s.t.} \quad & \mathbf{B}^T\mathbf{B} = n\mathbf{I}_k, \quad \mathbf{B}^T\mathbf{1}_{n\times 1} = \mathbf{0}, \quad \mathbf{B} \in \{-1, 1\}^{n\times k} \end{aligned} \tag{3.3}$$

As pointed out, even for a single bit, this problem is extremely difficult, and continuous relaxations are generally used for simplification. However, naive relaxation generally degrades the quality of the binary codes, and for this reason, finding better continuous relaxations has been an area of study. Now, we take the first step toward efficiency. we assume that the binary codes can be obtained from the following simple model:

$$\mathbf{B} = sgn(\mathbf{X}\mathbf{W}), \tag{3.4}$$

where $\mathbf{W}$ is a $d \times k$ matrix and $sgn(\cdot)$ denotes the element-wise sign operation. Given this model, the $n \times k$ optimization variable is replaced with the $d \times k$ matrix $\mathbf{W}$. Clearly, $d \ll n$, and thus the proposed model reduces the search space. In this case, the optimization

problem in Eq.3.3 will convert into the following problem:

$$\begin{aligned} \underset{\mathbf{W}}{\text{argmin}} \quad & -Tr\{sgn(\mathbf{W}^T\mathbf{X}^T)\mathbf{A}sgn(\mathbf{XW})\} \\ \text{s.t.} \quad & sgn(\mathbf{XW})^T sgn(\mathbf{XW}) = n\mathbf{I}_k, \\ & sgn(\mathbf{XW})^T\mathbf{1}_{n\times 1} = \mathbf{0}. \end{aligned} \tag{3.5}$$

Note that after obtaining $\mathbf{W}$, the binary matrix $\mathbf{B}$ can be calculated using Eq. 3.4. Although the aforementioned model reduces the search space, owing to $sgn(\cdot)$, this is still a discrete optimization problem and is difficult to solve. Removing $sgn(\cdot)$ results in poor binary codes owing to the accumulated quantization error. However, if we ensure that the elements of $\mathbf{XW}$ are sufficiently close to +1 or -1, then the $\mathbf{XW} \approx sgn(\mathbf{XW})$ approximation is no longer unrealistic. To this end, we propose a novel regularization term that pushes the elements of $\mathbf{XW}$ closer to ± 1 without adding any other optimization variables. This is particularly different from methods such as LGSHR, DGH, RDSH, and DSH, all of which do so by including one or more optimization variables, which leads to alternating optimization approaches. The proposed relaxed optimization problem function has the following form:

$$\begin{aligned} \underset{\mathbf{W}}{\text{argmin}} \quad & -Tr\{\mathbf{W}^T(\mathbf{X}^T\mathbf{AX})\mathbf{W}\} + \frac{\alpha}{2}\| \, |\mathbf{XW}| - \mathbf{J}\|_F^2 \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{X}^T\mathbf{XW} = n\mathbf{I}_k, (\mathbf{XW})^T\mathbf{1}_{n\times 1} = \mathbf{0}, \end{aligned} \tag{3.6}$$

where $\mathbf{J}$ is an $n \times k$ matrix with all elements equal to 1, $|\cdot|$ represents the element-wise absolute value, and $\|\cdot\|_F$ denotes the Frobenius norm. First, note that, because the data are zero-centered, the last constraint in Eq. 3.6 is already satisfied and thus we remove it from the problem statement. In addition, to further simplify the problem, we transform the orthogonality of the columns of $\mathbf{XW}$ into the orthonormality of the columns of $\mathbf{W}$. Finally, we normalize the cost function by the number of samples to obtain smoother training. Taking these changes into account and denoting $\mathbf{X}^T\mathbf{AX}$ as $\mathbf{S}$, which is a $d \times d$ matrix, the problem in Eq. 3.6 will take the following form:

$$\begin{aligned} \underset{\mathbf{W}}{\text{argmin}} \quad & \mathcal{L}(\mathbf{W}) = \frac{-1}{n} Tr\{\mathbf{W}^T\mathbf{SW}\} + \frac{\alpha}{2n}\| \, |\mathbf{XW}| - \mathbf{J}\|_F^2 \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{W} = \mathbf{I}_k. \end{aligned} \tag{3.7}$$

To solve this constraint optimization problem, two algorithms, namely, the projected gradient and Stiefel manifold optimization, are employed.

3.4.1 Projected Gradient-ESH1

In this algorithm, during each iteration, the matrix $\mathbf{W}$ is updated using the gradient descent method as if there is no constraint, and the updated matrix is then projected to the closest matrix in the feasible set. The derivative of the first term in Eq. 3.7 is $2\mathbf{SW}$. For the derivative of the second part, first note that we have $|\mathbf{XW}| = sgn(\mathbf{XW}) \odot \mathbf{XW}$, where $\odot$ denotes the Hadamard (element-wise) product. The derivative can then be calculated as follows:

$$\begin{aligned}
&\frac{\partial}{\partial \mathbf{W}} \| \, |\mathbf{XW}| - \mathbf{J}\|_F^2 \\
&= \frac{\partial}{\partial \mathbf{W}} Tr\left(\left(|\mathbf{XW}| - \mathbf{J}\right)^T \left(|\mathbf{XW}| - \mathbf{J}\right) \right) \\
&= 2\, \mathbf{X}^T sgn\left(\mathbf{XW}\right) \odot \left(|\mathbf{XW}| - \mathbf{J}\right) \\
&= 2\, \mathbf{X}^T \, \left(\mathbf{XW} - sgn\left(\mathbf{XW}\right)\right).
\end{aligned} \tag{3.8}$$

This derivative is valid everywhere, except for zero. For zero, we define the derivative as equal to zero. In this case, if the derivative of the cost function in Eq. 3.8 in iteration p is $\mathbf{G}$, it can then be calculated as follows:

$$\mathbf{G}_p = \frac{-2}{n}\mathbf{SW}_p + \frac{\alpha}{n}\mathbf{X}^T(\mathbf{XW}_p - sgn(\mathbf{XW}_p)), \tag{3.9}$$

In this case the learning rule for minimizing the expression can be written as follows:

$$\mathbf{W}_p = \mathbf{W}_{p-1} - \eta \mathbf{G}_{p-1}, \tag{3.10}$$

where η is the learning rate parameter. To complete the iteration step, we project $\mathbf{W}$ onto the feasible set. This is equivalent to finding the closest matrix $\mathbf{Q}$ to $\mathbf{W}$ such that $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}_k$. This problem is known as the projection on the Stiefel manifold, which can be formulated as follows:

$$\begin{aligned}
&\text{Proj}(\mathbf{W}) := \underset{\mathbf{Q}}{\text{argmin}} \quad \|\mathbf{W} - \mathbf{Q}\|_F^2 \\
&\text{s.t.} \quad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_k.
\end{aligned} \tag{3.11}$$

Fortunately, there is a closed form solution to this problem:

$$\mathbf{Q} = \text{Proj}(\mathbf{W}) = \mathbf{U}\mathbf{I}_{d\times k}\mathbf{V}^T, \tag{3.12}$$

where $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the Singular Value Decomposition (SVD) of $\mathbf{W}$. Interested readers can see [84] for a detailed proof. Algorithm 1 summarizes the proposed projected gradient method.

Algorithm 1 Proposed ESH1 Algorithm

Input: Training data $\mathbf{X} \in \mathbb{R}^{n\times d}$, affinity matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, number of iterations N, learning rate η.
Output: Binary matrix $\mathbf{B} \in \{-1, 1\}^{n\times k}$.
Initialization: Initialize an orthogonal matrix $\mathbf{W}$: $\mathbf{W}_0 \in \mathbb{R}^{d\times k}$

$\mathbf{S} \leftarrow \mathbf{X}^T\mathbf{A}\mathbf{X}$
Compute α according to Eq.3.19
for $p = 1, 2, \ldots, N$ **do**
  $\mathbf{W}_p \leftarrow \mathbf{W}_{p-1} - \ldots$
  $\ldots \quad \eta\left(\frac{-2}{n}\mathbf{S}\mathbf{W}_{p-1} + \frac{\alpha}{n}\mathbf{X}^T(\mathbf{X}\mathbf{W}_{p-1} - sgn(\mathbf{X}\mathbf{W}_{p-1}))\right)$
  $\mathbf{U}\Sigma\mathbf{V}^T \leftarrow \text{SVD}(\mathbf{W}_p)$
  $\mathbf{Q} \leftarrow \text{Proj}(\mathbf{W}) = \mathbf{U}\mathbf{I}_{d\times k}\mathbf{V}^T$
  $\mathbf{W}_p \leftarrow \mathbf{Q}$
end for
$\mathbf{B} \leftarrow sgn(\mathbf{X}\mathbf{W})$

3.4.2 Stiefel Manifold Optimization-ESH2

The problem in Eq. 3.7 is an optimization on the Stiefel manifold, and methods developed for the optimization on manifolds can be used to obtain a solution. In this study, we employ the method in [124], where an efficient algorithm has been proposed to preserve the updated $\mathbf{W}$ on the Stiefel manifold during each iteration. Briefly, to preserve the orthogonality constraint on $\mathbf{W}$ during each iteration, we define the skew-symmetric matrix $\mathbf{F}$ in iteration $p-1$ as $\mathbf{F}_{p-1} = \mathbf{G}_{p-1}\mathbf{W}_{p-1}^T - \mathbf{W}_{p-1}\mathbf{G}_{p-1}^T$, and denote the updated version of the $\mathbf{W}$ as $\mathbf{Y}(\tau)$, then using a Crank-Nicolson-like scheme we have

$$\mathbf{Y}(\tau)_p = \mathbf{W}_{p-1} - \frac{\tau}{2}\mathbf{F}_{p-1}\left(\mathbf{W}_{p-1} + \mathbf{Y}(\tau)_p\right), \tag{3.13}$$

where τ is the step size parameter. In this case, the closed-form solution for $\mathbf{Y}(\tau)$ is given as follows:

$$\mathbf{Y}(\tau)_p = \left(\mathbf{I} + \frac{\tau}{2}\mathbf{F}_{p-1}\right)^{-1}\left(\mathbf{I} - \frac{\tau}{2}\mathbf{F}_{p-1}\right)\mathbf{W}_{p-1}. \tag{3.14}$$

Following [124], we employ the Barzilai-Borwein (BB) method to reduce the total number of iterations:

$$\tau_p = \frac{|\,Tr\left((\mathbf{M}_p)^T(\mathbf{Y}_p)\right)|}{Tr\left((\mathbf{Y}_p)^T(\mathbf{Y}_p)\right)}, \tag{3.15}$$

where $\mathbf{M}_p = \mathbf{W}_p - \mathbf{W}_{p-1}$, and $\mathbf{Y}_p = \nabla\mathcal{L}(\mathbf{W}_p) - \nabla\mathcal{L}(\mathbf{W}_{p-1})$, in which $\nabla\mathcal{L}(\mathbf{W}) = \mathbf{G} - \mathbf{W}\mathbf{G}^T\mathbf{W}$ is the gradient of the loss function in the tangent planes. Algorithm 2 summarizes the proposed manifold optimization method.

Algorithm 2 Proposed ESH2 Algorithm

Input: Training data $\mathbf{X} \in \mathbb{R}^{\mathrm{n}\times\mathrm{d}}$, affinity matrix $\mathbf{A} \in \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$, number of iterations N.
Output: Binary matrix $\mathbf{B} \in \{-1, 1\}^{n\times k}$.
Initialization: Initialize an orthogonal matrix $\mathbf{W}$: $\mathbf{W}_0 \in \mathbb{R}^{\mathrm{d}\times\mathrm{k}}$, step size τ.

$\mathbf{S} \leftarrow \mathbf{X}^T\mathbf{A}\mathbf{X}$
Compute α according to Eq.3.19
for $p = 1, 2, \ldots, N$ **do**
  $\mathbf{G}_{p-1} \leftarrow \frac{-2}{n}\mathbf{S}\mathbf{W}_{p-1} + \frac{\alpha}{n}\mathbf{X}^T(\mathbf{X}\mathbf{W}_{p-1} - sgn(\mathbf{X}\mathbf{W}_{p-1}))$
  $\mathbf{F}_{p-1} \leftarrow \mathbf{G}_{p-1}\mathbf{W}_{p-1}^T - \mathbf{W}_{p-1}\mathbf{G}_{p-1}^T$
  $\mathbf{Y}(\tau)_p \leftarrow \left(\mathbf{I} + \frac{\tau}{2}\mathbf{F}_{p-1}\right)^{-1}\left(\mathbf{I} - \frac{\tau}{2}\mathbf{F}_{p-1}\right)\mathbf{W}_{p-1}$
  $\mathbf{W}_p \leftarrow \mathbf{Y}(\tau)_p$
  $\tau \leftarrow \frac{|Tr\left((\mathbf{M}_p)^T(\mathbf{Y}_p)\right)|}{Tr((\mathbf{Y}_p)^T(\mathbf{Y}_p))}$
end for
$\mathbf{B} \leftarrow sgn(\mathbf{X}\mathbf{W})$

3.4.3 Out of Sample: Hashing New Data

Thus far, we have derived a binary representation for the training data. To obtain a binary representation for a new data point $\mathbf{x}^*$, the same approach as in many spectral hashing

methods can be used [76, 72]. More precisely, let $\mathbf{b}(\mathbf{x}^*)$ be the binarized version of $\mathbf{x}^*$. Then, using a similar approach as applied for training, we can write the following:

$$\begin{aligned} \underset{\mathbf{b}(\mathbf{x}^*)}{\text{argmin}} \quad & \sum_{i}^{n} A(\mathbf{x}_i, \mathbf{x}^*) \|\mathbf{b}_i - \mathbf{b}(\mathbf{x}^*)\|_2^2, \\ \text{s.t.} \quad & \mathbf{b}(\mathbf{x}^*) \in \{-1, 1\}^k \end{aligned}, \tag{3.16}$$

where $A(\mathbf{x}_i, \mathbf{x}^*) = \mathbf{z}_i \Lambda^{-1} \mathbf{z}(\mathbf{x}^*)$, and $\mathbf{z}_i$ is the i-th row of matrix $\mathbf{Z}$. By expanding $\|\mathbf{b}_i - \mathbf{b}(\mathbf{x}^*)\|_2^2$, this problem can be written as

$$\begin{aligned} \underset{\mathbf{b}(\mathbf{x}^*)}{\text{argmax}} \quad & \langle \mathbf{b}(\mathbf{x}^*), \mathbf{B}^T \mathbf{Z} \Lambda^{-1} \mathbf{z}(\mathbf{x}^*) \rangle \\ \text{s.t.} \quad & \mathbf{b}(\mathbf{x}^*) \in \{-1, 1\}^k. \end{aligned} \tag{3.17}$$

In this case, the solution is

$$\mathbf{b}(\mathbf{x}^*) = sgn(\mathbf{B}^T \mathbf{Z} \Lambda^{-1} \mathbf{z}(\mathbf{x}^*)). \tag{3.18}$$

3.4.4 Implementation Note

To avoid tuning the parameters, we set the learning rate parameter for all datasets and experiments to a fixed value ($\eta = 0.01$). Furthermore, we propose a novel method for automatically determining the regularization parameter α for each dataset. To this end, we call the first and second terms in Eq. 3.7 $\mathbf{T}_1$ and $\mathbf{T}_2$, respectively. If we denote the initialization of $\mathbf{W}$ as $\mathbf{W}^0$, then we propose setting α such that the importance of the first and second parts of the cost function is the same in the first iteration:

$$\left|\mathbf{T}_1(\mathbf{W}^0)\right| = \frac{\alpha}{2} \left|\mathbf{T}_2(\mathbf{W}^0)\right| \Rightarrow \alpha = \left| \frac{2\mathbf{T}_1(\mathbf{W}^0)}{\mathbf{T}_2(\mathbf{W}^0)} \right|. \tag{3.19}$$

The ESH code is provided in this Github page.

3.5 Datasets, Evaluation, and Results for ESH

3.5.1 Datasets

The performance of ESH1 and ESH2 are evaluated on standard benchmark datasets, namely CIFAR-10 [67], LabelMe-12-50K [114], a medical image dataset NCT-CRC-HE-100K [82], and NUS-WIDE [13]. These datasets provide a total of **489,000 images** for learning and testing.

- The **CIFAR-10 dataset** is a 10-class dataset consisting of 60,000 color images of size 32×32 pixels containing classes like airplane, horse, cat, ship, frog and more.
- The **LabelMe-12-50K dataset** is a 12-class dataset containing 50,000 images of size 256×256 pixels. This dataset is highly imbalanced such that five classes constitute 91% of all images while there is one class that only contains 0.6% of the samples. The images of this dataset have multiple label values between zero and one. In our experiments, same as previous works that employed this dataset for evaluating hashing algorithms, we choose the class of the largest label value as the image label.
- The **NCT-CRC-HE-100K dataset** is 9-class histopathology dataset containing 100,000 non-overlapping image patches from hematoxylin & eosin stained (H&E) images of human colorectal cancer and normal tissue. All images are 224×224 pixels and color-normalized. Tissue classes include adipose, background, debris, lymphocytes, mucus, smooth muscle, normal colon mucosa, cancer-associated stroma, and colorectal adenocarcinoma epithelium.
- The **NUS-WIDE dataset** is a multi-label dataset that contains 269,000 images collected from Flickr. This database contains 81 ground-truth concepts.

3.5.2 Hash Code Evaluation

To evaluate the performance of the ESH1 and ESH2 in comparison to other methods, we use standard measures for image retrieval quality assessment. These measures include mean Average Precision (mAP), precision at N samples (i.e., precision@1000), and precision of Hamming distance with a radius of 2 (precision@r=2). Briefly, mAP measures the overall performance of the retrieval over all classes, whereas precision@N calculates the proportion of true positive over top N retrieved samples. Finally, precision@r=2 measures

the precision over all retrieved images that have a Hamming distance equal or less than 2 from the query image. For LabelMe-12-50K, which is highly imbalanced, following the common setting in previous works [37], the macro average mAP is reported to avoid biased performance increase for frequent classes to dominate the overall performance. In order to use the results provided by other methods, the experimental setting here has been adopted from previous works.

Table 3.1: Comparison of retrieval performance, for 16, 32, 64 and 128 bits based on macro mAP (average over classes) for LabelMe-12-50k dataset represented by 4096-D VGG-FC7 descriptor. The best performance values are highlighted in boldface.

	macro mAP %			
Method	16 Bit	32 Bit	64 Bit	128 Bit
SH	12.60	12.59	12.24	-
SpH	13.59	15.10	17.03	-
KMH	13.36	15.47	16.58	-
BA	16.96	18.42	20.80	-
ITQ	17.61	18.65	20.10	21.49
DGH	21.45	22.74	25.41	26.77
LGHSR	21.10	23.49	23.98	22.85
KNNH	20.13	23.34	26.06	27.62
DSH	**24.70**	23.78	24.35	22.39
SCQ	22.89	24.95	26.50	26.35
ESH1	22.95	25.67	27.59	28.94
ESH2	22.87	**26.85**	**28.20**	**29.61**

3.5.3 Results on LabelMe-12-50K

We report the macro mAP for this imbalanced dataset. A VGG network was employed for the feature extraction. For the test and train split, similar to the common setting, we sample 10% of each class as the test data and 90% as the training set. Table 3.1 presents the performance of ESH1 and ESH2 compared with the other methods. It is obvious that, except for the 16-bit setting where DSH obtains a better performance, for the 32, 64, and 128 bits, ESH1 and ESH2 attain a better macro mAP compared to other algorithms. In Table 3.1, SpH indicates spherical hashing [43], and KMH stands for k-means hashing [36].

Table 3.2: Comparison of retrieval performance, based on mAP, precision@1000, and precision@r=2 on CIFAR-10 represented by 4096-D VGG-FC7 features. The best performance values are highlighted in boldface.

	mAP %				precision % @1000				precision@r=2		
Method	16Bits	32 Bits	64Bits	128Bits	16Bits	32Bits	64Bits	128Bits	16Bits	32Bits	64Bits
SH	18.31	16.54	15.78	-	-	-	-	-	-	-	-
SpH	18.82	20.93	23.40	-	-	-	-	-	-	-	-
KMH	18.68	20.82	22.87	-	-	-	-	-	-	-	-
BA	25.38	26.16	27.99	-	-	-	-	-	-	-	-
ITQ	26.23	26.73	27.90	29.22	36.71	38.79	40.75	42.53	**39.33**	27.71	00.06
DGH	27.73	27.44	28.01	29.47	40.36	39.46	39.70	40.69	38.73	42.25	44.04
LGSHR	27.83	25.87	22.12	19.66	39.98	42.84	41.82	40.45	38.45	46.11	32.75
KNNH	29.25	30.55	32.60	33.68	38.13	40.51	43.32	44.57	37.66	24.95	03.43
DSH	27.72	25.36	22.12	19.55	40.36	42.87	41.80	40.57	38.73	45.69	26.15
SCQ	27.52	27.42	30.34	32.25	34.24	36.51	40.29	43.15	30.58	38.24	25.14
ESH1	**32.11**	33.08	34.47	34.92	38.67	42.15	44.75	45.70	35.03	43.59	**44.26**
ESH2	31.59	**33.44**	**34.72**	**35.29**	**41.32**	**43.47**	**45.16**	**45.95**	38.73	**46.69**	43.58

3.5.4 Results on CIFAR-10

For CIFAR-10, similar to [37], each image is represented by a deep 4096-D feature extracted from the VGG network [106]. For the test & train split, 10% of each class is sampled as the query set, and the remaining instances are sampled as the training set. Table 3.2 shows the results for CIFAR-10 based on mAP, precision@1000, and precision@r=2. As can be seen, ESH1 and ESH2 outperform the state-of-the-art approaches, namely, LGHSR, KNNH, and DSH, based on mAP and precision@1000 for all 16-, 32-, 64-, and 128-bit settings. For precision@r=2, ESH1 and ESH2 achieved the best performance for 64 and 16 bits, respectively, and competitive results for 32 bits. The first row in Fig. 3.1 compares the performance of the ESH algorithms with recent competitive algorithms, LGHSR, DSH, and KNNH based on precision-recall graphs. Clearly, the ESH algorithms achieved a better performance than the recent methods.

3.5.5 Results on NUS-WIDE

For this multi-label dataset, similar to the common setting [78], VGG features were used for image representation. Images with labels among the 21 most frequent labels (195,834

Table 3.3: Comparison of retrieval performance based on mAP, precision@5000, and precision@r=2 on NUS-WIDE dataset represented by VGG-F deep features. The best performance values are highlighted in boldface.

	mAP %				precision % @5000				precision@r=2		
Method	16Bits	32Bits	64Bits	128Bits	16Bits	32Bits	64Bits	128Bits	16Bits	32Bits	64Bits
LSH	40.45	48.04	46.75	-	47.95	55.29	58.95	-	-	-	-
SH	44.74	42.60	42.36	-	59.15	54.98	54.18	-	-	-	-
AGH	49.80	47.14	44.72	-	70.43	70.29	69.29	-	-	-	-
SGH	48.86	49.10	51.33	-	64.92	66.04	69.01	-	-	-	-
ITQ	52.09	53.12	54.04	54.85	64.34	66.62	68.48	69.54	67.31	51.91	05.21
DGH	51.35	52.21	53.34	55.96	66.33	65.35	66.02	67.20	65.14	68.41	**70.19**
LGHSR	50.06	47.72	45.71	44.11	**68.42**	68.09	67.92	66.95	**68.87**	71.98	45.27
KNNH	55.12	57.03	**58.61**	**59.45**	66.77	69.83	71.07	72.13	68.61	49.78	09.58
DSH	49.87	47.07	45.67	44.03	68.05	68.17	67.83	67.15	68.67	71.86	39.93
SCQ	56.34	56.21	56.10	53.54	68.45	70.64	70.81	69.00	68.74	70.92	08.98
ESH1	56.32	56.89	57.47	57.28	67.58	69.47	71.46	72.31	64.64	**72.30**	60.14
ESH2	**56.54**	**57.16**	57.71	57.53	68.20	**70.80**	**72.14**	**72.45**	64.34	72.06	62.08

images) were selected. We randomly sampled 100 images from each class to construct the test set, and the remaining images were used to train the hash function and populate the hash table. For the mAP and precision calculation, two images are considered neighbors if they share at least one common label. Table 3.3 presents the validation of the ESH1 and ESH2 algorithms on this dataset. Here, LSH indicates locality sensitivity hashing [31], PCAH is semi-supervised hashing [119], and SGH represents salable graph hashing [54]. For mAP, ESH1 and ESH2 outperform DSH, which is the most recent method based on a spectral hashing formulation and has a higher complexity than ESH. For the NUS-WIDE dataset, in some cases, ESH algorithms do not outperform the state-of-the-art method KNNH. Note that with the KNNH method, there exists an $O(n^2d)$ complexity for distance computing and an $O(n^2 \log_2 n)$ complexity for sorting, whereas the ESH methods are by far more efficient with a complexity of $O(2ndkN + ndm)$. Furthermore, for precision@r=2, compared with DGH, DSH, and LGHSR, the results of the ESH methods do not always show the best performance. In this regard, first, we should note that all DGH, DSH, and LGHSR have been formulated based on two decision variables (one discrete and one continuous), whereas the ESH is a function of one decision variable. This means that the runtime of the ESH methods are significantly lower (see Fig. 3.2) compared with these methods. Considering this lower complexity, providing a competitive performance

compared with the state-of-the-art approaches imply that the ESH algorithms propose a tradeoff between lower complexity and simultaneously achieving a good performance. Second, for precision@r=2, it is true that ESH values are not always the state-of-the-art, but we should note that, in precision@r=2, r = 2 is an empirical setting for reducing the number of retrieved images, which leads to the different performances achieved by the different methods. Finally, the precision-recall graphs in the second row of Fig. 3.1 shows that the ESH algorithms are competitive compared with the state-of-the-art approaches.

