Multi-label Classification with Optimal Thresholding for Multi-composition Spectroscopic Analysis

by

Luyun GAN
B.A., Nankai University, 2012

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Multi-label Classification with Optimal Thresholding for Multi-composition Spectroscopic Analysis

by

Luyun GAN
B.A., Nankai University, 2012

Supervisory Committee

Dr. Tao Lu, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Wu-Sheng Lu, Departmental Member
(Department of Electrical and Computer Engineering)

Supervisory Committee

Dr. Tao Lu, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Wu-Sheng Lu, Departmental Member
(Department of Electrical and Computer Engineering)

ABSTRACT

Spectroscopic analysis has several applications in physics, chemistry, bioinformatics, geophysics, astronomy, etc. It has been widely used for detecting mineral samples, gas emission, and food volatiles. Machine learning algorithms for spectroscopic analysis focus on either regression or single-label classification problems. Using multi-label classification to identify multiple chemical components from the spectrum, has not been explored. In this thesis, we implement Feed-forward Neural Network with Optimal Thresholding (FNN-OT) identifying gas species among a multi gas mixture in a cluttered environment. Spectrum signals are initially processed by a Feed-forward Neural Network (FNN) model, which produces individual prediction scores for each gas. These scores will be the input of a following Optimal Thresholding (OT) system. Predictions of each gas component in one testing sample will be made by comparing its output score from FNN against a threshold from the OT system. If its output score is larger than the threshold, the prediction is 1 and 0 otherwise, representing the existence/non-existence of that gas component in the spectrum.

Using infrared absorption spectroscopy and tested on synthesized spectral datasets, our approach outperforms FNN itself and conventional binary relevance - Partial Least Squares - Binary Relevance (PLS-BR). All three models are trained and tested on 18 synthesized datasets with 6 levels of Signal-to-noise Ratio (SNR) and 3 types of gas correlation. They are evaluated and compared with micro, macro and sample averaged precision, recall and F_1 score. For mutually independent and randomly correlated gas data, FNN-OT yields better performance than FNN itself or the conventional PLS-BR, by significantly by increasing recall without sacrificing much precision.

For positively correlated gas data, FNN-OT performs better in capturing information of positive label correlation from noisy datasets than the other two models.

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **v**

List of Tables **viii**

List of Figures **ix**

Acronyms **xii**

Acknowledgements **xiii**

Dedication **xiv**

1 Introduction **1**

1.1 Context 1

1.2 Research Problem 4

1.3 Proposed Approach 7

1.4 Method Illustration 9

1.5 Research Contributions 11

1.6 Thesis Outline 11

2 Background and Related Work **13**

2.1 Feed-forward Neural Network 13

2.1.1 Activation Functions 14

2.1.2 Loss Functions 16

2.1.3 Optimization Methods 18

2.1.4 Number of Neurons and Layers 20

2.1.5 Dropout . . . 21
2.2 Multi-label Classification . . . 21
2.2.1 Formal Definitions of Multi-label Classification . . . 22
2.2.2 Algorithms for Multi-label Learning . . . 23
2.2.3 Evaluation Metrics . . . 25
2.3 Absorption spectroscopy . . . 28

3 Feedforward Neural Network with Optimal Thresholding 32
3.1 Pre-processing Datasets . . . 33
3.2 FNN-1 . . . 37
3.3 Optimal Thresholding . . . 38

4 Evaluation and Comparison 41
4.1 Datasets . . . 41
4.1.1 Datasets with Mutually Independent Gas Labels . . . 42
4.1.2 Datasets with Positively Correlated Gas Labels . . . 44
4.1.3 Datasets with Randomly Correlated Gas Labels . . . 46
4.2 Models for Comparison . . . 48
4.2.1 PLS-BR . . . 48
4.2.2 FNN without OT . . . 49
4.3 Evaluation Metrics . . . 50

5 Results and Discussions 53
5.1 Hyper-parameter Tuning . . . 53
5.1.1 Latent Variables . . . 53
5.1.2 Optimizer . . . 53
5.1.3 Training Sample Size . . . 55
5.1.4 Dropout . . . 56
5.2 Performance Comparison of Mutually Independent Gas Data . . . 58
5.3 Performance Comparison of Positively Correlated Gas Data . . . 61
5.4 Performance Comparison of Randomly Correlated Gas Data . . . 66

6 Conclusion 70
6.1 Summary and contributions . . . 70
6.2 Limitations and Future Work . . . 71

A Additional Information 73

A.1 Software 73

Bibliography **74**

List of Tables

Table 1.1 Categorization of classification tasks 5
Table 1.2 Multi-class classification for sample images in Figure 1.3 7
Table 1.3 Multi-label classification for sample images in Figure 1.3 7

Table 2.1 Confusion Matrix for Binary Classification 25
Table 2.2 Evaluation Metrics for Single-label and Multi-label Classifications 28

Table 4.1 The numbers of positive and negative samples in each label at 30 dB when gases are mutually independent. "P" in the first column represents for number of positives, and "N" represents for number of negatives. 42
Table 4.2 The numbers of positive and negative samples in each label at 30 dB when gases are positively correlated. "P" in the first column represents for number of positives, and "N" represents for number of negatives. 44
Table 4.3 The numbers of positive and negative samples in each label at 30 dB when gases are randomly correlated. "P" in the first column represents for number of positives, and "N" represents for number of negatives. 46

Table 5.1 Number of LVs for Partial Least Squares (PLS) 53
Table 5.2 Retention probabilities of input layer (p_1) and hidden layer (p_2). Data 1 represents mutually independent gas data. Data 2 represents positively correlated gas data. Data 3 represents randomly correlated gas data. 58

List of Figures

Figure 1.1 Categorizations in machine learning 2
Figure 1.2 Information flow in FNN (left) and Recurrent Neural Network (RNN) (right) 3
Figure 1.3 The LOF caption 6
Figure 1.4 Illustration of FNN-OT model for multi-composition spectroscopic analysis 8
Figure 1.5 Caption for LOF 9

Figure 2.1 A neuron in Artificial Neural Network (ANN) model 15
Figure 2.2 Fraunhofer's experiment for detecting the dark lines of the solar spectrum. [79, 38] 29

Figure 3.1 FNN-OT training and testing procedure. 33
Figure 3.2 Percentage of cumulative explained variance vs. number of principal components adopted. 34
Figure 3.3 Comparison of Hamming loss with and without Principal Component Analysis (PCA) and dropout. 35
Figure 3.4 Illustration of FNN with dropout. 36
Figure 3.5 Illustration of optimal thresholding. 38
Figure 3.6 Mean F_1 for different values of combining the predicted upper and lower boundary. [77] 39

Figure 4.1 The (a) correlation between labels and (b) frequency of positive labels at 30 dB when gases are mutually independent. 42
Figure 4.2 The correlation between labels at 30 dB when gases are mutually independent. 43
Figure 4.3 The frequency of positive labels at 30 dB when gases are positively correlated. 44

Figure 4.4 The correlation between labels at 30 dB when gases are positively correlated. 45
Figure 4.5 The frequency of positive labels at 30 dB when gases are randomly correlated. 46
Figure 4.6 The correlation between labels at 30 dB when gases are randomly correlated. 47
Figure 4.7 FNN training and testing procedure 49

Figure 5.1 Mean squared errors of PLS with different number of latent variables. 54
Figure 5.2 Cross entropy loss vs number of iterations (SNR=30db) 55
Figure 5.3 Learning curves of FNN-OT without dropout (blue), with dropout (red) and PLS-BR (green). 56
Figure 5.4 Hamming loss vs retention probability on mutually independent gas data 57
Figure 5.5 The (a) micro-precision, (b) macro-precision and (c) sample-precision of FNN-OT, FNN and PLS-BR when gases are mutually independent. 59
Figure 5.6 The (a) micro-recall, (b) macro-recall and (c) sample-recall of FNN-OT, FNN and PLS-BR when gases are mutually independent. 60
Figure 5.7 The (a) micro-F_1 score, (b) macro-F_1 score and (c) sample-F_1 score of FNN-OT, FNN and PLS-BR when gases are mutually independent. 61
Figure 5.8 Caption for LOF 62
Figure 5.9 The (a) micro-precision, (b) macro-precision and (c) sample-precision of FNN-OT, FNN and PLS-BR when gases are positively correlated. 63
Figure 5.10 The (a) micro-recall, (b) macro-recall and (c) sample-recall of FNN-OT, FNN and PLS-BR when gases are randomly correlated. 64
Figure 5.11 The (a) micro-F_1 score, (b) macro-F_1 score and (c) sample-F_1 score of FNN-OT, FNN and PLS-BR when gases are randomly correlated. 65
Figure 5.12 The (a) micro-precision, (b) macro-precision and (c) sample-precision of FNN-OT, FNN and PLS-BR when gases are randomly correlated. 67

Figure 5.13The (a) micro-recall, (b) macro-recall and (c) sample-recall of FNN-OT, FNN and PLS-BR when gases are randomly correlated. 68
Figure 5.14The (a) micro-F_1 score, (b) macro-F_1 score and (c) sample-F_1 score of FNN-OT, FNN and PLS-BR when gases are randomly correlated. 69

Acronyms

ANN Artificial Neural Network. ix, 1, 2, 5, 13, 14, 21, 31

CNN Convolutional Neural Network. 21, 71

FNN Feed-forward Neural Network. iii, vi, ix, x, xi, 2, 3, 4, 5, 7, 8, 11, 13, 14, 19, 20, 21, 32, 36, 38, 49, 50, 55, 58, 57, 58, 60, 58, 61, 60, 61, 63, 64, 61, 64, 65, 66, 67, 68, 66, 70, 71, 72

FNN-OT Feed-forward Neural Network with Optimal Thresholding. iii, ix, x, xi, 7, 8, 9, 12, 13, 31, 32, 37, 49, 55, 54, 55, 56, 58, 57, 58, 59, 60, 58, 60, 61, 63, 61, 63, 64, 61, 64, 65, 66, 67, 68, 66, 70, 71

OT Optimal Thresholding. iii, vi, 8, 32, 49, 50, 70, 71, 72

PCA Principal Component Analysis. ix, 11, 32, 34, 35, 48, 50, 70, 71

PLS Partial Least Squares. viii, x, 31, 47, 48, 53, 70

PLS-DA Partial Least Squares - Discriminant Analysis. 48

PLS-BR Partial Least Squares - Binary Relevance. iii, vi, x, xi, 31, 47, 55, 54, 58, 57, 58, 61, 60, 61, 63, 64, 61, 64, 65, 66, 67, 68, 66, 70, 71

RNN Recurrent Neural Network. ix, 2, 21

SNR Signal-to-noise Ratio. iii, x, 10, 11, 35, 41, 42, 53, 54, 55, 56, 58, 60, 61, 63, 64, 65, 66, 67, 68, 70

ACKNOWLEDGEMENTS

I would like to express my gratitude to:

Dr. Tao Lu, for his mentoring, support, encouragement, and patience. His profound knowledge and academic integrity have been of great inspiration to me.

Dr. Wu-Sheng Lu and Dr. George Tzanetakis for their insightful courses and helpful comments.

my family and friends, for their endless support.

DEDICATION

To my lovely daughter Sylvie without whom this thesis would have been finished much earlier.

Chapter 1

Introduction

1.1 Context

Multi-label classification can be defined as the collection of tasks of mapping input features of an instance to multiple categorical outputs. As shown in Figure 1.1, it is a subset of classification tasks which belongs to supervised learning, and the superset of all of them is machine learning which enables "computers the ability to learn without being explicitly programmed"[69].

Supervised and unsupervised learning are two main subgroups of machine learning tasks that deal with different types of datasets. If the model is trained with the guidance of labeled output data, it is called supervised learning and the goal it to map the input data to the output variables. If the model is trained on unlabeled data samples, it is called unsupervised learning and the goal is to discover the underlying structure of the input data.

Supervised learning methods are applied on labeled dataset with prior knowledge of output values for training samples. Based on the type of output data, Supervised learning can be further categorized as classification and regression. Classification problems have categorical outputs, such as color (blue, red or yellow) and weather (sunny, rainy or snowy). The output variables of regression problems are numerical and continuous, such as house price and gas mileage. The goal of both classification and regression algorithms is to find the relationship between input and output data trough training samples and correctly and efficiently predict outcomes for future instances.

Multi-label classification is a type of supervised learning with more than one

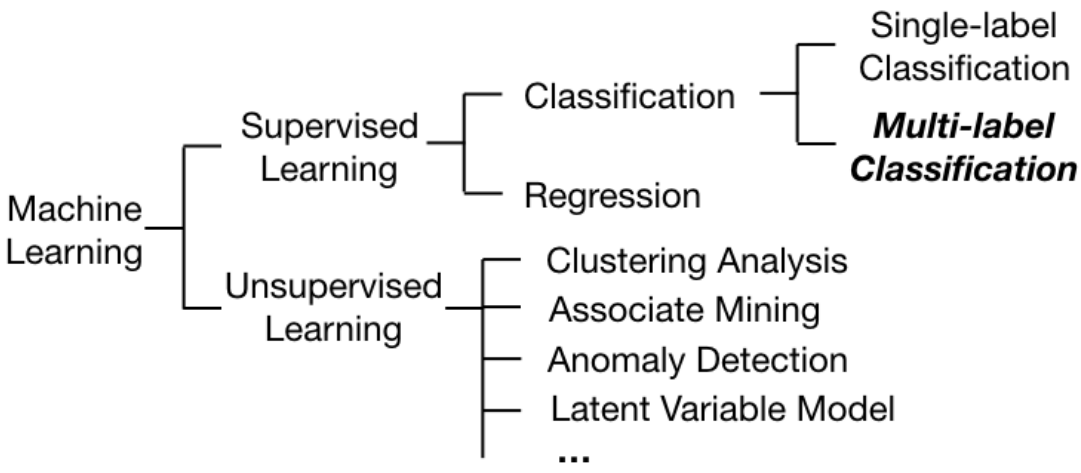

Figure 1.1: Categorizations in machine learning

categorical output labels. There is no specific relation between each output label. They can be independent, partially correlated or fully correlated. Intuitively, a simple solution for multi-label problems is to transform them into single-label classification tasks. That way classic single-label classifiers can be employed to deal with each classification task individually. This approach works well with small scale of output labels, because only a few single-label classifiers need to be trained. However, when the number of output labels is large, or single-label classifiers are complicated, the training process of the whole model can be extremely time consuming. Another solution for multi-label learning involves adapting single-label classifiers to produce more than one output label. Following this approach, in the past decades, various of machine learning methods have been adapted and proposed to solve this multi-label variant of classification problems. Many of them are based on ANN, which is a state-of-the-art machine learning technique.

ANNs are informed by our understanding of the human brain. Emulating the structure of our brains, ANNs have synapses that connect neurons and pass input through neurons to obtain the final output. Neurons in ANN can be aligned in different structures. The simplest architecture of ANN is a FNN [71]. It has one input layer, one output layer, and one or more hidden layers between them. Figure 1.2 shows the basic structure of aFNN. Unlike RNN illustrated in Figure 1.2, neurons in FNN only connect to neurons in the previous and following layers. There is no circle, loop or any other recursive structure in FNN. Information flows towards one direction:

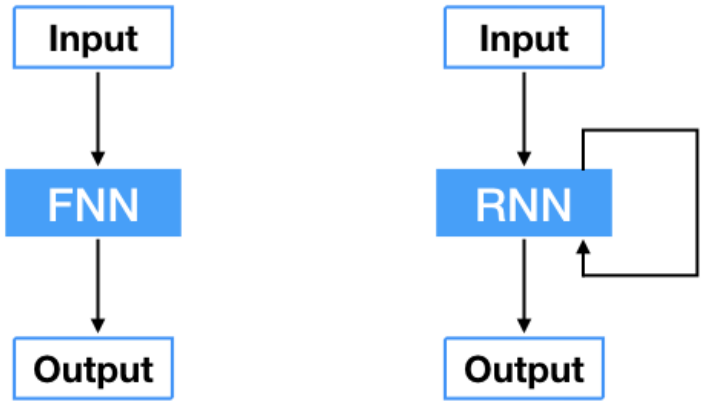

Figure 1.2: Information flow in FNN (left) and RNN (right)

from input layer to output layer.

Even though a FNN can be as simple as a three layer model (single hidden layer between input and output layer), it has the property of universal approximation, which means any continuous function can be approximated by a FNN model with desired accuracy [35][11]. In order to find the optimal weights in FNN model to approximate the relations between input features and output labels, the following approach is used:

1. Make observations and gather data.
2. Formulate hypothesis of weights and model structure.
3. Use training data and hypothesis to make predictions.
4. Compare predictions and targets and calculate the difference.

   If the difference is acceptable, continue to the next step.

   If not, return to Step 2 for new hypothesis.
5. Test hypothesis with testing data.
6. Analyze results and draw conclusions.

Data gathered in Step 1 is divided into two sets: a training set and a testing set. Data samples for training purpose in Step 3 cannot be used again in the testing phase

in Step 5. In some circumstances the model fits training data points so well that it causes generalization problem called overfitting. Overfitting models often have overly complex structures and excessive number of parameters so that they will memorize the dataset with inner noise and errors during training. They behave extremely well for training data points. For new entries, overfitting models can not make accurate predictions, because the models can not be generalized to data points that are outside the training set. If we use training samples for testing purposes, we will be mislead by the high accuracy of overfitting models. The overfitting problem will be discussed in Chapter 2, and the separation of training/testing datasets will be described in Chapter 3.

FNN is suitable for multi-label learing adaption, not only because of its universal approximation property, but also its flexible number of ouput labels in the output layer. It can produce predictions for all labels simultaneously. The problem is how to transform numerical outputs of FNN into categorical predictions for all instances in the testing set. An indicator function with a certain threshold will easily do the trick, so an optimal thresholding system becomes crucial in these processes. In this thesis, we propose an adaptive FNN model with an optimal thresholding system that can systematically generate optimal thresholds based on multi-composition spectroscopic analysis.

1.2 Research Problem

To our knowledge, no specific multi-label classification model has been proposed in detecting presence of a gas component inside mixtures using composition spectroscopy analysis. Research has been focusing on single-label classification [18] [72] or multi-output regression models [8] [81], which cannot be directly applied to our mulit-label classification problem.

As mentioned in Section 1.1, classification problems have categorical variables as their outputs. They can be divided into subgroups based on the characteristics of their outputs. One way to categorize classification problems is to count number of output labels. Labels of an instance are targets and outputs of the learning process. In statistics, these labels are often called dependent or response variables. For example, if there are L labels in a n instance of a classification problem, then if $L = 1$ it is a single-label classification, if $L \geq 2$ it belongs to multi-label learning. Another way to categorize classification is to count number of classes in each label. The learning

problem will be called binary classification when the number of class C equals to 2, and when $C \geq 3$ it is a multi-class classification problem. Depending on number of labels or classes of their outputs, classification tasks can be categorized into four groups in Table 1.1: single-label binary classification, multi-label binary classification, single-label multi-class classification, multi-label multi-class classification.

		CLASS	
		Binary	Multi-class
LABEL	Single-label	Single-label binary classification	Single-label multi-class classification
	Multi-label	Multi-label binary classification	Multi-label multi-class classification

Table 1.1: Categorization of classification tasks

Multi-label and multi-class, two different subsets of classification problems, often cause confusion because of their name similarity. Classes in multi-class learning are mutually exclusive but labels in multi-label learning are not. Each instance in the dataset can belong to one and only one of all possible classes, but it may be associated with none to all labels. Multi-class and multi-label classification can be demonstrated with two sample images in Figure 1.3. Given weather classes *sunny, rainy, cloudy* and *others*, Figure 1.3a can be classified as sunny, and Figure 1.3b will be rainy in the multi-class, single-label problem (Table 1.2). If more than one label — *tree, cloud* and *house*— need to be identified in the binary multi-label problem, Figure 1.3a will be positive in labels *tree* and *cloud*, and Figure 1.3a will be positive in *tree* only (Table 1.3).

The research problem that will be discussed and resolved in later chapters is identifying gas components from gas mixtures based on the multi-composition absorbance spectrum. It is a multi-label binary classification task which involves predicting the presence (binary classes) for all possible gases (multiple labels). In the rest of this thesis, this will be treated as multi-label classification problem only, because the main contribution of our research is to resolve classification issues caused by multiple labels of the output.

In other fields of research, such as text categorization and natural language processing, adaptive FNN models for multi-label classification have been proposed [87, 56], and more complicated ANNs have been introduced [80, 84]. Even though these models perform well on large scale text datasets, applying them on our problem is

(a)

(b)

Figure 1.3: Sample images of multi-class and multi-label classification. [a] [b]

[a] Source of Figure 1.3a: `https://www.wallpaperup.com/7661/Freshsunshinebehindthetree.html`

[b] Source of Figure 1.3b: `https://communityimpact.com/dallas-fort-worth/public-safety/2018/02/21/due-heavy-rain-avoid-23-closed-collin-county-roads/`

Image	Weather (sunny/rainy/cloudy/other)
(a)	Sunny
(b)	Rainy

Table 1.2: Multi-class classification for sample images in Figure 1.3

Image	Tree (Y/N)	Cloud (Y/N)	House (Y/N)
(a)	Yes	Yes	No
(b)	Yes	No	No

Table 1.3: Multi-label classification for sample images in Figure 1.3

problematic because:

- No pre-processing method such as data normalization or dimensionality reduction has been employed in [87] or [56]. The main reason is that input features of text datasets are sparse and categorical, but for spectroscopic datasets, input features are usually wavelengths. They are numerical and highly correlated in most cases. It is highly possible that multi-label classifiers will be less accurate and less efficient with original input data.

- Some of the state-of-the-art techniques of FNN are not applied.

More details of the existing machine learning algorithms for spectroscopic analysis and adaptive FNN models for multi-label learning will be provided in Chapter 2. On the basis of these studies, we propose a multiple-gas classification model for multi-composition spectroscopic analysis.

1.3 Proposed Approach

The approach proposed in this thesis for multi-label classification problem is called FNN-OT. It is an adaptive FNN model that can process multi-composition spectroscopic data for detection of gas species.

FNN models have been widely used for single-label classification tasks with multiple classes, because the mathematical constraint that these classes are mutually exclusive enables transforming numerical outputs into categorical predictions. Suppose we would like to use FNN to classify an instance into 3 possible classes c_1, c_2 and c_3, then the output of FNN will be a 1×3 vector (y_1, y_2, y_3). Since all classes are mutually exclusive, the instance belongs to one and only one of c_1, c_2 and c_3. So we

will compare output values y_1, y_2 and y_3 with each other, and the class with highest value will be the prediction of the instance. However, for multi-label classification tasks, even though FNN can provide an output value for each label, we still need other methods to determine the class of labels based on output values of FNN.

Following the basic scientific method in Section 1.1, we propose a FNN-OT model for multiple gas detection which will be trained and tested on spectroscopic data. Spectrum signals are firstly processed by a FNN model, which produces one output score for each gas. These output scores are then used as inputs to a following OT system. For every sample in the training set, its threshold will be determined by the OT which will be explained in Section 3.3. Then the output scores and thresholds are fed as input into a new FNN model that is used to calculate thresholds for testing samples as input and output variables. By comparing FNN output score with OT, prediction of each gas in one testing sample can be made. If its output score passes the threshold, the gas will be predicted as present in the sample. Otherwise, the prediction will be that such gas dose not exist in the sample.

