Bioinformatics, 34, 2018, i323–i332
doi: 10.1093/bioinformatics/bty252
ISMB 2018

SIMPLE: Sparse Interaction Model over Peaks of moLEcules for fast, interpretable metabolite identification from tandem mass spectra

Dai Hai Nguyen[1],*, Canh Hao Nguyen[1] and Hiroshi Mamitsuka[1,2]

[1]Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji 611-0011, Japan and [2]Department of Computer Science, Alato University, Espoo 02150, Finland

*To whom correspondence should be addressed.

Abstract

Motivation: Recent success in metabolite identification from tandem mass spectra has been led by machine learning, which has two stages: mapping mass spectra to molecular fingerprint vectors and then retrieving candidate molecules from the database. In the first stage, i.e. fingerprint prediction, spectrum peaks are features and considering their interactions would be reasonable for more accurate identification of unknown metabolites. Existing approaches of fingerprint prediction are based on only individual peaks in the spectra, without explicitly considering the peak interactions. Also the current cutting-edge method is based on kernels, which are computationally heavy and difficult to interpret.

Results: We propose two learning models that allow to incorporate peak interactions 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.

Availability and implementation: The code will be accessed through http://mamitsukalab.org/tools/SIMPLE/.

Contact: hai@kuicr.kyoto-u.ac.jp

1 Introduction

Metabolites are small molecules which are used in, or created by, chemical reactions occurring in living organisms (Wishart, 2007). They play a lot of important functions such as energy transport, signaling, building block of cells and inhibition/catalysis. Understanding biochemical characteristics of metabolites is an essential and important part of metabolomics to enlarge the knowledge of biological systems. It is also key to development of many applications and areas such as biotechnology, biomedicine or pharmaceutical sciences. Mass Spectrometry is a common technique in analytical chemistry (de Hoffmann and Stroobant, 2007) for metabolite identification. A mass spectrometer analyses a chemical sample by fragmenting it and measuring the mass-to-charge ratios (m/z) of its fragments. The resulting mass spectrum (MS) or tandem mass spectrum (MS/MS) 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 (Fig. 1). Another common representation of mass spectrum is a list of peaks, each defined by its m/z and intensity value (top-right corner of Fig. 1).

Many methods have been proposed for metabolite identification. A traditional approach is to compare a query MS or MS/MS spectrum of unknown compound against a database of a number of reference MS or MS/MS spectra (Scheubert *et al.*, 2013). The candidate molecules from the database are ranked based on the similarity, and the query spectrum and the best matched candidates are returned. However, the main disadvantage of this approach is that the reference database often contains only a small fraction of molecules in reality, leading to unreliable matching results if the molecule of the query spectrum is not in the database. Consequently, to mitigate the insufficiency of the reference databases, alternative approaches for metabolite identification are devised to deal with unavailable measured reference spectra. In general, we can divide computational methods for metabolite identification into the following three categories: i) searching in spectral libraries; ii) in silico fragmentation; iii) machine learning (ML).

We focus on the ML approach, where the common scheme is, given a set of MS/MS spectra, to learn a map from a MS/MS spectrum to a molecule (Fig. 2). This has two steps: i) fingerprint prediction: predict a fingerprint with supervised ML, and ii) candidate retrieval: use the predicted fingerprint to query the database. Our particular focus is the first step, which predicts the fingerprint that is a binary (or rarely real-valued) vector, indicating the presence (or absence) of certain substructure. This step has been tackled by many ML methods, including linear discriminative analysis (LDA) (Imre *et al.*, 2008) and decision tree (Hummel *et al.*, 2010). A notable method is FingerID (Heinonen *et al.*, 2012), 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.*, 2004).

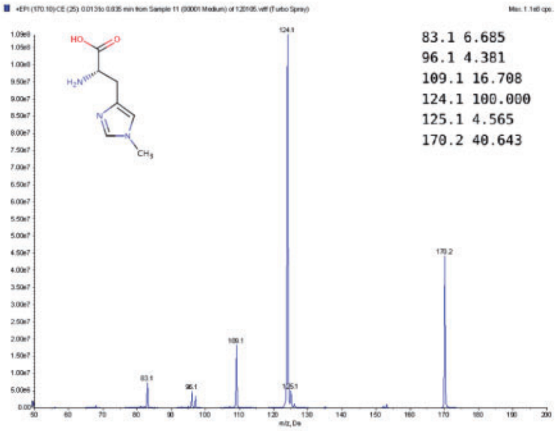

Fig. 1. Example MS spectrum from Human Metabolome Database (Wishart *et al.*, 2012) for 1-Methylhistidine (HMBD00001), with its corresponding chemical structure (top-left) and peak list (top-right)

We point out three drawbacks of the existing ML approaches: i) 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. ii) There is an interesting (not ML-based) software, which outputs, 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.*, 2015; Shen *et al.*, 2014) 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. iii) The size/length of a fingerprint is not short, while the number of peaks is usually small, meaning that most elements of a fingerprint vector are zero. So the training data is very sparse, while sparse models have not been considered yet. In addition, sparse models are advantageous in that their results are easily interpretable.

