SNARE-CNN: a 2D convolutional neural network architecture to identify SNARE proteins from high-throughput sequencing data

Nguyen Quoc Khanh Le [Corresp., 1] , **Van-Nui Nguyen** [2]

[1] School of Humanities, Nanyang Technological University, Singapore, Singapore

[2] University of Information and Communication Technology, Thai Nguyen University, Thai Nguyen, Vietnam

Corresponding Author: Nguyen Quoc Khanh Le
Email address: khanhle@ntu.edu.sg

Deep learning has been increasingly and widely used to solve numerous problems in various fields with state-of-the-art performance. It can also be applied in bioinformatics to reduce the requirement for feature extraction and reach high performance. This study attempts to use deep learning to predict SNARE proteins, which is one of the most vital molecular functions in life science. A functional loss of SNARE proteins has been implicated in a variety of human diseases (e.g., neurodegenerative, mental illness, cancer, and so on). Therefore, creating a precise model to identify their functions is a crucial problem for understanding these diseases, and designing the drug targets. Our SNARE-CNN model which uses two-dimensional convolutional neural networks and position-specific scoring matrices profiles could identify SNARE proteins with achieved sensitivity of 76.6%, specificity of 93.5%, accuracy of 89.7%, and MCC of 0.7 in cross-validation dataset. We also evaluate the performance of our model via an independent dataset and the result show that we are able to solve the overfitting problem. Compared with other state-of-the-art methods, this approach achieved significant improvement in all of the metrics. Throughout the proposed study, we provide an effective model for identifying SNARE proteins and a basis for further research that can apply deep learning in bioinformatics, especially in protein function prediction. SNARE-CNN are freely available at https://github.com/khanhlee/snare-cnn .

1 # SNARE-CNN: a 2D convolutional neural network
2 # architecture to identify SNARE proteins from high-
3 # throughput sequencing data

5 Nguyen Quoc Khanh Le[1], Van-Nui Nguyen[2]

7 [1] School of Humanities, Nanyang Technological University, 48 Nanyang Ave, Singapore 639798
8 [2] University of Information and Communication Technology, Thai Nguyen University, Vietnam

10 Corresponding Author:
11 Nguyen Quoc Khanh Le[1]
12 [1]School of Humanities, Nanyang Technological University, 48 Nanyang Ave, Singapore 639798
13 Email address: khanhle@ntu.edu.sg

Abstract

16 Deep learning has been increasingly and widely used to solve numerous problems in various
17 fields with state-of-the-art performance. It can also be applied in bioinformatics to reduce the
18 requirement for feature extraction and reach high performance. This study attempts to use deep
19 learning to predict SNARE proteins, which is one of the most vital molecular functions in life
20 science. A functional loss of SNARE proteins has been implicated in a variety of human diseases
21 (e.g., neurodegenerative, mental illness, cancer, and so on). Therefore, creating a precise model
22 to identify their functions is a crucial problem for understanding these diseases, and designing
23 the drug targets. Our SNARE-CNN model which uses two-dimensional convolutional neural
24 networks and position-specific scoring matrices profiles could identify SNARE proteins with
25 achieved sensitivity of 76.6%, specificity of 93.5%, accuracy of 89.7%, and MCC of 0.7 in
26 cross-validation dataset. We also evaluate the performance of our model via an independent
27 dataset and the result show that we are able to solve the overfitting problem. Compared with
28 other state-of-the-art methods, this approach achieved significant improvement in all of the
29 metrics. Throughout the proposed study, we provide an effective model for identifying SNARE
30 proteins and a basis for further research that can apply deep learning in bioinformatics, especially
31 in protein function prediction. SNARE-CNN are freely available at
32 https://github.com/khanhlee/snare-cnn.

Introduction

34 Deep learning is an advanced machine learning and artificial intelligent technique to learn the
35 representative data with multiple layers of neural networks (LeCun, Bengio, & Hinton, 2015).
36 Numerous difficult problems have been solved with deep learning, e.g., speech recognition,
37 visual object recognition, object detection. The advantages of deep learning are: (1) significantly

38 outperforms other solutions in multiple domains, (2) reduces the requirement for feature
39 extraction and time consumption with the use of graphic processing units (GPUs), and (3) easily
40 adapts to a new problem. Deep neural network models often achieve better performance
41 compared to shallow networks, especially in most of problems with big data. Therefore, deep
42 learning becomes popular and attracts numerous huge companies establishing their directions in
43 this field in recent years. Nowadays, much progress towards deep learning has been made using
44 different deep neural network architectures. A number of studies showed that using deep learning
45 can enhance results in various fields, e.g., prediction of cervical cancer diagnosis (Fernandes,
46 Chicco, Cardoso, & Fernandes, 2018), piRNA (Wang, Hoeksema, & Liang, 2018), and isolated
47 guitar transcription (Burlet & Hindle, 2017). Hence, deep learning is also a fascinating trend in
48 bioinformatics and computational biology research. This study attempts to present a framework
49 to apply deep learning in bioinformatics by using two-dimensional convolutional neural network
50 (2D CNN), which is one popular type of deep neural networks. We anticipate our method will
51 lead to a significant improvement when compared to traditional machine learning techniques in
52 the bioinformatics field.
53 In earlier years, researchers used shallow neural networks for solving a number of problems in
54 bioinformatics and computational biology. For example, Ou constructed QuickRBF package
55 (Oyang, Hwang, Ou, Chen, & Chen, 2005) for training radial basis function (RBF) networks and
56 applied them on several bioinformatics problems including classifying electron transport proteins
57 (Le, Nguyen, & Ou, 2017), transporters (Le, Sandag, & Ou, 2018), and binding sites (Le & Ou,
58 2016a, 2016b). (Chang & Lin, 2011) introduced LibSVM to help biologists implement
59 bioinformatics models by using support vector machines. Recently, as deep learning has been
60 successfully applied in various fields, researchers started to use it in bioinformatics problems,
61 e.g., prediction of piRNA (Wang et al., 2018) and ab initio protein secondary structure (Spencer,
62 Eickholt, & Cheng, 2015). Although those studies achieved very good performances, we believe
63 that we can obtain superior results by using 2D CNN in some bioinformatics applications. In this
64 study, we applied our architecture in the prediction of SNARE proteins, which is one of the most
65 vital molecules in the life sciences.
66 SNARE is an evolutionary superfamily of small proteins that have a conservation pattern of 60-
67 70 amino acids (SNAP motifs) in their cytoplasmic domain. SNARE proteins catalyze cell
68 membrane integration in eukaryotes and are essential for a wide range of cellular processes,
69 including cell growth, cytokinesis, and synaptic transmission (Jahn & Scheller, 2006; Wickner &
70 Schekman, 2008). Most SNAREs contain only one SNARE motif adjacent to a single C-terminal
71 membrane (e.g., synaptobrevin 2 and syntaxin 1). Some SNAREs contain two SNARE motifs
72 that are connected by a long linkage and non-transmembrane sequence (e.g., SNAP-25) but are
73 attached to the membrane through a post-translational modification such as palmitoylation.
74 Various types of SNARE proteins now identified and several studies demonstrated that a
75 functional loss of SNARE proteins has been implicated in numerous diseases (e.g.,
76 neurodegenerative (Hou, Wang, Liu, Wang, & Long, 2017), mental illness (Honer et al., 2002),
77 cancer (Meng & Wang, 2015; Sun et al., 2016), and so on). Therefore, SNARE proteins play an

78   important function in the cell and there is a need to develop some bioinformatics techniques to

79   identify them.