Table 3.4: Comparison of retrieval performance based on mAP, precision@1000, and precision@r=2 on NCT-CRC-HE-100K dataset represented by EfficientNet features. The best performance values are highlighted in boldface.

	mAP %				precision % @1000				precision@r=2		
Method	16Bits	32Bits	64Bits	128Bits	16Bits	32Bits	64Bits	128Bits	128Bits	32Bits	64Bits
ITQ	54.72	55.78	57.50	58.39	68.41	71.64	74.63	76.14	67.90	72.54	35.66
DGH	49.85	47.24	51.61	59.61	74.08	77.14	77.05	77.58	68.75	79.02	79.15
LGHSR	55.28	47.74	38.65	32.37	75.66	78.21	77.13	74.67	73.80	79.82	67.76
KNNH	58.02	61.47	64.28	65.79	70.27	74.91	78.16	80.30	68.37	69.56	42.73
DSH	58.75	45.63	37.27	32.52	**76.70**	78.71	77.73	75.30	**75.15**	79.56	61.49
SCQ	64.90	67.25	**67.75**	66.57	76.03	**80.07**	80.01	80.02	69.64	**80.51**	72.65
ESH1	**66.32**	66.77	67.14	66.26	74.82	77.36	79.84	80.32	67.14	76.92	80.02
ESH2	63.54	**67.30**	**67.75**	**67.04**	71.85	78.12	**80.16**	**80.46**	63.04	77.17	**80.41**

3.5.6 Results on NCT-CRC-HE-100K

For the NCT-CRC-HE-100K dataset, EfficientNet [111] pre-trained on ImageNet was used for feature extraction. The training set consists of 70,000 randomly sampled images, and the test set includes the remaining 30,000 images. Table 3.4 shows that ESH1 and ESH2 achieved the best performance in terms of mAP for 16, 32, 64, and 128 bits. For precision@1000, DSH achieved the best result, and ESH2 achieved a 2% lower precision. However, as the number of bits increases, the performance of the ESH algorithms increases with a stronger trend such that for 32-bit settings ESH2 and DSH perform equally, and ESH algorithms outperform DSH for 64 and 128 bits with a margin of 3% and 5%, respectively. Clearly, the ESH algorithms attain the best precision at @r=2 under the 64-bit setting. Although for 16-and 32-bit settings, DSH and LGHSR achieved the best performance, the ESH algorithms were still competitive. The third row in Fig. 3.1, illustrates how the ESH

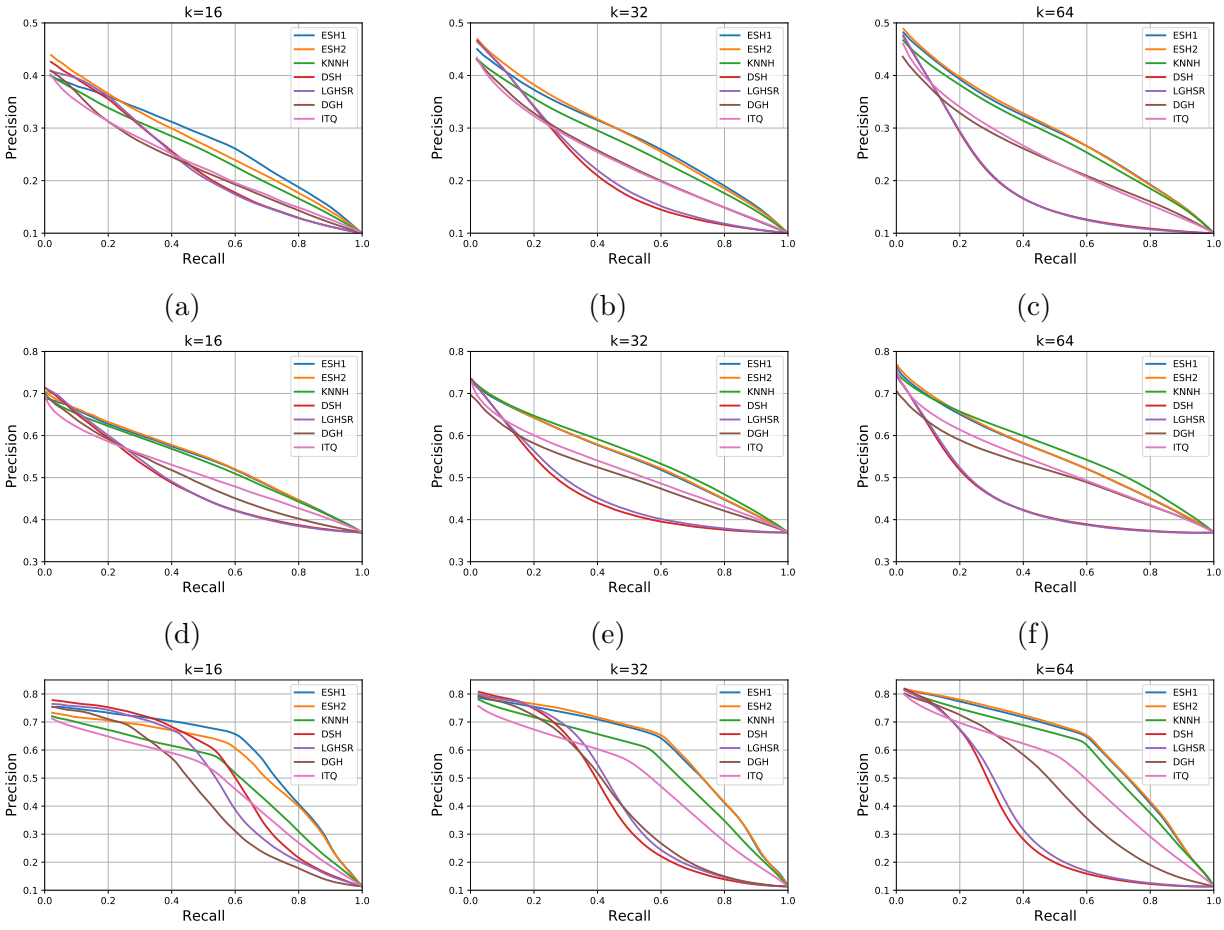


Figure 3.1: Precision-recall graphs for CIFAR-10 (first row), NUS-WIDE (second row), and NCT-CRC-HE-100K (third row) for different numbers of bits ($k = 16, 32, 64$). Clearly ESH1 and ESH2 achieve competitive performance against state-of-the-art unsupervised hashing techniques.

algorithms perform compared with recent methods in terms of precision-recall graphs. In almost all cases, the ESH algorithms achieved a better performance in comparison with the other methods.

3.5.7 Effect of Regularization

To determine how incorporating the regularization term in Eq. 3.7 improves the quality of the binary codes, we ran experiments with $\alpha = 0$ on NCT-CRC-HE-100K. Table 3.5 shows that the setting of $\alpha = 0$ significantly degrades the performance, confirming the effectiveness of the proposed regularization.

Table 3.5: Effect of the regularization term on retrieval performance in terms of mAP for NCT-CRC-HE-100K detaset.

	NCT-CRC-HE-100K			
Method, Regularization	16 Bit	32 Bit	64 Bit	128 Bit
ESH1, $\alpha = 0$	40.65	35.96	31.09	26.80
ESH1, α=automatic	**66.32**	**66.77**	**67.14**	**66.26**

3.5.8 Time Complexity and Runtime Comparison

In this section, we provide a complexity analysis of the ESH1 algorithm and compare it with recent representative graph hashing methods. The complexity required for calculating matrix $\mathbf{S}$, which only needs to be calculated once, is $O(ndm)$. Note that the complexity of k-means algorithm for selecting the centers is excluded as this is a common step in all graph hashing algorithms which employ the low-rank approximation method proposed in AGH paper for affinity matrix construction. For the learning rule applied in the projected gradient method, the complexity is $O(2ndkN)$, and as a result, the overall complexity is $O(2ndkN + ndm)$. Table 3.6 presents the complexity of the proposed ESH algorithms and recent representative graph hashing methods without including the complexity of calculating the affinity matrix. Clearly, the RDSH algorithm has the highest time complexity $O(n^3)$, which is due to solving a standard Sylvester equation to derive the spectral solution. In contrast, the AGH algorithm has the lowest complexity because it uses an eigen decomposition to obtain a spectral solution. The other algorithms listed in Table 3.6 are a function of at least two decision variables, which increases the complexity. For example,

the term $nkNlog_2n$ in LGHSR and DSH represents the complexity of updating additional variables non-existent in the ESH algorithms. On the other hand, ESH has lower complexity as the optimization is a function of one variable, and there is no inner loop, for example, terms such as N_GN in DSH or N_BN in DGH iterations.

To further validate the training time efficiency of the proposed ESH algorithms, we conduct a runtime comparison among graph hashing methods on NUS-WIDE dataset for 128 and 256-bit settings. As discussed earlier, due to the closed-form solution for AGH, this algorithm has the lowest time complexity. By contrast, the time complexity of RDSH is $O(n^3)$, which is significantly higher than that of the other algorithms. As a result, we compared the runtime of the ESH methods with DSH, DGH, and LGHSR. Fig. 3.2 shows the runtime of ESH1, ESH2, DSH, DGH, and LGHSR. Apparently, ESH2 and ESH1 have the lowest runtimes compared with the other graph hashing methods. In addition, based on this figure, the runtime of DGH and LGHSR increases quadratically with the number of bits i.e., k. This can also be seen in Table 3.6 where the k^2 term exists in the time complexity of DGH and LGHSR.

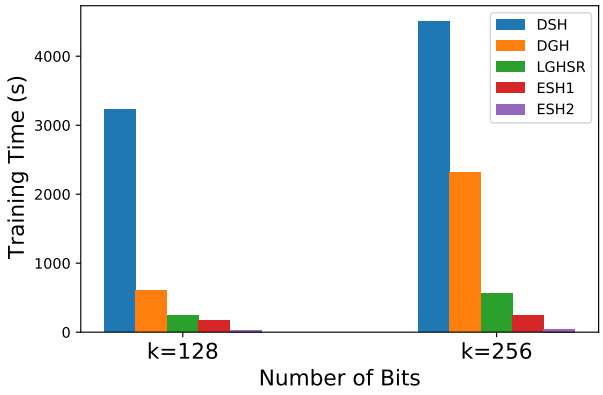


Figure 3.2: Runtime comparison of ESH1 and ESH2 against the graph hashing algorithms DSH, DGH, and LGHSR on NUS-WIDE dataset for 128- and 256-bit settings. Clearly ESH1 and ESH2 have lower training time compared with other graph hashing methods DSH, DGH, and LGHSR.

3.5.9 ESH1 versus ESH2

As indicated in the results section, ESH2 (manifold optimization) achieves a better performance in most cases compared with ESH1 (projected gradient). This is because, in ESH2,

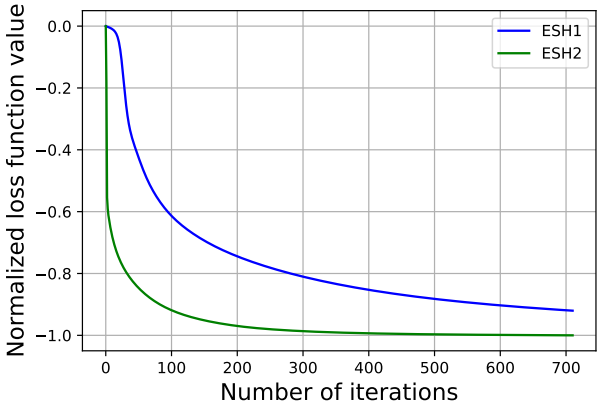


Figure 3.3: Convergence of ESH1 and 2 on NUS-WIDE dataset for 128-bit setting. According to the graph, ESH2 converges faster than ESH1.

the matrix $\mathbf{W}$ is updated on the Stiefel manifold. In other words, in ESH2, while $\mathbf{W}$ is updated, the orthogonality constraint is preserved. However, in ESH1, we update $\mathbf{W}$ using a gradient descent without considering the orthogonality constraint, and then find the closest orthogonal matrix to the updated $\mathbf{W}$. ESH2 and ESH1 can be compared in terms of the run time. As shown in Fig. 3.2, ESH2 has a lower training time compared with ESH1, which is due to the faster convergence of the manifold optimization. The graph in Fig. 3.3 compares the convergence of ESH1 and ESH2 for the NUS-WIDE dataset. Clearly, ESH2 has a faster convergence rate. Note that although ESH2 performs better in terms of both the quality of the binary codes and the runtime, it may face a memory bottleneck during training owing to the matrix inversion step (see Eq. 3.14). However, in our case, this is not a problem because the inversion is conducted on a $d \times d$ matrix instead of an $n \times n$ matrix where $d \ll n$. This is due to the fact that we transformed the problem from $n \times k$ parameters to a problem with $d \times k$ decision variables.

3.5.10 Comparison With Deep Unsupervised Hashing

Although deep learning has mainly been applied to supervised hashing problems, recent attempts have been made to develop deep unsupervised hashing methods. Some examples include DH [25], UHBDNN [20], DeepBit [73], and SADH [105]. Table 3.7 compares the performance of our proposed method compared with these algorithms. Clearly, based on Table 3.7, ESH 1 and ESH 2 outperform UHBDNN and DeepBit and provide competitive results compared with SADH. In this Table, "R+" implies that raw images are fed to

Table 3.6: Comparison of time complexities where n is number of data points, d, dimensionality of data, k, number of bits, m number of selected anchors, s the affinity matrix sparsity parameter for selecting a subset of s anchors, N number of iterations, and N_B, N_G are number of inner loop iterations.

Method	Time complexity
AGH	$O(m^2n + (s+1)kn)$
DGH	$O(nmkN_BN + k^2nN)$
RDSH	$O(n^3)$
LGHSR	$O(m^2n + (s+1)kn + 2nk^2N + nkN\log_2 n)$
DSH	$O(nmkN_GN + nkN\log_2 n)$
ESH	$O(2ndkN + ndm)$

the network, suggesting that these methods are end-to-end. By contrast, "V+" indicates that the vector representations are fed to the network. Considering the simplicity of our proposed method in comparison with deep learning methods, and that feature learning has not been included, the obtained results are promising

Table 3.7: ESH compared with deep unsupervised hashing algorithms for NUS-WIDE dataset. The R+ and V+ means the respective algorithm works on raw images and vector data (images after feature extraction) respectively.

	mAP %			precision % @5000		
Method	16 Bits	32 Bits	64 Bits	16 Bits	32 Bits	64 Bits
V+UHBDNN	54.26	51.72	54.74	70.18	69.60	72.74
R+DeepBit	39.22	40.32	42.06	45.54	51.34	57.72
R+SADH	**60.14**	**57.99**	56.33	**71.45**	**73.88**	75.04
ESH1	56.32	56.89	57.47	67.58	69.47	71.46
ESH2	56.54	57.16	**57.71**	69.74	72.49	**75.62**

3.6 Summary and Conclusions on ESH

In this Chapter, we proposed a novel formulation for spectral hashing that achieves a highly competitive performance compared with most recent methods, and at the same time achieves a low complexity. The proposed projected gradient method is highly efficient for three reasons. First, the formulation for ESH transforms the decision variable with a

dimensionality of $n \times k$ into $d \times k$, where $d \ll n$. Second, the affinity matrix, which is $n \times n$ in the spectral formulation, was removed, and instead, a $d \times d$ matrix $\mathbf{S}$ plays a similar role. Finally, and more importantly, unlike other graph hashing schemes, the proposed formulation achieves high-quality binary codes without adding any additional decision variables to the problem. We applied two different optimization techniques, that is, a projected gradient and manifold optimization, to obtain a solution. Using extensive experiments on four public datasets, we showed that the proposed method outperforms or achieves highly competitive results compared with recent methods and offers a low complexity at the same time. For future work, we plan to update the affinity matrix along with the proposed loss function, which needs the use of an end-to-end training framework where feature learning is achieved through training. To conduct feature learning, we may need to employ a reconstruction loss with the proposed loss function in Eq. 3.7. We believe that such a scheme can significantly improve the performance of the proposed method. Furthermore, another interesting path for future studies is to apply this more efficient non-alternating hashing scheme instead of the more complex alternating algorithm employed by [47] for the network quantization problem.

Chapter 4

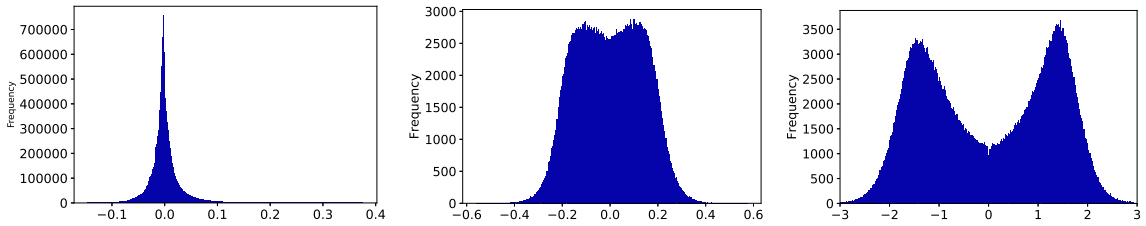
Beyond Neighbourhood-Preserving Transformations for Quantization-Based Unsupervised Hashing

4.1 Prologue

The content of this chapter is based on the following paper published during the Ph.D. research:

1. **S. Hemati**, et al. *Beyond neighbourhood-preserving transformations for quantization-based unsupervised hashing* Pattern Recognition Letters Volume 153, January 2022, Pages 44-50

In the previous Chapter , we proposed the efficient spectral hashing (ESH) for binary representation learning where the main idea was to preserve the neighbourhood of data in binary as much as possible. Although ESH achieves high quality binary codes while enjoys less memory usage and training time compares with other spectral hashing based methods, it still leads to difficult constrained optimization problem. Another well studied approach for binary representation learning is quantization based approach where generally formulations lead to more straightforward optimization problems. In this Chapter, we



(a) No Transformation (b) ITQ (c) SNRQ

Figure 4.1: Histogram of transformed MNIST dataset using a) Raw (no transformation) data, b) ITQ (rigid transformation) and c) SNRQ (rigid and non-rigid transformations). As can be seen, SNRQ is more effective that ITQ in preparing the data to be quantized at zero.

propose a novel form of quantization called non-rigid quantization which leads to high-quality binary representations compared with latest quantization based hashing methods.

Employing a neighbourhood preserving transformation is the common practice in quantization based hashing methods. For example, the ITQ [33], Angular Quantization (AQ) [32], Isotropic Hashing (IsoHash) [65], Efficient Spectral Hashing (ESH) [42], Optimal Projection Hashing (OPH) [12], and Simultaneous Compression and Quantization (SCQ) [44] employ neighbourhood preserving transformations in different forms to reduce the quantization loss. The main issue with binarization is destroying neighbourhood structure of data. Having this in mind, rotation matrices seem to be a good choice for reducing quantization loss as they preserve neighbourhood. Although they are effective, we argue that a single rotation is not powerful enough to minimize the quantization error. Instead, we propose to employ transformations beyond rotation that even corrupt neighbourhood structure of data in favour pushing for quantization. As we will see, such transformation coupled with a rotation leads to high quality binary codes that outperforms state-of-art linear unsupervised hashing methods. Although the proposed idea can be applied to many hashing algorithms that employ a single rotation for reducing quantization error, here we choose to develop this idea on top of ITQ which is very fast and still among competitive hashing methods. In ITQ, the data is projected to lower dimensionality and then the optimal rotation is found such that it minimizes the quantization error in two separate steps. In our method, we employ a transformation beyond rotation and also resolve the discontinuity in projecting data to lower dimensionality and reduce quantization error. We formulate and optimize our objective function such that in our algorithm:

1. The data is projected and quantized simultaneously. This means unlike many con-

ventional linear unsupervised hashing methods, the projection matrix is contributed towards reducing quantization error.

2. We relax the orthogonality on projection matrix in favor of reducing quantization error even if it corrupts the original neighbourhood. We experiment different matrix norms for achieving this non-rigid (non-similarity preserving) transformation.

3. An efficient sequential update scheme is proposed for learning the projection matrix.

The left plot in Fig. 4.1 shows the histogram of the raw data, while the middle and right plots show transformed data for a neighbourhood preserving transformation (ITQ) and transformed by a non-neighbourhood preserving operations using our proposed method Sequential Sequential Non-rigid Quantization (SNRQ). Apparently, SNRQ pushes data points more distinctively toward +1 and -1 modes compared to ITQ.

Using extensive quantitative experiments on five public detest and also qualitative results, we show that although our transformation corrupts the neighbourhood of data, the final binary codes obtained using our method, preserve more neighbourhood compared to many other linear hashing methods.

4.2 Method

4.2.1 Formulation

In this section, we formulate an optimization problem that follows two objectives. Firstly, it is desirable to jointly learn a projection matrix and minimize the quantization loss of projected data. Secondly, the orthogonality constraint on the projection matrix is relaxed such that we can employ a non-rigid transformation for reducing quantization error. These two objectives together enable non-orthogonal projection matrix to contribute to minimizing quantization loss in a way different from the orthogonal rotation matrix. The first objective is achieved by joining the two well-known ITQ steps, namely applying PCA to the data and minimizing quantization loss. Let $\mathbf{X} \in \mathbb{R}^{n \times D}$ represents zero-centered data in which n is the number of training data points and D is the data dimensionality. It is well understood that the projection matrix $\mathbf{W} \in \mathbb{R}^{D \times K}$ or K principal components of data can be obtained by maximizing the objective function

$$\arg\max_{\mathbf{W}} \quad Tr\{\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\} \quad \text{s.t.} \quad \mathbf{W}^T\mathbf{W} = \mathbf{I}_K, \tag{4.1}$$

where $\mathbf{I}_K$ is the $K \times K$ identity matrix and $Tr\{\}$ is the trace of a matrix. Here we define $\mathbf{V}$ as $\mathbf{V} = \mathbf{XW}$ with $\mathbf{V} \in \mathbb{R}^{n \times K}$. We would like to minimize the quantization loss of transformed data. To do this, we find an orthogonality relaxed matrix $\mathbf{W}$ (the relaxed orthogonality constraint on $\mathbf{W}$ will be formulated later in Eq. 4.3) and an $K \times K$ orthogonal rotation matrix $\mathbf{R}$ such that the quantization loss of thresholding the transformed data $\mathbf{VR} = \mathbf{XWR}$ at zero is minimized. This can be formulated as minimizing the following:

$$Q(\mathbf{B}, \mathbf{R}, \mathbf{W}) = \|\mathbf{XWR} - \mathbf{B}\|_F^2 \quad \text{s.t.} \quad \mathbf{R}^T\mathbf{R} = \mathbf{I}_K, \tag{4.2}$$

where $\mathbf{B} \in \{-1, 1\}^{n \times k}$ is the corresponding binary representation of $\mathbf{X}$. In order to jointly learn the projection matrix $\mathbf{W}$ and minimize the quantization loss using both non-rigid $\mathbf{W}$ and rigid $\mathbf{R}$, we relax the orthogonality constrain on the projection in PCA formulation and also regularize this by a quantization term which leads to **our proposed objective function**:

$$\underset{\mathbf{W},\mathbf{R},\mathbf{B}}{\arg\max} \quad J(\mathbf{W}, \mathbf{R}, \mathbf{B}) = Tr\{\mathbf{W}^T\mathbf{C_x}\mathbf{W}\} - \alpha\|\mathbf{XWR} - \mathbf{B}\|_F^2 - \beta\|\mathbf{W}^T\mathbf{W} - \mathbf{I}_K\|_F^2, \tag{4.3}$$

subject to $\mathbf{R}^T\mathbf{R} = \mathbf{I}_K$ and $\mathbf{B} \in \{-1, 1\}^{n \times k}$ where α and β are quantization and rigidness regularization parameters, and finally $\mathbf{C_x} = \mathbf{X}^T\mathbf{X}$. Note that in Eq. 4.3 both $\mathbf{W}$ and $\mathbf{R}$ are contributing to minimizing quantization loss, $\|\mathbf{XWR} - \mathbf{B}\|_F^2$, but in different ways as orthogonality constraint on $\mathbf{W}$ is smoothed whereas $\mathbf{R}$ is a pure orthogonal rotation matrix. This is one of the main differences between the proposed method and other existing methods e.g., in contrast to ITQ, IsoHash, AQ, and SCQ.

