BIOINFORMATICS

Vol. 30 ISMB 2014, pages i157–i164
doi:10.1093/bioinformatics/btu275

Metabolite identification through multiple kernel learning on fragmentation trees

Huibin Shen[1,2,*], Kai Dührkop[3], Sebastian Böcker[3] and Juho Rousu[1,2]

[1]Department of Information and Computer Science, Aalto University, Espoo, Finland, [2]Helsinki Institute for Information Technology, Espoo, Finland and [3]Chair for Bioinformatics, Friedrich Schiller University Jena, Jena, Germany

ABSTRACT

Motivation: Metabolite identification from tandem mass spectrometric data is a key task in metabolomics. Various computational methods have been proposed for the identification of metabolites from tandem mass spectra. Fragmentation tree methods explore the space of possible ways in which the metabolite can fragment, and base the metabolite identification on scoring of these fragmentation trees. Machine learning methods have been used to map mass spectra to molecular fingerprints; predicted fingerprints, in turn, can be used to score candidate molecular structures.

Results: Here, we combine fragmentation tree computations with kernel-based machine learning to predict molecular fingerprints and identify molecular structures. We introduce a family of kernels capturing the similarity of fragmentation trees, and combine these kernels using recently proposed multiple kernel learning approaches. Experiments on two large reference datasets show that the new methods significantly improve molecular fingerprint prediction accuracy. These improvements result in better metabolite identification, doubling the number of metabolites ranked at the top position of the candidates list.

Contact: huibin.shen@aalto.fi

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Metabolomics deals with the analysis of small molecules and their interactions in living cells. A central task in metabolomics experiments is the identification and quantification of the metabolites present in a sample. This is mandatory for subsequent analysis steps such as metabolic pathway analysis and flux analysis (Pitkänen *et al.*, 2010). Mass spectrometry (MS) is one of the two predominant analytical technologies for metabolite identification. Identification is done by fragmenting the metabolite, for example, by tandem MS (MS/MS), and measuring the mass-to-charge ratios of the resulting fragment ions. The measured mass spectra contain information about the metabolite, but extracting the relevant information is a highly non-trivial task.

Several computational methods have been suggested to identify the metabolites from MS/MS spectra. Mass spectral databases (spectral libraries) have been created (e.g. Hisayuki *et al.*, 2010; Oberacher *et al.*, 2009; Smith *et al.*, 2005; Tautenhahn *et al.*, 2012), which allow us to search measured mass spectra. Unfortunately, this approach can only identify 'known unknowns' where a reference measurement is available.

Fragmentation trees are combinatorial models of the MS/MS fragmentation process. Böcker and Rasche (2008) suggested fragmentation trees for identifying the molecular formula of an unknown compound. Later, fragmentation trees were shown to contain valuable structural information about the compound (Rasche *et al.*, 2011, 2012).

The relation between spectral and structural similarities has been studied by Demuth *et al.* (2004). A kernel-based machine learning approach for metabolite identification was recently introduced by Heinonen *et al.* (2012), relying on predicting the molecular fingerprints as an intermediate step. Molecular fingerprints are given as bit vectors with each bit describing the existence of certain molecular property such as substructures in the molecule. After the prediction, imposing some scoring strategy, the predicted molecular fingerprints are used for searching some chemical database and finally the ranked list of candidates are generated (Heinonen *et al.*, 2012; Shen *et al.*, 2013).

Besides these two approaches, methods have been suggested for predicting MS/MS spectra from molecular structures (Allen *et al.*, 2013; Kangas *et al.*, 2012); commercial software packages also exist for this task. Such simulated spectra can be used to replace the notoriously incomplete spectral libraries by molecular structure databases (Hill *et al.*, 2008). Combinatorial fragmentation of molecular structure serves the same purpose (Gerlich and Neumann, 2013; Wolf *et al.*, 2010). Finally, we can search spectral libraries for similar compounds, by comparing either MS/MS spectra (Demuth *et al.*, 2004; Gerlich and Neumann, 2013) or fragmentation trees (Rasche *et al.*, 2012). See Scheubert *et al.* (2013) and Hufsky *et al.* (2014) for recent reviews.

We propose a joint strategy that combines fragmentation trees and multiple kernel learning (MKL) to improve molecular fingerprint prediction and, subsequently, the metabolite identification. We first outline the metabolite identification framework and introduce fragmentation trees and their computation. Next, we introduce a family of kernels for fragmentation trees, consisting of simple node and edge statistics kernels as well as path and subtree kernels that use dynamic programming (DP) for efficient computation. We then describe state-of-the-art methods for MKL. In these experiments, we evaluate different MKL algorithms with regards to the fingerprint prediction and the metabolite identification.

2 METHODS

Figure 1 gives an overview for our metabolite identification framework through MKL. Fragmentation trees are computed first, followed by the computation of kernels. MKL approaches are used to integrate different kernels for molecular fingerprint prediction. The final step of the framework is to query molecular structure databases with the predicted molecular fingerprint using a probabilistic scoring function.

*To whom correspondence should be addressed.

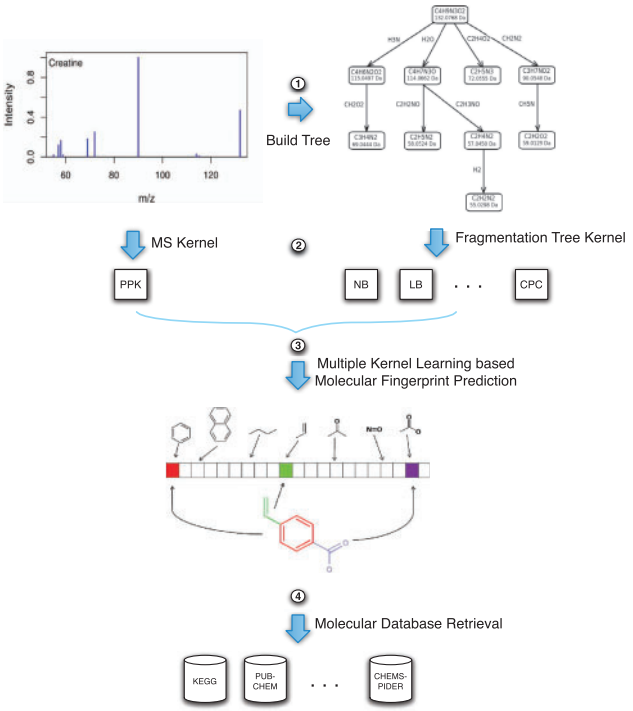

Fig. 1. The metabolite identification framework through MKL. First, we construct the fragmentation tree from the MS/MS spectrum. Second, we compute kernels for both MS/MS data and fragmentation trees. Third, MKL is used to combine kernels and predict molecular fingerprints. Finally, fingerprints are used for molecular structure database retrieval

The advantages of the kernel-based machine learning framework are: that it easily allows incorporating the combinatorial fragmentation trees by kernelizing the model; that it can query molecular structure databases which are much larger than MS/MS spectral libraries; and that molecular fingerprints can help to characterize the unknown metabolite and may shed light for *de novo* identification.