Motivated by these drawbacks, we address the following two problems: i) incorporation of peak interactions into the learning model to improve the predictive performance; ii) introduction of sparsity into the models for interpretation. For the above i), we

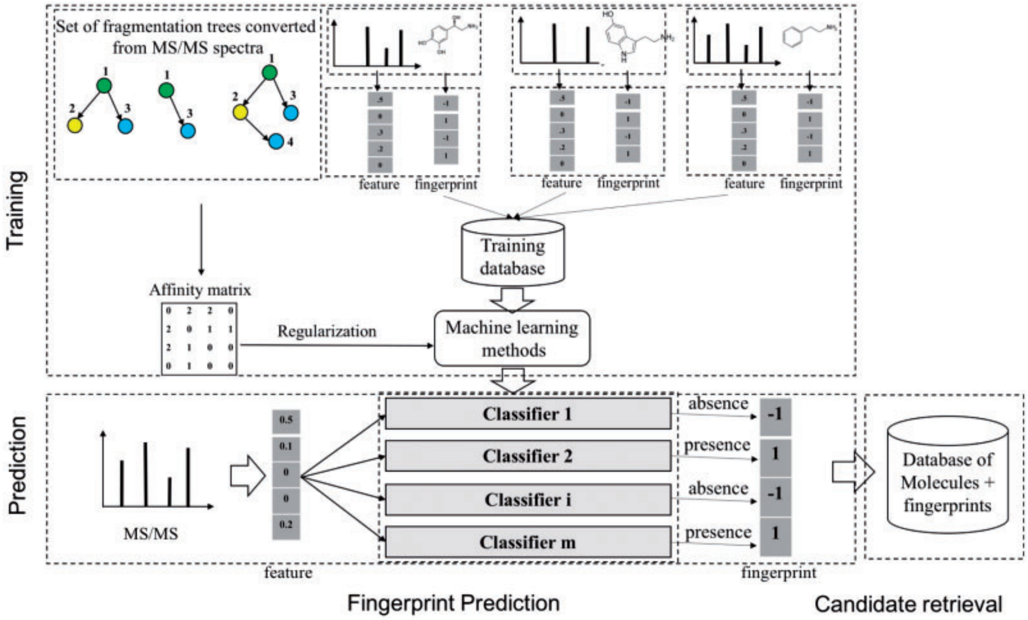

Fig. 2. A general scheme to identify unknown metabolites based on molecular fingerprint vectors. There are two main stages: 1) fingerprint prediction: learning a mapping from a molecule to the corresponding binary molecular fingerprint vector by classification methods, given a set of MS/MS spectra and fingerprints; 2) candidate retrieval: using the predicted fingerprints to retrieve candidate molecules from the databases of known metabolites. Note that the step of constructing an affinity matrix is optional and is used in L-SIMPLE only

propose a kernel for peak interactions and combine this kernel with other kernels through multiple kernel learning (MKL). For the above both i) and ii), 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.*, 2014). We formulate the model as a convex optimization problem, 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.*, 2014). We found that incorporation of peak interactions can significantly improve the prediction 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 predictive performance.

2 Related work

In Figure 2, 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.*, 2004), 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 3.1.1.

Also we briefly explain the current cutting-edge ML method by MKL, CSI: FingerID (Dührkop *et al.*, 2015; Shen *et al.*, 2014), 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.*, 2011). 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, kernels are difficult to interpret and deal with sparse data, such as MS/MS spectra, where each spectrum (fingerprint vector) has only a few number of peaks (nonzeros).

3 Materials and methods

3.1 Kernel method

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

3.1.1 Preliminary: kernel for peaks

MS/MS spectra have peak information, 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.*, 2004) was developed 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 Gaussians.

