IEEJ TRANSACTIONS ON ELECTRICAL AND ELECTRONIC ENGINEERING
IEEJ Trans 2021; **16**: 559–567
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI:10.1002/tee.23330

Paper

Deep Protein Subcellular Localization Predictor Enhanced with Transfer Learning of GO Annotation

Xin Yuan*, Non-member
Erli Pang**, Non-member
Kui Lin**, Non-member
Jinglu Hu*[a], Member

Since the large-scale protein sequence data is available, applying deep neural networks to mine better features from the sequences becomes possible. Eukaryotic protein subcellular localization prediction which makes a contribution in many biology process, has used protein sequences in many automatic predicting methods. Moreover, gene ontology (GO) annotation has been shown to be helpful in improving the prediction accuracy of subcellular localization. However, experimentally annotated proteins are not always available. On the other hand, experimentally annotated proteins are available for certain species such as human, mouse, *Arabidopsis thaliana*, etc. It is highly motivated to perform deep learning of GO annotations on the available experimentally annotated proteins and to transfer it to subcellular localization prediction on other species. In this paper, we propose a deep protein subcellular localization predictor, consisting of a linear classifier and a deep feature extractor of convolution neural network (CNN). The deep CNN feature extractor is first shared and pre-trained in a deep GO annotation predictor, and then is transferred to the subcellular localization predictor with fine-tuning using protein localization samples. In this way, we have a deep protein subcellular localization predictor enhanced with transfer learning of GO annotation. The proposed method has good performances on the Swiss-Prot datasets, when transfer learning using the protein samples both within and out species. Moreover, it outperforms the state-of-the-art traditional methods on benchmark datasets.

Keywords: subcellular localization prediction; GO annotation; deep feature extractor; deep neural network; transfer learning

Received 20 May 2020; Revised 17 September 2020

1. Introduction

The prediction of eukaryotic protein subcellular localization plays an important role in many biological processes, as the location information of protein reveals how a cell is working as a basic unit of life [1]. One of the applications is that it is utilized in studying targeted drugs. It has been proved that one protein may appear in multi-locations which makes the prediction more complex. In this situation, the automatic prediction is required. The subcellular location prediction as a multi-label classification task, has been realized by using many machine learning methods like support vector machine (SVM) [2,3], Autoencoder [4], decision trees [5], etc.

Protein sequences are often used as the input feature in machine learning methods, since a large number of protein sequences are becoming available through novel high-throughput sequencing technologies. Considering that a protein sequence consists of 20 different amino acids, which is similar to a natural language [6], the n-gram algorithm is naturally considered as a way to encode proteins into vectors. The n-gram method generated from natural language processing is now developed in the bioinformatics field. A vector of n-gram is defined as the appearance frequency of n consecutive amino acids [7,8]. Another encoding method, the one-hot encoding is also popular in sequence data analysis [9]. For example, in the method of *CONV A-BLSTM* [10], the authors investigated protein sequences dealing with some deep learning models. Protein sequences have been represented as one-hot encoding in units of the 20 amino acids with the sequence length of 1000 (input: $20 * 1000$, the work ignored the sequences larger than 1000). The method consists of four parts, one layer of convolutional neural network (CNN), one hidden state layer and one attention decoding layer of long short-term memory (LSTM), one dense layer of the fully-connected network and the predictor of hierarchical tree likelihood. Comparing with the one-hot coding, the n-gram method takes account of the entire sequences with units of amino acids, dipeptides or more. In this paper, we use a '$1-\text{gram}+2-\text{gram}$' encoding method to extract an appearance frequency as the input feature of a deep CNN model for predicting protein subcellular localization.

When a one-hot or n-gram encoding method is used to vectorize protein sequences, a feature extractor is usually needed to extract and map the feature to space where it is more linearly separable. A neural network, especially deep CNN can be used as such a feature extractor. However, it is inevitable to

[a] Correspondence to: Jinglu Hu. E-mail: jinglu@waseda.jp

*Graduate School of Information, Production and Systems, Waseda University, Kitakyushu-shi, 808-0135, Japan

**College of Life Science, Beijing Normal University, 19 Xinjiekou Outer St, BeiTaiPingZhuang, Haidian Qu, Beijing, 100875, China

use a large dataset for the training, which is difficult for many cases of protein subcellular localization prediction. On the other hand, gene ontology (GO) terms reveal that individual genes contribute to the biology of an organism at the molecular, cellular and organism levels. GO annotation which includes the Cellular Component terms describe parts of cells and structures associated with cells throughout the taxonomy range, if available, is useful genetic information for predicting protein subcellular localization. Cheng et al. [11] and Chou [12] investigated the cases like the proteins of human being, fungus where GO annotation is available, by using sequence feature with annotation feature for localization prediction and found that it is better than using only sequence feature. The works of Refs. (13,14) encoded feature vectors by GO correlation information instead of using the presence or frequency of GO terms. They exploit the hidden correlation between the annotation features of proteins. However, experimentally annotated proteins are not always available for many species. Fortunately, for some species such as human, mouse, *Arabidopsis thaliana*, etc., experimentally annotated proteins are available [15]. It therefore is highly motivated to enhance predicting protein subcellular localization by using a transfer learning of these available experimental GO annotations from various related species.

Since there are available many machine learning methods for GO annotations using protein sequence [16,17], one natural way that may be considered is to build a sequence-feature based prediction model for GO annotation, then apply the predicted GO annotation to enhance predicting protein subcellular localization. However, the error in the GO annotation prediction, especially in the case of transfer learning, may result in poor performance of protein subcellular localization prediction. The similarity of GO annotations plays an important role in many protein studies [18–21]. Considering the similarity between the predictions of protein sequence to GO annotation and to subcellular localization, another way that may be considered is to use the multi-task learning method. The GO annotation predictor and the subcellular localization predictor share a common feature extractor, which improves the accuracy of each predictor by extracting the common features. As the GO annotation information can be used to improve localization prediction, the multi-task learning method is undoubtedly very suitable for this problem. However, the prediction task of GO annotation has more than 3000 labels, while the subcellular localization task has only less than 20 labels, which results in an imbalance problem. The imbalance problem will decrease the performance of the task with fewer labels, which is our target task.