4.2.2 Optimization

In order to jointly minimize the quantization loss and learn the projection matrix, we need to find $\mathbf{W}$, $\mathbf{R}$, and $\mathbf{B}$ in a way to maximize the objective function in Eq. 4.3. Due to binary constraints on $\mathbf{B}$, the optimization is intractable. Hence, inspired by technique used in ITQ paper, we use a coordinate descent approach to optimize the objective function over $\mathbf{W}$, $\mathbf{R}$, and $\mathbf{B}$. To this end, in each step we consider one of the $\mathbf{W}$, $\mathbf{R}$, and $\mathbf{B}$ variable and the two other matrices are assumed to be constant.

Fix R and B, and update W: In order to optimize the objective in Eq. 4.3 with respect to $\mathbf{W}$, we easily calculate the derivative with respect to $\mathbf{W}$ as follow:

$$\frac{\partial J(\mathbf{W},\mathbf{R},\mathbf{B})}{\partial \mathbf{W}} = 2\mathbf{C_x}\mathbf{W} - \alpha\left(2\mathbf{X}^T(\mathbf{XWR}-\mathbf{B})\mathbf{R}^T\right) - \beta\left(4\mathbf{W}(\mathbf{W}^T\mathbf{W}-\mathbf{I}_K)\right). \tag{4.4}$$

Having the gradient w.r.t $\mathbf{W}$, we found that the L-BFGS-B optimizer [135] obtain a good solution for $\mathbf{W}$. However, as each variable should be updated multiple times (while other are fixed) we observed that optimizing for $\mathbf{W}$ directly is a time-consuming procedure. To mitigate this challenge, in the following we propose a sequential approach for updating $\mathbf{W}$. As we will show in experiments, sequential approach achieves comparable performance while reducing training time significantly. In sequential scheme, each time one column of $\mathbf{W}$ is updated while the rest of columns are considered fixed. First, let's expand the second term (quantization loss) in Eq. 4.3:

$$\begin{aligned} &\|\mathbf{XWR}-\mathbf{B}\|_F^2 = \|\mathbf{XWR}\|_F^2 - 2\,Tr\{\mathbf{XWRB}^T\} + \|\mathbf{B}\|_F^2 \\ =&Tr\{\mathbf{W}^T\mathbf{C_x}\mathbf{W}\} - 2\,Tr\{\mathbf{W}(\mathbf{RB}^T\mathbf{X})\} + const. \end{aligned} \tag{4.5}$$

Now let's expand the relaxed orthogonality constraint (the third term in Eq. 4.3):

$$\begin{aligned} &\|\mathbf{W}^T\mathbf{W}-\mathbf{I}_K\|_F^2 = \|\mathbf{W}^T\mathbf{W}\|_F^2 - 2\,Tr\{\mathbf{W}^T\mathbf{W}\mathbf{I}_K\} + const \\ =&Tr\{(\mathbf{WW}^T)(\mathbf{WW}^T)\} - 2\,Tr\{\mathbf{WW}^T\} + const. \end{aligned} \tag{4.6}$$

Using Eqs. 4.5 and 4.6 for the second and third terms in Eq. 4.3, respectively, and defining $\mathbf{C_y} = \mathbf{RB}^T\mathbf{X}$ provides us with the objective function for the case that $\mathbf{W}$ is variable:

$$\begin{aligned} \arg\max_{\mathbf{W}} \quad & (1-\alpha)\,Tr\{\mathbf{W}^T\mathbf{C_x}\mathbf{W}\} + 2\alpha\,Tr\{\mathbf{WC_y}\} \\ & - \beta\,Tr\{(\mathbf{WW}^T)(\mathbf{WW}^T)\} + 2\beta\,Tr\{\mathbf{WW}^T\}. \end{aligned} \tag{4.7}$$

Let's start formulating the problem such that one column of $\mathbf{W}$ is considered variable while the remaining columns are constant. To this end, the k-th column of $\mathbf{W}$ that is considered variable is denoted by $\mathbf{z}_k$ and all other columns are shown by $\mathbf{W}'$. In this case, for the first term we receive

$$\begin{aligned} &Tr\{\mathbf{W}^T\mathbf{C_x}\mathbf{W}\} = Tr\{\mathbf{C_x}\mathbf{WW}^T\} = \\ &Tr\{\mathbf{C_x}(\mathbf{W}'\mathbf{W}'^T + \mathbf{z}_k\mathbf{z}_k^T)\} = const + \mathbf{z}_k^T\mathbf{C_x}\mathbf{z}_k. \end{aligned} \tag{4.8}$$

Note that here first we used the fact that $Tr\{\mathbf{C_x}\mathbf{z}_k\mathbf{z}_k^T\} = Tr\{\mathbf{z}_k^T\mathbf{C_x}\mathbf{z}_k\}$ and then removed $Tr\{\cdot\}$ as the $\mathbf{z}_k^T\mathbf{C_x}\mathbf{z}_k$ is scalar. Similarly, if we define the k-th row of $\mathbf{C_y}$ as $\mathbf{u}_k^T$ and the

rest of the rows as $\mathbf{C}'_{\mathbf{y}}$, then the second term can be written as

$$\begin{aligned} Tr\{\mathbf{W}\mathbf{C}_{\mathbf{y}}\} = Tr\{\mathbf{W}'\mathbf{C}'_{\mathbf{y}} + \mathbf{z}_k\mathbf{u}_k^T\} = \\ const + Tr\{\mathbf{u}_k^T\mathbf{z}_k\} = const + \mathbf{u}_k^T\mathbf{z}_k. \end{aligned} \tag{4.9}$$

For the third term in Eq. 4.7 we write

$$\begin{aligned} &Tr\{(\mathbf{W}\mathbf{W}^T)(\mathbf{W}\mathbf{W}^T)\} = \\ &Tr\{(\mathbf{W}'\mathbf{W}'^T + \mathbf{z}_k\mathbf{z}_k^T)(\mathbf{W}'\mathbf{W}'^T + \mathbf{z}_k\mathbf{z}_k^T)\} = \\ &const + 2\mathbf{z}_k^T\mathbf{W}'\mathbf{W}'^T\mathbf{z}_k + \mathbf{z}_k^T\mathbf{z}_k\mathbf{z}_k^T\mathbf{z}_k. \end{aligned} \tag{4.10}$$

As $\mathbf{z}_k^T\mathbf{W}'\mathbf{W}'^T\mathbf{z}_k$ and $\mathbf{z}_k^T\mathbf{z}_k\mathbf{z}_k^T\mathbf{z}_k$ are scalars, $Tr\{\cdot\}$ can be removed. Similarly, for the last term we obtain

$$\begin{aligned} Tr\{\mathbf{W}\mathbf{W}^T\} = Tr\{\mathbf{W}'\mathbf{W}'^T + \mathbf{z}_k\mathbf{z}_k^T\} = \\ const + Tr\{\mathbf{z}_k^T\mathbf{z}_k\} = const + \mathbf{z}_k^T\mathbf{z}_k. \end{aligned} \tag{4.11}$$

Incorporating Eqs. 4.8, 4.9, 4.10, and 4.11 into Eq. 4.7, if we set

$$\mathbf{Q} = (1-\alpha)\mathbf{C}_{\mathbf{x}} - 2\beta\mathbf{W}'\mathbf{W}'^T + 2\beta\mathbf{I}_D, \tag{4.12}$$

then by maximizing the following objective function one can obtain the $\mathbf{z}_k$, k-th column of $\mathbf{W}$:

$$\arg\max_{\mathbf{z}_k} \quad J(\mathbf{z}_k) = \mathbf{z}_k^T\mathbf{Q}\mathbf{z}_k + 2\alpha\mathbf{u}_k^T\mathbf{z}_k - \beta\mathbf{z}_k^T\mathbf{z}_k\mathbf{z}_k^T\mathbf{z}_k. \tag{4.13}$$

To update each column of $\mathbf{W}$, i.e, $\mathbf{z}_k$, where $k = 1 \dots K$, we have to find $\mathbf{z}_k$ that maximizes Eq. 4.13. The gradient of objective function (Eq. 4.13) can be calculated as

$$\frac{\partial J(\mathbf{z}_k)}{\partial \mathbf{z}_k} = 2\mathbf{Q}\mathbf{z}_k + 2\alpha\mathbf{u}_k - 4\beta\mathbf{z}_k(\mathbf{z}_k\mathbf{z}_k^T). \tag{4.14}$$

Now, we use the L-BFGS-B optimizer to obtain solutions. As we will show in experiments, this sequential learning scheme significantly reduces the training time of the proposed algorithm without sacrificing the quality of binary codes. This efficiency is in part because of simplifying the objective function and also reducing the search space by dividing the problem to K sub-problems.

Fix R and W, and update B: In this case, the optimization in Eq. 4.3 takes the following form:

$$\arg\min_{\mathbf{B}} \quad \|\mathbf{XWR} - \mathbf{B}\|_F^2 \quad \text{s.t.} \quad \mathbf{B} \in \{-1, 1\}^{n \times k}. \tag{4.15}$$

Having in mind that $\mathbf{R}$ and $\mathbf{W}$ are fixed, clearly, minimization of Eq. 4.15 with respect to $\mathbf{B}$ is equivalent the maximization of $Tr\{(\mathbf{XWR})^T\mathbf{B}\}$ where elements of $\mathbf{B}$ can be either 1 or -1. As we know from ITQ [33], the optimal $\mathbf{B}$ can be calculated as

$$\mathbf{B} = sgn(\mathbf{VR}) = sgn(\mathbf{XWR}). \tag{4.16}$$

Fix W and B, and update R: For the case that $\mathbf{R}$ is variable, maximizing Eq. 4.3 is equivalent to

$$\arg\min_{\mathbf{R}} \quad \|\mathbf{VR} - \mathbf{B}\|_F^2 \quad \text{s.t.} \quad \mathbf{R}^T\mathbf{R} = \mathbf{I}_K. \tag{4.17}$$

This is the Orthogonal Procrustes problem where a rotation matrix is found such that two point sets are aligned with each other. Here, these two point sets are the target binary code matrix $\mathbf{B}$ and projected data $\mathbf{V}$. This problem has closed form solution when $\mathbf{R}$ is a square orthogonal (rotation) matrix [101] which is

$$\mathbf{R} = \hat{\mathbf{S}}\mathbf{S}^T. \tag{4.18}$$

where $\mathbf{S}\mathbf{\Lambda}\hat{\mathbf{S}}^T$ is the SVD of the K by K matrix $\mathbf{B}^T\mathbf{V}$.

In summary, in each iteration, Eq. 4.16 and Eq. 4.18 are used to update $\mathbf{B}$ and $\mathbf{R}$, respectively.To update $\mathbf{W}$, in each iteration, K optimization problems for K columns of $\mathbf{W}$ are solved. After obtaining $\mathbf{R}$, and $\mathbf{W}$, the binary representation of the data $\mathbf{X}$ can be obtained $\mathbf{B} = sgn(\mathbf{VR}) = sgn(\mathbf{XWR})$. When the direct optimization is used for updating $\mathbf{W}$ we denote our method NRQ and for the sequential case this denoted by SNRQ. Algorithm 1 summarizes the proposed scheme to update all three matrices method.

4.2.3 Implementation Note

We use K eigenvectors of $\mathbf{C}_x$ corresponding to K largest eigenvalues as initialization of $\mathbf{W}$. For $\mathbf{R}$, we use the rotation matrix obtained by ITQ as initialization of $\mathbf{R}$. Although we will show the algorithm is robust to different values for α and β, to avoid tuning regularization parameters, we set $\alpha = 3$ and $\beta = 0.01$ for all datasets and experiments. In order to choose α, the only consideration is to keep it larger than 1. This can be understood based

Algorithm 1 The Proposed SNRQ Algorithm

Input: Data matrix $\mathbf{X}$, number of iterations N
regularization parameters α and β
Output: Projection matrix $\mathbf{W}$, rotation matrix $\mathbf{R}$
Initialization: Initialize $\mathbf{W}$ and $\mathbf{R}$

for iteration$= 1, 2, \ldots, N$ **do**
  $\mathbf{V} \leftarrow \mathbf{XW}$
  $\mathbf{B} \leftarrow sgn(\mathbf{VR})$
  $\mathbf{S}\Lambda\hat{\mathbf{S}}^T \leftarrow \text{SVD}(\mathbf{B}^T\mathbf{V})$
  $\mathbf{R} \leftarrow \hat{\mathbf{S}}\mathbf{S}^T$
  $\mathbf{C_y} \leftarrow \mathbf{RB}^T\mathbf{X}$
  for $k = 1, 2, \ldots, K$ **do**
    $\mathbf{u}_k^T \leftarrow \mathbf{C_y}[k, :]$
    $\mathbf{Q} \leftarrow (1-\alpha)\mathbf{C_x} - 2\beta\mathbf{W'}\mathbf{W'}^T + 2\beta\mathbf{I}_K$
    Solve Eq. 4.13 for each column of $\mathbf{W}$, i.e., $\mathbf{z}_k$
    $J(\mathbf{z}_k) = \mathbf{z}_k^T\mathbf{Q}\mathbf{z}_k + 2\alpha\mathbf{u}_k^T\mathbf{z}_k - \beta\mathbf{z}_k^T\mathbf{z}_k\mathbf{z}_k^T\mathbf{z}_k$
    $\frac{\partial J(\mathbf{z}_k)}{\partial \mathbf{z}_k} = 2\mathbf{Q}\mathbf{z}_k + 2\alpha\mathbf{u}_k - 4\beta\mathbf{z}_k(\mathbf{z}_k\mathbf{z}_k^T)$
    Update the k-th column of $\mathbf{W}$, i.e., $\mathbf{z}_k$
    $\mathbf{W}[:, k] \leftarrow \mathbf{z}_k$
  end for
end for

on Eq. 4.12 which shows there is a trade-off between reducing quantization error and maximizing variance across projections. This is clear that the quantization error is our priority. The algorithm is fairly robust to β, and based on our experiments, any number between 0.01 to 50 easily does the job. For the number of iterations, experiments on a variety of detests show that 70 iterations are generally sufficient, and after that, the loss function does not change significantly. As a results, We set the number of iterations N to 70. An implementation of the proposed method is provided in this GitHub repository.

4.3 Experiments and Results

4.3.1 Datasets and Evaluation Protocol

The performance of the proposed SNRQ algorithm is evaluated on five standard benchmark datasets, MNIST [71], CIFAR-10 [67], an unbalanced dataset LabelMe-12-50K [114], a medical image dataset NCT-CRC-HE-100K [82], and a multi-label NUS-WIDE [13]. These datasets provide a total of **489,000 images** for learning and testing.

- The **MNIST Dataset** contains 70,000 gray-scale images all of size 28×28 pixels. There are 10 classes in this dataset for handwritten digits.
- The **CIFAR-10 Dataset** is a 10-class dataset consisting of 60,000 color images of size 32×32 pixels.
- The **LabelMe-12-50K Dataset** is a 12-class dataset containing 50,000 images of size 256×256 pixels. This dataset is highly imbalanced such that five classes constitute 91% of all images while there is one class that only contains 0.6% of the samples. The images of this dataset have multiple label values between zero and one. In our experiments, same as previous works that employed this dataset [37] for evaluating hashing algorithms, we choose the class of the largest label value as the image label.
- The **NCT-CRC-HE-100K Dataset** is 9-class histopathology dataset containing 100,000 non-overlapping image patches from hematoxylin & eosin stained (H&E) images of human colorectal cancer and normal tissue. All images are 224×224 pixels and color-normalized.
- The **NUS-WIDE Dataset** is a multilabel dataset that contains 269,000 images collected from Flickr. This database contains 81 ground-truth concepts.

4.3.2 Hash Code Evaluation

To evaluate the performance of the SNRQ, we use standard measures for image retrieval quality assessment. These measures include mean Average Precision (mAP), and precision at M samples (e.g., precision@1000). Briefly, mAP measures the overall performance of the retrieval over all classes, whereas precision@M calculates the proportion of true positive over top M retrieved samples.

Table 4.1: Comparison of retrieval performance based on mAP and precision@1000 on MNIST dataset represented by 512-D GIST descriptor. The best performance is highlighted in boldface. KMH: K-means Hashing [36], SpH: spherical hashing [43].

	mAP %			precision % @1000		
Method	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
SH	32.59	33.23	30.65	-	-	-
SpH	31.27	36.80	41.40	-	-	-
KMH	31.96	37.39	41.11	-	-	-
BA	48.48	51.72	52.73	-	-	- -
ITQ	46.37	50.59	53.69	69.67	75.13	80.45
KNNH	53.07	61.11	65.55	73.99	83.32	87.05
SCQ	62.39	74.49	72.23	79.26	88.46	**88.90**
NRQ (Ours)	**71.46**	69.44	**72.79**	**84.59**	85.49	87.30
SNRQ (Ours)	64.70	**76.98**	**73.48**	80.00	**88.53**	88.27

Table 4.2: Comparison of retrieval performance, based on mAP, for 16, 32 and 64 bits for CIFAR-10 and macro mAP (average over classes) for LabelMe-12-50k datasets both represented by 4096-D VGG-FC7 descriptors.

	CIFAR-10			LabelMe-12-50k		
Method	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
SH	18.31	16.54	15.78	12.60	12.59	12.24
SpH	18.82	20.93	23.40	13.59	15.10	17.03
KMH	18.68	20.82	22.87	13.36	15.47	16.58
BA	25.38	26.16	27.99	16.96	18.42	20.80
ITQ	26.82	27.38	28.73	18.06	19.40	20.73
DGH	27.73	27.44	28.01	21.45	22.74	25.41
LGHSR	27.83	25.87	22.12	21.10	23.49	23.98
DSH	27.72	25.36	22.12	24.70	23.78	24.35
SCQ	27.52	27.42	30.34	22.89	24.95	26.50
KNNH	29.06	30.82	32.60	20.13	23.79	26.22
NRQ (Ours)	**31.42**	30.87	33.05	24.27	**27.40**	**28.65**
SNRQ (Ours)	30.10	**31.82**	**33.10**	**25.14**	26.80	28.50

4.3.3 Results on MNIST dataset

Following the setting of [37] for MNIST data, each image is presented by a GIST 512-D descriptor, 10% of each class is considered for query set and the remaining data is used as training set. Table 4.1 shows the results for MNIST in terms of mAP, and precision@1000. Clearly, the NRQ and SNRQ methods provide an improvement over other methods in

all cases with only one exception for precision@1000 where the SCQ slightly outperforms SNRQ with less than 0.5% in 64 bit.

4.3.4 Results on CIFAR-10 Dataset

For this dataset, following the common setting [37, 133], we used deep 4096-D features extracted from VGG network [106] and sample 10% of each class as query set and the remaining instances as training set. The left half of Table 4.2 represents the results for this experiment setting. As it can be seen, NRQ and SNRQ outperform state-of-art namely KNNH [37] and SCQ [44].

4.3.5 Results on LabelMe-12-50k Dataset

As it was pointed out, this dataset is highly imbalanced. To ascertain a fair comparison, we calculate mAP values that are macro averages over all classes. Following the common setting [37] we sample 10% of each class to construct the query set, and the remaining data points as training set. Besides, VGG network has been used to extract feature vectors. As Table 4.2 shows, NRQ and SNRQ are performing better and outperform state-of-art methods with a large gap.

4.3.6 Results on NCT-CRC-HE-100K Dataset

For this dataset, we used EfficientNet [111] pre-trained on ImageNet to extract 1280-D feature vectors from images. We randomly sampled 70,000 images (out of 100K) for training set and the rest of images (30,000) for test. For efficiency, we reduced the feature vector dimensionality to 512 values by the PCA. Table 4.3 shows that NRQ and SNRQ outperform competitive methods based on mAP and precision@1000 with a significant gap.

4.3.7 Results on NUS-WIDE Dataset

For NUS-WIDE experiments, we used the common setup in many hashing papers [78, 105]. In this setting, images are selected which their labels are among the 21 most frequent labels. This leads to 195,834 images. We randomly sample 2,100 images (100 images from each class) from 195,834 images for the test set and the rest of the images are used for training the hash function and populating the hash table. We used VGG-F network [9] to extract

Table 4.3: Comparison of retrieval performance based on mAP, and precision@1000 on NCT-CRC-HE-100K dataset represented by features extracted by EfficientNet.

	mAP %			precision % @1000		
Method	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
ITQ	55.8	57.8	59.5	68.8	72.6	75.4
DGH	49.85	47.24	51.61	74.08	77.14	77.05
LGHSR	55.28	47.74	38.65	75.66	78.21	77.13
DSH	58.75	45.63	37.27	76.70	78.71	77.73
KNNH	58.02	61.47	64.28	70.27	74.91	78.16
SCQ	64.90	67.25	67.75	76.03	80.07	81.01
NRQ (Ours)	**75.32**	73.08	77.25	**83.94**	83.32	86.91
SNRQ (Ours)	68.51	**75.45**	**78.75**	80.19	**84.58**	**87.07**

Table 4.4: Comparison of retrieval performance on NUS-WIDE dataset represented by VGG-F deep features.

	mAP %			precision % @5000		
Method	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
SH	44.74	42.60	42.36	59.15	54.98	54.18
AGH	49.80	47.14	44.72	70.43	70.29	69.29
DGH	54.03	52.74	49.64	**71.15**	71.91	70.66
LGHSR	50.79	47.72	45.45	66.79	68.59	67.60
ITQ	53.39	54.61	55.75	65.80	68.69	70.59
DSH	49.87	47.07	45.67	68.05	68.17	67.83
KNNH	55.12	57.53	58.61	66.77	69.83	71.07
SCQ	58.34	56.21	56.10	68.45	70.64	70.81
NRQ (Ours)	60.83	62.08	**63.34**	69.91	72.07	**73.82**
SNRQ (Ours)	**61.78**	**62.74**	62.60	70.83	**72.33**	73.37

features from images. Results for this experiment are reported in Table 4.4. The results for other algorithms mainly are directly reported from literature. [105].

4.3.8 Comparison Between NRQ and SNRQ

Here we compare NRQ and SNRQ both in terms of run-time and performance. If we directly update the projection matrix, the training becomes significantly time consuming. To address this, we proposed sequential update scheme which achieves the same performance while significantly reduces the time complexity. The training times of NRQ and SNRQ for CIFAR-10 dataset are shown in Fig. 4.2. Although based on Fig. 4.2, SNRQ

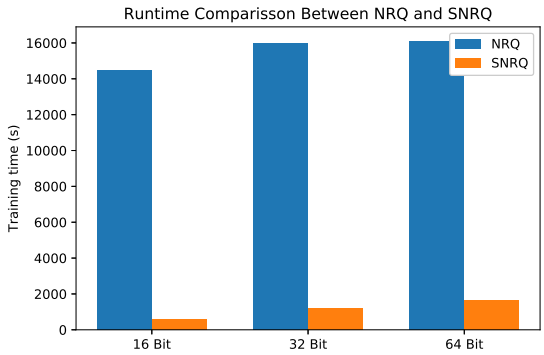
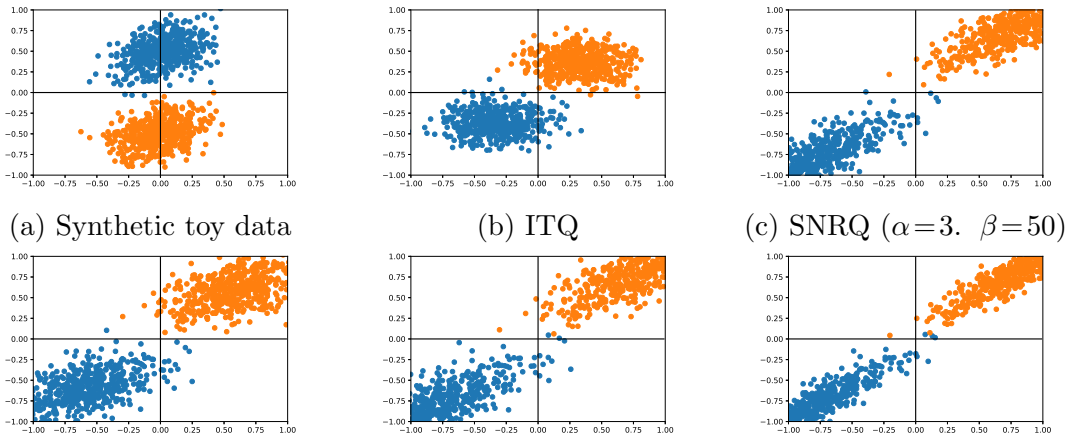


Figure 4.2: Training time for NRQ and SNRQ applied to the CIFAR-10 dataset for different number of bits. According to the graph, the training time for SNRQ is significantly less than NRQ.