2.1 Fragmentation trees

Böcker and Rasche (2008) introduced fragmentation trees to predict the molecular formula of an unknown compound using its MS/MS spectra. A fragmentation tree annotates the MS/MS spectra of a compound via assumed fragmentation processes. Nodes are molecular formulas, representing the unfragmented molecule and its fragments. Edges represent fragmentation reactions between fragments, or the unfragmented molecule and a fragment. Details on the computation can be found in Böcker and Rasche (2008) and Rasche *et al.* (2011); here, we quickly recapitulate the method. We assume that MS/MS spectra recorded at different collision energies have been amalgamated into a single spectrum, as described in Section 3. We decompose all peaks in the amalgamated spectrum, finding all molecular formulas that are within the mass accuracy of the measurement. For each decomposition of the parent peak, we build a fragmentation graph which contains all possible explanations for each peak, where nodes are colored by the peaks they originate from. We insert all edges between nodes that are not ruled out by the molecular formulas: that is, a product fragment can never gain atoms of any element through the fragmentation. Edges of this graph are then weighted, taking into account the intensity and mass accuracy of the product fragment, the mass of the loss and prior knowledge about the occurrence of certain losses.

Under the parsimony assumption, we then compute a colorful subtree of this graph with maximum weight. Unfortunately, finding this tree is an NP-hard problem (Rauf *et al.*, 2012). Nevertheless, we can compute optimal trees in a matter of seconds using Integer Linear Programming (Rauf *et al.*, 2012). For each peak, this tree implicitly decides whether it is noise or signal and, in the later case, assigns the molecular formula of the corresponding fragments and the fragmentation reaction it resulted from. The score of the tree is the sum of its edge weights. Candidate molecular formulas of the parent peak are ranked by this score, which is the maximum score of any tree that has this molecular formula as its root.

Different from Böcker and Rasche (2008) and Rasche *et al.* (2011), we used a modified weighting function for the edges of the fragmentation graph. With these new weights, the above optimization can be interpreted as a maximum a posteriori estimator of the observed data. We weight edges by the logarithmic likelihood that a certain fragmentation reaction occurs: for this, we consider the intensity and mass deviation of the product fragment peak, the loss mass and chemical properties of the molecular formula as proposed in Kind and Fiehn (2007): namely, the ring double bond equivalent and the hetero atoms and carbon atoms ratio. Furthermore, we favor a few common losses that were learned from the data, and penalize implausible losses and radicals. Such weights have already been used in Böcker and Rasche (2008) and Rasche *et al.* (2011); different from there, we did not choose parameters *ad hoc* but rather learned them from the data. Details about these new weights will be published elsewhere.

2.2 Kernels for fragmentation trees and MS/MS spectra

2.2.1 Probability product kernel Heinonen *et al.* (2012) compared several kernels that can be computed directly from the MS/MS spectra without the knowledge of the fragmentation trees. In their studies, simple peak and loss matching kernels were found inferior to the probability product kernel (PPK). Thus, we use the PPK as the baseline comparison with the fragmentation tree kernels. The idea of the PPK is the following: each peak in a spectrum is modeled by a 2D Gaussian distribution with the mass-to-charge ratio as one dimension, and the intensity as the other. All-against-all matching between the Gaussians is performed to avoid problems arising from alignment errors.

Formally, a spectrum is defined as $\chi = \{\chi(1), \ldots, \chi(\ell_\chi)\}$, a set of ℓ_χ peaks $\chi(k) = (\mu(k), \iota(k)) \in \mathbb{R}^2, (k = 1, \ldots, \ell_\chi)$ consisting of the peak mass $\mu(k)$ and the normalized peak intensity $\iota(k)$. The k-th peak of the mass spectrum χ is represented by $p_{\chi(k)} = \mathcal{N}(\chi(k), \Sigma)$ centered around the peak measurement and with covariance shared with all peaks

$$\Sigma = \begin{bmatrix} \sigma_\mu^2 & 0 \\ 0 & \sigma_\iota^2 \end{bmatrix}$$

where the variances σ_μ^2 for the mass is estimated from data and σ_ι^2 is tuned by cross-validation. No covariance is assumed between peak distributions. The spectrum χ is finally represented as a mixture of its peak distributions $p_\chi = \frac{1}{\ell_\chi} \sum_{k=1}^{\ell_\chi} p_{\chi(k)}$.

The PPK K_{peaks} (Jebara *et al.*, 2004) between the peaks of two spectra χ, χ' is given by:

$$\begin{aligned} K_{\text{peaks}}(\chi, \chi') &= K(p_\chi, p_{\chi'}) \\ &= \int_{\mathbb{R}^2} p_\chi(\mathbf{x}) p_{\chi'}(\mathbf{x}) d\mathbf{x} \\ &= \frac{1}{\ell_\chi \ell_{\chi'}} \sum_{k,k'=1}^{\ell_\chi, \ell_{\chi'}} \frac{1}{4\pi\sigma_\mu\sigma_\iota} \exp\left(-\frac{1}{2}\right. \\ &\quad \left.(\chi(k) - \chi'(k'))^T \Sigma^{-1} (\chi(k) - \chi'(k'))\right). \end{aligned}$$

The precursor ion is the compound selected in the first round of MS/MS and further fragmented in the second round. As a result, the difference (loss) between the peak $\chi(k)$ and the precursor ion $\mathrm{prec}(\chi) = (\mu(p),0)$ is also important, where $\mu(p)$ is the mass of the precursor ion. We can model the difference with distribution $p_{\hat{\chi}(k)} = \mathcal{N}(\hat{\chi}(k), \Sigma)$, where $\hat{\chi}(k) = |\mathrm{prec}(\chi) - \chi(k)|$. This feature is denoted as loss and corresponding kernel matrix as K_{loss}. Experiments in Heinonen *et al.* (2012) and Shen *et al.* (2013) showed that the combined kernel $K_{\mathrm{peaks}} + K_{\mathrm{loss}}$ achieved best accuracy and computational efficiency among the spectral kernels.

