To Encourage or to Restrict: the Label Dependency in Multi-Label Learning

Dissertation by

Zhuo Yang

In Partial Fulfillment of the Requirements

For the Degree of

Doctor of Philosophy

King Abdullah University of Science and Technology

Thuwal, Kingdom of Saudi Arabia

June, 2022

EXAMINATION COMMITTEE PAGE

The dissertation of Zhuo Yang is approved by the examination committee

Committee Chairperson: Prof. Xiangliang Zhang
Committee Members: Prof. Di Wang, Prof. Mikhail Moshkov, Prof. Zhuo Feng

ABSTRACT

To Encourage or to Restrict: the Label Dependency in Multi-Label Learning

Zhuo Yang

Multi-label learning addresses the problem that one instance can be associated with multiple labels simultaneously. Understanding and exploiting the Label Dependency (LD) is well accepted as the key to build high-performance multi-label classifiers, i.e., classifiers having abilities including but not limited to generalizing well on clean data and being robust under evasion attack.

From the perspective of generalization on clean data, previous works have proved the advantage of exploiting LD in multi-label classification. To **further verify the positive role of LD in multi-label classification** and address previous limitations, we originally propose an approach named Prototypical Networks for Multi-Label Learning (PNML). Specially, PNML addresses multi-label classification from the angle of estimating the positive and negative class distribution of each label in a shared nonlinear embedding space. PNML achieves the State-Of-The-Art (SOTA) classification performance on clean data.

From the perspective of robustness under evasion attack, as a pioneer, we firstly define the attackability of an multi-label classifier as the expected maximum number of flipped decision outputs by injecting budgeted perturbations to the feature distribution of data. Denote the attackability of a multi-label classifier as C^*, and the empirical evaluation of C^* is an NP-hard problem. We thus develop a method named Greedy Attack Space Exploration (GASE) to estimate C^* efficiently. More interestingly, we derive an information-theoretic upper bound for the adversarial risk

faced by multi-label classifiers. The bound unveils the key factors determining the attackability of multi-label classifiers and **points out the negative role of LD in multi-label classifiers' adversarial robustness, i.e. LD helps the transfer of attack across labels, which makes multi-label classifiers more attackable.** One step forward, inspired by the derived bound, we propose a Soft Attackability Estimator (SAE) and further develop Adversarial Robust Multi-label learning with regularized SAE (ARM-SAE) to improve the adversarial robustness of multi-label classifiers.

This work gives a more comprehensive understanding of LD in multi-label learning. The exploiting of LD should be encouraged since its positive role in models' generalization on clean data, but be restricted because of its negative role in models' adversarial robustness.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Xiangliang Zhang for the continuous support of my Ph.D study and related research, for her patience, motivation, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis.

Besides my advisor, I also want to thank Yufei Han, the co-author of my papers. Thanks for his helpful discussion in my research and comments on my writting of papers.

I also want to thank my labmates for our interesting academic talks, for their help in my life and the fun we have had in the last five years.

Last but not the least, I would like to thank my family. Thanks for their support of my pursing of a deepper understanding of knowledge. Thanks my wife for her consoling when I am down and her encouragement of my graduation.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ARM-SAE	Adversarial Robust Multi-label learning with regularized SAE
BR	Binary Relevance
CAMEL	CollAboration based Multi-labEl Learning
CMI	Conditional Mutual Information
DNN	Deep Neural Nets
ECC	Ensembles of Classifier Chains
EPS	Ensembles of Pruned Sets
GASE	Greedy Attack Space Exploration
JFSC	Joint Feature Selection and Classification for Multilabel Learning
LD	Label Dependency
LIFT	Label specIfic FeaTures
LM-KNN	Large Margin-K Nearest Neighbor
LP	Label Power-set
MINLP	Mixed-Integer Non-Linear constraint Problem
MLP	Multi-Layer Perceptron
MT-LMNN	Multi Task-Large Margin Nearest Neighbor
PGD	Projected Gradient Descent
PNML	Prototypical Networks for Multi-Label Learning
RAKEL	RAndom K labELsets
SAE	Soft Attackability Estimator
SOTA	State-Of-The-Art
SVM	Support Vector Machine

LIST OF SYMBOLS

C^*	Indicates the attackability evaluation of one multi-label classifier
$\mathcal{D}$	Indicates the underlying data distribution
$\mathbf{e}$	Indicates the embedding vector of feature vector $\mathbf{x}$
$\mathbb{E}_{neg\_k}$	Indicates the set of embeddings of negative instances for label k
$\mathbb{E}_{pos\_k}$	Indicates the set of embeddings of positive instances for label k
$\mathbb{X}_{pos\_k}$	Indicates the set of positive instances for label k
ε	Indicates the budget for evasion attack
F	Indicates a trained multi-label classifier
f_j	Indicates the classifier for label j
K	Indicates the number of labels
l_k+	Indicates the positive component of label k
l_k-	Indicates the negative component of label k
l_k	Indicates label k
$\boldsymbol{\mu}_c$	Indicates the expectation of the cluster c
$\mu(\boldsymbol{\theta})$	Indicates the expectation of the exponential family distribution
$\boldsymbol{\Omega}^{+/-}$	Indicates the set composed of learnable mixing coefficient and $\boldsymbol{\theta}_s^{+/-}$
$\mathbb{P}_{neg\_k}$	Indicates the prototype set for the negative component of label k
$\mathbf{P}_{neg\_k}$	Indicates the prototype vector for the negative component of label k
$\mathbb{P}_{pos\_k}$	Indicates the prototype set for the positive component of label k

$\mathbf{P}_{pos\_k}$	Indicates the prototype vector for the positive component of label k
$\mathbf{r}$	Indicates the perturbation vector
$\boldsymbol{\theta}_s^{+/-}$	Indicates the density function parameters of the mixture density models describing positive/negative component
$\mathbf{U}_k$	Indicates the distance metric for label k
$\mathbf{X}$	Indicates the feature matrix of the training data set
$\mathbf{x}$	Indicates the feature vector of one instance
$\mathbf{x}_i$	Indicates the feature vector of instance i
$\mathbf{Y}$	Indicates the label matrix of the training data set
$\mathbf{y}$	Indicates the label vector of one instance
y_i^k	Indicates the k-th element of label vector $\mathbf{y}$ for instance i
y^j	Indicates the j-th element of label vector $\mathbf{y}$
$\mathbf{z}$	Indicates a multi-label instance composed by feature vector $\mathbf{x}$ and label vector $\mathbf{y}$
$\mathbf{z}^n$	Indicates a multi-label data set including n instances

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Multi-label learning addresses the problem that one instance can be associated with multiple labels simultaneously. For example, people may have interests to several objects in one image, one article can be attached with multiple markers on Wikipedia, one movie can be labeled with *funny* and *romantic* annotation at the same time, etc. Formally, the goal of multi-label learning is to learn a function F, which maps an instance $\mathbf{x} \in \mathbb{R}^D$ to a label vector $\mathbf{y}= [l_1, l_2, \cdots, l_K]$ (l_k is 1 if $\mathbf{x}$ is associated with the k-th label, and l_k is -1 otherwise). Many real-world applications drive the study of this problem, such as the mentioned image object recognition [1, 2], text classification [3, 4], music retrieval [5] and bioinformatics [6].

The most remarkable characteristic in multi-label learning is the dependency among labels, i.e. one instance's ownership of some labels may have influence on its' ownership of other labels. Let's take a toy example to explain the dependency. Imagine a picture depicting an object in an area of blue. Usually, it's hard to tell if the blue area is sea or sky without other information. While, by leveraging the label of the object, we can easily recognise the blue area as sea if the object is a ship, or sky if the object is an airplane. Of course, the LD in practice is usually much more complicated than what is described in this toy example. While no matter how complicated it is, the common sense is that this dependency among labels plays an important role in the learning of multi-label classifier, and it's exactly this dependency which makes multi-label learning such a complicated and interesting problem. Undoubtedly, for the building of high-performance multi-label models, we must have

a good understanding about the role that LD plays in multi-label learning.

People usually have an impression that LD plays a positive role in multi-label learning with the setting that data is clean without noise, i.e. exploiting LD can help models generalize better on clean data. This positive role has been supported by many successful methods [7, 8, 9, 10, 11, 12, 13]. For example, Joint Feature Selection and Classification for Multilabel Learning (JFSC) calculates a pairwise label correlation matrix to regularize the multi-label learning objective [7], CollAboration based Multi-labEl Learning (CAMEL) learns to represent one given label by a linear combination of all labels [10], and so on. All these methods have shown their effectiveness in the improvement of generalization on clean data.

However, everything has two sides. machine learning models have been proved venerable to small designed perturbations [14, 15]. That is by injecting designed noise into the input, we can make a well trained model give wrong prediction easily, and this is known as adversarial evasion attack. The adversarial robustness under evasion attack has been the key to trustworthy machine learning and attracts lots of studies [16, 17, 18, 19, 14, 20, 21, 22, 23, 24, 25, 26, 27]. An important yet unsolved problem is whether **the introduction of LD hurts the adversarial robustness of multi-label classifiers.**

The adversarial robustness of multi-label classifiers. Despite of the widely existence of multi-label attack threats [28, 29, 30], it has been a rarely investigated, yet important topic to evaluate the adversarial robustness of a multi-label classifier (a.k.a. attackability). Intuitively, assessing the attackability of a multi-label classifier F with an input instance is to explore the maximal perturbation on F's output that an input adversarial noise of bounded magnitudes can ever cause. The problem of attackability assessment in a general setting can be defined as: given a magnitude bound of the adversarial perturbation and the distribution of legal input data instances, what is the worst-case missclassification risk of F under the attack? Classifier F is more

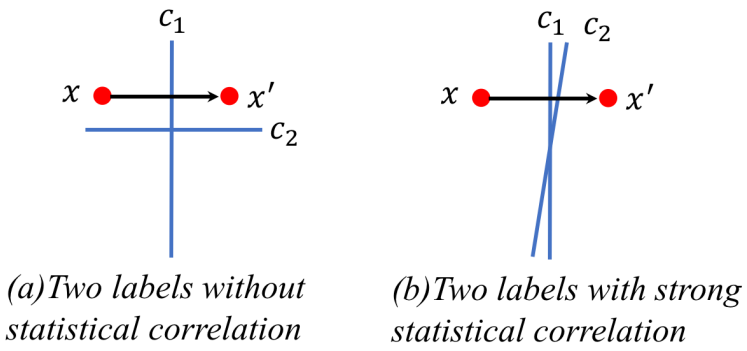


Figure 1.1: A toy example of multi-label evasion attack at two different conditions of label correlation.

attackable if it has a higher risk, while F is certified to be not attackable if its output cannot be changed by any adversarial noise within the magnitude bound.

Roughly, Fig. 1.1 demonstrates a toy example to give us some intuition about the role of LD in the assesment of attackability of multi-label models. Simply, Fig. 1.1 depicts a threat scenario with two labels l_1 and l_2, with decision hyperplanes c_1 and c_2, respectively. Fig.1 (a) assumes no statistical correlation between the two labels. Thus, c_1 and c_2 are orthogonal. In contrast, the two boundaries are well aligned in Fig.1 (b), implying a strong correlation between l_1 and l_2. The injected evasion noise in both scenarios has the same magnitude to change x to be x', indicating the same attack strength. As we can see, the evasion attack can flip simultaneously the classifier's output with respect to both labels in Fig.1.1 (b), due to the alignment between the decision boundaries of l_1 and l_2. However, in Fig.1.1 (a), the evasion perturbation can only bring impacts to the decision output of l_1. As shown in the toy example, whether the attack can transfer across different labels depends on the alignment of the decision hyperplanes, which is determined intrinsically by the correlation between the labels. On the closely dependent labels, the multi-label classifier tends to produce the consistently same or converse decisions. The adversarial noise that successfully perturbs the decision over one label is likely to cause misclassification on the other labels. In other words, LD plays a negative role in multi-label classifiers' adversarial

robustness.

In summary, **LD has two sides in multi-label learning and the objective of this dissertation is to comprehensively analyze the roles of LD in multi-label learning**, i.e., the positive influence of LD on models' generalization on clean data, as well as its' negative impact on models' adversarial robustness. Specially, this dissertation targets to answer following questions:

- The previous study has proved the positive role of LD in models' generalization on clean data. However, these methods suffer from limitations of ignoring the information from feature side and oversimplifying the label dependency by using linear extractors. Can we propose a new method to address these limitations, and reverify the positive role of LD in multi-label classification further.

- With the intuition that the strengthened LD can make multi-label models more attackable, can we derive any analytical relation between LD and the attackability of multi-label models? One step further, can we improve the adversarial robustness of multi-label models, while preserving the generalization capability?

By answering the questions above, this dissertation makes the following contributions:

- We propose PNML as a new solution to the multi-label classification problem. PNML estimates the positive and negative class distribution of each label in a shared nonlinear embedding space, which effectively catches the nonlinear LD and feature-label predicative relation. Mixture density estimation and distance metric learning are then employed to model the distribution of positive/negative component of each label. As another support to the positive role of LD in multi-label classification, PNML achieves the SOTA generalization performance on clean data.

- As a pioneer, we firstly define the attackability of multi-label models, which is denoted by C^*. Specially, the calculation of C^* is NP-hard. We then reformulate

the calculation of C^* as a bilevel set function optimization problem. By proving the submodularity of the reformulated problem, our proposed greedy-based method GASE obtains guaranteed performance.

- We establish an information-theoretic upper bound of the adversarial risk faced by multi-label models. On one hand, the bound unveils the key factors determining the attackability of multi-label classifiers, i.e., i) the Conditional Mutual Information (CMI) between the training data and the learnt classifier [31]; ii) the transferability level of the attack; and iii) the strength of the evasion attack. On the other hand, the bound points out the potential harm of LD to the robustness multi-label models, i.e., LD increases the attackability of multi-label models by aggravating the transfer of attack. Based on the derived bound, we propose SAE, which has the ability of measuring the transferability of attack. SAE is then integrated into the multi-label learning paradigm as a regularization term to suppress the attack transfer and enhance the adversarial robustness of the derived multi-label classifier. We name this method ARM-SAE.

- We conduct extensive experiments on real data sets to validate the correctness of our theoretical analysis and the effectiveness of our proposed methods, i.e. PNML, GASE and ARM-SAE.

The rest of this dissertation is organized as follows. In chapter 2, we review the related work. In chapter 3, we introduce the proposed method PNML, which serves as another support of the positive role of LD in multi-label classification. After that, in chapter 4, we defines the attackability of multi-label models and propose GASE to empirically estimate the attackability of multi-label models. Then, in chapter 5, we derive an upper bound of the adversarial risk faced by multi-label models and unveils the negative role of LD in the attackability measurement of multi-label models. We also propose ARM-SAE to improve the adversarial robustness of multi-label models

in this chapter. Finally, in chapter 6, we summarise our work and discuss the future work.

Chapter 2

Related Work

The previous study of LD in multi-label learning mainly focuses on the problem of classification on clean data. We will review these works in section 2.1. Since we are the first to study the influence of LD on the adversarial robustness of multi-label classifiers, there is no existing closely related work. Thus, in section 2.2, we focus on reviewing the related works studying the adversarial robustness of single-label classification. Last, in section 2.3, we discuss the related works of noise-tolerant multi-label learning, since it studies the robustness of multi-label learning from another perspective.

2.1 Multi-label Classification

There have been many designs of various types of multi-label learning models. We concentrate on the discussion of the most recent and relevant work regarding the ways of exploiting LD.

Binary Relevance (BR) based methods [32] decompose a multi-label classification problem into K independent binary classification problems while ignoring label dependency. It is known as *the first-order approach*. Besides, multi-label learning with Label specIfic FeaTures (LIFT) [33], which builds new label specific features for each instance, is also a first-order approach. In contrast, methods in [7, 8] make use of the pairwise label co-occurrence pattern, and are thus featured as *the second-order methods*. For example, in JFSC [7], a pairwise label correlation matrix is calculated

from label matrix $\mathbf{Y}$ and used to regularize the multi-label learning objective. Label Power-set (LP) [34] and approaches proposed based on LP including RAndom K labELsets (RAKEL) [35] and Ensembles of Pruned Sets (EPS) [36] exploit *higher-order* label dependency from $\mathbf{Y}$ by grouping labels as mutually exclusive meta-labels, so as to transform a multi-label classification task as a multi-class problem w.r.t. the meta-labels. Ensembles of Classifier Chains (ECC) [9] adds the predictions of previous labels to the feature vector for current label's classification, forming a chain of classifiers. Moreover, CAMEL [10] learns to represent any given label as a linear combination of all the labels of $\mathbf{Y}$. Among above approaches, JFSC [7] and CAMEL achieved SOTA performance. Other methods [11, 12, 13] exploit high-order label dependency by learning a low-rank representation of $\mathbf{Y}$ and a predicative mapping function between $\mathbf{X}$ and the low-rank label embedding. These two steps are conducted independently in these approaches. Feature consistency between $\mathbf{X}$, despite its usefulness, is not used for encoding label dependency. The recently proposed algorithms [37, 38, 39, 40] overcome this shortage by jointly learning both label embedding and the predicative mapping function. Nevertheless, the main limit of these methods is the low-rank assumption of the label matrix, which doesn't necessarily hold in practices [41]. Besides, the label dependency usually has a more complicated structure than the simple linear model.

Our proposed PNML exploits the high-order and potentially nonlinear LD by learning a nonparametric mixture distribution model for each component of different labels in *a nonlinear feature embedding space*. The positive and negative instances of each label are squeezed into separated and compact subspaces, whose distribution is modeled by prototypes. These prototypes characterize simultaneously *the LD and the predicative feature-label mapping*. Notably, Multi Task-Large Margin Nearest Neighbor (MT-LMNN) [42] and Large Margin-K Nearest Neighbor (LM-KNN) [43] also exploit both $\mathbf{X}$ and $\mathbf{Y}$ to extract the label dependency from the angle of metric

learning. MT-LMNN treats the classification of each label as an individual task and a distance metric is learned for this label to keep an instance with this label stay closer to its neighbors also with this label. Especially, the distance metric in [42] has the form of $\mathbf{M_0} + \mathbf{M_t}$, in which $\mathbf{M_0}$ is the part shared by all labels and $\mathbf{M_t}$ is the part tuned upon labels. LM-KNN maps one instance's feature profile and label vector onto a low dimensional manifold, where the projections of the features and labels stay close to each other. Our approach can be also interpreted as a metric learning process, where an instance carrying or not a specific label k is supposed to be close to the positive or negative prototypes of the label. Compared to MT-LMNN and LM-KNN, we jointly learn the prototypes and the distance metric of each label in the nonlinear embedding space. Our approach is designed to be more flexible to capture the label dependency and the feature-label relation.

Using mixture distribution in multi-label classification was previously discussed in [44, 45, 46]. However, they need to empirically set the number of component distributions in the mixtures and choose a distance metric for classification. Differently, our method directly learns both settings from the data distribution, which makes our method more flexible to capture the underlying LD.

2.2 Adversarial Robustness of Single-label Classification Models

The emergence of evasion attack raises a severe challenge to practitioners' trust on machine learning in performance-critic applications [47, 48]. Considerable efforts have been dedicated to detect adversarial samples, improve model designs and propose robust training methods [16, 17, 18, 19, 14, 20, 21, 22, 23, 24, 25, 26, 27, 49, 50, 51, 52, 53]. Especially, [54, 55, 56] discussed the convergence guarantee and high sample complexity of adversarial training. In contrast, few literature focuses on the essential problem of evaluating the vulnerability of a classifier under a given evasion

attack setting and identifying the key factors determining the feasibility of evasion attack against the targeted classifier. Pioneering works of this topic [24, 57, 19, 58] focused on identifying the upper bound of adversarial noise, which guarantees the stability of the targeted classifier's output, a.k.a adversarial sphere. Notably, [19] pointed out the association between adversarial robustness and the curvature of the classification boundary. Strengthened further by [59, 60, 61], the expected classification risk under adversarial perturbation can be bounded by the Rademacher complexity of the targeted classifier. Moreover in [62, 63], attackability of a recurrent neural net based classifier on discrete inputs was measured by checking the regularity of the targeted classifier. Inspiring as they are, these works focus on single-label learning tasks. The research about the attackability of multi-label models remains untouched.

2.3 Noise-tolerant Multi-label Classification Models

Another relevant topic is to learn multi-label classifiers with imperfect training instances. Miss-observations and noise corruptions of features and labels of training instances can introduce severe bias into the derived classifier. Most research efforts in this domain recognised that the key to success is to encode label correlation and the predicative relation between features and labels [64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78]. They exploited not only low-rank structures of feature/label matrices for missing data imputation, but also gained stable performances by enforcing the low-rank regularization on the predictive model capturing the feature-label correlation. Especially [79] proposed to regularize the local Rademacher complexity of a linear multi-label classifier in the training process. Nevertheless, all the previous works focus on the learning paradigm that the training data is noised and there is no adversarial attack, which is not in the scope of this dissertation.

Chapter 3

Prototypical Networks for Multi-Label Learning

In this chapter, we introduce the proposed approach PNML, which supports the positive role of LD in multi-label classifiers' generalization on clean data.

3.1 The Intuitive Idea behind PNML

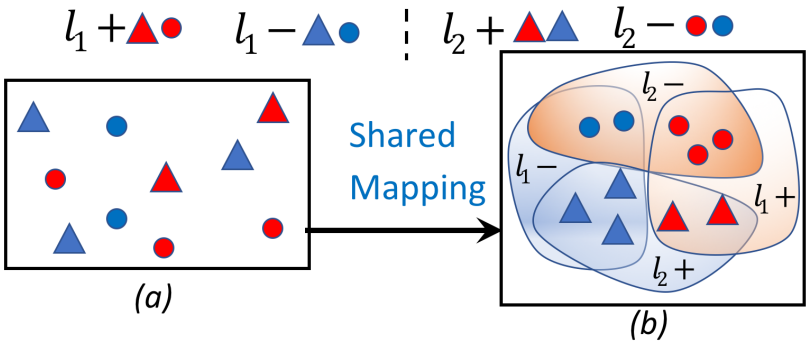


Figure 3.1: Intuition of our study: non-separable instances in the original feature space *(a)* are mapped to a non-linear embedding space *(b)* via feature embedding process applied to all labels. Label l_1 is indicated by color (red for positive and blue for negative). Label l_2 is presented by shape (triangle for positive and circle for negative). In space *(b)*, l_k+/l_k- indicates the positive/negative component of label k, and data instances dropped into the area ▲ are tagged with both labels $\{l_1,l_2\}$, those in ● or ▲ only have $\{l_1\}$ or $\{l_2\}$ and those in ● carry neither of them. Label dependency is then captured by the distribution of instance embeddings in the new space *(b)*.

