https://doi.org/10.1016/j.jtbi.2016.03.013

Mem-ADSVM: A Two-Layer Multi-Label Predictor for Identifying Multi-Functional Types of Membrane Proteins

Shibiao Wan[a,*], Man-Wai Mak[a,*], Sun-Yuan Kung[b]

[a]*Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong SAR, China*
[b]*Department of Electrical Engineering, Princeton University, New Jersey, USA.*

Abstract

Identifying membrane proteins and their multi-functional types is an indispensable yet challenging topic in proteomics and bioinformatics. However, most of the existing membrane-protein predictors have the following problems: (1) they do not predict whether a given protein is a membrane protein or not; (2) they are limited to predicting membrane proteins with single-label functional types but ignore those with multi-functional types; and (3) there is still much room for improvement for their performance. To address these problems, this paper proposes a two-layer multi-label predictor, namely Mem-ADSVM, which can identify membrane proteins (Layer I) and their multi-functional types (Layer II). Specifically, given a query protein, its associated gene ontology (GO) information is retrieved by searching a compact GO-term database with its homologous accession number. Subsequently, the GO information is classified by a binary support vector machine (SVM) classifier to determine whether it is a membrane protein or not. If yes, it will be further classified by a multi-label multi-class SVM classifier equipped with an adaptive-decision (AD) scheme to determine to which functional type(s) it belongs. Experimental results show that Mem-ADSVM significantly outperforms state-of-the-art predictors in terms of identifying both membrane proteins and their multi-functional types. This paper also suggests that the two-layer prediction architecture is better than the one-layer for prediction performance. For reader's convenience, the Mem-ADSVM server is available online at `http://bioinfo.eie.polyu.edu.hk/MemADSVMServer/`.

Keywords: membrane protein type prediction; multi-label classification; adaptive-decision scheme; gene ontology; two-layer classification.

1. Introduction

According to the characteristic sequence and structural features, proteins are divided into four common types [1]: membrane, soluble (or globular), fibrous and intrinsically disordered. Among them, membrane proteins are found to play essential roles in a variety of vital biological processes [2] by interacting with the membranes of a cell or an organelle. Membrane proteins are targets of almost half of all medicinal drugs [3, 4], because they mediate many interactions between cells and extracellular surroundings as well as between the cytosol and membrane-bound organelles. Despite owning the same basic phospholipid bilayer structure [5], membrane proteins perform various and diversified functions. Therefore, given a query protein, identifying whether it is a membrane protein or not is an indispensable yet challenging topic in proteomics and bioinformatics.

Membrane proteins can be further divided into different functional types. Conventionally, some studies [5] broadly classified membrane proteins into two categories, namely integral (or intrinsic) and peripheral (or extrinsic), depending upon the interactions between membrane proteins and the membrane. Other studies [6] grouped membrane proteins into three distinct classes: integral, peripheral and lipid-anchored. With the exponentially growing number of protein sequences discovered in the post-genomic era, these three classes of membrane proteins are further divided into eight types [7]: (1) single-pass type I; (2) single-pass type II; (3) single-pass type III; (4) single-pass type IV; (5) multi-pass; (6) lipid-anchor; (7) GPI-anchor and (8) peripheral. More information about the hierarchical relationships between these eight types and the former three classes can be found in [8]. Particularly, GPI-anchored proteins (Type 7) is a kind of special lipid-anchored proteins (Type 6).[1] Type 7 is singled out from Type 6 because GPI-anchored proteins ubiquitously exist in many species and have been intensively studied for their unique functions [9]. Details about these eight functional types are also elaborated in [8].

Knowing the functional types of membrane proteins can be helpful to elucidate the biological functions of membrane proteins. For example, phospholipases [10], belonging to Type-8, are a group of water-soluble enzymes that are temporarily bound to the polar head groups of membrane phospholipids. Their major functions are lipid signaling, which can be achieved by hydrolizing various bonds linking phospholipases with the lipid layer to which they are temporarily at-

*Corresponding author
Email addresses: `shibiao.wan@connect.polyu.hk` (Shibiao Wan), `enmwmak@polyu.edu.hk` (Man-Wai Mak), `kung@princeton.edu` (Sun-Yuan Kung)

[1] http://www.uniprot.org/locations/SL-9902

tached. Moreover, about 20%∼35% of genes contain the instructions for producing membrane proteins, whereas the structurally annotated membrane proteins only account for less than 1% of the proteins with known structures [11]. Knowing the functional types of membrane proteins can accelerate the process of annotating their structures. Besides, because of their fluidity property, membrane proteins can freely move within the lipid bilayer to the location where their functions are performed. The knowledge of the functional type of a membrane protein can help reveal the mechanisms of this kind of biological activities. Therefore, it is highly necessary to develop computational approaches for timely and accurate prediction of the functional types of membrane protein. Ideally, the computational approaches should perform two-layer predictions.

1. Layer I: Given a query protein, the predictor determines whether the query protein is a membrane protein or not.
2. Layer II: If the answer in Layer I prediction is 'yes', the predictor determines the functional type(s) of the protein.

In recent years, impressive progress has been made in predicting the functional types of membrane proteins [7, 11, 16, 17, 18, 19, 20, 21] and in the prediction of membrane proteins in specific subcellular locations [22, 23]. While many advanced predictors have been developed, they still have several limitations, which are elaborated below.

1. These predictors assume that all query proteins are membrane proteins. If a query protein is not a membrane protein, these predictors will attempt to determine the most likely functional type of the protein. This is obviously undesirable because a non-membrane protein does not have a *membrane* functional type. Given that membrane proteins are just one of the four common types of proteins [1], it is important to ensure that the query protein is really a membrane protein prior to determining its functional type(s). As far as we know, only three predictors, namely LeastEudist [12], ProtLoc [13] and MemType-2L [7],[2] are capable of predicting whether a query protein is a membrane protein or not. These predictors are summarized in Layer I of Table 1. As can be seen, these predictors use sequence-based features (i.e., amino-acid compositions and pseudo position-specific score matrices) to discriminate membrane proteins from non-membrane proteins.
2. They are limited to the prediction of membrane proteins with single-label functional types. However, many membrane proteins were found to simultaneously belong to multiple functional types. For example, the envelope glycoprotein p57 [24, 25] was reported to belong to single-pass type I (Type 1) when locating in the host endoplasmic reticulum membrane, and simultaneously it belongs to peripheral (Type 8) when locating in the host cell membrane. Table 1 lists the existing multi-label predictors for membrane protein type prediction (Layer II). To the best of our knowledge, only three predictors, namely Mem-PseAA [14], iMem-Seq [15] and Mem-mEN [8],[3] are able to predict multi-label membrane proteins. In terms of feature extraction, iMem-Seq uses the information from position-specific score matrices and physical-chemical property matrices; Mem-PseAA uses feature information from pseudo-amino acid compositions; Mem-mEN uses the information from homologous gene ontology term frequencies. In terms of classification, both Mem-PseAA and iMem-Seq use multi-label kNN classifiers, whereas Mem-mEN uses a multi-label elastic net (EN) classifier. Particularly, Mem-mEN possesses the property of 'interpretability' [8], which can provide biological reasons on why a query protein belongs to the predicted type(s).
3. The performance of existing predictors are far from satisfactory. In particular, it has been shown [8] that Mem-mEN performs better than Mem-PseAA and iMem-Seq. However, the performance of all of these predictors still remains to be improved. Besides, the one-layer architecture of these predictors cannot model or capture the intrinsic inter-class correlations, such as the exclusiveness between the non-membrane type and the other eight membrane types, causing poor prediction performance.

To address the aforementioned problems, this paper proposes an efficient two-layer multi-label predictor, namely Mem-ADSVM, which can identify membrane proteins (Layer I) and predict membrane proteins with single- and multi-label functional types (Layer II). Similar to Mem-mEN, Mem-ADSVM extracts features by exploiting the gene ontology (GO) information retrieved from a compact GO-term database. Unlike Mem-mEN that uses a multi-label elastic net classifier for Layer II prediction, Mem-ADSVM harnesses a binary SVM classifier for Layer I prediction and a multi-label SVM classifier equipped with an adaptive-decision scheme for Layer II prediction.

Experimental results on three recent benchmark datasets demonstrate the superiority of Mem-ADSVM over existing state-of-the-art predictors in terms of both identifying membrane proteins and predicting multi-functional types. Besides, the results suggest that GO information is more discriminative for predicting functional types of membrane proteins than amino acid sequences. This work also found that the proposed adaptive-decision scheme is beneficial to improving the performance of Mem-ADSVM. Besides, the superiority of using two-layer prediction architecture over one-layer is also presented.

2. Feature Extraction

Fig. 1 shows the flowchart of Mem-ADSVM, including (a) the whole architecture, (b) feature extraction, (c) Layer I classification and (d) Layer II classification. Cylinders in blue represent databases, and rectangles in orange represent processing/procedures. As shown in Fig. 1(a), a query protein will

[2]Note that LeastEudist and ProtLoc were implemented in [7]. For ease of reference, we use the name LeastEudist to denote the method proposed in [12].

