BMC Systems Biology

RESEARCH Open Access

Scalable prediction of compound-protein interactions using minwise hashing

Yasuo Tabei[1*], Yoshihiro Yamanishi[2,3]

From 24th International Conference on Genome Informatics (GIW 2013)
Singapore, Singapore. 16-18 December 2013

Abstract

The identification of compound-protein interactions plays key roles in the drug development toward discovery of new drug leads and new therapeutic protein targets. There is therefore a strong incentive to develop new efficient methods for predicting compound-protein interactions on a genome-wide scale. In this paper we develop a novel chemogenomic method to make a scalable prediction of compound-protein interactions from heterogeneous biological data using minwise hashing. The proposed method mainly consists of two steps: 1) construction of new compact fingerprints for compound-protein pairs by an improved minwise hashing algorithm, and 2) application of a sparsity-induced classifier to the compact fingerprints. We test the proposed method on its ability to make a large-scale prediction of compound-protein interactions from compound substructure fingerprints and protein domain fingerprints, and show superior performance of the proposed method compared with the previous chemogenomic methods in terms of prediction accuracy, computational efficiency, and interpretability of the predictive model. All the previously developed methods are not computationally feasible for the full dataset consisting of about 200 millions of compound-protein pairs. The proposed method is expected to be useful for virtual screening of a huge number of compounds against many protein targets.

Background

The identification of compound-protein interactions is an important part in the drug development toward discovery of new drug leads and new therapeutic protein targets. The completion of the human genome sequencing project has made it possible for us to analyze the genomic space of possible proteins coded in the human genome. At the same time, many efforts have also been devoted to the constitution of molecular databanks to explore the entire chemical space of possible compounds including synthesized molecules or natural molecules extracted from animals, plants, or microorganisms. However, there is little knowledge about the interactions between compounds and proteins. For example, the US PubChem database stores more than 30 million chemical compounds, but the number of compounds with information on their target proteins is very limited [1]. In that field, the importance of chemogenomics research has recently grown fast to investigate the relationship between the chemical space and the genomic space [2,3]. A key issue in chemogenomics is computational prediction of compound-protein interactions on a genome-wide scale.

Recently, a variety of *in silico* chemogenomic approaches have been developed to predict compound-protein interactions or drug-target interactions, assuming that similar compounds are likely to interact with similar proteins. The state-of-the-art in the chemogenomic approach is to built the chemogenomic space of compound-protein pairs as the tensor product of the chemical space of compounds and the genomic space of proteins, and analyze compound-protein pairs by machine learning classifiers such as support vector machine (SVM) [4-8]. However, the input of the SVM method in most previous works is the pairwise kernel similarity matrix of compound-protein *pairs*, which makes it difficult to analyze large-scale data. For example, it is impossible to apply standard implementations such as LIBSVM [9] and SVM^{light}[10], because it requires prohibitive computational time and the size of

* Correspondence: yasuo.tabei@gmail.com
[1]PRESTO, Japan Science and Technology Agency, Kawaguchi, Saitama 332-0012, Japan
Full list of author information is available at the end of the article

the kernel matrix for compound-protein pairs is too huge to construct explicitly in the memory. All previous chemogenomic methods are not suitable for scalable screening of millions of or billions of compound-protein pairs.

Fingerprint is a powerful way to efficiently summarize information about various bio-molecules (e.g., compounds, proteins), that is, encoding their molecular structures or physicochemical properties into finite-dimensional binary vectors. The fingerprint representation has a long history in chemoinformatics, and many 1D, 2D or 3D descriptors for molecules have been proposed [11] and adopted in many molecular databases such as PubChem [1] and ChemDB [12]. The fingerprints can be used for exploring the chemical space based on their Euclidian distance or Tanimoto coefficients, and can also be used as inputs of various machine learning classifiers to predict various biological activities of compounds [13]. The fingerprint representation is applicable to proteins as well [14,15].

In this study we consider representing compound-protein pairs by the fingerprints to use them as inputs of linear SVM, because the linear SVM provides us with interpretable predictive models and works well for super-high dimensional data [16]. A straightforward way is to represent each compound-protein pair by taking the tensor product of the compound fingerprint and the protein fingerprint, which enables biological interpretation of chemogenomic features (functional associations between compound substructures and protein domains) behind interacting compound-protein pairs [8]. However, the resulting fingerprint is sparse and super-high dimensional. Even worse, the total number of fingerprints is the product of the number of compounds and the number of proteins, so it is difficult to train classical linear SVM for extremely large-scale data. Although optimization techniques of linear SVM have recently advanced [17-20], they are not enough to analyze a huge number of compound-protein pairs in practice.

In this paper we develop a novel chemogenomic method to make a scalable prediction of compound-protein interactions from heterogeneous biological data using minwise hashing, which is applicable for virtual screening of a huge number of compounds against many human proteins. The proposed method mainly consists of two steps: 1) construction of new compact fingerprints for compound-protein pairs by an improved minwise hashing algorithm, and 2) application of the linear SVM to the compact fingerprints. A unique feature of the proposed method is that the linear SVM with the compact fingerprints generated by the minwise hashing is able to simulate the nonlinear property of the kernel SVM. We test the proposed method on its ability to make a large-scale prediction of compound-protein interactions from compound substructure fingerprints and protein domain fingerprints, and show superior performance of the proposed method compared with the previous chemogenomic methods in terms of prediction accuracy, computational efficiency, and interpretability of the predictive model. All the previously developed methods are not computationally feasible for the full dataset consisting of about 200 millions of compound-protein pairs.

Materials

Compound-protein interactions involving human proteins were obtained from the STITCH database [21]. Compounds are small molecules and proteins belong to many different classes such as enzymes, transporters, ion channels, and receptors. The dataset consists of 300,202 known compound-protein interactions out of 216,121,626 possible compound-protein pairs, involving 35,366 compounds and 6,111 proteins. Note that duplicated compounds were removed. The set of known interactions is used as gold standard data.

Chemical structures of compounds were encoded by a chemical fingerprint with 881 chemical substructures defined in the PubChem database [1]. Each compound was represented by a substructure fingerprint (binary vector) whose elements encode for the presence or absence of each of the 881 PubChem substructures by 1 or 0, respectively.

Genomic information about proteins was obtained from the UniProt database [22], and the associated protein domains were obtained from the PFAM database [23]. Proteins in our dataset were associated with 4,137 PFAM domains. Each protein was represented by a domain fingerprint (binary vector) whose elements encode for the presence or absence of each of the retained 4,137 PFAM domains by 1 or 0, respectively.

Methods

We deal with the in-silico chemogenomics problem as the following machine learning problem: given a set of n compound-protein pairs $(C_1, P_1),..., (C_n, P_n)$, then estimate a function $f(C, P)$ that would predict whether a compound C binds to a protein P . In addition, we attempt to estimate an interpretable function f in order to extract informative features. Since our dataset consists of about 216 millions of compound-protein pairs, we propose an efficient and general approach to solve these problems.

Model

Linear models are a feasible tool for large-scale classification and regression tasks such as linear support vector machines (linear SVM) and logistic regression which provide comprehensible models for these tasks. Generally, linear models represent each example E as a feature vector $\Phi(E) \in \Re^D$ and then estimate a linear function $f(E) = \boldsymbol{w}^{\mathrm{T}}\Phi(E)$ whose sign is used to predict whether or not the example E is positive or negative. Note that fingerprints are used for feature vectors in this study. The weight vector $\boldsymbol{w} \in \Re^D$ is estimated based on its ability to correctly predict the classes of examples in the training set. Since each element of the weight vector $\boldsymbol{w}$ corresponds to an element of the fingerprint $\Phi(E)$, we can interpret salient features by sorting elements of $\Phi(E)$ according to the values of the corresponding elements of $\boldsymbol{w}$.

In this study each compound-protein pair corresponds to an example. Thus, it is necessary to represent each compound-protein pair (C, P) as a single fingerprint $\Phi(C, P)$ and then estimate a function $f(C, P) = \boldsymbol{w}^T\Phi(C, P)$ whose sign is used to predict whether a compound C interacts with a protein P or not. As in the previous case, we can extract effective features in $\Phi(C, P)$ for compound-protein interaction predictions.