2.2.2 Fragmentation tree kernels Fragmentation trees can be considered as an annotated representation of the original MS/MS spectra. Recent advancement (Rasche *et al.*, 2012; Rojas-Chertó *et al.*, 2012) in comparing and aligning the fragmentation trees enables similarity metrics to be defined between fragmentation patterns for small molecules. Rasche *et al.* (2012) introduced fragmentation tree alignments, and showed alignment scores to be correlated with chemical similarity. However, alignment scores of this type do not, in general, yield positive semidefinite kernels. In the following, we define a set of kernels for fragmentation trees that will allow us to transfer the power of the fragmentation tree approach to the kernel-based learning algorithms for molecular fingerprint prediction and metabolite identification.

A fragmentation tree $T = (V, E)$ consists of a nodes set V of molecular formulas (corresponding to the fragments) and an edges set $E \subseteq V \times V$ (corresponding to the losses). Let r denote the root of T. For an edge $e = (u, v) \in E$ let $\lambda(e) = \lambda(u, v) := u - v$ be the molecular formula of the corresponding loss. Clearly, different edges may have identical losses; let $\lambda(E)$ be the multiset of all losses. For some loss molecular formula l, let $N(l)$ be the number of edges $e \in E$ with $\lambda(e) = l$. Each path from the root r to a node v implies a root loss $r - v$; let $\mathcal{E} := \{r - v : v \in V\}$ be the set of all root losses. For a MS/MS spectrum x, let $T_x = (V_x, E_x)$ be the corresponding fragmentation tree, with root losses $\mathcal{E}_x$ and loss multiplicities $N_x(\cdot)$. For any node $v \in V_x$ let $\iota_x(v)$ be the corresponding peak intensity; for an edge $e = (u, v) \in E_x$ let $\iota_x(e)$ be the intensity of the terminal node v.

For the loss- and node-based kernels, feature vectors ϕ are constructed and the kernel function is just a simple dot product between two feature vectors. Path-based kernels are more complicated, and details on their computation will be given below.

Loss-based kernels: edges in the fragmentation trees represent the losses from the parent node to the child node. The following feature vectors are devised based on the losses in a fragmentation tree T_x:

- `LB`: Loss binary, indicates the presence of a loss l in a fragmentation tree T_x, that is, $\phi_l^{\mathrm{LB}}(x) = 1_{l \in \lambda(E_x)}$.
- `LC`: Loss count, counts the number of occurrences of a loss l in a fragmentation tree T_x, that is, $\phi_l^{\mathrm{LC}}(x) = N_x(l)$.
- `LI`: Loss intensity, uses the average intensity of the terminal nodes with loss l in a fragmentation tree T_x, that is, $\phi_l^{\mathrm{LI}}(x) = \frac{1}{N_x(l)} \sum_{e \in E_x \lambda(e) = l} \iota_x(e)$.
- `RLB`: Root loss binary, indicates the presence of a root loss l in a fragmentation tree T_x, that is, $\phi_l^{\mathrm{RLB}}(x) = 1_{l \in \mathcal{E}_x}$.
- `RLI`: Root loss intensity uses the intensity of the terminal node of a root loss if it is present in a fragmentation tree T_x. For root r we set $\phi_l^{\mathrm{RLI}}(x) = \iota_x(r - l)$ if $r - l \in V_x$, and $\phi_l^{\mathrm{RLI}}(x) = 0$ otherwise.

Node-based kernels: the nodes in the fragmentation tree explain peaks in the MS/MS by some chemical formula of the hypothetical fragment. The nodes are unique in a fragmentation tree T, and so are the root losses. To this end, we can omit root losses from the feature vectors.

- `NB`: Nodes binary, indicates the presence of a node v in a fragmentation tree T_x, that is, $\phi_v^{\mathrm{NB}}(x) = 1_{v \in V_x}$.
- `NI`: Nodes intensity, uses the intensity of the node if it is presented in a fragmentation tree T_x; that is, $\phi_v^{\mathrm{NI}}(x) = \iota_x(v)$ for $v \in V_x$, and $\phi_v^{\mathrm{NI}}(x) = 0$ otherwise.

Path-based kernels: these kernels are count common path between two fragmentation trees—here, 'common path' refers to an identical sequence of losses in the two trees. We use DP to efficiently count the number of common paths, that is, the dot product of two feature vectors which are not explicitly constructed. For two fragmentation trees $T_1 = (V_1,E_1)$ and $T_2 = (V_2,E_2)$ we compute a DP table $D[u,v]$ for all $u \in V_1$ and $v \in V_2$. In all cases, the number of common paths is $D[r_1,r_2]$ where r_i is the root of T_i. We initialize

$$D[u, v] = 0, \forall u \in \mathcal{L}(T_1), v \in T_2$$
$$D[u, v] = 0, \forall u \in T_1, v \in \mathcal{L}(T_2)$$

where $\mathcal{L}(T)$ denotes the leaves of a tree T. Let $C(v)$ be the children of a node v.

- Common path counting (`CPC`). The DP table entry $D[u,v]$ records the count of common path for the subtrees rooted in u and v, respectively. This leads to the following recurrence:

$$D[u, v] = \sum_{\substack{a \in C(u), b \in C(v) \\ \lambda(u,a) = \lambda(v,b)}} (1 + D[a, b]).$$

- Common paths of length 2 (`CP2`). In this case, only common losses for paths of length two are considered:

$$D[u, v] = \sum_{\substack{x \in C(u), a \in C(x), y \in C(v), b \in C(y) \\ \lambda(u,x) = \lambda(v,y), \lambda(x,a) = \lambda(y,b)}} (1 + D[a, b]).$$

- Common path with K_{peaks} (`CPK`). Instead of simply counting the common paths, we use the PPK K_{peaks} to score the terminal peaks. We omit the straightforward but somewhat tedious details.
- Common subtree counting (`CSC`). In this case, we count the number of 'common subtrees' between T_1 and T_2, which can be defined analogously to the common paths above. Entry $D[u,v]$ now counts the number of common subtrees for the two subtrees rooted in u of T_1, and v of T_2. We have to consider three cases: for each pair of children $a \in C(u)$ and $b \in C(v)$ with $\lambda(u,a) = \lambda(v,b)$ we can either attach the subtrees rooted in a and b; we can use solely the edges (u, a) and (v, b) as a common subtree; or, we can attach no common subtree for this pair of children. But if we choose no subtree for all matching pairs of children, the result would be a tree without edges and, hence, not a valid common subtree. Thus, we have to correct for this case by subtracting one. Hence, the recurrence is:

$$D[u, v] = \prod_{\substack{a \in C(u), b \in C(v) \\ \lambda(u,a) = \lambda(v,b)}} (2 + D[a, b]) - 1.$$

2.3 MKL

In many applications, multiple kernels from different kernel functions or multiple sources of information are available. MKL becomes a natural way to combine information contained in the kernels. Instead of choosing the best kernel via cross-validation as in Heinonen *et al.* (2012) and Shen *et al.* (2013), MKL seeks a linear, convex or even non-linear combination of the kernels. An overview of MKL algorithms can be found in a survey by Gönen and Alpaydin (2011).