(a) Synthetic toy data (b) ITQ (c) SNRQ ($\alpha=3$. $\beta=50$)

(d) SNRQ ($\alpha=3$, $\beta=200$) (e) SNRQ ($\alpha=2$, $\beta=50$) (f) SNRQ ($\alpha=4$, $\beta=50$)

Figure 4.3: Visualization of the ITQ and SNRQ performance in projecting the data to a space with less quantization loss on a 2D toy dataset: (a) The reference 2-D data, (b) data rotated & projected by ITQ, (c-f) Data rotated & projected by SNRQ using different values of β & α. In SNRQ, the data points are pushed more towards Hamming vertices at cost of corrupting the neighbour structure of data.

is significantly faster compared with NRQ, as can be seen from Tables 4.1, 4.2, 4.3, and 4.4 their performance is similar. The time difference for training may be quite significant for petabyte archives with gigapixel files like satellite imaging and digital pathology where images are commonly quite large, $>$ 50,000 by 50,000 pixels. For the latter each patient often comes with many images.

4.3.9 SNRQ vs. ITQ

To validate how employing a non-rigid transformation acts compared with rigid one, we generated a synthetic toy dataset. Fig. 4.3 shows the transformed data by ITQ and SNRQ on this toy dataset under for values of α and β. Fig. 4.3 part (a) represents the toy data. Fig. 4.3 part (b) shows the same data after transformation by ITQ. Figs. 6.3 part (c), (d), (e), and (f) show the transformed data using SNRQ for different values of α and β. This graph reveals role of these regularization parameters in SNRQ algorithm which is controlling the trade-off between preserving neighbourhood structure of data and pushing data points. They can cause unsafe quantization. The smaller the β (larger the α) the unsafer the quantization. The quantization loss is increasingly reduced by corrupting neighbourhood.

4.3.10 Comparison of SNRQ With Deep Unsupervised Hashing Algorithms

Recently deep learning has been applied to unsupervised hashing including DH [25], UHBDNN [20], DeepBit [73], and SADH [105]. To compare SNRQ with Deep unsupervised hashing methods, we employ NUS-WIDE in the setting same as Table 4.4. As Table 4.5 shows, SNRQ outperforms UHBDNN and DeepBit by a considerable gap. Compared with SADH, results are highly competitive. Considering DeepBit and SADH are feeding images into a CNN and learn the features, which implies that the quality of features used in SADH, and DeepBit are expectedly better than VGG-F extracted features, it is quite impressive that SNRQ is superior or delivering on par results.

4.3.11 Ablation Study

We set $\alpha = 3$ and $\beta = 0.01$ for all experiments and datasets. Based on the following experiments, although β and α regulates the trade-off between preserving the neighbourhood

Table 4.5: SNRQ compared with deep unsupervised hashing algorithms for NUS-WIDE dataset. The terms R+ and V+ mean the respective algorithm works on raw images and vector data (images after feature extraction) respectively. The performance is measured based on based on mAP and precision@5000. R+ and V+ means raw images and VGG features are fed to the network respectively.

	mAP %			precision % @5000		
Method	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
V+UHBDN	54.26	51.72	54.74	70.18	69.60	72.74
R+DeepBit	39.22	40.32	42.06	45.54	51.34	57.72
R+SADH	60.14	57.99	56.33	**71.45**	**73.88**	**75.04**
V+SNRQ	**61.78**	**62.74**	**62.60**	70.83	72.33	73.37

structure and the quality of quantization, SNRQ is robust to changes. Results in Table 4.6 show the SNRQ performance on MNIST and LabelMe-12-50k datasets for different values $\beta = 0.01, 0.05, 0.1$, and 0.5 for the 16 bit setting. Even the worst performance after changing $\beta = 0.01$ is still state-of-art. We also tested the robustness of SNRQ to changes in α. We should not use $\alpha = 1$ as it removes the $\mathbf{Q}$ from objective. For α, first the β value is set to 0.01, then we evaluated SNRQ for α =2, 3, 4 and 5 in Table 4.6. The performance for α =2,3, 4 and 5 is consistently high reaching highest for $\alpha = 4$. Considering the proposed objective function in Eq. 4.3, if we put $\alpha = 0$, there will be no quantization step, and the problem becomes a PCA-like objective function where the orthogonality constraint has been smoothed. Similar to PCA, this would lead to poor binary representations due to accumulated quantization error. For $\beta = 0$, the projection matrix $\mathbf{W}$ becomes too dominant and destroys the neighbourhood of data leading to loss of information. Besides, from an optimization point of view, for $\alpha = 0$ and $\beta = 0$, the objective function would become a quadratic function where, given the fact that covariance matrix is positive semi-definite, the maximization is unbounded. As a result, $\alpha = 0$ and $\beta = 0$ do not make sense in the context of this method.

4.3.12 Achieving Non-rigid Projections

We achieved a non-rigid transformation by relaxing orthogonality constraint on projection matrix. We used the well-known Soft Orthogonality (SO) $\|\mathbf{W}^T\mathbf{W}-\mathbf{I}_K\|_F^2$. However, there are other ways of achieving this non-rigid projection matrix [3]. We also tried Double Soft Orthogonality (DSO) $\|\mathbf{W}^T\mathbf{W}-\mathbf{I}_K\|_F^2 + \|\mathbf{W}\mathbf{W}^T-\mathbf{I}_D\|_F^2$, and Mutual Coherence (MC) $\|\mathbf{W}^T\mathbf{W}-\mathbf{I}_K\|_\infty^2$ where we resorted to auto differentiation for implementation. Based on Table 4.7, the simple soft orthogonality SO is achieving better performance in most cases.

Table 4.6: Comparison of retrieval performance for MNIST (GIST 512-D) and LabelMe-12-50k (4096-D VGG-FC7) datasets based on mAP for different values of β and α.

	16 bits, $\alpha = 3$			
Dataset	$\beta = 0.01$	$\beta = 0.05$	$\beta = 0.1$	$\beta = 0.5$
MNIST	64.70	64.29	**67.17**	66.38
LabelMe	**26.08**	25.29	25.29	25.10
	16 bits, $\beta = 0.01$			
Dataset	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
MNIST	63.3	63.18	**65.58**	65.30
LabelMe	24.20	24.54	**25.73**	25.63

Table 4.7: Comparison of SO, DSO, and MC for obtaining non-rigid transformation on NCT-CRC-HE-100K dataset based on mAP and precision@1000.

	mAP %			precision % @1000		
Method	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
SO	71.34	**76.83**	76.57	**82.94**	**86.13**	**86.30**
DSO	71.07	76.29	75.40	81.59	85.60	86.16
MC	**72.43**	73.94	**76.86**	82.70	83.72	85.86

4.4 Conclusions

We introduced Sequential Non-rigid Quantization (SNRQ). The backbone of SNRQ is based on ITQ (iterative quantization) [33]. Although the ideas presented in ITQ are interesting, we argued that learning projection and rotation in two separate steps could be sub-optimal. Furthermore, a rigid transformation may not be enough for reducing quantization to the ultimate limit. Motivated by these limitations, we proposed an algorithm to reduce both dimensionality and quantization loss simultaneously. We also employed a non-rigid transformation to push for quantization beyond rotation. Employing non-rigid transformations is generally against intuition. It does not preserve the neighborhood of data (and all these efforts for reducing quantization error is to preserve more neighborhood after binarization). However, we showed that corrupting neighborhood in favor of reducing quantization eventually leads to better codes. An efficient nested coordinate descent algorithm was employed to update all three matrices. The results on five public datasets

totaling almost half a million images showed that the proposed method outperforms the state-of-art linear hashing methods.

As the future work, we plan to extend the idea of binary representation learning using the rigid and non-rigid transformations to the deep architectures. In this framework, while the non-rigid transformations can be realized using regularization terms similar to the soft orthogonality in Eq. 4.3. For the rigid transformation, one can project the gradient on to orthogonal feasible set using the projection operation proposed in [84] to update the rotation matrix. Besides, it has been recently demonstrated that hashing can be used in the network quantization task [30]. This would be interesting to see how the proposed quantization method performs for network quantization.

Chapter 5

CNN and Deep Sets for End-to-End Whole Slide Image Representation Learning

5.1 Prologue

The content of this Chapter is based on the following paper published during the Ph.D. research:

1. **S. Hemati**, et al. *CNN and Deep Sets for End-to-End Whole Slide Image Representation Learning* Proceedings of Machine Learning Research 143:301–311, 2021

5.2 Introduction

So far in the last two Chapters, we explored methods for learning high quality binary representation learning. However, due to the giagantic size of WSIs, the proposed methods cannot be applied to WSIs directly. Motivated by this limitation,in this Chapter we investigate a simple end-to-end WSI representation learning and then in the next Chapter, we attempt to unify binary and WSI representation learning ideas to achieve a framework for learning efficient representations for WSIs.

As it was pointed out in the related work section, patch extraction is typically the first step for the representation learning of a WSI. Commonly, thousands of representative

patches can be extracted from a WSI. Processing the patches separately instead of the entire WSI eases the memory bottleneck; however, this leads to multi-vector embedding, which is non-trivial to transform to a single vector representation introducing new challenges, e.g., high data usage and compromised retrieval speed [58]. Computing a single-vector representation of a WSI is an active area of research [112, 57]. Ideally, we are interested in a deep-learning solution that can be efficiently trained on WSI patches (at various magnifications), yielding a compact single-vector representation for the WSI, much more suitable for efficient retrieval tasks.

Representing each WSI as bag of image patches makes MIL [18, 57] a natural approach for WSI representation learning [93]. Considering this, employing permutation invariant networks have potential to be used as an effective approach for developing end-to-end WSI representation learning. One recent simple yet effective work on permutation invariant networks is Deep Sets [130]. In the original work detailing Deep Sets, the authors specified a permutation-invariant function and proposed to employ universal set function approximators in neural network. They showed that despite its simplicity, their proposed permutation-invariant architecture can achieve promising performance in a variety of tasks including point cloud classification.

The objective of this Chapter is to propose an end-to-end permutation invariant CNN capable of obtaining a vector representation for a WSI. We use Deep Sets as a simple permutation-invariant neural network which makes it suitable for patch set data for WSI representation learning. We propose to employ a CNN along with Deep Sets to achieve a single global representation per WSI. To this end, we propose two reshape layers to connect our CNN to Deep Sets such that we can train a deep network in an end-to-end manner. Note that having one global representation for each WSI enables us to train our network in a multi-label classification scheme such that the targets for each WSI are primary site and primary diagnosis. This enables the proposed CNN-Deep Sets (CNN-Deep Sets (CNN-DS)) architecture to be used for WSI search in both horizontal search (search for primary site) and vertical search (search for primary diagnosis) [58]. In order to further guide the proposed CNN-DS, we employ hierarchical multi-label training where primary site information is used to predict primary diagnosis labels. This idea is based on the fact that every primary site has its own disease sybtypes so we prevent the network from predicting meaningless diseases/primary site pairs. We show that the proposed network coupled with hierarchical multi-label training can be used for WSI representation. We validate the proposed scheme against Yottixel for the image search task on The Cancer Genome Atlas (TCGA) dataset [122, 14] both in terms of retrieval performance and speed.

5.3 Method

5.3.1 Preprocessing

The common practice to deal with gigapixel WSIs is patch extraction which leads to a bag of patches (set representation) [113]. This type of image embedding pulls out some patches so that the network can train on a smaller set without sacrificing too much of tissue information. As it is not always clear which areas of a WSI are the regions of interest, patch extraction is challenging. In particular a chosen patch may not be relevant to the WSI diagnosis as it may contain exclusively healthy tissue or a combination of healthy and malignant tissue. Considering this, the patch extraction step is crucial as we may loose valuable information. This problem can be more severe in patch-based training as we assign WSI label to each patch which may not be correct. In this paper, we employ a patch extraction algorithm used in Yottixel [58]. The patch selection method selects the representative patches from a WSI. We removed non-tissue portions of WSIs using colour threshold. The remaining tissue-containing patches are grouped into a pre-set number of categories through a clustering algorithm (we chose 9, and K-means algorithms). A portion of all clustered patches (e.g., 10%) are randomly selected within each cluster, yielding a mosaic. The mosaic is transformed into a set of features, obtained through a deep network (shown in Figure 5.1). The mosaic is meant to be representative of the full WSI, and enables much computational convenient computation for training of neural networks we randomly accept 40 patches from the mosaic.

5.3.2 Proposed CNN-Deep Sets (CNN-DS)

Representing a WSI by a mosaic of patches reduces our WSI to a set representation learning problem. Motivated by this, we propose applying Deep Sets [130] to learn a permutation-invariant representation for each WSI in an end-to-end manner. The general architecture proposed in Deep Sets for representation of set X that contains elements $x_1, x_2, \dots x_n$ follows the following form, $f_X = \phi(\theta(x_1), \dots, \theta(x_n))$, where, f_X is the set representation, θ is a non-linear mapping and ϕ is a pooling operation, including sum, mean, and max. In the Deep Sets paper, the authors proved that their proposed architecture was capable of acting invariantly and universally on set inputs approximate any set function [130]. The universal invariance refers to the property that shuffling the input vector does not result in a change of the output vector; mathematically, for any reordering, $\pi(i)$: $F(\{x_1, x_2, x_3, \dots, x_n\}) = F(\{x_{\pi(1)}, x_{\pi(2)}, x_{\pi(3)}, \dots, x_{\pi(n)}\})$. In this Chapter, we employ the max pooling operation

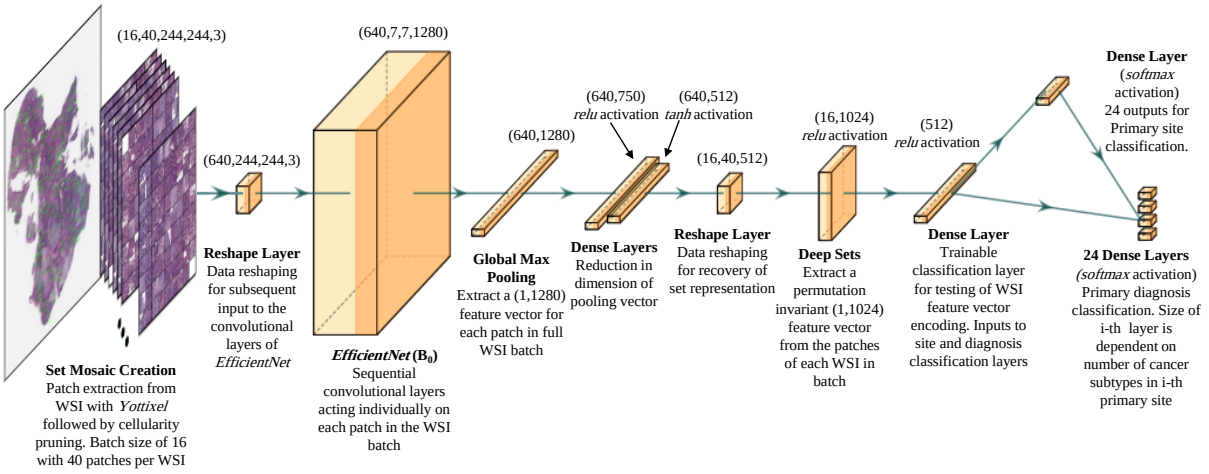


Figure 5.1: Proposed architecture for end-to-end WSI representation leaning. This architecture contains four main modules namely reshape layers, a CNN, Deep Sets, and a hierarchical multi-label learning layer.

for symmetric function part of Deep Sets as it it has been shown to be superior to other pooling layers for set representation learning [130].

CNN-DS Design for End-To-End Training

To have an end-to-end algorithm that learns high-quality permutation invariant representation per WSI, we employ EfficientNet B0 [111] prior to the Deep Sets model. Fig. 5.1 shows our proposed CNN-DS architecture.

Crucial to the design of our network are two reshape layers: one before the CNN (EfficientNet B_0 here) and one before the permutation-invariant Deep Sets. The first reshape layer is necessary to feed into the convolutional layers. Considering batch size of 16, extracting 40 patches per each WSI (set size= 40), and resizing patches from 1000 $\times$ 1000 to 224 $\times$ 224, the input tensor of our network has the shape (16,40,224,224,3). To feed this 5-dimensional tensor to the CNN-Deep Sets, we use the reshape layer to turn the input tensor into a 4-dimensional tensor with shape (640,224,224,3) so that in EfficientNet each patch is treated as an individual image and not part of a set. EfficientNet then transforms the data into shape (640,7,7,1280) which is further reduced to shape (640,1280) by the global max pooling. To prepare this matrix for Deep Sets, we process it with two dense layers. These layers reduce the dimensionality and apply a symmetric activation function

$\tanh(\cdot)$ which is helpful before symmetric functions employed by Deep Sets. After these dense layers, the data shape is (640,512). To retain the set nature of the data, our second reshape layer changes the dimensionality to (16,40,512). This data is then given to Deep Sets to obtain a global representation for each WSI, which was represented by a set of patches. After Deep Sets we have a (16,1024) representation where each WSI has been embedded as a 1024 dimensional vector.

CNN-DS Design for Multi-label Training

To update the network parameters, the vector embedding of the WSI outputted by Deep Sets is used in a multi-label classification task where labels are primary site and primary cancer subtypes. First, the output of Deep Sets is inputted to two different dense layers for primary site and cancer subtype classification. The elevated layer in Fig. 5.1 is the primary site classifier component with 24 outputs and a softmax activation function where each output predict a primary site probability for the WSI. Since every primary site has its own cancer subtype, we can use the primary site predicted label to predict the primary diagnosis label. We therefore design the final lower layer to be a set of 24 layers associated with 24 primary sites where number of outputs for each layer is equal to the number of cancer subtypes for that primary site. For example, if the first layer in the lower final layer represents the brain as the primary site, then the primary diagnosis type layer will either be Glioblastoma Multiforme (GBM) or Lower Grade Glioma (LGG) - only two possible outputs with a softmax activation function. This layer therefore calculates the $P(\text{GBM}|\text{Brain})$ and $P(\text{LGG}|\text{Brain})$ probabilities. However, we aimed to calculate $P(\text{GBM})$ and $P(\text{LGG})$ probabilities with this assumption that we know the probability of the given WSI is Brain, i.e., $P(\text{Brain})$ which we can obtained from upper final layer. To do this we use law of total probability as follows:

$$P(\text{GBM}) = P(\text{GBM}|\text{Brain})P(\text{Brain}) \tag{5.1}$$

$$P(\text{LGG}) = P(\text{LGG}|\text{Brain})P(\text{Brain}) \tag{5.2}$$

The multiplication between $P(\text{Brain})$ and $P(\text{GBM}|\text{Brain})$, or $P(\text{LGG}|\text{Brain})$ is shown using the connection between upper and lower final layers. We develop these layers for all other primary sites and their corresponding cancer subtypes where categorical cross entropy is used as loss function. Compared with a simple multi-label training with a sigmoid activation and binary cross entropy, this guided multi-label training needs significantly fewer epochs.

Training

All patches were reduced from 1000 by 1000 to 224 by 244 images. Then, for each WSI we ended up with a tensor of shape (40, 224, 244, 3) where 40 is number of patches per WSI. We set the batch size to 16 which leads to a tensor shape (16, 40, 224, 244, 3) for one batch of data. A batch of this size is quite large, leading to issues in regular GPU memory and run times. To handle data of this size we employed four Tesla V 100 GPUs in parallel mode. We employed the Adam optimizer [61] with 0.000001 learning rate to avoid instabilities. The Albumentations library [5] was used to apply horizontal and vertical flip, 90 degree rotation, shifting and scaling data augmentation. Finally, in the last two dense layers we employed dropout at a 0.25 rate.

5.4 Results

To validate the proposed architecture for WSI representation, we employ the CNN-DS to obtain one feature vector for set of patches (here 40) per WSI. The output of the feature extractor for the proposed architecture is obtained from the dense layer after the Deep Sets layer, a 512 dimensional representation for each WSI. Unlike the training, obtaining WSI representations for test data can be done using a regular GPU. To investigate the quality of obtained WSI representations we validate the obtained features in the image search task for test data. We compare the proposed method with Yottixel search engine [58] on two different WSI search tasks, namely, horizontal and vertical search. Horizontal search refers to how accurate we can find the tumour type across the entire test database. Vertical search quantifies how accurately we find the correct cancer subtype of a tumour type among the slides of a specific primary site including different primary diagnoses. Due to small size of test set, we employ leave-one-out strategy and report the average scores.

5.4.1 Dataset

We employ 5861, 281, and 604 WSIs unfrozen sections from TCGA for training, validation, and testing, respectively. The dataset spanned 24 primary sites and 30 primary cancer diagnoses. The tumour types available in the dataset include brain, breast, endocrine, gastrointestinal tract, gynecological, hematopoietic, liver/pancreaticobiliary, melanocytic, head and neck, prostate/testis, pulmonary, and urinary tract.

5.4.2 WSI Search

The k-nearest neighbors (k-NN) horizontal search results both for $k = 3$ and $k = 5$ are shown in Table 5.1. Clearly, almost in all primary sites there is a significant improvement in retrieval performance compared with Yottixel search engine. Table 5.2 presents the k-NN vertical search result using Yottixel and WSI embeddings obtained from CNN-DS. Unlike horizontal search, CNN-DS obtained better results in all cases compared with Yottixel in vertical search; in some cases Yottixel achieves better results. Looking more closely at these cases, the improvement of Yottixel against CNN-DS is not significant in most cases. Fig. 5.2 shows the 2-D representation of obtained WSI embedding using CNN-DS labelled based on primary site and primary diagnosis labels.

Table 5.1: Majority-3 and 5 search accuracy (%) for the horizontal search (primary site identification) among 604 WSIs for Yottixel and CNN-DS (best results in green).

		Accuracy (in %)			
Tumor Type	Patient #	Yottixel ($k = 3$)	CNN-DS ($k = 3$)	Yottixel ($k = 5$)	CNN-DS ($k = 5$)
Brain	46	73	91	73	89
Breast	77	45	77	38	79
Endocrine	71	61	66	59	62
Gastro.	69	50	75	49	74
Gynaec.	18	16	33	0	27
Head/neck	23	17	69	13	65
Liver	44	43	56	36	43
Melanocytic	18	16	50	5	38
Mesenchymal	12	8	100	0	83
Prostate/testis	44	47	81	43	77
Pulmonary	68	58	91	54	89
Urinary tract	112	67	76	62	74

5.4.3 WSI Classification

The Lung Adenocarcinoma (LUAD) and Lung Squamous Cell Carcinoma (LUSC) are two main cancer types of non-small cell lung cancer. The classification of LUAD versus LUSC can aid pathologists in diagnosis of these cancer subtypes that include 65-70% of all lung cancers [131]. To validate the performance of CNN-DS, we apply it to LUAD/LUAC classification task. We gathered 2,580 (H&E) stained WSIs of lung cancer from TCGA repository. Among this, we employ 1,806 for training set and the remaining 774 WSIs for test set [57]. The patch selection and the architecture design of CNN-DS is the same as the one that used in transfer learning task. We avoid training convolutional layers to have a fair comparison against other transfer-learning based methods. The results have

been reported in Table 5.3 where CNN-DS can achieve competitive performance against the state-of-art.

Table 5.2: Majority-3 and -5 search through k-NN for the vertical search among 604 WSIs. Best F1-measure values highlighted.

Site	Subtype	n_{slides}	F1-measure (in %)			
			Yottixel	CNN-DS	Yottixel	CNN-DS
Brain	LGG	23	78	89	75	81
	GBM	23	82	89	83	84
Endocrine	THCA	50	92	98	91	98
	ACC	6	25	28	28	0
	PCPG	15	61	81	61	79
Gastro.	ESCA	10	12	44	25	55
	COAD	27	62	69	54	70
	STAD	22	61	64	57	78
	READ	10	30	55	16	0
Gynaeco.	UCS	3	75	80	50	50
	CESC	6	92	66	76	80
	OV	9	80	82	66	82
Liver, panc.	CHOL	4	26	0	25	0
	LIHC	32	82	95	87	95
	PAAD	8	94	94	77	94
Prostate/testis	PRAD	31	98	97	95	96
	TGCT	13	96	93	86	93
Pulmonary	LUAD	30	62	61	62	61
	LUSC	35	69	60	69	62
	MESO	3	0	50	0	0
Urinary tract	BLCA	31	89	95	86	94
	KIRC	47	91	87	89	84
	KIRP	25	75	84	79	81
	KICH	9	70	53	66	0

5.4.4 Query time comparison against Yottixel

We inteded to obtain one global representation for a WSI. We argued that this is particularly useful for WSI search as the set representation is bypassed. Hence, we measured query time for the leave-one-out approach used for 604 WSIs. Results showed that while for Yottixel it takes around 16 minutes to calculate pairwise distances between 604 WSIs, in our case it takes around 20 seconds to reproduce the results.

In the following, the full description of the abbreviations for cancer subtypes in Table 5.2 have been presented in Table 5.4.

Table 5.3: CNN-DS evaluation on lung cancer classification via transfer learning.