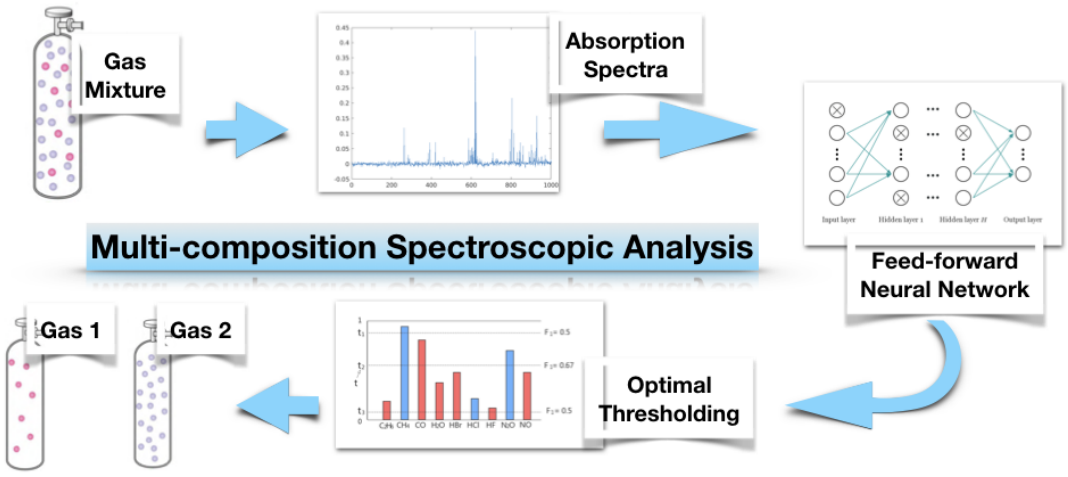

Figure 1.4: Illustration of FNN-OT model for multi-composition spectroscopic analysis

By selecting optimal thresholds, FNN-OT increase the performance of FNN, and outperforms multi-label adaption of conventional methods for spectroscopic analysis. Our test in Chapter 5 shows that it dynamically select a threshold thus reduces events where existing gases are mislabeled as absent. In addition, FNN-OT utilizes

correlation among the components to enhance its classification capability. Both of these unique features make FNN-OT a favorable choice for spectroscopic analysis in cluttered environments.

1.4 Method Illustration

In order to illustrate and evaluate our proposed classifier on multi-label learning, we apply FNN-OT to synthesized spectroscopic datasets for gas detection. Intuitively, the most straightforward way to detect gases from mixtures is to do chemical dynamic measurement directly for each target gas component. However, such measurement can be extremely difficult, expensive and time-consuming. Absorption spectroscopy provides a much simpler, faster and more practical way to conduct multi-composition analysis for gas mixtures.

Absorption spectroscopy is the experimental techniques that measures attenuation of radiation when passing through a medium. The outcome of the measurements is usually a function of radiation frequency or wavelength. The variation of the energy absorbed by a sample as a function of frequency comprises the absorption spectrum. This function, with its frequency eigenvalues, is very often used as an analytic tool to determine whether the radiation has interacted with a particular matter in a sample, and can further more be used to calculate the quantity or concentration of the substance present.

Usually these spectroscopic experimental set-ups are very intuitive. Simple idea

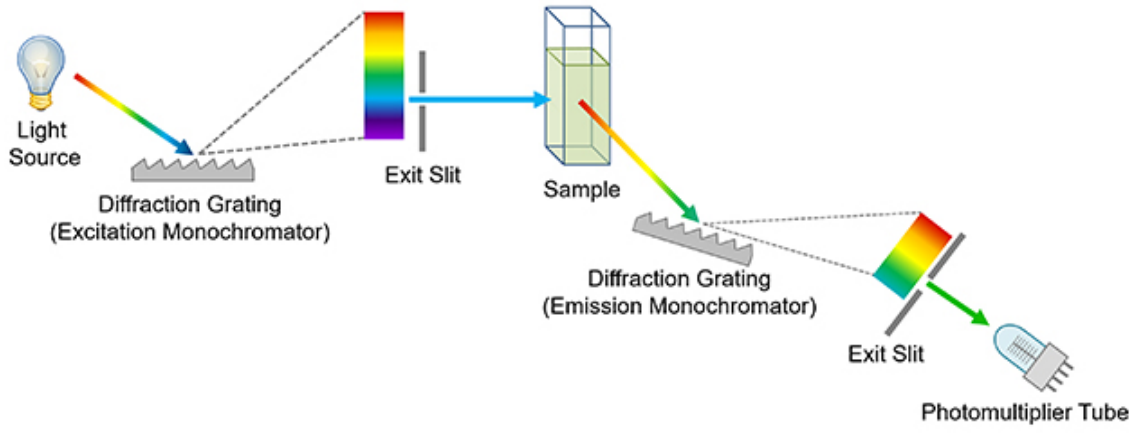

Figure 1.5: Illustration of an absorbance measurement. [a]

[a]Source: Edinburge Instrument: `https://www.edinst.com/blog/what-is-a-spectrometer/`

is to point a source at a detector with a sample in between. We need at least a reference spectrum of the radiation with nothing between source and detector, than the spectrum with sample, as only the sample spectrum would not be sufficient because it will be not only affected by the characters of the source, the experimental optic piece but also quality and wavelength dependencies of the detector. Combining these two spectrum determines materials‘ absorption spectrum.

The classical design of an absorbance measurement usually involves a white light source and a monochromator. Scanning wavelength is typically done by mechanical moving positions of a adjustable aperture. A photo-resistor and post amplifier is usually used to translate the radiation signal to an analog signal. The design is illustrated in Figure 1.5.

Modern spectral measurements normally input wavelength spectrum with a wavelength scanning laser. Scanning no longer depends on hard mechanics. It can scan much faster and can be fine tuned to a specific bandwidth of interest. We also use more advanced detectors, such as photo-diode with amplifier between cathode and anode.

When implemented properly, an absorbance measurement can be done with extreme accuracy, and all measurements are real time and immediate. To most materials that are not photon-sensitive, measurements can be repeated without any interference with the sample. Optical measurements, unless process is taking too long, are usually non-destructive. The method does not only keep the sample undisturbed, but can be made without bringing any instrument into contact with the sample. Light traveled through the sample already carries all information needed so the measurements can be done remotely. In our situation, these measurement cell can be put into set-up without placing an operator or instrument at risk.

A worldwide standard database for molecular absorption spectrum is called HITRAN (an acronym for High Resolution Transmission) [65]. It is developed and maintained by Harvard-Smithsonian Center for Astrophysics. In our study, synthesized datasets are based on single gas spectrums of C_2H_6, CH_4, CO, H_2O, HBr, HCl, HF, N_2O, and NO from the HITRAN database. Artificial Gaussian noise will be added to all datasets. Pre-set SNR of each dataset will be 0 dB, 10 dB,20 dB,30 dB,40 dB, or 50 dB. Correlations between nine gas labels are not constant for all datasets. Gases can be mutually independent, positively correlated, or randomly correlated. Therefore, our model will be evaluated, compared and analyzed under different combinations of SNRs and label correlations.

1.5 Research Contributions

The main contributions of our research are listed as follows:

- We have introduced multi-label classification technique to spectral analysis.
  - Results of our multi-label classifier have been compared with multi-label extensions of commonly used data processing techniques in spectroscopic analysis.
  - Micro, macro and sample averaged evaluation metrics have been employed to show the influence of label balance and label frequency to the classifier performance.
  - A standard estimate of detection capability called minimum concentration detection has been applied to analyze our classification results.
  - Relations of SNR and classifier performance have been revealed in our research.
  - Influence of correlations between gas labels have been illustrated and discussed in our work.
- We have improved existing multi-label adaptive models of FNN so adaptive FNN model designed for multiple gas detection can be used on spectroscopic dataset
  - Input data has been pre-processed by data scaling and PCA be for the input layer of FNN. PCA does not perform feature dimension reduction as it usually does. It is used to deal with the high correlation between input features in spectroscopic datasets.
  - A more flexible and efficient setting for optimal thresholding has been designed.
  - State-of-the-art techniques of FNN such as Adam Optimizer have been applied in our model.

1.6 Thesis Outline

The rest of thesis is organized as follows:

- In Chapter 2 we present background information and related work of FNN, multi-label classification and spectroscopic analysis.
- In Chapter 3 we present our proposed FNN-OT model.
- In Chapter 4 we introduce datasets, models and metrics that will be used for evaluation and comparison.
- In Chapter 5 we evaluate performance of FNN-OT and compare it with other multi-label classification models.
- In Chapter 6 we conclude results and contributions of our research. Limitations and possible future work are discussed as well.

Chapter 2

Background and Related Work

The model we will present in Chapter 3 is called FNN-OT. It is a multi-label classification model designed for detecting two or more gases simultaneously based on the absorption spectrum of the gas mixture. Before the discussion of our proposed approach, background and related work of FNN, multi-label classification and spectroscopic analysis will be presented and discussed in this chapter.

2.1 Feed-forward Neural Network

FNN is a branch of ANN that has no loops or circles in their structure. It dates back to 1940s when McCulloch and Pitts show that artificial neurons can compute and approximate functions [52]. ANN was initially applied to a limited class of problems of pattern recognition [64]. However, because of inherent flaws in their work [54] and limitations of computing powers, research of ANN was suspended for decades. In 1970s with increasing computing capabilities, studies on ANN grow rapidly. New neural networks [45, 2] and self-organizing networks [27] were proposed. Back-propagation, which is one of the most powerful and popular gradient-based training technique for parameters in FNN, was also proposed and developed in 1970s and 1980s [67, 68]. It is the algorithm that resolved the inherent problem proposed by [54] in the 1960s.

ANN consists of neurons and their connections. Neurons are basic computational units. The biological neurons are connected by synapses, dendrites and axons. Input signals are received by synapses. Signal is then influenced by synaptic strength, which is the result of a learning and training process. Then the signal passing through dendrites arrives at the body of a neuron where signal strength is compared against a

certain threshold. If the signal passes the threshold, the neuron will be activated and produce an output signal along the axon. Then the axon branches out and makes connection with other neurons through either neuron body, synapses or dendrites.

FNN mimics the functions of human brains by creating layers of neurons and building connections between them. As shown in Figure 2.1, in FNN, input signal x_1 is received and multiplied with synaptic strength w_1. All weighted input signals $w_1x_1, w_2x_2, ..., w_nx_n$ received from neurons in the previous layer will be added up. If the sum of input signals ($\sum_i w_ix_i + b$, b is the bias term) received by the neuron is large enough, the neuron will be activated and produce output signal $f(\sum_i w_ix_i + b)$ for the connected neuron in the next layer. If the input signal is below the threshold, the neuron will stay inactive and no output signal will be passed to other neurons. This "if...if not..." process can be represented by an activation function $f()$, which controls the activation rate of the neuron. An intuitive choice of activation functions is an indicator function $\mathbb{1}_{\sum_i w_ix_i+b>t}(x_i)$ where t is the activation threshold. However, the indicator function only has 0 gradient, which will cause trouble in adjusting parameters during the optimization process. The most common choices of activation functions are sigmoid, tanh and ReLU functions, which will be discussed in detail in Subsection 2.1.1. The output ($f(\sum_i w_ix_i + b)$) will be the input signal of subsequent neurons.

2.1.1 Activation Functions

In neural networks, activation functions are set to decide whether neurons should be "activated" or not. In each layer of a neural network model, an activation function transfers the weighted sum of input into output in an appropriate range. These activation functions and topological arrangements of neural computational units introduce non-linearity into the model. Without non-linear activation functions, neural networks with multiple hidden layers would be equivalent to single layer linear models.

Sigmoid Function

Sigmoid functions are some of the most widely used activation functions. A sigmoid function is a special logistic function with "S" shape curve. It is defined by the following formula:

$$h(z) = \frac{1}{1 + e^{-z}} = \frac{e^z}{e^z + 1} \tag{2.1}$$

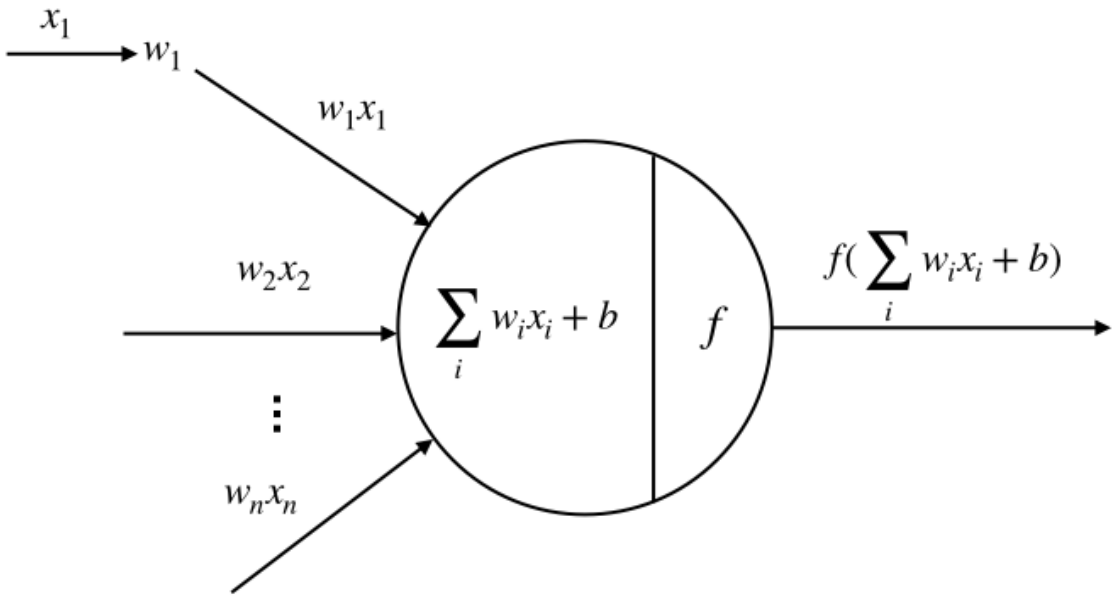

Figure 2.1: A neuron in ANN model

where $z = \boldsymbol{wx} + b$. The outputs of sigmoid functions are in range $(0, 1)$. With a steep slope around $z = 0$ and flat slope elsewhere, sigmoid function tend to map z into either end of the range $(0, 1)$ and make clear distinctions of predictions.

The function is bounded, continuously differentiable and monotonic increasing with respect to z. One disadvantage of sigmoid activation functions is that they have a vanishing gradient. As the absolute value of z increases, the gradient of sigmoid function decreases rapidly and will be close to zero. For deep neural networks with a large number of layers, after multiplication, the product of gradients that are close to zero will "vanish".

Tanh Function

Tanh is the hyperboic tangent function. It has the following mathematical formula:

$$h(z) = tanh(z) = \frac{sinh(z)}{cosh(z)} = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.2}$$

where $z = \boldsymbol{wx} + b$. Tanh function is a scaled sigmoid function $tanh(z) = 2sigmoid(2z) - 1$. It has similar advantages and disadvantages as the sigmoid function. Unlike sigmoid function, the output range of tanh function is $(-1, 1)$, which is

more convenient than the range $(0, 1)$ for some classification problems.

ReLU Function

ReLU is one of the most popular activation function nowadays, especially for deep learning and convolutional neural networks. ReLU is short for rectified linear units with mathematical formula:

$$h(z) = max\{0, z\} \tag{2.3}$$

where $z = \boldsymbol{wx} + b$. Neural networks with ReLU activation functions have significant improvement in convergence performance compared top sigmoid and tanh functions[46].

ReLU increases computational efficiency, because no expensive operations such as the exponential functions in sigmoid need to be performed in ReLU functions. Also, ReLU reduces the likelihood of vanishing gradient. When $z > 0$, the gradient will be constant, and it will remain positive after multiplication through a large number of layers. When $z \leq 0$, the gradient will be zero and result in sparsity of the model. Since ReLU turns all negative input into zeros, it can not map negative values well in neural networks. Some neurons may never be activated on any sample point. To solve this problem, extensions and variations of ReLU such as Leaky ReLU functions are proposed. Instead of setting zeros for all negative inputs, Leaky ReLU and other variations of ReLU functions use linear or exponential formula to deal with output less than zero.

2.1.2 Loss Functions

Loss functions mathematically measure errors of predictions. With the help of optimization methods mentioned above, the model will 'learn' from data and reduce prediction errors by minimizing results of loss function.

There are two main categories for loss functions: regression loss functions and classification loss functions.

Regression Loss Functions

- Mean Absolute Error (MAE)

  MAE calculates the mean of absolute deviations between observations and predictions.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{2.4}$$

where y_i is the observation of i-th sample and $\hat{y}_i$ is the prediction of i-th sample.

- Mean Square Error (MSE)

  MSE is the average of squared difference between predictions and true values. The formulation of MSE is as follows:

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n} \tag{2.5}$$

  where y_i is the observation of i-th sample and $\hat{y}_i$ is the prediction of i-th sample.

  Since in MSE the error is squared when compared to MAE it is more sensitive to outliers in the data which often can have large errors. Therefore, MSE is a good choice for problems where one needs to be attention to outliers. The L-2 norm of errors also gives MSE a nice mathematical property of gradient for optimization.

Classification Loss Functions

- Cross Entropy Loss(CEL)

  Cross Entropy is the most commonly used loss function for classification problems. It increases as the predictions diverge from observations.

$$CEL = -y_i log(\hat{y}_i) - (1 - y_i) log(1 - \hat{y}_1) \tag{2.6}$$

  where y_i is the observation of i-th sample and $\hat{y}_i$ is the prediction of i-th sample.

- Hinge Loss (HL)

  Hinge loss is mostly used to maximize margins for support vector machines with decision set $\{-1, +1\}$.

$$HL = \sum_{i=1}^{n} max(0, 1 - y_i \cdot \hat{y}_i) \tag{2.7}$$

  where y_i is the observation of i-th sample and $\hat{y}_i$ is the prediction of i-th sample. HL is non-differentiable, but it is convex with respect to $\hat{y}_i$ which makes it easier

to work with convex optimizers.

2.1.3 Optimization Methods

Optimizers are tools to minimize prediction errors by moving parameters towards the optimal choices. They update the model according to the output of loss functions.

Gradient Descent Optimizer

Gradient descent is one of the most basic iterative methods of optimization. It calculates the results of loss function $J(\theta)$ with respect to small changes of parameters θ, and adjust parameters based on the gradient of the loss function $\nabla_\theta J(\theta)$. Learning rate η controls the steps of moving parameters. One popular variant of gradient descent is stochastic gradient descent. In iteration t, it performs parameter update for each sample with the following formula:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta J_t(\theta) \tag{2.8}$$

One problem with the gradient descent method is that parameters often get stuck in local minima.If the loss function has multiple local minima, the optimization results will be affected by learning rate and the starting point of parameters. Another problem comes from fixed learning rate η. If learning rate is too large for the loss function, parameters will probably skip the minimum and the algorithm may never converge to an optimum. If learning rate is too small then it will take too much time and too many steps to converge, and the optimizer has higher probability of reaching to a local minimum, not the global one.

AdaGrad Optimizer

Unlike Gradient Descent Optimizer, AdaGrad has per parameter adaptive learning rate. AdaGrad performs larger iterative updates for sparse parameters and smaller updates for less sparse ones. At iteration t, parameter θ_i is updated as follows:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,i} + \epsilon}} \nabla_\theta J_t(\theta) \tag{2.9}$$

where ϵ is a small term (usually 10^{-8}) to avoid zero denominator. $G_{t,i}$ is the sum of all the squares of gradients from the first to t-th iteration with respect to θ_i. It is

the i-th diagonal element in the diagonal matrix G_t.

It outperforms stochastic gradient descent for sparse data sets because AdaGrad improves the convergence performance. It also improves the robustness for large scale neural nets [12].

The main weakness of Adagrad is caused by shrinking learning rate. Since $G_{t,i}$ in the denominator in the accumulated sum of the squared gradients, the adaptive learning rate of AdaGrad monotonically decreases with respect to iterations. After a certain number of iterations, the learning rate will be extremely small so that parameters stop updating.

Adam Optimizer

Adam is short for Adaptive Moment Estimation. In Adam optimizer, past gradients are used to calculate current gradients. It calculates the decaying average of first (m_t) and second moments (v_t) of past gradients as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla_\theta J_t(\theta) \tag{2.10}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)[\nabla_\theta J_t(\theta)]^2 \tag{2.11}$$

where hyperparameters β_1 and β_2 are close to 1. Since the two estimates m_t and v_t have biases towards zero, the following estimates $\hat{m}_t$ and $\hat{v}_t$ are used instead:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{2.12}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{2.13}$$

Parameter θ_t is updated with the following rule:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t \tag{2.14}$$

Adam solves vanishing learning rate in Adagrad, and it performs well for sparse gradients which often appears at the end of optimization [43]. It is one of the best choice for complex neural network models [66].

2.1.4 Number of Neurons and Layers

FNN consists of an input layer, an output layer and one or more hidden layers in between. Even though neurons on hidden layers do not directly interact with input features or output predictions, they play an crucial role on the final output of the FNN model. So the number of hidden layers and their sizes need to be determined before training and testing any dataset.

Number of Hidden Layers

In most cases, FNN models have 1 or more hidden layers. Otherwise, it will become a linear function (when no activation function is applied in the output layer) or sigmoid function (when sigmoind function is employed as activate function in the output layer). Without any hidden layer, FNN models can only represent linearly separable functions. Based on universal approximation therom, single hidden layer models are capable of approximating any continuous mapping from one finite space to another. For many practical problems, FNN models with one hidden layer is the optimal choice. In most cases, theoretically, it is not necessary to employ two or more hidden layers in FNN models. [30]

Number of Neurons

The number of neurons inside each hidden layer is another decision that has to be made for the architecture of FNN model. Even though universal approximation theorem states that one hidden layer FNN can approximate any continuous function, it does not specify the number of neurons the approximation needs in the hidden layer. If the number of neuron is too small, the FNN model will not have enough freedom to learn highly complex and nonlinear functions. If the number is too large, the FNN model will suffer from overfitting problems [88].

Blum provides an answer called 'rules of thumb' for hidden layer sizes [4]. It states that the number of neurons in hidden layer should between the size of input and output layers. However, [74] argues that such rules are not universally applicable to all FNN models. The sizes of hidden layers depend not only on input and output layer sizes, but also on training sample size, regularization method and function complexity.

2.1.5 Dropout

Overfitting is one of the common problems happened to large neural network models with fixed-sized datasets. Those models learns the noise in training sets so well that they can not be generalized into separate testing or validation data sets. Overfitting will result in excellent performance in training set and poor performance in testing set. It can be observed by the large gap between the low training loss and high testing loss in learning curves.

One way to reduce overfitting is to ensemble different neural networks, but it may give rise to expensive computational cost by training and storing those models. Another way is to simulate multiple parallel neural networks by randomly dropping out neurons in input and hidden layer during the training phase. By dropping out a percentage of neurons along with their incoming and outgoing connections, the neural network model can reduce the co-adaption of input features and the interdependent learning of units in the training phase [73].

In every iteration of the training phase, each neuron in visible (input) or hidden layer will be dropped randomly with probability $1 - p^{(i)}$ where $p^{(i)}$ is the probability of units in layer (i) that will be kept in the training. In the testing or validation phase, all neurons in layer (i) will be used for prediction, but they will be reduced by the factor $1 - p^{(i)}$. Suppose N of all neurons in the network have the possibility of being dropped, then there will be 2^N possible models in training phase, and the entire network will be considered in the testing phase.

The hyperparameter introduced by dropout is retention rate of input and hidden layers. A typical choice of retention rate is 0.8 for input layer and 0.5 for hidden layer, but the optimal dropout hyperparameter varies from one model to another.

2.2 Multi-label Classification

The development of multi-label classification dates back to 1990s when binary relevance [85] and boosting method [70] were introduced to solve text categorization problems. Significant amount of research was done after that, and the multi-label learning has been widely used in areas such as natural language processing and image recognition [76, 22]. Most of the multi-label classification algorithms fall into two basic categories: problem transformation and algorithm adaption. Problem transformation algorithms transform a multi-label problem into one or more single-label

problems. After the transformation, existing single-label classifiers can be implemented to make predictions, and the combined outputs will be transformed back into multi-label representations. One of the simplest problem transformation methods is binary relevance. It transforms a multi-label problem by splitting it into one binary problem for each label [24, 41]. Under the assumption of label independence, it ignores the correlations between labels. If such assumption fails, label powerset and classifier chains are known transformation alternatives where label powerset maps one subset of original labels into one class of the new single label [78] and classifier chains passes label correlation information along a chain of classifiers [63]. Algorithm adaption methods modify existing single-label classifiers to produce multi-label outputs. For instance, the extensions of decision tree [9], Adaboost [70], and k-nearest neighbors [86] are all designed to deal with multi-label classification problems. ANN is suited to be adapted for multi-label learning as well. Restricted Boltzman machine [62], FNN [87, 56], Convolutional Neural Network (CNN) [10, 25], and RNN [80] are employed to characterize label dependency in image processing or find feature representations in text classification. Those adaptive methods can identify multiple labels simultaneously and efficiently without repeatedly trained for sets of labels or chains of classifiers.

2.2.1 Formal Definitions of Multi-label Classification

Definitions related to multi-label classification problems are listed as follows. Those definitions and related symbols will be used throughout the rest of this thesis.