More specifically, given a mass spectrum **S**, being a list of peaks, i.e. $\{(m_1, I_1), (m_2, I_2), \ldots, (m_{N_\mathbf{S}}, I_{N_\mathbf{S}})\}$. The *k*th peak of the spectrum **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: $\Sigma = \mathrm{diag}(\sigma_m^2, \sigma_I^2)$, where σ_m^2 and σ_I^2 are the variances for the mass and intensity, respectively. Hence, the spectrum **S** can be represented as a uniform mixture of the peaks contained in it, i.e. $p(\mathbf{S}) = \frac{1}{N_\mathbf{S}}\sum_{k=1}^{N_\mathbf{S}} p_k$. 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 **S** and $\mathbf{S}'$ is given by:

$$\mathbf{K}_{peak}(\mathbf{S}, \mathbf{S}') = \int_{\mathbb{R}^2} p_\mathbf{S}(x) q_{\mathbf{S}'}(x) dx \tag{1}$$

$$= \frac{1}{N_\mathbf{S} N_{\mathbf{S}'}} \sum_{\substack{1 \le i \le N_\mathbf{S} \\ 1 \le j \le N_{\mathbf{S}'}}} \mathbf{K}(p_i, q_j) \tag{2}$$

where $\mathbf{K}(p, q)$ is the PPK between two component Gaussian distributions p and q, computed by (3).

$$\mathbf{K}(p, q) = \frac{1}{4\pi\sigma_m\sigma_I} \exp\left(-0.5\left(\frac{\left(\mu_p^m - \mu_q^m\right)^2}{\sigma_m} + \frac{\left(\mu_p^I - \mu_q^I\right)^2}{\sigma_I}\right)\right) \tag{3}$$

where μ_p^m and μ_p^I denote the mass and intensity values of the peaks corresponding to p.

3.1.2 Kernel for peak interactions

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

$$\mathbf{K}_{interaction}(\mathbf{S}, \mathbf{S}') = \sum_{\substack{1 \le i \le j \le N_\mathbf{S} \\ 1 \le k \le l \le N_{\mathbf{S}'}}} \mathbf{K}(p_i, q_k) * \mathbf{K}(p_j, q_l), \tag{4}$$

where **K** is the function defined by (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.

3.1.3 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.*, 2012).

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:

$$\mathbf{K}_\alpha = \sum_{j=1}^{m} \alpha_j \mathbf{K}_j \tag{5}$$

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 dataset. For the purposes of MKL, alignment is often measured between the combined kernel in (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:

$$\mathbf{K}_c = \left[\mathbf{I} - \frac{\mathbf{e}\mathbf{e}^T}{n}\right]\mathbf{K}\left[I - \frac{\mathbf{e}\mathbf{e}^T}{n}\right] \tag{6}$$

where **I** is the identity matrix and **e** is the vector with all ones.

The centered alignment between two kernels **K** and **K**′ is defined by:

$$\rho(\mathbf{K}, \mathbf{K}') = \frac{< \mathbf{K}_c, \mathbf{K}'_c >}{||\mathbf{K}_c||_F ||\mathbf{K}_c||_F} \tag{7}$$

$$\text{where } < \mathbf{K}_c, \mathbf{K}'_c > = \text{trace}\left(\mathbf{K}_c^T \mathbf{K}'_c\right) \text{ and } ||\mathbf{K}_c||_F = \sqrt{\text{trace}\left(\mathbf{K}_c^T \mathbf{K}_c\right)} \tag{8}$$

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

$$\alpha^* = \underset{\alpha}{\text{argmax}} \frac{< \mathbf{K}_\alpha, \mathbf{K}_y >}{||\mathbf{K}_\alpha||_F} \tag{9}$$

$$\text{subject to } ||\alpha|| = 1 \ , \ \alpha \geq 0 \tag{10}$$

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

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].

3.2.1 Prediction model

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

$$f(\mathbf{x}; w, W) = b + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} W_{ij} x_i x_j \tag{11}$$

$$= b + w^T \mathbf{x} + \mathbf{x}^T W \mathbf{x} \tag{12}$$

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. 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 $y(\mathbf{x}) = \text{sign}(f(\mathbf{x}; w, W)) \in \{-1, 1\}$.

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 ith 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:

$$\min_{b,w,W} \sum_{i=1}^{n} l(\mathbf{y}_i, f(\mathbf{x}_i; w, W)) + \mathcal{R}(w) + \mathcal{R}(W) \tag{13}$$

where $l(\mathbf{y}, \widehat{\mathbf{y}})$ is the hinge loss function and computed by $l(\mathbf{y}, \widehat{\mathbf{y}}) = (1 - \mathbf{y}, \widehat{\mathbf{y}})_+$, 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. $W \succcurlyeq 0$. Therefore, putting all above things together, the objective function of **SIMPLE** becomes:

$$\begin{aligned} &\min_{b,w,W} && \sum_{i=1}^{n} l(\mathbf{y}_i, f(\mathbf{x}_i; w, W)) + \alpha||w||_1 + \beta||W||_* \\ &\text{subject to} && W \succcurlyeq 0 \end{aligned} \tag{14}$$

where α and β are hyperparameters to control the sparsity of w and low-rankness of W, respectively. Note that (15) is the convex function which guarantees to find the global optimal solution.