Algorithm	Accuracy (in %)
Kalra & Adnan et al. [57]	84
Khosravi et al. [60]	83
Yu et al. [128]	75
CNN-DS (Ours)	86

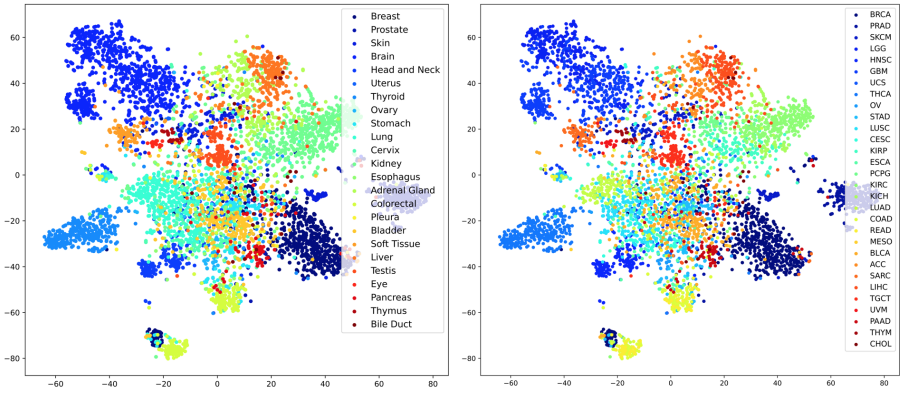


Figure 5.2: 2-D representation of obtained WSI embedding using CNN-DS labelled based on 24 primary sites (left) ad 30 primary diagnoses (right). As can be seen, embeddings for WSIs with different classes have been separated to a good extent.

5.5 Conclusion

We employed Deep Sets along with a CNN for end-to-end WSI representation. This was inspired by bag of patches (set) representation per WSI. Two reshape layers connected CNN with Deep Sets. We propose to train our CNN-DS in the multi-label scheme. We used the law of total probability to capture the primary site predicted probability for obtaining probability of primary diagnosis. We validated the proposed topology in a transfer learning scheme for WSI search. We showed that the proposed architecture can obtain WSI embeddings leading to comparable retrieval performance compared with Yottixel while reducing the retrieval time significantly. We also applied the proposed scheme to lung classifcation task and achieved competitive results compared with the state-of-art. One limitation of the proposed method is number of small patches (here 40) that has been used to model each WSI. In future works we would to sample more patches per WSI and see how it improve

Table 5.4: Full description for primary diagnosis abbreviations used in the paper.

Abbreviation	Primary Diagnosis
ACC	Adrenocortical Carcinoma
BLCA	Bladder Urothelial Carcinoma
CESC	Cervical Squamous Cell Carcinoma and Endocervical Adenoc.
CHOL	Cholangiocarcinoma
COAD	Colon Adenocarcinoma
ESCA	Esophageal Carcinoma
GBM	Glioblastoma Multiforme
KICH	Kidney Chromophobe
KIRC	Kidney Renal Clear Cell Carcinoma
KIRP	Kidney Renal Papillary Cell Carcinoma
LGG	Brain Lower Grade Glioma
LIHC	Liver Hepatocellular Carcinoma
LUAD	Lung Adenocarcinoma
LUSC	Lung Squamous Cell Carcinoma
MESO	Mesothelioma
OV	Ovarian Serous Cystadenocarcinoma
PAAD	Pancreatic Adenocarcinoma
PCPG	Pheochromocytoma and Paraganglioma
PRAD	Prostate Adenocarcinoma
READ	Rectum Adenocarcinoma
STAD	Stomach Adenocarcinoma
TGCT	Testicular Germ Cell Tumors
THCA	Thyroid Carcinoma
UCS	Uterine Carcinosarcoma

the performance of the model. Further, here we started from the ImageNet wights which may not be the best possible option for weight initialization. Given the recent success of self supervised methods, it would be interesting to explore how the performance changes if we start from a set of weights that have been obtained from self supervised training.

Chapter 6

Sparse and Binary Permutation-Invariant Whole Slide Image Representation Learning Without Memory Bottleneck

6.1 Prologue

The content of this Chapter is based on the following paper published during the Ph.D. research:

1. **S. Hemati**, et al. *Compact Whole Slide Image Representation Learning Without Memory Bottleneck* under submission.

6.2 Introduction

So far, in Chapters 3 and 4 we explored methods for binary representation learning of images. The main issue with these methods is the fact that they can only be used for patches and not WSIs. Then in Chapter 5 we explored a simple WSI representation learning framework based on the MIL and the permutation invariant neural network. The main issues with this approach were the huge memory usage during the training and also lack of ability to obtain binary representations for WSIs. In this Chapter, we propose

a framework that unifies the ideas from earlier Chapter where the objective is obtaining compact (binary and sparse) permutation invariant representations for WSIs while the memory usage during the training is reduced.

MIL enables learning on set data instead of using single instances during training. MIL is an appropriate method applicable to WSI representation, and as a result, there is a large body of papers exploring various MIL schemes for WSI representation learning [18, 41, 93, 45, 49, 57]. Although MIL has become a preferred method for WSI representation, it does have several limitations. Among others, MIL requires all instances to be processed at once as a set (called *bag*), making it difficult to develop end-to-end training in a memory-efficient manner. Another issue with existing WSI representation methods is that the obtained embeddings cannot be directly used for WSI search in its raw form. Searching within large archives of WSIs through the nearest neighbour search would lead to a prohibitively large increase in memory demand and retrieval time [117]. As a result, the ancillary processing method is usually necessary to encode these embeddings into more suitable forms, i.e., binary and sparse embeddings facilitating the speed and memory efficiency in nearest neighbour search. Finally, current WSI engines, i.e., *Yottixel* [58], *SMILY* [40] ignore the a-priori knowledge, such as tumor type, about WSIs for performing the search. It is ideal to employ all known attributes of WSIs for producing a more effective embedding. For this Chapter, our contribution is design of neural network for WSI representation learning which provides:

1. The compact (sparse and binary) and permutation-invariant WSI representations ideal for efficient WSI search in large archives.

2. the permutation-invariant representation of WSIs, trained end-to-end by feeding individual instances instead of a bag of instances which eases up the time and memory bottlenecks, enabling our methods to even incorporate patches at multiple magnification levels.

3. Learning representations guided by a-priori information, i.e., the tumor type as a way of self-supervision.

To the best of our knowledge, such a network that provides the above-mentioned functionalities does not exist in the literature of medical imaging and designing it can be very helpful. Having the mentioned objectives in mind, here we propose our idea for learning compact WSI/patch representation with no memory issue. The proposed scheme is based on the combination of old-school machine learning algorithms and recent advances

in deep generative models. We further present some novel ideas to unify learning WSI representation learning and learning binary & sparse representations.

The rest of this Chapter is organized as follows: In section 6.3 we briefly review the current related works on WSI representation learning. Then, in section 6.4 we provide the details of our proposed framework. Next, in section 6.5 we validate the effectiveness of our approach for search and classification tasks on two publicly available benchmark datasets. Finally, we conclude the Chapter in the section 6.6.

6.3 Related works

In this section, we review the related literature on WSI representation learning. We organized the related literature into three main themes, i.e., heuristic deep architectures, glsMIL-based methods, and dictionary learning approaches.

6.3.1 Heuristic Architectures

These methods generally split the task into multiple separate steps to simplify the problem. First, there is an instance-based training where instances are smaller parts of WSIs, typically patches. Then, another network is trained to obtain WSI embeddings while capturing the spatial relationship between patches. For example, Bejnordi et al. [4] proposed to employ two sub-networks for processing high and low-resolution information separately and then attaching two networks together. Other works in this category are Spatio-Net [64] and the neural compression scheme proposed by Tellez et al. [112]. In Spatio-Net, a grid of embeddings for each patch and its neighbours are obtained by a CNN feature extractor, and then they are processed by 2D-LSTM layers to capture the spatial information. Tellez et al. [112] proposed a two-stage neural compression where the first stage is devoted to unsupervised representation learning of grid of all image patches per WSI. Then, they employed this trained model to obtain compressed patches and WSI. Finally, in one recent work, authors in [83] proposed a framework to choose between low and high-resolution information for WSI classification.

6.3.2 Multiple Instance Learning

Representing each WSI as a bag of patches makes MIL-based schemes a natural approach for end-to-end WSI representation learning [18, 93, 57, 51, 49]. One of the early works in

MIL-based WSI classification was conducted by Hou et al. [45] where they first trained a patch level classifier and then a fusion model using MIL scheme to achieve WSI classification. In fact, one can regard this approach as a two-step instance-based MIL method where an algorithm determines instance classes. Motivated by this, Chikontwe et al. [11] proposed an end-to-end MIL-based method for simultaneous patch and WSI representation learning in a single framework where a center loss is introduced to map patch embeddings from the same WSI to a single centroid. They showed that their approach achieves promising results compared with other MIL-based methods, especially two-stage MIL methods. Other recent MIL methods include [49] and [41] where pooling layers based on attention mechanism and Deep Sets [130] have been proposed. Finally, Kalra et al. [56] employed focal factor learning to modulate the aggregated patch-level predictions.

6.3.3 Dictionary Learning

Another approach that can be used for the WSI representation is the *bag of visual words* (BoVW) [17] for encoding local image descriptors into one embedding. A more advanced version of BoVW that captures higher-order statistics to obtain the set representation is based on the *Fisher Kernel* theory and generative models [52]. Authors in [89] introduced Gaussian mixture model (GMM)-based Fisher Vector which can be calculated using the normalized gradient of the log-likelihood of the GMMs with respect to parameters, such as mixing coefficients, means, and variances, given a set of observations. Further, recently there has been some research to extract Fisher Vector from deep generative models [92, 132]. Although this set encoding ability makes Fisher Kernel a natural candidate for WSI representation learning. There are only a few papers that use Fisher theory and dictionary learning in general for the WSI representation task [109, 110, 136]. The reason could be attributed to the fact that Fisher Vector is formulated in a fully unsupervised manner using GMMs. However, considering the challenges inherent to pathology images (e.g., complex textures and colour variations), employing available WSI information, i.e., tumor type and primary diagnosis, in obtaining an efficient global representation is necessary. Further, GMM-based Fisher Vector captures no more than second-order statistics of data for set encoding. Besides, the training of GMMs is sub-optimal and not end-to-end. Finally, the obtained encodings are generally high-dimensional embeddings in Euclidean space, which are less desirable for WSI search due to their increased distance computation times.

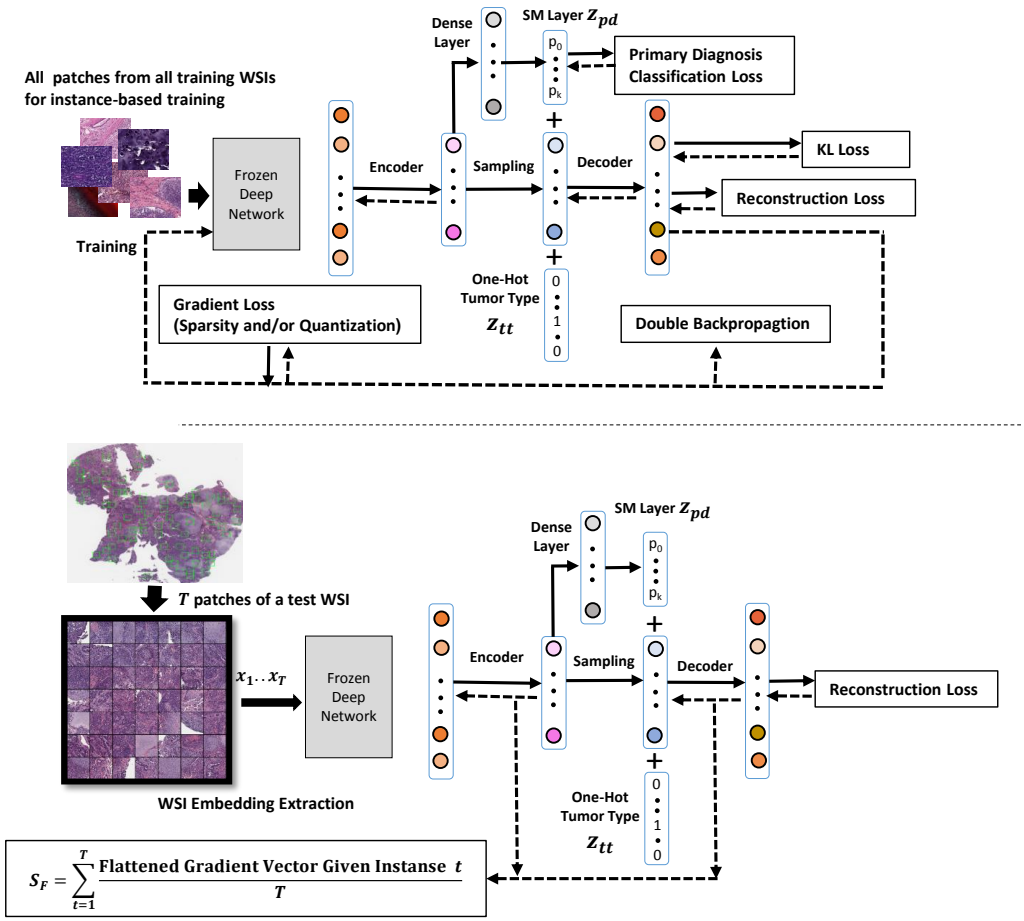


Figure 6.1: The first row represents the proposed architecture and associated instance-based training scheme. The second row shows the procedure for obtaining the WSI embedding for a set of patches of this WSI given the trained model.

6.4 Method

This section presents the proposed framework for learning compact WSI representations. First, we briefly review the relevant concepts, i.e., Fisher Kernel [52] and Fisher Vector theories [89]. Next, we describe the proposed method based upon VAEs and Fisher Vector theory. The proposed method is memory efficient during training and learns representations that are permutation-invariant, compact (sparse/binary), and can be conditioned on known information (e.g., the given tumor type) for the self-supervision. The proposed method is trained in an end-to-end manner on individual instances instead of a bag of instances to obtain representations for both patches and the WSI in its entirety.

6.4.1 Preparation

The key idea behind Fisher Kernel is to derive the kernel function from a generative probability model. Initially, the main motivation for deriving such kernels was bridging the gap between generative and discriminative models [52]: "the gradient of the log-likelihood with respect to a parameter describes how that parameter contributes to the process of generating a particular example". As a result, to take advantage of generative models in discriminative tasks, Jaakkola and Haussler proposed to employ the gradient space of the generative models to use the generative process as a similarity metric between examples (or set of examples, i.e., $\mathbf{X} = \{\mathbf{x}_t, t = 1, \ldots, T\}$ where T is the number of examples in the set) [52]. Let us consider a class of probability models $p(\mathbf{X} \mid \boldsymbol{\theta})$ where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is a parameter vector and $\mathbf{X}$ is set of examples, i.e., $\mathbf{X} = \{\mathbf{x}_t, t = 1, \ldots, T\}$. The Fisher Score is then defined as

$$U_{\mathbf{X}} = \nabla_{\boldsymbol{\theta}} \log p(\mathbf{X} \mid \boldsymbol{\theta}), \tag{6.1}$$

where the $U_{\mathbf{X}} \in \mathbb{R}^d$. The dimensionality d of the Fisher Score is equal to the number of parameters in the generative model $p(\mathbf{X} \mid \boldsymbol{\theta})$ independent of the number of data points in the set T. The FIM is

$$\mathbf{I} = E_{\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\theta})}\{U_{\mathbf{X}} U_{\mathbf{X}}^T\}. \tag{6.2}$$

Subsequently, the Fisher Kernel can be defined as

$$K(\mathbf{X}, \mathbf{Y}) = U_{\mathbf{X}}^T \mathbf{I}^{-1} U_{\mathbf{Y}} \tag{6.3}$$

Fisher Kernel can be used to calculate the similarity between two sets of data points [52].

Authors in [89] proposed the GMM-based Fisher Vector as a way to encode a set of local descriptors in a single embedding where the Fisher Vector is the normalized Fisher Score ($\mathbf{s}_F$) calculated as

$$\mathbf{s}_F = \frac{1}{T}\mathbf{L}\nabla_{\boldsymbol{\theta}} \log p(\mathbf{X} \mid \boldsymbol{\theta}) = \mathbf{L}\frac{1}{T}\sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_t \mid \boldsymbol{\theta}), \tag{6.4}$$

where $\mathbf{L}$ is calculated from Cholesky decomposition of inverse FIM, i.e., $\mathbf{I}^{-1} = \mathbf{L}^T\mathbf{L}$, with that assumption that data points in $\mathbf{X}$ are statistically independent.

6.4.2 Deep Compact Fisher Vector

Although GMM-based Fisher Vector outperforms BOW in the sense that it captures second-order statistics for obtaining the set representation, there are some issues that makes them unsuitable for WSI representation learning. First, GMMs are trained in a fully unsupervised manner while considering the challenges inherent to pathology images (e.g., challenging textures, colour variations, etc.) employing available information, i.e., primary site or primary diagnosis of the WSI in obtaining an compact global representation is necessary. Second, GMMs are sub-optimal as this cannot be applied in an end-to-end manner to the images. Besides, GMMs are not able to fully capture the natural clustering of the patch descriptors. This is due to the inefficient training scheme in GMMs and the fact that by employing the GMMs, no more than second-order statistics of data are captured using Fisher Vector. Motivated to remove these limitations, we propose a new type of Fisher Vector based on the deep generative models for the WSI representation learning. The contributions of our method are as follows.

1. To capture higher-order statistics while learning the set representation, we propose to employ generative models VAE here [62] for WSI representation learning.
2. We add a classification loss to the training such that the available WSI level primary diagnosis labels are employed during the training of the VAE.
3. We design the VAE to be conditioned on available information, e.g., the tumor type. Given the fact that every tumor type has its own specific cancer subtypes, this conditioning is expected to improve the quality of WSI embeddings.
4. More importantly, we propose two novel loss functions for compact (sparse and binary) and permutation-invariant WSI representation learning.

We start by training a VAE and then modify the VAE to be conditioned on tumor type. We add a classification loss to the end of the encoder part such that primary diagnosis label information is injected into the model space. Finally, we propose two novel loss functions for learning sparse and binary permutation-invariant representations.

VAE Loss Function

To learn the encoder and decoder parameters of the VAE, i.e., $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, that models distribution of $\mathbf{x}$, we assume the prior distribution on the random variable $\mathbf{z}$ is $p_{\boldsymbol{\theta}}(\mathbf{z})$ and as a result, $\mathbf{x}_t$ is sampled from $p_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{z})$. In this case, one can show that the lower bound for the $\log p_{\boldsymbol{\theta}}(\mathbf{x})$ can be calculated as

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) \geq -q_{\boldsymbol{\phi}}(\mathbf{z} \mid \mathbf{x}) p_{\boldsymbol{\theta}}(\mathbf{z}) + E_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{z})\right], \tag{6.5}$$

where $q_{\boldsymbol{\phi}}(\mathbf{z} \mid \mathbf{x})$ is the approximate posterior with parameters $\boldsymbol{\phi}$. The lower bound in Eq. 6.5 is known as variational lower bound and on the patch $\mathbf{x}_t$ and it is represented with $\mathcal{LB}(\boldsymbol{\phi}, \boldsymbol{\theta}, \mathbf{x}_t)$. We aim to maximize $\mathcal{LB}$ to learn the generative model parameters. In context of the VAE model, $q_{\boldsymbol{\phi}}(\mathbf{z}_t \mid \mathbf{x}_t)$ and $p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t)$ are encoder and decoder, respectively. In order to learn the encoder and decoder parameters, i.e., $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, first we assume the prior distribution $p_{\boldsymbol{\theta}}(\mathbf{z}_t)$ is $\mathcal{N}(\mathbf{z}_t; 0, \mathbf{I})$ and $q_{\boldsymbol{\phi}}(\mathbf{z}_t \mid \mathbf{x}_t)$ and $p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t)$ follow the normal distributions $\mathcal{N}(\mathbf{z}_t; \boldsymbol{\mu}_{\mathbf{z}_t}, \boldsymbol{\sigma}^2_{\mathbf{z}_t}\mathbf{I})$ and $\mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_{\mathbf{x}_t}, \boldsymbol{\sigma}^2_{\mathbf{x}_t}\mathbf{I})$. In this case, by estimating the latent code using one-step Monte Carlo, the variational lower bound is

$$\mathcal{LB}(\boldsymbol{\phi}, \boldsymbol{\theta}, \mathbf{x}_t) = \log p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t) + \frac{1}{2}\sum_{j=1}^{d}(1 + \log \boldsymbol{\sigma}^2_{\mathbf{z}_t(j)}) - \frac{1}{2}\|\boldsymbol{\mu}_{\mathbf{z}_t}\|^2 - \frac{1}{2}\|\boldsymbol{\sigma}_{\mathbf{z}_t}\|^2, \tag{6.6}$$

where $\mathbf{z}_t$ is sampled from $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{z}_t}, \boldsymbol{\sigma}^2_{\mathbf{z}_t}\mathbf{I})$.

Conditioned VAE

As the tumor type of a WSI is always available, we conditioned VAE on the tumor type of the given WSI to draw benefit from this apriori knowledge. Following the [107], let's represent the tumor type as a one-hot encoded vector $\mathbf{z}_{tt}$, then we concatenate this vector to the $\mathbf{z}_t$. Furthermore, to inject WSI-level primary diagnosis information into the generative model, we add a classification loss to the last layer of the encoder before the sampling layer. We assign the WSI label to all patches extracted from that WSI. Then, we concatenate the softmax of predicted primary diagnosis $\mathbf{z}_{pd}$, with length k, to the latent space. The

latent space associated with $\mathbf{x}_t$ that is fed to decoder is modified as $\mathbf{z}_t \leftarrow [\mathbf{z}_t, \mathbf{z}_{tt}, \mathbf{z}_{pd}]$. Considering the classification loss, so far, the loss function for training the Conditioned Variational Autoencoder (CVAE) has the form

$$\mathcal{L}_{\text{CVAE}} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{kl} + \lambda_3 \mathcal{L}_{cls}, \tag{6.7}$$

where minimizing the first two terms is equivalent to maximizing the variational lower bound, and the third loss is the classification loss of predicting cancer subtypes.

Deep Sparse Fisher Vector

Now, we propose a novel method for learning Conditioned Deep Sparse Fisher Vector (C-Deep-SFV). As the gradient space represents the WSI, we encourage sparsity in the gradient by adding the l_1 norm of the gradient of the loss function in Eq.6.7 to the overall training loss. To regularize the gradient, we utilize the *double backpropagation*, where given a batch of data points $\mathbf{X}$; the loss function can be written as

$$\mathcal{L}_{\text{SFV}} = \mathcal{L}_{\text{CVAE}} + \lambda_4 \sum_{\mathbf{W}_i \in \mathbb{W}} \|\nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X})\|_1, \tag{6.8}$$

where $\mathbb{W}$ is the set of CVAE parameters for all layers, $\mathbf{W}_i$ and $\nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X})$ are the is parameters and the gradient of the CVAE loss with respect to the i^{th} layer parameters. To the best of our knowledge, such an end-to-end Sparse Fisher Vector learning does not exist in the literature.

Deep Binary Fisher Vector

For learning Conditioned Deep Binary Fisher Vector (C-Deep-BFV), inspired by the quantization-based learning in hashing literature [33, 42], we propose to reduce the quantization loss of the gradient of the CVAE loss with respect to each layer's parameters. We propose to find

$$\begin{aligned} \underset{\mathbf{B}_i, \nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X})}{\arg\min} \quad & \sum_{\mathbf{W}_i \in \mathbb{W}} \|\nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X}) - \mathbf{B}_i\|_2^2 \\ \text{s.t.} \quad & \mathbf{B}_i \in \{-1, 1\}^{d_i \times 1}, \end{aligned} \tag{6.9}$$

where $\mathbf{B}_i$ is the flattened binary representation of the gradient of the CVAE loss w.r.t parameters of the i^{th} layer. In this case, the loss function to obtain BFV can be written as

$$\mathcal{L}_{\text{BFV}} = \mathcal{L}_{\text{CVAE}} + \lambda_5 \sum_{\mathbf{W_i} \in \mathbb{W}, \mathbf{B_i} \in \mathbb{B}} \|\nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X}) - \mathbf{B}_i\|_2^2, \tag{6.10}$$

where $\mathbb{B}$ is the set of closest hamming vertices to gradients w.r.t all layers. Given the binary optimization variable $\mathbf{B}_i$, on each epoch, we employ the coordinate descent approach and update each of $\mathbf{B}_i$ and $\mathbf{W}_i$ while the other is fixed. For the case that $\mathbf{W}_i$ is fixed the problem turns to

$$\begin{aligned} \arg\min_{\mathbf{B}_i} \quad & \|\nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X}) - \mathbf{B}_i\|_2^2 \\ \text{s.t.} \quad & \mathbf{B}_i \in \{-1, 1\}^{d_i \times 1}, \end{aligned} \tag{6.11}$$

where by expanding Eq.6.11 it turns out the above minimization is equivalent to

$$\begin{aligned} \arg\max_{\mathbf{B}_i} \quad & \mathbf{B}_i^T . \nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X}) \\ \text{s.t.} \quad & \mathbf{B}_i \in \{-1, 1\}^{d_i \times 1}. \end{aligned} \tag{6.12}$$

This problem has the following closed-form solution [33]:

$$\mathbf{B}_i = sgn(\nabla_{\mathbf{W}_i} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X})). \tag{6.13}$$

The loss function can be for fixed $\mathbf{B}_i$ as

$$\mathcal{L}_{\text{BFV}} = \mathcal{L}_{\text{CVAE}} + \lambda_5 \sum_{\mathbf{W_i} \in \mathbb{W}} \|\nabla_{\mathbf{W_i}} \mathcal{L}_{\text{CVAE}}(\mathbb{W}, \mathbf{X}) - \mathbf{B}_i\|_2^2 \tag{6.14}$$

This is similar to SFV learning in Eq. 6.8. The variables can be updated using double backpropagation.

