frontiers in Genetics

ORIGINAL RESEARCH
published: 22 January 2019
doi: 10.3389/fgene.2018.00714

mlDEEPre: Multi-Functional Enzyme Function Prediction With Hierarchical Multi-Label Deep Learning

Zhenzhen Zou[1†], *Shuye Tian*[2†], *Xin Gao*[1] *and Yu Li*[1*]

[1] Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, Computational Bioscience Research Center (CBRC), King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia, [2] Department of Biology, Southern University of Science and Technology (SUSTC), Shenzhen, China

As a great challenge in bioinformatics, enzyme function prediction is a significant step toward designing novel enzymes and diagnosing enzyme-related diseases. Existing studies mainly focus on the mono-functional enzyme function prediction. However, the number of multi-functional enzymes is growing rapidly, which requires novel computational methods to be developed. In this paper, following our previous work, DEEPre, which uses deep learning to annotate mono-functional enzyme's function, we propose a novel method, mlDEEPre, which is designed specifically for predicting the functionalities of multi-functional enzymes. By adopting a novel loss function, associated with the relationship between different labels, and a self-adapted label assigning threshold, mlDEEPre can accurately and efficiently perform multi-functional enzyme prediction. Extensive experiments also show that mlDEEPre can outperform the other methods in predicting whether an enzyme is a mono-functional or a multi-functional enzyme (mono-functional vs. multi-functional), as well as the main class prediction across different criteria. Furthermore, due to the flexibility of mlDEEPre and DEEPre, mlDEEPre can be incorporated into DEEPre seamlessly, which enables the updated DEEPre to handle both mono-functional and multi-functional predictions without human intervention.

Keywords: multi-functional enzyme, function prediction, EC number, deep learning, hierarchical classification, multi-label learning

OPEN ACCESS

Edited by:
Quan Zou,
University of Electronic Science and Technology of China, China

Reviewed by:
Huiluo Cao,
The University of Hong Kong, Hong Kong
Ka-Chun Wong,
City University of Hong Kong, Hong Kong

***Correspondence:**
Yu Li
yu.li@kaust.edu.sa

[†]*These authors have contributed equally to this work*

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Genetics

Received: *13 November 2018*
Accepted: *20 December 2018*
Published: *22 January 2019*

Citation:
Zou Z, Tian S, Gao X and Li Y (2019) mlDEEPre: Multi-Functional Enzyme Function Prediction With Hierarchical Multi-Label Deep Learning. Front. Genet. 9:714. doi: 10.3389/fgene.2018.00714

1. INTRODUCTION

Enzymes, which catalyze reactions *in vivo*, play a vital role in metabolism in every species. Predicting enzyme function is an important bioinformatics task, for helping researchers design more efficient novel enzymes and assisting people in diagnosing enzyme-related diseases (Hoffmann et al., 2007). To predict enzyme function, a clear and standard enzyme function ontology should be defined. Currently, the most popular way of standardizing enzyme function is to use the EC number system (Cornish-Bowden, 2014). An enzyme commission (EC) number is composed of 4 digits, i.e., EC 3.1.21.4, with the first digit denoting the main class of the enzyme; and the second digit indicating the subclass of the enzyme, etc. Each further digit defines the function of an enzyme more specifically, combining with the previous digits. As shown in **Figure 1** in Shen and Chou (2007), the label space of the EC system has a tree structure. As an important bioinformatics task, a number of methods have been proposed to deal with the problem, based on

structure similarity (Dobson and Doig, 2005; Roy et al., 2012; Yang et al., 2015), sequence similarity (Tian et al., 2004; Quester and Schomburg, 2011), or machine learning techniques (des Jardins et al., 1997; Cai et al., 2003; Shen and Chou, 2007; Zhou et al., 2007; Li et al., 2018d). Despite the success of those methods in predicting mono-functional enzyme function with very high accuracy, seldom have people worked on the prediction of multi-functional enzyme function, which actually constitutes a relatively large part of all the enzymes. Until now, to our knowledge, only five methods (De Ferrari et al., 2012; Zou et al., 2013; Che et al., 2016; Zou and Xiao, 2016; Amidi et al., 2017) are able to address that specific type of enzymes. Among them, De Ferrari et al. (2012) use InterPro signatures as the features and multi-label k-nearest neighbor (KNN) as the algorithm. Zou et al. (2013) took advantage of a number of manually designed features, such as a 20-D feature vector extracted from the position-specific scoring matrix (PSSM) and a 188-D feature vector based on the composition and physical-chemical properties of the protein, and the conventional multi-label machine learning algorithm. Che et al. (2016) also utilized features extracted from PSSM, combined with the multi-label KNN algorithm. Zou and Xiao (2016) deployed three variants of the famous feature, Pseudo Amino Acid Composition (PseAAC), and the multi-label KNN algorithm. Amidi et al. (2017) used the predicted structure information, combined with the sequence information, as the feature, and multi-label KNN and multi-label support vector machine (SVM) as the classifier. Despite their satisfactory performance on a specific dataset, we can find that all of those algorithms can be improved in two ways. Firstly, all the methods utilize very specific expert-designed features. It is both time and knowledge consuming to do so, and the extracted features can be local minima for this particular problem (Dai et al., 2017; Li et al., 2018d). The features automatically designed and extracted from the raw representation of a protein by an algorithm are more desirable than the human designed features. Secondly, almost all the aforementioned methods rely on KNN. KNN is a similarity based algorithm. Unlike the probability based algorithms, KNN is unable to annotate novel enzymes which do not have homologs with high sequence similarities in the current databases. Both the feature design process and the classification process have great potential to be improved.