In practice, it is often difficult for MKL algorithms to outperform the uniform combination of the kernels (`UNIMKL`) where the weights for kernels are equal. However, in some cases, some methods have seen improvements over the uniform combinations. Three algorithms coupled with SVM are considered in the following: centered alignment-based algorithms

(Cortes *et al.*, 2012), quadratic combination of the kernels (Li and Sun, 2010) and ℓ_p-norm $P>1$ for the kernel weights (Kloft *et al.*, 2011).

For all the three algorithms, the input will be a set of kernels $\mathbf{K}=\{\mathbf{K}_k|\mathbf{K}_k\in\mathbb{R}^{n\times n}, k=1,\ldots,q\}$ computed from n data points. The output is a set of m fingerprint properties $\mathbf{Y}\in\{-1,+1\}^{n\times m}$ which is a multi-label prediction task and each label is trained independently in the experiments.

2.3.1 Centered alignment-based MKL The centered alignment-based MKL algorithms are based on the observation that the centered alignment score with the target kernel $\mathbf{K}_Y=\mathbf{y}\mathbf{y}^T$ correlates very well with the performance of the kernel, where $\mathbf{y}$ is a single label. Experiments by Cortes *et al.* (2012) show consistent improvements over the uniform combination. In the molecular fingerprint prediction setting, the target kernel is defined as $\mathbf{K}_Y=\mathbf{Y}\mathbf{Y}^T$.

Two-stage model are considered in which the kernel weights are learned first and then can be applied to all kernel-based learning algorithms (SVM in this work). The centered kernel matrices are defined by Equation (1):

$$\mathbf{K}_c=\left[\mathbf{I}-\frac{\mathbf{e}\mathbf{e}^T}{n}\right]\mathbf{K}\left[\mathbf{I}-\frac{\mathbf{e}\mathbf{e}^T}{n}\right] \tag{1}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{e}$ is the vector with all ones. $\forall\mathbf{A},\mathbf{B}\in\mathbb{R}^{n\times n}$, let $\langle\cdot,\cdot\rangle_F$ denotes the Frobenius product and $||\cdot||_F$ denotes the Frobenius norm which are defined by

$$\langle\mathbf{A},\mathbf{B}\rangle_F=Tr[\mathbf{A}^T\mathbf{b}] \text{ and } ||\mathbf{A}||_F=\sqrt{\langle\mathbf{A},\mathbf{A}\rangle_F}.$$

Let now $\mathbf{K}\in\mathbb{R}^{n\times n}$ and $\mathbf{K}'\in\mathbb{R}^{n\times n}$ be two kernel matrices such that $||\mathbf{K}_c||_F\neq 0$ and $||\mathbf{K}'_c||_F\neq 0$. Then the centered alignment between $\mathbf{K}$ and $\mathbf{K}'$ is defined by

$$\hat{\rho}(\mathbf{K},\mathbf{K}')=\frac{\langle\mathbf{K}_c,\mathbf{K}'_c\rangle_F}{||\mathbf{K}_c||_F||\mathbf{K}'_c||_F}. \tag{2}$$

The simple independent centered alignment-based algorithm (ALIGN) (Cortes *et al.*, 2012) computes the alignment score between each kernel matrix $\mathbf{K}_i$ and the target kernel matrix $\mathbf{K}_Y$ and combine the kernels as

$$\begin{aligned}\mathbf{K}_\mu &\propto \sum_{k=1}^{q}\hat{\rho}(\mathbf{K}_k,\mathbf{K}_Y)\mathbf{K}_k\\ &=\frac{1}{||\mathbf{K}_Y||_F}\sum_{k=1}^{q}\frac{\langle\mathbf{K}_k,\mathbf{K}_Y\rangle_F}{||\mathbf{K}_k||_F}\mathbf{K}_k.\end{aligned}$$

The alignment maximization algorithm (ALIGNF) (Cortes *et al.*, 2012) jointly seeks the weight μ_i to maximize the alignment score defined by Equation (2) between the convex combination of the kernel in $\mathbf{K}$ and the target kernel $\mathbf{K}_Y=\mathbf{y}\mathbf{y}^T$, that is, the following optimization problem:

$$\max_{\mu\in\mathcal{M}}\frac{\langle\mathbf{K}_\mu,\mathbf{K}_Y\rangle_F}{||\mathbf{K}_\mu||_F}$$

where $\mathcal{M}=\mu:||\mu||_2=1,\mu\geq 0$.

2.3.2 Quadratic combination MKL In this setting, the quadratic combination of kernels (QCMKL) is included in the formulation and the MKL problem is solved by semidefinite programming (Lanckriet *et al.*, 2002; Li and Sun, 2010). The kernels in $\mathbf{K}$ are enriched to a new set $\tilde{\mathbf{K}}=\{\tilde{\mathbf{K}}_t|t=1,\ldots,q(q+1)/2\}$ by the following transformation:

$$\tilde{\mathbf{K}}_{t(i,j)}=\begin{cases}\mathbf{K}_i\circ\mathbf{K}_j & i\neq j\\ \mathbf{K}_i & i=j\end{cases}$$

where $i,j=1,\ldots,q$ and $\circ$ denotes the Hadamard product.

The convex combinations of the kernels is given by $\tilde{\mathbf{K}}_\mu=\sum_{t=1}^{q(q+1)/2}\mu_t\tilde{\mathbf{K}}_t$ with $\mu\geq 0$ and $\mathbf{e}^T\mu=1$. Adapting the soft margin SVM formulation reveals the following dual problem (in epigraph form) (Li and Sun, 2010):

$$\begin{aligned}&\max_{\alpha,u}\ u\\ &\text{s.t.}\quad u\geq\alpha^T\mathbf{e}-\frac{1}{2}\alpha^T G(\tilde{\mathbf{K}}_\mu)\alpha,\\ &\qquad 0\leq\alpha\leq C\mathbf{e},\alpha^T\mathbf{y}=0,\\ &\qquad \mu\geq 0,\mathbf{e}^T\mu=1.\end{aligned}$$