Fingerprint representation of compound-protein pairs

A fingerprint representation of compound-protein pairs has a large impact on not only classification ability of linear models but also interpretability of features. To meet both demands, we represent each compound-protein pair by a fingerprint using the compound fingerprint and the protein fingerprint.

The fingerprint of a compound C is represented by a D-dimensional binary vector: $\Phi(C) = (c_1, c_2, ..., c_D)^T$ where $c_i \in \{0, 1\}$, $i = 1, ..., D$. The fingerprint of a protein P is represented by a D'-dimensional binary vector as well: $\Phi(P) = (p_1, p_2, ..., p_{D'})$T where $p_i \in \{0, 1\}$, $i = 1, ..., D'$. We define the fingerprint of each compound-protein pair as the tensor product of $\Phi(C)$ and $\Phi(P)$ as follows:

$$\begin{aligned}\Phi(C, P) &= \Phi(C) \otimes \Phi(P) \\ &= (c_1p_1, \ldots, c_1p_{D'}, \ldots, c_Dp_1, \ldots, c_Dp_{D'})^T.\end{aligned}$$

$\Phi(C, P)$ consists of all possible products of elements in two fingerprints $\Phi(C)$ and $\Phi(P)$, so the fingerprint is a $D \times D'$ dimensional binary vector. The dimensions of $\Phi(C)$, $\Phi(P)$, and $\Phi(C, P)$ in this study are $D = 881$, $D' = 4, 137$, and $DD' = 3, 644, 697$, respectively.

Minwise hashing

We propose to use *minwise hashing* for analyzing fingerprints efficiently. In this section, we make a brief review of minwise hashing [24]. A key observation is that any fingerprint can be represented by a set uniquely. Each fingerprint $\Phi(C, P)$ is represented by a set $S \subseteq \Omega = \{1, 2, ..., D \times D'\}$. Given two sets S_i and S_j, Jaccard similarity $J(S_i, S_j)$ of Si and S_j is defined as

$$J(S_i, S_j) = \frac{|S_i \cap S_j|}{|S_i \cup S_j|} \quad (i, j = 1, 2, \cdots, n).$$

Minwise hashing is a random projection of sets such that the expected Hamming distance of obtained symbol strings is proportional to the Jaccard similarity [24]. We pick ℓ random permutations π_k, $k = 1, ..., \ell$, each of which maps $[1, M]$ to $[1, M]$. Let $T_i = t_{i1}, ..., t_{i\ell}$ be a resultant string projected from S_i. The projection is defined as the minimum element of the random permutation of the given set,

$$t_{ik} = \min\{\pi_k(S_i)\}$$

For example, if π_k is defined as

$$(1, 2, 3, 4, 5, 6, 7, 8) \rightarrow (3, 8, 7, 1, 2, 6, 4, 5),$$

$S_i = (1, 4, 6, 7)$ is transformed to $\pi_k(S_i) = (3, 1, 6, 4)$, and the final product is $t_{ik} = 1$. The collision probability, which is a probability that two sets S_i and S_j are projected to the same elements t_{ik} and $t_{jk}(t_{ik} = t_{jk})$, is described as

$$\Pr[t_{ik} = t_{jk}] = J(S_i, S_j).$$

Therefore, the expected Hamming distance between t_i and t_j is identical to $\ell(1 - J(S_i, S_j))$.

Saving memory by additional hashing

The common practice of minwise hashing is to store each hashed value using 64bits [24]. The storage (and computational) cost is prohibitive in large-scale applications. To overcome this problem, Li et al. proposed b-bit minwise hashing [25,26], which rounds each hashing value to only lower b-bits value. However, a theoretical analysis of the collision probability is complicated.

Here we introduce a simple yet effective method such that a theoretical estimation of collision probability can easily be derived. In our method, the hashing values are further hashed to a set $\{1, ..., N\}$ randomly, where $N << M$. This projection is defined as follows:

$$s_{ik} = h(t_{ik}),$$

where $h : \{1, ..., M\} \rightarrow \{1, ..., N\}$ is a random hash function. If t_{ik} and t_{jk} are identical, s_{ik} and s_{jk} always collide. If not, they collide with probability $1/N$. Thus, the collision probability is obtained as follows:

$$Pr[s_{ik} = s_{jk}] = 1 - \frac{N-1}{N}(1 - J(S_i, S_j)). \quad (1)$$

Figure 1 shows collision probability for each hashing value, where four different Jaccard similarities, 0.1, 0.3, 0.5 and 0.7, are chosen. It is observed that collision probabilities do not increase for hashing values of no less than 2^8. Thus, small hashing values can be chosen without loss of accuracy.

Building compact fingerprints by minwise hashing

Learning linear models with large-scale high-dimensional data is a difficult problem in terms of computational cost. Here we propose a method to represent the

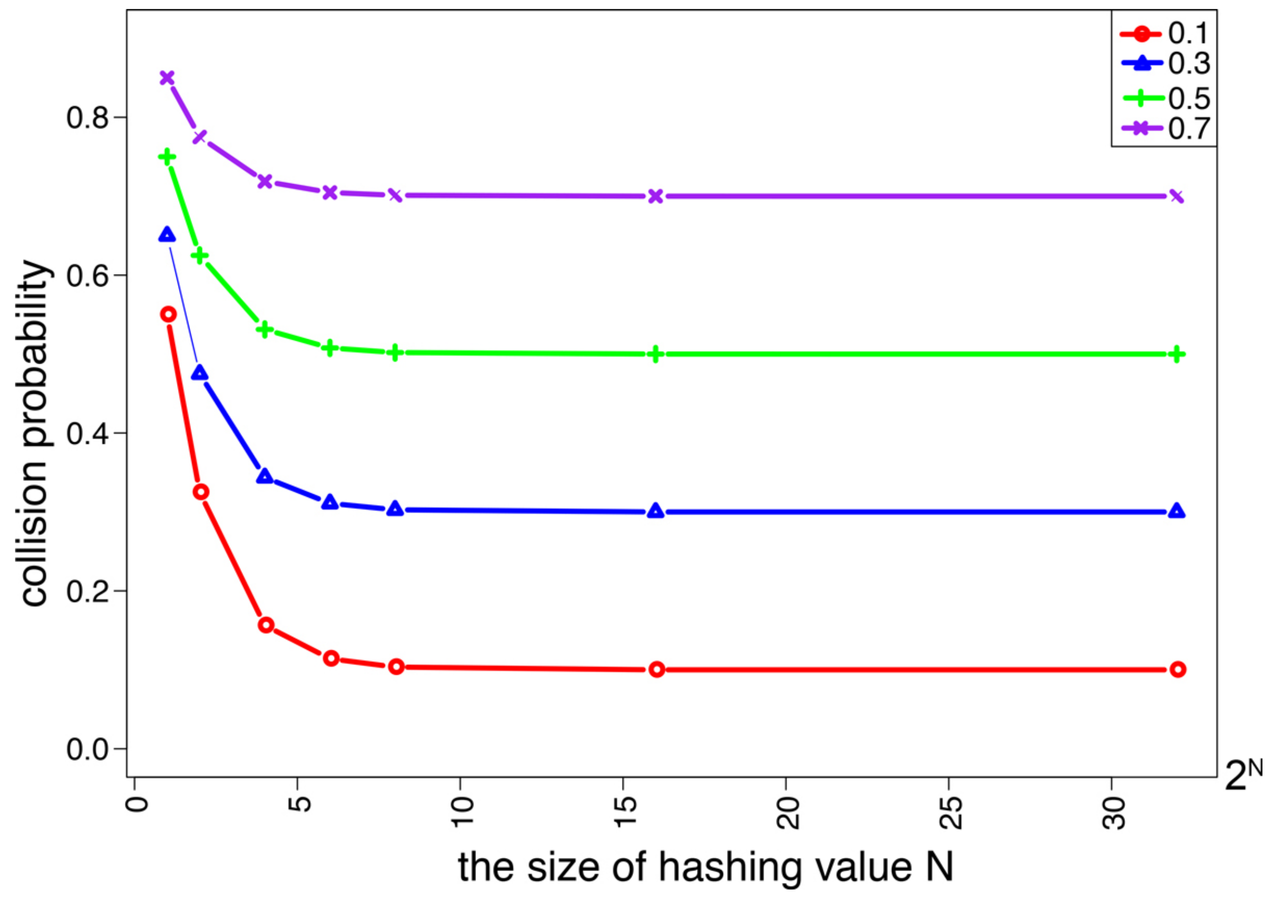


Figure 1 Collision probabilities for varying the size of additional hashing value *N*.

original fingerprint of compound-protein pair by a new fingerprint whose size is smaller than that of the original fingerprint.