On the other hand, deep learning (LeCun et al., 2015), an end-to-end learning algorithm which wraps the representation learning and classifier learning into one model, has shown great potential in the bioinformatics field (Dai et al., 2017; Li et al., 2018a,d,e; Umarov et al., 2018; Xia et al., 2018), especially in the enzyme function prediction direction (Li et al., 2018d), in which the newly proposed deep learning based method, DEEPre (Li et al., 2018d), has improved the state-of-the-art performance significantly. In general, Li et al. (2018d) built one deep learning model for each of the internal nodes in the tree structure of the EC number system. In particular, for level 0, that is, predicting whether the input protein sequence is an enzyme or not, there is one model; for level 1, whose task is to predict the main class of an enzyme from the six classes, there is one model; for level 2, which predicts the subclass of an enzyme, there are six models since there are six main classes and we need to build a model for each of those different main classes. In terms of the deep learning model architecture, Li et al. (2018d) proposed a novel architecture which can extract convolutional information and sequential information from three raw representations (PSSM, sequence encoding and functional domain encoding) and combine them together automatically for the downstream classification. They only fed the very raw encoding of the protein sequence to the deep learning model and the model is responsible for both feature extraction and classification. In this way, the algorithm is likely to find a better hidden feature representation implicitly, which benefits the classification results.

However, despite the success of DEEPre, the original version of DEEPre was designed specifically for mono-functional enzyme function prediction and not capable of handling multi-functional enzymes. Following the success of DEEPre and its research direction, we propose a novel hierarchical multi-label deep learning method, mlDEEPre, for predicting the multi-functional enzyme functions. In particular, mlDEEPre first predicts whether an enzyme is a mono-functional enzyme or a multi-functional enzyme as a binary classification problem. If the enzyme is a multi-functional enzyme, it will take the input enzyme sequence and predict its main classes as a multi-label prediction problem. To equip the deep learning model with multi-label prediction ability, we adopt the idea of backpropagation for multilabel learning (BP-MLL) (Zhang and Zhou, 2006) into the original DEEPre architecture. Meanwhile, since the entire DEEPre package can also take the main class of the sequence as input and start the prediction from the second level, after obtaining the main classes of the multi-functional enzymes, we can feed the mlDEEPre result to DEEPre, predicting all the four digits for each function of an enzyme. In this work, we make the following contributions:

- We extend the capability of DEEPre from mono-functional enzyme function prediction to multi-functional enzyme function prediction.
- We propose a novel multi-label deep learning framework based on BP-MLL which can be useful for other multi-label prediction problems in bioinformatics.

2. MATERIALS AND METHODS

In this section, we first introduce the dataset used to evaluate the proposed method (section 2.1) and the needed raw encodings we feed to the deep learning model (section 2.2). Then, we provide a big picture of the mlDEEPre method in section 2.3. After that, we introduce the deep learning architecture used in our method (section 2.4). Following the model introduction, we describe how we equip the model with the ability to perform multi-label prediction (section 2.5 and section 2.6). Finally, we wrap up the mlDEEPre method and combine it with the original version of DEEPre (section 2.7).

2.1. Dataset

For the mono-functional enzyme data, we use the dataset from Li et al. (2018d). As for the multi-functional enzyme data, we use the dataset from Che et al. (2016). Li et al. (2018d) constructed

TABLE 1 | Dataset I: 22,168 single-labeled enzymes.

Class	EC 1	EC 2	EC 3	EC 4	EC 5	EC 6
Name	Oxidoreductase	Transferase	Hydrolase	Lyase	Isomerase	Ligase
Number	3343	8517	5917	1532	1193	1666

TABLE 2 | Dataset II: 4,076 multi-labeled enzymes.

Number of classes	2													3			4
EC numbers	1	1	1	1	2	2	2	2	3	3	3	4	4	1	1	1	1
														2	3	4	2
	2	3	4	5	3	4	5	6	4	5	6	5	6	4	6	5	3
																	4
Number	147	841	63	37	1148	235	38	131	622	22	4	308	34	215	10	211	10

This table shows the number of multi-functional enzymes in the dataset with different EC main class combinations.

TABLE 3 | Dataset II: 1,085 multi-labeled enzymes with 65% sequence similarity cut-off.

Multifunctional enzymes	EC 1	EC 2	EC 3	EC 4	EC 5	EC 6	Total
Name	Oxidoreductase	Transferase	Hydrolase	Lyase	Isomerase	Ligase	
Before redundancy	1534	1924	2657	1698	616	179	4076
After CD-HIT	386	503	689	473	137	52	1085

a dataset containing 22,168 sequences from UniProt which have mono-function with 40% sequence similarity filtered by CD-hit. Che et al. (2016) provided us with 4,076 multi-functional enzymes. More detailed descriptions of how to construct the datasets can be referred to Li et al. (2018d) and Che et al. (2016). Here, we provide the statistics of the mono-functional and the multi-functional enzyme datasets in **Tables 1**–**3**.

2.2. Protein Raw Encoding

Similar to DEEPre (Li et al., 2018d), we use the following three raw protein encodings to represent a protein sequence, which will be fed to the deep learning model as inputs.