Input

Let $\mathcal{X} = X_1 \times X_2 \times ... \times X_d$ be the d-dimensional input space, then the input vector of one instance will be $\boldsymbol{x} = (x_1, x_2, ..., x_d)$, which is defined in $\mathcal{X}$. Those input attributes x_i can be either numerical or categorical. However, categorical inputs needs to be transformed into numerical representatives. For example, the two classes of binary input variables can be written as 0 and 1, or -1 and 1. Suppose there are n samples in a dataset, then the input is a $n \times d$ matrix $\mathbf{X}$.

Output

Let $\mathcal{Y} = Y_1 \times Y_2 \times ... \times Y_l$ be the l-dimensional output space, then the output vector of one instance will be $\boldsymbol{y} = (y_1, y_2, ..., y_l)$. Those output labels y_i can be numerical

or categorical. In classification problems, only categorical outputs are taken into consideration. In a dataset with n samples, the output matrix is $\mathbf{Y}$ with size $n \times l$.

Suppose there are k possible classes $C_1, C_2, ..., C_k$ in every label, then the output of one label y_i will belongs to one of those classes. So the output vector of l-label k-class classification problems can be written as $\boldsymbol{y} = (y_1, y_2, ..., y_l) = \{C_1, C_2, ..., C_k\}^l$.

Classifier

The multi-label classifier will be defined as $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Y}$. For a new instance with attributes $\boldsymbol{x} = (x_1, x_2, ..., x_d)$, its prediction will be $\hat{\boldsymbol{y}} = \mathcal{F}(\boldsymbol{x})$.

2.2.2 Algorithms for Multi-label Learning

Most multi-label classification algorithms fall into two basic categories: problem transformation methods and algorithm adaption methods.

Problem Transformation Methods

Problem transformation methods transform a multi-label problem into one or more single-label problems. After the transformation, well-developed single-label classifiers can be implemented to make predictions, and the combined outputs will be transformed back into multi-label representations. Commonly used problem transformation methods are listed as follows.

- Instances selection

  Instances selection is a simple transformation technique [6]. For every instance in the training data set, only one label y_i is selected from its label vector $\boldsymbol{y}$. Label selection can be random or based on label frequency (select-max or select-min). This method ignores labels that are considered to be irrelevant and the possible correlations between different labels for each instance. It will result in inaccuracy due to information loss.

- Label Powerset

  Label powerset converts multi-label problems into multi-class ones [78]. It maps one subset of original labels into one class of the new single label. So every instance in the data set is assigned to a class of new label, and the single-label problem can be handled with classic classification methods. For testing or

predicting new entries, new label can be mapped back to original label combinations. This method is only suitable for small label number l, because for k class labels, there are k^l possible combinations, resulting in a sparse data set which is difficult to classify. Moreover, the computation complexity will increase exponentially as well.

- Binary Relevance

  Binary relevance is the most well known transformation method that splits a multi-label problem into one binary problem for each label [24]. In the original data set, a label will be considered as relevant or irrelevant for each instance regardless of relevance of other labels, thus k new data sets will be generated. Each new data set will be processed by single-label classification method seperately. Under the assumption of label independence, it ignores the correlations between labels.

- Classifier Chains

  Classifier Chains extend the binary relevance method and pass label correlation information along a chain of classifiers[63] . Like binary relevance method, it deals with k new data sets with k binary classifiers. The difference between classifier chains and binary relevance is that classifier chains incorporates predictions of all previous classifiers into the current one as additional features. For an instance with original input vector $\boldsymbol{x} = (x_1, x_2, ..., x_d)$, the classifier for the j-th data set has new input vector $(\boldsymbol{x}, \hat{y}_1, \hat{y}_2, ..., \hat{y}_{j-1})$ where $\hat{y}_1, \hat{y}_2, ..., \hat{y}_{j-1}$ are predictions of previous classifiers. The outcome of j-th data set will be used in following classifiers as well. The order of classifiers can be random, and an ensemble method based on the random order of classifiers is a useful extension of classifier chains. Another extension use Monte Carlo methods to search for an optimal sequence of the classifier chain.

Algorithm Adaption Methods

Algorithm adaption methods modify existing single-label classifiers to produce multi-label outputs. It is hard to categorize algorithm adaption methods since almost all classic single-label classification methods have extensions for multi-label problems. For instance, the extensions of Adaboost called Adaboost.MH and Adaboost.MR [70], and the adaption of K-nearest neighbours called ML-KNN [86] are designed to

deal with multi-label classification problems. BP-MLL [87] is also a adaptive neural network for multi-label learning.

2.2.3 Evaluation Metrics

In order to test the performance of supervised models, evaluation metrics are employed to demonstrate model performances in testing data sets. Presenting numerical testing results, those metrics are important criteria for model comparison and selection. For single-label classification problems, evaluation metrics are easily defined and calculated. Precision, recall and F1 score are commonly used for model evaluation. They are also foundations of evaluation metrics for multi-label classification models

Evaluation Metrics for Single-label Classification

- Confusion Matrix

  Confusion matrix is also known as error matrix. It is a $k \times k$ table that describes the confusion/error made by the supervised model. k denotes number of classes for each label. In our binary classification case, $k = 2$. Columns of this table represents numbers of each predicted class while rows of confusion matrix counts the actual number of instances in each class. A simple illustration of confusion matrix is in Table 2.1. True negatives (TN) are negative cases that predicted as negatives. Similarly, true positives (TP) are positive cases with correct predictions. False negatives (FN) are negative predictions for positive cases, and false positives (FP) are negative cases with positive predictions. TN, TP, FN and FP are fundamental elements for following evaluation metrics.

		PREDICTED CLASSES	
		Negative	Positive
ACTUAL CLASSES	Negative	True Negatives (TN)	False Positives (FP)
	Positive	False Negatives (FN)	True Positives (TP)

Table 2.1: Confusion Matrix for Binary Classification

- Accuracy

  Accuracy is the percentage of correctly predicted instances over all instances. In other words, it is the ratio of true positives and true negatives to the sum of all four categories.

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{2.15}$$

- Precision

  Precision is the proportion of the predicted correct labels to the total number of actual labels averaged over all testing instances. In other words, it is the ratio of true positives to the sum of true positives and false positives averaged over all instances.

$$Precision = \frac{TP}{TP + FP} \tag{2.16}$$

- Recall

  Recall is the proportion of the predicted correct labels to the total number of predicted labels averaged over all instances. In other words, it is the ratio of true positives to the sum of true positives and false negatives averaged over all instances.

$$Recall = \frac{TP}{TP + FN} \tag{2.17}$$

- F1 score

  F1 score is given by the harmonic mean of Precision and Recall. It balances precision and recall, indicating an overall accuracy of prediction. Mathematically it can be expressed as follows.

$$\begin{aligned} F1\ score &= 2 \cdot \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}} \\ &= \frac{2 \cdot TP}{2 \cdot TP + FN + FP} \end{aligned} \tag{2.18}$$

Precision, recall and F1 score above are defined and calculated on one dimension of the output variables. They can be applied to either one output label of all testing samples, or all output labels of one testing sample. But for multi-label classification models, one complexity arises when we discuss evaluation metrics for the whole model: how to obtain the average metrics of all instances and all labels.

Averaging Methods for Multi-label Classification

Generally speaking, there are three types of averaging methods for evaluation metrics [49]. In the following metric averaging methods, $M(TN_j, TP_j, FN_j, FP_j)$ denotes a specific evaluation metric for label j where $M \in \{precision, recall, F1\ score\}$[3].

- Micro Averaging Method

  Micro averaging method calculates metrics globally by counting the total TP, TN, FP and FN of the whole datasets. Label and sample differences are ignored for micro averaging metrics.

$$M_{micro} = M(\sum_{j=1}^{l} TN_j, \sum_{j=1}^{l} TP_j, \sum_{j=1}^{l} FN_j, \sum_{j=1}^{l} FP_j) \tag{2.19}$$

- Macro Averaging Method

  The macro is the unweighted average of the metrics taken separately for each label. It is an average over labels without considering the data imbalance between labels.

$$M_{macro} = \frac{1}{l} \sum_{j=1}^{l} M(TN_j, TP_j, FN_j, FP_j) \tag{2.20}$$

- Sample/Example Based Averaging Method

  Sample(or example) based averaging methods calculate metrics separately for every sample, then find their unweighted mean.

$$M_{sample} = \frac{1}{n} \sum_{i=1}^{n} M(TN_i, TP_i, FN_i, FP_i) \tag{2.21}$$

  where n is the sample size.

In addition to the nine combination of evaluation metrics and averaging methods mentioned above, we will use hamming loss to evaluate our models as well. It is the proportion of incorrectly predicted labels. For single-label binary classification problems, $hamming\ loss = 1 - accuracy$. For multi-label cases,

$$hamming\ loss = \frac{1}{l \cdot n} \sum_{i=1}^{n} \sum_{j=1}^{l} \frac{FN_{ij} + FP_{ij}}{TN_{ij} + TP_{ij} + FN_{ij} + FP_{ij}} \tag{2.22}$$

Four evaluation metrics for single-label learning and ten corresponding extensions for multi-label cases are listed in Table 2.2.

Single-label Metrics	Multi-label Metrics		
Accuracy	Hamming Loss		
Precision	Micro-precision	Macro-precision	Sample-precision
Recall	Micro-recall	Macro-recall	Sample-recall
F1 Score	Micro-F1 Score	Macro-F1 Score	Sample-F1 Score

Table 2.2: Evaluation Metrics for Single-label and Multi-label Classifications

2.3 Absorption spectroscopy

Absorbance spectroscopy is used in a wide range of areas. In life sciences, absorbance of in-vivo [39] or in-vitro [39, 48] blood smaples are measured to analyze nucleic acids and proteins. In industrial plants as well as environmental monitoring towers, it was used to determine airborne and pollutants [15], soil contamination, even to predict volcanic activities by calculating sulfur-dioxide [75] concentration from these optical spectrum. In chemistry, absorption spectroscopy was used to monitor reactions [58], to analysis kinetic endpoints, unless the reaction kinetics is highly sensitive to photons, optical measurement are always the first choice to use as it does not disturb the reaction and are totally non-intrusive and non-destructive. In consumer products, absorption spectroscopy were installed in devices that alert people about food decay [14], for example, it shines light in dairy products and determine its poisonous components, or predict order and flavor in wines and fruits.

Absorption spectroscopy was already in existence long before modern science or technology. Old alchemist already managed to identify and understand their substance by looking at the color and opacity of solutions as different reagents were mixed, heated, and stirred.

Absorption spectrum was noticed as dark lines in a rainbow (a spectrum), some centuries ago. A huge step forward in the systematic understanding of such absorbance behavior of a substance took place in the early 19 century, using the underlining set-up (Figure 2.2) William Wollaston [83] and Fraunhofer [29, 17, 40] are able to definitively observe 'Fraunhofer lines' in a expanded spectrum of sun light and some other artificial light source passes through opaque matters onto a wall,

these tiny dark lines, are thousands of missing pieces of a original spectrum, were the absorption spectrum [44].

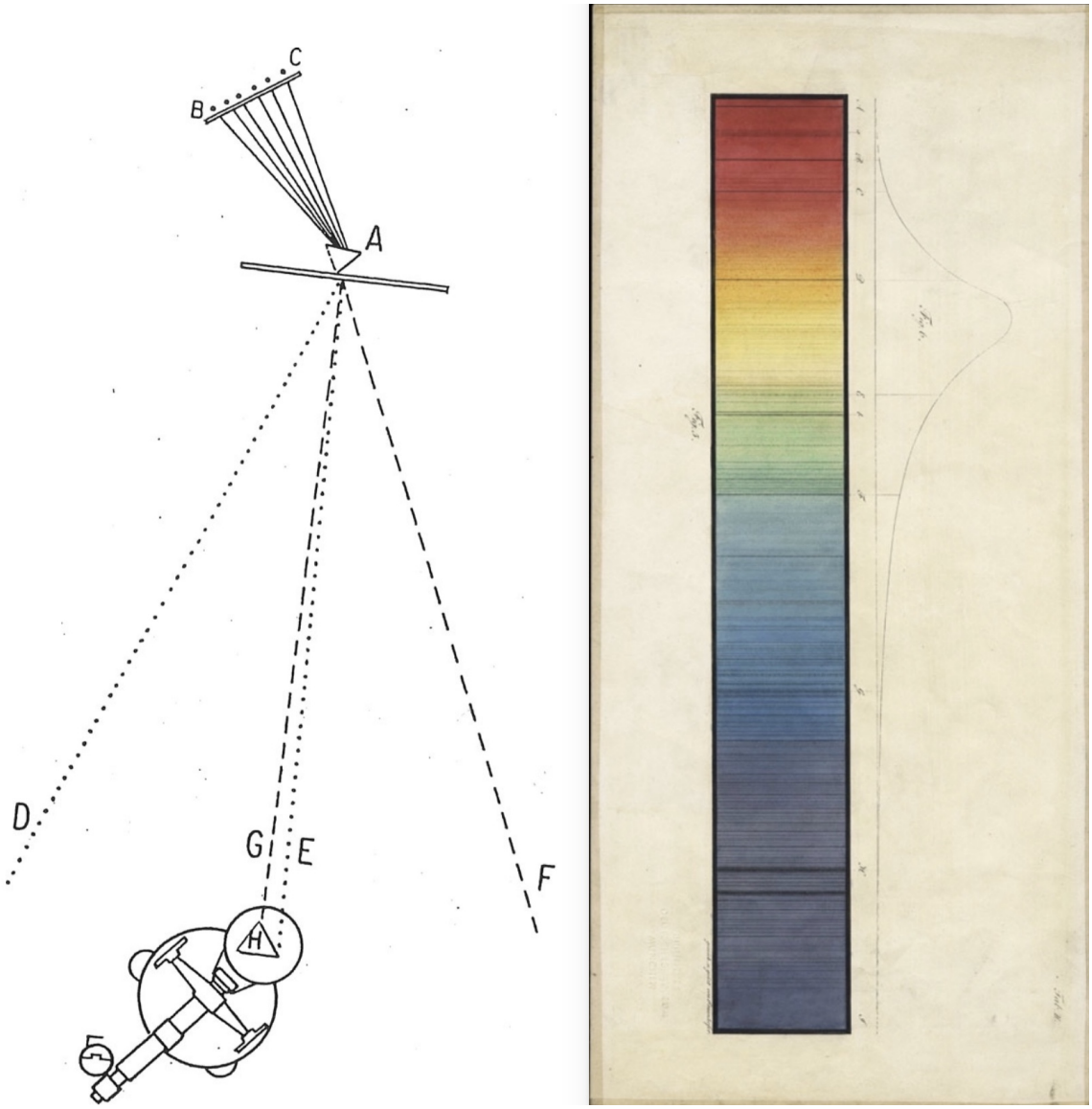

Figure 2.2: Fraunhofer's experiment for detecting the dark lines of the solar spectrum. [79, 38]

When light passes through gas in the atmosphere some of the light at particular wavelengths is scattered or absorbed resulting in darker bands. These lines came to be known as 'spectral lines' and were cataloged by heating common elements until they produced light and measuring the wavelengths emitted. [65]

With modern scientific instruments and methods, especially invention and im-

provement of laser and optical pieces, absorbance measures are one of the most widely used spectroscopic technique for studying fluid matters (liquid, gases, even plasma and some solid). The optical measurement of a substance' absorbance spectrum is simple, accurate, and easy to perform. Optical absorbance is also extremely quantifiable, make it the first-thought-of when people are trying to find a molecular's chemical signature, or the concentration of it in a diffused solution, weather it is liquid or gas form.

These fingerprint of particular chemicals enabled astronomers to determine composition of a distant stars, and also enabled examiners in a factory to determine composition of a product.

The most common experimental set up is to detect the intensity of the output beam that past through the sample. The most common calculation and theory associated with the intensity attenuation is the Beer-Lambaert law [37] where intensity decreases with depth inside the material, accordance with the exponential decay [5, 47, 3]:

$$T = e^{-\mu l} \tag{2.23}$$

Where T is the transmittance of material sample, μ is the attenuation coefficient, l is the path length of the beam of light through the material sample.

This law dictates a monochromatic behavior of the radiation intensity. When a spectrum was considered, the intensity been absorbed correspond to the integrated area under the absorption line, this is also proportional to the amount of absorbing substance present in the sample.

The absorption line considered here, usually take the shape of a Gaussian or Lorentzian distribution [34]. This shape usually depend on the instrument used in conducting this experiment [61]. More importantly, the width of downer limit of the absorption line distribution, which is also the resolving limit, is also determined by the instrument used [20]. A line width is only meaningful when its larger than the resolution of the instrument, this is broadening [59] is caused by the absorber, and can be used to determine substances in the sample or how dense the this substance is inside the sample, or even how different substances were convoluted under the temperature.

In principle, we expect lines of absorption spectrum of gases to correspond to the exact wavelength/frequency that fills the material particles' energy gap. These

energy state structures of the molecules, sometime of various mechanism, sometime act together to dictate a substance's absorptive behaviour of radiation. That is why normally the set of wavelengths or intensity of these spectral lines are normally not independent of each other, given different spectral lines sometime contains the information of the same energy gap between the same set of energy state. Such collinearity of spectral data will cause failures in some classical regression methods such as ordinary least squares. Spectral lines of gases are also consider to be 'cleaner', given molecules in gas-form, unless in extreme states, are often in a more independent state of each other because distance between gas molecules are much greater than their interactive ranges. There are fewer elements, for example collision and interaction between molecules, that broadens the gas spectral lines. The sparsity of spectral data will results in redundancy of input signals.

Such collinearity and redundancy in the spectroscopic datasets make multivariate regression algorithms such as principal component regression (PCR) [8] and PLS [81] the most fundamental and popular tools for spectroscopic analysis. PCR and PLS solves correlations by rotating coordinates and transferring original input signals into uncorrelated principal components (PC) or latent variables (LV). Moreover, by keeping the most relevant PCs or LVs, they reduce the input dimension and resolve the redundancy in spectral data. So in our study, a multi-label extension of PLS called PLS-BR will be applied and evaluated.

In addition to multivariate regression algorithms, non-linear methods, such as support vector machine [72], genetic programming [26] and ANNs [18], are also adopted to increase prediction accuracy. These linear and non-linear algorithms focus on either regression or single-label classification problems. Using multi-label classification to identify multiple chemical components from the spectrum, is under explored in spectroscopic analysis. Consequently, relations between labels in multi-label tasks can be either independent or correlated. So our adaptive model FNN-OT will be compared with PLS-BR on uncorrelated, positively correlated and mixed correlated gas data.

Chapter 3

Feedforward Neural Network with Optimal Thresholding

In single-label learning, a typical approach to classify an instance is to rank the probabilities (or scores) of all classes and choose the class with the highest probability as prediction. For multi-label problems, the same ranking system can be used to compute scores for all labels instead, then a threshold will be determined to assign all labels whose scores are higher than the threshold to the sample. In the FNN-OT model, scores of all labels need to be calculated for ranking purpose, and an optimal thresholding system will be employed to assign a set of labels to all samples. The whole process of FNN-OT is shown in Figure. 3.1. Spectrum Signals are firstly pre-processed by PCA. The output principal components are the input features of a FNN model called FNN-1, which produces one output score for each gas. Output scores will be the input of a following optimal thresholding (OT) system. For every sample in the training set, its threshold will be determined by OT illustrated in Figure. 3.5. Its mechanism will be explained in Section 3.3. Then the output scores and thresholds are the input and output variables of a new FNN model called FNN-2 which will be used to calculate thresholds for testing samples. Predictions of each gas component in one testing sample will be made by comparing its output score from FNN-1 against threshold from OT. If its output score is larger than the threshold, the prediction is 1 and 0 otherwise, representing the existence/non-existence of that gas component in the spectrum.

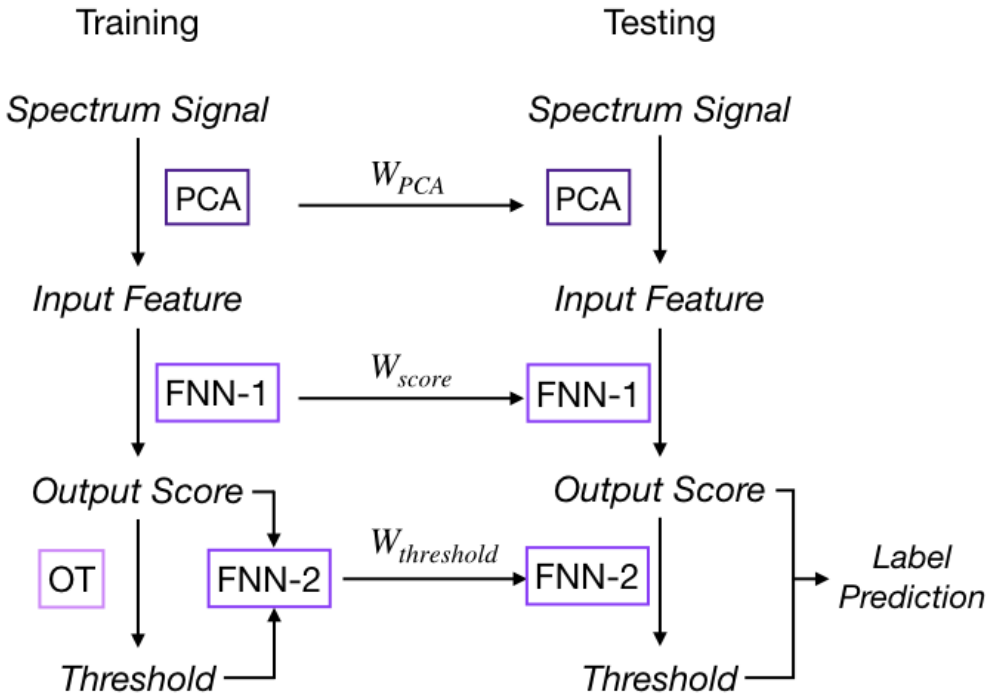

Figure 3.1: FNN-OT training and testing procedure.

3.1 Pre-processing Datasets

In both training and testing, the absorption spectra will be pre-processed with PCA, and the principal components will be the input of the FNN model ($\mathbf{x}$). PCA is a commonly used pre-processing method for spectroscopic datasets. It is conventionally employed to reduce feature dimension by transferring original input variables into a smaller set of uncorrelated principal components (PC) that preserves highest explained variance [33].

Data Normalization

The data pre-processing procedure begins with mean-centering and scaling features. These are basic pre-processing techniques that will make calculations easier. For the $n \times m$ feature matrix $\boldsymbol{X}$ with m features and n samples, each feature vector $\boldsymbol{x_j} = (x_{1j}, x_{2j}, ..., x_{nj})^T$ will be normalized by

$$\bar{\boldsymbol{x}}_{\boldsymbol{j}} = \frac{\boldsymbol{x_j} - \mu_j \boldsymbol{e_n}}{\sigma_j} \tag{3.1}$$

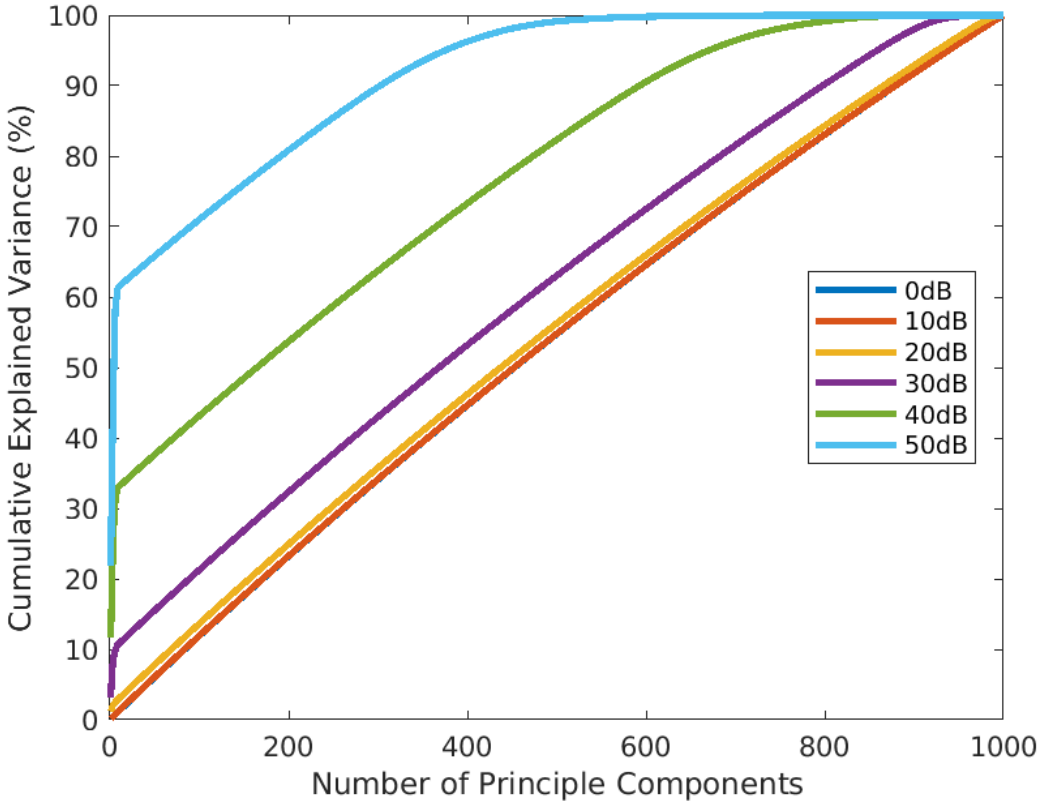

Figure 3.2: Percentage of cumulative explained variance vs. number of principal components adopted.

where μ_j, σ_j are mean and standard deviation of $\boldsymbol{x_j}$, and $\boldsymbol{e_n} = (1, 1, ..., 1)^T$.