PNML addresses the problem of multi-label learning from a novel perspective of distribution estimation. The intuition behind this idea is that there exists an embedding space, where the positive instances of each label distribute compactly to form a positive component. The remained negative instances form a negative component, which

separates as much as possible from the positive one. In this paper, for each label, we use positive/negative component to denote the subspace formed by the data instances of positive/negative class of the label in the embedding space. Fig. 3.1 demonstrates this intuition. Instances are mapped into the embedding space described by Fig. 3.1 *(b)*, in which for each label $l_k(k = 1, 2)$, its positive instances are grouped into the positive component l_k+, which is separated from the negative component l_k-. Given the distribution of the components l_k+ and l_k-, classification for label k can be simply done by comparing the probabilities of one instance's belongingness to the components l_k+ and l_k-, which process is promoted to all labels in the label set to finish multi-label classification. The key to capture LD in our approach is the shared mapping function applied to all labels. Instead of learning a mapping function per label independently, we learn this shared mapping function jointly with all the labels. Intuitively, this shared mapping function pushes instances carrying similar label occurrence patterns and feature profiles together in the new space, which encodes non-linear LD into the distribution of embeddings using information from both feature side and label side.

We employ mixture density estimation to describe the distribution of a positive/negative component of each label. Depending on the complexity of the distribution pattern, the positive/negative component can be modeled by a single cluster or multiple ones. For the convenience of analysis, we constrain each cluster to be modeled as an exponential family distribution function, which is actually associated to a unique Bregman divergence function [80]. With this connection, one instance's likelihood belonging to a cluster can be approximated by its Bregman-divergence based distance to the expectation of this cluster. We then define the expectation of a cluster as a **prototype**. Our proposed multi-label classification approach is thus named as **PNML** (*Prototypical Networks for Multi-Label Learning*). The proposed PNML works with two modes, namely **PNML-multiple** and **PNML-single**. With

the PNML-multiple mode, the optimal number and parameters of prototypes are learned automatically via an adaptive clustering process, which brings us strong representation power to depict the distribution of positive/negative components. The PNML-single mode is a computationally economic alternative. It computes only one prototype for each of the positive/negative component. By choosing between PNML-multiple and PNML-single, we provide a trade-off between computation efficiency and representation power. At the stage of prediction, for a given instance, its positive/negative class membership to one label is measured by its Bregman-divergence based distance to the positive/negative prototypes of this label. More specifically, unlike the simple Euclidean distance used in [81, 82], we learn a Mahalanobis distance metric for each label in PNML to cope with the varying distribution pattern of each label. In PNML, LD is actually encoded by the profiles of these learnt prototype vectors. We further exploit the LD by incorporating a label correlation-based regularization term into the object function, so as to stress LD in these prototypes .

3.2 The PNML Model

3.2.1 Overview of the PNML Model

Fig. 3.2 illustrates the overall architecture of PNML. In the training stage, the embedding module learns the non-linear feature embedding function applied to all the labels $f_\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M$. Specially, this shared embedding function f_ϕ defines the feature embedding of an input data instance based on its label membership and original feature profile. Learning the embedding representation naturally encodes the information from both the label matrix $\mathbf{Y}$ and feature matrix $\mathbf{X}$ into the distribution of the embeddings in the new space, and thus captures the non-linear LD. In Fig. 3.2, the *Embedding Layer* network f_ϕ is defined by a one-layer fully connected neural network with LeakyRelu [83] as activation function. In the new embedding space, the distribution of the data instances of each label k is described as a mixture of local

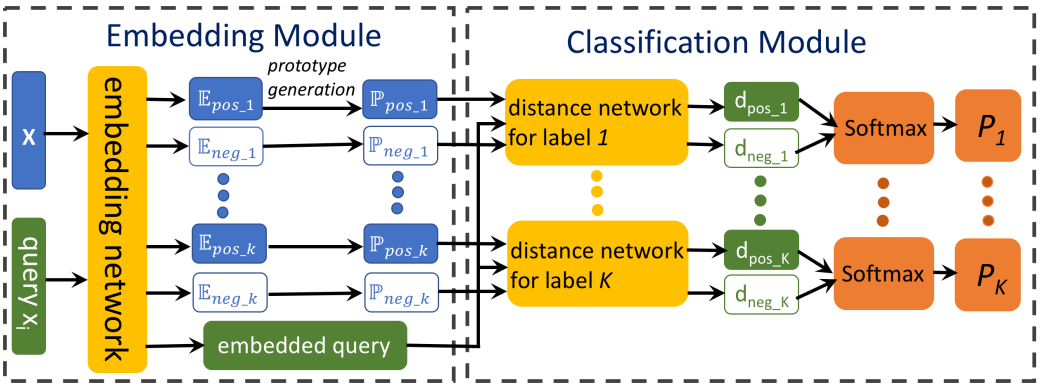


Figure 3.2: Overview of the proposed model PNML. For k=1, ..., K, $\mathbb{E}_{pos\_k}$ ($\mathbb{E}_{neg\_k}$) is the set of embeddings of positive (negative) instances for label k, i.e., $\mathbb{E}_{pos\_k}$= $\{f_\phi(\mathbf{x}), \mathbf{x} \in \mathbb{X}_{pos\_k}\}$, where $\mathbb{X}_{pos\_k}$ is the positive instance set of label k. $\mathbb{P}_{pos\_k}/\mathbb{P}_{neg\_k}$ is the positive/negative prototype set of label k. d_{pos_k}/d_{neg_k} is the distance from embedding of query $\mathbf{x}_i$ to $\mathbb{P}_{pos\_k}/\mathbb{P}_{neg\_k}$. P_k is the predicted probability of $\mathbf{x}_i$ having label k. The *embedding network* maps the instances into a common space in which instances with similar feature and label profiles will be grouped together. *Distance network for label k* learns the specific distribution pattern of embeddings for label k.

prototype sets $\mathbb{P}_{pos\_k}$ and $\mathbb{P}_{neg\_k}$ of the positive and negative components $\mathbb{E}_{pos\_k}$ and $\mathbb{E}_{neg\_k}$. $\mathbb{E}_{pos\_k} = \{f_\phi(\mathbf{x}), \mathbf{x} \in \mathbb{X}_{pos\_k}\}$, where $\mathbb{X}_{pos\_k}$ is the positive instance set of label k and vice versa. With respect to the different strategies of prototype generation, PNML works with two different modes, PNML-multiple mode and PNML-single mode:

- **PNML-multiple**: In this mode, the positive and negative prototype sets $\mathbb{P}_{pos\_k}$ and $\mathbb{P}_{neg\_k}$ are generated via a process of adaptive clustering. The number and parameters of the generated prototypes are tuned jointly in the training process. Usually multiple prototypes are generated in this self-tuning process.

- **PNML-single**: In this mode, only one prototype for the positive or negative component of each label is generated. This prototype is the expectation of the positive or negative components' distribution.

Intuitively, compared to PNML-single, PNML-multiple can derive more accurate description of the distribution of the components $\mathbb{E}_{pos\_k}$ and $\mathbb{E}_{neg\_k}$ by generating multiple prototypes for each component. However, the self-tuning prototype generation pro-

cess requires more computational cost. Alternatively, PNML-single computes only one prototype for each component. While it gains improved computationally efficiency, it may cause loss of classification accuracy. We will discuss the performances of both modes of prototype generation. A comparative study will demonstrate the trade-off between computational cost and representation/classification accuracy in the experiment section.

The classification module learns a Mahalanobis distance function for each label, based on the prototype sets $\mathbb{P}_{pos\_k}$ and $\mathbb{P}_{neg\_k}$. In the testing stage, a query instance $\mathbf{x}_i$ is classified by going through the embedding layer first to generate an embedding of $\mathbf{x}_i$. For each label, the distance between the embedding vector of $\mathbf{x}_i$ and the positive/negative prototypes is computed with the distance metric module. The derived distance indicates the likelihood of $\mathbf{x}_i$ belonging to each prototype. *softmax* is then performed on these probabilities to determine which class label $\mathbf{x}_i$ should be assigned. We elaborate the design of PNML in the followings.

3.2.2 Mixture Density Estimation

Let $\mathbf{e}= f_\phi(\mathbf{x})$ be the embedding vector of $\mathbf{x}$ in the embedding space. We assume that the class-conditional probability $p(\mathbf{e}|\hat{y}_e = \pm 1)$ of $\mathbf{e}$ belonging to positive or negative class of each label follows a mixture of the distributions of positive or negative clusters, noted as $p(\mathbf{e}|\,\mathbf{\Omega}^{+/-})$:

$$p(\mathbf{e}|\hat{y}_e = \pm 1) = p(\mathbf{e}|\,\mathbf{\Omega}^{+/-}) = \sum_{s=1}^{k_{+/-}} \pi_s^{+/-} p_\psi(\mathbf{e}|\boldsymbol{\theta}_s^{+/-}) \tag{3.1}$$

where $\mathbf{\Omega}^{+/-}= \{\pi_s^{+/-}, \boldsymbol{\theta}_s^{+/-}\}$ are the learnable mixing coefficient and density function parameters of the mixture density models of the positive and negative component for one label. $k_{+/-}$ denotes the number of positive or negative clusters, which is tuned jointly in the training process in the PNML-multiple mode. In the PNML-single mode, $k_{+/-} = 1$. For the convenience of analysis, $p_\psi(\mathbf{e}|\boldsymbol{\theta}_s^{+/-})$ is constrained to be an

exponential family distribution function with canonical parameters $\boldsymbol{\theta}_s^{+/-}$:

$$p_\psi(\mathbf{e}|\boldsymbol{\theta}_s^{+/-}) = h(\mathbf{e}) \exp(T(\mathbf{e})\boldsymbol{\theta}_s^{+/-} - \psi(\boldsymbol{\theta}_s^{+/-})) \tag{3.2}$$

where $T(\mathbf{e})$ denotes sufficient statistics of the distribution of $\mathbf{e}$ and $\psi(\boldsymbol{\theta}_s^{+/-})$ is the cumulant generating function, defined as the logarithm of the normalization factor to generate a proper probabilistic measure, and $h(\mathbf{e})$ is the carrier measure.

Given learned $\boldsymbol{\Omega}^{+/-}$, we measure the posterior probability of $\mathbf{e}$ belonging to the positive or negative class of each label as:

$$p(\hat{y}_e = +1|\mathbf{e}) = \frac{p(\mathbf{e}|\,\boldsymbol{\Omega}^{+})}{p(\mathbf{e}|\,\boldsymbol{\Omega}^{+}) + p(\mathbf{e}|\,\boldsymbol{\Omega}^{-})} \tag{3.3}$$

where the prior probabilities of the positive and negative class are set equally as 0.5. Without specified prior domain knowledge, the non-informative prior is a reasonable choice. Eq.(3.3) can be thus interpreted as a likelihood ratio test.

According to Theorem 4 in [80], there is a unique Bregman divergence function associated with every member of the exponential family. For example, squared Euclidean distance and Kullback-Leibler divergence are associated with spherical Gaussian distribution and multinomial distribution respectively. Therefore, we can reformulate the regular exponential family distribution given in Eq.(3.2) with the Bregman divergence d_φ [81, 80] as:

$$p_\psi(\mathbf{e}|\boldsymbol{\theta}_s^{+/-}) = \exp(-d_\varphi(\mathbf{e}, \mu(\boldsymbol{\theta}_s^{+/-})) - g_\varphi(\mathbf{e})) \tag{3.4}$$

where d_φ is the unique Bregman divergence function determined by the conjugate Legendre function of ψ. $g_\psi(\mathbf{e})$ absorbs all the rest terms that are not related to $\mathbf{e}$, but determined by ψ. $\mu(\boldsymbol{\theta})$ is the expectation of the exponential family distribution defined by Eq. (3.5). As we can see, the output of Eq. (3.3) is decided by Bregman

divergence $d_{\psi}(\mathbf{e}, \mu(\theta_s^{+/-}))$ from the embedding of one instance to the positive and negative cluster expectations of each label.

$$\mu(\boldsymbol{\theta}) = E_{p_{\psi}}[\mathbf{e}] = \int_{\mathbb{R}^M} \mathbf{e} p_{\psi}(\mathbf{e}|\boldsymbol{\theta}) d\mathbf{e} \tag{3.5}$$

We define the expectation of cluster in Eq. (3.5) as prototype.

Prototype Generation for PNML-single. In the PNML-single mode, one positive/negative component of label k is treated as one positive/negative cluster. The positive/negative prototype is computed as:

$$\mathbf{P}_{pos\_k/neg\_k} = \frac{1}{\left|\mathbb{X}_{pos\_k/neg\_k}\right|} \sum_{\mathbf{x}_i \in \mathbb{X}_{pos\_k/neg\_k}} f_{\phi}(\mathbf{x}_i) \tag{3.6}$$

where $\left|\mathbb{X}_{pos\_k/neg\_k}\right|$ is the size of set $\mathbb{X}_{pos\_k/neg\_k}$. Here $\mathbf{P}_{pos\_k}$ composes the positive prototype set $\mathbb{P}_{pos\_k}$ and $\mathbf{P}_{neg\_k}$ composes the negative prototype set $\mathbb{P}_{neg\_k}$.

Prototype Generation for PNML-multiple. The PNML-multiple mode proposes to model the positive or negative component with multiple prototypes (clusters). Specifically, we adopt an adaptive process in the PNML-multiple mode to determine adaptively the number and the profiles of the prototypes to fit the statistical profiles of the data instances. Without loss of generality, we use Eq. (3.4) to evaluate the probability of $\mathbf{e}$'s belonging to each cluster. To learn these prototypes, we follow the idea of infinite mixture distribution applied previously in [84, 82]. The main steps are :

1. Initialize $\boldsymbol{\mu}_c = \mu_{\mathbb{E}_{pos\_k}}$ ($\mu_{\mathbb{E}_{neg\_k}}$) as the mean of set $\mathbb{E}_{pos\_k}$ ($\mathbb{E}_{neg\_k}$), $C = 1$ as the initial number of prototypes, and $\sigma_c = \sigma$, which is the trainable variance of one cluster from which instance is assumed to be sampled. *ite_clustering* is the

iteration number of clustering.

2. Estimate the distance threshold (Eq.3.7) for creating a new prototype:

$$\lambda = -2\sigma \log \left(\frac{\alpha}{(1+\frac{\rho}{\sigma})^{M/2}} \right) \tag{3.7}$$

where ρ is a measure of the standard deviation for the base distribution from which prototypes are drawn, M is the dimension of the embedding vector and α is a hyperparameter named concentration parameter.

3. For each embedding vector $\mathbf{e}_i$ in $\mathbb{E}_{pos\_k}$ ($\mathbb{E}_{neg\_k}$), compute its' distance to each prototype c in $\{1, \cdots, C\}$ as $d_{i,c} = d_\psi(\mathbf{e_i}, \boldsymbol{\mu_c})$. If $\min_c d_{i,c} > \lambda$, set $C = C + 1$, update $\boldsymbol{\mu}_c = \mathbf{e}_i$ and $\sigma_c = \sigma$. After that, compute the normalized probability of $\mathbf{e}_i$ belonging to each cluster by $z_{i,c} = \frac{\exp(-d_\varphi(\mathbf{e}_i, \boldsymbol{\mu}_c))}{\sum_c \exp(-d_\varphi(\mathbf{e}_i, \boldsymbol{\mu}_c))}$, and then recompute the cluster mean as $\boldsymbol{\mu}_c = \frac{\sum_i z_{i,c} \mathbf{e}_i}{\sum_i z_{i,c}}$.

4. Repeat step 3 for *ite_clustering* rounds of iterations. Finally, each $\boldsymbol{\mu}_c$ is taken as a prototype and all $\boldsymbol{\mu}_c$, $c = 1, ..., C$ compose the prototype set $\mathbb{P}_{pos\_k}$ ($\mathbb{P}_{neg\_k}$).

Time Complexity of Prototype Generation. For PNML-single, we need to calculate the mean of positive component and negative component respectively for each label. The time complexity of the calculation of component centroids is $\mathcal{O}(N)$, where N is the number of training points. So the time complexity of prototype generation for all labels is $\mathcal{O}(NK)$. For PNML-multiple, we need $\mathcal{O}(N^2)$ times to adaptively generate the clusters in *Step 3*. So the complete time complexity of prototype generation is $\mathcal{O}(N^2 K * ite)$, where *ite* is the number of iteration rounds in *Step 4*.

With the non-informative class and cluster prior probability, we combine Eq. (3.1), (3.3) and (3.4) to give the posterior probability of the query $\mathbf{x}_i$ carrying the label k

as:

$$p(\hat{y}_i^k = +1|\mathbf{x}_i) = \frac{\frac{1}{\left|\mathbb{P}_{pos\_k}\right|}\sum_{c=1}^{\left|\mathbb{P}_{pos\_k}\right|}\exp(-d_\varphi(f_\phi(\mathbf{x}_i),\mathbf{P}_{pos\_k\_c}))}{\sum_{l\in\{pos,neg\}}\left[\frac{1}{\left|\mathbb{P}_{l\_k}\right|}\sum_{c=1}^{\left|\mathbb{P}_{l\_k}\right|}\exp(-d_\varphi(f_\phi(\mathbf{x}_i),\mathbf{P}_{l\_k\_c}))\right]} \tag{3.8}$$

where $\left|\mathbb{P}_{pos\_k/neg\_k}\right|$ is the size of positive/negative prototype set of label k. $\mathbf{P}_{pos\_k\_c/neg\_k\_c}$ is the c-th prototype vector in set $\mathbb{P}_{pos\_k/neg\_k}$.

Based on Eq. (3.8), we can formulate the learning objective function of PNML in Eq. (3.9) with a cross-entropy loss, where y_i^k indicates the k-th element of label vector $\mathbf{y}$ for instance i.

$$\begin{aligned} L_e = -\sum_{i=1}^{N}\sum_{k=1}^{K} & y_i^k \log p(\hat{y}_i^k = +1|\mathbf{x}_i) \\ & +(1-y_i^k)\log(1-p(\hat{y}_i^k = +1|\mathbf{x}_i)) \end{aligned} \tag{3.9}$$

3.2.3 Label-Wise Distance Metric Learning

In [80], various distance functions $d_\varphi(\cdot,\cdot)$ are presented for popular distribution functions in exponential families. For example, squared Euclidean distance $||\mathbf{e}-\mu(\boldsymbol{\theta})||^2$ is associated with multivariate spherical Gaussian distribution. Despite of the simplicity of the Euclidean distance and its effectiveness demonstrated in [81], it is not an appropriate choice in the setting of multi-label learning. In real-world data, the positive and negative components of each label may follow non-spherical Gaussian distribution.

The non-spherical distribution can be firstly caused by the instances' common membership among different categories. One component may be pulled to get close to several different components due to their common instances. Secondly, for multi-label learning, features may play different roles in different labels' discriminant processes [33, 7], which implies that each label has its own specific distribution pattern.

We therefore propose to approximate the Bregman divergence $d_\psi(\mathbf{e}, \boldsymbol{\mu}(\theta_s))$ for each label by learning a Mahalanobis distance function d_m^k in the embedding space, which shows:

$$d_m^k(\mathbf{e}, \boldsymbol{\mu}(\theta_s)) = \sqrt{(\mathbf{e} - \boldsymbol{\mu}(\theta_s))^T \mathbf{U}_k{}^T \mathbf{U}_k (\mathbf{e} - \boldsymbol{\mu}(\theta_s))} \tag{3.10}$$

The non-spherical covariance structure is approximated by the weight matrix $\mathbf{U}_k \in \mathbb{R}^{M \times M}$ learned from the data [85, 42].

In this paper, we encode $\mathbf{U}_k$ with a one-layer fully connected neural network with a linear activation function, as shown by *Distance Network* in Fig. 3.2. The learned metric function d_m^k computes $p(\hat{y}_i^k = +1|\mathbf{x}_i)$ in the loss function. $\mathbf{U}_k$ of the distance metric and the parameters of the positive and negative prototypes are jointly learned by optimizing the loss function in Eq. (3.9). Moreover, we add a regularization term defined by Eq. (3.11) to the overall loss function to prevent potential over-fitting of the distance metric learning.

$$L_m = \sum_{k=1}^{K} \|\mathbf{U}_k\|_F^2 \tag{3.11}$$

3.2.4 Label Correlation Regularizer

In the proposed method, the positive and negative prototypes can be considered as the representative feature profiles of the positive and negative class of each label. Especially, the positive prototypes depict the typical feature representation of the data instances carrying a given label. In practice, the negative instances for a label usually contain much more variances compared to the positive opponents. It is likely that the negative instances of different labels might overlap with each other to much extent. Consequently the negative prototypes of different labels may be of similar profiles. We thus enforce regularization on the positive prototypes in the objective function, which aims at enhancing the correlation between the positive prototypes of different labels in the training process. Intuitively, it is designed to make the prototypes' profiles consistent with the label correlation in the embedding space.