In this paper, we propose a new method of predicting subcellular localization by using a transfer learning of GO annotation. It is structurally similar to the multi-task learning method. The GO annotation predictor and the subcellular localization predictor share a deep CNN feature extractor, but they work on a pre-training and fine-tuning manner. The GO annotation predictor is a hierarchical multi-label model [22–24], consisting of a deep CNN feature extractor and a set of linear multi-label classifiers [25]. It is trained by using a large amount of available experimental GO annotations from various related species. The pre-trained deep CNN feature extractor is then transferred to the subcellular localization predictor. It will be fine-tuned by using a limited size of subcellular localization dataset. In the subcellular localization prediction model, the deep CNN feature extractor plays two roles: extracting features from the vectorized protein sequence and mapping the feature onto a feature space where it is more linearly separable. Therefore, the transfer learning of GO annotation, especially on closely related species, is expected to significantly improve the prediction of subcellular localization. The proposed method is applied to Swiss-Prot datasets and the results are compared with the state-of-the-art methods. Experiment results demonstrate the effectiveness of the proposed method.

The rest of our paper is organized as follows. Section 2 introduces the problem formulation. Section 3 introduces two parallel predictors of the subcellular localization and the GO annotation, sharing a deep CNN feature extractor. Section 4 describes the training processes in a pre-training and fine-tuning manner. Section 5 carries out experiments on a set of Swiss-Prot datasets, and transfer learning experiments compared with different methods. Finally, Section 6 gives conclusions.

2. Problem Formulation

In this section, we first introduce the notations and then formulate the problem.

2.1. Problem and notation Suppose we are given a dataset of protein subcellular localization from a specific target species, $\{x_s(k), y_s(k) | \, k = N_s\}$, where N_s is the size of dataset, $x_s(k)$ is vectorized protein k, and $y_s(k)$ is the subcellular location vector, $y_s(k) = [s_1, \ldots, s_L]^T$, which L is the number of localizations. The problem is to build a multi-label prediction model for predicting protein subcellular localization.

For some species such as human, mouse and *Arabidopsis thaliana*, etc., there are available experimentally annotated proteins. Suppose we collect a large dataset of protein GO annotation from those species, $\{x_g(k), y_g(k) | \, k = N_g\}$, where N_g is the size of dataset, $x_g(k)$ is vectorized protein k, and $y_g(k)$ is the hierarchical GO annotation vector, $y_g(k) = [g^{(1)}(k), \ldots, g^{(q)}(k)]^T$, $g^{(i)}(k) = [g_1^{(i)}(k), \ldots, g_{n_i}^{(i)}(k)]$ $(i = 1, \ldots, q)$ where q is the number of levels, and n_i is the number of label in i level. The Go annotation dataset will be used to help building a subcellular localization prediction model by using transfer learning.

2.2. '1-gram + 2-gram' encoding method Proteins are available in amino acid sequences and we consider a '1-g + 2-g' encoding method to vectorize them. The encoding method encodes a protein sample, an amino acid sequence into an appearance frequency vector which is the column vector with a length of 420, the number of 20 amino acids and 400 dipeptides. For a given protein sequence p_k which contains the subcellular localization samples and GO annotation samples, the vector $t_s(k)$ and $t_g(k)$ represent the appearance time of localization samples and GO annotation samples are first obtained by counting the number of times of each amino acid and dipeptide appearing in the protein sequence p_k. Then the appearance frequency vectors $x_s(k)$ and $x_g(k)$ are defined by

$$x_s(k) = \frac{t_s(k)}{\dim(x_s(k))}, \quad x_g(k) = \frac{t_g(k)}{\dim(x_g(k))} \tag{1}$$

where $\dim(x_s(k)) = \dim(x_g(k)) = 20^1 + 20^2 = 420$. $x_s(k)$, $x_g(k)$ are used as the input vector of localization predictor and annotation predictor respectively.

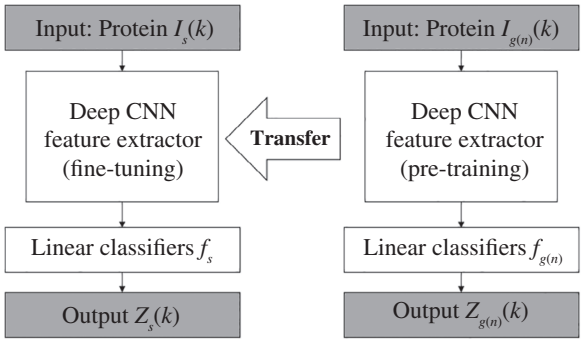


Fig. 1. An image of two parallel predictors of GO annotation and subcellular localization, sharing a deep feature extractor

2.3. Enhanced subcellular localization predictor

When designing protein subcellular localization predictor, we consider two parallel predictors of GO annotation predictor and subcellular localization predictor sharing a powerful deep CNN feature extractor, as shown in Fig. 1, described by

$$Zg(k) = f_g(\phi(Ig(k)) \tag{2}$$

$$Zs(k) = f_s(\phi(Is(k)) \tag{3}$$

where $I_g(k)$, $Z_g(k)$ and $I_g(k)$, $Z_g(k)$ are the input–output vectors of GO annotation predictor (right side of Fig. 1) and subcellular localization predictor (left side of Fig. 1), respectively. Efforts are made to design a powerful deep CNN feature extractor, $\phi(\cdot)$, to extract and map to a feature space where it is more linearly separable. The deep CNN feature extractor is first trained by using a large dataset of protein GO annotation from various species, then it is transferred to subcellular localization predictor where being fine-tuned by using the protein subcellular localization dataset. We will introduce the details of the two predictors in the following sections.

3. Deep CNN Feature Extractor

With a powerful feature extractor, the subcellular localization predictor can be simply designed as a linear multi-label classifier. We consider an ordinal 1-d deep CNN model with fixed input and output size as the feature extractor, defined by

$$\phi(k) = \phi(W, I(k)) \tag{4}$$

where the input vector $I(k) \in R^{420}$ and the output vector $\phi(k) \in R^{N_o}$. In our experiments, $N_o = 512$. Since the deep CNN model will first be trained as feature extractor in the GO annotation predictor using a large dataset from various related species, and then transferred to the subcellular localization predictor with fine-tuning, it therefore is key point to avoid overfitting and to make it sure only extracting common feature.