3.2.2 Fragmentation trees (FTs) as prior information

As the number of interactions is large, it would be a good idea to regularize (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: i) How to represent the structural information from FTs; ii) 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 (Fig. 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 ith and jth bins of spectrum x and correspond to representation of ith and jth features in the space $\mathbb{R}^k$.

We assume that if ith and jth 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 (14):

$$\mathcal{R}(V) = \sum_{i,j=1}^{d} A_{ij}||v_i - v_j||_2^2 \quad (15)$$

$$= \text{trace}\big(V^T LV\big) \quad (16)$$

$$= \text{trace}\big(VV^T L\big) = \text{trace}(WL) \quad (17)$$

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**:

$$\min_{b,w,W} \quad \sum_{i=1}^{n} l(\mathbf{y}_i, \mathbf{f}_i) + \alpha||w||_1 + \beta||W||_* + \gamma\text{trace}(WL), \quad (18)$$
$$\text{subject to} \quad W \succcurlyeq 0$$

where $\mathbf{f}_i = f(\mathbf{x}_i; w, W)$, γ is an additional hyperparameter to control Laplacian smoothness of the objective function (18). It is noteworthy that, our derived formulation is still convex due to convexity of regularization terms, thus, the solution of (18) is guaranteed to be the global optimal. Below we will present an optimization method for efficiently solving (18).

3.2.3 Optimization: ADMM algorithm

Equation (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 $\mathbf{C} = (C_1, C_2, \ldots, C_n)^T$ and $C_i = 1 - \mathbf{y}_i\mathbf{f}_i$, then (18) can be reformulated into the following equivalent constrained problem:

$$\min_{b,w,W,\mathbf{C}} \quad \sum_{i=1}^{n} (C_i)_+ + \alpha||w||_1 + \beta||W||_* + \gamma\text{trace}(WL) \quad (19)$$
$$\text{subject to} \quad \mathbf{C} = \mathbf{1} - \mathbf{YF} \text{ and } W \succcurlyeq 0$$

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

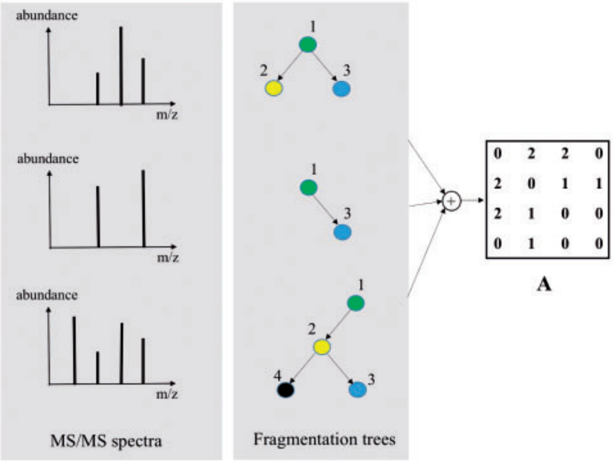

Fig. 3. Illustration of constructing affinity matrix A from the set of fragmentation trees. The constructed matrix A is used as prior information for regularizing interaction matrix W

The augmented Lagrange function of (19) is defined by:

$$\mathcal{L}(w_0, w, W, \mathbf{C}, \mathbf{u}) = \sum_{i=1}^{n} (C_i)_+ + \alpha||w||_1 + \beta||W||_* + \gamma\text{trace}(WL)$$
$$+\mathbf{u}^T(1 - \mathbf{YF} - \mathbf{C}) + \frac{1}{2}||1 - \mathbf{YF} - \mathbf{C}||_2^2 \quad (20)$$

where $\mathbf{u} = [u_1, u_2, \ldots, u_n]^T$ is a dual variable corresponding to the constraint $\mathbf{C} = 1 - \mathbf{YF}$. Note that the constraint $W \succcurlyeq 0$ is not included in (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 (20) through an iterative algorithm between the primal and the dual optimization as follows:

$$\begin{cases} b^{t+1}, w^{t+1} & = \text{argmin}_{b,w}\, \mathcal{L}(b, w, W^t, \mathbf{C}^t, \mathbf{u}^t) \\ W^{t+1} & = \text{argmin}_W\, \mathcal{L}\big(b^{t+1}, w^{t+1}, W, \mathbf{C}^t, \mathbf{u}^t\big) \\ \mathbf{C}^{t+1} & = \text{argmin}_{\mathbf{C}}\, \mathcal{L}\big(b^{t+1}, w^{t+1}, W^{t+1}, \mathbf{C}, \mathbf{u}^t\big) \\ \mathbf{u}^{t+1} & = \mathbf{u}^t + 1 - \mathbf{YF}^{t+1} - \mathbf{C}^{t+1} \end{cases} \quad (21)$$

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 (20) depends on whether the subproblems in (21) can be solved quickly. Algorithm 1 summarizes the ADMM steps for solving the optimization problem (20). To avoid confusion, the detailed derivation of the update rules for subproblems of (21) are in Appendix.