More specifically, if label j and k co-occur frequently in $\mathbf{Y}$, their positive prototypes should be of similar profiles and thus exhibit a large inner product. Otherwise, the inner product between them should be small. We introduce an regularization term defined as:

$$L_c = \frac{1}{2}\sum_{j=1}^{K}\sum_{k=1}^{K}(1 - c_{jk})\mu(\mathbb{P}_{pos\_j})^T\mu(\mathbb{P}_{pos\_k}) \tag{3.12}$$

where c_{jk} indicates the co-occurrence frequency measurement between j-th column and k-th column of label matrix $\mathbf{Y}$. $\mu(\mathbb{P}_{pos\_j})$ and $\mu(\mathbb{P}_{pos\_k})$ are the mean vector of prototype sets $\mathbb{P}_{pos\_j}$ and $\mathbb{P}_{pos\_k}$.

The overall loss function is given in Eq. (3.13), where λ_1 and λ_2 are the non-negative trade-off parameters.

$$L_{all} = L_e + \lambda_1 L_m + \lambda_2 L_c \tag{3.13}$$

3.2.5 Training Procedure

In the training process, for each label, we need to load the feature matrix $\mathbf{X}$ for embedding mapping and prototype learning. In the PNML-single mode, the mean of embedding set $\mathbb{E}_{pos\_k}$ and $\mathbb{E}_{neg\_k}$ needs to be computed and updated in each training iteration. In the PNML-multiple mode, it is required to compute the distance between the embedding of each instance $\mathbf{e}_i$ to the cluster centroids (prototypes). The computational cost increases significantly when the volume of the training data set becomes increasingly larger. To mitigate the issue, we can sample instances for each label to reduce the amount of data involved for prototype computing. We denote r_{pos_k} and r_{neg_k} as the sampling rates for positive and negative instances, respectively. Usually, r_{pos_k} is set bigger than r_{neg_k} to sample as many positive instances as possible. The positive instances are more informative in classification but relatively less in quantity compared to the negative instances. r_{neg_k} is smaller, in order to reduce computation cost while without losing much information. The influence of the sampling rate on

model performance will be evaluated in next section.

The training procedure first samples positive and negative instances respectively. Then the network weights for embedding f_ϕ, distance metric $d_m^k(\cdot,\cdot)$ and the prototypes' parameters are updated. Adam [86] is used as the optimizer.

3.3 Experimental Validation of the Effectiveness of PNML

3.3.1 Experimental Setup

Table 3.1: Used multi-label benchmark data sets. N/D denotes the number of instances/features of a data set. *Labels* denotes the number of labels in data set. $Card$ denotes the average number of labels associated with each instance.

data set	N	D	*Labels*	$Card$	domain
emotions	593	72	6	1.869	music
scene	2407	294	5	1.074	image
image	2000	294	5	1.240	image
arts	5000	462	26	1.636	text(web)
science	5000	743	40	1.451	text(web)
education	5000	550	33	1.461	text(web)
enron	1702	1001	53	3.378	text
genbase	662	1186	27	1.252	biology
rcv1-s1	6000	944	101	2.880	text
rcv1-s3	6000	944	101	2.614	text
rcv1-s5	6000	944	101	2.642	text
bibtex	7395	1836	159	2.402	text
corel5k	5000	499	374	3.522	image
bookmark	87856	2150	208	2.028	text
imdb	120919	1001	28	2.000	text

Data sets and Evaluation Metrics. Fifteen public benchmark data sets are used to evaluate all the involved approaches comprehensively. These data sets have different application contexts, including text, biology, music and image. Table 3.1 summarizes the details of these data sets. We choose 5 popularly applied evaluation metrics to measure the performances [87]. They are *Accuracy*, *Macro-averaging* F_1, *Micro-averaging* F_1, *Average Precision* and *Ranking Loss*.

Comparing Approaches. To comprehensively verify the performance of the proposed PNML, we compare PNML with the following 6 multi-label learning approaches, including first-order, second-order and high-order approaches. Some of these approaches achieve SOTA multi-label classification performance and half of them were proposed within 3 years. Specially, a SOTA extreme multi-label classification approach is also introduced for broader comparison. The comparing approaches are:

- *BR* [32]: It decomposes multi-label problem as K independent single-label classification problems and is a first-order approach.
- *ML-KNN* [88]: It is derived from traditional KNN method for multi-label problem and it is a high-order approach.
- *MLTSVM* [89]: It is adapted from SVM method.
- *JFSC* [7]: It does feature selection and classification for each label jointly. It is a second-order approach and is one of the SOTA approaches.
- *CAMEL* [10]: It treats each label as a linear combination of all other labels. It is a high-order approach and is one of the SOTA approaches.
- *Parabel* [90]: It is one of the SOTA extreme multi-label classification methods. Since *Parabel* is designed specially for ranking, so to keep fairness, we just compare their performance in terms of the ranking metrics, that is *Ranking Loss* and *Average Precision*.

Baseline models of *BR*, *ML-KNN* and *MLTSVM* are implemented with the scikit-multilearn package [91] and a two layer Multi-Layer Perceptron (MLP) is used as the base classifier for *BR*. The unit number of hidden layer of MLP is determined by Eq. (3.14) in which D is the original feature dimension. Besides, the number of nearest neighbors of *ML-KNN* is searched in $\{3, 5, \cdots, 21\}$. For *MLTSVM*, the

empirical risk penalty parameter c_k and regularization parameter λ_k are searched in $\{2^{-6}, 2^{-5}, 2^{-4}, \cdots, 2^4\}$. Codes and suggested parameters in the original papers are used for *JFSC*, *CAMEL* and *Parabel*.

In our approach, the embedding dimension M, slope of LeakyRelu β, concentration parameter α (only needed by PNML-multiple mode) and loss trade off parameters λ_1, λ_2 are hyperparameters to be determined. Empirically, β is set to 0.2 and M is determined by Eq.(3.14). λ_1 and λ_2 are searched in $\{10^{-7}, 5 \times 10^{-6}, 10^{-6}, 5 \times 10^{-6}, \cdots, 10^{-2}\}$. When under PNML-multiple mode, α is searched in $\{0.0001, 0.001, 0.01, 0.1, 0.5, 0.8\}$.

$$M = \begin{cases} 32 & D \leq 100 \\ 64 & 100 < D \leq 200 \\ 128 & D > 200 \end{cases} \tag{3.14}$$

3.3.2 Classification Results

For each comparing approach, 5-fold cross-validation is performed on the training data of each data set. Tables 3.2 reports the average results of each comparing algorithm over 15 data sets in terms of each evaluation metric. Table 3.3 summarizes the overall pairwise comparison results by comparing PNML-single (PNML-multiple) with other baselines. Based on these experimental results, the following observations can be made.

- Our approach *PNML-single (PNML-multiple)* outperforms the baseline approaches in most cases. Concretely, if we treat one evaluation metric for one data set as one case, there are 75 cases in total. *PNML-single* outperforms all the other baselines in 57% (43/75) evaluation cases, and *PNML-multiple* outperforms them in 77% (58/75) evaluation cases. Besides, the average rank of *PNML-multiple* is higher than all the other baselines with every evaluation metric and the average rank of *PNML-single* is higher than all the other baselines with

Table 3.2: Experimental results of evaluated algorithms on 15 data sets on 5 evaluation metrics. ↑ (↓) indicates the larger (smaller) the value, the better the performance. The best results are in bold . *AR* is the average rank of algorithm on 15 data sets with corresponding metric.

Comparing	*Accuracy* ↑															
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**	*AR*
BR	0.478	0.672	0.530	0.341	0.350	0.365	0.419	0.988	0.324	0.378	0.381	0.321	0.106	0.233	0.065	4.47
ML-KNN	0.418	0.692	0.522	0.189	0.280	0.324	0.273	0.973	0.338	0.342	0.349	0.313	0.079	0.242	0.145	5.27
MLTSVM	0.320	0.418	0.500	0.241	0.309	0.283	0.273	0.952	0.235	0.245	0.312	0.201	0.091	0.173	0.071	6.47
JFSC	0.255	0.603	0.451	0.377	0.365	0.346	0.418	**0.994**	0.354	0.391	0.397	0.352	**0.142**	0.250	**0.248**	3.60
CAMEL	0.505	0.702	0.589	0.342	0.331	0.360	0.440	0.992	0.300	0.354	0.358	0.312	0.079	0.239	0.098	4.20
Parabel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
PNML-single	0.489	0.708	0.583	0.381	0.394	0.394	0.472	0.987	0.395	0.422	0.428	0.387	0.121	0.253	0.109	2.53
PNML-multiple	**0.519**	**0.728**	**0.603**	**0.393**	**0.402**	**0.400**	**0.475**	0.990	**0.398**	**0.427**	**0.431**	**0.390**	0.123	**0.258**	0.121	1.33
Comparing	*Micro − averaging* F_1 ↑															
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**	*AR*
BR	0.609	0.735	0.615	0.400	0.410	0.442	0.540	0.990	0.426	0.435	0.449	0.426	0.185	0.244	0.105	4.73
ML-KNN	0.548	0.748	0.592	0.250	0.348	0.403	0.425	0.979	0.426	0.394	0.413	0.416	0.143	**0.317**	0.180	5.33
MLTSVM	0.484	0.559	0.560	0.319	0.376	0.376	0.436	0.963	0.379	0.362	0.453	0.340	0.160	0.226	0.137	6.27
JFSC	0.407	0.697	0.552	0.444	0.446	0.445	0.542	**0.994**	0.495	0.487	0.497	0.471	**0.243**	0.281	**0.342**	3.33
CAMEL	0.636	**0.770**	0.659	0.409	0.421	0.450	0.564	0.993	0.403	0.413	0.431	0.421	0.110	0.278	0.166	3.87
Parabel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
PNML-single	0.630	0.744	0.648	0.440	0.454	0.471	0.596	0.988	0.533	0.516	0.532	0.501	0.198	0.262	0.194	2.87
PNML-multiple	**0.653**	0.757	**0.662**	**0.446**	**0.462**	**0.474**	**0.598**	0.991	**0.534**	**0.518**	**0.540**	**0.502**	0.207	0.268	0.203	1.53
Comparing	*Macro − averaging* F_1 ↑															
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**	*AR*
BR	0.578	0.746	0.612	0.245	0.241	0.221	0.231	0.747	0.271	0.244	0.244	0.311	0.041	0.173	0.049	3.47
ML-KNN	0.528	0.755	0.589	0.146	0.169	0.176	0.140	0.674	0.239	0.184	0.208	0.267	0.027	0.191	0.040	5.33
MLTSVM	0.479	0.570	0.564	0.200	0.173	0.156	0.127	0.600	0.201	0.162	0.135	0.205	0.036	0.104	0.032	6.40
JFSC	0.352	0.701	0.546	0.235	0.222	0.144	0.196	0.746	0.256	0.225	0.232	0.354	0.039	0.136	0.080	4.80
CAMEL	0.619	**0.779**	0.662	0.232	0.206	0.198	0.212	**0.755**	0.177	0.156	0.156	0.251	0.026	0.139	0.087	4.40
Parabel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
PNML-single	0.634	0.756	0.655	**0.328**	0.296	**0.310**	0.258	0.746	0.378	0.365	0.372	**0.418**	0.064	**0.232**	**0.156**	1.87
PNML-multiple	**0.652**	0.767	**0.666**	0.321	**0.298**	0.304	**0.262**	0.733	**0.389**	**0.375**	**0.377**	0.414	**0.066**	0.228	0.144	1.67
Comparing	*Average precision* ↑															
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**	*AR*
BR	0.604	0.721	0.632	0.378	0.369	0.388	0.418	0.989	0.314	0.376	0.377	0.318	0.276	0.424	**0.504**	7.00
ML-KNN	0.652	0.824	0.682	0.380	0.410	0.453	0.521	0.982	0.537	0.541	0.558	0.348	0.218	0.299	0.399	6.93
MLTSVM	0.636	0.800	0.770	0.545	0.560	0.592	0.680	0.997	0.569	0.605	0.585	0.520	0.241	0.401	0.424	5.60
JFSC	0.720	0.851	0.788	0.604	0.592	0.626	0.641	0.997	0.600	0.619	0.630	0.594	0.292	0.441	0.496	3.87
CAMEL	0.793	**0.893**	0.825	0.596	0.604	0.621	0.678	0.997	0.604	0.633	0.631	0.606	**0.293**	0.463	0.479	2.80
Parabel	0.613	0.861	0.794	0.349	0.481	0.474	0.479	**0.998**	0.589	0.616	0.635	0.604	0.267	0.458	0.483	4.87
PNML-single	0.769	0.867	0.816	0.609	0.608	0.635	**0.682**	0.992	0.627	0.630	0.640	0.608	0.273	0.476	0.445	2.93
PNML-multiple	**0.795**	0.872	**0.828**	**0.622**	**0.611**	**0.638**	**0.682**	0.994	**0.633**	**0.635**	**0.644**	**0.611**	0.286	**0.481**	0.450	1.73
Comparing	*Ranking loss* ↓															
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**	*AR*
BR	0.490	0.289	0.437	0.611	0.591	0.587	0.477	0.010	0.583	0.546	0.535	0.616	0.153	0.095	**0.159**	6.93
ML-KNN	0.308	0.101	0.257	0.192	0.135	0.104	0.111	0.012	0.067	0.067	0.061	0.160	0.134	0.187	0.218	5.73
MLTSVM	0.374	0.102	0.175	0.128	0.099	0.077	**0.077**	0.004	0.043	**0.040**	0.060	0.065	0.143	0.172	0.204	4.00
JFSC	0.242	0.090	0.176	0.162	0.151	0.104	0.130	0.002	0.065	0.065	0.064	0.081	0.178	0.146	0.167	4.80
CAMEL	0.173	**0.065**	0.150	0.170	0.140	0.135	0.103	0.003	0.073	0.069	0.063	0.095	0.225	0.101	0.189	4.87
Parabel	0.356	0.075	0.172	0.272	0.171	0.168	0.208	**0.001**	0.068	0.065	0.061	0.084	0.312	0.132	0.163	4.93
PNML-single	0.192	0.077	0.150	0.118	0.101	0.072	0.079	0.002	0.037	0.043	0.039	0.071	0.143	0.092	0.174	2.93
PNML-multiple	**0.171**	0.076	**0.147**	**0.110**	**0.093**	**0.070**	**0.077**	**0.001**	**0.035**	0.041	**0.038**	**0.062**	**0.131**	**0.088**	0.171	1.40

four evaluation metrics except for *Average Precision*. Moreover, in Table 3.3, pairwise comparison is done between *PNML-single (PNML-multiple)* and other baseline algorithms. The "win | tie | lose" means how many times that *PNML* achieves a better/tied/worse performance than the compared algorithm. Generally, we have $15 \times 5 = 75$ cases, while for *Parabel*, we have $15 \times 2 = 30$ cases. The sign test [92] is employed to test whether *PNML-single (PNML-multiple)* achieves a competitive performance against the other comparing algorithms. If

Table 3.3: Results of pairwise comparison applied to PNML-single (PNML-multiple) with baseline algorithms.

comparing algorithms	win	tie	lose	$V/2 + 1.96\sqrt{V}/2$	superior
BR	70 (72)	0 (0)	5 (3)		yes (yes)
ML-KNN	71 (73)	0 (0)	4 (2)		yes (yes)
MLTSVM	69 (72)	1 (1)	5 (2)		yes (yes)
JFSC	61 (63)	2 (0)	12 (12)	45.987 ($V = 75$)	yes (yes)
CAMEL	55 (65)	1 (0)	19 (10)		yes (yes)
Parabel	25 (26)	0 (0)	5 (4)	20.368 ($V = 30$)	yes (yes)

the number of wins is at least $V/2 + 1.96\sqrt{V}/2$ (V is the number of cases), the algorithm is significantly better with significance level $\alpha < 0.05$. The results of sign test indicate *PNML-single (PNML-multiple)* is significantly superior to other baselines.

- In terms of evaluation metric, *PNML-single (PNML-multiple)* performs best on *Macro-averaging* F_1, which indicates *PNML-single (PNML-multiple)* is more friendly to rarely encountered labels comparing to other approaches. This observation matches the results of [81], in which prototypical networks are used to solve few-shot learning problem.

- *PNML-multiple* outperforms *PNML-single* in 69 cases, which indicates the learned multiple prototypes can describe the embedding distribution more comprehensively than one prototype.

3.3.3 Ablation Study

Without loss of generality, we tune the architecture of our proposed approach to verify the effectiveness of different parts under PNML-multiple mode.

- To confirm that our approach can learn well the label dependency and perform an effective mapping f_ϕ, we assign different independently trained embedding

layers to each label. Therefore K single-label classifications are conducted independently without interaction. We name this variant of the proposed approach as ***PNML-I***.

- To demonstrate the effectiveness of the distance metric learning component, we build another variant of our approach in which Euclidean distance is used for each label and we name it ***PNML-D***.

Table 3.4 reports the average results of *PNML-I*, *PNML-D* and *PNML-multiple*. *PNML-multiple* outperforms *PNML-I* and *PNML-D* in 71 and 75 cases respectively (75 cases in total), both of which are bigger than the threshold 45.987 of sign test in Table 3.3. These results verify that *PNML-multiple* achieves statistically superior performance against *PNML-I* and *PNML-D*, which indicates that the learned embedding function f_ϕ does transfer meaningful knowledge across labels and the learned label-specific distance metric achieves better distribution modelling than Euclidean.

Table 3.4: Experimental results of ablation study on 15 data sets on 5 evaluation metrics. ↑ (↓) indicates the larger (smaller) the value, the better the performance. The best results are in bold .

Comparing	*Accuracy* ↑														
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**
PNML-I	0.492	0.697	0.572	0.364	0.351	0.385	0.420	0.987	0.352	0.384	0.397	0.356	0.121	0.221	0.108
PNML-D	0.415	0.669	0.556	0.361	0.366	0.386	0.430	0.974	0.366	0.387	0.389	0.307	0.109	0.198	0.089
PNML-multiple	**0.519**	**0.728**	**0.603**	**0.393**	**0.402**	**0.400**	**0.475**	**0.990**	**0.398**	**0.427**	**0.431**	**0.390**	**0.123**	**0.258**	**0.121**
Comparing	*Micro − averaging* F_1 ↑														
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**
PNML-I	0.628	0.751	0.642	0.417	0.407	0.450	0.540	0.989	0.470	0.453	0.475	0.452	0.190	0.239	0.191
PNML-D	0.552	0.711	0.623	0.421	0.425	0.460	0.556	0.974	0.501	0.478	0.499	0.411	0.174	0.214	0.183
PNML-multiple	**0.653**	**0.757**	**0.662**	**0.446**	**0.462**	**0.474**	**0.598**	**0.991**	**0.534**	**0.518**	**0.540**	**0.502**	**0.207**	**0.268**	**0.203**
Comparing	*Macro − averaging* F_1 ↑														
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**
PNML-I	0.641	0.762	0.642	0.288	0.240	0.255	0.242	**0.737**	0.343	0.280	0.294	0.348	0.061	0.224	**0.146**
PNML-D	0.554	0.739	0.633	0.313	0.282	0.280	0.253	0.719	0.367	0.353	0.354	0.375	0.058	0.209	0.137
PNML-multiple	**0.652**	**0.767**	**0.666**	**0.328**	**0.298**	**0.311**	**0.262**	0.733	**0.389**	**0.375**	**0.377**	**0.420**	**0.066**	**0.228**	0.144
Comparing	*Average precision* ↑														
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**
PNML-I	0.791	0.867	0.815	0.601	0.571	0.620	0.630	**0.996**	0.580	0.591	0.594	0.544	0.271	0.432	0.389
PNML-D	0.693	0.860	0.807	0.601	0.599	0.627	0.646	0.989	0.593	0.604	0.606	0.577	0.260	0.417	0.384
PNML-multiple	**0.795**	**0.872**	**0.828**	**0.622**	**0.611**	**0.638**	**0.682**	0.994	**0.633**	**0.635**	**0.644**	**0.611**	**0.286**	**0.481**	**0.450**
Comparing	*Ranking loss* ↓														
Algorithm	**emotions**	**scene**	**image**	**arts**	**science**	**education**	**enron**	**genbase**	**rcv1-s1**	**rcv1-s3**	**rcv1-s5**	**bibtex**	**corel5k**	**bookmark**	**imdb**
PNML-I	0.183	**0.075**	0.156	0.118	0.112	0.081	0.110	**0.001**	0.056	0.063	0.057	0.090	0.158	0.113	0.184
PNML-D	0.281	0.080	0.160	0.117	0.100	0.072	0.079	0.003	0.040	0.044	0.045	0.070	0.164	0.127	0.193
PNML-multiple	**0.171**	0.076	**0.147**	**0.110**	**0.093**	**0.070**	**0.077**	**0.001**	**0.035**	**0.041**	**0.038**	**0.062**	**0.131**	**0.088**	**0.171**

3.3.4 Influence of Instance Sampling Rate

In our approach, to reduce the computational cost, we sample positive instances and negative instances for each label at rate r_{pos_k} and r_{neg_k} in the training process. Here, we show the influence of sampling rates on model predictive performance and efficiency. We choose data set *arts* for the experimental study and record the change of *Macro-averaging* F_1, *Micro-averaging* F_1 and run-time of 5 folds under different sampling rates in Fig. 3.3. Here PNML works under the PNML-single mode. It can be observed that the performance keeps nearly the same under different sampling rates, while run-time drops down with the decreasing of sampling rate. Therefore, a small sampling rate is acceptable. Empirically, r_{pos_k} is set big to sample as many as possible positive instances since they are more informative and rare, and r_{neg_k} is adjusted to sample hundreds of negative instances.