80   Because of the essential role in human diseases, SNARE proteins attracted various researchers

81   who conducted their research on them. For instance, Kloepper team attempted to build a database

82   to store and classify SNARE proteins (Kienle, Kloepper, & Fasshauer, 2009; Kloepper, Kienle,

83   & Fasshauer, 2007, 2008). Next, (van Dijk, Bosch, ter Braak, van der Krol, & van Ham, 2008)

84   built a framework to predict functions of SNAREs in sub-Golgi localization. Moreover, (Weimbs

85   et al., 1997) used bioinformatics techniques to analyze conserved domains in SNARE.

86   (Yoshizawa et al., 2006) extracted sequence motifs and the phylogenetic features of

87   SNARE-dependent membrane trafficking. (Shi, Halder, Yavuz, Jahn, & Shuman, 2016) directed

88   targeting of membrane fusion by SNARE mimicry by convergent evolution of Legionella

89   effectors. (Lu, 2015) analyzed the destructive effect of botulinum neurotoxins on the SNARE

90   protein and proposed that the truncated SNAP-25 mutants will disrupt the assembly of the

91   SNARE core complex, and then inhibit the synaptic membrane fusion accordingly.

92   Most published works on SNARE proteins achieved high performance, but to our knowledge, no

93   researcher conducted the prediction of SNARE proteins using machine learning techniques. It is

94   challenging and motivates us to create a precise model for this. Besides that, we also applied

95   deep learning in this problem, which is a modern technique for classification and obtain high

96   accuracies in various fields. Based on the advantages of deep learning, this study consequently

97   proposes the use of a 2D convolutional neural network (CNN) constructed from position-specific

98   scoring matrix (PSSM) profiles to identify SNARE proteins. The basic principle has already

99   been successfully applied to identify electron transporting proteins (Le, Ho, & Ou, 2017) and

100   Rab GTPases (Le, Ho, & Ou, 2018). Thus, in this paper, we extend this approach to identify the

101   molecular functions of SNARE proteins. The main achievements, including contributions to the

102   field, are presented as follows: (i) development of a deep learning framework to identify SNARE

103   functions from protein sequences, in which our model exhibited a significant improvement

104   beyond traditional machine learning algorithms; (ii) first computational study to identify SNARE

105   proteins and provide useful information to biologists to discover the SNARE molecular

106   functions; (iii) valid benchmark dataset to train and test SNARE proteins with high accuracy,

107   which forms a basis for future research on SNARE proteins.

108   As shown in a series of recent publications (W. Chen, Ding, Zhou, Lin, & Chou, 2018; Cheng,

109   Xiao, & Chou, 2018a, 2018b; Chou, Cheng, & Xiao, 2018; P. Feng et al., 2018; Jia, Li, Qiu,

110   Xiao, & Chou, 2019; Khan, Rasool, Hussain, Khan, & Chou, 2018; Xiao, Xu, et al., 2018), to

111   develop a really useful statistical predictor for a biological system, one should observe the

112   guidelines of Chou's 5-step rule (Chou, 2011) to make the following five steps very clear: (i)

113   how to construct or select a valid benchmark dataset to train and test the predictor; (ii) how to

114   formulate the statistical samples with an effective mathematical expression that can truly reflect

115   their intrinsic correlation with the target to be predicted; (iii) how to introduce or develop a

116   powerful algorithm (or engine) to operate the prediction; (iv) how to properly perform cross-

117   validation tests to objectively evaluate the anticipated accuracy of the predictor; (v) how to

118   provide source code and dataset that are accessible to the public. Below, we are to describe how
119   to deal with these steps one-by-one.

Materials & Methods

121   We implemented an efficient framework for identifying SNARE proteins by using a 2D CNN
122   and PSSM profiles. The framework consists of four procedures: data collection, feature
123   extraction, CNN generation, and model evaluation. Fig. 1 presents the flowchart of our
124   framework, and its details are described as follows.

Dataset

126   The dataset was retrieved from the UniProt database (by 22-10-2018) (Consortium, 2014), which
127   is one of the comprehensive resources for the protein sequence. First of all, we collected all
128   SNAREs proteins from the UniProt annotation (by using keyword "SNARE"). Note that only
129   reviewed proteins (records with information extracted from literature and curator-evaluated
130   computational analysis) were collected. Subsequently, BLAST (Altschul et al., 1997) was
131   applied to remove the redundant sequences with similarity more than 30%, and the rest of the
132   proteins reaches 245 SNAREs. However, after this process, the rest of proteins only reaches 245
133   SNAREs, and the number of data points was insufficient for a precise deep learning model.
134   Hence, we used a cut-off level of 100% in the cross-validation dataset for more data to create a
135   significant model. We still used similarity of 30% in the independent dataset to evaluate the
136   performance of the model. This step is a very important step to check if the model was
137   overfitting or not.
138   The proposed problem was the binary classification between SNARE proteins and general
139   proteins, thus we collected a set of general proteins as negative data. In order to create a precise
140   model, there is a need to collect negative dataset which has a similar function and structure with
141   the positive dataset. From that, it is challenging to build a precise model but it increases our
142   contribution to the predictor. It will also help us decrease the number of negative data collected.
143   After considering the structure and function, we chose vesicular transport protein, which is a
144   general protein including SNARE protein. We counted it as negative data to perform the
145   classification problem. We removed the redundant data between two datasets as well as the
146   sequences with similarity more than 30%. Finally, there were 682 SNARE proteins and 2583
147   non-SNARE proteins used. We then divided data into cross-validation and independent dataset.
148   The detail of the dataset using in this study is listed in Table 1.

Encoding feature sets from the protein sequence information

150   In order to convert the protein sequence information into feature sets, we applied the PSSM
151   matrices for FASTA sequences. A PSSM profile is a matrix represented by all motifs in
152   biological sequences in general and in protein sequences in particular. It is created by rendering
153   two sequences having similar structures with different amino acid compositions. Therefore,
154   PSSM profiles have been adopted and used in a number of bioinformatics problems, e.g.,
155   prediction of protein secondary structure (Jones, 1999), protein disorder (Shimizu, Hirose, &

156 Noguchi, 2007), and transport protein (Ou, Chen, & Gromiha, 2010) with significant
157 improvements.
158 Since the retrieved dataset is in FASTA format, it needs to be converted into PSSM profiles. To
159 perform this task, we used PSI-BLAST (Altschul et al., 1997) to search all the sequence
160 alignments of proteins in the non-redundant (NR) database with two iterations. The query to
161 produce the PSSM profile is as follows:
162   *psiblast.exe -num_iterations 2 -db <nr> -in_msa <fasta_file> -out_ascii_<pssm_file>*
163 The feature extraction part of Fig. 1 indicates the information of generating the 400 PSSM
164 capabilities from original PSSM profiles. Each amino acid in the sequence is represented by a
165 vector of 20 values (each row). First, we summed up all rows with the same amino acid to
166 transform the original PSSM profiles to PSSM profiles with 400 dimensions. The purpose of this
167 step is to force this data type into something easier for the neural network to deal with. Each
168 element of the 400D input vector was then divided by the sequence length and then be scaled
169 before inserting into neural networks.