2.2.1. PSSM

For each protein sequence, we run PSI-BLAST (Altschul et al., 1997) from BLAST+ (Camacho et al., 2009) against SWISS-PROT (Bairoch and Apweiler, 2000), with three iterations and the e-value as 0.002, to find the sequence homologies. Then we align those sequences, and for each position in the query protein, calculate a vector which indicates the appearance frequency of each amino acid in the alignment. The evolutionary information of the protein sequence is encoded by an L by 20 matrix.

2.2.2. Sequence One-Hot Encoding

To represent the original protein sequence information, we use one-hot encoding. For each type of amino acids, we use a vector composed with nineteen 0s and one 1 to represent it. For example, 'A' is represented as $(1, 0_1, ..., 0_{19})$ and 'C' is represented as $(0_1, 1, ..., 0_{19})$. In this way, each position of the protein sequence is encoded into a vector. Putting those vectors together, we have an L by 20 matrix to represent the original raw sequence.

2.2.3. Functional Domain Encoding

This representation encodes the functional domain within a protein sequence. We use HMMER (Eddy, 2011) to search a query protein against Pfam (Finn et al., 2016), which is a functional domain database. If one functional domain is hit, we use 1 to encode it; otherwise, we use 0 to encode it. Consequently, we have a vector composed of 0s and 1s to show the functional domain information of a protein.

2.3. mlDEEPre

The primary task of mlDEEPre is to predict the main classes of multi-functional enzymes. However, we start from predicting whether a query enzyme is a multi-functional enzyme or not. As shown in **Figure 1**, mlDEEPre has two levels. Given an enzyme sequence, the first level predicts whether the enzyme is a mono-functional enzyme or a multi-functional one. If the sequence is a multi-functional enzyme, the second level of mlDEEPre will predict the main classes of the enzyme's multi-functions. The model architecture of the two levels is discussed in section 2.4 and the specific design for multi-label prediction is discussed in sections 2.5 and 2.6. mlDEEPre has very close relationship with the original version of DEEPre, which is discussed in details in section 2.7.

2.4. Model Architecture

Regarding the deep learning model, we use a similar model architecture as in Li et al. (2018d). As shown in **Figure 2**, we adopt

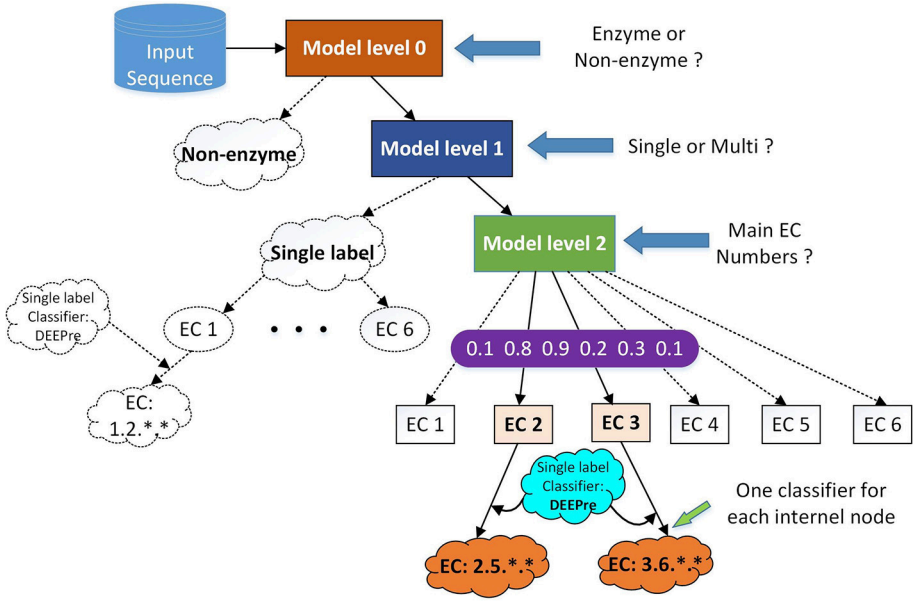

FIGURE 1 | The hierarchical classification strategy combining DEEPre and mlDEEPre. When we input a protein sequence to the system, we first use DEEPre level 0 to predict whether the input sequence is an enzyme or not. If it is an enzyme, we use mlDEEPre level 1 to predict whether the enzyme is a mono-functional enzyme or a multi-functional enzyme. If it is a mono-functional enzyme, DEEPre will take over. If not, we use mlDEEPre to predict that multi-functional enzyme's main classes. Inputting the main classes and the sequence to DEEPre, we can obtain the full annotation for each function of the enzyme.