[3]For ease of reference, we refer to the predictor proposed in [14] as Mem-PseAA.

Table 1: Summary of existing predictors for identifying membrane proteins (Layer I) and predicting membrane protein functional types. *Pse-PSSM*: pseudo position-specific score matrix (PSSM); *AA*: amino-acid composition; *PseAA*: pseudo amino acid composition; *improved PSSM*: PSSM with AA physical-chemical properties; *KNN*: K-nearest neighbor; *OET-KNN*: optimized evidence-theoretic KNN; *EN*: elastic net.

Stratification	Predictor	Features	Classifier	Multi-label	No. of classes
Layer I	MemType-2L [7]	Pse-PSSM	ensemble OET-KNN	No	2
	LeastEudist [12]	AA	least Euclidean distance	No	2
	ProtLoc [13]	AA	least Mahalanobis distance	No	2
Layer II	Mem-PseAA [14]	PseAA	multi-label KNN	Yes	8
	iMem-Seq [15]	improved PSSM	multi-label KNN	Yes	8
	Mem-mEN [8]	GO terms	multi-label EN	Yes	8

go through three steps: (1) feature extraction, (2) Layer I classification and (3) Layer II classification. Particularly, if the query protein is predicted to be a non-membrane protein in Step 2, then the predictor will return 'non-membrane' as the result without proceeding to Step 3.

In this work, we used the gene ontology (GO) information as the features for discriminating membrane proteins and their functional types. The flowchart of feature extraction is shown in Fig. 1(b). To avoid null-GO vectors and to address the storage complexity problem, we used two compact databases, namely ProSeq and ProSeq-GO [26] to replace the traditional Swiss-Prot and GOA databases. After retrieving GO terms, term-frequency based approach was used to construct the GO vectors. Thus, this section includes the following three subsection: (1) GO information as features; (2) creation of compact databases; and (3) GO-Vector Construction.

2.1. GO Information as Features

Gene Ontology (GO) is an influential and major bioinformatics initiative to standardize the attribute representations of genes and gene products across all species [27]. The past decades have witnessed successful applications of GO-based approaches in various bioinformatics domains, such as protein-protein interaction inference [28, 29, 30], microarray clustering [31], protein function prediction [32, 33], subnuclear localization prediction [34] and protein subcellular localization prediction [35, 36, 37, 38, 39, 40, 41]. Particularly, extensive analyses and comparisons among different GO-based subcellular-location predictors have been reported in a recent book [40]. However, GO-based approaches are hardly without disadvantages. One of the challenges is how to deal with query proteins whose GO information is not available in the gene ontology annotation (GOA) database.[4] This situation is especially prevalent in those newly discovered proteins which have not been functionally annotated yet.

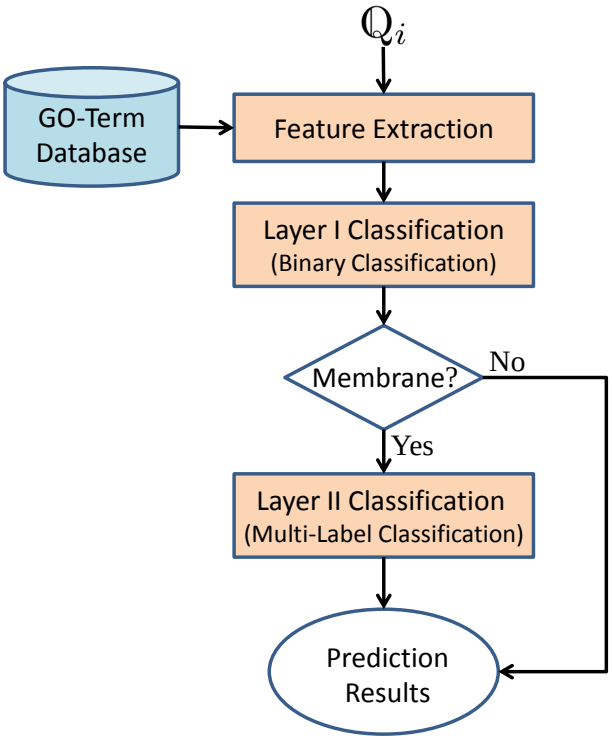
Traditionally, given a query protein, if its accession number (AC) does not associate with any GO terms in the GOA database, BLAST search [42] is used to find the AC of the top homolog of the query protein. Subsequently, the homologous AC is used as a key to search against the GOA database to find a set of GO terms and a GO vector (see Section 2.3 below) can be constructed. In this way, the homologous GO information is effectively transferred to the query protein. This strategy, nonetheless, will still lead to null GO vectors when the top homologous protein has not been annotated in the GOA database, i.e., its AC does not associate with any GO terms. To address this problem, some predictors [43, 44] use the AC of lower-rank homologs as a replacement until a non-null GO vector can be found. Some others give up using GO information and apply back-up methods that rely on other features such as pseudo-amino-acid composition [45, 46] and sorting signals [47]. Nevertheless, the backup methods usually lead to poor prediction accuracy for the proteins to which GO-based approaches are not applicable.

2.2. Compact Databases Creation

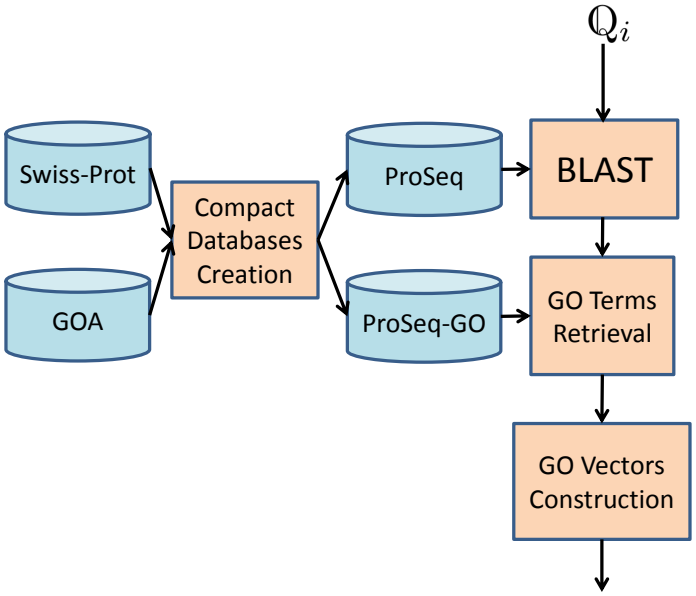
In this work, similar to our previous studies [26], we used two compact yet efficient databases so that GO-based approaches are applicable to all proteins. These two databases, namely ProSeq and ProSeq-GO, were obtained by filtering the traditional Swiss-Prot database and the GOA database. ProSeq is a subset of Swiss-Prot, whereas ProSeq-GO is a subset of the GOA database. These two databases possess the following properties:

1. **Efficiency.** Using ProSeq and ProSeq-GO for retrieving GO terms can avoid null-GO vectors because the filtering process guarantees that ProSeq will only keep the sequences in the Swiss-Prot database whose ACs have at least one GO term in ProSeq-GO. With the rapid progress of the Swiss-Prot database, in most circumstances we can always use BLAST to find a homolog for a query protein. In case no close homolog is found, we can increase the E-value until we can find some lower-rank homologs. The same argument also applies to ProSeq. As a result, all of the homologous ACs will be associated with at least one GO terms and the GO vectors will have at least one non-null entry.
2. **Compactness.** ProSeq and ProSeq-GO are very compact in that they do not require large storage space and their software implementation does not consume significant memory space. In fact, reducing the storage complexity is one of the important issues that GO-based methods