A crucial observation is that any fingerprint can be represented as a set uniquely, and can also be converted into a string uniquely. First, we convert the original fingerprint of each compound-protein pair into a string by applying minwise hashing and additional hashing. Next, we expand hashing values organizing the string into a new binary vector whose dimension is much smaller than that of the original fingerprint.

Let $\boldsymbol{S}(C, P)$ be a set representation of $\Phi(C, P)$ where i is contained in $\boldsymbol{S}(C, P)$ iff the i-th element of $\Phi(C, P)$ is 1. We apply minwise hashing $\pi_k(k = 1, ..., \ell)$ to $\boldsymbol{S}(C, P)$ to generate a string $\boldsymbol{T}(C, P) = t_1, t_2, ..., t_\ell$, where each element t_k takes a value ranging from 1 to M. We additionally hash each element t_k to a new small value t'_k ranging from 1 to $N(N << M)$ by applying additional hash h, and generate a new string $T'(C, P) = t'_1, t'_2, \ldots, t'_\ell$. Each value t'_k in the string $\boldsymbol{T}'(C, P)$ is expanded to an N-dimensional binary vector f_k, where the t'_k-th element is 1 and the others are 0. Finally, we concatenate $f_1, ..., f_\ell$ into a single one, and obtain an ℓN-dimension binary vector $\boldsymbol{F}(C, P) = (f_1, ..., f_\ell)$. The newly obtained $\boldsymbol{F}(C, P)$ is referred to as "compact fingerprint". Figure 2 shows an illustration of the proposed procedure.

Linear support vector machines (Linear SVM)

We use linear SVM as a classifier. The predictive model is typically learned by minimizing objective functions with a regularization. The most common regularization is L_2-regularization which keeps most elements in the weight vector to be non-zeros, so one suffers from difficulty in interpreting the predictive model with many non-zero weights. L_2-regularized linear SVM is referred to as L2SVM. Another regularization is L_1-regularization which keeps most elements in the weight vector to be zeros, so the L_1-regularization is popularly used for its high interpretability owing to the induced sparsity. L_1-regularized linear SVM is referred to as L1SVM.

Given a training set of compound-protein pairs and labels $\{F(C_i, P_i), y_i\}_{i=1}^n$, $y_i \in \{+1, -1\}$, linear SVM is

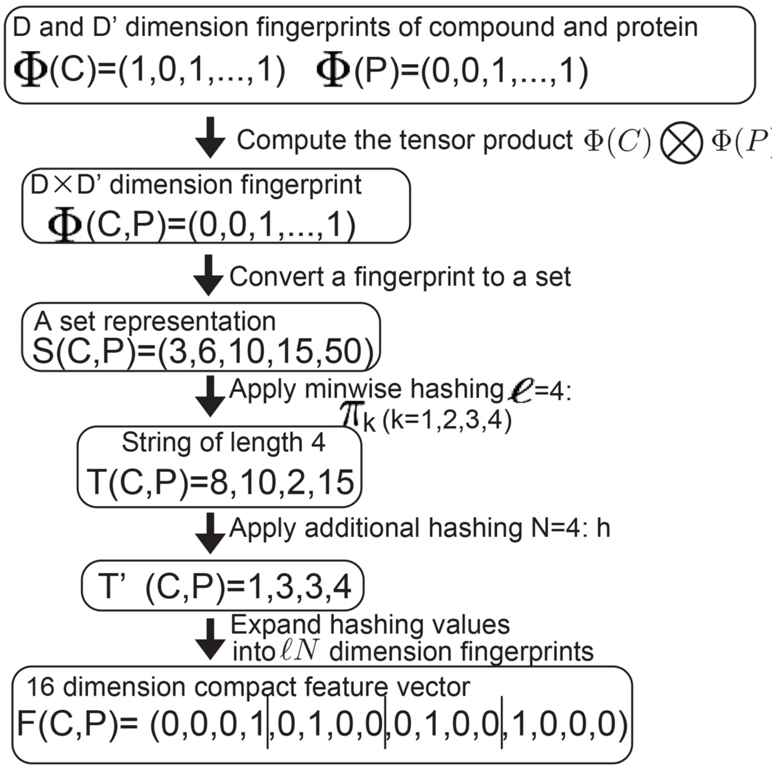

D and D' dimension fingerprints of compound and protein
Φ(C)=(1,0,1,...,1) Φ(P)=(0,0,1,...,1)
Compute the tensor product Φ(C) ⊗ Φ(P)
D×D' dimension fingerprint
Φ(C,P)=(0,0,1,...,1)
Convert a fingerprint to a set
A set representation
S(C,P)=(3,6,10,15,50)
Apply minwise hashing ℓ=4:
π_k (k=1,2,3,4)
String of length 4
T(C,P)=8,10,2,15
Apply additional hashing N=4: h
T' (C,P)=1,3,3,4
Expand hashing values into ℓN dimension fingerprints
16 dimension compact feature vector
F(C,P)= (0,0,0,1|,0,1,0,0|,0,1,0,0|,1,0,0,0)

Figure 2 Construction of a compact fingerprint. The vertical bars in *F*(*C, P*) are inserted for readability. Each range represented by the vertical bars in *F*(*C, P*) includes elements expanded from a hashing value.

formulated as the following unconstrained optimization problem:

$$\min_{\boldsymbol{w}} \sum_{i=1}^{n} \max\left\{1 - y_i \boldsymbol{w}^{\mathrm{T}} \boldsymbol{F}(C_i, P_i), 0\right\}. \quad (2)$$

To prevent overfitting, the weight vector is optimized with L_1-regularization and L_2-regularization as follows:

$$\min_{\boldsymbol{w}} ||\boldsymbol{w}||_1 + \sum_{i=1}^{n} \max\left\{1 - y_i \boldsymbol{w}^{\mathrm{T}} \boldsymbol{F}(C_i, P_i), 0\right\}. \quad (3)$$

and

$$\min_{\boldsymbol{w}} ||\boldsymbol{w}||_2 + \sum_{i=1}^{n} \max\left\{1 - y_i \boldsymbol{w}^{\mathrm{T}} \boldsymbol{F}(C_i, P_i), 0\right\}. \quad (4)$$

where $||...||_1$ and $||...||_2$ are L_1 and L_2 norms, and C is a hyper-parameter. Recently, optimization algorithms for linear SVM have rapidly advanced. In this study, we use an efficient optimization algorithm named LIBLINEAR [18][1].

[1]The software is available from http://www.csie.ntu.edu.tw/~cjlin/liblinear/

In our method, we propose to use the compact fingerprint $\boldsymbol{F}(C, P)$ instead of the original fingerprint $\Phi(C, P)$ as an input for L1SVM and L2SVM. L1SVM and L2SVM with the compact fingerprints $\boldsymbol{F}(C, P)$ are referred to as Minwise Hashing-based L1SVM (MH-L1SVM) and Minwise Hashing-based L2SVM (MH-L2SVM), respectively. In contrast, L1SVM and L2SVM with the original fingerprints $\Phi(C, P)$ are referred to as L1SVM and L2SVM, respectively, which correspond to previous methods [8].

In most previous works the kernel SVM method was used, but the input of kernel SVM is the kernel similarity matrix for compound-protein pairs [5,6], which makes it difficult to apply the kernel SVM to large-scale interaction prediction. This is because the time complexity of the quadratic programming problem for kernel SVM is $O(n_c^3 \times n_p^3)$, where n_c is the number of compounds and n_p is the number of proteins, and the space complexity is $O(n_c^2 \times n_p^2)$, which is just for storing the kernel matrix. Moreover, kernel SVM does not have any interpretability of the predictive model because it is not able to extract features.

