Correlation Discovery for Multi-view and Multi-label Learning

A Dissertation Presented

by

Lichen Wang

to

The Department of Electrical and Computer Engineering

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Northeastern University

Boston, Massachusetts

2021

To my family.

Contents

List of Figures

List of Tables

Acknowledgments

My deepest and sincerest gratitude is first to Professor Yun Raymond Fu, my supervisor, for his strong support, patient guidance, and dependable encouragement across my PhD study in the past five years. He not only guided me how to do good research work, but also taught me how to methodically, effectively, and efficiently achieve the goals. What I learned from him would undoubtedly benefit my career and my whole life. I would not have my current achievements without his tremendous and selfless help.

I would also like to thank my committee members, Prof. Ehsan Elhamifar, Prof. Lili Su, and Prof. Hongfu Liu for forming my PhD committee especially in this pandemic period of time. Many thanks for their constructive comments, insightful suggestions, valuable time. I also learned a lot from this process.

In addition, I would like to thank all my teammates from Synergetic Media Learning Lab Prof. Zhengming Ding, Prof. Sheng Li, Prof. Zhiqiang Tao, Prof. Yu Kong, Dr. Handong Zhao, Dr. Shuyang Wang, Dr. Yue Wu, Dr. Joseph P. Robinson, Songyao Jiang, Bin Sun, Haiyi Mao, Kunpeng Li, Yulun Zhang, Kai Li, Can Qin, Chang Liu, Huan Wang, Yu Yin, Yue Bai, Yi Xu, Yizhou Wang, Xu Ma, Sara Al Bunian. I also want to thank my volunteers, Yunyu Liu, Hang Di, Kasey Lee, Taotao Jing, Allyson Vakhovskaya, Daniel J. Peluso, and Emily Freed, for helping me handle large-scale projects. For other lab members, Qianqian Ma, Dr. Bo Zong, Prof. Jun Li, Prof. Bineng Zhong, Prof. Shuhan Chen, Prof. Gan Sun, Prof. Siyu Xia, Prof. Qianqian Wang, Prof. Guoshuai Zhao, Prof. Wencang Zhao, Prof. Haitao Xiong, Prof. Yi Tian, I also thank you very much. I have spent my wonderful five years with these excellent colleagues and left the impressive memory.

Finally, I would like to express my gratitude to my family and friends for their continuously encouraging and helping me when I face challenges and difficulties. They made me happy, calm, and confident in my PhD study.

Abstract of the Dissertation

Correlation Discovery for Multi-view and Multi-label Learning

by

Lichen Wang

Doctor of Philosophy in Electrical and Computer Engineering

Northeastern University, 2021

Dr. Yun Fu, Advisor

Correlation indicates the interactions or connections across different instances. It exists in a wide range of real-world scenarios such as scene understanding, social network, time-series data, and human-object interactions. Correlation provides the unique and informative knowledge to reveal the latent connections across instances, and it plays an essential and important role in the machine learning field.

However, recovering and utilizing correlation is challenging. First, it is hard to explicitly and clearly define the correlations, which leads to relatively small and high-level noise datasets. Second, the correlation is task-specific, which cannot be generalized to more diverse tasks. This challenge increases the cost of correlation learning and its down-stream applications. Third, even if the correlation is given, how to efficiently utilize the learned/given correlations and enhance the final performance is still difficult. This point has not been well-explored.

In this dissertation research, we investigate the techniques to effectively discover various kinds of correlations in machine learning tasks including multi-view learning, multi-label learning, image/scene understanding, time-series data analysis, and human action recognition. Specifically, we propose algorithms from the following perspectives: 1) designing correlation exploration frameworks to automatically explore the label correlations in multi-label scenarios, 2) proposing a multi-view fusion strategy which effectively dig the latent correlations across different views to achieve high-accuracy human action recognition, and 3) exploring the inductive and unsupervised graph representation learning task, which aims to preserve the correlation knowledge in graph structured objects. To demonstrate the effectiveness of the proposed algorithms, various experiments on commonly used datasets have been implemented and the results show the superiority of our algorithms over the other state-of-the-art methods.

Chapter 1

Introduction

1.1 Background

Formally, correlation is a mutual relationship or connection between two or more things. In our work, correlation denotes the interactions or connections across different instances. It exists in a wide range of real-life environments. Correlation provides people with unique and informative knowledge, which is always utilized in their daily decision making, learning, and planning procedures. For example, if today is *sunshine* in summer, it is probably a *hot* day. Similarly, a *cloudy* weather always reminds people to check if there will be *raining*. These correlations are explicit and explicable, which become the general common senses. More diverse correlations also exist. In human action analysis scenario, human-object interaction (e.g., *checking watch*, *drinking water*, and *fighting*) contains different types of correlations which provide information for down-stream tasks such as action recognition, prediction, and interaction. In social network (e.g., Facebook, Twitter, and LinkedIn), the background of the users (e.g., education, experience, career, and the achievements), the posters (e.g., sentences and images), and the likes/comments from their connected friends contain various correlations in different aspects. This information could benefit the recommendation, advertising, and searching performances.

To further improve the learning performance of machine learning applications, effectively exploring and utilizing the correlation knowledge are the feasible and promising research directions. Conventional machine learning methods mainly focus on exploring the given samples separately, which cannot well solve the correlation learning challenges. A small partial of machine learning methods (e.g., clustering and transfer learning) explore the data distribution knowledge from unlabeled samples to enhance the learning performance. However, most of these methods explore the

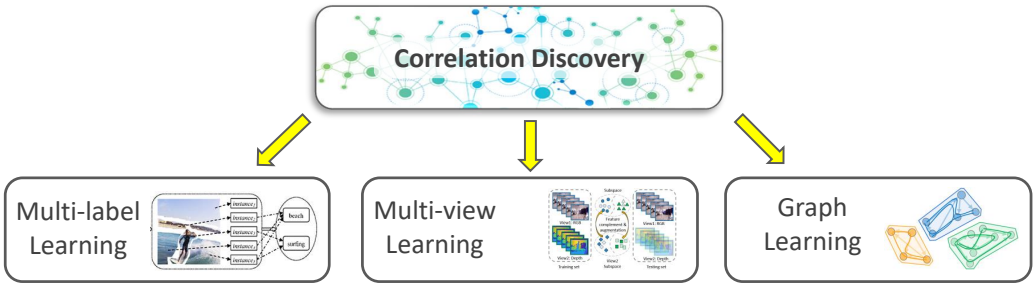


Figure 1.1: The organization of our thesis. We explore correlations from different data formats in various machine learning tasks. Specifically, we explored multi-label learning, multi-view learning, and graph structural object representation learning by conventional and deep learning-based strategies.

distribution information in pure feature space based on single and simple metrics (e.g, Euclidean distance and Cosine distance). While, these metrics cannot handle sophisticated correlations and various correlation types. In addition, an instance could have multiple modalities/representations (e.g., visual, semantic, and attributes). Exploring the correlation in one feature space may not be comprehensive. To this end, how to explore the correlations in different views is also a potential point in correlation learning.

In summary, compared with traditional machine learning methods, recognizing and utilizing correlations and improving the down-stream applications are difficult tasks. The major challenges are listed below:

- The correlation is hard to explicitly and clearly define. For example, how to define the correlation between *sunshine* and *hot*. A naive correlation score could be one solution, however, the score is not reliable such as in *winter*. Thus, the fixed correlation score could lead to negative influence for final performance.

- The correlation is task-specific, which means different tasks (e.g., scene understanding and object detection) have totally different labels and correlations. Existing datasets cannot be generalized to more diverse tasks. This challenge causes relatively small and high-level noise datasets, and also makes correlation learning costly for real-world deployments.

- It is still difficult to effectively and efficiently utilize the given/learned correlation to reliably improve the performance. The correlation and the feature representation should be complementary to each other instead of contradictory.

In this thesis, we concentrate on solving the aforementioned challenges. Figure 1.1 illustrates the topic structure of our thesis. Specifically, we investigate the techniques to effectively discover various kinds of correlations in machine learning tasks including multi-view learning, multi-label learning, and graph representation learning for different applications such as image/scene understanding, human action recognition, and graph data classification. The details are listed below:

- We designed frameworks to automatically predict multiple labels from a single instance. The frameworks are used for multi-label learning and scene understanding tasks. Label correlations residing inside each instance and across different instances are learned. Then, the correlation is further used for multi-label prediction tasks.
- We proposed a view-correlation discovery network for RGB-Depth action recognition. Our approach fully digs correlations across RGB and depth views, and the view-specific prediction results are effectively fused and achieved higher performance.
- We proposed a general framework for inductive and unsupervised representation learning on graph structured objects which contain sophisticated instance representations and correlation types. Instead of directly dealing with the computational challenges raised by graph similarity evaluation, a novel sampling, encoding, and embedding framework is proposed.

1.2 Related works

1.2.1 Multi-label Learning

In real-world applications, one object could relate to tens or hundreds of semantic descriptions or attributes. For instance, an image illustrates "It is a *sunny day* with *blue sky* and a *lake/water* nearby". This image contains multiple labels (i.e., *sunny*, *blue sky*, and *water*) selected from a large number of candidate labels. Compared to single label classification tasks, multi-label tasks assume multiple labels exist in each instance. There are numerous real-world applications that require multi-label learning including data mining [5], large-scale image retrieval [6, 7], multi-label classification [6, 7], natural language processing [8], and advertising [9]. As a result, multi-label learning has become an attractive and critical area of research in recent years [10, 11, 12, 13, 14].

There are multiple unique challenges in multi-label learning. First, the available datasets (e.g., SUN [15], CUB [16], and AWA [17]) are relatively small, since multi-label data collection and labeling procedures are labor intense and expensive compared with the single-label setting. In

addition, the labels in most datasets follow a long-tail distribution. It means that some "common" labels (e.g., *blue sky*, *outdoor*, and *trees*) are much more prevalent than "rare" labels (e.g., *fight* and *fire*). For instance, the SUN dataset has 14340 samples in total. The most common label (i.e., *Man-made*) shows up 8089 times, while the rarest label (i.e., *Fire*) only shows up 73 times. The significant unbalanced training samples could negatively affect the learning performance. Moreover, multi-label datasets suffer from high-level label noise due to the subjective nature of the labels (e.g., *hot*, *warm*, and *stressful*). It is hard to obtain consistent label results since different people hold different opinions. To this end, more robust and noise insensitive models are required for multi-label learning tasks. Second, correlation is the most essential knowledge to achieve high performance prediction. There always exists certain trends between labels, for instance, *"Moist"* and *"Dry"* should not exist in the same time, but the pairs *"Dry-Desert"* and *"Moist-Water"* usually emerge simultaneously. Meanwhile, there are no strong correlations across *"Water"*, *"Stone"*, and other labels. These label correlations are crucial for making accurate predictions and could improve performance significantly [18, 19]. Nevertheless, this syntactic knowledge is not available in most existing datasets. Moreover, creating the correlation/syntactical map would require well-trained experts with task-specific definitions, meaning that the created maps would not be easily generalized to other similar applications. This specificity considerably limits the potential applications of existing approaches.

To solve these challenges, semi-supervised multi-label learning [20, 21, 22, 23, 24, 25] is a practical solution to enhance the learning performance by exploring unlabeled samples. [26] proposed an error correcting output correcting scheme to achieve the multi-class heterogeneous domain adaptation. [27] learned a low-rank kernel strategy which eliminates the noise and enhances the representation ability. [28] proposed a reliable graph learning strategy. It obtains robust graphs by adaptively removing errors and noise from the original samples. [29] mapped the data into a higher dimensional space and deployed a multiple-kernel-based algorithm for the recommendation system.

1.2.2 Multi-view Learning

Multi-view learning aims to integrate complementary information from different views, where the views refer to various feature representations, modalities or sensors. Most existing methods focus on analyzing static multi-view data (e.g., image, description, and attributes) to improve the performances of down-stream tasks including classification, clustering, detection, and segmentation. It is explored in a wide range of applications such as action recognition, object detection, semantic

segmentation, and information retrieval [30, 31, 32, 33, 34]. It has become attractive and urgent as the increasing multi-modal sensors are widely deployed in a great number of real-world applications.

Multi-view learning contains several challenges compared with a single-view scenario. First, the feature domains of different views are heterogeneous. They have significant various feature distributions. Naively fusing multi-view features (e.g., concatenation and summation) could induce a negative effect and hurt the performance. In addition, it is a common situation where one or more views are missing such as sensor malfunction, equipment deficiency, and signal loss in data transformation.

Conventional research efforts [35, 36, 37, 38, 39] mainly utilize effective feature extraction approaches to obtain view-specific representation first, then deploy fusion mechanism to integrate these representations together. However, these methods assume data are accessible for all the views, yet without considering the practical and common incomplete view scenarios. Moreover, different views could provide class-level unique distinctiveness, and it is crucial to explore the correlation across action classes and views to further improve the learning performance. Hence, their performances inevitably degrade when dealing with partial multi-view data.

1.2.3 Graph Representation Learning

Graph representation learning has been the core problem of machine learning tasks. Given a graph structured object, the goal is to represent the input graph as a dense low-dimensional vector so that we are able to feed this vector into off-the-shelf machine learning or data management techniques. Consider graph contains comprehensive structural/correlation information across different instances, it is an ideal data format to fully explore and utilize the correlation knowledge to further improve the performance of a wide spectrum of downstream tasks such as classification [40], anomaly detection [41], information retrieval [42], and many others [43, 44].

Different from consistent data formats (e.g., image, video, and audio), the format of graph structural data is inconsistent. It is based on the number of nodes (i.e., instances) and the edges (i.e., correlations) between these instances. To this end, it is difficult for conventional machine learning or deep learning methods to process graph data. This unique character makes graph representation learning become more challenging compared with conventional data formats. In addition, inductive and unsupervised graph learning is a critical technique for predictive or information retrieval tasks where label information is difficult to obtain. It is also challenging to make graph learning inductive and unsupervised at the same time, as learning processes guided by reconstruction error based loss

functions inevitably demand graph similarity evaluation that is usually computationally intractable.

Previous deep graph learning techniques mainly focus on transductive [45] or supervised settings [42]. A few recent studies focus on autoencoding specific structures, such as directed acyclic graphs [46], trees or graphs that can be decomposed into trees [47], and so on. From the perspective of graph generation, [48] propose to generate graphs of similar graph statistics (e.g., degree distribution), and [49] provide a GAN based method to generate graphs of similar random walks.

1.3 Dissertation Organization

The rest of this dissertation is organized as follows.

In chapter 2, we develop a generic multi-label learning framework based on Adaptive Graph and Marginalized Augmentation in a semi-supervised scenario. Generally speaking, our approach makes use of a small amount of labeled data associated with a lot of unlabeled data to boost the learning performance. An adaptive similarity graph, a marginalized augmentation strategy, and a feature-label autoencoder is proposed to solve the challenges. In addition, a deep learning-based generative correlation discovery network is proposed. Specifically, a generative model is utilized to conditionally generate more diverse samples, and a correlation discovery network is designed to automatically learn the label correlations and further improve the prediction performance.

In chapter 3, we propose a Generative Multi-View Action Recognition framework for RGB-D action recognition. Specifically, an adversarial generative network is leveraged to generate one view conditioning on the other view, which fully explores the latent correlation in both intra-view and cross-view aspects. Moreover, an effective View Correlation Discovery Network is proposed to further fuse the multi-view information in a higher-level label space.

In chapter 4, we propose a general framework SEED (Sampling, Encoding, and Embedding Distributions) for inductive and unsupervised representation learning on graph structured objects. Given an input graph, the SEED framework samples a number of subgraphs whose reconstruction errors could be efficiently evaluated, encodes the subgraph samples into a collection of subgraph vectors, and employs the embedding of the subgraph vector distribution as the output vector representation for the input graph.

In chapter 5, we present a conclusion of our methods.

Chapter 2

Multi-label Learning

2.1 Background

In real-world applications, one object could relate to tens or hundreds of semantic descriptions or attributes. For instance, an image illustrates "It is a *sunny day* with *blue sky* and a *lake/water* nearby". This image contains multiple labels (i.e., *sunny*, *blue sky*, and *water*) selected from a large number of candidate labels. Compared with single label classification tasks, multi-label tasks assume multiple labels exist in each instance [10, 11, 12, 13, 14]. As illustrated in Figure 2.1, multi-label classification aims to accurately predict all positive labels from the given instances. It is a more challenging, practical, and potential classification task for a large number of real-world applications, e.g., video concept recognition [50], image annotation [14], retrieval, and natural language processing [51].

There are several unique challenges in multi-label scenarios. First, the available datasets (e.g., SUN [15], CUB [16], and AWA [17]) are relatively small. Since multi-label data collection and labeling procedures are labor intense and expensive compared with the single-label setting. In addition, multi-label datasets suffer from high-level label noise due to the subjective nature of the labels (e.g., *hot*, *warm*, and *stressful*). It is hard to obtain consistent label results since different people hold different opinions. Third, the labels in most datasets follow a long-tail distribution. It means that some "common" labels (e.g., *blue sky*, *outdoor*, and *trees*) are much more prevalent than "rare" labels (e.g., *fight* and *fire*). For instance, the SUN dataset has $14,340$ samples in total. The most common label (i.e., *Man-made*) shows up $8,089$ times, while the rarest label (i.e., *Fire*) only shows up 73 times. The significant unbalanced training samples could negatively affect the learning performance. More sophisticated and specifically designed models are required for multi-label learning tasks.

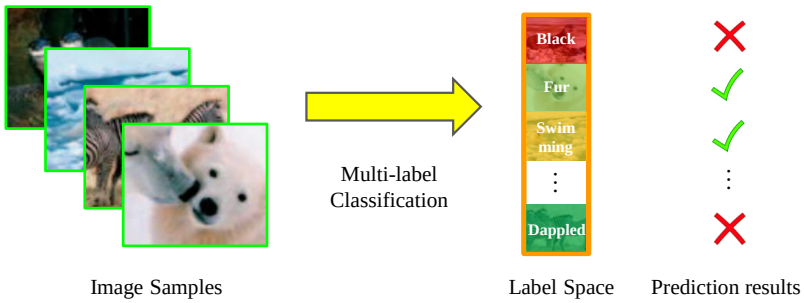


Figure 2.1: Concept of multi-label classification. Compared with conventional classification tasks, multi-label learning assumes a single instance could contain multiple positive labels. Multi-label classification aims to accurately predict all the labels from the given instance.

One straightforward solution for multi-label learning is utilizing multiple single-label learning classifiers to recover each label individually [14]. However, the latent correlations between labels are not considered in this strategy (e.g., *blue sky* usually show up with *outdoor*). Label relation plays an important role for multi-label learning [52]. [53] designed a contextual merging step based on the output of each classifier to leverage the correlations. [54] handles the missing label problem via learning the semantic structural information to build the label correlations. It projects samples to the semantic space with an effective semantic descriptor. [55] learned the labels as well as the correlations simultaneously in the training stage for multi-view scenarios. [56] designed a dependence maximization strategy for multi-label dimension deduction based on Hilbert-Schmidt independence criterion. [57] proposed a non-negative matrix factorization to obtain robust prediction performance. [58] proposed a model which automatically identifies easy and hard prediction samples. It then uses the obtained easy samples to enhance the prediction of hard samples. However, most of these approaches are still in supervised learning manner which cannot perform well in the training data shortage situation.

In addition, although training classification models with large-scale multi-label datasets is one solution, as introduced above, generating such multi-label datasets is a challenging and expensive task. However, relevant and unlabeled data are easy to obtain. Based on this, semi-supervised learning [20, 21] is a practical solution to enhance the learning performance by exploring unlabeled samples. Semi-supervised learning utilizes a small-scale well-labeled samples associated with a large-scale unlabeled samples to improve the learning performance [59, 21, 22, 20, 60, 61]. There are various ways to achieve semi-supervised learning. A detailed introduction can be found in [21]. Its essential insight is to explore the feature distribution knowledge from unlabeled samples and

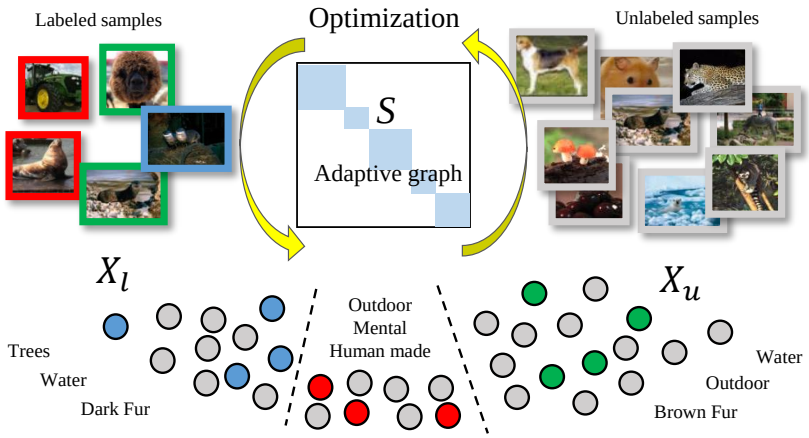


Figure 2.2: Concept of the adaptive graph semi-supervised learning strategy. The representations of labeled and unlabeled samples, and the adaptive graph are jointly optimized, which could effectively handle the graph robustness and model effectiveness challenges.

improve the effectiveness of down-stream tasks. [22] filters the training sets and obtains a model which is independent to the training initialization procedure. [62] utilized the hashing and transfer learning strategies to achieve transfer hashing for privileged information. It could handle data sparsity issues in the deep learning framework. [63] proposed a self-supervised mechanism which contains two losses to achieve semi-supervised learning scenario. [64] utilizes a differentiable surrogate of the non-differentiable Hungarian algorithm to achieve the view-specific alignment. [65] effectively utilizes the knowledge from both feature and label space. The pairwise sample assignments are minimized across each data point. However, most of the aforementioned approaches mainly handle the single-label classification tasks, which ignore the unique challenges of multi-label setting such as the "long-tail" label distribution issue, and the latent label correlation aspect, which lead to low prediction performance.

2.2 Adaptive Graph and Marginalized Augmentation

2.2.1 Motivation

Due to the aforementioned challenges, a specifically designed semi-supervised multi-label learning approach is potential for solving the challenges. From all the various semi-supervised strategies, graph-based approaches [66] have attracted great attention due to their high performance. It deploys an affiliate graph to explore the latent data structure residing in both source and target

samples. Although graph-based methods achieve high performance, there is a main drawback. Specifically, the classification performance heavily depends on the quality of the affiliate graph, while it is difficult to always obtain an effective affiliate graph. Moreover, most graph generation methods are parameter sensitive. Thus, the same set of graph generation configurations could not achieve the best performance for other resources. For example, the graph which is generated based on the original sample representation which could be influenced by noise and the configurations of the similarity metrics. These factors could significantly affect the graph generation and decrease the final performance.

Previous works exploited adaptive graphs to handle the sensitivity issue [22, 23, 24, 25, 67]. The concept is shown in Figure 2.2. The main idea of adaptive graph is to jointly update the learned representations of labeled and unlabeled samples and the graph, which could effectively handle the graph robustness and further improve the performance. An adaptive affiliate graph is proposed in [68] which is adaptively optimized in the training stage. [32, 69] deploy graph optimization strategy for unsupervised feature selection and representation learning tasks. [70] extended this approach to image and video scenarios. [71] deployed affiliate graph associated with subspace learning to learn more distinctive feature representation and helped the adaptive graph learning. A Gaussian random field and a harmonic function were proposed to improve the performance [20]. [26] proposed an error correcting output correcting scheme to achieve the multi-class heterogeneous domain adaptation. [27] learned a low-rank kernel strategy which eliminates the noise and enhances the representation ability. [28] proposed a reliable graph learning strategy. It obtains robust graphs by adaptively removing errors and noise from the original samples. [29] mapped the data into a higher dimensional space and deployed a multiple-kernel-based algorithm for recommendation system. However, most of the graph based approaches still rely on the similarity measurement in either the feature space or a learned subspace. The performance of this strategy is easily affected by noise and outliers. Moreover, most of the aforementioned approaches mainly handle the single-label classification tasks, which ignore utilizing the latent label correlation knowledge residing inside the samples which is crucial for multi-label setting.

Augmenting samples from the auxiliary domain is a promising direction for multi-label learning. Marginalized Corrupted Features (MCF) is an effective and efficient feature augmentation strategy. MCF "corrupts" existing samples and "generates" infinite artificial samples for model training [72]. It is specifically designed for the situation in which only limited training samples are available. More details are introduced in [73]. [74] proposed a marginalized Denoising Auto-encoder (mDAE) approach for non-linear representation learning. mDAE achieves similar or even better

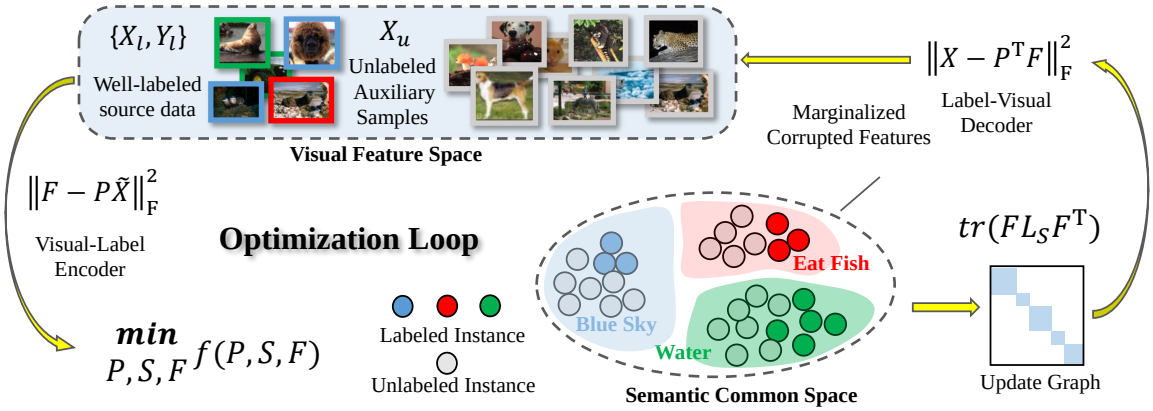


Figure 2.3: Framework of AGMA approach. A visual-label encoder, P, maps data from visual space to label space. An adaptive affinity graph, S, is adaptively optimized based on both label space and feature space, which also explores the pairwise latent connections across both labeled and unlabeled data. A marginalized feature augmentation strategy is further deployed to extend the feature space and enhance the model robustness. The predicted label matrix F, the adaptive graph S, and the encoder P are jointly optimized which helps the model to obtain the best and reliable performance.

performance with much fewer training samples. [75] proposed a Regularized Marginalized Cross-View learning (RMCV) framework with marginalized denoising autoencoder, which effectively improves the model robustness. However, these methods either focus on representation learning tasks or supervised classification tasks which cannot effectively explore unlabeled data.

In this thesis, a novel and generic multi-label learning framework via Adaptive Graph and Marginalized Augmentation strategy (AGMA) in semi-supervised scenario is proposed. The framework is shown in Figure 2.3. The core insight is jointly propagating the labeled and unlabeled data by an adaptive graph and seeking an effective and robust visual-label encoder with marginalized feature augmentation strategy. Such two strategies could assist each other to enhance the final performance. The main contributions of this work are listed below:

- An adaptive graph is proposed to explore the latent correlations of labeled and unlabeled samples. It is jointly updated with other components to obtain the best performance.
- A feature-label autoencoder is proposed to project the samples between label space and feature space. In addition, this framework is able to fully explore the feature-label connection and reduce the computational cost in the testing stage.
- A marginalized feature augmentation strategy is deployed which extends infinite samples from the limited samples and further improves the model robustness.

Table 2.1: Symbol Description Table

Symbol	Description
x_i, x_j	Feature vector of i-th and j-th samples.
X_l, X_u	Feature matrix of labeled and unlabeled samples.
X	$X = [X_l, X_u]$.
f_i, f_j	Predicted label vector of x_i and x_j.
F_l, F_u	Predicted multi-label of X_l and X_u, and $F = [F_l, F_u]$.
y_i, y_j	Groundtruth label vector of x_i and x_j.
Y_l	Groundtruth label of X_l.
d, d_l	Dimensions of feature space and label space.
S	Adaptive affinity graph.
L_S	Graph Laplacian matrix of S.
P	Visual-Label Encoder, and Label-Visual Decoder is $P^\top$.
n_l, n_u	Number of labeled and unlabeled samples, $n = n_l + n_u$.
λ, μ	Trade-off parameters.
δ	Gaussian distribution variance.

- An optimization approach is designed to solve all variables. Five datasets are deployed in the experiments and the results illustrate the efficiency and effectiveness of the model.

AGMA is an extension of our previous research work [71]. There are three-fold modifications to improve the performance. First, we deploy an autoencoder strategy to directly project the samples between feature space and label space. It avoids the negative influence from the uncontrollable latent subspace of [71]. Second, a marginalized augmentation approach is designed to extend the feature distribution for further improving the performance. Third, our approach is efficient in inferring step, since our model is able to project the new samples from feature space to label space without extra optimization process. Extensive experiments indicate that AGMA achieves better performance. In the following sections, we introduce related works including semi-supervised and multi-label learning. Section 2.2 introduces the details of our model. Experiments and analysis are presented in section 2.3. Conclusion is provided in Section 2.6.

2.2.2 Conference Version Revisit

In this section, we first briefly present the preliminaries of multi-label learning and the framework of our previous research work, AG^{2}E [71], then we will derive our AGMA approach.

The notations utilized in this work are summarized in Table 2.1. Scale values or vectors are represented by lowercase letters and the matrices are illustrated by uppercase letters. $X_l \in \mathbb{R}^{d\times n_l}$ is the feature matrix of labeled data, where $X_l = [x_1, x_2, \cdots, x_{n_l}]$. d is the feature dimension, n_l is the sample number. $x_i \in \mathbb{R}^d$ represents a feature vector of the i-th sample. $Y_l \in \mathbb{R}^{d_l\times n_l}$ is the ground truth label matrix of X_l, where d_l is the label dimension. $Y_l = [y_1, y_2, \cdots, y_{n_l}]$ and $y_i \in \mathbb{R}^{d_l}$ represents a label vector. Similarly, $X_u \in \mathbb{R}^{d\times n_u}$ is the feature matrix of unlabeled data. F_l and F_u are the predicted label matrix of X_l and X_u. $F_l = [f_1, f_2, \cdots, f_{n_l}]$ and $F_u = [f_1, f_2, \cdots, f_{n_u}]$. In semi-supervised multi-label setting, X_l, Y_l and X_u are given. The goal of our approach is to obtain F_u as accurate as possible.

Conventional semi-supervised multi-label learning methods obtain label propagation based on a pre-defined affiliate graph [76]. This approach assumes that the pairwise samples which have high similarity scores should have similar multiple labels. In this scenario, the pre-defined affiliate graph directly determines the recovered label. However, the quality of the affiliate graph is easily affected by several aspects including different similarity metrics (e.g., Euclidean and Cosine distance), the metric configurations, and the feature/label noise. To avoid this limitation, adaptive graph-based methods are explored to automatically obtain the best graph.

Our previous work [71] learns a low-dimensional subspace to obtain distinctive representations. An adaptive affinity graph is jointly updated based on the representations. The main objective function is shown below:

$$\begin{aligned} &\min_{F,S,P} \sum_{i,j=1}^{n} \|f_i - f_j\|_2^2 s_{ij} + \mu \sum_{i,j=1}^{n} \|Px_i - Px_j\|_2^2 s_{ij}, \\ &\text{s.t. } F_l = Y_l, S \geq 0, S\mathbb{1} = \mathbb{1}, \end{aligned} \tag{2.1}$$

where $S \in \mathbb{R}^{n\times n}$ is the similarity matrix across all samples, each element s_{ij} is the obtained similarity score between x_i and x_j. $n = n_l + n_u$. The constraint $S\mathbb{1} = \mathbb{1}$ is included, where $\mathbb{1}$ is a vector of ones. It indicates the sum of the elements in each row is 1. This constraint controls the scale of S and avoids a trivial solution (i.e., $S = 0$). The negative influence from outliers could also be suppressed. In addition, instead of calculating the pairwise distances in the original feature space (i.e., $\|x_i - x_j\|_2^2 s_{ij}$), a linear projection $P \in \mathbb{R}^{r\times d}$ is deployed to project the original feature vectors to a low-dimensional subspace (i.e., $\|Px_i - Px_j\|_2^2 s_{ij}$). $F_l = Y_l$ since F_l is the given ground truth. F, P, and S are simultaneously optimized. By this way, S is adaptively learned based on both the feature similarity and label similarity to achieve higher prediction accuracy.

There are several drawbacks in [71] may still limit its potential performance. First, it is

difficult to guarantee that the learned subspace can obtain the most distinctive representations. High level noise could reduce the quality of the subspace. Second, the approach does not well solve the limited training data challenge. Third, if new/unseen samples should be predicted in the testing stage, the whole optimization procedure has to be operated again to obtain the prediction result, F. To this end, this pipeline is not efficient for large-scale applications.