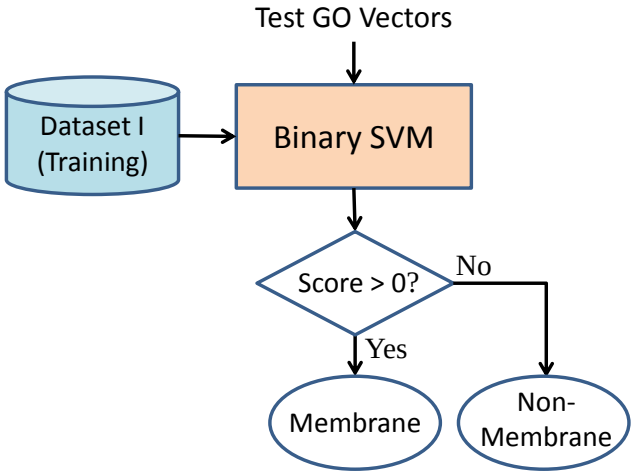
[4] http://www.ebi.ac.uk/GOA



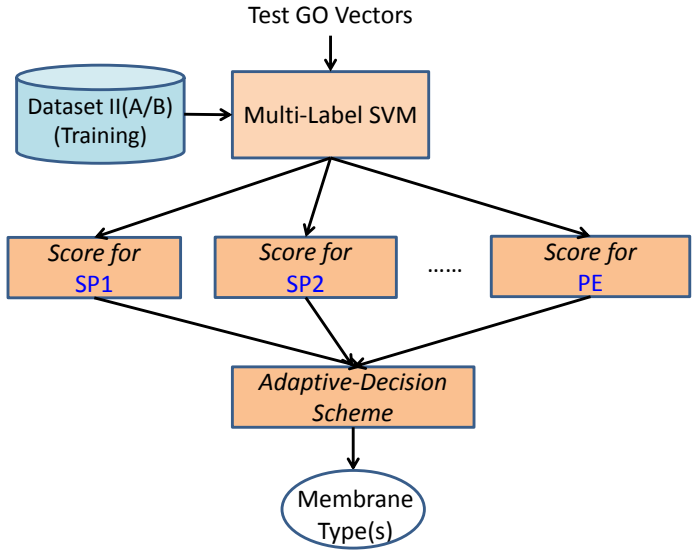
(a) The whole architecture



(b) Feature Extraction



(c) Layer I Classification



(d) Layer II Classification

Figure 1: Flowcharts of Mem-ADSVM, including (a) the whole architecture, (b) feature extraction, (c) Layer I classification and (d) Layer II classification. $\mathbb{Q}_i$: the i-the query protein; *Score*: the SVM score; *SP1*: single-pass type I; *SP2*: single-pass type II; *PE*: peripheral. Cylinders in blue represent databases, and rectangles in orange represent processing/procedures.

are facing. This is because GO-based approaches need to store the mapping between the ACs and their GO terms as a hash map or hash table in memory. Given the large number of ACs in the GOA database, the hash map will easily occupy tens of gigabytes of memory. With the rapid growth in the number of entries in the GOA database, the memory consumption will further increase in the future. The ProSeq-GO can reduce the memory consumption from tens of gigabytes to several hundred megabytes because the number of ACs in ProSeq-GO is substantially smaller than that in the GOA database.

2.3. GO Vectors Construction

The construction of term-frequency based GO vectors involves two steps: (1) GO terms retrieval; and (2) GO vectors construction.

For GO-terms retrieval, given a query protein, its amino acid sequence is presented to BLAST [42] to find its homologs in the ProSeq database created in Section 2.2. The homologous ACs are then used as keys to search against the ProSeq-GO database. We used the default parameter setting for BLAST in our experiments. Given a dataset, the GO terms of all of its proteins are retrieved. Then, the number of distinct GO terms is found. Specifically, let $\mathbb{W}$ denotes a set of distinct GO terms corresponding to a dataset of interest. $\mathbb{W}$ is constructed as follows: (1) identify all of the GO terms in the dataset and (2) remove the repetitive GO terms. Suppose W distinct GO terms are found, i.e., $|\mathbb{W}| = W$; these GO terms form a GO Euclidean space with W dimensions.

For GO-vectors construction, because term-frequency (TF) based GO vectors [44, 43] were found to outperform the conventional 1-0 vectors, we adopted the TF method to construct GO vectors. Specifically, for each protein sequence in the dataset, a GO vector is constructed by matching its GO terms against $\mathbb{W}$ determined in Step 1, using the number of occurrences of individual GO terms in $\mathbb{W}$ as the coordinates. Mathematically, the frequency GO vector $\mathbf{q}_i$ of the i-th protein $\mathbb{Q}_i$ is defined as:

$$\mathbf{q}_i = [f_{i,1}, \ldots, f_{i,j}, \ldots, f_{i,W}]^\mathsf{T}, \tag{1}$$

where $f_{i,j}$ is the number of occurrences of the j-th GO term (term-frequency) in the i-th protein sequence. Detailed information about GO vectors can be found in [43, 44].

3. Two-Layer Classification

Essentially speaking, membrane-protein type prediction is a two-layer classification problem. Specifically, given a query protein, we have to identify whether it is a membrane protein or not. If it is a non-membrane protein, we stop the prediction. This is 'Layer I classification'. If it is a membrane protein, we need to predict to which membrane type(s) it belongs. This is 'Layer II classification'. Thus, Layer I is a binary classification problem, whereas Layer II is a multi-label multi-class classification problem. To address this two-layer classification problem, we used a binary SVM classifier for Layer I and an adaptive-decision multi-label SVM classifier [48] for Layer II, which are elaborated below.

3.1. Layer I: Identifying Membrane Proteins

The Layer I is a typical binary classification problem, i.e., differentiating membrane proteins from non-membrane proteins. The flowchart of Layer I classification is shown in Fig. 1(c). Here, we utilize support vector machines (SVMs) to tackle the classification task.

SVMs, initially proposed by Vapnik [49], have become one of the most popular classification tools in recent years due to their attractive features and promising performance. SVMs are usually defined to map a set of input patterns to a high-dimensional space and then find a hyperplane to obtain the largest possible margin of separation. This is achieved by optimizing the weights via removing the influence of those patterns at a distance from the decision boundary. The obtained hyperplane can classify the patterns into two categories and maximize their distance from the hyperplane. To construct a robust SVM classifier, slack variables are usually incorporated, which, to some extent, can tolerate some data to violate the constraints that define the minimum safety margin. A user-defined penalty parameter (C) is used to control the penalties imposed on those patterns violating the constraints. SVMs can also be generalized to deal with nonlinear classification problems by kernelization. Three commonly used kernels are linear, polynomial and RBF kernels.

Because GO vectors defined in Eq. 1 are of very high dimension (> 13,000), we use a linear SVM with the penalty parameter C set to 0.1.

3.2. Layer II: Predicting Functional Types

Because membrane proteins may belong to multiple functional types, Layer II classification is essentially a multi-label multi-class classification problem. The flowchart of Layer II classification is shown in Fig. 1(d). To tackle the multi-label problem, we use a multi-label SVM classifier equipped with an adaptive-decision scheme, which is elaborated below.

3.2.1. Scoring for Multi-Label SVM

Without loss of generality, suppose we have a dataset of N membrane proteins belonging to M functional types.[5] Denote $\{\boldsymbol{\mathcal{Y}}_i\}_{i=1}^N$, where $\boldsymbol{\mathcal{Y}}_i \subset \{1, 2, \ldots, M\}$ as the label set of this dataset. By using the definitions of *transformed labels* [44], we convert the label set of the i-th protein to a label vector $\mathbf{y}_i = [y_{i,1}, \ldots, y_{i,m}, \ldots, y_{i,M}]^\mathsf{T}$, where $y_{i,m} \in \{-1, +1\}$. Here, we used the GO term-frequency vectors computed in Eq. 1 to train the multi-label one-vs-rest SVMs. Specifically, for an M-class problem (here M is the number of membrane functional types), M independent binary SVMs are trained, one for each type. Then, given the t-th test protein $\mathbb{Q}_t$, the score of the m-th SVM is:

$$s_m(\mathbb{Q}_t) = \sum_{r \in \mathcal{S}_m} \alpha_{r,m} y_{r,m} K(\mathbf{p}_r, \mathbf{q}_t) + b_m, \tag{2}$$

where $\mathcal{S}_m$ is the set of support vector indexes corresponding to the m-th SVM, $\mathbf{p}_r$ is the r-th training vector in the set $\mathcal{S}_m$,

[5] Note that for Layer II, the dataset of interest contains membrane proteins only because Layer I has already identified them as membrane proteins.

$\alpha_{m,r}$ is the r-th Lagrange multiplier for the m-th SVM, $K(\cdot,\cdot)$ is a kernel function, b_m is the bias for the m-th SVM, $y_{m,r} \in \{-1,+1\}$ is the m-th element of the label vector $\mathbf{y}_r$ for the r-th training protein $\mathbb{Q}_r$ in the set $\mathcal{S}_m$. When $y_{m,r} = 1$, $\mathbb{Q}_r$ belongs to the m-th class; and vice versa for $y_{m,r} = -1$. Similar to the reasons stated in Section 3.1, the linear kernel is used here, i.e., $K(\mathbf{p}_r, \mathbf{q}_t) = \langle \mathbf{p}_r, \mathbf{q}_t \rangle$.