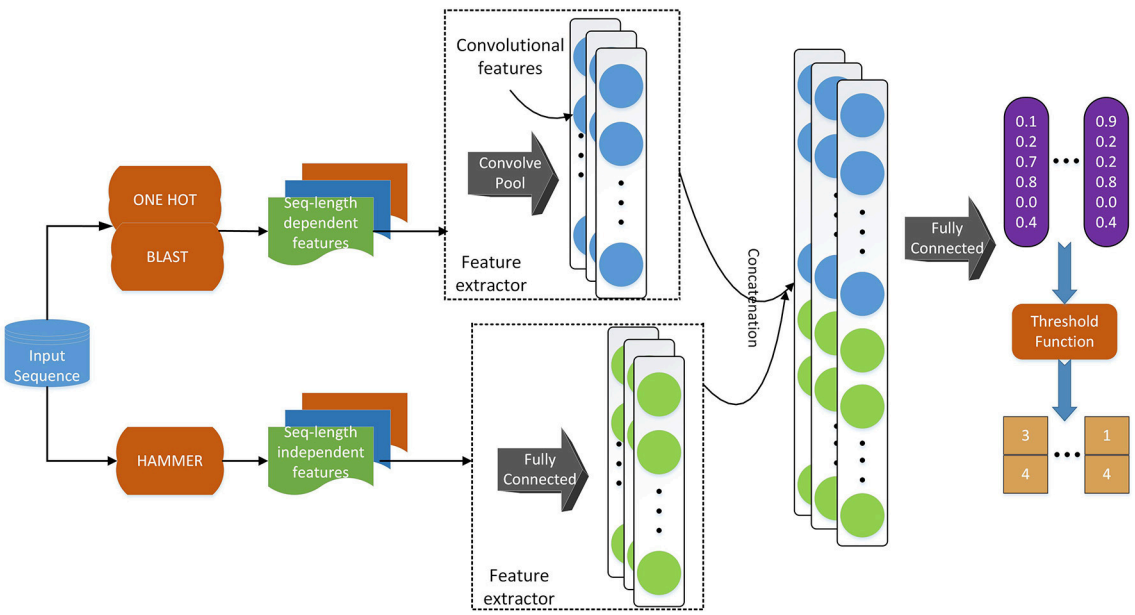

FIGURE 2 | The deep learning model architecture. We use a convolutional neural network component to deal with sequence-length dependent features, such as PSSM and sequence one-hot encoding, and fully connected neural network component to handle functional domain encoding. After those components, we concatenate their outputs into one vector, which is fed to a fully connected classifier. We apply a threshold function to the output of the model to obtain the labels of the input sequence.

convolutional layers for the sequence-length dependent features, i.e., PSSM and sequence one-hot encoding, to extract useful information from those encodings. Since functional domain encoding has already been a high level feature with a fixed length, we use fully-connected layers to reduce the dimensionality and further extract information. After those separated layers

for each feature, we concatenate their outputs and feed the concatenated vector to a fully connected layer, which can be considered as the classifier. Training the model in an end-to-end manner, we are able to optimize the feature extractors (the layers before concatenation) and the classifiers (the final fully connected layers) at the same time, resulting in a better hidden feature representation and thus a classification model with better performance.

2.5. Multi-Label: Loss Function

Deep learning methods are often suitable for multi-label classification. As shown in **Figure 2**, the model's last layer has multiple nodes, whose outputs correspond to the predicted probability of each label. If we use the model to perform single label classification, we will find the label with the highest probability score and assign the query with that label. When we use the model to perform multi-label prediction, we can still use the predicted probabilities. However, we need to change the way of assigning labels. Instead of assigning the label with the highest probability, we may want to assign the labels whose probability score is higher than a certain threshold so that multiple labels can be predicted. On the other hand, when we train the model, we also need to consider the multi-label information in the training data. One of the most straightforward way of incorporating such information when training the model is by changing the loss function to let the model know that we are performing multi-label prediction using a multi-label dataset. In terms of such threshold and loss function, we adopt the idea from Zhang and Zhou (2006), i.e., BP-MLL. We introduce the loss function in this section in details and discuss the threshold in the next section.

Formally, denote the i^{th} enzyme instance as x_i, and its corresponding label vector as D_i. Each element of D_i is a binary value, which indicates whether that enzyme instance belongs to a certain class. We use d_i^j to denote that element, where $j \in [1, 6]$ for our problem. If d_i^j is 1, the enzyme x_i belongs to the class j, 0 otherwise. As for a classification problem, the most intuitive way to define the global error of the network is to measure the distance between the predicted labels and the real labels of the training set:

$$E = \sum_{i=1}^{m} E_i, \tag{1}$$

where E_i represents the network error on the instance x_i and m is the size of the training data. For a multi-label classification problem, we can define E_i as below:

$$E_i = \sum_{j=1}^{Q} (l_i^j - d_i^j)^2, \tag{2}$$

where l_i^j and d_i^j are the actual output and the true label of the network on x_i on the class j, respectively; Q is the total number of classes, which is 6 in our problem. Using Equation 2, we are able to incorporate the multi-label information into the model to a certain degree since all the label information is considered in that loss function. However, the loss function in Equation 2 assumes that each class label is independent, which ignores any relationship between different class labels. In reality, one of the most straightforward relationships between labels is that labels in L_i^{true} should have higher ranks than those not in L_i^{true}, where L_i^{true} is set of labels that the instance x_i has. Accordingly, we can use the following function as the loss which considers the rank relationship between labels:

$$E = \sum_{i=1}^{m} E_i = \sum_{i=1}^{m} \frac{1}{|L_i^{true}||\overline{L_i^{true}}|} \sum_{(k,q) \in L_i^{true} \times \overline{L_i^{true}}} e^{(-(l_i^k - l_i^q))}, \tag{3}$$

where $\overline{L_i^{true}}$ is the complementary set of L_i^{true}, that is, the label set which the instance x_i does not have, and $|\bullet|$ is the cardinality measure of a set. From the equation, we can find that $(l_i^k - l_i^q)$ measures the difference between the outputs of the network on the labels belonging to the training instance and the ones not belonging to it, which is further fed to the exponential function. When l_i^q happens to be much larger than l_i^k, which causes large discrepancy, the exponential function can penalize the error severely. By minimizing Equation 3, we can make the model output much higher values for the true labels while very small values for the labels that the training data do not have. Thus, labels in L_i^{true} have higher ranks than those not in L_i^{true}, which is in agreement with our goal.