The derived Lagrangian for the problem is (Li and Sun, 2010):

$$\begin{aligned}L(\alpha,\beta,\delta,\gamma)=\alpha^T\mathbf{e}-\frac{1}{2}\alpha^T G(\tilde{\mathbf{K}}_\mu)\alpha+\beta^T\alpha\\ +\gamma\alpha^T\mathbf{y}+\delta(C\mathbf{e}-\alpha)\end{aligned}$$

with $\alpha,\beta\geq 0,\delta\geq 0,\gamma$ as dual variables, and $G(\mathbf{K})=\text{diag}(\mathbf{y})\mathbf{K}\text{diag}(\mathbf{y})$. Applying Schur's lemma to convert the first inequality constraint to Linear Matrix Inequality (LMI) unveils the following semidefinite program (SDP) (Li and Sun, 2010):

$$\begin{aligned}&\min_{\alpha,u}\ u\\ &\text{s.t.}\quad\begin{pmatrix}G(\tilde{\mathbf{K}}_\mu) & \mathbf{e}+\beta+\gamma\mathbf{y}-\delta\\ (\mathbf{e}+\beta+\gamma\mathbf{y}-\delta)^T & u-2C\delta^T\mathbf{e}\end{pmatrix}\succcurlyeq 0\\ &\qquad\mu\geq 0,\mathbf{e}^T\mu=1,\beta\geq 0,\delta\geq 0.\end{aligned}$$

Many standard SDP solvers can be used to find the optimal solutions such as cvx (http://cvxr.com/).

2.3.3 ℓ_p-norm MKL While ℓ_1 norm on the kernel weights μ produces sparse solutions, higher norms $p>1$ produces non-sparse solutions which may be beneficial. A general framework for ℓ_p-norm MKL (ℓ_p-MKL) was proposed by Kloft *et al.* (2011). The q kernels correspond to q feature mappings $\Psi_k:\chi\to\mathcal{H}_k$, $k=1,\ldots,q$ and l is some convex loss function and the primal problem is then:

$$\begin{aligned}&\min_{w,b,\mu}\ C\sum_{i=1}^{n}l(\sum_{k=1}^{q}\langle w_k,{}_k(x_i)\rangle_{\mathcal{H}_k}+b,y_i)+\frac{1}{2}\sum_{k=1}^{q}\frac{||w_k||^2_{\mathcal{H}_k}}{\mu_k}\\ &\text{s.t.}\quad\mu\geq 0,\ ||\mu||_{\text{p}}^2\leq 1.\end{aligned}$$

when the optimization is coupled with hinge loss, the problem has a simple dual form (Kloft *et al.*, 2011):

$$\max_{\alpha}\ \alpha^T\mathbf{e}-\frac{1}{2}||(\alpha^T G(\tilde{\mathbf{K}}_i)\alpha)_{k=1}^{q}||_{p^*},$$

where all the variables are all as defined before but $p^*=\frac{p}{p-1}$.

The optimization problem can be solved by alternating the dual variables α and the kernel weights μ via the squared norm on w by the following equations:

$$||w_k||^2=\mu_k^2\alpha^T\mathbf{K}_k\alpha,\forall k=1,\ldots,q. \tag{3}$$

$$\mu_k=\frac{||w_k||^2}{(\sum_{k'=1}^{q}||w_{k'}||^{\frac{2p}{p+1}})^{\frac{1}{p}}},\forall k=1,\ldots,q. \tag{4}$$

Based on the above equations, a simple alternating algorithm has been proposed by Kloft *et al.* (2011) as Algorithm 1.

Algorithm 1 Wrapper algorithm for ℓ_p-norm MKL

Input feasible α and μ
while optimization conditions are not satisfied **do**
  Solve α with current μ using standard SVM.
  Compute $||w_k||^2$ with equation (3).
  Update μ by equation (4).
end while

The optimization conditions can be the difference of objective function or the duality gap between two subsequent iterations. More detailed, theoretical results and a faster chunking-based algorithm are also presented in Kloft *et al.* (2011).

2.4 Probabilistic scoring of candidate metabolites

Given a predicted fingerprint associated with a mass spectrum, for metabolite identification, we need to retrieve metabolites with similar fingerprints from a molecular database. Assume $\hat{\mathbf{y}} \in \{-1, +1\}^m$ is a predicted fingerprint and an arbitrary fingerprint $\mathbf{y} \in \{-1, +1\}^m$ for some molecule in some molecular database, one can score the $\mathbf{y}$ by the following equation as used in *FingerID* (Heinonen *et al.*, 2012; Shen *et al.*, 2013):

$$P_{PB}(\mathbf{y}|\gamma, \hat{\mathbf{y}}) = \prod_{j=1}^{m} \gamma_j^{1_{y_j=\hat{y}_j}} (1-\gamma_j)^{1_{y_j=\hat{y}_j}}$$

that is, the Poisson binomial probability for the fingerprint vector $\mathbf{y}$ where the cross-validation accuracies $(\gamma_j)_{j=1}^m \in [0.5, 1]^m$ of the fingerprints prediction are taken as the reliability scores.

3 RESULTS

Two MS/MS datasets, 978 compounds downloaded from METLIN (Tautenhahn *et al.*, 2012) and 402 compounds from MassBank (Hisayuki *et al.*, 2010), both measured by QTOF MS/MS instruments are tested. For each compound, mass spectra recorded at different collision energies were amalgamated before further processing: we normalize MS/MS spectra such that intensities sum up to 100%. We merge peaks from different collision energies with m/z difference at most 0.1, using the m/z of the highest peak and summing up intensities. We discard all but the 30 highest peaks, as well as peaks with relative intensity <0.5%.

Next, we compute the fragmentation tree. We assume that we can identify the correct molecular formula from the data: limiting candidate molecular formulas to those present in KEGG (Kanehisa and Goto, 2000), which is used for searching molecular structures below, the best scoring fragmentation tree identified the correct molecular formula of the compound in 97.1% (96.0%) of the cases for the METLIN (MassBank) dataset. Integrating other sources of information such as MS1 isotope patterns (Böcker *et al.*, 2009) or retention times would reach even better identification rates. To allow for a meaningful comparison of the power of the different kernels, we therefore use the best scoring fragmentation tree of the correct compound molecular formula.

All 11 fragmentation tree kernels proposed in the previous section were computed, along with PPK used in Heinonen *et al.* (2012) and Shen *et al.* (2013) computed directly from MS/MS, resulting in 12 kernels to be evaluated.

Molecular fingerprints were generated using OpenBabel (O'Boyle *et al.*, 2011) which contains four types of fingerprints (http://openbabel.org/wiki/Tutorial:Fingerprints). FP3, FP4 and MACCS fingerprints (528 bits in total) were generated based on the software predefined SMARTS patterns. In our dataset, more than half of the fingerprint properties have high-class bias rate, with a large majority of the dataset belonging to the positive class (most compounds match the property) or respectively the negative class (most compounds do not match the property). For such fingerprints, the default classifier, one that always predicts the majority class, has high accuracy, although the model is not meaningful. For our performance comparisons, we opted to only include fingerprints with class bias rate <0.9.