3.2.2. Adaptive-Decision Scheme for Multi-Label SVM

Unlike the single-label problem where each protein has one predicted label only, a multi-label protein should have more than one predicted labels. This paper uses an adaptive decision scheme similar to the one in mPLR-Loc [50]. In this scheme, the predicted functional type(s) of the t-th query protein are given by:

$$\mathcal{M}^*(\mathbb{Q}_t) = \begin{cases} \bigcup_{m=1}^{M} \{m : s_m(\mathbb{Q}_t) \geq \min(1.0, f(s_{\max}(\mathbb{Q}_t)))\}, \\ \qquad\qquad \text{where } \exists\ s_m(\mathbb{Q}_t) > 0; \\ \arg\max_{m=1}^{M} s_m(\mathbb{Q}_t), \qquad \text{otherwise}, \end{cases} \tag{3}$$

where $s_m(\mathbb{Q}_t)$ is defined in Eq. 2, $f(s_{\max}(\mathbb{Q}_t))$ is a function of $s_{\max}(\mathbb{Q}_t)$, where $s_{\max}(\mathbb{Q}_t) = \max_{m=1}^{M} s_m(\mathbb{Q}_t)$. Here, a linear function was used, i.e. ,

$$f(s_{\max}(\mathbb{Q}_t)) = \theta s_{\max}(\mathbb{Q}_t), \tag{4}$$

where $\theta \in [0.0, 1.0]$ is a parameter that can be determined by using cross-validation experiments. Because $f(s_{\max}(\mathbb{Q}_t))$ is linear, Eq. 3 turns the linear SVMs into piecewise linear SVMs (See reasons in [40, 51]). Eq. 3 also suggests that the predicted labels depend on $s_{\max}(\mathbb{Q}_t)$, a function of the test instance (or protein). This means that the decision and the corresponding threshold are adaptive to the test protein.

When $\theta = 0$, Eq. 3 becomes the conventional binary-relevance based decision which uses a fixed threshold to determine the predicted class [44]; on the contrary, when $\theta = 1$, Eq. 3 becomes a single-label multi-class classification decision scheme. In particular, $s_{\max}(\mathbb{Q}_t)$ in Eq. 3 adaptively normalizes the scores of all one-vs-rest SVMs so that for SVMs to be considered as runner-ups, they need to have a sufficiently large score relative to the winner. This strategy effectively reduces the chance of over-prediction. The condition $s_m(\mathbb{Q}_i) > 1$ in Eq. 3 aims to avoid under-prediction when the SVM with the maximum score has very high confidence (i.e., $s_{\max}(\mathbb{Q}_t) \gg 1$) but the runners-up still have enough confidence ($s_m(\mathbb{Q}_t) > 1$) in making a right decision.[6] On the other hand, when the maximum score is small (say $0 < s_{\max}(\mathbb{Q}_t) \leq 1$), $f(s_{\max}(\mathbb{Q}_t))$ in Eq. 3 can strike a balance between over-prediction and under-prediction. When all of the SVMs have very low confidence (say $s_{\max}(\mathbb{Q}_t) < 0$), the classifier switches to single-label mode.

For ease of presentation, we refer to the proposed predictor as Mem-ADSVM.

[6] SVM scores larger than one means that the test proteins fall beyond the margin of separation; therefore, the confidence is fairly high.

4. Datasets and Performance Metrics

Four datasets were used to evaluate the performance of Mem-ADSVM. For identifying membrane proteins, a benchmark dataset [7] containing both membrane proteins and non-membrane proteins was used; for predicting membrane functional types, two benchmark datasets [14, 15] containing both single- and multi-functional-type proteins were used; to evaluate the two-layer architecture, a dataset containing non-membrane, single- and multi-functional-type proteins was created (see below). For ease of representation, we refer to the dataset for Layer I as Dataset I, the two datasets for Layer II as Dataset II(A) and Dataset II(B), and the dataset for evaluating two-layer architecture as Dataset II(C). Details of these four datasets are shown in Table 2 and Fig 2.

Datasets I [7], II(A) [15] and II(B) [14] were extracted from Swiss-Prot released in October 2006, March 2013 and June 2012, respectively. In Dataset I, there are 15,547 single-label proteins, of which 7,582 and 7,965 are membrane and non-membrane proteins, respectively. In Dataset II(A), there are 5,502 virtual proteins [15] corresponding to 5,307 actual proteins, of which 5,117 belong to one type, 185 to two types and 5 to three types. In Dataset II, there are 14,016 virtual proteins corresponding to 13,659 actual proteins, of which 13,313 belong to one type, 335 to two types and 11 to three types. *Virtual proteins* are defined as follows: If a protein belongs to two different functional types, then it will be counted as two virtual proteins; if a protein belongs to three types, then it will be counted as three virtual proteins; and so forth. As can be seen from Fig. 2, the majority (70%/74%) of membrane proteins in both datasets II(A) and II(B) belong to multi-pass type and peripheral type, while proteins in other 6 types totally account for no more than 30% in these two datasets. This means that both datasets are very imbalanced. The sequence identities of Datasets I, II(A) and II(B) were cut off at 80%, 25% and 80%, respectively.

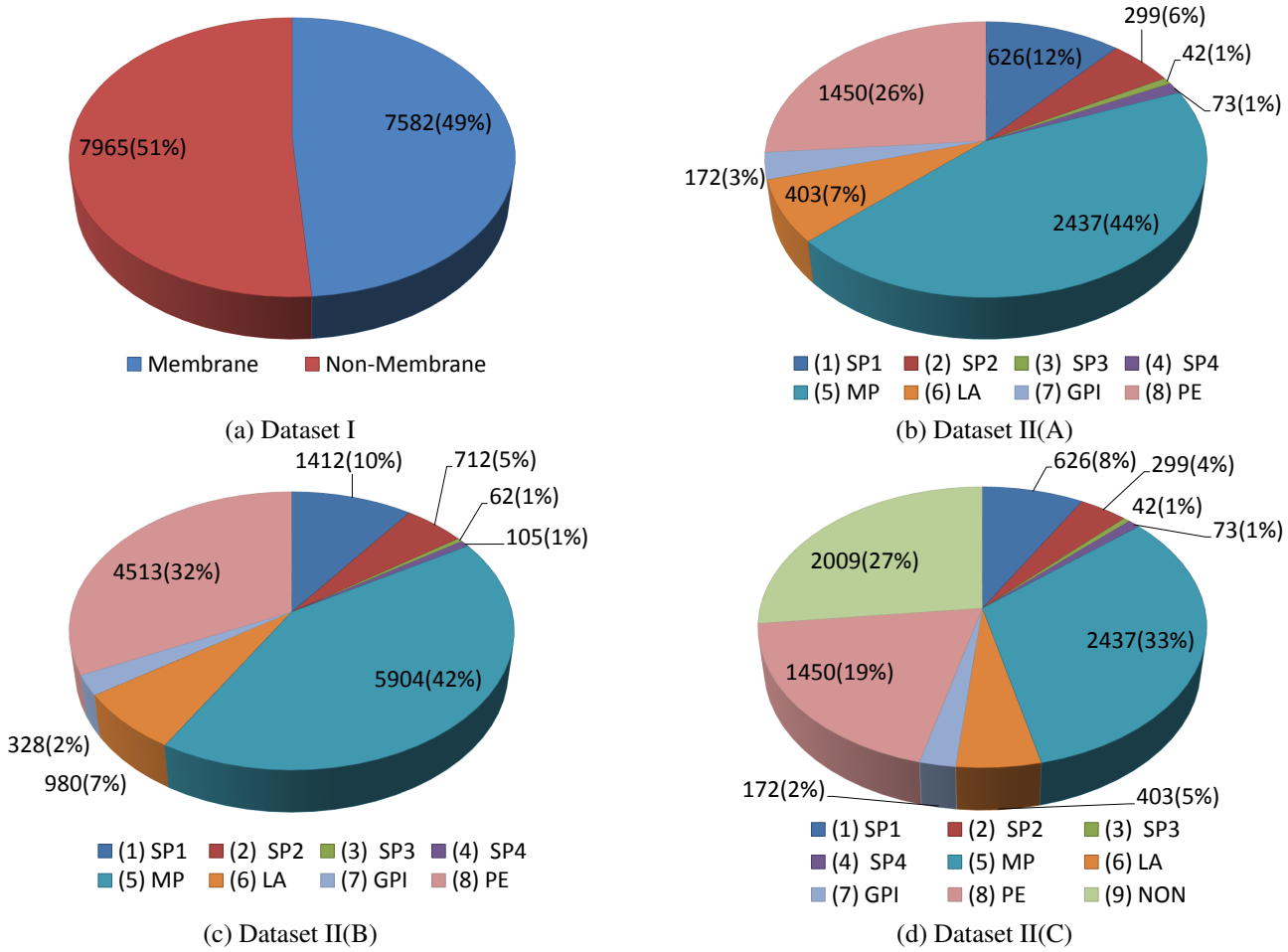
Dataset II(C) was created based on Dataset I and Dataset II(A). First, we collected all of the non-membrane 7,965 proteins in Dataset I. Because the sequence identity of Dataset I was much higher than that of Dataset II(A), we used the BLASTCLUST[7] to further reduce the sequence similarity to 25%, leading to 2,009 non-membrane proteins selected. Then, we combined these 2,009 non-membrane proteins with Dataset II(A) (5,307 membrane proteins) to constitute Dataset II(C) with a total of 7,316 proteins, of which 7,126 belong to one type, 185 to two types and 5 to three types. The breakdown of Dataset II(C) is shown in Fig. 2(d). More information about these datasets can be found in [52] as well as the Mem-ADSVM server.

