Machine Learning for Metabolite Identification with Mass Spectrometry Data

Kyoto University

NGUYEN Dai Hai

Dedication

This thesis is dedicated to my father NGUYEN Van-Dai, my sister NGUYEN Thi-Dung for their continuous support of my studies. I hope that this achievement will complete a part of the dream that you had for me all those many years ago when you to give me the best education you could.

NGUYEN Dai-Hai

Kyoto, Japan

Abstract

Metabolites are small molecules which are used in, or created by, chemical reactions occurring in living organism. They play important functions such as energy transport, signaling, building block of cells and inhibition/catalysis. Understanding biochemical characteristics (or identification) of metabolites is an essential part of metabolomics to enlarge the knowledge of biological systems. It is also key to the development of many applications and areas such as biotechnology, biomedicine or pharmaceutical sciences. However, this still remains a challenging task with a huge number of potentially interesting but unknown metabolites. Mass Spectrometry is a common analytical technique that measures the mass-to-charge ratio of ions converted from a portion of a chemical sample. The results are typically presented as a mass spectrum, a plot of intensity as a function of mass-to-charge ratio. Another way to represent a mass spectrum is as a list of peaks, each is defined by its mass-to-charge ratio and intensity value.

Identification of metabolites based on mass spectra can be regarded as a retrieval task: given a query spectrum of an unknown molecule, we aim to find molecules which have similar spectra from a reference database. A traditional approach is to compare the query against reference spectra in the database. The candidate molecules from the reference database are ranked based on the similarity between their reference spectra and the query, and the best matched candidates are returned. However, the reference databases often contain spectra of a small fraction of molecules in reality, leading to unreliable matching results if the molecule of query spectrum is not in the reference database. Consequently, to mitigate the insufficiency of such databases, alternative approaches for the task are devised. In this thesis, we explore computational methods for metabolite identification from spectra data with a focus on machine learning (ML), which has two stages: (i) mapping a spectrum to an intermediate representation (usually a molecular fingerprint, which is a binary vector to encode the presence of predetermined substructures or chemical properties in a molecule)

and (ii) retrieving candidate molecules from the reference database. The contributions of this thesis include: 1) we present a comprehensive survey on recent advances and prospects of computational methods for metabolite identification from mass spectra with an emphasis on ML approach; 2) we present SIMPLE, a method for predicting molecular fingerprints from spectra with ability to explicitly incorporate peak interactions and has interpretation, which are not addressed by the current cutting-edge methods for fingerprint prediction (stage (i)); 3) we present ADAPTIVE, a method for predicting chemical structures from spectra through learnable intermediate representations to overcome the drawbacks of molecular fingerprints: being very large to cover all possible substructures and redundant. We summarize each topic below in more detail.

In Chapter 1, we thoroughly survey computational methods for metabolite identification from mass spectra. The primary purpose of this survey is not only to summarize the proposed techniques in literature, but also to systematically organize them into groups according to their methodology and approaches. It would be beneficial for researchers to comprehend the key differences between techniques as well as rationale behind their groupings. We grouped computational techniques for the task into the following main categories: 1) spectra library; 2) *in silico* fragmentation and 3) ML. Given a query spectrum, spectra library is to compare it against a database of reference spectra of known molecules and rank the candidates based on their similarity to the query. In contrast, *in silico* fragmentation attempts to generate simulated spectra from the chemical structures in a compound database and then compare them with the query spectrum. ML is to predict intermediate representations between spectra and chemical structures of compounds and then use such representations for matching or retrieval. Our research focuses on developing ML models for predicting the intermediate representations with high accuracy and interpretation.

In Chapter 2, we present SIMPLE, a sparse learning based tool for fingerprint prediction. It takes a query spectrum of an unknown molecule as an input and predicts binary fingerprints as output, indicating which substructures or chemical properties are present in the molecule corresponding to the query spectrum. We then can use these predicted fingerprints to query candi-

date molecules with most similar fingerprints in the reference database. SIMPLE achieved around accuracy of 78.86%, which was comparable to the top-performance kernel based methods, which achieved around 76-80%, obtained by 10-fold cross validation on the MassBank dataset with 402 spectra. On the other hand, these kernel based methods needed around 1500 milliseconds, which is more than 300 times slower than that of SIMPLE, which required less than 5 milliseconds on the same dataset. This is a sizable difference when we process a huge amount of spectra produced by the current high-throughput mass spectrometry. One advantage of sparse learning models over kernel based methods is interpretation. SIMPLE clearly revealed individual peaks and peak interactions that contribute to enhancing the performance of predicting a particular fingerprint, shown by some case studies. In more technical detail, we formulate a sparse interaction model for spectra data. The model encourages sparsity over peaks and low-rankness over peak interactions while minimizing the classification errors for predicting the presence of fingerprints. The formulation of model is convex and guarantees global optimization, for which we develop an alternating direction method of multipliers algorithm.

In Chapter 3, we present ADAPTIVE, a tool for metabolite identification with learnable intermediate representations from given pairs of spectra and corresponding chemical structures of known molecules. It takes a spectrum of an unknown molecule as input and outputs a list of candidate compounds from the reference database. Instead of using fingerprints as in existing methods, ADAPTIVE could learn intermediate representations (called molecular vectors) between spectra and chemical structures of compounds. The benefits of learning molecular vectors are: 1) specific to both given data and task of metabolite identification and 2) more compact than molecular fingerprints, leading to a significant improvement in terms of both predictive performance and computational efficiency. ADAPTIVE with the molecular vector size of 300 achieved top-10 and -20 accuracies of 71.1% and 78.52%, which are 4% and 5% higher than those of the current best method, input output kernel regression (IOKR), respectively, obtained by 10-fold cross validation on a benchmark dataset with 4138 spectra. Furthermore, ADAPTIVE took 1000 milliseconds for retrieving one spectrum, while IOKR needed more than 3000 milliseconds on the same

dataset, meaning that ADAPTIVE was three times faster than IOKR. Technically, ADAPTIVE has two parts for learning two mappings: (i) from chemical structures to molecular vectors; (ii) from spectra to molecular vectors. The first part learns molecular vectors for molecular structures by maximizing the correlation between given spectra and molecular structures. The second part uses input output kernel regression, the current cutting-edge method for mapping spectra to molecular vectors obtained by the first part.

Acknowledgements

Undertaking this PhD study has been a truly life-changing decision for me. It would not have been accomplished without the support and guidance I received from many people and I would like to take this opportunity to show my gratefulness to them.

Firstly, I would like to express my sincere gratitude to my supervisor, professor Hiroshi Mamitsuka, for the continuous support of my PhD study and research activities, for his patience, encouragement, and of course immense knowledge. His excellent guidance helped me in overcoming the difficult time of research and writing of this thesis. I could not have though of having a better supervisor and mentor for my PhD study.

Besides my supervisor, I would like to thank my fellow labmates Dr. Canh Hao Nguyen, Dr. Kishan, Dr. Lu Sun and Mr. Duc Anh Nguyen (members of Mamitsuka Laboratory) for the scientific discussions on various machine learning topics, and also Dr. Ulrike, Dr. Chun-Yu Lin (former members of Akutsu laboratory) for all the fun we have had when attending conferences.

I would also like to thank Japan Society for the Promotion of Science (JSPS) for selecting me as an awardee of the DC2 scholarship. The PhD study could not have been possible without the prestigious funding from this award.

Last but not the least, I would like to thank my family members: my father Nguyen Van Dai, my sister Nguyen Thi Dung and my wife Luu Thi Nhu for their spiritual support during my PhD study and my life in general. A very special acknowledgement goes into my little girl Hai An who has changed the goal of my life and has given me an enormous happiness. Taking care of her and writing this thesis simultaneously at the time of global pandemic was truly the hardest but a great experience for me.

Publication notes

Chapter 1 is based on a review paper on computational methods for metabolite identification from mass spectra, which appeared in the journal Briefings in Bioinformatics [Nguyen et al., 2019b]. Chapter 2 is based on a paper on machine learning models with the incorporation of peak interactions for predicting molecular fingerprints, which appeared in the journal Bioinformatics (also the proceedings of the 26th International Conference on Intelligent Systems for Molecular Biology (ISMB 2018)) [Nguyen et al., 2018]. Chapter 3 is based on a paper on another machine learning model for predicting representations for metabolites from their chemical structures and spectra, which appeared in the journal Bioinformatics (also the proceedings of the 27th International Conference on Intelligent Systems for Molecular Biology (ISMB 2019)) [Nguyen et al., 2019a]. All together, the content of this thesis is based on the all three scientific publications in the following publication list:

List of publications by the author

1. Dai Hai Nguyen, Canh Hao Nguyen, Hiroshi Mamitsuka, "Recent advances and prospects of computational methods for metabolite identification: a review with emphasis on machine learning approaches", Briefings in Bioinformatics, bby066, 2019.

2. Dai Hai Nguyen, Canh Hao Nguyen, Hiroshi Mamitsuka, "ADAPTIVE: leArning DAta-dePendenT, concIse molecular VEctors for fast, accurate metabolite identification from tandem mass spectra", Bioinformatics, (Proceedings of the 27th International Conference on Intelligent Systems for Molecular Biology (ISMB/ECCB 2019), Basel, Switzerland), Pages i164–i172.

3. Dai Hai Nguyen, Canh Hao Nguyen, Hiroshi Mamitsuka, "SIMPLE: Sparse Interaction Model over Peaks of MoLEcules for Fast, Interpretable Metabolite Identification from Tandem Mass Spectra". Bioinformatics, 34 (13) (Proceedings of the 26th International Conference on Intelligent Systems for Molecular Biology (ISMB 2018), Chicago, USA), Pages i323–i332

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1. Motivation

Metabolomics involves studies of a great number of metabolites, which are small molecules present in biological systems. They play a lot of important functions such as energy transport, signaling, building block of cells and inhibition/catalysis. Understanding biochemical characteristics (or identification) of the metabolites is an essential and significant part of metabolomics to enlarge the knowledge of biological systems. It is also the key to the development of many applications and areas such as biotechnology, biomedicine or pharmaceuticals. However, the identification of metabolites remains a challenging task in metabolomics with a huge number of potentially interesting but unknown metabolites. The standard method for identifying metabolites is based on the mass spectrometry (MS) preceded by a separation technique. The output of the mass spectrometer, given a sample (of molecules or metabolites), is a mass spectrum, which is simply the m/z ratios of the ions present in a sample plotted against their intensities. Another way to represent a mass spectrum is as a list of peaks. Each peak shows a component of unique m/z in the sample, and its height implies the relative abundance of the various components in the sample. An illustration of an example mass spectrum is shown in Figure 1.1.

Identification of metabolites can be regarded as a retrieval task: given a query spectrum of an unknown molecule, we aim to find molecules which have similar spectra from a reference database. A traditional approach is to compare

the query against reference spectra in the reference database. The candidate molecules from the reference database are ranked based on the similarity between their reference spectra and the query, and the best matched candidates are returned. However, the reference databases often contain spectra of a small fraction of molecules in reality, leading to unreliable matching results if the molecule of query spectrum is not in the reference database. Consequently, to mitigate the insufficiency of such databases, alternative approaches for the task are devised.

A number of computational methods or tools have been developed to tackle the task of metabolite identification. Remarkably, machine learning (ML) is the key to recent development of the task, and it can be divided into two main categories:

- *Supervised learning* is to learn a relationship or mapping from input to output. Here, the input is a mass spectrum, and the output is a binary vector (or so-called molecular fingerprints) to indicate which predetermined substructures or chemical properties are present in the measured molecule of the input spectrum. The predicted molecular fingerprints by the learned mapping can be used to characterize the measured molecule or retrieve and score candidate molecules in the reference database. FingerID [Heinonen et al., 2012b], CSI:FingerID [Shen et al., 2014b] and IOKR [Brouard et al., 2016a] are examples of this category.

- *Unsupervised learning* is to learn underlying structures from a set of input without output specified. Here, the input are only a collection of MS spectra. Metabolites may have common substructures, yielding shared subsets of peaks in their spectra. Unsupervised learning allows to extract such relevant substructures. Extracted substructures may be regarded as important biochemical processes and subsequenly used to group similar metabolites or improve the accuracy of metabolite identification. MS2LDA [van Der Hooft et al., 2016] and MSSAR [Mrzic et al., 2017] are examples of this category.

In this thesis, computational methods with a focus on supervised ML are proposed to tackle the metabolite identification task. We aim to develop su-

pervised learning models for identifying metabolites with the two following main criteria: (i) *High predictive performance*: given a query mass spectrum of unknown molecule, the proposed methods are expected to produce a highly accurate list of candidate molecules from the reference database with most similar reference spectra; (ii) *Computational efficiency*: in order to be able to process large-scale datasets of molecules (e.g. PubChem), it is desirable for the proposed methods to produce good list of candidate molecules with fast prediction as well; (iii) *Interpretability*: a mass spectrum is represented by a list of peaks, each of which corresponds to a fragment captured by the device; a set of few peaks (or fragments) may comprise a substructure or chemical property, therefore it is desirable to identify which peaks determine which chemical property from spectra data.

1.2. Overview of the thesis

In this thesis, we explore computational methods for metabolite identification from mass spectra data with a focus on machine learning (ML). More specifically, the contributions of the thesis are presented as follows:

In Chapter 1, we present necessary background knowledge regarding the metabolite identification task from mass spectra data, and thoroughly survey computational methods for dealing with this task. The primary purpose of this survey is not only to summarize the proposed techniques in literature, but also to systematically organize them into groups according to their methodology and approaches. It would be beneficial for researchers to comprehend the key differences between techniques as well as rationale behind their groupings. We grouped computational techniques for the task into the following categories: 1) spectra library; 2) *in silico* fragmentation; 3) fragmentation trees and 4) ML. This thesis is focused on the ML approach, which has two stages: (i) mapping a spectrum to an intermediate representation (usually molecular fingerprints and this stage is often referred to as fingerprint prediction) and (ii) retrieving candidate molecules from the reference database using the predicted fingerprints.

In Chapter 2, we propose two learning models that allow to incorporate

peak interactions along with individual peaks for fingerprint prediction. First, we extend the state-of-the-art kernel learning method by developing kernels for peak interactions to combine with kernels for peaks through multiple kernel learning (MKL). Second, we formulate a sparse interaction model for metabolite peaks, which we call SIMPLE, which is computationally light and interpretable for fingerprint prediction. The formulation of SIMPLE is convex and guarantees global optimization, for which we develop an alternating direction method of multipliers (ADMM) algorithm. Experiments using the MassBank dataset show that both models achieved comparative prediction accuracy with the current top-performance kernel method. Furthermore, SIMPLE clearly revealed individual peaks and peak interactions which contribute to enhancing the performance of fingerprint prediction.

In Chapter 3, we present ADAPTIVE, another method for metabolite identification with learnable intermediate representations from given pairs of spectra and corresponding chemical structures of known molecules. It takes a spectrum of an unknown molecule as input and outputs a list of candidate molecules from the reference database. Instead of using (binary) molecular fingerprints as in existing ML based methods, ADAPTIVE could learn intermediate representations (called molecular vectors) between spectra and chemical structures of molecules. The benefits of learning molecular vectors are: 1) specific to both given data and task of metabolite identification and 2) more compact than molecular fingerprints. By conducting experiments on a publicly available benchmark data set, we demonstrated that ADAPTIVE could obtain a significant improvement in terms of both predictive performance and computational efficiency compared to the existing state-of-the-art methods for the same task.

We conclude this thesis in Chapter 4.

1.3. Background on Mass Spectrometry

In order to better understand metabolites, various techniques, most commonly used Mass Spectrometry (MS) and Nuclear Magnetic Resonance (NMR) have been employed to measure them in a high-throughput manner with different approaches [Wishart, 2009]. Both of the techniques are quite complemen-

tary and promising in the area, but neither has been shown to be clearly preferred over the other, because different techniques might also be used, depending on various factors such as the type and quality of sample to be analyzed, as well as the concentration and molecular properties of the metabolites. In general, NMR allows for a detailed characterization of the chemical structure of the compound, and it is opted for unambiguous identification of a chemical structure. However a disadvantage of NMR is that it requires abundant and pure sample, yielding low sensitivity. By contrast, MS is more sensitive and specific, requiring less amount of samples, but providing less information about the chemical structures, namely its elemental composition and some structural fragments. Furthermore, the most important information exclusively obtained by MS is the molecular weight of the target molecule. We focus on the use of MS rather than NMR throughout the rest of this thesis.

MS is a commonly used technique in analytical chemistry [De Hoffmann and Stroobant, 2007, Gross, 2006, McLafferty and Turecek, 1993] for measuring the mass-to-charge ratio (m/z) of one or more molecules in a chemical sample. The output is a mass spectrum, which is represented by a graph with m/z on the x-axis and the relative abundance of ions with m/z values on the y-axis (Figure 1.1). Another way to represent a mass spectrum is as a list of peaks, each of which is defined by its m/z and intensity value (top-right corner of Figure 1.1). The intensity values are often normalized such that the highest peak has a relative intensity of 100 for the subsequent processing stages.

A mass spectrometer consists of at least these three components: ionization source, mass analyzer and a detector (Figure 1.2). The ionization source is the component by which input molecules become charged ions. Two commonly used form of ionization are: Electron Ionization (EI) and Electrospray Ionization (ESI) which are introduced later. The mass analyzer is the component to physically separates ions according to their m/z. Commonly used mass analyzer types includes: quadrupole, time-of-flight and orbitrap devices. The details of these devices can be found in [Dass, 2007, De Hoffmann and Stroobant, 2007, Makarov, 2000]. Once the ions have been separated according to their m/z, the responsibility of the detector is to detect and quantify the ions.

In order to analyse complex biological mixtures, an initial separation of

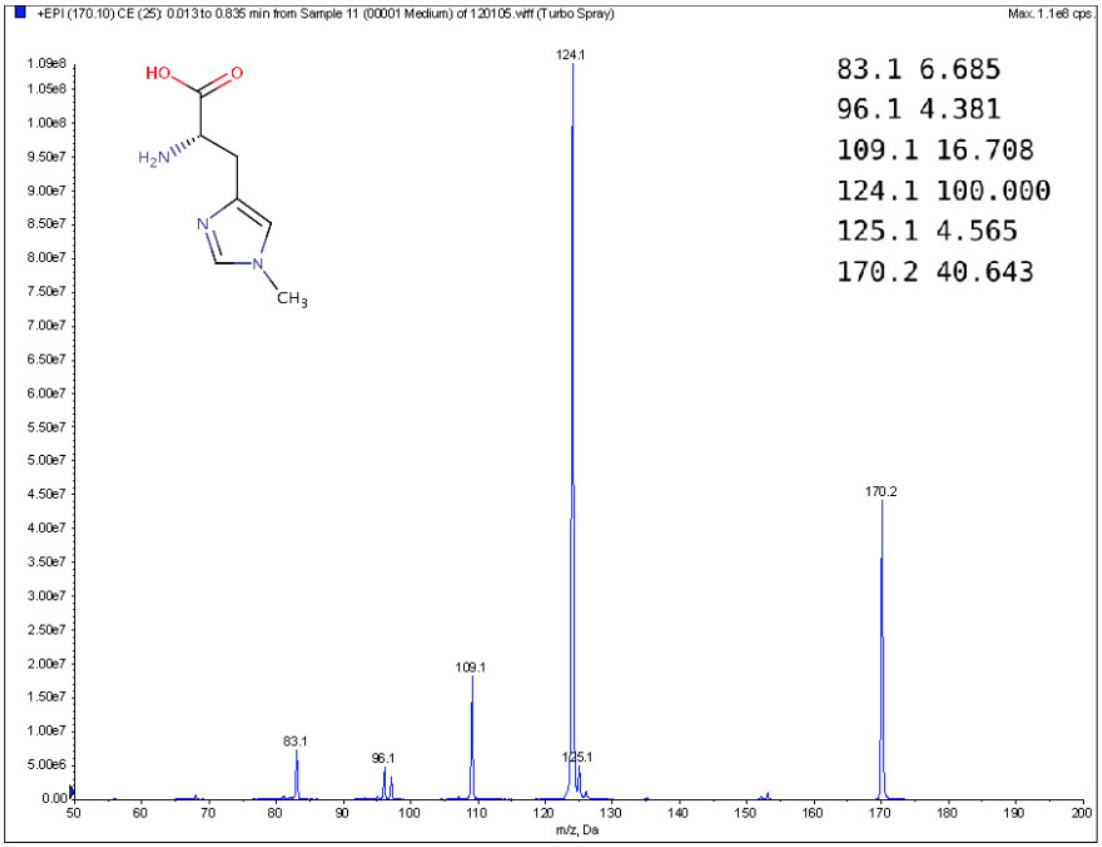

Figure 1.1: Example mass spectrum from the public Human Metabolome Database for 1-methylhistidine (HMBD00001) [Wishart et al., 2017], with its corresponding chemical structure (top-left) and peak list (top-right).

the mixture is often performed by a chromatographic step to provide pure or near pure compounds to the mass spectrometer [De Hoffmann and Stroobant, 2007, McLafferty and Turecek, 1993]. There are two commonly used forms of chromatography: gas chromatography (GC) and liquid chromatography (LC). GC requires the input sample to be in the gaseous phase and targets typically thermally stable and volatile compounds. GC is commonly coupled with EI method which is previously mentioned in mass spectrometry, called GC-MS, or GC-EI-MS. The gas-phase compounds eluted by GC are taken to the ionization source and then ionized by the bombarding electrons. The resulting molecular ion is a positively charged radical and then is broken into fragments, some of which will be charged, some will be neutral. The mass spectrum contains

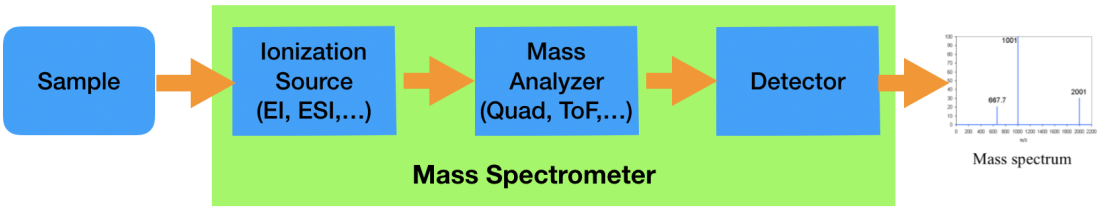

Figure 1.2: Main components of a mass spectrometer: Ionization source, mass analyzer and detector

peaks corresponding to the masses of the charged fragments and the precursor ion as well. Since these values provide the masses of some its substructures, they can be used elucidate the structure of the molecule. Different from GC, LC uses liquid mobile phase and suitable for the study of more thermally unstable and non-volatile compounds. This form is most commonly coupled with ESI mass spectrometry, known as LC-ESI-MS. Instead of losing an electron as in previously mentioned GC-EI-MS, most of the compounds result in protonated (or deprotonated) molecular ions and adduct ions by LC-ESI-MS. ESI method is one of the softest ionization methods, which means that the molecular ions are unlikely to be fragmented further, providing no information about the structures of compounds. A MS/MS system, also known as tandem MS (denoted MS/MS) consists of two mass analyzers coupled with Collision Induced Dissociation (CID) has been a versatile and powerful for many applications. Ions are separated in the first mass analyzer (MS1), then enter a collision or fragmentation cell and fragmented, leading to generation of ions called product ions which are separated in the second mass analyzer (MS2) and detected, eventually resulting in MS/MS spectra or tandem mass spectra. Multistage mass spectrometry (MS^n) allows to further fragment the product ions, providing ways to link these product ions to their precursor ions, thus, providing more information about fragmentation process. For this purpose, product ions found in MS/MS are chosen as precursor ions and fragmented to smaller product ions, resulting in MS^3 spectra. By the same way, MS^4 and so on are produced.

1.4. Computational approaches for metabolite identification from mass spectra data

Identification of metabolites from mass spectra is an important step for further chemi-biological interpretation of metabolomics samples. In practice, this process is presumed to be one challenging and also time-consuming task in metabolomics experiments. Different from peptides and protein where the fragmentation is generally simple due to the repetition of their structures, the fragmentation process of metabolites under varying fragmentation energies is a more complicated stochastic process. Therefore, the interpretation of mass spectra is cumbersome and require expert knowledge. There have been lots of computational techniques/software proposed and developed to deal with the task of metabolite identification. The primary purpose of this survey is not only to summarize the proposed techniques in the literature, but also to systematically organize them into groups according to their methodology and approaches. It would be beneficial in making researchers comprehend the key differences between techniques as well as the rationale behind their groupings. In general, we grouped computational techniques for the task into the following categories: (1) mass spectral library; (2) *in silico* fragmentation; (3) fragmentation tree and (4) ML. Given a query MS/MS spectrum of an unknown compound, mass spectral library is to compare the query spectrum against a database of MS/MS spectra of reference compounds and rank the candidates based on their similarity to the query spectrum. In contrast, *in silico* fragmentation attempts to generate simulated spectra from the chemical structures of reference compounds in a database and compare them to the query MS/MS spectrum. Fragmentation trees are constructed from MS/MS spectra by optimization techniques and can be used to cluster compounds into groups. ML is to learn and predict intermediate representations between spectra and compound structures and then use such representation for matching or retrieval. The scheme of this grouping is illustrated in Figure 1.3 and the details of the approaches and their difference will be presented in the following subsections.