170 **Input layers for 2D convolutional neural networks**
171 The architecture of our CNN is described in the below part of Fig. 1. The CNN contains three
172 layers: an input layer, hidden layers (including convolutional, pooling and fully connected
173 layers), and an output layer. CNN had been applied in numerous applications in various fields
174 and convinced wonderful results (Amidi et al., 2018; Palatnik de Sousa, 2018). In our study, an
175 input of the CNN is a PSSM corresponding to the protein sequences. We then propose a method
176 to predict SNARE proteins by using their PSSM profiles as the input data. With this type of
177 dataset, we assumed the PSSM profile with 20x20 matrix as a grayscale image with 20x20
178 pixels, we can then train the model with two-dimensional CNN. The input PSSM profile was
179 then connected to our 2D CNN in which we set a variety of parameters to improve the
180 performance of the model. By using a 2D CNN rather than other neural network structures, we
181 aimed to capture as many hidden spatial features as possible in the PSSM matrices. This
182 approach guarantees the correctness of the generated features and prevents the disorder problem
183 inside the amino acid sequences. The more hidden layers generated, the more hidden features
184 generated in CNN to identify SNARE proteins easily. In this work, we used four filter layers
185 (with 32, 64, 128, and 256 filters) and three different kernel sizes in each filter.

186 **Multiple hidden layers for deep neural networks**
187 Following the input layer, hidden layers aim to generate matrices to learn the features. We
188 established the hidden layers that contained various sub-layers with different parameters and
189 shapes. Those 2D sub-layers are zero padding, convolutional, max pooling and fully-connected
190 layers with different numbers of filters. All of the layers are combined together to become the
191 nodes in the deep neural networks. The quality of our model was determined by the number of
192 layers and parameters. The first layer of our 2D CNN architecture is the zero padding 2D layer,
193 which added zero values at the beginning and the end of 20x20 matrices. The shape matrix
194 changed to 22x22 dimensions when we added the zero padding layer into our network. After we

195   applied the filters into the input shape, the output dimension was not different under the effect of
196   the zero padding.

$$zp = \frac{k-1}{2} \quad (1)$$

198   where k is the filter size. Next, the 2D convolution layer was used with a kernel size of 3x3,
199   meaning that the features will be learned with the 3x3 matrices and sliced to the end. After each
200   step, the next layer will take the weights and biases from its previous layer and train again.
201   Normally, a 2D max-pooling layer follows the 2D convolution layer. There are several
202   parameters for a max-pooling layer, i.e., loop size and stride. In our study, we performed max
203   pooling by a stride of 2 through the selection of the maximum value over a window of 22. By
204   using this process, we can reduce the processing time in the next layers. The output size of a
205   convolutional layer is computed as follow.

$$os = \frac{w-k+2p}{s} + 1 \quad (2)$$

207   where w is the input size, k is the filter size, p is the padding and s is the stride size.

208   **Output layers**
209   The first layer in the output layer is a flatten layer. A flatten layer is always included before fully
210   connected layers to convert the input matrix into a vector. We applied two fully connected layers
211   in which each node is fully-connected to all the nodes of the previous layer. Fully connected
212   layers are typically used in the last stages of CNNs. All the nodes of the first layer are connected
213   to the flatten layer to allow the model to gain more knowledge and perform better. The second
214   layer connects the first fully-connected layer to the output layer. Moreover, we inserted the next
215   layer, dropout, to enhance the performance results of the model and it also helps our model
216   prevent overfitting (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). In the
217   dropout layer, the model will randomly deactivate a number of neurons with a certain probability
218   p. By tuning the dropout value (from 0 to 1), we will save a lot of computing time for the next
219   layers, and the training time will be faster. Furthermore, an additional non-linear operation called
220   ReLU (Rectified Linear Unit) was performed after each convolution operation. To define the
221   ReLU output, we used this formula:
222

$$f(x) = \max(0,x) \quad (3)$$

223   where x is the number of inputs into a neural network. The output of the model was computed
224   through a softmax function by which the probability for each possible output was determined.
225   The softmax function is a logistic function which is defined by the following formula:

226

$$\sigma(z)_i = \frac{e^{z_i}}{\sum\limits_{k=1}^{K} e^{z_k}} \quad (4)$$

227 where z is the input vector with K-dimensional vector, K-dimensional vector σ(z) is real values
228 in the range (0, 1) and j^{th} class is the predicted probability from sample vector x. In summary, we
229 set a total of 233,314 trainable parameters in the model (Table 2).

Performance evaluation

231 The most important purpose of the present study was to predict whether or not a sequence is
232 SNARE protein; therefore, we used "Positive" to define the SNARE protein, and
233 "Negative" to define the non-SNARE protein. For each dataset, we first trained the model by
234 applying 5-fold cross-validation technique on the training dataset. Based on the 5-fold cross-
235 validation results, hyper-parameter optimization process was employed to find the best model for
236 each dataset. Finally, the independent dataset was used to assess the predictive ability of the
237 current model.
238 Based on the Chou's symbols introduced for studying protein signal peptides (Chou, 2001), a set
239 of four intuitive metrics were derived, as given in Eq.14 of (W. Chen, Feng, Lin, & Chou, 2013)
240 or in Eq.19 of (Xu, Shao, Wu, Deng, & Chou, 2013). For evaluating the performance of the
241 methods, we also adopted Chou's criterion used in many bioinformatics studies (J. Chen, Liu,
242 Yang, & Chou, 2007; P.-M. Feng, Chen, Lin, & Chou, 2013; Taju, Nguyen, Le, Kusuma, & Ou,
243 2018). Either the set of traditional metrics copied from math books or the intuitive metrics
244 derived from the Chou's symbols (Eqs.5-8) are valid only for the single-label systems (where
245 each sample only belongs to one class). For the multi-label systems (where a sample may
246 simultaneously belong to several classes), whose existence has become more frequent in system
247 biology (Cheng, Xiao, & Chou, 2017a, 2017b; Cheng, Zhao, Lin, Xiao, & Chou, 2017; Xiao,
248 Cheng, Chen, Mao, & Chou, 2018), system medicine (Cheng, Zhao, Xiao, & Chou, 2017) and
249 biomedicine (Qiu, Sun, Xiao, Xu, & Chou, 2016), a completely different set of metrics as
250 defined in (Chou, 2013) is absolutely needed. Some standard metrics were used, such as
251 sensitivity, specificity, accuracy, and Matthews correlation coefficient (MCC) using below given
252 formulae (TP, FP, TN, FN are true positive, false positive, true negative, and false negative
253 values, respectively):

254
$$Sensitivity = 1 - \frac{N_-^+}{N^+}, \quad 0 \le Sen \le 1 \qquad (5)$$

255
$$Specificity = 1 - \frac{N_+^-}{N^-}, \quad 0 \le Spec \le 1 \qquad (6)$$

256
$$Accuracy = 1 - \frac{N_-^+ + N_+^-}{N^+ + N^-}, \quad 0 \le Acc \le 1 \qquad (7)$$

257
$$MCC = \frac{1 - \left(\frac{N_-^+}{N^+} + \frac{N_+^-}{N^-}\right)}{\sqrt{\left(1 + \frac{N_+^- - N_-^+}{N^+}\right)\left(1 + \frac{N_-^+ - N_+^-}{N^-}\right)}}, \quad -1 \le MCC \le 1 \qquad (8)$$

258   The relations between these symbols and the symbols in Eqs. (5, 6, 7 and 8) are given by:

259
$$\begin{cases} N_+^{\,-} = FP \\ N_-^{\,+} = FN \\ N^+ = TP + N_-^{\,+} \\ N^- = TN + N_+^{\,-} \end{cases} \qquad (9)$$

Results and Discussions

261   The quality and reliability of the modeling techniques of research is an important factor in the
262   study. Initially, we designed an experiment by analyzing data, perform calculations and take
263   various comparisons in the results and discussions section.