Relation to kernel SVM

In this section, we describe a theoretical foundation for using linear SVM with compact fingerprints and discuss the relation to kernel SVM [5,6]. Kernel matrix is an $n \times n$ matrix $\boldsymbol{K}$ satisfying $\sum_{ij} c_i c_j K_{ij} \leq 0$ for all real vectors c.

Such a property is called positive definite (PD), which is necessary to effectively train an SVM classifier with a kernel matrix. A matrix $\mathbf{A}$ is PD if it can be written as an inner product of matrices $\mathbf{B}^T\mathbf{B}$.

Our linear SVM with compact fingerprints simulates non-linear SVMs with the Jaccard similarity matrix for the following reasons.

1. Each element of the pairwise kernel matrix of compound-protein pairs is defined as the number of common elements between two sets $\boldsymbol{S}(C, P)$ and $\boldsymbol{S}(C', P')$, i.e, $|\boldsymbol{S}(C, P) \cap \boldsymbol{S}(C', P')|$. The pairwise kernel matrix is PD. Jaccard similarity is a pairwise kernel normalized by the cardinality of the union of two sets $\boldsymbol{S}(C, P)$ and $\boldsymbol{S}(C', P')$, i.e., $|\boldsymbol{S}(C, P) \cup \boldsymbol{S}(C', P')|$. The Jaccard similarity matrix of compound-protein pairs, where each element is Jaccard similarity of two sets $\boldsymbol{S}(C, P)$ and $\boldsymbol{S}(C', P')$, is also PD.
2. Let the minwise hashing matrix of compound-protein pairs be a matrix whose element is defined as the inner product of two compact fingerprints $\boldsymbol{F}(C, P)$ and $\boldsymbol{F}(C', P')$. The minwise hashing matrix is PD.
3. The (i, j)-element of the Jaccard similarity matrix correlates with the (i, j)-element of the minwise hashing matrix.
4. While Jaccard similarity is a non-linear function, the inner product is a linear function.

The third reason is true because the collision probability, which is a probability that two minwise hashing and additional hashing values for two sets $\boldsymbol{S}(C, P)$ and $\boldsymbol{S}(C', P')$ are the same, is positively correlated with Jaccard similarity $\boldsymbol{J}(\boldsymbol{S}(C, P), \boldsymbol{S}(C', P'))$ (Equation 1).

Feature extraction for biological interpretation

Extracting informative features in the original fingerprint for predicting compound-protein interactions is also an important task. Since each value of the weight vector in a linear model corresponds to the importance of the corresponding feature of the original fingerprint in the classification task. In our method, we apply minwise hashing and additional hashing to the original fingerprint, and build the compact fingerprint to efficiently train a linear SVM classifier. Thus, it is not trivial to extract features in the original fingerprint in our framework.

We propose to keep inverse mappings π_k^{-1} and h^{-1} for permutation π_k and additional hashing h, and apply h^{-1} and π_k^{-1} to each element in the compact fingerprint in order to recover the weight vector for the original fingerprint. Let $\pi_k^{-1} : [1, M] \rightarrow [1, M]$ ($k = 1, ..., \ell$) be an inverse mapping for permutation $\pi_k : [1, M] \rightarrow [1, M]$. Let $h^{-1} : [1, N] \rightarrow [1, M]^*$ be an inverse mapping for additional hashing $h : [1, M] \rightarrow [1, N]$. Note that h^{-1} is, basically, a one-to-many mapping $N << M$.

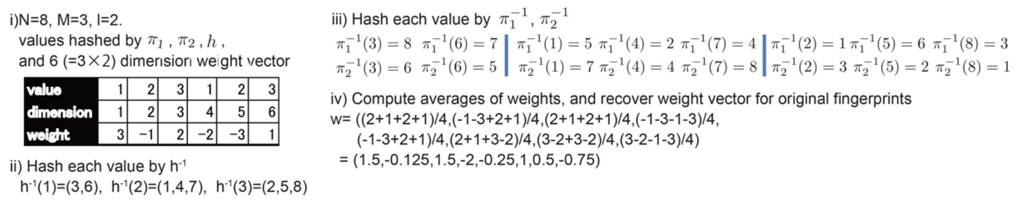
First, we apply inverse mapping h^{-1} to each element in the compact fingerprint to recover values hashed by additional hashing h. Since h^{-1} is a one-to-many mapping, several values are recovered. Then, inverse mapping π^{-1} is applied to each value in order to recover an element in the original fingerprint. Finally, we compute an average of the weights learned by linear SVMs, which provides the recovered weight vector for the original fingerprint. Figure 3 shows an illustration of the proposed procedure.

Results

Performance evaluation

We tested MH-L1SVM and MH-L2SVM (newly proposed methods) on their abilities to predict compound-protein interactions from compound substructure fingerprints and protein domain fingerprints, and compared the performance with L1SVM and L2SVM (previous methods [8]) in terms of prediction accuracy and computational cost. Note that the kernel SVM (the state-of-the-art [4-7]) was not computationally feasible for our large data. Our full dataset is too huge (consists of about 216 millions of compound-protein pairs), so we used a subset of the full data for efficient evaluation of the four different methods. In the sub-dataset, the numbers of positive and negative examples were balanced, i.e., 300,202, respectively and 600,404 in total. We performed two types of 5-fold cross-validations: pair-wise cross-validation and block-wise cross-validation.

In the pair-wise cross-validation we perform the following procedure: 1) We randomly split compound-protein pairs in the gold standard set into five subsets of roughly



i)N=8, M=3, l=2.
values hashed by π_1 , π_2 , h ,
and 6 (=3×2) dimension weight vector

value	1	2	3	1	2	3
dimension	1	2	3	4	5	6
weight	3	−1	2	−2	−3	1

ii) Hash each value by h^{-1}
$h^{-1}(1)=(3,6)$, $h^{-1}(2)=(1,4,7)$, $h^{-1}(3)=(2,5,8)$

iii) Hash each value by π_1^{-1}, π_2^{-1}
$\pi_1^{-1}(3) = 8$ $\pi_1^{-1}(6) = 7$ | $\pi_1^{-1}(1) = 5$ $\pi_1^{-1}(4) = 2$ $\pi_1^{-1}(7) = 4$ | $\pi_1^{-1}(2) = 1$ $\pi_1^{-1}(5) = 6$ $\pi_1^{-1}(8) = 3$
$\pi_2^{-1}(3) = 6$ $\pi_2^{-1}(6) = 5$ | $\pi_2^{-1}(1) = 7$ $\pi_2^{-1}(4) = 4$ $\pi_2^{-1}(7) = 8$ | $\pi_2^{-1}(2) = 3$ $\pi_2^{-1}(5) = 2$ $\pi_2^{-1}(8) = 1$

iv) Compute averages of weights, and recover weight vector for original fingerprints
w= ((2+1+2+1)/4,(-1-3+2+1)/4,(2+1+2+1)/4,(-1-3-1-3)/4,
(-1-3+2+1)/4,(2+1+3-2)/4,(3-2+3-2)/4,(3-2-1-3)/4)
= (1.5,-0.125,1.5,-2,-0.25,1,0.5,-0.75)

Figure 3 Recovery of a weight vector.

equal sizes, and take each subset in turn as a test set. 2) We train a predictive model on the remaining four subsets. 3) we compute the prediction scores for compound-protein pairs in the test set. 4) Finally, we evaluate the prediction accuracy over the five folds. The pair-wise cross-validation assumes the situation where we want to detect missing interactions between known ligand compounds and known target proteins with information about interaction partners. In the block-wise cross-validation we perform the following procedure: 1) We randomly split compounds and proteins in the gold standard set into five compound subsets and five protein subsets, and take each compound subset and each protein subset in turn as test sets. 2) We train a predictive model on compound-target pairs in the remaining compound subsets and four protein subsets. 3) We compute the prediction scores for compound-protein pairs involving test compound set and test protein set. 4) Finally, we evaluate the prediction accuracy over the five folds. The block-wise cross-validation assumes the situation where we want to detect new interactions for newly arriving ligand candidate compounds and target candidate proteins with no information about interaction partners. In the both cases, we evaluated the performance by the area under the ROC curve (AUC) and execution time. The cross-validations were performed by varying the hyper-parameter $C = 10^{-5}, 10^{-4}, ..., 10^{5}$ and chosen as the one to achieve the best AUC score.