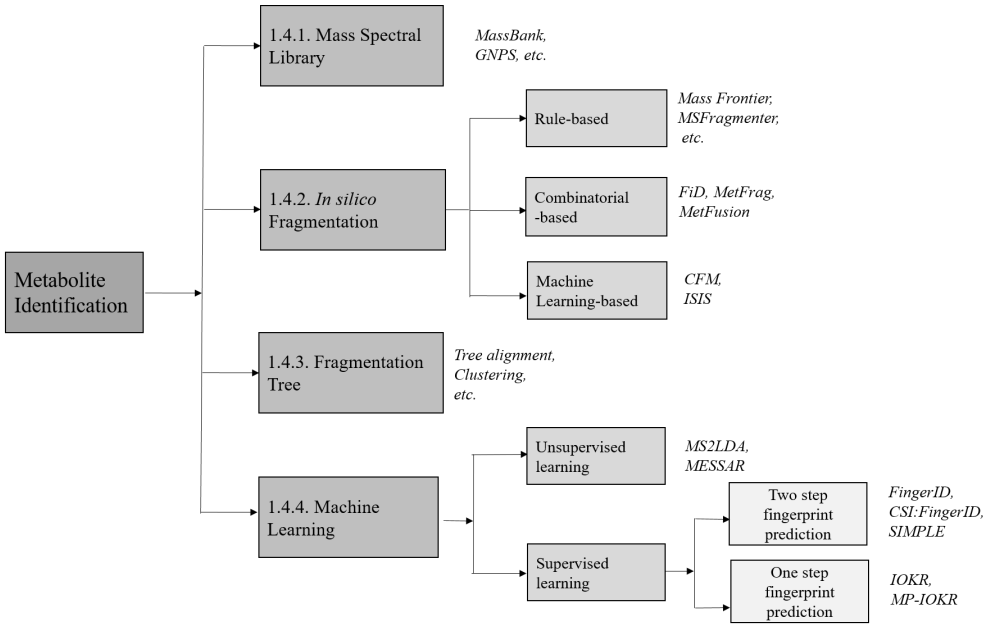

Figure 1.3: The overview of approaches for metabolite identification. The numbers show the corresponding (sub)sections for each category

1.4.1 Mass spectral library

A traditional approach is to compare an unkown MS or MS/MS spectrum of unknown compound against a database of a number of reference MS or MS/MS spectra [Dunn and Ellis, 2005, Scheubert et al., 2013, Tautenhahn et al., 2012]. The candidate molecules from the database are ranked based on their similarity of their spectra and the query spectrum and the best matching candidates are returned. In order to do that, various similarity or distance function have been proposed, from simple weighted counts of matching peaks [Stein and Scott, 1994], to more complicated probability-based measures [Mylonas et al., 2009].

However, the main disadvantage of these methods is that, the reference database is often incomplete and represents only a small fraction of molecules in reality, leading to unreliable matching results if the reference spectrum of the targeted compound is not contained in the database. For example, the public Human Metabolome Database [Wishart et al., 2017] consists of MS/MS spectrum for only approximately 2000 compounds, compared to more than 40,000

known human metabolites. The Metlin database [Smith et al., 2005] contains MS/MS spectra for more than 13,000, compared to over 240,000 endogenous and exogenous metabolites. The Global Natural Products Social Networking (GNPS) Library [Wang et al., 2016] contains MS/MS spectra for around 4000 compounds.

Consequently, to mitigate the insufficiency of the reference database, alternative approaches for identifying metabolites have been devised to deal with unavailability of measured reference spectra.

1.4.2 *In silico* fragmentation tools to aid metabolite identification

Due to the lack of MS/MS data of compounds in mass spectral databases, the ability to identify unknown molecules through search in mass spectra databases is limited as mentioned in the previous subsection. Therefore, the advent of software tools for predicting fragments and their abundance from the molecular structures of compounds can fill the gap between spectral and structural databases. This strategy has been successfully applied in protein studies to construct database containing data on trypsin-associated cleavage and MS/MS spectra of peptides, such as MASCOT [Cottrell and London, 1999] and SEQUEST [Eng et al., 1994]. However, compared to the prediction of fragmentation mechanism for peptides and protein which is simple due to the repetition in their structures, the fragmentation of precursor ions of metabolites in a tandem mass spectrometer is a much more complicated stochastic process and depends on various factors including: the detailed three-dimensional structure of metabolites, the amount of energy to break several certain bonds to obtain the product ion, the probabilities of different dissociation reactions which can be considered as a function of the applied collision energy and the pressure in the collision chamber and so on. Nowadays, many *in silico* fragmentation software tools have been developed and are used to identify MS/MS spectra when the reference spectrum is not available. In this section we surveys different tools/methods using various algorithms for *in silico* fragmentation. The algorithms differ in the way that they deploy different strategies to generate *in*

silico fragments from the chemical structure/graph of the candidate compound. We can divide them into three subgroups: rule-, combinatorial- and ML-based fragmentation tools (see Figure 1.3).

Rule-based methods

The rule-based *in silico* fragmentation tools are used to predict/generate theoretical spectra from molecular structures/graph of compounds in the database using a set of rules. This set of rules is a collection of general and heuristic rules of fragmentation processes extracted from data sets of elucidated MS/MS spectra. The predicted spectra of candidate compounds from the database will be compared with the queried spectrum [Hill et al., 2008, Kumari et al., 2011].

A typical commercial software tool, Mass Frontier [Mistrik, 2004], developed by HighChem, can generate fragments according to general rules, or to specific rule libraries. The libraries can be defined by users or provided by HighChem or combination of both. ACD/MS Fragmenter (available at: http://www.acdlabs.com), another commercial tool, also uses a comparable set of rules to generate fragments. MOLGEN-MSF [Schymanski et al., 2009], developed by the University of Bayreuth, uses general fragmentation rules and also is able to accept additional rules as an optional input file when calculating fragments. Besides, non-comercial rule based software tools, like MASSIS Chen et al. [2003] and MASSIMO Gasteiger et al. [1992] adopted different ways. In particular, structure-specific cleavage rules contained in MASSIS are divided into 26 different molecular classes. A molecule is classified into one or some of these classes and the corresponding fragmentation rules are applied to obtain a set of fragments. MSSIMO uses a small set of general fragmentation reactions parameterized with reaction probabilities drawn from a collection of determined fragmentations.

In fact, these rule-based methods are not preferred in practice due to several disadvantages: 1) the fragmentation process can significantly be variant due to small changes in structure of a molecule. Hence, a fragmentation rule collected from a known fragmentation of a molecule may not be applied to another, even though they have very similar chemical structures; 2) experimental results showed that, a set of general rules is insufficient to identify some ob-

served fragments with a reasonably high accuracy. Although specific rules are constantly added to rule databases, they do not need to be applied to a new undiscovered compound in many cases and 3) the product ions of generated spectra have the same intensities because the bond cleavage rates are ignored. In reality, different molecules can generate the same product ions and the relative intensities can play a meaningful role in distinguishing these molecules.

Combinatorial-based methods

Different from the above software tools which rely on fragmentation rule databases, combinatorial-based methods are to generate a graph of substructures from the chemical structure of a candidate compound in the database (see Figure. 1.4), then find the most likely subset of the substructures or so-called fragmentation trees that best matches the query spectrum by solving optimization problems. An advantage offered by this approach is in situations where MS/MS spectra of compounds with less known fragmentation rules are queried. Some typical methods are reviewed in this subsection. In general, methods belonging to this subsection differ in the way of how they find the fragmentation tree best matches to the query spectra to produce a similarity score.

FiD (Fragment iDentificator, Heinonen et al. [2008]) performs a search over all possible fragmentation paths and outputs a ranked list of alternative structures. More specifically, given a graph structure of a precursor ion and its MS/MS spectrum, FiD first generates all possible connected subgraphs by a depth-first graph traversal (see Fig. 1.4), then computing the masses of product ions corresponding to the generated subgraphs to match with observed peak masses in the spectrum. After that, a list of candidate fragments are obtained then each of which is assigned a cost, namely, the standard bond energy required to cleave bonds from the precursor ion. Obviously, the candidate fragment with smaller cost will be preferred. Finally, a combinatorial optimization method, such as mix integer linear programming (MILP) is used to assign candidate fragments to measured peaks with minimal cost. Their experimental results show that, the product ions predicted by FiD agree better with the manual identification produced by domain experts than those of the rule-based

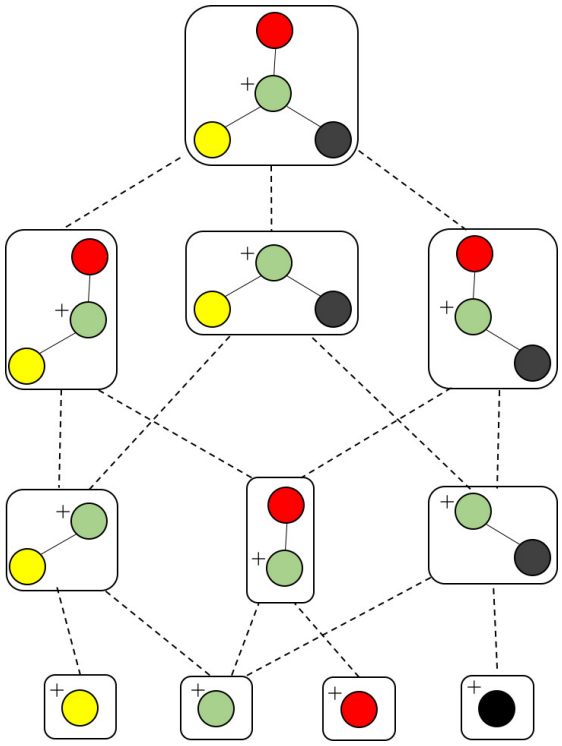

Figure 1.4: An illustration of generating all connected subgraphs of the precursor graph

fragment identification tools mentioned in the previous section. However, the main drawback of FiD is the computational expensiveness due to the following reasons: 1) rapid increase in the number of connected subgraphs; 2) the computational complexity of MILP to explain peaks with most likely candidate fragments. For these reasons, FiD can be applied to only small sized molecules.

Another combinatorial based method is MetFrag [Wolf et al., 2010] using heuristic strategies, such as breadth-first search algorithm with a maximum tree depth parameter or removing duplicated subgraphs, to limit the search space of candidate fragments, overcoming the computational difficulty of FiD which employs depth-first graph traversal to generate subgraphs, as illustrated in Figure 1.4. Hence, it is much faster than FiD and can be applied to a full structure database to find the compound that explains best the spectrum. MetFrag use bond dissociation energies for the cost of cleaving bonds. The candidate fragments are then used to rank the candidate molecules in the database with-

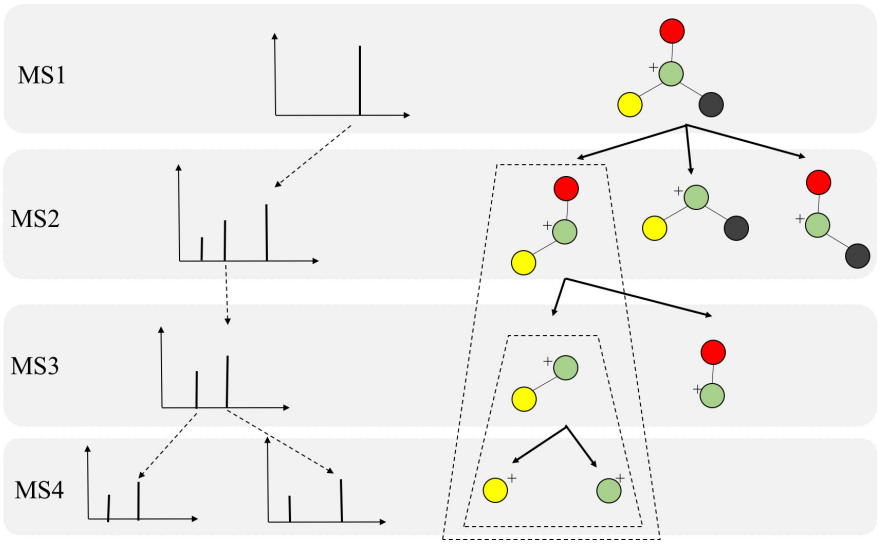

Figure 1.5: An illustration of MAGMA to recursively rank structure candidates with multiple levels

out finding the most likely fragments corresponding to the spectrum. In the same vein, MAGMA, introduced in [Ridder et al., 2012], is an extended version to multistage spectral trees MS^n. Different from MetFrag, when a substructure is considered to explain an MS^2 product ion which is the precursor ion of MS^3 spectrum, in addition to its substructure score, the resulting MS^3 spectrum is also taken into account. This spectrum is temporarily annotated with only subset of the substructure, similarly to MS^2 level fragmentation spectrum. Then, the substructure scores obtained at level 3 are added to the sore at level 2 and this total core is for ranking substructure candidates for MS/MS peak and its fragmentation spectrum. This procedure is applied recursively to handle MS^n with any level, as illustrated in Figure 1.5.

Gerlich and Neumann [2013] presented a system, namely MetFusion, to combine the results from MassBank (search in spectral database) and MetFrag as illustrated in Figure 1.6. The aim of this combination is to take advantage of complementary approaches to improve the compound identification, that is, the vast coverage of the structural databases queried by MetFrag and reliable matching results achieved by search in spectral libraries if similar spectra are available. The experimental results show that a combination of an *in silico* frag-

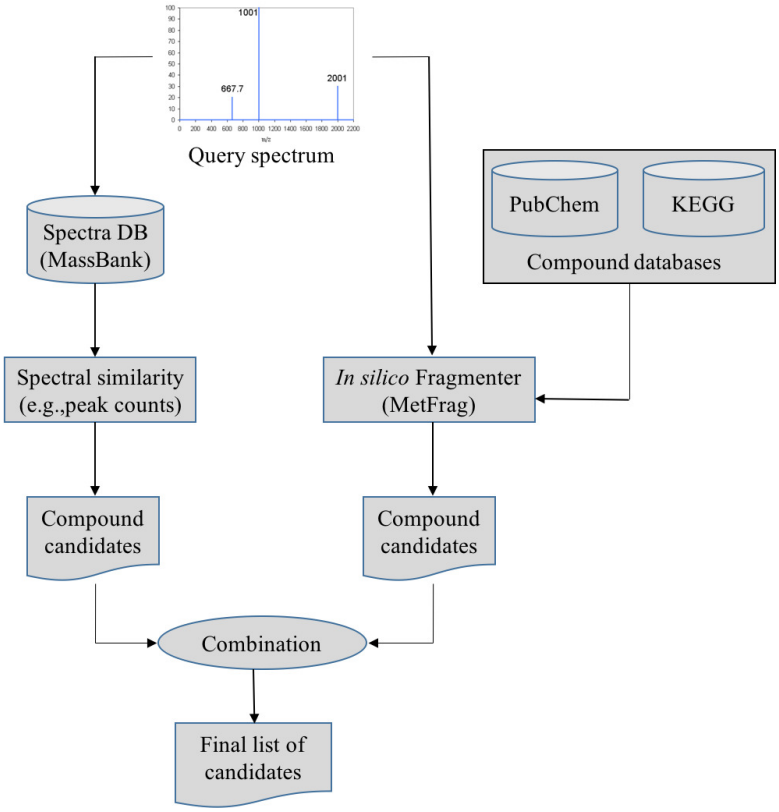

Figure 1.6: The flowchart of MetFusion: MassBank and MetFrag process the query spectrum and return two individual ranked list of compound candidates. The lists are then combined into a single integrated list of re-ranked candidates by calculating the similarity between candidate structures.

mentation based method with curated reference measurements can improve compound identification and achieve the best of two approaches. More details about this method and results can be found in [Gerlich and Neumann, 2013].

A drawback of this approach is that the above methods are mainly based on a bond disconnection based approach to generate fragments from molecules, e.g. standard bond energy and bond dissociation energy used by FiD and MetFrag, respectively. However, these are solely approximate estimates and bond dissociation energies are much more complicated in reality. These limitations have been tackled with some methods based on learning models which are presented the following subsections.

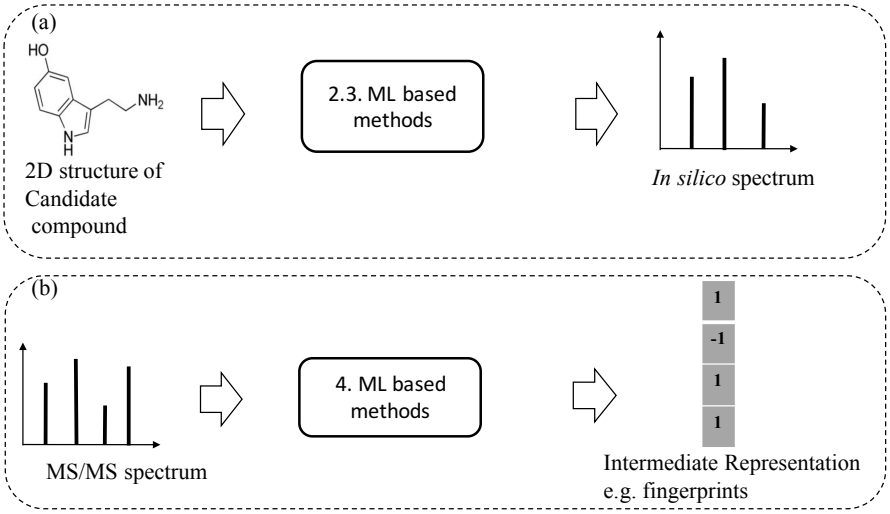

Figure 1.7: An illustration to clarify the difference between ML based methods for learning and predicting *in silico* spectra from 2D structures of compounds (a) and ML based methods for learning and predicting substructures or chemical properties from MS/MS spectra (b). The numbers indicate the (sub)sections for each category.

Machine Learning- based methods

Besides the above approaches to generate *in silico* fragmentation from graph structure of compounds, there are a few work proposed to use ML models to learn the fragmentation process from the training data and have shown great promise in generating *in silico* spectra for the structural identification purpose. To avoid the confusion with the content in section 4, we clarify here that ML methods are used to learn and predict the presence of certain fragments (e.g. whether a bond between two atoms is broken or not) to generate *in silico* spectra from chemical structures. In a different sense, methods in section 4 are to learn and perform classification or clustering from spectra (see Figure 1.7 for illustration).

The previously mentioned methods to generate *in silico* fragments from chemical structure of compounds are based on either chemical reaction equations or approximate bond strength. None of them have shown sufficient accuracy in generating *in silico* spectra for enable automated and correct identifica-

tion of metabolites. To overcome the difficulty, Kangas et al. [2012] presented a method, named ISIS, using ML to generate *in silico* MS/MS spectra for lipids solely from chemical structure of compounds without fragmentation rules and no need to define bond dissociation energy. The main idea is that, for every bond in the molecular structure, one artificial neural network (ANN) is designed to predict bond cleavage energy from which bond cleavage rates can be calculated to determine the relative intensities; another is to predict which side of the bond is charged and captured by the detector in the mass spectrometer. These ANNs are iterated over all bonds in a molecule to find bond cleavage energies and charged ions. For the leaning process, the weights of the former ANN are trained by genetic algorithm (GA) to better predict the bond cleavage energies that produce ions and their corresponding intensities in the *in silico* spectra. The objective of GA is to have the *in silico* spectra match those in the experimental spectra using a Pearson R^2 correlation. The latter ANN is trained by backpropagation algorithm in which the labels can be found by comparing the fragment masses to the experimental spectra.

Allen et al. [2015] proposed a probabilistic generative model, namely Competitive fragmentation mode (CFM), for the fragmentation process. They assume that each peak in the spectrum is generated by a fixed length sequence of random fragment states. It consists of two models: transition model to define the probability of each fragment leads to another at one step in the process and an observation model to map the final intermediate fragment state to the give peak. The parameter estimation for the transition and observation models is performed by an Expectation Maximization -like algorithm. The trained CFM can be used to predict peaks in the spectrum and for metabolite identification. The results showed that, CFM obtained substantially better ranking for the correct candidate than MetFrag and FingerID. However, like other above methods, this method is limited to small molecules due to the combinatorial enumeration of fragmentation possibilities. It is worthy noting that, while ISIS is based on supervised ML, CFM is based on unsupervised learning to predict spectra.

1.4.3 Fragmentation tree

Fragmentation tree (FT) plays an important role in interpreting the structure of molecules since it is usually assumed that only MS/MS spectrum is not sufficient to describe the fragmentation process. It is noteworthy that these FTs are constructed from spectra while the trees mentioned in combinatorial based methods are generated from chemical structures of candidate compounds. This section is devoted to review the benefits of the use of fragmentation trees for metabolite identification and summarize methods to construct them directly from the MS/MS spectra.

Unlike proteins and glycans where molecules are only fragmented at specific chemical bonds and thus the fragmentation process can be well understood, this process for small metabolites can happen at almost any bonds, hence, being difficult to predict and interpret MS/MS data. Böcker and Rasche [2008] proposed using FTs for interpretation of MS/MS spectra. The FT as shown in Figure 1.8 can bring several benefits such as, they can be used to identify the molecular formula of a molecule , also to interpret the fragmentation process of a precursor ion by MS/MS spectrum (see Rasche et al. [2010]). Because of this reason, there are some efforts [Brouard et al., 2016a, Shen et al., 2014b] to use FTs combined with MS/MS spectra in identifying metabolites which will be discussed later. Moreover, we can align FTs of two unknown compounds to compare them based on their corresponding trees, by which, useful information about unknown compounds that cannot be identified also can be derived such as a clustering (see Rasche et al. [2012], Rojas-Cherto et al. [2012] for more details).

A FT is represented by a set of vertexes, each of which corresponds to a fragment or precursor ion, and is annotated with its molecular formula. Edges connecting pairs of vertexes represent fragmentation reaction and are annotated with the molecular formulas of neutral loss. Briefly, FT computation is performed in two main steps: 1) Construction of weighted fragmentation graph containing all possible trees corresponding to the given MS/MS data; 2) Searching for the highest score tree inside the graph. More specifically, the fragmentation graph is constructed as follows: each peak in the MS/MS spectra is assigned to one or more molecular formulas with mass sufficiently close

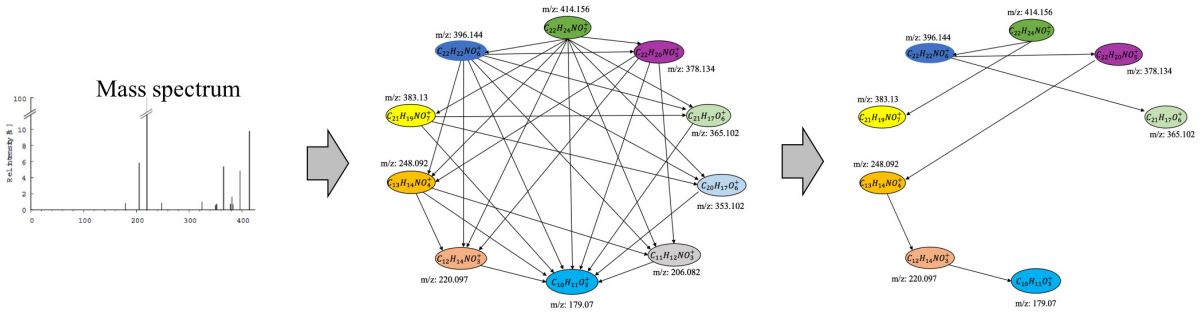

Figure 1.8: Noscapine and the corresponding hypothetical fragmentation tree computed by the method introduced in Rasche et al. [2010].

to the peak mass. These resulting molecular formulas are vertexes of a directed acyclic graph (DAG). Two vertexes u and v are connected by an edge (u, v) if the molecular formula of u is sub-formula of the formula of v and that edge is assigned a score using the annotated neutral loss (i.e., the fragment not being captured by the device) and/or other properties such as peak intensities, mass deviation, representing how likely the neutral loss is. Also, vertexes in the graph are colored so that two vertices with the same color correspond to the same peak. To avoid the case that, there are two vertexes in the FT to represent the same peak, another constraint is added, that is any two vertexes in the tree have different colors, (or so-called colorful tree) must be imposed, leading to *the Maximum Colorful Subtree problem* (**MCS**):

MCS problem: Given a vertex-colored DAG $G = (V, E)$ with a set of colors C and weights $w : E \to \mathbb{R}$. Find the induced colorful subtree $T = (V_T, E_T)$ of G of maximum weight $w(T) = \sum_{e \in E_T} w(e)$.

Despite the fact that finding MCS is an NP-hard problem which was proved in Rasche et al. [2012], many algorithms have been proposed to solve this and they can be categorized into two main groups: exact algorithms and heuristics.

Exact algorithms: Böcker and Rasche [2008] used dynamic programming over vertices and color subsets to solve exactly the MCS problem. Denote $W(v, S)$ the maximal score of a colorful tree with root v and a color subset

$S \subset C$. The matrix W can be computed by the following recurrence:

$$W(v, S) = \tag{1.1}$$

$$\max \begin{cases} \max\limits_{u:c(u)\in S\setminus\{c(v)\}, vu\in E} W(u, S \setminus \{c(v)\}) + w(v, u) \\ \max\limits_{(S_1,S_2):S_1\cap S_2=\{c(v)\}, S_1\cup S_2=S} W(v, S_1) + W(v, S_2) \end{cases}$$

Then, the two-dimensional matrix W can be computed from the above recurrence with the initial condition $W(v, \{c(v)\}) = 0$. The time and space complexity of this algorithm are $O(3^k k|E|)$ and $O(2^k|V|)$, respectively. This algorithm should be used for only small size molecules due to these exponential running time.

A brute-force approach also is introduced by the same authors where all combinations of vertices that can form a colorful set are considered. Maximum Spanning Tree (MST) is calculated for each combination with some additional constraints that the resulting tree is a colorful subtree. Finally, the MST with the highest score will be selected as the fragmentation tree.