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.
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.

Table 1. Micro-average performance of kernels: PPK (Heinonen *et al.*, 2012) is used to compute $\mathbf{K}_{peak}$

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

Note: ComUNIMKL, ALIGN, ALIGNF are combinations of $\mathbf{K}_{peak}$ and $\mathbf{K}_{interaction}$ by algorithms UNIMKL, ALIGN, ALIGNF, respectively.

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.

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?

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.*, 2014), and followed the same preprocessing steps as in (Shen *et al.*, 2014): for each compound, peaks recorded from different energies were merged for each MS/MS spectra. Then, spectra were normalized such that the sum of intensities is up to 100%. 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.

4.2 Benefit of incorporating interaction

We show the benefit of incorporating peak interactions using kernel methods described in Section 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 $C \in \{2^{-3}, 2^{-2}, \ldots, 2^{6}, 2^{7}\}$.

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

Algorithm 1 ADMM algorithm for optimizing the objective function (19).

```
Inputs:
        A set of MS/MS spectra $\mathbf{X} = \left[\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_n^T\right]^T$ and
        associated output $\mathbf{Y} = [y_1, y_2, \ldots, y_n]^T$. Laplacian $L$
        (only for L-SIMPLE).
Outputs:
        weight vector $w$ and interaction matrix $W$.
Initialize:
        $b \leftarrow 0, w \leftarrow \mathbf{0}, W \leftarrow \mathbf{0}, \mathbf{C} \leftarrow 0, \mathbf{u} \leftarrow 0$
while not converged do
    1. Fix W, C, u and update b, w
    Precompute $\widehat{\mathbf{F}}_1 \leftarrow \mathbf{Y}(\mathbf{1} - \mathbf{C} + \mathbf{u}) - \mathrm{diag}\left(\mathbf{X}W\mathbf{X}^T\right)$
    while not converged do
        $\begin{cases} \mathbf{z} \leftarrow w - \rho\mathbf{X}^T\left(\mathbf{X}w + b - \widehat{\mathbf{F}}_1\right) \\ w \leftarrow \mathcal{S}_\alpha(z) \\ b \leftarrow \frac{1}{n}\mathrm{sum}\left(\widehat{\mathbf{F}}_1 - \mathbf{X}w\right) \end{cases}$
    end while
    2: Fix $b$, $w$, C, u and update $W$
    precompute $\widehat{\mathbf{F}}_2 \leftarrow \mathbf{Y}(1 - \mathbf{C} + \mathbf{u}) - b - \mathbf{X}w$
    while not converged do
        $\begin{cases} \mathbf{R} \leftarrow \mathrm{diag}\left(\mathbf{X}W\mathbf{X}^T\right) - \widehat{\mathbf{F}}_2 \\ \Delta_W\mathcal{L} \leftarrow \mathbf{X}^T\mathbf{R}\mathbf{X} \\ \mathbf{Z} \leftarrow W - \rho(\Delta_W\mathbf{L} + \gamma L) \\ \mathbf{U}, \mathbf{E} \leftarrow \mathbf{EVD}(\mathbf{Z}) \\ \widehat{\mathbf{E}} \leftarrow \mathcal{S}_\beta(\mathbf{E}) \\ W \leftarrow \mathbf{U}\widehat{\mathbf{E}}\mathbf{U}^T \end{cases}$
    end while
    3: Update C
    $\mathbf{F} = b + \mathbf{X}w + \mathrm{diag}\mathbf{X}W\mathbf{X}^T$, $\mathbf{C} \leftarrow \mathcal{T}_1(1 - \mathbf{Y}\mathbf{F} + \mathbf{u})$
    4: Update the dual variable u
    $\mathbf{u} \leftarrow \mathbf{u} + \mathbf{1} - \mathbf{Y}\mathbf{F} - \mathbf{C}$
end while
```

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

Since kernel methods in Section 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 was 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 sparsity,