Because Layer I is a binary classification problem, we used some common performance measures, such as *Accuracy*, *F1* and *MCC*. Because Layer II is a multi-label multi-class classification problem, compared to single-label classification, more sophisticated performance metrics were used. Specifically, we

[7] http://www.ncbi.nlm.nih.gov/Web/Newsltr/Spring04/blastlab.html

Table 2: Summary of Datasets for evaluating the performance of Mem-ADSVM. *All-membrane*: all of the proteins in the dataset of interest are membrane proteins; *multi-label*: some (or all) of the proteins in the dataset of interest are multi-label membrane proteins; *Similarity cutoff*: sequence similarity cutoff threshold; *No. of actual proteins*: number of total proteins; *No. of virtual proteins*: number of total labels to which total proteins belong.

Stratification	Layer I	Layer II		Both Layers
Datasets	Dataset I	Dataset II(A)	Dataset II(B)	Dataset II(C)
All-membrane	No	Yes	Yes	No
Multi-label	No	Yes	Yes	Yes
Similarity Cutoff	80%	25%	80%	25%
No. of classes	2	8	8	9
No. of actual proteins	15,547	5,307	13,659	7,316
No. of virtual proteins	15,547	5,502	14,016	7,511
Distribution	Fig. 2(a)	Fig. 2(b)	Fig. 2(c)	Fig. 2(d)
Source	[7]	[15]	[14]	This paper



(a) Dataset I

(b) Dataset II(A)

(c) Dataset II(B)

(d) Dataset II(C)

Figure 2: Breakdown of datasets, including (a) Dataset I, (b) Dataset II(A), (c) Dataset II(B) and (d) Dataset II(C). *SP1*: single-pass type I; *SP2*: single-pass type II; *SP3*: single-pass type III; *SP4*: single-pass type IV; *MP*: multi-pass; *LA*: lipid-anchor; *GPI*: GPI-anchor; *PE*: peripheral; *NON*: non-membrane.

used the popular multi-label evaluation metrics [14, 15, 53, 54], including *Hamming loss, Ranking loss, One-error, Coverage, Average precisions, Accuracy, Precision, Recall* and *Absolute true*. For the first four metrics, the smaller the better, and for the remaining metrics, the larger the better. Among these performance metrics, *Absolute true* (equivalent to *overall actual accuracy* in [50]) is the most objective and stringent [50]. The definitions of these metrics can be found in supplementary materials on the Mem-ADSVM server.

5. Results and Analysis

5.1. Two-Layer vs One-Layer

Table 3: Comparing the Two-Layer architecture (Fig. 1(a)) against the One-Layer architecture (Fig. 1(d)) based on leave-one-out cross-validation tests on Dataset II(C). *One-Layer* denotes treating the non-membrane proteins as the 9-th functional type, whereas *Two-Layer* first identifies membrane/non-membrane proteins and then predicts membrane types. ↓ means the lower the better; ↑ denotes the higher the better.

Evaluation Criteria	Predictors	
	One-Layer	Two-Layer
Hamming loss ↓	0.0451	**0.0342**
Ranking loss ↓	0.0466	**0.0357**
One-error ↓	0.1450	**0.1420**
Coverage ↓	0.3229	**0.3010**
Average precision ↑	0.8956	**0.9095**
Accuracy ↑	0.8265	**0.8487**
Precision ↑	0.8309	**0.8554**
Recall ↑	0.8483	**0.8598**
Absolute-true ↑	0.8009	**0.8311**

To investigate the advantages of using the two-layer prediction architecture, we have compared the performance of using the two-layer architecture (Two-Layer) with that of using the one-layer architecture (One-Layer) based on leave-one-out cross-validation (LOOCV) on Dataset II(C), which is shown in Table 3. One-Layer treats the non-membrane proteins as another 'functional type', thus making the original eight-type multi-label membrane type classification problem into a nine-type multi-label classification task. On the contrary, Two-Layer first identifies membrane/non-membrane proteins by a binary classifier and then predicts membrane types by an eight-type multi-label classifier. We used the same GO features and the same adaptive decision scheme for both cases.

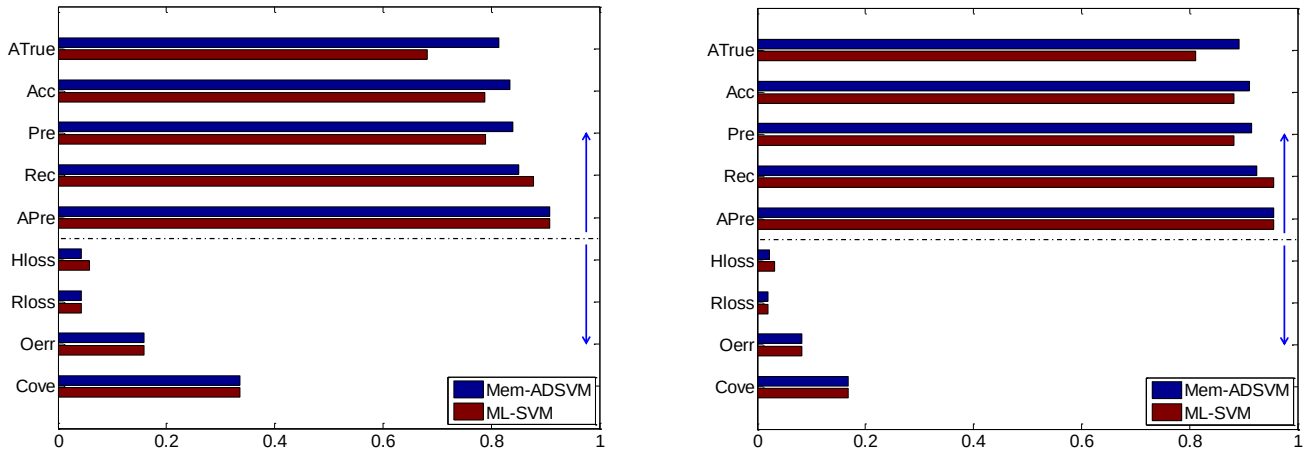
As can be seen from Table 3, Two-Layer significantly outperforms One-Layer in terms of all of the performance metrics. In particular, for the most objective and stringent criterion *Absolute-true*, the performance of Two-Layer is more than 3% (absolute) better than that of One-Layer. These results suggest that using the two-layer is more conducive to identifying membrane proteins and their multi-functional types. This is understandable because using the two-layer prediction architecture is equivalent to making full use of the inter-class relationships between membrane and non-membrane proteins. In the one-layer prediction architecture, the non-membrane type is treated equally important as other membrane types; in other words, a protein may be predicted as belonging to the non-membrane type and one (or more) of the eight membrane types simultaneously, which is logically problematic. On the contrary, the two-layer architecture explicitly removes this possibility. Besides, the high performance in Layer I dismisses the concerns that the prediction performance will be dramatically affected by Layer I. In our experiments, in the 7,316 proteins (5,307 membrane and 2,009 non-membrane) of Dataset II(C), the accuracy of Layer I is 97.2% (=7112/7316), and only 125 and 79 are false positives ('non-membrane' predicted as 'membrane') and false negatives ('membrane' predicted as 'non-membrane').

5.2. Adaptive-Decision vs Fixed-Decision

Fig. 3 compares the performance of using the adaptive-decision scheme (Mem-ADSVM) with that of using the fixed-decision scheme, i.e., multi-label SVM (ML-SVM) [44] on (a) Dataset II(A) and (b) Dataset II(B), respectively. In Fig. 3(a), the performance is based on leave-one-out cross-validation (LOOCV) tests, whereas in Fig. 3(b), the performance is based on five 5-fold cross-validation tests.[8] For a fair comparison, we used the same features obtained in Section 2 for both Mem-ADSVM and ML-SVM. The blue arrows denote that for the performance metrics above the black dotted line, the larger the better, whereas for those below the black dotted line, the smaller the better.