Interestingly, Rauf et al. [2013] formulates the MCS problem as an integer linear program (ILP). Specifically, denote x_{uv} be a binary variable for each edge uv of the fragmentation graph $G = (V, E)$ and $V(c)$ be the set of all vertices in G, which have the color c, for each $c \in C$. Then, the MCS problem can be represented by:

$$\max \sum_{uv\in E} w(u, v)x_{uv} \tag{1.2}$$

$$\text{s.t.} \begin{cases} \sum_{u \text{ with } uv\in E} x_{uv} \leq 1 & \text{for all } v \in V \setminus \{r\}, \\ x_{vw} \leq \sum_{u \text{ with } uv\in E} x_{uv} & \text{for all } vw \in E \text{ with } v \neq r, \\ \sum_{uv\in E \text{ with } v\in V(c)} x_{uv} \leq 1 & \text{for all } c \in C, \\ x_{uv} \in \{0, 1\} & \text{for all } uv \in E. \end{cases}$$

The constraints in the above optimization problem ensure that the obtained solution satisfies the condition of being colorful subtree. In particular, the first

constraint ensures that there is at most one parent vertex for each vertex in the graph, meaning that the solution is a forest. The second constraint require the solution to be connected and the third one make sure that at most one vertex of each color will be selected in the solution.

Heuristics: Böcker and Rasche [2008] proposed a simple greedy heuristic by considering the edges according to their weights in descending order. The first edge is pick first. Then, the next edge from the ordered list that satisfies the constraint of colorful subtree is selected or rejected, otherwise. This procedure is iterated until enough edges have been selected. Another strategy, top-down heuristic, starts at root and repeatedly select the outgoing edges satisfying the constraint of colorful subtree with highest score or move back to the root if no such edge exists. The algorithm terminates if no edge at the root can be chosen.

Besides the above methods, Böcker and Rasche [2008] presented a heuristic that combines dynamic programming and a greedy approach. The idea behind is that, dynamic programming strategy is used to construct a preliminary subtree (or so-called backbone) with small number of vertices and then complete it by using a greedy approach (see Böcker and Rasche [2008] for more details).

Some experimental results show that, for small molecules, the brute-force and dynamic programming based methods can quickly find the optimal solution. Especially, for task requiring construction of accurate FTs, such as tree-alignment for MS/MS spectra [Rojas-Cherto et al., 2012], it is advised that exact methods should be used. In the case of dealing with a huge number of molecules, to decrease the running time, the heuristic in combination with an exact method (e.g., tree completion heuristic, Rauf et al. [2013]) may be preferred. Increasing the size of backbone tree improves the quality of the results but endures the price of significantly increased running time.

1.4.4 Machine learning-based metabolite identification

Recently, several ML frameworks have been introduced to deal with the task of metabolite identification. Besides identifying chemical compounds by searching in structural databases as presented in the previous sections, there are some methods proposed to predict structural substructures or general chemical properties such as [Brouard et al., 2016a, Heinonen et al., 2008, 2012b]. An-

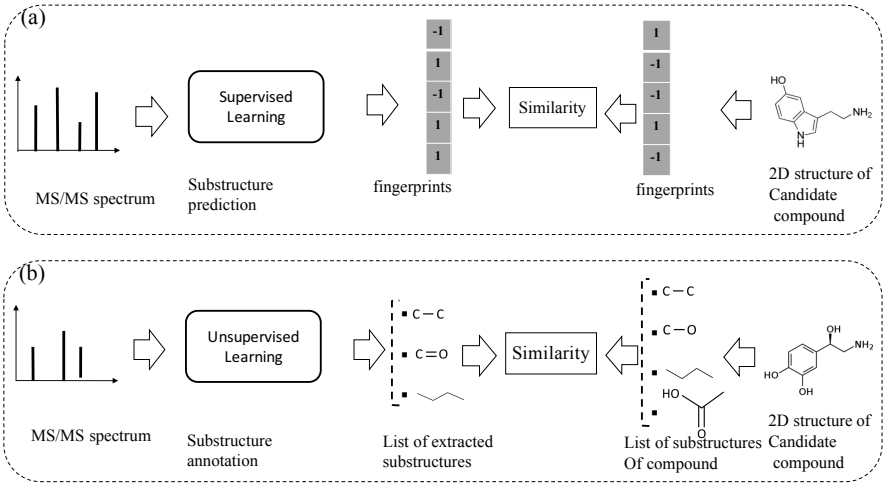

Figure 1.9: An illustration to clarify the difference between supervised and unsupervised learning for metabolite identification: (a) substructure prediction using supervised learning to map a given MS/MS spectrum to an intermediate representation (e.g. fingerprints), which is subsequently used to retrieve candidate metabolites in the database. (b) substructure annotation using unsupervised learning to extract biochemically relevant substructures with certain confidence from the given spectrum. Then, the similarity between the MS/MS spectrum and a chemical structure of a metabolite is estimated according to their common substructures. Note that the output of supervised learning (e.g. fingerprints) may indicate the presence/absence of all *predefined* substructures whereas that of unsupervised learning may be a list of substructures *frequently occurring* in the database.

other direction is to automatically discover substructures directly from a set of MS/MS spectra from which we can identify the candidate compounds from the database based on their substructures, such as [Mrzic et al., 2017, van Der Hooft et al., 2016]. More importantly, ML approach is the key technology behind the progress. In this subsection, we cover ML frameworks for this task which can be divided into two subgroups: supervised learning for substructure prediction and unsupervised learning for substructure annotation. The difference between the two subgroups can be intuitively illustrated as in Figure 1.9.

Supervised learning for substructure prediction

The task of supervised learning for metabolite identification is that, given a set of MS/MS spectra and corresponding chemical structures of known molecules, one may want to learn a mapping function from a MS/MS spectrum to a chemical structure of molecule. However, this task is challenging because both input and output spaces (mass spectra and chemical structures) are highly structured objects. Instead of learning directly a mapping from a spectrum to a molecule, fingerprint-based approach has been used in many systems. This can be called two-step approach in many publications. A molecular fingerprint is a feature vector which is used to encode the structure of a molecule. In general, the values of this vector are binary indicating the presence or absence of a certain substructure or more general chemical property. The methods based on fingerprint prediction are usually carried out in two steps as illustrated in Figure 1.9. The first step is to predict a fingerprint with supervised ML, which is regarded as a collection of binary classification tasks, each task corresponds to a bit in the fingerprint. The second step then uses the predicted fingerprint to query the database with techniques in ranking/information retrieval.

The first step can be dealt with by classification tools such as linear discriminative analysis (LDA), partial least squares discriminative analysis (PLS-DA, Yoshida et al. [2001]), or decision tree [Hummel et al., 2010b]. A notable method is FingerID [Heinonen et al., 2008], which uses support vector machine (SVM, Burges [1998]) with kernels to predict fingerprint. The kernels for pairs of mass spectra were defined, including integral mass kernel and probability product kernel (PPK, Jebara et al. [2004b]). It is noteworthy that the above methods are mainly based on the information from individual peaks present in the spectra while ignoring their interactions. In fact, such information is proved to be useful in predicting fingerprint.

CSI:FingerID [Dührkop et al., 2015b, Heinonen et al., 2012b], an extended version of FingerID, jointly takes MS/MS spectra and corresponding FTs as input to improve the predictive performance since FTs, reviewed in the previous subsection, can be used to provide prior knowledge about the structure of compound (i.e., dependencies between peaks in spectra), which was ignored in the previous system. In order to do this, kernels for FTs have to be defined which

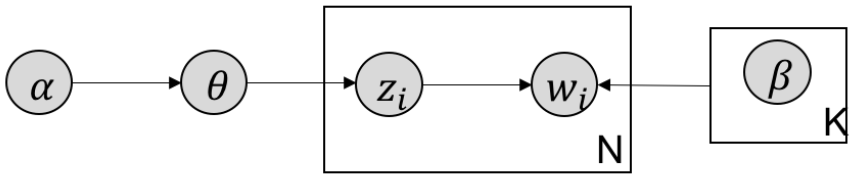

Figure 1.10: Simplified graphical representation of LDA.

range from simple ones for node including node binary (NB), node intensity (NI); for edges including loss binary (LB), loss count (LC), loss intensity (LI) to more complicated ones like common paths counting (CPC), common subtree counting (CSC) and so on. Subsequently, multiple kernel learning (MKL, Gönen and Alpaydın [2011]) is used to combine these kernels using several methods, including centered alignment (ALIGNF, Cortes et al. [2012b]), quadratic combination [Li and Sun, 2010] and l_p-norm regularized combination [Kloft et al., 2011b]. The combined kernel is then used in learning the final model for fingerprint prediction. CSI:FingerID presented improved scores against other benchmarked tools but has the current limitation of processing MS/MS spectra one at a time due to the need of computationally heavy conversion of spectra into FTs

Unsupervised learning for substructure annotation

The metabolites may have common substructures, and this may yield common product ions in their MS/MS spectra. Many substructures among them contain information pertaining to the biochemical processes present. Therefore, extraction of such biochemically relevant substructures allows molecules to be grouped based on their shared substructures regardless of classical spectral similarity. Also, this can be used to improve the accuracy of metabolite identification.

One of typical software tools for chemical substructure exploration is MS2Analyzer [Ma et al., 2014] which is a library-independent tool, allowing to exploit the potential structure information contained in mass spectra. It were developed to elucidate substructures of small molecules from accurate MS/MS

spectra. The main function of this tool is to search mass spectral features including: neutral loss, precursor, fragment ions mass and mass differences in a large number of mass spectra. By combining the searching results and substructure/ compound class relationship knowledge, compounds can be identified. However, MS2Analyzer can find all molecules sharing a specific set of mass spectral features provided by users and sample-specific features are likely to be ignored. Another technique, namely molecular networking [Wang et al., 2016, Watrous et al., 2012, Yang et al., 2013], groups precursor ions i.e., MS1 peaks, based on their MS2 spectral similarity, e.g., cosine score, such that one metabolites which are structural annotated in a cluster can be used to annotate their neighbors. The drawback of molecular networks is that only MS1 peaks with high similarity are grouped and spectral features specifying the clusters have to be manually extracted. Thus, it may be failed to cluster molecules sharing small substructures with low MS2 spectral similarity.

MS2LDA, presented in [van Der Hooft et al., 2016], is a software tool offering benefits of both of these methods while overcoming their disadvantages. It can automatically extract relevant substructures in molecules based on their co-occurrence of mass fragments and neutral losses, and cluster the molecules accordingly. Based on the assumption that, each observed MS/MS spectrum is composed of one or more substructures, MS2LDA adopt Latent Dirichlet Allocation (LDA, Blei et al. [2003]) initially developed for text mining for extracting such substructures. LDA is a bayesian version of probabilistic latent semantic analysis . In standard setting for text mining, LDA models each of D documents as a discrete distribution over T latent topics, each of which is a discrete distribution over a vocabulary of V words. For document d, the distribution over topics, denoted by θ_d, is drawn from a Dirichlet distribution $Dir(\alpha)$, and for each topic t, the distribution over words, denoted by ϕ_t, is drawn from a Dirichlet distribution $Dir(\beta)$. A generative process in LDA is defined on document d as follows (note that the index d for document d is omitted for simplification):

1. Choose $\theta \sim Dir(\alpha)$.

2. For each word w_i in document d:

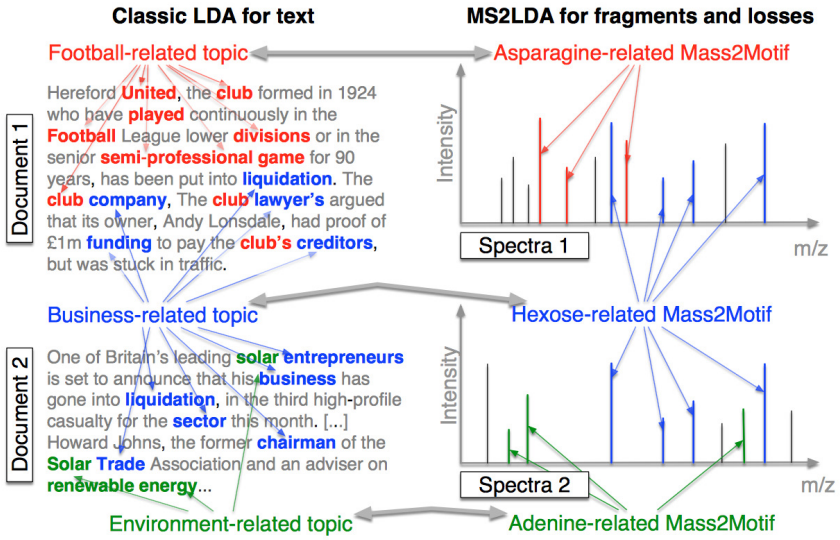

Figure 1.11: The correspondence between LDA for text and MS2LDA for mass spectra: LDA finds topics based on the co-occurrence of words while MS2LDA finds substructures based on the co-occurrence of mass fragments and neutral losses. This figure is adapted from van Der Hooft et al. [2016].

(a) Choose a topic $z_i \sim \text{Multinomial}(\theta)$.

(b) Choose a word $w_i \sim \text{Multinomial}(\phi_{z_i})$

where latent variable z_{di} is a topic assignment for i^{th} word w_{di} in the document d. The parameters to be learned include α and β. The graphical representation of this process is illustrated in Figure 1.10.

The correspondence between text documents and fragmentation spectra can be obviously observed from ML perspective. LDAs decompose a document into topics based on the co-occuring words, while MS2LDA decompose MS/MS spectra into patterns of co-occuring fragments and losses. Learning LDA (MS2LDA) is to extract these topics (patterns or so-called (Mass2) Motifs) as illustrated in Figure 4.1. For reference, either collapsed Gibb sampling [Griffiths and Steyvers, 2004] or Variational Bayes [Blei et al., 2003] can be used to assign topics (Mass2Motifs) to words (peaks). This step applied to mass spectra is called substructure annotation. By MS2LDA, each metabolite can be explained by one or more Mass2Motifs by which we can partly identify un-

known metabolites via their spectra. Also, It can be used to quickly classify metabolites into functional classes without knowing the complete structures.

A drawback of the above mentioned MS2LDA is that, the extracted motifs still need to be structurally annotated based on expert knowledge which is a complex process and time-consuming. To overcome this difficulty, Mrzic et al. [2017] introduces an automated method named MESSAR for substructure recommendation from mass spectra, motivated by frequent set mining which is a popular class of methods in unsupervised data mining. Similarly to MS2LDA, this method is also capable of capturing recurring patterns from mass spectra. In brief, molecular substructures are first generated for a database of metabolites for which both experimental MS/MS spectrum and molecular structures are available, then are combined with fragment ions and mass differences between product fragment ions to construct a single dataset in transactional format. Subsequently, frequent set mining techniques are applied to this set to extract rules of the following format: peaks p (or mass difference md) can be associated with substructure s with support f and confidence c. Such rules can be used to annotate substructures with calculated scores of support and confidence for mass spectra in which, the given peaks and mass differences are observed. Moreover, the recommended substructures can also be used to rank candidate metabolites retrieved from a database. The ranking is performed based on the similarity between recommended substructures and candidate molecular structures. Metabolites with a high number of substructures with high confidence are assigned a higher rank.

It is worth noting that, the aim of the above mentioned methods are similar, i.e., substructure annotation. While MS2LDA only needs a set of unlabeled MS/MS spectra for learning without prior information about the molecular structures, MESSAR utilizes both experimental spectra and the corresponding structures, hence, providing an automated substructure recommendation as opposed to expert-driven substructure annotation by MS2LDA.

1.5. Summary

In this Chapter, we have focused on the introduction of backgrounds on metabolite identification task from MS or MS/MS data. We also have thoroughly reviewed the proposed techniques in the literature to tackle the task. Many techniques and software tools are systematically organized into groups according to their methodologies and approaches so that researchers can comprehend the key differences between techniques as well as the rationale behinds their groupings.

In general, we grouped computational techniques for the task of metabolite identification into the following categories: (1) mass spectral library; (2) *in silico* fragmentation; (3) fragmentation tree and (4) ML. Given a query MS/MS spectrum of an unknown compound, mass spectral library is to compare the query spectrum against a database of MS/MS spectra of reference compounds and rank the candidates based on their similarity to the query spectrum. In contrast, *in silico* fragmentation attempts to generate simulated spectra from the chemical structures of reference compounds in a database and compare them to the query MS/MS spectrum. Fragmentation trees are constructed from MS/MS spectra by optimization techniques and can be used to cluster compounds into groups. ML is to learn and predict an intermediate representation (typically molecular fingerprint) between spectra and compound structures and then use such representation for matching or retrieval. Our research is focused on the ML approach. That is, we develop statistical ML models fo predicting intermediate representations of chemical compounds (or molecules) from MS/MS spectra effectively and efficiently.

Chapter 2

Interaction models for fingerprint prediction

2.1. Introduction

Recent success in computational methods for metabolite identification from mass spectra data has been led by ML, which has two stages: 1) fingerprint prediction: predict a fingerprint with supervised ML, and 2) candidate retrieval: use the predicted fingerprint to query the reference database. Figure 2.1 illustrates the general scheme of twos-stage fingerprint prediction based methods for metabolite identification. Our particular focus in this chapter is the first stage, which predicts the fingerprint that is a binary (or rarely real-valued) vector, indicating the presence (or absence) of a certain substructure or generally chemical property. This stage has been tackled by many ML methods, including linear discriminative analysis (LDA) [Imre et al., 2008] and decision tree [Hummel et al., 2010a]. A notable method is FingerID [Heinonen et al., 2012a], which used support vector machine (SVM) with kernels for pairs of mass spectra, including integral mass kernel and probability product kernel (PPK) [Jebara et al., 2004a].

We point out three drawbacks of the existing ML approaches: 1) all methods are based on the information from individual peaks in spectra, without explicitly considering peak interactions, which leads to the limitation of predictive performance; 2) There is an interesting (not ML-based) software, which out-

puts, given a spectrum, a so-called fragmentation tree (FT) [Böcker and Rasche, 2008, Rasche et al., 2011]. In a FT, possible fragments corresponding to peaks in spectrum are shown as node labels, where parent-child relationships in this tree are inclusive relations of fragments in chemical structure. FT is indeed interesting, but this software of converting a given spectrum into a FT is very slow. CSI:FingerID [Dührkop et al., 2015a, Shen et al., 2014a] used both mass spectra and FTs as input, thinking that structural information of chemical compounds can be captured by FTs. Indeed using FTs might be similar to considering peak interactions. However if FTs are used as input features, spectrum must be converted into a FT not only in training and but also in prediction, which needs a heavy computational load, leading to slow prediction in the second stage, candidate retrieval. 3) Each bit in a fingerprint represents a predetermined chemical property or substructure and its presence is often decided by a few number of peaks. Also the number of training data is small, while sparse learning models have not been considered yet. In addition, sparse learning models are advantageous in that their results are easily interpretable.

Motivated by these drawbacks, we address the following two problems: 1) incorporation of peak interactions into the learning model to improve the predictive performance; 2) introduction of sparsity into the models for interpretation. For the above 1), we propose a kernel for peak interactions and combine this kernel with other kernels defined for individual peaks through multiple kernel learning (MKL). For the above both 1) and 2), we propose a sparse, interpretable model, which we call Sparse Interaction Model over Peaks of moLEcules (SIMPLE). Additionally, we also propose a FT-induced Laplacian regularization to make SIMPLE more robust. We note that in SIMPLE, FTs are used for regularization only and not for input, by which we do not need FTs for prediction, meaning that computationally SIMPLE is much lighter for prediction than [Shen et al., 2014a]. We formulate objective functions to optimize the parameters of models as convex optimization problems, for which we develop an alternating direction method of multipliers (ADMM) algorithm. We evaluated our two proposed models by using real data obtained from the MassBank dataset [Horai et al., 2010], which was used in [Shen et al., 2014a]. We found that incorporation of peak interactions can significantly improve the prediction

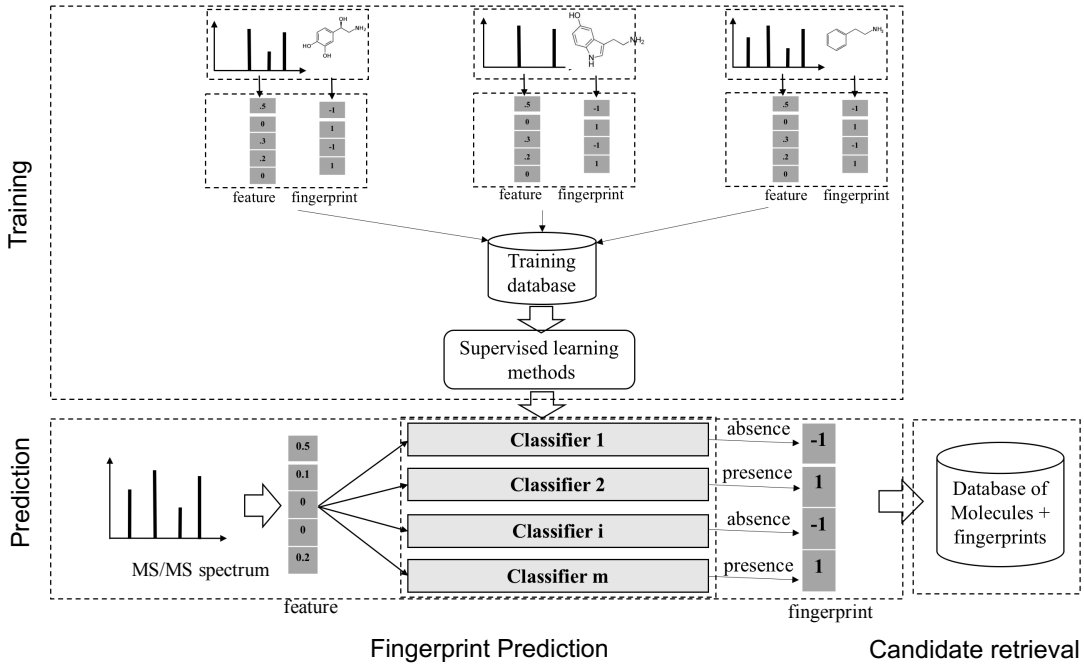

Figure 2.1: A general scheme to identify unknown metabolites based on the molecular fingerprint vectors. There are two main stages, which are as follows: (1) learning a mapping from a molecule to the corresponding binary molecular fingerprint vector by classification methods, given a set of MS/MS spectra and fingerprints; (2) using the predicted fingerprints to retrieve candidate molecules from the databases of known metabolites.

accuracy of not using interactions, resulting in comparable performance with the current top method. Furthermore, SIMPLE could show the interpretability of results, i.e. peaks and peak interactions which contribute to high predicitive performance.

2.2. Related work

The standard data preprocessing converts spectra into high-dimensional feature vectors by dividing *m/z* range into bins and taking accumulated intensity within each bin as a feature value. However the width of bins is hard to determine. In fact, wide bins can cause noise, and narrow bins can induce

alignment errors due to mass error. One idea to overcome this issue is using a kernel, say probability product kernel (PPK, Jebara et al. [2004a]), which can be computed directly from mass spectra. PPK treats each peak in a spectrum as a two-dimensional Gaussian distribution and a spectrum as a uniform mixture of these distributions. Then, kernel between two spectra is computed by all-against-all matching between the component Gaussians. The detail is in Section 2.3.1.

Also we briefly explain the current cutting-edge ML method by MKL, CSI:FingerID [Dührkop et al., 2015a, Shen et al., 2014a], where the input is both MS/MS spectra and FTs. Motivation behind this method is to use structural information, which might be captured by FTs. So they use numerous kernels for FTs to capture any information of FTs, ranging from simple ones, such as node kernels: node binary (NB), node intensity (NI), and edge kernels: loss binary (LB), loss count (LC), loss intensity (LI) to more complex one like common path counting (CPC) which is one of path based kernels. Subsequently, these kernels are combined by MKL methods, such as centered alignment [Gönen and Alpaydin, 2011], quadratic combination and l_p-norm regularized combination [Kloft et al., 2011a]. The combined kernel is then used in learning the final model for fingerprint prediction.

We again emphasize that these methods consider mainly only peaks in MS/MS spectra without explicitly taking peak interactions into account. Also kernels using FTs have the limitation of processing MS/MS spectra, particularly for prediction, due to the need of computationally heavy conversion of spectra into FTs. More importantly, despite the effectiveness for fingerprint prediction in existing work, kernels are generally difficult to interpret and deal with sparse data, such as MS/MS spectra, where each spectrum is high-dimensional and each bit in the fingerprint (output) is decided by a few number of peaks (or features).

2.3. Methods

2.3.1 Kernel method

We develop kernel for peak interactions and combine this kernel with PPK (kernel for peaks) [Heinonen et al., 2012a] through the framework of MKL.

Preliminary: Kernel for peaks

MS/MS spectra have information about individual peaks, i.e. *m/z* and intensity. The problem is that peaks are not correctly aligned each other, due to measurement errors in mass spectrometry devices (alignment error). To overcome this problem, PPK [Jebara et al., 2004a] was utilized to calculate the similarity between two spectra. The idea is that a peak can be considered as a two-dimensional Gaussian distribution and the spectrum is a uniform mixture of peaks (or distributions). The kernel between two spectra is computed by all-against-all matching between the Gaussian distributions.

More specifically, given a mass spectrum $\mathbf{S}$, being a list of peaks, i.e., $\{(m_1, I_1), (m_2, I_2), ..., (m_{N_\mathbf{S}}, I_{N_\mathbf{S}})\}$. The k-th peak of the spectrum $\mathbf{S}$ is represented by a Gaussian distribution p_k centered around the peak measurement (m_k, I_k) and with covariance shared with all peaks: $\Sigma = \text{diag}(\sigma_m^2, \sigma_I^2)$, where σ_m^2 and σ_I^2 are the variances for the mass and intensity, respectively. Hence, the spectrum $\mathbf{S}$ can be represented as a uniform mixture of Gaussian distribution corresponding to peaks contained in it, i.e., $p(\mathbf{S}) = \frac{1}{N_\mathbf{S}} \sum_{k=1}^{N_S} p_k$. Likewise, another spectrum $\mathbf{S}^{'}$ is also represented by a mixture of distributions q_l, $l = 1, 2, 3, .., N_{\mathbf{S}'}$ and $p(\mathbf{S}^{'}) = \frac{1}{N_{\mathbf{S}'}} \sum_{l=1}^{N_{\mathbf{S}'}} q_l$.