The normalization step has two components: mean centering $(\boldsymbol{x} - \mu\boldsymbol{e})$ and variance scaling $(\frac{1}{\nu})$. Both of them are essential preliminary procedures for the feature dimensionality reduction method — PCA — in the next step. If mean centering is not performed, the first principal component will describe more of mean of input data, not the direction of maximum variance [55]. The reason for variance scaling is that PCA is sensitive to the scale of input features. If the unit or scale of one feature is changed, then variance of this input vector changes correspondingly, leading to a different direction of maximum variance, but in fact the input data is still the same.

PCA

The second pre-processing step is to reduce the dimension of input features. PCA is one of the most popular methods for linear spectral dimensionality reduction. It

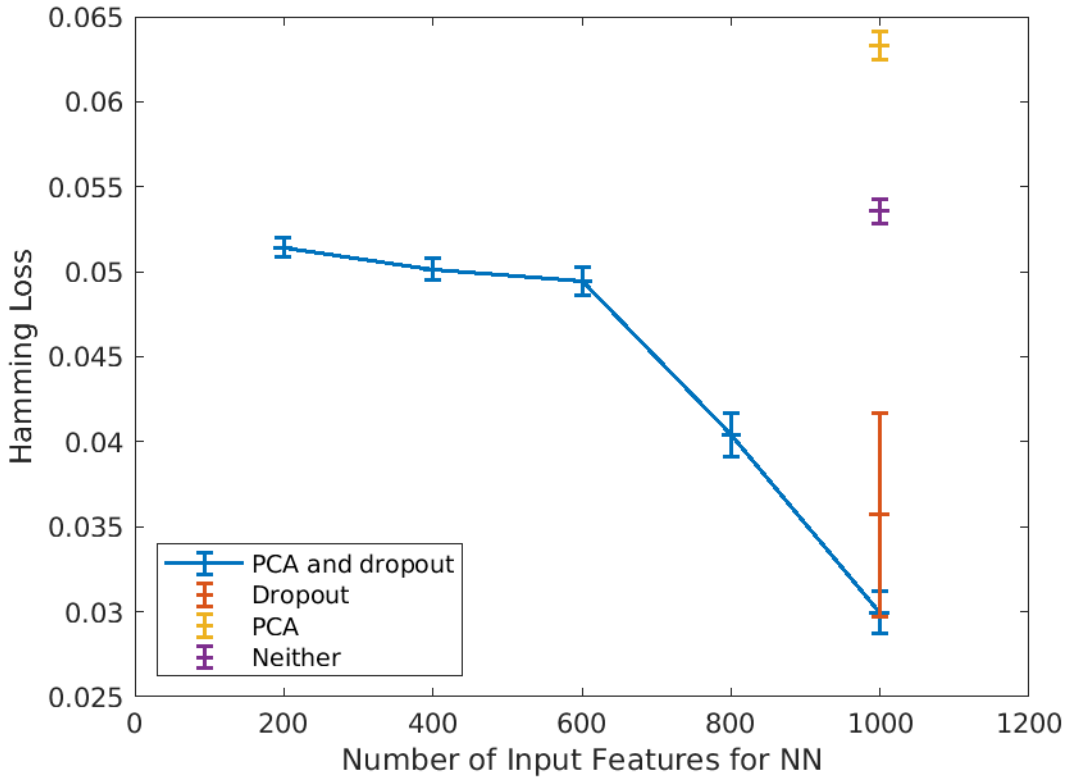

Figure 3.3: Comparison of Hamming loss with and without PCA and dropout.

seeks a set of coordinates to project input data to a lower dimensional subspace to reduce the dimension. PCA can be defined as a linear transformation method to rotate the original coordinate system into a new one to coincide with the directions of maximum variance. The first coordinate that the maximum variance of the original input data lies on is called the first principal components. The successive principal components are orthogonal to the previous ones and they maximize the variance of projected points. In this transformation, the first principal component keeps as much information contained in the variance of the original data set as possible, and each of the succeeding principal components represent as much of the remaining variability of the input data as possible. The percentage of variance explained by principal components can be calculated to evaluate the information loss in PCA transformation.

In our datasets, training samples are used to estimate the covariance matrix. Then both training and testing data are transformed using the same PCA transformation. As shown in Figure 3.2, at high SNR, PCA is an efficient technique for dimension reduction as only a small number of principal components is sufficient to preserve

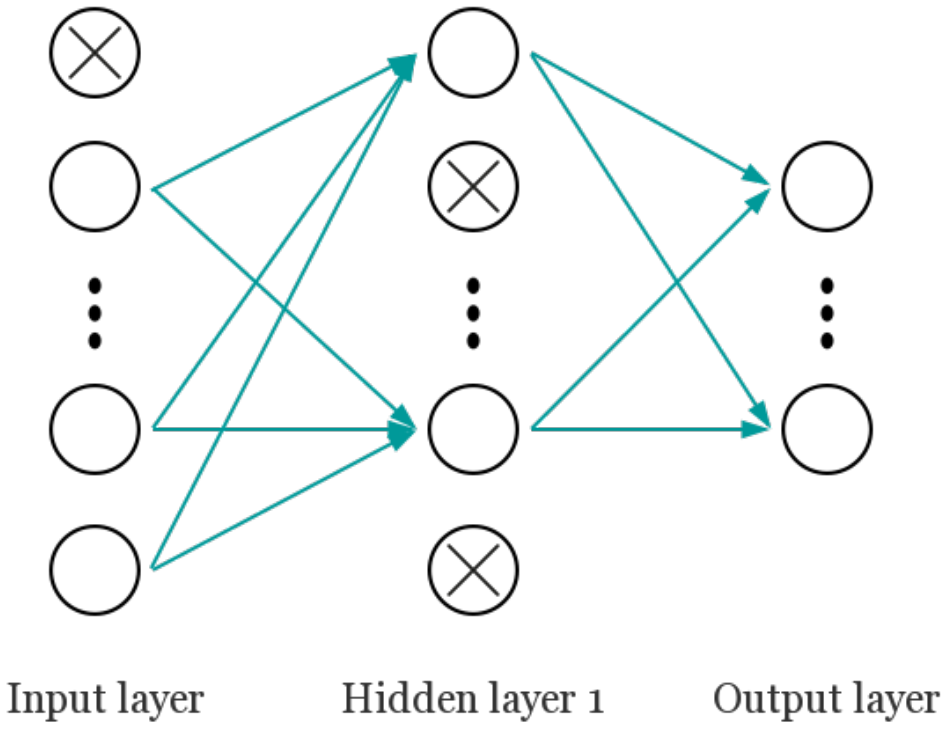

Figure 3.4: Illustration of FNN with dropout.

most of the variances. However, when the SNR drops to below 30 dB, variance of original data is almost evenly projected into PCs. Under such circumstances, PCA will not be efficient for dimension reduction. So, in a preliminary 10-fold test on the SNR=40 dB dataset, Hamming loss has higher means when number of PCs is less than the number of original pixels (blue line in Figure. 3.3). However, as shown in the same plot, when PCA is adopted in conjunction with dropout (blue markers), the Hamming loss is significantly reduced compared to the models that only adopts PCA (yellow markers) or dropout (red marker) or neither of them (purple marker). Therefore, in this thesis, we adopt PCA for all SNRs for Hamming loss reductions.

3.2 FNN-1

FNN has outstanding performance with large scale datasets [56]. As shown in Figure. 3.4, a typical FNN is formed by an input layer, an output layer and one hidden layer in-between. Each layer has a number of active neurons (circles without cross in Figure. 3.4) that use the neuron outputs from previous layer as input and produces output to the neurons in next layer. In our case of multi-label learning, a simple one hidden layer FNN model can achieve a state-of-the-art result with great computational efficiency [56]. To get output score $\mathbf{s}$ based on input feature set $\mathbf{x}$, our FNN for calculating output scores called FNN-1 can be written as [73]:

$$\begin{aligned} \mathbf{h} &= f_h(\mathbf{W}^{(1)}\mathbf{P}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) \\ \mathbf{s} &= f_s(\mathbf{W}^{(2)}\mathbf{P}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}) \end{aligned} \tag{3.2}$$

where $\mathbf{h}$ is a hidden layer that lies between input and output layer, f_h is the Rectified Linear Units (ReLU) activation function in hidden layer, f_s is the sigmoid function for output layer, and $\mathbf{W}^{(1)}$, $\mathbf{W}^{(2)}$, $\mathbf{b}^{(1)}$, $\mathbf{b}^{(2)}$ are the parameters that need to be trained from data. In our model, the loss function $f_L(\mathbf{s}, \mathbf{y})$ is defined as the cross entropy of label score $\mathbf{s}$ and classification target $\mathbf{y}$ which can be expressed as:

$$f_L(\mathbf{s}, \mathbf{y}) = -\sum_{i=1}^{L} y_i \log(s_i) + (1 - y_i) \log(1 - s_i) \tag{3.3}$$

where L is the number of labels.

In our model, we adopted dropout to mitigate overfitting [73]. Dropout is a widely used method for preventing overfitting problems in neural networks. It randomly drops out a percentage of neurons in training, and the weights of remaining neurons will be trained by back-propagation [73]. Retention probability $\boldsymbol{p} = (p_1, p_2)$ is the hyperparameter of dropout that will be tuned for our model. p_1 and p_2 are the probabilities of retaining units in input and the hidden layer of the neural network model. Retention probabilities set for the FNN-OT model are the ones that result in minimum losses. The dropout is activated by two diagonal matrices of Bernoulli random variables $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ with parameters p_1 and p_2. Both parameters are retention probabilities of input and hidden layer for dropout.

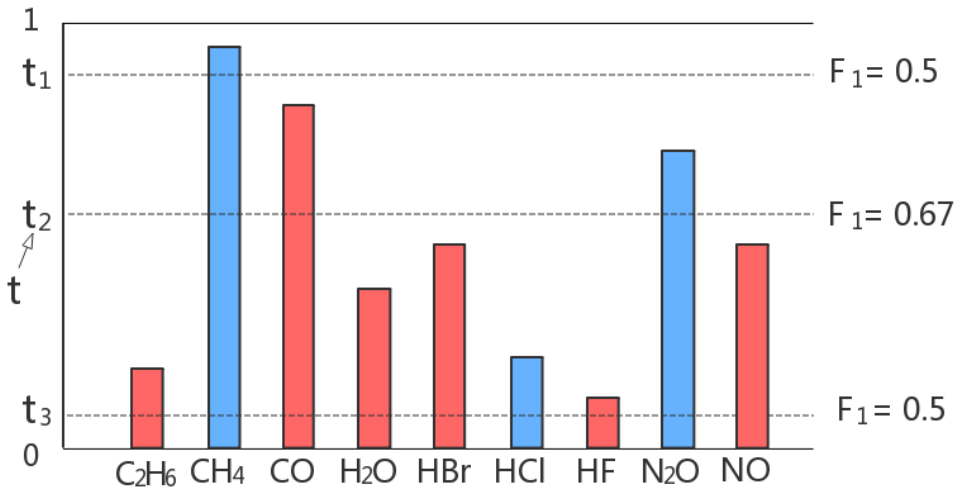

Figure 3.5: Illustration of optimal thresholding.

3.3 Optimal Thresholding

Once we obtain the output score $\mathbf{s}$ for a specific instance, we need to find a threshold t_i to convert i-th label score s_i in $\mathbf{s}$ to i-th label predictions $\hat{y}_i$ in $\hat{\mathbf{y}}$. Here, $\hat{y}_i$ can be expressed by an indicator function $\hat{y}_i = \mathbb{1}_{s_i > t_i}(s_i)$. That is, for the i-th specific gas component label that has a score higher than t_i, the prediction is 1 and 0 otherwise, representing the existence/non-existence of that gas component in the spectrum.

For binary classification problems in single-label learning, the sigmoid activation function of the output layer results in output scores that are between 0 and 1, and those output scores are often interpreted as probabilities of the two possible classes. For each sample in the testing set, its predicted class will be the one with more than 0.5 probability (output score), so the classifier can be viewed as an FNN model with a threshold $t = 0.5$. As shown in our result section, mislabelling of extremely low concentration of a specific gas species as absent from the sample occurs more frequently than mislabeling a non-existing gas species as existing in the sample. This results an imbalance between recall and precision. To re-balance recall and precision for higher F_1, adopting an optimal threshold t for each label in each instance is desirable.

Since we would like to balance precision and recall in the final output, maximizing

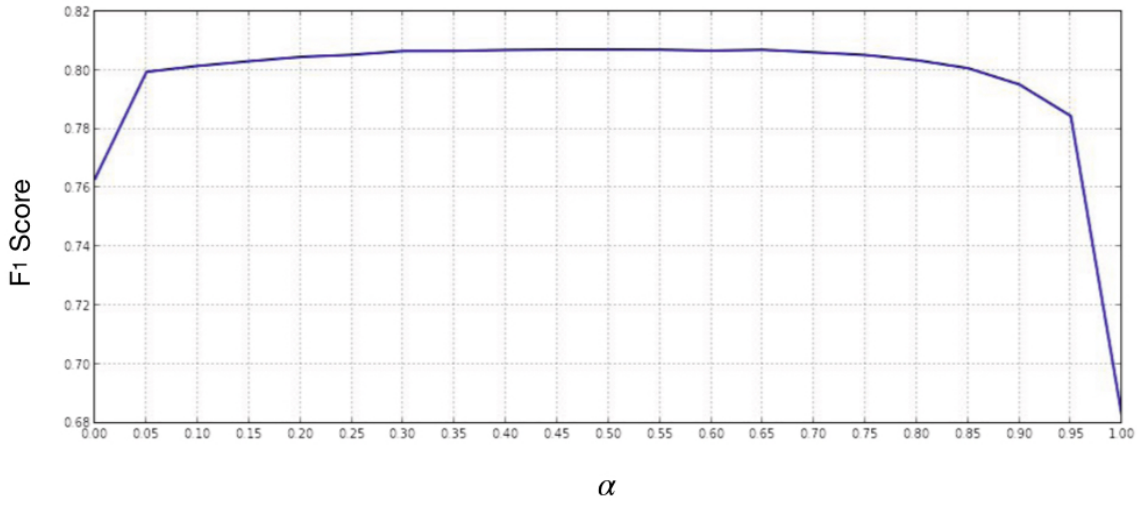

Figure 3.6: Mean F_1 for different values of combining the predicted upper and lower boundary. [77]

F_1 score will be the criteria of choosing threshold. For each sample, the function that maps threshold t to classification result F_1 score is not continuous. If the nine output scores are formed into an increasing order: $s_1 \leq s_2 \leq, ..., \leq s_9$, then the F_1 score with respect to t will jump at points s_1, s_2, ..., s_9, and remain constant within intervals $[0, s_1]$, $(s_1, s_2]$, $(s_2, s_3]$, ..., $(s_9, 1]$, because when t changes within one of the ten intervals, the classification results of nine gas labels will not change. So 10 candidates of t — t_1, t_2, ..., t_{10} — needs to be compared for the optimal t, where $t_1 \in [0, s_1]$, $t_2 \in (s_1, s_2]$, $t_3 \in (s_2, s_3]$, ..., $t_{10} \in (s_9, 1]$. For samples in the training set, the method of determining t is illustrated in Figure 3.5. Suppose we have obtained output scores for all nine labels of a gas mixture. Three of them (blue ones) have the ground truth value 1 (gas species exists in the sample), and the rest labels in red are 0 (gas species is absent in the sample). Then we calculate the F_1 scores for the three candidates t_1, t_2 and t_3 of t (dash lines), and the candidate with the highest F_1 score, which is t_2 in this example, is the t we need.

The remaining problem is to determine candidates of t in each of the 10 intervals. [77] have studied the relationship between threshold candidate and the upper/lower bond of its interval. Suppose the threshold candidate in interval $(l, u]$ is the linear combination of the lower-bond l and upper-bond u, then it can be expressed as $\alpha l + (1 - \alpha)u$ where $\alpha \in (0, 1]$. The relationship between α and F_1 score is illustrated in Figure 3.6. It shows that F_1 score reaches highest value when α is around 0.5, and it decreases drastically when α moves close to 0 or 1. So in our model, we set $\alpha = 0.5$,

and the threshold candidate t_1 (i=1,2,...,10) can be written as:

$$t_i = \begin{cases} 0 & i = 1 \\ 0.5s_{i-1} + 0.5s_i & i = 2, 3, ..., 9 \\ 1 & i = 10 \end{cases} \tag{3.4}$$

In order to systematically get thresholds for all instances in the testing set, we assume that threshold t is determined by the label scores $\mathbf{s}$, and their relationship can be recognized by the following FNN-2 model:

$$\begin{aligned} \mathbf{h}_t &= f_h(\mathbf{W}_t^{(1)}\mathbf{s} + \mathbf{b}_t^{(1)}) \\ \hat{t} &= \mathbf{W}_t^{(2)}\mathbf{h}_t + \mathbf{b}_t^{(2)} \end{aligned} \tag{3.5}$$

where $\mathbf{h}_t$ is a hidden layer with ReLU activation function f_h, and $\mathbf{W}_t^{(1)}$, $\mathbf{W}_t^{(2)}$, $\mathbf{b}_t^{(1)}$, $\mathbf{b}_t^{(2)}$ are the parameters that need to be estimated. We will use instances in the training set to train FNN-2 model, and the loss function is the mean squared error between t and $\hat{t}$.

Chapter 4

Evaluation and Comparison

4.1 Datasets

Datasets were synthesized based on single gas spectrums of C_2H_6, CH_4, CO, H_2O, HBr, HCl, HF, N_2O, and NO. They were selected from the HITRAN [65] database. Concentrations of gases are distributed between $0 - 10$ μM. Artificial Gaussian noises were added to each of the datasets with a pre-set SNR of 0 dB, 10 dB, 20 dB, 30 dB, 40 dB or 50 dB. For each SNR, three data sets were generated respectively to represent uncorrelated, positively and randomly correlated cases. In uncorrelated cases, nine gas labels are mutually independent. In positively correlated cases, nine gases are evenly divided into three subsets. Gas labels within the same subset are highly positively correlated, and labels from different subsets are independent. In randomly correlated cases, correlations between nine gas labels can be either positive or negative. In total, there will be $6 \times 3 = 18$ datasets used in later chapters. Details of generating those datasets are presented in [19].

Exploratory data analysis is an essential step before training and testing models on those datasets. By checking import attributes of the dataset, such as number of features, balance between classes and correlations between labels, we can not only rule out unsuitable models [23], but also determine necessary pre-processing steps for a certain model [28] to mitigate problems caused by unbalanced datasets. Moreover, data balance should be taken into consideration while choosing evaluation metrics for models. For datasets that are unbalanced between labels, it is highly possible that there will be significant variance of model performance between labels. So metrics with different average methods, such as micro and macro averaged metrics, should be

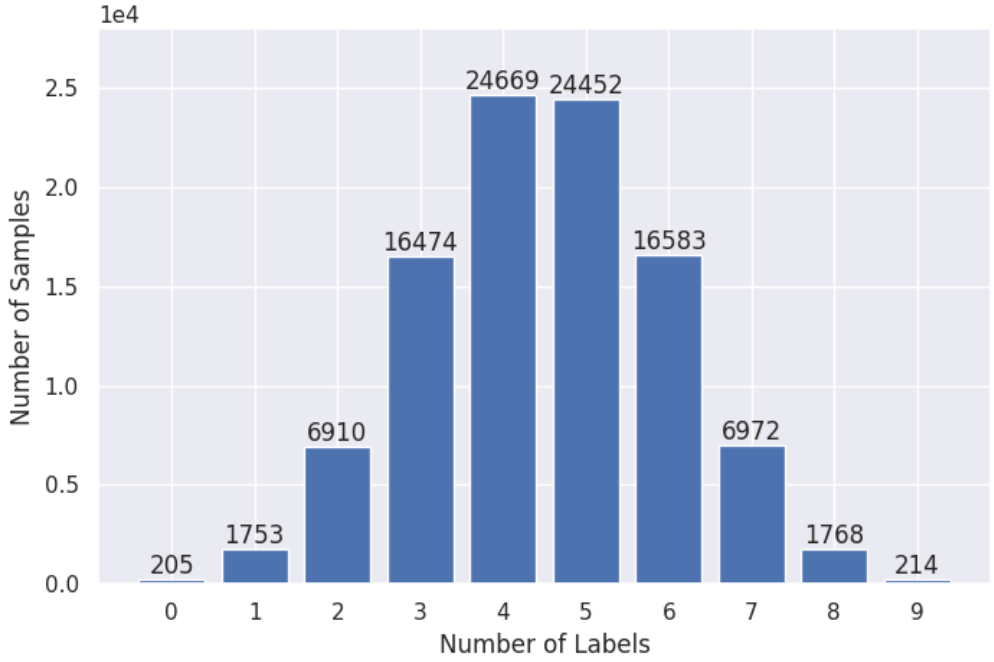

Figure 4.1: The (a) correlation between labels and (b) frequency of positive labels at 30 dB when gases are mutually independent.

	C_2H_6	CH_4	CO	H_2O	HBr	HCl	HF	N_2O	NO
P	50,117	49,922	49,948	50,010	50,081	50,237	49,833	50,165	49,990
N	49,883	50,078	40,052	49,990	49,919	49,763	50,167	49,835	50,010

Table 4.1: The numbers of positive and negative samples in each label at 30 dB when gases are mutually independent. "P" in the first column represents for number of positives, and "N" represents for number of negatives.

employed to make thorough and reliable evaluation and comparison.

In this section, we will check data balance, label frequency and label correlation for the uncorrelated, positively correlated and randomly correlated datasets at 30 dB as examples. Datasets at other SNRs have similar characteristics of gas labels.