Composition of amino acid in SNARE and non-SNARE proteins

265   We analyzed the composition of amino acid and the variance of amino acid composition in
266   SNARE sequences and non-SNARE sequences by computing the frequency between them. Fig.
267   2 illustrates the amino acids which contributed the significantly highest frequency in two
268   different datasets. We realized that the amino acid E, and K, and P occur at the highest
269   frequencies surrounding the SNARE proteins. On the other hand, amino acids G and P occur at
270   the highest frequencies surrounding the non-SNARE proteins. Therefore, these amino acids
271   certainly had an essential role in identifying SNARE proteins. Thus, our model might predict
272   SNARE proteins accurately via the special features from those amino acids contributions.

Performance for identifying SNARE proteins with 2D CNN

274   We implemented our 2D CNN architecture by using Keras package with Tensorflow backend.
275   First, we tried to find the optimal setup for the hidden layers by doing experiments using four
276   different filter sizes: 32, 64, 128, and 256. Table 3 demonstrates the performance results from
277   various filter layers in the cross-validation dataset. We easily observe that during the 5-fold
278   cross-validation to identify SNAREs, the model with 256 filters was prominent identifying these
279   sequences with an average 5-fold cross-validation accuracy of 88.2%. The performance results
280   are higher than the performances from the other results with other filters. The sensitivity,
281   specificity, and MCC for cross-validation data achieved 70.5%, 93.3%, and 0.65, respectively.
282   Therefore, we used 256 filters for the hidden layer to develop our model. We then optimized the
283   neural networks using a variety of optimizers: rmsprop, adam, nadam, sgd, and adadelta. The
284   model was reinitialized, i.e. a new network is built, after each round of optimization so as to
285   provide a fair comparison between the different optimizers. Overall, the performance results are
286   shown in Fig. 3 and we decided to choose nadam, an optimizer with consistent performance to
287   create our final model.

Improving the performance results and preventing over-fitting problem with dropout

290 It can be seen that there was a fair difference in performance between using the cross-validation
291 dataset and the independent dataset. It is due to the non-removing similarity in cross-validation,
292 and now we address this issue. To solve this issue, we applied an important technique called
293 dropout (Srivastava et al., 2014). Table 4 presents the performances of the model when we varied
294 the dropout value from 0 to 1. It can be seen that the performance from the dropout value of 0.1
295 was higher than others, with the sensitivity, specificity, accuracy, and MCC of 72.4%, 94.4%,
296 89.5%, and 0.69, respectively. In the independent dataset, the sensitivity, specificity, accuracy,
297 and MCC were 44.7%, 95.4%, 90.4%, and 0.43, respectively. We can see that the performance
298 of the independent dataset has been already improved and moved closer to that of the cross-
299 validation dataset. Therefore, the over-fitting problem was gradually resolved, and we used this
300 dropout value for our final model.
301 Moreover, the number of epochs used in the experiment extremely affects the performance
302 results. To discover the optimal epoch, we ran our experiments by ranging the epoch value from
303 the first epoch to the 500th epoch. During this process, we saved the checkpoint with the highest
304 performance and used its parameters to create our model. Until this final step, the independent
305 sensitivity, specificity, accuracy, and MCC reached 65.8%, 90.3%, 87.9% and 0.46, respectively.
306 This result is close to that of the cross-validation dataset at the same level of 2D CNN
307 architecture. Finally, our model applied 256 filter layers, nadam optimizer, and dropout value of
308 0.1 to identify SNARE proteins with the highest performance.

309 **Comparative performance for identifying SNAREs between 2D CNN and shallow**
310 **neural networks**
311 We examined the performances of using different machine learning classifiers for identifying
312 SNARE proteins. We used four different classifiers (i.e., nearest neighbor (kNN), Gaussian,
313 Random Forest, and support vector machine (SVM)) to evaluate the model and compared 2D
314 CNN results with their results. For a fair comparison, we definitely used the optimal parameters
315 for all the classifiers in all the experiments. Table 5 shows the performance results between our
316 method and other machine learning algorithms. It can be seen that our 2D CNN exhibited higher
317 performance than those of the other traditional machine learning techniques using the same
318 experimental setup. Especially, our 2D CNN outperformed other algorithms when using the
319 independent dataset.

320 **Comparative performance for identifying SNAREs between 2D CNN and BLAST**
321 **search pipeline**
322 To make our prediction have convincing, we aim to simply BLASTing the SNARE and non-
323 SNARE sequences. The objective of this step is to check whether the first non-identical match
324 was a SNARE/non-SNARE protein. We then compared with our PSSM via PSI-BLAST and the
325 performance results were shown in Table 6. It is easy to say that we are able to reach a better
326 performance when using the PSSM profiles to build a classifier. It also means that BLAST can
327 search a sequence within motifs, but it cannot capture hidden information in sequences.

328   Therefore, it is necessary and useful to create an advanced classifier with stronger features e.g.,
329   PSSM profiles in this study.
330   Furthermore, source codes and publicly accessible web-servers represent the current trend for
331   developing various computational methods (W. Chen et al., 2018; Cheng et al., 2018a, 2018b;
332   Chou et al., 2018; P. Feng et al., 2018; Jia et al., 2019; Khan et al., 2018; Le, Ho, & Ou; Xiao,
333   Xu, et al., 2018). Actually, they have significantly enhanced the impacts of computational
334   biology on medical science (Chou, 2015), driving medicinal chemistry into an unprecedented
335   revolution (Chou, 2017), here we also publish our source codes and dataset at
336   https://github.com/khanhlee/snare-cnn for presenting the new method reported in this paper.

Conclusions

338   Deep learning, a leading technique in various fields, has been increasingly applied in
339   bioinformatics and computational biology. This study approaches a novel for identifying SNARE
340   proteins by using deep learning. The idea is to transform PSSM profiles into matrices and use
341   them as the input to 2D CNN architectures. We evaluated the performance of our model, which
342   was developed by using a 2D CNN and PSSM profiles, using 5-fold cross-validation and an
343   independent testing dataset. Our method produced superior performance, and compared to other
344   state-of-the-art neural networks, it achieved a significant improvement in all the typical
345   measurement metrics. Using our model, new SNARE proteins can be accurately identified and
346   used for drug development. Moreover, the contribution of this study could help further research
347   to promote the use of 2D CNN in bioinformatics, especially in protein function prediction.
348

Acknowledgements

350   Not applicable.

References

352

353   Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J.
354   (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search
355   programs. *Nucleic acids research, 25*(17), 3389-3402.
356   Amidi, A., Amidi, S., Vlachakis, D., Megalooikonomou, V., Paragios, N., & Zacharaki, E. I.
357   (2018). EnzyNet: enzyme classification using 3D convolutional neural networks on
358   spatial representation. *PeerJ, 6*, e4750. doi:10.7717/peerj.4750
359   Burlet, G., & Hindle, A. (2017). Isolated guitar transcription using a deep belief network. *PeerJ*
360   *Computer Science, 3*, e109. doi:10.7717/peerj-cs.109
361   Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM*
362   *transactions on intelligent systems and technology (TIST), 2*(3), 27.
363   Chen, J., Liu, H., Yang, J., & Chou, K.-C. (2007). Prediction of linear B-cell epitopes using
364   amino acid pair antigenicity scale. *Amino acids, 33*(3), 423-428.
365   Chen, W., Ding, H., Zhou, X., Lin, H., & Chou, K.-C. (2018). iRNA(m6A)-PseDNC: Identifying
366   N6-methyladenosine sites using pseudo dinucleotide composition. *Analytical*
367   *Biochemistry, 561-562*, 59-65. doi:https://doi.org/10.1016/j.ab.2018.09.002