2.2.3 Visual-Label Encoder via Adaptive Graph

To solve the aforementioned drawbacks, we improve the model by directly projecting the data from feature space to label space. In our new model, a projection P is trained to output the label prediction as shown below:

$$f_i = Px_i. \tag{2.2}$$

By this way, the connection between features and labels could be further tightened and it avoids the potential negative influence from the arbitrary subspace. Furthermore, we assume the predicted label vector could still recover the original features, inspired by the work of semantic autoencoder [77], we let the encoder share the same weight as P. This strategy could further help the model to reduce the computational cost and mitigate overfitting. To this end, we have

$$x_i = P^\top f_i. \tag{2.3}$$

By replacing the second term in Eq. (2.1) with Eq. (2.2) and Eq. (2.3), we can have the objective function shown below:

$$\begin{aligned} &\min_{F,P,S} \sum_{i,j=1}^{n} \|f_i - f_j\|_2^2 s_{ij} + \mu \sum_{i=1}^{n} \|f_i - Px_i\|_2^2 + \lambda \sum_{i=1}^{n} \|x_i - P^\top f_i\|_2^2, \\ &\text{s.t.} \quad F_l = Y_l, S \geq 0, S\mathbb{1} = \mathbb{1}, \end{aligned} \tag{2.4}$$

where P projects visual feature to the label/semantic space and $P^\top$ maps the predicted labels back to the original feature space. The second and the third term calculate the encoder error and decoder error respectively. λ and μ are the trade-off parameters which balance the weight between label space and visual space. S is initialized as a dense matrix in the optimization process. It gradually converged to a sparse matrix due to the constraint $S\mathbb{1} = \mathbb{1}$. The sparsity of S is influenced by the data distribution of different datasets.

To make Eq. (2.4) more compact and efficient to solve, we rewrite Eq. (2.4) as a matrix format which is shown below:

$$\begin{aligned} &\min_{F,P,S} \text{tr}(FL_SF^\top) + \mu\|F - PX\|_{\text{F}}^2 + \lambda\|X - P^\top F\|_{\text{F}}^2, \\ &\text{s.t.} \quad F_l = Y_l, S \geq 0, S\mathbb{1} = \mathbb{1}, \end{aligned} \tag{2.5}$$

where tr(.) indicates the matrix trace calculation which is the sum of the main diagonal elements. $L_S \in \mathbb{R}^{n\times n}$ is the Laplacian matrix. $L_S = D - S$ where $D \in \mathbb{R}^{n\times n}$ and $D_{ii} = \sum_{i=1}^{n} s_{ij}$. $X = [X_l, X_u]$ and $F = [F_l, F_u]$.

2.2.4 Generic Encoder Learning via Marginalized Augmentation

As mentioned in the above section, the long-tail label distribution is a common and one of the main challenges of multi-label learning tasks. Long-tail label distribution means some labels only have very limited training samples, while some labels dominate the whole label space. This challenge also suppresses the learning performance. To address this problem, we explore the idea of Marginalized Corrupted Features (MCF) [72]. It effectively extends/enlarges the feature distribution by corrupting the existing training examples with a fixed noise distribution. By this way, the feature distribution gaps between samples could be filled up.

Given a feature vector $x_i \in \mathbb{R}^d$. We let x_i^k ($k = \{1, 2, \cdots, d\}$) represent the value of each dimension of x_i. MCF assumes that the augmentation distribution factorizes over all dimensions of x_i. It considers each individual distribution as a combination of a set of natural exponential family:

$$p(\widetilde{x}_i|x_i) = \prod_{k=1}^{d} P_E(\widetilde{x}_i^k|x_i^k; \eta_k), \tag{2.6}$$

where $\widetilde{x}_i$ is the corrupted version of x_i. η_k is the augmentation distribution parameter on the dimension k. MCF constrains $\mathbb{E}[\widetilde{x}_i]_{p(\widetilde{x}_i|x_i)} = x_i$, where $\mathbb{E}(\widetilde{x}_i)$ is the expectation of $\widetilde{x}_i$. It means that the expectation of the augmented features should be the same as x_i.

In our model, all the samples from labeled and unlabeled sets are utilized to obtain the corrupted features. Given the whole samples $\mathcal{D} = [(x_i, f_i)]_{i=1}^n$, assume we augment the samples M times and obtain the augmented features $\widetilde{x}_{im}$ ($m = 1, 2, 3, \ldots, M$). Then, our model can utilize these features $\widetilde{\mathcal{D}}$ to train any classification models by minimizing the equation below:

$$\mathcal{L}(\mathcal{D}; \Theta) = \sum_{i=1}^{n} \frac{1}{M} \sum_{m=1}^{M} L(\widetilde{x}_{im}, f_i; \Theta), \tag{2.7}$$

where Θ is the model parameters with $\widetilde{x}_{im} \sim p(\widetilde{x}_{im}|x_i)$, and $L(x_i, f_i; \Theta)$ is the objective function of a proposed model. However, such approach is not elegant and could increase the computational cost significantly. To this end, the limiting case in which $M \to \infty$ can be used for Eq. (2.7) as follow:

$$\mathcal{L}(\mathcal{D}; \Theta) = \sum_{i=1}^{n} \mathbb{E}[L(\widetilde{x}_i, y_i; \Theta)]_{p(\widetilde{x}_i|x_i)}. \tag{2.8}$$

where $\mathbb{E}(.)$ is the expectation of the objective value. Minimizing Eq. (2.8) under the corruption model is the crucial module for MCF. The solution of Eq. (2.8) relies heavily on the objective function and the augmentation distributions. Coincidentally, for projections that employ exponential or quadratic objective function, the expectations in Eq. (2.8) could be obtained for all augmentation distributions in the natural exponential family [72]. To this end, we modify Eq. (2.5) based on the MCF strategy and the expression can be formulated as follows:

$$\begin{aligned} &\min_{F,P,S} \text{tr}(FL_SF^\top) + \mu\mathbb{E}\big[\|F - P\tilde{X}\|_{\text{F}}^2\big] + \lambda\|\tilde{X} - P^\top F\|_{\text{F}}^2, \\ &\text{s.t.} \quad F_l = Y_l, S \geq 0, S\mathbb{1} = \mathbb{1}. \end{aligned} \tag{2.9}$$

where $\tilde{X}$ is the corrupted features of X. We preserve the quadratic objective loss and deploy the isotropic Gaussian distribution to augment the feature with mean x_i and variance $\delta^2\mathbf{I}$. In this way, the expectation can be written as a simple case as follows:

$$\begin{aligned} \mathbb{E}\big[\|F - P\tilde{X}\|_{\text{F}}^2\big] &= P\big(\mathbb{E}[\tilde{X}]\mathbb{E}[\tilde{X}]^\top + V[\tilde{X}]\big)P^\top - 2\text{tr}\big(Y\mathbb{E}[\tilde{X}]\big)^\top P^\top + \text{tr}\big(FF^\top\big), \\ &= P\text{tr}\big(XX^\top\big)P^\top - 2\big(YX\big)^\top P^\top + \delta^2 nPP^\top + \text{tr}\big(FF^\top\big), \end{aligned} \tag{2.10}$$

where $V[\tilde{X}]$ is a diagonal matrix storing the variance of X. It is the standard l_2-regularized quadratic objective function. Combined with other terms, Eq. (2.7) can be shown as follows:

$$\begin{aligned} &\min_{F,P,S} \text{tr}(FL_SF^\top) + \mu\text{tr}(PXX^\top P^\top) - 2\mu\text{tr}(FX^\top P^\top) \\ &\qquad + \mu\text{tr}(\delta^2 nPP^\top + FF^\top) + \lambda\|X - P^\top F\|_{\text{F}}^2, \\ &\text{s.t.} \quad F_l = Y_l, S \geq 0, S\mathbb{1} = \mathbb{1}. \end{aligned} \tag{2.11}$$

Eq. (2.11) is the complete objective function of our model. Deploying MCF does not increase the computational cost significantly since the complexity of the training algorithms remains linear in n. Our model is easy to degrade to non-augmented version. From Eq. (2.11), we observe that Eq. (2.11) becomes exactly the same as Eq. (2.5) when $\delta = 0$. We will further prove the effectiveness of marginalization augmentation by tuning the value of δ.

2.2.5 Model Discussion

Compared with our previous work, AG^{2}E [71], our AGMA approach has another extra advantage. When new/unseen samples come, our model could directly infer their labels by $F_{new} = PX_{new}$, where X_{new} are the new samples and F_{new} are the predicted labels. Such a strategy avoids the optimization procedure. Although the feature distribution knowledge of new data could not be fully explored, it is an effective and efficient way and the performance is still high and stable since P is well trained. More theoretical analysis is provided in Section 2.2.7, and the empirical evaluation is shown in Section 2.3.8.

The adaptive graph strategy is also deployed in semi-supervised domain adaptation scenario. Compared with domain adaptation setting, our approach is able to handle domain shift issues between labeled and unlabeled samples, which is similar to domain adaptation approaches. While, there are several differences between them. Conventional domain adaptation approaches explicitly learn the domain-invariant representation, while our approach achieves domain adaptation by exploring the sample similarities across different domains and adjusting the similarity matrix. Moreover, conventional methods mainly diminish the domain shift only in feature space, while our approach adaptively explores the similarities in both feature space and label space.

2.2.6 Solving Objective Function

Three variables in Eq. (2.11) are required to be optimized. It is difficult to obtain an explicit solution. We adopt the Alternative Directions Method of Multipliers (ADMM) [78] to solve the problem. ADMM is driven by alternatively optimizing the equation with respect to P, S, and F. The pseudocode of the optimization procedure is provided in Algorithm 1. P_0 is the initialization of P, it is initialized based on the objective function $\min_{P_0} \|Y_l - P_0 X_l\|_F^2 + \mu_0 \|P_0\|_F^2$, where μ_0 is a trade-off parameter and empirically set to 100. Then F_u is initialized by $F_u = PX_u$. After that, ADMM is deployed to update one variable each time where other variables are fixed. All the variables are iteratively optimized until the objective value of Eq. (2.11) is convergent. We introduce the details of the optimization procedure below:

Update P: When others are fixed, Eq. (2.11) can be written as below:

$$\min_{P} \text{tr}(PXX^\top P^\top) - 2\text{tr}\left[(FX^\top)P^\top\right] + \text{tr}\left[\delta^2 nPP^\top + FF^\top\right] + \frac{\lambda}{\mu}\|X - P^\top F\|_{\text{F}}^2. \quad (2.12)$$

To obtain the optimized point, we assign the derivation of Eq. (2.12) with respect of P to

Algorithm 1. *Solution to Eq.* (2.11).

Input:

labeled and unlabeled feature matrices X_l and X_u,

label matrix of Y_l, Gaussian distribution variance δ,

trade off parameter μ, λ and convergent threshold ϵ.

Output:

The recovered label F_u, semantic projection P.

Initialization:

Train $\min_P \|Y_l - P^\top X_l\|_{\mathrm{F}}^2 + \mu\|P\|_{\mathrm{F}}^2$ and initial $F_u = P^\top X_u$,

Obtain F by concatenating Y_l and $F = [Y_l, F_u]$.

Optimization:

while not converged **do**
  Update $P_{(k+1)}$ from the solution of (2.14);
  while not converged **do**
    Update $S_{i(k+1)}$ using Eq.(2.16);
  end while
  Calculate $L_s = D_s - (S + S^\top)/2$, $D_{sii} = \sum_i (S_{ij} + S_{ji})/2$;
  Update F_u using Eq. (2.21), given others fixed;
  $k = k + 1$;
  Obtain $\mathcal{L}_k$, which is the objective value of Eq. (2.11)
  Check if $|\mathcal{L}_{k-1} - \mathcal{L}_k| < \epsilon$.
end while

zero and obtain:

$$2PXX^\top - 2\left[(FX^\top)\right] + 2\delta^2 nP + \frac{2\lambda}{\mu}F(F^\top P - X^\top) = 0, \tag{2.13}$$

then Eq. (2.13) can be simplified to the following equation:

$$(\delta^2 nI + \frac{\lambda}{\mu}FF^\top)P + P(XX^\top) = (1 + \frac{\lambda}{\mu})FX^\top. \tag{2.14}$$

Since Eq. (2.14) is a Sylvester equation, the Bartels-Stewart algorithm [79] can be deployed to efficiently solve the equation.

Update S: By ignoring other variables, Eq. (2.11) can be written as below:

$$\begin{aligned} &\min_S \mathrm{Tr}(FL_SF^\top), \\ &\text{s.t.} \quad S \geq 0, S\mathbb{1} = \mathbb{1}. \end{aligned} \tag{2.15}$$

S cannot be explicitly solved due to the two constraints $S \geq 0$ and $S\mathbb{1} = \mathbb{1}$. We optimize S row by row, based on this strategy, the equation can be written as follows:

$$\min_{S} \sum_{i=1}^{n} \|f_i - f_j\|_2^2 s_{ij} = \sum_{i=1}^{n} a_i \mathbf{s}_i^\top, \tag{2.16}$$

where $a_i = \{a_{ij}, 1 \leq j \leq n\} \in \mathbb{R}^{1\times n}$ with $a_{ij} = \|f_i - f_j\|_2^2$, $\mathbf{s}_i$ is the i-th row of S. KKT [80] approach can be used for solving this problem, then the updated graph S is obtained.

Update F: When others are fixed, the objective function can be written as follows:

$$\begin{aligned} &\min_{F} \text{tr}(FL_sF^\top) - 2\mu\text{tr}\left[(FX^\top)P^\top\right] + \mu\text{tr}(FF^\top) + \lambda\|X - P^\top F\|_{\text{F}}^2, \\ &\text{s.t.} \quad F_l = Y_l. \end{aligned} \tag{2.17}$$

Since label matrix F is the concatenation of labeled and unlabeled data (i.e., $F = [F_l, F_u]$), thus, we can decompose Eq. (2.17) and obtain the equation shown below:

$$\begin{aligned} &\min_{F_u} \text{tr}([F_l, F_u]L_s[F_l, F_u]^\top) - 2\mu\text{tr}([F_l, F_u]X^\top)P^\top \\ &\qquad + \mu\text{tr}([F_l, F_u][F_l, F_u]^\top) + \lambda\|X - P^\top[F_l, F_u]\|_{\text{F}}^2, \\ &\text{s.t.} \quad F_l = Y_l. \end{aligned} \tag{2.18}$$

Meanwhile, L_s can also be decomposed as $L_s = \begin{bmatrix} L_{ll} & L_{lu} \\ L_{ul} & L_{uu} \end{bmatrix}$. Then, Eq. (2.18) can be further decomposed as shown below:

$$\begin{aligned} &\min_{F_u} \text{tr}\left(F_lL_{ll}F_l^\top + F_uL_{ul}F_l^\top + F_lL_{lu}F_u^\top + F_uL_{uu}F_u^\top\right) - 2\mu\text{tr}\left[(F_lX_l^\top + F_uX_u^\top)P^\top\right] \\ &\qquad + \mu\text{tr}\left(F_lF_l^\top + F_uF_u^\top\right) + \lambda\|X_u - P^\top F_u\|_{\text{F}}^2, \\ &\text{s.t.} \quad F_l = Y_l. \end{aligned} \tag{2.19}$$

To obtain the optimized point, we assign the derivation of Eq. (2.19) with respect of F_u to zero and obtain:

$$(L_{ul}F_l^\top)^\top + F_lL_{lu} + F_uL_{uu} + F_uL_{uu}^\top - 2\mu PX_u + 2\mu F_u + 2\lambda P(P^\top F_u - X_u) = 0. \tag{2.20}$$

By simplifying Eq. (2.20), Bartels-Stewart algorithm [79] can be used to solve the equation:

$$(\mu I + \lambda PP^\top)F_u + F_uL_{uu} = (\mu + \lambda)PX_u - F_lL_{lu}. \tag{2.21}$$

We set a threshold ϵ, if the difference is less than ϵ, then we consider the optimization process is converged. Then we stop the process and report the final performance.

2.2.7 Complexity Analysis

In the optimization stage, updating P and F requires the Bartels-Stewart approach and the complexity becomes $\mathbf{O}(d^3)$ and $\mathbf{O}(n^3)$ respectively. These steps have more efficient solution by Coppersmith-Winograd algorithm [81] and the computational cost can be reduced to $\mathbf{O}(d^{2.37})$ and $\mathbf{O}(n^{2.37})$. To this end, the sum of the complexity is $\mathbf{O}(td^{2.37} + tn^{2.37})$ where t is the iteration number. The obtained computational cost is the cost for the whole optimization procedure. It could fully explore the data structure from both labeled and unlabeled samples. However, as mentioned in Section 2.2.5, we can utilize the learned projection P to directly infer the new/unseen samples (i.e., $F_{new} = PX_{new}$). This strategy avoids the optimization procedure which is more efficient. By this way, we reduce the complexity to $\mathbf{O}(n)$. It is more suitable for large-scale real-world applications. The empirical evaluation in Section 2.3.8 further shows the time consumption which demonstrates the efficiency of our approach in the testing phase.

2.3 Experiment

In this section, five real-world multi-label datasets are utilized. Then, we will introduce the experimental settings in our work. To comprehensively compare our approach with the state-of-the-art algorithms, we evaluate our AGMA approach as well as other baselines in both general and zero-shot multi-label learning scenario. Zero-shot setting is more challenging which attempts to recover labels from the "unseen" samples. The details will be introduced in Section 2.3.3. Finally, we analyze some properties of our proposed methods.

2.3.1 Datasets and Experimental Setting

Five multi-label datasets including one emotion dataset, one acoustic dataset, and three image datasets are explored. Brief introductions are listed as follows, and the statistical summary of the datasets is listed in Table 2.2.

- **SUN Dataset** [15] is widely used in fine-grained scene understanding and high-level scene recognition. It contains 14000 samples collected from 700 classes. Each sample has a 102-dimensional label vector which contains averagely 6.3 labels. The label value is in $\{0, 0.33, 0.66, 1\}$, since there are three annotators label each image, and the dataset averages the assigned label from all the annotators.

Table 2.2: Datasets statistical summary

Datasets	Setting	Labeled	Unlabeled	Labels	Ave
SUN [15]	General	6,387	6,513	102	6.3
	Zero-shot	12,900	1,440		
CUB [16]	General	4,374	4,468	312	31.4
	Zero-shot	8,842	2,946		
AWA [82]	General	12,154	12,141	85	15.0
	Zero-shot	24,295	6,180		
BIRD [83]	General	322	323	19	1.1
EMO [84]	General	391	202	6	1.9

- **CUB Dataset** [16] is an augmentation dataset derived from CUB-200 dataset [85]. It contains 200 categories of birds. There are 312 attribute label candidates. The elements in the label vector are binary values, i.e., 0 and 1.

- **AWA Dataset** [82] is a large-scale animal attribute datasets, where more than $30,000$ samples are collected from 50 animal categories. The label is a 85-dimensional vector with the continuous element values from 0 to 100. There are around 15 labels of each sample.

- **BIRD Dataset** [83] contains the acoustic recordings collected from 19 different kinds of bird. Each recording is around 10-seconds length. The recordings are paired with its attributes assigned by several experts along with their confidence. Each label vector contains binary value in $\{0, 1\}$.

- **EMO Dataset** [84] captures the music from 233 musical albums. It aims to test the music emotion evaluation approaches. There are 593 songs where each song is extracted to a 30-seconds recording and classified to 6 emotions assigned by music experts.

For the image datasets (i.e., SUN, CUB, and AWA datasets), Very Deep Convolutional Networks [86] pre-trained by ImageNet [87] is utilized to extract deep features. It obtains a 4096-dimensional feature vector for each instance. We also evaluate GoogleNet [88] features on these datasets and observe that different features may cause different performances, while our approach always achieves high performance. For the BIRD dataset, we use the features provided by [83]. Both the Rhythmic and the Timbre features provided by [84] are utilized for the EMO dataset.

Table 2.3: Multi-label Classification Performance

Dataset	Method	Prec	Recall	F1	N-R	mAP
SUN	Regression	0.6318±0.0070	0.1504±0.0011	0.2429±0.0016	100.0±0.0000	0.3907±0.0026
	SSMLDR	0.5625±0.0021	0.1239±0.0011	0.2031±0.0045	67.8±2.0736	0.6315±0.0038
	FastTag	0.6187±0.0251	0.1473±0.0027	0.2379±0.0083	101.0±0.4265	0.6935±0.0189
	ML-PGD	0.3218±0.0178	0.1521±0.0009	0.2513±0.0010	100.2±0.3235	0.7013±0.0016
	SAE	0.7415±0.0089	0.1976±0.0005	0.3123±0.0011	101.4±0.5477	0.6928±0.0019
	AG^{2}E	**0.7460±0.0063**	0.1625±0.0019	0.2669±0.0028	102.0±0.0000	**0.7174±0.0013**
	Ours	0.7046±0.0144	**0.2040±0.0015**	**0.3164±0.0018**	**102.0±0.0000**	0.6821±0.0028
CUB	Regression	0.2728±0.0080	0.0317±0.0007	0.0568±0.0013	166.6±1.7889	0.2831±0.0035
	SSMLDR	0.2162±0.0031	0.0399±0.0003	0.0674±0.0006	163.8±2.8636	0.2135±0.0033
	FastTag	0.3231±0.0244	0.0496±0.0028	0.0860±0.0052	163.0±4.2426	0.2457±0.0255
	ML-PGD	0.3029±0.0067	0.0448±0.0002	0.0781±0.0004	132.4±3.1937	0.4081±0.0049
	SAE	0.2947±0.0062	0.0424±0.0007	0.0742±0.0014	175.6±5.4498	0.4020±0.0027
	AG^{2}E	0.3351± 0.0079	0.0525±0.0009	0.0908±0.0015	194.2±3.1195	0.4011±0.0027
	Ours	**0.3976±0.0048**	**0.0578±0.0007**	**0.1010±0.0009**	**200.4±1.1670**	**0.4115±0.0046**
AWA	Regression	0.8198±0.0098	0.0819±0.0001	0.1489±0.0003	74.8±0.8366	0.9282±0.0003
	SSMLDR	0.8085±0.0087	0.0948±0.0002	0.1698±0.0004	74.0±0.8366	0.8323±0.0031
	FastTag	0.7848±0.0316	0.0857±0.0031	0.1545±0.0096	67.2±3.1852	0.8851±0.0183
	ML-PGD	0.5283±0.0019	0.0631±0.0001	0.1127±0.0004	44.6±1.6733	0.9103±0.0001
	SAE	**0.9506±0.0010**	0.1029±0.0005	**0.1857±0.0007**	75.2±0.8944	0.8630±0.0001
	AG^{2}E	0.7745±0.0096	**0.1285±0.0016**	0.2204±0.0027	71.8±1.0062	0.9211±0.0074
	Ours	0.9013±0.0092	0.0971±0.0018	0.1766±0.0030	**81.0±0.4472**	**0.9355±0.0073**
EMO	Regression	0.3793±0.0053	0.9114±0.0118	0.5357±0.0069	6.0±0.0000	0.5431±0.0127
	SSMLDR	0.3556±0.0048	0.8965±0.0094	0.5093±0.0078	6.0±0.0000	0.5590±0.0103
	FastTag	0.3833±0.0198	0.9459±0.0215	0.5456±0.0272	6.0±0.0000	0.5894±0.0428
	ML-PGD	0.3784±0.0079	0.9265±0.0078	0.5373±0.0090	6.0±0.0000	0.5677±0.0135
	SAE	0.3923±0.0143	0.8389±0.0083	0.5346±0.0157	6.0±0.0000	0.5770±0.0153
	AG^{2}E	0.3995±0.0122	**0.9714±0.0131**	0.5762±0.0121	6.0±0.0000	0.5825±0.0181
	Ours	**0.4474±0.0080**	0.8361±0.0230	**0.5829±0.0118**	**6.0±0.0000**	**0.5962±0.0201**
BIRD	Regression	0.0764±0.0078	0.3726±0.0367	0.1268±0.0128	12.8±0.7071	0.2364±0.0546
	SSMLDR	0.0709±0.0052	0.3465±0.0282	0.1178±0.0093	12.2±0.7071	0.1436±0.0382
	FastTag	0.1005±0.0144	0.3783±0.0421	0.1601±0.0153	15.6±1.1400	0.1643±0.0857
	ML-PGD	0.0809±0.0089	0.3883±0.0267	0.1338±0.0134	15.4±1.0000	0.2423±0.0329
	SAE	0.0964±0.0107	0.3665±0.0435	0.1526±0.0156	15.2±1.3038	0.1779±0.0480
	AG^{2}E	0.1021±0.0150	0.4529±0.0186	0.1653±0.0187	16.8±0.7786	0.2454±0.0466
	Ours	**0.1065±0.0131**	**0.5216±0.0181**	**0.1780±0.0143**	**18.0±0.0000**	**0.3519±0.0311**

Traditional multi-label scenario and the zero-shot multi-label scenario [17, 89] settings are deployed in our experiments. In the conventional setting, we randomly extract the samples from the whole datasets and build a labeled set and an unlabeled set. Each set has half of the whole sample. Our model is evaluated five times based on the randomly generated training/testing sets and report the average performance. The standard deviation is also provided. Five-fold cross-validation is deployed to tune the trade-off parameters λ and μ. The parameter sensitivity analysis will be introduced in the experiments. We evaluate our methods as well as other state-of-the-art multi-label learning methods. The brief introduction of all the baselines are shown below:

- **Least Squares Regression (Regression)** is a ridge regression approach. It obtains a projection

based on the training samples and then recovers the target samples.

- **Semi-Supervised Multi-Label Dimensionality Reduction (SSMLDR)** [76] enlarges the multiple label information from the labeled samples to the unlabeled samples. In addition, a transformation matrix is proposed to obtain the distinctive low-dimensional representations.
- **FastTag** [90] proposes two linear projections that are simultaneously optimized in a joint convex objective function. Even if the training samples contain incomplete/noisy ground truth labels, FastTag is able to effectively and efficiently predict the complete list of labels.
- **Multi-Label with a Mixed Graph (ML-PGD)** [19] designs a mixed graph which fully explores the label dependencies. It considers the co-occurrence across each pair of the candidate labels and the instance-level similarities as the graph edges.
- **Semantic AutoEncoder (SAE)** [77] proposes an effective and efficient autoencoder strategy. It recovers multiple labels without other sophisticated constraints. SAE achieves high performance in both conventional and zero-shot learning settings.
- **Adaptive Graph Guided Embedding (AG^{2}E)** [71] proposes a novel approach which simultaneously updates the affinity graph, recovers labels, and optimizes projected subspaces. It effectively overcomes the label noise and long-tail distribution issues.

We deploy the metrics utilized in [91]. Specifically, the recall R and the precision (Prec) P are obtained. $P = \frac{t_p}{t_p+f_p}$ and $R = \frac{t_p}{t_p+f_n}$, where t_p denotes True-Positive. f_n and f_p represent the False-Negative and the False-Positive respectively. We calculate harmonic mean of the precision and the recall, F1-score (F1), to compare the results easier. $F_1 = 2\frac{P \times R}{P+R}$. A non-zero recall (N-R) which denotes the number of non-zero labels are further reported. Moreover, the mean average precision (mAP) utilized in [19] is further deployed for a comprehensive evaluation. For all evaluations, higher value denotes better performance.

2.3.2 Performance Comparison

Table 2.3 shows the classification evaluations. The result illustrates the higher performance is obtained by our approach than other methods in most of the metrics. In addition, we can see that the deviations of all the evaluated methods are relatively low. Although the deviations of our approach are not the smallest, it is small enough to demonstrate the significance and stability of our method.

Table 2.4: Zero-shot multi-label learning performance

Dataset	Method	Prec	Recall	F1	N-R	mAP
SUN	Regression	0.4301±0.0083	0.1243±0.0018	0.1929±0.0023	**62.0±0.0000**	0.4142±0.0035
	SSMLDR	0.2611±0.0029	0.1055±0.0018	0.1503±0.0061	48.2±2.2893	0.3516±0.0046
	FastTag	0.3924±0.0316	0.1317±0.0042	0.1972±0.0152	60.6±3.1825	0.3775±0.0227
	ML-PGD	0.2972±0.0198	0.1138±0.0013	0.1646±0.0020	34.6±2.5273	0.5181±0.0025
	SAE	0.4838±0.0128	0.1210±0.0007	0.1943±0.0015	55.8±0.6285	**0.5357±0.0021**
	AG^{2}E	**0.4925±0.0059**	0.1235±0.0028	0.1975±0.0041	55.2±0.1685	0.5132±0.0017
	Ours	0.4710±0.0162	**0.1326±0.0017**	**0.2069±0.0020**	57.8±2.2114	0.4739±0.0031
CUB	Regression	0.2026±0.0091	0.0268±0.0009	0.0474±0.0018	143.6±1.9128	0.1982±0.0044
	SSMLDR	0.1949±0.0042	0.0360±0.0004	0.0607±0.0008	131.4±3.1010	0.2535±0.0038
	FastTag	0.2821±0.0286	0.0428±0.0033	0.0743±0.0074	143.0±3.6278	0.2229±0.0266
	ML-PGD	0.1953±0.0081	0.0357±0.0002	0.0604±0.0006	81.8±2.4681	0.3095±0.0061
	SAE	0.2206±0.0083	0.0355±0.0009	0.0611±0.0019	138.4±5.1826	0.3064±0.0035
	AG^{2}E	0.2749±0.0086	0.0415±0.0011	0.0720±0.0017	172.0±2.1983	**0.3115±0.0036**
	Ours	**0.2838±0.0062**	**0.0446±0.0009**	**0.0768±0.0011**	**172.2±1.8315**	0.3004±0.0050
AWA	Regression	0.7761±0.0151	0.0761±0.0004	0.1386±0.0007	68.4±1.0425	0.8818±0.0012
	SSMLDR	0.7380±0.0121	0.0787±0.0003	0.1423±0.0004	67.6±1.2185	0.8423±0.0082
	FastTag	0.7753±0.0451	0.0852±0.0052	0.1535±0.0165	65.8±3.8195	**0.8838±0.0267**
	ML-PGD	0.4570±0.0026	0.0607±0.0002	0.1073±0.0005	39.8±2.1066	0.8431±0.0004
	SAE	0.8914±0.0016	**0.0920±0.0007**	0.1648±0.0011	71.6±1.1528	0.8432±0.0004
	AG^{2}E	0.8810±0.0132	0.0897±0.0018	0.1511±0.0035	71.8±1.1225	0.8381±0.0093
	Ours	**0.9129±0.0129**	0.0906±0.0028	**0.1657±0.0052**	**84.0±0.6385**	0.8493±0.0085

We observe that the mAP performance is not competitive in the AWA dataset. We conjecture several reasons. First, in the AWA dataset, the samples which belong to the same class have consistent label vectors. Consider there are only 50 different label vectors corresponding to the 50 classes. The label distribution/diversity is narrow and this situation is unique in the AWA dataset. We assume it is hard for our approach to learn comprehensive distribution knowledge and augment diverse features. Second, the AWA dataset contains 24295 samples with averagely 15 labels in each sample. The dataset scale is bigger than other datasets. We conjecture that the data scale is already big enough for training a good classifier, and our model gains limited benefits from the feature augmentation strategy. Meanwhile, our model still gets the best performance in mAP metric which is considered as one of the most important metrics (i.e., F1 and mAP) for multi-label learning scenario. For the SUN dataset, we observed that the precision and mAP are not the highest performance. We assume that although the feature augmentation strategy is effective for improving the performance, the precision-recall improvement balances of different datasets are uncertain. We observe that in most of the cases either precision or recall is higher than other state-of-the-art methods. F1 metric is

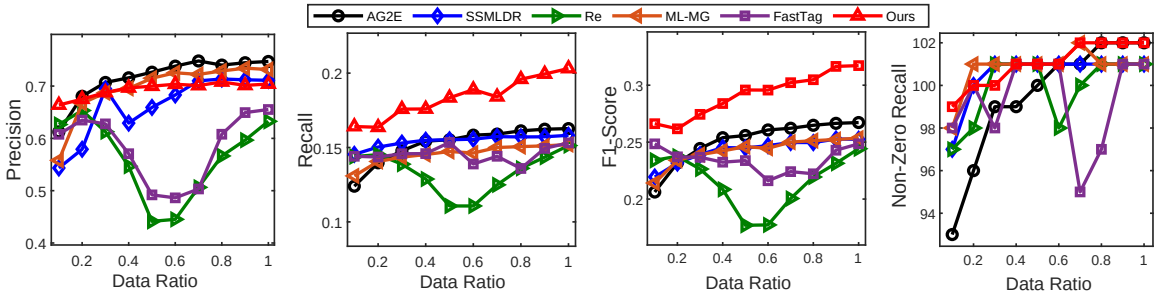


Figure 2.4: Multi-label annotation performance based on part of the available training samples. It denotes that our approach still achieves high performance when fewer and fewer labeled samples are provided in the training stage. It demonstrates the effectiveness and stability of the trained model.