With definitions above, the kernel for peaks, $\mathbf{K}_{peak}$, between two spectra $\mathbf{S}$ and $\mathbf{S}^{'}$ is given by:

$$\mathbf{K}_{peak}(\mathbf{S}, \mathbf{S}^{'}) = \int_{\mathbb{R}^2} p_\mathbf{S}(x) q_{\mathbf{S}'}(x) dx \tag{2.1}$$

$$= \frac{1}{N_\mathbf{S} N_{\mathbf{S}'}} \sum_{\substack{1 \le i \le N_\mathbf{S} \\ 1 \le j \le N_{\mathbf{S}'}}} \mathbf{K}(p_i, q_j) \tag{2.2}$$

where $\mathbf{K}(p, q)$ is the PPK between two component Gaussian distributions p and q, computed by 2.3.

$$\mathbf{K}(p, q) = \frac{1}{4\pi\sigma_m\sigma_I}\exp(-0.5(\frac{(\mu_p^m - \mu_q^m)^2}{\sigma_m} + \frac{(\mu_p^I - \mu_q^I)^2}{\sigma_I})) \tag{2.3}$$

where μ_p^m and μ_p^I denote the mass and intensity values of the peaks contained in p, μ_q^m and μ_q^I denote the mass and intensity values of the peaks contained in q.

Kernel for peak interactions

Being rather straight-forward, kernel for peak interactions between two spectra $\mathbf{S}$ and $\mathbf{S}^{'}$ can be defined as follows:

$$\mathbf{K}_{interaction}(\mathbf{S}, \mathbf{S}^{'}) = \sum_{\substack{1 \le i \le j \le N_{\mathbf{S}} \\ 1 \le k \le l \le N_{\mathbf{S}'}}} \mathbf{K}(p_i, q_k) * \mathbf{K}(p_j, q_l), \tag{2.4}$$

where $\mathbf{K}$ is the function defined by 2.3.

Note that $\mathbf{K}_{interaction}$ also can overcome the alignment error problem by taking advantage of $\mathbf{K}_{peak}$. Intuitively, $\mathbf{K}_{peak}$ can be considered as a probabilistic version of the number of common peaks. Similarly $\mathbf{K}_{interaction}$ can be considered as a probabilistic version of the number of common edges or interactions between two spectra.

Combining kernels for peaks and peak interactions

We combine kernels for peaks and peak interactions by using a regular approach: Multiple Kernel Learning (MKL, [Gönen and Alpaydin, 2011]), which can combine kernels from different sources. We use three approaches: the first is the uniform combination of the kernels (**UNIMKL**: the weights for kernels are equal), which can produce good results for prediction in many practical applications. The second and third are two different MKL algorithms: centered alignment (**ALIGN**) and alignment maximization algorithm (**ALIGNF**) [Cortes et al., 2012a].

These algorithms have the same setting. Given a set of kernels $\{\mathbf{K}_j \in \mathbb{R}^{n\times n}, j = 1,..,m\}$, computed from n data points and representing different sources of information. The vector $\mathbf{y} = \{-1,1\}^n$ corresponds to the output or labels of data points. The aim of MKL is to seek a combination of these kernels, as defined by:

$$\mathbf{K}_\alpha = \sum_{j=1}^{m} \alpha_j \mathbf{K}_j \tag{2.5}$$

A popular approach to MKL, which is called two-stage MKL, separate kernel learning from prediction learning. In the first stage, the combined kernel is learned through the use of an objective function, such as centered alignment that measures the similarity of two kernels over the training data set. For the purposes of MKL, alignment is often measured between the combined kernel in 2.5 and the target kernel $\mathbf{K}_y = \mathbf{y}\mathbf{y}^T$. Subsequently, the learned combined kernel is used in the second stage with kernel based classifiers such as relevance vector machine (RVM) used in [Burden and Winkler, 2015] or SVM used in our experiments.

Formally, the centered kernel of a given kernel $\mathbf{K} \in \mathbb{R}^{n\times n}$ is defined by:

$$\mathbf{K}_c = [\mathbf{I} - \frac{\mathbf{e}\mathbf{e}^T}{n}]\mathbf{K}[I - \frac{\mathbf{e}\mathbf{e}^T}{n}] \tag{2.6}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{e}$ is the vector with all ones.

The centered alignment between two kernels $\mathbf{K}$ and $\mathbf{K}^{'}$ is defined by:

$$\rho(\mathbf{K}, \mathbf{K}^{'}) = \frac{< \mathbf{K}_c, \mathbf{K}_c^{'} >}{||\mathbf{K}_c||_F ||\mathbf{K}_c||_F} \tag{2.7}$$

$$\text{where } < \mathbf{K}_c, \mathbf{K}_c^{'} > = \text{trace}(\mathbf{K}_c^T \mathbf{K}_c^{'}) \text{ and } ||\mathbf{K}_c||_F = \sqrt{\text{trace}(\mathbf{K}_c^T \mathbf{K}_c)} \tag{2.8}$$

Cortes et al. [2012a] proposed two MKL algorithms: 1) a simple centered alignment algorithm (**ALIGN**), which assigns alignment scores computed in (7) to combination weights in (5), i.e., $\alpha_j = \rho(\mathbf{K}_j, \mathbf{K}_\mathbf{Y})$, $j = 1,2,3,...,m$. and 2) the alignment maximum algorithm (**ALIGNF**) seeks the weights α by maximizing the alignment scores between the combined kernels $\mathbf{K}_\alpha$ and target kernel $\mathbf{K}_y$,

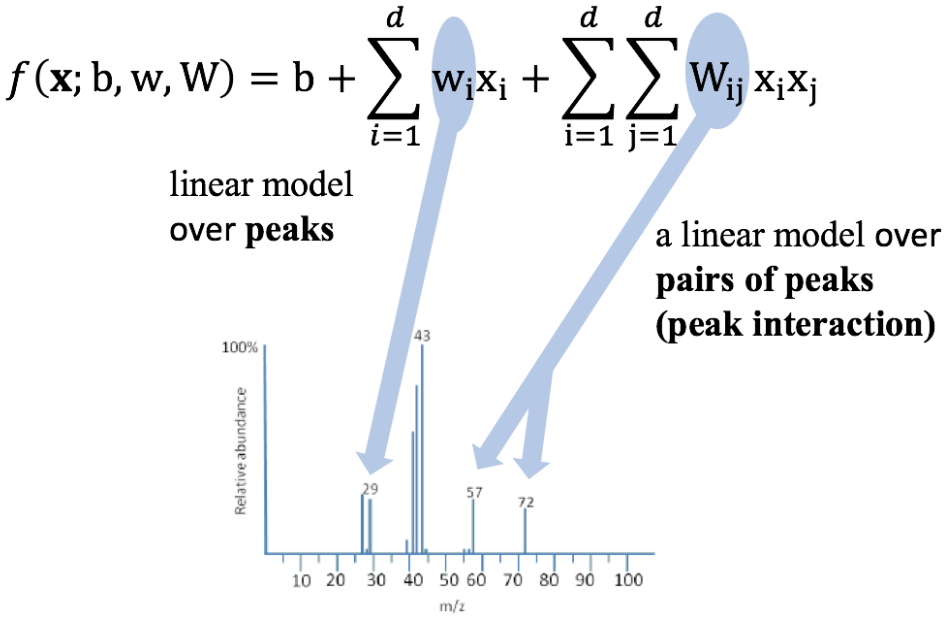

Figure 2.2: Illustration of the predictive model of SIMPLE: the weight vector w of the main effect term captures information about the individual peaks, while interaction weight matrix W of the interaction term captures information about the peak interactions.

resulting in the objective function:

$$\alpha^* = \underset{\alpha}{\text{argmax}} \frac{< \mathbf{K}_\alpha, \mathbf{K}_y >}{||\mathbf{K}_\alpha||_F} \tag{2.9}$$

$$\text{subject to } ||\alpha|| = 1 \, , \alpha \geq 0 \tag{2.10}$$

In our experiments, we conducted comparative experiments by using these algorithm for combining kernels for peaks and interactions.

2.3.2 Sparse Interaction Model over Peaks of moLEcules (SIMPLE)

Kernel-based methods are difficult to deal with sparse data and lack of interpretation. Thus we present a more *interpretable, fast, sparse* learning: Sparse Interaction Model over Peaks of moLEcules (**SIMPLE**), to incorporate peak interactions explicitly. We first preprocess each MS/MS spectrum to generate a

feature vector: we divide *m/z* range into bins and taking accumulated peak intensities in a bin as a feature value to obtain high dimensional vector for each spectrum. These feature vectors are normalized such that all feature values are in [0, 1].

Prediction model

Given a MS/MS spectrum, represented by a feature vector, $\mathbf{x} = [x_1, x_2, ..., x_d]^T \in \mathbb{R}^d$, for one particular bit, we formulate the model for individual peaks and peak interactions as follows:

$$f(\mathbf{x}; w, W) = b + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=1}^{d} W_{ij} x_i x_j \tag{2.11}$$

$$= b + w^T \mathbf{x} + \mathbf{x}^T W \mathbf{x} \tag{2.12}$$

where $b \in \mathbb{R}$, $w \in \mathbb{R}^d$ and $W \in \mathbb{R}^{d \times d}$. The prediction function consists of a bias b and two terms: main effect term parameterized by the weight vector w and interaction term parameterized by the weight matrix W. Their roles are different, as illustrated in Figure 2.2. While the former capture information about the peaks, the latter captures information about peak interactions. Since our task is classification which predicts the presence or absence of properties in fingerprint vector, the output of the model can be computed by $y(\mathbf{x}) = \text{sign}(f(\mathbf{x}; w, W)) \in \{-1, 1\}$. It is important to note that fingerprint prediction is regarded as a collection of a number of binary classification task, each task corresponds to prediction of one particular bit in the fingerprint. Therefore, we have a number of models 2.12 to learn separately.

In order to predict a binary vector of fingerprint, we predict one response variable (hereafter a *property* or a *task*) at each time with a separate classifier. Let consider a property and give a set of n training input/output pairs $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ represent the i-th spectrum and $\mathbf{y}_i \in \{-1, 1\}$ indicates the presence (+1) or absence (-1) of the property. We can learn the parameters by minimizing the following optimization function:

$$\min_{b,w,W} \quad \sum_{i=1}^{n} l(\mathbf{y}_i, f(\mathbf{x}_i; w, W)) + \mathcal{R}(w) + \mathcal{R}(W) \tag{2.13}$$

where $l(\mathbf{y}, \hat{\mathbf{y}})$ is the hinge loss function and computed by $l(\mathbf{y}, \hat{\mathbf{y}}) = (1 - \mathbf{y}, \hat{\mathbf{y}})_+$, in which the operator $(z)_+$ denotes $\max(0, z)$. $\mathcal{R}(w)$ and $\mathcal{R}(W)$ are the regularization terms for vector w and matrix W, respectively.

Our purpose is to seek a model which is accurate as well as interpretable. Different from SVM using l_2-norm regularizer, $\mathcal{R}(w) = \alpha||w||_2^2$, our model uses sparsity-induced regularizer [Tibshirani, 1994], $\mathcal{R}(w) = \alpha||w||_1$ to yield sparse solution and interpretation. As for interaction term, it is natural to impose low-rankness on matrix W due to the existence of groups of peaks interacting with each other. Thus, we propose to use trace norm, similar to [Blondel et al., 2015], i.e., $\mathcal{R}(W) = \beta||W||_*$. Furthermore, due to the symmetry of matrix W, we also impose the positive semidefiniteness on matrix W, i.e., $W \succeq 0$. Therefore, putting all above things together, the objective function of **SIMPLE** becomes:

$$\begin{aligned} \min_{b,w,W} \quad & \sum_{i=1}^{n} l(\mathbf{y}_i, f(\mathbf{x}_i; w, W)) + \alpha||w||_1 + \beta||W||_* \\ \text{subject to} \quad & W \succeq 0 \end{aligned} \tag{2.14}$$

where α and β are hyperparameters to control the sparsity of w and low-rankness of W, respectively. Note that (2.14) is the convex function (see Appendix) which guarantees to find the global optimal solution.

Fragmentation trees (FTs) as prior information

As the number of interactions is large, it would be a good idea to regularize (2.14) with background knowledge. We propose to regularize the interaction matrix W by FTs of the spectra. However, there are two questions to deal with : 1) How to represent the structural information from FTs; 2) how to incorporate them into the convex objective function (14) while preserving its convexity to guarantee the global optimal solutions.

To answer the first question, our idea is to construct an affinity (adjacency) matrix A for all peaks in the spectra. Concretely, each spectrum is converted into a FT by algorithms in [Rasche et al., 2011]. Our assumption is that, the frequency of an interaction present in the fragmentation trees reflects how strongly the corresponding features are interacting with each other. From that, we calculate the co-occurrence of all peak pairs in the trees to construct the affinity

matrix (Figure 2.3).

To answer the second question, we impose the constraint of being positive semidefinite on interaction matrix W. By this, W can be decomposed into the low rank matrix, i.e., $W = VV^T$ where $V \in \mathbb{R}^{nxk}$, $k = \text{rank}(W)$. Thus, the interaction term in the prediction function (12) can be rewritten as following:

$$\mathbf{x}^T W \mathbf{x} = \sum_{i,j=1}^{d} (v_i^T v_j) \mathbf{x}_i \mathbf{x}_j,$$

where v_i and v_j are the i-th and j-th bins of spectrum x and correspond to representation of i-th and j-th features in the space $\mathbb{R}^k$.

We assume that if i-th and j-th peaks are strongly interacting with each other, then their representation in the vector space should be close, i.e., $||v_i - v_j||_2$ should be small. From this observation, we include the following term, Laplacian smoothness, into the objective function (2.14):

$$\begin{aligned} \mathcal{R}(V) &= \sum_{i,j=1}^{d} A_{ij} ||v_i - v_j||_2^2 && (2.15) \\ &= \text{trace}(V^T L V) && (2.16) \\ &= \text{trace}(VV^T L) = \text{trace}(WL) && (2.17) \end{aligned}$$

where $L = D - A$, D is the degree matrix of A, i.e, $D_{ii} = \sum_{j=1}^{n} A_{ij}, i = 1, .., n$

Thus, we can formulate the following optimization problem, which we call **L-SIMPLE**:

$$\begin{aligned} &\min_{b,w,W} \sum_{i=1}^{n} l(\mathbf{y}_i, \mathbf{f}_i) + \alpha ||w||_1 + \beta ||W||_* + \gamma \text{trace}(WL), \\ &\text{subject to } W \succeq 0 \end{aligned} \tag{2.18}$$

where $\mathbf{f}_i = f(\mathbf{x}_i; w, W)$, γ is an additional hyperparameter to control Laplacian smoothness of the objective function (2.18). It is noteworthy that, our derived formulation is still convex due to convexity of regularization terms, thus, the solution of (2.18) is guaranteed to be the global optimal. Below we will present an optimization method for efficiently solving (2.18).

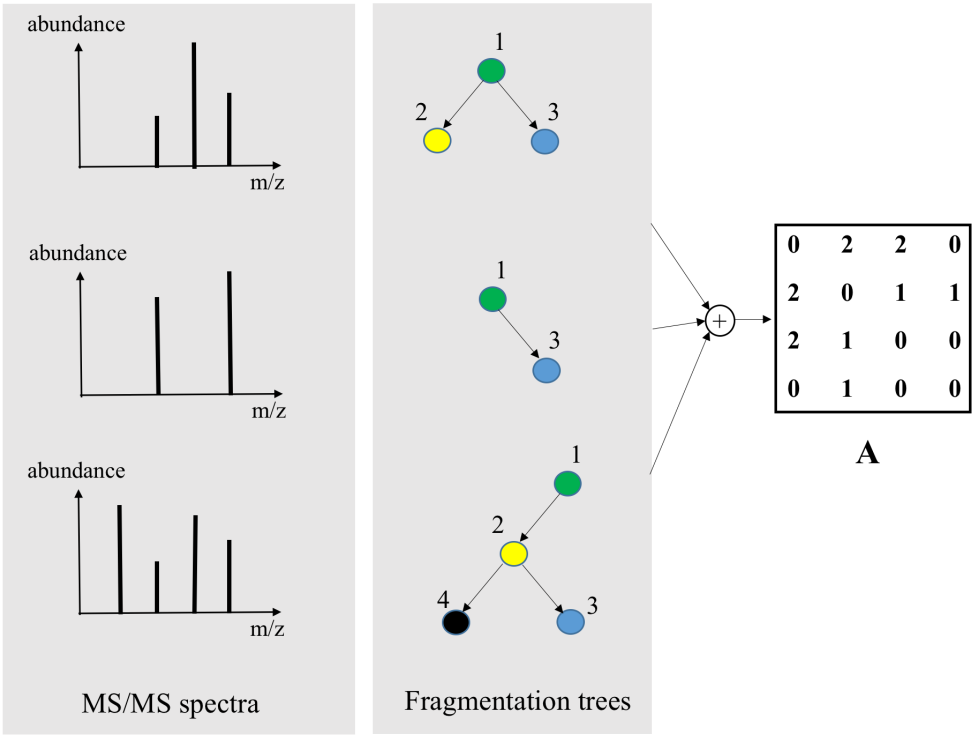

Figure 2.3: Illustration of constructing affinity matrix A from the set of fragmentation trees. The constructed matrix A is used as prior information for regularizing interaction matrix W.

Optimization: ADMM algorithm

(2.18) is convex and guaranteed that the algorithm converges to the global optimal solution. However, it is challenging to solve this problem directly, because terms are nondifferentiable. Our optimization method is based on alternating direction method of multipliers (ADMM). Due to the nondifferentiability of the hinge loss, we introduce the auxiliary variable $\mathbf{C}$, where $\mathbf{C} = (C_1, C_2, ..., C_n)^T$ and $C_i = 1 - \mathbf{y}_i \mathbf{f}_i$, then (2.18) can be reformulated into the following equivalent constrained problem:

$$\begin{aligned} \min_{b,w,W,\mathbf{C}} \quad & \sum_{i=1}^{n} (C_i)_+ + \alpha ||w||_1 + \beta ||W||_* + \gamma \text{trace}(WL) \\ \text{subject to} \quad & \mathbf{C} = \mathbf{1} - \mathbf{YF} \text{ and } W \succeq 0 \end{aligned} \tag{2.19}$$

where $\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_n]^T$.

The augmented Lagrange function of (2.19) is defined by:

$$\begin{aligned}\mathcal{L}(w_0, w, W, \mathbf{C}, \mathbf{u}) = \sum_{i=1}^{n}(C_i)_+ + \alpha||w||_1 + \beta||W||_* + \gamma\text{trace}(WL) \\ + \mathbf{u}^T(\mathbf{1} - \mathbf{YF} - \mathbf{C}) + \frac{1}{2}||\mathbf{1} - \mathbf{YF} - \mathbf{C}||_2^2\end{aligned} \tag{2.20}$$

where $\mathbf{u} = [u_1, u_2, ..., u_n]^T$ is a dual variable corresponding to the constraint $\mathbf{C} = \mathbf{1} - \mathbf{YF}$. Note that the constraint $W \succeq 0$ is not included in (2.20) because this property will be imposed automatically on W after each iteration of updating W, as explained in the Appendix section. We solve the problem of finding the saddle point $(b^*, w^*, W^*, \mathbf{C}^*, \mathbf{u}^*)$ of the augmented Lagrangian function (2.20) through an iterative algorithm between the primal and the dual optimization as follows:

$$\begin{cases} b^{t+1}, w^{t+1} & = \text{argmin}_{b,w}\, \mathcal{L}(b, w, W^t, \mathbf{C}^t, \mathbf{u}^t) \\ W^{t+1} & = \text{argmin}_{W}\, \mathcal{L}(b^{t+1}, w^{t+1}, W, \mathbf{C}^t, \mathbf{u}^t) \\ \mathbf{C}^{t+1} & = \text{argmin}_{\mathbf{C}}\, \mathcal{L}(b^{t+1}, w^{t+1}, W^{t+1}, \mathbf{C}, \mathbf{u}^t) \\ \mathbf{u}^{t+1} & = \mathbf{u}^t + \mathbf{1} - \mathbf{YF}^{t+1} - \mathbf{C}^{t+1} \end{cases} \tag{2.21}$$

where the first three steps update the primal variables based on the current estimate of the dual variable $\mathbf{u}^t$ and the final step updates the dual variable based on the current estimate of the primal variables. Note that the efficiency of ADMM for solving (2.20) depends on whether the subproblems in (2.21) can be solved quickly. Algorithm 1 summarizes the ADMM steps for solving the optimization problem (2.20). To avoid confusion, the detailed derivation of the update rules for subproblems of (2.21) are in Appendix.

2.3.3 Model summary

We here summarize the advantageous features of (L-)SIMPLE:
1) **Peak interactions**: SIMPLE has, in its formulation, the term for peak interactions explicitly, which has not been considered by existing methods, particularly kernel-based methods.

Table 2.1: Micro-average performance of kernels: PPK [Heinonen et al., 2012a] is used to compute $\mathbf{K}_{peak}$. ComUNIMKL, ALIGN, ALIGNF are combinations of $\mathbf{K}_{peak}$ and $\mathbf{K}_{interaction}$ by algorithms UNIMKL, ALIGN, ALIGNF, respectively.

Method	**Acc (%)**	**F1-score (%)**
PPK	75.74($\pm$8.13)	60.59($\pm$13.75)
ComUNIMKL	78.41($\pm$6.82)	65.05($\pm$12.16)
ComALIGN	78.57($\pm$6.24)	65.34($\pm$11.99)
ComALIGNF	**79.03($\pm$7.89)**	**65.67($\pm$13.02)**

2) **Sparse interpretability**: MS spectra are with many zeros and sparse data. We formulate SIMPLE as a sparse model, by which peaks or peak interactions which contribute to improve the predictive performance can be checked and found easily.
3) **Convex formulation**: our formulation keeps SIMPLE (L-SIMPLE) a convex model, which guarantees to find the global optimum. We have developed an alternating direction method of multipliers (ADMM) algorithm, to realize the detection of the global optimum.
4) **No fragmentation trees in prediction**: L-SIMPLE uses fragmentation trees for regularization in training, and they are not inputs, meaning that we do not need fragmentation trees in prediction, which can avoid heavy computational cost of generating fragmentation trees from spectrum.

2.4. Experimental evaluation

Our focus is to incorporate peak interactions of mass spectra into fingerprint prediction, and to build sparse models for model interpretability, and so we conducted experiments to answer the following two questions:

- **(Q1)**: Can peak interactions due to sets of correlated peaks in spectra be used to predict fingerprint vectors more accurately?
- **(Q2)**: For the purpose of interpretation, how to identify a smaller subset of predictors (i.e, peaks or peak interactions) that exhibit the strongest

effects on fingerprints?

2.4.1 Data, preprocessing and evaluation measures

MassBank [Horai et al., 2010] was used. We used the same dataset with 402 compounds as [Shen et al., 2014a], measured by QTOF MS/MS instruments, and followed the same preprocessing steps as in [Shen et al., 2014a]: for each compound, peaks recorded from different collision energies were merged for each MS/MS spectra. Then, spectra were normalized such that the sum of intensities is up to 100%. Peaks merged from different collision energies with m/z difference at most 0.1, using the highest peaks and summing up intensities. In terms of the output for the learning models, molecular fingerprints, which are binary vectors of 528 bits in total, were generated using OpenBabel [O'Boyle et al., 2011]. Fingerprints have a high class imbalance, i.e. mostly +1 or reverse. Thus we only trained models for predicting fingerprints in which the majority class occupies less than 90% of instances.

We used micro-average accuracy, F1 score to evaluate the performance of different methods, computed by taking the average of accuracies and F1 scores over all tasks.

2.4.2 Benefit of incorporating interaction

We show the benefit of incorporating peak interactions using kernel methods described in Section 2.3.1. The MKL algorithms, i.e. UNIMKL, ALIGN and ALIGNF, were used to combine two kernels, $\mathbf{K}_{peak}$ and $\mathbf{K}_{interaction}$ (we call their results ComUNIMKL, ComALIGN, ComALIGNF, respectively) and compared with PPK, which has only kernel for peaks. The combined kernels were coupled with SVM to predict the fingerprint properties. Each property was separately trained by a classifier. Five-fold cross-validation was conducted to seek suitable margin parameter C where $C \in \{2^{-3}, 2^{-2}, ..., 2^{6}, 2^{7}\}$.

Table 2.1 shows the results, in which the micro-average accuracy and F1 score of combined kernels were higher than PPK. This shows that incorporating peak interactions can improve the performance on predicting molecular

Table 2.2: Performance comparison between **SIMPLE** and **L-SIMPLE**.

Task Id/Name	**SIMPLE**		**L-SIMPLE**	
	Acc (%)	**F1** (%)	**Acc** (%)	**F1** (%)
3 (Aldehyde)	71.16	69.24	73.14	70.75
27 (Hydroxy)	91.24	95.19	90.29	94.82
29 (Primary alcohol)	79.36	53.74	79.85	53.98
30 (Secondary alcohol)	80.35	54.18	81.84	56.17
37 (Ether)	80.35	71.39	80.61	72.03
38 (Dialkyl etherEther)	82.35	70.98	82.6	71.16
45 (Aryl)	83.83	81.67	83.34	82.16
50 (Carboxylic acid)	69.38	62.00	69.65	62.15
56 (Primary Carbon)	73.12	40.42	73.88	44.46
57 (Secondary Carbon)	71.88	67.15	72.39	67.28
60 (Alkene)	81.35	23.49	84.08	27.75
Avg±Std	78.33±6.05	66.69±13.03	78.86±5.87	67.59±12.35

fingerprint properties. Additionally, the performance of ALIGNF algorithm was slighly better than the rest.