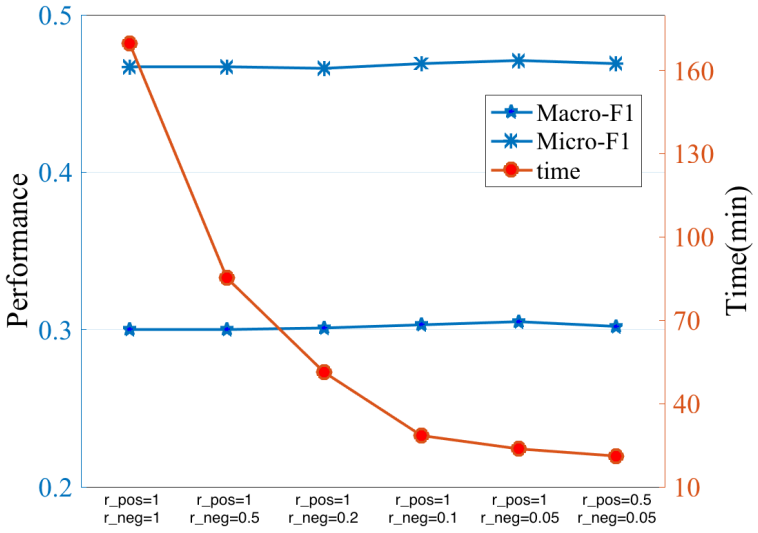


Figure 3.3: Performance and run-time under different sampling rates on data set *arts*.

3.3.5 Run Time Evaluation

In this part, we show the run time evaluation experiments of PNML. Table 3.5 lists the one-fold training time of mode PNML-single and PNML-multiple on representative

data sets under chosen sampling rates. The experiments were conducted on a Linux system using Python and our method was implemented using the Keras library. Each experiment was conducted on an Nvidia 1080TI GPU. We can observe that PNML can finish training within hours even for large data set, especially PNML-single mode. Besides, t_{single} is much smaller than $t_{multiple}$, indicating that the adaptive prototype generation process dominates the training complexity. In previous section 3.3.2, we show that using adaptive prototype generation improves the classification performance in terms of all the used five evaluation metrics, comparing to the usage of a single prototype. So, for high-accuracy demanding applications, PNML-multiple can be adopted, and for applications with efficiency requirement, PNML-single performs also well, better than the previous existing approaches.

Table 3.5: Run-time evaluation on representative data sets. N/D denotes the number of instances/features of a data set. $Labels$ denotes the number of labels in a data set. r_{pos_k} and r_{neg_k} are the sampling rates. t_{single}(h) is the total training time of PNML-single mode in hours. $t_{multiple}$ is the total training time of PNML-multiple mode.

data set	N	D	$Labels$	r_{pos_k},r_{neg_k}	t_{single}(h)	$t_{multiple}$(h)
emotions	593	72	6	1, 0.5	0.001	0.006
image	2000	294	5	1, 0.1	0.003	0.017
arts	5000	462	26	0.5, 0.05	0.071	0.597
enron	1702	1001	53	1, 0.1	0.029	0.233
yeast	2417	103	14	1, 0.1	0.008	0.063
genbase	662	1186	27	1, 0.5	0.002	0.010
medical	978	1449	45	1, 0.5	0.004	0.028
rcv1-s1	6000	944	101	0.5, 0.05	0.356	3.685
bibtex	7395	1836	159	0.5, 0.05	0.735	8.649
corel5k	5000	499	374	0.5, 0.05	0.944	10.576
bookmark	87856	2150	208	0.5, 0.002	1.371	12.469
imdb	120919	1001	28	0.5, 0.002	1.680	16.572

3.3.6 Prototypes Visualization and LD

In our approach, positive and negative prototypes are learned for positive and negative components of each label respectively. Based on our idea, positive and negative

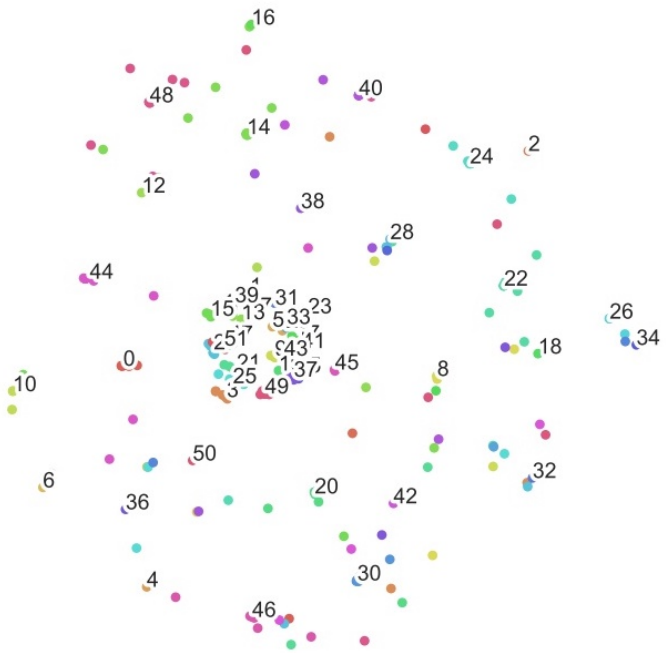


Figure 3.4: tSNE visualization of prototypes learned on data set *arts* under PNML-multiple mode. Each point corresponds to one prototype, and prototypes belonging to the same positive or negative component of one label have the same color. Odd number locates at the mean of one label's negative prototypes. Even number locates at the mean of one label's positive prototypes. For example, number 0 indicates the positive prototype mean of label 0, and number 1 indicates the negative prototype mean of label 0.

prototypes of one label should be pushed away from each other. Besides, in section 3.2.4, positive prototypes are adopted to build label correlation regularizer and the negative ones are proposed to be similar and less informative. Here we visualize the learned prototypes of data set *arts* under PNML-multiple mode in Fig. 3.4 to verify these points. We can observe in Fig. 3.4 that negative prototypes (with odd numbers) stay together, implying that they are similar, and positive prototypes (with even numbers) are pushed away from negative ones. Though the similar are these negative prototypes, they play important roles as the opposites of positives ones, which eases the classification process.

Besides, in our PNML, LD is actually encoded into prototypes, especially positive prototypes. Intuitively, if label j and label k have high positive label dependency, they

should have many common positive instances, and thus similar positive prototypes. Fig. 3.5 shows the roughly estimated label dependency represented by correlation coefficients computed by columns of label matrix $\mathbf{Y}$. In Fig. 3.5, label correlation coefficients of data set *arts* is showed and for the clear view of highly correlated labels, correlation coefficients $-0.1 < coef < 0.1$ are set to 0. Comparing Fig. 3.4 with Fig. 3.5, we can observe that prototypes present the label dependency. For example, in Fig. 3.5, label 7 has high correlation coefficients with label 20 and label 24, and you can observe in Fig. 3.4 that the positive prototypes of these three labels (located by number 14, 40 and 48) stay close. A pair of negatively correlated examples is the label 15 and label 22. We can observe that they have correlation coefficient -0.2 and the positive prototypes of them (located by number 30 and 44) are far away from each other.

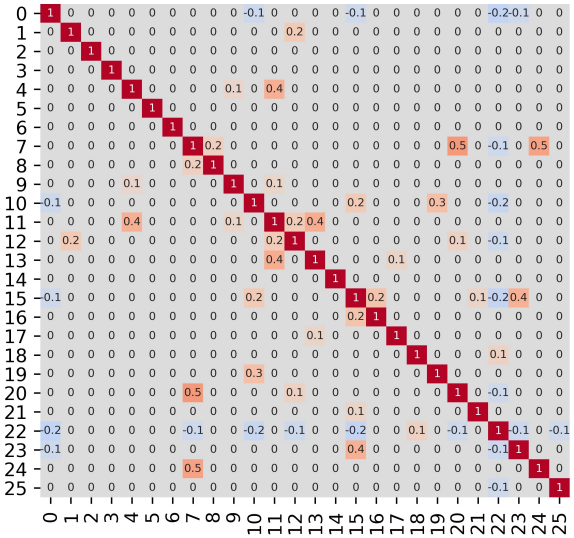
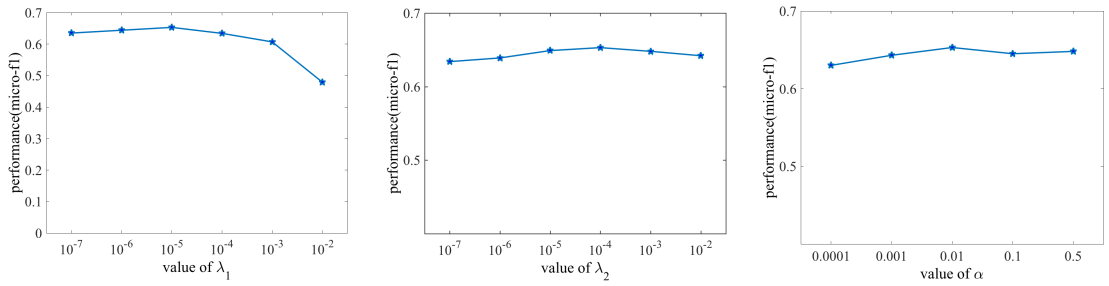


Figure 3.5: Correlation coefficients computed by label matrix of data set *arts*. Correlation coefficients $-0.1 < coef < 0.1$ are set to zero.

3.3.7 Parameter Sensitivity

In this part, we show the sensitivity of hyperparameters. There are four/five hyperparameters in PNML-single/PNML-multiple mode, M for embedding dimension, β for



(a) λ_1 (b) λ_2 (c) α

Figure 3.6: Parameter sensitivity on dataset *emotions* under mode PNML-multiple

slope of LeakyRelu, α for concentration parameter to determine the distance threshold (only used in PNML-multiple mode) and λ_1, λ_2 for loss trade off parameters. β is set as 0.2 and M is set by Eq. (3.14). These two hyperparameters are empirically set and work well for all data sets we used. For λ_1, λ_2 and α, Fig. 3.6 shows their sensitivity test on data set *emotions* with metric *Micro-averaging* F_1 under PNML-multiple mode. The sensitivity test results for λ_1 and λ_2 under PNML-single mode are omitted due to the similar trends.

3.4 Summary

In this chapter, we propose PNML to cope with challenges in multi-label learning. PNML addresses multi-label learning by mixture density estimation of positive and negative class distribution of each label in a new embedding space. The embedding space is obtained by a mapping function which can map label-wise training instances to two compact components, one for positive and the other one for negative. Then positive and negative prototypes are defined for each label based on the distribution of embeddings (PNML-multiple mode) or as the expectations of positive embeddings and negative embeddings (PNML-single mode). After that, label-wise classification is performed based on the distance from instances to positive and negative prototypes, measured by label specific distance metric learned by neural networks. Extensive experiments including ablation study clearly verify the effectiveness of our approach.

Our approach can thus serve as another strong support of the positive role of LD in multi-label classification.

Chapter 4

Attackability of Multi-Label Classifiers: Definition and Empirical Evaluation

This chapter serves as the preliminary for our study about the influence of LD on the adversarial robustness (attackability) of multi-label classifiers. Specially, we will answer two basic questions about the attackability of multi-label classifiers, i.e. its' mathematical definition and its' empirical evaluation method.

4.1 Notations and Problem Definition

Notations. We use $\mathbf{z}$= $(\mathbf{x}, \mathbf{y})$ as a multi-label instance. Let $\mathcal{D}$ be the underlying distribution of $\mathbf{z}$. Let F denote the multi-label classifier to learn from the data instances sampled from $\mathcal{D}$. The learning paradigm (possibly randomized) is thus noted as $\mathcal{A} : \mathbf{z}^n \rightarrow F$. The probability distribution of the learning paradigm is $\mathcal{P}_{\mathcal{A}}$. The corresponding loss function of $\mathcal{A}$ is $\ell : F \times \mathbf{z} \rightarrow \mathbb{R}$. $\|\mathbf{x}\|_p$ $(p \geq 1)$ denotes the L_p norm of a vector $\mathbf{x}$. Without loss of generality, we choose $p = 2$ hereafter.

Attackability of a multi-label Classifier. The attackability of F is defined as the expected maximum number of flipped decision outputs by injecting the perturbation $\mathbf{r}$ to $\mathbf{x}$ within an attack budget ε:

$$
\begin{aligned}
&C^*(\mathcal{D}) = \mathop{\mathbb{E}}_{\mathbf{z}\sim\mathcal{D}} \left(\max_{T, \|\mathbf{r}^*\| \leq \varepsilon} \sum_{j=1}^{K} \mathbb{1}(y^j \neq sgn(f_j(\mathbf{x} + \mathbf{r}^*))) \right), \\
&\text{where } \mathbf{r}^* = \mathop{argmin}_{\mathbf{r}} \|\mathbf{r}\|_p, \\
&s.t.\ y^j f_j(\mathbf{x} + \mathbf{r}^*) \leq 0 \ (j \in T),\ \ y^j f_j(\mathbf{x} + \mathbf{r}^*) > 0 \ (j \notin T).
\end{aligned}
\tag{4.1}
$$

T denotes the set of the attacked labels. y^j denotes the true label for label j. f_j

denotes the classifier for label j, thus $f_j(\mathbf{x}+\mathbf{r})$ denotes the decision score of the label j of the adversarial input. $\mathbb{1}(\cdot)$ is the indicator function. It is valued as 1 if the attack flips the decision of the label j and 0 otherwise. With the same input $\mathbf{x}$ and the same attack strength $\|\mathbf{r}\|_p$, one multi-label classifier F is more vulnerable to the evasion attack than the other F', if $C^*_F > C^*_{F'}$.

4.2 Empirical Attackability Evaluation by Greedy Exploration

4.2.1 Problem Reformulation

Solving Eq.(4.1) over legal input instances is an NP-hard Mixed-Integer Non-Linear constraint Problem (MINLP). Traditional solutions to this problem, such as Branch-and-Bound, has an exponential complexity in the worst case. To achieve an efficient evaluation, we propose to empirically approximate C^* via greedy forward expansion of the set of the attacked labels. We reformulate the label exploration problem in Eq.(4.1) as a bi-level set function optimization problem:

$$\begin{aligned} S^* &= \arg\max_S \psi(S), \\ \text{where } \psi(S) &= \max_{S,|S|\geq k} \{|S| - g(S)\}, \\ g(S) &= \min_{T\subseteq S, \|\mathbf{r}\|\leq\varepsilon} \|\mathbf{r}\|_2^2, \\ s.t.\ (1-2b_j)y^j f_j(\mathbf{x}+\mathbf{r}) &\geq t_j,\ \ j=1,2,...,m, \\ b_j = 1\ \ (\text{for } j\in T),&\ \ b_j = 0\ \ (\text{for } j\notin T), \end{aligned} \tag{4.2}$$

where label $y^j = \{+1, -1\}$, and t_j is the minimum classification margin value enforced on label j. The core components of the constraints are the binary indicators $\{b_j\}$. With $b_j = 1$, label y^j is flipped, while with $b_j = 0$, the label remains unchanged. The set function $g(S)$ returns the minimal magnitude of the perturbation $\mathbf{r}$ ever achieved via attacking the labels indicated by subsets of S. In this sense, the inner layer of Eq.(4.2) defines an evasion attack against the multi-label classifier targeting at the labels indicated by S. The optimization objective of the outer layer aims at expanding

the set S as much as possible while minimizing as much as possible the required attack cost $\|\mathbf{r}\|_2$. Notably, we set a lower bound k for $|S|$ in Eq.(4.2) for the convenience of presentation. In a naive way, we can gradually increase the lower bound k until the attack cost valued by $\psi(S)$ surpasses the budget limit. The volume $|S|$ of the derived set gives the result of C^*.

Lemma 1. *The outer layer of Eq.(4.2) defines a problem of non-monotone submodular function maximization. Let $\psi(\hat{S})$ and $\psi(S^*)$ denote respectively the objective function value obtained by randomized greedy forward search proposed in [93] and the underlying global optimum following the cardinality lower bound constraint. The greedy search based solution has the following certified approximation accuracy:*

$$\psi(\hat{S}) \geq \frac{1}{4}\psi(S^*). \tag{4.3}$$

4.2.2 Fast Greedy Attack Space Exploration

According to Lemma.1, the set S derived from the random greedy search produces an attack cost $\|\mathbf{r}\|_2$ that is close to the one achieved by the global optimum solution. It guarantees the quality of the greedy search based solution. The *primitive greedy forward expansion* is thus designed as follows:

- We initialize an empty S (flipped labels), which assumes no labels are attacked at the beginning.

- In each round of the greedy expansion, for the current set S and current adversarial noise $\mathbf{r}(S)$, we choose each of the candidate labels j out of S and compute the marginal gain $\|\mathbf{r}(S \cup j)\|_2 - \|\mathbf{r}(S)\|_2$ by conducting targeted multi-label evasion attack. $\|\mathbf{r}(S \cup j)\|_2$ is the magnitude of the adversarial noise to flip all the labels in $S \cup j$. We then select randomly one of the candidate labels j with the least marginal gains to update $S = S \cup j$.

- We update $\|\mathbf{r}\|_2$ by conducting an evasion attack targeting at the labels indicated by S. The expansion stops when $\|\mathbf{r}\|_2 \geq \mu_r$.

In each iteration, the *primitive greedy forward expansion* needs to perform evasion attack for each candidate label. It requires $(m+1)k - k(k-1)/2 - 1$ evasion attacks before including k labels in S. It is costly when the label dimension is high. To break the bottleneck, we propose a computationally economic estimator to the magnitude of the marginal gain $\Delta = \|\mathbf{r}(S \cup j)\|_2 - \|\mathbf{r}(S)\|_2$.

Lemma 2. *In each iteration of the greedy forward expansion, the magnitude of the marginal gain* Δ *is proportional to* $\frac{|y^j f_j(\mathbf{x}+\mathbf{r})|}{\|\nabla_j(\mathbf{x}+\mathbf{r})\|}$*, where* $\mathbf{r}$ *is the current feasible adversarial perturbation.* $\|\nabla_j(\mathbf{x}+\mathbf{r})\|$ *denotes the L2 norm of the gradient vector* $\frac{\partial f_j(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}$ *at the point* $\hat{\mathbf{x}} = \mathbf{x} + \mathbf{r}$.

Therefore, instead of running evasion attack for each candidate label, we can simply choose the one with the smallest ratio $\frac{|y^j f_j(\mathbf{x}+\mathbf{r})|}{\|\nabla_j(\mathbf{x}+\mathbf{r})\|}$. Algorithm 2 presents the proposed *Greedy Attack Space Expansion* (GASE) algorithm. It only runs in total $k-1$ evasion attacks to reach $|S| = k$.

In the proposed GASE algorithm, the step of *greedy label expansion* is equivalent to conducting the orthogonal matching pursuit guided greedy search [94]. It enjoys fast computation, the optimal value of the objective function in Eq.(4.2) achieved by GASE has a guaranteed approximation accuracy to the underlying global optimum according to Theorem 1.3 in [93].

The step of *greedy label expansion* in Algorithm.2 benefits from label correlation in multi-label instances. A successful attack targeted at one label tends to bias the classification output of another highly correlated label simultaneously. The candidate label with the weakest classification margin while a large $\|\nabla_j(\mathbf{x}+\mathbf{r})\|$ is thus likely to be flipped with minor update on the adversarial perturbation. Notably, the proposed GASE algorithm is independent of *the choice of evasion attack methods* in the step

targeted evasion attack. Once the greedy search for each input instance $\mathbf{x}$ finishes, we use the average $|S|$ computed over all $\mathbf{x}$ as *the empirical attackability indicator.* A larger average $|S|$ indicates a higher attackability of the targeted multi-label classifier.

Algorithm 1: Greedy Attack Space Expansion

1 **Input:** Instance example $\mathbf{x}$, a trained multi-label classifier h, perturbation norm budget μ_r.

2 **Output:** The set of attacked labels S.

3 Initialize S as an empty set and $\mathbf{r} = 0$.

4 **while** $|S| < K$ *and* $\|\mathbf{r}\|_2 < \mu_r$ **do**

5 **Greedy label expansion:** Calculate d_j in Eq.(4.4) for each label j outside S, where $f_j(\mathbf{x}+\mathbf{r})$ is the probabilistic classification output of label j, and t_j is the threshold of label decision;

$$d_j = \frac{\left|y^j f_j(\mathbf{x}+\mathbf{r})\right|}{\|\nabla_j(\mathbf{x}+\mathbf{r})\|}. \tag{4.4}$$

Update $S = S \cup j$, where label $j(j \notin S)$ is selected randomly from the labels with the least values of Eq.(4.4).

6 **Targeted evasion attack:** Solve the targeted evasion attack problem with updated S and get the optimized perturbation $\mathbf{r}^*$; Update $\mathbf{r} = \mathbf{r}^*$.

7 **end**

Time Complexity of GASE. In each iteration of *Step 4*, we need to select the least value of d_j from $K - |S|$ labels and conduct the targeted evasion attack. The whole time complexity for GASE is $\mathcal{O}(K^2 + KT_e)$, where T_e denotes the time complexity for one time of targeted evasion attack.

4.3 Experimental Validation of the Effectiveness of GASE

Data sets. We include 4 data sets collected from various real-world multi-label cyber security practices (*Creepware*) biology research (*Genbase*)[95], object recognition (*VOC2012*)[96] and environment research (*Planet*[97]). Except from the well-known *VOC2012* data set, *Creepware* data include 966 stalkerware app instances intercepted

by the mobile AV service of a private security vendor. Each app has 16 labels indicating different types of surveillance on the victim's mobile device. The surveillance types include malicious remote control functions, such as recording messages/phone calls/call logs, logging key pressing, tracking GPS locations, extracting photos, remotely accessing cameras of the victim's mobile device and so on. Each app is profiled by the introductory texts of the app available in the third-party app stores and signatures of its mobile service access. *GenBase* data set contains 662 proteins. Each protein molecule may belong to one or more classes among the 10 protein families concerned in a biomedicine clinical study. One protein molecule is described by a binary string, denoting whether or not a specific signature of the molecule structure is present. *Planet* data collects daily satellite imagery of the entire land surface of the earth at 3-5 meter resolution. Each image is equipped with labels denoting different atmospheric conditions and various classes of land cover/land use. The 4 data sets are summarized in Table.4.1.