Table 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

Table 3. Micro-average performance and computation time (for prediction) of kernel-based methods in Shen *et al.* (2014) 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 (±7.03)	61.64 (±15.48)	1501.02
LC (Loss count)	75.33 (±5.4)	61.25 (±13.99)	1501.02
LI (Loss intensity)	74.54 (±8.49)	58.46 (±16.01)	1501.02
NB (Node binary)	79.11 (±5.02)	67.34 (±11.75)	1501.09
NI (Node intensity)	78.41 (±4.99)	66.87 (±12.11)	1501.01
CPC (Common path count)	79.02 (±7.4)	67.55 (±12.93)	1501.11
ComFT (combining all above)	80.98 (±6.05)	69.04 (±11.98)	1559.20
ComALIGNF (Proposed: MKL)	79.03 (±7.89)	65.67 (±13.02)	471.71
SIMPLE (Proposed)	78.33 (±6.05)	66.70 (±13.03)	4.57
L-SIMPLE (Proposed)	78.86 (±5.87)	67.59 (±12.35)	4.32

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 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.*, 2014). 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 and is efficiently computable in prediction.

Table 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 dataset. The first four methods including PPK achieved around the accuracy of 75%, which is clearly worse than the other methods, which achieved around 78–80% 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 than 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 5 ms. 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.

4.4 Model interpretation

One advantage of sparse learning models over kernel based methods is interpretation. For illustration purposes, Figure 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 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.

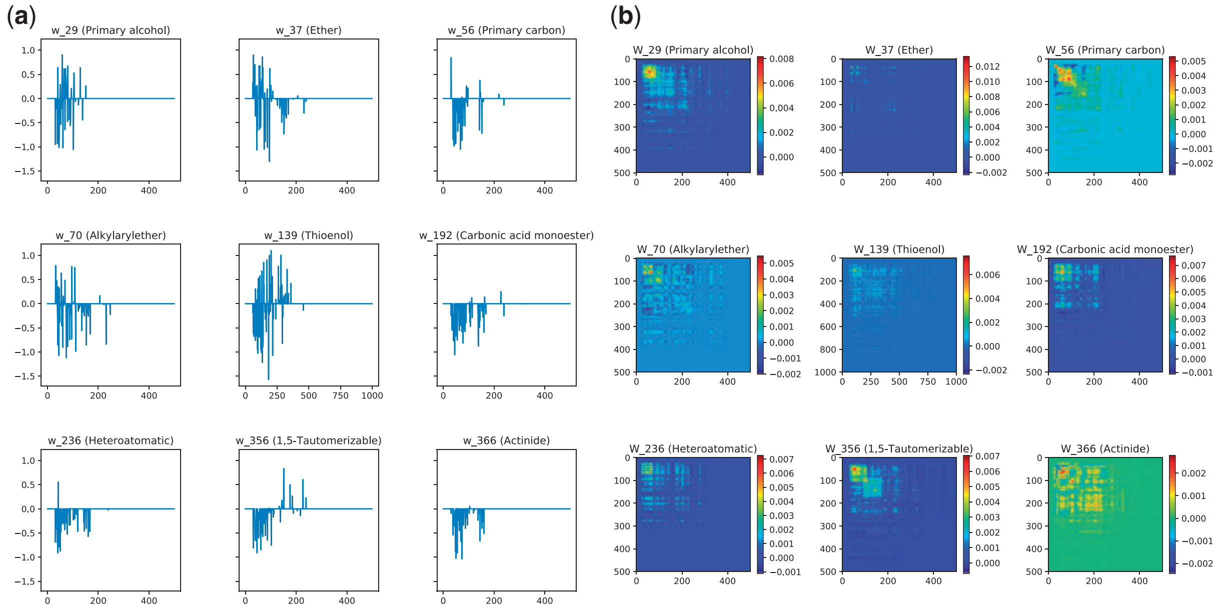

Fig. 4. (**a**) Weight vectors (*w*) of the main effect terms and (**b**) smooth heat map of weight matrices (*W*) of the interaction terms learned by L-SIMPLE for properties or tasks: 29 (Primary alcohol), 37 (Ether), 56 (Primary Carbon), 70 (Alkylarylether), 139 (Thioenol), 192 (Carbonic acid monoester), 236 (Heteroaromatic), 356 (1,5-Tautomerizable) and 366 (Actinide)

Table 4. Case studies of weight vector *w* and interaction matrix *W* learned by **L-SIMPLE** over a set of randomly selected tasks

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

Note: 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.

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.

5 Discussion and conclusion

The goal of this work 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.* (2014) used fragmentation trees for prior structural information of spectral, and 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 very few number of mass positions in each given spectrum.

Our sparse interaction models, **(L-)SIMPLE**, have a number of advantages, which are summarized in Section 3.3: (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.