2.4.3 Benefit of (L-)SIMPLE, sparse interaction models

Since kernel methods in Section 2.3.1 are unable to provide sparse solutions for interpretability, we examined (L-)SIMPLE to gain insight into the models to predict fingerprints. (L-)SIMPLE needs to construct feature vectors from MS/MS spectra: we divided *m/z* range into 500 bins and took accumulated peak intensities in a bin as a feature value to obtain high dimensional vector for each spectrum. These feature vectors were normalized such that all feature values are in [0, 1]. (L-)SIMPLE was trained by ADMM (see algorithm 1) with all variables initialized at zero. The algorithm were iterated until the relative difference in training errors fell below 0.0001 or the number of iterations reaches 100. Five-fold cross-validation was used to evaluate the generalization of the learning machines. Specifically, parameters α, β and γ , for controlling

Table 2.3: Micro-average performance and computation time (for prediction) of kernel-based methods in [Shen et al., 2014a] and proposed methods in this paper.

Method	**Acc** (%)	**F1 score** (%)	**Run. time** (ms)
PPK (Peaks)	75.74(+/-6.72)	60.59(+/-14.54)	52.37
LB (Loss binary)	76.63($\pm$7.03)	61.64($\pm$15.48)	1501.02
LC (Loss count)	75.33($\pm$5.4)	61.25 ($\pm$13.99)	1501.02
LI (Loss intensity)	74.54($\pm$8.49)	58.46($\pm$16.01)	1501.02
NB (Node binary)	79.11($\pm$5.02)	67.34($\pm$11.75)	1501.09
NI (Node intensity)	78.41($\pm$4.99)	66.87($\pm$12.11)	1501.01
CPC (Common path count)	79.02 ($\pm$7.4)	67.55 ($\pm$12.93)	1501.11
ComFT (combining all above)	80.98($\pm$6.05)	69.04($\pm$11.98)	1559.20
ComALIGNF (Proposed:MKL)	79.03($\pm$7.89)	65.67($\pm$13.02)	471.71
SIMPLE (Proposed)	78.33 ($\pm$6.05)	66.70 ($\pm$13.03)	4.57
L-SIMPLE (Proposed)	78.86($\pm$5.87)	67.59 ($\pm$12.35)	4.32

sparsity, low-rankness and Laplacian smoothness were chosen from the lists: $\{1, 2, 3\}$, $\{2, 3, 4, 5\}$ and $\{0.1, 0.5, 1.0\}$, respectively.

Also, to evaluate the effects of adding the Laplacian smoothness term into the objective function, we compared the accuracy and F1 score over tasks by performing five-fold cross-validation. Table 2.2 shows the accuracy and F1 score obtained for the first ten tasks (Note that we used only tasks in which the majority class occupies less than 90% of instances). The micro-average over all tasks is also displayed at the bottom. We can see that Laplacian regularization worked to make SIMPLE more robust, resulting in better predictive performance of L-SIMPLE.

We further compared (L-)SIMPLE with various kernel, namely PPK, NB,NI, LB, LC, LI, CPC and their combination, which we call ComFT, all from [Shen et al., 2014a]. While these kernels (except for PPK) are all computed from the fragmentation trees, in which the cost for converting MS/MS spectra to these trees is heavy and time-consuming, our method uses peaks from spectra only

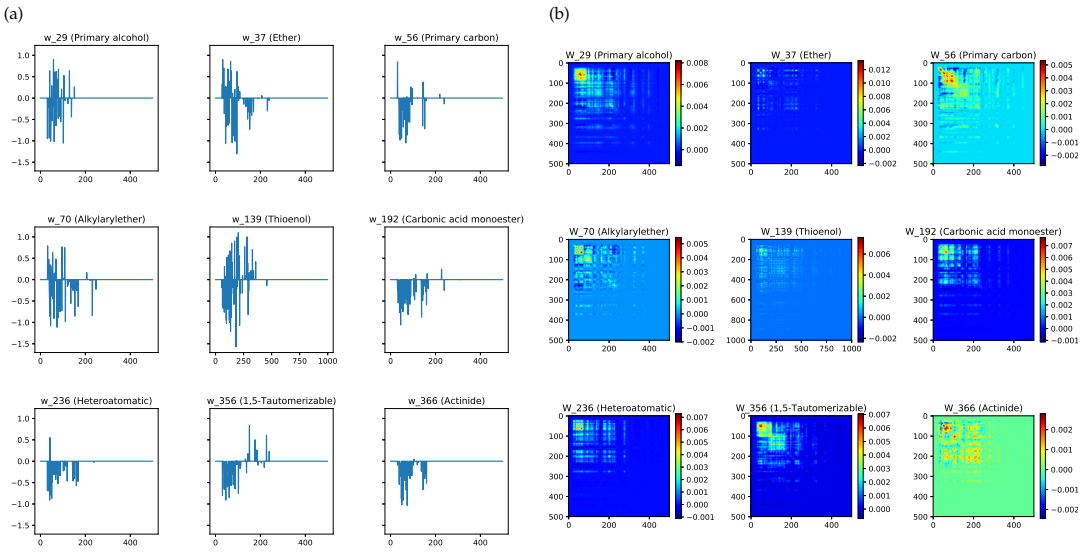

Figure 2.4: (a) Weight vectors (w) of the main effect terms and (b) smooth heat map of weight matrices (W) of the interaction terms learned by L-SIMPLE for properties or tasks: 29 (Primary alcohol), 37 (Ether), 56 (Primary Carbon), 70 (Alkylarylether), 139 (Thioenol), 192 (Carbonic acid monoester), 236 (Heteroaromatic), 356 (1,5-Tautomerizable) and 366 (Actinide).

and is efficiently computable in prediction.

Table 2.3 shows the results of accuracy, F1 scores and computation time for prediction by all compared methods. The prediction time (in milliseconds) was averaged over all spectrum in the data set. The first four methods including PPK achieved around the accuracy of 75%, which is clearly worse than the other methods, which achieved around 78% to 80% and are very comparable each other. In fact ComFT, the current cutting-edge MKL-based method, performed best in both accuracy and F1 score, while the second best was not clear (NB by accuracy and L-SIMPLE by F1 score). On the other had, about computation time, FT-based methods, i.e. from LB to ComFT, were clearly slower then the others, because they need convert spectra into FTs. In fact ComFT needed more than 1500 ms, which is more than 300 times slower than that of (L-)SIMPLE, which just spent only less than 5ms. This is a sizable difference when we have to process a huge amount of spectra produced by the current high-throughput MS/MS. We stress that the performance difference between (L-)SIMPLE and

ComFT was very slight and statistically insignificant, while (L-)SIMPLE was exceedingly faster than ComFT.

2.4.4 Model interpretation

One advantage of sparse learning models over kernel based methods is interpretation. For illustration purposes, Figure 2.4 shows the weights of main effect terms (w) and the interaction weight matrix (W) obtained by **L-SIMPLE** for nine fingerprint properties (or tasks): 29, 37, 56, 70, 139, 192, 236, 356 and 370 (these are randomly selected for investigation). As observed, the weight of main effect and interaction terms were different between properties, suggesting that different properties are strongly affected by different subsets of a few peaks in spectra.

Table 2.4 shows case studies to illustrate the effects of peak interactions. We consider four interaction pairs frequently present in these tasks. w_1 and w_2 denote the weights for peaks and W_{12} denotes the weight for their interactions. We can raise the following three interesting findings:

1. Either w_1 or w_2 (or both) can be zero but the interaction weights are often nonzero: For example, W_{12} of interaction (42, 85) are mostly nonzero, while w_1 and w_2 are both zero with respect to tasks 29, 37, 56, 70. This means that individual peaks are not good predictive descriptors of properties, while their interactions are. Thus this result clearly shows the importance of considering peak interactions in fingerprint prediction.

2. Despite of negative impacts of individual peaks, their interactions can be positive: For example, interaction (85, 227) with respect to tasks 29, 56 and 366. This means that while individual peaks indicate the absence of a property, the interactions of two peaks mean the presence of the property.

3. Interaction of peaks can be zero, indicating these peaks are independent of each other even if their corresponding weights are nonzeros: For example, interaction (42, 85) with respect to task 139.

They are just part of numerous findings, but even these examples show a fact that even individual peaks are not good indicators of some fingerprints,

their interactions with others may significantly contribute to the prediction. This again confirms the importance of peak interactions for fingerprint prediction.

Algorithm 1 ADMM algorithm for optimizing the optimization problem (2.20).

Inputs:
A set of MS/MS spectra $\mathbf{X} = [\mathbf{x}_1^T, \mathbf{x}_2^T, ..., \mathbf{x}_n^T]^T$ and associated output $\mathbf{Y} = [y_1, y_2, ..., y_n]^T$. Laplacian L (only for **L-SIMPLE**).

Outputs:
weight vector w and interaction matrix W.

Initialize:
$b \leftarrow 0, w \leftarrow \mathbf{0}, W \leftarrow \mathbf{0}, \mathbf{C} \leftarrow \mathbf{0}, \mathbf{u} \leftarrow \mathbf{0}$

while not converged **do**

1. Fix W, **C**, **u** and update b, w

Precompute $\hat{\mathbf{F}}_1 \leftarrow \mathbf{Y}(\mathbf{1} - \mathbf{C} + \mathbf{u}) - \text{diag}(\mathbf{X}W\mathbf{X}^T)$

while not converged **do**

$$\begin{cases} \mathbf{z} \leftarrow w - \rho\mathbf{X}^T(\mathbf{X}w + b - \hat{\mathbf{F}}_1) \\ w \leftarrow \mathcal{S}_\alpha(z) \\ b \leftarrow \frac{1}{n}\text{sum}(\hat{\mathbf{F}}_1 - \mathbf{X}w) \end{cases}$$

end while

2: Fix b, w, **C**, **u** and update W

precompute $\hat{\mathbf{F}}_2 \leftarrow \mathbf{Y}(\mathbf{1} - \mathbf{C} + \mathbf{u}) - b - \mathbf{X}w$

while not converged **do**

$$\begin{cases} \mathbf{R} \leftarrow \text{diag}(\mathbf{X}W\mathbf{X}^T) - \hat{\mathbf{F}}_2 \\ \Delta_W\mathcal{L} \leftarrow \mathbf{X}^T\mathbf{R}\mathbf{X} \\ \mathbf{Z} \leftarrow W - \rho(\Delta_W\mathbf{L} + \gamma L) \\ \mathbf{U}, \mathbf{E} \leftarrow \mathbf{EVD}(\mathbf{Z}) \\ \hat{\mathbf{E}} \leftarrow \mathcal{S}_\beta(\mathbf{E}) \\ W \leftarrow \mathbf{U}\hat{\mathbf{E}}\mathbf{U}^T \end{cases}$$

end while

3: Update **C**

$\mathbf{F} = b + \mathbf{X}w + \text{diag}\mathbf{X}W\mathbf{X}^T$, $\mathbf{C} \leftarrow \mathcal{T}_1(\mathbf{1} - \mathbf{Y}\mathbf{F} + \mathbf{u})$

4: Update the dual variable **u**

$\mathbf{u} \leftarrow \mathbf{u} + \mathbf{1} - \mathbf{Y}\mathbf{F} - \mathbf{C}$

end while

Table 2.4: Case studies of weight vector w and interaction matrix W learned by **L-SIMPLE** over a set of randomly selected tasks. w_1 and w_2 denote weights corresponding two mass positions and W denotes the weight of their interactions. Four pairs of mass positions which are frequently present in these tasks, including (42, 85), (42, 163), (85, 227) and (130, 201) are shown.

Tasks/ Name	**(42, 85)**			**(42, 163)**			**(85, 227)**			**(130, 201)**		
	w_1	w_2	W_{12}	w_1	w_2	W_{12}	w_1	w_2	W_{12}	w_1	w_2	W_{12}
29 (Primary Alcohol)	0.0	0.0	0.0016	-0.0545	0.0	0.0442	-0.4085	-0.0545	0.0765	-0.4085	0.0	0.0218
37 (Ether)	0.0	0.0	0.0120	0.0260	0.0	0.0471	0.0	0.0260	0.1264	0.0	0.0	0.3389
56 (Primary Carbon)	0.0	0.0	0.0271	-0.5657	0.0	0.0047	-0.9833	-0.5657	0.0271	-0.9833	0.0	0.0104
70 (Alkylarylether)	0.0	0.0	0.0159	0.0	0.0	0.0551	-0.1238	0.0	0.0972	-0.1238	0.0	0.0265
139 (Thioenol)	0.1575	0.3939	0.0	-0.2308	-0.4094	0.0	-0.3589	-0.2308	0.0	-0.3589	-0.1126	0.0
192 (Carbonic acid monoester)	-0.1542	0.0	0.0	0.0	0.0	0.0	-0.6945	0.0	0.0	-0.6945	0.0	0.0
236 (Heteroaromatic)	0.0	0.0	0.0107	0.0	0.2499	0.0537	-0.3069	0.0	0.1062	-0.3069	0.1401	0.0201
356 (1,5-Tautomerizable)	0.0	0.0	0.0170	0.0	0.0	0.0301	0.5539	0.0	0.0607	0.5539	0.0	0.0107
366 (Actinide)	0.0	0.0	0.0245	-0.1891	0.6065	0.0282	-0.5373	-0.1891	0.0399	-0.5373	0.0	0.0153

2.5. Summary

The goal of this chapter is to propose machine learning models which are able to incorporate peak interactions for fingerprint prediction. Our experiments showed that peak interactions are definitely useful to improve fingerprint prediction, along with discriminative information about peaks.

Our first model is based on kernel learning, defining two kernels, one for peaks and the other for peak interactions, which are combined through MKL. Again we note that Shen et al. [2014a] used fragmentation trees for prior structural information of spectral, and converting spectra into fragmentation trees is definitely computation- ally expensive. On the other hand, our model of kernel learning uses only peaks in the spectrum as input for prediction, indicating that our model is much more efficient. Kernel learning does not have to construct feature vectors for spectra, and instead this is done implicitly by kernels defined, which can avoid any error caused when generating feature vectors. However, a big issue of kernel learning is interpret- ability. That is, it is difficult for kernel learning to figure out which subset of peaks or peak interactions exhibit the strongest effects on fingerprint prediction, despite that clearly each property depends on a very few number of mass positions in each given spectrum.

Our sparse interaction models, (L-)SIMPLE, have a number of advantages: (L-)SIMPLE is formulated as a sparse, convex optimization model, which can capture peak interactions and also give interpretable solutions. We emphasize that next generation fingerprint prediction needs a ML model, which should learn, from huge but sparse spectra, peaks as well as peak interactions comprehensibly and predict fingerprints against again huge spectra highly efficiently. (L-)SIMPLE would be a reasonable solution for this situation.

Chapter 3

Learning data-dependent, concise molecular vector for fast, accurate metabolite identification from tandem mass spectra

3.1. Introduction

Kernel methods have been shown effective for fingerprint prediction, such as FingerID [Heinonen et al., 2012a], CSI:FingerID [Dührkop et al., 2015a] and Input Output Kernel Regression (IOKR, Brouard et al. [2016b]). In particular, IOKR is recognized as the current cutting-edge method for metabolite identification due to the following advantages: 1) structures (e.g. feature interaction in the molecular fingerprint vectors) in the output can be incorporated into the learning model by the kernel defined in the output space, leading to accuracy improvement; 2) fingerprints are simultaneously predicted by the learned model, rather than being considered as a set of separate tasks, resulting in faster computation. However, still IOKR has a limitation of using molecular fingerprints as intermediate representation vectors between spectra and chemical structures of the two-step based methods. These molecular fingerprint vectors have some drawbacks: they should be large in size to encode all possible

substructures and chemical properties related with metabolites, causing slow prediction in the candidate retrieval step; they are neither necessarilly specific to any task nor data, and therefore redundant in the sense that they might contain much information irrelevant to the task and data, resulting in limited predictive performance.

Motivated by the drawbacks of molecular fingerprints used in existing methods for metabolite identification, we propose a ML framework, named ADAPTIVE, which allows to generate representations for molecules using their chemical structures, which we call molecular vectors to distinguish with traditional binary molecular fingerprints. These vectors are specific to both data and the task of metabolite identification and, therefore non-redundant. In a technical detail, ADAPTIVE has two subtasks in the learning step: 1) learning a mapping from chemical structures to molecular vectors and 2) learning a mapping from spectra to molecular vectors. Figure 3.1 shows a schematic picture of ADAPTIVE, where the left and right blue boxes correspond to the first and second subtasks, respectively. In Subtask 1, ADAPTIVE learns a model to generate molecular vectors for metabolites using their chemical structures, where these vectors are specific to both data and the task of metabolite identification and, therefore non-redundant. The model in Subtask 1 is parameterized a model, named a message passing neural network (MPNN) for mapping chemical structures of molecules to the molecular vectors. The **main contribution** of this paper is in the Subtask 1, that is, to learn the correspondence between given pairs of spectra and structures for metabolites. Thus, the parameters of MPNN are trained so that the correlation between the spectra and vectors mapped from the structures is maximized. We use Hilbert-Schmidt Independence Criterion (HSIC, Gretton et al. [2005]) for evaluating the correlation, due to its theoretically nice properties and kernel based calculation. Specifically, we formulate an objective function for the maximization problem through HSIC and solve this problem to have the best molecular vectors adapted to given data. For Subtask 2, ADAPTIVE uses IOKR to learn a mapping from spectra to molecular vectors generated by the Subtask 1, since IOKR is the current cutting-edge method for this task.

We emphasize that the key difference between ADAPTIVE and the origi-

nal IOKR is that IOKR uses "manually-designed" fingerprints, which are large in size, possibly redundant and non-specific to metabolite identification (and given data), while ADAPTIVE learns representations for metabolites from given data, as molecular vectors, resulting in that the molecular vectors generated by ADAPTIVE are data-driven and concise.

In order to validate the performance of ADAPTIVE, we conducted extensive experiments using a benchmark data. Experimental results showed the following two main advantages of ADAPTIVE over existing methods, including the original IOKR:

- **Predictive performance**.

  ADAPTIVE achieved the best performance, followed by IOKR, CSI:FingerID and FingerID. For example, the top-20 accuracy of ADAPTIVE was 78.52% with the parameters of Gaussian kernel, ALIGNF and molecular vector size of 300. On the other hand, IOKR, CSI:FingerID and FingerID achieved 74.79%, 73.07% (or 68.20%) and 58.17%, respectively, using Gaussian kernel (for IOKR) and ALIGNF. The top-k accuracy was computed by the average over all trials of 10-fold cross-validation, and so the performance advantage of ADAPTIVE was significant and very clear.

- **Computational efficiency for prediction**.

  Under the same experimental setting, ADAPTIVE was four to seven times faster than IOKR, which was already known as the fastest method. We can then say that ADAPTIVE is the current fastest method while keeping the highest predictive performance for metabolite identification.

3.2. Related work

As mentioned in Introduction, fingerprint prediction is important in supervised learning for metabolite identification, because we can retrieve metabolite candidates more reliably if fingerprints are predicted more accurately. For fingerprint prediction, kernel learning has been shown to be the most powerful approach. For example, a typical approach, FingerID [Heinonen et al., 2012a]

uses probability product kernel (PPK, Jebara et al. [2004a]), which can be directly computed from spectra and runs support vector machine with this kernel for solving fingerprint prediction as a classification problem. CSI:FingerID [Dührkop et al., 2015a], an extension of FingerID, uses not only spectra but also fragmentation trees (FTs, Rasche et al. [2011]) as input to generate kernels over spectra and FTs, which are then combined via multiple kernel learning (MKL, Gönen and Alpaydin [2011]). FTs may capture structural information behind spectra which is missing in the approach of FingerID. This is the motivation of CSI:FingerID. However, the computational cost for converting FTs from MS/MS spectra is very expensive, leading to heavy computational load, which causes a problem particularly in prediction. Thus we can say that kernel-based supervised learning, particularly complex kernels, have a computation issue, regardless of high performance in prediction.

Among the series of kernel-based approaches, Input-Output Kernel Regression (IOKR, Brouard et al. [2016b]) has been shown to outperform the previous methods, in terms of both predictive performance and computational speed. That is, simply IOKR is the cutting-edge kernel-based method for metabolite identification. IOKR learns mapping from spectra, i.e. input $\mathcal{X}$, to molecular fingerprints (or structures behind fingerprints), i.e. output $\mathcal{Y}$. In order to do this mapping, IOKR defines kernels to encode similarities in the input space (e.g., spectra and/or FTs) and the output space (molecular fingerprints or structures). Then the advantage of IOKR comes from the following two points: 1) unlike previous kernel-based methods, IOKR handles the structured output space by the kernel defined for the output, which improves the predictive performance. 2) IOKR simultaneously predicts fingerprints rather than considering fingerprint prediction as another task, leading to an efficient computation in prediction. Some part (mapping from spectra to feature vectors) of IOKR is a part of ADAPTIVE, and so further technical details of the corresponding part of IOKR is described more in Section 3.3.

Conventionally molecular fingerprints for fingerprint prediction have been manually-designed feature vectors, to encode a predefined set of substructures or chemical properties, which are possibly found in metabolites. However recently, machine learning-based (or data-driven) algorithms for generating fin-

gerprints have been proposed. A typical approach is Neural Fingerprint (NFP, Duvenaud et al. [2015]), which takes graphs with arbitrary sizes and shapes as inputs. NFP uses the idea of *graph convolution*, an extension of convolution operation from multi-dimensional arrays, like images or texts, to graph structures. NFP is then trained in a supervised manner by using available labels, such as log Mol/L for solubility, EC_{50} for drug efficacy. Finally NFP results in fingerprint vectors (for molecules) specific to given task and data. An extension of NFP is for unsupervised (as well as semi-supervised) settings to learn representations of molecular graph without labels [Nguyen et al., 2017], since label information can be experimentally obtained and precious. More recently, Gilmer et al. [2017] showed that several graph convolution-based models, including NFP, Gated Graph Neural Networks [Li et al., 2015], spectral graph convolutional network [Kipf and Welling, 2016], etc., can be formulated in an unified model, namely Message Passing Neural Network (MPNN), with the following three functions: *message passing*, *update* and *readout*. A key advantage of MPNN is that defining the above components generates a proper model for learning graphs, depending on a given task. Also another advantage of MPNN as well as other neural network-based methods in this paragraph use differentiable operations and thus their parameters can be effectively trained by using a stochastic gradient descent algorithm.

3.3. Methods

3.3.1 ADAPTIVE: Overview

We first introduce the framework of ADAPTIVE for metabolite identification. This is also the general framework of approaches using machine learning for metabolite identification. It has two subtasks: **Subtask 1**) learning a function which maps metabolites from their structures to molecular vectors and **Subtask 2**) learning a function which maps metabolites from spectra to the vectors generated in Subtask 1. Fig. 3.1 shows an illustration of the entire framework of ADAPTIVE. In this figure, the left and right blue boxes correspond to Subtasks 1 and 2, respectively.

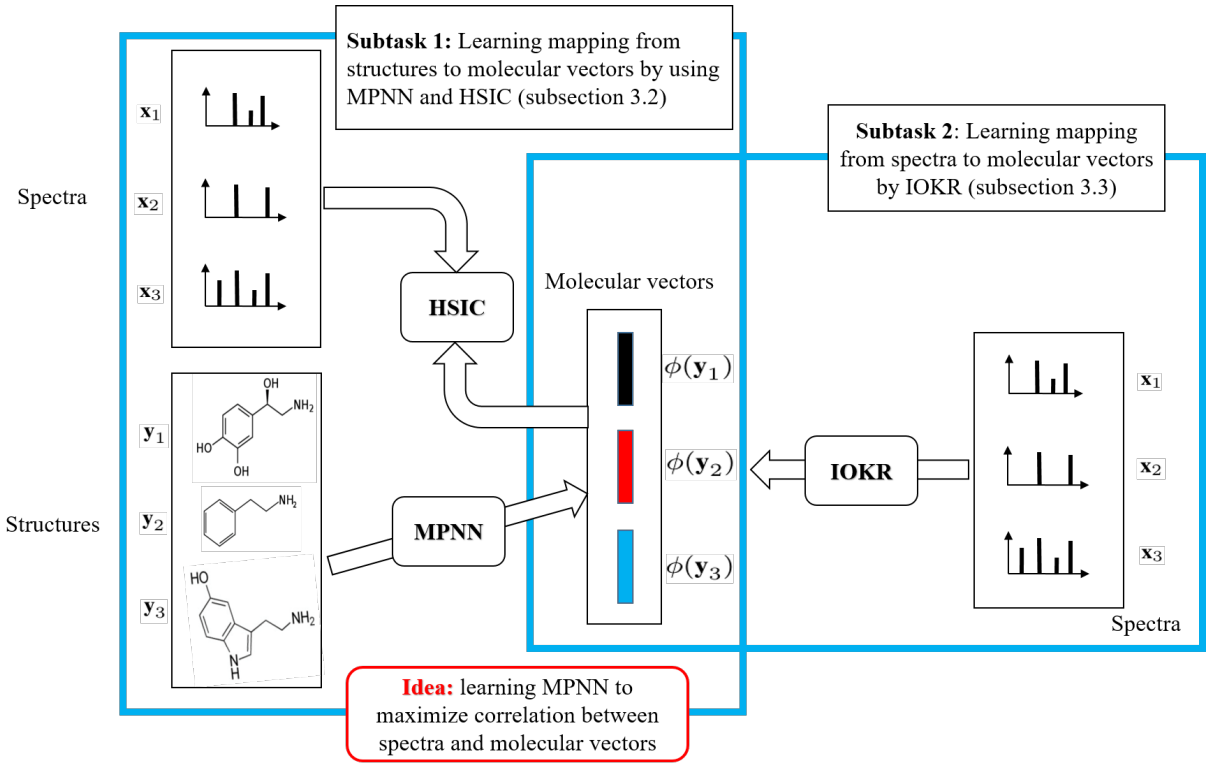

Figure 3.1: Overview of ADAPTIVE for metabolite identification. ADAPTIVE has two components: 1) Subtask 1: estimates parameters of a function mapping metabolites from structures to molecular vectors, given a set of spectra-structure pairs; 2) Subtask 2: learns a function mapping from spectra to molecular vectors (generated by Subtask 1), given a set of spectrum-vector pairs.