2.6. Multi-Label: Threshold

As discussed in the previous section, when we use the model, to determine and assign the labels, there should be a threshold $t(x)$, which is applied to the output of the deep learning model, so that we predict the labels as $L_i^{pred} = \{j | l_i^j > t(x), j \in [1, 6]\}$. A straightforward and natural solution of the threshold function is to set $t(x)$ as a constant. However, that constant threshold does not consider the difference between different data points. To solve the problem, Elisseeff and Weston (2002) proposed an excellent idea to incorporate the information of each single data point into the threshold, which replaces the constant with a linear function $t(x_i) = \mathbf{w}^\intercal \cdot l(x_i) + b$, where $l(x_i)$ is the output of the network on the instance x_i. In this way, each data point can have its own threshold, which is more flexible than a constant. To obtain the threshold function, we need to solve the following problem:

$$t(x_i) = argmin_t(|\{k | k \in L_i^{true}, l_i^k \leqslant t\}| + |\{q | q \in \overline{L_i^{true}}, l_i^q \geqslant t\}|). \tag{4}$$

If the solution of Equation 4 is not unique and the solution composes a segment, the middle value of the value range is chosen as the threshold. For example, assume the real label and predicted label set of x_i are {1,1,0,0,0,0} and {0.9, 0.8, 0.3, 0.1, 0.1, 0.1}, when $0.3 < t < 0.8$, $|\{k | k \in L_i^{true}, l_i^k \leqslant t\}| + |\{q | q \in \overline{L_i^{true}}, l_i^q \geqslant t\}|$ always takes the minimum value as 0. Consequently, we choose the middle value of $(0.3, 0.8)$, which is 0.55, as the threshold. In BP-MLL, the solution of the threshold equation can be obtained through the linear least square method.

To sum up, after we have a well-trained model and the threshold function parameters, and when we need to use the model to perform prediction, firstly, we feed the test instance

to the trained network and get the outputs $l(\mathbf{x})$. Secondly, we calculate the threshold using $t(\mathbf{x}) = \mathbf{w}^{\mathsf{T}} \cdot l(\mathbf{x}) + b$ and apply the threshold to the output of the model, obtaining the predicted labels for the enzyme instance x.

2.7. DEEPre and mlDEEPre

Although DEEPre is designed for mono-functional enzyme function prediction, it is very flexible, being able to predict the detailed function of an enzyme from the first level or the second level. For example, if we have already known that an enzyme has the follow incomplete EC number: 1.-.-.-, we can run DEEPre from the second level to fulfill the missing digits. Taking into consideration the enzyme's feature representation and the fact that the query sequence is an Oxidoreductase, we run the model trained specifically for the enzyme with the first EC digit as 1. With such flexibility, we can easily combine mlDEEPre and DEEPre to predict the detailed functionality of multi-functional enzymes. Using mlDEEPre, we can predict the main classes of those multi-functional enzymes, such as 2.-.-.- and 3.-.-.-. Feeding the sequence and the main classes annotation to DEEPre, we are able to fill in the missing digits for each incomplete annotation of a multi-functional enzyme. The idea of combining DEEPre and mlDEEPre is illustrated in **Figure 1**. Starting from a protein sequence, we first use level 0 of DEEPre to predict whether the protein is an enzyme or not. If yes, we use mlDEEPre first level to predict whether the enzyme is a mono-functional enzyme or multi-functional enzyme. If that is a mono-functional enzyme, we will further run DEEPre to get full annotation of that enzyme. If not, we will run the second level of mlDEEPre to predict the main classes of the enzyme. For each function, we run DEEPre to obtain the full annotation. Considering that most multi-functional enzymes have multiple EC number annotations for its different functions diverging in the first digit, our method is efficient and reliable under most circumstances.

3. RESULTS

In this section, we first briefly introduce the methods with which we are going to compare mlDEEPre (section 3.1). And then, we define the evaluation criteria for the comparison in details in section 3.2. After that, we show the performance of our method in predicting whether an enzyme is a mono-functional enzyme or a multi-functional enzyme in section 3.3. Furthermore, section 3.4 gives the main classes prediction results of multi-functional enzymes. Finally, we show our method's performance on fatty acid synthase (FAS) function prediction in section 3.5.

3.1. Compared Methods

For mono-function prediction, we compared our method with Pse-ACC (Chou and Ho, 2006), ACC (Che et al., 2016), EnzML (De Ferrari et al., 2012), and SVM. Pse-ACC (Chou and Ho, 2006) is a widely used tool, which predicts the enzymatic attribute of proteins by considering the functional domain composition of a given enzyme sequence. In ACC (Che et al., 2016), the authors utilized autocross-covariance (ACC) feature representation which consists of two feature models, autocovariance (AC) and cross-covariance (CC) (Dong et al., 2009). Another compared method here is EnzML, which can efficiently utilize the InterPro signatures. All the above three methods used K-nearest neighbors (KNN) based classification algorithm as the base classifier. We also compared our method with a baseline method, which used SVM as the algorithm and ACC as the features.