As can be seen from Fig. 3(a), Mem-ADSVM performs impressively better than ML-SVM in terms of *Absolute true, Accuracy* and *Precision*, whereas the performance of the former is a bit inferior to that of the latter in terms of *Recall*. Particularly, for the most objective and stringent criteria *Absolute true*, Mem-ADSVM outperforms ML-SVM by more than 10%, whereas for *Recall*, the latter performs around 3% better than the former. This is understandable because in traditional fixed-threshold decision scheme, e.g, ML-SVM, the number of over-predictions is significantly more than that of under-predictions. This problem can be overcome by the adaptive-decision scheme because it focuses on reducing false-positives (or over-predictions), leading to a significant improvement on *Absolute true*. However, the adaptive decision scheme may cause few cases of under-prediction which may slightly reduce the *Recall*. For *Hamming loss*, Mem-ADSVM also significantly outperforms ML-SVM, which is consistent with the aforementioned analysis that the proposed adaptive-decision scheme can reduce the prediction errors. Interestingly, ML-SVM achieves the same results as Mem-ADSVM for other performance measures, such as *Average precision, Ranking loss, One-error, Coverage*. This is because these performance measures are for multi-label ranking tasks [53], which are more sensitive to the scores than to the decision-making schemes. As Mem-ADSVM and ML-SVM produce the same one-vs-rest

[8] Because the standard deviation is inconsiderable compared to the mean, we only show the mean performance on Fig. 3(b)



(a) Dataset II(A) (b) Dataset II(B)

Figure 3: Comparing the performance of using the adaptive-decision scheme (Mem-ADSVM) with that of using the fixed-decision scheme (ML-SVM) on (a) Dataset II(A) and (b) Dataset II(B). In (a), the performance is based on leave-one-out cross-validation (LOOCV) tests, whereas in (b), the performance is based on the average of five 5-fold cross-validation tests. *ATrue*: Absolute true; *Acc*: Accuracy; *Pre*: Precision; *Rec*: Recall; *APre*: Average Precision; *Hloss*: Hamming loss; *Rloss*: Ranking loss; *Oerr*: One-error; *Cove*: Coverage. The blue arrows denote that for the performance metrics above the black dotted line, the larger the better, whereas for those below the black dotted line, the smaller the better.

SVM scores, both predictors produce the same results in terms of these measures.

Similar conclusions can also be drawn from Fig. 3(b) except that the superiority of Mem-ADSVM over ML-SVM is less apparent. This is possibly because the sequence identity in Dataset II(B) is much higher than that in Dataset II(A) (80% vs. 25%), which increases the bias on the prediction performance.

5.3. Layer I: Comparing with State-of-the-Art Predictors

Table 4 compares Mem-ADSVM with existing state-of-the-art predictors for identifying membrane proteins (Layer I) based on leave-one-out cross-validation (LOOCV) tests on Dataset I. To the best of our knowledge, there are three predictors, namely LeastEudist [12], ProtLoc [13] and MemType-2L [7], which are able to determine whether a query protein is a membrane protein or not. LeastEudist uses the conventional amino acid compositions as features and adopts least Euclidean distance for classification. ProtLoc also uses the amino acid compositions as features but uses least Mahalanobis distance as classification. MemType-2L uses pseudo position-specific score matrices (PSSM) as features and an ensemble optimized evidence-theoretic K-nearest neighbor (OET-KNN) algorithm for classification. Our proposed Mem-ADSVM uses the term-frequency based GO information as features and a binary SVM as the classifier for Layer I classification.

As shown in Table 4, Mem-ADSVM performs impressively better than the three state-of-the-art predictors in terms of all the performance metrics. Specifically, the *Accuracy, F1* and MCC of Mem-ADSVM are around 20% (absolute), 20% (absolute) and 37% (absolute), respectively, higher than those of LeastEudist and ProtLoc, and 5% (absolute), 5% (absolute) and 10% (absolute), respectively, higher than those of MemType-2L. The superiority of Mem-ADSVM over these three predictors is also observed in terms of individual accuracies (i.e., membrane/non-membrane). The results suggest that Mem-ADSVM is more advanced than existing predictors in terms of identifying membrane proteins.

5.4. Layer II: Comparing with State-of-the-Art Predictors

Table 5 and Table 6 compare Mem-ADSVM with several state-of-the-art multi-label predictors on Datasets II(A) and II(B), respectively. As far as we know, there are three predictors, namely Mem-PseAA [14], iMem-Seq [15] and Mem-mEN [8], capable of predicting membrane proteins with both single- and multi-label functional types. In terms of constructing feature vectors, iMem-Seq uses position-specific score matrices, Mem-PseAA uses pseudo-amino acid compositions and Mem-mEN uses term-frequency based GO information. In terms of multi-label classification, the former two predictors use a multi-label kNN classifier to deal with the multi-label classification problem, whereas the latter uses a multi-label elastic net classifier. Particularly, Mem-mEN possesses the 'interpretable' property so that its prediction results are interpretable based on the selected GO terms. Our proposed Mem-ADSVM is a two-layer classifier that can identify membrane proteins and their multi-functional types. Similar to Mem-mEN, Mem-ADSVM extracts homologous GO frequency information from ProSeq and ProSeq-GO databases; however, in Layer II, unlike Mem-mEN, Mem-ADSVM uses an adaptive-decision based multi-label SVM classifier to deal with the multi-label classification problem.

The results in Table 5 were based on LOOCV tests on Dataset II(A), where the sequence identity was cut off at 25%. For the measures of *Hamming loss, Ranking loss, One-error and Coverage*, the smaller the better; while for the measures of *Average precision, Accuracy, Precision, Recall and Absolute-true*, the larger the better. As shown in Table 5, Mem-ADSVM significantly outperforms iMem-Seq and Mem-mEN in terms of all performance metrics. Specifically, the *Average precision* of Mem-ADSVM is more than 7% (absolute) and 2% (absolute)

Table 4: Comparing Mem-ADSVM with state-of-the-art predictors based on leave-one-out cross-validation (LOOCV) tests on Dataset I.

Class	Predictors			
	LeastEudist [12]	ProtLoc [13]	MemType-2L [7]	Mem-ADSVM
Membrane	$\frac{5320}{7582} = 0.7017$	$\frac{5512}{7582} = 0.7270$	$\frac{6897}{7582} = 0.9097$	$\mathbf{\frac{7471}{7582} = 0.9854}$
Non-membrane	$\frac{6688}{7965} = 0.8397$	$\frac{6754}{7965} = 0.8480$	$\frac{7520}{7965} = 0.9441$	$\mathbf{\frac{7761}{7965} = 0.9744}$
Accuracy	0.7724	0.7890	0.9273	**0.9797**
F1	0.7504	0.7706	0.9243	**0.9794**
MCC	0.5475	0.5801	0.8548	**0.9595**

Table 5: Comparing Mem-ADSVM with a state-of-the-art predictor based on leave-one-out cross-validation tests on Dataset II(A). ↓ means the lower the better; ↑ denotes the higher the better.

Evaluation Criteria	Predictors		
	iMem-Seq [15]	Mem-mEN [8]	Mem-ADSVM
Hamming loss ↓	0.0635	0.0493	**0.0423**
Ranking loss ↓	0.0902	0.0521	**0.0424**
One-error ↓	0.2572	0.1892	**0.1594**
Coverage ↓	0.6735	0.4046	**0.3362**
Average precision ↑	0.8335	0.8881	**0.9086**
Accuracy ↑	0.6804	0.8056	**0.8347**
Precision ↑	0.6825	0.8085	**0.8403**
Recall ↑	0.6813	0.8135	**0.8504**
Absolute-true ↑	0.6774	0.7948	**0.8138**

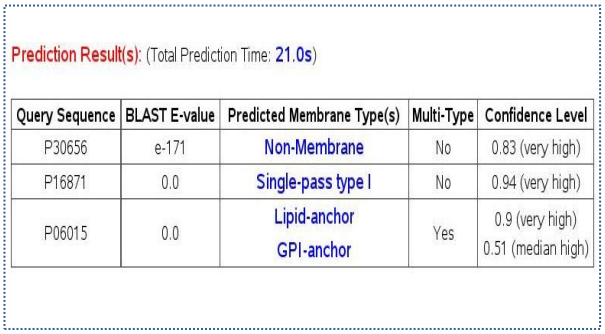
Table 6: Comparing Mem-ADSVM with state-of-the-art predictors based on the average of five 5-fold cross-validation tests on Dataset II(B). ↓ means the lower the better; ↑ denotes the higher the better. Because [14] and [15] do not report the *Precision, Recall* and *Absolute-True* on Dataset II(B), for consistency, we do not report these results.

Evaluation Criteria	Predictors			
	Mem-PseAA [14]	iMem-Seq [15]	Mem-mEN [8]	Mem-ADSVM
Hamming loss ↓	0.0495±0.0019	0.0317±0.0013	0.0303±0.0008	**0.0228±0.0003**
Ranking loss ↓	0.0600±0.0025	0.0425±0.0008	0.0339±0.0006	**0.0202±0.0004**
One-error ↓	0.1964±0.0033	0.1192±0.0011	0.1154±0.0013	**0.0820±0.0012**
Coverage ↓	0.4470±0.0215	0.3266±0.0031	0.2636±0.0029	**0.1684±0.0026**
Average precision ↑	0.8780±0.0025	0.9211±0.0007	0.9225±0.0009	**0.9557±0.0008**

higher than that of iMem-Seq and Mem-mEN, respectively. In terms of both *Accuracy* and *Precision*, Mem-ADSVM outperforms iMem-Seq and Mem-mEN by more than 15% (absolute) and 3% (absolute), respectively. Besides, the *Recall* of the former is 16% (absolute) and 3% (absolute) better than the latter two predictors, respectively. In particular, for the most stringent and object criteria *absolute-true*, Mem-ADSVM outpaces iMem-Seq and Mem-mEN by more than 13% (absolute) and 2% (absolute), respectively.