For Subtask 1, given pairs of metabolite structure-spectrum, we estimate parameters of a function mapping metabolites from their structures to molecular vectors by maximizing the correlation between the vectors mapped from the structures and also the corresponding spectra. In more detail, we model the mapping function by message passing neural network (MPNN) and evaluate the correlation between the vectors and spectra by using Hilbert-Schmidt Independence Criterion (HSIC), due to the computational simplicity and provably theoretical properties of HSIC. For Subtask 2, we simply borrow the corresponding part of IOKR, which is the current cutting-edge method for metabolite identification, to learn a function mapping metabolites from spectra to vec-

tors generated by Subtask 1.

We explain these two subtasks in the following subsections, being followed by the subsection on kernels we used in ADAPTIVE.

3.3.2 Subtask 1: learning molecular vectors for metabolites via Hilbert-Schmidt Independence Criterion (HSIC)

For this subtask, we need to estimate a function to map metabolites from structures to molecular vectors, given spectrum-structure pairs. For this problem, we use Message Passing Neural Network (MPNN) as the mapping function, which can extract meaningful representation for graphs (molecules for our problem) by supervised learning from training data. That is, MPNN requires labeled training data, which are, however, unavailable for this subtask. Then we manage this problem by taking advantage of given spectrum-structure pairs. We estimate parameters of MPNN by using the idea of maximizing the correlation between the given spectra and vectors (mapped from structures). The correlation is evaluated by Hilbert-Schmidt Independence Criterion (HSIC). We describe the detail of MPNN, HSIC and related optimization procedures in the following subsections.

Message Passing Neural Network (MPNN, Gilmer et al. [2017])

MPNN is a framework, which takes graphs of arbitrary sizes and structures as inputs, to learn their representation vectors at different levels (i.e. nodes, subgraphs and the whole graph) in a supervised manner. A key advantage is that MPNN allows to learn features specific to the given task from the given data. Below we explain the procedure of MPNN.

First let G be an undirected graph, and v and vw be a node (atom in molecules) and an edge (bond in molecules), respectively. Each node v is assigned with *state vectors* at different levels, where each level represents a substructure (or subgraph) rooted at the corresponding node, denoted by h_v^r, where r shows a level. We can compute state vector h_v^r as well as *message* m_v^r in a hierarchical manner, by using the following two functions: *message passing* (3.1) and *update*

(3.2):

$$m_v^{r+1} = \sum_{w \in \mathcal{N}(v)} H_{e(vw)}^r h_w^r, \tag{3.1}$$

$$h_v^{r+1} = g(h_v^r + m_v^{r+1}), \tag{3.2}$$

where $\mathcal{N}_v$ denotes the set of neighbors of node v in graph G; $e(v, w)$ indicates the type of edge between two nodes v and w (this edge type is like a single, double, triple or aromatic bond); $H_{e(vw)}^r$ is a (square) weight matrix to be learned, specific to the edge type $e(vw)$ at the r^{th} level; g is a nonlinear activation function (e.g., ReLU or sigmoid).

Intuitively, the *message passing* function (3.1) on node v plays the role of collecting information from the neighbors of node v and *update* function (3.2) on node v is to update the state of node v based on the collected information and the former state of node v. Thus, by applying two functions (3.1) and (3.2) multiple times, the updated features at node v (e.g., h_v^{r+1}) can be used to represent a certain number of substructures with the root of node v. Then, the values for these series of substructures can be used to generate a vector at node v with different levels (sizes) of substructures. Figure 3.2 shows a schematic and illustrative picture of this procedure (Figure 3.2 is from Nguyen et al. [2017]).

After obtaining the state vectors of substructures rooted at node v, i.e. h_v^r, we have the *readout* phase to combine all vectors at different levels into a single representation vector of the whole molecule (namely, neural fingerprints). Figure 3.3 shows a schematic picture of summing up the state vectors at different levels. As in Duvenaud et al. [2015], we adopt the softmax operation on the states and then perform linear projections (parameterized by different weight matrices W_r) and finally sum them up to obtain a single vector over different levels which represents the whole graph. In short, the molecular vector for the entire molecule can be written as following:

$$\sum_r \sum_v \text{softmax}(W_r h_v^r). \tag{3.3}$$

We note that operations are all differentiable with respect to parameters, which makes learning the parameters possible, given an objective function, by a stochastic or minibatch gradient descent algorithm. Algorithm 1 shows a pseudocode of the procedure of repeating the *message passing* and *update* functions.

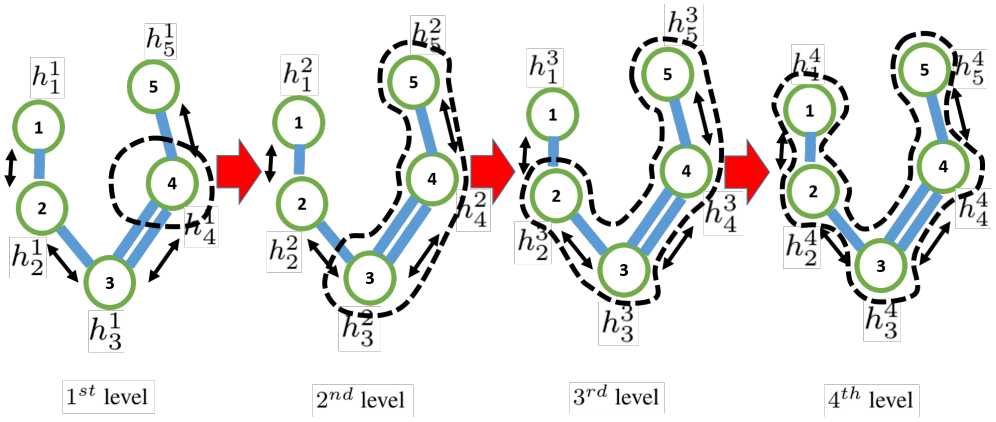

Figure 3.2: Message passing and update functions are used to represent rooted substructures in a hierarchical manner. At the first level (left-most graph), each node is represented by feature vector, with only information of the node itself. We note that by repeatedly applying message passing and update functions (from left to right), more neighboring information are incorporated. For example, the updated feature (2nd level) has information on nodes 3 and 5, and then 3rd level has that on nodes 2 to 5. Finally, the whole graph is covered.

Hilbert-Schmidt Independence Criterion (HSIC)-based objective function

We estimate parameters of MPNN by maximizing the correlation (dependency) between given spectra and molecular vectors. A lot of measures can be used to evaluate and estimate the correlation, while we use Hilbert-Schmidt Independence Criterion (HSIC) due to its theoretically sound properties. More importantly, estimation of HSIC is based on kernel calculation, which can effectively deal with the uncertainty of peaks in spectra caused by measurement errors.

Formally, we are given dataset $\mathcal{D} = (\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, where $\mathbf{x}_i, \mathbf{y}_i$ are spectrum and molecular structure, respectively, of the i^{th} metabolite. First, for the spectra, i.e. $\mathbf{x}$, we consider kernels which combine spectra with fragmentation trees, namely $k(\mathbf{x}_i, \mathbf{x}_j)$. We describe the detail of the kernels for spectra in Section 3.3.4. Then, given the kernel over $\mathbf{x}$ is fixed, the goal is to learn function $\phi : \mathcal{Y} \mapsto \mathcal{F}_d$ from $\mathcal{D}$ such that the correlation between the input and output is maximized. The $\phi(\mathbf{y})$ is the output of MPNN (or molecular vectors) which

Algorithm 2 Message Passing Neural Network (MPNN).

Inputs:
  minibatch of molecular structures $\mathbf{Y}_b = \{\mathbf{y}_i\}_{i=1}^{B}$, radius R
  weight matrices of edges: $\mathrm{H}_1^1, \mathrm{H}_2^1, ..., \mathrm{H}_4^R$,
  weight matrices of *readout* function: $\mathrm{W}_1, \mathrm{W}_2, ..., \mathrm{W}_R$
Outputs:
  molecular vectors $\phi(\mathbf{Y}_b)$
for $i \leftarrow 1$ to B **do**
  for each atom v in $\mathbf{y}_i$ **do**
    $h_v \leftarrow$initial hidden rep. vector of v ▷ atom feature
  end for
  $\phi_i \leftarrow 0_d$ ▷ Initialize each molecular vector with a zero vector
  for $r \leftarrow 1$ to R **do**
    for each node v in $\mathbf{y}_i$ **do**
      $m_v^{r+1} = \sum_{w \in \mathcal{N}(v)} \mathrm{H}_{vw}^r h_w^r$ ▷ *message* function
      $h_v^{r+1} = g(h_v^r + m_v^{r+1})$ ▷ *update* function
      $\phi_i = \phi_i + \mathrm{softmax}(\mathrm{W}_{r+1} h_v^{r+1})$ ▷ *readout* function
    end for
  end for
end for
$\phi(\mathbf{Y}_b) = [\phi_1, \phi_2, ..., \phi_B]$

belongs to space $\mathcal{F}_d$. The linear kernel function induced by this space can be written as follows:

$$l(\mathbf{y}_i, \mathbf{y}_j) = \langle \phi(\mathbf{y}_i), \phi(\mathbf{y}_j) \rangle \tag{3.4}$$

To evaluate the correlation between spectra and molecular vectors (output of MPNN), we use an unbiased empirical estimate of Hilbert-Schmidt Independence Criterion (HSIC, Gretton et al. [2005]), which can be given as follows:

$$\begin{aligned} \mathrm{uHSIC}(\mathcal{X}, \mathcal{Y}) = \frac{1}{n(n-3)} [\mathrm{trace}(\bar{\mathrm{K}_n}\bar{\mathrm{L}_n}) + \frac{\mathbf{1}_n^\top \bar{\mathrm{K}_n} \mathbf{1}_n \mathbf{1}_n^\top \bar{\mathrm{L}_n} \mathbf{1}_n}{(n-1)(n-2)} \\ - \frac{2}{n-2} \mathbf{1}_n^\top \bar{\mathrm{K}_n} \bar{\mathrm{L}_n} \mathbf{1}_n], \end{aligned} \tag{3.5}$$

where $\bar{\mathrm{K}_n} = \mathrm{K}_n - \mathrm{diag}(\mathrm{K}_n)$ denotes the kernel matrix for the set of n spectra

$\mathcal{X}$ with diagonal elements set to zero; $\mathbf{1}_n$ is a vector of 1s of n dimensions. Likewise $\bar{\mathrm{L}_n} = \mathrm{L}_n - \mathrm{diag}(\mathrm{L}_n)$, where L_n is the kernel matrix of n molecular vectors output by MPNN. By arranging terms in (3.5), we can rewrite (3.5) as the objective function to learn parameters as follows:

$$\mathrm{uHSIC}(\mathcal{X}, \mathcal{Y}) = \mathrm{trace}(\mathrm{S}_n \bar{\mathrm{L}_n}) \tag{3.6}$$

where

$$\mathrm{S}_n = \frac{1}{n(n-3)}[\bar{\mathrm{K}_n} + \frac{\mathbf{1}_n \mathbf{1}_n^\top \bar{\mathrm{K}_n} \mathbf{1}_n \mathbf{1}_n^\top}{(n-1)(n-2)} - \frac{2}{n-2}\mathbf{1}_n \mathbf{1}_n^\top \bar{\mathrm{K}_n}] \tag{3.7}$$

However, directly optimizing (3.6) is prohibitively expensive in computation, particularly for large-scale data, since the complexity reaches $\mathrm{O}(n^2)$, both in space and time. In order to overcome this limitation, following Zhang et al. [2018], we disjointly divide samples $(\mathcal{X}, \mathcal{Y})$ into n/B blocks with the size of B, $\{\{(\mathbf{x}_i^{(b)}, \mathbf{y}_i^{(b)})\}_{i=1}^{B}\}_{b=1}^{n/B}$ and then apply HSIC on each block independently. An empirical estimate of the unbiased block HSIC can be defined by:

$$\mathrm{ubHSIC}(\mathcal{X}, \mathcal{Y}) = \frac{1}{n/B} \sum_{b=1}^{n/B} \mathrm{trace}(\mathrm{S}_b \bar{\mathrm{L}}_b), \tag{3.8}$$

where S_b can be defined by a similar manner to (3.7), and $\bar{\mathrm{L}}_b$ is the kernel matrix for the b^{th} block.

Furthermore, in order to avoid the effect by biased partition of the dataset, following Yamada et al. [2018], we repeat shuffling dataset T times, compute ubHSIC on each permutation and take the average over them. HSIC by this procedure is known as bagging block HSIC, which can be written as follows:

$$\mathrm{ubHSIC}(\mathcal{X}, \mathcal{Y}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{n/B} \sum_{b=1}^{n/B} \mathrm{trace}(\mathrm{S}_{t,b} \bar{\mathrm{L}}_{t,b}), \tag{3.9}$$

We use (3.9) as objective function J to learn parameters.

Optimization algorithm

An advantage of objective function (3.9) is that we can use the gradient descent (minibatch gradient descent) for estimating parameters of MPNN. We

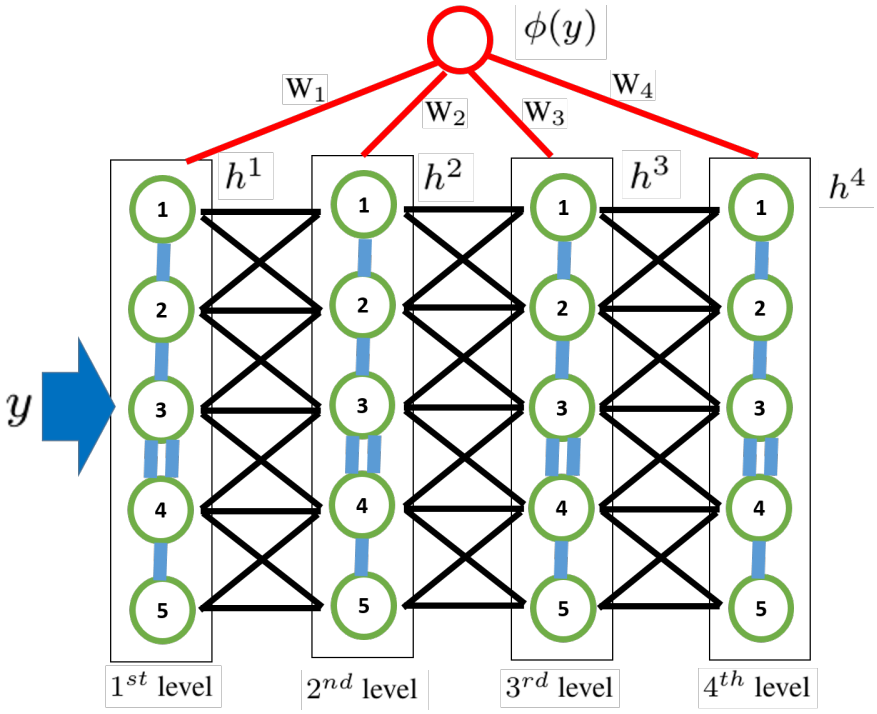

Figure 3.3: Representation vectors of substructures, which are rooted at nodes, are computed from the input graph by the message passing and update functions. These functions contribute to computing the molecular representation vector of the whole molecule.

.

here explain details on how to conduct the minibatch gradient descent procedure for the HSIC-based loss, which has three steps.

Step 1: Feed forward and loss calculation.
For samples of size n, at each iteration, we perform random permutation and then split all samples into batches, where the size of each batch is B. Batches are sequentially fed into MPNN. The output of MPNN for the b^{th} batch at the t^{th} iteration is denoted by $\phi(\mathbf{Y}_{t,b}) = (\phi(\mathbf{y}_1^{t,b}), \phi(\mathbf{y}_2^{t,b}), ...\phi(\mathbf{y}_B^{t,b}))$. Then using these outputs, the objective function on the whole samples can be calculated as in (3.9).

Step 2: Gradient calculation of the loss layer.

As we can compute the loss directly with the output of MPNN (i.e., $\phi(\mathbf{Y}_{t,b})$), we need to compute the gradient of J with respect to $\phi(\mathbf{Y}_{t,b})$. Suppose that the output of MPNN is already normalized, i.e. $\phi(\mathbf{y})^\top \phi(\mathbf{y}) = 1$ for all $\mathbf{y} \in \mathcal{Y}$, the gradient can be obtained by the following:

$$\frac{\partial J}{\partial \phi(\mathbf{Y}_{t,b})} = \frac{B}{Tn} \mathrm{S}_{t,b} \phi(\mathbf{Y}_{t,b}) \tag{3.10}$$

Step 3: Gradient calculation of the MPN and weight update.
Having calculated the gradient of J, i.e. (3.10), the next step is to compute the gradient of $\phi(\mathbf{Y}_{t,b})$ with respect to model parameters θ, namely $\frac{\partial \phi(\mathbf{Y}_{t,b})}{\partial \theta}$, to update the whole parameters for each batch at each step.

Algorithm 2 is a pseudocode of the entire algorithm of learning parameters of MPNN.

3.3.3 Subtask 2: learning a mapping from spectra to molecular vectors by Input Output Kernel Regression (IOKR)

For Subtask 2, we use Input Output Kernel Regression (IOKR). That is, we learn a mapping from spectra to molecular vectors generated in Subtask 1 by using IOKR. Again, we explain two technical reasons why we use IOKR for this mapping below: 1) IOKR allows to incorporate the structures behind outputs, such as feature interactions in molecular vectors, into the learning model, by which the prediction accuracy can be improved. 2) Furthermore, all features in molecular vectors are predicted simultaneously, which is not like separate tasks in prediction. This leads to faster computation.

We now present the technical detail of IOKR below, which has two consecutive steps.

Step 1: learning spectra-vectors mapping

Once parameters, i.e. function ϕ, are learned, we convert the structures of metabolites into their molecular vectors to obtain a new set of pairs, $\{(\mathbf{x}_i, \phi(\mathbf{y}_i))\}_{i=1}^n$.

Algorithm 3 Learning molecular representation vectors via HSIC.

Inputs:
set $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$ of spectra-structure pairs,
T: number of iterations, B: size of minibatch
Outputs:
$\theta = \{\mathrm{H}_1^1, \mathrm{H}_2^1, ..., \mathrm{H}_4^R, \mathrm{W}_1, \mathrm{W}_2, ..., \mathrm{W}_R\} \cup \{h_v | v \in \text{set of atoms}\}$
for $t \leftarrow 1$ to T **do**
  $\mathcal{D}_t = \{\{(\mathbf{x}_i^{(t,b)}, \mathbf{y}_i^{(t,b)})\}_{i=1}^{B}\}_{b=1}^{n/B}$ ▷ shuffled and split
  for $b \leftarrow 1$ to B **do**
    $\mathbf{Y}_{t,b} = \{\mathbf{y}_i^{t,b}\}_{i=1}^{B}$, $\mathbf{X}_{t,b} = \{\mathbf{x}_i^{t,b}\}_{i=1}^{B}$
    $\mathrm{F}_{t,b} = \phi(\mathbf{Y}_{t,b})$ ▷ Call Algorithm 1
    $\mathbf{S}_{t,b}$ is calculated from $\mathbf{X}_{t,b}$ by (3.7)
    $\mathbf{J}_{t,b} = \mathrm{trace}(\mathrm{S}_{t,b}\mathrm{F}_{t,b}^{\top}\mathrm{F}_{t,b})$
    gradient of loss layer $\leftarrow \mathrm{S}_{t,b}\mathrm{F}_{t,b}$
    Grad_θ is calculated by chain rule
    $\theta \leftarrow \theta - \gamma \mathrm{Grad}_\theta$ ▷ Update the whole parameters
  end for
end for

Now the goal is to find the optimal function $h : \mathcal{X} \rightarrow \mathcal{F}_d$ by minimizing the following objective function:

$$\hat{h} = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} ||h(\mathbf{x}_i) - \phi(\mathbf{y}_i)||_{\mathcal{F}_d}^2 + \lambda ||h||_{\mathcal{H}}^2, \tag{3.11}$$

where λ (> 0) is a regularization parameter to prevent overfitting and $\mathcal{H}$ is an approximate functional space that contains h; $\mathcal{F}_d$ is a space of molecular vectors of dimension d. By using the representer theorem in Micchelli and Pontil [2005], the optimal function $\hat{h}$ can be written as (see Appendix 4 for more detail):

$$\hat{h}(x) = \mathcal{K}_n(x, .)(\lambda \mathbf{I}_{nd} + \mathcal{K}_n)^{-1} \mathrm{vec}(\phi(\mathbf{Y}_n)) \tag{3.12}$$

where $\mathcal{K}_n$ is an operator-valued kernel, defined on spectra $\mathcal{X}$, satisfying certain constraints (see Micchelli and Pontil [2005]). As dimensionality d of space $\mathcal{F}_d$ is finite, the kernel value $\mathcal{K}_n(\mathbf{x}_i, \mathbf{x}_j)$ is a matrix with the size of $d \times d$. $\mathbf{I}_{nd}$ is the identity matrix of size $nd \times nd$. $\phi(\mathbf{Y}_n) = (\phi(\mathbf{y}_1), \phi(\mathbf{y}_2), ..., \phi(\mathbf{y}_n))$ is a matrix

with the size of $d \times n$, and vec(.) is the vectorization of the input matrix, where the output is a vector obtained by repeatedly stacking each column of the input matrix on the top of the next column.

Considering that the operator-valued kernel keeps $\mathcal{K}_n(\mathbf{x}, \mathbf{x}^{'}) = k(\mathbf{x}, \mathbf{x}^{'}) * \mathbf{I}_d$, the optimal solution can be simplified as:

$$\hat{h}(x) = \phi(\mathbf{Y}_n)(\lambda \mathbf{I}_n + \mathrm{K}_n)^{-1} k(\mathbf{X}, \mathbf{x}) \tag{3.13}$$

where K_n is real-valued kernel defined on the set of spectra. $\mathbf{I}_n$ is the identity matrix of size $n \times n$ and $k(\mathbf{X}, \mathbf{x}) = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}) \\ \vdots \\ k(\mathbf{x}_n, \mathbf{x}) \end{bmatrix}$ is a column vector.

Step 2: candidate retrieval

Given mapping $\hat{h}$ learned in Step 1, we now turn to the problem of finding the output metabolite in the database which corresponds to the query spectrum $\mathbf{x}$. To this end, we search metabolite $\mathbf{y}$ in the list of given candidates $\mathcal{Y}^*$, such that the squared distance between $\phi(\mathbf{y})$ and $\hat{h}(\mathbf{x})$ can be minimized:

$$f(\mathbf{x}) = \underset{\mathbf{y} \in \mathcal{Y}^*}{\operatorname{argmin}} \, ||\hat{h}(\mathbf{x}) - \phi(\mathbf{y})||^2_{\mathcal{F}_d} \tag{3.14}$$

Considering that the output kernel is normalized and the operator-valued kernel keeps $\mathcal{K}_n(\mathbf{x}, \mathbf{x}^{'}) = k(\mathbf{x}, \mathbf{x}^{'}) * \mathbf{I}_d$, the optimal solution of $f(\mathbf{x})$ can be estimated as the following :

$$\hat{f}(\mathbf{x}) = \underset{\mathbf{y} \in \mathcal{Y}^*}{\operatorname{argmax}} \, l(\mathbf{Y}, \mathbf{y})^\top (\lambda \mathbf{I}_n + \mathrm{K}_n)^{-1} k(\mathbf{X}, \mathbf{x}), \tag{3.15}$$

where $l(\mathbf{Y}, \mathbf{y}) = \begin{bmatrix} l(\mathbf{y}_1, \mathbf{y}) \\ \vdots \\ l(\mathbf{y}_n, \mathbf{y}) \end{bmatrix}$ and $k(\mathbf{X}, \mathbf{x}) = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}) \\ \vdots \\ k(\mathbf{x}_n, \mathbf{x}) \end{bmatrix}$ are column vectors.

Practically, the values given by objective function (3.15) are used as scores for ranking candidate metabolites in Step 2: candidate retrieval.

3.3.4 Kernels

ADAPTIVE uses kernels for the input and output.

Kernels for input

A various types of kernels are already defined and used for the input from MS/MS spectra. These kernels are typically divided into the following two groups: 1) kernels defined for spectra such as PPK [Jebara et al., 2004a] and 2) kernels defined for fragmentation trees [Rasche et al., 2011]. Details on these kernels can be found in Dührkop et al. [2015a].

In fact Dührkop et al. [2015a] suggested 24 different input kernels. ADAPTIVE combines these input kernels into a single kernel through Multiple Kernel Learning (MKL, Gönen and Alpaydin [2011]). ADAPTIVE uses two options for MKL: 1) UNIMKL (uniform MKL): assigns the same weights to all component kernels, and 2) ALIGNF: uses weights over kernels to combine. That is, in ALIGNF, weights over component kernels are optimized (trained) by maximizing the centered kernel alignment between the combined kernel and the target kernel defined on the molecular vectors, which generate trained parameters (model).