Table 2.5: Ablation Study of Marginalized Augmentation Strategy

Dataset	Aug.	Prec	Recall	F1	N-R	mAP
EMO	×	0.4215±0.0071	0.8357±0.0294	0.5611±0.0152	5.0±0.0000	0.5832±0.0187
	√	**0.4474±0.0080**	**0.8361±0.0230**	**0.5829±0.0118**	**6.0±0.0000**	**0.5962±0.0201**
BIRD	×	0.1051±0.0189	0.5113±0.0201	0.1735±0.0113	17.0±0.0000	0.3391±0.0253
	√	**0.1065±0.0131**	**0.5216±0.0181**	**0.1780±0.0143**	**18.0±0.0000**	**0.3519±0.0311**
SUN	×	0.6953±0.0096	0.1914±0.0024	0.3011±0.0031	100.0±0.0000	0.6785±0.0030
	√	**0.7046±0.0144**	**0.2040±0.0015**	**0.3164±0.0018**	**102.0±0.0000**	**0.6821±0.0028**

a comprehensive evaluation which considers both precision and recall, and our method obtains the highest performance in most of the target datasets.

2.3.3 Zero-shot Multi-label Classification

More challenging zero-shot scenario is deployed for evaluating our approach. In zero-shot setting, the classes in the training set and the test set have no overlap, which means the feature distribution gaps between training and test sets are more significant. Specifically, in multi-label scenario, all the samples share the same set of multi-label candidates, while the training and test samples are extracted from non-overlapped categories (e.g., *horse* and *zebra* could be in training and test sets respectively. They share similar shape labels but different color/texture labels). SUN, CUB and AWA datasets have the default splits for zero-shot scenario. Specifically, in the SUN dataset, it contains 645 training classes and 72 test classes. In CUB dataset, 150 bird categories are used for training and the rest 50 categories are used for testing. Moreover, AWA dataset consists 40 training classes and 10 test classes. The detailed sample numbers are further summarized in Table 2.2.

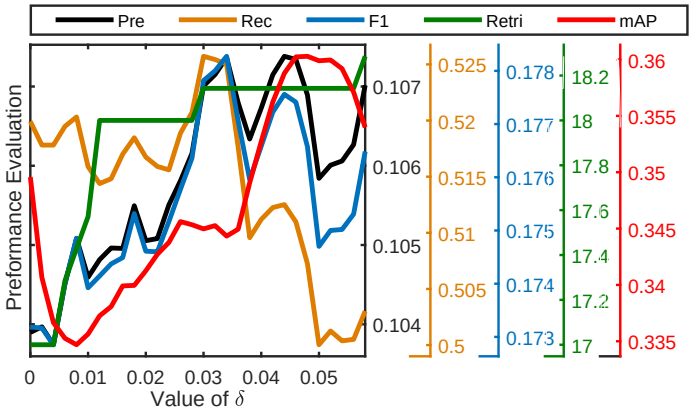


Figure 2.5: Learning performance based on different values of the Gaussian distribution variance δ for feature augmentation. Different colors indicate the five metrics respectively. The result shows that almost all the metrics improve as δ increases. It demonstrates the effectiveness of MCF module.

The same evaluation metrics as a general multi-label task are deployed and the results are illustrated in Table 2.4. We can observe that our approach achieves higher performance compared with other baselines. The result illustrates the ability of our approach for handling domain shift scenario, where labeled and unlabeled samples are shifted. The standard deviations are still small while slightly higher than conventional multi-label setting. We assume it is due to the larger distribution gap across training and test data in zero-shot scenario.

2.3.4 Model Robustness Analysis

To estimate the robustness of our model, we use only partial samples from the labeled set (from 10% to 100%) and the final results are shown in Figure 2.4. From Figure 2.4, we observe that our approach is still able to secure the high performance even only 20% labeled samples are provided, and it achieves the highest performances in most of the metrics when the ratio is from 20% to 100%. The results prove the robustness of our model with limited samples.

2.3.5 Marginalized Feature Augmentation

To demonstrate the effectiveness of the marginalized augmentation strategy, we evaluated the performance with and without augmentation module. As we discussed in section 2.2.4, our model can degrade to a non-augmentation version when the variation of the augmented feature distribution, δ, is reduced to zero. To this end, we tested the performance with and without it by tuning $\delta = 0$, and the result is shown in Table 2.5. Moreover, we gradually increase δ value and report the performance.

The results in the BIRD dataset are illustrated in Figure 2.5. From the results, we observe that as δ increases, almost all the metrics have some improvements. This result demonstrates the effectiveness of the marginalized augmentation for improving the performance. In the experiments, we notice that the same type of feature achieves the highest performance based on the same δ, and different features require different δ. We utilize cross-validation to tune δ and report the performances. In addition, we observe that the performances of different values of δ are relatively independent to other variables (i.e., μ and λ). Therefore, we tune δ after other parameters are tuned. It is a more practical strategy in real-world applications.

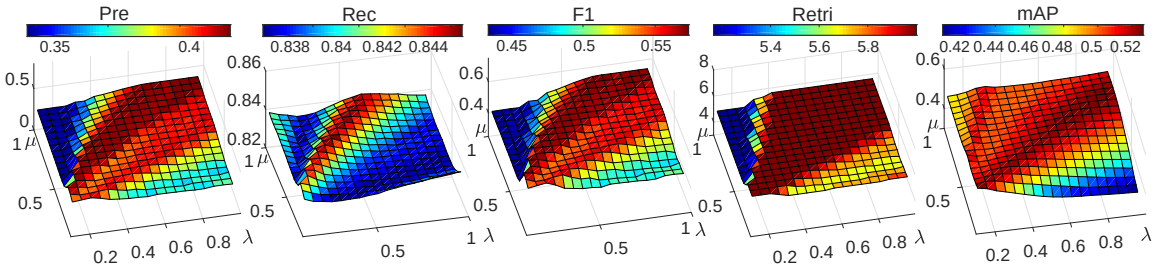


Figure 2.6: Parameter sensitivity analysis of μ and λ. The much "redder" of the color indicates the higher of the performance, and vice versa. From the results we observe that there are a wide range of values which could make our model achieve the best performance. It proves the effectiveness and robustness of our model. In real applications, cross-validation can be utilized for parameter tuning.

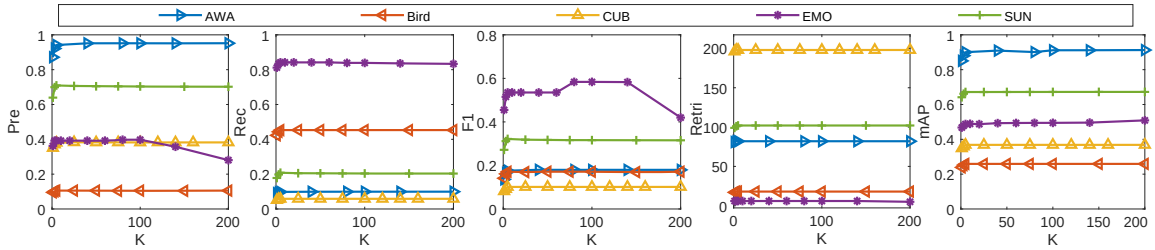


Figure 2.7: Parameter sensitivity analysis of the graph nearest neighbor K in S optimization procedure. It shows that instead of optimizing S for all sample pairs, calculating several nearest sample pairs could achieve the similar performance. It indicates the robustness of adaptive graph and we can further reduce the computational complexity in training phase by reducing the value of K.

2.3.6 Parameter Sensitivity

We further visualize the performance based on different values of μ and λ to analyze the parameter sensitivity. The result is shown in Figure 2.6. The color scale bar from blue to red indicates

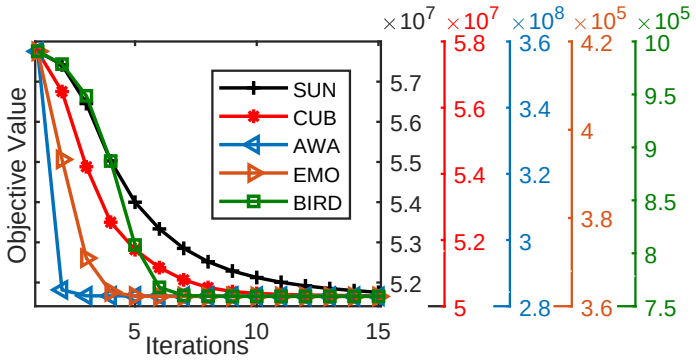


Figure 2.8: The value of objective function, Eq. (2.11)), as iteration increases. Different color denotes different datasets, and all values converged after 10-15 iterations. The result illustrates the convergence of the optimization procedure.

the performance from low to high. From Figure 2.6, we can obtain two conclusions. First, both μ and λ could affect the performance. Second, there is a large region (i.e., red region) in the visualization result where μ and λ are roughly equal to each other. This configuration usually leads to the best performance. In our parameter tuning process, we usually set one parameter fixed (e.g., $\mu = 1$) and utilize cross-validation strategy to tune the value of λ. Based on our observation, this strategy could achieve the best performance for all the datasets.

There is another hyper-parameter K in the our model, which denotes the number of the nearest sample points in the feature space. We observe that most low-similarity pairwise samples have the similarity value close to zero and they have almost no influence to the final obtained S. In our implementation, we update S based on the nearest K pairwise samples. To prove this, we evaluate the performance with different $K = [0, 200]$ in Figure 2.7. It shows that the performance drops considerably when only a few (i.e., 0, 1, or 2) of the nearest neighbors are utilized for updating the adaptive graph. Meanwhile, $K > 200$ seems to have no distinctive negative influence to most of the datasets. We observed that most of the elements in S are very close to 0, which means S is usually sparse after the optimization procedure. Thus, K does not have any negative influence on the final performance if K is great enough. From the result, we conclude that $K \geq 30$ is an appropriate value for most cases, and we do not need further parameter tuning for K, which reduces the unnecessary calculation on updating S without loss the performance for the final prediction.

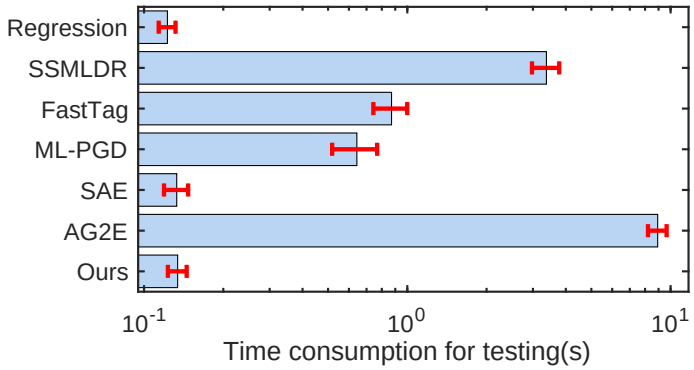


Figure 2.9: Time consumption of all methods in the testing stage. It illustrates that our approach is one of the most efficient methods which is suitable for large-scale applications.

2.3.7 Convergence Analysis

In the training stage, we utilize the Alternating Direction Method of Multipliers (ADMM) [78] algorithm for solving the objective function. Specifically, the three target variables are alternatively optimized to its optimal point until the final objective loss is converged (i.e., Eq. 2.11). Considering multiple variables are optimized independently in the training stage, thus, it is difficult to theoretically guarantee the obtained solution is the global optimal point. In practice, we empirically analyzed the global convergence of our approach. The objective function value of Eq. (2.11) is shown in Figure 2.8 as the ADMM iteration increases, and different colors denote all five datasets. From Figure 2.8, we observe that the objective function values significantly decrease in the first 10 iterations and become stable afterward. The result empirically indicates that our optimization strategy is effective and could converge in most real-world datasets.

2.3.8 Time Consumption

The time consumption of each method is illustrated in Figure 2.9. We can see from the results that our model associated with SAE [77] and regression approaches are the most efficient approaches. The main explanation is that although in the training stage, our approach requires to alternatively optimize all the variables including P, S, and F. While, after the training procedure is finished, our approach could directly utilize the learned projection P to project new/unseen samples between visual and semantic/label spaces (i.e., Eq. (2.2)). By this way, the inferring process could be degraded to a matrix multiplication operation without any extra computational costing calculations (e.g., eigen-decomposition). The complexity is $\mathbf{O}(n)$ where n is the input sample

numbers. Our previous work, AG^{2}E [71], requires to update the entire adaptive graph based on labeled and unlabeled samples, which is both space and computational costly.

Figure 2.10: Case study of the label prediction results from the SUN dataset. Black font means correct prediction and red font means incorrect prediction. In addition, blue font indicates the "correct" prediction based on our judgments while are missing in ground truth.

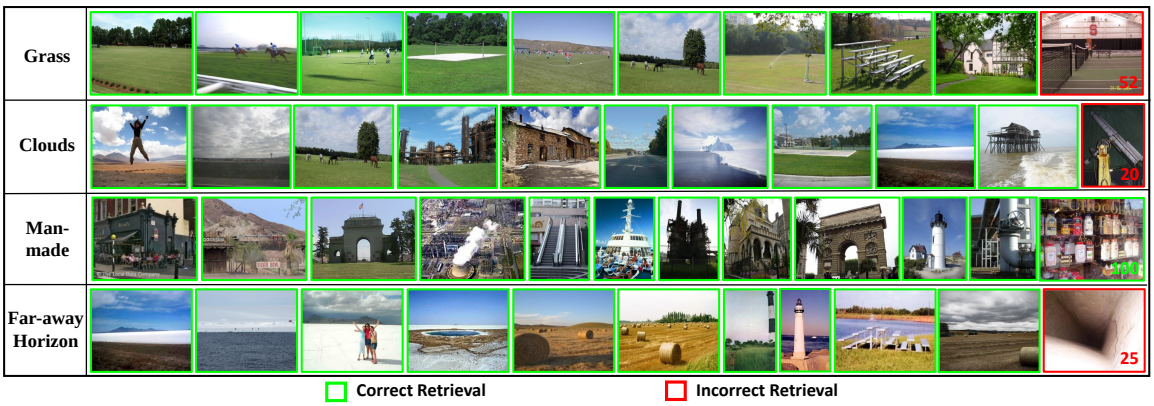


Figure 2.11: Zero-shot image retrieval result from SUN dataset. Given a target retrieval label, the samples in the testing set which have the highest prediction score are selected. Green and red boxes are the correct and incorrect retrievals. The numbers in right indicate the rankings of the samples.

2.3.9 Image Annotation

Image annotation setting is evaluated in the SUN dataset. Figure 2.10 listed the sample images as well as the corresponding predicted labels. Different colors indicate different prediction results. Considering some samples have a large number of labels, we only list the top 15 labels for discussion. In Figure 2.10, the red font is the incorrect prediction and the black font is the correct prediction. Blue font indicates the "correct" prediction based on our judgments while missing in ground truth. Figure 2.10 illustrates that most of the prediction results are correct and our model is

able to reveal several "missing" labels. From Figure 2.10, we observed that most of the predictions are correct, and our model could further explore extra missing labels compared with ground truth. The result demonstrates the efficiency and effectiveness of our method.

2.3.10 Image Retrieval

Image retrieval aims to retrieve specific images from a set of images [92]. This is deployed in a wide range of real-world applications such as searching, recommendation, and captioning. In our implementation, the obtained P assigns labels to the candidate images. The candidate images are ranked based on the prediction confidence. Zero-shot setting is utilized which means the target image categories are unseen in the training stage. The retrieved samples are listed in Figure 2.11. Each row shows the retrieval label and the obtained images. The images with green and red boxes are the corrected and incorrect retrieval. We observe that our model effectively retrieves the target images even based on the target label even if the image categories are unseen in the training stage.

2.4 Deep Learning-based Multi-label Learning

2.4.1 Motivation

As deep learning-based approaches achieved significant improvement in machine learning tasks, it is rational to explore the deep neural networks for solving the challenges in multi-label learning scenario. However, compared with conventional machine learning approaches, deep learning requires large-scale training samples to well train the neural network, and this drawback is more significant for multi-label setting due to the long-tail label distribution and the high-level label noise.

Supervised approaches need a large amount of labeled training samples to reach high performance. Compared with the single-label scenario, the multi-label scenario is more challenging [14]. For instance, if there is a large number of label candidates, the task would become difficult since the number of possible label sets will become tremendous. And the combinations across various labels. MEFF [93] utilizes a multi-view fusion approach for multi-label classification. Modulation approach is proposed in [94] for encouraging the coupling of relevant tasks for image retrieval. However, the scales of multi-label datasets [95, 96, 16, 15] are always limited which reduces the potential of the approaches. Semi-supervised learning [97, 60] is able to explore a more compatible model by making use of a small scale labeled dataset as well as a large scale unlabeled data [20, 98]. However, the performances of these kinds of approaches significantly rely on the quality of the auxiliary data and

the optimization process is complicated which is hard to control [99, 100]. Moreover, in multi-label scenario, label correlation is crucial and important to further improve the learning performance. [19, 101, 102] builds a semantic label hierarchy as prior knowledge to generate a label dependency graph. [18] utilizes a label semantic structure to deduce label noise and cover diverse and distinct labels. Label embedding [103] projects labels into a latent space to explore the label relations. [104] uses attention and RNN based approach to obtain the object relations in an image space. However, these approaches require the pre-defined label correlation information which is expensive and difficult to obtain.

Generative Adversarial Networks (GAN) [105] explores the feature distribution from real samples and generates diverse samples as real as possible. Specifically, GAN model contains two neural network structures: First, a generator network is trained to generate fake samples and confuse a discriminator network. Second, the discriminator tries to differentiate the real and generated samples. The generator and the discriminator which are trained in opposition to one another. The competition between the two networks lets both of them enhance their abilities until the fake samples are indistinguishable. Many variations of GAN are proposed for various goals and applications. Least Squares Generative Network [106] adopts the least squares loss objective for the discriminator. It overcomes the vanishing gradient challenges during the training process. Mode Regularized GAN [107] introduces several ways of regularizing the objective function, which can dramatically stabilize the training of GAN models. Cycle GAN [108] proposes a structure which translates and utilizes the absence of paired examples for source and target domain translations. GMVAR [55] generates samples of different views for multi-view classification task. Conditional GAN (CGAN) [109, 110] extends GAN strategy by adding conditional knowledge, such as classification labels, on both the discriminator and generator. ACGAN [111, 112] is proposed based on CGAN but is specifically associated with an auxiliary classifier, which is utilized to guide and stabilize the training process for the generator. However, CGAN and ACGAN were mainly designed to subjectively diversify images and utilize the human perceptual model MS-SSIM [113] to evaluate the generation diversity. It is not designed for objective classification purposes. Conditional Loss-Sensitive GAN [114] designs a loss function to make the fake images more real and can also classify target images. However, it is designed for single label classification and is difficult to extend to the multi-label scenario since it utilizes optimization strategy to classify.

In this thesis, we explore the idea of the Generative Adversarial Network (GAN) [105] and propose a novel Generative Correlation Discovery Network (GCDN) for multi-label learning. The framework of GCDN is shown in Figure 2.12. GCDN belongs to a supervised learning scenario. It

automatically learns the feature distribution from the training data and even the visual components across different samples. Our approach is able to span the feature area and overcome the limited training data scenario. In addition, inspired by the use of graph strategy in deep model [115], such as Graph-based CNN [116]; in our work, we design a simple but effective Correlation Discovery Network (CDN) to learn the correlation among different labels. Different from previous works, our approach is proposed to explore the generative model in the multi-label learning scenario. Specifically, our model applies to multi-label classification rather than single-label classification. The model builds connections across labels and features, and is designed to increase the visual feature diversity for boosting learning performance rather than increasing diversity for the subjective human perceptual evaluation [113].

In summary, GCDN captures the feature distribution of each label, and generates fake features, which completes the distribution to obtain more general samples. Meanwhile, GCDN learns the correlations across different labels and takes advantage of the learned semantic structure knowledge to significantly improve the learning performance. Our main contributions are listed as follows:

- A specifically-designed multi-label conditional feature generative strategy is proposed. It synthesizes and diversifies the feature space to improve the model robustness and generalization.
- A graph-based Correlation Discovery Network (CDN) is proposed to automatically learn semantic correlations across different labels and utilize the knowledge to further improve learning performance.
- A similarity constraint is deployed associated with the multi-label prediction to stabilize the generator training, which is effective in multi-label learning scenario.

All designed networks are trained simultaneously in an end-to-end scenario without other semantic information as prior knowledge, which is easy to deploy to a wide range of potential multi-label learning and relevant applications.

2.4.2 Preliminaries

Given the multi-label training data $\{X_l, Y_l\}$, $X_l \in \mathbb{R}^{d \times n_l}$ is the feature matrix, where each column $x_i \in \mathbb{R}^d$ represents one instance, n_l is the instance number, and d is the feature dimension. $Y_l \in \mathbb{R}^{d_l \times n_l}$ is the label matrix, where d_l is the dimension of the label. Each column y_i denotes the corresponding multi-label vector of x_i. Generally, our approach aims to train based on $\{X_l, Y_l\}$

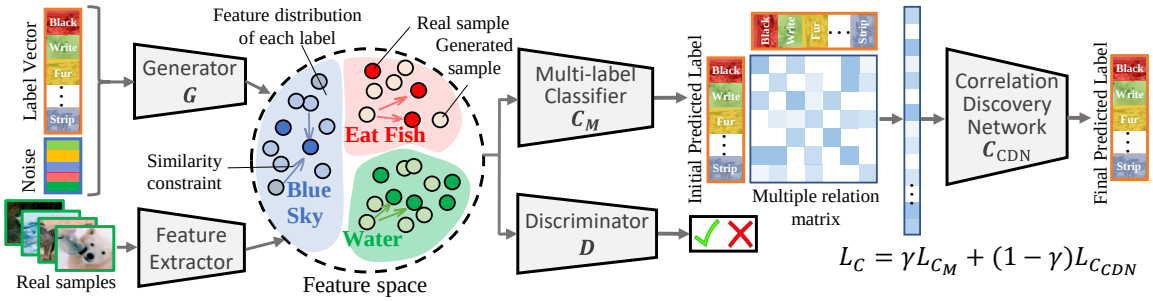


Figure 2.12: Framework of our approach, where a generator $G(\cdot)$, a discriminator $D(\cdot)$, and a multi-label classifier $C_M(\cdot)$ are simultaneously trained. The generator synthesizes augmented samples conditioned on the provided labels to handle the limited data and long-tail label distribution drawbacks; while the classifier predicts initial multi-label results, and the results are transferred to the correlation discovery network to learn correlations and obtain final high accuracy results. All networks are jointly trained in an end-to-end scenario to achieve the highest performance.

without any other prior knowledge, and predict the multi-label Y_u of X_u. Since the feature space is much more diverse than the label space, thus, it is challenging to collect enough labeled visual data to capture the data variance. Moreover, there are sophisticated correlations residing across different labels. It is useful and crucial information to further improve the learning performance, but it is difficult and expensive to obtain.

To this end, we aim to compensate for the visual feature and mitigate the gap between the training and testing samples. Inspired by the idea of generative model, it is natural to synthesize more diverse features conditioned on each multi-label vector. Meanwhile, a simple but effective graph structure is proposed to automatically explore the label correlation knowledge to further improve the learning performance. These two parts are crucial to improve the learning performance, since it allows the model to fully utilize the feature-label mapping and label-label correlation knowledge from both the feature space and label space of the training samples.

2.4.3 Multi-label Generation

Figure 2.12 illustrates the structure of our approach. It contains a generator $G(\cdot)$, a discriminator $D(\cdot)$, a multi-label classifier $C_M(\cdot)$, and a correlation discovery network $C_{CDN}(\cdot)$. $X_g = G(z|Y)$, where Y is the label matrix for conditionally generating samples and z is the random noise. The $D(\cdot)$ outputs the probability of the samples being real or fake. The generator captures the feature distribution of the existing data and borrows the shared components from other categories.

$C_{CDN}(\cdot)$ further learns the label correlation and helps to improve the final label prediction. The objective function of $D(\cdot)$ is shown in Eq. (3.4) which manages to maximize L_D:

$$L_D = E_{X\sim p_X(X)} \log D(X|Y) + E_{z\sim p_z(z)} \log(1 - D(G(z|Y))), \tag{2.22}$$

where $D(\cdot)$ is a three-layer network including a fully connected layer with ReLU activation, a mini-batch [117] layer with the LeakyReLU [118] activation, and a fully-connected layer with Sigmoid function. Multi-label classifier $C_M(\cdot)$ includes two objectives. The first one is trained based on real samples, while the second one is based on the generated samples associated with the conditional labels to improve the robustness and generalization of the classifier. The objective function is shown as follow:

$$L_{C_M} = \mu\|Y - C_M(X)\|_{\mathrm{F}}^2 + (1-\mu)\|Y - C_M(G(z|Y))\|_{\mathrm{F}}^2, \tag{2.23}$$

where μ is the trade-off parameter which is used to balance the weights between real and fake samples. μ is empirically set to 0.5 in our implementation, which expects both the real and fake samples are evenly utilized for training. And it also avoids extra parameter tuning. Meanwhile, we have observed that slightly tuning μ near 0.5 does increase the performance a little, and cross validation could be employed for automatic parameter tuning. $C_M(\cdot)$ is a two-layer network with ReLU activation in the hidden layer and a Sigmoid in the output layer. We observe that two layers are enough for label prediction, and the model is not sensitive to the number of layers. For discriminator, we include more constraints. The first term is the major competing component with $D(\cdot)$ and makes the generated samples as real as possible:

$$L_{Gd} = -E_{z\sim p_z(z)} \log(1 - D(G(z|Y))). \tag{2.24}$$

Compared with single label learning, multi-label learning provides more abstract information for each sample. Inspired by ACGAN [111], we further utilize the classification results as another clue to stabilize the generator training:

$$L_{Gc} = \|Y - C_M(G(z|Y))\|_{\mathrm{F}}^2. \tag{2.25}$$

Moreover, considering the various feature distributions across labels, the proposed terms may not be strong enough to achieve stable and robust generation performance. Thus, we further include similarity constraint which pulls the generated samples and real samples to be similar:

$$L_{Gs} = \|G(z|Y) - X\|_{\mathrm{F}}^2. \tag{2.26}$$

After combining all the objectives together and the generator loss is shown as follow:

$$L_G = L_{Gd} + \alpha L_{Gc} + \lambda L_{Gs}, \tag{2.27}$$

where α and λ are the trade-off parameters which balance the scales across binary discriminator loss, multi-label space, and visual feature space. The major goal of L_{G_s} is to stabilize the training process, and we could tune λ to balance the strength of L_{G_s}. We did observe that large λ decreased the final performance, while a small-scale λ on L_{G_s} indeed reduced training fluctuation and sped up the training process. $G(\cdot)$ is a two-layer neural network in addition with a batch-normalization layer [119] to normalize input vectors and improve model robustness.

2.4.4 Correlation Discovery Network

Simply deploying GAN model is not enough to achieve the highest performance. As introduced before, label correlation is crucial to further improve learning performance. Thus, we propose a simple while effective Correlation Discovery Network (CDN), $C_{CDN}(\cdot)$, to automatically explore the label correlation knowledge. After the predicted label $f_{ci} = C_M(x_i)$ is obtained, where $f_{ci} \in \mathbb{R}^{d_l \times 1}$ is the prediction of each instance x_i. We make a transformation from f_{ci} to an adjacency matrix $\mathbf{m_{ci}}$ by multiplying f_{ci} and its transposition as $\mathbf{m_{ci}} = f_{ci} \times f_{ci}^{\top}$, where $\mathbf{m_{ci}} \in \mathbb{R}^{d_l \times d_l}$ is the adjacency matrix and d_l is the label dimension. The obtained $\mathbf{m_{ci}}$ is reshaped to a $\mathbb{R}^{d_l^2 \times 1}$ vector and forwarded to a fully connected layer network and further predicts the multi-label result. To this end, the objective of CDN is shown below:

$$L_{C_{CDN}} = \sum_{i=1}^{n_l} \|y_i - C_{CDN}(C_M(x_i)C_M(x_i)^{\top})\|_2^2, \tag{2.28}$$

where $y_i \in \mathbb{R}^{d_l \times 1}$ is the corresponding multi-label vector of x_i. In this network, the elements in $\mathbf{m_{ci}}$ are the multiplication of each pair of the predicted labels of f_{ci}, which could be considered as a similarity metric of the pairwise labels (including the similarity with itself). CDN is trained based on the similarities structure. By this way, CDN explores the latent correlation knowledge residing inside the training data based on the obtained similarities, and further refines the predicted label from $C_M(.)$ to improve performance.

In summary, $C_M(\cdot)$ obtains initial (low-accurate) results first, then $C_{CDN}(\cdot)$ further utilizes the available prediction to "tune" the result to high-accurate. Specifically, $C_{CDN}(.)$ can be considered as a refining strategy over $C_M(.)$. It explores the latent structure knowledge (correlation) across labels and further improves the prediction performance. Jointly optimizing $C_M(.)$ and $C_{CDN}(.)$ by

combining their losses together could 1) control the training of $C_M(.)$ to predict rough labels and 2) intentionally force $C_{CDN}(.)$ to capture the label correlations based on the rough labels from $C_M(.)$. This strategy balances the update processing between $C_M(.)$ and C_{CDN} to further help each other in the training stage and achieve the promising performance at last. To this end, the objective function is shown as below:

$$L_C = \gamma L_{C_M} + (1-\gamma) L_{C_{CDN}}, \tag{2.29}$$

where $\gamma \in [0, 1]$ is the trade-off parameter which is used to balance the weights of the two objective terms. We empirically set $\gamma = 0.5$ for the experiments, and its parameter sensitivity will be analyzed in the following sections.

In our implementation, $C_{CDN}(\cdot)$ is a fully-connected two-layer network with ReLU activation in the first layer and Sigmoid activation before output. Considering $\mathbf{m_{ci}}$ is a symmetric matrix, thus, to reduce the redundant weights, we remove almost half of the $\mathbf{m_{ci}}$ and forward to $C_{CDN}(\cdot)$. This strategy improves model efficiency without losing any information, and the input dimension of $C_{CDN}(\cdot)$ becomes $(d_l^2 + d_l)/2$.

2.4.5 Model Discussion

Our proposed model contains three networks jointly optimized in a minimax strategy, which brings in several advantages. First, it is an end-to-end framework without the requirement of any other prior knowledge (e.g., semantic label hierarchy), which is easy to train and compatible for a wide-range of applications; second, the learning performance is robust. Since the generated data enlarges and diversifies the feature distribution, which effectively reduces the over-fitting issue; third, other than the discriminator, the classifier as well as the similarity constraint further guide the generator optimization process and make the training process be efficient and stable; fourth, the GCDN can be directly deployed for testing without any other optimization operations which is more simple and efficient in inference compared with graph-based approaches. To this end, our model jointly trains the components and enables each component to benefit others. The experiments demonstrate its necessity in multi-label scenario.

Compared with the conventional generative model, our approach is different in the following aspects. First, our model conditions multi-label information (either binary or continuous values) to synthesize the visual features which is more challenging than the single-label generation scenario. Second, the label correlation knowledge is automatically learned in the training procedure without any extra semantic knowledge, which is more compatible in a wide range of real-world

applications. Third, our model generates samples in feature space instead of image space, thus, it is not only limited for image level application, but also potentially works well with other data types (which is demonstrated in experiments).

2.5 Experiment

We evaluate our approach associated with the state-of-the-art approaches on six fine-grained datasets. We further extend the experiments to zero-shot multi-label learning, image annotation, as well as image retrieval scenarios in the experiments.

2.5.1 Multi-label Datasets

Six image datasets collected from different data formats are utilized for evaluation. Brief introductions of the datasets are as follows:

- **Corel5K Dataset** [95] is a subset from the Corel Photo CD dataset. It contains $4,500$ images assigned for training and 499 images assigned for testing. Each label is a 260-dimensional semantic description vector in binary format. The average descriptions per sample is 3.40.

- **ESP Game Dataset** [96] is labeled by an ESP interactive system, which is designed like a computer game in the labeling process. It includes $18,689$ samples assigned for training and $2,081$ samples assigned for testing. The label vector is a 268-dimensional vector in binary value. On average, each sample is assigned with 4.69 labels.

- **IAPRTC-12 Dataset** [120] CLEF cross-language dataset which is generated for image retrieval task. It has $19,627$ samples including landscapes, animals, actions, etc. $17,665$ samples are assigned for training and $1,962$ samples are assigned for testing. The label vector is 291-dimension in binary format with averagely 5.72 labels.

- **SUN Dataset** [15] is a scene multi-label database including images such as *bakery*, *ballroom*, and *balcony*. There are 717 scene classes in total. Each instance contains a 102-dimensional label vector in a continuous value format, ranged between $[0, 1]$ assigned by multiple trained labors, with averagely 6.31 labels per sample. There are $12,900$ samples for training and $1,440$ samples for testing.