We investigated the effects of the length of strings l and the size of hashing values N in the minwise hashing process of MH-L1SVM and MH-L2SVM on the performance. We tried five different lengths of string $\ell = 5, 10, 15, 30, 50$. The size of additional hashing values N is varied from 2^2 to 2^{32}. Figure 4 and 5 shows the AUC scores for MH-L1SVM and MH-L2SVM in the pair-wise cross validation. It was observed that the AUC scores reached the maximum with the length of string $\ell = 10$ and the size of additional hashing value $N = 2^{16}$, and the AUC score was comparable to that for the original fingerprint.

Figure 6 and 7 shows the execution time for performing the minwise hashing and for learning SVM classifiers, where the length of string ℓ is varied from 5 to 50 and the size of additional hashing value is fixed to $N = 2^{16}$. The AUC scores of MH-L1SVM and MH-L2SVM with the length of string $\ell = 10$ and the size of additional hashing $N = 2^{16}$ were comparable to those of L1SVM and L2SVM. In addition, MH-L1SVM and MH-L2SVM achieved certain speedup compared with L1SVM and L2SVM.

The same trends of these results in the pair-wise cross-validation were observed in the case of the block-wise cross-validation as well. The corresponding results for the block-wise cross-validation are shown in Figures 8, 9, 10 and 11. The AUC scores in the block-wise cross-validation were lower than those in the pair-wise cross-validation, which implies that predicting unknown interactions for newly coming compounds and proteins outside of the learning set is much more difficult than detecting missing interactions between compounds and proteins in the learning set.

Experiments on large-scale datasets

We evaluated the performance for the full data consisting of 216,121,626 compound-protein pairs, where the best parameter values for each method in the cross-

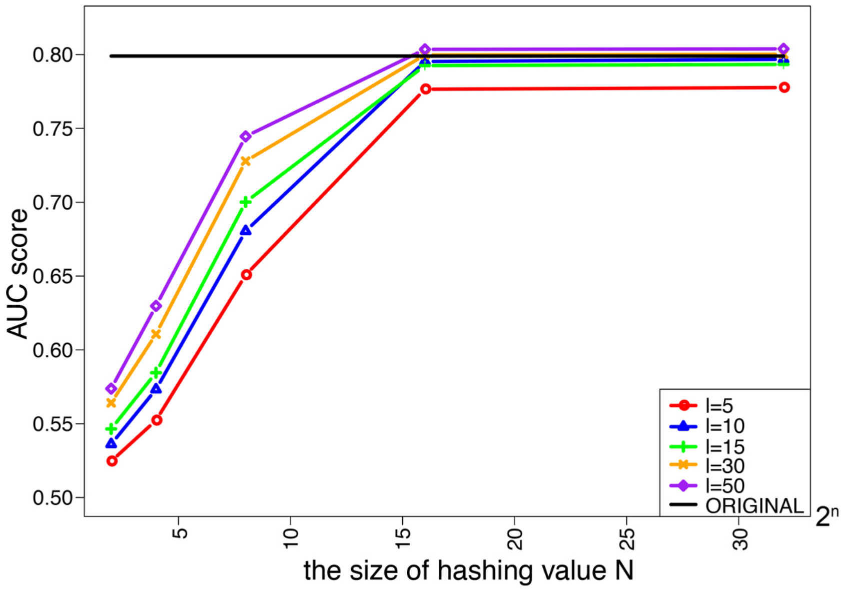


Figure 4 AUC score of L1SVM for varying the size of additional hashing value *N*.

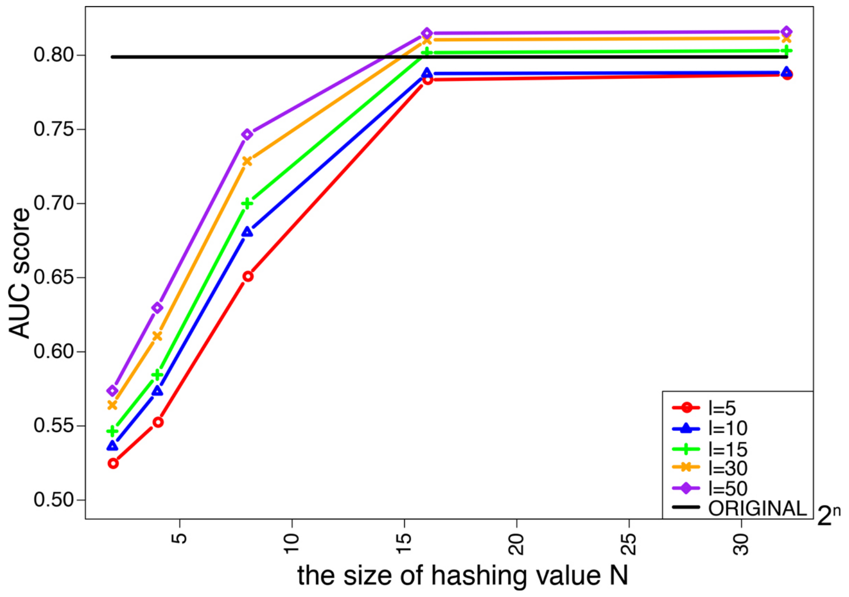


Figure 5 AUC score of L2SVM for varying the size of additional hashing value *N*.

validation experiments in the previous subsection were used. We examined the effect of the ratio of positive compound-protein pairs against negative compound-protein pairs on the performance. Note that the number of negative examples is much larger than that of positive examples in our dataset. We varied the number of negative examples in the cross-validation from the same number of positive examples to the number of all possible negative examples.

Figure 12 shows the memory usages of the four different methods. It was observed that the memory usage grew linearly as the number of compound-protein pairs increased in each method. Especially, both L1SVM and L2SVM required about 200GB in memory. On the other hand, MH-L1SVM and MH-L2SVM took only about 30GB in memory. There is little difference of memory usage between L1-regularization and L2-regularization.

Table 1 shows the AUC scores in the pair-wise cross-validation. It was observed that the AUC scores of MH-L1SVM and MH-L2SVM were comparable to those of L1SVM and L2SVM, respectively. Table 2 shows training time on the pair-wise cross-validation, where the training

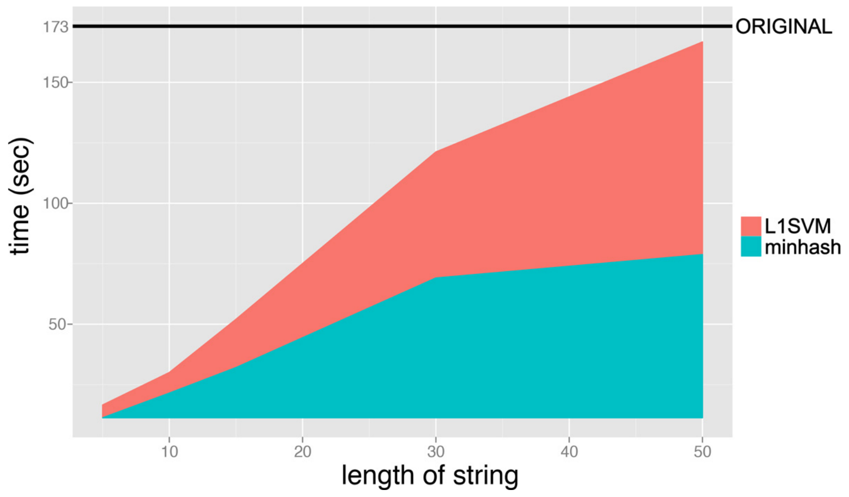


Figure 6 Learning time of L1SVM for fixing the size of additional hashing value $N = 2^{16}$ and varying the length of string ℓ.