Funding

This work was partially supported by MEXT KAKENHI Grant Number 16H02868, Grant Number JPMJAC1503 ACCEL JST, FiDiPro Tekes (currently Business Finland) and AIPSE Academy of Finland.

Conflict of Interest: none declared.

References

Blondel,M. *et al.* (2015). Convex factorization machines. In Appice,A. *et al.* (eds.) *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, Cham, pp. 19–35.

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

Burden,F.R. and Winkler,D.A. (2015) Relevance vector machines: sparse classification methods for qsar. *J. Chem. Inf. Model.*, **55**, 1529–1534.

Cai,J.-F. *et al.* (2010) A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.*, **20**, 1956–1982.

Cortes,C. *et al.* (2012) Algorithms for learning kernels based on centered alignment. *J. Mach. Learn. Res.*, **13**, 795–828.

de Hoffmann,E. and Stroobant,V. (2007) *Mass Spectrometry, Principles and Applications*, 3rd edn. John Wiley & Sons.

Dührkop,K. *et al.* (2015) Searching molecular structure databases with tandem mass spectra using csi: fingerid. *Proc. Natl. Acad. Sci. USA*, **112**, 12580–12585.

Gönen,M. and Alpaydin,E. (2011) Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, **12**, 2211–2268.

Heinonen,M. *et al.* (2012) Metabolite identification and molecular fingerprint prediction through machine learning. *Bioinformatics*, **28**, 2333–2341.

Horai,H. *et al.* (2010) Massbank: a public repository for sharing mass spectral data for life sciences. *J. Mass Spectrom.*, **45**, 703–714.

Hummel,J. *et al.* (2010) Decision tree supported substructure prediction of metabolites from gc-ms profiles. *Metabolomics*, **6**, 322–333.

Imre,T. *et al.* (2008) Mass spectrometric and linear discriminant analysis of n-glycans of human serum alpha-1-acid glycoprotein in cancer patients and healthy individuals. *J. Proteomics*, **71**, 186–197.

Jebara,T. *et al.* (2004) Probability product kernels. *J. Mach. Learn. Res.*, **5**, 819–844.

Kloft,M. *et al.* (2011) lp-norm multiple kernel learning. *J. Mach. Learn. Res.*, **12**, 953–997.

Ma,S. *et al.* (2011) Fixed point and bregman iterative methods for matrix rank minimization. *Math. Program.*, **128**, 321–353.

O'Boyle,N.M. *et al.* (2011) Open babel: an open chemical toolbox. *J. Cheminf.*, **3**, 33.

Rasche,F. *et al.* (2011) Computing fragmentation trees from tandem mass spectrometry data. *Anal. Chem.*, **83**, 1243–1251. PMID: 21182243.

Scheubert,K. *et al.* (2013) Computational mass spectrometry for small molecules. *J. Cheminf.*, **5**, 12.

Shen,H. *et al.* (2014) Metabolite identification through multiple kernel learning on fragmentation trees. *Bioinformatics*, **30**, i157–i164.

Tibshirani,R. (1994) Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser. B*, **58**, 267–288.

Watanabe,T. *et al.* (2014) Scalable fused lasso svm for connectome-based disease prediction. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5989–5993.

Wishart,D.S. (2007) Current progress in computational metabolomics. *Brief. Bioinf.*, **8**, 279–293.

Wishart,D.S. *et al.* (2012) Hmdb 3.0 – the human metabolome database in 2013. *Nucleic Acids Res.*, **41**, D801–D807.

Appendix

We will derive the updating rules of ADMM for steps in (21).

A1. ADMM for updating *b* and *w*

The subproblem for optimizing b, w in (21) is equivalent to

$$\begin{aligned} b^{t+1}, w^{t+1} &= \underset{b,w}{\operatorname{argmin}} \quad \mathcal{L}(b, w, W^t, \mathbf{C}^t, \mathbf{u}^t) \\ &= \underset{b,w}{\operatorname{argmin}} \quad \alpha||w||_1 + \frac{1}{2}||\mathbf{1} - \mathbf{Y}\mathbf{F}^t - \mathbf{C} + \mathbf{u}||_2^2 \\ &= \underset{b,w}{\operatorname{argmin}} \quad \alpha||w||_1 + \frac{1}{2}||\mathbf{X}w + b\mathbf{1} - \widehat{\mathbf{F}}_1||_2^2 \end{aligned} \tag{22}$$