368   Chen, W., Feng, P.-M., Lin, H., & Chou, K.-C. (2013). iRSpot-PseDNC: identify recombination
369   spots with pseudo dinucleotide composition. *Nucleic acids research, 41*(6), e68-e68.
370   doi:10.1093/nar/gks1450
371   Cheng, X., Xiao, X., & Chou, K.-C. (2017a). pLoc-mPlant: predict subcellular localization of
372   multi-location plant proteins by incorporating the optimal GO information into general
373   PseAAC. *Molecular BioSystems, 13*(9), 1722-1727. doi:10.1039/C7MB00267J
374   Cheng, X., Xiao, X., & Chou, K.-C. (2017b). pLoc-mVirus: Predict subcellular localization of
375   multi-location virus proteins via incorporating the optimal GO information into general
376   PseAAC. *Gene, 628*, 315-321. doi:https://doi.org/10.1016/j.gene.2017.07.036
377   Cheng, X., Xiao, X., & Chou, K.-C. (2018a). pLoc-mEuk: Predict subcellular localization of multi-
378   label eukaryotic proteins by extracting the key GO information into general PseAAC.
379   *Genomics, 110*(1), 50-58. doi:https://doi.org/10.1016/j.ygeno.2017.08.005
380   Cheng, X., Xiao, X., & Chou, K.-C. (2018b). pLoc-mGneg: Predict subcellular localization of
381   Gram-negative bacterial proteins by deep gene ontology learning via general PseAAC.
382   *Genomics, 110*(4), 231-239. doi:https://doi.org/10.1016/j.ygeno.2017.10.002
383   Cheng, X., Zhao, S.-G., Lin, W.-Z., Xiao, X., & Chou, K.-C. (2017). pLoc-mAnimal: predict
384   subcellular localization of animal proteins with both single and multiple sites.
385   *Bioinformatics, 33*(22), 3524-3531. doi:10.1093/bioinformatics/btx476
386   Cheng, X., Zhao, S.-G., Xiao, X., & Chou, K.-C. (2017). iATC-mISF: a multi-label classifier for
387   predicting the classes of anatomical therapeutic chemicals. *Bioinformatics, 33*(3), 341-
388   346. doi:10.1093/bioinformatics/btw644
389   Chou, K.-C. (2001). Prediction of signal peptides using scaled window. *Peptides, 22*(12), 1973-
390   1979. doi:https://doi.org/10.1016/S0196-9781(01)00540-X
391   Chou, K.-C. (2011). Some remarks on protein attribute prediction and pseudo amino acid
392   composition. *Journal of Theoretical Biology, 273*(1), 236-247.
393   doi:https://doi.org/10.1016/j.jtbi.2010.12.024
394   Chou, K.-C. (2013). Some remarks on predicting multi-label attributes in molecular biosystems.
395   *Molecular BioSystems, 9*(6), 1092-1100. doi:10.1039/C3MB25555G
396   Chou, K.-C. (2015). Impacts of Bioinformatics to Medicinal Chemistry. *Medicinal Chemistry,
397   11*(3), 218-234. doi:http://dx.doi.org/10.2174/1573406411666141229162834
398   Chou, K.-C. (2017). An Unprecedented Revolution in Medicinal Chemistry Driven by the
399   Progress of Biological Science. *Current Topics in Medicinal Chemistry, 17*(21), 2337-
400   2358. doi:http://dx.doi.org/10.2174/1568026617666170414145508
401   Chou, K.-C., Cheng, X., & Xiao, X. (2018). pLoc_bal-mHum: Predict subcellular localization of
402   human proteins by PseAAC and quasi-balancing training dataset. *Genomics*.
403   doi:https://doi.org/10.1016/j.ygeno.2018.08.007
404   Consortium, U. (2014). UniProt: a hub for protein information. *Nucleic acids research, 43*(D1),
405   D204-D212.
406   Feng, P.-M., Chen, W., Lin, H., & Chou, K.-C. (2013). iHSP-PseRAAAC: Identifying the heat
407   shock protein families using pseudo reduced amino acid alphabet composition.
408   *Analytical Biochemistry, 442*(1), 118-125.
409   Feng, P., Yang, H., Ding, H., Lin, H., Chen, W., & Chou, K.-C. (2018). iDNA6mA-PseKNC:
410   Identifying DNA N6-methyladenosine sites by incorporating nucleotide physicochemical
411   properties into PseKNC. *Genomics*. doi:https://doi.org/10.1016/j.ygeno.2018.01.005
412   Fernandes, K., Chicco, D., Cardoso, J. S., & Fernandes, J. (2018). Supervised deep learning
413   embeddings for the prediction of cervical cancer diagnosis. *PeerJ Computer Science, 4*,
414   e154. doi:10.7717/peerj-cs.154
415   Honer, W. G., Falkai, P., Bayer, T. A., Xie, J., Hu, L., Li, H.-Y., . . . Trimble, W. S. (2002).
416   Abnormalities of SNARE Mechanism Proteins in Anterior Frontal Cortex in Severe
417   Mental Illness. *Cerebral Cortex, 12*(4), 349-356. doi:10.1093/cercor/12.4.349

418 Hou, C., Wang, Y., Liu, J., Wang, C., & Long, J. (2017). Neurodegenerative Disease Related
419   Proteins Have Negative Effects on SNARE-Mediated Membrane Fusion in Pathological
420   Confirmation. *Frontiers in molecular neuroscience, 10*, 66.
421 Jahn, R., & Scheller, R. H. (2006). SNAREs — engines for membrane fusion. *Nature Reviews*
422   *Molecular Cell Biology, 7*, 631. doi:10.1038/nrm2002
423 Jia, J., Li, X., Qiu, W., Xiao, X., & Chou, K.-C. (2019). iPPI-PseAAC(CGR): Identify protein-
424   protein interactions by incorporating chaos game representation into PseAAC. *Journal of*
425   *Theoretical Biology, 460*, 195-203. doi:https://doi.org/10.1016/j.jtbi.2018.10.021
426 Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring
427   matrices1. *Journal of molecular biology, 292*(2), 195-202.
428 Khan, Y. D., Rasool, N., Hussain, W., Khan, S. A., & Chou, K.-C. (2018). iPhosT-PseAAC:
429   Identify phosphothreonine sites by incorporating sequence statistical moments into
430   PseAAC. *Analytical Biochemistry, 550*, 109-116.
431   doi:https://doi.org/10.1016/j.ab.2018.04.021
432 Kienle, N., Kloepper, T. H., & Fasshauer, D. (2009). Phylogeny of the SNARE vesicle fusion
433   machinery yields insights into the conservation of the secretory pathway in fungi. *BMC*
434   *evolutionary biology, 9*(1), 19.
435 Kloepper, T. H., Kienle, C. N., & Fasshauer, D. (2007). An elaborate classification of SNARE
436   proteins sheds light on the conservation of the eukaryotic endomembrane system.
437   *Molecular biology of the cell, 18*(9), 3463-3471.
438 Kloepper, T. H., Kienle, C. N., & Fasshauer, D. (2008). SNAREing the basis of multicellularity:
439   consequences of protein family expansion during evolution. *Molecular biology and*
440   *evolution, 25*(9), 2055-2068.
441 Le, N. Q. K., Ho, Q.-T., & Ou, Y.-Y. Using two-dimensional convolutional neural networks for
442   identifying GTP binding sites in Rab proteins. *Journal of Bioinformatics and*
443   *Computational Biology, 0*. doi:10.1142/s0219720019500057
444 Le, N. Q. K., Ho, Q.-T., & Ou, Y.-Y. (2017). Incorporating deep learning with convolutional
445   neural networks and position specific scoring matrices for identifying electron transport
446   proteins. *Journal of Computational Chemistry, 38*(23), 2000-2006.
447   doi:doi:10.1002/jcc.24842
448 Le, N. Q. K., Ho, Q.-T., & Ou, Y.-Y. (2018). Classifying the molecular functions of Rab GTPases
449   in membrane trafficking using deep convolutional neural networks. *Analytical*
450   *Biochemistry, 555*, 33-41. doi:https://doi.org/10.1016/j.ab.2018.06.011
451 Le, N. Q. K., Nguyen, T.-T.-D., & Ou, Y.-Y. (2017). Identifying the molecular functions of
452   electron transport proteins using radial basis function networks and biochemical
453   properties. *Journal of Molecular Graphics and Modelling, 73*, 166-178.
454   doi:https://doi.org/10.1016/j.jmgm.2017.01.003
455 Le, N. Q. K., & Ou, Y.-Y. (2016a). Incorporating efficient radial basis function networks and
456   significant amino acid pairs for predicting GTP binding sites in transport proteins. *BMC*
457   *Bioinformatics, 17*(19), 501. doi:10.1186/s12859-016-1369-y
458 Le, N. Q. K., & Ou, Y.-Y. (2016b). Prediction of FAD binding sites in electron transport proteins
459   according to efficient radial basis function networks and significant amino acid pairs.
460   *BMC Bioinformatics, 17*(1), 298. doi:10.1186/s12859-016-1163-x
461 Le, N. Q. K., Sandag, G. A., & Ou, Y.-Y. (2018). Incorporating post translational modification
462   information for enhancing the predictive performance of membrane transport proteins.
463   *Computational Biology and Chemistry, 77*, 251-260.
464   doi:https://doi.org/10.1016/j.compbiolchem.2018.10.010
465 LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature, 521*(7553), 436.
466 Lu, B. (2015). The destructive effect of botulinum neurotoxins on the SNARE protein: SNAP-25
467   and synaptic membrane fusion. *PeerJ, 3*, e1065. doi:10.7717/peerj.1065