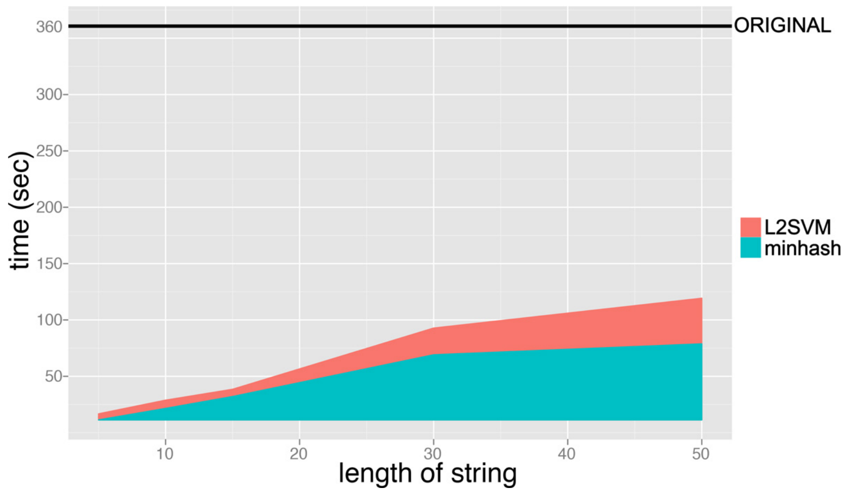


Figure 7 **Learning time of L2SVM for fixing the size of additional hashing value $N = 2^{16}$ and varying the length of string ℓ.**

time includes the minwise hashing process and the upper limitation is put on the execution time for all methods to 24 hours. MH-L1SVM and MH-L2SVM are significantly faster than L1SVM and L2SVM, respectively. Especially, MH-L2SVM is about 10 times faster than L2SVM. L1SVM did not finish the computation for such a large number of compound-protein pairs within 24 hours. On the other hand, our MH-L1SVM finished the computation and took only 25,060 seconds on average.

The same trends for these results in the pair-wise cross-validation were observed in the block-wise cross-validation as well (See Tables 3 and 4).

Table 5 shows the AUC scores and training times in using all possible negative examples, where only 1-fold of the 5-fold cross-validation was performed on this dataset. On this extremely large data, L1SVM and L2SVM did not finish the computation within 24 hours. On the other hand, MH-L1SVM and MH-L2SVM finished the computation, and the AUC scores were reasonable. The training times of MH-L1SVM and MH-L2SVM were 157,013 and 10,054 seconds, respectively. These results suggest the usefulness of our proposed methods in large-scale applications.

Figure 13 shows the numbers of features extracted by MH-L1SVM and MH-L2SVM. The number of features

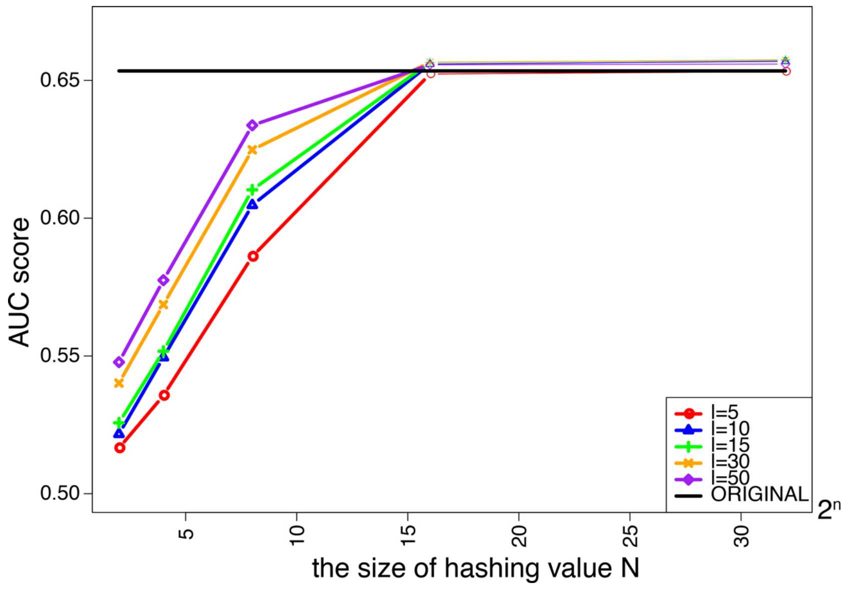


Figure 8 **AUC score of L1SVM for varying the size of additional hashing value *N*.**

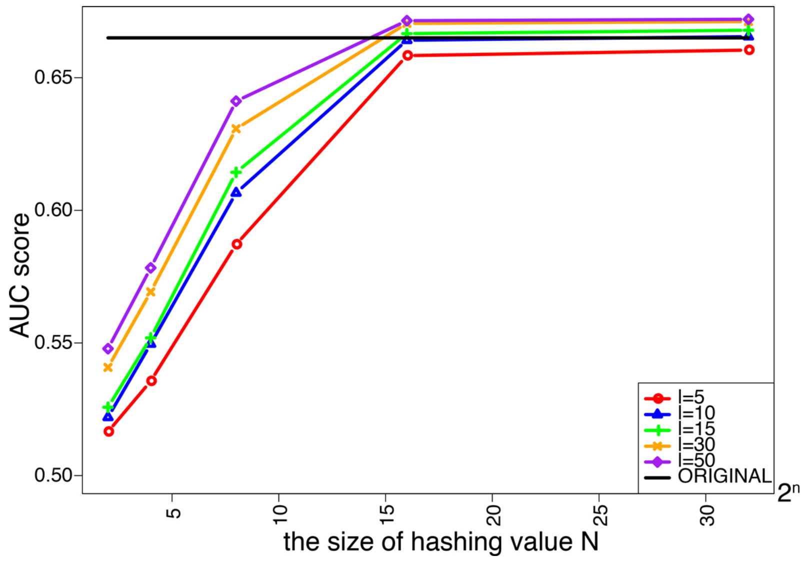


Figure 9 AUC score of L2SVM for varying the size of additional hashing value *N*.

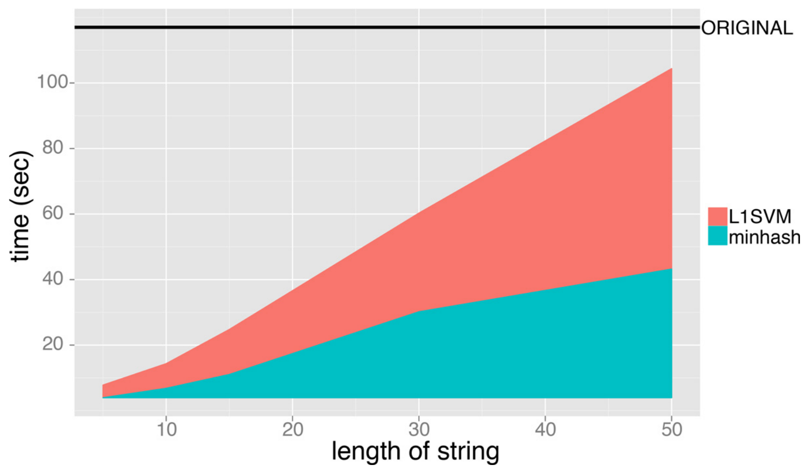


Figure 10 Learning time of L1SVM for fixing the size of additional hashing value 2^{16} and varying the length of string.

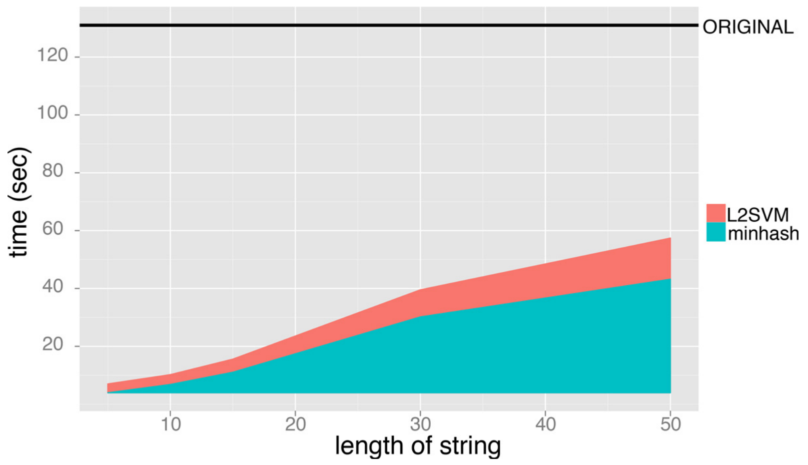


Figure 11 Learning time of L2SVM for fixing the size of additional hashing value 2^{16} and varying the length of string.