Table 2.6: multi-label learning performance

Data	Method	Pre	Rec	F1	N-R	mAP	Data	Method	Pre	Rec	F1	N-R	mAP
	LR	0.2859	0.3211	0.3025	128	0.3630		LR	0.6209	0.1473	0.2457	102	0.6807
	SSMLDR	0.2741	0.3366	0.3022	143	0.3410		SSMLDR	0.6879	0.1700	0.2726	102	0.6723
	FastTag	0.3123	0.3657	0.3369	143	0.3871		FastTag	0.6816	0.1473	0.2457	102	0.6914
Corel	ML-PGD	0.2575	0.2911	0.2732	122	0.3727	SUN	ML-PGD	0.7110	0.1614	0.2631	101	0.7087
	SAE	0.2962	0.3442	0.3184	141	0.3823		SAE	0.7183	0.1638	0.2668	98	0.7012
	AG2E	0.3011	0.3520	0.3245	**157**	0.3568		AG2E	0.7685	0.1765	0.2871	99	0.6778
	Ours	**0.3335**	**0.3714**	**0.3514**	148	**0.4417**		Ours	**0.7985**	**0.1835**	**0.2985**	**102**	**0.7093**
	LR	0.3793	0.2038	0.2653	215	0.3440		LR	0.2010	0.0239	0.0428	157	0.0638
	SSMLDR	0.3298	0.1885	0.2399	226	0.3156		SSMLDR	0.3410	0.0473	0.0832	178	0.2329
	FastTag	0.4011	0.1927	0.2617	208	0.3904		FastTag	0.2147	0.0359	0.0615	167	0.3144
ESP	ML-PGD	0.3239	0.2012	0.2482	210	0.4077	CUB	ML-PGD	0.3334	0.0451	0.0794	155	0.3288
	SAE	0.3861	0.1743	0.2402	194	0.3842		SAE	0.3383	0.0514	0.0908	196	0.3255
	AG2E	0.3548	0.1525	0.2133	213	0.3730		AG2E	0.3409	0.0531	0.0911	190	0.3106
	Ours	**0.4032**	**0.2178**	**0.2828**	**239**	**0.4327**		Ours	**0.3718**	**0.0541**	**0.0944**	**214**	**0.3561**
	LR	0.4287	0.2041	0.2765	199	0.4211		LR	0.8798	0.0821	0.1500	75	0.8626
	SSMLDR	0.3491	0.2520	0.2927	229	0.3981		SSMLDR	0.7812	0.0858	0.1546	67	0.8346
	FastTag	0.4346	0.2267	0.2980	227	0.4596		FastTag	0.7861	0.0949	0.1694	72	0.8791
IAP	ML-PGD	0.4132	0.2441	0.3011	230	0.4674	AWA	ML-PGD	0.5395	0.0635	0.1136	57	0.9121
	SAE	0.3537	0.2282	0.2774	213	0.4309		SAE	0.9683	**0.0957**	**0.1742**	73	**0.9397**
	AG2E	0.3829	0.2330	0.2897	229	0.4353		AG2E	0.8483	0.0827	0.1507	73	0.9033
	Ours	**0.4732**	**0.2648**	**0.3396**	**237**	**0.5295**		Ours	**0.9716**	0.0871	0.1599	**83**	0.9291

- **CUB Dataset** [16] has 200 birds. Each instance has roughly 31.39 annotations in a binary 312-dimensional label vector. There are several options to split the images for training and testing with roughly $8,800$ samples for training and $1,440$ samples for evaluation.

- **AWA Dataset** [82] consists of more than $30,000$ images captured from 50 animal species. The label vector is a 85-dimensional vector and each instance has roughly 15 labels. Different from other datasets, the label vectors are continuous values that range from 0 to 100. There are $24,295$ samples for training and $6,180$ samples for testing.

2.5.2 Experimental Setup

In our implementation, all three networks are fully connected networks. Other sophisticated deep networks can also be applied to attain higher performance. For ESP Game, IAPRTC, and Corel5K datasets, we utilize 15 different visual descriptors, which are extracted by [91]. For AWA, CUB, and SUN dataset, due to the limited training data which is difficult obtain a well trained convolutional neural network from scratch; hence, the pre-trained VGG Networks [86] based on ImageNet [87] is deployed to extract deep visual features. As shown in Figure 2.12, the label vector concatenated with random noise is set as input to $G(\cdot)$. α is empirically set to 0.01. λ limits the

feature scales which is set to 5 for VGG [86] features and 20 for handcrafted features [91]. ADAM optimizer [121] is employed and the learning rates are set to 0.00002, 0.00002, 0.00005, and 0.001 for $C_M(\cdot)$ and $C_{CDN}(\cdot)$, $D(\cdot)$, and $G(\cdot)$, respectively. In the training procedure, $C_M(\cdot)$ and $G(\cdot)$ are pre-trained to have stable initialization, while $G(\cdot)$ is optimized by $L_G = L_{Gc} + \frac{\lambda}{\alpha} L_{Gs}$ without including L_{Gd} at first, and after around 50 epoch, we switch L_G back to Eq. (4.2) and train $D(\cdot)$ simultaneously with the other networks. The same number of generated and real samples are utilized in each training iteration. We randomly separate the samples into a training and a testing subset with relatively even sample numbers and run our model 5 times and report the average performance. The model is implemented on TensorFlow and trained with Nvidia Titan XP GPU for acceleration. The regular training time is around 20 minutes for model convergence.

2.5.3 Multi-label Classification

For the multi-label classification scenario, we evaluate our approach on two settings: (a) Conventional multi-label learning; (b) Zero-shot multi-label learning which is a more challenging task. We compare our approach with several state-of-the-art representative multi-label learning approaches. Brief introductions of the methods are listed below:

- **Least Square Regression (LR)** is a straightforward linear regression model, which learns a linear mapping between the feature and label spaces.
- **Semi-Supervised Multi-Label Dimension Reduction (SSMLDR)** [76] effectively utilizes the information from both labeled and unlabeled data by designing a special label propagation strategy to improve the model's robustness and accuracy.
- **Fast Image Tagging (FastTag)** [90] proposes two co-regularized linear mappings in one loss function. It is able to infer the full list of tags based on the incomplete ground truth training labels.
- **Multi-Label learning using a Mixed Graph (ML-PGD)** [19] proposes a label dependencies model by constructing a mixed graph and combines instance level similarity with class co-occurrence.
- **Semantic AutoEncoder (SAE)** [77] proposes an effective auto-encoder with an additional reconstruction constraint to recover labels.

Table 2.7: multi-label learning performance on augmented label sets

Data	Methods	Pre	Rec	F1	N-R	mAP	Data	Methods	Pre	Rec	F1	N-R	mAP
Corel	LR	0.2842	0.2304	0.2545	103	0.3762	ESP	LR	0.3848	0.1256	0.1894	178	0.3913
	SSMLDR	0.3036	0.2791	0.2908	134	0.3660		SSMLDR	0.3253	0.1697	0.2231	202	0.3357
	FastTag	0.3329	0.3145	0.3234	136	0.4127		FastTag	0.3886	0.1531	0.2197	196	0.4254
	ML-PGD	0.3245	0.3011	0.3124	140	0.4275		ML-PGD	0.3713	0.1184	0.1795	162	0.4211
	SAE	0.3168	0.3037	0.3101	128	0.4192		SAE	0.3153	0.1425	0.1966	156	0.4050
	AG2E	0.3273	0.3172	0.3221	**143**	0.3985		AG2E	0.3518	0.1492	0.2095	196	0.4030
	Ours	**0.3438**	**0.3219**	**0.3325**	138	**0.4773**		Ours	**0.4772**	**0.1944**	**0.2763**	**225**	**0.4436**

- **Adaptive Graph Guided Embedding (AG2E)** [71] proposes an adaptive graph strategy which jointly obtains the similarity graph and predicts multiple label in a semi-supervised fashion.

For the SSMLDR method, we directly set testing data as unlabeled data and evaluate its recovery performance. To fully compare our approach with other methods, we utilize the same metrics adopted in [91]. When the labels are recovered, we select the top 5 ranked labels as the recovered label. Then, the recovery precision (Pre) P and the recall (Rec) R are calculated. $P = \frac{t_p}{t_p+f_p}$, and $R = \frac{t_p}{t_p+f_n}$, where t_p represents truth-positive. f_p and f_n represent the false positive and the false negative respectively. To compare the results easier, we calculate the F1-score (F1) which is the harmonic mean of the precision and the recall, where $F1 = 2\frac{P\times R}{P+R}$. We further obtain the number of labels with a non-zero recall (N-R) value. The mean average precision (mAP) from [19] is utilized for comprehensive evaluation. In all metrics, higher value indicates better performance.

The experimental result in the conventional multi-label learning setting is illustrated in Table 2.6. We can see that our approach significantly outperforms other baselines in most of the metrics, which demonstrates the high accuracy and robustness of our approach. The work of [19] proposes a complete/augmented label set for Corel5K and ESP Game datasets, increasing Corel5K label from averagely 3.40 to 4.84 labels, and the ESP Game label from 4.69 to 7.27 labels. We evaluate our model based on these label sets (2.7). The results are shown in Table 2.7, and it indicates that our approach still achieves the best performance in most matrices.

2.5.4 Zero-shot Multi-label Classification

We extend our method to the zero-shot multi-label scenario where the classes in training and test are non-overlapped. It is a more challenging task since the distribution gaps are more

Table 2.8: Zero-shot multi-label learning performance

Data	Method	Pre	Rec	F1	N-R	mAP
SUN	LR	0.7047	0.1548	0.2539	97	0.6616
	SSMLDR	0.6637	0.1481	0.2422	95	0.6581
	FastTag	0.6906	0.1522	0.2494	90	0.6706
	ML-PGD	0.7037	0.1471	0.2433	95	0.6829
	SAE	0.6978	0.1710	0.2747	100	0.6513
	AG2E	0.7125	0.1618	0.2637	88	0.6693
	Ours	**0.7531**	**0.1857**	**0.2979**	**101**	**0.6911**
CUB	LR	0.2600	0.0307	0.0549	160	0.2693
	SSMLDR	0.2926	0.0383	0.0677	166	0.2329
	FastTag	0.2231	0.0434	0.0726	143	0.2967
	ML-PGD	0.2392	0.0365	0.0635	117	0.3178
	SAE	0.2552	0.0469	0.0798	167	0.3102
	AG2E	0.2808	0.0481	0.0821	163	0.2693
	Ours	**0.3091**	**0.0488**	**0.0843**	**179**	**0.3264**
AWA	LR	0.7555	0.0766	0.1392	66	0.8809
	SSMLDR	0.7017	0.0764	0.1378	66	0.7858
	FastTag	0.8610	0.0912	0.1649	81	0.8918
	ML-PGD	0.4338	0.0623	0.1091	49	0.8677
	SAE	0.9015	**0.0926**	**0.1679**	78	**0.8918**
	AG2E	0.8247	0.0811	0.1476	71	0.8874
	Ours	**0.9249**	0.0804	0.1480	**83**	0.8784

significant.

We evaluate our model based on the SUN, CUB and AWA datasets. These datasets have default training and testing splits for ZSL. In the SUN dataset, 645 classes are used for training and 72 classes are used for testing. In the AWA dataset, 40 classes of animals are assigned for training and the other 10 are set for testing. In the CUB dataset, 150 classes are set for training and 50 are set for testing. There are 4 different splits in CUB, thus, we execute the testing four times and report the mean results. For SUN and AWA datasets, we run the testing five times and obtain the average performance.

The performance is illustrated in Table 2.8, which indicates the high performance of our approach compared with other baselines. It shows that our model is robust and works well even when the testing classes are unobserved during the training stage. This advantage is suitable for real-world applications since the target images are not controllable. On the AWA dataset, our model still cannot achieve the best performance of all metrics. The reasons are similar to the explanation discussed in the conventional multi-label scenario. Since the label distribution is narrow and the scale of training samples is large, the performance of our model cannot achieve significant improvement.

Moreover, we visualize 10 unseen classes of CUB dataset from the predicted labels.

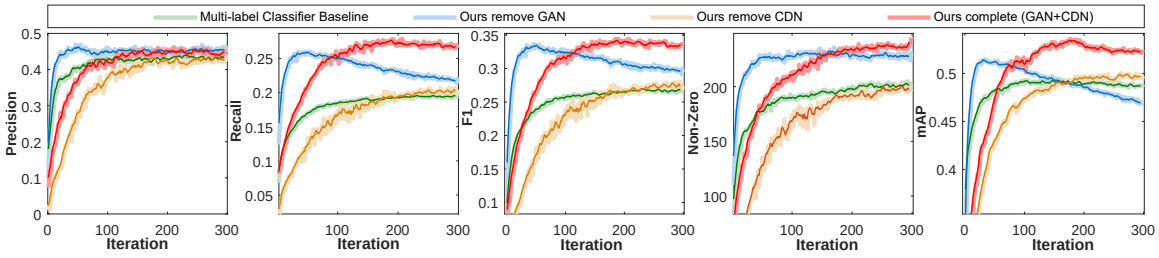


Figure 2.13: Ablation study: classification performance along training iterations in the IAPRTC-12 dataset. Different color indicates generative and CDN modules are removed/deployed in our approach. The red line indicates the results of our complete approach; blue line is our model without generative strategy; yellow line is our model without CDN; and green line is the result which both the generative and CDN modules are removed. It illustrates that CDN dramatically improves the learning performance in all metrics especially Recall, F1, and mAP metrics. Only CDN-based strategy causes over-fitting easily due to the limited training data and long-tail feature distribution, while generative model could effectively increase the robustness and stabilize the learning performance. The result demonstrates the effectiveness of both generative and CDN modules in our approach. (Please view the color figures for better visualization)

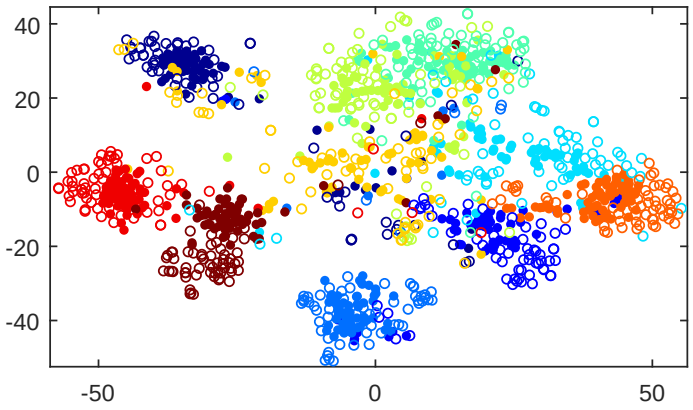


Figure 2.14: Visualization of 10 hard unseen classes of both generated (hollow circle) and ground-truth (solid circle) samples. The same color denotes the same class samples. It further demonstrates the generated samples are similar but not same to ground-truth samples, and they do enlarge/diversify the distribution area.

Table 2.9: Multi-label learning performance based on different noise levels. Gaussian noise with different variance is added on the original feature of the CUB samples.

Noise	Pre	Rec	F-1	N-R	mAP
0.00	**0.3718**	**0.0541**	**0.0944**	**214**	**0.3561**
0.05	0.3711	0.0540	0.0941	214	0.3561
0.10	0.3692	0.0538	0.0943	214	0.3537
0.15	0.3668	0.0537	0.0941	214	0.3511
0.20	0.3647	0.0534	0.0938	212	0.3482
0.25	0.3612	0.0533	0.0936	211	0.3467
0.30	0.3591	0.0531	0.0932	209	0.3416
0.35	0.3505	0.0530	0.0930	208	0.3389
0.40	0.3393	0.0529	0.0929	206	0.3351
0.45	0.3314	0.0528	0.0927	204	0.3232
0.50	0.3248	0.0526	0.0926	202	0.3215

Specifically, we deploy t-SNE [2] to map both the generated samples (hollow circle) and the ground truth samples (solid circle) to a 2-D subspace in Figure 2.14. We can see that the samples belonging to the same class become closer, while different classes samples are more separated. It indicates that our model could improve discriminability and generalizability to predict multiple labels given an unseen image.

2.5.5 Ablation Study

To demonstrate the effectiveness of all the proposed strategies in our approach, we intentionally run our approach with or without the generative and CDN modules in IAPRTC-12 dataset. Figure 2.13 illustrates the performance with the iteration increasing; different colors indicate different settings. The details are introduced in the caption. We can see that our approach achieves the highest performance when both generation and CDN modules are deployed. CDN improves the learning performance significantly; however, only utilizing CDN without generative strategy could easily cause over-fitting due to the long-tail label distribution, while generative strategy diversifies the feature distribution and effectively reduces the over-fitting issue. From Figure 2.13, we observe that only generative model without CDN could also improve the performance but the improvement is not significant. To this end, we conclude that both GAN or CDN can effectively improve the performance independently, and the combination of the two components can let GAN and CDN help each other and dramatically improve and stabilize the performance.

To further demonstrate the effectiveness of the generative strategy, we add low-level Gaussian noise on original features. Table 2.9 shows the classification performance based on various

Figure 2.15: Samples of recovered labels from SUN dataset. Each image contains several semantic labels. **Black** font denotes labels that match with the ground truth. **Blue** font denotes labels that do not exist in the ground truth but match our judgments. **Red** font denotes incorrect labels from our model. The result shows that our approach is robust and able to recover labels even when labels are missed from the ground truth.

noise levels. It illustrates that the classifier obtains the highest performance if no noise is included in the features. To this end, we conclude that noise cannot increase the sample diversity and it could further destroy the feature structure and eliminate learning performance. This result indicates that the generator is indeed an effective approach to synthesize appropriate features to diversify and enlarge corresponding distributions in feature space.

In our model, $\gamma \in [0, 1]$ is a critical hyper-parameter, as introduced in Eq. (2.29), which balances the weights between $C_M(\cdot)$ and $C_{CDN}(\cdot)$. Now, we tune γ in $[0, 1]$ on IAPRTC12 dataset and Figure 2.16 shows the performance. We observe that our model achieves stable and highest performance when $\gamma \in [0.1, 0.9]$, which indicates the parameter insensitivity of our model. If γ is too close to 0, that means there is no control on $C_M(.)$ and $C_M(\cdot)$ could not to be trained to output initial label prediction. By this way, the label relation matrix can only be considered as a regular feature extraction layer but without any reasonable logic which may decrease the generalization quality and cause the overfitting issue. Thus, we can see the clear performance decreases when γ is close to 0. Meanwhile, if γ is too close to 1 would cause $C_{CDN}(\cdot)$ not be trained which significantly reduces the learning performance. These results demonstrate the necessity of jointly training $C_{CDN}(\cdot)$ and $C_M(\cdot)$ in our model. In the implementation, we empirically set $\lambda = 0.5$ to achieve the results of all datasets which denotes that 0.5 is appropriate enough for most applications without extra tuning

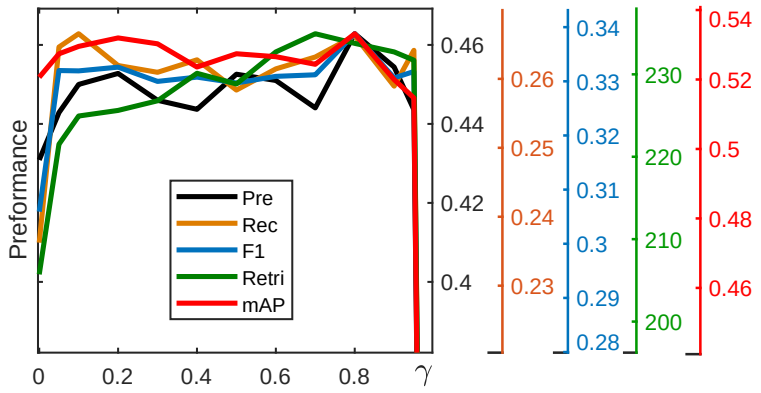


Figure 2.16: Parameter sensitivity analysis: The performance of GCDN as γ changes from 0 to 1 in IAPRTC12 dataset. The result illustrates that evaluation metrics are high and stable when $\gamma \in [0.1, 0.9]$ which demonstrates the robustness and parameter insensitivity of our model.

procedure.

2.5.6 Discussion

We notice that our approach cannot achieve the best performance in AWA dataset in some metrics. We consider this for the following reasons. First, different from other datasets, AWA samples that belong to the same class share only one consistent semantic description (label vector), thus, it is difficult to comprehensively learn neither image-label mappings nor cross-label correlations; second, due to the consistent label issue, there are limited correlation information learnt by CDN to extend to other samples/classes. The result reveals the limitation of the proposed model but this scenario is unique which is not seen very often.

2.5.7 Image Annotation

We test the image annotation performance on the SUN dataset and the result samples are shown in Figure 2.15. Figure 2.15 shows target images and the recovered labels are listed on the right. We set different colors to indicate different labels. The black font denotes correct labels. Considering there are more than 10 labels of an image in some cases, we only visualize and discuss the labels with the top 10 highest scores. The blue font indicates the missing annotations in ground truth but our model still promisingly recovers these labels based on our judgments. The red font denotes incorrect recovered labels from our model. From the result, we can see that most recovered labels are correct with several discovered "new" labels. The results indicate that our model is effective and robust,

Figure 2.17: Image retrieval result of SUN dataset in zero-shot scenario. Each row shows the images with the highest corresponding label score retrieved from the testing set. Green and red boxes indicate correct and incorrect retrieval, respectively. For each target label, we show the first incorrect retrieval result and its score ranking on the image right corner.

which is able to recover the vast majority of labels from target images in high accuracy. Moreover, our model can find the missed and error labels of the ground truth.

2.5.8 Image Retrieval

We further test our approach in the image retrieval scenario. Image retrieval is a visual search task that aims at retrieving target images from large-scale image sets. The target features can be visual, semantic or content descriptions. Image retrieval has a lot of real-world applications such as image search, person identification and data mining. In the implementation, the trained classifier is used to predict labels of the testing images. Then we rank the images by each score of the label. When inputting a retrieval label, we can find the corresponding images from the ranking results. We run the test based on the same zero-shot settings, that means the retrieved classes don't exist in the training process. It is a more challenging task than the conventional setting. Figure 2.17 shows the retrieved image samples. The left part lists the target label, and the right part shows the retrieved images. Green and red image edges indicate correct and incorrect retrieval results respectively. Since most top ranking images are correct, we intentionally select the first incorrect results of them and mark the images associated with the corresponding ranking numbers on the bottom corner of each image. From 2.17, we can see that most images are correctly retrieved with only a few errors.

There are discussions for some phenomena we find in the results. First, the performance varies in different labels. For example, the retrieval performance of *metal* is better than *digging*.

We observe that adjective and verb labels are more challenging than noun labels. Since it needs to analyze interactions between different features and more sophisticated context based structures are required for this challenge. Second, the model prefers specific scenes than others. Such as *sports*, the model prefers to retrieve all field scenes first instead of specific sport classes such as biking and swimming. Thus, more works can be done for these issues to get better retrieval performance.

2.6 Conclusion

In this Chapter, we introduced two multi-label learning frameworks. The first one is the generic multi-label learning framework via Adaptive Graph and Marginalized Augmentations (AGMA) in a semi-supervised learning scenario. It efficiently utilizes limited labeled samples associated with unlabeled samples to improve learning performance. In AGMA model, an adaptive similarity graph is learned to effectively obtain the intrinsic structure within the data; moreover, a marginalized strategy is explored to further augment the samples to reinforce the generalization and robustness of the learned model. An autoencoder is utilized to connect visual space and label space. Extensive experiments prove the usefulness of all designed modules in our framework, and demonstrate the high robustness, accuracy, and efficiency of our AMGA method. The second is a Generative Correlation Discovery Network (GCDN) for Multi-label Learning. GCDN captures the visual distribution and generates diverse samples to fill up gaps between training and testing samples. A multi-label classifier is jointly trained based on both the generated and real samples to improve the robustness and accuracy. A simple but effective Correlation Discovery Network (CDN) is proposed to automatically explore the correlations across labels and dramatically improve the learning performance without any extra semantic information as prior knowledge. All networks are jointly trained in an end-to-end scenario. Our model is quantitatively and visually evaluated based on six datasets with four settings and significantly improves the performance. Ablation study demonstrates the necessities of all proposed strategies in our model for reaching high accuracy.

Chapter 3

Multi-view Learning

3.1 Background

Multi-view learning aims to integrate complementary information from different views to improve the performances of down-stream tasks such as clustering, classification, and segmentation [30, 31, 32, 33, 34]. The views refer to various feature representations, modalities or sensors. Multi-view learning is a challenging task due to the distinct gap between heterogeneous feature domains. Most existing methods focus on analyzing static multi-view data (e.g., image, description, and attributes). In various multi-view learning tasks, multi-view action recognition [122, 123, 124, 125] has become attractive and urgent as the increasing multi-modal sensors are widely deployed in a great number of real-world applications. More specifically, RGB-D action recognition is one of the most important and urgent research directions due to the popularity of depth/3D sensors and the corresponding applications such as action recognition, emotion classification, skeleton/pose estimation, and human-computer interaction [126, 127, 128].

There are two categories in the multi-view action recognition scenario. The first category explores action sequences captured by multiple sensors which belong to the same visual modality (e.g., surveillance systems usually capture videos with RGB-only cameras). These methods assume that actions recorded by different viewpoints (e.g., front, back, and top) or distances could provide distinctive aspects for recognition tasks [123, 124, 125]. The second category methods analyze action sequences captured from different types of sensors (e.g., RGB, depth, skeleton, acceleration, trajectory, 3D, and electromyography [4, 3, 1, 129]) and attempt to integrate the complementary information among various modalities. For instance, Kinect sensor [126, 130] provides high-quality RGB, depth, and skeleton sequences simultaneously, where both depth [131, 132] and

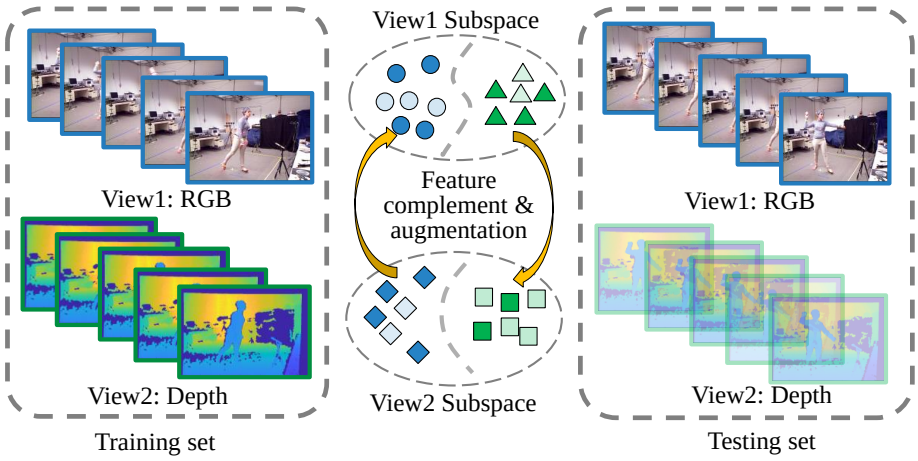


Figure 3.1: Illustration of our GMVAR approach, which is trained on both RGB and depth views. However, in the test stage, GMVAR is capable of dealing with different scenarios including complete multi-view, partially missing view, or even single-view. It is due to the generative mechanism in our model which significantly extends the potential applications of our approach.

skeleton [133, 134, 135] modalities have been demonstrated to provide effective and unique motion knowledge for action recognition. Electromyography (EMG) signal which reflects the electrical activity produced by skeletal muscles is utilized for action/motion analysis [136, 129]. Acoustical and acceleration are also utilized for multi-view event detection and action recognition tasks [137, 138].

3.2 Generative Multi-View Human Action Recognition

In this part, we briefly present our motivation, then provide our Generative Multi-View Human Action Recognition framework. Finally, we illustrate the experimental results and the ablation studies to demonstrate the effectiveness of our approach.

3.2.1 Motivation

In this thesis, we focus on the second category of multi-view action recognition. As shown in Figure 3.1, both RGB and depth views are available in the training stage while either complete or incomplete/missing views are available in the test stage. Incomplete view is a more challenging while practical scenario, which usually happens due to a lot of reasons such as sensor malfunction, equipment deficiency, and signal loss in the data transformation process. Naively fusing multi-view features (e.g., concatenation or summation) could induce a negative effect and hurt the

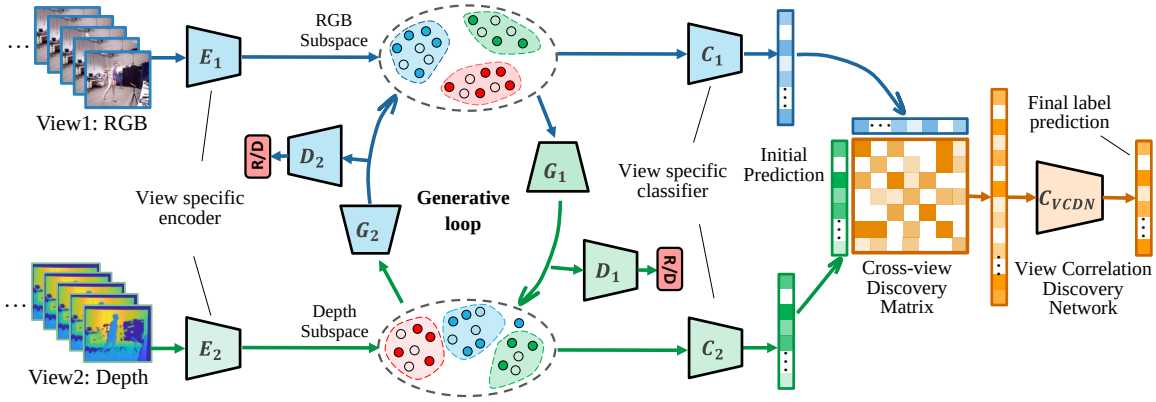


Figure 3.2: Framework of our proposed model. The RGB and depth views first go through the feature encoders $E_1(\cdot)$ and $E_2(\cdot)$ respectively to obtain more distinctive representations in the latent subspaces Z_1 and Z_2. Two generators $G_1(\cdot)$ and $G_2(\cdot)$ generate representations conditionally based on the other subspace. This generative mechanism fully explores the feature distribution across Z_1 and Z_2. Two view-specific classifiers $C_1(\cdot)$ and $C_2(\cdot)$ are trained to obtain initial recognition prediction from each view, then the proposed View Correlation Discovery Network (VCDN), $C_{VCDN}(\cdot)$, is utilized to further enhance the multi-view final prediction. Our model fully reveals the latent cross-view connection by the generative model in latent subspaces, and further explores the high-level view-correlation knowledge in label space. Due to the generative model, our model is compatible for both multi-view and single-view scenarios.

performance. Previous research efforts [35, 36, 37, 38, 39] mainly utilize effective feature extraction approaches to obtain view-specific representation first, then deploy fusion mechanism to integrate these representations together. However, these methods assume data are accessible for all the views, yet without considering the possible incomplete view issue which limits their potential applications. Hence, their performances inevitably degrade when dealing with partial multi-view data. This drawback further limits their potential compatibility in real-world applications. Moreover, different views could provide class-level unique distinctiveness, and it is crucial to explore the correlation across action classes and views to further improve the learning performance.

To this end, we propose a Generative Multi-View Action Recognition (GMVAR) framework to address the challenges above. The adversarial generative network is leveraged to generate one view conditioning on the other view, which fully explores the latent connections in both intra-view and cross-view aspects. Our approach enhances the model robustness by employing adversarial training, and naturally handles the incomplete view case by imputing the missing data. Particularly, two generative networks are developed to learn the instance-level pairwise cross-view connection knowledge, which could fully leverage the complementary information among views. More specifi-

cally, each view's generator is trained to reproduce its own latent representation, conditioning on the other view's information. By this way, our approach is able to effectively enrich the multi-view representations, and handle the missing modality case. Moreover, an effective View Correlation Discovery Network (VCDN) is proposed to further fuse the multi-view information in a higher-level cross-view correlation in the label space. VCDN aims to fully explore the latent correlations in both intra-view and cross-view aspects. Extensive experiments demonstrate the effectiveness of our proposed approach by comparing state-of-the-art algorithms. The main contributions of our approach are listed below:

- We proposed a generative multi-view action recognition framework, which can simultaneously handle complete-view, partial-view, and missing-view scenarios using a unified strategy.
- The adversarial training is encapsulated into our model to explore the complementary information shared by different modalities, which works as a regularizer to enhance the accuracy and robustness of our model.
- A simple yet effective View Correlation Discovery Network (VCDN) is proposed to learn the intra-view and cross-view label correlations in the higher-level label space. It further explores the label information and significantly improves model performance.