468   Meng, J., & Wang, J. (2015). Role of SNARE proteins in tumourigenesis and their potential as
469   targets for novel anti-cancer therapeutics. *Biochimica et Biophysica Acta (BBA) -*
470   *Reviews on Cancer, 1856*(1), 1-12. doi:https://doi.org/10.1016/j.bbcan.2015.04.002
471   Ou, Y. Y., Chen, S. A., & Gromiha, M. M. (2010). Classification of transporters using efficient
472   radial basis function networks with position-specific scoring matrices and biochemical
473   properties. *Proteins: Structure, Function, and Bioinformatics, 78*(7), 1789-1797.
474   Oyang, Y.-J., Hwang, S.-C., Ou, Y.-Y., Chen, C.-Y., & Chen, Z.-W. (2005). Data classification
475   with radial basis function networks based on a novel kernel density estimation algorithm.
476   *IEEE transactions on neural networks, 16*(1), 225-236.
477   Palatnik de Sousa, I. (2018). Convolutional ensembles for Arabic Handwritten Character and
478   Digit Recognition. *PeerJ Computer Science, 4*, e167. doi:10.7717/peerj-cs.167
479   Qiu, W.-R., Sun, B.-Q., Xiao, X., Xu, Z.-C., & Chou, K.-C. (2016). iPTM-mLys: identifying
480   multiple lysine PTM sites and their different types. *Bioinformatics, 32*(20), 3116-3123.
481   doi:10.1093/bioinformatics/btw380
482   Shi, X., Halder, P., Yavuz, H., Jahn, R., & Shuman, H. A. (2016). Direct targeting of membrane
483   fusion by SNARE mimicry: Convergent evolution of Legionella effectors. *Proceedings of*
484   *the National Academy of Sciences, 113*(31), 8807.
485   Shimizu, K., Hirose, S., & Noguchi, T. (2007). POODLE-S: web application for predicting protein
486   disorder by using physicochemical features and reduced amino acid set of a position-
487   specific scoring matrix. *Bioinformatics, 23*(17), 2337-2338.
488   Spencer, M., Eickholt, J., & Cheng, J. (2015). A deep learning network approach to ab initio
489   protein secondary structure prediction. *IEEE/ACM transactions on computational biology*
490   *and bioinformatics (TCBB), 12*(1), 103-112.
491   Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a
492   simple way to prevent neural networks from overfitting. *The Journal of Machine Learning*
493   *Research, 15*(1), 1929-1958.
494   Sun, Q., Huang, X., Zhang, Q., Qu, J., Shen, Y., Wang, X., . . . Ren, B. (2016). SNAP23
495   promotes the malignant process of ovarian cancer. *Journal of Ovarian Research, 9*(1),
496   80. doi:10.1186/s13048-016-0289-9
497   Taju, S. W., Nguyen, T.-T.-D., Le, N.-Q.-K., Kusuma, R. M. I., & Ou, Y.-Y. (2018). DeepEfflux: a
498   2D convolutional neural network model for identifying families of efflux proteins in
499   transporters. *Bioinformatics, 34*(18), 3111-3117. doi:10.1093/bioinformatics/bty302
500   van Dijk, A. D. J., Bosch, D., ter Braak, C. J. F., van der Krol, A. R., & van Ham, R. C. H. J.
501   (2008). Predicting sub-Golgi localization of type II membrane proteins. *Bioinformatics,*
502   *24*(16), 1779-1786. doi:10.1093/bioinformatics/btn309
503   Wang, K., Hoeksema, J., & Liang, C. (2018). piRNN: deep learning algorithm for piRNA
504   prediction. *PeerJ, 6*, e5429. doi:10.7717/peerj.5429
505   Weimbs, T., Low, S. H., Chapin, S. J., Mostov, K. E., Bucher, P., & Hofmann, K. (1997). A
506   conserved domain is present in different families of vesicular fusion proteins: a new
507   superfamily. *Proceedings of the National Academy of Sciences, 94*(7), 3046-3051.
508   Wickner, W., & Schekman, R. (2008). Membrane fusion. *Nature structural & molecular biology,*
509   *15*(7), 658.
510   Xiao, X., Cheng, X., Chen, G., Mao, Q., & Chou, K.-C. (2018). pLoc_bal-mGpos: Predict
511   subcellular localization of Gram-positive bacterial proteins by quasi-balancing training
512   dataset and PseAAC. *Genomics*. doi:https://doi.org/10.1016/j.ygeno.2018.05.017
513   Xiao, X., Xu, Z.-C., Qiu, W.-R., Wang, P., Ge, H.-T., & Chou, K.-C. (2018). iPSW(2L)-PseKNC:
514   A two-layer predictor for identifying promoters and their strength by hybrid features via
515   pseudo K-tuple nucleotide composition. *Genomics*.
516   doi:https://doi.org/10.1016/j.ygeno.2018.12.001

517 Xu, Y., Shao, X.-J., Wu, L.-Y., Deng, N.-Y., & Chou, K.-C. (2013). iSNO-AAPair: incorporating
518   amino acid pairwise coupling into PseAAC for predicting cysteine S-nitrosylation sites in
519   proteins. *PeerJ, 1*, e171. doi:10.7717/peerj.171
520 Yoshizawa, A. C., Kawashima, S., Okuda, S., Fujita, M., Itoh, M., Moriya, Y., . . . Kanehisa, M.
521   (2006). Extracting Sequence Motifs and the Phylogenetic Features of SNARE-
522   Dependent Membrane Traffic. *Traffic, 7*(8), 1104-1118. doi:doi:10.1111/j.1600-
523   0854.2006.00451.x
524