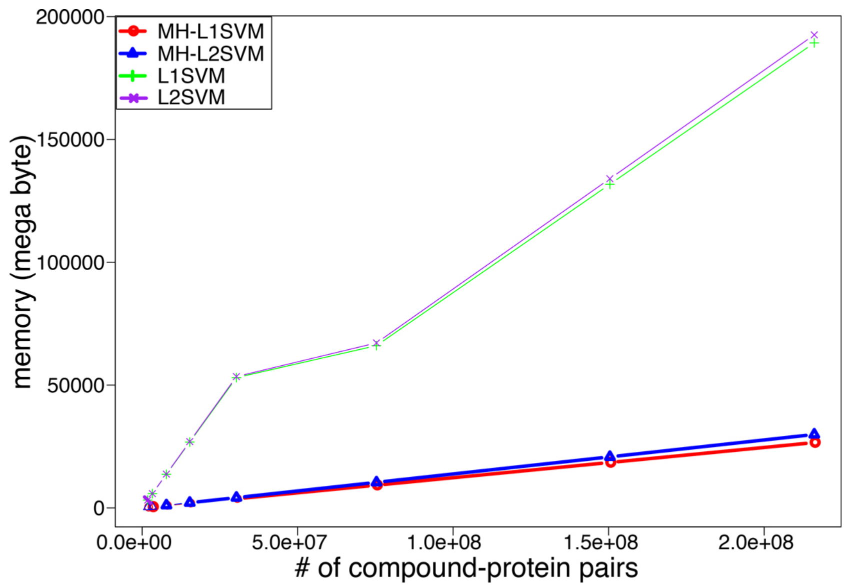


Figure 12 Memory usage for increasing the number of compound-protein pairs.

Table 1 AUC score on pair-wise cross validation experiments

Ratio	Number	MH-L1SVM	MH-L2SVM	L1SVM	L2SVM
1	600, 404	$0.78 \pm 2.31 \times 10^{-6}$	$0.79 \pm 2.31 \times 10^{-6}$	$0.79 \pm 3.22 \times 10^{-6}$	$0.80 \pm 4.97 \times 10^{-6}$
5	1, 801, 212	$0.79 \pm 7.23 \times 10^{-7}$	$0.80 \pm 8.30 \times 10^{-7}$	$0.81 \pm 2.04 \times 10^{-7}$	$0.81 \pm 2.04 \times 10^{-7}$
10	3, 302, 222	$0.79 \pm 1.84 \times 10^{-6}$	$0.80 \pm 1.35 \times 10^{-6}$	$0.81 \pm 5.34 \times 10^{-7}$	$0.81 \pm 4.31 \times 10^{-7}$
25	7, 805, 252	$0.79 \pm 2.89 \times 10^{-7}$	$0.80 \pm 6.28 \times 10^{-8}$	$0.81 \pm 9.87 \times 10^{-8}$	$0.81 \pm 1.30 \times 10^{-7}$
50	15, 310, 302	$0.79 \pm 3.21 \times 10^{-7}$	$0.81 \pm 3.79 \times 10^{-7}$	$0.81 \pm 3.40 \times 10^{-8}$	$0.81 \pm 1.72 \times 10^{-7}$
100	30, 320, 402	$0.79 \pm 2.38 \times 10^{-7}$	$0.81 \pm 1.49 \times 10^{-7}$	-	$0.81 \pm 2.43 \times 10^{-7}$
250	75, 350, 702	$0.79 \pm 2.91 \times 10^{-7}$	$0.81 \pm 2.42 \times 10^{-7}$	-	$0.81 \pm 3.66 \times 10^{-7}$

The number of negative examples is varied from the same number of positive examples to the number of negative examples 250 times larger than the number of all positive examples.

extracted by MH-L1SVM are about third times smaller than that of features extracted by MH-L2SVM. This result suggests that MH-L1SVM provides us with more selective features, which would help to make a biological interpretation about the functional associations between compound substructures and protein domains behind compound-protein interactions.

Discussion and conclusion

In this paper we proposed a novel chemogenomic method to predict unknown compound-protein interactions on a large scale, which was made possible by using an improved minwise hashing algorithm to efficiently represent the fingerprints of compound-protein pairs. Interestingly, the linear SVM with the compact fingerprints generated by the

Table 2 Training time on pair-wise cross validation experiments

Ratio	Number	MH-L1SVM	MH-L2SVM	L1SVM	L2SVM
1	600, 404	29 ± 1	28 ± 1	188 ± 32	387 ± 63
5	1, 801, 212	172 ± 5	38 ± 2	1, 655 ± 156	963 ± 81
10	3, 302, 222	448 ± 41	261 ± 7	1, 261 ± 579	10, 798 ± 1, 981
25	7, 805, 252	1, 808 ± 181	732 ± 17	20,067 ± 1,453	4, 623 ± 782
50	15, 310, 302	1,140 ± 90	811 ± 41	58, 045 ± 5, 678	8, 936 ± 1, 412
100	30, 320,402	7, 601 ± 627	1,643 ± 50	> 24hours	16, 608 ± 2, 732
250	75, 350, 702	25,060 ± 12,417	4,631 ± 795	> 24hours	43, 843 ± 7, 200

The training time includes minwise hashing time. The number of negative examples is varied from the same number of positive examples to the number of negative examples 250 times larger than the number of all positive examples.

Table 3 AUC scores on block-wise cross validation experiments

Ratio	Number	MH-L1SVM	MH-L2SVM	L1SVM	L2SVM
1	600, 404	0.66 ± 0.00	0.66 ± 0.01	0.65 ± 0.01	0.67 ± 0.01
5	1, 801, 212	0.66 ± 0.01	0.66 ± 0.01	0.66 ± 0.01	0.67 ± 0.01
10	3, 302, 222	0.66 ± 0.01	0.67 ± 0.01	0.66 ± 0.01	0.67 ± 0.01
25	7, 805, 252	0.66 ± 0.01	0.66 ± 0.01	0.65 ± 0.01	0.66 ± 0.01
50	15, 310, 302	0.66 ± 0.01	0.66 ± 0.01	0.65 ± 0.01	0.66 ± 0.01

The number of negative examples is varied from the same number of positive examples to the number of negative examples 50 times larger than the number of all positive examples.

Table 4 Training times on block-wise cross validation experiments

Ratio	Number	MH-L1SVM	MH-L2SVM	L1SVM	L2SVM
1	600, 404	8 ± 0	7 ± 1	131 ± 7	117 ± 28
5	1, 801, 212	76 ± 9	31 ± 1	982 ± 184	252 ± 30
10	3, 302, 222	237 ± 23	55 ± 4	3925 ± 92	475 ± 93
25	7, 805, 252	582 ± 79	107 ± 4	2606 ± 265	322 ± 20
50	15, 310, 302	1889 ± 82	243 ± 8	7133 ± 664	729 ± 18

The training time includes minwise hashing time. The number of negative examples is varied from the same number of positive examples to the number of negative examples 50 times larger than the number of all positive examples.

Table 5 AUC score and training time on the full data consisting of all 216,121,626 compound-protein pairs

Measure	MH-L1SVM	MH-L2SVM	L1SVM	L2SVM
AUC score	0.79	0.81	-	-
Training Time (sec)	157, 013	10, 054	> 24hours	> 24hours

minwise hashing is able to simulate the nonlinear property of the kernel SVM (the state-of-the-art). The originality of the proposed method lies in the scalable prediction of compound-protein interactions, in the computational efficiency, and in the interpretability of the predictive model. It should be pointed out that all previous methods were not computationally feasible for the full data. The proposed method is expected to be useful for virtual screening of a large number of compounds against many protein targets.