For each fingerprint property, we separately trained a SVM; for all properties, we used identical training and testing compounds. Five-fold cross-validation was performed and the SVM margin softness parameter ($C \in \{2^{-3}, 2^{-2}, \ldots, 2^6, 2^7\}$) was tuned based on the training accuracy.

Table 1. Micro-average performance of individual kernels

	METLIN		MassBank	
	Acc (%)	F1 (%)	Acc (%)	F1 (%)
LB	79.5 ± 0.5	69.9 ± 0.9	78.9 ± 1.0	69.0 ± 2.2
LC	79.4 ± 0.3	69.6 ± 0.4	78.5 ± 1.2	68.4 ± 2.7
LI	77.8 ± 0.5	66.8 ± 0.7	77.4 ± 1.0	66.7 ± 2.0
RLB	81.6 ± 0.8	73.2 ± 1.1	78.6 ± 1.0	68.4 ± 1.2
RLI	78.4 ± 0.6	68.5 ± 0.8	76.7 ± 0.9	65.4 ± 1.6
NB	**81.9 ± 0.4**	**73.9 ± 0.3**	**81.4 ± 0.7**	**73.2 ± 1.2**
NI	80.3 ± 0.7	71.1 ± 0.8	79.8 ± 1.0	70.5 ± 0.9
CPC	80.6 ± 0.5	71.6 ± 0.7	78.7 ± 1.4	68.9 ± 2.4
CP2	78.7 ± 0.7	68.4 ± 1.2	76.4 ± 1.0	65.5 ± 1.1
CPK	72.9 ± 0.3	58.8 ± 0.5	72.2 ± 0.6	57.9 ± 0.5
CSC	74.9 ± 0.4	61.9 ± 0.8	77.8 ± 0.8	67.2 ± 2.0
PPK	76.7 ± 0.6	64.0 ± 0.7	72.9 ± 1.1	58.6 ± 1.2

PPK is the method from Heinonen *et al.* (2012), which we compare against.

3.1 Fingerprint prediction performance

The micro-average (simultaneous average over fingerprint properties and compounds) accuracy and F1 of the individual kernels on the predictions of fingerprint properties with bias rate <0.9 are shown in Table 1 with the SDs computed from different cross-validation folds. The kernel NB achieves the best accuracy and F1 on both METLIN and MassBank. Compared with the PPK, the fragmentation tree kernels are markedly more accurate on average.

The improvement of MKL approaches over single kernel SVMs are clear. The t-test between NB and ALIGNF shows the differences of mean accuracy and F1 are indeed very significant with P-values of 4×10^{-6} and 1.7×10^{-3}, respectively. The kernel weights learned by different MKL algorithms are shown in the supplementary file.

The micro-average accuracy and F1 of the MKL algorithms on the fingerprint properties predictions are shown in Table 2, where it can be concluded that averaged overall fingerprints of the MKL methods are quite close. We conducted further pairwise difference testing, where the performance difference of each method on each individual fingerprint property is evaluated. Table 3 shows the significance level of the sign test on the

accuracy and F1 on the METLIN and MASSBANK datasets using the different MKL methods. The sign test describes whether one of the methods has higher probability of success (better than the other on a fingerprint) than the other (alternative hypothesis) or not (null hypothesis). From the table, we can deduce that ALIGN and ALIGNF rise slightly above the competition whereas ℓ_2-MKL and QCMKL are slightly inferior to the rest. The performance of UNIMKL is also respectable. The scatter plots of accuracy and F1 between every pair of the MKL algorithms are shown in the supplementary file.

Table 2. Micro-average performance of MKL algorithms

	METLIN		MassBank	
	Acc (%)	F1 (%)	Acc (%)	F1 (%)
UNIMKL	85.0 ± 0.6	78.3 ± 0.7	82.2 ± 0.6	73.9 ± 1.5
ALIGN	**85.2 ± 0.6**	78.6 ± 0.7	82.4 ± 0.7	74.4 ± 1.4
ALIGNF	85.0 ± 0.5	**78.6 ± 0.4**	**82.8 ± 0.4**	**75.2 ± 1.2**
QCMKL	84.9 ± 0.5	77.8 ± 0.5	82.1 ± 0.6	74.0 ± 0.7
ℓ_2-MKL	84.7 ± 0.5	77.5 ± 0.5	82.2 ± 0.5	74.0 ± 0.9
ℓ_3-MKL	**85.2 ± 0.6**	78.5 ± 0.7	82.4 ± 0.6	74.4 ± 1.3
ℓ_4-MKL	**85.2 ± 0.6**	78.5 ± 0.8	82.3 ± 0.6	74.2 ± 1.0
ℓ_5-MKL	85.1 ± 0.6	78.5 ± 0.7	82.3 ± 0.6	74.1 ± 1.3

3.2 Metabolite identification performance

The molecular fingerprint prediction can serve as an intermediate step for metabolites identification, and can be used to search a molecular structure database (Heinonen *et al.*, 2012; Shen *et al.*, 2013). We want to evaluate whether improvements in fingerprint prediction propagate to better metabolites identifications. We will search for molecular structures from the KEGG database. As we assume to know the correct molecular formula, we may filter based on this information to generate our candidate lists. But it turns out that this filter is too strict for a meaningful evaluation, as the number of candidates for each MS/MS spectrum becomes very small and, hence, *all* kernels show good performance. For a

Table 3. Sign test for the performance of MKL algorithms on the METLIN and MassBank datasets