4.1.1 Datasets with Mutually Independent Gas Labels

Table 4.1 presents the data balance by counting numbers of positives (blue bar) and negatives (orange bar) in each gas. Positives of a certain gas indicate its presence in gas mixtures, and negatives of a gas suggest that those samples do not contain such gas. Summation of positives with respect to different gases describes the layout of

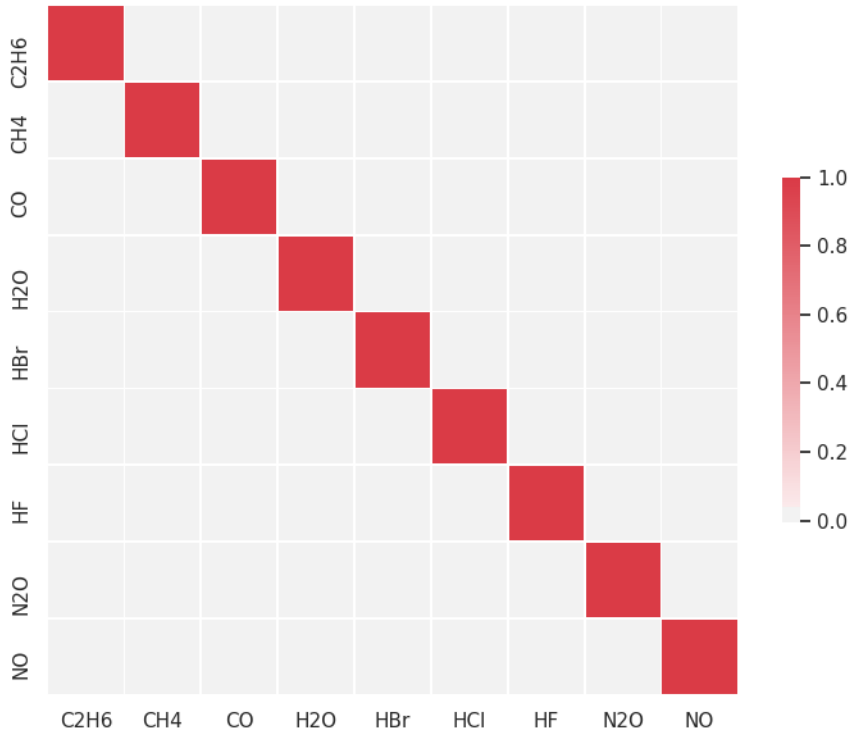

Figure 4.2: The correlation between labels at 30 dB when gases are mutually independent.

label balance in the datasets, and comparison between positives and negatives within a certain gas shows its class balance. In Table 4.1, Numbers of positives are all extremely close to 50,000 for all nine gases. It suggests that positives and negatives are well balanced between gas labels. For each of the nine gases, since the dataset has 100,000 samples, number of negatives is around 50,000 as well. So positives and negatives are well balanced between classes for all gases. Therefore, we do not need to apply any pre-processing method to balance the data.

Figure 4.1 shows the frequency of positive labels. The whole dataset is divided into ten groups, and all samples are assigned to one of the ten groups based on the number of target gases contained in them. From Figure 4.1 we can see that samples are distributed from 0 positive labels to 9 positive labels. Mean and median of this distribution are all between 4 and 5 labels. It proves the necessity of our multi-label design for the classification task, because the majority of samples have 2 or more

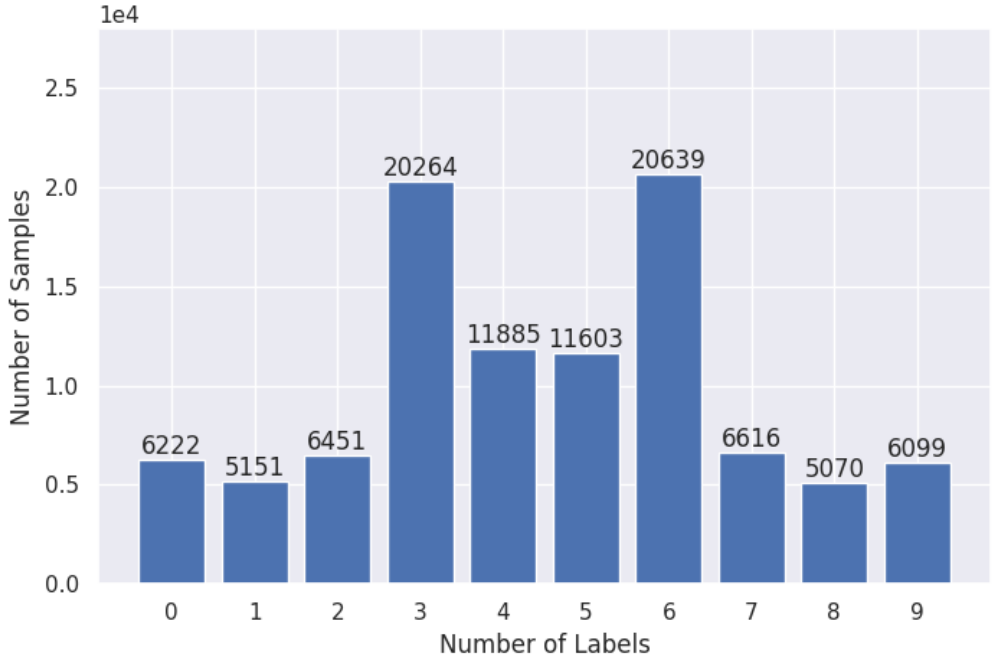

Figure 4.3: The frequency of positive labels at 30 dB when gases are positively correlated.

	C_2H_6	CH_4	CO	H_2O	HBr	HCl	HF	N_2O	NO
P	50,000	50,000	49,999	50,000	50,000	50,000	49,999	49,999	50,000
N	50,000	50,000	50,001	50,000	50,000	50,000	50,001	50,001	50,000

Table 4.2: The numbers of positive and negative samples in each label at 30 dB when gases are positively correlated. "P" in the first column represents for number of positives, and "N" represents for number of negatives.

positive gas labels.

Figure 4.2 is a heat map for the correlation matrix between nine gas labels. Diagonal elements in the correlation matrix are all 1, and the rest of the matrix is 0. It suggests that the synthesized dataset is in accordance of our setting that all gas labels are mutually independent.

4.1.2 Datasets with Positively Correlated Gas Labels

Similar to the dataset with uncorrelated gas labels, Table 4.2 presents the data balance by counting numbers of positives (blue bar) and negatives (orange bar) in each gas. In Table 4.2, Numbers of positives are all extremely close to 50,000 for all nine gases.

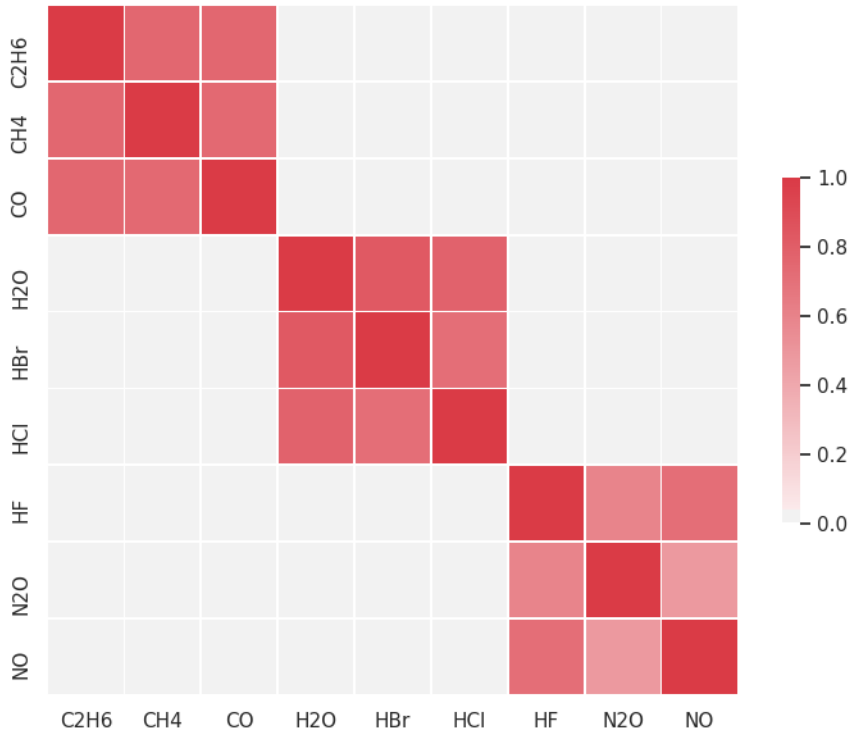

Figure 4.4: The correlation between labels at 30 dB when gases are positively correlated.

It suggests that positives and negatives are well balanced between gas labels. For each of the nine gases, since the dataset has 100,000 samples, number of negatives is around 50,000 as well. So positives and negatives are well balanced between classes for all gases. Therefore, we do not need to apply any pre-processing method to balance the data.

Figure 4.3 shows the frequency of positive labels. Same as the uncorrelated case, samples are distributed from 0 positive labels to 9 positive labels. Mean and median of this distribution are all between 4 and 5 labels. It proves the necessity of our multi-label design for the classification task, because the majority of samples have 2 or more positive gas labels.

Figure 4.4 is a heat map for the correlation matrix between nine gas labels. Diagonal elements in the correlation matrix are all 1, and the rest of the matrix is 0 or close to 1. It shows that the the nine gases are evenly divided into three subsets, and

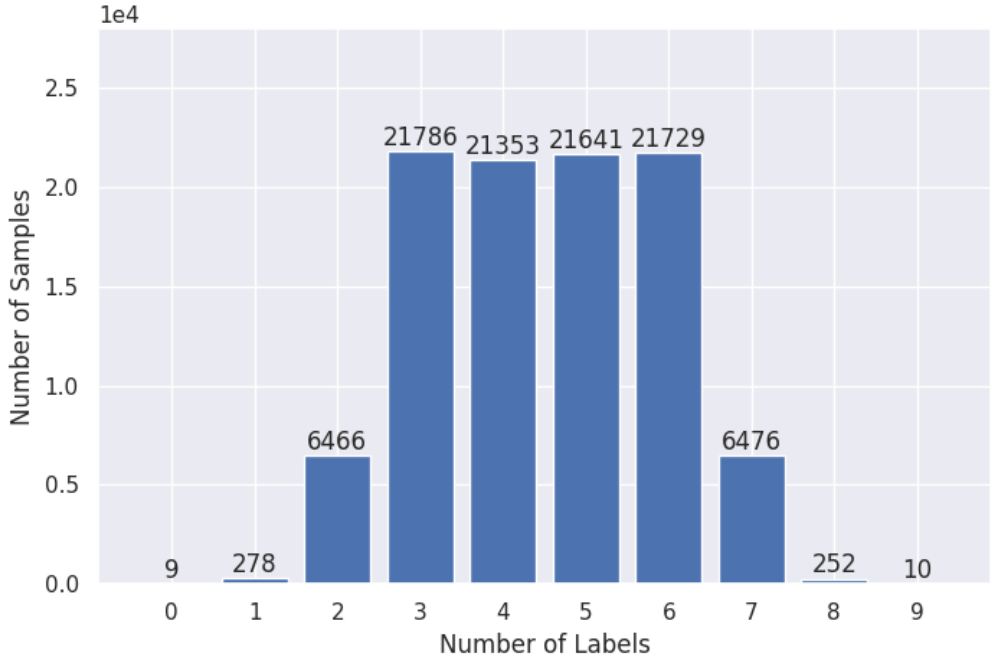

Figure 4.5: The frequency of positive labels at 30 dB when gases are randomly correlated.

	C_2H_6	CH_4	CO	H_2O	HBr	HCl	HF	N_2O	NO
P	50,000	50,000	49,999	49,998	50,000	49,999	50,000	50,001	50,000
N	50,000	50,000	50,001	50,002	50,000	50,001	50,000	49,999	50,000

Table 4.3: The numbers of positive and negative samples in each label at 30 dB when gases are randomly correlated. "P" in the first column represents for number of positives, and "N" represents for number of negatives.

within each subset, three gases are positively correlated with high coefficients.

4.1.3 Datasets with Randomly Correlated Gas Labels

Similar to the datasets above, Table 4.3 presents the data balance by counting numbers of positives (blue bar) and negatives (orange bar) in each gas. In Table 4.3, Numbers of positives are all extremely close to 50,000 for all nine gases. It suggests that positives and negatives are well balanced between gas labels. For each of the nine gases, since the dataset has 100,000 samples, number of negatives is around 50,000 as well. So positives and negatives are well balanced between classes for all gases. Therefore, we do not need to apply any pre-processing method to balance the data.

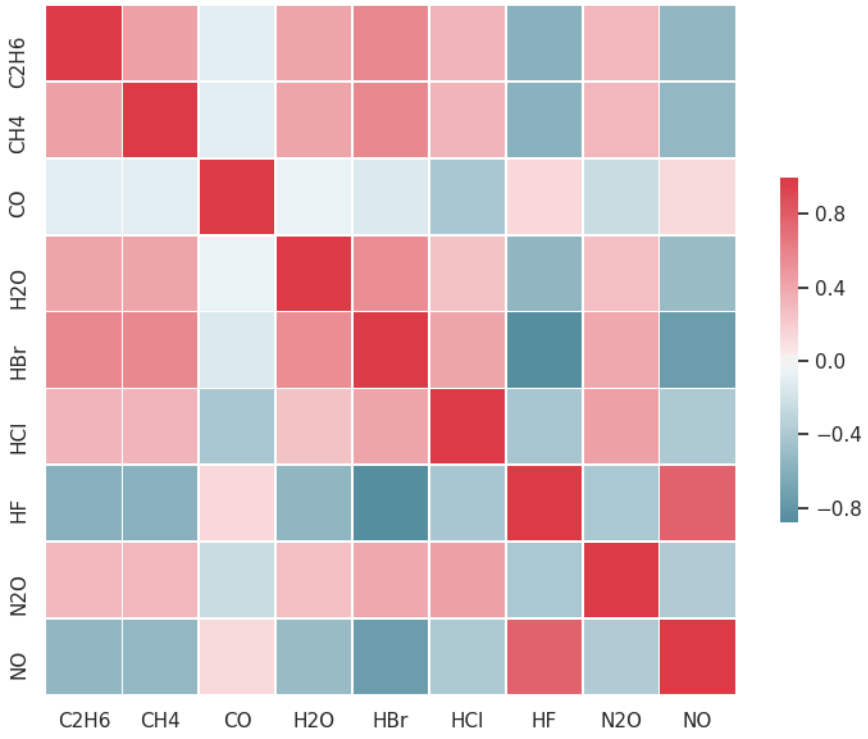

Figure 4.6: The correlation between labels at 30 dB when gases are randomly correlated.

Figure 4.5 shows the frequency of positive labels. Same as the uncorrelated case, samples are distributed from 0 positive labels to 9 positive labels. Mean and median of this distribution are all between 4 and 5 labels. It proves the necessity of our multi-label design for the classification task, because the majority of samples have 2 or more positive gas labels.

Figure 4.6 is a heat map for the correlation matrix between nine gas labels. Diagonal elements in the correlation matrix are all 1, and the rest of the matrix is distributed from -1 to 1. There is not a certain pattern for the correlations between gases. It can be negative, positive or zero.

4.2 Models for Comparison

4.2.1 PLS-BR

PLS-BR is a multi-label classifier adapted from PLS. It utilizes BR to split the multi-label task into several single-label classification problems. BR decomposes the learning of output labels into a set of binary classification tasks, one per label, where each single model is learned independently, using only the information of that particular label and ignoring the information of all other labels[49]. It has got various advantages such that the base learner can be selected from any of the binary learning methods, or the complexity is linear with the number of labels. Apart from this, BR also optimizes some loss functions. The main disadvantage of BR is that it assumes that all labels are independent thus ignores the correlations between them.

PLS is a widely used quantitative technique in advanced spectral analysis [50]. Unlike PCA by which only input features X are being analyzed, PLS is a multivariate regression method that correlates information of feature matrix x and output matrix Y. In order to make prediction of output Y out of feature X, PLS describes the common structure of X and Y by combining PCA and multivariate regression. [21] Similar to PCA, PLS decomposes $\boldsymbol{X}$ and $\boldsymbol{Y}$ by projecting them into a new set of coordinates called latent variables. However, unlike PCA, there are two matrices involved, and the goal of PLS is to reveal the relations between them. Latent variables which can optimally explain $\boldsymbol{X}$ in PCA are not necessarily relevant to $\boldsymbol{Y}$. So information from $\boldsymbol{Y}$ will be incorporated to the process of calculating latent variables so that covariance of $\boldsymbol{X}$ and $\boldsymbol{Y}$ will be maximized by latent variables.

PLS decomposes $\boldsymbol{X}$ and $\boldsymbol{Y}$ as follows:

$$\begin{aligned} \boldsymbol{X} &= \boldsymbol{T}\boldsymbol{P}^T \\ \boldsymbol{Y} &= \boldsymbol{U}\boldsymbol{Q}^T \end{aligned} \tag{4.1}$$

Where $\boldsymbol{T}$ and $\boldsymbol{U}$ are projections of $\boldsymbol{X}$ and $\boldsymbol{Y}$, $\boldsymbol{P}^T$ and $\boldsymbol{Q}^T$ are transpose of orthogonal loading matrices. Then regression of $\boldsymbol{T}$ and $\boldsymbol{U}$ will be performed following the standard multivariate regression procedure.

PLS itself is not designed for classification, so an extension of PLS called Partial Least Squares - Discriminant Analysis (PLS-DA) (Partial Least Squares - Discriminant Analysis) is adopted to classify categorical outputs. PLS-DA has been success-

fully used to classify milk and lubricant based on spectroscopic data sets in [16], [31], and [32]. In binary classification ($y = 0$ or 1) cases, PLS-DA creates two dummy variables $y_1(y = 0)$ and $y_2(y = 1)$ for the y label, and then calculates the PLS regression scores for y_1 and y_2. If y_1 has higher score, y is classified as 0. Otherwise the prediction class of y is 1.

4.2.2 FNN without OT

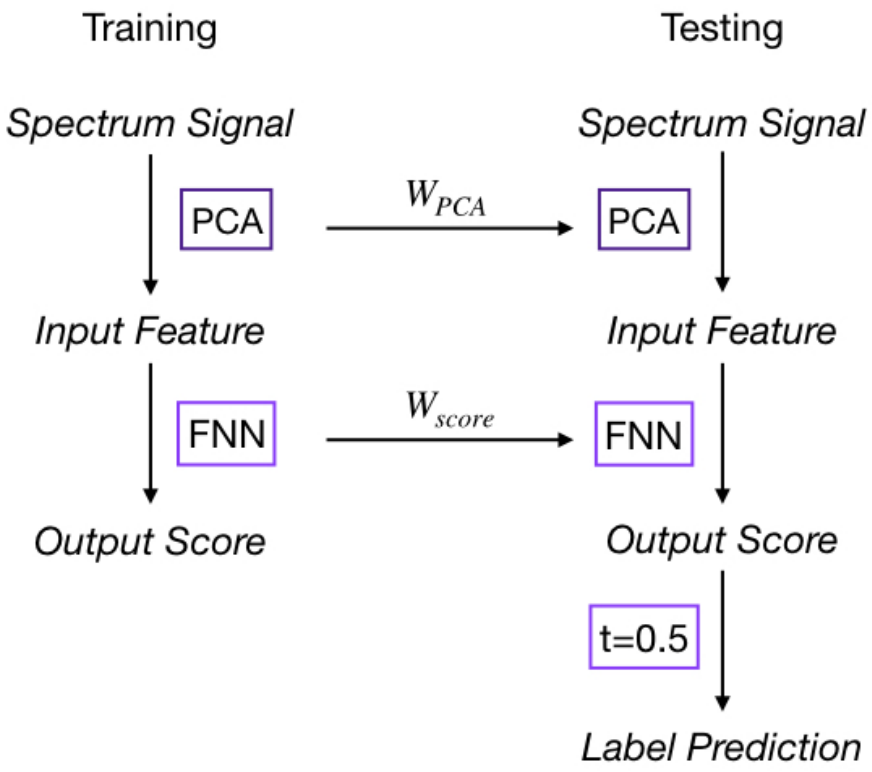

Figure 4.7: FNN training and testing procedure

FNN follows the design of FNN-OT in Section 3.2. It is equipped with all components of FNN-OT except for the OT system. FNN itself is capable of producing categorical predictions with a sigmoid or softmax activation function in the output layer. As stated in 3.3, for binary classification problems in single-label learning, the sigmoid activation function of the output layer results in output scores that are between 0 and 1, and those output scores are often interpreted as probabilities of the two possible classes. For each sample in the testing set, its predicted class will be the one with more than 0.5 probability (output score), so the classifier can be viewed

as an FNN model with a threshold $t = 0.5$. For binary classification problems in multi-label learning, since FNN can also produce multiple numerical outcomes at one time, it can easily solve multi-label classification problems with sigmoid activation function and 0.5 threshold.

The process of FNN is shown in figure 4.7. In FNN, Spectrum Signals are firstly pre-processed by PCA. The output principal components are the input features of an FNN model, which produces one output score for each gas. Instead of an OT system, an arbitrary threshold 0.5 is set for all the samples in training and testing sets. Predictions of each gas in one testing sample will be made by comparing its output score from FNN with the 0.5 threshold. If its output score is larger than the threshold, the gas will be predicted as present in the sample. Otherwise, the prediction is that such gas is not a composition in the sample.

4.3 Evaluation Metrics

To evaluate our models, we use the following metrics as our figures of merit [76]. Suppose there are n samples in the dataset, and each sample has l labels, then the mathematical definition of our evaluation metrics are as follows.

- Micro averaged precision

$$micro - precision = \frac{\sum_{j=1}^{l} TP_j}{\sum_{j=1}^{l} TP_j + \sum_{j=1}^{l} FP_j} \tag{4.2}$$

- Micro averaged recall

$$micro - recall = \frac{\sum_{j=1}^{l} TP_j}{\sum_{j=1}^{l} TP_j + \sum_{j=1}^{l} FN_j} \tag{4.3}$$

- Micro averaged F_1 score

$$micro - \mathrm{F}_1 = \frac{2\sum_{j=1}^{l} TP_j}{2\sum_{j=1}^{l} TP_j + \sum_{j=1}^{l} FN_j + \sum_{j=1}^{l} FP_j} \tag{4.4}$$

- Macro averaged precision

$$macro - precision = \frac{1}{l}\sum_{j=1}^{l} \frac{TP_j}{TP_j + FP_j} \tag{4.5}$$

- Macro averaged recall

$$macro - recall = \frac{1}{l}\sum_{j=1}^{l} \frac{TP_j}{TP_j + FN_j} \tag{4.6}$$

- Macro averaged F_1 score

$$macro - \mathrm{F}_1 = \frac{1}{l}\sum_{j=1}^{l} \frac{2TP_j}{2TP_j + FN_j + FP_j} \tag{4.7}$$

- Sample averaged precision

$$sample - precision = \frac{1}{n}\sum_{i=1}^{n} \frac{TP_i}{TP_i + FP_i} \tag{4.8}$$

- Sample averaged recall

$$sample - recall = \frac{1}{n}\sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i} \tag{4.9}$$

- Sample averaged F_1 score

$$sample - \mathrm{F}_1 = \frac{1}{n}\sum_{i=1}^{n} \frac{2TP_i}{2TP_i + FN_i + FP_i} \tag{4.10}$$

where TP_j, FN_j and FP_j are true positive, false negative, false positive of all

testing samples for label j. Respectively, TP_i, FN_i and FP_i are true positive, false negative, false positive of all gas labels for a testing sample i. In our context, true negative (TN) is the absence of a certain gas that has been correctly predicted in a sample. Similarly, true positive (TP) is the case that an existing gas is marked as present in a sample. False negative (FN) is the case that the classifier fails to identify an existing gas, and false positives (FP) is a false alarm where the classifier identifies a non-existence gas. Definitions and mathematical expressions of the 9 evaluation metrics are also introduced in Section 2.2.3.

Chapter 5

Results and Discussions

5.1 Hyper-parameter Tuning

ALL original datasets with 100,000 sample size are split into training sets of 80,000 samples and testing size of 20,000 samples. In the first step, hyper-parameters are tuned with the training datasets leaving testing datasets for final evaluation and comparison.

5.1.1 Latent Variables

The quantity of latent variables (LV) is an important hyper-parameter for PLS. A typical way of choosing optimal number is by calculating MSE of predictions for each LV [13]. 10-fold cross-validation tests are conducted on the 80,000-sample training sets of all 6 label-independent datasets. Results of MSE for $0 - 50$ dB datasets are presented in Figure 5.1. The quantity numbers of latent variables chosen are the ones with minimum or stably low mean squared error. They are listed in Table 5.1.