Deep Sparse Binary Fisher Vector

Knowing that the length of obtained WSI embeddings is equal to the number of parameters in the generative model, we may be interested in compact (short) binary codes for more efficient WSI retrieval. We propose to employ both gradient sparsity and gradient

quantization losses to achieve Conditioned Deep Sparse Binary Fisher Vector (C-Deep-BFV). Gradient sparsity pushes the generative model to use fewer parameters to generate a data point. As a result, the quality of embedding will be more robust to dropping some dimensions, i.e., gradient w.r.t some parameters of VAE. To choose effective dimensions for each tumor type we find the top M parameters that provide the highest variance in their respective gradient values for the training data.

VAE Architecture and Training Scheme

The architecture of the proposed conditioned VAE is given in the first half of 6.1. We employed a frozen pre-trained CNN (DenseNet-121 [48]) as the backbone of the VAE. Each encoder and decoder parts contain three fully connected layers. The last layer of the encoder is fed to a softmax layer (SM Layer in 6.1) for primary diagnosis prediction. In order to condition the VAE, for each patch, the output of the softmax layer along with a one-hot encoded vector representing the available tumor type information of the patch is concatenated to the latent vector to create the $\mathbf{Z}_t$. Then, this vector is fed to the decoder part. As it can be seen from the Fig. 6.1, the CVAE is trained on a per-instance basis enabling to include even patches from multiple magnifications.

WSI Embedding Extraction

After the training phase, to obtain a single embedding for a WSI, all patches of that WSI are fed to the CVAE (see the second half of Fig. 6.1). Then, given the reconstruction loss, we calculate the average gradient over all patches using backpropagation to obtain the Fisher Score ($\mathbf{s}_F$). Based on Fisher Theory, we also need $\mathbf{L}$ obtained from FIM to normalize the vector and derive the Fisher Vector. However, given the computational load of calculating $\mathbf{L}$, we replace this with identity matrix and normalize the gradient using power and l_2 normalization steps proposed by [90]. In other words, representing the power and l_2 normalization steps as $\mathcal{S}(\cdot)$ operator, the conditioned deep compact Fisher Vector $\mathbf{v}_F$ is calculated from the Fisher Score $\mathbf{s}_F$:

$$\mathbf{v}_F = \mathcal{S}\left(\frac{1}{T}\sum_{t=1}^{T}\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}}\|\mathbf{x}_t - \hat{\mathbf{x}}_t(\boldsymbol{\theta},\boldsymbol{\phi})\|_2^2\right) = \mathcal{S}\left(\mathbf{s}_F\right). \tag{6.15}$$

where $\mathbf{x}_t$ and $\hat{\mathbf{x}}_t(\boldsymbol{\theta},\boldsymbol{\phi})$ are the patch embedding and its reconstruction. The size of the proposed feature vector is equal to the number of parameters in CVAE. The test-time, the one-hot vector of the tumor type, will be fed to the CVAE as a known parameter while the

$\mathbf{z}_{pd}$ is calculated by the classifier. For cases that there are multiple WSIs per patient, still we can feed all patches from all WSIs of the patient to the generative model, calculate the gradients, average them, and eventually obtain one embedding per "patient".

6.5 Results

We evaluate the quality of the WSI embeddings obtained by the proposed method for both search and classification tasks. The datasets we employed are diagnostic slides from The Cancer Genomic Atlas (TCGA) repository [122] and the Liver-Kidney-Stomach (LKS) immunofluorescence [83] to conduct experiments.

6.5.1 WSI search

For this experiment, we randomly selected 40% of the TCGA diagnostic WSIs as a test set and the rest for training. For both test and training WSIs, 15% of patches with 1000×1000 patch size have been selected based on the Yottixel (the same clustering method has been applied)[58]. Similar to [58], the vertical search has been applied on the test set (3,761 WSIs), and leave-one-out patient performed for searching WSIs through the same primary site. The majority of the top 3 similar cases have been used for predicting each query cancer subtype. Table. 6.1 compares F1-measure between Yottixel, Conditional GMM-based Fisher Vector (C-GMM-FV), Conditioned Deep Fisher Vector (C-Deep-FV), C-Deep-SFV, C-Deep-BFV, and C-Deep-BFV. Yottixel takes the median of minimum patch distances to calculate two WSIs dissimilarity, while C-GMM-FV and our proposed method obtain one embedding per WSI. Our proposed method improved the search F1-measure for all 29 cancer subtypes while the embeddings are binary and/or sparse. Although in almost all subtypes of two primary sites (Gynecological and Prostate/testis), C-Deep-FV performed better than other methods, almost in all cases, compact WSI embeddings obtained by gradient sparsity and quantization losses achieve even better search performance (see Table. 6.1). The compactness of the proposed embeddings leads to high efficiency for WSI search in terms of memory usage and retrieval times. Fig. 6.2 (a) shows the embedding for C-Deep-FV and C-Deep-SFV across the first 5,000 high variance dimensions given the tissue type of the given WSI out of 1,407,105 parameters of our CVAE. Considering Fig. 6.2 (a), after encouraging sparsity on the gradients, the C-Deep-SFV can represent the WSI by significantly much fewer parameters leading to compact representations. Fig. 6.2 (b) shows the effectiveness of incorporating gradient sparsity loss in reducing the l_1 norm of the loss function gradient during the epochs.

Table 6.1: F1-measure (in %) for majority-3 search through k-NN of the vertical search among 3770 test WSIs for Yottixel, C-GMM-FV, C-Deep-FV (C-D-FV), C-Deep-SFV (C-D-SFV), C-Deep-BFV (C-D-BFV), and C-Deep-SBFV (C-D-SBFV). Best F1-measure values highlighted.

Site	Subtype	n_{slides}	Yottixel	C-GMM-FV	C-D-FV	C-D-SFV	C-D-BFV	C-D-SBFV
Brain	LGG	323	86.60	85.35	92.83	93.07	93.10	93.43
	GBM	387	88.68	87.63	93.44	93.76	93.99	94.24
Endocrine	THCA	198	97.98	97.02	98.74	99.24	99.24	98.50
	ACC	93	93.68	91.01	94.62	95.08	94.62	95.13
	PCPG	70	92.53	87.14	91.97	92.95	90.64	91.97
Gastro.	ESCA	55	60.95	58.82	72.00	65.42	75.43	66.66
	COAD	174	72.62	71.95	72.72	74.93	74.44	76.42
	STAD	157	79.75	79.75	78.59	80.75	83.48	83.97
	READ	61	24.24	29.62	23.52	31.77	30.30	31.37
Gynaeco.	UCS	37	65.62	66.66	78.26	72.13	74.62	73.01
	UCEC	206	84.23	82.82	89.31	87.29	83.33	88.16
	CESC	113	71.71	76.10	86.36	81.44	70.47	81.65
	OV	42	64.78	68.42	76.74	83.95	77.33	80.95
Haematopoietic.	THYM	80	93.41	93.49	93.56	93.97	96.93	94.04
	DLBC	14	47.61	42.10	35.29	54.54	80.00	50.00
Liver, panc.	CHOL	17	32.00	38.46	40.00	48.27	41.66	32.00
	LIHC	146	94.31	93.55	92.61	93.91	94.00	94.38
	PAAD	65	93.93	91.85	92.18	94.65	93.93	93.75
Melanocytic mal	SKCM	184	96.08	98.37	98.11	98.37	98.92	98.92
	UVM	40	76.92	92.30	90.90	92.30	94.73	94.73
Prostate/testis	PRAD	176	98.56	98.00	99.42	98.55	99.14	99.14
	TGCT	112	97.79	96.88	99.11	97.81	98.67	98.66
Pulmonary	LUAD	218	67.44	74.77	77.96	79.12	74.88	79.13
	LUSC	198	67.75	70.02	71.65	72.95	72.04	76.14
	MESO	27	7.14	43.24	50.00	51.28	40.00	31.25
Urinary tract	BLCA	193	90.41	88.26	92.34	92.83	94.20	95.93
	KIRC	195	88.88	88.26	91.82	93.75	91.47	93.81
	KIRP	142	77.73	75.53	82.97	84.01	84.05	84.78
	KICH	47	79.06	84.78	86.36	89.58	89.36	87.50

The length of our proposed WSI embedding is equal to the number of trainable parameters of the generative model, which is 1,407,105. Although here we employed a relatively small generative model for models with millions of parameters, the embeddings may not be suitable for efficient WSI search. The proposed gradients sparsity solves this issue by enforcing the generative model to use a smaller number of parameters for generating samples. In other words, by imposing sparsity on gradients, one can argue that the parameters that the gradients are zero w.r.t them do not have a significant contribution to generating those samples, so they can be removed from embeddings. We have validated the effect of the sparsity loss by selecting the gradients w.r.t. a subset of some parameters that leads to high-variance gradients per tissue type. Table. 6.2 shows the feature reduction results for the same search with keeping $500, 5,000$, and $40,000$ high variance bits. Based

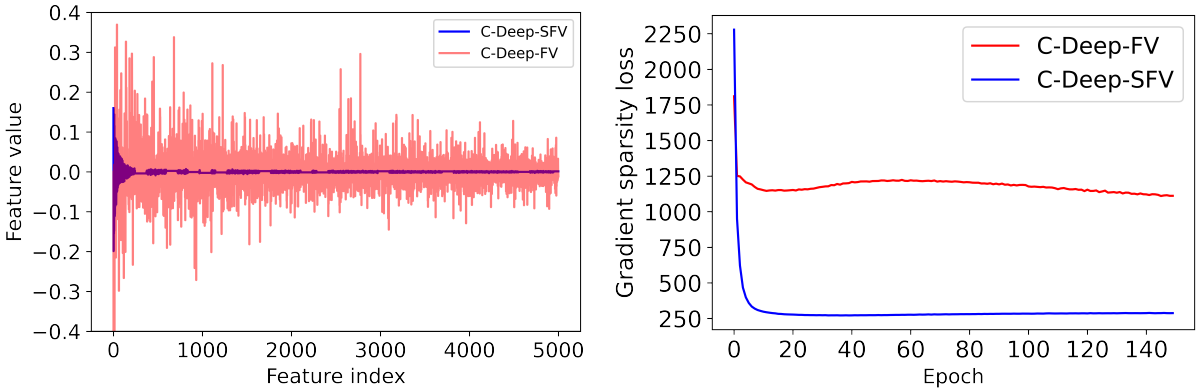


Figure 6.2: (Left). Feature values across first 5,000 high variance dimensions for a WSI using C-Deep-SFV and C-Deep-FV. (Right) Gradient sparsity loss (l_1 norm of the loss function gradient) of C-Deep-SFV and C-Deep-FV during the training epochs. According to the left figure, the C-Deep-SFV is highly sparsified compared with C-Deep-FV. The right figure indicates the effectiveness of the learning scheme in reducing l_1 norm of the loss function gradient during the epochs.

on Table. 6.2 and Fig. 6.3, keeping even 4000 high variance bits not only outperforms Yottixel and C-GMM-FV in terms of search performance but also leads to significantly faster search speed. Table. 6.2 clearly shows that the sparsity term helps the network to produce embedding with fewer but more informative bits. The right column in most subtypes outperforms the left column. Based on the *Fisher Vector Theory* this is intuitive that the gradients w.r.t generative model parameters show the contribution of each parameter to generating a sample. More precisely, by encouraging sparsity on the gradients, fewer parameters are contributing to generation; consequently, more parameters can be dropped in the final embedding.

6.5.2 WSI Classification

For this task, we validated the quality of obtained WSI embeddings on both TCGA and LKS datasets. For both cases, we trained a simple, fully connected network with two layers on top of the SFV embeddings for the purpose of WSI classification. Lung Adenocarcinoma (LUAD) and Lung Squamous Cell Carcinoma (LUSC) are two main types of non-small cell lung cancer (NSCLC) that account for 65-70% of all lung cancers [34]. Automated classification of these two main subtypes of NSCLC is a crucial step to building computerized decision support systems. The dataset for this experiment consists of all lung cancer

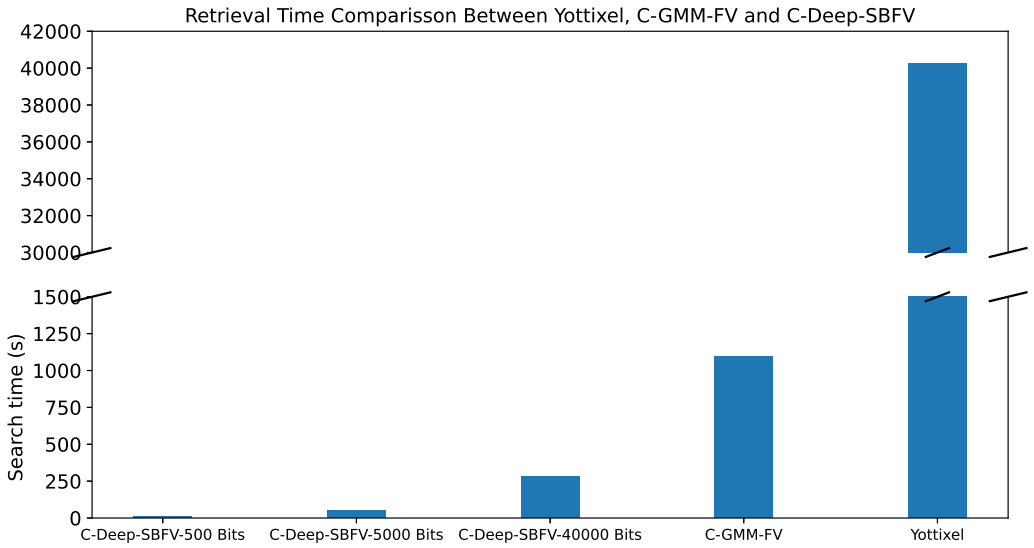


Figure 6.3: Retrieval times for the leave-one-patient-out search experiment presented in Table 6.1 for Yottixel and C-Deep-SBFV with different number of bits. Clearly, binary WSI embeddings from C-Deep-SBFV lead to faster WSI search.

slides in TCGA (comprising of 2 TB of data). We used 2,580 WSIs from TCGA public repository [122] with 774 WSIs for the test and the rest for the train. We have achieved greater than or similar to all existing research works without utilizing any expert's opinions or domain-specific techniques, as listed in Table. 6.3. However, the compared methods use their own subset of the WSIs with their own test-train split, making it difficult to benchmark them.

The Liver-Kidney-Stomach (LKS) is the other publicly available dataset that we use for validating quality of WSI embeddings. The LKS dataset contains immunofluorescence WSIs realized by authors in [83]. The dataset contains 684 WSIs from four classes Anti-Mitochondrial Antibodies (AMA), Negative (Neg), Vessel-Type Anti-Smooth Muscle Antibodies (SMA-V), and Tubule-Type Anti-Smooth Muscle Antibodies (SMA-T). This dataset contains one low-resolution image and also a set of patches per WSI. Following the same split in [83], we compared C-Deep-SFV against the proposed method in the paper Selective Objective Switch (SOS), Reinforced Dynamic Multi-Scale (RDMS), [22] and three techniques for WSI classification, namely Image-Level, Patch-Level, and Conventional Multi-Scale (see [83] for more detail). For this experiment, we only employed low-resolution images for training the backbone and then for each WSI we used is low-resolution image along with 5% of high-resolution patches for training the CVAE and

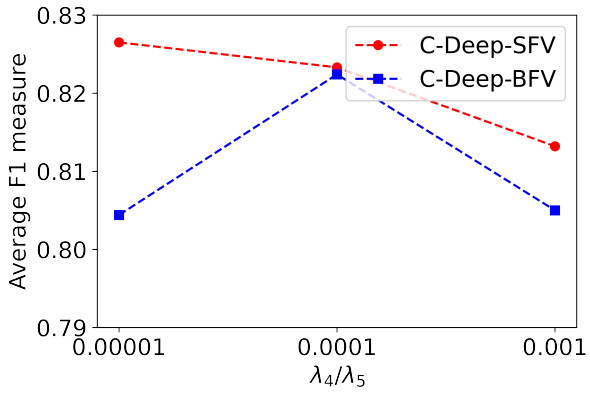


Figure 6.4: Effect of changing regularization parameter for gradient sparsity and quantization of C-Deep-SFV and C-Deep-BFV. As can be seen, the search performance of obtained WSI embeddings is fairly robust to the change of regularization parameters. See 6.5 the supplementary material for full results.

extracting the WSI embedding. The results for this experiment are presented in Table. 6.4 where our method outperforms the Image-Level, Patch-Level, Conventional Multi-Scale, RDMS methods and achieves on par result compared with SOS. It is worth mentioning that our proposed architecture has one CNN backbone, while in SOS, two networks for low and high resolutions images have been used. Besides, embeddings obtained by or method are compact and suitable for WSI search. Further extensive results on LKS dataset have been presented in the Table. 6.6, Table. 6.7,Table. 6.8, and Table. 6.9.

6.5.3 Ablation study

To study how gradient sparsity and quantization loss may affect retrieval performance, we conducted comprehensive ablation experiments. Fig. 6.4 shows the average F1 measure across all sites for C-Deep-SFV and C-Deep-BFV where values 1×10^{-5}, 1×10^{-4}, and 1×10^{-3} have been tested for both λ_4 and λ_5. The average F1 decreases with increasing the λ_4. However, we should note that this experiment has been conducted with the same number of epochs (150). Our experiments showed that by increasing the λ_4 and also the number of epochs, the average F1 measure does not decrease. To see the effect of changing regularization parameters in more detail, we refer the readers to Table. 6.5.

Table 6.2: F1-measure (in %) for majority-3 search through k-NN of the vertical search among 3,761 test data points for C-Deep-BFV (C-D-BFV), C-Deep-SBFV (C-D-SBFV) WSI embeddings with top $500, 1000$, and 5000 high variance features.

Site	Subtype	500 Bits		5000 Bits		40000 Bits	
		C-D-BFV	C-D-SBFV	C-D-BFV	C-D-SBFV	C-D-BFV	C-D-SBFV
Brain	LGG	84.48	89.02	86.27	90.93	93.21	94.06
	GBM	86.55	89.83	87.68	92.38	93.91	94.76
Endocrine	THCA	92.97	93.62	94.43	95.26	98.48	98.24
	ACC	82.87	84.37	86.91	93.19	94.68	96.25
	PCPG	73.43	75.40	69.49	84.61	91.30	92.64
Gastro.	ESCA	35.41	52.08	41.37	51.02	73.58	72.89
	COAD	64.37	70.96	68.16	70.96	77.65	74.58
	STAD	60.81	78.01	71.83	75.54	84.01	85.80
	READ	20.00	17.47	30.30	19.80	37.83	27.77
Gynaeco.	UCS	53.33	61.53	63.49	60.60	81.81	67.60
	UCEC	82.01	81.69	82.77	84.21	88.53	86.99
	CESC	64.03	64.48	63.73	73.87	82.24	82.35
	OV	64.93	52.17	74.66	70.42	82.50	83.95
Haematopoietic.	THYM	91.76	94.04	92.39	91.56	94.54	91.01
	DLBC	22.22	50.00	23.52	36.36	60.86	28.57
Liver, panc.	CHOL	25.00	9.52	20.00	23.07	34.78	38.46
	LIHC	81.36	84.24	85.09	85.34	92.30	93.95
	PAAD	58.18	66.12	68.42	74.79	98.29	90.90
Melanocytic mal	SKCM	96.02	94.31	97.57	95.58	97.57	94.91
	UVM	78.87	79.16	88.31	81.39	88.31	80.85
Prostate/testis	PRAD	91.37	96.04	94.05	95.18	98.29	98.85
	TGCT	86.84	93.69	90.58	92.37	97.32	98.24
Pulmonary	LUAD	62.30	67.89	69.27	68.62	76.64	75.71
	LUSC	61.65	62.31	61.08	64.51	72.72	72.53
	MESO	12.90	50.90	25.80	50.00	52.63	65.11
Urinary tract	BLCA	75.88	82.95	81.42	82.84	96.12	94.84
	KIRC	77.47	82.46	77.87	85.78	93.93	91.83
	KIRP	60.00	62.94	60.60	63.67	88.64	85.71
	KICH	42.25	52.27	50.00	54.34	89.79	87.23

6.5.4 Does Gradient Sparsity Improve Generalization?

In this Chapter, we proposed a new loss function, namely gradient sparsity that encourages sparsity on the gradients to obtain sparse permutation-invariant representations. Although, we introduced this loss function to obtain sparse permutation-invariant representations, during the experiments we observed that models trained using this loss function achieve better generalization. For instance, Looking more closely at Table 6.1, an interesting property of learning with sparse gradients can be discovered. By comparing the results of C-Deep-SFV (trained with sparse) and C-Deep-FV (trained with no constraint on gradients), one can easily observe for the majority of cases that the number of slides is too low and the network is prone to overfitting, the network with trained with sparse gradients achieves

Table 6.3: Comparison of WSI classification methods against Deep-SFV on lung cancer classification via transfer learning.

Method	Accuracy (in %)
Yu et al. [128]	75
Khosravi et al. [60]	83
Kalra & Adnan et al. [57]	84
Hemati et al. [41]	86
Deep-SFV (Ours)	**88**

Table 6.4: Comparison of different WSI classification methods against Deep-SFV on LKS dataset. For more detailed experiments see the 6.6, 6.7,6.8, and 6.9 in supplementary material.

Method	Accuracy (in %)
Image-Level	81.95
Patch-Level	69.27
Multi-Scale	85.37
RDMS [22]	88.78
SOS [83]	**90.73**
Deep-SFV (Ours)	**90.73**

superior performance. Further, in Table 6.2, we can see that including gradient sparsity loss improves the performance of the model on the test set. To further explore this, we conduct another experiment using a simple CNN with three convolutional layers on CIFAR-10 [67] dataset. Naming the the gradient sparsity regularization parameter as λ, this simple CNN is trained in two different settings i.e., with gradient sparsity $\lambda = 0.0005$ and without gradient sparsity losses $\lambda = 0$. Surprisingly, as can be seen in Fig. 6.5, both test accuracy and loss plots confirm that gradient sparsity can improve the generalization of the neural network.

Another interesting way to measure quantify the generalization ability of a machine learning model is through lens of Membership Inference Attack (MIA) [98]. In short, MIA on machine learning models employ the model parameters and/or parameters and aim to identify whether a data sample was used to train the target machine learning model or not. As a result, if the machine learning model generalize well or equivalently behave more similar on the test and train sets, it will be more robust against this attack. we employed TensorFlow privacy to run attacks on the two mentioned models trained on

Table 6.5: Ablation study on λ_4 and λ_5 regularization parameters based on Majority-3 search through k-NN of the vertical search among 3761 test WSIs.

		F1-measure (in %)					
		$\lambda_4 = 10^{-5}$	$\lambda_5 = 10^{-5}$	$\lambda_4 = 10^{-4}$	$\lambda_5 = 10^{-4}$	$\lambda_4 = 10^{-3}$	$\lambda_5 = 10^{-3}$
Site	Subtype	C-D-SFV	C-D-BFV	C-D-SFV	C-D-BFV	C-D-SFV	C-D-BFV
Brain	LGG	90.88	92.87	93.07	93.10	92.72	93.80
	GBM	91.67	93.91	93.76	93.99	93.53	94.70
Endocrine	THCA	98.48	97.29	99.24	99.24	98.74	98.75
	ACC	92.55	91.30	95.08	94.62	95.13	94.62
	PCPG	88.40	90.07	92.95	90.64	91.42	90.37
Gastro.	ESCA	69.42	67.76	65.42	75.43	77.04	70.17
	COAD	73.38	75.97	74.93	74.44	76.21	74.01
	STAD	84.07	79.23	80.75	83.48	84.45	84.01
	READ	25.49	33.33	31.77	30.30	30.18	29.90
Gynaeco.	UCS	81.15	74.28	72.13	74.62	73.23	71.64
	UCEC	89.97	86.18	87.29	83.33	89.42	85.58
	CESC	82.72	84.30	81.44	70.47	86.84	74.17
	OV	79.48	73.68	83.95	77.33	79.01	81.01
Haematopoietic.	THYM	94.54	93.49	93.97	96.93	93.56	95.23
	DLBC	60.86	42.10	54.54	80.00	35.29	60.00
Liver, panc.	CHOL	41.66	25.00	48.27	41.66	14.81	16.66
	LIHC	94.91	92.66	93.91	94.00	88.66	92.35
	PAAD	91.97	90.90	94.65	93.93	86.82	93.12
Melanocytic malig	SKCM	98.13	98.37	98.37	98.92	97.23	97.86
	UVM	90.41	92.30	92.30	94.73	88.37	89.18
Prostate/testis	PRAD	99.42	99.14	98.55	99.14	98.56	98.85
	TGCT	99.11	98.66	97.81	98.67	98.56	98.23
Pulmonary	LUAD	77.84	75.84	79.12	74.88	76.95	78.70
	LUSC	72.67	74.20	72.95	72.04	71.72	72.20
	MESO	63.63	50.00	51.28	39.99	72.72	50.00
Urinary tract	BLCA	94.20	94.62	92.83	94.20	96.14	94.02
	KIRC	92.73	90.90	93.75	91.47	93.50	89.35
	KIRP	86.69	83.94	84.01	84.05	87.63	78.41
	KICH	90.52	90.32	89.58	89.36	90.72	87.64

Table 6.6: Comparison of different WSI classification methods against Deep-SFV on LKS dataset for SMA-T class.