Similar conclusions can be drawn for Dataset II(B) in Table 6. The results in Table 6 were based on the average of five 5-fold cross-validation tests on Dataset II(B), where the sequence identity was cut off at 80%. As can be seen, in terms of the first four metrics, Mem-ADSVM performs better than Mem-PseAA, iMem-Seq and Mem-mEN, and the *average precision* of the former is higher than that of the latter three. When comparing Table 5 and Table 6, we notice that the superiority of Mem-ADSVM over iMem-Seq and Mem-PseAA on Dataset II(B) is less apparent than that on Dataset II(A). Similar to the reason explained in Section 5.2, this is caused by the difference in sequence identity, i.e., high sequence identity increases the bias on the prediction performance. Therefore, the performance comparison in Table 5 is more trustworthy than that in Table 6. Interestingly, the difference of the sequence similarity cutoff does not compromise the advantages of Mem-ADSVM. This is possibly because the superiority of the adaptive-decision based multi-label SVM classifier (Mem-ADSVM) is robust regardless of the sequence identities in the datasets.

(a) Input to the Mem-ADSVM server



Query Sequence	BLAST E-value	Predicted Membrane Type(s)	Multi-Type	Confidence Level
P30656	e-171	Non-Membrane	No	0.83 (very high)
P16871	0.0	Single-pass type I	No	0.94 (very high)
P06015	0.0	Lipid-anchor GPI-anchor	Yes	0.9 (very high) 0.51 (median high)

(b) Prediction results

Figure 4: An example output of Mem-ADSVM when confidence scores (Eq. 5) are used in the decision-making process (Eq. 6). (a) Using three protein sequences as input to the Mem-ADSVM server; (b) Prediction results with confidence levels for the proteins in (a). See our web-server Mem-ADSVM for more details.

6. Discussion

In bioinformatics, a classifier will be more useful if every decision made by the classifier also comes with a confidence score (or posterior probability). Confidence scores, which reflect the confidence of prediction decisions, can help users to decide whether to accept the decisions or not. In multi-label bioinformatics classification, confidence scores have an additional purpose in that they help to decide the number of classes of a query protein.

Standard SVMs or adaptive-decision based SVMs proposed in Section 3.2.2 can only produce uncalibrated, non-probabilistic output scores. One possible way to solve this problem is to convert the SVM output scores into calibrated posterior probabilities using a sigmoid function [55]. This idea can be extended to multi-label, multi-class classification as follows. Given a query protein $\mathbb{Q}_t$, the calibrated probabilistic score $p_m(\mathbb{Q}_t)$ for the m-th class is defined as:

$$p_m(\mathbb{Q}_t) = \frac{1}{1 + e^{(A_m s_m(\mathbb{Q}_t) + B_m)}}, \tag{5}$$

where $m \in \{1, 2\}$ for Layer I (i.e., either membrane or non-membrane) and $m \in \{1, \ldots, 8\}$ for Layer II (i.e., any one of the eight membrane types), A_m and B_m are the parameters for the m-th class, which can be trained via cross validation, and $s_m(\mathbb{Q}_t)$ is the uncalibrated SVM score of the query protein $\mathbb{Q}_t$ for the m-th class. For ease of understanding, when $\mathbb{Q}_t$ is predicted to be a non-membrane protein, we use the $p_m(\mathbb{Q}_t)$ in Layer I as the final confidence score; when $\mathbb{Q}_t$ is predicted to be a membrane protein (either single-type or multi-type), we use the $p_m(\mathbb{Q}_t)$ in Layer II as the final confidence score.[9]

Motivated by our previous study [50], to make the confidence scores easier to understand, we further divide the confidence into four levels:

$$\mathbb{C} = \begin{cases} \text{very high} & \text{if } 0.8 \le p_m(\mathbb{Q}_t) \le 1.0, \\ \text{median high} & \text{if } 0.5 \le p_m(\mathbb{Q}_t) < 0.8, \\ \text{median low} & \text{if } 0.2 \le p_m(\mathbb{Q}_t) < 0.5, \\ \text{very low} & \text{if } 0 \le p_m(\mathbb{Q}_t) < 0.2. \end{cases} \tag{6}$$

In other words, if $p_m(\mathbb{Q}_t) \ge 0.8$, the confidence of the decision is very high; on the contrary, if $p_m(\mathbb{Q}_t) < 0.2$, then the confidence is very low, meaning that the decision may be wrong.

Fig. 4 shows an example output of Mem-ADSVM when confidence scores (Eq. 5) are used in the decision-making process (Eq. 6). In Fig. 4(a), three protein sequences were used as the input to the Mem-ADSVM server; and Fig. 4(b) shows the prediction results with confidence levels for the query proteins. As can be seen, three cases are presented: non-membrane, single-type membrane and multi-type membrane. The first (P30656) and the second (P16871) proteins are predicted with confidence

[9] When $\mathbb{Q}_t$ is predicted to belong to multiple membrane types, the confidence scores will be multiple, one for each predicted membrane type.

scores 0.83 and 0.94, respectively. According to Eq. 6, the confidence levels for these two predictions are *"very high"*. On the contrary, the third (P06015) protein is predicted to belong to two membrane types: lipid-anchor and GPI-anchor with confidence scores of 0.9 and 0.51, respectively. Thus, the former predicted type is of *"very high"* confidence and the latter one is of *"median high"* confidence. This is understandable because the GPI-anchor is a special lipid-anchor type, which is more difficult to predict.

7. Conclusion

This paper proposes an efficient two-layer predictor, namely Mem-ADSVM, to identify membrane proteins and their functional types. Given a query protein, its feature information is extracted from the frequencies of occurrences of its associated GO terms retrieved from the ProSeq-GO database. Based on its GO information, Mem-ADSVM first determines whether the query protein is a membrane protein or not by a binary SVM classifier. If yes, by utilizing an adaptive-decision scheme in a multi-label SVM classifier, Mem-ADSVM subsequently identifies to which functional type(s) the query protein belongs.

Experimental results show that Mem-ADSVM performs impressively better than state-of-the-art membrane-protein predictors in terms of identifying both membrane proteins and their types. Besides, this paper also found that the proposed adaptive-decision scheme is conducive to improving the prediction performance of Mem-ADSVM. This paper also found that the two-layer prediction architecture performs better than one-layer. For readers' convenience, Mem-ADSVM is freely accessible at `http://bioinfo.eie.polyu.edu.hk/MemADSVMServer/`.

Acknowledgment

This work was in part supported by the RGC of Hong Kong SAR Grant No. PolyU152117/14E and PolyU152068/15E.

References

[1] A. Andreeva, D. Howorth, C. Chothia, E. Kulesha, A. G. Murzin, SCOP2 prototype: a new approach to protein structure mining, Nucleic Acids Research 42 (D1) (2014) D310–D314.

[2] M. S. Almén, K. J. V. Nordström, R. Fredriksson, H. B. Schiöth, Mapping the human membrane proteome: a majority of the human membrane proteins can be classified according to function and evolutionary origin, BMC Biology 7 (1) (2009) 50.

[3] T. M. Bakheet, A. J. Doig, Properties and identification of human protein drug targets, Bioinformatics 25 (4) (2009) 451–457.

[4] J. P. Overington, B. Al-Lazikani, A. L. Hopkins, How many drug targets are there?, Nature Reviews Drug Discovery 5 (12) (2006) 993–996.

[5] H. F. Lodish, A. Berk, S. L. Zipursky, P. Matsudaira, D. Baltimore, J. Darnell, Molecular cell biology, 4th Edition, W. H. Freeman New York, 2000.

[6] K. Gerald, Cell and molecular biology: concepts and experiments, 7th Edition, John Wiley and Sons, Hoboken, NJ, 2013.

[7] K. C. Chou, H. B. Shen, MemType-2L: a web server for predicting membrane proteins and their types by incorporating evolution information through Pse-PSSM, Biochemical and Biophysical Research Communications 360 (2) (2007) 339–345.

[8] S. Wan, M. W. Mak, S. Y. Kung, Mem-mEN: Predicting multi-functional types of membrane proteins by interpretable elastic nets, IEEE/ACM Transactions on Computational Biology and Bioinformatics to appear. doi:10.1109/TCBB.2015.2474407.

[9] H. Ikezawa, Glycosylphosphatidylinositol (GPI)-anchored proteins, Biological and Pharmaceutical Bulletin 25 (4) (2002) 409–417.

[10] P. S. Tappia, N. S. Dhalla, Phospholipases in health and disease, 1st Edition, Springer, 2014.