SNR (dB)	0	10	20	30	40	50
Number of LV	2	2	6	16	10	4

Table 5.1: Number of LVs for PLS

5.1.2 Optimizer

Optimizers Stochastic Gradient Descent(SGD), Adagrad and Adaptive Moment Estimation(Adam) are all popular tools to minimize prediction errors by moving pa-

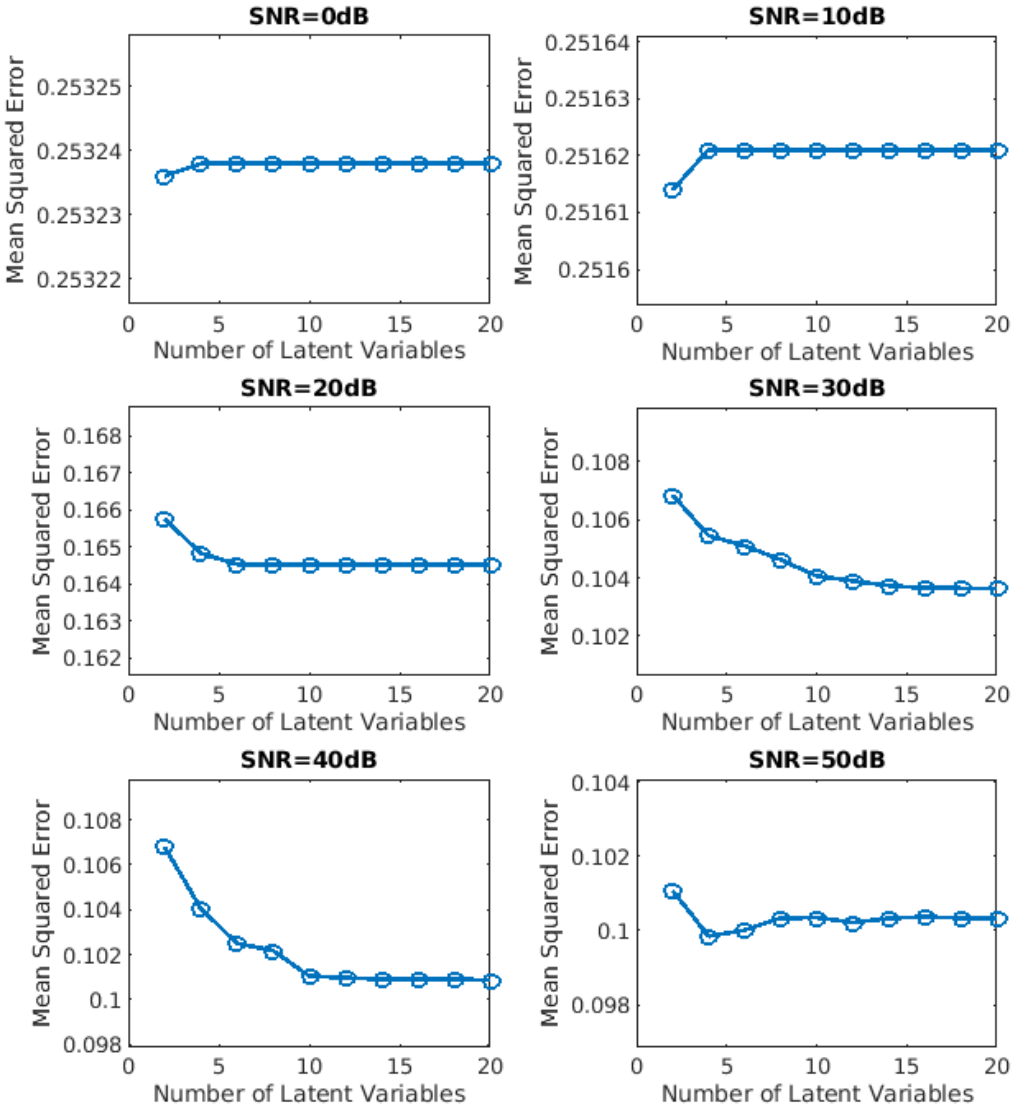

Figure 5.1: Mean squared errors of PLS with different number of latent variables.

rameters towards optimal. They update the model according to the output of loss functions. On mutually independent gas data at SNR=30 dB, we conduct a 10-fold cross validation test for these 3 optimizers. Results in Figure 5.2 shows that Adam optimizer works well and compares favorably to the other two optimization algorithms. It converge faster with lower loss. Upon investigation we found datasets with all SNRs and gas correlations demonstrated similar results under these 3 optimizers.

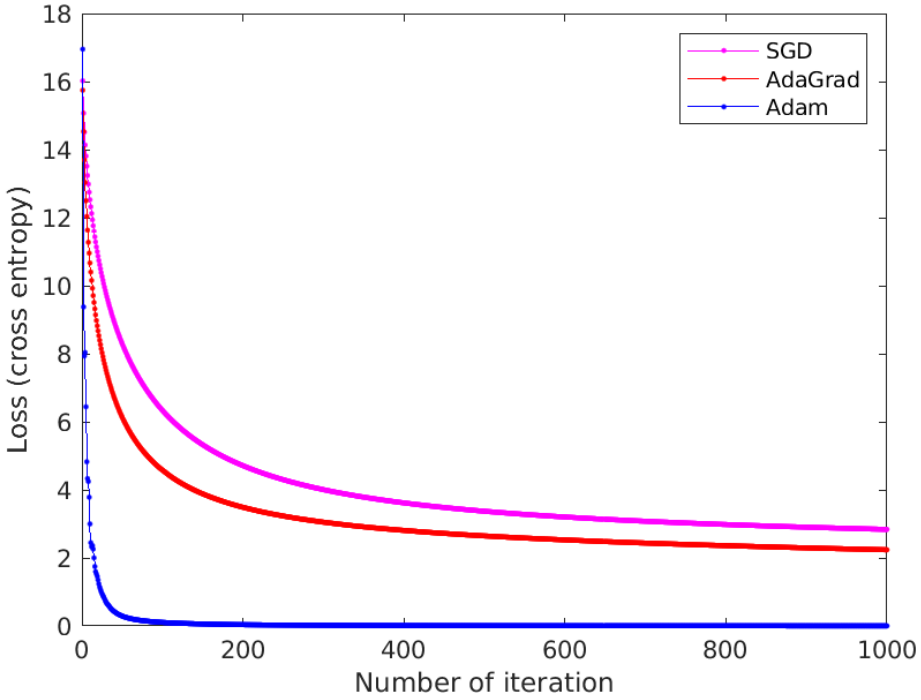

Figure 5.2: Cross entropy loss vs number of iterations (SNR=30db)

5.1.3 Training Sample Size

To determine the sufficient size of training sets for our models, we plotted the learning curve as shown in Fig. 5.3. Here, FNN-OT is compared with PLS-BR and FNN with 0.5 threshold.

As shown in the learning curves plot, we change the size of training sets while keeping the 20,000-sample testing set intact. Both training (solid lines) and testing (dashed lines) Hamming losses are plotted as a function of training sample sizes. At an SNR level of 30 dB, without dropout (blue markers), our FNN-OT model displays large variance and low bias as the training loss is almost 0 while the testing loss is above 0.1 even when the training sample size is around 100,000. This is a clear indication of overfitting for training samples fewer than 100,000. In contrast, by adopting dropout (red markers), the overfitting issue is solved and both training and testing loss converge to around 0.05 at around 100,000 training samples. In comparison, PLS-BR (green markers) does not display overfitting at the aforementioned sample size. However, the converged training and testing losses are higher (> 0.1) than our FNN-OT model with dropout, indicating our model outperforms this conventional

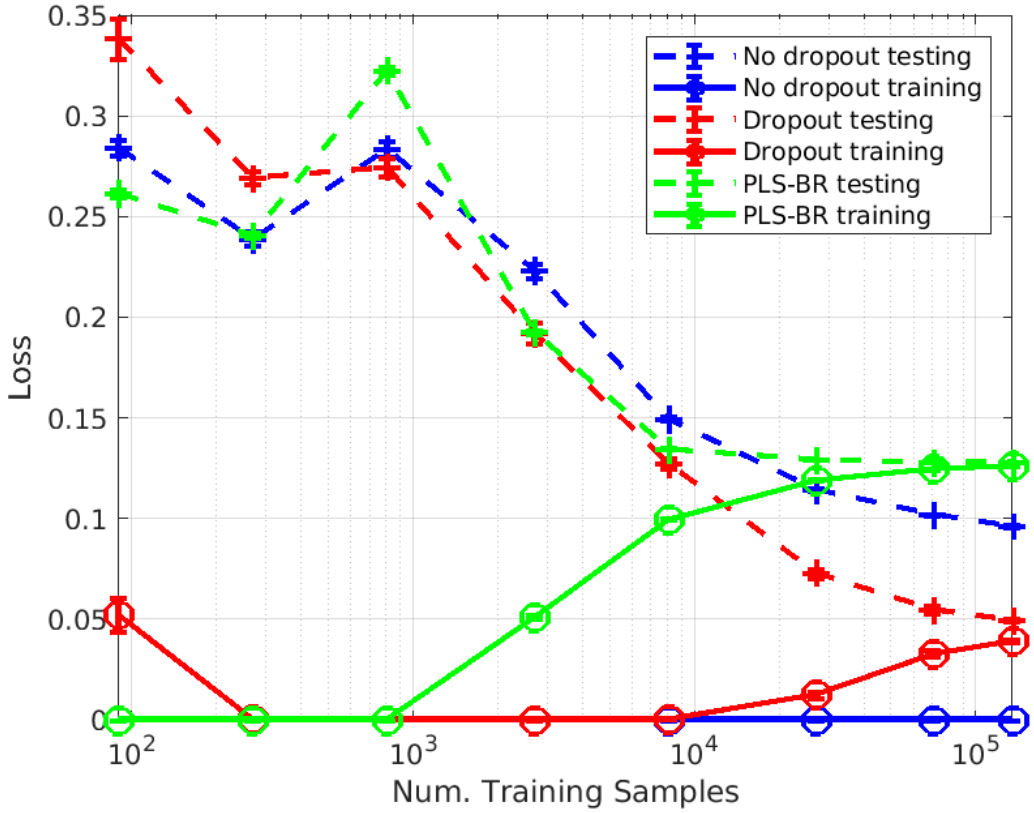

Figure 5.3: Learning curves of FNN-OT without dropout (blue), with dropout (red) and PLS-BR (green).

technique. Nevertheless, the plot clearly shows that it is sufficient to use around 100,000 samples to train our FNN-OT with dropout model.

5.1.4 Dropout

Learning curves in Figure 5.3 illustrate the necessity of regularization method such as dropout to overcome overfitting problems for FNN-OT. In order to tune the hyper-parameters for dropout, a grid search has been conducted on retention probabilities $\boldsymbol{p} = (p_1, p_2)$ of input and hidden layers. A typical choice of retention rate is 0.8 for input layer and 0.5 for hidden layer [73], and a preliminary search on our datasets shows that the optimal choice of $\boldsymbol{p}$ is around (0.95, 0.2). Figure 5.4 illustrates means of hamming losses of independent gas data with respect to different combination of p_1 and p_2. Positively and randomly correlated gas data has similar patterns of hamming losses with respect to retention probabilities. Table 5.2 lists retention probabilities

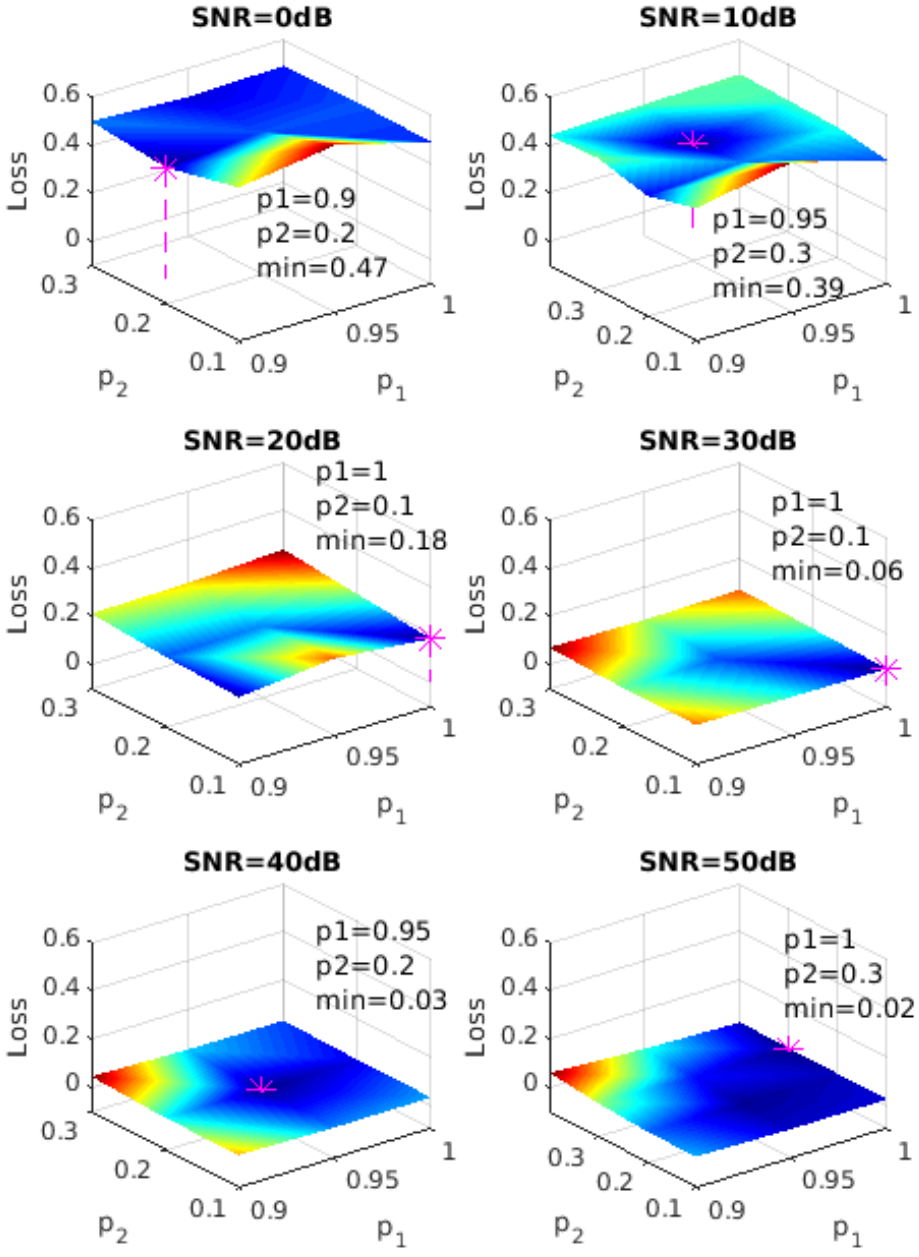

Figure 5.4: Hamming loss vs retention probability on mutually independent gas data that will be used in the Feed-forward neural networks.

SNR (dB)	Data 1		Data 2		Data 3	
	p_1	p_2	p_1	p_2	p_1	p_2
0	0.9	0.2	0.9	0.1	0.95	0.5
10	0.95	0.3	0.95	0.2	1	0.2
20	1	0.1	0.95	0.1	1	0.2
30	1	0.1	1	0.2	1	0.1
40	0.95	0.2	1	0.2	1	0.2
50	1	0.3	1	0.2	1	0.2

Table 5.2: Retention probabilities of input layer (p_1) and hidden layer (p_2). Data 1 represents mutually independent gas data. Data 2 represents positively correlated gas data. Data 3 represents randomly correlated gas data.

5.2 Performance Comparison of Mutually Independent Gas Data

We firstly compare our model using the datasets where all gas components are mutually independent.

Figure 5.5 illustrates the micro, macro and sample averaged precision at six different SNRs. Generally speaking, as expected, all models perform better at higher SNRs. When SNR is 0 dB, all three classifiers failed to identify gases because 0.5 precision is as good as random guess. Comparing the three sub-figures in Figure 5.5, we can observe that no significant difference exist between the three types of precision. Similarity of micro and macro average precision results from the label balance of datasets, which was stated in Section 4.1. For each type of precision, three models share similar performance for all SNRs. FNN-OT even slightly under-perform other two models at SNR=20 dB and 30 dB. Standard deviation of PLS-BR is always close to 0 for all SNRs. For FNN and FNN-OT, precision is less stable at SNR=20 dB

Figure 5.6 shows the micro, macro and sample averaged recall at six different SNRs. Same as precision above, all models perform better at higher SNRs. When SNR is 0 dB, all three classifiers failed to identify gases because 0.5 recall is as good as random guess. Between the three types of recall, no significant difference can be observed in any dataset. For each type of recall, FNN-OT significantly outperforms the other two models at all SNRs. Standard deviation of PLS-BR is always close to 0 for all SNRs. For FNN, recall is less stable when SNR is large ($\geqslant$ 20 dB), but for FNN-OT, recall is more stable for higher SNRs ($\geqslant$ 30 dB).

Figure 5.7 shows the micro, macro and sample averaged F_1 score at six different

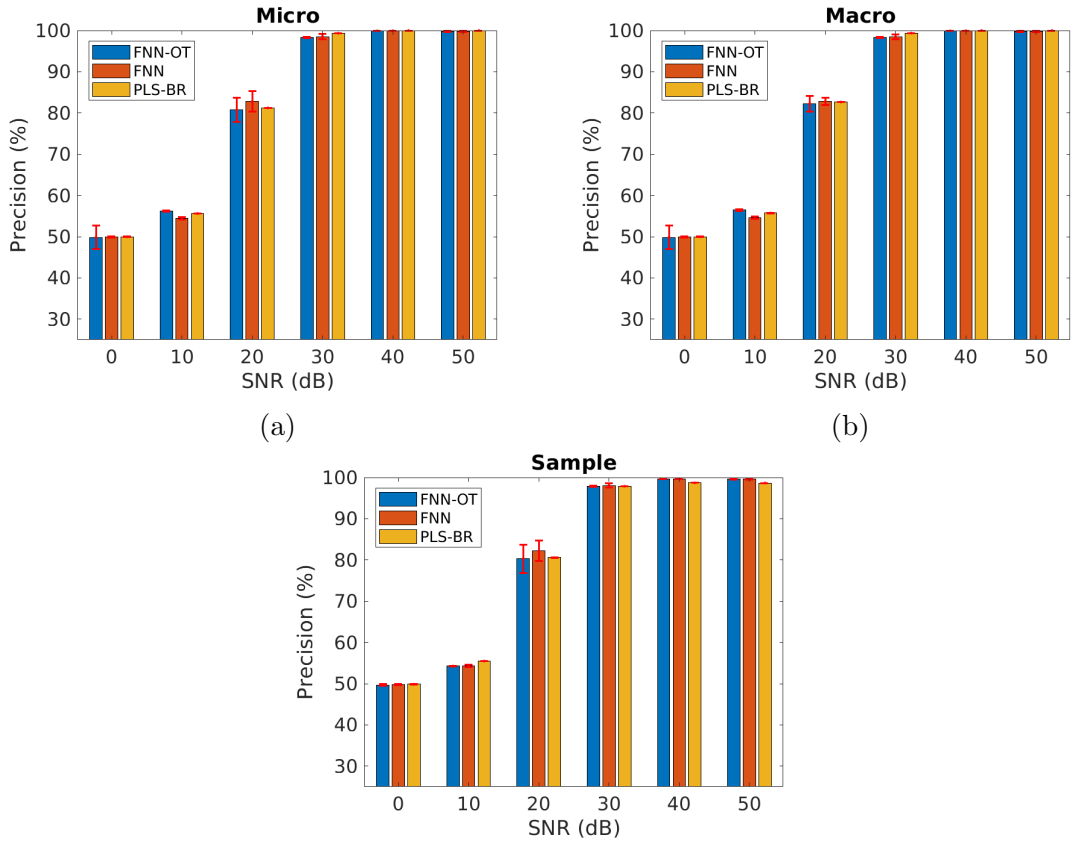

Figure 5.5: The (a) micro-precision, (b) macro-precision and (c) sample-precision of FNN-OT, FNN and PLS-BR when gases are mutually independent.

SNRs. Expected, all models perform better at higher SNRs. When SNR is 0 dB, all three classifiers failed to identify gases because 0.5 micro-F_1 score is as good as random guess. Across all SNRs, FNN-OT yields better micro, macro and sample averaged F_1 than FNN itself or the conventional PLS-BR, clearly indicating it a superior approach for gas identification. As for the standard deviation of three models, F_1 score share a similar pattern with recall, where stability occurs at low SNRs for FNN, high SNRs for FNN-OT, and all SNRs for PLS-BR.

Three figures above illustrate that all three models display higher values of precision than recall. This is due to the fact that most mislabelling occurs when a gas species' concentration is too low to produce detectable signal above noise background, all model will mistakenly predict the absence of that gas and produce a FN. However, as evident from Figure 5.6, selecting optimal threshold will significantly reduce the occurrence of FN and increase recall without significantly reducing the precision,

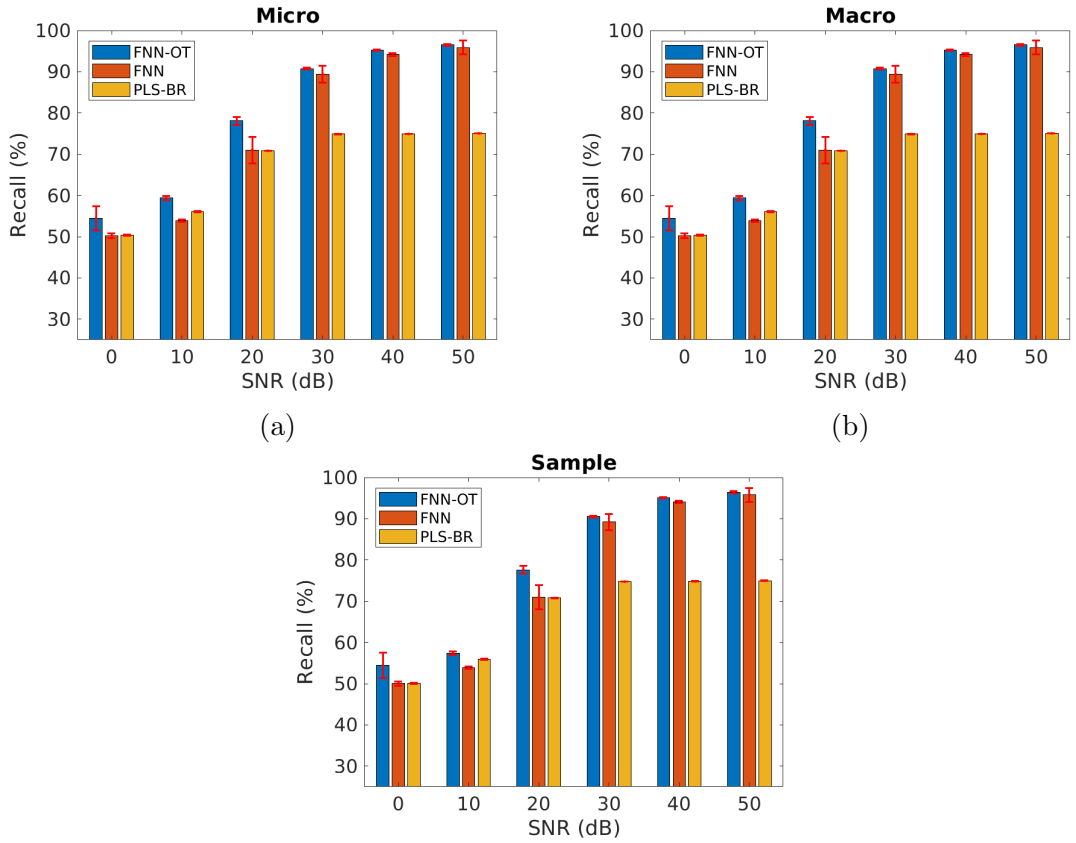

Figure 5.6: The (a) micro-recall, (b) macro-recall and (c) sample-recall of FNN-OT, FNN and PLS-BR when gases are mutually independent.

resulting a better F_1 score. This clearly justifies the necessity of adopt FNN-OT.

The advantages of FNN-OT are further confirmed by comparing minimum detectable concentrations (C_{min}) of nine gases. In original datasets, gas concentration is distributed in $[0\ \mu\text{M}, 10\ \mu\text{M}]$. If a gas has 0 μM concentration, it will be marked as 0 (absent). Otherwise it is marked as 1 (present). For a gas with a positive concentration, its spectrum signal has more similarity to that in the absent case when it has lower original concentration. So in general, classifiers have higher probability of making errors on cases with lower gas concentrations. F_1 score of a gas will increase with respect to its original concentration. In our context, C_{min} is the minimum concentration level that the F_1 score of a model reaches 0.8.

As shown in Figure 5.8, C_{min} for all models decrease with respect to SNR. Both FNN-OT and FNN consistently show lower minimum detectable concentration at all SNRs while in general FNN-OT outperforms FNN.