Targeted Classifiers. We instantiate the study empirically with linear Support Vector Machine (SVM) and Deep Neural Nets (DNN) based multi-label classifiers. Linear SVM is applied on *Creepware* and *Genbase*. DNN model Inception-V3 is used on *VOC2012* and *Planet*. On each data set, we randomly choose 50%, 30% and 20% data instances for training, validation and testing to build the targeted multi-label classifier. In Table.4.1, we show *Micro-F1* and *Macro-F1* scores derived on the unperturbed testing data. Note that feature engineering and model design of the classifiers for better classification is beyond the scope of this study. These classifiers are trained to achieve comparable classification accuracy w.r.t. the reported SOTA methods on their corresponding data sets, so as to set up the test bed for the attackability analysis.

Attack. We use adversarial-robustness-toolbox [98] to implement the step of targeted adversarial attack in Algorithm.2. Specifically, *Projected Gradient Descent*

Table 4.1: Summary of the used real-world data sets. N is the number of instances. K is the total number of labels. l_{avg} is the average number of labels per instance. The F1-scores of the targeted classifiers on different data sets are also reported.

Data set	N	K	l_{avg}	Micro F1	Macro F1	Classifier$_{target}$
Creepware	966	16	2.07	0.76	0.66	SVM
Genbase	662	27	1.25	0.99	0.73	SVM
VOC2012	17,125	20	1.39	0.83	0.74	Inception-V3
Planet	40,479	17	2.87	0.82	0.36	Inception-V3

(PGD) [21] is employed to conduct the targeted attack in Algorithm.2. The decision threshold t_i in Algorithm.2 is set to 0 without loss of generality. When imposing attacks, we project the perturbed data in *VOC2012* and *Planet* to $[-1, 1]$, while we don't limit the value range of data in *Creepware* and *GenBase*.

Performance Benchmark. We gradually increase the attack strength by varying the attack budget ε. Given a fixed value of ε, we calculate *the average number of flipped labels on test data* induced by the attack. This is the empirical attackability indicator, as defined in the end of the section of *fast greedy attack space exploration.*

Implementation Platforms.: Our codes were written in Python and all the models were built by Keras package [99]. Our experiments were conducted on GPU rtx2080ti.

4.3.1 Validation of Empirical Attackability Indicator

We assess here the empirical attackability indicator estimated by the proposed GASE algorithm, by comparing it with four baselines of label exploration strategies.

- **PGS** (Primitive Greedy Search): In each round of the greedy search, the primitive greedy search method calculates the magnitude of r with the combination of the current set S and each of the candidate labels. Then it chooses the label contributing the least increasing of $\|\mathbf{r}\|^2$. Though PGS can achieve the exact greedy search, it requires to run evasion attack for each candidate label. Therefore it is significantly heavier than the proposed **GASE** method.

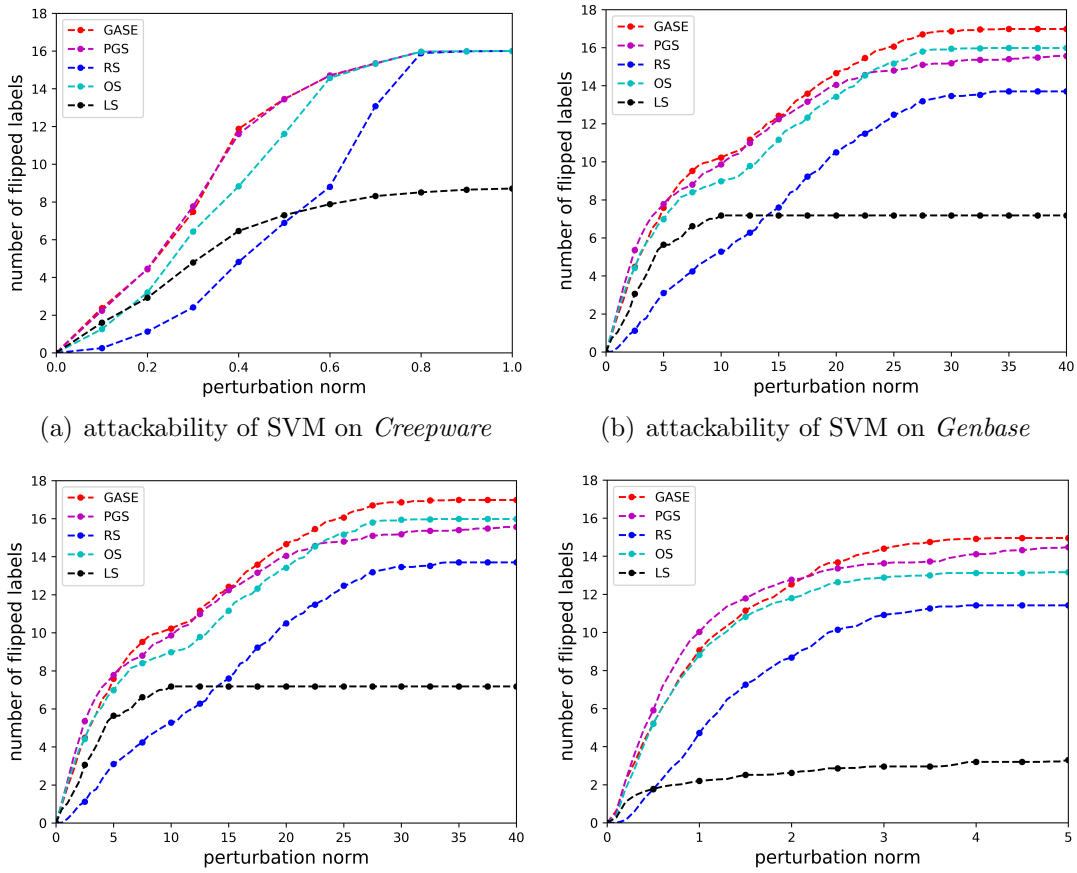
- **RS** (Random Search): In each round of RS, one label is selected purely ran-

domly from the candidate set and added to the current set S. Randomized search doesn't pursue to optimize the exploration objective function in Eq.4.2. It is involved to show the necessity of the heuristic rule in the label exploration, such as the principle of the greedy search.

- **OS** (Oblivious Search): The oblivious method doesn't conduct iterative expansion of the set. This method first compute the norm of the adversarial perturbation induced by flipping each candidate label and keeping the other labels unchanged. The labels causing the least perturbation magnitudes are selected to form the set S. It is required to check if flipping all the labels in S can deliver a feasible evasion attack.

- **LS** (Loss-guided Search) : In each iteration of the LS method, it updates the adversarial perturbation $\mathbf{r}$ along the direction where the multi-label classification loss is increased the most. The iterative update of $\mathbf{r}$ is stopped until $\|\mathbf{r}\|^2$ surpasses the cost limit. The set of the attacked labels given the derived $\mathbf{r}$ are reported. **LS** doesn't use any attack method in its implementation. It is simply a gradient ascent process. Maximizing the loss only, though sounding reasonable, is a rough search strategy. The increasing of the loss can be caused by pushing originally misclassified labels even further from the correct decision, instead of flipping originally correctly predicted labels. As a result, it misleads the search of the attackable labels.

Fig. 4.1 shows the number of flipped labels obtained by the proposed *GASE* algorithm and the baselines on linear and DNN based multi-label classifiers. Since we limit the maximum iterations and perturbation norm bounds of attacks in our experiments, few cases of the involved label exploration methods can flip all of the labels in each data set. Not surprisingly, the proposed *GASE* and *PGS* method achieve significantly more flipped labels than *RS*, *OS* and *LS* methods, especially when the



(a) attackability of SVM on *Creepware* (b) attackability of SVM on *Genbase*

(c) attackability of Inception-V3 on *VOC2012* (d) attackability of Inception-V3 on *Planet*

Figure 4.1: The empirical attackability indicator estimated by different label exploration strategies.

constraint of attack budget is strict (with small perturbation norms). It confirms the reasonableness of greedy search stated in Lemma.1. Over all the data sets, *GASE* performs similarly or even better compared to *PGS*. It empirically demonstrates the merits of *GASE*: it is much less costly than *PGS*, while obtains attackability indicators with certified quality.

4.4 Summary

The definition of attackability in this chapter focuses on the non-targeted setting, i.e., we pursue flipping as many as possible labels, while not some specific label combinations. A future work can be studying the targeted attackability of multi-

label classifiers.

To compute the empirical attackability of multi-label classifiers, we proposed the method GASE to address this NP-hard problem. GASE is efficient by leveraging each label's prediction score instead of the needed attack strength. More importantly, GASE can serve as a general baseline to evaluate multi-label classifiers' generalization performance on perturbed data.

Chapter 5

Attack Transferability Characterization for Adversarially Robust Multi-label Classification

In this chapter, we will unveil the negative role of LD in multi-label classifiers' attackability, i.e., LD will help the attack transfer among labels, which then makes the classifiers more attackable. Based on this insight, a regularization term will be designed to improve the classifiers' adversarial robustness by suppressing the transfer of attack across labels.

5.1 Information-theoretic Adversarial Risk Bound

In this section, we are interested in **1)** establishing an upper bound of the expected misclassification risk of F with the presence of adversary. It is helpful for characterizing the key factors deciding the adversarial risk of F; **2)** understanding the role of LD in shaping the adversarial threat.

For a multi-label classifier F, n legal instances $\mathbf{z}^n = \{\mathbf{z}_i\}$ (i=$1, 2, ..., n$, $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}$) and the attack budget ε, we can estimate the expected adversarial risk of F by evaluating the worst-case classification risk over the neighborhood defined in Eq. (5.1). The expected and empirical adversarial risks $R_{\mathcal{D}}(h, \varepsilon)$ and $R_{\mathcal{D}}^{emp}(h, \varepsilon)$ are given in Eq. (5.2).

$$N(\mathbf{z}_i) = \left\{ (\mathbf{x'}_i, \mathbf{y}_i) \,\middle|\, \|\mathbf{x'}_i - \mathbf{x}_i\|_p \leq \varepsilon \right\} \tag{5.1}$$

$$
\begin{aligned}
R_{\mathcal{D}}(F,\varepsilon) &= E_{\mathcal{A},\mathbf{z}^n\sim\mathcal{D}^n}[E_{\mathbf{z}\sim\mathcal{D}}[\max_{(\mathbf{x}',\mathbf{y})\in N(\mathbf{z})}\ell(F(\mathbf{x}'),\mathbf{y})]],\ \ F=\mathcal{A}(\mathbf{z}^n),\\
R^{emp}_{\mathcal{D}}(F,\varepsilon) &= E_{\mathcal{A},\mathbf{z}^n\sim\mathcal{D}^n}[\frac{1}{n}\sum_{i=1}^{n}[\max_{(\mathbf{x}'_i,\mathbf{y}_i)\in N(\mathbf{z}_i)}\ell(F(\mathbf{x}'_i),\mathbf{y}^i)]],\ \ F=\mathcal{A}(\mathbf{z}^n).
\end{aligned}
\tag{5.2}
$$

The expectation in Eq.(5.2) is taken with respect to the joint distribution $\mathcal{D}^{\otimes n}\otimes\mathcal{P}_{\mathcal{A}}$ and $\mathcal{D}^n$ denotes the data distribution with n instances. The expected adversarial risk $R_{\mathcal{D}}(F,\varepsilon)$ reflects the vulnerability level of the trained classifier F. Intuitively, a higher $R_{\mathcal{D}}(F,\varepsilon)$ indicates that the classifier F trained with the learning paradigm $\mathcal{A}$ is easier to attack (more attackable). $R^{emp}_{\mathcal{D}}(F,\varepsilon)$ is the empirical evaluation of the attackability level. By definition, if $\mathcal{A}$ is deterministic and the binary 0-1 loss is adopted, $\sum_{i=1}^{n}C^*_F(\mathbf{z}_i)$ gives $R^{emp}_{\mathcal{D}}(F,\varepsilon)$.

Theorem.1 establishes the upper bound of the adversarial risk $R_{\mathcal{D}}(F,\varepsilon)$ based on the conditional mutual information $CMI_{\mathcal{D},\mathcal{A}}$ between the legal data and the learning paradigm. Without loss of generality, the hinge loss is adopted to compute the misclassification risk of each $\mathbf{z}$, i.e., $\ell(F,\mathbf{z}=(\mathbf{x},\mathbf{y}))=\sum_{j=1}^{K}\max\{0,1-y^jf_j(\mathbf{x})\}$. We consider one of the most popularly used structures of multi-label classifiers, i.e., $F(\mathbf{x})=\mathbf{W}Rep(\mathbf{x})$, where $\mathbf{W}\in R^{m*d'}$ is the weight of a linear layer and $Rep(\mathbf{x})\in R^{d'}$ is a d'-dimensional representation vector of $\mathbf{x}\in R^d$, e.g., from a nonlinear network architecture. In Theorem.1, we assume a linear hypothesis F, i.e., $Rep(\mathbf{x})=\mathbf{x}$ for the convenience of analysis. The conclusion holds for more advanced architectures, such as feedforward neural networks.

Theorem 1. *Let $F=\mathbf{W}\mathbf{x}$ be a linear multi-label classifier. We further denote $\mathcal{D}=(\mathcal{D}_1,\cdots,\mathcal{D}_K)$ and $\mathbf{W}=(\mathbf{w}_1,\cdots,\mathbf{w}_K)$, where $\mathcal{D}_j$ is the data distribution w.r.t. each label j and $\mathbf{w}_j$ is the weight vector of the classifier of label j.*

$$
\begin{aligned}
R_{\mathcal{D}}(F,\varepsilon)\leq\ & R^{emp}_{\mathcal{D}}(F,\varepsilon)+\\
&\left(\frac{2}{n}CMI_{\mathcal{D},A}\mathop{\mathbb{E}}_{\mathbf{z}=(\mathbf{x},\mathbf{y})\sim\mathcal{D}}\left[\sup_{\mathbf{W}\in\mathcal{W}_A}\left(l(\mathbf{W},\mathbf{z})+C_{\mathbf{W},\mathbf{z}}\varepsilon\right)^2\right]\right)^{1/2},
\end{aligned}
\tag{5.3}
$$

where $\mathcal{W}_{\mathcal{A}}$ is the set including all possible weight vectors learned by $\mathcal{A}$ using the data set $\mathbf{z}^n$ sampled from $\mathcal{D}^n$. $C_{\mathbf{W},\mathbf{z}}=\max_{\{b_1,\cdots,b_K\}}\left\|\sum_{j=1}^{K}b_jy^j\mathbf{w}_j\right\|_2, b_j=\{0,1\}$. The empirical

adversarial risk $R_{\mathbf{z}^n}(A,\varepsilon)$ *has the upper bound:*

$$R^{emp}_{\mathcal{D}}(F,\varepsilon) \leq R^{emp}_{\mathcal{D}}(F,0) + \mathop{\mathbb{E}}_{\mathbf{z}^n\sim\mathcal{D}^n,\mathcal{A}}\left[\sup_{\mathbf{W}\in\mathcal{W}_{\mathcal{A}}} \mathop{\mathbb{E}}_{\mathbf{z}\in\mathbf{z}^n}\left(C_{\mathbf{W},\mathbf{z}}\varepsilon\right)\right], \tag{5.4}$$

where $R^{emp}_{\mathcal{D}}(F,0)$ *denotes the empirical and adversary-free classification risk.*

We further provide the upper bound of $CMI_{\mathcal{D},\mathcal{A}}$ *as:*

$$CMI_{\mathcal{D},\mathcal{A}} \leq ent(\mathbf{w}_1,\cdots,\mathbf{w}_m) + ent(\mathcal{D}_1,\cdots,\mathcal{D}_m) \tag{5.5}$$

where $ent(\cdot)$ *denotes the entropy of the concerned random variables.*

Key Factors of Attackability. The three key factors determining the adversarial risk (thus the attackability level) of the targeted multi-label classifier are: 1) $CMI_{\mathcal{D},\mathcal{A}}$; 2) $\mathop{\mathbb{E}}_{\mathbf{z}} C_{\mathbf{W},\mathbf{z}}$ ($\mathop{\mathbb{E}}_{\mathbf{z}\leftarrow\mathcal{D}} C_{\mathbf{W},\mathbf{z}}$ in Eq.(5.3) and $\mathop{\mathbb{E}}_{\mathbf{z}\in\mathbf{z}^n} C_{\mathbf{W},\mathbf{z}}$ in Eq.(5.4)); and 3) the attack budget ε.

The last factor of the attack budget ε is easy to understand. The targeted classifier is intuitively attackable if the adversary has more attack budget. The larger ε is, the stronger the attack becomes and the adversarial risk rises accordingly. We then analyze the first factor $CMI_{\mathcal{D},\mathcal{A}}$. For a multi-label classifier F accurately capturing the label correlation in the training data, the output from f_j and f_k are closely aligned w.r.t. the positively or negatively correlated labels j and k. Specifically, in the linear case, the alignment between f_j and f_k can be presented by $s(f_j,f_k)$=$\max\left\{cos\left\langle\mathbf{w}_j,\mathbf{w}_k\right\rangle, cos\left\langle-\mathbf{w}_j,\mathbf{w}_k\right\rangle\right\}$, where $cos\left\langle *,*\right\rangle$ denotes the cosine similarity. As shown in Eq.(5.5), the alignment of the decision hyperplanes of the correlated labels reduce the uncertainty of $\mathbf{W}=\mathcal{A}(\mathcal{D})$. Correspondingly, the conditional mutual information $CMI_{\mathcal{D},\mathcal{A}}$ decreases if the label correlation is strong and the classifier perfectly encodes the correlation into the alignment of the label-wise decision hyperplanes. According to Eq.(5.3), it is consistent with the well recognized fact of adversary-free multi-label learning: encoding the label correlation in the classifier helps to achieve an accurate adversary-free multi-label classification.

Lemma 3. $\mathop{\mathbb{E}}_{\mathbf{z}} C_{\mathbf{W},\mathbf{z}}$ *reaches the maximum value, if for each pair of labels* j *and* k, $\mathop{\mathbb{E}}_{\mathbf{z}}\{cos\left\langle y^j\mathbf{w}_j, y^k\mathbf{w}_k\right\rangle\} = 1$.

The second factor $\mathop{\mathbb{E}}_{\mathbf{z}} C_{\mathbf{W},\mathbf{z}}$ *measures the transferability of the attack noise and demonstrates the impact of the transferability level on the attackability of the classifier*. With Lemma.3, we make the following analysis. **First**, for two labels j and k with strong positive or negative correlation in the training data, a large value of $\mathop{\mathbb{E}}_{\mathbf{z}}\{cos\left\langle y^j\mathbf{w}_j, y^k\mathbf{w}_k\right\rangle\}$ indicates a high intensity of $s(f_j, f_k) = \max\{cos\left\langle \mathbf{w}_j, \mathbf{w}_k\right\rangle, cos\left\langle -\mathbf{w}_j, \mathbf{w}_k\right\rangle\}$. It represents that the decision hyperplanes $\mathbf{w}_j$ and $\mathbf{w}_k$ of the classifier F are consistently aligned. Therefore, with the same attack strength encoded by $\|r\|_2 \leq \varepsilon$, the adversarial sample $\mathbf{x}' = \mathbf{x} + \mathbf{r}$ tends to cause misclassification on both $f_j(\mathbf{x}')$ and $f_k(\mathbf{x}')$. Therefore, the attack perturbation's impact is easy to transfer between the correlated labels. Otherwise, $\mathop{\mathbb{E}}_{\mathbf{z}}\{cos\left\langle y^j\mathbf{w}_j, y^k\mathbf{w}_k\right\rangle\} = 0$ indicates an orthogonal pair of $\mathbf{w}_j$ and $\mathbf{w}_k$. The adversarial perturbation $\mathbf{r}$ may cause misclassification on one of the labels, but induce little bias to the decision output of the other. The attack cannot be transferred between the labels. Therefore, a higher / lower $\mathop{\mathbb{E}}_{\mathbf{z}} C_{\mathbf{W},\mathbf{z}}$ denotes higher / lower transferability of the attack perturbation. **Second**, according to Eq.(5.3) and Eq.(5.4), with an increasingly higher $\mathop{\mathbb{E}}_{\mathbf{z}}\{cos\left\langle y^j\mathbf{w}_j, y^k\mathbf{w}_k\right\rangle\}$, the adversarial risk of the targeted classifier F rises given a fixed attack budget ε. In summary, the alignment between the classifier's decision hyperplanes of different labels captures the label correlation. The alignment facilitates the attack to transfer across the labels. A multi-label classifier is more attackable if the attack is more transferable across the labels, as the attack can impact the decision of more labels at the same time.

Remark 1. ***Trade-off between the generalization capability of the classifier on clean data and its adversarial robustness.***

Capturing the label correlation in the learnt multi-label classifier can be a double-edged sword. ***On one hand****, encouraging alignment between the decision hyperplanes*

of the correlated labels reduces $CMI_{\mathcal{D},\mathcal{A}}$ under the adversary-free scenario ($\varepsilon = 0$ in Eq.(5.3)), thus reduces the expected misclassification risk. ***On the other hand****, the alignment between the decision hyperplanes increases the transferability of the attack, which makes the classifier more vulnerable. Controlling the alignment between the decision outputs of different labels can tune the tradeoff between the utility and the adversarial robustness of the classifier.*

5.2 Transferrability Regularization for Adversarially Robust Multi-label Classification

Following the above discussion, an intuitive solution to achieve adversarially robust multi-label classification is to regularize $\mathop{\mathbb{E}}_{\mathbf{z}\in\mathbf{z}^n} C_{\mathbf{W},\mathbf{z}}$ empirically, while minimizing the multi-label classification loss over the training data set $\mathbf{z}^n$. We denote this training paradigm as **ARM-Primal**:

$$F^* = \arg\min_{F} \frac{1}{n}\ell(F, \mathbf{z}_i) + \frac{\lambda}{n}\sum_{i=1}^{n} C_{\mathbf{W},\mathbf{z}_i} \tag{5.6}$$

where λ is the penalty parameter, and $C_{\mathbf{W},\mathbf{z}_i}$ is given as in Theorem.1. As discussed in Section.5.1, the magnitude of $C_{\mathbf{W},\mathbf{z}_i}$ in Eq.(5.6) reflects the alignment between the classifier's parameters $\{\mathbf{w}_1, ..., \mathbf{w}_K\}$. Penalizing large $C_{\mathbf{W},\mathbf{z}_i}$ thus reduces the transferability of the input attack manipulation among different labels, which makes the learnt classifier F more robust against the adversarial perturbation. However, *ARM-Primal* only considers the alignment between the parameters of the linear layer $\mathbf{w}_j$ (j=1, ..., K). This setting limits the flexibility of the regularization scheme from two perspectives. First, whether F is attackable given a bounded attack budget also depends on the magnitude of the classification margin of the input instance [100, 101]. Second, the regularization is only enforced over the linear layer's parameters of F. However, it is possible that the other layers could be relevant with the transferability of the attack noise. Adjusting the parameters of these layers can also help to control

the attackability.