Kernels for output

After learning parameters (model) to generate the molecular vectors for structures, we define kernels for output $\mathcal{Y}$ by directly computing kernels on the corresponding molecular vectors. In our experiments, we consider the following two typical kernels:

- Linear kernel: $l(\mathbf{y}, \mathbf{y}') = \phi(\mathbf{y})^\top \phi(\mathbf{y}')$.
- Gaussian kernel: $l(\mathbf{y}, \mathbf{y}') = \exp(-\gamma ||\phi(\mathbf{y}) - \phi(\mathbf{y}_\prime)||^2)$,

where $\mathbf{y}$ and $\mathbf{y}'$ are molecular structures in $\mathcal{Y}$.

3.4. Experimental results

3.4.1 Data set and evaluation measures

We used a benchmark dataset in Brouard et al. [2016b] to evaluate ADAPTIVE and compare with existing methods. The dataset consists of 4,138 MS/MS

Table 3.1: Parameter values used for experiments

	Parameter	Values
T	#epoch	100
B	batchsize	100
R	#updates	6
d	#dim of molecular vectors	100,200,300
m	#dim of atom feature	50
N/A	#atom types	12 (C, O, N, P, S, etc)
N/A	#bond types	4 (single, double, triple, acromatic)

spectra extracted from the GNPS (Global Natural Products Social) public spectra library
(https://gnps.ucsd.edu/ProteoSAFe/libraries.jsp).

To compare ADAPTIVE with existing methods, we used the same setting for all competing methods. Specifically we used 10-fold cross-validation (CV), and the results are averaged over all 10-folds. The performance was checked by the top-k accuracies (where $k = 1, 10, 20$), which is the ratio of the number of the cases that the true structures are ranked at lower than or equal to k to the number of all cases. Also the speed was checked by computation time for prediction, measured by milliseconds per example (ms/example).

Hyperparameters, such as regularization parameter λ and parameter γ of the output kernel, were chosen by using leave-one-out CV on each training fold. For prediction in ADAPTIVE, at the retrieval stage, given test example $\mathbf{x}$, we computed the molecular vectors of $\mathbf{x}$, $\hat{h}(\mathbf{x})$ (see (3.11)) and those of all candidates $\phi(\mathbf{y})$ (see Algorithm 1). These candidates including the correct molecular structure of test example $\mathbf{x}$ were ranked, according to their distances to $\hat{h}(\mathbf{x})$ (from the smallest to the highest). These ranked candidates were used for computing the top-k accuracy. Table 3.1 shows a set of parameter values, which were used to train the model of generating molecular vectors.

All experiments were performed on a server with 2.7 GHz Intel Core i5 CPU and 8GB memory. The code was written in Python and Matlab with the support

Table 3.2: Comparison of the top-k accuracy (k=1, 10 and 20) of FingerID [Heinonen et al., 2012a], CSI:FingerID [Dührkop et al., 2015a], IOKR [Brouard et al., 2016b] and ADAPTIVE. The highest value (indicating the most accurate prediction) are in boldface for each k.

Method	Mol. vec. size	MKL	Accuracy (%)		
			Top 1	Top 10	Top 20
FingerID	2765	None	17.74	49.59	58.17
CSI:FingerID unit	2765	ALIGNF	24.82	60.47	68.20
CSI:FingerID mod Platt	2765	ALIGNF	28.84	66.07	73.07
IOKR linear	2765	UNIMKL	30.02	66.05	73.66
		ALIGNF	28.54	65.77	73.19
ADAPTIVE linear	100	UNIMKL	29.42	70.01	77.48
		ALIGNF	29.19	69.52	77.64
	200	UNIMKL	29.60	69.39	76.99
		ALIGNF	29.11	69.53	77.56
	300	UNIMKL	30.22	70.48	78.18
		ALIGNF	**30.61**	**70.51**	**78.23**
IOKR Gaussian	2765	UNIMKL	30.66	67.94	75.00
		ALIGNF	29.78	67.84	74.79
ADAPTIVE Gaussian	100	UNIMKL	29.47	70.01	77.51
		ALIGNF	29.37	69.91	77.48
	200	UNIMKL	29.47	69.86	77.09
		ALIGNF	28.98	69.65	77.17
	300	UNIMKL	30.3	70.58	78.26
		ALIGNF	**31.03**	**70.89**	**78.52**

of the Chainer framework [Tokui et al., 2015].

3.4.2 Performance results

Predictive performance

We compared the predictive performances of ADAPTIVE with three existing methods: FingerID [Heinonen et al., 2012a], CSI:FingerID [Dührkop et al., 2015a] and IOKR [Brouard et al., 2016b] in terms of the top-k accuracy (k=1,10 and 20). Table 3.2 shows the top-k accuracies of the competing methods with UNIMKL and ALIGNF for MKL and linear and Gaussian kernels for the output kernel, changing k from 1 to 20 and also changing the size of fingerprints from 100 to 300 (for ADAPTIVE only). This table first shows that ADAPTIVE achieved the best performance, being followed by IOKR, CSI:FingerID and FingerID. For example, ADAPTIVE with ALIGNF, Gaussian kernel and the fingerprint size of 300 achieved 31.03% for k=1, while IOKR with ALIGNF and Gaussian kernel was 29.78% and CSI:FingerID with ALIGNF was 28.84% or 24.82%. That of Finger:ID was only 17.74%. Interestingly, for k=1, the performance advantage of ADAPTIVE against IOKR was rather slight, while k= 10 and 20, ADAPTIVE outperformed IOKR much more clearly, with the difference of around 3 to 4% under the same condition for the two methods. This indicates that the performance advantage of ADAPTIVE was confirmed by checking a larger number of top candidates. Another finding is the performance difference between linear and Gaussian kernels was very slight (almost nothing) for ADAPTIVE under the same other conditions. Also this is true of UNIMKL and ALIGNF, the performance for them was rather the same. However the size of fingerprints strongly affected the performance in the sense that a larger size of fingerprints achieved a higher performance. In summary, ADAPTIVE clearly outperformed competing methods with, for example, for k=20, the difference of 3–4%, which is very sizable.

Computation time for prediction

IOKR was already shown to be faster than previous kernel-based methods in prediction [Brouard et al., 2016b]. Thus, we consider only IOKR as a competing method for examining computational efficiency. Table 3.3 shows the computation time of ADAPTIVE and IOKR with linear and Gaussian kernels for

Table 3.3: Computation time for prediction by ADAPTIVE and IOKR. The smallest values (indicating the fastest) were in boldface for linear and Gaussian kernels.

Method	Mol. vec. size	prediction time (ms/example)	
		linear	Gaussian
IOKR	2765	140.22	3352.4
ADAPTIVE	100	**20.32**	**802.6**
	200	39.88	844.33
	300	54.14	1071.8

prediction. The computation time was averaged over the 10-fold CV. This table shows that ADAPTIVE was significantly faster than IOKR. Specifically under both linear and Gaussian kernels, ADAPTIVE with the fingerprint size of 100 was four to seven times faster than IOKR. This is because molecular vectors by ADAPTIVE are much more precise and adaptive to given data than those used in IOKR.

3.4.3 Case study

To understand the results obtained by ADAPTIVE more, in the obtained molecular vectors, we examined substructures rooted at atoms, which activated several example features most. As shown in Section 3.3.2, each substructure rooted at an atom is represented by a state vector and contributes to computing the molecular vector of the whole molecule. Then, given a feature, we can estimate the contribution of each substructure by simply computing the softmax value from the corresponding state vector. We use these values of substructures as scores to rank substructures to activate the given feature.

Figure 3.4 shows three example features (#2, #39 and #83). For each feature, we show three substructures with the highest scores (each score is shown above the substructure). The first row shows three substructures which activated fea-

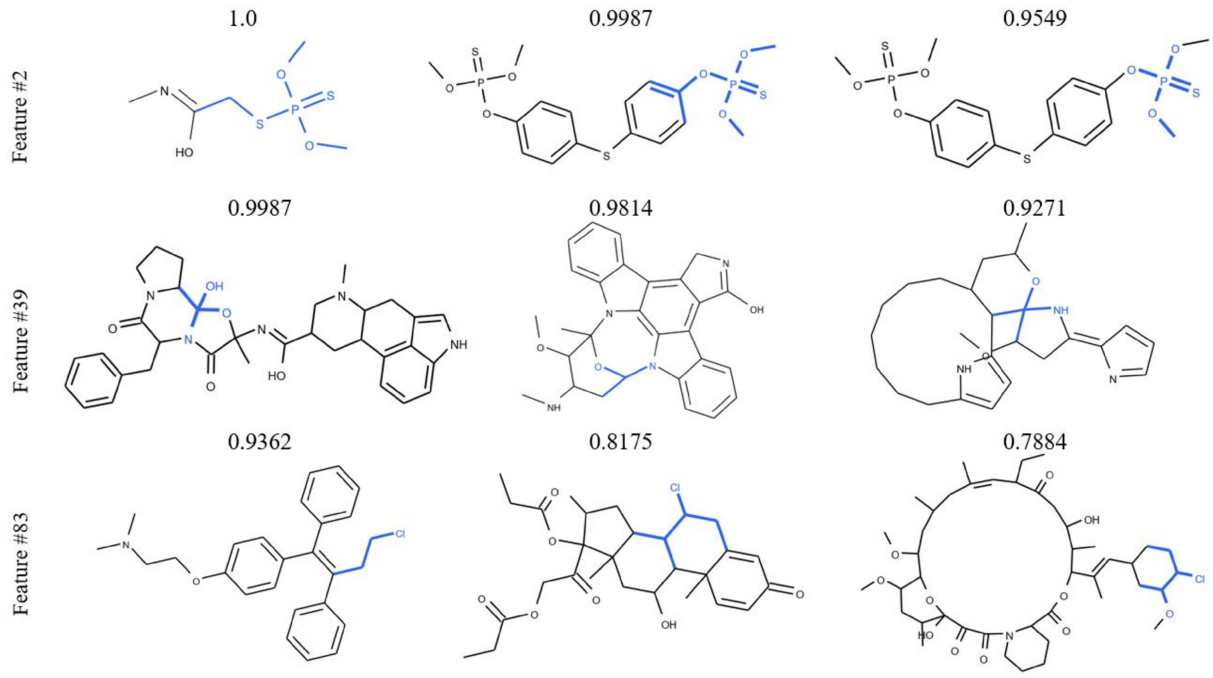

Figure 3.4: Example features (#2, #39 and #83) and their three substructures (and their scores) which activated the corresponding feature most. Note that three substructures of each feature share a similar group (set) of atoms which are shown in blue.

ture #2 most. Interestingly, we can see that these substructures share a further smaller, similar group of atoms: O, P and S (highlighted in blue). Similarly, the second row shows three substructures sharing a group of atoms: O and N, where these substructures activated feature #39 most. Also the third row shows substructures which activated feature #83, all having atom: Cl. Thus, Figure 3.4 shows that each feature of ADAPTIVE is activated by multiple different substructures sharing some similar properties, which must be important in data and probably for prediction. In contrast, each feature in regular molecular fingerprints is activated by only one predefined substructure. In summary, from this case study, learned features of ADAPTIVE are more concise and specific to the task of metabolite identification than regular molecular fingerprints, leading to the advantage of predictive performance and computation time.

3.5. Summary

Supervised learning for metabolite identification uses fingerprints as intermediate representation vectors between spectra and metabolites, while such vectors are too redundant to cover all possible substructures and chemical properties in metabolites, causing limitations in predictive performance and high computational costs. To overcome this problem, we have proposed ADAPTIVE, which generates representations of metabolites specific to given spectrum-structure pairs. ADAPTIVE learns a model to generate molecular vectors for metabolites, which is parameterized by a message passing neural network over given molecular structures and trained through optimizing the objective function to maximize the correlation between molecular vectors and corresponding spectra. Our empirical validation of ADAPTIVE with the benchmark data set showed the advantage of ADAPTIVE over existing methods including IOKR, the current cutting- edge method, both in predictive performance and computation time for prediction.

A drawback of ADAPTIVE would be interpretability, because structural information are implicitly encoded in compact vectors in ADAPTIVE and cannot be made explicit easily. In metabolite identification, it would be desirable to connect the set of peaks to the corresponding substructures/ chemical properties of metabolites (see Chapter 2). Developing a model with such interpretability would be interesting future work.

Chapter 4

Conclusion and future work

In the present thesis, we studied computational methods for metabolite identification from mass spectra data, with the focus on the machine learning approach. In Chapter 1, we reviewed many techniques/software tools with different approaches to deal with the task of metabolite identification, which can be divided into the following main groups: mass spectra library, *in silico* fragmentation and machine learning. We mainly focus on machine learning based methods (used in *in silico* fragmentation and machine learning approaches) for the task, which are the key to the recent progress in metabolite identification.

In Chapter 2, we have proposed machine learning models which are able to incorporate peak interactions for fingerprint prediction (the first stage in machine learning based approach). Our experiments showed that peak interactions are definitely useful to improve fingerprint prediction, along with discriminative information about peaks. Our first model is based on kernel learning, defining two kernels, one for peaks and the other for peak interactions, which are then combined through MKL. Again we note that Shen et al. [2014b] used fragmentation trees and spectra as input, while converting spectra into fragmentation trees is definitely computationally expensive. On the other hand, our model of kernel learning uses only peaks in the spectrum as input for prediction, indicating that our model is much more efficient. Kernel learning does not have to construct feature vectors for spectra, and instead this is done implicitly by kernels defined, which can avoid any error caused when generating feature vectors. However, a big issue of kernel learning is interpretability.

That is, it is difficult for kernel learning to figure out which subset of peaks or peak interactions exhibit the strongest effects on fingerprint prediction, despite that clearly each property depends on a few number of mass positions in each given spectrum Our sparse interaction models,(L-)SIMPLE, have a number of advantages, which are summarized in Chapter 2: (L-)SIMPLE is formulated as a sparse, convex optimization model, which can capture peak interactions and also give interpretable solutions. We emphasize that next generation fingerprint prediction needs a ML model, which should learn, from huge but sparse spectra, peaks as well as peak interactions comprehensibly and predict fingerprints.

In Chapter 3, we have proposed ADAPTIVE, which generates representations of metabolites specific to given spectrum-structure pairs. The model can overcome the limitations of molecular fingerprints which are lengthy and irrelevant to the task and data. ADAPTIVE learns a model to generate representations, named molecular vectors, for metabolites. The model is parameterized by a MPNN that takes given molecular structures as inputs and output molecular vectors. Its parameters are trained through optimizing the objective function to maximize the correlation between molecular vectors and corresponding spectra. Our empirical validation of ADAPTIVE with the benchmark dataset showed the advantage of ADAPTIVE over existing methods including IOKR, the current cutting-edge method, both in predictive performance and computation time for prediction. A drawback of ADAPTIVE would be interpretability, because structural information is implicitly encoded in compact vectors in ADAPTIVE and cannot be made explicit easily. In metabolite identification, it would be desirable to connect the set of peaks to the corresponding substructures/chemical properties of metabolites. Developing a model with such interpretability would be interesting future work.

Although encouraging results have been already obtained in this thesis, the approach must be extended further before the proposed methods become useful tools for the task of metabolite identification. In this thesis, supervised learning methods for predicting intermediate representations for metabolites have been proposed. Unsupervised learning models which are expected to discover potentially meaningful substructures from MS data (also known as

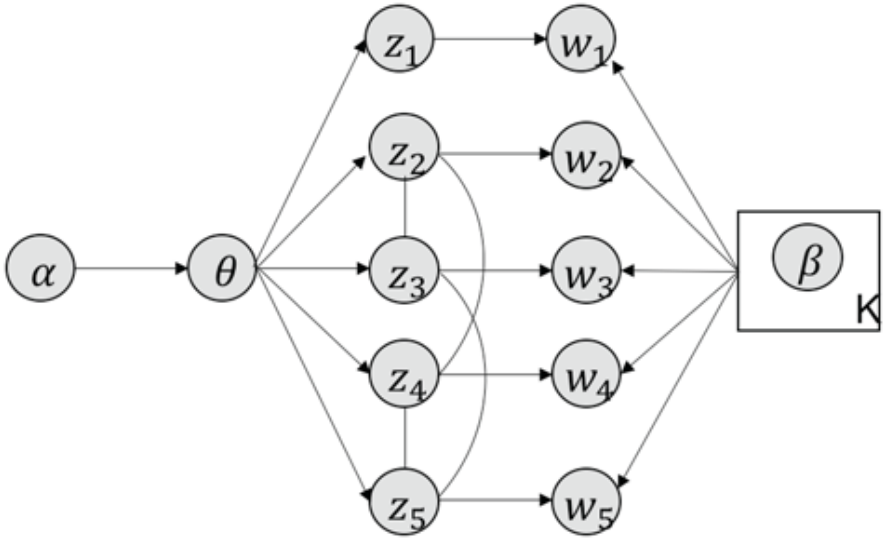

Figure 4.1: Graphical representation of Markov random field regularized LDA; if two words are correlated according to the external knowledge, an undirected edge between their topic labels is created. Finally, a graph in which nodes are latent topic labels and edges connect topic labels of semantically related words. In this example, the graph contains five nodes z_1, z_2, z_3, z_4, z_5 and two edges (z_2, z_4), (z_3, z_5).

substructure annotation mentioned in Chapter 1) will also be considered in future work. For instance, a key limitation of the existing unsupervised probabilistic topic models for extracting substructures (called motifs) from MS peaks, including LDA in Chapter 1, is that, words (peaks) are assumed to be uncorrelated or so-called bag-of-word assumption, meaning that the topic assignment for each word (peak) is irrelevant to all other words (peaks). This assumption results in losing rich information about the word (peak) dependencies and incoherent learned topics (motifs). Some methods have been proposed to incorporate external knowledge regarding the word correlation in the text application, such as WordNet [Miller, 1995], which can be considered to learn more coherent topics. Andrzejewski et al. [2009] proposed an approach to incorporate such knowledge into LDA by imposing Dirichlet Forest Prior, replacing the Dirichlet prior over topic-word multinomial to encode the Must-links and Cannot-links between words. Words having Must-links are imposed to have similar proba-

bilities within all topics while those with Cannot-links are not allowed to have high probabilities in any topics simultaneously. In a similar fashion, Newman et al. [2011] proposed a quadratic regularizer and a convolved Dirichlet over the topic-word distribution to incorporate the dependencies between words. One point is that these methods ignored the fact that there are some words correlated depending on the topic they appear in. Xie et al. [2015] proposed to use a Markov random field for regularization of LDA to encourage words similarly labeled to share the same topic label (Figure 4.1). Under this model, the topic assignment of each word is not independent, but depends on the topic labels of its correlated words. Motivated by these advanced learning models designed for text applications, FTs constructed directly from mass spectra can be used as a source of external knowledge to provide rich information about peak correlations, making the learned motifs more coherent.

In silico fragmentation is one of alternating approaches to overcome the insufficiency of mass spectra libraries. The good point of this approach is its ability to take advantage of compound databases with a huge number of chemical structures of known compounds to generate simulated spectra. However, generation of such spectra from chemical structures is challenging because the fragmentation process of a molecular structure is truly stochastic. Even from the same structure, different spectra can be generated at different times. Therefore, prediction of spectra from structures will be more challenging for machine learning methods and requires more research in the future. Our future work will be the development of generative machine learning models for producing realistic spectra from chemical structures, which can be then used for metabolite identification in combination with the reverse process (from spectra to chemical structures).

Additionally, we also emphasize that the combination of different approaches should be also taken into account, by which we can take advantages of them for significant improvements. For instance, (MP-)IOKR and CSI:FingerID are using machine learning and fragmentation trees as input. Another is MetFusion, mentioned in Chapter 1, combines the results from MassBank (mass spectral library) and MetFrag (*in silico* fragmentation) to take advantages of complementary approaches.

Bibliography

Felicity Allen, Russ Greiner, and David Wishart. Competitive fragmentation modeling of esi-ms/ms spectra for putative metabolite identification. *Metabolomics*, 11(1):98–110, 2015.

David Andrzejewski, Xiaojin Zhu, and Mark Craven. Incorporating domain knowledge into topic modeling via dirichlet forest priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 25–32. ACM, 2009.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Mathieu Blondel, Akinori Fujino, and Naonori Ueda. Convex factorization machines. In Annalisa Appice, Pedro Pereira Rodrigues, Vítor Santos Costa, João Gama, Alípio Jorge, and Carlos Soares, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 19–35, Cham, 2015. Springer International Publishing. ISBN 978-3-319-23525-7.

Sebastian Böcker and Florian Rasche. Towards de novo identification of metabolites by analyzing tandem mass spectra. *Bioinformatics*, 24(16):i49–i55, 2008.

Sebastian Böcker and Florian Rasche. Towards de novo identification of metabolites by analyzing tandem mass spectra. *Bioinformatics*, 24(16):i49–i55, 2008. doi: 10.1093/bioinformatics/btn270. URL `+http://dx.doi.org/10.1093/bioinformatics/btn270`.

Céline Brouard, Huibin Shen, Kai Dührkop, Florence d'Alché Buc, Sebastian Böcker, and Juho Rousu. Fast metabolite identification with input output kernel regression. *Bioinformatics*, 32(12):i28–i36, 2016a.

Céline Brouard, Huibin Shen, Kai Dührkop, Florence d'Alché Buc, Sebastian Böcker, and Juho Rousu. Fast metabolite identification with input output kernel regression. *Bioinformatics*, 32(12):i28–i36, 2016b. doi: 10.1093/bioinformatics/btw246. URL `http://dx.doi.org/10.1093/bioinformatics/btw246`.

Frank R Burden and David A Winkler. Relevance vector machines: sparse classification methods for qsar. *Journal of chemical information and modeling*, 55(8): 1529–1534, 2015.

Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

Jian-Feng Cai, Emmanuel J. CandÃšs, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010. doi: 10.1137/080738970. URL `https://doi.org/10.1137/080738970`.

HaiFeng Chen, BoTao Fan, HaiRong Xia, Michel Petitjean, ShenGang Yuan, Annick Panaye, and Jean-Pierre Doucet. Massis: a mass spectrum simulation system. 1. principle and method. *European Journal of Mass Spectrometry*, 9(3): 175–186, 2003.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *J. Mach. Learn. Res.*, 13(1):795–828, March 2012a. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=2503308.2188413`.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13(Mar):795–828, 2012b.

John S Cottrell and U London. Probability-based protein identification by searching sequence databases using mass spectrometry data. *electrophoresis*, 20(18):3551–3567, 1999.

Chhabil Dass. *Fundamentals of contemporary mass spectrometry*, volume 16. John Wiley & Sons, 2007.

Edmond De Hoffmann and Vincent Stroobant. *Mass spectrometry: principles and applications*. John Wiley & Sons, 2007.

Kai Dührkop, Huibin Shen, Marvin Meusel, Juho Rousu, and Sebastian Böcker. Searching molecular structure databases with tandem mass spectra using csi:fingerid. *Proceedings of the National Academy of Sciences*, 112(41):12580–12585, 2015a. ISSN 0027-8424. doi: 10.1073/pnas.1509788112. URL `http://www.pnas.org/content/112/41/12580`.

Kai Dührkop, Huibin Shen, Marvin Meusel, Juho Rousu, and Sebastian Böcker. Searching molecular structure databases with tandem mass spectra using csi: Fingerid. *Proceedings of the National Academy of Sciences*, 112(41):12580–12585, 2015b.

Warwick B Dunn and David I Ellis. Metabolomics: current analytical platforms and methodologies. *TrAC Trends in Analytical Chemistry*, 24(4):285–294, 2005.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2224–2232. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5954-convolutional-networks-on-graphs-for-learning-molecular-fingerprints.pdf`.

Jimmy K Eng, Ashley L McCormack, and John R Yates. An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *Journal of the American Society for Mass Spectrometry*, 5(11): 976–989, 1994.

Johann Gasteiger, Wolfgang Hanebeck, and Klaus-Peter Schulz. Prediction of mass spectra from structural information. *Journal of Chemical Information and Computer Sciences*, 32(4):264–271, 1992.

Michael Gerlich and Steffen Neumann. Metfusion: integration of compound identification strategies. *Journal of Mass Spectrometry*, 48(3):291–298, 2013.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL `http://proceedings.mlr.press/v70/gilmer17a.html`.

Mehmet Gönen and Ethem Alpaydin. Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, 12:2211–2268, July 2011. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1953048.2021071`.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of machine learning research*, 12(Jul):2211–2268, 2011.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, ALT'05, pages 63–77, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3-540-29242-X, 978-3-540-29242-5. doi: 10.1007/11564089_7. URL `http://dx.doi.org/10.1007/11564089_7`.

Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

Jürgen H Gross. *Mass spectrometry: a textbook*. Springer Science & Business Media, 2006.

Markus Heinonen, Ari Rantanen, Taneli Mielikäinen, Juha Kokkonen, Jari Kiuru, Raimo A Ketola, and Juho Rousu. Fid: a software for ab initio structural

identification of product ions from tandem mass spectrometric data. *Rapid Communications in Mass Spectrometry*, 22(19):3043–3052, 2008.

Markus Heinonen, Huibin Shen, Nicola Zamboni, and Juho Rousu. Metabolite identification and molecular fingerprint prediction through machine learning. *Bioinformatics*, 28(18):2333–2341, 2012a. doi: 10.1093/bioinformatics/bts437. URL +http://dx.doi.org/10.1093/bioinformatics/bts437.

Markus Heinonen, Huibin Shen, Nicola Zamboni, and Juho Rousu. Metabolite identification and molecular fingerprint prediction through machine learning. *Bioinformatics*, 28(18):2333–2341, 2012b.

Dennis W Hill, Tzipporah M Kertesz, Dan Fontaine, Robert Friedman, and David F Grant. Mass spectral metabonomics beyond elemental formula: chemical database querying by matching experimental with computational fragmentation spectra. *Analytical chemistry*, 80(14):5574–5582, 2008.