3.2.2 Multi-view Action Recognition

Multi-view action recognition uses data taken from multiple views/resources to achieve higher performance. It assumes different views are complementary which provide extra information and help to distinguish actions. DA-Net [139] obtains both view-independent and view-specific representations and utilizes a view classifier to combine the classification score from each view. PM-GANs [140] deploys generative and feature fusion strategies for inferred action recognition. [141] proposed a shared-specific feature factorization network which effectively fuses RGB and depth information. [38] presents a joint learning model to simultaneously explore the shared and feature-specific components to improve learning performance. [142] achieves modality hallucination through shared weights neural network for image classification. [143] proposed a cascaded residual autoencoder to handle missing view scenario. [123] fuses the action descriptors by utilizing a Multi-view Super Vector. [124] designs a novel approach for combining optical flow into enhanced 3D motion vector fields to achieve feature fusion. [144] proposes a first-person hand action recognition baseline based on 3D hand pose and RGB view. [145, 146] explores a view-invariant feature

extraction approach which is robust for actions captured from different views. Depth view is considered in [131, 132] and there are skeleton based recognition approaches [133, 134, 135] for action recognition.

Compared with existing methods, our approach is different in the following two aspects. First, it is a general multi-view action recognition approach which could handle complete-view, partial-view, and missing-view scenarios in a unified framework; second, instead of fusing views in feature space, our approach explores the correlations residing in the high-level label space which could deliver more accurate recognition results.

3.2.3 Generative Adversarial Network

GAN [105] consists of two networks: the generator and the discriminator. The generator is trained to make generated samples while the discriminator tries to differentiate the samples. Competition strategy drives both networks to enhance their abilities. Many GAN variants are recently proposed. Mode-Regularized GAN [107] introduces ways to dramatically stabilize the training process. Conditional GAN (CGAN) [109] extends the GAN model by adding extra conditional information (e.g., label knowledge) to regularize the generation process. Auxiliary Classifier GAN (ACGAN) [111] combines an auxiliary classifier with CGAN for image synthesis applications. Ding et al. explore two-stage conditional generative model for zero-shot learning [110]. Small Object Detection GAN (SOD-MTGAN) [147] generates high resolution small objects to improve multi-class detection performance. [148] deploys generative strategy to handle missing view clustering task, and [34] uses ensemble strategy to achieve final clustering result. Cycle GAN [108] utilizes the generative approach and its inverse direction to achieve unpaired image style translation. However, current models are mainly (e.g., GAN, CGAN) designed to subjectively diversify images and utilize the human perceptual aspect (e.g., MS-SSIM [113]) to evaluate the diversity; while we want to generate representations from one view to another view to solve the multi-view, partial-view, and missing view problems.

Compared with other generative models, our model builds connections across views and is designed to complement/boost the feature diversity for classification goal. Specifically, there are two major differences compared with other generative models: first, our approach is proposed to explore the generative strategy in the multi-view scenario. In addition, we deploy the generative strategy in latent subspace instead of raw feature space which hopefully explores the data structure and obtains more distinctive feature representations; second, a triplet loss is deployed to an autoencoder which

fully utilizes the available supervision information to obtain high quality subspace.

3.2.4 Preliminaries

Given the multi-view training data X_{tr}^1 and X_{tr}^2, where $X_{tr}^1 \in \mathbb{R}^{d_1 \times n_{tr}}$ and $X_{tr}^2 \in \mathbb{R}^{d_2 \times n_{tr}}$ are the feature matrices of two views, where each column represents one instance, n_{tr} is the training instance number, and d_1, d_2 are the feature dimensions of view1 and view2. $Y_{tr} \in \mathbb{R}^{d_l \times n_{tr}}$ is the one-hot label matrix, where d_l is the dimension of the label space. Correspondingly, $X_{te}^1 \in \mathbb{R}^{d_1 \times n_{te}}$, $X_{te}^2 \in \mathbb{R}^{d_2 \times n_{te}}$, and $Y_{te} \in \mathbb{R}^{d_l \times n_{te}}$ are the test features and the label matrices. Considering some of the test samples only containing single-view data, thus, the goal of our approach is to predict the label matrix Y_{te}, when either only single-view (X_{te}^1 or X_{te}^2) or both views (X_{te}^1 and X_{te}^2) are available. Generally, the feature space is much more diverse than the label space especially in multi-view action recognition scenario. To this end, we aim to compensate the visual feature and mitigate the gap between the training and test samples especially when the other view is not available.

3.2.5 Subspace Conditional Feature Generation

Inspired by the idea of generative models [105, 109, 111], we propose the generative networks to synthesize one view conditioned on the other view. By this way, the generators learn the cross-view connections and also borrow shared motion components from other actions which effectively diversifies the generated representations. Moreover, considering the original visual feature contains high-level noise, and directly generating features conditioned on visual space could bring in negative influence to the label prediction [60, 61]. To this end, we further propose a subspace conditional generative mechanism to utilize the samples projected into the corresponding subspace for view complementing/augmentation. The framework of our proposed model is shown in Figure 3.2. Our approach contains two generators, $G_1(\cdot)$ and $G_2(\cdot)$, and their corresponding discriminators, $D_1(\cdot)$ and $D_2(\cdot)$, which are trained in inverse direction; meanwhile, two view-specific encoders $E_1(\cdot)$ and $E_2(\cdot)$ are introduced to encode both views from original feature spaces to the latent subspaces Z_1 and Z_2, respectively. Moreover, in order to make the projected samples more distinctive across views, thus, the available label information associated with the triplet loss function [149] is utilized, where the goal of triplet loss is to make the projected representations closer to the samples of the same action than it is to any other actions. To this end, the objectives of $E_1(\cdot)$ and $E_2(\cdot)$ are introduced

below:

$$L_{E_m} = \sum_{i=1}^{M} \max\Big([\|E_m(X^a_{tr_i}) - E_m(X^p_{tr_i})\|_2^2 - \|E_m(X^a_{tr_i}) - E_m(X^n_{tr_i})\|_2^2 + \alpha], 0\Big), \tag{3.1}$$

where M means there are M semi-hard triplets in the given embeddings and labels, $m = \{1, 2\}$ indicates $E_1(\cdot)$ and $E_2(\cdot)$. $X^a_{tr_i}$, $X^p_{tr_i}$, and $X^n_{tr_i}$ represent the i-th training sample as *anchor*, *positive*, and *negative* respectively. α is a margin that is enforced between positive and negative pairs. By this way, the learned subspace could obtain more distinctive and robust feature representations in the corresponding subspace compared with the original feature space. Both $E_1(\cdot)$ and $E_2(\cdot)$ are implemented by a two-layer fully-connected network with the LeakyReLU activation [118] deployed in the first layer.

Then, two generative structures including $G_1(\cdot)$, $D_1(\cdot)$, $G_2(\cdot)$, and $D_2(\cdot)$, are designed for cross-view representation generation goal. Since the two networks are in symmetrical positions and have the same objective equations, thus, we only discuss $G_1(\cdot)$ and $D_1(\cdot)$ in this section. In our model, the first term is the competing approach with $D_1(\cdot)$ and makes the generated samples as real as possible:

$$L_{G_1 d} = -E_{z\sim p_z(z)} \log\Big(1 - D_1\big(G_1(z|E_1(X^1_{tr}))\big)\Big), \tag{3.2}$$

where z is the noise matrix and $E_1(X^1_{tr})$ is the learned representation as the generation condition of $G_1(\cdot)$. Since the subspaces Z_1 and Z_2 are changed when encoders $E_1(\cdot)$ and $E_2(\cdot)$ are optimized, it is difficult to directly obtain stable generative results. Thus, we include similarity constraint which pulls the generated samples and real samples to be similar in subspace. The objective term is shown as follows:

$$L_{G_1 s} = E_{z\sim p_z(z)}\Big(\|G_1(z|E_1(X^1_{tr})) - E_2(X^2_{tr})\|_{\mathrm{F}}^2\Big). \tag{3.3}$$

To this end, the overall objective of $G_1(\cdot)$ is represented as $L_{G_1} = L_{G_1 d} + \lambda L_{G_1 s}$, where λ is the trade-off parameter to balance the scales across discriminator loss and similarity loss. $G_1(\cdot)$ is a three-layer neural network with a batch normalization layer [119] to normalize input vectors and stabilize the training procedure. The goal of $D_1(\cdot)$ is to differentiate the generated samples and the real samples in subspace Z_2. And the objective function is shown below which manages to maximize L_{D_1}:

$$L_{D_1} = E_{X\sim p_X(X)} \log D_1\big(E_2(X^2_{tr})\big) + E_{z\sim p_z(z)} \log\Big(1 - D_1\big(G_1(z|E_1(X^1_{tr}))\big)\Big). \tag{3.4}$$

In our implementation, $D_1(\cdot)$ is a three-layer network. The first layer is a fully connected layer with LeakyReLU activation [118]. The second layer is a mini-batch [117] layer, which increases

the diversity of the fake samples. The activation functions of both layers are LeakyReLU and the last layer is the Sigmoid function to output the real-fake possibility of the input representations. After the generated representation is obtained in subspace, both the real and fake representations are forwarded to the view-specific classifiers $C_1(\cdot)$ and $C_2(\cdot)$ to obtain the initial label prediction. The objective functions of the classifiers include two objectives. The first one is trained to let the classifier predict labels from real samples:

$$L_{C_{m_r}} = \|Y_{tr} - C_m(E_m(X_{tr}^m))\|_{\mathrm{F}}^2, \tag{3.5}$$

where $m = \{1, 2\}$ indicates the classifiers $C_1(\cdot)$, $C_2(\cdot)$ and the encoders $E_1(\cdot)$, $E_2(\cdot)$. The second one further obtains generated samples associated with the conditional subspace representations to improve the robustness and generalization of the classifier:

$$L_{C_{1_g}} = \|Y_{tr} - C_1(G_2(z|E_2(X_{tr}^2))\|_{\mathrm{F}}^2, \tag{3.6}$$

$$L_{C_{2_g}} = \|Y_{tr} - C_2(G_1(z|E_1(X_{tr}^1))\|_{\mathrm{F}}^2. \tag{3.7}$$

To this end, the objective function of $C_m(\cdot)$ is $L_{C_m} = \beta L_{C_{m_r}} + (1-\beta)L_{C_{m_g}}$, where β is the trade-off parameters and we always set $\beta = 0.5$ in our experiments. $C_m(\cdot)$ aims to minimize L_C based on both real and generated features by benefiting from the augmented features.

3.2.6 View Correlation Discovery Network

Existing multi-view classification methods [30, 32, 31] either learn the score weights of each view or try to fuse the multi-view features in low-level feature space. However, it is hard to well align various views and easy to cause negative influence. While, in multi-view action recognition scenario, we notice that some actions are distinctive in one view (e.g., *Turning Around* in RGB view), and others are distinctive in the other view (e.g., *Answering Phone* in depth view). Thus, simply learning the weights of each view cannot take the full advantage of the view-specific motion characteristics, while exploring the latent relation hidden inside the label [71, 150] is crucial to obtain higher performance.

To this end, we further propose a simple yet effective View Correlation Discovery Network (VCDN), $C_{VCDN}(\cdot)$, to refine the action prediction by exploring the label-level knowledge across views. Instead of naively averaging/weighting the view-specific classification scores, VCDN explores the initial scores and discovers the latent correlations across different views. To this end, the final prediction is based on both the view-specific prediction and the learned across-view label-correlation knowledge.

The framework of $C_{VCDN}(\cdot)$ is shown in Figure 3.2. After the initial classification results are achieved by $y_i^1 = C_1(E_1(x_{tr_i}^1))$ and $y_i^2 = C_2(E_2(x_{tr_i}^2))$, where $y_{tr_i}^1 \in \mathbb{R}^{d_l}$ and $y_{tr_i}^2 \in \mathbb{R}^{d_l}$ are the initial predictions of the corresponding i-th sample from the two views, $x_{tr_i}^1$ and $x_{tr_i}^2$. We make a transformation from the two view predictions, $y_{tr_i}^1$ and $y_{tr_i}^2$, to obtain an cross-view label-level adjacency matrix $\mathbf{c}_i$ by multiplying $y_{tr_i}^2$ and the transpose of $y_{tr_i}^1$ as $\mathbf{c}_i = y_{tr_i}^2 \cdot y_{tr_i}^{1\top}$, where $\mathbf{c}_i \in \mathbb{R}^{d_l \times d_l}$ is the adjacency matrix. By this way, the elements in $\mathbf{c}_i$ are the multiplication of the pair-wise predicted scores. Then, the obtained $\mathbf{c}_i$ is reshaped to a d_l^2-dimensional vector and forwarded to $C_{VCDN}(\cdot)$ to predict the final prediction. To this end, $C_{VCDN}(\cdot)$ could reveal the latent correlation between the two views and help the model improve the learning performance. Since both label vectors are achieved from real samples, thus, the objective function can be written as:

$$L_{C_{VCDN}^{rr}} = \sum_{i=1}^{n_{tr}} \|y_i - C_{VCDN}(y_{tr_i}^2 \cdot y_{tr_i}^{1\top})\|_2^2, \tag{3.8}$$

where $y_i \in \mathbb{R}^{d_l}$ is the ground-truth label vector of i-th sample, and *rr* means *real-real* setting. Moreover, since $G_1(\cdot)$ and $G_2(\cdot)$ also contain effective cross-view structure information, thus, we also want this knowledge to be transferred to $C_{VCDN}(\cdot)$. To this end, we assign the predicted label vector of the fake representations $y_{fi}^1 = C_1(G_2(z|E_2(X_{tr_i}^2)))$, and $y_{fi}^2 = C_2(G_1(z|E_1(X_{tr_i}^1)))$ be utilized in the VCDN training procedure, where $y_{fi}^1 \in \mathbb{R}^{d_l}$, and $y_{fi}^2 \in \mathbb{R}^{d_l}$. We deploy both *real-fake* and *fake-real* combinations to design the objective functions:

$$L_{C_{VCDN}^{rf}} = \sum_{i=1}^{n_{tr}} \|y_i - C_{VCDN}(y_{f_i}^2 \cdot y_{tr_i}^{1\top})\|_2^2, \tag{3.9}$$

$$L_{C_{VCDN}^{fr}} = \sum_{i=1}^{n_{tr}} \|y_i - C_{VCDN}(y_{tr_i}^2 \cdot y_{f_i}^{1\top})\|_2^2. \tag{3.10}$$

Then, we obtain the final objective of $C_{VCDN}(\cdot)$:

$$L_{C_{VCDN}} = \gamma L_{C_{VCDN}^{rr}} + \frac{1-\gamma}{2}(L_{C_{VCDN}^{rf}} + L_{C_{VCDN}^{fr}}), \tag{3.11}$$

where γ is a trade-off parameter which balances the weights between real and fake label instances for training the classifiers. $C_{VCDN}(\cdot)$ is a two-layer fully connected network with Leak-ReLU activation in the first layer.

Our model is an end-to-end model and all networks are trained simultaneously. It can also be easily deployed to a wide range of applications. There are two major differences compared with other methods: first, a generative mechanism is utilized to synthesize view information from the other view, which fully explores the latent connection across the views; second, a View Correlation

Discovery Network (VCDN) is proposed to fully explore the cross-view label correlations and improve the learning performance. This strategy is effective due to the high correlation of actions across different views.

3.3 Experiment

We visually and quantitatively evaluate the performance of our approach. In addition, complete multi-view and incomplete multi-view settings are both utilized. The experimental results demonstrate the effectiveness of our proposed approach.

3.3.1 Datasets and Experimental Setting

Three real-world multi-view action datasets are deployed to evaluate the performance of our approach. The brief introduction of the datasets are listed below:

- **Berkeley Multimodal Human Action Database (MHAD)** [3] is a comprehensive multi-modal human action dataset. It contains RGB, depth, skeleton, acceleration, and audio views. MHAD contains 11 actions performed by 12 subjects for 5 repetitions of each action, yielding 660 action sequences in total.

- **Depth-included Human Action dataset (DHA)** [4] is an RGB-D multi-model dataset which contains 23 categories performed by 21 subjects, and there are 483 video clips in total for training and test. Each action has RGB images, human masks and depth data.

- **UWA3D Multiview Activity (UWA)** [1] is a multi-view dataset collected by Kinect sensors. There are 10 subjects performing 30 human activities in a continuous manner without breaks or pauses. The dataset is challenging because of varying viewpoints, self-occlusion and high similarity among activities.

In our experiments, we utilize roughly half of the available samples for training and another half for test. Specifically, there are 254 samples for training and 253 for test in the UWA dataset. 244 samples for training and 283 for test in MHAD dataset. 240 samples for training and the rest 243 for test in the DHA dataset. In the training procedure, both RGB and depth features are utilized. In the test procedure, there are three settings including single-view (RGB or depth) and multi-view (RGB-D) scenarios.

3.3.2 Multi-view Action Recognition Baselines

We test our approach in multi-view (RGB-D) scenarios. In each setting, we also deploy the state-of-the-art methods to demonstrate the effectiveness of our model. Comparison baselines are briefly introduced below.

- **Least Square Regression (LSR)** is a straightforward linear regression model. The multi-view features are concatenated together and LSR learns a linear mapping between the feature and label spaces.

- **Support Vector Machine (SVM)** [151] is a classical and robust classifier which constructs one hyperplane or multiple hyperplanes in high-dimensional space to achieve classification, regression, or other tasks. We utilize the implementation from [152] for our baseline.

- **Action Vector of Local Aggregated Descriptor (VLAD)** [153] is an effective action representation that aggregates local convolutional features and the video spatio-temporal content by an extension of Net-VLAD layer. It integrates two-stream networks and is trainable in an end-to-end framework.

- **Temporal Segment Networks (TSN)** [154] proposes a strategy that combines a sparse temporal sampling with video-level supervision. In this way, the entire video was learned effectively while it still achieves accurate and stable performance. **Weighted Depth Motion Maps (WDMM)** [131] aims to recognize human gestures from depth views, which is based on linear aggregation of spatio-temporal information. It proposed a video summarization procedure for hierarchical representation, which results in increasing intra-class similarity and also effectively reduces the inter-class similarities.

- **Auto-Weight Multiple Graph Learning (AMGL)** [31] is a multi-view classification methods. It learns the optimal weight for each graph automatically without introducing any additive parameters, which is convex and easy to get the global optimal result in a semi-supervised learning scenario.

- **Multi-view Learning with Adaptive Neighbours (MLAN)** [32] designes an adaptive graph-based method which performs semi-supervised and local structure learning simultaneously. It learns the ideal weight for each view without any parameter tuning.

- **Partial-modal Generative Adversarial Networks (PM-GANs)** [140] learns a full-modal representation based on partial modalities and implements feature-level fusion for infrared action classification tasks.

3.3.3 Implementation

We deploy the TSN [154] structure to extract RGB features. Each video is divided into 5 segments. A snippet is randomly chosen from each segment. The ResNet-101 [155] with weights pre-trained on ImageNet produces class scores for each snippet. After the training procedure, we sample 3 snippets from each video instead of 25 which is utilized in TSN since we did not observe significant improvements (less than 0.5%) between these two configurations. We obtain the final features by concatenating the output of the last layer. To this end, each video is represented in a 6144-dimensional feature vector. We utilize WDMM [131] to extract depth features. WDMM samples each video in three three projection views. After that, HOG and LBP are used to extract the features associated with VLAD and PCA for feature dimension reduction. We follow a similar scheme to WDMM [131] and obtain 110-dimensional feature vectors. As shown in Figure 3.2, the label vector concatenated with random noise is set as input to $G_1(\cdot)$ and $G_2(\cdot)$. We set the batch size to 64. The Adam optimizer [121] is used for optimization and the learning rates are set to 0.00002, 0.0001, and 0.0002 for $C_m(\cdot)$, $D_{1/2}(\cdot)$, and $G_{1/2}(\cdot)$ respectively. λ limits the feature similarity scales which is set to 0.1. In the training procedure, $D_{1/2}(\cdot)$ and $G_{1/2}(\cdot)$ are pre-trained to obtain stable initialization, while $G_{1/2}(\cdot)$ is optimized by minimizing $L_{G_{1/2}s}$ without including $L_{G_{1/2}d}$ at first, and after 50 epochs, we switch $L_{G_{1/2}}$ back and train $D_{1/2}(\cdot)$ simultaneously with the other networks. The model is implemented using TensorFlow with GPU acceleration.

Since VLAD and TSN are specifically designed for action recognition in RGB view (single view), thus, we follow the same protocol to pre-process the action data and run the code provided by the authors and report the highest performance. The same strategy is also used to evaluate WDMM in depth view. For general classification algorithms, we utilize the RGB features extracted from TSN, and depth features from WDMM since these methods are new and achieve high performance in RGB and depth representation learning respectively. To evaluate the SVM and LSR performance in the multi-view scenario, we concatenate both RGB and depth features after normalization and achieve a single feature vector for classification. Since AMGL and MLAN are designed for multi-view learning, thus, we input RGB and depth features separately and evaluate the performance. PM-GANs utilizes one view to complement another view for classification in the test stage, and we follow the

Table 3.1: Action recognition performance on UWA dataset [1]

Method	RGB	RGB→Depth	Depth	Depth→RGB	RGB+Depth
LSR	67.59	69.17	45.45	37.73	68.77
SVM [151]	69.44	68.53	34.92	34.33	72.72
VLAD [153]	71.54	-	-	-	-
TSN [154]	71.01	-	-	-	-
WDMM [131]	-	-	46.58	-	-
AMGL [31]	69.17	71.54	39.92	35.96	68.53
MLAN [32]	67.19	67.19	33.28	33.61	66.64
PM-GANs [140]	-	71.36	-	49.01	-
Ours	-	**73.53**	-	**50.35**	**76.28**

Table 3.2: Action recognition performance on MHAD dataset [3]

Method	RGB	RGB→Depth	Depth	Depth→RGB	RGB+Depth
LSR	96.46	97.17	47.63	42.51	97.17
SVM [151]	96.09	96.80	45.39	45.13	96.80
VLAD [153]	97.17	-	-	-	-
TSN [154]	97.31	-	-	-	-
WDMM [131]	-	-	66.41	-	-
AMGL [31]	96.46	97.11	30.03	29.96	94.70
MLAN [32]	96.05	96.10	41.48	41.25	96.46
PM-GANs [140]	-	96.76	-	66.84	-
Ours	-	**98.23**	-	**68.32**	**98.94**

Table 3.3: Action recognition performance on DHA dataset [4]

Method	RGB	RGB→Depth	Depth	Depth→RGB	RGB+Depth
LSR	65.02	65.43	82.30	48.56	77.36
SVM [151]	66.11	**70.24**	78.92	78.18	83.47
VLAD [153]	67.13	-	-	-	-
TSN [154]	67.85	-	-	-	-
WDMM [131]	-	-	81.05	-	-
AMGL [31]	64.61	59.05	72.84	67.33	74.89
MLAN [32]	67.91	67.91	72.96	72.83	76.13
PM-GANs [140]	-	68.72	-	76.02	-
Ours	-	69.72	-	**83.48**	**88.72**

same setting and evaluation in our experiments.

3.3.4 Performance Analysis

The experimental results are shown in Table 3.1, Table 3.2, and Table 3.3, where *RGB*, *Depth*, and *R+D* indicate the classification accuracy of single RGB view, single depth view, and RGB-D views respectively. Since our model conditionally generates another view based on the available view, thus we show *R*→*D* and *D*→*R* which indicate these settings (e.g., *R*→*D* means the depth view is conditionally generated by RGB view).To prove the effectiveness of the generated view, we deploy the pseudo feature which is the average feature from the training samples as the "generated" view of and forward to SVM, AMGL, and MLAN baselines. The results are also shown in the same column of the tables.

From the results, we observe that in the single-view scenario, our model achieves the highest performance. In *D*→*R* scenario, our generative strategy gains averagely 3% improvements in all baseline datasets. For other pseudo feature baselines, only parts of the results have slight improvements (e.g., 0.5%) while others are even lower than the single-view scenario. Therefore, the consistent pseudo feature cannot provide any extra distinctive information for improving classification performance, and concatenating the available and generated features directly (with/without normalization) could even hurt the data structure and diminish final recognition performance. These results demonstrate the effectiveness of the generative strategy of our model.

Table 3.4: Recognition performance of our model and the modified fusion strategies in both low-level feature space and high-level label space. It demonstrates the effectiveness of the VCDN framework which considerably improves the performance. Please note that the performance is lower than our complete model since we removed the generative module for a fair comparison.

Setting	UWA	MHAD	DHA
RGB-C_1	69.18	96.42	68.15
Depth-C_2	45.28	63.05	79.79
RGBD-Fea-En-Con	68.78	96.82	70.85
RGBD-Fea-Ori-Con	69.22	97.32	70.83
RGBD-Lab-Con	70.38	96.28	80.95
RGBD-Lab-Ave	71.84	97.56	83.28
RGBD-Lab-Wei	71.15	97.17	83.95
RGBD-VCDN (Ours)	**74.07**	**98.06**	**84.32**

Table 3.5: Classification performance of our VCDN model compared with the multi-layer neural networks. Different number of layers are evaluated and our VCDN achieves the highest performance.

Dataset	1-layer	2-layer	3-layer	4-layer	VCDN
UWA	74.31	74.70	73.52	75.10	**76.28**
MHAD	97.83	97.88	96.47	95.76	**98.94**
DHA	86.01	87.24	85.19	82.72	**88.72**

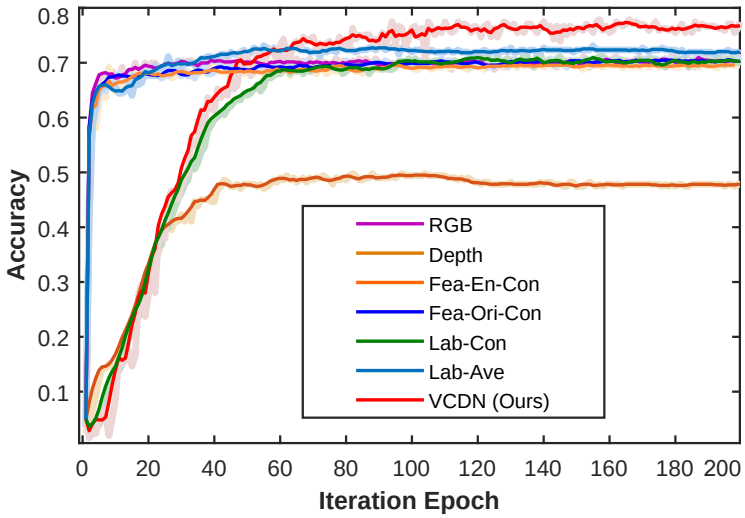


Figure 3.3: Recognition performance as the training epoch increases in UWA3D dataset [1]. The shadow lines indicate the exact performances per iteration. It shows that our VCDN framework achieves the highest performance after tens of iterations and keeps stable eventually. It demonstrates the robustness and stability of VCDN in this multi-view scenario.

For the multi-view recognition scenario, which means both the RGB and depth views are available, the generative strategy further augments the feature distribution which helps both view-specific classifiers and the VCDN framework. The results shown in column *R+D* illustrate that our model further improves the accuracy which is considerably higher than any single view scenario.

3.3.5 Ablation Study

To prove the effectiveness of VCDN, we utilize several feature/label fusion strategies to achieve multi-view classification. In addition, to avoid the influence of the augmented samples from the generative components, we first evaluate our model without including any generated samples. The result is shown in Table 3.4. The first two lines show the single-view baseline performance from the view-specific classifier $C_1(\cdot)$ and $C_2(\cdot)$; *RGBD-Fea-Ori-Con* indicates the performance when the straightforward feature concatenation approach is processed; *RGBD-Fea-En-Con* indicates the obtained features are concatenated together from $E_1(\cdot)$, $E_2(\cdot)$ and then goes through a network which has the same structure as $C_{VCDN}(\cdot)$; while *RGBD-Lab-Con* denotes the concatenated labels from $C_1(\cdot)$, $C_2(\cdot)$ and also goes through the same structure classifier as $C_{VCDN}(\cdot)$; meanwhile, *RGBD-Lab-Con* shows the performance when the obtained labels from $C_1(\cdot)$ and $C_2(\cdot)$ are averaged;

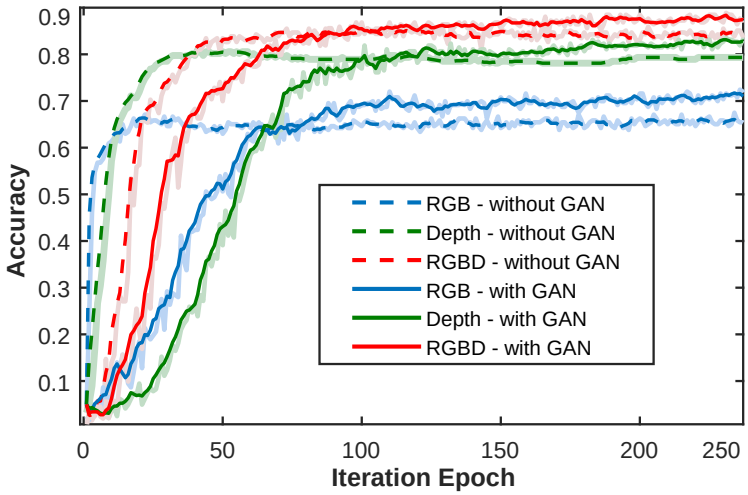


Figure 3.4: Performance of our GMVAR approach with (solid lines) and without (dashed lines) the generative strategy in DHA dataset. Different colors indicate different settings. The shadow lines indicate the exact performances per iteration. It demonstrates that the generative model does learn the cross-view connection knowledge and further improves the recognition performance.

in addition, *RGBD-Lab-Wei* shows the weighted sum of $C_1(\cdot)$ and $C_2(\cdot)$ where the weight is learned simultaneously in the training process; and the last line is our VCDN model. In this experiment, we show the performance of fusing view information in both low-level (e.g., *RGBD-Fea-En-Con* and *RGBD-Fea-Ori-Con*) and high-level (e.g., *RGBD-Lab-Con* and *RGBD-Lab-Con*). To further prove the effectiveness of VCDN, we concatenate the outputs and forward to a deeper network (i.e., 2,3,4-layer structures). The results (Table 3.5) show 2-layer structure tends to be enough. However, it still works worse than our VCDN. The result indicates that multiple views knowledge does provide extra distinctive features for action recognition; while high-level fusion performs better than low-level fusion due to the significant difference across views, and our VCDN achieves the best performance since it fully explores the label correlations.

Following the previous experimental setting, we further visualize the recognition performance as the training epoch increases, and the result is shown in Figure 3.3, where we observe that most fusion strategies cannot outperform the highest single-view classification performance. We assume that simply feature level fusion cannot provide clear distinctive clue for classifier, and it is too difficult to capture the correlation by itself; while label average approach achieves slight improvement which indicates the high-level fusion performs well in multi-view action scenario; meanwhile, our approach achieves the highest performance and keeps stable after around 100 epoch

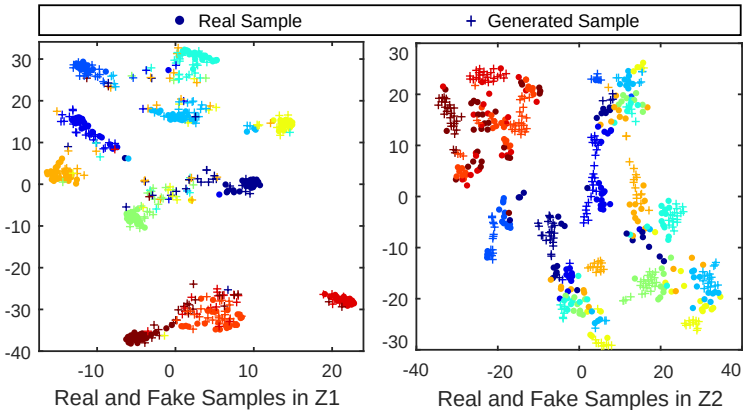


Figure 3.5: t-SNE [2] visualization results of the real and the generated test sample representations in Z_1 and Z_2 respectively. The solid circles and the cross marks indicate the real and generated representations, and different colors denote different action categories. We observe that real and generated representations which belong to the same category are close to each other. It illustrates that the generative model is capable of "recover" one view conditioned on the other view. And it further demonstrates the effectiveness of the generative strategy in this multi-view scenario.

which further demonstrates the effectiveness of the VCDN model.

We evaluate our GMVAR with and without the generative strategy to prove its effectiveness in our model. Figure 3.4 shows the recognition accuracy of GMVAR with and without the generative model in single-view (RGB and depth) and multi-view (RGB-D) settings on DHA dataset. From the results, we observe that the generative strategy indeed improves the performance of all settings considerably. Moreover, we changed the GAN module to a mapping module for further comparison. In this case, one modality is a mapping of the other, and the obtained performance (*i.e.*, UWA: 74.52%, MHAD: 98.23%, DHA: 88.07%) is lower than the model with the generative model. We assume GAN captures better feature distribution and diversifies the training space to achieve higher performance.

Furthermore, we visualize the distribution of the real and generated representations of the test samples in Z_1 and Z_2 by t-SNE [2] method respectively. The results are in Figure 3.5 which illustrate that the real and generated representations which belong to the same action category are close to each other and vice-versa. It indicates that this generative approach effectively learns the across-view correlations in the subspace which can accurately generate similar representations to complement/augment the other view. And the view-specific classifiers associated with the proposed VCDN further utilizes the knowledge to improve the action recognition performance.