3.1. The GO annotation predictor

As mentioned in Section 2, labels in the GO annotation problem are organized in q levels. An efficient way to capture the label relations at different levels is to design a hierarchical multi-label classifier, consisting of hierarchically organized $q+1$ nonlinear classifiers. In our previous work, by sharing a deep CNN with the $q+1$ classifiers we designed a deep CNN model with multiple heads and multiple ends (MHME) to implement the $q+1$ classifiers in one deep neural network, and design a sophisticated recursive algorithm to train the MHME CNN model to perform a set of hierarchically organized powerful classifiers. We will briefly summarize as follows. Referred to Ref. (25) for more details.

The MHME CNN model has three parts: the body part, the multi-end part and the multi-head part, which realizes the $q+1$ classifiers with sharing the deep CNN feature extractor as the body part, and represented by:

$$Z_{gn}(k) = f_{gn}(\omega_n, \phi(W, A_n(\lambda_n, I_{gn}(k)))), \tag{5}$$

where $n = 1, 2, \ldots, q+1$, ϕ, A_n and f_n denote the body part, the multi-end part and the multi-head part, and ω_n, W, λ_n are the parameters, respectively. The input vectors $I_{gn}(k)$ and the output vectors $Z_{gn}(k)$ for the $q+1$ classifiers are defined by

$$I_{g1}(k) = x_g(k), Z_{g1}(k) = g^{(1)}(k) \tag{6}$$

$$I_{g2}(k) = [x_g^T(k), \widehat{Z}_{g1}^T(k)]^T, Z_{g2}(k) = [g^{(1)}(k), g^{(2)}(k)]^T$$

$$\ldots$$

$$I_{gq}(k) = [x_g^T(k), \widehat{Z}_{g(q-1)}^T(k)]^T$$
$$Z_{gq}(k) = [g^{(1)}(k), \ldots, g^{(q)}(k)]^T$$

$$I_{g(q+1)}(k) = [x_g^T(k), \widehat{Z}_{gq}^T(k)]^T$$
$$Z_{g(q+1)}(k) = [g^{(1)}(k), \ldots, g^{(q)}(k)]^T$$

where $\widehat{Z}_{gq}(k)$ denotes the prediction value of $Z_{gq}(k)$. And finally $Z_g(k) = Z_{g(q+1)}(k)$ is the output vector of GO annotation predictor.

The multi-end part corresponds to the inputs of the $q+1$ hierarchically related classifiers. As can be seen from the description in (7), the $q+1$ classifiers have input vectors with different lengths. To share the body part for feature extraction, The multi-end part is a set of autoencoder trained to transform the input vectors into a set of feature vectors with the same length so as to share the deep CNN feature extractor

$$\begin{cases} a_1(k) = x_g(k) \\ a_n(k) = A_n(\lambda_n, I_{gn}(k)), \quad n = 2, \ldots, q+1 \end{cases} \tag{7}$$

where $a_n(k)$ is the outputs of multi-end part with $\dim(a_n(k)) = \dim(x_g(k))$.

The multi-head part is a set of linear multi-label classifiers with input vectors from the outputs of the deep CNN featrue extractor. As the linear multi-label classifiers, we simply use a linear network with logistic sigmoid outputs.

$$Z_{gn}(k) = f_{gn}(\omega_n, \phi_n(k)), \quad n = 1, \ldots, q+1 \tag{8}$$

where $\phi_n(k) = \phi(W, I_{gn}(k))$.

Note that by sharing the deep CNN feature extractor with the $q+1$ classifiers in the GO annotation predictor, we are expecting to be able to prevent overfitting and to ensure to extract only common features.

3.2. The subcellular localization predictor As described in Section 2, labels in the subcellular localization problem are organized in only one level. It corresponds to the GO annotation predictor at $q = 1$. By using the deep CNN feature extractor transferred from the deep GO annotation predictor, a hierarchical multi-label classifier can be designed by using a linear network with logistic sigmoid outputs:

$$Z_{s1}(k) = f_{s1}(\Omega_1, \phi(W, x_s(k))) \tag{9}$$

$$Z_s(k) = f_s(\Omega_2, \phi(W, A(\lambda, I_{s1}(k)))) \tag{10}$$

where $I_{s1} = [x_s^T(k), \widehat{Z}_{s1}^T(k)]^T$, $Z_{s1}(k) = y_s(k)$, $Z_s(k) = y_s(k)$, and $\widehat{Z}_{s1}(k)$ denotes the prediction value of $Z_{s1}(k)$.

4. Training Process

The training process consists of two parts: the pre-training and the fine-tuning.

4.1. Pre-training the deep CNN feature extractor

The GO annotation predictor which is a hierarchical multi-label model is represented by:

$$Z_{g1}(k) = f_{g1}(\omega_1, \phi(W, I_{g1}(k))), \;\; I_{g1}(k) = x_g(k) \tag{11}$$

$$Z_{g2}(k) = f_{g2}(\omega_2, \phi(W, A_2(\lambda_2, I_{g2}(k)))), \quad I_{g2}(k) = a_{g1}(k) \tag{12}$$

$$\cdots$$

$$Z_{gq}(k) = f_{gq}(\omega_q, \phi(W, A_q(\lambda_q, I_{gq}(k)))), \quad I_{gq}(k) = a_{g(q-1)}(k) \tag{13}$$

$$Z_{g(q+1)}(k) = f_{g(q+1)}(\omega_{q+1}, \phi(W, A_{q+1}(\lambda_{q+1}, I_{g(q+1)}(k)))), \quad I_{g(q+1)}(k) = a_{g(q)}(k) \tag{14}$$

The deep CNN feature extractor is pre-trained during the training of GO annotation predictor.

- *Step* 1 The training of the parameters ω_1, W from (11): ω_1 is the parameter of classifier f_{g1} and W is the parameter of the deep CNN feature extractor. They are trained in a supervised manner with logistic output by optimizing the sigmoid cross-entropy (SCE) loss function $E = \sum_{i=1}^{n} \varepsilon_{SCE}(\widehat{Z}_{g(n)}(k), Z_{g(n)}(k))$. In this formulation, ε_{SCE} is defined by

$$\varepsilon_{SCE}(\widehat{z}, z) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C_n}[z_{ij} \times log(\widehat{z}_{ij}) + (1 - z_{ij}) \times log(1 - \widehat{z}_{ij})] \tag{15}$$

where N is the number of samples and $C_n = \dim(Z_n)$.