List of Figures

List of Tables

Figure 1

Flowchart for identifying SNARE proteins using two-dimensional convolutional neural networks

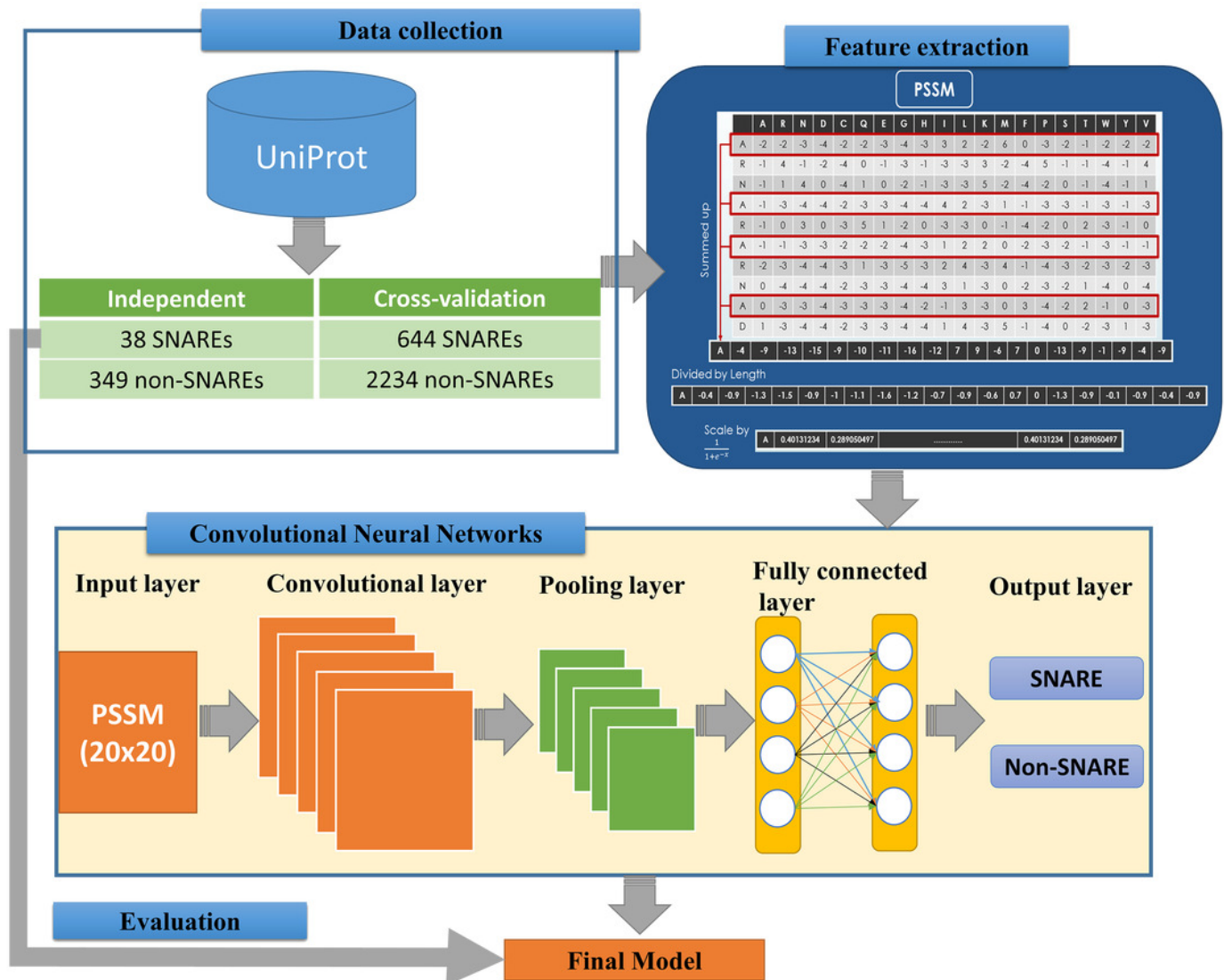



Figure 2

Amino acid composition in SNARE and non-SNARE proteins

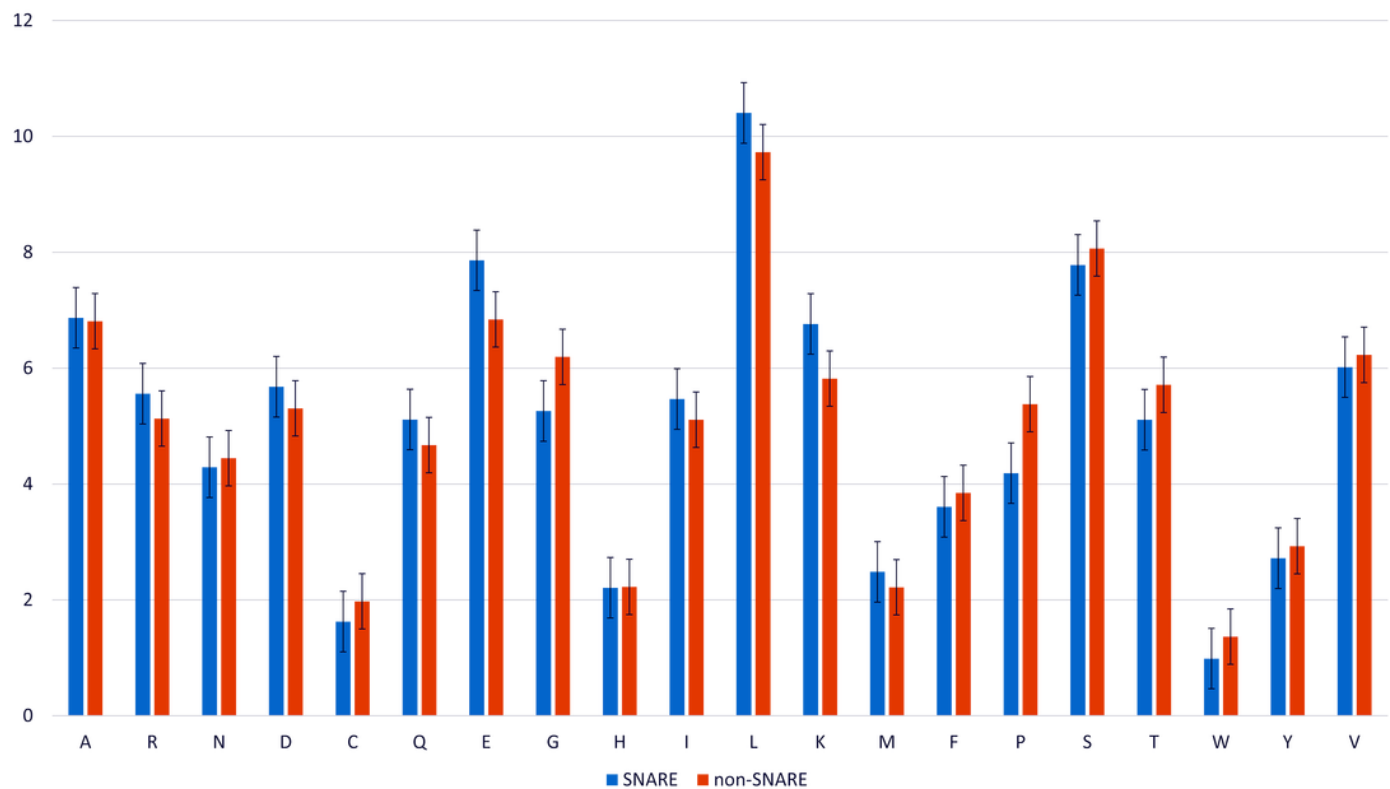
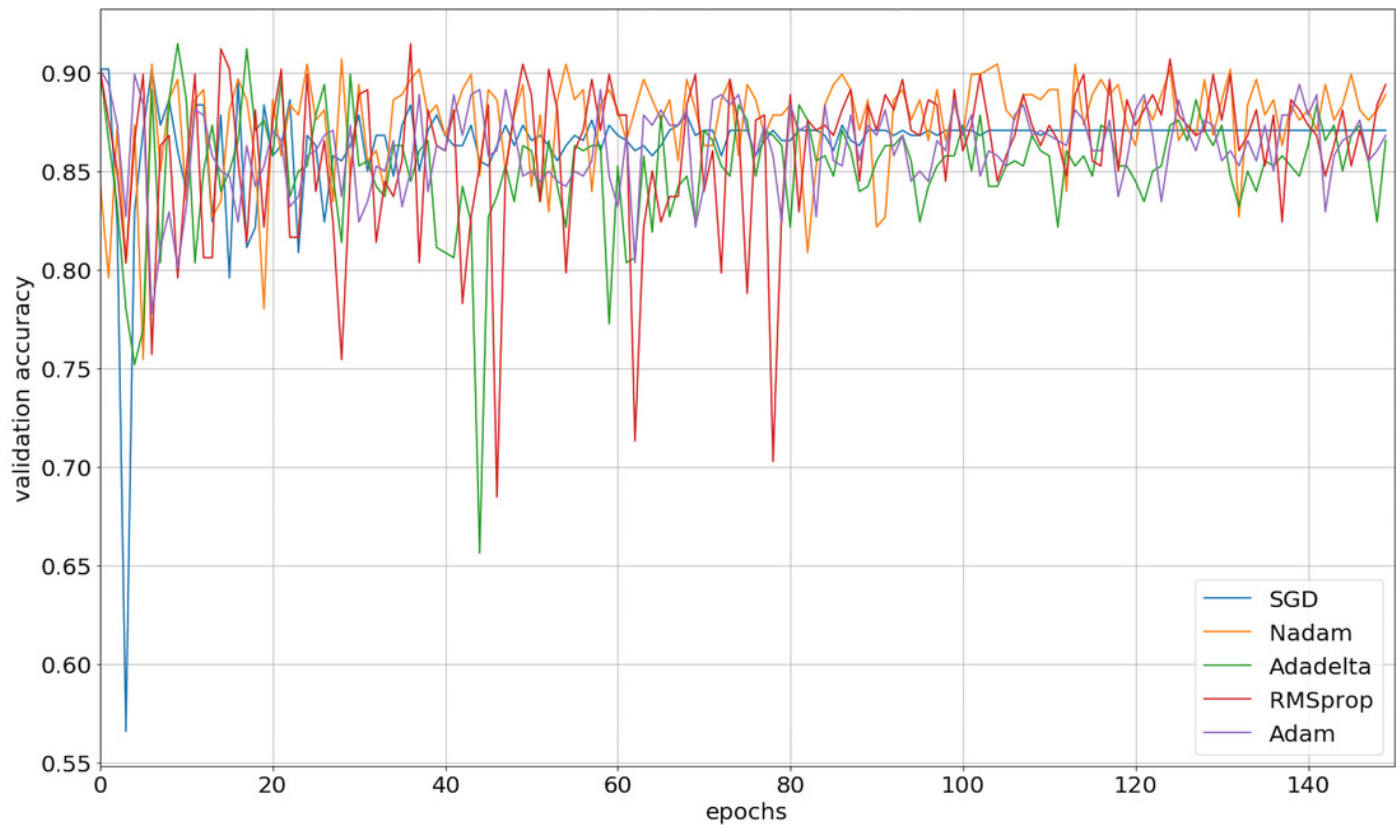



Figure 3

The validation accuracy on identifying SNARE proteins using different optimizers

Statistics of all retrieved SNARE and non-SNARE proteins

1   **Table 1.** Statistics of all retrieved SNARE and non-SNARE proteins

	Cross-validation	Independent
SNARE	644	38
Non-SNARE	2234	349

All layers and trainable parameters of the two-dimensional convolutional neural networks in this study

Table 2. All layers and trainable parameters of the two-dimensional convolutional neural networks in this study

Layer (type)	Output shape	Parameters #
Zeropadding2d_1	(None, 3, 22, 20)	0
Conv2d_1	(None, 1, 20, 32)	5792
Max_pooling2d_1	(None, 1, 10, 16)	0
Zeropadding2d_2	(None, 3, 12, 16)	0
Conv2d_2	(None, 1, 10, 64)	9280
Max_pooling2d_2	(None, 1, 5, 32)	0
Zeropadding2d_3	(None, 3, 7, 32)	0
Conv2d_3	(None, 1, 5, 128)	36992
Max_pooling2d_3	(None, 1, 2, 64)	0
Zeropadding2d_4	(None, 3, 4, 64)	0
Conv2d_4	(None, 1, 2, 256)	147712
Max_pooling2d_4	(None, 1, 1, 128)	0
Flatten_1	(None, 128)	0
Dense_1	(None, 256)	33024
Dropout_1	(None, 256)	0
Dense_2	(None, 2)	514
Activation_1	(None, 2)	0

Performance results of identifying SNAREs with different filter layers