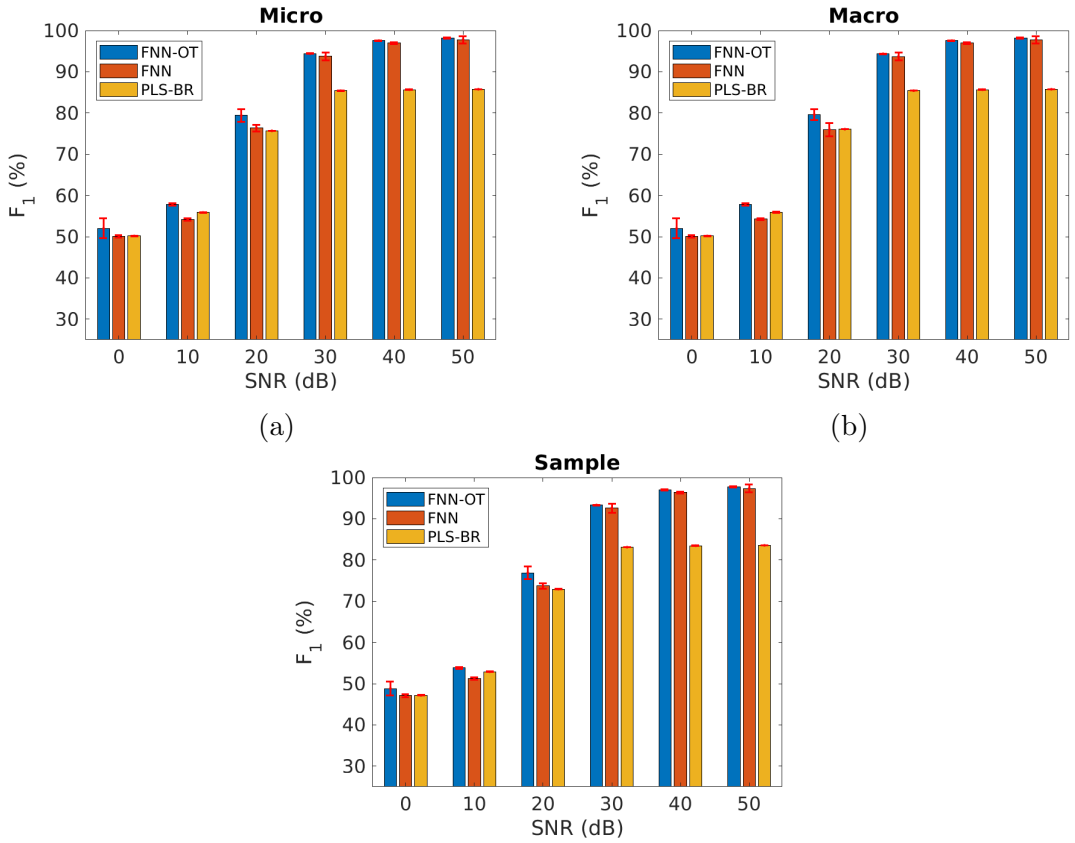

Figure 5.7: The (a) micro-F_1 score, (b) macro-F_1 score and (c) sample-F_1 score of FNN-OT, FNN and PLS-BR when gases are mutually independent.

5.3 Performance Comparison of Positively Correlated Gas Data

We further apply our models to the cases when the gases are positively correlated.

Figure 5.9 illustrates the micro, macro and sample averaged precision at six different SNRs. As expected, all models generally perform better at higher SNRs. Comparing the three sub-figures in Figure 5.9, we can observe that no significant difference exist between macro and micro averaged precision. Their similarity results from the label balance of datasets, which was stated in Section 4.1. Sample-precision of all three models, on the other hand, are significantly lower than micro or macro averaged precision.

For micro and macro averaged precision, when SNR is 0 dB, FNN and PLS-BR failed to identify gases because 0.5 precision is as good as random guess, but

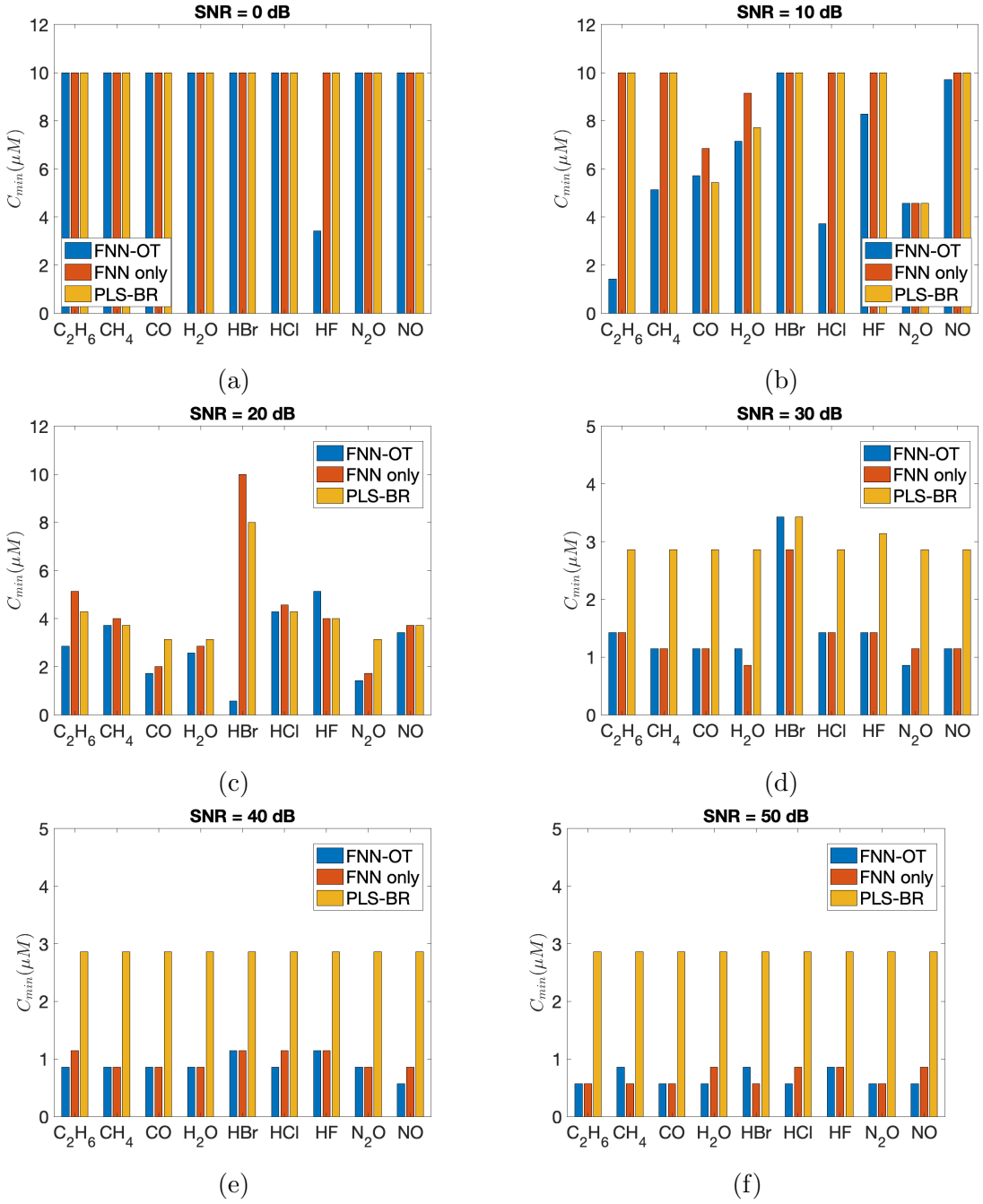

Figure 5.8: Minimum detectable concentration (C_{min}) of nine gases. [a]

[a]Calculation of C_{min} is done by Brosnan Yuen (email:brosnany@uvic.ca).

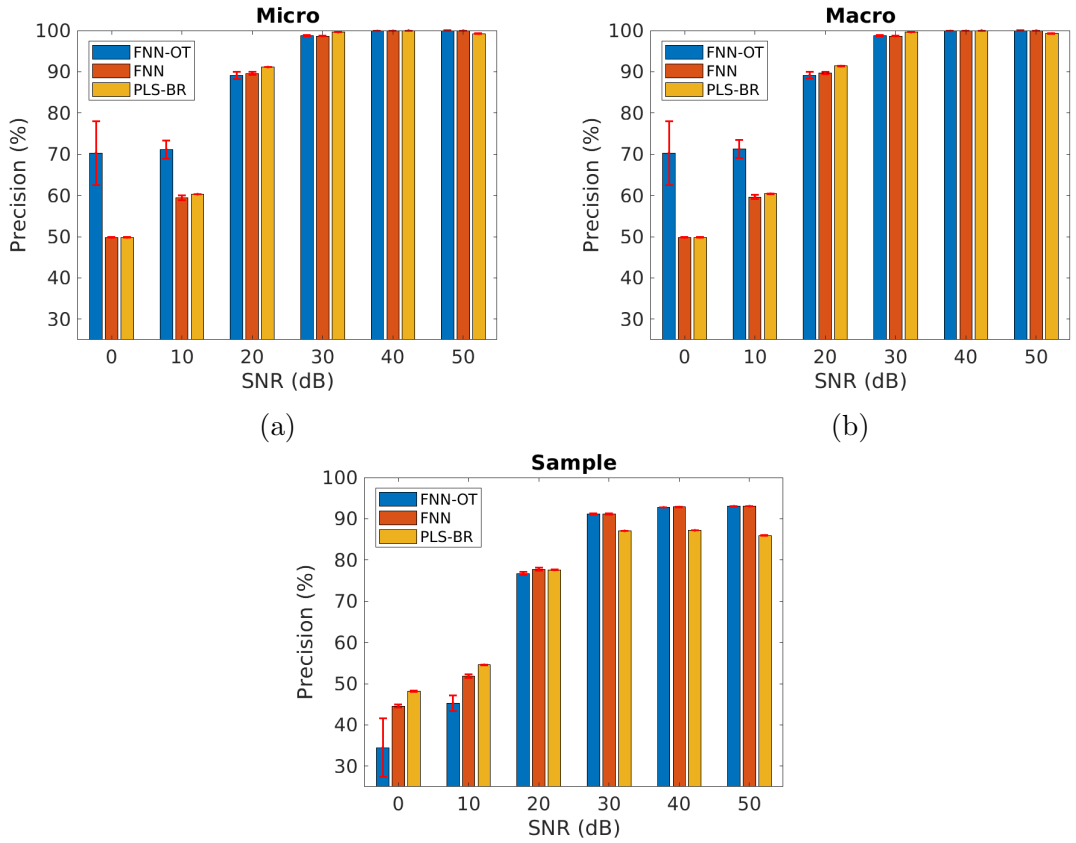

Figure 5.9: The (a) micro-precision, (b) macro-precision and (c) sample-precision of FNN-OT, FNN and PLS-BR when gases are positively correlated.

FNN-OT has much higher micro and macro averaged precision at SNR=0 dB and 10 dB. Standard deviation of FNN-OT is much higher than the other two models. When SNR is equal to or larger than 20 dB, three models share similar performance. Sample-precision, however, shows a completely different pattern. When SNR is equal to or less than 20 dB, FNN-OT is significantly underperform PLS-BR and FNN. Its sample-precision is below 0.5. When SNR is equal to or larger than 30 dB, FNN and FNN-OT have similar sample-precision, which is larger than that of PLS-BR.

Figure 5.10 shows the micro, macro and sample averaged recall at six different SNRs. Same as precision above, all models perform better at higher SNRs. No significant difference can be observed between macro and micro averaged recall, but sample-recall of all three models have different patterns. All models display lower values of sample averaged recall at all SNRs.

Micro and macro averaged recall of FNN-OT has much higher mean and standard

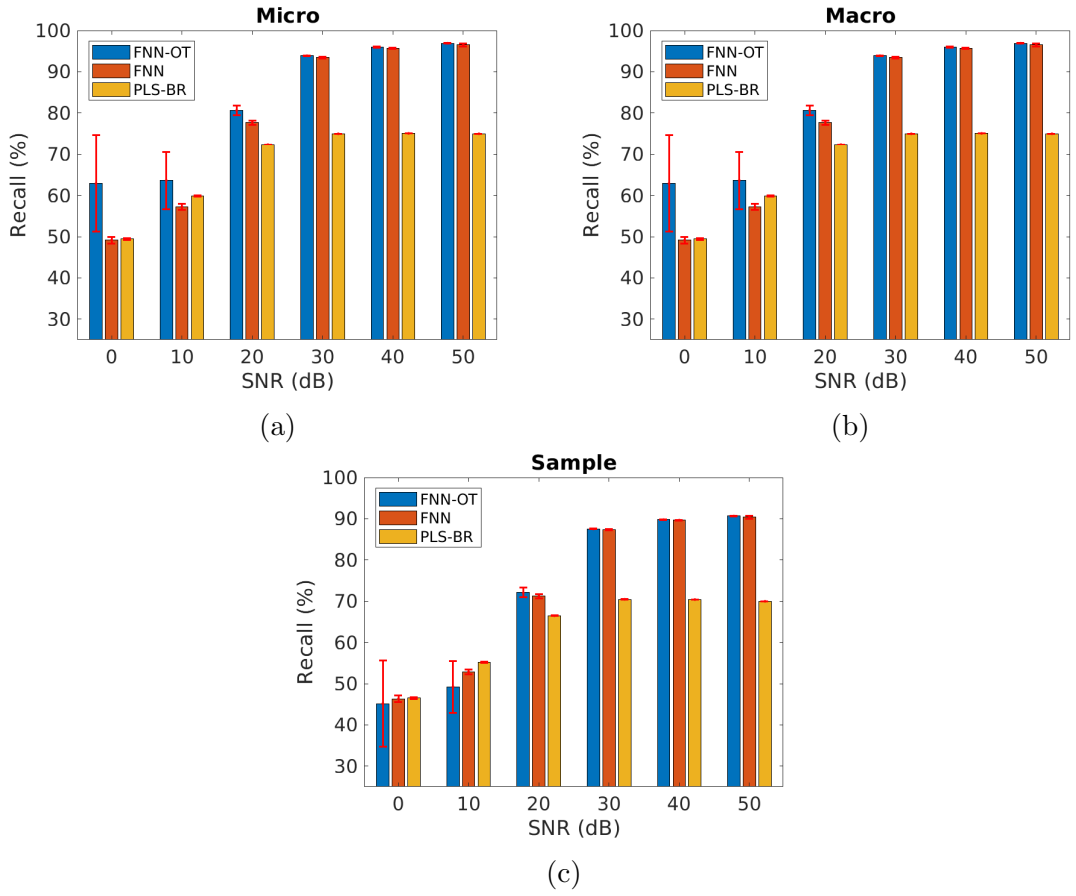

Figure 5.10: The (a) micro-recall, (b) macro-recall and (c) sample-recall of FNN-OT, FNN and PLS-BR when gases are randomly correlated.

deviation than that of FNN or PLS-BR at SNR=0 dB and 10 dB. When SNR is 20 dB and above, FNN display a closer value of micro and macro-recall to FNN-OT while recall of PLS-BR stay at a low value around 0.7. Standard deviation of all three models are close to 0 at higher SNRs ($\geqslant$ 30 dB). Mean values of Sample-recall for FNN-OT and FNN are similar, but FNN-OT is less stable with much higher standard deviation.

Figure 5.11 shows the micro, macro and sample averaged F_1 score at six different SNRs. As expected, all models perform better at higher SNRs. For each model at all SNRs, micro and macro averaged F_1 scores display similar value and pattern. Sample-F_1 of each model is lower than its micro or macro-averaged F_1 score.

For micro-F_1 and macro-F_1, across all SNRs, FNN-OT yields better performance than FNN itself or the conventional PLS-BR. As for the standard deviation of three models, F_1 score share a similar pattern with recall, where stability occurs at low

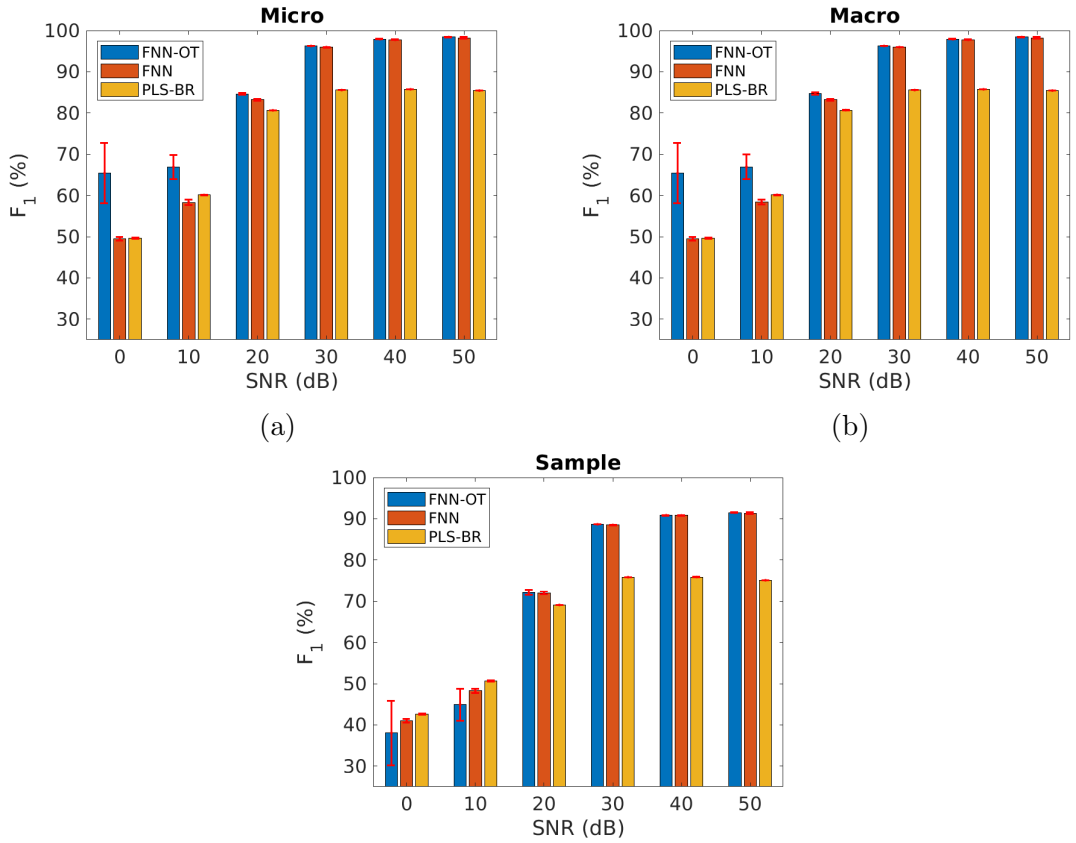

Figure 5.11: The (a) micro-F_1 score, (b) macro-F_1 score and (c) sample-F_1 score of FNN-OT, FNN and PLS-BR when gases are randomly correlated.

SNRs for FNN and PLS-BR, high SNRs for all models. For sample-F_1, FNN-OT display slightly lower mean value and much larger standard deviation when SNR is equal to or less than 20 dB. For higher SNRs, FNN-OT and FNN have similar performance of sample averaged F_1 score, and they both outperform PLS-BR.

The three figures above shows that unlike mutually independent gas data, evaluation results of three models are sensitive to average methods of evaluation metrics. Because of the data balance between all gas labels, micro and macro averaged metrics share similar patterns of performance. When SNR is above 20 dB, performance of the 3 models is similar to the uncorrelated case and FNN-OT outperforms. Further at SNR=0 dB or 10 dB, FNN-OT significantly outperforms the other 2 models and its own results of the uncorrelated case due to the fact that FNN-OT can collaboratively identify gas species through organize their correlation while FNN and PLS-BR are not capable of. Sample averaged results have a different pattern. For sample

averaged metrics, all three models display much lower performance at all SNRs. Low sample averaged precision suggests that FNN-OT, FNN and PLS-BR tend to make more errors at samples that a large portion of the nine target gases are predicted as present. Low sample averaged recall suggests that all three models have higher probability to mislabel gases at samples that contain many target gases. This confirms our intuition that models have more difficulty identifying a certain gas when many positively correlated gases occur. Comparing to FNN and PLS-BR, at low SNRs, FNN-OT has much higher values in micro/macro averaged metrics and much lower values in sample averaged metrics. It shows that FNN-OT is more capable of capturing information of positive label correlation from noisy datasets. This capability of FNN-OT has both advantages and disadvantages. On one hand, positive correlations between labels provide more information for FNN-OT to identify a certain gas. On the other hand, such information may mislead FNN-OT if many correlated gases are exist in one sample.

5.4 Performance Comparison of Randomly Correlated Gas Data

Last but not least, we compare our model using the datasets where all gas components are randomly correlated.

Figure 5.12 illustrates the micro, macro and sample averaged precision at six different SNRs. Same as other datsets, all models perform better at higher SNRs. When SNR is 0 dB, all three classifiers failed to identify gases because 0.5 precision is as good as random guess. Comparing the three sub-figures in Figure 5.12, we can observe that no significant difference exist between the three types of precision at all SNRs except for SNR=10 dB. Similarity of micro and macro average precision results from the label balance of datasets, and similarity of micro and sample average precision comes from similar performance of models on samples with different number of target gases. For each type of precision, three models share similar performance for all SNRs except for SNR=10 dB. FNN-OT even slightly under-perform other two models at SNR=20 dB. When SNR is 10 dB, sample-precision of FNN-OT is significantly lower than micro or macro averaged precision. It suggests that FNN-OT makes more mistakes on samples with many positively predicted gases at SNR=10 dB. Standard deviation of PLS-BR is always close to 0 for all SNRs.

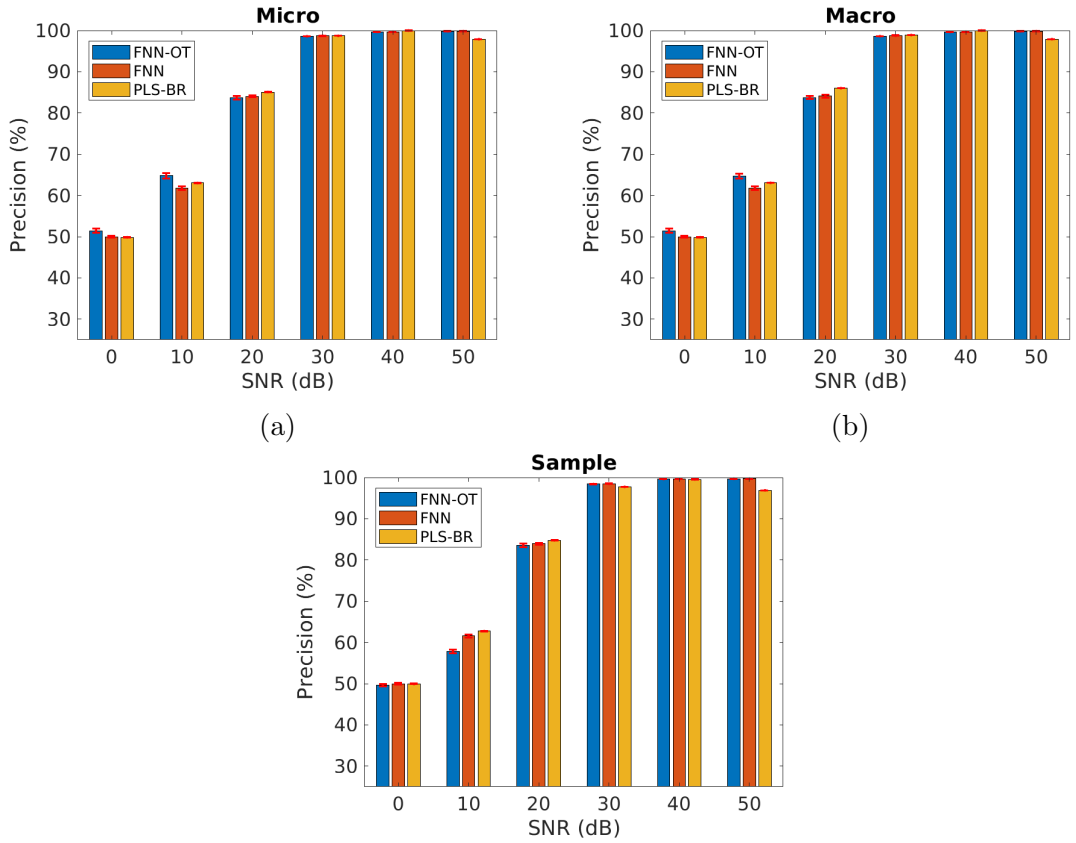

Figure 5.12: The (a) micro-precision, (b) macro-precision and (c) sample-precision of FNN-OT, FNN and PLS-BR when gases are randomly correlated.