Set the input $I_{g1} = x_g(k)$, and after training we get the predictions $\widehat{Z}_{g1}(k)$ by (11).

- *Step* 2 The training of the parameters λ_2 from (12): λ_2 is the parameter of the autoencoder A_2 which is trained in an unsupervised manner by optimizing the mean squared error (MSE) loss function $E_n = \varepsilon_{MSE}(\widehat{I}_{g(n)}, I_{g(n)})$. Let z be the input of encoder and $\widehat{z}$ be the output of decoder. Then ε_{MSE} is defined by

$$\varepsilon_{MSE}(\widehat{z}, z) = \frac{1}{N}\sum_{i=1}^{N}(\widehat{z}_i - z_i)^2 \tag{16}$$

where N is the number of samples.

Set $I_{g2} = [x_g^T(k), \widehat{Z}_{g1}^T(k)]^T$, and after training the autoencoder we get the output $a_2(k)$.

- *Step* 3 The training of the parameters W, ω_n, λ_n, $(n = 2, 3, \ldots q+1)$: ω_n, λ_n are the parameters of the classifier f_{gn}, the autencoder A_n respectively. Set the input $I_{gn} = a_{gn}(k)$, and train the parameters ω_n, λ_n, W based on (13)–(16), the training process is as the same as the *Steps* 1 and 2. Then repeat *Steps* 2 and 3 until an appropriate stop condition is satisfied.

4.2. Fine-tuning the deep CNN feature extractor

With the pre-trained deep CNN feature extractor $\phi(W,)$, we build the localization predictor represent by (9) and (10). The deep CNN feature extractor is fine-tuned during the training of subcellular localization predictor.

- *Step* 1 The training of the parameters Ω_1, W from (9): Ω_1 is the parameter of classifier f_{s1} and they are trained in a supervised manner with logistic output by optimizing the sigmoid cross-entropy (SCE) loss function $E = \sum_{i=1}^{n} \varepsilon_{SCE}(\widehat{Z}_{s1}(k), Z_{s1}(k))$. Set the input as $x_s(k)$, and train the parameters based on (9) and (15). After training we get the predictions $\widehat{Z}_{s1}(k)$.
- *Step* 2 The training of the parameters Ω_2, λ, W: λ is the parameter of the autoencoder A which is trained in an unsupervised manner by optimizing the mean squared error (MSE) loss function $E_n = \varepsilon_{MSE}(\widehat{I}_{s1}(k), I_{s1}(k))$. Ω is the parameter of the classifier f_s and the training method is the same as the *Step* 1. Set the input $I_{s1} = [x_s^T(k), \widehat{Z}_{s1}^T(k)]^T$, and train the parameters based on (10), (15) and (16) until an appropriate stop condition is satisfied.

5. Experiment Results

In this section, the proposed enhanced subcellular localization predictor is applied to Swiss-Prot datasets. Experiment results are compared with different methods.

5.1. Datasets The proteins on the database Uniprot [26] included reviewed entries named swiss-prot. In the experiments, we create several swiss-prot datasets as shown in Table I. DataSet1 consists of 52 774 protein sequences from eight different species (human, mouse, rat, *Arabidopsis thaliana*, cerevisiae, rice, fruit fly, fungus) with 3281 GO annotations [27] organized in six levels ($q = 6$). The DataSet1 is used for pre-training the deep CNN feature extractor by a transfer learning of a deep GO annotation predictor. DataSet2 consists of 34 300 protein sequence samples also from these eight species with 14

Table I. Swiss-Prot datasets

Dataset	Sequences	GO-terms	locations
DataSet1	52 774	3281	—
DataSet2	34 300	—	14
DataSet3-z	2631	—	14
DataSet3-d	1078	—	14
DataSet3-c	2181	—	14
DataSet3-ch	643	—	14
DeepLoc	14 004	—	10
HumT	379	—	12
Höglund	5959	—	11

subcellular locations: 'Membrane', 'Nucleus', 'Cytoplasm', 'Cell membrane', 'Endoplasmic reticulum', 'Golgi apparatus', 'Mitochondrion', 'Secreted', 'Chromosome', 'Plastid', 'Peroxisome', 'Lysosome', 'cytoskeleton' and 'endosome'. DataSet2 will be divided into 80% of training set, 10% of testing set and 10% of validation set, when building the deep subcellular localization predictor. DataSet3's are a set of datasets consisting of protein sequences from different species: DataSet3-z from 'zebrafish', DataSet3-d from 'dog', DataSet3-c from 'cat', DataSet3-ch from 'chimpanzee', which will be used for testing the performance of transfer learning. Datasets of DeepLoc, HumT and Höglund are three benchmark datasets used in Refs. [10,14,28], which will be used to compare the proposed method with the state-of-the-art methods.

5.2. Evaluation metrics The evaluation metrics for multi-label classification include the multi-label precision (MPrecision), the multi-label recall (MRecall), the multi-label F1-score (MF1-score) [29], the multi-label G-means (MG-means) and the multi-label Accuracy (MAcc). Considering that the multi-label classification method in our work is a combination of a multi-binary classifier for each class, MPre and MRe are the averages for the Precision and Recall of all classes. They correspond to the micro average of the multi-label precision and the multi-label recall. Let us assume that D is a multi-label dataset, and the sample in $|D|$ is (x_i, y_i), $i = 1 \ldots |D|$, $y_i \in \{0, 1\}^m$ is the set of labels. Suppose $z_i \in \{0, 1\}^m$ be the labels predicted by the multi-label classifier for sample x_i. The definitions of the evaluation metrics are given by

$$MPrecision = \frac{1}{m}\sum_{i=1}^{|D|}\frac{| y_i \cap z_i |}{| z_i |} \tag{17}$$

$$MRecall = \frac{1}{m}\sum_{i=1}^{|D|}\frac{| y_i \cap z_i |}{| y_i |} \tag{18}$$

$$MF1 - score = \frac{2 * MPre * MRe}{MPre + MRe} \tag{19}$$

$$MG - means = \sqrt{MPre * MRe} \tag{20}$$

$$MAcc = \frac{1}{m}\sum_{i=1}^{|D|}\frac{| y_i \cap z_i |}{| y_i \cup z_i |} \tag{21}$$

In the multi-label classification, considering that the number of labels is imbalanced, so the result of MF-score and MG-means are more important.