[11] L. Nanni, A. Lumini, An ensemble of support vector machines for predicting the membrane protein type directly from the amino acid sequence, Amino Acids 35 (3) (2008) 573–580.

[12] H. Nakashima, K. Nishikawa, T. OOI, The folding type of a protein is relevant to the amino acid composition, Journal of Biochemistry 99 (1) (1986) 153–162.

[13] J. Cedano, P. Aloy, J. A. Perez-Pons, E. Querol, Relation between amino acid composition and cellular location of proteins, J. Mol. Biol. 266 (1997) 594–600.

[14] C. Huang, J. Q. Yuan, A multilabel model based on Chou's pseudo–amino acid composition for identifying membrane proteins with both single and multiple functional types, The Journal of Membrane Biology 246 (4) (2013) 327–334.

[15] X. Xiao, H. L. Zou, W. Z. Lin, iMem-Seq: A multi-label learning classifier for predicting membrane proteins types, The Journal of Membrane Biology (2015) 745–752.

[16] K. C. Chou, Y. D. Cai, Using GO-PseAA predictor to identify membrane proteins and their types, Biochemical and Biophysical Research Communications 327 (3) (2005) 845–847.

[17] Y. D. Cai, G. P. Zhou, K. C. Chou, Support vector machines for predicting membrane protein types by using functional domain composition, Biophysical Journal 84 (5) (2003) 3257–3263.

[18] H. L. Zou, A multi-label classifier for prediction membrane protein functional types in animal, The Journal of Membrane Biology 247 (11) (2014) 1141–1148.

[19] M. Hayat, A. Khan, M. Yeasin, Prediction of membrane proteins using split amino acid and ensemble classification, Amino acids 42 (6) (2012) 2447–2460.

[20] C. Ding, L. F. Yuan, S. H. Guo, H. Lin, W. Chen, Identification of mycobacterial membrane proteins and their types using over-represented tripeptide compositions, Journal of Proteomics 77 (2012) 321–328.

[21] T. Wang, T. Xia, X. M. Hu, Geometry preserving projections algorithm for predicting membrane protein types, Journal of Theoretical Biology 262 (2) (2010) 208–213.

[22] V. Tripathi, D. K. Gupta, Discriminating lysosomal membrane protein types using dynamic neural network, Journal of Biomolecular Structure and Dynamics 32 (10) (2014) 1575–1582.

[23] Z. Yuan, R. D. Teasdale, Prediction of Golgi Type II membrane proteins based on their transmembrane domains, Bioinformatics 18 (8) (2002) 1109–1115.

[24] R. Clemente, C. Juan, Cell entry of Borna disease virus follows a clathrin-mediated endocytosis pathway that requires Rab5 and microtubules, Journal of Virology 83 (20) (2009) 10406–10416.

[25] T. W. Vahlenkamp, A. Konrath, M. Weber, H. Müller, Persistence of Borna disease virus in naturally infected sheep, Journal of Virology 76 (19) (2002) 9735–9743.

[26] S. Wan, M. W. Mak, S. Y. Kung, R3P-Loc: A compact multi-label predictor using ridge regression and random projection for protein subcellular localization, Journal of Theoretical Biology 360 (2014) 34–45.

[27] The Gene Ontology Consortium, The Gene Ontology project in 2008, Nucleic Acids Research 36 (2008) D440–D444.

[28] X. Wu, L. Zhu, J. Guo, D. Y. Zhang, K. Lin, Prediction of yeast protein-protein interaction network: insights from the gene ontology and annotations, Nucleic Acids Res. 34 (2006) 2137–2150.

[29] X. Guo, R. Liu, C. D. Shriver, H. Hu, M. N. Liebman, Assessing semantic similarity measures for the characterization of human regulatory pathways, Bioinformatics 22 (2006) 967–973.

[30] T. Xu, L. Du, Y. Zhou, Evaluation of GO-based functional similarity measures using S. cerevisiae protein interaction and expression profile data, BMC Bioinformatics 9 (2008) 472.

[31] D. Yang, Y. Li, H. Xiao, Q. Liu, M. Zhang, J. Zhu, W. Ma, C. Yao, J. Wang, D. Wang, Z. Guo, B. Yang, Gaining confidence in biological interpretation of the microarray data: the functional consistence of the

significant GO categories, Bioinformatics 24 (2) (2008) 265–271.
[32] M. Zhu, L. Gao, Z. Guo, Y. Li, D. Wang, J. Wang, C. Wang, Globally predicting protein functions based on co-expressed protein-protein interaction networks and ontology taxonomy similarities, Gene 391 (1-2) (2007) 113–119.
[33] C. Pesquita, D. Faria, A. O. Falcao, P. Lord, F. M. Counto, Metrics for GO based protein semantic similarity: a systematic evaluation, BMC Bioinformatics 9 (Suppl 5) (2008) S4.
[34] Z. Lei, Y. Dai, Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction, BMC Bioinformatics 7 (2006) 491.
[35] K. C. Chou, Z. C. Wu, X. Xiao, iLoc-Euk: A multi-label classifier for predicting the subcellular localization of singleplex and multiplex eukaryotic proteins, PLoS ONE 6 (3) (2011) e18258.
[36] S. Wan, M. W. Mak, S. Y. Kung, Semantic similarity over gene ontology for multi-label protein subcellular localization, Engineering 5 (2013) 68–72.
[37] S. Mei, Multi-label multi-kernel transfer learning for human protein subcellular localization, PLoS ONE 7 (6) (2012) e37716.
[38] S. Wan, M. W. Mak, S. Y. Kung, mLASSO-Hum: A LASSO-based interpretable human-protein subcellular localization predictor, Journal of Theoretical Biology 382 (2015) 223–234.
[39] S. Wan, M. W. Mak, S. Y. Kung, HybridGO-Loc: Mining hybrid features on gene ontology for predicting subcellular localization of multi-location proteins, PLoS ONE 9 (3) (2014) e89545.
[40] S. Wan, M. W. Mak, Machine learning for protein subcellular localization prediction, De Gruyter, 2015.
[41] S. Wan, M. W. Mak, S. Y. Kung, Sparse regressions for predicting and interpreting subcellular localization of multi-label proteins, BMC Bioinformatics 17 (97). doi:10.1186/s12859-016-0940-x.
[42] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, D. J. Lipman, Gapped BLAST and PSI-BLAST: A new generation of protein database search programs, Nucleic Acids Res. 25 (1997) 3389–3402.
[43] S. Wan, M. W. Mak, S. Y. Kung, GOASVM: A subcellular location predictor by incorporating term-frequency gene ontology into the general form of Chou's pseudo-amino acid composition, Journal of Theoretical Biology 323 (2013) 40–48.
[44] S. Wan, M. W. Mak, S. Y. Kung, mGOASVM: Multi-label protein subcellular localization based on gene ontology and support vector machines, BMC Bioinformatics 13 (290). doi:10.1186/1471-2105-13-290.
[45] K. C. Chou, Prediction of protein cellular attributes using pseudo amino acid composition, Proteins: Structure, Function, and Genetics 43 (2001) 246–255.
[46] S. Wan, M. W. Mak, S. Y. Kung, Transductive learning for multi-label protein subchloroplast localization prediction, IEEE/ACM Transactions on Computational Biology and Bioinformatics to appear. doi:10.1109/TCBB.2016.2527657.
[47] K. Nakai, Protein sorting signals and prediction of subcellular localization, Advances in Protein Chemistry 54 (1) (2000) 277–344.
[48] S. Wan, M. W. Mak, Predicting subcellular localization of multi-location proteins by improving support vector machines with an adaptive-decision scheme, International Journal of Machine Learning and Cybernetics (2015) to appear.
[49] V. N. Vapnik, The nature of statistical learning theory, Springer Verlag, 2000.
[50] S. Wan, M. W. Mak, S. Y. Kung, mPLR-Loc: An adaptive decision multi-label classifier based on penalized logistic regression for protein subcellular localization prediction, Analytical Biochemistry 473 (2015) 14–27.
[51] S. Wan, M. W. Mak, S. Y. Kung, Adaptive thresholding for multi-label SVM classification with application to protein subcellular localization prediction, in: 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'13), 2013, pp. 3547–3551.
[52] S. Wan, M. W. Mak, S. Y. Kung, Benchmark data for identifying multi-functional types of membrane proteins, Data in Brief (2015) submitted.
[53] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining multi-label data, in: Data Mining and Knowledge Discovery Handbook, O. Maimon, l. Rokach (Ed.). Springer, 2nd edition, 2010, pp. 667–685.
[54] R. E. Schapire, Y. Singer, Boostexter: A boosting-based system for text categorization, Machine Learning 39 (2/3) (2000) 135–168.
[55] J. C. Platt, Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods, Advances in Large Margin Classifiers 10 (3) (1999) 61–74.