	Acc	UNIMKL	ALIGN	ALIGNF	QCMKL	ℓ_2-MKL	ℓ_3-MKL	ℓ_4-MKL	ℓ_5-MKL
METLIN	UNIMKL		−−	−	++	++		−−	−−
	ALIGN	++			++	++	+		++
	ALIGNF	+			++	++			+
	QCMKL	−−	−−	−−		++	−−	−−	−−
	ℓ_2-MKL	−−	−−	−−	−−		−−	−−	−−
	ℓ_3-MKL		−		++	++			
	ℓ_4-MKL	++			++	++			
	ℓ_5-MKL	++	−−	−	++	++			
MassBank	UNIMKL		−	−−	+	+			+
	ALIGN	+		−−	+	++		++	++
	ALIGNF	++	++		++	++	++	++	++
	QCMKL	−	−	−−			−	−−	−
	ℓ_2-MKL	−	−−	−−				−	−
	ℓ_3-MKL			−−	+			+	
	ℓ_4-MKL		−−	−−	++	+	−		
	ℓ_5-MKL	−	−−	−−	+	+			
	F1	UNIMKL	ALIGN	ALIGNF	QCMKL	ℓ_2-MKL	ℓ_3-MKL	ℓ_4-MKL	ℓ_5-MKL
METLIN	UNIMKL		−			+	−−	−−	−−
	ALIGN	+			++	++			
	ALIGNF				+	++			
	QCMKL		−−	−		+	−−	−−	
	ℓ_2-MKL	−	−−	−−	−		−−	−−	−−
	ℓ_3-MKL	++			++	++			
	ℓ_4-MKL	++			++	++			
	ℓ_5-MKL	++				++			
MassBank	UNIMKL		−						+
	ALIGN	+			++	++			
	ALIGNF				++	++			
	QCMKL		−−	−−			−−	−−	−
	ℓ_2-MKL		−−	−−					
	ℓ_3-MKL				++				
	ℓ_4-MKL				++				
	ℓ_5-MKL	−			+				

'+' indicates the method in the row is better than the method in the column ('−' otherwise) with significance *P*-value between 0.01 and 0.05; blank indicates no significance. Similarly, '++' and '−−' indicate significance with *P*-value < 0.01. Upper table is for accuracy and lower table is for F1.

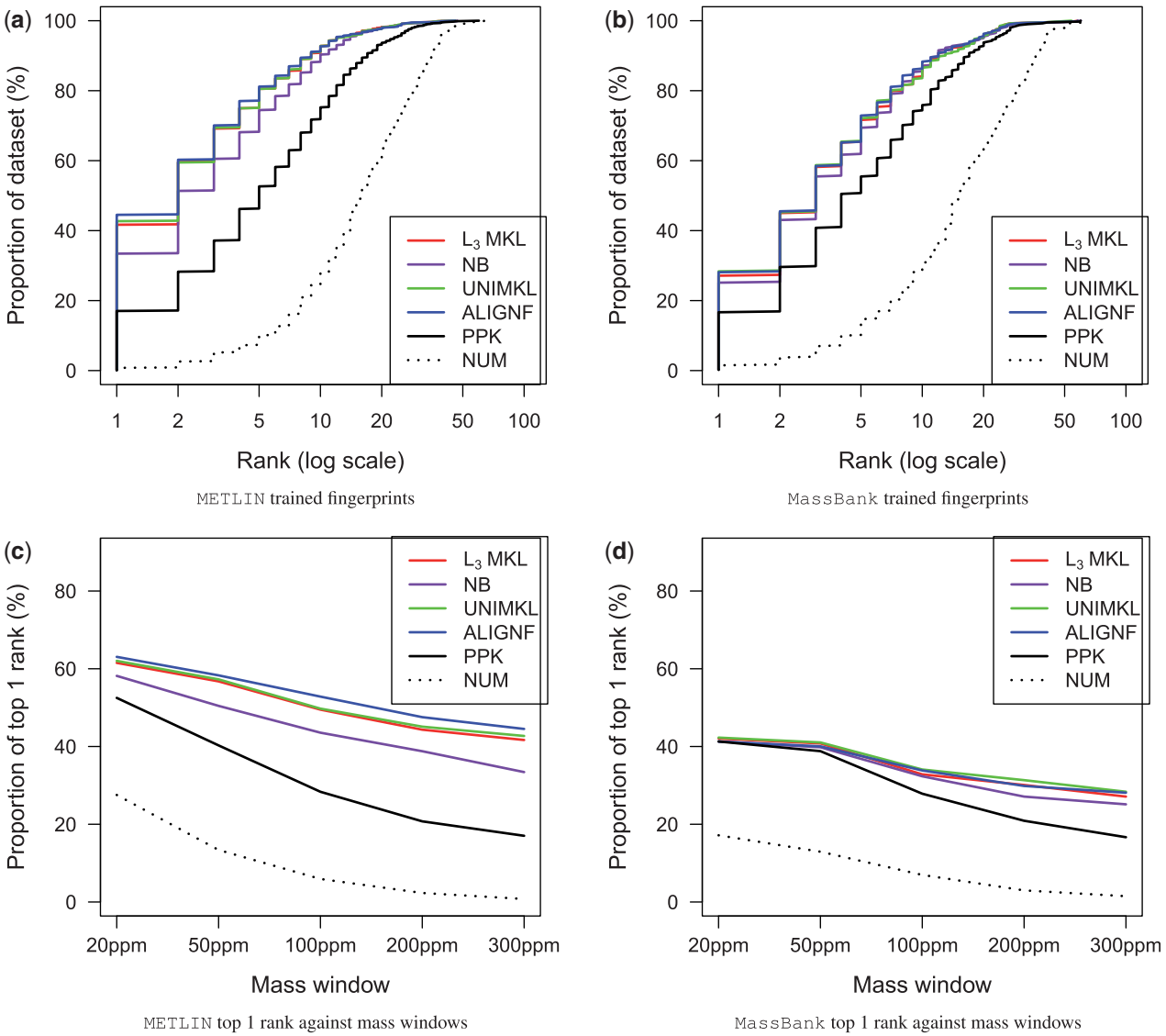

Fig. 2. (**a** and **b**) show the performance for identification when searching KEGG using 300-ppm mass window with predicted molecular fingerprints, with fingerprints trained with METLIN and MassBank datasets, respectively. NUM denotes the number of candidate molecules returned per query. (**c** and **d**) show the proportion of data that were correctly identified in the top 1 rank against a series of mass windows

more discriminative evaluation of the kernels, we artificially enlarge the set of candidates: we use all molecular structures in KEGG with mass accuracy window $[\mu_M - \Delta, \mu_M + \Delta]$ as candidates, where μ_M is the true mass of the unknown molecule. For sufficiently large mass accuracy Δ, this results in candidate lists that allow a meaningful comparison of the kernels.

For identification, we want the true molecular structure to be ranked as high as possible in the candidates list. Figure 2a and b shows the fraction of compounds that were ranked higher than certain rank for the two datasets, when searching KEGG with 300 ppm mass inaccuracy to generate the candidates for the two datasets.

We notice that the NB kernel is consistently more accurate than PPK. In addition, MKL clearly improves the identification performance, especially the number of top-ranked identifications increases significantly. *T*-test between the ranks of the ALIGNF and PPK shows a *P*-value of 0.06 which verifies the improvements in identification by ALIGNF over the PPK is indeed significant. ALIGNF comes on top of the MKL approaches, which is in line with its good fingerprint prediction accuracy and F1 score.