Method	F1 (in %)	PR (in %)	RE (in %)
Image-Level	16.67	100.00	9.09
Patch-Level	00.00	00.00	00.00
Multi-Scale	47.06	66.67	36.36
RDMS [22]	55.56	71.43	45.45
SOS [83]	**70.00**	71.43	**63.64**
Deep-SFV (Ours)	66.67	**85.71**	54.55

Table 6.7: Comparison of different WSI classification methods against Deep-SFV on LKS dataset for Negative class.

Method	F1 (in %)	PR (in %)	RE (in %)
Image-Level	88.00	81.15	96.12
Patch-Level	79.67	68.53	95.15
Multi-Scale	90.83	86.09	96.12
RDMS [22]	93.00	95.88	90.29
SOS [83]	**94.06**	**95.96**	92.23
Deep-SFV (Ours)	93.84	91.67	**96.12**

Table 6.8: Comparison of different WSI classification methods against Deep-SFV on LKS dataset for AMA class.

Method	F1 (in %)	PR (in %)	RE (in %)
Image-Level	89.89	90.90	88.89
Patch-Level	84.71	90.00	80.00
Multi-Scale	87.06	92.50	82.22
RDMS [22]	91.49	87.76	**95.56**
SOS [83]	93.48	91.49	**95.56**
Deep-SFV (Ours)	**94.51**	**93.48**	**95.56**

CIFAR-10 using Adam ($\lambda = 0$) and Adam + gradient sparsity ($\lambda = 0.0005$). Fig. 6.6 shows the receiver operating characteristic curve for the MIA against models trained with two mentioned optimizers. Clearly, the model trained with Adam + gradient sparsity shows superior robustness against MIA compared with Adam and DP-Adam.

The sting point here is the fact that unlike differential private optimizers that improve the model generalization at cost of sacrificing model utility (accuracy), here based on the Figs. 6.5, 6.6, both utility and privacy robustness has been improved.

6.5.5 Why Gradient Sparsity Achieves Better Results?

The observations that gradient sparsity loss improves the generalization motivated us to explore possible reasons behind generalization improvements caused by gradient sparsity. In the following, we provide our intuitive and theoretical perspectives.

Table 6.9: Comparison of different WSI classification methods against Deep-SFV on LKS dataset for SMA-V class.

Method	F1 (in %)	PR (in %)	RE (in %)
Image-Level	66.67	73.68	60.87
Patch-Level	23.53	36.36	17.39
Multi-Scale	77.78	79.55	76.09
RDMS [22]	83.67	78.85	**89.13**
SOS [83]	**85.42**	82.00	**89.13**
Deep-SFV (Ours)	84.44	**86.36**	82.61

Intuitive Explanation

In the following we provide some of our intuitions that can explain generalization improvements caused by gradient sparsity.

1. Increasing the sparsity in gradients leads to many zero values for gradients. Considering the learning rule for updating the network parameters, zero gradients lead to no updates in some specific weights. This can be seen as the network is learning no to learn some patterns in data (hopefully the noise) which may reduce overfitting.

2. Encouraging sparsity in gradients leads to many zero values in gradients given batch of data. One can see this as a dropout like operation. More precisely this can be seen as gradient dropout.

3. Encouraging sparsity in gradients can supply avoids the over training problem.

4. From Fisher Kernel theory perspective, when gradients with respect to more parameters are zero, this means that the generative model is using smaller number of parameters to generate samples. As a result, when we encourage sparsity in the gradients, we are forcing the neural net to use smaller portion of its power to generate samples.

Theoretical Explanation

We explored possible reasons of superior generalization of models trained with sparse gradients and we realized that through loss landscape we can achieve this. More precisely, there is a large body of works that show loss landscape with flatter local minimums are

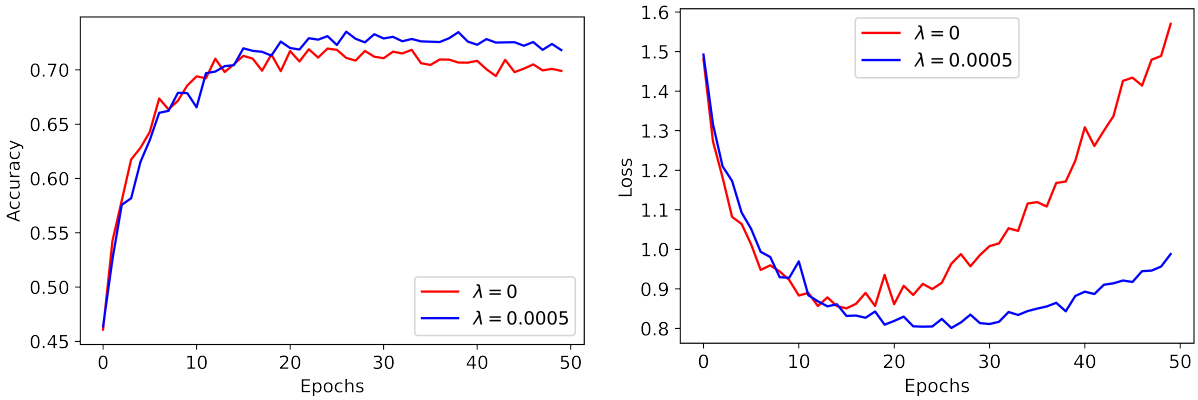


Figure 6.5: (Left). The CIFAR-10 test accuracy during the epochs for Adam ($\lambda = 0$) and Adam + gradient sparsity ($\lambda = 0.0005$). (Right) The CIFAR-10 test loss during the training epochs for Adam ($\lambda = 0$) and Adam + gradient sparsity ($\lambda = 0.0005$). Clearly gradient sparsity improves generalization.

associated with better generalization [59, 28]. In other words deep neural networks that their local minimum has less sharpness will generalize better. In the following, first we define the sharpness of the loss function and then we show that encouraging sparsity in the gradients of loss function reduces its sharpness.

$$sharpness := \|\mathcal{L}(\mathbb{W} + \mathcal{E}, \mathbf{X}) - \mathcal{L}(\mathbb{W}, \mathbf{X})\|, \tag{6.16}$$

where $\mathcal{E}$ is a small perturbation vector with the length same as the number model of parameters, $\mathbb{W}$ is the set of deep network parameters of all layers, $\mathbf{X}$ input data and $\mathcal{L}(\mathbb{W}, \mathbf{X})$ is the loss value.

In order to show that gradient sparsity can reduce the sharpness, first we show that l_1 norm of the gradient of $\mathcal{L}(\mathbb{W}, \mathbf{X})$ i.e., $\|\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X})\|$ is an upper bound for loss sharpness in Eq. 6.16. To show this, using the first order Taylor Approximation we can write

$$\mathcal{L}(\mathbb{W} + \mathcal{E}, \mathbf{X}) \approx \mathcal{L}(\mathbb{W}, \mathbf{X}) + \mathcal{E}^T.\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X}). \tag{6.17}$$

Now, one can show that [1]

$$\underset{\|\mathcal{E}\|_\infty \leq \epsilon}{\arg\max} \quad \mathcal{E}^T.\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X}) = \epsilon.\|\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X})\|_1, \tag{6.18}$$

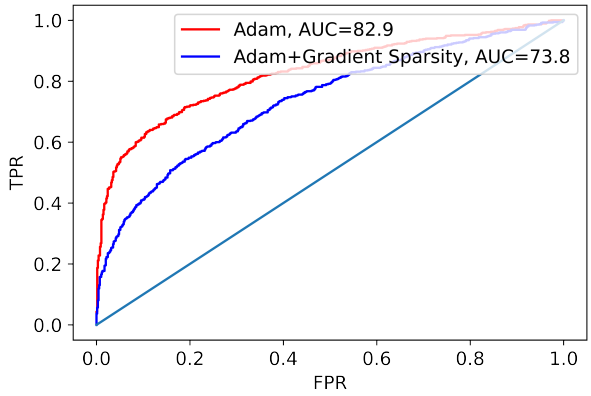


Figure 6.6: Membership inference attack performance when the model is trained with Adam ($\lambda = 0$) and Adam + gradient sparsity ($\lambda = 0.0005$). The CNN model trained with gradient sparsity is more robust to the membership inference attack.

which means that the maximum of $\mathcal{E}^T.\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X})$ for a l_∞ bounded perturbation i.e., $\|\mathcal{E}\|_\infty \leq \epsilon$ is equal to the l_1 norm of the gradient of $\mathcal{L}(\mathbb{W}, \mathbf{X})$ i.e., $\|\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X})\|_1$. Considering Eqs. 6.17 and 6.18 we can conclude that

$$\|\mathcal{L}(\mathbb{W} + \mathcal{E}, \mathbf{X}) - \mathcal{L}(\mathbb{W}, \mathbf{X})\| \leq \epsilon.\|\nabla_{\mathbb{W}}\mathcal{L}(\mathbb{W}, \mathbf{X})\|_1. \tag{6.19}$$

The Eq. 6.19 suggests that reducing l_1 norm of the gradients reduce loss function sharpness and as a result improve the generalization. To Validate this, we also monitored the sharpness during the training i.e., $\|\mathcal{L}(\mathbb{W} + \epsilon, \mathbf{X}) - \mathcal{L}(\mathbb{W}, \mathbf{X})\|_1$, for C-Deep-SFV ($\lambda_4 = 10^{-4}$) and C-Deep-FV ($\lambda_4 = 0$). Fig. 6.7 shows the sharpness for model trained with and without gradient sparsity. Clearly, gradient sparsity reduces the loss landscape sharpness significantly.

Another measure of generalization that is also related to loss sharpness is the entropy of predictions or equivalently the confidence of the network predictions [134, 88]. More precisely, networks that their output has higher entropy or equivalently there is less confidence in predictions, enjoy better generalization. Considering this relation ship between loss sharpness and confidence of predictions in Figs. 6.8 and 6.9 we compare the average of predictions over all data points for (test and train split in the CIFAR-10 dataset) for an example case when the ground-truth class label is 2. It is evident that the model trained with gradient sparsity has predictions with less confidence (higher entropy).

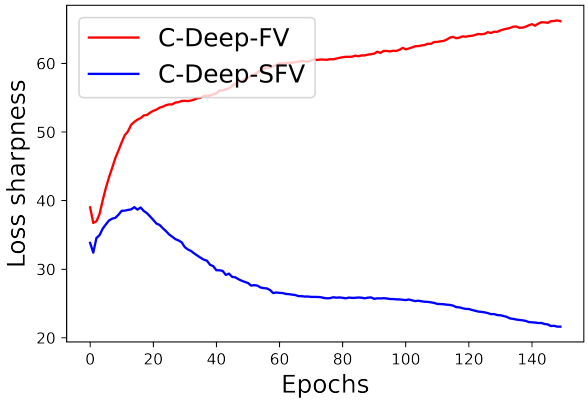


Figure 6.7: Loss sharpness $\|\mathcal{L}(\mathbb{W}+\epsilon,\mathbf{X})-\mathcal{L}(\mathbb{W},\mathbf{X})\|$ for C-Deep-SFV ($\lambda_4 = 10^{-4}$) and C-Deep-FV ($\lambda_4 = 0$) through epochs.

6.6 Conclusions

We proposed a new framework based on deep conditional generative modelling and *Fisher Vector Theory* for compact WSI representation. Unlike the common practice for WSI representation, i.e., MIL scheme, the training for the proposed method is instance-based, and as a result, GPU memory usage is the same as conventional training. Furthermore, we introduced new loss functions, gradient sparsity and gradient quantization for learning sparse and binary permutation-invariant representations, namely C-Deep-SFV and C-Deep-BFV, suitable for efficient WSI retrievals. We showed that gradient sparsity loss function pushes the generative model to use parameters for generating a sample, and as a result, one can reduce the dimensionality of the WSI embeddings and still achieve a good performance. The WSIs representations were validated on the largest public archive of WSIs, The TCGA WSIs and also the LKS dataset for both WSI search and classification tasks. The proposed method outperforms *Yottixel* a recent search engine for histopathology images and *GMM-based Fisher Vector*. Furthermore, we also achieved competitive results against state-of-the-art in WSI classifications on both lung and LKS public benchmark datasets.

Although the gradient sparsity loss function was proposed to achieve sparse permutation-invariant WSI representations, we observed that encouraging sparsity on the gradients improves the model generalization. Following this observation, additional experiments on CIFAR-10 dataset were conducted to explore effect of gradient sparsity on the generalization. The test loss and accuracy plots showed that gradient sparsity improves generalization. Besides, we employed the membership inference attack as another measure of

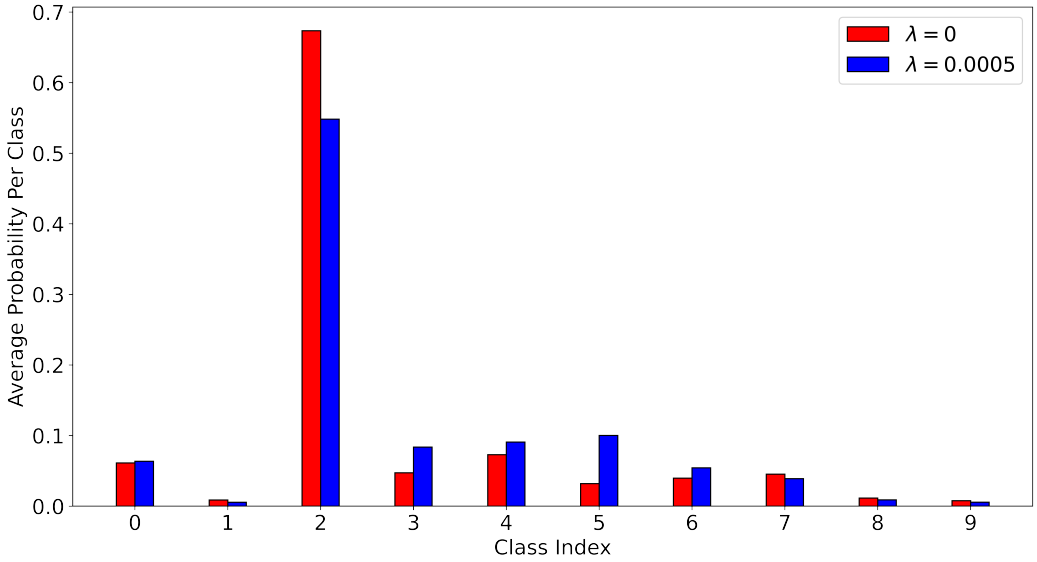


Figure 6.8: Average of predictions for train split over class 2 for Adam $\lambda = 0$ and Adam+gradient sparsity $\lambda = 0.0005$.

generalization where the results indicated that gradient sparsity improves the generalization. We also presented some intuitive and theoretical investigations that explains possible reasons why gradient sparsity improves the generalization. More precisely, we showed that gradient sparsity reduces the loss landscape sharpness and as a result improves the generalization.

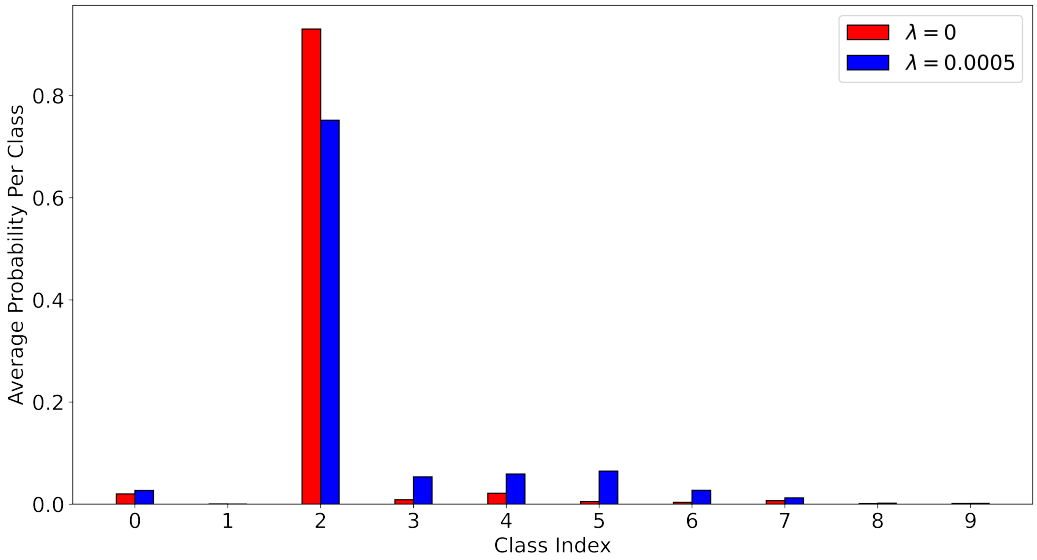


Figure 6.9: Average of predictions for train split over class 2 for Adam $\lambda = 0$ and Adam+gradient sparsity $\lambda = 0.0005$.

Chapter 7

Summary and Conclusions

Computer-assisted Diagnosis (CAD) is an active research area in the domain of medical image analysis. The main motivation is to help the physicians by automating some of the technical tasks performed by them. CAD is basically a learning algorithm (for example WSI classification) and this is expected to reduce inter-observer and support a decision taken by physicians. CAD is popular in different image modalities including X-ray, MRI, Endoscopy, ultrasound, and digital pathology. For digital pathology, CAD systems are being used in different tasks including detection or segmentation of tumor are in a WSI [35], nuclei density estimation [85], and cancer staging [95].

Another scheme which we are specifically interested in is fast and efficient content-based retrieval of WSIs/patches were given a WSI/patch as an input, this is expected that the search engine returns and ranks the most similar WSI/patch by capturing semantic similarity to the other WSIs/patches in a large archive of images. Content-based retrieval systems can be beneficial in different ways, including teleconsultation, workload efficiency, collaborations, improving diagnostic accuracy, virtual education, and research [113, 115]. More precisely, Pathologists examine biopsy tissues to detect the tumours and to investigate their characteristics in order to evaluate tumour aggressiveness which is s a complex task that needs many years of experience, and sub-speciality expertise [55]. However, having a fast and reliable retrieval system can act as additional knowledge that shares the experience of many other pathologists that investigated similar cases in the past. This will lead to a more reliable and objective diagnosis with less inter-observer variability.

Although the content-based retrieval of medical images has been studied [94], there are some limitations with traditional systems. On one hand, many of these systems are based on old-school computer vision techniques to represent the WSI while these representations

are sub-optimal compared with deep features. On the other hand, due to the large size of WSIs, this is not straightforward to obtain compact (binary/sparse) deep embeddings for WSIs. Considering these difficulties, the objective of this thesis was to propose a deep neural network for learning WSI representations ideal for fast and memory efficient WSI search in a gigantic archive. Image search in extremely large archives is challenging even for regular-size images. However, in histopathology whole slide images, due to their gigantic size, the problem becomes even more difficult. This is because each WSI is translated to thousands of patches and as a result, in the patch search problem the size of the archive is typically thousands of times larger and as a result we are faced with a bottleneck in memory usage and search speed. Further, in WSI search problem, we end up with a bag of embedding representations which makes it unclear to employ regular image retrieval algorithms. To this end, in chapters 3 and 4, we proposed efficient spectral hashing and non-rigid quantization methods for learning to hash or learning similarity preserving binary representations suitable for large patch archives. Considering the fact that the developed methods were not applicable to WSIs, in chapter 5 we proposed to employ Deep Sets as a simple permutation invariant neural network for end-to-end WSI representation learning. Although using this multi-instance learning scheme we were able to obtain WSI embeddings, there were two issues with the proposed approach. First, the obtained embeddings were in euclidean space which cannot be used for search in a large archive due to search speed and memory bottleneck issues. Second, due to keeping multiple bags of patches in memory during the training, the proposed scheme needed considerable GPU memory during the training. Motivated by these limitations, in chapter 6, we proposed a framework based on deep generative models and Fisher Vector theory for learning binary and sparse permutation invariant representations. We showed that using the proposed scheme, one can obtain compact binary and sparse WSI embeddings that outperform recently developed WSI search engine Yotixxel both in terms of search speed and retrieval accuracy. Further, considering that the proposed method is a dictionary learning-based approach, and in the training phase of the generative model (creating the dictionary) there is no need to keep bags of patches in memory, the GPU memory usage was also significantly reduced.

In the following, we discuss our contributions, limitations, and possible future directions of the proposed methods in each chapter in more detail.

7.1 Contributions, Limitations, and Future Works

7.1.1 ESH

Contributions in chronological order

In this algorithm which we proposed in chapter 3, we proposed a novel formulation for spectral hashing which achieves highly competitive performance compared with most recent methods and at the same time, enjoys very low time complexity. The proposed projected gradient method is highly efficient for three reasons. Firstly, the formulation for ESH transforms the decision variable with dimensionality of $n \times k$ to $d \times k$ where $d \ll n$. Secondly, the affinity matrix which is $n \times n$ in the spectral formulation has been removed, and instead, a $d \times d$ matrix $\mathbf{S}$ plays a similar role. Finally, and more importantly, unlike other graph hashing schemes, the proposed formulation achieves high quality binary codes without adding any additional decision variables to the problem and as a result it is non-alternating. We applied two different optimization techniques i.e., projected gradient and manifold optimization to obtain a solution. Using extensive experiments on four public datasets, we showed that the proposed method either outperforms or achieves highly competitive results compared with recent methods and offering low complexity at the same time.

Limitations

The first limitation of this work is the difficult optimization problem which is inherent to all spectral hashing based formulations. The second downside of the proposed method is the fact that it cannot be applied to WSIs directly. More precisely, we can only obtain binary representations for patches and not WSIs.

Future Works

For future work, we plan to update the affinity matrix along with the proposed loss function, which needs the use of an end-to-end training framework where feature learning is achieved through training. To conduct feature learning, we may need to employ a reconstruction loss with the proposed loss function in Eq. 6.7. We believe that such a scheme can significantly improve the performance of the proposed method. Furthermore, another interesting path for future studies is to apply this more efficient non-alternating hashing scheme instead of the more complex alternating algorithm employed by [47] for the network quantization problem.

7.1.2 NRQ

Contributions

In chapter 4, we introduced Non-rigid Quantization NRQ which was built on top of ITQ (iterative quantization) [33] idea. In ITQ, projection and rotation matrices are learned in two separate steps which makes it sub-optimal. Furthermore, a rigid transformation may not be powerful enough for reducing quantization to the ultimate limit. Motivated by these limitations, we proposed an algorithm to reduce both dimensionality and quantization loss simultaneously. We also employed a non-rigid transformation to push for quantization beyond rotation. Employing non-rigid transformations is generally against intuition. It does not preserve the neighborhood of data (and all these efforts for reducing quantization error is to preserve more neighborhood after binarization). However, we showed that corrupting neighborhood in favor of reducing quantization eventually leads to better binary codes. An efficient nested coordinate descent algorithm was employed to update all three matrices. The results on five public datasets totaling almost half a million images showed that the proposed method outperforms the state-of-art linear hashing methods.

Limitations

Although the NRQ approach leads to slightly easier optimization problem, but still the solution to the problem is not straightforward and a coordinate descent approach was used to obtain a solution. Further, similar to the ESH method, the NRQ algorithm cannot be directly applied to WSIs.

Future Works

As the future work, we plan to extend the idea of binary representation learning using the rigid and non-rigid transformations to the deep architectures. In this framework, while the non-rigid transformations can be realized using regularization terms similar to the soft orthogonality in Eq. 6.3. For the rigid transformation, one can project the gradient on to orthogonal feasible set using the projection operation proposed in [84] to update the rotation matrix. Besides, it has been recently demonstrated that hashing can be used in the network quantization task [30]. This would be interesting to see how the proposed quantization method performs for network quantization.