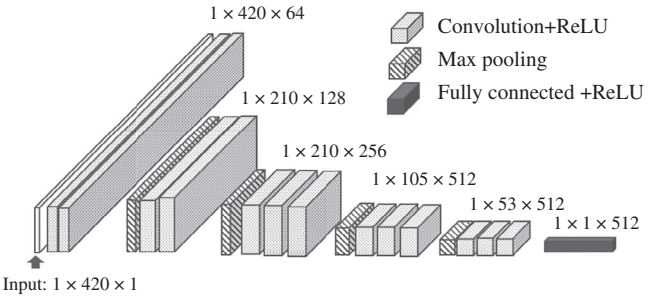


Fig. 2. Structure of the deep CNN feature extractor

5.3. Experiment settings

5.3.1. Details of the deep CNN feature extractor Figure 2 shows the structure of the deep CNN feature extractor with an input vector size of 1 * 420, which is the output of '1 g + 2 g' encoding. This encoding model make each protein sequence to be a heat map of amino acids and dipeptides. The deep CNN feature extractor is designed by referring VGG16 network [30]. It has thirteen 1-d convolution layers with the ReLU activation function and one fully-connected layer with the ReLU activation function. Four max-pooling layers (pool-size: 1*2) are applied to reduce the dimensional and merge features in subregion. The notation such as 1 * 420 * 64 is the input or output size of this layer. We use a batch normalization [31] and dropout in each layer and the dropout with a ratio of 0.4, which is a way to introduce regularization in deep learning reducing interdependent learning [32]. The output of the model is a new feature of protein from the last flatten layer, which is a vector with a size of 1 * 512. The above hyper-parameters of the CNN feature extractor are determined by referring to the VGG-16 network [30], which is known to have a good performance on image recognition.

5.3.2. Details on the hyper-parameters of training algorithms As the deep learning framework, we use Chainer developed by Community, Preferred Networks, Inc. [33]. The training algorithm described in Section 4 is implemented by using the Python programming language. In the experiments, each dataset is divided into three sets: training set, validation set and test set.

Since the deep CNN feature extractor is designed by referring to the VGG16 model, in our experiments, the default values of hyper-parameters suggested by the VGG16 model are used. But some hyper-parameters such as learning rate and stop epochs are determined by using trial and error method. The training of the supervised learning model: $f_{gn}(\omega_n, \phi(W,))$ of the GO annotation predictor and $f_{s1}(\Omega_1\ \phi(W,))$, $f_s(\Omega_2, \phi(W,))$ of the subcellular localization predictor, the initial learning rate used is 0.0075 and training process stops after 120 epochs where the performance is not improved anymore for the validation set. On the other hand, when the unsupervised learning of the autoencoders: $A_n(\lambda_n, \cdot)$ of the GO annotation predictor and $A(\lambda, \cdot)$ of the subcellular localization predictor, the initial learning rate used is 0.05 and the learning was stopped after 30 epochs where the performance for validation set stops improving.

5.4. Comparisons with the state-of-the-art methods

In this subsection, the proposed enhanced subcellular localization predictor (enhanced-SL) is applied to three benchmark datasets of subcellular localization, to compare with the state-of-the-art

methods. The results are also compared with a deep subcellular localization predictor (SL-seq), which is used as a baseline. The deep subcellular localization predictor has the same structure of the proposed model which is trained using only the protein sequences of subcellular localization.

When building the proposed enhanced-SL model, the deep feature extractor is pre-trained using DataSet1, then fine-tuned by using each benchmark dataset. When building the baseline (SL-seq) model, the deep feature extractor is trained only using the benchmark dataset. Table II shows the results of the three benchmark datasets. Two state-of-the-art methods are considered for the comparisons. One is the CONV A-BLSTM model which used multiply deep learning methods (CNN, RNN, LSTM) without GO annotations features [10]. Row 1 in Table II shows the results of the CONV A-BLSTM copied from Refs. [10,28]. The other state-of-the-art method is the Hum-mPLoc3.0 model which is trained using more GO annotations also from swiss-prot (GO: 14737) [14]. Row 2 in Table II shows the results of the Hum-mPLoc3.0 model copied from Refs. [14,28].

For each model, these were calculated from the 30 resampled estimates produced by repeated cross-validation, and Table II shows the results: mean ± standard deviation. Besides the five evaluation metrics, a statistical hypothesis test (Hypothesis test in Table I) is conducted to see if the new models are statistically significant based on the baseline model. And in this case, the statistical significance is represented by the P-value.

Table II shows the results of the baseline (SL-seq) and the proposed enhanced deep subcellular prediction model (enhanced-SL) for the three benchmark datasets: DeepLoc, HumT and Höglund. From the results in Table II, we can see that the proposed enhanced deep subcellular prediction model not only has better performance than the baseline, but also outperforms the two state-of-the-art methods. According to the hypothesis test of the P-values, the proposed enhanced-SL model outperforms the two state-of-the-art methods significantly. In the hypothesis test, the F1-score of SL-seq model is the baseline and the enhanced-SL model performed significantly better than the other two models, which we can find out by the P-values.