For multi-function prediction, as discussed before, until now, there are only a few works focused on this problem, and all of them are based on KNN. The key idea of KNN is that similar instances should share the same labels and we can assign the labels to a query sequence with the most frequent ones from its K-most similar instances, the idea of which is shown in **Figure S1**. We compared our method with ML-KNN (Zhang and Zhou, 2007), BR-KNN (Spyromitros et al., 2008), IBLR-ML (Cheng and Hüllermeier, 2009), GM (Zou and Xiao, 2016), and SVM-NN (Amidi et al., 2017). In ML-KNN (Zhang and Zhou, 2007), for each unseen instance, its K-nearest neighbors in the training set are firstly identified. And then, maximum a posteriori estimation (MAP) principle is applied to determine the label set of the unseen instance based on the statistical information of its neighbor samples. As for Binary Relevance KNN (BR-KNN) (Spyromitros et al., 2008), it learns M binary classifiers, one for each class. In terms of instance-based learning and logistic regression (IBLR) (Cheng and Hüllermeier, 2009), it combines instance-based learning and logistic regression with ML-KNN. For the above three methods, the ACC (Dong et al., 2009) is used as the feature representation. Furthermore, Zou and Xiao (2016) utilized ML-KNN and a different feature extraction model, Grey Model (GM) (Lin et al., 2011), to perform the task. The last method is SVM-NN (Amidi et al., 2017). In this method, the authors combined structural and amino acid sequence information together, investigating two fusion approaches both in the feature level and the algorithm level (SVM and KNN), resulting in a method for general enzymatic function prediction.

3.2. Evaluation Criteria

3.2.1. Single-Label Measurement

Given multi-label and mono-label test datasets $S = \{(x_i, L_i^{true})\}$, the binary classifier performance is evaluated by the four criteria: accuracy, precision, recall, and F1-score, which are defined below:

$$Accuracy = B(TP_j, FP_j, TN_j, FN_j) = \frac{TP_j + TN_j}{TP_j + FP_j + TN_j + FN_j}, \tag{5}$$

$$Precision = B(TP_j, FP_j, TN_j, FN_j) = \frac{TP_j}{TP_j + FP_j}, \tag{6}$$

$$Recall = B(TP_j, FP_j, TN_j, FN_j) = \frac{TP_j}{TP_j + FN_j}, \tag{7}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall}, \tag{8}$$

in which $B(TP_j, FP_j, TN_j, FN_j)$ represents the binary classification indicator; TP_j indicates the number of true positive instance; TN_j is the number of true negative instance; FP_j stands for the number of false positive instances; and FN_j represents the number of false negative instances.

3.2.2. Multi-Label Measurement

Regarding the multi-label classification evaluation, the measurement criteria cannot be exactly the same as those in single-label classification. The assessment method is much more complicated in multi-label learning. The previous works (Chou and Ho, 2006; Zhang and Zhou, 2007) have defined various metrics, including example-based and label-based metrics. For example-based methods, the classification results for each instance are calculated first. After that, the average value for the entire dataset is obtained. For label-based metrics, the binary classification results for each class are calculated first, and then the average value for all classes is given. Here, we adopt both example-based and label-based methods. The metrics that have been utilized to assess the performance of mlDEEPre are described below:

3.2.2.1. Hamming-loss

Hamming-loss evaluates the frequency of incorrect prediction of an instance-label pair. This index is averaged over all classes and the entire dataset. The smaller hamming loss is, the better the performance of the classifier is. It is defined as follows:

$$Hamming-loss = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{6}|L_i^{pred}\Delta L_i^{true}|_1, \tag{9}$$

where Δ is symmetric difference between two sets, $|\bullet|_1$ represents l_1-norm, and N is the number of example enzymes.

3.2.2.2. Subset accuracy

Subset accuracy is the strictest evaluation in multi-label classification. For each sample, the entire set of labels must be correctly predicted, otherwise the subset accuracy for that instance is equal to 0. In literature, it is also known as zero-one-loss:

$$Subset\ accuracy = \frac{1}{N}\sum_{i=1}^{N}\delta(L_i^{pred}, L_i^{true}), \tag{10}$$

where δ is the Kronecker delta:

$$\begin{cases} \delta(L_i^{pred}, L_i^{true}) = 1, & if\ and\ only\ if\ all\ the\ labels\ in\ L_i^{pred} \\ & are\ equal\ to\ those\ in\ L_i^{true}, \\ \delta(L_i^{pred}, L_i^{true}) = 0, & otherwise. \end{cases} \tag{11}$$

Opposite to hamming-loss, and just as the following Macro and Micro methods, the higher the subset accuracy is, the better the performance is.

3.2.2.3. Macro-precision, Macro-recall, Macro-F1-score

Macro-precision, Macro-recall, and Macro-F1-score, which have been used in multi-label classification, calculate precision, recall, and F1-score separately for each class. The Macro-average method is straightforward: just take the average of the precision and recall of the system on different classes. When we want to evaluate system performance on different datasets, macro-averaged metrics are the best choice:

$$Macro-precision = \frac{1}{6}\sum_{j=1}^{6}\frac{TP_j}{TP_j + FP_j}, \tag{12}$$

$$Macro-recall = \frac{1}{6}\sum_{j=1}^{6}\frac{TP_j}{TP_j + FN_j}, \tag{13}$$

$$Macro-F1-score = \frac{2}{6}\sum_{j=1}^{6}\frac{Macro-precision_j \times Macro-recall_j}{Macro-precision_j + Macro-recall_j}. \tag{14}$$