3.4 Conclusion

We proposed a novel Generative Multi-View Action Recognition (GMVAR) framework in this paper. A generative mechanism is designed to generate one view conditioned on the other view. By this way, the comprehensive cross-view motion structure knowledge can be revealed. Due to this generative strategy, our model works well in single-view and missing-view scenarios which are difficult for other multi-view approaches. Moreover, we proposed an effective View Correlation Discovery Network (VCDN) which further explores the cross-view correlation in high-level label space and obtains more accurate classification results. Evaluation of three multi-view action datasets and extensive ablation studies show the effectiveness of both generative model and VCDN framework. All experimental results illustrate that our GMVAR is an effective, accurate, robust framework, and compatible with a wide range of multi-view action recognition tasks.

Chapter 4

Graph Representation Learning

4.1 Background

Representation learning has been the core problem of machine learning tasks. Graphs representation learning is one of the challenging, practical, and potential task. The concept is shown in Figure 4.1. Given a graph structured object, the goal is to represent the input graph as a dense low-dimensional vector so that we are able to feed this vector into off-the-shelf machine learning or data management techniques for a wide spectrum of downstream tasks, such as classification [40], anomaly detection [41], information retrieval [42], and many others [43, 44].

Inductive and unsupervised graph learning is a critical requirement for predictive or information retrieval tasks where label information is difficult to obtain. There are several unique challenges in representation learning in graph structured data compared with other consistent data formats. First, it is challenging to make graph learning inductive and unsupervised at the same time even compared with its transductive or supervised counterparts, as learning processes guided by reconstruction error based loss functions inevitably demand graph similarity evaluation that is usually computationally intractable. Second, when inductive capability is required, it is necessary to deal with the problem of node alignment such that we can discover common patterns across graphs. Third, in the case of unsupervised learning, we have limited options to design objectives that guide learning processes. To evaluate the quality of the learned latent representations, reconstruction errors are commonly adopted. When node alignment meets reconstruction error, we have to answer a basic question: Given two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, are they identical or isomorphic [156]? To this end, it could be computationally intractable to compute reconstruction errors (e.g., using graph edit distance [157] as the metric) in order to capture detailed structural information.

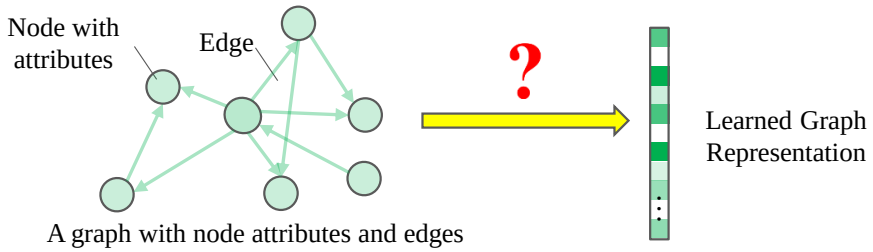


Figure 4.1: Given an input graph, graph representation learning aims to obtain the dense presentation of the given graph, where the edges and nodes could contain attributes.

Kernel-based methods were commonly utilized for learning from graph structured objects. Similarity evaluation is one of the key operations in graph learning. Conventional graph kernels rely on handcrafted substructures or graph statistics to build vector representations for graphs [158, 159, 160, 161, 162, 163]. Although kernel methods are potentially unsupervised and inductive, it is difficult to make them handle rich node and edge attributes in many applications, because of the rigid definition of substructures.

As deep neural networks achieve great progress in a wide range of machine learning and computer vision tasks. Various deep-based graph representation learning methods also proposed. Deep graph representation learning suggests a promising direction where one can learn unified vector representations for graphs by jointly considering both structural and attribute information. While most of existing works are either transductive [45, 164, 165] or supervised settings [166, 167, 168, 169, 170, 171, 172, 173, 174], a few recent studies focus on autoencoding specific structures, such as directed acyclic graphs [46], trees or graphs that can be decomposed into trees [47], and so on. In the case of graph generation, [48] proposes to generate graphs of similar graph statistics (e.g., degree distribution), and [49] provides a method to generate graphs of similar random walks. In addition, [42] propose a supervised method to learn graph similarity, and [175] theoretically analyses the expressive power of existing message-passing based graph neural networks. [176] propose anonymous walks for reconstruction tasks. It reconstructs a Markov process from the records collected by limited/partial observations. In an anonymous walk procedure, the states are visited according to the underlying transition probabilities, but no global state names are known. [177] deploy anonymous walks as a crucial strategy for obtaining data-driven and feature-based graph representations. An efficient sampling approach is designed which approximates the distributions for large networks.

4.2 SEED: Sampling, Encoding, and Embedding Distributions

4.2.1 Motivation

In this thesis, our work focuses on learning graph representations in an inductive and unsupervised manner. As inductive methods provide high efficiency and generalization for making inference over unseen data, they are desired in critical applications. For example, we could train a model that encodes graphs generated from computer program execution traces into vectors so that we can perform malware detection in a vector space. During real-time inference, efficient encoding and the capability of processing unseen programs are expected for practical usage. Meanwhile, for real-life applications where labels are expensive or difficult to obtain, such as anomaly detection [178] and information retrieval [179], unsupervised methods could provide effective feature representations shared among different tasks.

Specifically, we propose a general framework SEED (Sampling, Encoding, and Embedding Distributions) for inductive and unsupervised representation learning on graph structured objects. Instead of directly dealing with the computational challenges raised by graph similarity evaluation, given an input graph, the SEED framework samples a number of subgraphs whose reconstruction errors could be efficiently evaluated, encodes the subgraph samples into a collection of subgraph vectors, and employs the embedding of the subgraph vector distribution as the output vector representation for the input graph. One can further feed such vector representations to off-the-shelf machine learning or data management tools for downstream learning or retrieval tasks. By theoretical analysis, we demonstrate the close connection between SEED and graph isomorphism. Using public benchmark datasets, our empirical study suggests the proposed SEED framework is able to achieve up to 10% improvement, compared with competitive baseline methods.

Instead of directly addressing the computational challenge raised by evaluation of graph reconstruction errors, SEED decomposes the reconstruction problem into the following two subproblems.

Q1: How to efficiently autoencode and compare structural data in an unsupervised fashion?

SEED focuses on a class of subgraphs whose encoding, decoding, and reconstruction errors can be evaluated in polynomial time. In particular, we propose random walks with earliest visiting time (WEAVE) serving as the subgraph class, and utilize deep architectures to efficiently autoencode WEAVEs. Note that reconstruction errors with respect to WEAVEs are evaluated in linear time.

Q2: How to measure the difference of two graphs in a tractable way?

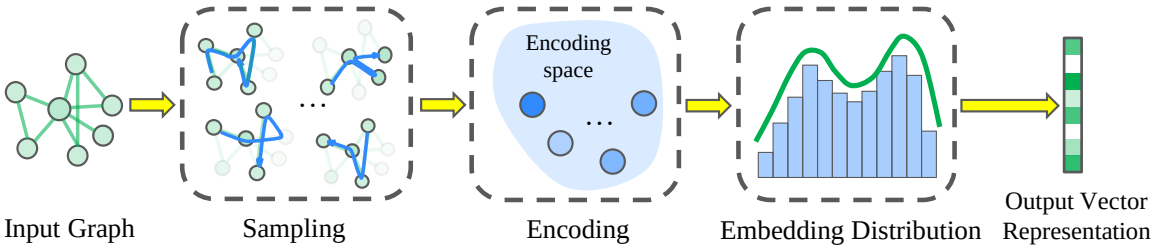


Figure 4.2: SEED consists of three components: sampling, encoding, and embedding distribution. Given an input graph, its vector representation can be obtained by going through the components.

As one subgraph only covers partial information of an input graph, SEED samples a number of subgraphs to enhance information coverage. With each subgraph encoded as a vector, an input graph is represented by a collection of vectors. If two graphs are similar, their subgraph distribution will also be similar. Based on this intuition, we evaluate graph similarity by computing distribution distance between two collections of vectors. By embedding distribution of subgraph representations, SEED outputs a vector representation for an input graph, where distance between two graphs' vector representations reflects the distance between their subgraph distributions.

Unlike existing message-passing based graph learning techniques whose expressive power is upper bounded by Weisfeiler-Lehman graph kernels [175, 180], we show the direct relationship between SEED and graph isomorphism in theoretical analysis.

To more comprehensively demonstrate the effectiveness of the proposed model, we empirically evaluate the effectiveness of the SEED framework via classification and clustering tasks on public benchmark datasets. We observe that graph representations generated by SEED are able to effectively capture structural information, and maintain stable performance even when the node attributes are not available. Compared with competitive baseline methods, the proposed SEED framework could achieve up to 10% improvement in prediction accuracy. In addition, SEED achieves high-quality representations when a reasonable number of small subgraphs are sampled. By adjusting sample size, we are able to make trade-off between effectiveness and efficiency. Unlike existing kernel or deep learning methods, our SEED framework is unsupervised with inductive capability, and naturally supports complex attributes on nodes and edges. Moreover, it works for arbitrary graphs, and provides graph representations that simultaneously capture both structural and attribute information.

4.2.2 SEED Overview

The core idea of SEED is to efficiently encode subgraphs as vectors so that we can utilize subgraph distribution distance to reflect graph similarity. We first give an abstract overview on the SEED framework in Section 4.2.2, and then discuss concrete implementations for each component in Section 4.2.3, 4.2.4, and 4.2.5, respectively. In Section 4.2.6, we share the theoretical insights in SEED. For the ease of presentation, we focus on undirected graphs with rich node attributes in the following discussion. With minor modification, our technique can also handle directed graphs with rich node and edge attributes.

SEED encodes an arbitrary graph into a vector by the following three major components, as shown in Figure 4.2.

- **Sampling**. A number of subgraphs are sampled from an input graph in this component. The design goal of this component is to find a class of subgraphs that can be efficiently encoded and decoded so that we are able to evaluate their reconstruction errors in a tractable way.
- **Encoding**. Each sampled subgraph is encoded into a vector in this component. Intuitively, if a subgraph vector representation has good quality, we should be able to reconstruct the original subgraph well based on the vector representation. Therefore, the design goal of this component is to find an autoencoding system that provides such encoding functionality.
- **Embedding distribution**. A collection of subgraph vector representations are aggregated into one vector serving as the input graph's representation. For two graphs, their distance in the output vector space approximates their subgraph distribution distance. The design goal of this component is to find such an aggregation function that preserves a pre-defined distribution distance.

Although there could be many possible implementations for the above three components, we propose a competitive implementation in this paper, and discuss them in detail in the rest of this section.

4.2.3 Sampling

In this paper, we propose to sample a class of subgraphs called WEAVE (random Walk with EArliest Visit timE). Let $\mathcal{G}$ be an input graph of a node set $V(\mathcal{G})$ and an edge set $E(\mathcal{G})$. A WEAVE of length k is sampled from $\mathcal{G}$ as follows.

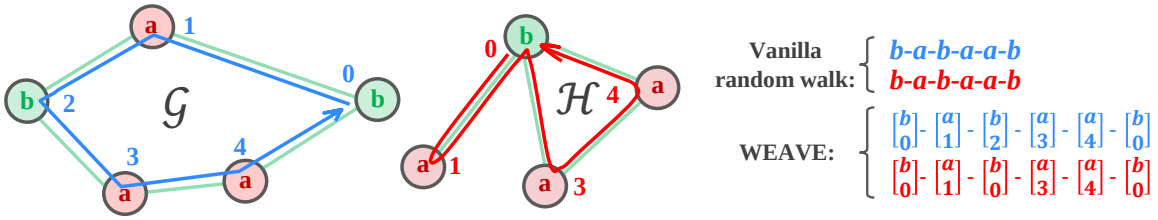


Figure 4.3: Expressive power comparison between WEAVEs and vanilla random walks: while blue and orange walks cannot be differentiated in terms of vanilla random walks, the difference under WEAVEs is outstanding.

- **Initialization**. A starting node $v^{(0)}$ is randomly drawn from $V(\mathcal{G})$ at timestamp 0, and its earliest visiting time is set to 0.
- **Next-hop selection**. Without loss of generality, assume $v^{(p)}$ is the node visited at timestamp p ($0 \leq p < k$). We randomly draw a node $v^{(p+1)}$ from $v^{(p)}$'s one-hop neighborhood as the node to be visited at timestamp $p+1$. If $v^{(p+1)}$ is a node that we have not visited before, its earliest visiting time is set to $p+1$; otherwise, its earliest visiting is unchanged. We hop to $v^{(p+1)}$.
- **Termination**. The sampling process ends when the timestamp reaches k.

In practical computation, a WEAVE is denoted as a matrix $X = [\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(k)}]$. In particular, $\mathbf{x}^{(p)} = [\mathbf{x}_a^{(p)}, \mathbf{x}_t^{(p)}]$ is a concatenation of two vectors, where $\mathbf{x}_a^{(p)}$ includes attribute information for the node visited at timestamp p, and $\mathbf{x}_t^{(p)}$ contains its earliest visit time. As earliest visit time is discrete, we use one-hot scheme to represent such information, where $\mathbf{x}_t^{(p)}$ is a k-dimensional vector and $\mathbf{x}_t^{(p)}[q] = 1$ means the earliest visit time is timestamp q. If one aims to sample s WEAVEs from an input graph, the output of this component is a set of s matrices $\{X_1, X_2, ..., X_s\}$.

Difference between WEAVEs and vanilla random walks. The key distinction comes from the information of the earliest visit time. Vanilla random walks include coarser-granularity structural information, such as neighborhood density and neighborhood attribute distribution [45]. As vanilla random walks have no memory on visit history, detailed structural information related to loops or circles is ignored. While it is also efficient to encode and decode vanilla random walk, it is difficult to evaluate finer-granularity structural difference between graphs. Unlike vanilla random walks, WEAVEs utilize earliest visit time to preserve loop information in sampled subgraphs. As shown in Figure 4.3, while we cannot tell the difference between walk w_1 and walk w_2 using vanilla

random walk, the distinction is outstanding under WEAVEs. Note that it is equally efficient to encode and decode WEAVEs, compared with vanilla random walks.

In addition, WEAVE is also related to anonymous random walks [177, 176]. By excluding attribution information, a WEAVE is reduced to an anonymous random walk.

4.2.4 Encoding

Given a set of sampled WEAVEs of length k $\{X_1, X_2, ..., X_s\}$, the goal is to encode each sampled WEAVE into a dense low-dimensional vector. As sampled WEAVEs share the same length, their matrix representations also have identical shapes. Given a WEAVE X, one could encode it by an autoencoder [181] as follows.

$$\begin{aligned} \mathbf{z} &= f(X; \theta_e), \\ \hat{X} &= g(\mathbf{z}; \theta_d), \end{aligned} \tag{4.1}$$

where $\mathbf{z}$ is the dense low-dimensional representation for the input WEAVE, $f(\cdot)$ is the encoding function implemented by an MLP with parameters θ_e, and $g(\cdot)$ is the decoding function implemented by another MLP with parameters θ_d. The quality of $\mathbf{z}$ is evaluated through reconstruction errors as follows,

$$\mathcal{L} = \|X - \hat{X}\|_2^2. \tag{4.2}$$

By conventional gradient descent based backpropagation [121], one could optimize θ_e and θ_d via minimizing reconstruction error $\mathcal{L}$. After such an autoencoder is well trained, the latent representation $\mathbf{z}$ includes both node attribute information and finer-granularity structural information simultaneously. Given s sampled WEAVEs of an input graph, the output of this component is s dense low-dimensional vectors $\{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_s\}$.

4.2.5 Embedding distribution

Let $\mathcal{G}$ and $\mathcal{H}$ be two arbitrary graphs. Suppose subgraph (e.g., WEAVE) distributions for $\mathcal{G}$ and $\mathcal{H}$ are $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$, respectively. In this component, we are interested in evaluating the distance between $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$. In this work, we investigate the feasibility of employing empirical estimate of the maximum mean discrepancy (MMD) [182] to evaluate subgraph distribution distances, without assumptions on prior distributions, while there are multiple candidate metrics for distribution distance evaluation, such as KL-divergence [183] and Wasserstein distance [184]. We leave the detailed comparison among different choices of distance metrics in our future work.

Given s subgraphs sampled from $\mathcal{G}$ as $\{\mathbf{z}_1, \cdots, \mathbf{z}_s\}$ and s subgraphs sampled from $\mathcal{H}$ as $\{\mathbf{h}_1, \cdots, \mathbf{h}_s\}$, we can estimate the distance between $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$ under the MMD framework:

$$\begin{aligned}\widehat{MMD}(P_{\mathcal{G}}, P_{\mathcal{H}}) =& \frac{1}{s(s-1)} \sum_{i=1}^{s} \sum_{j \neq i}^{s} k(\mathbf{z}_i, \mathbf{z}_j) + \frac{1}{s(s-1)} \sum_{i=1}^{s} \sum_{j \neq i}^{s} k(\mathbf{h}_i, \mathbf{h}_j) \\ & - \frac{2}{s^2} \sum_{i=1}^{s} \sum_{j=1}^{s} k(\mathbf{z}_i, \mathbf{h}_j) \\ =& \|\hat{\mu}_{\mathcal{G}} - \hat{\mu}_{\mathcal{H}}\|_2^2. \end{aligned} \tag{4.3}$$

$\hat{\mu}_{\mathcal{G}}$ and $\hat{\mu}_{\mathcal{H}}$ are empirical kernel embeddings of $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$, respectively, and are defined as follows,

$$\begin{aligned}\hat{\mu}_{\mathcal{G}} &= \frac{1}{s} \sum_{i=1}^{s} \phi(\mathbf{z}_i), \\ \hat{\mu}_{\mathcal{H}} &= \frac{1}{s} \sum_{i=1}^{s} \phi(\mathbf{h}_i), \end{aligned} \tag{4.4}$$

where $\phi(\cdot)$ is the implicit feature mapping function with respect to the kernel function $k(\cdot, \cdot)$. To this end, $\hat{\mu}_{\mathcal{G}}$ and $\hat{\mu}_{\mathcal{H}}$ are the output vector representation for $\mathcal{G}$ and $\mathcal{H}$, respectively.

In terms of kernel selection, we find the following options are effective in practice.

Identity kernel. Under this kernel, pairwise similarity evaluation is performed in the original input space. Its implementation is simple, but surprisingly effective in real-life datasets,

$$\begin{aligned}\hat{\mu}_{\mathcal{G}} &= \frac{1}{s} \sum_{i=1}^{s} \mathbf{z}_i, \\ \hat{\mu}_{\mathcal{H}} &= \frac{1}{s} \sum_{i=1}^{s} \mathbf{h}_i, \end{aligned} \tag{4.5}$$

where output representations are obtained by average aggregation over subgraph representations.

Commonly adopted kernels. For popular kernels (e.g., RBF kernel, inverse multi-quadratics kernel, and so on), it could be difficult to find and adopt their feature mapping functions. While approximation methods could be developed for individual kernels [185], we could train a deep neural network that approximates such feature mapping functions. In particular,

$$\begin{aligned}\hat{\mu}'_{\mathcal{G}} &= \frac{1}{s} \sum_{i=1}^{s} \hat{\phi}(\mathbf{z}_i; \theta_m), \\ \hat{\mu}'_{\mathcal{H}} &= \frac{1}{s} \sum_{i=1}^{s} \hat{\phi}(\mathbf{h}_i; \theta_m), \\ D(P_{\mathcal{G}}, P_{\mathcal{H}}) &= \|\hat{\mu}'_{\mathcal{G}} - \hat{\mu}'_{\mathcal{H}}\|_2^2, \end{aligned} \tag{4.6}$$

where $\hat{\phi}(\cdot;\theta_m)$ is an MLP with parameters θ_m, and $D(\cdot,\cdot)$ is the approximation to the empirical estimate of MMD. Note that $\hat{\mu}'_{\mathcal{G}}$ and $\hat{\mu}'_{\mathcal{H}}$ are output representations for $\mathcal{G}$ and $\mathcal{H}$, respectively. To train the function $\hat{\phi}(\cdot;\theta_m)$, we evaluate the approximation error by

$$J(\theta_m) = \|D(P_{\mathcal{G}}, P_{\mathcal{H}}) - \widehat{MMD}(P_{\mathcal{G}}, P_{\mathcal{H}})\|_2^2, \tag{4.7}$$

where θ_m is optimized by minimizing $J(\theta_m)$.

4.2.6 Theoretical insights

In this section, we sketch the theoretical connection between SEED and well-known graph isomorphism [156], and show how walk length in WEAVE impacts the effectiveness in graph isomorphism tests. The full proof of theorems and lemmas is detailed in Appendix.

To make the discussion self-contained, we define graph isomorphism and its variant with node attributes as follows.

- **Graph isomorphism**. $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ and $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H})))$ are isomorphic if there is a bijection function $f : V(\mathcal{G}) \Leftrightarrow V(\mathcal{H})$ such that $\forall(u,v) \in E(\mathcal{G}) \Leftrightarrow (f(u), f(v)) \in E(\mathcal{H})$.
- **Graph isomorphism with node attributes**. Let $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}), l_1), \mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}), l_2)$ be two attributed graphs, where l_1, l_2 are attribute mapping functions $l_1 : V(\mathcal{G}) \to \mathbb{R}^d, \quad l_2 : V(\mathcal{H}) \to \mathbb{R}^d$, and node attributes are denoted as d-dimensional vectors. Then $\mathcal{G}$ and $\mathcal{H}$ are isomorphic with node attributes if there is a bijection $f : V(\mathcal{G}) \Leftrightarrow V(\mathcal{H})$, s.t., $\forall(u,v) \in E(\mathcal{G}) \Leftrightarrow (f(u), f(v)) \in E(\mathcal{H})$, and $\forall u \in V(\mathcal{G}), l_1(u) = l_2(f(u))$.
- **Identical distributions**. Two distributions P and Q are identical if and only if their first order Wasserstein distance [186] $W_1(P,Q) = 0$.

The following theory suggests the minimum walk length for WEAVEs, if every edge in a graph is expected to be visited.

Lemma 1. *Let $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ be a connected graph, then there exists a walk of length k which can visit all the edges of $\mathcal{G}$, where $k \geq 2|E(\mathcal{G})| - 1$.*

Proof. We will use induction on $|E(\mathcal{G})|$ to complete the proof.

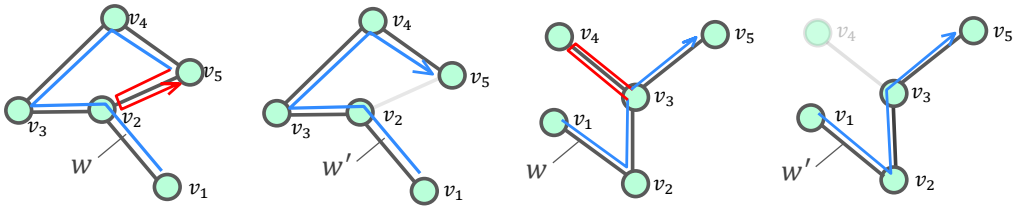
Basic case: Let $|E(\mathcal{G})| = 1$, the only possible graph is a line graph of length 1. For such a graph, the walk from one node to another can cover the only edge on the graph, which has length $1 = 2 \cdot 1 - 1$.

Induction: We assume for all the connected graphs on less than m edges (i.e., $|E(\mathcal{G})| \leq m-1$), there exist a walk of length k which can visit all the edges if $k \geq 2|E(\mathcal{G})| - 1$. Then we will show for any connected graph with m edges, there also exists a walk which can cover all the edges on the graph with length $k \geq 2|E(\mathcal{G})| - 1$.

Let $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ be a connected graph with $|E(\mathcal{G})| = m$. Firstly, we assume $\mathcal{G}$ is not a tree, which means there exist a cycle on $\mathcal{G}$. By removing an edge $e = (v_i, v_j)$ from the cycle, we can get a graph $\mathcal{G}'$ on $m-1$ edges which is still connected. This is because any edge on a cycle is not a bridge. Then according to the induction hypothesis, there exists a walk $w' = v_1 v_2 \dots v_i \dots v_j \dots v_t$ of length $k' \geq 2(m-1)+1$ which can visit all the edges on $\mathcal{G}'$ (The walk does not necessarily start from node 1, v_1 just represents the first node appears in this walk). Next, we will go back to our graph $\mathcal{G}$, as $\mathcal{G}'$ is a subgraph of $\mathcal{G}$, w' is also a walk on $\mathcal{G}$. By replacing the first appeared node v_i on walk w' with a walk $v_i v_j v_i$, we can obtain a new walk $w = v_1 v_2 \dots v_i v_j v_i \dots v_j \dots v_t$ on $\mathcal{G}$. As w can cover all the edges on $\mathcal{G}'$ and the edge e with length $k = k' + 2 \geq 2(m-1) - 1 + 2 = 2m - 1$, which means it can cover all the edges on $\mathcal{G}$ with length $k \geq 2|E(\mathcal{G})| - 1$.

Next, consider graph $\mathcal{G}$ which is a tree. In this case, we can remove a leaf v_j and its incident edge $e = (v_i, v_j)$ from $\mathcal{G}$, then we can also obtain a connected graph $\mathcal{G}'$ with $|E(\mathcal{G}')| = m - 1$. Similarly, according to the induction hypothesis, we can find a walk $w' = v_1 v_2 \dots v_i \dots v_t$ on $\mathcal{G}'$ which can visit all the $m-1$ edges of $\mathcal{G}'$ of length k', where $k' \geq 2(m-1) - 1$. As $\mathcal{G}'$ is a subgraph of $\mathcal{G}$, any walk on $\mathcal{G}'$ is also a walk on $\mathcal{G}$ including walk w'. Then we can also extend walk w' on $\mathcal{G}$ by replacing the first appeared v_i with a walk $v_i v_j v_i$, which produce a new walk $w = v_1 v_2 \dots v_i v_j v_i \dots v_t$. w can visit all the edges of $\mathcal{G}'$ as well as the edge e with length $k = k' + 2 \geq 2(m-1) - 1 + 2 = 2m - 1$. In other words, w can visit all the edges on $\mathcal{G}$ with length $k \geq 2|E(\mathcal{G})| - 1$. Now, we have verified our assumption works for all the connected graphs with m edges, hence we complete our proof. (To give an intuition for our proof of lemma 1, we provide an example of 5 edges in Figure 4.4)

Figure 4.4 $(a1)$ illustrates an example graph $\mathcal{G}$ which is a connected graph on 5 edges but not a tree. By removing an edge (v_2, v_5) from the cycle, we can get a connected graph $\mathcal{G}'$ (Figure 4.4 $(a2)$) with 4 edges. $\mathcal{G}'$ has a walk $w' = v_1 v_2 v_3 v_4 v_5$ which covers all the edges of $\mathcal{G}'$, as w' is also a walk on $\mathcal{G}$, by replacing v_5 with walk $v_5 v_2 v_5$ in w', we can get $w = v_1 v_2 v_3 v_4 v_5 v_2 v_5$ which can visit all the edges of $\mathcal{G}$. Figure 4.4 $(b1)$ shows an example graph $\mathcal{G}$ which is a tree on 5 edges. By removing the leaf v_4 and its incident edge (v_4, v_3), we can get a tree $\mathcal{G}'$ with 4 edges (Figure 4.4 (b2)). $\mathcal{G}'$ has a walk $w' = v_1 v_2 v_3 v_5$ which covers all the edges of $\mathcal{G}'$, as w' is also a walk on $\mathcal{G}$, by replacing v_3 with $v_3 v_4 v_3$ in w' we can get a walk $w = v_1 v_2 v_3 v_4 v_3 v_5$ which can cover all the edges



(a1) Graph $\mathcal{G}$ which is not a tree (a2) Graph $\mathcal{G}'$ corresponds to $\mathcal{G}$ in (a1) (b1) Graph $\mathcal{G}$ which is a tree (b2) Graph $\mathcal{G}'$ corresponds to $\mathcal{G}$ in (b1)

Figure 4.4: Different types of graphs with random walk w which can visit all the edges.

of $\mathcal{G}$. ■

Now, we are ready to present the connection between SEED and graph isomorphism.

Theorem 1. *Let $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ and $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}))$ be two connected graphs. Suppose we can enumerate all possible WEAVEs from $\mathcal{G}$ and $\mathcal{H}$ with a fixed-length*

$$k \geq 2\max\{|E(\mathcal{G})|, |E(\mathcal{H})|\} - 1, \tag{4.8}$$

where each WEAVE has a unique vector representation generated from a well-trained autoencoder. The Wasserstein distance between $\mathcal{G}$'s and $\mathcal{H}$'s WEAVE distributions is 0 if and only if $\mathcal{G}$ and $\mathcal{H}$ are isomorphic.

The following theory shows the connection in the case of graphs with nodes attributes.

The following lemma is crucial for the proof of Theorem 1.

Lemma 2. *Suppose that w, w' are two random walks on graph $\mathcal{G}$ and graph $\mathcal{H}$ respectively, if the representation of w and w' are the same, i.e., $r_w = r_{w'}$, the number of the distinct edges on w and w' are the same, as well as the number of the distinct nodes on w and w'.*

Proof. Let n_1, n_2 be the number of distinct nodes on w, w' respectively, let m_1, m_2 be the number of distinct edges on w and w' respectively. First, let's prove $n_1 = n_2$. We will prove this by contradiction. Assume $n_1 \neq n_2$, without loss of generality, let $n_1 > n_2$. According to our encoding rule, the largest number appears in a representation vector is the number of the distinct nodes in the corresponding walk. Hence, the largest element in vector r_w is n_1 while the largest element in vector $r_{w'}$ is n_2. Thus, $r_w \neq r_{w'}$, which contradicts our assumption. Therefore, we have $n_1 = n_2$.

Next, we will show $m_1 = m_2$. We will also prove this point by contradiction. Assume $m_1 \neq m_2$, without loss of generality, let $m_1 > m_2$. As we have proved $n_1 = n_2$, each edge on w and w' will be encoded as a vector like $[k_1, k_2]^\top$, where $k_1, k_2 \in [n_1]$. A walk consists of edges, hence the representation of a walk is formed by the representation of edges. Since $m_1 > m_2$, which

means there exists at least two consecutive element $[k_1, k_2]^\top$ in r_w which will not appear in $r_{w'}$, thus $r_w \neq r_{w'}$, which is a contradiction of our assumption. As a result, we can prove $m_1 = m_2$. □

Proof. We will first prove the sufficiency of the theorem, i.e., suppose graphs $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ and $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}))$ are two isomorphic graphs, we will show that the WEAVE's distribution on $\mathcal{G}$ and $\mathcal{H}$ are the same.

Let A be the set of all the possible walks with length k on $\mathcal{G}$, B be the set of all the possible walks with length k on $\mathcal{H}$. Each element of A and B represents one unique walk on $\mathcal{G}$ and $\mathcal{H}$ respectively. As we have assumed a WEAVE is a class of subgraphs, which means a WEAVE may correspond to multiple unique walks in A or B. Consider a walk $w = v_1 v_2 \dots v_i \dots v_t \in A$ (v_i represent the ith node appears in the walk), for any edge $e = (v_i, v_j)$ on w_i, as $e \in E(\mathcal{G})$, according to the definition of isomorphism, there exists a mapping $f : V(\mathcal{G}) \to V(\mathcal{H})$ such that $(f(v_i), f(v_j)) \in E(\mathcal{H})$. If we map each node on w_i to graph $\mathcal{H}$, we can get a new walk $w_i' = f(v_1)f(v_2)...f(v_t)$ on $\mathcal{H}$ as each edge $(f(v_i), f(v_j)) \in E(\mathcal{H})$, besides, as the length of w_i' is also k, we have $w_i' \in B$. Hence, we can define a new mapping $g : A \to B$, s.t.

$$\forall w_i = v_1 v_2 \dots v_t \in A, \quad g(w_i) = f(v_1)f(v_2)\dots f(v_t) = w_i' \in B. \tag{4.9}$$

Next, we will show that g is a bijective mapping. Firstly, we will show that f is injective. Suppose $g(w_1) = g(w_2)$, we want to show $w_1 = w_2$. Assume $w_1 \neq w_2$, there must exists one step i such that $w_1(i) \neq w_2(i)$, let $w_1(i) = (v_i^{(1)}, v_j^{(1)})$, $w_2(i) = (v_i^{(2)}, v_j^{(2)})$, then we have $(f(v_i^{(1)}), f(v_j^{(1)})) \neq (f(v_i^{(2)}), f(v_j^{(2)}))$ due to the definition of isomorphism. According to the mapping rule of f, $(f(v_i^{(1)}), f(v_j^{(1)}))$ is the ith step of $f(w_1)$, $(f(v_i^{(2)}), f(v_j^{(2)}))$ is the ith step of $g(w_2)$, thus the walk $g(w_1) \neq g(w_2)$, which contradicts our assumption. Therefore, the assumption is false, we have $w_1 = w_2$. Then we will show that g is surjective, i.e., for any $w' \in B$, there exists a $w \in A$ such that $g(w) = w'$. We will also prove this by contradiction, suppose there exists a walk $w' \in B$ such that we can't find any $w \in A$ to make $g(w) = w'$. Let $w' = v_1 v_2 \dots v_t$, according to the definition of isomorphism, for any edge $(v_i, v_j) \in E(\mathcal{H})$ on w', we have $(f^{-1}(v_i), f^{-1}(v_j)) \in E(\mathcal{G})$, where f^{-1} represents the inverse mapping of f. Hence

$$w = f^{-1}(v_1)f^{-1}(v_1)\dots f^{-1}(v_t) \in A, \tag{4.10}$$

as w is a walk on graph $\mathcal{H}$ with length k. Now consider $g(w)$, based on the mapping rule of g, we need to map each node on w via f, i.e.,

$$g(w) = f(f^{-1}(v_1))f(f^{-1}(v_1))\dots f(f^{-1}(v_t)) = v_1 v_2 \dots v_t = w', \tag{4.11}$$

which is a contradiction to our assumption. Thus we have proved g is an injective mapping as well as a surjective mapping, then we can conclude that g is a bijective mapping.