Table II. Comparison results with the state-of-the-art methods

Datasets	Methods	Evaluation metrics					Hypothesis test (MF_1)		
		MPrecision	MRecall	MF1-score	MG-mean	MAcc	Z-statistic	P-value	Sig[a]
DeepLoc	CONV A-BLSTM	—	—	—	—	0.7511	baseline(MAcc)		
	SL-seq	0.4669 ± 0.0008	0.5037 ± 0.0036	0.4815 ± 0.0027	0.4844 ± 0.0023	0.8151 ± 0.0109	2.3031	0.014319	**
	Enhanced-SL	0.5012 ± 0.0016	0.5030 ± 0.0071	**0.5020 ± 0.0041**	**0.4902 ± 0.0034**	**0.8724 ± 0.0207**	4.5401	0.000045	***
HumT	Hum-mPLoc3.0	—	—	0.6300	—	0.6500	baseline (MF1-score)		
	SL-seq	0.5187 ± 0.0015	0.6306 ± 0.0038	0.6214 ± 0.0017	0.6237 ± 0.0034	0.9046 ± 0.0055	—		
	Enhanced-SL	0.6717 ± 0.0017	0.6589 ± 0.0054	**0.6589 ± 0.0032**	**0.6605 ± 0.0039**	**0.9095 ± 0.0141**	2.7511	0.005063	***
Höglund	CONV A-BLSTM	—	—	—	—	**0.9138**	—		
	Hum-mPLoc3.0	—	—	0.5900	—	0.6400	baseline (MF1-score)		
	SL-seq	0.4755 ± 0.0015	0.5093 ± 0.0052	0.4739 ± 0.0031	0.4905 ± 0.0049	0.8268 ± 0.0145	—		
	Enhanced-SL	0.6170 ± 0.0007	0.6325 ± 0.0054	**0.6207 ± 0.0029**	**0.6237 ± 0.0041**	0.8903 ± 0.0176	3.0700	0.002307	***

[a] *Significance*: *P-value ≤0.1, **P-value ≤0.05, ***P-value ≤0.01.
Note: The bold values are the best result in each group of the experiments.

5.5. Comparisons with different transfer learning methods In this subsection, we perform experiments on DataSet1 and DataSet2 described in Table I. The target task is to build a multi-label subcellular localization classifier using DataSet2. Same to the way building the proposed enhanced-SL model in Section 5.4, the deep feature extractor is first pre-trained using DataSet1, then fine-tuned by using DataSet2. The proposed method is compared with the other two ways to transfer the learning of GO annotation. One is building a GO annotation predictor, then using the prediction of GO annotation as additional input features (SL-GoP). Another is the multi-task learning method (multi-task), parallelly building a GO annotation and a subcellular localization predictor that sharing a deep feature extractor. Same to Section 5.4, 30 resampled estimates produced by repeated cross-validation, a statistical hypothesis test (Hypothesis test in Table III) is conducted to see if the new models are statistically significant based on the baseline model (the state-of-the-art methods). And the statistical significance is represented by the P-value.

Table III shows a comparison of different ways test on the testing set of DataSet2. The first row of Table III shows the results of the baseline without the transfer learning of GO annotation. We can see that both the multi-task method and the transfer

Table III. Results of different transfer learning methods

Methods	Evaluation metrics					Hypothesis test		
	MPrecision	MRecall	MF1-score	MG-mean	MAcc	Z-statistic	*P*-value	Sig
SL-seq	0.6188 ± 0.0023	0.6276 ± 0.0031	0.6158 ± 0.0028	0.6214 ± 0.0028	0.8847 ± 0.0024	baseline (MF1-score)		
SL-GoP	0.6283 ± 0.0010	0.6330 ± 0.0021	0.6272 ± 0.0019	0.6298 ± 0.0041	0.8800 ± 0.0071	1.4084	0.084822	*
Multi-task	0.6595 ± 0.0083	0.6892 ± 0.0042	0.6727 ± 0.0085	0.6737 ± 0.0021	0.9051 ± 0.0102	3.3235	0.001208	***
Enhanced-SL	**0.6973 ± 0.0057**	**0.7275 ± 0.0014**	**0.7114 ± 0.0256**	**0.7119 ± 0.0113**	**0.9395 ± 0.0316**	3.5869	0.000606	***

Note: The bold values are the best result in each group of the experiments.

learning methods are better than the baseline (SL-seq) which only uses protein sequence of subcellular localization, without transfer learning of GO annotation. The multi-task method works better than the SL-GoP method but it could only apply to the problem in which the protein samples have experimental GO annotations. The SL-GoP method has poor performance maybe due to the overfitting of predicting GO annotations. The proposed enhanced deep subcellular localization prediction model works the best and it could transfer the feature extractor which carries the knowledge of predicting GO annotation from different proteins to the subcellular localization predictor.

5.6. Performance of transfer learning within or out species As shown in Table III, the experiments in Section 5.5 have proved that the learning of GO annotation on some species can be transferred to significantly enhance the subcellular localization predictors of the same species. In this subsection, we show the results of transfer learning out species. That is, the learning of GO annotation on some species is transferred to enhance the subcellular localization predictors of different species. The three prediction models were trained in Section 5.5 using DataSet1 and DataSet2 are directly applied to test on four other datasets from different species without any further training. From the results shown in Table IV, we can see that the enhanced-SL model has better performance than the SL-seq model and the SL-GoP model.

As mentioned before, the deep feature extractor plays two roles in the subcellular localization prediction. The first role is to extract features from the vectorized protein sequence; the second role is to map the features onto a feature space where it is more linearly separable. Therefore, when enhancing the deep feature extractor by using transfer learning, for the first role, any protein sequence-based prediction may be helpful, while for the second role, a related function prediction is required in order to transfer useful knowledge. Since the subcellular localization is related to GO annotation, we employ the transfer learning of GO annotation on eight different species to improve the prediction of subcellular localization. In such a case, the species that are more closely related to the eight species are expected to get more improvement, while the species that are distantly related to the eight species are expected to get less improvement. From the results shown in Table IV, the three different datasets show different improvements demonstrating that the enhanced feature extractor by transfer learning is effective:

- *Distantly related species*: From Table IV, we find that the DataSet-z is improving very less. It is the data of zebrafish which is distantly related to the species of the training data(DataSet1).
- *Closely related species*: The Dataset3-ch is improving more which is the data of Chimpanzee. Considering that the similarity of genes in chimpanzee and human (in the DataSet1), which is proved that the proposed method is more effective to the samples of closely related species.
- *Within species*: From Table III, the DataSet2 is the within species data which has an obvious improvement by the proposed method.