The effect of mass accuracy windows during the database retrieval are shown in Figure 2c and d. A narrower 20-ppm mass search window filters out many false candidates, and thus significantly elevates the identification accuracies to 60% on METLIN dataset and 40% on MassBank dataset. However, the effect of improved molecular fingerprint prediction is softened due to the fewer but possibly more similar candidates. An extreme case is observed in Figure 2d in which all the methods shrink to the same result when searching with 20-ppm mass accuracy window.

4 DISCUSSION

The present work combines the combinatorial fragmentation tree approach with machine learning through a kernel-based approach. We suggest several kernels for fragmentation trees, and show how to fuse their information through MKL. The result significantly enhances molecular fingerprint prediction and metabolite identification.

The closest analogs to our fragmentation tree kernels in literature are those defined for parse trees in natural language processing (Collins and Duffy, 2001); our fragmentation trees can be seen as parses of the MS/MS spectra. DP techniques similar to ours are used there for computing kernels between trees (Collins and Duffy, 2001; Kuboyama, 2007). However, fragmentation trees have important differences to the trees defined between parses of natural language and to kernels comparing molecular structures (Mahé and Vert, 2009). Differently from natural language parses, the node labels have partial order (via their molecular weights) and also the edges have labels. Differently from kernels for molecular graphs, the label spaces of both nodes and edges are vast (subsets of molecular formulae).

The comparison with the PPK employed by the FingerID (Heinonen *et al.*, 2012) software shows that the fragmentation tree kernels are able to extract more information out of the MS/MS spectra. Improvements are seen in both the prediction accuracy and the F1 score. Comparing with FingerID (PPK), the uniform combination of the kernels (UNIMKL) improves the molecular fingerprint prediction significantly in accuracy and F1. As witnessed by many MKL applications, the `UNIMKL` algorithm is hard to beat. In our result, several MKL algorithms such as `ALIGNF` and ℓ_3-`norm` can give slightly better result than `UNIMKL`. The improvements in the molecular fingerprint prediction translate to improved metabolite identification.

There are several possible routes forward with the current metabolite identification framework. First, post-processing on the candidates list, such as the one proposed by Allen *et al.* (2013), is necessary when searching a large compound database such as PubChem, because the returned candidates (hundreds to thousands) may share the same fingerprints and there is no way to differ them based only on molecular fingerprints. Second, training a separate SVM for each fingerprint property is clearly an aspect that can be improved upon, for example, by a multi-label classification approach. A still more tempting yet challenging direction would be to replace the two-step identification by an integrated prediction approach. Such an approach would potentially learn to predict the fingerprint properties that are important for discriminating metabolites from each other.

Funding: Academy of Finland grant 268874 (MIDAS); Deutsche Forschungsgemeinschaft grant (BO 1910/16-1) (IDUN).

Conflict of Interest: none declared.

REFERENCES

Allen,F. *et al.* (2013) Competitive fragmentation modeling of ESI-MS/MS spectra for metabolite identification. *Pre-print*. arXiv:1312.0264.

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

Böcker,S. *et al.* (2009) Sirius: decomposing isotope patterns for metabolite identification. *Bioinformatics*, **25**, 218–224.

Collins,M. and Duffy,N. (2001) Convolution kernels for natural language. In Dietterich,T.G. *et al.* (ed.) *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA, pp. 625–632.

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

Demuth,W. *et al.* (2004) Spectral similarity versus structural similarity: mass spectrometry. *Anal. Chim. Acta*, **516**, 75–85.

Gerlich,M. and Neumann,S. (2013) MetFusion: integration of compound identification strategies. *J. Mass Spectrom.*, **48**, 291–298.

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.

Hill,D.W. *et al.* (2008) Mass spectral metabonomics beyond elemental formula: chemical database querying by matching experimental with computational fragmentation spectra. *Anal. Chem.*, **80**, 5574–5582.

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

Hufsky,F. *et al.* (2014) Computational mass spectrometry for small molecule fragmentation. *Trends Anal. Chem.*, **53**, 41–48.

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

Kanehisa,M. and Goto,S. (2000) KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, **28**, 27–30.

Kangas,L.J. *et al.* (2012) In silico identification software (ISIS): a machine learning approach to tandem mass spectral identification of lipids. *Bioinformatics*, **28**, 1705–1713.

Kind,T. and Fiehn,O. (2007) Seven golden rules for heuristic filtering of molecular formulas obtained by accurate mass spectrometry. *BMC Bioinformatics*, **8**, 105.

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

Kuboyama,T. (2007) Matching and learning in trees. PhD Thesis, University of Tokyo.

Lanckriet,G. *et al.* (2002) Learning the kernel matrix with semi-definite programming. *J. Mach. Learn. Res.*, **5**, 2004.

Li,J. and Sun,S. (2010) Nonlinear combination of multiple kernels for support vector machines. In *International Conference on Pattern Recognition, Istanbul*. IEEE, pp. 2889–2892.

Mahé,P. and Vert,J.-P. (2009) Graph kernels based on tree patterns for molecules. *Mach. Learn.*, **75**, 3–35.

Oberacher,H. *et al.* (2009) On the inter-instrument and the inter-laboratory transferability of a tandem mass spectral reference library: 2. optimization and characterization of the search algorithm. *J. Mass Spectrom.*, **44**, 494–502.

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

Pitkänen,E. *et al.* (2010) Computational methods for metabolic reconstruction. *Curr. Opin. Biotechnol.*, **21**, 70–77.

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

Rasche,F. *et al.* (2012) Identifying the unknowns by aligning fragmentation trees. *Anal. Chem.*, **84**, 3417–3426.

Rauf,I. *et al.* (2012) Finding maximum colorful subtrees in practice. In Benny,C. (ed.) *Research in Computational Molecular Biology*. Volume 7262 of *Lecture Notes in Computer Science*. Springer, Berlin Heidelberg, pp. 213–223.

Rojas-Chertó,M. *et al.* (2012) Metabolite identification using automated comparison of high-resolution multistage mass spectral trees. *Anal. Chem.*, **84**, 5524–5534.

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

Shen,H. *et al.* (2013) Metabolite identification through machine learning—tackling casmi challenge using FingerID. *Metabolites*, **3**, 484–505.

Smith,C.A. *et al.* (2005) Metlin: a metabolite mass spectral database. *Drug Monit.*, **27**, 747–751.

Tautenhahn,R. *et al.* (2012) An accelerated workflow for untargeted metabolomics using the METLIN database. *Nat. Biotechnol.*, **30**, 826–828.

Wolf,S. *et al.* (2010) In silico fragmentation for computer assisted identification of metabolite mass spectra. *BMC Bioinformatics*, **11**, 148.