Then we will show the WEAVEs' distribution of $\mathcal{G}$ and $\mathcal{H}$ are the same. Since in our assumption, $|E(\mathcal{G})|$ is limited, then $|A|$ and $|B|$ are limited, besides, according to our encoding rule, different walks may correspond to one specific WEAVE while each WEAVE corresponds to a unique representation vector, thus the number of all the possible representation vectors is limited for both $\mathcal{G}$ and $\mathcal{H}$. Thus, the representation vector's distributions $P_{\mathcal{G}}$ for graph $\mathcal{G}$ and representation's distributions $P_{\mathcal{H}}$ for graph $\mathcal{H}$ are both discrete distributions. To compare the similarity of two discrete probability distributions, we can adopt the following equation to compute the Wasserstein distance and check if it is 0.

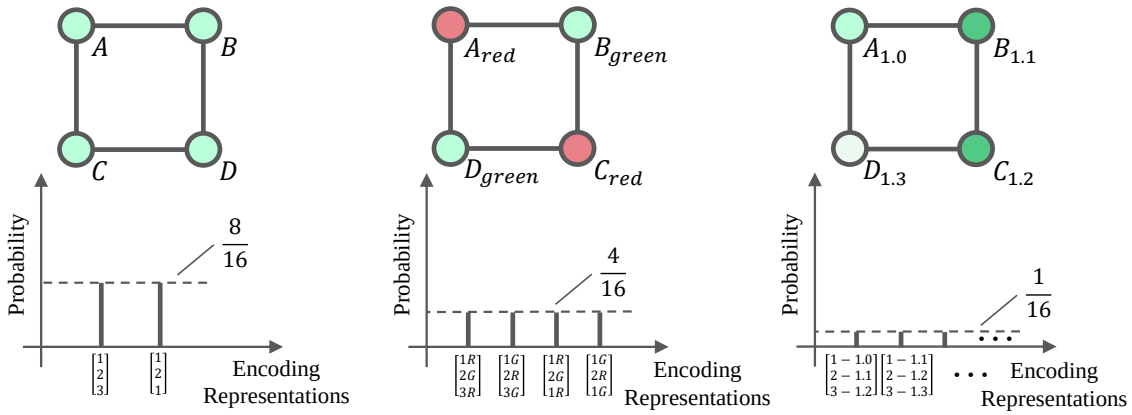
$$\begin{aligned}
&(\mathbb{P}, \mathbb{Q}) = \min_{\pi} \sum_{i=1}^{m} \sum_{j=1}^{n} \pi(i,j) s(i,j), \\
&\text{s.t.}, \sum_{i=1}^{m} \pi(i,j) = w_{q_j}, \forall j, \\
&\quad \sum_{j=1}^{n} \pi(i,j) = w_{p_i}, \forall i, \\
&\quad \pi(i,j) \geq 0, \forall i, j,
\end{aligned} \tag{4.12}$$

where $W_1(\mathbb{P}, \mathbb{Q})$ is the Wasserstein distance of probability distribution $\mathbb{P}$ and $\mathbb{Q}$, $\pi(i,j)$ is the cost function and $s(i,j)$ is a distance function, w_{q_j} and w_{p_j} are the probabilities of q_j and p_j respectively.

Since we have proved $g : A \to B$ is a bijection, besides, according to our encoding rule, $g(w)$ and w will correspond to the same WEAVE, hence they will share the same representation vector. As a consequence, for each point (g_i, w_{g_i}) (g_i corresponds to a representation vector, w_{g_i} represents the probability of g_i) in the distribution $P_{\mathcal{G}}$, we can find a point (h_i, w_{h_i}) in $P_{\mathcal{H}}$ such that $g_i = h_i$, and $w_{g_i} = w_{h_i}$. Then consider (4.12), for $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$, if we let π be a diagonal matrix with $[w_{p_1}, w_{p_2}, \ldots, w_{p_m}]$ on the diagonal and all the other elements be 0, we can make each element in the sum $\sum_{i=1}^{m} \sum_{j=1}^{n} \pi(i,j) s(i,j)$ be 0, as this sum is supposed to be nonnegative, its minimum is 0, hence $W_1(P_{\mathcal{G}}, P_{\mathcal{H}}) = 0$, which means for two isomorphic graphs $\mathcal{G}$ and $\mathcal{H}$, their WEAVE's distributions $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$ are the same.

Next we will prove the necessity of this theorem. Suppose that the Wasserstein distance between the walk representation distributions $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$ is 0, we will show that graph $\mathcal{G}$ and $\mathcal{H}$ are isomorphic. Let the number of the nodes of graph $\mathcal{G}$ is n_1, the number of the nodes of graph $\mathcal{H}$ is n_2, let the number of the edges on graph $\mathcal{G}$ is m_1, the number if the edges on graph $\mathcal{H}$ is m_2. Let $k = 2\max\{m_1, m_2\} - 1$.

Now, we will give a bijective mapping $f : V(\mathcal{G}) \to v(\mathcal{H})$. First, consider the walks on graph $\mathcal{G}$, as $k = 2\max\{m_1, m_2\} - 1 \geq 2m_1 - 1$, according to Lemma 1, there exists at least one walk of length k on graph $\mathcal{G}$ which can cover all the edges of $\mathcal{G}$. Consider such a walk $w_{\mathcal{G}}$, let $r_{\mathcal{G}} = [1, 2, 3, ..., t]^\top$ be the representation vector (corresponds to a WEAVE) we obtained according to our encoding rule. Now, we will use this representation to mark the nodes on graph $\mathcal{G}$. Mark the first node in this walk as u_1 (corresponds to 1 in the representation), the second node as u_2, the ith appearing node in $w_{\mathcal{G}}$ is u_i, continue this process until we marked all the new appearing nodes in this walk. Since $w_{\mathcal{G}}$ can visit all the edges of graph $\mathcal{G}$, all the nodes on this graph will definitely be marked, hence the last new appearing node will be marked as u_{n_1}. Now, let's consider the walks on graph $\mathcal{H}$. As we have assumed that $W_1(P_{\mathcal{G}}, P_{\mathcal{H}}) = 0$, which means that for each point (g_i, w_{g_i}) on $P_{\mathcal{G}}$, we can find a point (h_i, w_{h_i}) in $P_{\mathcal{H}}$ such that $g_i = h_i$, and $w_{g_i} = w_{h_i}$. As a consequence, as r_g is a point on $P_{\mathcal{G}}$, there must be a point r_h on $\mathcal{H}$ such that $r_h = r_g = [1, 2, 3, ..., t]^\top$. Then choose any walk w_h on $\mathcal{H}$ which produce r_h, and apply the same method to mark the nodes in this walk in order as $v_1, v_2, ..., v_{n_1}$. Now we can define the mapping f, let $f : V(\mathcal{G}) \to V(\mathcal{H})$, s.t., $f(u_i) = v_i$ for $\forall i \in [n_1]$, which is exactly the mapping we are looking for.



(a) Graph without node attributes (b) Graph with discrete node attributes (c) Graph with continuous node attributes

Figure 4.5: Walk representation distributions of graphs without attributes, graphs with discrete attributes, and graphs with continuous attributes.

Next, we just need show for each edge $(u_i, u_j) \in E(\mathcal{G})$, we have $(f(u_i), f(u_j)) \in E(\mathcal{H})$, and vice versa, then we can prove $\mathcal{G}$ and $\mathcal{H}$ are isomorphic. The first direction is obviously true as $w_{\mathcal{G}}$ covers all the edges on $\mathcal{G}$, for any edge (u_i, u_j) in $w_{\mathcal{G}}$, we have $(f(u_i), f(u_j)) = (v_i, v_j)$ which belongs to w_h, since w_h is walk on $\mathcal{H}$, we have $(v_i, v_j) \in E(\mathcal{H})$. Then we will prove the reverse

direction, i.e., for any $(v_i, v_j) = (f(u_i), f(u_j)) \in E(\mathcal{H})$, we have $(u_i, u_j) \in E(\mathcal{G})$. To prove this, we will first show that the number of edges of graph $\mathcal{G}$ and $\mathcal{H}$ are the same, i.e., $m_1 = m_2$. Suppose this is not true, without loss of generality, let $m_1 > m_2$. Since $P_{\mathcal{G}}$ and $P_{\mathcal{H}}$ are the results of random walks for infinite times. Then there must exists some walks which can visit the additional edges on $\mathcal{G}$, as a consequence, we can obtain some representation vector which will not appear in $P_{\mathcal{H}}$, which contradicts our assumption. Hence, we have $m_1 = m_2$. Besides, since we have $r_g = r_h$, according to Lemma 2, we can derive that the number of distinct edges on w_g and w_h are the same. As w_g covers all the edges on $\mathcal{G}$, hence the number of distinct edges on w_g is m_1. Therefore, the number of distinct edges on w_h is also m_1, which means w_h also has visited all the edges on $\mathcal{H}$. As for any edge (v_i, v_j) on w_h, we have (u_i, u_j) on w_h, in other words, we have $(u_i, u_j) = (f^{-1}(v_i), f^{-1}(v_j)) \in E(\mathcal{G})$. Hence we complete the proof. □

Figure 4.5 shows the walk representation distributions for a 4 nodes ring with walk length $k = 2$ in three different cases: without node attributes, with discrete node attributes, and with continuous node attributes. We can see the attributes will have an influence to the distributions, more specifically, the probability of each unique walk keeps the same no matter what the attributes are, however, the probability of each representation vector may vary as different unique walks may correspond to one representation vector, and the attributes may influence how many representation vectors there will be and how many unique walks correspond to a representation vector. To clarify, in Figure 4.5 (a), the ring graph does not have nodes attributes, there exists 16 unique walks in total, among them walk ABD, BDC, DCA, CAB, DBA, CDB, ACD, BAC will all be encoded as $r_1 = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^\top$, walk ABA, BAB, BDB, DBD, CDC, DCD, CAC, ACA will be encoded as $r_2 = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}^\top$. Hence, for a graph in Figure 4.5 (a), we have $Pr(r_1) = \frac{8}{16}$, $Pr(r_2) = \frac{8}{16}$. In Figure 4.5 (b), each node has a discrete attribute, i.e., red or green, there are still 16 unique walks in total. However, in this case, there exits four different representation vectors, walk ABC, CBA, ADC, CDA will be encoded as $r_1 = \begin{bmatrix} 1R & 2G & 3R \end{bmatrix}^\top$, where R represents Red while G represents Green; walk BCD, DCB, DAB, DCB correspond to $r_2 = \begin{bmatrix} 1G & 2R & 3G \end{bmatrix}^\top$; walk ABA, ADA, CDC, CBC correspond to $r_3 = \begin{bmatrix} 1R & 2G & 3R \end{bmatrix}^\top$; walk BAB, BCB, DCD, DAD correspond to $r_3 = \begin{bmatrix} 1R & 2G & 3R \end{bmatrix}^\top$. In this case, we have $Pr(r_1) = Pr(r_2) = Pr(r_3) = Pr(r_4) = \frac{4}{16}$. In the last, let's consider the case when there exists continuous node attributes, for such a graph, the value of node attributes has infinite choices, hence, it is very likely that each node may have different attributes. As a consequence, each unique walk will correspond to a unique representation

vector. In our example Figure 4.5 (c), there also exists 16 unique walks, each walk has a particular representation vector, hence, the probability of each representation vector is $\frac{1}{16}$.

Theorem 2. *Let $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ and $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}))$ be two connected graphs with node attributes. Suppose we can enumerate all possible WEAVEs on $\mathcal{G}$ and $\mathcal{H}$ with a fixed-length $k \geq 2\max\{|E(\mathcal{G})|, |E(\mathcal{H})|\} - 1$, where each WEAVE has a unique vector representation generated from a well-trained autoencoder. The Wasserstein distance between $\mathcal{G}$'s and $\mathcal{H}$'s WEAVE distributions is 0 if and only if $\mathcal{G}$ and $\mathcal{H}$ are isomorphic with node attributes.*

Proof. The proof for Theorem 2 is quite similar as the proof of Theorem 1, this is because the attributes just influence the representation vector form and how many unique walks correspond to a representation vector, however, the probability of each unique walk keeps the same. Hence, we can use a similar method to complete the proof. Similarly, we will first prove the sufficiency. Let $\mathcal{G}$ and $\mathcal{H}$ be two isomorphic graphs with attributes, we will prove that the walk representations distribution of $\mathcal{G}$ and $\mathcal{H}$ are the same. Suppose that A and B are the sets of possible walks of length k on $\mathcal{G}$ and $\mathcal{H}$ respectively. By applying the same analysis method as in the proof of Theorem 1, we can show that there exists a bijective mapping $g : A \to B$ such that for $\forall w_i = v_1 v_2 v_3 \dots v_t \in A$, we have

$$g(w_i) = f(v_1)f(v_2)\dots f(v_t) \in B, \tag{4.13}$$

where $f : V(\mathcal{G}) \to V(\mathcal{H})$ satisfies $\forall (v_i, v_j) \in E(\mathcal{G})$, we have $(f(v_i), f(v_j)) \in E(\mathcal{H})$ and for $\forall v_i \in V(\mathcal{G})$, the attribute of v_i and $f(v_i)$ are the same. Hence, according to our encoding rule, w_i and $f(w_i)$ will be encoded as the same representation vector, which means for each point $(r_{g_i}, Pr(r_{g_i}))$ in the representation distribution of $\mathcal{G}$, we can find a point $(r_{h_i}, Pr(r_{h_i}))$ in the distribution of $\mathcal{H}$ such that $r_{g_i} = r_{h_i}$, $Pr(r_{g_i}) = Pr(r_{h_i})$. Thus, we can obtain the Wasserstein distance of distribution $P_{\mathcal{G}}$ and the distribution $P_{\mathcal{H}}$ is $W_1(P_{\mathcal{G}}, P_{\mathcal{H}}) = 0$ via a similar approach as in Theorem 1. In other words, we have $P_{\mathcal{G}} = P_{\mathcal{H}}$. In addition, the necessity proof of Theorem 2 is the same as Theorem 1. □

Note that similar results can be easily extended to the cases with both node and edge attributes.

If both the nodes and edges in a graph have attributes, the graph is an attributed graph denoted by $\mathcal{G} = (V, E, \alpha, \beta)$, where $\alpha : V \to L_N$ and $\beta : E \to L_E$ are nodes and edges labeling functions, L_N, L_E are sets of labels for nodes and edges. In this case, the graph isomorphism are defined as:

Definition . *Given two graphs* $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}), \alpha_{\mathcal{G}}, \beta_g)$ *and* $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}), \alpha_{\mathcal{H}}, \beta_{\mathcal{H}})$*, then* $\mathcal{G}$ *and* $\mathcal{H}$ *are isomorphic with node attributes as well as edge attributes if there is a bijection* $f : V(\mathcal{G}) \Leftrightarrow V(\mathcal{H})$

$$\forall uv \in E(\mathcal{G}) \Leftrightarrow f(u)f(v) \in E(\mathcal{H}), \tag{4.14}$$

$$\alpha_{\mathcal{G}}(u) = \alpha_{\mathcal{H}}(f(u)), \forall u \in V(\mathcal{G}), \tag{4.15}$$

$$\beta_{\mathcal{G}}(u, v) = \beta_{\mathcal{H}}(f(u), f(v)). \tag{4.16}$$

Corollary 1. *Let* $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ *and* $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}))$ *be two connected graphs with node attributes. Suppose we can enumerate all possible WEAVEs on* $\mathcal{G}$ *and* $\mathcal{H}$ *with a fixed-length* $k \geq 2\max\{|E(\mathcal{G})|, |E(\mathcal{H})|\} - 1$*, where each WEAVE has a unique vector representation generated from a well-trained autoencoder. The Wasserstein distance between* $\mathcal{G}$*'s and* $\mathcal{H}$*'s WEAVE distributions is* 0 *if and only if* $\mathcal{G}$ *and* $\mathcal{H}$ *are isomorphic with both node attributes and edge attributes.*

Proof. When both nodes and edges of a graph are given attributes, the representation vectors of random walks will be different. However, just like the cases with only node attributes, the probability of each unique walk on the graph keeps the same. Hence, we can follow a similar analysis method as Theorem 2 to complete this proof. □

The theoretical results suggest the potential power of the SEED framework in capturing structural difference of graph data. As shown above, in order to achieve the same expressive power of graph isomorphism, we need to sample a large number of WEAVEs with a long walk length so that all possible WEAVEs can be enumerated. The resource demand is impractical. However, in the empirical study in Section **??**, we show that SEED can achieve state-of-the-art performance, when we sample a small number of WEAVEs with a reasonably short walk length.

4.3 Experiment

4.3.1 Datasets and Experimental Setting

We employ seven public benchmark datasets to evaluate the effectiveness of SEED. The brief introductions of the datasets are listed below.

- **Deezer User-User Friendship Networks (Deezer)** [187] is a social network dataset which is collected from the music streaming service Deezer. It represents a friendship network of users

from three European countries (i.e., Romania, Croatia and Hungary). There are three graphs which correspond to the three countries. Nodes represent the users and edges are the mutual friendships. For the three graphs, the numbers of nodes are $41,773$, $54,573$, and $47,538$, respectively, and the number of edges are $125,826$, $498,202$, and $222,887$, respectively. There exist 84 distinct genres, and genre notations are considered as node features. Thus, node features are represented as 84-dimensional multi-hot vectors.

- **Mutagenic Aromatic and Heteroaromatic Nitro Compounds (MUTAG)** [188] is a chemical bioinformatics dataset, which contains 188 chemical compounds. The compounds can be divided into two classes according to their mutagenic effect on a bacterium. The chemical data can be converted to graph structures, where each node represents an atom. Explicit hydrogen atoms have been removed. In the obtained graph, the node attributes represent the atom types (i.e., C, N, O, F, I, Cl and Br) while the edge attributes represent bond types (i.e., single, double, triple or aromatic).

- **NCI1** [189] represents a balanced subset of datasets of chemical compounds screened for activity against non-small cell lung cancer and ovarian cancer cell lines, respectively. The label is assigned based on this characteristic. Each compound is converted to a graph. There are $4,110$ graphs in total with $122,747$ edges.

- **PROTEINS** [190] is a bioinformatics dataset. The proteins in the dataset are converted to graphs based on the sub-structures and physical connections of the proteins. Specifically, nodes are secondary structure elements (SSEs), and edges represent the amino-acid sequence between the two neighbors. PROTEINS has 3 discrete labels (i.e., *helix*, *sheet*, and *turn*). There are $1,113$ graphs in total with $43,471$ edges.

- **COLLAB** [191] is a scientific collaboration dataset. It belongs to a social connection network in general. COLLAB is collected from 3 public collaboration datasets (i.e., Astro Physics, Condensed Matter Physics, and High Energy Physics). The ego-networks are generated for individual researchers. The label of each graph represents the field which this researcher belongs to. There are $5,000$ graphs with $24,574,995$ edges.

- **IMDB-BINARY** [192] is a collaboration dataset of the film industry. The ego-network of each actor/actress is converted to a graph data. Each node represents an actor/actress and each edge is the indication of two actors/actresses if they appear in the same movie. IMDB-BINARY has $1,000$ graphs associated with $19,773$ edges in total.

- **IMDB-MULTI** extends the IMDB-BINARY dataset to a multi-class version. It contains a balanced set of ego-networks derived from *Sci-Fi*, *Romance*, and *Comedy* genres. Specifically, there are $1,500$ graphs with $19,502$ edges in total.

Three state-of-the-art representative techniques are implemented as baselines in the experiments.

- **Graph Sample and Aggregate (GraphSAGE)** [174] is an inductive graph representation learning approach in either supervised or unsupervised manner. GraphSAGE explores node and structure information by sampling and aggregating features from the local neighborhood of each node. A forward propagation algorithm is specifically designed to aggregate the information together. We evaluate GraphSAGE in its unsupervised setting.

- **Graph Matching Network (GMN)** [42] utilizes graph neural networks to obtain graph representations for graph matching applications. A novel Graph Embedding Network is designed for better preserving node features and graph structures. In particular, Graph Matching Network is proposed to directly obtain the similarity score of each pair of graphs. In our implementation, we utilize the Graph Embedding Networks and deploy the graph-based loss function proposed in [174] for unsupervised learning fashion.

- **Graph Isomorphism Network (GIN)** [175] provides a simple yet effective neural network architecture for graph representation learning. It deploys the sum aggregator to achieve more comprehensive representations. The original GIN is a supervised learning method. Thus, we follow the GraphSAGE approach, and modify its objective to fit an unsupervised setting.

Two downstream tasks, classification and clustering, are deployed to evaluate the quality of the learned graph representations.

For classification task, a simple multi-layer fully connected neural network is built as a classifier. We report the average accuracy (ACC) for classification performance. For clustering task, an effective conventional clustering approach, Normalized Cuts (NCut) [193], is used to cluster graph representations. We consider two widely used metrics for clustering performance, including Accuracy (ACC) and Normalized Mutual Information (NMI) [194]. ACC comes from classification with the best mapping, and NMI evaluates the mutual information across the ground truth and the recovered cluster labels based on a normalization operation. Both ACC and NMI are positive measurements (i.e., the higher the metric is, the better the performance will be).

Setting	Datasets	Methods	SAGE	GIN	GMN	SEED	SAGE	GIN	GMN	SEED
		Metric	Node Feature **Excluded**				Node Feature **Included**			
Clustering	Dezzer	ACC	0.3853	0.4913	0.4924	**0.4927**	0.3840	**0.4930**	0.4808	0.4810
		NMI	0.0079	0.0958	0.0726	**0.1277**	0.0003	**0.0893**	0.0651	0.0566
	MUTAG	ACC	0.6649	0.4997	0.4990	**0.8014**	0.6649	0.4963	0.4910	**0.7260**
		NMI	0.0150	0.0946	0.0825	**0.3214**	0.0070	0.0933	0.0917	**0.1567**
	NCI1	ACC	0.5098	0.5221	0.5022	**0.5510**	0.5070	0.5204	0.5005	**0.5441**
		NMI	0.0003	0.0015	0.0034	**0.0073**	0.0002	0.0013	0.0042	**0.0089**
	PROTEINS	ACC	0.5657	0.5957	**0.5966**	0.5957	0.5657	0.5957	0.5957	**0.5957**
		NMI	0.0013	0.0038	0.0117	**0.0518**	0.0004	0.0034	0.0067	**0.0689**
	COLLAB	ACC	0.5208	0.5458	0.5173	**0.5973**	-	-	-	-
		NMI	0.0025	0.0729	0.0193	**0.2108**	-	-	-	-
	IMDB-BINARY	ACC	0.5069	0.6202	0.5010	**0.5776**	-	-	-	-
		NMI	0.0002	0.0459	0.0093	**0.0241**	-	-	-	-
	IMDB-MULTI	ACC	0.3550	3607	0.3348	**0.3816**	-	-	-	-
		NMI	0.0019	0.0185	0.0112	**0.0214**	-	-	-	-
Classification	Dezzer	ACC	0.3775	0.5094	0.5427	**0.6327**	0.3754	0.5270	0.5627	**0.7451**
	MUTAG	ACC	0.6778	0.6778	0.6889	**0.8112**	0.6889	0.6778	0.6889	**0.8222**
	NCI1	ACC	0.5410	0.5571	0.5123	**0.6105**	0.5328	0.5231	0.5133	**0.6151**
	PROTEINS	ACC	0.6846	**0.7387**	0.6216	0.7207	0.7027	0.7207	0.6357	**0.7462**
	COLLAB	ACC	0.5650	0.6170	0.5460	**0.6720**	-	-	-	-
	IMDB-BINARY	ACC	0.5400	0.7310	0.5140	**0.7660**	-	-	-	-
	IMDB-MULTI	ACC	0.3866	0.3843	0.3478	**0.4466**	-	-	-	-

Table 4.1: Evaluating graph representation quality by classification and clustering tasks

4.3.2 Performance Analysis

In this section, we discuss the performance of SEED and its baselines in the downstream tasks. The performance with and without the node features are reported. In this set of experiments, SEED adopts identity kernel in the component of embedding distributions.

As shown in Table 4.1, SEED consistently outperforms the baseline methods in both classification and clustering tasks. For GIN and GMN, supervision information could be crucial in order to differentiate structural variations. As GraphSAGE mainly focuses on aggregating feature information from neighbor nodes, it could be difficult for GraphSAGE to extract effective structural information from an unsupervised manner. In the unsupervised setting, SEED is able to differentiate structural differences at finer granularity and capture rich attribute information, leading to high-

quality graph representations with superior performance in downstream tasks. Interestingly, for NCI and PROTEINS datasets, we see node features bring little improvement in the unsupervised setting. One possible reason could be that node feature information has high correlation with structural information in these cases.

Sampling Number	Classification Accuracy	Clustering ACC	Clustering NMI
25	0.6832	0.6649	0.0031
50	0.6778	0.6649	0.0005
100	0.7778	0.6649	0.0537
150	0.7889	0.6968	0.1081
200	0.7778	0.7633	**0.2100**
300	0.7833	0.7502	0.1995
400	**0.8389**	0.7628	0.1928
800	0.8111	**0.7660**	0.1940

Table 4.2: Representation quality with different sampling numbers

Walk Length	Classification Accuracy	Clustering ACC	Clustering NMI
5	0.7278	0.6649	0.0534
10	0.7778	0.7633	0.2100
15	0.8167	0.7723	0.2495
20	0.8778	0.8245	0.3351
25	0.8722	0.8218	**0.3380**
30	**0.8743**	**0.8285**	0.3321

Table 4.3: Representation quality with different walk lengths

4.3.3 Ablation Study

Walk length and sample numbers are two meta-parameters in the SEED framework. By adjusting these two meta-parameters, we can make trade-off between effectiveness and computational efficiency. In the experiment, we empirically evaluate the impact of the two meta-parameters on the MUTAG dataset. In Table 4.2, each row denotes the performance with different sampling numbers

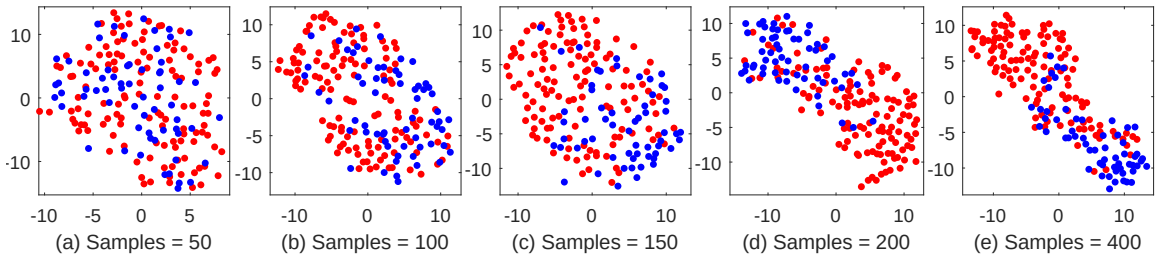


Figure 4.6: t-SNE visualziation of the MUTAG representations with different sampling numbers

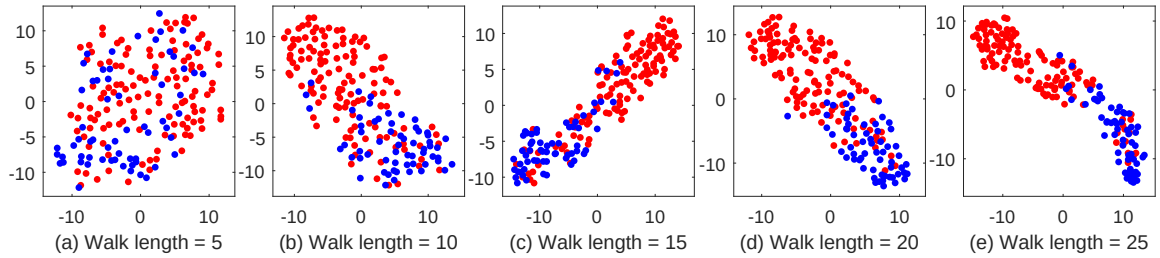


Figure 4.7: t-SNE visualziation of MUTAG representations with different walk lengths

(from 25 to 800) while the walk length is fixed to 10. Moreover, we adjust the walk length from 5 to 25 while sampling number is fixed to 200 in Table 4.3. We can see that the performance of SEED in both classification and clustering tasks increases as there are more subgraphs sampled, especially for the changes from 25 to 200. Meanwhile, we observe the increasing rates diminish dramatically when sampling number ranges from 200 to 800. Similarly, the performance of SEED increases as the walk length grows from 5 to 20, and the performance starts to converge when the length goes beyond 20.

Embedding Distribution

We employ t-SNE [2] to visualize learned graph representations in Figure 4.6 and Figure 4.7. Red and blue colors indicate two labels. We observe that the boundary becomes clearer when sample number or walk length increases.

Embedding	Classification ACC	Clustering ACC	Clustering NMI
Identity kernel	0.8112	0.8014	0.3214
RBF kernel	0.7958	0.7984	0.3115

Table 4.4: Graph representation quality comparison between identity and RBF kernel on MUTAG

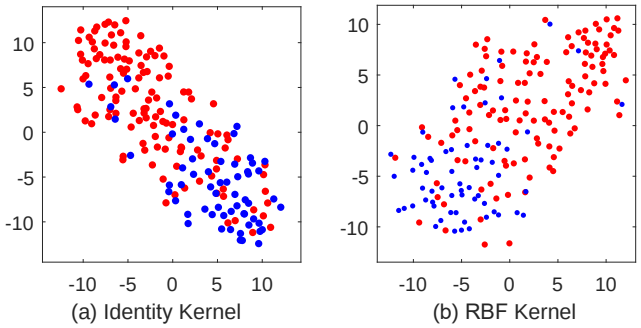


Figure 4.8: t-SNE visualization of the learned representations from different kernels on MUTAG

Identity kernels or commonly adopted kernels could be deployed in the component of embedding subgraph distributions. In our implementation, we utilize a multi-layer deep neural network to approximate a feature mapping function, for kernels whose feature mapping function is difficult to obtain. Figure 4.8 shows the t-SNE visualization of learned graph representations based on identity kernel and RBF kernel. As shown in Table 4.4, SEED variants with different kernels for distribution embedding could distinguish different classes with similar performance on the MUTAG dataset.

DeepSet

In this section, we investigate whether DeepSet [195] is an effective technique for distribution embedding. In particular, we employ DeepSet to replace the multi-layer neural network for feature mapping function approximation, and similarity values generated by MMD serve as supervision signals to guide DeepSet training. In our experiments, we compare the SEED implementation based on DeepSet with MMD (DeepSet in Table 4.5) with the SEED implementation based on the identity kernel (Identity Kernel in Table 4.5). We also observe that the MMD does not have a significant performance difference. The result confirms that DeepSet could be a strong candidate for the component of Embedding subgraph distributions.

WEAVE

In this section, we investigate the impact of node features and earliest visit time in WEAVE. In Table 4.6, *Only node feature* means only node features in WEAVE are utilized for subgraph encoding (which is equivalent to vanilla random walks), *only earliest visit time* means only earliest visit time information in WEAVE is used for subgraph encoding, and *Node feature + earliest visit time* means both information is employed. We evaluate the impact on the MUTAG dataset. As shown

Dataset	Identity Kernel			DeepSet-MMD		
	Classification	Clustering		Classification	Clustering	
	ACC	ACC	NMI	ACC	ACC	NMI
NCI1	0.6105	0.5510	0.0073	0.6382	0.5630	0.0095
PROTEINS	0.7207	0.5957	0.0518	0.7103	0.5965	0.0438
COLLAB	0.6720	0.5973	0.2108	0.6572	0.5668	0.2015
IMDB-BINARY	0.7660	0.5776	0.0241	0.7210	0.5219	0.0225
IMDB-MULTI	0.4466	0.3816	0.0214	0.4258	0.3647	0.0168

Table 4.5: Representation evaluation based on classification and clustering down-stream tasks

Feature utilized	Classification ACC	Clustering ACC	Clustering NMI
Only node feature	0.6444	0.6744	0.0625
Only earliest visit time	0.8112	**0.8014**	**0.3214**
Node feature + Earliest visit time	**0.8222**	0.7260	0.1567

Table 4.6: The impact of node feature and earliest visit time in WEAVE based on MUTAG dataset

above, it is crucial to use both node feature and earliest visit time information in order to achieve the best performance. Interestingly, on the MUTAG dataset, we observe that clustering could be easier if we only consider earliest visit time information. On the MUTAG dataset, node features seem to be noisy for the clustering task. As the clustering task is unsupervised, noisy node features could negatively impact its performance when both node features and earliest visit time information are considered.