6. Conclusions

In this paper, we have proposed a new subcellular localization predictor with a powerful deep CNN feature extractor

Table IV. Performance of transfer learning from different species

Datasets	Methods	Evaluation metrics				
		MPrecision	MRecall	MF1-score	MG-mean	MAcc
DataSet3-z	SL-seq	0.4674 ± 0.0017	0.5212 ± 0.0051	**0.4914 ± 0.0013**	0.4931 ± 0.0021	0.8774 ± 0.0013
	SL-GoP	0.4688 ± 0.0005	0.5089 ± 0.0013	0.4850 ± 0.0037	0.4877 ± 0.0029	0.8725 ± 0.0013
	Enhanced-SL	0.4752 ± 0.0015	0.5133 ± 0.0022	0.4906 ± 0.0016	**0.4933 ± 0.0014**	**0.8807 ± 0.0022**
DataSet3-d	SL-seq	0.5168 ± 0.0011	0.5598 ± 0.0027	0.5362 ± 0.0015	0.5374 ± 0.0029	0.8902 ± 0.0012
	SL-GoP	0.5007 ± 0.0008	0.5059 ± 0.0016	0.4858 ± 0.0023	0.5032 ± 0.0015	0.8846 ± 0.0013
	Enhanced-SL	0.5116 ± 0.0015	0.5665 ± 0.0026	**0.5363 ± 0.0027**	**0.5378 ± 0.0017**	**0.9052 ± 0.0021**
DataSet3-c	SL-seq	0.4974 ± 0.0004	0.4992 ± 0.0033	0.4874 ± 0.0015	0.4980 ± 0.0018	0.8774 ± 0.0019
	SL-GoP	0.4834 ± 0.0001	0.5321 ± 0.0015	0.5061 ± 0.0021	0.5070 ± 0.0014	0.8871 ± 0.0027
	Enhanced-SL	0.5033 ± 0.0007	0.5390 ± 0.0023	**0.5199 ± 0.0019**	**0.5206 ± 0.0023**	**0.9214 ± 0.0016**
DataSet3-ch	SL-seq	0.5091 ± 0.0009	0.5016 ± 0.0012	0.4726 ± 0.0014	0.5049 ± 0.0011	0.8807 ± 0.0015
	SL-GoP	0.5345 ± 0.0013	0.5909 ± 0.0018	0.5597 ± 0.0016	0.5612 ± 0.0015	0.8824 ± 0.0013
	Enhanced-SL	0.6113 ± 0.0009	0.5788 ± 0.0016	**0.5615 ± 0.0015**	**0.5933 ± 0.0021**	**0.9046 ± 0.0017**

Note: The bold values are the best result in each group of the experiments.

which is enhanced using a transfer learning of GO annotation. The proposed method consists of two main steps: pre-training the deep CNN feature extractor by building a deep GO annotation predictor, and fine-tuning the deep CNN feature extractor when building the subcellular localization predictor. The deep CNN GO annotation predictor consists of a set of hierarchical multi-label classifiers sharing one common deep CNN feature extractor. The deep CNN subcellular localization predictor consists of a multi-label classifier and the deep CNN feature extractor which is transferred from the learning of GO annotation predictor. A large set of experimentally annotated proteins from various species are used to pre-train the deep CNN feature extractor, which then transfers the common feature mapping of annotations to improve the performance of subcellular localization prediction. The proposed method has good performance on the swiss-port datasets when transfer learning using the protein samples both within and out species. Furthermore, it outperforms the state-of-the-art methods on all three benchmark datasets. Especially on dataset DeepLoc our model is 12% enhanced of Accuracy than the CONV A-BLSTM method, and the dataset Höglund 3% enhanced of F1-score than the Hum-mPLoc3.0 method.

In future research, we will improve the transfer learning model with other deep learning models. On the other hand, GO annotations as one of the important features in many biology progress, we will investigate the transfer learning of GO annotation utilized for more areas in bioinformatics prediction.

References

(1) Thul PJ, Åkesson L, Wiking M, Mahdessian D, Geladaki A, Blal HA, Alm T, Asplund A, Björk L, Breckels LM, Bäckström A, Danielsson F, Fagerberg L, Fall J, Gatto L, Gnann C, Hober S, Hjelmare M, Johansson F, Lee S, Lindskog C, Mulder J, Mulvey CM, Nilsson P, Oksvold P, Rockberg J, Schutten R, Schwenk JM, Sivertsson Å, Sjöstedt E, Skogs M, Stadler C, Sullivan DP, Tegel H, Winsnes C, Zhang C, Zwahlen M, Mardinoglu A, Pontén F, von Feilitzen K, Lilley KS, Uhlén M, Lundberg E. A subcellular map of the human proteome. *Science* 2017; **356**(6340):eaal3321.

(2) Hua S, Sun Z. Support vector machine approach for protein subcellular localization prediction. *Bioinformatics* 2001; **17**(8):721–728.

(3) Shi J-Y, Zhang S-W, Pan Q, Cheng Y-M, Xie J. Prediction of protein subcellular localization by support vector machines using multi-scale energy and pseudo amino acid composition. *Amino Acids* 2007; **33**(1):69–74.

(4) Wei L, Ding Y, Su R, Tang J, Zou Q. Prediction of human protein subcellular localization using deep learning. *Journal of Parallel and Distributed Computing* 2018; **117**:212–217.

(5) Lorena AC, de Carvalho AC. Protein cellular localization prediction with support vector machines and decision trees. *Computers in Biology and Medicine* 2007; **37**(2):115–125.

(6) Cao R, Freitas C, Chan L, Sun M, Jiang H, Chen Z. ProLanGO: Protein function prediction using neural machine translation based on a recurrent neural network. *Molecules* 2017; **22**(10):1732–1745.

(7) Zeng J, Li D, Wu Y, Zou Q, Liu X. An empirical study of features fusion techniques for protein-protein interaction prediction. *Current Bioinformatics* 2016; **11**(1):4–12.

(8) Wei L, Xing P, Zeng J, Chen J, Su R, Guo F. Improved prediction of protein–protein interactions using novel negative samples, features, and an ensemble classifier. *Artificial Intelligence in Medicine* 2017; **83**:67–74.

(9) Nauman M, Rehman HU, Politano G, Benso A. Beyond homology transfer: Deep learning for automated annotation of proteins. *Journal of Grid Computing* 2019; **17**(2):225–237.

(10) Almagro Armenteros JJ, Sønderby CK, Sønderby SK, Nielsen H, Winther O. DeepLoc: Prediction of protein subcellular localization using deep learning. *Bioinformatics* 2017; **33**(21):3387–3395.