Figure 5.13 shows the micro, macro and sample averaged recall at six different SNRs. Same as precision above, all models perform better at higher SNRs. When SNR is 0 dB, all three classifiers failed to identify gases because 0.5 recall is as good as random guess. Between the three types of recall, no significant difference can be observed in any dataset except for the one that SNR=10 dB. For each type of recall, FNN-OT outperforms the other two models at all SNRs except for the sample recall at SNR=10 dB. When SNR is 10 dB, sample-recall of FNN-OT is lower than micro or macro averaged precision. It suggests that FNN-OT makes more mistakes on samples that contain many target gases at SNR=10 dB. Standard deviation of PLS-BR is always close to 0 for all SNRs. For FNN, recall is less stable when SNR is large ($\geqslant$ 20 dB), but for FNN-OT, recall is more stable for higher SNRs ($=$ 30 dB).

Figure 5.14 shows the micro, macro and sample averaged F_1 score at six different SNRs. Same as other evaluation metrics, all models perform better at higher SNRs.

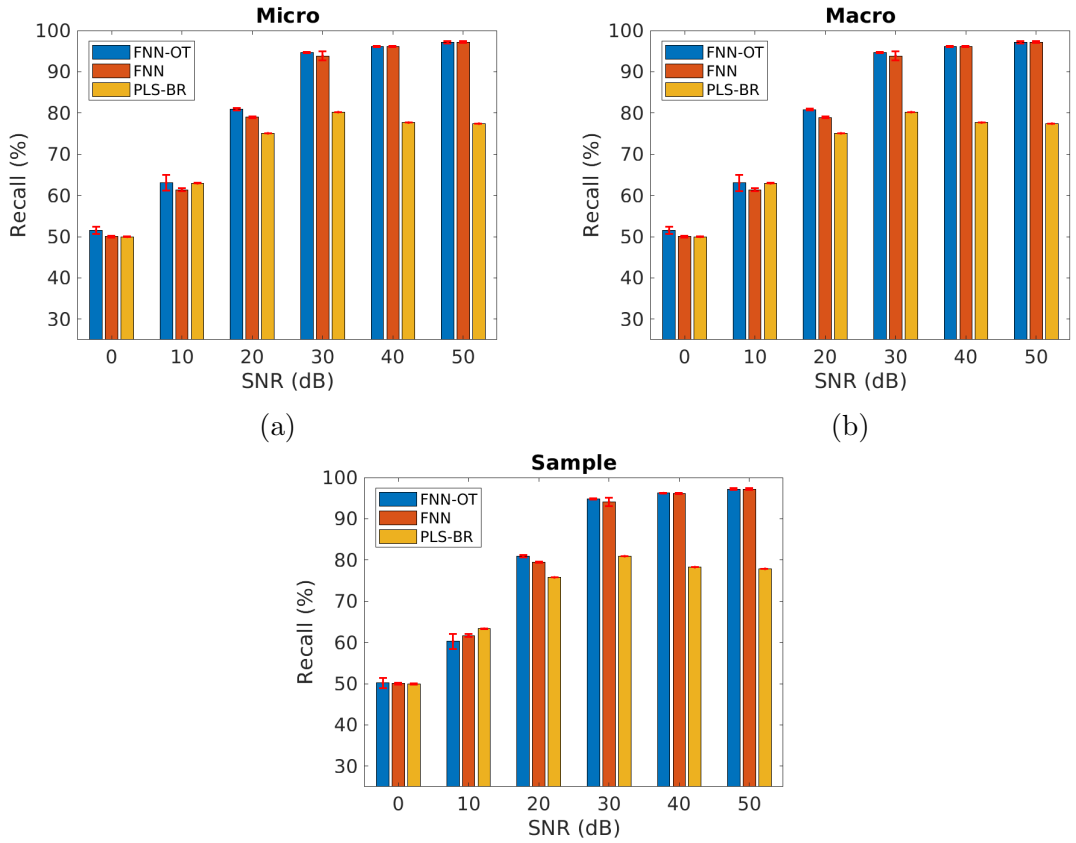

Figure 5.13: The (a) micro-recall, (b) macro-recall and (c) sample-recall of FNN-OT, FNN and PLS-BR when gases are randomly correlated.

When SNR is 0 dB, all three classifiers fail to identify gases because 0.5 micro-F_1 score is as good as random guess. Across all SNRs, FNN-OT yields better micro, macro and sample averaged F_1 than FNN itself or the conventional PLS-BR, clearly indicating it a superior approach for gas identification. As for the standard deviation of three models, F_1 score share a similar pattern with recall, where stability occurs at low SNRs for FNN, high SNRs for FNN-OT, and all SNRs for PLS-BR.

Three figures above illustrate that all three models display similar performance on uncorrelated and randomly correlated gas data. Unlike the results of datasets with positively correlated gas labels, FNN-OT only slightly outperforms PLS-BR and FNN for micro and macro averaged metrics at low SNRs. It suggests that compared to FNN and PLS-BR, FNN-OT can not extract much more useful information from label correlation if positive and negative correlations occur between gas labels.

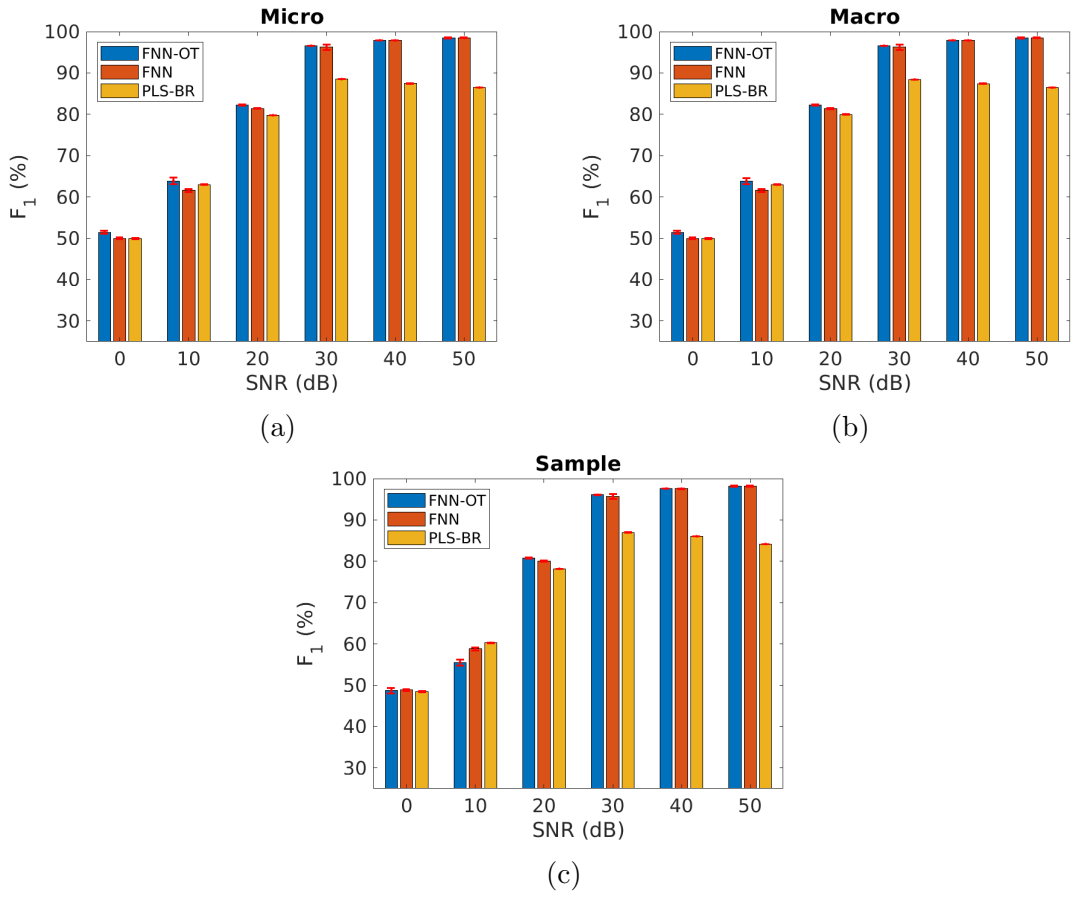

Figure 5.14: The (a) micro-F_1 score, (b) macro-F_1 score and (c) sample-F_1 score of FNN-OT, FNN and PLS-BR when gases are randomly correlated.

Chapter 6

Conclusion

In this chapter, we summarize our work and results, draw conclusions, and demonstrate our contributions and advantages. Furthermore, we will also discuss limitations of our methods, and present some future views about our research objectives.

6.1 Summary and contributions

FNN-OT is an adaptive FNN model for multi-label classficiation proposed to process absorption spectrum to detect target gas species from gas mixtures. It consists of three main parts: PCA, FNN and OT. PCA is a pre-processing step designed for spectroscopic datasets with highly correlated input features. FNN is a simple but effective model for producing output scores of target gases. OT is a decision making system for thresholds of all target gases in each sample. Output scores from FNN and thresholds from OT will determine the predictions of target gases.

Subsequently, FNN-OT is compared with other two multi-label classiers: FNN and PLS-BR. FNN itself is an adaptive method that can solve multi-label classification problems without any OT system. PLS-BR, built on PLS — a fundamental and popular tool that have been successfully applied to spectroscopic analysis, is a problem transformation method for multi-label learning. All three models are trained and tested on 18 synthesized datasets with 6 levels of SNR and 3 types of gas correlation. They are evaluated and compared with micro, macro and sample averaged precision, recall and F_1 score. For mutually independent and randomly correlated gas data, FNN-OT yields better performance than FNN without OT or the conventional PLS-BR, by significantly increasing its recall without much sacrifice of its precision. For

positively correlated gas data, FNN-OT is more capable of capturing information of positive label correlation from noisy datasets than the other two models.

In conclusion, by selecting optimal thresholds, FNN-OT outperforms conventional PLS-BR and FNN in two aspects. FNN-OT can dynamically select a threshold to reduce FN events. In addition, FNN-OT is capable of utilizing positive correlation among the components to enhance its classification capability. Both of these unique features make FNN-OT a favorable choice for spectroscopic analysis in cluttered environments.

6.2 Limitations and Future Work

During our research, we also found the following limitations of FNN-OT:

- In FNN-OT, FNN with only one hidden layer inside may not be complex enough to represent all possible relations of spectrum signals and gas predictions.
- OT system in FNN-OT is established on an assumption that all gases in one sample should have a uniform threshold. This assumption may not hold true in some real world practices.
- FNN-OT is evaluated and compared on synthesized datasets. It may not perform well on data gathered from experiments.

Those limitations point to some potential directions of future work. We propose these future steps, aiming to achieve a better performance on multi-composition spectroscopic analysis.

- Experiment on other NN models to replace PCA and FNN in FNN-OT. More complicated NN models, such as CNN, may provide more efficient and reliable ways to extract information from input spectroscopic data.
- Test OT systems with different assumptions of the threshold. For example, it can be assumed that thresholds of a certain gas in all sample mixtures are the same.
- Use input features of FNN as the input variables of OT. FNN-OT employs output scores of FNN as the input vectors of OT, but it contains much less information than the input features of the original dataset. So replacing output scores with original inputs may increase the performance of classifiers.

- Experiment on other models to determine thresholds in OT. Instead of a three-layer FNN, simpler models such as linear regression or more complex settings such as CNN or deep learning can be used for mapping output scores or original inputs to thresholds in OT.
- Replace OT with special designs of loss functions to balance precision and recall or represent label relations for better performance. For example, in the field of multi-task learning, weighted loss functions such as weighted cross entropy are used to represent the relations and hierarchy of tasks [42].
- Other problem transformation and algorithm adaption methods for multi-label learning can be implemented on the synthesized spectroscopic datasets for comparison.
- Explore potential methods to adapt FNN or other NN models for multi-label quantitative analysis of certain target gases in clustering environment.

Appendix A

Additional Information

A.1 Software

Most of the code is written in Matlab [51] and Python.

Machine Learning Python Libraries

- Tensorflow [1]
- keras [7]
- scikit-learn [60]

General Python Libraries

- numpy [57]
- pandas [53]
- matplotlib [36]
- seaborn [82]
- csv

Latex

- Overleaf

Bibliography

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016.

[2] James A Anderson. A simple neural network generating an interactive memory. *Mathematical biosciences*, 14(3-4):197–220, 1972.

[3] August Beer. Bestimmung der absorption des rothen lichts in farbigen flussigkeiten. *Ann. Physik*, 162:78–88, 1852.

[4] Adam Blum. *Neural networks in C++: an object-oriented framework for building connectionist systems*. John Wiley & Sons, Inc., 1992.

[5] Pierre Bouguer. *Essai d'optique sur la gradation de la lumière*. chez Claude Jombert, ruë S. Jacques, au coin de la ruë des Mathurins, à l . . . , 1729.

[6] Weizhu Chen, Jun Yan, Benyu Zhang, Zheng Chen, and Qiang Yang. Document transformation for multi-label feature selection in text categorization. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 451–456. IEEE, 2007.

[7] François Chollet et al. Keras. `https://keras.io`, 2015.

[8] Colin D Christy. Real-time measurement of soil attributes using on-the-go near infrared reflectance spectroscopy. *Computers and electronics in agriculture*, 61(1):10–19, 2008.

[9] Amanda Clare and Ross D King. Knowledge discovery in multi-label phenotype data. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 42–53. Springer, 2001.

[10] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[11] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[12] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

[13] Michael C Denham. Choosing the number of factors in partial least squares regression: estimating and minimizing the mean squared error of prediction. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 14(4):351–361, 2000.

[14] Daming Dong, Leizi Jiao, Chuanxia Li, and Chunjiang Zhao. Rapid and real-time analysis of volatile compounds released from food using infrared and laser spectroscopy. *TrAC Trends in Analytical Chemistry*, 2018.

[15] E Dupuit, A Dandrieux, P Kvapil, J Ollivier, G Dusserre, and O Thomas. Uv spectrophotometry for monitoring toxic gases. *Analusis*, 28(10):966–972, 2000.

[16] M Elbassbasi, F Kzaiber, G Ragno, and A Oussama. Classification of raw milk by infrared spectroscopy (ftir) and chemometric. *Journal of Scientific Speculations and Research*, 1(2):28–33, 2010.

[17] Joseph Fraunhofer. Bestimmung des brechungs-und des farbenzerstreungs-vermögens verschiedener glasarten, in bezug auf die vervollkommnung achromatischer fernröhre. *Annalen der Physik*, 56(7):264–313, 1817.

[18] Marcus Gallagher and Peter Deacon. Neural networks and the classification of mineralogical samples using x-ray spectra. In *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, volume 5, pages 2683–2687. IEEE, 2002.

[19] Luyun Gan, Brosnan Yuen, and Tao Lu. Multi-label Classification with Optimal Thresholding for Multi-composition Spectroscopic Analysis. *arXiv e-prints*, page arXiv:1906.10242, Jun 2019.

[20] Peter Gans. *Data fitting in the chemical sciences: by the method of least squares*. Wiley, 1992.

[21] Paul Geladi and Bruce R Kowalski. Partial least-squares regression: a tutorial. *Analytica chimica acta*, 185:1–17, 1986.

[22] Eva Gibaja and Sebastián Ventura. Multi-label learning: a review of the state of the art and ongoing research. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(6):411–444, 2014.

[23] Andrés Felipe Giraldo-Forero, Jorge Alberto Jaramillo-Garzón, José Francisco Ruiz-Muñoz, and César Germán Castellanos-Domínguez. Managing imbalanced data sets in multi-label problems: a case study with the smote algorithm. In *Iberoamerican Congress on Pattern Recognition*, pages 334–342. Springer, 2013.

[24] Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. *Advances in knowledge discovery and data mining*, pages 22–30, 2004.

[25] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013.

[26] Royston Goodacre. Explanatory analysis of spectroscopic data using machine learning of simple, interpretable rules. *Vibrational Spectroscopy*, 32(1):33–45, 2003.

[27] Stephen Grossberg. Adaptive pattern classification and universal recoding: I. parallel development and coding of neural feature detectors. *Biological cybernetics*, 23(3):121–134, 1976.

[28] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, (9):1263–1284, 2008.

[29] John B Hearnshaw. *The analysis of starlight: one hundred and fifty years of astronomical spectroscopy*. CUP Archive, 1990.

[30] Jeff Heaton. *Introduction to neural networks with Java*. Heaton Research, Inc., 2008.

[31] Aziz Hirri, Mahfoud Bassbasi, and Abdelkhalek Oussama. Classification and quality control of lubricating oils by infrared spectroscopy and chemometric. *Int J Adv Technol Eng Res*, 3:59–62, 2013.

[32] Aziz Hirri, Mahfould Bassbasi, Stefan Platikanov, Roma Tauler, and Abdelkhalek Oussama. Ftir spectroscopy and pls-da classification and prediction of four commercial grade virgin olive oils from morocco. *Food Analytical Methods*, 9(4):974–981, 2016.

[33] Steven M Holland. Principal components analysis (pca). *Department of Geology, University of Georgia, Athens, GA*, pages 30602–2501, 2008.

[34] J Michael Hollas. *Modern spectroscopy*. John Wiley & Sons, 2004.

[35] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[36] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[37] James D Ingle Jr and Stanley R Crouch. Spectrochemical analysis. 1988.

[38] Myles W Jackson. *Spectrum of belief: Joseph von Fraunhofer and the craft of precision optics*. Mit Press, 2000.

[39] Steven L Jacques. Optical properties of biological tissues: a review. *Physics in Medicine & Biology*, 58(11):R37, 2013.

[40] Francis A Jenkins and Harvey E White. *Fundamentals of optics*. Tata McGraw-Hill Education, 1937.

[41] Ioannis Katakis, Grigorios Tsoumakas, and Ioannis Vlahavas. Multilabel text classification for automated tag suggestion. *ECML PKDD discovery challenge*, 75, 2008.

[42] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, pages 7482–7491, 2018.

[43] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[44] Gustav Kirchhoff. Ueber das verhältniss zwischen dem emissionsvermögen und dem absorptionsvermögen der körper für wärme und licht. *Annalen der Physik*, 185(2):275–301, 1860.

[45] Teuvo Kohonen. Correlation matrix memories. *IEEE transactions on computers*, 100(4):353–359, 1972.

[46] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[47] Jean-Henri Lambert. *JH Lambert,... Photometria, sive de Mensura et gradibus luminis, colorum et umbrae*. sumptibus viduae E. Klett, 1760.

[48] Tom Lister, Philip A Wright, and Paul H Chappell. Optical properties of human skin. *Journal of biomedical optics*, 17(9):090901, 2012.

[49] Oscar Luaces, Jorge Díez, José Barranquero, Juan José del Coz, and Antonio Bahamonde. Binary relevance efficacy for multilabel classification. *Progress in Artificial Intelligence*, 1(4):303–313, 2012.

[50] Michael G Madden and Tom Howley. A machine learning application for classification of chemical spectra. In *Applications and Innovations in Intelligent Systems XVI*, pages 77–90. Springer, 2009.

[51] MATLAB. *version 9.6 (R2019a)*. The MathWorks Inc., Natick, Massachusetts, 2019.

[52] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[53] Wes McKinney et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.

[54] Marvin Minsky and Seymour A Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[55] Abhilash Alexander Miranda, Yann-Aël Le Borgne, and Gianluca Bontempi. New routes from minimal approximation error to principal components. *Neural Processing Letters*, 27(3):197–207, 2008.

[56] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. Large-scale multi-label text classification-revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer, 2014.

[57] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[58] Masatoshi Osawa. Dynamic processes in electrochemical reactions studied by surface-enhanced infrared absorption spectroscopy (seiras). *Bulletin of the Chemical Society of Japan*, 70(12):2861–2880, 1997.

[59] G Peach. Theory of the pressure broadening and shift of spectral lines. *Advances in Physics*, 30(3):367–474, 1981.

[60] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[61] J Pitha and R Norman Jones. A comparison of optimization methods for fitting curves to infrared band envelopes. *Canadian Journal of Chemistry*, 44(24):3031–3050, 1966.

[62] Jesse Read and Jaakko Hollmén. Multi-label classification using labels as hidden nodes. *arXiv preprint arXiv:1503.09022*, 2015.

[63] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning and Knowledge Discovery in Databases*, pages 254–269, 2009.

[64] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[65] Laurence S Rothman, Iouli E Gordon, Yury Babikov, Alain Barbe, D Chris Benner, Peter F Bernath, Manfred Birk, Luca Bizzocchi, Vincent Boudon, Linda R Brown, et al. The HITRAN 2012 Molecular Spectroscopic Database. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 130:4–50, 2013.

[66] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[67] David E Rumelhart. Parallel distributed processing: Explorations in the microstructure of cognition. *Learning internal representations by error propagation*, 1:318–362, 1986.

[68] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[69] Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal*, 3:211–229, 1959.

[70] Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168, 2000.

[71] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[72] Wilm Schumacher, Michael Kühnert, Petra Rösch, and Jürgen Popp. Identification and classification of organic and inorganic components of particulate matter via raman spectroscopy and chemometric approaches. *Journal of Raman Spectroscopy*, 42(3):383–392, 2011.

[73] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[74] Daniel Svozil, Vladimir Kvasnicka, and Jiri Pospichal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.

[75] Augusta Syty. Determination of sulfur dioxide by ultraviolet absorption spectrometry. *Analytical Chemistry*, 45(9):1744–1747, 1973.

[76] Grigorios Tsoumakas and Ioannis Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3), 2006.

[77] Grigorios Tsoumakas, Apostolos Papadopoulos, Weining Qian, Stavros Vologiannidis, Alexander D'yakonov, Antti Puurula, Jesse Read, Jan Švec, and Stanislav Semenov. Wise 2014 challenge: Multi-label classification of print media articles to topics. In *International Conference on Web Information Systems Engineering*, pages 541–548. Springer, 2014.

[78] Grigorios Tsoumakas and Ioannis Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *European conference on machine learning*, pages 406–417. Springer, 2007.

[79] Joseph von Fraunhofer. *Bestimmung des Brechungs-und Farbenzerstreuungs-Vermögens verschiedener Glasarten, in Bezug auf die Vervollkommnung achromatischer Fernröhre*. Franz, 1817.

[80] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.

[81] Yin Wang, Yubin Wei, Tongyu Liu, Tong Sun, and Kenneth T Grattan. TDLAS detection of propane/butane gas mixture by using reference gas absorption cells and partial least square approach. *IEEE Sensors Journal*, 18(20):8587–8596, 2018.

[82] Michael Waskom, Olga Botvinnik, Drew O'Kane, Paul Hobson, Joel Ostblom, Saulius Lukauskas, David C Gemperline, Tom Augspurger, Yaroslav Halchenko, John B. Cole, Jordi Warmenhoven, Julian de Ruiter, Cameron Pye, Stephan Hoyer, Jake Vanderplas, Santi Villalba, Gero Kunter, Eric Quintero, Pete Bachant, Marcel Martin, Kyle Meyer, Alistair Miles, Yoav Ram, Thomas Brunner, Tal Yarkoni, Mike Lee Williams, Constantine Evans, Clark Fitzgerald, Brian, and Adel Qalieh. mwaskom/seaborn: v0.9.0 (july 2018), July 2018.

[83] William Hyde Wollaston. Xii. a method of examining refractive and dispersive powers, by prismatic reflection. *Philosophical Transactions of the Royal Society of London*, (92):365–380, 1802.

[84] Guanghao Xu, Hyunjung Lee, Myoung-Wan Koo, and Jungyun Seo. Convolutional neural network using a threshold predictor for multi-label speech act classification. In *2017 IEEE international conference on big data and smart computing (BigComp)*, pages 126–130. IEEE, 2017.

[85] Yiming Yang. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1-2):69–90, 1999.

[86] Min-Ling Zhang and Zhi-Hua Zhou. A k-nearest neighbor based algorithm for multi-label classification. In *Granular Computing, 2005 IEEE International Conference on*, volume 2, pages 718–721. IEEE, 2005.

[87] Min-Ling Zhang and Zhi-Hua Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

[88] Qi-jun Zhang and Kuldip C Gupta. *Neural networks for RF and microwave design (Book+ Neuromodeler Disk)*. Artech House, Inc., 2000.