3.2.2.4. Micro-precision, Micro-recall, Micro-F1-score

Regarding Micro-precision, Micro-recall, and Micro-F1-score, in Micro-average methods, we sum up the individual TP, FP, TN, and FN of the system for different sets and then apply them to get the statistics. The Micro-metrics pay more attention to whether the enzymes are correctly classified, regardless their original distribution. Thus, in case of the dataset size being variable, Micro-averaged indexes are the better choice:

$$Micro-precision = \frac{\sum_{j=1}^{6} TP_j}{\sum_{j=1}^{6} TP_j + \sum_{j=1}^{6} FP_j}, \tag{15}$$

$$Micro-recall = \frac{\sum_{j=1}^{6} TP_j}{\sum_{j=1}^{6} TP_j + \sum_{j=1}^{6} FN_j}, \tag{16}$$

$$Micro-F1-score = 2 \cdot \frac{Micro-precision \times Micro-recall}{Micro-precision + Micro-recall}. \tag{17}$$

3.3. Mono-Functional vs. Multi-Functional Prediction

In this section, we describe the performance of the proposed method in predicting whether an enzyme is a mono-functional or multi-function enzyme. The training and testing datasets used in this work are shown in **Tables 1**, **3**. It is worthy pointing out that the data are imbalanced, with 22,168 mono-functional enzymes and 1085 multi-functional enzymes. In this work, we employed penalized models to overcome the imbalance, forcing the model to pay more attention to the multi-functional class. We ran the model 30 times on a GPU node with 32 CPU cores and one GTX 1080 Ti card, each time with 70% of all the data as training data and 30% as testing data. The average training

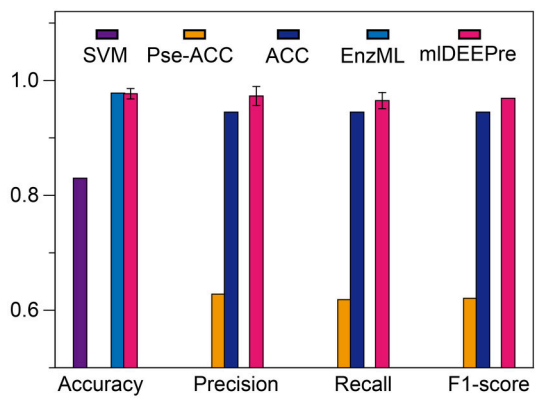

FIGURE 3 | The mono-functional enzyme VS multi-functional enzyme classification testing performance of different models. Performance lower than 0.6 are not shown in the figure.

time is 11 h for 40 epochs and the average testing time for one batch is < 1 min. We show the comparison results, both the average and the standard deviation, in **Figure 3**. As suggested by **Figure 3**, our method can outperform all the other methods consistently across different criteria. Besides, our method is very stable, with the standard deviation of accuracy being as low as 0.09.

3.4. Multi-Functional Enzyme Main Classes Prediction

Similar to the experiments in section 3.3, we also ran the model 30 times on a GPU node with 32 CPU cores and one GTX 1080 Ti card, each time with 70% of all the data as training data and 30% as testing data. The average training time is about 14 h for 40 epochs and the average testing time for one batch is around one minute. Using the criteria for multi-label learning that have been discussed in section 3.2, we evaluated mlDEEPre and compared it with other models introduced in section 3.1, obtaining the performance results shown in **Table 4** and **Figure 4**. According to Hamming-loss, the proposed multi-label model, mlDEEPre, predicts 97.6% of all the actual main classes in the test dataset correctly, with the corresponding standard deviation being 2.7%, which outperforms all the other methods.

TABLE 4 | The multi-functional classification performance of mlDEEPre on dataset II shown in **Table 3**.

Hamming-loss	Subset accuracy	Macro-precision	Macro-recall
3.3 ± 0.4%	82.6 ± 2.7%	96.7 ± 0.3%	96.4 ± 0.6%
Macro-F1	**Micro-precision**	**Micro-recall**	**Micro-F1**
96.5 ± 0.5%	96.7 ± 0.4%	95.1 ± 1.6%	96.2 ± 0.8%

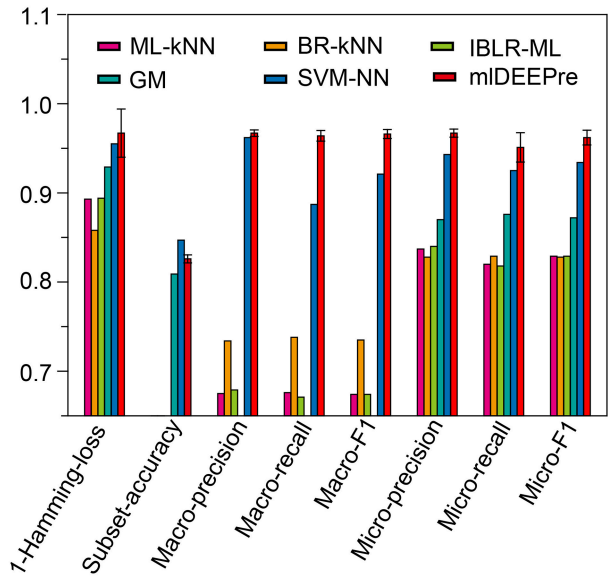

FIGURE 4 | The multi-functional classification testing performance of different models. Performance lower than 0.65 are not shown in the figure.