(11) Cheng X, Xiao X, Chou K-C. pLoc-mEuk: Predict subcellular localization of multi-label eukaryotic proteins by extracting the keyGO information into general pseaac. *Genomics* 2018; **110**(1):50–58.

(12) Chou K-C. Prediction of protein cellular attributes using pseudo-amino acid composition. *Proteins: Structure, Function, and Bioinformatics* 2001; **43**(3):246–255.

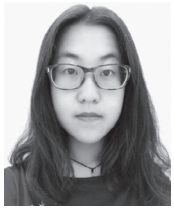
(13) Zhou H, Yang Y, Shen H-B. A new subcellular localization predictor for human proteins considering the correlation of annotation features and protein multi-localization. In *Chinese Conference on Pattern Recognition*. Chengdu, China: Springer; 2016; 499–512.

(14) Zhou H, Yang Y, Shen H-B. Hum-mPLoc 3.0: Prediction enhancement of human protein subcellular localization through modeling the hidden correlations of gene ontology and functional domain features. *Bioinformatics* 2016; **33**(6):843–853.

(15) Consortium GO. Gene ontology consortium: Going forward. *Nucleic Acids Research* 2015; **43**(D1):D1049–D1056.

(16) Kulmanov M, Khan MA, Hoehndorf R. DeepGO: Predicting protein functions from sequence and interactions using a deep ontology-aware classifier. *Bioinformatics* 2017; **34**(4):660–668.

(17) Armean IM, Lilley KS, Trotter MW, Pilkington NC, Holden SB. Co-complex protein membership evaluation using maximum entropy on GO ontology and InterPro annotation. *Bioinformatics* 2018; **34**(11):1884–1892.
(18) Bandyopadhyay S, Mallick K. A new feature vector based on gene ontology terms for protein-protein interaction prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2016; **14**(4):762–770.
(19) Montañez G, Cho Y-R. Predicting false positives of protein-protein interaction data by semantic similarity measures. *Current Bioinformatics* 2013; **8**(3):339–346.
(20) Lei X, Zhao J, Fujita H, Zhang A. Predicting essential proteins based on RNA-Seq, subcellular localization and GO annotation datasets. *Knowledge-Based Systems* 2018; **151**:136–148.
(21) Liu W, Liu J, Rajapakse JC. Gene ontology enrichment improves performances of functional similarity of genes. *Scientific Reports* 2018; **8**(1):12100.
(22) Chen B, Hu J. Hierarchical multi-label classification based on oversampling and hierarchy constraint for gene function prediction. *IEEJ Transactions on Electical and Electronic Engineering* 2012; **7**(2):183–189.
(23) Cerri R, Rodrigo CB, AndrÃl C d C, Jin Y. Reduction strategies for hierarchical multi-label classification in protein function prediction. *BMC Bioinformatics* 2016; **17**(1):373.
(24) Wehrmann J, Cerri R, Barros R. Hierarchical multi-label classification networks. *International Conference on Machine Learning*, 2018, 5225–5234.
(25) Yuan X, Pang E, Lin K, Hu J. Hierarchical multi-label classification for gene ontology annotation using multi-head and multi-end deep CNN model. *IEEJ Transactions on Electical and Electronic Engineering* 2020; **7**:15.
(26) UniProt Consortium. Uniprot: A hub for protein information. *Nucleic Acids Research* 2014; **43**(D1):D204–D212.
(27) GeneOntology Consortium. The gene ontology (GO) database and informatics resource. *Nucleic Acids Research* 2004; **32**(suppl_1):D258–D261.
(28) Höglund A, Dönnes P, Blum T, Adolph H-W, Kohlbacher O. MultiLoc: Prediction of protein subcellular localization using n-terminal targeting sequences, sequence motifs and amino acid composition. *Bioinformatics* 2006; **22**(10):1158–1165.
(29) Tsoumakas G, Ioannis K. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)* 2007; **3**(3):1–13.
(30) Simonyan K, Zisserman A. Very deep convolutional networks for largescale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
(31) Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
(32) Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 2014; **15**(1):1929–1958.
(33) Tokui S, Oono K, Hido S. Clayton J. Chainer: A next-generation open source framework for deep learning. *Proceedings of Workshop on Machine Learning Systems (LearningSys) in the Twenty-Ninth Annual Conference on Neural Information Processing Systems (NIPS)*, vol. **5**, 2015, pp. 1–6.



Xin Yuan (Non-member) received the B.E. degree in Measurement and Control Technology and Instrument from Beijing University of Chemical Technology, China in 2015 and the M.Sci. degree in Waseda University in 2017. Since 2017 September, she is currently working toward the PhD degree in the Graduate School of Information,Production and System, Waseda University. Her research interests include hierarchical multi-label classification, feature extraction and application of bioinformatics.

Erli Pang (Non-member) received the B.Sci. degree and the Ph.D. degree from Beijing Normal University, China. Now she works as an Associate Professor at College of Life, Beijing Normal University. Her research interests are on the areas fo computational molecular biology.

Kui Lin (Non-member) received the B.Sci. degree in 1983, the M.Sci. degree in 1990 and the Ph.D. degree in 1997 from Lanzhou University, China. From 1998 to 2002, he worked as a Research Fellow at BIC, NUS. From 2002 to 2007, he worked as an Associate Professor and Since July 2007, he has been a Professor at College of Life, Beijing Normal University. His research interests are on the areas fo computational molecular biology.

Jinglu Hu (Member) received the M.Sci. degree in Electronic Engineering from Sun Yat-Sen University, China in 1986, and a Ph.D. degree in Computer Science and System Engineering from Kyushu Institute of Technology, Japan in 1997. From 1986 to 1993, he worked as a Research Associate and Lecturer at Sun Yat-Sen University. From 1997 to 2003, he worked as a Research Associate at Kyushu University. From 2003 to 2008, he worked as an Associate Professor and Since April 2008, he has been a Professor at Graduate School of Information, Production and Systems of Waseda University. His research interests include Computational Intelligence such as neural networks and genetic algorithms, and their applications to system modeling and identification, bioinformatics, time series prediction, and so on. Dr. Hu is a member of IEEE, IEEJ, SICE and IEICE.