As an echo, we address accordingly the limits of *ARM-Primal*: **First**, a soft attackability estimator (**SAE**) for the targeted multi-label classifier F is proposed to relax the NP-hard attackability assessment in Eq.(4.1). We show that the proposed *SAE* assesses quantitatively the transferability level of the input attack noise by considering both the alignment of the decision boundaries and the classification margin of the input data instance. The attackability of the classifier is unveiled to be proportional to the transferability of the attack. **Second**, *SAE* is then introduced as a regularization term to achieve a tunable tradeoff between transferability control and classification accuracy of the targeted classifier F. It thus reaches a customized balance between adversarial attackability and utility of F for multi-label learning practices.

5.2.1 Soft Attackability Estimator (SAE)

We first introduce the concept of *SAE* with the single-label classification setting and then extend it to the multi-label case. Suppose f is a binary classifier and instance $\mathbf{x}$ is predicted as positive if $f(\mathbf{x}) > 0$ and vice versa. Let the adversarial perturbation be decomposed as $\mathbf{r} = c\tilde{\mathbf{r}}$, where $c = \|\mathbf{r}\|_p$ and $\|\tilde{\mathbf{r}}\|_p = 1$, i.e., $\tilde{\mathbf{r}}$ shows the direction of the attack noise and c indicates the strength of the attack along this direction. For the perturbed input $\mathbf{x}' = \mathbf{x} + c\tilde{\mathbf{r}}$, the first-order approximation of $f(\mathbf{x}')$ is given as:

$$f(\mathbf{x} + c\tilde{\mathbf{r}}) = f(\mathbf{x}) + c\tilde{\mathbf{r}}^T \nabla f(\mathbf{x}), \;\; s.t. \;\; \|\tilde{\mathbf{r}}\|_p = 1, \;\; c \geq 0 \tag{5.7}$$

where $\nabla f(\mathbf{x})$ denotes the gradient of f to $\mathbf{x}$. To deliver the attack successfully, the magnitude of the attack noise follows:

$$c \geq \frac{-f(\mathbf{x})}{\tilde{\mathbf{r}}^T \nabla f(\mathbf{x})}. \tag{5.8}$$

The attackability of f on $\mathbf{x}$ along the direction of $\tilde{\mathbf{r}}$ is proportional to $\frac{1}{c}$. The smaller c is, the more attackable the classifier f becomes.

Extending the notions to the multi-label setting, we define the multi-label classifier F**'s attackability at x along the direction of** $\tilde{\mathbf{r}}$:

$$A_{F(\mathbf{x}),\tilde{\mathbf{r}}} = \sum_{j=1}^{K} \max\{\frac{-\tilde{\mathbf{r}}^T \nabla f_j(\mathbf{x})}{f_j(\mathbf{x})}, 0\}. \tag{5.9}$$

Note that in the multi-label setting, the adversarial perturbation $\tilde{\mathbf{r}}$ may cause misclassification of $\mathbf{x}$ for some labels, while enhancing the correct classification confidence for other labels, i.e., $\frac{-\tilde{\mathbf{r}}^T \nabla f_j(\mathbf{x})}{f_j(\mathbf{x})}$ can be negative for the labels with enhanced correct classification confidences. We set the corresponding attackability level to 0, as the attack perturbation fails to cause misclassification.

The intensity of $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ is proportional to the number of the labels whose decision outputs are flipped by the perturbation $\tilde{\mathbf{r}}$. Compared to the hard-count based attackability measurement C_F^* in Eq.(4.1), $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ is a soft score quantifying the impact of the attack perturbation over the outputs of the classifier. It is therefore regarded as a *soft attackability estimator*.

Transferrability defines attackability. For simplicity, we denote $\frac{-\nabla f_j(\mathbf{x})}{f_j(\mathbf{x})}$ as $\mathbf{a}_j$, and $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ can be further described as

$$\begin{aligned} A_{F(\mathbf{x}),\tilde{\mathbf{r}}} &= \tilde{\mathbf{r}}^T \sum_{j\in S, S=\{j;\mathrm{sgn}(-y^j \tilde{\mathbf{r}}^T \nabla f_j(\mathbf{x}))>0\}} \mathbf{a}_j \\ &= \|\tilde{\mathbf{r}}\|_2 \sqrt{\sum_{j\in S} \|\mathbf{a}_j\|_2^2 + 2 \sum_{j<k;j,k\in S} \|\mathbf{a}_j\|_2 \|\mathbf{a}_k\|_2 \cos\langle \mathbf{a}_j, \mathbf{a}_k\rangle} \cos\left\langle \tilde{\mathbf{r}}, \sum_{j\in S} \mathbf{a}_j \right\rangle \end{aligned} \tag{5.10}$$

As shown in Eq.(5.10), the transferability of the attack noise $\tilde{\mathbf{r}}$ is measured by the cosine similarity between $\mathbf{a}_j$ and $\mathbf{a}_k$. Each $\mathbf{a}_j$ aligns with the principal eigenvector of the Fisher Information Matrix (FIM) of f_j at the input instance $\mathbf{x}$ [102]. It depicts the local geometrical profile of the decision boundaries of different labels near $\mathbf{x}$. A larger cosine similarity between $\mathbf{a}_j$ and $\mathbf{a}_k$ indicates a stronger alignment of the decision boundaries of label j and k within the neighborhood of $\mathbf{x}$. The attack noise $\tilde{\mathbf{r}}$ thus causes closer magnitude of perturbation over $f_j(\mathbf{x})$ and $f_k(\mathbf{x})$ according to Eq.(5.9). It confirms the association between the transferability and the attackability, as unveiled by Eq.(5.3) and Eq.(5.4). Besides, the magnitude of the gradient $\mathbf{a_k} = \nabla f_k(\mathbf{x})$

also shapes the attackability level. A larger norm $\|\nabla f_k(\mathbf{x})\|_2$ indicates a less stable classification output within the L_p-ball centered at $\mathbf{x}$, i.e., a higher attackability level of the classifier. Integrating both factors, $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ is thus adopted as an empirical attackability estimator of F.

It is worth noting that **the proposed *SAE* reflects the transferability of the attack, regardless of the setting of attack budget**. As shown by Eq.(5.9), *SAE* is evaluated only with the gradient information of the classifier, which is independent of the attack capability of the adversary. In contrast, *GASE* in [103] depends on the prior knowledge about the attack budget of the adversary. In practical applications, the attack budget is usually case-dependent, which limits the use of *GASE* as a generic adversarial robustness evaluation tool. As an attack-strength-independent assessment, *SAE* can help to evaluate the attackability level of a classifier, before it is compromised by any specific attack. It is therefore can be used as a predicative guide for choosing adversarially robust multi-label learning architectures. In the linear case where $F(\mathbf{x}) = \mathbf{W}\mathbf{x}$, the cosine similarity $cos\left\langle \mathbf{a}_j, \mathbf{a}_k \right\rangle$ produces a similar alignment metric as $s(f_j, f_k) = \max\left\{cos\left\langle \mathbf{w}_j, \mathbf{w}_k \right\rangle, cos\left\langle -\mathbf{w}_j, \mathbf{w}_k \right\rangle\right\}$. According to Eq.(5.3) and (5.10), the higher the cosine similarity score $cos\left\langle \mathbf{a}_j, \mathbf{a}_k \right\rangle$ is, the higher $C_{\mathbf{W},\mathbf{z}}$ in Eq.(5.3) and $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ in Eq.(5.10) becomes. We thus measure the **attackability of F at $\mathbf{x}$** as the maximum $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ as:

$$\phi_{F,\mathbf{x}} = \max_{\tilde{\mathbf{r}}} A_{F(\mathbf{x}),\tilde{\mathbf{r}}},\ s.t.\ \|\tilde{\mathbf{r}}\|_p = 1 \tag{5.11}$$

We inherit the constraint $\|\tilde{\mathbf{r}}\|_p = 1$ from Eq.(5.7). The resultant $\tilde{\mathbf{r}}$ denotes the directions of the adversarial noise vector along which the attack can be maximally transferred. With this setting, we separate the derived transferability measurement with the attack strength. With the primal-dual conversion, we can obtain the solution

to Eq.(5.11) as:

$$\begin{aligned} \phi_{F,\mathbf{x}} &= \max_{\{b_1,b_2,\cdots,b_K\}} \left\| \sum_{j=1}^{K} \frac{-b_j \nabla f_j(\mathbf{x})}{f_j(\mathbf{x})} \right\|_q, \\ s.t.\ \ &\tfrac{1}{p} + \tfrac{1}{q} = 1,\ \ b_j = \{0,1\}, \end{aligned} \tag{5.12}$$

where p denotes the L_p norm of the perturbations. Without loss of generality, we only discuss $p = 2$ of the l_p-norm in Eq.(5.12). As the objective function of Eq.(5.12) enjoys the submodularity property [94], we employ a simple yet effective greedy based algorithm to solve Eq.(5.12). Algorithm 2 describes the greedy-search based solution to compute the *SAE* score.

Algorithm 2: The Greedy Solution to Soft Attackability Estimation

1 **Input:** $\left\{ \frac{-\nabla f_1(\mathbf{x})}{f_1(\mathbf{x})}, \cdots, \frac{-\nabla f_K(\mathbf{x})}{f_K(\mathbf{x})} \right\}$.

2 **Output:** The set of selected labels S.

3 Initialize S as an empty set. Set $LB = 0$ and $CB = 0$, where LB denotes the best result of last iteration and CB denotes the best result of current iteration.

4 **while** $|S| < K$ **do**

5 $\quad LB = CB$;

6 $\quad CB = \max_{\{1,\cdots,K\}-S} \left(\sum_{i \in S} \frac{-\nabla f_i(\mathbf{x})}{f_i(\mathbf{x})} + \frac{-\nabla f_j(\mathbf{x})}{f_j(\mathbf{x})} \right)$;

7 $\quad$ if $CB < LB$, break;

8 $\quad S = S + j$

9 **end**

Time Complexity of SAE. In each iteration of *Step 4*, we need to select the label increasing CB mostly from $K - |S|$ labels. It is easy to calculate the whole time complexity is $\mathcal{O}(K^2)$.

5.2.2 SAE Regularized Multi-label Learning

We propose to enhance the adversarial robustness of a multi-label classifier by enforcing the control over the *SAE* score of the classifier explicitly during training. While we suppose $\mathbf{x}$ is correctly classified during the theoretical analysis of attackability, it

doesn't necessarily hold during training. For an originally misclassified data instance $\mathbf{x}$, it is possible that $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ can be valued to 0. In this case, the attack perturbation $\mathbf{r}$ can augment the confidence of the misclassification. However, with $A_{F(\mathbf{x}),\tilde{\mathbf{r}}} = 0$, bare penalization can be enforced to suppress the bias. It may encourage further negative impact in the learnt classifier. To mitigate this issue, we slightly modify the definition of $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ and use it as the transferability regularization term of multi-label learning, which gives:

$$\hat{A}_{F(\mathbf{x}),\tilde{\mathbf{r}}} = \sum_{j=1}^{K} \max\{\frac{-\tilde{\mathbf{r}}^T y^j \nabla f_j(\mathbf{x})}{\max(e^{y^j f_j(\mathbf{x})}, \alpha)}, 0\}, \;\; \alpha > 0 \tag{5.13}$$

where α is set to prevent overweighing. For an originally correctly classified instance ($y^j f_j(\mathbf{x}) > 0$), $\hat{A}_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ penalizes the attack transferability as $A_{F(\mathbf{x}),\tilde{\mathbf{r}}}$. For a misclassified instance ($y^j f_j(\mathbf{x}) \leq 0$), minimizing $\hat{A}_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ helps to reduce the confidence of the misclassification output. Using the exponential function in $\hat{A}_{F(\mathbf{x}),\tilde{\mathbf{r}}}$, the misclassified instance with stronger confidence (more biased decision output) is assigned with an exponentially stronger penalty. This setting strengthens the error-correction effect of $\hat{A}_{F(\mathbf{x}),\tilde{\mathbf{r}}}$.

Similarly as in Eq.(5.12), we can define $\hat{\phi}_{F,\mathbf{x}} = \max_{\tilde{\mathbf{r}}} \hat{A}_{F(\mathbf{x}),\tilde{\mathbf{r}}}$ in Eq.(5.14). The objective function of the SAE regularized multi-label learning (named hereafter as **ARM-SAE**) gives in Eq.(5.15):

$$\begin{aligned} &\hat{\phi}_{F,\mathbf{x}} = \max_{\{b_1,b_2,\cdots,b_K\}} \left\| \sum_{j=1}^{K} \frac{-b_j y^j \nabla f_j(\mathbf{x})}{\max(e^{y^j f_j(\mathbf{x})},\alpha)} \right\|_q, \\ &s.t. \;\; \tfrac{1}{p} + \tfrac{1}{q} = 1, \;\; b_j = \{0,1\}, \end{aligned} \tag{5.14}$$

$$l = \frac{1}{n}\sum_{i=1}^{n} \ell(F, \mathbf{z}_i) + \frac{\lambda}{n}\sum_{i=1}^{n} \hat{\phi}_{F,\mathbf{x}_i}, \tag{5.15}$$

where λ is the regularization weight. $\hat{\phi}_{F,\mathbf{x}}$ can be calculated using the greedy search solution as $\phi_{F,\mathbf{x}}$. If the classifier F takes a linear form, we can find that *ARM-SAE* reweighs the linear layer parameters of the classifier $\{\mathbf{w}_1, \cdots, \mathbf{w}_K\}$ with the weight $\frac{1}{\max(e^{y^j f_j(\mathbf{x})},\alpha)}$. Compared to *ARM-Primal* (see Eq.(5.12) to $C_{\mathbf{W},\mathbf{z}}$ in Theorem

1), *ARM-SAE* enforces more transferability penalty over the instances with smaller classification margins. As unveiled in [103], these instances are easier to be perturbed for the attack purpose. Instead of penalizing each instance with the same weight as in *ARM-Primal*, *ARM-SAE* can thus perform a more flexible instance-adapted regularization.

5.3 Experiments

5.3.1 Experimental Setup

Data sets and Targeted Classifiers. We inherit the experimental study on data sets *Creepware*, *VOC2012* and *Planet*, and also the corresponding targeted classifiers introduced in Table 4.1.

Performance Benchmark. Given a fixed attack strength of ε, we compute the number of flipped labels $C_F^*(\mathbf{z})$ on each testing instance according to Eq.(4.1) and take the average of the derived $\{C_F^*(\mathbf{z})\}$ (noted as C_a) as an overall estimation of attackability on the testing data set. Due to the NP-hard intrinsic of the combinatorial optimization problem in Eq.(4.1), we use GASE to estimate empirically $C_F^*(\mathbf{z})$ and C_a.

Besides, we measure the multi-label classification performance on the clean and adversarially modified testing instances with *Micro-F1* and *Macro-F1* scores.

Input Normalization and Reproduction. When imposing attacks, we project the perturbed data in *VOC2012* and *Planet* to $[-1, 1]$, while we don't limit the value range of data in *Creepware*. The α in Eq.(5.15) is empirically set to 0.01 in all experiments. The regularization parameters λ in Eq.5.15 and other baselines are chosen empirically from the range $\left\{10^{-8}, 10^{-7}, \cdots, 10^{7}, 10^{8}\right\}$

Table 5.1: Attackability estimation by SAE. $\lambda_{nuclear}$ denotes the strength of nuclear-norm based regularization. CC and P denote the Spearman coefficient and the p-value between GASE and SAE scores on the testing instances.

Data set		$\longrightarrow$ *robustness increase*					$CC, P(Spearman)$
Creepware	$\lambda_{nuclear}$	0	0.00001	0.0001	0.001	0.01	$CC = 1$ $P = 0$
	GASE ($C_a, \varepsilon = 0.5$)	13.5	11.4	10.8	6.9	4.3	
	SAE	31.5	19.16	18.06	14.55	11.22	
VOC2012	$\lambda_{nuclear}$	0	0.0001	0.001	0.01	0.1	$CC = 1$ $P = 0$
	GASE ($C_a, \varepsilon = 10$)	10.8	10.1	9.3	8.5	4.9	
	SAE	157.6	127.3	77.6	69.1	61.0	
Planet	$\lambda_{nuclear}$	0	0.0001	0.001	0.01	0.1	$CC = 1$ $P = 0$
	GASE ($C_a, \varepsilon = 2$)	13.1	12.2	11.6	10.5	7.1	
	SAE	267.1	221.5	186.3	158.2	102.0	
		$\longrightarrow$ *attackability decrease*					

5.3.2 Effectivity of SAE

In Table.5.1, we demonstrate the validity of the proposed SAE by checking the consistency between the SAE and the GASE-based attackability measurement [103]. We adopt the nuclear-norm regularized training [103] to obtain an adversarially robust multi-label classifier. On the same training set, we increase the nuclear-norm regularization strength gradually to derive more robust architectures against the evasion attack. For each regularization strength, we can compute the SAE score of the classifier on the unperturbed testing instances. Similarly, by freezing the attack budget ε on each data set, we can generate the GASE score (C_a) corresponding to each regularization strength. Note that only the ranking orders of the SAE and GASE score matters in the attackability measurement by definition. We use the ranking relation of the scores to select adversarially robust models. Therefore, we adopt the Spearman rank correlation coefficient to measure the consistency between SAE and GASE.

We use the GASE score as a baseline of attackability assessment. The SAE and GASE score are strongly and positively correlated over all the data sets according to the correlation metric. Furthermore, with a stronger robustness regularization, the SAE score decreases accordingly. It confirms that the intensity of the proposed

SAE score capture the attackability level of the targeted classifier. This observation further validates empirically the motivation of using SAE in adjusting the adversarial robustness of the classifier.

The experimental study also shows the attack-strength-independent merit of the SAE over GASE. SAE is computed without knowing the setting of the attack budget. It thus reflects the intrinsic property of the classifier determining its adversarial vulnerability. In practice, this attack-strength-independent assessment can help to evaluate the attackability level of the deployed classifier, before it is compromised by any specific attack.

5.3.3 Effectiveness Evaluation of ARM-SAE

We compare the proposed ARM-SAE method to the SOTA techniques in improving adversarial robustness of multi-label learning: the L_2-norm and the nuclear-norm regularized multi-label training [103]. Besides, we conduct an ablation study to verify the effectiveness of ARM-SAE.

- **L_2 Norm and Nuclear Norm Regularized Training.** Enforcing the L_2 and nuclear norm constraint helps to reduce the model complexity and thus enhance the model's adversarial robustness [61, 103].

- **ARM-Single.** This variant of ARM-SAE is built by enforcing the transferability regularization with respect to individual labels separately:

$$\phi_{H\_\text{single}} = \sum_{i=1}^{n} \sum\nolimits_{k=1}^{m} \left\| \frac{\nabla h_k(x_i)}{\max(e^{y_k^i h_k(x)}, \alpha)} \right\|_2 . \tag{5.16}$$

  We compare ARM-SAE with ARM-Single to show the merit of jointly measuring and regularizing the impact of the input attack noise over all the labels. ARM-SAE tunes the transferability of the attack jointly, while ARM-Single enforces the penalization with respect to each label individually.

- **ARM-Primal.** We compare this variant to ARM-SAE to demonstrate the merit of ARM-SAE by 1) introducing the flexibility of penalizing the whole model architecture, instead of only the linear layer; 2) taking the impact of classification margin on adversarial risk [100] into the consideration.

Table 5.2: Effectiveness evaluation of ARM-SAE. For convenience, *non*, L_2, *nl*, *sg*, *pm* and *SAE* are used to denote the absence of regularization, L_2 *norm, nuclear-norm, ARM-single, ARM-Primal* and *ARM-SAE* based methods respectively. The best results are in bold.

	Creepware : Micro F1 = 0.76, Macro F1 = 0.66 (on clean data)											
Budget	$\varepsilon = 0.05$						$\varepsilon = 0.2$					
Regularizers	*non*	L_2	*nl*	*sg*	*pm*	*SAE*	*non*	l_2	*nl*	*sg*	*pm*	*SAE*
Micro F1	0.34	0.40	0.45	0.44	0.43	**0.53**	0.10	0.13	0.15	0.15	0.16	**0.22**
Macro F1	0.33	0.39	**0.43**	0.39	**0.43**	0.42	0.12	0.15	0.20	0.17	0.20	**0.25**
	*VOC*2012 : Micro F1 = 0.83, Macro F1 = 0.74 (on clean data)											
Budget	$\varepsilon = 0.1$						$\varepsilon = 1$					
Regularizers	*non*	L_2	*nl*	*sg*	*pm*	*SAE*	*non*	l_2	*nl*	*sg*	*pm*	*SAE*
Micro F1	0.49	0.53	0.56	0.54	0.57	**0.61**	0.20	0.22	0.27	0.26	0.26	**0.30**
Macro F1	0.29	0.31	0.33	0.31	0.36	**0.38**	0.12	0.16	0.22	0.17	0.20	**0.23**
	Planet : Micro F1 = 0.82, Macro F1 = 0.36 (on clean data)											
Budget	$\varepsilon = 0.1$						$\varepsilon = 1$					
Regularizers	*non*	L_2	*nl*	*sg*	*pm*	*SAE*	*non*	l_2	*nl*	*sg*	*pm*	*SAE*
Micro F1	0.41	0.49	0.45	0.48	0.49	**0.53**	0.06	0.09	0.08	0.10.	0.09	**0.13**
Macro F1	0.13	0.22	0.17	0.20	0.18	**0.24**	0.03	0.04	0.04	0.06	0.06	**0.08**

Table 5.3: Trade-off Between Generalization Performance on Clean Data and Adversarial Robustness on *Creepware*. The attack budget $\varepsilon = 0.05$.