1   **Table 3.** Performance results of identifying SNAREs with different filter layers

Filters	Sens	Spec	Acc	MCC
32	68.3	91.9	86.6	0.61
32-64	69.9	93.2	88	0.65
32-64-128	73.3	91.5	87.5	0.64
32-64-128-256	70.5	93.3	88.2	0.65

Performance results of identifying SNAREs with different dropout levels

1   **Table 4.** Performance results of identifying SNAREs with different dropout levels

Dropout	Cross-validation				Independent			
	Sens	Spec	Acc	MCC	Sens	Spec	Acc	MCC
0	70.5	93.3	88.2	0.65	57.9	85.7	82.9	0.33
0.1	72.4	94.4	89.5	0.69	44.7	95.4	90.4	0.43
0.2	69.3	93.9	88.4	0.65	50	87.4	83.7	0.3
0.3	69.6	94.2	88.7	0.66	42.1	86	81.7	0.22
0.4	72	92.6	88	0.65	39.5	91.4	86.3	0.29
0.5	69.7	94.8	89.1	0.68	36.8	92.8	87.3	0.29

Comparative performance between 2D CNN and other shallow neural networks

1   **Table 5.** Comparative performance between 2D CNN and other shallow neural networks

Classifier	Sens	Spec	Acc	MCC	Sens	Spec	Acc	MCC
kNN	60.1	95.4	87.5	0.62	28.9	95.1	88.6	0.28
RandomForest	59.6	98.2	89.5	0.68	15.8	98	89.9	0.23
Gaussian	93.5	30.5	44.6	0.23	81.6	23.2	28.9	0.03
SVM	35.2	98.1	84	0.48	28.9	97.1	90.4	0.34
2D CNN	76.6	93.5	89.7	0.7	65.8	90.3	87.9	0.46

Comparative performance between our classification method and BLAST search pipeline

1   **Table 6.** Comparative performance between our classification method and BLAST search
2   pipeline

Method	Cross-validation				Independent			
	Sens	Spec	Acc	MCC	Sens	Spec	Acc	MCC
BLAST	37.0	99.3	85.3	0.53	26.3	99.4	92.2	0.44
2D CNN	76.6	93.5	89.7	0.7	65.8	90.3	87.9	0.46