7.1.3 CNN-Deep Sets

Contributions

In chapter 5, inspired by bag of patches (set) representation per WSI, we employed Deep Sets along with a CNN for end-to-end WSI representation learning. We introduced two reshape layers to connect the CNN backbone with Deep Sets while avoiding batch size one. We proposed a novel multi-label training scheme where the primary site prediction is used to help the network in primary diagnosis labels prediction. To this end, we used the law of total probability to capture the primary site predicted probability for obtaining probability of primary diagnosis. We validated the proposed topology in a transfer learning scheme for WSI search. We showed that the proposed architecture can obtain WSI embeddings leading to comparable retrieval performance compared with Yottixel while reducing the retrieval time significantly. We also applied the proposed scheme to lung classifcation task and achieved competitive results compared with the state-of-art. Although the gradient sparsity loss function was proposed to achieve sparse permutation-invariant WSI representations, we observed that encouraging sparsity on the gradients improves the model generalization. Following this observation, additional experiments on CIFAR-10 dataset were conducted to explore effect of gradient sparsity on the generalization. The test loss and accuracy plots showed that gradient sparsity improves generalization. Besides, we employed the membership inference attackMIA as another measure of generalization where the results indicated that gradient sparsity improves the generalization. We also presented some intuitive and theoretical investigations that explains possible reasons why gradient sparsity improves the generalization. More precisely, showed that gradient sparsity reduces the loss landscape sharpness and as a result improves the generalization.

Limitations

One limitation of the proposed method is number of small patches (here 40) that has been used to model each WSI. Even for keeping 40 patches per WSI and batch size of 16, we employed four Tesla V 100 GPUs to keep this data in the memory which shows the memory bottleneck limitation of this work. To get a better idea of the memory consumption in this approach, considering batch size of 16, extracting 40 patches per each WSI (set size= 40), and resizing patches from 1000×1000 to 224×224, the input tensor of our network has the shape (16,40,224,224,3). This tensor will be reshaped into a 4-dimensional tensor with shape (640,224,224,3) that means the task can be seen as a regular instance-based training while the batch size is 640 which obviously is not feasible with current GPUs. The other

limitation of the proposed approach is the fact that the obtained WSI embeddings are in Euclidean space and as a result they are not ideal for fast and memory efficient WSI search within gigantic archives.

Future Works

In future works we would to sample more patches per WSI and see how it improve the performance of the model. Further, here we started from the ImageNet wights which may not be the best possible option for weight initialization. Given the recent success of self supervised methods, it would be interesting to explore how the performance changes if we start from a set of weights that have been obtained from self supervised training.

Considering the fact that removing position embeddings from transformer makes them permutation invariant, an interesting future research direction is to replace permutation invariant unit Deep Sets with transformer models. Although given the representation learning capability of transformers this approach probably will lead to WSI embeddings with higher quality, the memory bottleneck during end-to-end not only exists but also it becomes more severe.

7.1.4 Conditioned-Sparse and Binary Fisher Vector

Contributions

In chapter 6, we proposed a new framework based on deep conditional generative modelling and *Fisher Vector Theory* for compact WSI representation. Unlike the common practice for WSI representation, i.e., MIL scheme, the training for the proposed method is instance-based, and as a result, GPU memory usage is the same as conventional training. Furthermore, we introduced new loss functions, gradient sparsity and gradient quantization for learning sparse and binary permutation-invariant representations, namely C-Deep-SFV and C-Deep-BFV, suitable for efficient WSI retrievals. We showed that gradient sparsity loss function pushes the generative model to use parameters for generating a sample, and as a result, one can reduce the dimensionality of the WSI embeddings and still achieve a good performance. The WSIs representations were validated on the largest public archive of WSIs, The TCGA WSIs and also the LKS dataset for both WSI search and classification tasks. The proposed method outperforms *Yottixel* a recent search engine for histopathology images and *GMM-based Fisher Vector*. Furthermore, we also achieved competitive results against state-of-the-art in WSI classifications on both lung and LKS public benchmark datasets. Although the gradient sparsity loss function was proposed to achieve sparse

permutation-invariant WSI representations, we observed that encouraging sparsity on the gradients improves the model generalization. Following this observation, additional experiments on CIFAR-10 dataset were conducted to explore effect of gradient sparsity on the generalization. The test loss and accuracy plots showed that gradient sparsity improves generalization. Besides, we employed the membership inference attackMIA as another measure of generalization where the results indicated that gradient sparsity improves the generalization. We also presented some intuitive and theoretical investigations that explains possible reasons why gradient sparsity improves the generalization. More precisely, we showed that gradient sparsity reduces the loss landscape sharpness and as a result improves the generalization.

Limitations

The proposed method as a scheme for learning binary and sparse permutation invariant representations showed great potential for WSI search. However as a technique that is designed for clinical use cases, this is important to provide explainable representations.

Future Works

In our proposed method we employed VAEs along with Fisher Vector theory for learning compact permutation invariant WSI representations. An interesting future research direction for the proposed learning scheme is to employ other deep generative models specially generative Adversarial Networks (GANs) for learning binary and/or sparse WSI representations. Note the Fisher Vector theory can formulated on top of any generative models and considering the fact that GANs can be better generative models compared with VAEs in some senses, one can expect that GANs-based Fisher Vectors obtain better search performance.

References

[1] Milad Alizadeh, Arash Behboodi, Mart van Baalen, Christos Louizos, Tijmen Blankevoort, and Max Welling. Gradient l_1 regularization for quantization robustness. *arXiv preprint arXiv:2002.07520*, 2020.

[2] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 459–468, Oct 2006.

[3] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? In *Advances in Neural Information Processing Systems*, pages 4261–4271, 2018.

[4] Babak Ehteshami Bejnordi, Guido Zuidhof, Maschenka Balkenhol, Meyke Hermsen, Peter Bult, Bram van Ginneken, Nico Karssemeijer, Geert Litjens, and Jeroen van der Laak. Context-aware stacked convolutional neural networks for classification of breast carcinomas in whole-slide histopathology images. *Journal of Medical Imaging*, 4(4):044504, 2017.

[5] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020.

[6] Fatih Cakir, Kun He, Sarah Adel Bargal, and Stan Sclaroff. Mihash: Online hashing with mutual information. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 437–445, 2017.

[7] Fatih Cakir, Kun He, and Stan Sclaroff. Hashing with binary matrix pursuit. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 332–348, 2018.

[8] Miguel A Carreira-Perpinán and Ramin Raziperchikolaei. Hashing with binary autoencoders. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 557–566, 2015.

[9] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[10] Wafa Chenni, Habib Herbi, Morteza Babaie, and Hamid R Tizhoosh. Patch clustering for representation of histopathology images. In *European Congress on Digital Pathology*, pages 28–37. Springer, 2019.

[11] Philip Chikontwe, Meejeong Kim, Soo Jeong Nam, Heounjeong Go, and Sang Hyun Park. Multiple instance learning with center embeddings for histopathology classification. In Anne L. Martel, Purang Abolmaesumi, Danail Stoyanov, Diana Mateus, Maria A. Zuluaga, S. Kevin Zhou, Daniel Racoceanu, and Leo Joskowicz, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, pages 519–528, Cham, 2020. Springer International Publishing.

[12] Chaoqun Chu, Dahan Gong, Kai Chen, Yuchen Guo, Jungong Han, and Guiguang Ding. Optimized projection for hashing. *Pattern Recognition Letters*, 117:169–178, 2019.

[13] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval*, pages 1–9, 2009.

[14] Lee Ad Cooper, Elizabeth G Demicco, Joel H Saltz, Reid T Powell, Arvind Rao, and Alexander J Lazar. Pancancer insights from the cancer genome atlas: the pathologist's perspective. *The Journal of pathology*, 244(5):512–524, 2018.

[15] Nicolas Coudray, Paolo Santiago Ocampo, Theodore Sakellaropoulos, Navneet Narula, Matija Snuderl, David Fenyö, Andre L Moreira, Narges Razavian, and Aristotelis Tsirigos. Classification and mutation prediction from non–small cell lung cancer histopathology images using deep learning. *Nature medicine*, 24(10):1559–1567, 2018.

[16] Gabriela Csurka and Florent Perronnin. Fisher vectors: Beyond bag-of-visual-words image representations. In *International Conference on Computer Vision, Imaging and Computer Graphics*, pages 28–42. Springer, 2010.

[17] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[18] Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1-2):31–71, 1997.

[19] Neofytos Dimitriou, Ognjen Arandjelović, and Peter D Caie. Deep learning for whole slide image analysis: an overview. *Frontiers in medicine*, 6:264, 2019.

[20] Thanh-Toan Do, Anh-Dzung Doan, and Ngai-Man Cheung. Learning to hash with binary deep neural network. In *European Conference on Computer Vision*, pages 219–234. Springer, 2016.

[21] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[22] Nanqing Dong, Michael Kampffmeyer, Xiaodan Liang, Zeya Wang, Wei Dai, and Eric Xing. Reinforced auto-zoom net: towards accurate and fast breast cancer segmentation in whole-slide images. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 317–325. Springer, 2018.

[23] Harris Drucker and Yann Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

[24] Sepehr Eghbali and Ladan Tahvildari. Deep spherical quantization for image search. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[25] Venice Erin Liong, Jiwen Lu, Gang Wang, Pierre Moulin, and Jie Zhou. Deep hashing for compact binary codes learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2475–2483, 2015.

[26] Jason Farquhar, Sandor Szedmak, Hongying Meng, and John Shawe-Taylor. Improving" bag-of-keypoints" image categorisation: Generative models and pdf-kernels. 2005.

[27] Kevin Faust, Quin Xie, Dominick Han, Kartikay Goyle, Zoya Volynskaya, Ugljesa Djuric, and Phedias Diamandis. Visualizing histopathologic deep learning classification and anomaly detection using nonlinear feature space dimensionality reduction. *BMC bioinformatics*, 19(1):173, 2018.

[28] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[29] Lianli Gao, Jingkuan Song, Fuhao Zou, Dongxiang Zhang, and Jie Shao. Scalable multimedia retrieval by deep learning hashing with relative similarity learning. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 903–906, 2015.

[30] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021.

[31] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529, 1999.

[32] Yunchao Gong, Sanjiv Kumar, Vishal Verma, and Svetlana Lazebnik. Angular quantization-based binary codes for fast similarity search. In *Advances in neural information processing systems*, pages 1196–1204, 2012.

[33] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 35(12), 2012.

[34] Simon Graham, Muhammad Shaban, Talha Qaiser, Navid Alemi Koohbanani, Syed Ali Khurram, and Nasir Rajpoot. Classification of lung cancer histology images using patch-level summary statistics. In *Medical Imaging 2018: Digital Pathology*, volume 10581, page 1058119. International Society for Optics and Photonics, 2018.

[35] Zichao Guo, Hong Liu, Haomiao Ni, Xiangdong Wang, Mingming Su, Wei Guo, Kuansong Wang, Taijiao Jiang, and Yueliang Qian. A fast and refined cancer regions segmentation framework in whole-slide breast pathological images. *Scientific reports*, 9(1):882, 2019.

[36] Kaiming He, Fang Wen, and Jian Sun. K-means hashing: An affinity-preserving quantization method for learning binary compact codes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2938–2945, 2013.

[37] Xiangyu He, Peisong Wang, and Jian Cheng. K-nearest neighbors hashing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[38] Xiaofei He, Deng Cai, Shuicheng Yan, and Hong-Jiang Zhang. Neighborhood preserving embedding. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1208–1213. IEEE, 2005.

[39] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Advances in neural information processing systems*, pages 153–160, 2004.

[40] Narayan Hegde, Jason D Hipp, Yun Liu, Michael Emmert-Buck, Emily Reif, Daniel Smilkov, Michael Terry, Carrie J Cai, Mahul B Amin, Craig H Mermel, et al. Similar image search for histopathology: Smily. *NPJ digital medicine*, 2(1):1–9, 2019.

[41] Sobhan Hemati, Shivam Kalra, Cameron Meaney, Morteza Babaie, Ali Ghodsi, and Hamid Tizhoosh. Cnn and deep sets for end-to-end whole slide image representation learning. 2021.

[42] Sobhan Hemati, Mohammad Hadi Mehdizavareh, Shojaeddin Chenouri, and Hamid R Tizhoosh. A non-alternating graph hashing algorithm for large scale image search. *arXiv preprint arXiv:2012.13138*, 2020.

[43] Jae-Pil Heo, Youngwoon Lee, Junfeng He, Shih-Fu Chang, and Sung-Eui Yoon. Spherical hashing. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2957–2964. IEEE, 2012.

[44] Tuan Hoang, Thanh-Toan Do, Huu Le, Dang-Khoa Le-Tan, and Ngai-Man Cheung. Simultaneous compression and quantization: A joint approach for efficient unsupervised hashing. *Computer Vision and Image Understanding*, 191:102852, 2020.

[45] Le Hou, Dimitris Samaras, Tahsin M Kurc, Yi Gao, James E Davis, and Joel H Saltz. Patch-based convolutional neural network for whole slide tissue image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2424–2433, 2016.

[46] Di Hu, Feiping Nie, and Xuelong Li. Discrete spectral hashing for efficient similarity retrieval. *IEEE Transactions on Image Processing*, 28(3):1080–1091, 2018.

[47] Qinghao Hu, Peisong Wang, and Jian Cheng. From hashing to cnns: Training binary weight networks via hashing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[48] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[49] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pages 2127–2136. PMLR, 2018.

[50] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pages 2127–2136. PMLR, 2018.

[51] Maximilian Ilse, Jakub M Tomczak, and Max Welling. Deep multiple instance learning for digital histopathology. In *Handbook of Medical Image Computing and Computer Assisted Intervention*, pages 521–546. Elsevier, 2020.

[52] Tommi Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *Advances in neural information processing systems*, pages 487–493, 1999.

[53] Andrew Janowczyk and Anant Madabhushi. Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of pathology informatics*, 7, 2016.

[54] Qing-Yuan Jiang and Wu-Jun Li. Scalable graph hashing with feature transformation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[55] Shivam Kalra. Learning discriminative representations for gigapixel images. 2022.

[56] Shivam Kalra, Mohammed Adnan, Sobhan Hemati, Taher Dehkharghanian, Shahryar Rahnamayan, and Hamid R Tizhoosh. Pay attention with focus: A novel learning scheme for classification of whole slide images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 350–359. Springer, 2021.

[57] Shivam Kalra, Mohammed Adnan, Graham Taylor, and Hamid R Tizhoosh. Learning permutation invariant representations using memory networks. In *European Conference on Computer Vision*, pages 677–693. Springer, 2020.

[58] Shivam Kalra, HR Tizhoosh, Charles Choi, Sultaan Shah, Phedias Diamandis, Clinton JV Campbell, and Liron Pantanowitz. Yottixel–an image search engine for large archives of histopathology whole slide images. *Medical Image Analysis*, 65:101757, 2020.

[59] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[60] Pegah Khosravi, Ehsan Kazemi, Marcin Imielinski, Olivier Elemento, and Iman Hajirasouliha. Deep convolutional neural networks enable discrimination of heterogeneous digital pathology images. *EBioMedicine*, 27:317–328, 2018.

[61] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[62] Diederik P Kingma and Max Welling. An introduction to variational autoencoders. *arXiv preprint arXiv:1906.02691*, 2019.

[63] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[64] Bin Kong, Xin Wang, Zhongyu Li, Qi Song, and Shaoting Zhang. Cancer metastasis detection via spatially structured deep network. In *International Conference on Information Processing in Medical Imaging*, pages 236–248. Springer, 2017.

[65] Weihao Kong and Wu-Jun Li. Isotropic hashing. *Advances in neural information processing systems*, 25:1646–1654, 2012.

[66] Weihao Kong and Wu-Jun Li. Isotropic hashing. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1646–1654, Red Hook, NY, USA, 2012. Curran Associates Inc.

[67] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[68] Brian Kulis and Trevor Darrell. Learning to hash with binary reconstructive embeddings. In *Advances in neural information processing systems*, pages 1042–1050, 2009.

[69] Brian Kulis and Kristen Grauman. Kernelized locality-sensitive hashing for scalable image search. In *2009 IEEE 12th international conference on computer vision*, pages 2130–2137. IEEE, 2009.

[70] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan. Simultaneous feature learning and hash coding with deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3270–3278, 2015.

[71] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[72] Xuelong Li, Di Hu, and Feiping Nie. Large graph hashing with spectral rotation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[73] Kevin Lin, Jiwen Lu, Chu-Song Chen, and Jie Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1183–1192, 2016.

[74] Zijia Lin, Guiguang Ding, Jungong Han, and Jianmin Wang. Cross-view retrieval via probability-based semantics-preserving hashing. *IEEE transactions on cybernetics*, 47(12):4342–4355, 2016.

[75] Wei Liu, Junfeng He, and Shih-Fu Chang. Large graph construction for scalable semi-supervised learning. In *ICML*, 2010.

[76] Wei Liu, Cun Mu, Sanjiv Kumar, and Shih-Fu Chang. Discrete graph hashing. In *Advances in neural information processing systems*, pages 3419–3427, 2014.

[77] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. Supervised hashing with kernels. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081. IEEE, 2012.

[78] Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Hashing with graphs. 2011.

[79] Xiao Liu, Dacheng Tao, Mingli Song, Ying Ruan, Chun Chen, and Jiajun Bu. Weakly supervised multiclass video segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[80] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218*, 1(2), 2020.

[81] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[82] Marc Macenko, Marc Niethammer, James S Marron, David Borland, John T Woosley, Xiaojun Guan, Charles Schmitt, and Nancy E Thomas. A method for normalizing histology slides for quantitative analysis. In *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 1107–1110. IEEE, 2009.

[83] Sam Maksoud, Kun Zhao, Peter Hobson, Anthony Jennings, and Brian C Lovell. Sos: Selective objective switch for rapid immunofluorescence whole slide image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3862–3871, 2020.

[84] Jonathan H Manton. Optimization algorithms exploiting unitary constraints. *IEEE Transactions on Signal Processing*, 50(3):635–650, 2002.

[85] Akito Nagase, Masanobu Takahashi, and Masayuki Nakano. Automatic calculation and visualization of nuclear density in whole slide images of hepatic histological sections. *Bio-medical materials and engineering*, 26(s1):S1335–S1344, 2015.

[86] Mohammad Norouzi, David J Fleet, and Russ R Salakhutdinov. Hamming distance metric learning. In *Advances in neural information processing systems*, pages 1061–1069, 2012.

[87] Loïc Paulevé, Hervé Jégou, and Laurent Amsaleg. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern recognition letters*, 31(11):1348–1358, 2010.

[88] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

[89] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

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

[91] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[92] Zhaofan Qiu, Ting Yao, and Tao Mei. Deep quantization: Encoding convolutional activations with deep generative model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6759–6768, 2017.

[93] Gwenolé Quellec, Guy Cazuguel, Béatrice Cochener, and Mathieu Lamard. Multiple-instance learning for medical image and video analysis. *IEEE reviews in biomedical engineering*, 10:213–234, 2017.

[94] Md Mahmudur Rahman, Prabir Bhattacharya, and Bipin C Desai. A framework for medical image retrieval using machine learning and statistical similarity matching techniques with relevance feedback. *IEEE transactions on Information Technology in Biomedicine*, 11(1):58–69, 2007.

[95] Emad A Rakha, Mohamed Aleskandarani, Michael S Toss, Andrew R Green, Graham Ball, Ian O Ellis, and Leslie W Dalton. Breast cancer histologic grading using digital microscopy: concordance and outcome association. *Journal of clinical pathology*, 71(8):680–686, 2018.

[96] Abtin Riasatian, Morteza Babaie, Danial Maleki, Shivam Kalra, Mojtaba Valipour, Sobhan Hemati, Manit Zaveri, Amir Safarpoor, Sobhan Shafiei, Mehdi Afshari, et al. Fine-tuning and training of densenet for histopathology image representation using tcga diagnostic slides. *arXiv preprint arXiv:2101.07903*, 2021.

[97] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Adversarially robust training through structured gradient regularization. *arXiv preprint arXiv:1805.08736*, 2018.

[98] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

[99] Jorge Sánchez and Florent Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR 2011*, pages 1665–1672. IEEE, 2011.

[100] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.

[101] Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[102] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep

networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[103] Jian Shao, Fei Wu, Chuanfei Ouyang, and Xiao Zhang. Sparse spectral hashing. *Pattern recognition letters*, 33(3):271–277, 2012.

[104] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 37–45, 2015.

[105] Fumin Shen, Yan Xu, Li Liu, Yang Yang, Zi Huang, and Heng Tao Shen. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):3034–3044, 2018.

[106] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[107] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

[108] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo. Effective multiple feature hashing for large-scale near-duplicate video retrieval. *IEEE Transactions on Multimedia*, 15(8):1997–2008, 2013.

[109] Yang Song, Hang Chang, Heng Huang, and Weidong Cai. Supervised intra-embedding of fisher vectors for histopathology image classification. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 99–106. Springer, 2017.

[110] Yang Song, Ju Jia Zou, Hang Chang, and Weidong Cai. Adapting fisher vectors for histopathology image classification. In *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pages 600–603. IEEE, 2017.

[111] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

[112] David Tellez, Geert Litjens, Jeroen van der Laak, and Francesco Ciompi. Neural image compression for gigapixel histopathology image analysis. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[113] Hamid Reza Tizhoosh and Liron Pantanowitz. Artificial intelligence and digital pathology: challenges and opportunities. *Journal of pathology informatics*, 9, 2018.

[114] Rafael Uetz and Sven Behnke. Large-scale object recognition with cuda-accelerated hierarchical neural networks. In *2009 IEEE international conference on intelligent computing and intelligent systems*, volume 1, pages 536–541. IEEE, 2009.

[115] Jeroen Van der Laak, Geert Litjens, and Francesco Ciompi. Deep learning in histopathology: the path to the clinic. *Nature medicine*, 27(5):775–784, 2021.

[116] Avery Wang et al. An industrial strength audio search algorithm. In *Ismir*, volume 2003, pages 7–13. Washington, DC, 2003.

[117] J. Wang, T. Zhang, J. Song, N. Sebe, and H. Shen. A survey on learning to hash. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 40(04):769–790, apr 2018.

[118] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. A survey on learning to hash. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):769–790, 2017.

[119] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for large-scale search. *IEEE transactions on pattern analysis and machine intelligence*, 34(12):2393–2406, 2012.

[120] Qifan Wang, Bin Shen, Shumiao Wang, Liang Li, and Luo Si. Binary codes embedding for fast image tagging with incomplete labels. In *European Conference on Computer Vision*, pages 425–439. Springer, 2014.

[121] Xiaofang Wang, Yi Shi, and Kris M Kitani. Deep supervised hashing with triplet labels. In *Asian conference on computer vision*, pages 70–84. Springer, 2016.

[122] John N Weinstein, Eric A Collisson, Gordon B Mills, Kenna R Mills Shaw, Brad A Ozenberger, Kyle Ellrott, Ilya Shmulevich, Chris Sander, and Joshua M Stuart. The cancer genome atlas pan-cancer analysis project. *Nature genetics*, 45(10):1113–1120, 2013.

[123] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.

[124] Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1-2):397–434, 2013.

[125] Xing Xu, Fumin Shen, Yang Yang, Heng Tao Shen, and Xuelong Li. Learning discriminative binary codes for large-scale cross-modal retrieval. *IEEE Transactions on Image Processing*, 26(5):2494–2507, 2017.

[126] Yang Yang, Fumin Shen, Heng Tao Shen, Hanxi Li, and Xuelong Li. Robust discrete spectral hashing for large-scale image semantic indexing. *IEEE Transactions on Big Data*, 1(4):162–171, 2015.

[127] Ting Yao, Fuchen Long, Tao Mei, and Yong Rui. Deep semantic-preserving and ranking-based hashing for image retrieval. In *IJCAI*, pages 3931–3937, 2016.

[128] Kun-Hsing Yu, Ce Zhang, Gerald J Berry, Russ B Altman, Christopher Ré, Daniel L Rubin, and Michael Snyder. Predicting non-small cell lung cancer prognosis by fully automated microscopic pathology image features. *Nature communications*, 7(1):1–10, 2016.

[129] Yunchao Gong, M. Pawlowski, Fei Yang, L. Brandy, L. Boundev, and R. Fergus. Web scale photo hash clustering on a single machine. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19–27, 2015.

[130] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.

[131] Cecilia Zappa and Shaker A Mousa. Non-small cell lung cancer: current treatment and future advances. *Translational lung cancer research*, 5(3):288, 2016.

[132] Shuangfei Zhai, Walter Talbott, Carlos Guestrin, and Joshua Susskind. Adversarial fisher vectors for unsupervised representation learning. *Advances in Neural Information Processing Systems*, 32:11158–11168, 2019.

[133] Ruimao Zhang, Liang Lin, Rui Zhang, Wangmeng Zuo, and Lei Zhang. Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *IEEE Transactions on Image Processing*, 24(12):4766–4779, 2015.

[134] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4320–4328, 2018.

[135] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.

[136] Shujin Zhu, Yuehua Li, Shivam Kalra, and Hamid R Tizhoosh. Multiple disjoint dictionaries for representation of histopathology images. *Journal of Visual Communication and Image Representation*, 55:243–252, 2018.

[137] Roland S Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. *arXiv preprint arXiv:2102.08850*, 2021.