Hisayuki Horai, Masanori Arita, Shigehiko Kanaya, Yoshito Nihei, Tasuku Ikeda, Kazuhiro Suwa, Yuya Ojima, Kenichi Tanaka, Satoshi Tanaka, Ken Aoshima, Yoshiya Oda, Yuji Kakazu, Miyako Kusano, Takayuki Tohge, Fumio Matsuda, Yuji Sawada, Masami Yokota Hirai, Hiroki Nakanishi, Kazutaka Ikeda, Naoshige Akimoto, Takashi Maoka, Hiroki Takahashi, Takeshi Ara, Nozomu Sakurai, Hideyuki Suzuki, Daisuke Shibata, Steffen Neumann, Takashi Iida, Ken Tanaka, Kimito Funatsu, Fumito Matsuura, Tomoyoshi Soga, Ryo Taguchi, Kazuki Saito, and Takaaki Nishioka. Massbank: a public repository for sharing mass spectral data for life sciences. *Journal of Mass Spectrometry*, 45(7):703–714, 2010. ISSN 1096-9888. doi: 10.1002/jms.1777. URL http://dx.doi.org/10.1002/jms.1777.

Jan Hummel, Nadine Strehmel, Joachim Selbig, Dirk Walther, and Joachim Kopka. Decision tree supported substructure prediction of metabolites from gc-ms profiles. *Metabolomics*, 6(2):322–333, Jun 2010a. ISSN 1573-3890. doi: 10.1007/s11306-010-0198-7. URL https://doi.org/10.1007/s11306-010-0198-7.

Jan Hummel, Nadine Strehmel, Joachim Selbig, Dirk Walther, and Joachim

Kopka. Decision tree supported substructure prediction of metabolites from gc-ms profiles. *Metabolomics*, 6(2):322–333, 2010b.

Timea Imre, Tibor Kremmer, Karoly Heberger, Éva Molnár-Szöllősi, Krisztina Ludanyi, Gabriella Pocsfalvi, Antonio Malorni, Laszlo Drahos, and Karoly Vekey. Mass spectrometric and linear discriminant analysis of n-glycans of human serum alpha-1-acid glycoprotein in cancer patients and healthy individuals. *Journal of proteomics*, 71(2):186–197, 2008.

Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *J. Mach. Learn. Res.*, 5:819–844, December 2004a. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1005332.1016786`.

Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *Journal of Machine Learning Research*, 5(Jul):819–844, 2004b.

Lars J Kangas, Thomas O Metz, Giorgis Isaac, Brian T Schrom, Bojana Ginovska-Pangovska, Luning Wang, Li Tan, Robert R Lewis, and John H Miller. In silico identification software (isis): a machine learning approach to tandem mass spectral identification of lipids. *Bioinformatics*, 28(13):1705–1713, 2012.

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

Marius Kloft, Ulf Brefeld, Sören Sonnenburg, and Alexander Zien. lp-norm multiple kernel learning. *J. Mach. Learn. Res.*, 12:953–997, July 2011a. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1953048.2021033`.

Marius Kloft, Ulf Brefeld, Sören Sonnenburg, and Alexander Zien. Lp-norm multiple kernel learning. *Journal of Machine Learning Research*, 12(Mar):953–997, 2011b.

Sangeeta Kumari, Doug Stevens, Tobias Kind, Carsten Denkert, and Oliver Fiehn. Applying in-silico retention index and mass spectra matching for identification of unknown metabolites in accurate mass gc-tof mass spectrometry. *Analytical chemistry*, 83(15):5895–5902, 2011.

Jinbo Li and Shiliang Sun. Nonlinear combination of multiple kernels for support vector machines. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 2889–2892. IEEE, 2010.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. Gated graph sequence neural networks. *CoRR*, abs/1511.05493, 2015.

Shiqian Ma, Donald Goldfarb, and Lifeng Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1):321–353, Jun 2011. ISSN 1436-4646. doi: 10.1007/s10107-009-0306-5. URL `https://doi.org/10.1007/s10107-009-0306-5`.

Yan Ma, Tobias Kind, Dawei Yang, Carlos Leon, and Oliver Fiehn. Ms2analyzer: A software for small molecule substructure annotations from accurate tandem mass spectra. *Analytical chemistry*, 86(21):10724–10731, 2014.

Alexander Makarov. Electrostatic axially harmonic orbital trapping: a high-performance technique of mass analysis. *Analytical chemistry*, 72(6):1156–1162, 2000.

Fred W McLafferty and Frantisek Turecek. *Interpretation of mass spectra*. University science books, 1993.

Charles A. Micchelli and Massimiliano A. Pontil. On learning vector-valued functions. *Neural Comput.*, 17(1):177–204, January 2005. ISSN 0899-7667. doi: 10.1162/0899766052530802. URL `http://dx.doi.org/10.1162/0899766052530802`.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

R Mistrik. A new concept for the interpretation of mass spectra based on a combination of a fragmentation mechanism database and a computer expert system. *Ashcroft AE, Brenton G, Monaghan JJ (Edr) Advances i n Mass Spectrometry. Elsevier, Amsterdam*, 2004.

Aida Mrzic, Pieter Meysman, Wout Bittremieux, and Kris Laukens. Automated recommendation of metabolite substructures from mass spectra using frequent pattern mining. *bioRxiv*, page 134189, 2017.

Roman Mylonas, Yann Mauron, Alexandre Masselot, Pierre-Alain Binz, Nicolas Budin, Marc Fathi, Véronique Viette, Denis F Hochstrasser, and Frederique Lisacek. X-rank: a robust algorithm for small molecule identification using tandem mass spectrometry. *Analytical chemistry*, 81(18):7604–7610, 2009.

David Newman, Edwin V Bonilla, and Wray Buntine. Improving topic coherence with regularized topic models. In *Advances in neural information processing systems*, pages 496–504, 2011.

Dai Hai Nguyen, Canh Hao Nguyen, and Hiroshi Mamitsuka. Simple: Sparse interaction model over peaks of molecules for fast, interpretable metabolite identification from tandem mass spectra. *Bioinformatics*, 34(13):i323–i332, 2018.

Dai Hai Nguyen, Canh Hao Nguyen, and Hiroshi Mamitsuka. Adaptive: learning data-dependent, concise molecular vectors for fast, accurate metabolite identification from tandem mass spectra. *Bioinformatics*, 35(14):i164–i172, 2019a.

Dai Hai Nguyen, Canh Hao Nguyen, and Hiroshi Mamitsuka. Recent advances and prospects of computational methods for metabolite identification: a review with emphasis on machine learning approaches. *Briefings in bioinformatics*, 20(6):2028–2043, 2019b.

Hai Nguyen, Shin ichi Maeda, and Kenta Oono. Semi-supervised learning of hierarchical representations of molecules using neural message passing. *CoRR*, abs/1711.10168, 2017.

Noel M. O'Boyle, Michael Banck, Craig A. James, Chris Morley, Tim Vandermeersch, and Geoffrey R. Hutchison. Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3(1):33, Oct 2011. ISSN 1758-2946. doi: 10.1186/1758-2946-3-33. URL `https://doi.org/10.1186/1758-2946-3-33`.

Florian Rasche, Aleš Svatoš, Ravi Kumar Maddula, Christoph Böttcher, and Sebastian Böcker. Computing fragmentation trees from tandem mass spectrometry data. *Analytical Chemistry*, 83(4):1243–1251, 2010.

Florian Rasche, Aleš Svatoš, Ravi Kumar Maddula, Christoph Böttcher, and Sebastian Böcker. Computing fragmentation trees from tandem mass spectrometry data. *Analytical Chemistry*, 83(4):1243–1251, 2011. doi: 10.1021/ac101825k. URL `http://dx.doi.org/10.1021/ac101825k`. PMID: 21182243.

Florian Rasche, Kerstin Scheubert, Franziska Hufsky, Thomas Zichner, Marco Kai, Aleš Svatoš, and Sebastian Böcker. Identifying the unknowns by aligning fragmentation trees. *Analytical chemistry*, 84(7):3417–3426, 2012.

Imran Rauf, Florian Rasche, François Nicolas, and Sebastian Böcker. Finding maximum colorful subtrees in practice. *Journal of Computational Biology*, 20 (4):311–321, 2013.

Lars Ridder, Justin JJ Hooft, Stefan Verhoeven, Ric CH Vos, René Schaik, and Jacques Vervoort. Substructure-based annotation of high-resolution multistage msn spectral trees. *Rapid Communications in Mass Spectrometry*, 26(20): 2461–2471, 2012.

Miquel Rojas-Cherto, Julio E Peironcely, Piotr T Kasper, Justin JJ van der Hooft, Ric CH de Vos, Rob Vreeken, Thomas Hankemeier, and Theo Reijmers. Metabolite identification using automated comparison of high-resolution multistage mass spectral trees. *Analytical chemistry*, 84(13):5524–5534, 2012.

Kerstin Scheubert, Franziska Hufsky, and Sebastian Böcker. Computational mass spectrometry for small molecules. *Journal of cheminformatics*, 5(1):12, 2013.

Emma L Schymanski, Markus Meringer, and Werner Brack. Matching structures to mass spectra using fragmentation patterns: are the results as good as they look? *Analytical chemistry*, 81(9):3608–3617, 2009.

Huibin Shen, Kai Dührkop, Sebastian Böcker, and Juho Rousu. Metabolite identification through multiple kernel learning on fragmentation trees. *Bioin-*

formatics, 30(12):i157–i164, 2014a. doi: 10.1093/bioinformatics/btu275. URL +http://dx.doi.org/10.1093/bioinformatics/btu275.

Huibin Shen, Kai Dührkop, Sebastian Böcker, and Juho Rousu. Metabolite identification through multiple kernel learning on fragmentation trees. *Bioinformatics*, 30(12):i157–i164, 2014b.

Colin A Smith, Grace O'Maille, Elizabeth J Want, Chuan Qin, Sunia A Trauger, Theodore R Brandon, Darlene E Custodio, Ruben Abagyan, and Gary Siuzdak. Metlin: a metabolite mass spectral database. *Therapeutic drug monitoring*, 27(6):747–751, 2005.

Stephen E Stein and Donald R Scott. Optimization and testing of mass spectral library search algorithms for compound identification. *Journal of the American Society for Mass Spectrometry*, 5(9):859–866, 1994.

Ralf Tautenhahn, Kevin Cho, Winnie Uritboonthai, Zhengjiang Zhu, Gary J Patti, and Gary Siuzdak. An accelerated workflow for untargeted metabolomics using the metlin database. *Nature biotechnology*, 30(9):826, 2012.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. Chainer: a next-generation open source framework for deep learning. In *Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS)*, volume 5, pages 1–6, 2015.

Justin Johan Jozias van Der Hooft, Joe Wandy, Michael P Barrett, Karl EV Burgess, and Simon Rogers. Topic modeling for untargeted substructure exploration in metabolomics. *Proceedings of the National Academy of Sciences*, 113 (48):13738–13743, 2016.

Mingxun Wang, Jeremy J Carver, Vanessa V Phelan, Laura M Sanchez, Neha Garg, Yao Peng, Don Duy Nguyen, Jeramie Watrous, Clifford A Kapono, Tal

Luzzatto-Knaan, et al. Sharing and community curation of mass spectrometry data with global natural products social molecular networking. *Nature biotechnology*, 34(8):828, 2016.

T. Watanabe, C. D. Scott, D. Kessler, M. Angstadt, and C. Sripada. Scalable fused lasso svm for connectome-based disease prediction. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5989–5993, May 2014. doi: 10.1109/ICASSP.2014.6854753.

Jeramie Watrous, Patrick Roach, Theodore Alexandrov, Brandi S Heath, Jane Y Yang, Roland D Kersten, Menno van der Voort, Kit Pogliano, Harald Gross, Jos M Raaijmakers, et al. Mass spectral molecular networking of living microbial colonies. *Proceedings of the National Academy of Sciences*, 109(26):E1743–E1752, 2012.

David S Wishart. Computational strategies for metabolite identification in metabolomics. *Bioanalysis*, 1(9):1579–1596, 2009.

David S Wishart, Yannick Djoumbou Feunang, Ana Marcu, An Chi Guo, Kevin Liang, Rosa Vázquez-Fresno, Tanvir Sajed, Daniel Johnson, Carin Li, Naama Karu, et al. Hmdb 4.0: the human metabolome database for 2018. *Nucleic acids research*, 46(D1):D608–D617, 2017.

Sebastian Wolf, Stephan Schmidt, Matthias Müller-Hannemann, and Steffen Neumann. In silico fragmentation for computer assisted identification of metabolite mass spectra. *BMC bioinformatics*, 11(1):148, 2010.

Pengtao Xie, Diyi Yang, and Eric Xing. Incorporating word correlation knowledge into topic modeling. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 725–734, 2015.

Makoto Yamada, Yuta Umezu, Kenji Fukumizu, and Ichiro Takeuchi. Post selection inference with kernels. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*,

pages 152–160, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR. URL http://proceedings.mlr.press/v84/yamada18a.html.

Jane Y Yang, Laura M Sanchez, Christopher M Rath, Xueting Liu, Paul D Boudreau, Nicole Bruns, Evgenia Glukhov, Anne Wodtke, Rafael De Felicio, Amanda Fenner, et al. Molecular networking as a dereplication strategy. *Journal of natural products*, 76(9):1686–1699, 2013.

H Yoshida, R Leardi, K Funatsu, and K Varmuza. Feature selection by genetic algorithms for mass spectral classifiers. *Analytica Chimica Acta*, 446(1-2):483–492, 2001.

Qinyi Zhang, Sarah Filippi, Arthur Gretton, and Dino Sejdinovic. Large-scale kernel methods for independence testing. *Statistics and Computing*, 28(1):113–130, January 2018. ISSN 0960-3174. doi: 10.1007/s11222-016-9721-7. URL https://doi.org/10.1007/s11222-016-9721-7.

Appendix

A.1. Optimization problem 2.14 is convex w.r.t. w and W

The predictive model 2.12 can be rewritten as:

$$f(\mathbf{x}; w, W) = b + w^T\mathbf{x} + \mathbf{x}^T W\mathbf{x} \tag{4.1}$$

$$= b + \left(w^T \text{vec}(W)^T\right)\begin{pmatrix} x \\ \text{vec}(xx^T)\end{pmatrix} \tag{4.2}$$

$$= b + z^T\begin{pmatrix} x \\ \text{vec}(xx^T)\end{pmatrix} \tag{4.3}$$

where $z = \begin{pmatrix} w \\ \text{vec}(W)\end{pmatrix}$. Since this is a linear model, the classification loss term in 2.14 is jointly convex with respect to both w and W if loss function employed is convex.

For the regularization term, we define the following function for z: $|z| = \alpha||w||_1 + \beta||W||_*$. We will prove that it is a norm. Indeed, it is trivial to verify that if $|z| = 0$ only if $z = 0$, and that $|tz| = t|z|$ if t is a scalar. The remaining requirement is that the function satisfies the triangle inequality; that is, given $z = \begin{pmatrix} w \\ \text{vec}(W)\end{pmatrix}$ and $z' = \begin{pmatrix} w' \\ \text{vec}(W)'\end{pmatrix}$, we have:

$$\begin{aligned} |z + z'| &= |\begin{pmatrix} w + w' \\ \text{vec}(W) + \text{vec}(W)'\end{pmatrix}| = \alpha||w + w'||_1 + \beta||\text{vec}(W) + \text{vec}(W)'||_* \\ &\leq \alpha(||w||_1 + ||w'||_1) + \beta(||\text{vec}(W)||_* + ||\text{vec}(W)'||_*) = |z| + |z'| \end{aligned} \tag{4.4}$$

The last inequality hold as $||.||_1$ and $||.||_*$ are norms, satisfying the triangle inequality. Therefore, the regularization term in 2.14 is also convex.

A.2. ADMM for updating b and w

The subproblem for optimizing b, w in (2.21) is equivalent to

$$\begin{aligned} b^{t+1}, w^{t+1} &= \underset{b,w}{\text{argmin}} \quad \mathcal{L}(b, w, W^t, \mathbf{C}^t, \mathbf{u}^t) \\ &= \underset{b,w}{\text{argmin}} \quad \alpha||w||_1 + \frac{1}{2}||\mathbf{1} - \mathbf{Y}\mathbf{F}^t - \mathbf{C} + \mathbf{u}||_2^2 \\ &= \underset{b,w}{\text{argmin}} \quad \alpha||w||_1 + \frac{1}{2}||\mathbf{X}w + b\mathbf{1} - \hat{\mathbf{F}}_1||_2^2 \end{aligned} \tag{4.5}$$

where $\hat{\mathbf{F}}_1 = \mathbf{Y}(\mathbf{1} - \mathbf{C} + \mathbf{u}) - \text{diag}(\mathbf{X}W\mathbf{X}^T)$. We denote the operator diag(.) to produce a vector composed of diagonal elements of given square matrix.

In fact, solving (4.5) can be done easily with proximal gradient descent through alternately updating estimate of w and b as follows:

$$\begin{cases} \mathbf{z}^{k+1} &= w^k - \delta_w \mathbf{X}^T(\mathbf{X}w^k + b^k - \hat{\mathbf{F}}_1) \\ w^{k+1} &= \mathcal{S}_\alpha(\mathbf{z}^{k+1}) \\ b^{k+1} &= \frac{1}{n}\text{sum}(\hat{\mathbf{F}}_1 - \mathbf{X}w^{k+1}) \end{cases} \tag{4.6}$$

where δ_w is the learning rate for the proximal gradient updates and set to 0.01 in our experiments, k is the index of inner loop for optimizing b, w, $\mathcal{S}_\lambda(t)$ is the element-wise soft-thresholding operator, defined by:

$$\mathcal{S}_\lambda(t) = \text{sign}(t)\max(0, |t| - \lambda) \tag{4.7}$$

A.3. ADMM for updating W

The subproblem of optimizing W in (2.21) is equivalent to:

$$\begin{aligned} W^{t+1} &= \underset{W}{\operatorname{argmin}}\, \mathcal{L}(b^{t+1}, w^{t+1}, W, \mathbf{C}^t, \mathbf{u}^t) \\ &= \underset{W}{\operatorname{argmin}}\, \beta||W||_* + \gamma \operatorname{trace}(WL) \\ &\qquad + \frac{1}{2}||\mathbf{1} - \mathbf{Y}\mathbf{F}^t - \mathbf{C}^t + \mathbf{u}^t||_2^2 \\ &= \underset{W}{\operatorname{argmin}}\, \beta||W||_* + \gamma \operatorname{trace}(WL) \\ &\qquad + \frac{1}{2}||\operatorname{diag}(\mathbf{X}W\mathbf{X}^T) - \hat{\mathbf{F}}_2||_2^2 \end{aligned} \tag{4.8}$$

where $\hat{\mathbf{F}}_2 = \mathbf{Y}(\mathbf{1} - \mathbf{C} + \mathbf{u}) - b\mathbf{1} - \mathbf{X}w$. It is obvious that the objective function (4.8) split in two components: $\gamma \operatorname{trace}(WL) + \frac{1}{2}||\operatorname{diag}(\mathbf{X}W\mathbf{X}^T) - \hat{\mathbf{F}}_2||_2^2$, which is convex, differentiable and $\beta||W||_*$, which is also convex with inexpensive proximal operator. It is known that the solution of (4.8) is given by the matrix shrinkage operation which corresponds to a singular value decomposition (SVD) (see, e.g., [Cai et al., 2010] and [Ma et al., 2011] for more details). Hence, proximal gradient descent for updating W again is given as follows:

$$\begin{cases} \mathbf{R}^{k+1} & = \operatorname{diag}(\mathbf{X}W^k\mathbf{X}^T) - \hat{\mathbf{F}}_2 \\ \Delta_W\mathcal{L} & = \gamma L + \mathbf{X}^T\mathbf{R}^{k+1}\mathbf{X} \\ \mathbf{Z}^{k+1} & = W^k - \delta_W\Delta_W\mathcal{L} \\ \mathbf{U}^{k+1}, \mathbf{E}^{k+1} & = \mathbf{EVD}(\mathbf{Z}^{k+1}) \\ \hat{\mathbf{E}}^{k+1} & = \mathcal{S}_\beta(\mathbf{E}^{k+1}) \\ W^{k+1} & = \mathbf{U}^{k+1}\hat{\mathbf{E}}^{k+1}\mathbf{U}^{k+1} \end{cases} \tag{4.9}$$

where $\Delta_W\mathcal{L}$ is the derivative of the differentiable component of (4.8) with respect to W, δ_W is the learning rate for the proximal gradient updates and set to 0.05 in our experiments, k is the index of inner loop for optimizing W. $\mathbf{U}$ and $\mathbf{E}$ are columns of eigenvectors and diagonal matrix of eigenvalues obtained by applying eigendecomposition (**EVD**) to $\mathbf{Z}$, by which we can guarantee the semidefiniteness of weight matrix W after each iteration.

A.4. ADMM for updating C

For the subproblem of optimizing $\mathbf{C}$ in (2.21), it is equivalent to

$$\begin{aligned}\mathbf{C}^{t+1} &= \underset{\mathbf{C}}{\operatorname{argmin}}\, \mathcal{L}(b^{t+1}, w^{t+1}, W^{t+1}, \mathbf{C}, \mathbf{u}^t) \\ &= \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{n} (C_i)_+ + \frac{1}{2}||\mathbf{1} - \mathbf{Y}\mathbf{F}^{t+1} + \mathbf{u}^t - \mathbf{C}||^2\end{aligned} \tag{4.10}$$

In order to solve (4.10), we use the following proposition in [Watanabe et al., 2014]:

Proposition 1: the solution $\mathcal{T}_\lambda(t) = \operatorname{argmin}_{x\in\mathbb{R}} \lambda(x)_+ + \frac{1}{2}(x-t)^2$ has the following form:

$$\mathcal{T}_\lambda(t) = \begin{cases} t-\lambda & \text{if } t \geq \lambda \\ 0 & \text{if } 0 \leq t \leq \lambda \\ t & \text{if } t < 0 \end{cases} \tag{4.11}$$

Note that components of $\mathbf{C}$ are independent of each other in (4.10). By applying Proposition 1 in element-wise, we can derive the update for C in the following closed form solution:

$$\mathbf{C}^{t+1} = \mathcal{T}_1(\mathbf{1} - \mathbf{Y}\mathbf{F}^{t+1} + \mathbf{u}^t) \tag{4.12}$$

A.5. Solving 3.11

The goal is to seek the optimal function $h : \mathcal{X} \to \mathcal{F}_d$ by solving the optimization problem 3.11. By using the representer theorem in Micchelli and Pontil [2005], optimal solution $\hat{h}$ of (3.11) can be represented by a linear combination of vector-valued kernels on training set $\mathcal{X}$:

$$\hat{h}(\mathbf{x}_i) = \sum_{i=1}^{n} \mathcal{K}_n(\mathbf{x}_i, \mathbf{x}_j) c_j, \tag{4.13}$$

where $c_i (i = 1, ..., n)$ are vectors in $\mathcal{F}_d$; $\mathcal{K}_n$ is an operator-valued kernel, defined on spectra $\mathcal{X}$, satisfying certain constraints (see Micchelli and Pontil [2005]). As dimensionality d of space $\mathcal{F}_d$ is finite, the kernel is a matrix with the size of $d \times d$.

By replacing $\hat{h}(\mathbf{x}_i)$ in (3.11) with (4.13), the objective function in (3.11) becomes:

$$\sum_{i=1}^{n} || \sum_{j=1}^{n} \mathcal{K}_n(\mathbf{x}_i, \mathbf{x}_j) c_j - \phi(\mathbf{y}_i)||^2_{\mathcal{F}_d} + +\lambda \sum_{i=1}^{n} \sum_{j=1}^{n} c_i^T \mathcal{K}_n(\mathbf{x}_i, \mathbf{x}_j) c_j \tag{4.14}$$

In matrix form, the above can be rewritten as:

$$||\mathcal{K}_n \text{vec}(\mathbf{C}_n) - \text{vec}(\phi(\mathbf{Y}_n))||^2_{\mathcal{F}} + \lambda \text{vec}(\mathbf{C}_n)^T \mathcal{K}_n \text{vec}(\mathbf{C}_n) \tag{4.15}$$

where $\mathbf{C}_n = (c_1, c_2, ..., c_n)$ and $\phi(\mathbf{Y}_n) = (\phi(\mathbf{y}_1), \phi(\mathbf{y}_2), ..., \phi(\mathbf{y}_n))$ are both matrices with the size of $d \times n$.

By taking the derivative of (4.15) with respect to $\text{vec}(\mathbf{C}_n)$ and setting it to zero, we obtain the optimal solution for $\text{vec}(\mathbf{C}_n)$:

$$\text{vec}(\mathbf{C}_n) = (\lambda \mathbf{I}_{nd} + \mathcal{K}_n)^{-1} \text{vec}(\phi(\mathbf{Y}_n)) \tag{4.16}$$

From (4.13) and (4.16), we get the solution $\hat{h}$ in Equation (3.12).

In the case that the operator-valued kernel keeps $\mathcal{K}_n(\mathbf{x}, \mathbf{x}^{'}) = k(\mathbf{x}, \mathbf{x}^{'}) * \mathbf{I}_d$, 4.15 can be simplified as:

$$||\phi(\mathbf{Y}_n) - \mathbf{C}_n \mathrm{K}_n||^2_{\mathcal{F}} + \lambda \text{trace}(\mathbf{C}_n^T \mathrm{K}_n \mathbf{C}_n) \tag{4.17}$$

By taking the derivative of (4.17) with respect to $\mathbf{C}_n$ and setting it to zero, we obtain: $(\mathbf{C}_n \mathrm{K}_n - \phi(\mathbf{Y}_n))\mathrm{K}_n + \lambda \mathbf{C}_n \mathrm{K}_n = 0$, then we can get (3.13).