λ	0	10^{-7}	10^{-6}	10^{-5}	10^{-4}
ϕ_{align}	0.23	0.22	0.20	0.15	0.12
Micro F1(clean)	0.76	0.76	0.75	0.72	0.70
Macro F1(clean)	0.66	0.63	0.56	0.50	0.46
Micro F1(pert)	0.34	0.35	0.39	0.44	0.53
Macro F1(pert)	0.33	0.33	0.35	0.40	0.42

Two different attack budgets ε on each data set are introduced denoting varied at-

tack strength. With each fixed ε, we compute the *Micro-F1* and *Macro-F1* scores of the targeted classifiers after retraining with the techniques above. Table.5.2 lists the classification accuracy over the adversarial testing instances using different robust training methods. In Table.5.2, we also show the multi-label classification accuracy (measured by two F1 scores) on the clean testing instances as a baseline. Consistently observed on the three data sets, even a small attack budget can deteriorate the classification accuracy drastically, which shows the vulnerability of multi-label classifiers. Generally, all the regularization method can improve the classification accuracy on the adversarial input. Among all the methods, ARM-SAE achieves the highest accuracy on the adversarial samples. It confirms the merit of SAE in controlling explicitly the transferability and then suppressing the attackability effectively. In addition, by regularizing jointly the attack transfer and exploiting classification margin for the attackability measurement, ARM-SAE achieves superior robustness over the two variants.

5.3.4 Validation of Trade-off Between Generalization Performance on Clean Data and Adversarial Robustness

We validate the tradeoff described in Remark.1. Without loss of generality, we conduct the case study on *Creepware*. Tuning the alignment between decision boundaries of different labels is achieved by conducting the ARM-SAE training as described in Eq.5.15. We freeze ε as 0.05 and vary the regularization weight λ in Eq.(5.15) from 10^{-7} to 10^{-4} to show increasingly stronger regularization effects enforced on the alignment between decision boundaries of different labels. For each regularization strength, we train a multi-label classifier F and evaluate quantitatively the averaged alignment level $\phi_{align} = \frac{1}{K^2} \sum_{j,k \in \{1,\cdots,K\}} |\cos \langle \mathbf{w}_j, \mathbf{w}_k \rangle|$ between the decision hyperplanes of different labels. Table.5.3 shows the variation of ϕ_{align} and the Micro- / Macro-F1 accuracy of the trained multi-label classifier F over the clean and adversarially per-

turbed data instances (Micros / Macro F1 (clean / pert)). With increasingly stronger robustness regularization, the averaged alignment level ϕ_{align} between the label-wise decision hyperplanes decreases accordingly. Simultaneously, we witness the rise of the classification accuracy of F on the adversarially perturbed testing instances. It indicates the classifier F is more robust to the attack perturbation. However, the Macro- and Micro-F1 scores of F on the clean testing data drop with stronger alignment regularization. This observation is consistent with the discussion in Remark.1.

5.4 Summary

In this chapter, by establishing an information-theoretical adversarial risk bound, we unveil the negative role of LD in multi-label models' adversarial robustness. Our study identifies that the transferability of evasion attack across different labels, which is strongly correlated to the LD, determines the adversarial vulnerability of the classifier. Though capturing the label correlation improves the accuracy of adversary-free multi-label classification, our work unveils that it can also encourage transferable attack, which increases the adversarial risk. We show that the tradeoff between the utility of the classifier and its adversarial robustness can be achieved by explicitly regularizing the transferability level of evasion attack in the learning process of multi-label classification models. Our empirical study demonstrates the applicability of the proposed transferability-regularized robust multi-label learning paradigm for both linear and nonlinear classifies.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

A high performance multi-label classifier should not only generalize well on clean data, but also hold robustness, like the robustness under adversarial evasion attack. The key to build a high performance multi-label classifier is to understand and exploit LD comprehensively. In this dissertation, we unveil the two sides of LD in multi-label classifiers' generalization on clean data and adversarial robustness. To support our claim, on one hand, we propose the approach PNML to further prove the positive role of LD in multi-label classifiers' generalization on clean data. Specially, PNML addresses multi-label classification from a novel view that estimates the positive and negative class distribution of each label in a shared nonlinear embedding space. Effectively, PNML achieves the SOTA classification performance on clean data. On the other hand, we firstly define and empirically evaluate the attackability (adversarial robustness) of multi-label classifiers. After that, we analyze the role of LD in the attackability measurement of multi-label classifiers. Concretely, we derive an information-theoretic upper bound of the adversarial risk faced by multi-label classifiers. The bound demonstrates that LD can help the transfer of attack, which then makes multi-label classifiers more attackable. Inspired by the derived bound, we also propose ARM-SAE to reduce multi-label classifiers' attackability by suppressing the transfer of attack across labels. Our work unveils the tradeoff between multi-label classifiers' generalization on clean data and adversarial robustness from the view of

LD, which suggests that we should seek a balance between them in practice.

6.2 Future Work

Our future works stay with the field of adversarial machine leaning. In this dissertation, we have studied the negative role of LD in multi-label classifiers' robustness under evasion attack. Specially, we consider the untargeted threat model of evasion attack, i.e. the threat model aims to flip as many as possible labels, while it doesn't care about which the attacked labels are. We can extend the threat model to targeted setting, in which the threat model is required to flip labels in label set A, while keep the remained labels unchanged. The future study under targeted threat model may further show the influence of LD on multi-label classifiers.

Another interesting future work is considering the threat model in training stage, i.e. analyzing the role of LD in multi-label classifiers' robustness under poisoning attack. It is possible that poisoned training data will have more influence on highly depended labels.

Lastly, understanding the reason of adversarial examples is also my interest. I will spend more effort on this topic in the future.

REFERENCES

[1] H. Yang, J. T. Zhou, Y. Zhang, B.-B. Gao, J. Wu, and J. Cai, "Exploit bounding box annotations for multi-label object recognition," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, 2016, pp. 280–288.

[2] Z. Chen, X. Wei, P. Wang, and Y. Guo, "Multi-label image recognition with graph convolutional networks," in *Proceedings of 2019 IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, 2019, pp. 5177–5186.

[3] X. Li, J. Ouyang, and X. Zhou, "Supervised topic models for multi-label classification," *Neurocomputing*, vol. 149, pp. 811–819, 2015.

[4] T. N. Rubin, A. Chambers, P. Smyth, and M. Steyvers, "Statistical topic models for multi-label document classification," *Machine Learning*, vol. 88, pp. 157–208, 2012.

[5] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 16, pp. 467–476, 2008.

[6] G. Yu, C. Domeniconi, H. Rangwala, G. Zhang, and Z. Yu, "Transductive multi-label ensemble classification for protein function prediction," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, Beijing, China, 2012, pp. 1077–10 856.

[7] J. Huang, G. Li, Q. Huang, and X. Wu, "Joint feature selection and classification for multilabel learning," *IEEE TRANSACTIONS ON CYBERNETICS*, vol. 48, pp. 876–889, 2018.

[8] X. Cai, F. Nie, W. Cai, and H. Huang, "New graph structured sparsity model for multi-label image annotations," in *Proceedings of the 2013 IEEE International Conference on Computer Vision*, Sydney, NSW, Australia, 2013, pp. 801–808.

[9] J. Read, B. Pfahringe, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Proceedings of the 2009th European Conference on Ma-*

chine Learning and Knowledge Discovery in Databases, Bled, Slovenia, 2009, pp. 254–269.

[10] L. Feng, B. An, and S. He, "Collaboration based multi-label learning," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*. Honolulu, Hawaii: AAAI Press, 2019, pp. 3550–3557.

[11] F. Tai and H.-T. Lin, "Multilabel classification with principal label space transformation," *Neural Computation*, vol. 24, pp. 2508–2542, 2012.

[12] Y. Zhang and J. Schneider, "Multi-label output codes using canonical correlation analysis," *Journal of Machine Learning Research*, vol. 15, pp. 873–882, 2011.

[13] Y. Zhang and J. G. Schneider, "Maximum margin output coding," in *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.

[14] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *ICLR*, 2013.

[15] H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang, and A. K. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," 2019.

[16] D. Cullina, A. Bhagoji, Ramchandran, and P. Mittal, "Pac-learning in the presence of adversaries," in *NeurIPS*, 2019.

[17] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[18] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *ICML*, vol. 80, 2018, pp. 274–283.

[19] A. Fawzi, S. Moosavi-Dezfooli, and P. Frossard, "Robustness of classifiers: From adversarial to random noise," in *NIPS*, 2016, p. 1632–1640.

[20] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," in *NDSS*, 2018.

[21] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[22] A. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *AAAI*, 2018.

[23] D. Jakubovitz and R. Giryes, "Improving dnn robustness to adversarial attacks using jacobian regularization," in *ECCV*. Springer International Publishing, 2018, pp. 525–541.

[24] M. Hein and M. Andriushchenko, "Formal guarantees on the robustness of a classifier against adversarial manipulation," in *NIPS*, 2017, pp. 2266–2276.

[25] D. Zugner and S. Gunnemann, "Certifiable robustness and robust training for graph convolutional networks," in *KDD*, 2019, p. 246–256.

[26] A. Bojchevski and S. Günnemann, "Certifiable robustness to graph perturbations," in *NeurIPS*, 2019, pp. 8319–8330.

[27] A. Raghunathan, J. Steinhardt, and P. Liang, "Semidefinite relaxations for certifying robustness to adversarial examples," in *NeurIPS*, 2018, p. 10900–10910.

[28] A. Gupta, H. Lamba, P. Kumaraguru, and A. Joshi, "Faking sandy: Characterizing and identifying fake images on twitter during hurricane sandy," in *WWW*, 2013, p. 729–736.

[29] K. A. Roundy, P. Mendelberg, N. Dell, D. McCoy, D. Nissani, T. Ristenpart, and A. Tamersoy, "The many kinds of creepware used for interpersonal attacks," in *IEEE S&P)*, may 2020, pp. 626–643.

[30] D. Freed, J. Palmer, D. Minchala, K. Levy, T. Ristenpart, and N. Dell, ""a stalker's paradise": How intimate partner abusers exploit technology," in *CHI*, 2018, p. 1–13.

[31] T. Steinke and L. Zakynthinou, "Reasoning about generalization via conditional mutual information," in *COLT*, 2020.

[32] M. R.Boutell, J. Luo, X. Shen, and C. M.Browna, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, pp. 1757–1771, 2004.

[33] M. Zhang, "Lift: Multi-label learning with label-specific features," in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Barcelona, Catalonia, Spain, 2011, pp. 1609–1614.

[34] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," *Data Mining and Knowledge Discovery Handbook*, p. 667–685, 2010.

[35] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in *Proceedings of the 18th European conference on Machine Learning*, Warsaw, Poland, 2007, pp. 406–417.

[36] J. Read, B. Pfahringer, and G. Holmes, "Multi-label classification using ensembles of pruned sets," in *Proceedings of 2008 IEEE International Conference on Data Mining*, Pisa, Italy, 2008, pp. 995–1000.

[37] Y. Guo, "Convex co-embedding for matrix completion with predictive side information," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, California, USA, 2017, pp. 1955–1961.

[38] H.-F. Yu, P. Jain, P. Kar, and I. S. Dhillon, "Large-scale multi-label learning with missing labels," in *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China, 2014, pp. 593–691.

[39] Z. Lin, G. Ding, M. Hu, and J. Wang, "Multi-label classification via feature-aware implicit label space encoding," in *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China, 2014.

[40] L. Xu, Z. Wang, Z. Shen, Y. Wang, and E. Chen, "Learning low-rank label correlations for multi-label classification with missing labels," in *Proceedings of 2014 IEEE International Conference on Data Mining*, Shenzhen, China, 2014, pp. 1067–1072.

[41] K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain, "Sparse local embeddings for extreme multi-label classification," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Montreal, Canada, 2015, pp. 730–738.

[42] S. Parameswaran and K. Q. Weinberger, "Large margin multi-task metric learning," in *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, Vancouver, British Columbia, Canada, 2010, pp. 1867–1875.

[43] W. Liu and I. W. Tsang, "Large margin metric learning for multi-label prediction," in *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, Austin Texas, USA, 2015, pp. 2800–2806.

[44] L. Cheng, W. Bingyu, P. Virgil, and A. Javed, "Conditional bernoulli mixtures for multi-label classification," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, p. 2482–2491.

[45] C. Hong, I. Batal, and M. Hauskrecht, "A generalized mixture framework for multi-label classification," in *SIAM International Conference on Data Mining 2015*, 2015.

[46] Z. He, J. Wu, and P. Lv, "Label correlation mixture model for multi-label text categorization," in *2014 IEEE Spoken Language Technology Workshop (SLT)*, 2014.

[47] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *ECML PKDD*, 2013.

[48] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE S&P*, 2017.

[49] T. Florian, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *ICLR*, 2018.

[50] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *IEEE S&P*, 2016, pp. 582–597.

[51] D. Zugner and S. Günnemann, "Certifiable robustness of graph convolutional networks under structure perturbations," in *KDD*, 2020, p. 1656–1665.

[52] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *ICML*, 2019, pp. 1310–1320.

[53] G. Lee, Y. Yuan, S. Chang, and T. Jaakkola, "Tight certificates of adversarial robustness for randomly smoothed classifiers," in *NeurIPS*, 2019, pp. 4910–4921.

[54] Y. Wang, X. Ma, J. Bailey, J. Yi, B. Zhou, and Q. Gu, "On the convergence and robustness of adversarial training," in *ICML*, 2019, pp. 6586–6595.

[55] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!" in *NeurIPS*, 2019, pp. 3358–3369.

[56] R. Gao, T. Cai, H. Li, C. J. Hsieh, L. Wang, and J. D. Lee, "Convergence of adversarial training in overparametrized neural networks," in *NeurIPS*, 2019, pp. 13 029–13 040.

[57] Y. Wang, S. Jha, and K. Chaudhuri, "Analyzing the robustness of nearest neighbors to adversarial examples," in *ICML*, 2018.

[58] J. Gilmer, L. Metz, F. Faghri, S. Schoenholz, M. Raghu, M. Wattenberg, and I. Goodfellow, "Adversarial spheres," *CoRR*, 2018. [Online]. Available: http://arxiv.org/abs/1801.02774

[59] D. Yin, K. Ramchandran, and P. Bartlett, "Rademacher complexity for adversarially robust generalization," in *ICML*, 2019.

[60] J. Khim and P. Loh, "Adversarial risk bounds for binary classification via function transformation," *arXiv*, 2018.

[61] Z. Tu, J. Zhang, and D. Tao, "Theoretical analysis of adversarial learning: A minimax approach," in *NeurIPS*, 2019, pp. 12 259–12 269.

[62] L. Qi, L. Wu, P. Chen, A. Dimakis, I. Dhillon, and M. Witbrock, "Discrete attacks and submodular optimization with applications to text classification," in *SysML*, 2019.

[63] Y. Wang, Y. Han, H. Bao, Y. Shen, F. Ma, J. Li, and X. Zhang, "Attackability characterization of adversarial evasion attack on discrete data," in *KDD*, 2020.

[64] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou, "Multi-label learning with weak label," in *AAAI*, 2010, pp. 593–598.

[65] G. Zhu, S. Yan, and Y. Ma, "Image tag refinement towards low-rank, content-tag prior and error sparsity," in *ACM MultiMedia*, 2010, pp. 461–470.

[66] D. Liu, X.-S. Hua, M. Wang, and H.-J. Zhang, "Image retagging," in *ACM MultiMedia*, 2010, pp. 491–500.

[67] Z. Lin, G. Ding, M. Hu, J. Wang, and X. Ye, "Image tag completion via image-specific and tag-specific linear sparse reconstructions," in *CVPR*, 2013, pp. 1618–1625.

[68] L. Wu, R. Jin, and A. K. Jain, "Tag completion for image retrieval," *TPAMI*, vol. 35, no. 3, pp. 716–727, 2013.

[69] F. Zhao and Y. Guo, "Semi-supervised multi-label learning with incomplete labels," in *IJCAI*, 2015, pp. 4062–4068.

[70] H.-F. Yu, P. Jain, P. Kar, and I. S. Dhillon, "Large-scale multi-label learning with missing labels," in *ICML*, 2014.

[71] W. Bi and J. T. Kwok, "Multilabel classification with label correlations and missing labels," in *AAAI*, 2014, pp. 1680–1686.

[72] A. B. Goldberg, X. Zhu, B. Recht, J.-M. Xu, and R. Nowak, "Transduction with matrix completion: Three birds with one stone," in *NIPS*, 2010, pp. 757–765.

[73] R. Cabral, F. D. la Torre, J. P. Costeira, and A. Bernardino, "Matrix completion for weakly-supervised multi-label image classification," *TPAMI*, vol. 37, no. 1, pp. 121–135, 2015.

[74] M. Xu, R. Jin, and Z.-H. Zhou, "Speedup matrix completion with side information: Application to multi-label learning," in *NIPS*, 2013, pp. 2301–2309.

[75] K.-Y. Chiang, C.-J. Hsieh, and I. S. Dhillon, "Matrix completion with noisy side information," in *NIPS*, 2015, pp. 3447–3455.

[76] Y. Guo, "Convex co-embedding for matrix completion with predictive side information." in *AAAI*, 2017, pp. 1955–1961.

[77] Y. Zhu, J. T.Kwok, and Z.-H. Zhou, "Multi-label learning with global and local label correlation," *IEEE Transaction on Knowledge and Data Engineering*, 2018.

[78] C.-J. Hsieh, N. Natarajan, and I. S. Dhillon, "PU learning for matrix completion," in *ICML*, 2015, pp. 663–672.

[79] C. Xu, T. Liu, D. Tao, and C. Xu, "Local rademacher complexity for multi-label learning," *IEEE Transactions on Image Processing*, vol. 25, no. 3, pp. 1495–1507, 2016.

[80] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with bregman divergences," *Journal of machine learning research*, vol. 6, p. 1705–1749, 2005.

[81] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Long Beach, CA, 2017.

[82] K. Allen, E. Shelhamer, H. Shin, and J. Tenenbaum, "Infinite mixture prototypes for few-shot learning," in *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[83] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013.

[84] B. Kulis and M. I. Jordan, "Revisiting k-means: New algorithms via bayesian nonparametrics," in *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[85] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[86] D. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, 2015.

[87] M. S. Sorower, "A literature survey on algorithms for multi-label learning," 2010.

[88] M. Zhang and Z. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, pp. 2038–2048, 2007.

[89] W. Chen, S. Yuanhai, L. Chunna, and D. Naiyang, "Mltsvm: a novel twin support vector machine to multi-label learning," *Pattern Recognition*, 2016.

[90] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma, "Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising," in *Proceedings of the 27th International Conference on World Wide Web*, 2018.

[91] P. Szymański and T. Kajdanowicz, "A scikit-based python environment for performing multi-label classification," *ArXiv*, 2017.

[92] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, 2006.

[93] N. Buchbinder, M. Feldman, J. Naor, and R. Schwartz, "Submodular maximization with cardinality constraints," in *SODA*, 2014.

[94] E. R. Elenberg, R. Khanna, A. G. Dimakis, and S. Negahban, "Restricted strong convexity implies weak submodularity," *Annuals of Statistics*, 2016.

[95] G. Tsoumakas, I. Katakis, and I. Vlahavas, *Mining Multi-label Data*. Springer US, 2010, pp. 667–685.

[96] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge 2012 (voc2012) results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html", 2012.

[97] Kaggle, "Planet: Understanding the amazon from space," https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/overview, 2017.

[98] M. Nicolae, M. Sinn, T. N. Minh, A. Rawat, M. Wistuba, V. Zantedeschi, I. M. Molloy, and B. Edwards, "Adversarial robustness toolbox v0.2.2," *CoRR*, 2018. [Online]. Available: http://arxiv.org/abs/1807.01069

[99] Chollet and Francois. (2015) Keras. [Online]. Available: https://github.com/fchollet/keras

[100] Y. Yang, R. Khanna, Y. Yu, A. Gholami, K. Keutzer, J. E. Gonzalez, K. Ramchandran, and M. W. Mahoney, "Boundary thickness and robustness in learning models," in *NeuIPS*, 2020.

[101] G. F. Elsayed, D. Krishnan, H. Mobahi, and K. Regan, "Large margin deep networks for classification," in *NeuIPS*, 2018.

[102] C. Zhao, P. Fletcher, M. Yu, Y. Peng, G. Zhang, and C. Shen, "The adversarial attack and detection under the fisher information metric," in *AAAI*, 2019, pp. 5869–5876.

[103] Z. Yang, Y. Han, and X. Zhang, "Characterizing the evasion attackability of multi-label classifiers," in *AAAI*, 2021.

[104] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, 1963.