4.3.4 Nystrom approximation in the SEED framework

The Nystrom method [196] is a fast and efficient technique for obtaining a low-rank approximation of a large kernel matrix based on a subset of its columns. We tested Nystrom based algorithm to obtain the Kernel approximation, which is faster than conventional way. We consider a pure feature based random walk baseline with earliest visit time removed, and evaluate its performance. Moreover, we present how to leverage Nystrom approximation in embedding distribution, and its performance is reported

First, we investigate the impact to the effectiveness in the downstream tasks. In this set of

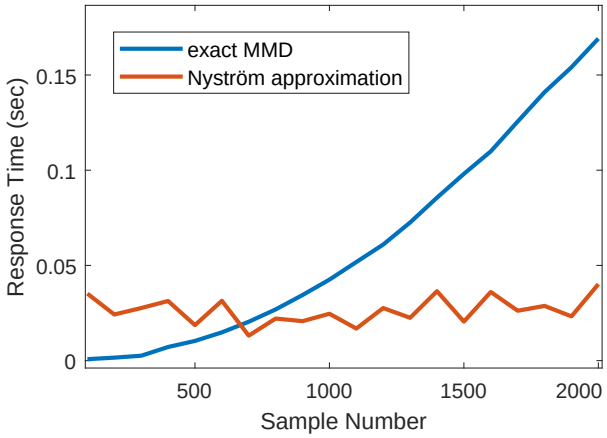


Figure 4.9: Response time comparison between exact MMD and its Nystrom approximation

	RBF Kernel			SEED-Nystrom		
Dataset	Classification	Clustering		Classification	Clustering	
	ACC	ACC	NMI	ACC	ACC	NMI
NCI1	0.6211	0.5610	0.0079	0.6281	0.5518	0.0081
PROTEINS	0.7161	0.5857	0.0476	0.7054	0.5738	0.0389
COLLAB	0.6718	0.5212	0.1831	0.6447	0.5217	0.1983
IMDB-BINARY	0.7421	0.5582	0.0218	0.7280	0.5018	0.0211
IMDB-MULTI	0.4541	0.3985	0.0241	0.4148	0.3676	0.0172

Table 4.7: Representation evaluation based on classification and clustering down-stream tasks

experiments, we implement a baseline named SEED-Nystrom, where the Nystrom method is applied to approximate RBF kernel based MMD during training phases with 200 sampled WEAVEs. In particular, top 30 eigenvalues and the corresponding eigenvectors are selected for the approximation. As shown in Table 4.7, across five datasets, SEED-Nystrom achieves comparable performance, compared with the case where an identity kernel is adopted.

In addition, we evaluate the response time of the exact RBF kernel based MMD and its Nystrom approximation. Top 30 eigenvalues and the corresponding eigenvectors are selected for the Nystrom approximation. As shown in Figure 4.9, when we range the number of WEAVE samples from 100 to 2000, the Nystrom approximation scales better than the exact MMD evaluation.

In summary, the Nystrom method is a promising method that can further improve the

scalability of the SEED framework in training phases, especially for the cases where a large number of WEAVE samples are required.

4.4 Conclusion

In this thesis, we propose a novel framework SEED (Sampling, Encoding, and Embedding distribution) framework for unsupervised and inductive graph learning. Instead of directly dealing with the computational challenges raised by graph similarity evaluation, given an input graph, the SEED framework samples a number of subgraphs whose reconstruction errors could be efficiently evaluated, encodes the subgraph samples into a collection of subgraph vectors, and employs the embedding of the subgraph vector distribution as the output vector representation for the input graph. By theoretical analysis, we demonstrate the close connection between SEED and graph isomorphism. Our experimental results suggest the SEED framework is effective, and achieves state-of-the-art predictive performance on public benchmark datasets.

Chapter 5

Conclusion

Correlation learning is an essential, practical, and important research topic for a wide range of real-world applications such as multi-view learning, multi-label learning, and graph structured object representation learning. In this thesis, we consider several challenges and applications of correlation learning including multi-label learning with limited labeled samples, multi-view learning with heterogeneous feature domains and incomplete views. Moreover, a novel graph representation learning strategy is proposed to further learn the correlation representations from data.

In chapter 2, we propose an novel Adaptive Graph and Marginalized Augmentation strategy for multi-label learning. AGMA fully utilizes the distribution information from both feature space and label space, and adaptively learns the similarity graph. Marginalized augmentation approach is used to improve the model robustness. The similarity graph, feature-label mapping, and the recovered labels are jointly optimized to achieve the best performance. In addition, to explore the potential of deep neural networks, a deep learning based generative correlation discovery network is proposed. A generative adversarial module is used to generate diverse samples, a label-correlation learning module is designed to explore the latent correlations across labels, and further improve the multi-label learning performance.

In chapter 3, we proposed a novel Generative Multi-View Action Recognition (GMVAR) framework in this paper. A generative mechanism is designed to generate one view conditioned on the other view. By this way, the comprehensive cross-view motion structure knowledge can be revealed. Due to this generative strategy, our model works well in single-view and missing-view scenarios which are difficult for other multi-view approaches. Moreover, we proposed an effective View Correlation Discovery Network (VCDN) which further explores the cross-view correlation in high-level label space and obtains more accurate classification results. Evaluation of three multi-view

action datasets and extensive ablation studies show the effectiveness of both generative model and VCDN framework.

In chapter 4, we propose a novel framework SEED (Sampling, Encoding, and Embedding distribution) framework for unsupervised and inductive graph learning. Instead of directly dealing with the computational challenges raised by graph similarity evaluation, given an input graph, the SEED framework samples a number of subgraphs whose reconstruction errors could be efficiently evaluated, encodes the subgraph samples into a collection of subgraph vectors, and employs the embedding of the subgraph vector distribution as the output vector representation for the input graph. By theoretical analysis, we demonstrate the close connection between SEED and graph isomorphism.

Bibliography

[1] H. Rahmani, A. Mahmood, D. Huynh, and A. Mian, "Histogram of oriented principal components for cross-view action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 12, pp. 2430–2443, 2016.

[2] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[3] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Berkeley MHAD: A comprehensive multimodal human action database," in *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2013, pp. 53–60.

[4] Y.-C. Lin, M.-C. Hu, W.-H. Cheng, Y.-H. Hsieh, and H.-M. Chen, "Human action recognition and retrieval using sole depth information," in *Proceedings of the ACM Multimedia*, 2012, pp. 1053–1056.

[5] S. Huang, W. Gao, and Z. Zhou, "Fast multi-instance multi-label learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[6] Y. Verma and C. Jawahar, "Image annotation by propagating labels from semantic neighbourhoods," *International Journal of Computer Vision*, vol. 121, no. 1, pp. 126–148, 2017.

[7] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos, "Supervised learning of semantic classes for image annotation and retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 394–410, 2007.

[8] L. Zhang, S. Wilson, and R. Mihalcea, "Multi-label transfer learning for multi-relational semantic similarity," in *Proceedings of the Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, Jun. 2019.

[9] Y. Song, L. Zhang, and C. L. Giles, "Automatic tag recommendation algorithms for social recommender systems," *ACM Transactions on the Web*, vol. 5, no. 1, p. 4, 2011.

[10] F. Briggs, X. Z. Fern, R. Raich, and Q. Lou, "Instance annotation for multi-instance multi-label learning," *ACM Transactions on Knowledge Discovery from Data*, vol. 7, no. 3, Sep. 2013.

[11] S. Ji, L. Tang, S. Yu, and J. Ye, "A shared-subspace learning framework for multi-label classification," *ACM Transactions on Knowledge Discovery from Data*, vol. 4, no. 2, May 2010.

[12] Y. Guo, F. Chung, G. Li, J. Wang, and J. C. Gee, "Leveraging label-specific discriminant mapping features for multi-label learning," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 2, Apr. 2019.

[13] M. Roseberry, B. Krawczyk, and A. Cano, "Multi-label punitive KNN with self-adjusting memory for drifting data streams," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 6, Nov. 2019.

[14] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[15] G. Patterson and J. Hays, "SUN attribute database: Discovering, annotating, and recognizing scene attributes," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2012, pp. 2751–2758.

[16] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," Tech. Rep. CNS-TR-2011-001, 2011.

[17] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2009, pp. 951–958.

[18] B. Wu, W. Chen, P. Sun, W. Liu, B. Ghanem, and S. Lyu, "Tagging like humans: Diverse and distinct image annotation," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2018, pp. 7967–7975.

[19] B. Wu, S. Lyu, and B. Ghanem, "ML-MG: Multi-label learning with missing labels using a mixed graph," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4157–4165.

[20] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the International Conference on Machine Learning*, 2003, pp. 912–919.

[21] X. Zhu, "Semi-supervised learning literature survey," *Technical Report 1530, University of Wisconsin-Madison*, 2005.

[22] F. Nie, D. Xu, and X. Li, "Initialization independent clustering with actively self-training method," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 1, pp. 17–27, 2012.

[23] F. Nie, W. Zhu, and X. Li, "Unsupervised feature selection with structured graph optimization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, pp. 1302–1308.

[24] Z. Ding and Y. Fu, "Low-rank common subspace for multi-view learning," in *Proceedings of the IEEE International Conference on Data Mining*, 2014, pp. 110–119.

[25] Q. Ma and A. Olshevsky, "Adversarial crowdsourcing through robust rank-one matrix completion," in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 21 841–21 852.

[26] J. T. Zhou, I. W. Tsang, S. J. Pan, and M. Tan, "Multi-class heterogeneous domain adaptation," *Journal of Machine Learning Research*, 2019.

[27] Z. Kang, L. Wen, W. Chen, and Z. Xu, "Low-rank kernel learning for graph-based clustering," *Knowledge-Based Systems*, vol. 163, pp. 510–517, 2019.

[28] Z. Kang, H. Pan, S. C. H. Hoi, and Z. Xu, "Robust graph learning from noisy data," *IEEE Transactions on Cybernetics*, pp. 1–11, 2019.

[29] Z. Kang, C. Peng, M. Yang, and Q. Cheng, "Exploiting nonlinear relationships for Top-N recommender systems," in *Proceedings of the IEEE International Conference on Big Knowledge*, 2017, pp. 49–56.

[30] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *arXiv preprint arXiv:1304.5634*, 2013.

[31] F. Nie, J. Li, X. Li *et al.*, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2016, pp. 1881–1887.

[32] F. Nie, G. Cai, and X. Li, "Multi-view clustering and semi-supervised classification with adaptive neighbours," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, pp. 2408–2414.

[33] H. Zhao, Z. Ding, and Y. Fu, "Multi-view clustering via deep matrix factorization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[34] Z. Tao, H. Liu, S. Li, Z. Ding, and Y. Fu, "From ensemble clustering to multi-view clustering," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2017, pp. 2843–2849.

[35] A. A. Chaaraoui, J. R. Padilla-López, P. Climent-Pérez, and F. Flórez-Revuelta, "Evolutionary joint selection to improve human action recognition with rgb-d devices," *Expert systems with applications*, vol. 41, no. 3, pp. 786–794, 2014.

[36] M. Yu, L. Liu, and L. Shao, "Structure-preserving binary representations for RGB-D action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1651–1664, 2016.

[37] J. Lei, X. Ren, and D. Fox, "Fine-grained kitchen activity recognition using RGB-D," in *Proceedings of the ACM Conference on Ubiquitous Computing*, 2012, pp. 208–211.

[38] J.-F. Hu, W.-S. Zheng, J. Lai, and J. Zhang, "Jointly learning heterogeneous features for RGB-D activity recognition," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2015, pp. 5344–5352.

[39] A. Chaaraoui, J. Padilla-Lopez, and F. Flórez-Revuelta, "Fusion of skeletal and silhouette-based features for human action recognition with RGB-D devices," in *Proceedings of the IEEE Computer Vision and Pattern Recognition Workshop*, 2013, pp. 91–97.

[40] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proceedings of the International Conference on Machine Learning*, 2016, pp. 2014–2023.

[41] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[42] Y. Li, C. Gu, T. Dullien, O. Vinyals, and P. Kohli, "Graph matching networks for learning the similarity of graph structured objects," in *Proceedings of the International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97, Jun 2019, pp. 3835–3845.

[43] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Proceedings of the Advances in Neural Information Processing Systems*, 2017, pp. 4967–4976.

[44] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2015.

[45] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 701–710.

[46] M. Zhang, S. Jiang, Z. Cui, R. Garnett, and Y. Chen, "D-VAE: A variational autoencoder for directed acyclic graphs," *arXiv preprint arXiv:1904.11088*, 2019.

[47] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," in *Proceedings of the International Conference on Machine Learning*, 2018.

[48] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "GraphRNN: Generating realistic graphs with deep auto-regressive models," in *Proceedings of the International Conference on Machine Learning*, 2018, pp. 5694–5703.

[49] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann, "NetGAN: Generating graphs via random walks," in *Proceedings of the International Conference on Machine Learning*, 2018.

[50] G. Qi, X. Hua, Y. Rui, J. Tang, T. Mei, and H. Zhang, "Correlative multi-label video annotation," in *Proceedings of the ACM Multimedia*, 2007, pp. 17–26.

[51] N. Ghamrawi and A. McCallum, "Collective multi-label classification," in *Proceedings of the Conference on Information and Knowledge Management*, 2005, pp. 195–200.

[52] Y. Zhang and D. Yeung, "Multilabel relationship learning," *ACM Transactions on Knowledge Discovery from Data*, vol. 7, no. 2, Aug. 2013.

[53] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2004, pp. 22–30.

[54] H. Yang, J. T. Zhou, and J. Cai, "Improving multi-label learning with missing labels by structured semantic correlations," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 835–851.

[55] L. W. Wang, Z. Ding, Z. Tao, Y. L. Liu, and Y. Fu, "Generative multi-view human action recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[56] Y. Zhang and Z. Zhou, "Multilabel dimensionality reduction via dependence maximization," *ACM Transactions on Knowledge Discovery from Data*, vol. 4, no. 3, 2010.

[57] M. Chen and A. Hauptmann, "Discriminative fields for modeling semantic concepts in video," in *Large scale semantic access to content*, 2007, pp. 151–166.

[58] W. Liu, I. W. Tsang, and K.-R. Müller, "An easy-to-hard learning paradigm for multiple classes and multiple labels," *Journal of Machine Learning Research*, vol. 18, no. 94, pp. 1–38, 2017.

[59] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-supervised learning*. The MIT Press, 2006.

[60] L. Wang, Z. Ding, and Y. Fu, "Learning transferable subspace for human motion segmentation," in *Proceedings of the the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[61] L. Wang, Z. D ing, and Y. Fu, "Low-rank transfer human motion segmentation," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 1023–1034, 2018.

[62] J. T. Zhou, H. Zhao, X. Peng, M. Fang, Z. Qin, and R. S. M. Goh, "Transfer hashing: From shallow to deep," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6191–6201, 2018.

[63] T. Zhou, S. Wang, and J. Bilmes, "Time-consistent self-supervision for semi-supervised learning," in *Proceedings of the International Conference on Machine Learning*, 2020, pp. 11 523–11 533.

[64] Z. Huang, P. Hu, J. T. Zhou, J. Lv, and X. Peng, "Partially view-aligned clustering," *Proceedings of the Advances in Neural Information Processing Systems*, vol. 33, 2020.

[65] X. Peng, H. Zhu, J. Feng, C. Shen, H. Zhang, and J. T. Zhou, "Deep clustering with sample-assignment invariance prior," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4857–4868, 2019.

[66] Z. Zha, T. Mei, J. Wang, Z. Wang, and X. Hua, "Graph-based semi-supervised learning with multiple labels," *Journal of Visual Communication and Image Representation*, vol. 20, no. 2, pp. 97–103, 2009.

[67] Q. Ma, Y.-Y. Liu, and A. Olshevsky, "Optimal lockdown for pandemic control," *arXiv preprint arXiv:2010.12923*, 2020.

[68] J. Liu, M. Li, W. Ma, Q. Liu, and H. Lu, "An adaptive graph model for automatic image annotation," in *Proceedings of the ACM International Workshop on Multimedia Information Retrieval*, 2006, pp. 61–70.

[69] F. Nie, S. Yang, R. Zhang, and X. Li, "A general framework for auto-weighted feature selection via global redundancy minimization," *IEEE Transactions on Image Processing*, 2018.

[70] W. Wang, Y. Yan, F. Nie, S. Yan, and N. Sebe, "Flexible manifold learning with optimal graph for image and video representation," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 2664–2675, 2018.

[71] L. Wang, Z. Ding, and Y. Fu, "Adaptive graph guided embedding for multi-label annotation." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018, pp. 2798–2804.

[72] L. Maaten, M. Chen, S. Tyree, and K. Weinberger, "Learning with marginalized corrupted features," in *Proceedings of the International Conference on Machine Learning*, 2013, pp. 410–418.

[73] L. Maaten., M. Chen, S. Tyree, and K. Weinberger, "Marginalizing corrupted features," *arXiv preprint arXiv:1402.7001*, 2014.

[74] M. Chen, K. Weinberger, F. Sha, and Y. Bengio, "Marginalized denoising auto-encoders for nonlinear representations," in *Proceedings of the International Conference on Machine Learning*, 2014, pp. 1476–1484.

[75] Y. Li, M. Yang, Z. Xu, and Z. Zhang, "Learning with marginalized corrupted features and labels together." in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, pp. 1251–1257.

[76] B. Guo, C. Hou, F. Nie, and D. Yi, "Semi-supervised multi-label dimensionality reduction," in *Proceedings of the IEEE International Conference on Data Mining*, 2016, pp. 919–924.

[77] T. X. Elyor Kodirov and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, July 2017.

[78] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[79] R. H. Bartels and G. Stewart, "Solution of the matrix equation AX+XB=C [F4]," *ACM Communications*, vol. 15, no. 9, pp. 820–826, 1972.

[80] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[81] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," in *Proceedings of the ACM Symposium on Theory of Computing*, 1987, pp. 1–6.

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

[83] F. Briggs, B. Lakshminarayanan, L. Neal, X. Fern, R. Raich, S. Hadley, A. Hadley, and M. Betts, "New methods for acoustic classification of multiple simultaneous bird species in a noisy environment," in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing*, 2013, pp. 1–8.

[84] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. P. Vlahavas, "Multi-label classification of music into emotions," in *Proceedings of the ISMIR*, 2008, pp. 325–330.

[85] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," 2010.

[86] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[87] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A large-scale hierarchical image database," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[88] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[89] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *Proceedings of the Advances in Neural Information Processing Systems*, 2009, pp. 1410–1418.

[90] M. Chen, A. Zheng, and K. Weinberger, "Fast image tagging," in *Proceedings of the International Conference on Machine Learning*, 2013, pp. 1274–1282.

[91] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid, "TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 309–316.

[92] W. Liu, D. Xu, I. Tsang, and W. Zhang, "Metric learning for multi-output tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[93] W. Ge, S. Yang, and Y. Yu, "Multi-evidence filtering and fusion for multi-label classification, object detection and semantic segmentation based on weakly supervised learning," in *Proceedings of the IEEE International Conference on Computer Vision*, June 2018.

[94] X. Zhao, H. Li, X. Shen, X. Liang, and Y. Wu, "A modulation module for multi-task learning with applications in image retrieval," in *Proceedings of the European Conference on Computer Vision*, September 2018.

[95] P. Duygulu, K. Barnard, J. F. de Freitas, and D. A. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *Proceedings of the European Conference on Computer Vision*, 2002, pp. 97–112.

[96] L. Von Ahn and L. Dabbish, "Labeling images with a computer game," in *Proceedings of the SIGCHI*, 2004, pp. 319–326.

[97] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[98] M. Belkin, I. Matveeva, and P. Niyogi, "Regularization and semi-supervised learning on large graphs," in *Proceedings of the International Conference on Computational Learning Theory*. Springer, 2004, pp. 624–638.

[99] Q. Ma, H. Xia, G. Ma, Y. Xia, and C. Wang, "Improved stability and stabilization criteria for TS fuzzy systems with distributed time-delay," in *Proceedings of the International Conference on Data Mining and Big Data*, 2017, pp. 517–526.

[100] Q. Ma, L. Li, H. Xia, M. Yang, and G. Ma, "New results on stability and stabilization analyses for TS fuzzy systems with distributed time-delay under imperfect premise matching," in *Proceedings of the International Conference on Intelligent Control and Information Processing*, 2016, pp. 5–10.

[101] B. Wu, F. Jia, W. Liu, B. Ghanem, and S. Lyu, "Multi-label learning with missing labels using mixed dependency graphs," *International Journal of Computer Vision*, pp. 1–22, 2018.

[102] C.-W. Lee, W. Fang, C.-K. Yeh, and Y.-C. Frank Wang, "Multi-label zero-shot learning with structured knowledge graphs," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2018, pp. 1576–1585.

[103] F. Tai and H.-T. Lin, "Multilabel classification with principal label space transformation," *Neural Computation*, vol. 24, no. 9, pp. 2508–2542, 2012.

[104] S. Chen, Y. Chen, C. Yeh, and Y. F. Wang, "Order-free RNN with visual attention for multi-label classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[105] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[106] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2017, pp. 2794–2802.

[107] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li, "Mode regularized generative adversarial networks," *arXiv preprint arXiv:1612.02136*, 2016.

[108] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2017, pp. 2223–2232.

[109] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[110] Z. Ding, M. Shao, and Y. Fu, "Generative zero-shot learning via low-rank embedded semantic dictionary," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[111] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proceedings of the International conference on machine learning*. PMLR, 2017, pp. 2642–2651.

[112] Z. Ding, Y. Guo, L. Zhang, and Y. Fu, "One-shot face recognition via generative learning," in *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 2018, pp. 1–7.

[113] Z. Wang *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, 2004.

[114] G.-J. Qi, "Loss-sensitive generative adversarial networks on lipschitz densities," *arXiv preprint arXiv:1701.06264*, 2017.

[115] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *arXiv preprint arXiv:1812.04202*, 2018.

[116] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of the Advances in Neural Information Processing Systems*, 2016, pp. 3844–3852.

[117] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proceedings of the Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.

[118] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

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

[120] M. Grubinger, P. Clough, H. Müller, and T. Deselaers, "The IAPR TC-12 benchmark: A new evaluation resource for visual information systems," in *Proceedings of the International Workshop OntoImage*, 2006.

[121] D. Kingma and J. Ba, "ADAM: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[122] Z. Cheng, L. Qin, Y. Ye, Q. Huang, and Q. Tian, "Human daily action analysis with multi-view and color-depth data," in *Proceedings of the European Conference on Computer Vision*. Springer, 2012, pp. 52–61.

[123] Z. Cai, L. Wang, X. Peng, and Y. Qiao, "Multi-view super vector for action recognition," in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 596–603.

[124] M. B. Holte, T. B. Moeslund, N. Nikolaidis, and I. Pitas, "3D human action recognition for multi-view camera systems," in *Proceedings of the IEEE International conference on 3D imaging, modeling, processing, visualization and transmission*, 2011, pp. 342–349.

[125] X. Ji, C. Wang, and Y. Li, "A view-invariant action recognition based on multi-view space hidden markov models," *International Journal of Humanoid Robotics*, vol. 11, no. 01, p. 1450011, 2014.

[126] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE Multimedia*, vol. 19, no. 2, pp. 4–10, 2012.

[127] L. Keselman, J. Iselin Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, "Intel realsense stereoscopic depth cameras," in *Proceedings of the IEEE International Conference on Computer Vision Workshop*, 2017, pp. 1–10.

[128] R. Horaud, M. Hansard, G. Evangelidis, and C. Ménier, "An overview of depth cameras and range scanners based on Time-of-Flight technologies," *Machine Vision and Applications*, vol. 27, no. 7, pp. 1005–1020, 2016.

[129] L. Wang, B. Sun, J. Robinson, T. Jing, and Y. Fu, "EV-Action: Electromyography-vision multi-modal action dataset," in *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 2020, pp. 160–167.

[130] D. Pagliari and L. Pinto, "Calibration of Kinect for Xbox one and comparison between the two generations of microsoft sensors," *Sensors*, vol. 15, pp. 27 569–27 589, 10 2015.

[131] R. Azad, M. Asadi-Aghbolaghi, S. Kasaei, and S. Escalera, "Dynamic 3D hand gesture recognition by learning weighted depth motion maps," *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[132] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. O. Ogunbona, "Action recognition from depth maps using deep convolutional neural networks," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 4, pp. 498–509, 2016.

[133] T. Soo Kim and A. Reiter, "Interpretable 3D human action analysis with temporal convolutional networks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2017, pp. 20–28.

[134] H. Wang and L. Wang, "Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2017.

[135] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[136] N. Bu, M. Okamoto, and T. Tsuji, "A hybrid motion classification approach for EMG-based human–robot interfaces using Bayesian and neural networks," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 502–511, 2009.

[137] F. De la Torre, J. Hodgins, A. Bargteil, and others., "Guide to the carnegie mellon university multimodal activity (CMU-MMAC) database," *Robotics Institute*, p. 135, 2008.

[138] C. Chen, R. Jafari, and N. Kehtarnavaz, "UTD-MHAD: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor," in *Proceedings of the IEEE International Conference on Image Processing*, 2015, pp. 168–172.

[139] D. Wang, W. Ouyang, W. Li, and D. Xu, "Dividing and aggregating network for multi-view action recognition," in *Proceedings of the European Conference on Computer Vision*, September 2018.

[140] L. Wang, C. Gao, L. Yang, Y. Zhao, W. Zuo, and D. Meng, "PM-GANs: Discriminative representation learning for action recognition using partial-modalities," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 384–401.

[141] A. Shahroudy, T.-T. Ng, Y. Gong, and G. Wang, "Deep multimodal feature analysis for action recognition in RGB-D videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 5, pp. 1045–1058, 2018.

[142] J. Hoffman, S. Gupta, and T. Darrell, "Learning with side information through modality hallucination," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2016, pp. 826–834.

[143] L. Tran, X. Liu, J. Zhou, and R. Jin, "Missing modalities imputation via cascaded residual autoencoder," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2017, pp. 1405–1414.

[144] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim, "First-person hand action benchmark with RGB-D videos and 3D hand pose annotations," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2018, pp. 409–419.

[145] J. Zheng, Z. Jiang, and R. Chellappa, "Cross-view action recognition via transferable dictionary learning," *IEEE Transactions on Image Processing*, vol. 25, no. 6, pp. 2542–2556, 2016.

[146] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, vol. 103, no. 1, pp. 60–79, 2013.

[147] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem, "SOD-MTGAN: Small object detection via multi-task generative adversarial network," in *Proceedings of the European Conference on Computer Vision*, 2018.

[148] Q. Wang, Z. Ding, Z. Tao, Q. Gao, and Y. Fu, "Partial multi-view clustering via consistent GAN," in *Proceedings of the IEEE International Conference on Data Mining*, 2018, pp. 1290–1295.

[149] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[150] L. Wang, Z. Ding, S. Han, J.-J. Han, C. Choi, and Y. Fu, "Generative correlation discovery network for multi-label learning." in *Proceedings of the IEEE International Conference on Data Mining*, 2019.

[151] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[152] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, p. 27, 2011.

[153] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "ActionVLAD: Learning spatio-temporal aggregation for action classification," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, vol. 2, 2017, p. 3.

[154] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *Proceedings of the European Conference on Computer Vision*, 2016, pp. 20–36.

[155] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[156] G. Chartrand, *Introductory graph theory*. Courier Corporation, 1977.

[157] Z. Zeng, A. K. Tung, J. Wang, J. Feng, and L. Zhou, "Comparing stars: On approximating graph edit distance," *VLDB Endowment*, vol. 2, no. 1, pp. 25–36, 2009.

[158] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Proceedings of the IEEE International Conference on Data Mining*, 2005, pp. 8–pp.

[159] H. Kashima, K. Tsuda, and A. Inokuchi, "Marginalized kernels between labeled graphs," in *Proceedings of the International Conference on Machine Learning*, 2003, pp. 321–328.

[160] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, "Graph kernels," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1201–1242, 2010.

[161] T. Horváth, T. Gärtner, and S. Wrobel, "Cyclic pattern kernels for predictive graph mining," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004, pp. 158–167.

[162] N. Shervashidze and K. Borgwardt, "Fast subtree kernels on graphs," in *Proceedings of the Advances in Neural Information Processing Systems*, 2009, pp. 1660–1668.

[163] N. M. Kriege, F. D. Johansson, and C. Morris, "A survey on graph kernels," *Applied Network Science*, vol. 5, no. 1, pp. 1–42, 2020.

[164] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[165] Q. Liu, M. Allamanis, M. Brockschmidt, and A. Gaunt, "Constrained graph variational autoencoders for molecule design," in *Proceedings of the Advances in Neural Information Processing Systems*, 2018, pp. 7795–7804.

[166] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[167] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende *et al.*, "Interaction networks for learning about objects, relations and physics," in *Proceedings of the Advances in Neural Information Processing Systems*, 2016, pp. 4502–4510.

[168] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of the Advances in Neural Information Processing Systems*, 2016, pp. 3844–3852.

[169] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Proceedings of the Advances in Neural Information Processing Systems*, 2015, pp. 2224–2232.

[170] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley, "Molecular graph convolutions: moving beyond fingerprints," *Journal of Computer-Aided Molecular Design*, vol. 30, no. 8, pp. 595–608, 2016.

[171] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proceedings of the International Conference on Learning Representations*, 2018.

[172] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Proceedings of the Advances in Neural Information Processing Systems*, 2017, pp. 4967–4976.

[173] K. Xu, C. Li, Y. Tian, T. Sonobe, K. ichi Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *Proceedings of the International Conference on Machine Learning*, 2018.

[174] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[175] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proceedings of the International Conference on Learning Representations*, 2019.

[176] S. Micali and Z. A. Zhu, "Reconstructing Markov processes from independent and anonymous experiments," *Discrete Applied Mathematics*, vol. 200, pp. 108–122, 2016.

[177] S. Ivanov and E. Burnaev, "Anonymous walk embeddings," in *Proceedings of the International Conference on Machine Learning*, vol. 80. PMLR, Jul 2018, pp. 2186–2195.

[178] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *Proceedings of the International Conference on Learning Representations*, 2018.

[179] X. Yan, P. S. Yu, and J. Han, "Substructure similarity search in graph databases," in *Proceedings of the ACM SIGMOD international conference on Management of data*, 2005, pp. 766–777.

[180] N. Shervashidze, P. Schweitzer, E. J. v. Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-Lehman graph kernels," *Journal of Machine Learning Research*, vol. 12, no. Sep, pp. 2539–2561, 2011.

[181] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[182] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.

[183] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[184] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv preprint arXiv:1701.07875*, 2017.

[185] M. Ring and B. M. Eskofier, "An approximation of the Gaussian RBF kernel for efficient classification with SVMs," *Pattern Recognition Letters*, vol. 84, pp. 107–113, 2016.

[186] L. Rüschendorf, "The Wasserstein distance and approximation theorems," *Probability Theory and Related Fields*, vol. 70, no. 1, pp. 117–129, 1985.

[187] B. Rozemberczki, R. Davies, R. Sarkar, and C. Sutton, "GEMSEC: Graph embedding with self clustering," *arXiv preprint arXiv:1802.03997*, 2018.

[188] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch, "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity," *Journal of Medicinal Chemistry*, vol. 34, no. 2, pp. 786–797, 1991.

[189] N. Wale, I. A. Watson, and G. Karypis, "Comparison of descriptor spaces for chemical compound retrieval and classification," *Knowledge and Information Systems*, vol. 14, no. 3, pp. 347–375, 2008.

[190] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinformatics*, vol. 21, no. suppl_1, pp. i47–i56, 2005.

[191] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, 2005, pp. 177–187.

[192] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *Proceedings of the the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1365–1374.

[193] Jianbo Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, Aug 2000.

[194] J. Wu, H. Xiong, and J. Chen, "Adapting the right measures for K-means clustering," in *Proceedings of the the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 877–886.

[195] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Proceedings of the Advances in Neural Information Processing Systems*, 2017, pp. 3391–3401.

[196] C. K. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Proceedings of the Advances in Neural Information Processing Systems*, 2001, pp. 682–688.