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

In fact, solving (22) can be done easily with proximal gradient descent through alternately updating estimate of w and b as follows:

$$\begin{cases} \mathbf{z}^{k+1} & = w^k - \delta_w \mathbf{X}^T\left(\mathbf{X}w^k + b^k - \widehat{\mathbf{F}}_1\right) \\ w^{k+1} & = \mathcal{S}_\alpha\left(\mathbf{z}^{k+1}\right) \\ b^{k+1} & = \dfrac{1}{n}\operatorname{sum}\left(\widehat{\mathbf{F}}_1 - \mathbf{X}w^{k+1}\right) \end{cases} \tag{23}$$

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:

$$\mathcal{S}_\lambda(t) = \operatorname{sign}(t)\max(0, |t| - \lambda) \tag{24}$$

A2. ADMM for updating *W*

The subproblem of optimizing W in (22) is equivalent to:

$$\begin{aligned} W^{t+1} &= \underset{W}{\operatorname{argmin}}\mathcal{L}\left(b^{t+1}, w^{t+1}, W, \mathbf{C}^t, \mathbf{u}^t\right) \\ &= \underset{W}{\operatorname{argmin}}\beta||W||_* + \gamma\operatorname{trace}(WL) \\ &\quad + \frac{1}{2}||\mathbf{1} - \mathbf{Y}\mathbf{F}^t - \mathbf{C}^t + \mathbf{u}^t||_2^2 \\ &= \underset{W}{\operatorname{argmin}}\beta||W||_* + \gamma\operatorname{trace}(WL) \\ &\quad + \frac{1}{2}||\operatorname{diag}\left(\mathbf{X}W\mathbf{X}^T\right) - \widehat{\mathbf{F}}_2||_2^2 \end{aligned} \tag{25}$$

where $\widehat{\mathbf{F}}_2 = \mathbf{Y}(1 - \mathbf{C} + \mathbf{u}) - b1 - \mathbf{X}w$. It is obvious that the objective function (25) split in two components: $\gamma\operatorname{trace}(WL) + \frac{1}{2}||\operatorname{diag}\left(\mathbf{X}W\mathbf{X}^T\right) - \widehat{\mathbf{F}}_2||_2^2$, which is convex, differentiable and $\beta||W||_*$, which is also convex with inexpensive proximal operator. It is known that the solution of (25) 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:

$$\begin{cases} \mathbf{R}^{k+1} & = \operatorname{diag}\left(\mathbf{X}W^k\mathbf{X}^T\right) - \widehat{\mathbf{F}}_2 \\ \Delta_W\mathcal{L} & = \gamma L + \mathbf{X}^T\mathbf{R}^{k+1}\mathbf{X} \\ \mathbf{Z}^{k+1} & = W^k - \delta_W\Delta_W\mathcal{L} \\ \mathbf{U}^{k+1}, \mathbf{E}^{k+1} & = \mathbf{EVD}\left(\mathbf{Z}^{k+1}\right) \\ \widehat{\mathbf{E}}^{k+1} & = \mathcal{S}_\beta\left(\mathbf{E}^{k+1}\right) \\ W^{k+1} & = \mathbf{U}^{k+1}\widehat{\mathbf{E}}^{k+1}\mathbf{U}^{k+1} \end{cases} \tag{26}$$

where $\Delta_W \mathcal{L}$ is the derivative of the differentiable component of (26) 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. **U** and **E** are columns of eigenvectors and diagonal matrix of eigenvalues obtained by applying eigendecomposition (**EVD**) to **Z**, by which we can guarantee the semidefiniteness of weight matrix W after each iteration.

A3. ADMM for updating C

For the subproblem of optimizing **C** in (22), it is equivalent to

$$\begin{aligned}\mathbf{C}^{t+1} &= \underset{\mathrm{C}}{\operatorname{argmin}}\ \mathcal{L}\left(b^{t+1}, w^{t+1}, W^{t+1}, \mathbf{C}, \mathbf{u}^t\right) \\ &= \underset{\mathrm{C}}{\operatorname{argmin}} \sum_{i=1}^{n} (C_i)_+ + \frac{1}{2}||\mathbf{1} - \mathbf{YF}^{t+1} + \mathbf{u}^t - \mathbf{C}||^2\end{aligned} \tag{27}$$

In order to solve (27), 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:

$$\mathcal{T}_\lambda(t) = \begin{cases} t - \lambda & \text{if } t \ \geq \lambda \\ 0 & \text{if } 0 \leq t \leq \lambda \\ t & \text{if } t \ < 0 \end{cases} \tag{28}$$

Note that components of **C** are independent of each other in (27). By applying Proposition 1 in element-wise, we can derive the update for C in the following closed form solution:

$$\mathbf{C}^{t+1} = \mathcal{T}_1\left(1 - \mathbf{YF}^{t+1} + \mathbf{u}^t\right) \tag{29}$$