The proposed method can be used, as soon as compounds and proteins are represented by binary descriptors

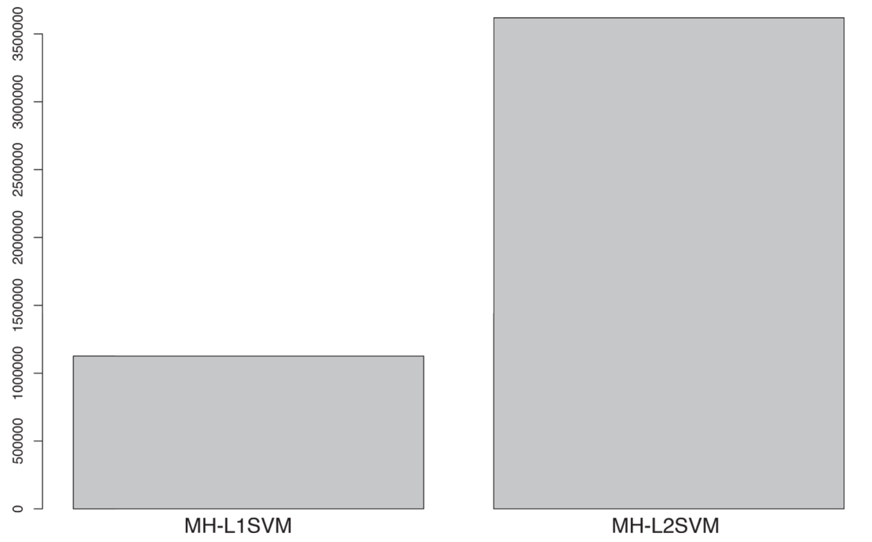


Figure 13 The number of features extracted by MH-L1SVM and MH-L2SVM.

(chemical substructures and protein domains in this study). However, a limitation of the proposed method is that the performance depends on the definitions of chemical substructures of compounds and functional domains of proteins. The use of other descriptors (e.g., KlekotaRoth, ECFP6, Daylight, and Dragon) could improve the generalization properties of the method. Datasets, all results and softwares are available at https://sites.google.com/site/interactminhash/.

Competing interests
None declared.

Authors' contributions
YT implemented the algorithm of the methods, made all analyses, and drafted the manuscript. YY prepared the datasets, and drafted the manuscript.

Acknowledgements
This work was supported by MEXT/JSPS KAKENHI Grant Numbers 24700140 and 25700029. This work was also supported by the Program to Disseminate Tenure Tracking System, MEXT, Japan, and Kyushu University Interdisciplinary Programs in Education and Projects in Research Development. This work was also supported by the PRESTO program of the Japan Science and Technology Agency (JST).

Declarations
The publication cost for this work was supported by the PRESTO program of the Japan Science and Technology Agency (JST).
This article has been published as part of *BMC Systems Biology* Volume 7 Supplement 6, 2013: Selected articles from the 24th International Conference on Genome Informatics (GIW2013). The full contents of the supplement are available online at http://www.biomedcentral.com/bmcsystbiol/supplements/7/S6.

Authors' details
[1]PRESTO, Japan Science and Technology Agency, Kawaguchi, Saitama 332-0012, Japan. [2]Division of System Cohort, Medical Institute of Bioregulation, Kyushu University, 3-1-1 Maidashi, Higashi-ku, Fukuoka, Fukuoka 812-8582, Japan. [3]Institute for Advanced Study, Kyushu University, 6-10-1, Hakozaki, Higashi-ku, Fukuoka, Fukuoka 812-8581, Japan.

Published: 13 December 2013

References

1. Chen B, Wild D, Guha R: **PubChem as a source of polypharmacology.** *J Chem Inf Model* 2009, **49**:2044-2055.
2. Stockwell B: **Chemical genetics: ligand-based discovery of gene function.** *Nat Rev Genet* 2000, **1**:116-125.
3. Dobson C: **Chemical space and biology.** *Nature* 2004, **432(7019)**:824-828.
4. Nagamine N, Sakakibara Y: **Statistical prediction of protein-chemical interactions based on chemical structure and mass spectrometry data.** *Bioinformatics* 2007, **23**:2004-2012.
5. Faulon JL, Misra M, Martin S, Sale K, Sapra R: **Genome scale enzyme-metabolite and drug-target interaction predictions using the signature molecular descriptor.** *Bioinformatics* 2008, **24**:225-233.
6. Jacob L, Vert JP: **Protein-ligand interaction prediction: an improved chemogenomics approach.** *Bioinformatics* 2008, **24**:2149-2156.
7. Yabuuchi H, Niijima S, Takematsu H, Ida T, Hirokawa T, Hara T, Ogawa T, Minowa Y, Tsujimoto G, Okuno Y: **Analysis of multiple compound-protein interactions reveals novel bioactive molecules.** *Mol Syst Biol* 2011, **7**:472.
8. Tabei Y, Pauwels E, Stoven V, Takemoto K, Yamanishi Y: **Identification of chemogenomic features from drug-target interaction networks using interpretable classifiers.** *Bioinformatics* 2012, **28**:i487-i494.
9. Chang C, Lin C: **LIBSVM: a library for support vector machines.** *ACM Transactions on Intelligent Systems and Technology (TIST)* 2011, **2**:1-27.
10. Joachims T: In *Learning to classify text using support vector machines: Methods, Theory and Algorithms. Volume 186.* Kluwer Academic Publishers; 2002.
11. Todeschini R, Consonni V: *Handbook of Molecular Descriptors* New York, USA: Wiley-VCH; 2002.
12. Chen J, Swamidass S, Dou Y, Bruand J, Baldi P: **ChemDB: a public database of small molecules and related chemoinformatics resources.** *Bioinformatics* 2005, **21**:4133-4139.
13. Lodhi H, Yamanishi Y: **Chemoinformatics and Advanced Machine Learning Perspectives: Complex Computational Methods and Collaborative Techniques.** *IGI Global* 2010.
14. Park K, Kanehisa M: **Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs.** *Bioinformatics* 2003, **19**:1656-1663.
15. Lanckriet G, Deng M, Cristianini N, Jordan M, Noble W: **Kernel-based data fusion and its application to protein function prediction in yeast.** *Pac Symp Biocomput* 2004, 300-311.
16. Ben-Hur A, Soon Ong C, Sonnenburg S, Schölkopf B, Rätsch G: **Support Vector Machines and Kernels for Computational Biology.** *PLoS Computational Biology* 2008, **4(10)**:e1000173.
17. Fan RE, Chang KW, Hsieh CJ, Wang X, Lin CJ: **LIBLINEAR:A library for large linear classification.** *The Journal of Machine Learning Research* 2008, **9**:1871-1874.
18. Hsieh CJ, Chang KW, Lin CJ, Keerthi SS, S S: **A Dual Coordinate Descent Method for Large-scale Linear SVM.** *Proceedings of the 25th international conference on Maching Learning* 2008, 408-415.
19. Joachims T: **Training Linear SVMs in Linear Time.** *Proceedings of the 12th ACM SIGKDD Conference on Knowledge Discover and Data Mining* 2006, 217-226.
20. Shalev-Shwartz S, Singer Y, Srebro N: **Pegasos: primal estimated sub-gradient solver for SVM.** *Proceedings of the 24th international conference on Machine learning* 2007, 807-814.
21. Kuhn M, Szklarczyk D, Franceschini A, Campillos M, von Mering C, Jensen L, Beyer A, Bork P: **STITCH 2: an interaction network database for small molecules and proteins.** *Nucleic Acids Res* 2010, **38(suppl 1)**:D552-D556.
22. Consortium TU: **The Universal Protein Resource (UniProt) in 2010.** *Nucleic Acids Res* 2010, **38**:D142-D148.
23. Finn R, Tate J, Mistry J, Coggill P, Sammut J, Hotz H, Ceric G, Forslund K, Eddy S, Sonnhammer E, Bateman A: **The Pfam protein families database.** *Nucleic Acids Res* 2008, **36**:D281-D288.
24. Broder A, Charikar M, Frieze A: **Min-Wise Independent Permutations.** *Journal of Computer and System Sciences* 2000, **60**:630-659.
25. Li P, Köning AC: **b-bit minwise hashing.** *Proceedings of 27th International World Wide Web Conference* 2010, 671-680.
26. Li P, Köning AC, Gui W: **b-bit minwise hashing for estimating three-way similarities.** *Twenty-Fourth Annual Conference on Neural Information Processing Systems* 2010.

doi:10.1186/1752-0509-7-S6-S3
Cite this article as: Tabei and Yamanishi: **Scalable prediction of compound-protein interactions using minwise hashing.** *BMC Systems Biology* 2013 **7**(Suppl 6):S3.