APPENDICES

A Proofs

Proof of Lemma 1

We first verify the supermodularity of $g(S)$ in Eq.4.2. $g(S)$ is a non-decreasing set function with increasingly larger S. We first derive an analytical solution to $g(S)$ with lagrangian multipliers $\{\lambda_i\}$:

$$\begin{aligned} & J(\lambda_i, \mathbf{r}) = \|\mathbf{r}\|^2 \\ & + \sum_{i=1}^{K} \lambda_i (2 b_i y^i f_i(\mathbf{x}+\mathbf{r}) - y^i f_i(\mathbf{x}+\mathbf{r}) + t_i), \\ s.t.\ \ & \lambda_i \geq 0, \\ & b_i = 1 \ \ for \ \ i \in T, \\ & b_i = 0 \ \ for \ \ i \notin T, \end{aligned} \tag{A.1}$$

where f_i denotes the output of the classifier F corresponding to the i-th label. Since the adversarial noise $\mathbf{r}$ is usually of small magnitude, we further approximate $f_i(\mathbf{x}+\mathbf{r})$ with its Taylor expansion: $f_i(\mathbf{x}+\mathbf{r}) \approx f_i(\mathbf{x}) + \mathbf{r}^T f_i'(\mathbf{x})$. Eq.A.1 can be further simplified as a quadratic programming problem with affine constraints:

$$
\begin{aligned}
& J(\lambda_i, \mathbf{r}) = \|\mathbf{r}\|^2 \\
& + \sum_{i=1}^{K} \lambda_i (2 b_i y^i (f_i + \mathbf{r}^T \phi_i) + t_i - y^i f_i - y^i \mathbf{r}^T \phi_i), \\
s.t.\ & \lambda_i \geq 0, \\
& b_i = 1 \ \ for \ \ i \in T, \\
& b_i = 0 \ \ for \ \ i \notin T,
\end{aligned}
\tag{A.2}
$$

where $\phi_i(\mathbf{x}) = f_i'(\mathbf{x})$ denotes the gradient of $f_i(\mathbf{x})$ with respect to the input feature vector $\mathbf{r}$.

By taking the first-order condition $\frac{\partial J}{\partial \mathbf{r}} = 0$, we can derive the optimal $\|\mathbf{r}\|^2$ as :

$$
\|\mathbf{r}\|^2 = \frac{1}{4} \| \sum_{i=1}^{K} (\lambda_i y^i \phi_i - 2 \lambda_i \phi_i b_i y^i) \|_2^2, \tag{A.3}
$$

and according to KKT conditions, we can get for non-zero λ_i:

$$
\begin{aligned}
& f_i + \frac{1}{2} \sum_{k \in \{k | \lambda_k > 0\}} (\lambda_k y^k \phi_k^T \phi_i - 2 \lambda_k \phi_k^T \phi_i \hat{y}^k) = 0 \\
& (if \ \ \lambda_i > 0).
\end{aligned}
\tag{A.4}
$$

Observation 1.1. *To obtain the values of* $\lambda_k > 0$ *(*$k = 0, 1, 2, ..., K$*), we turn to solve*

the equation system such as:

$$
\begin{aligned}
&-\frac{1}{2}\Pi[\Phi_0^T, \Phi_1^T, ..., \Phi_K^T] = H, \\
&\Pi = [\lambda_0\phi_0^T - 2b_0y^0\phi_0^T, \lambda_1\phi_1^T - 2b_1y^1\phi_1^T, \\
&..., \lambda_K\phi_K^T - 2b_Ky^K\phi_K^T], \\
&\Phi_k = [\phi_k, \phi_k, ..., \phi_k], \\
&H = [h_0, h_1, h_2, ..., h_K].
\end{aligned} \tag{A.5}
$$

Given a set of $\{b_0, b_1, ..., b_K\}$*, the value of* λ_k *(*$k = 0, 1, 2, ..., K$*) is determined uniquely by* f_i *and* ϕ_i*.*

Observation 1.2. $\|\mathbf{r}\|^2$ *is a set function defined over the set* T*, as* $\hat{y}^i$ *is determined by the binary variable* b_i*.*

Observation 1.3. $\|\mathbf{r}\|^2$ *is a convex quadratic function with respect to the variable* $\{\hat{y}^i\}, (i = 1, 2, 3, ..., m)$*, given a set of* $\{b_0, b_1, b_2, ...b_K\}$ *fixed in Eq.A.3.*

Furthermore, by simply flipping the sign of $g(S)$, we can find that $-g(S) = \min_S - \|\mathbf{r}\|^2$ is non-increasing and submodular function, since $-\|\mathbf{r}\|^2$ is concave, according to Theorem.1 in [94]. Correspondingly, $g(S)$ is a non-decreasing supermodular set function. In Eq.4.2, $|S|$ is a monotonically increasing modular function. As a result, the objective $\psi(S) = |S| - g(S)$ of the maximization problem defined in Eq.4.2 is a non-monotone submodular function. According to Theorem 1.5 in [93], randomized greedy forward expansion of the set S can provide a guarantee to the approximation accuracy:

$$\psi(\hat{S}) \geq \frac{1}{4}\psi(S^*). \tag{A.6}$$

where $\psi(\hat{S})$ and $\psi(S^*)$ denote respectively the objective function value obtained by randomized greedy forward search proposed in [93] and the underlying global optimum following the cardinality lower bound constraint.

Proof of Lemma 2

In each iteration of the greedy forward search, the current set of flipped labels is noted as T and the current perturbed input is noted as $\tilde{\mathbf{x}}$. $F(\tilde{\mathbf{x}})$ and $\phi(\tilde{\mathbf{x}})$ denotes the current classifier output and gradient vector with respect to $\tilde{\mathbf{x}}$. We further assume that the i^*-th label is selected to be flipped and added to the set T in the current iteration of the greedy search. Given the F, we inject a small adversarial perturbation $\mathbf{r}$ to form the perturbed input $\tilde{\mathbf{x}} + \mathbf{r}$ centering at $\tilde{\mathbf{x}}$, in order to flip the label i^*. By taking $\frac{\partial J}{\partial r} = 0$ and $\frac{\partial J}{\partial \lambda_i} = 0$ and the complementary slackness conditions, we have:

$$
\begin{aligned}
&\|\mathbf{r}_{-i^*}\|_2 \leq \frac{1}{2}\|\sum_{j\neq i}^{K}(\lambda_j y^j \phi_j - 2\lambda_j \phi_j \hat{y}^j)\|_2 \\
&+ \frac{1}{2}\|\lambda_{i^*} y^{i^*} \phi_{i^*}\|_2, \\
&y^i f_i + t_i \\
&= -\frac{y^i}{2}(\sum_{j\neq i}(\lambda_j y^j \phi_j^T \phi_i - 2\hat{y}^j \lambda_j \phi_j^T \phi_i)) + \frac{\lambda_i \|\phi_i\|^2}{2}, \quad i \in T, \\
&- y^i h_i + t_i \\
&= \frac{y^i}{2}(\sum_{j\neq i}(\lambda_j y^j \phi_j^T \phi_i - 2\hat{y}^j \lambda_j \phi_j^T \phi_i)) - \frac{\lambda_i \|\phi_i\|^2}{2}, \quad i \notin T, \\
&\lambda_i(2b_i y^i f_i - y^i f_i + t_i + (2b_i y^i - y^i)\mathbf{r}^T \phi_i) = 0,
\end{aligned}
\tag{A.7}
$$

where $\mathbf{r}_{-i^*}$ denotes the adversarial perturbation that can flip both the labels in the current attacked label set T and the latest added label i^*.

Observation 1.4. *To minimize $\|\mathbf{r}_{-i^*}\|_2$, we can set $\lambda_j = 0$, $(j \neq i^*)$ and $\lambda_{i^*} > 0$. In this case, we can derive a feasible solution:*

$$
\mathbf{r}_{-i^*} = -\frac{1}{2}\lambda_{i^*} \phi_{i^*} y^{i^*}. \tag{A.8}
$$

Furthermore, we can observe that the marginal gain of the greedy search is then pro-

portional to $\lambda_{i^*}\|\phi_{i^*}\|$ *of the candidate label* i^**. To minimize the marginal gain, we choose the candidate label* i^* *producing the minimal* $\lambda_{i^*}\|\phi_{i^*}\|$*.*

By taking $\frac{\partial J}{\partial \lambda_i} = 0$ and substituting the above expression of $\mathbf{r}$, we can obtain:

$$\lambda_{i^*} = \frac{2(t_{i^*} + y^{i^*} f_{i^*}(\tilde{\mathbf{x}}))}{\|\phi_{i^*}\|_2^2}. \tag{A.9}$$

Consequently, we can derive that the upper bound of the required adversarial perturbation norm $\mathbf{r}_{-i^*}$ as follows:

$$\frac{|y^{i^*} f_{i^*}(\tilde{\mathbf{x}}) + t_{i^*}|}{\|\phi_{i^*}\|_2} \geq \|\mathbf{r}_{-i^*}\|_2. \tag{A.10}$$

As indicated by Eq.A.10 and $t_i > 0$, $\|\mathbf{r}_{-i^*}\|_2$ is proportional to the ratio $\frac{|y^{i^*} f_{i^*}(\tilde{\mathbf{x}})|}{\|\phi_{i^*}\|_2}$. It gives the conclusion of Lemma.2 in Section.4. Based on Eq.A.8 and Eq.A.9, we can find that $\|\frac{\partial \|\mathbf{r}_{-i^*}\|_2}{y^i}\|_2 = \frac{|y^i f_i(\tilde{\mathbf{x}}) + t_i|}{\|\phi_i\|_2}$. Therefore, the greedy feedforward expansion can be considered as conducting orthogonal matching pursuit based greedy search for the submodular function maximization problem [94].

We supple the proof of the theorem in our paper, especially the Eq.(5.3) and Eq.(5.5), and the proof from Eq.(5.11) to Eq.(5.12)

Lemma 4. *(Thomas 2020 [31], Corollary 5) Let* E*,* E' *and* Z *be independent random variables where* E *and* E' *have identical distributions. Let* A *be a random function whose randomness is independent from* E*,* E' *and* Z*. Let* g *be a fixed function. Then*

$$\begin{aligned} &\mathop{\mathbb{E}}_{A,E,Z}[g(A(E,Z),E,Z)] \\ &\leq \inf_{t>0} \frac{I(A(E,Z);E|Z) + \mathop{\mathbb{E}}_{Z}\left[\log \mathop{\mathbb{E}}_{A,E,E'Z}\left[e^{t\cdot g(A(E,Z),E',Z)}\right]\right]}{t} \end{aligned} \tag{A.11}$$

Lemma 5. *(Hoeffding 1963 [104].) Let* $\mathbf{x} \in [a,b]$ *be a random variable with mean* μ*.*

Then for all $t \in \mathbb{R}$,

$$\mathbb{E}(e^{t\mathbf{x}}) \leq e^{t\mu + t^2(b-a)^2/8} \tag{A.12}$$

Proof from Eq.5.11 to Eq.5.12

We can rewrite Eq.5.9 as

$$A_{F(\mathbf{x}),\tilde{\mathbf{r}}} = \tilde{\mathbf{r}'} \sum_{k=1}^{K} \frac{-\nabla f_k(\mathbf{x}) * \max\{\text{sgn}(-\tilde{\mathbf{r}'} y^k \nabla f_k(\mathbf{x})), 0\}}{f_k(\mathbf{x})}. \tag{A.13}$$

If there is no sgn function and max function in Eq.A.13, Eq.5.11 is actually the definition of dual norm. To eliminate the sgn and max function, we can break the domain of $\tilde{\mathbf{r}}$ into a group of subsets according the output of those sgn functions. Denote the domain of $\tilde{\mathbf{r}}$ as I and I_S is a subset of I which is defined by Eq.(A.14). S is an element from the power set of $\{1, \cdots, K\}$.

$$I_S = \left\{ \tilde{\mathbf{r}} \;\middle|\; \begin{array}{l} \tilde{\mathbf{r}} y^k \nabla f_k < 0, k \in S \\ \tilde{\mathbf{r}} y^k \nabla f_k \geq 0, k \notin T \end{array}, \tilde{\mathbf{r}} \in \mathbb{R}^n \right\} \tag{A.14}$$

Based on Eq.(A.14), we redefine Eq.(5.9) and Eq.(5.11) over the subdomain I_S of $\mathbf{r}$ as:

$$A_{F(\mathbf{x}),\tilde{\mathbf{r}}s} = \sum_{k \in S} \frac{-\tilde{\mathbf{r}}' \nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \tag{A.15}$$

$$\begin{aligned} &\phi_s = \max_{\tilde{\mathbf{r}} \in I_S} A_{F(\mathbf{x}),\tilde{\mathbf{r}}s}, \\ &s.t. \;\; \left\|\tilde{\mathbf{r}}\right\|_p = 1 \end{aligned} \tag{A.16}$$

Now, we get $\phi_{F,\mathbf{x}} = \max\limits_{S \in P(S)} \phi_s$. It's easy to know that:

$$\begin{array}{l} \phi_s = \max\limits_{\tilde{\mathbf{r}} \in I_S} A_{F(\mathbf{x}),\tilde{\mathbf{r}}s}, \\ s.t. \;\; \left\|\tilde{r}\right\|_p = 1 \end{array} \leq \begin{array}{l} \phi_s = \max\limits_{\mathbf{c} \in \mathbb{R}^n} A_{F(\mathbf{x}),\mathbf{c}s}, \\ s.t. \;\; \left\|\mathbf{c}\right\|_p = 1 \end{array} = \left\| \sum_{k \in S} \frac{-\nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \right\|_q \tag{A.17}$$

The equality holds when the optimal $\mathbf{c}^*$ exactly locates in I_S. Now, if we want to prove that Eq.(5.11) = Eq.(5.12), we just need to prove that $\phi_{S^*} = \left\| \sum_{k \in S^*} \frac{-\nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \right\|_q$, that is we need to prove that the optimal $\mathbf{c}^*$ for S^* locates in I_{S^*}. We can prove that by contradiction. That is we assume $\mathbf{c}^*_{S^*} \in I_{S'}(S' \neq S^*)$, then it is proved by Eq.(A.18).

$$\begin{aligned} \left\| \sum_{k \in S^*} \frac{-\nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \right\|_q &= \sum_{k \in S^*} \frac{-\mathbf{c}^*_{S^*} \nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \\ &< \sum_{k \in S^* \cap S'} \frac{-\mathbf{c}^*_{S^*} \nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \\ &\leq \left\| \sum_{k \in S^* \cap S'} \frac{-\nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \right\|_q \\ &< \left\| \sum_{k \in S^*} \frac{-\nabla f_k(\mathbf{x})}{f_k(\mathbf{x})} \right\|_q \end{aligned} \tag{A.18}$$

Proof of Eq.5.3

We define the worst-case loss $l(F, \mathbf{z}, \varepsilon)$ as:

$$\begin{aligned} &l(F, \mathbf{z}, \varepsilon) = \max_{\mathbf{z}' \in N(\mathbf{z})} l(F, \mathbf{z}'), \\ &\text{where } N(\mathbf{z}) = \left\{ (\mathbf{x}', \mathbf{y}') \,\middle|\, \|\mathbf{x}' - \mathbf{x}\|_p \leq \varepsilon,\, \mathbf{y}' = \mathbf{y} \right\}. \end{aligned} \tag{A.19}$$

We first upperly bound $l(F, \mathbf{z}, \varepsilon)$ defined in Eq.(A.19) with the setting of linear classifier and hinge loss:

$$\begin{aligned} &l(F, \mathbf{z}, \varepsilon) \leq l(F, \mathbf{z}) + \\ &\max_{\|\mathbf{r}\|_2 \leq \varepsilon} \left\| \sum_{k=1}^{K} y^k \mathbf{r}' \cdot \mathbf{w}_k * \max\{\text{sgn}(y^k \mathbf{r}' \cdot \mathbf{w}_k), 0\} \right\|_2 \\ &\leq l(F, \mathbf{z}) + C_{\mathbf{W}, \mathbf{z}} \varepsilon. \end{aligned} \tag{A.20}$$

The last step borrows the proof from Eq.5.11 to Eq.5.12. Then we have

$$
\begin{aligned}
&R_{\mathcal{D}}(A,\varepsilon) - R_{Z^n}(A,\varepsilon) \\
&= \mathop{\mathbb{E}}_{Z^n\leftarrow\mathcal{D}^n,A} l(A(Z^n),\mathcal{D},\varepsilon) - \mathop{\mathbb{E}}_{Z^n\leftarrow\mathcal{D}^n,A} l(A(Z^n),Z^n,\varepsilon) \\
&= \mathop{\mathbb{E}}_{\bar{Z},E,A}\left[l(A(\bar{Z}_E),\bar{Z}_{\bar{E}},\varepsilon) - l(A(\bar{Z}_E),\bar{Z}_E,\varepsilon)\right], \quad (\bar{Z}\leftarrow\mathcal{D}^{n\times 2}) \\
&= \mathop{\mathbb{E}}_{\bar{Z},E,A}\left[f_{\bar{Z}}(A(\bar{Z}_E),E,\varepsilon)\right] \\
&\quad \textit{by } LEMMA\,2 \\
&\le \inf_{t>0} \frac{I(A(\bar{Z}_E);E|\bar{Z}) + \mathop{\mathbb{E}}_{\bar{Z}}\left[\log \mathop{\mathbb{E}}_{\mathbf{W},E'}\left[e^{t f_{\bar{Z}}(\mathbf{W},E',\varepsilon)}\right]\right]}{t}, \\
&\quad \textit{by independence} \\
&= \inf_{t>0} \frac{CMI_{\mathcal{D},A} + \mathop{\mathbb{E}}_{\bar{Z}}\left[\log \mathop{\mathbb{E}}_{\mathbf{W}}\left[\prod_{i=1}^{n} \mathop{\mathbb{E}}_{E'_i}\left[e^{\frac{t}{n}(l(W,(\bar{Z}_{E'})_i,\varepsilon) - l(\mathbf{W},(\bar{Z}_{\bar{E}'})_i,\varepsilon))}\right]\right]\right]}{t}, \\
&= \inf_{t>0} \frac{CMI_{\mathcal{D},A} + \mathop{\mathbb{E}}_{\bar{Z}}\left[\log \mathop{\mathbb{E}}_{\mathbf{W}}\left[\prod_{i=1}^{n} \mathop{E}_{E'_i}\left[e^{\frac{t}{n}(1-2E'_i)(l(\mathbf{W},\bar{Z}_{i,1},\varepsilon) - l(\mathbf{W},\bar{Z}_{i,2},\varepsilon))}\right]\right]\right]}{t} \\
&\quad \textit{by } LEMMA\ 3 \\
&\le \inf_{t>0} \frac{CMI_{\mathcal{D},A} + \mathop{\mathbb{E}}_{\bar{Z}}\left[\log \mathop{\mathbb{E}}_{\mathbf{W}}\left[\prod_{i=1}^{n} e^{\frac{t^2}{2n^2}(l(\mathbf{W},\bar{Z}_{i,1},\varepsilon) - l(\mathbf{W},\bar{Z}_{i,2},\varepsilon))^2}\right]\right]}{t}, \\
&\le \inf_{t>0} \frac{CMI_{\mathcal{D},A}}{t} + \frac{t}{2n}\mathop{\mathbb{E}}_{\bar{Z}}\left[\sup_{\mathbf{W}\in\mathcal{W}_A} \frac{1}{n}\sum_{i=1}^{n}\left(l(\mathbf{W},\bar{Z}_{i,1},\varepsilon) - l(\mathbf{W},\bar{Z}_{i,2},\varepsilon)\right)^2\right] \\
&\le \inf_{t>0} \frac{CMI_{\mathcal{D},A}}{t} + \frac{t}{2n}\mathop{\mathbb{E}}_{Z\leftarrow\mathcal{D}}\left[\sup_{\mathbf{W}\in\mathcal{W}_A} l(\mathbf{W},Z,\varepsilon)^2\right] \\
&\le \inf_{t>0} \frac{CMI_{\mathcal{D},A}}{t} + \frac{t}{2n}\mathop{\mathbb{E}}_{Z\leftarrow\mathcal{D}}\left[\sup_{\mathbf{W}\in\mathcal{W}_A} \left(l(\mathbf{W},Z) + C_{\mathbf{W},Z}\cdot\varepsilon\right)^2\right] \\
&= \sqrt{\frac{2}{n} CMI_{\mathcal{D},A} \cdot \mathop{\mathbb{E}}_{Z\leftarrow\mathcal{D}}\left[\sup_{\mathbf{W}\in\mathcal{W}_A} \left(l(\mathbf{W},Z) + C_{\mathbf{W},Z}\cdot\varepsilon\right)^2\right]}
\end{aligned}
\tag{A.21}
$$

Proof of Eq.5.5

Here we use H to denote the entropy.

$$
\begin{aligned}
& CMI_{\mathcal{D},A} \\
&= I(A; S, \bar{Z}) - I(A; \bar{Z}) \\
&= H(A) + H(S, \bar{Z}) - H(A, S, \bar{Z}) - H(A) - H(\bar{Z}) + H(A, \bar{Z}) \\
&= H(A, \bar{Z}) + H(S|\bar{Z}) - H(S) - H(A, \bar{Z}|S) \quad : S \; is independent to \, Z \\
&= H(A, \bar{Z}) - H(A, \bar{Z}|S) \\
&\leq H(A, \bar{Z}) \\
&\leq H(A) + H(\bar{Z}) \\
&= H(\mathbf{W}) + H(\bar{Z}) \\
&= ent(\mathbf{w}_1, \cdots, \mathbf{w}_m) + ent(\mathcal{D}_1, \cdots, \mathcal{D}_m)
\end{aligned}
\tag{A.22}
$$

B Papers Submitted and Under Preparation

• Zhuo Yang, Yufei Han and Xiangliang Zhang. "Characterizing the Evasion Attackability of Multi-label Classifiers", *the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021)*

• Zhuo Yang, Yufei Han and Xiangliang Zhang. "Attack Transferability Characterization for Adversarially Robust Multi-label Classification", *2021 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD 2021)*

• Zhuo Yang, Yufei Han, Guoxian Yu, Qiang Yang and Xiangliang Zhang. "Prototypical Networks for Multi-Label Learning", *submitted to IEEE Transactions on Neural Networks and Learning Systems (TNNLS), under review.*

• Zhuo Yang, Yufei Han, Guang Cheng and Xiangliang Zhang. "Which Frequency Matters? A Fourier Perspective on Adversarial Risk Characterization for Deep Neural Networks", *submitted to IJCAI 2022.*

• Zhuo Yang, Yufei Han and Xiangliang Zhang. "On Attackability Characterization of Targeted Evasion Attack Against Multi-label Classifiers", *under preparation*