Furthermore, we also compared our method with the other methods using other criteria. Although, because SVM-NN is good at predicting those rare class labels caused by imbalanced training samples (only 52 sequences belonging to class 6), the performance of SVM-NN (84.7%) is slightly better than that of mlDEEPre (82.6%) and GA (80.8 %) in subset accuracy, mlDEEPre performs better than all the other methods in term of all the other criteria, including Macro-precision, Macro-recall, Macro-F1, Micro-precision, Micro-recall, and Micro-F1.

3.5. Case Study: FAS

Fatty acid synthase (FAS) is a homodimeric multi-functional enzyme that performs the anabolic conversion of dietary carbohydrate or proteins to fatty acids (Chakravarty et al., 2004). Many human cancers can cause high level expression of FAS. Meanwhile, the regulation of human FAS in a variety of cancers makes FAS a candidate target for anticancer therapy (Camassei et al., 2003). FAS subunit alpha includes two parts, reductase and synthase, whose EC number are 1 and 2, respectively. To assess our model's ability in predicting the whole EC number sets of certain multi-functional enzyme sequences, we excluded the FAS sequences from the training data and fed those sequences to our model during testing. The outputs of our network show that mlDEEPre can exactly predict FAS's main classes which are Oxidorreductase and Transferase, being consistent with the experimental results. Furthermore, the integration of mlDEEPre and DEEPre can annotate the two sets of FAS's EC numbers correctly.

4. DISCUSSION

In this paper, based on multi-label deep learning, we propose a novel method, mlDEEPre, to annotate the functionality of multi-functional enzymes. It works seemlessly with DEEPre, which enables DEEPre to perform mono-functional enzyme and multi-functional enzyme function predictions at the same time automatically. Despite of the state-of-the-art performance of mlDEEPre, this tool can still be improved in the following ways. Firstly, when designing the tool, we assume that the multiple functions of an enzyme diverge in the main class. Although that assumption holds under most circumstances, it is inevitable that there are some enzymes with different sub-class or even subsub-class functions. Those kind of enzymes need to be investigated in the future. Secondly, we also assume that the EC system remains static. Although the EC system is stable most of the time, it is a dynamic system if we exam it over a long time period, and the number of classes can increase as we discover more enzymes, which may invalidate the previous classifier. In machine learning, this problem is called class incremental learning (Li et al., 2018b). In the future, efforts will be made to enable the system to perform enzyme function prediction in dynamic labeling space. Finally, since much of the performance gain of mlDEEPre is contributed to the superior performance of deep learning in handling classification problems, some recent works in investigating the nature of deep learning (Soudry et al., 2017; Li et al., 2018c) can be helpful for further improving the performance of deep learning and thus the performance of mlDEEPre. We believe that the idea

of mlDEEPre, combining multi-label learning with deep learning, can be helpful for solving other similar bioinformatics problems. For example, it has the potential to be applied to predict the properties of antibiotic-resistant genes (ARG) in multidrug-resistant pathogens (Zhu et al., 2013; Cao et al., 2018), perform classification of multicomponent transporter system (Saier et al., 2015), and predict CRISPR-Cas9 gene editing off-target regions (Fu et al., 2013; Pattanayak et al., 2013; Lin and Wong, 2018; Zhang et al., 2018).

DATA AVAILABILITY STATEMENT

The datasets for this study can be found in the http://www.cbrc.kaust.edu.sa/DEEPre/dataset.html and http://server.malab.cn/MEC/download.jsp.

AUTHOR CONTRIBUTIONS

YL and XG initialized and designed the project. ZZ and ST implemented the idea and run the experiments. YL and ZZ wrote the manuscript. XG and ST helped revise the manuscript. All authors provided critical feedback and helped shape the research, analysis and manuscript.

FUNDING

This work was supported by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Awards No. FCC/1/1976-17-01, FCC/1/1976-18-01, FCC/1/1976-23-01, FCC/1/1976-25-01, FCC/1/1976-26-01, URF/1/3007-01, and URF/1/3450-01.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2018.00714/full#supplementary-material

REFERENCES

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., et al. (1997). Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.* 25, 3389–3402. doi: 10.1093/nar/25.17.3389

Amidi, S., Amidi, A., Vlachakis, D., Paragios, N., and Zacharaki, E. I. (2017). Automatic single-and multi-label enzymatic function prediction by machine learning. *Peer J.* 5:e3095. doi: 10.7717/peerj.3095

Bairoch, A., and Apweiler, R. (2000). The swiss-prot protein sequence database and its supplement trembl in 2000. *Nucleic Acids Res.* 28, 45–8. doi: 10.1093/nar/28.1.45

Cai, C. Z., Han, L. Y., Ji, Z. L., Chen, X., and Chen, Y. Z. (2003). Svm-prot: Web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Res.* 31, 3692–7.

Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., et al. (2009). Blast+: architecture and applications. *BMC Bioinform.* 10:421. doi: 10.1186/1471-2105-10-421

Camassei, F. D., Cozza, R., Acquaviva, A., Jenkner, A., Rava', L., Gareri, R., et al. (2003). Expression of the lipogenic enzyme fatty acid synthase (fas) in retinoblastoma and its correlation with tumor aggressiveness. *Invest. Ophthalmol. Vis. Sci.* 44:2399. doi: 10.1167/iovs.02-0934

Cao, H., Xia, T., Li, Y., Xu, Z., Bougouffa, S., Lo, Y. K., et al. (2018). *A Multidrug Resistant Clinical P. aeruginosa Isolate in the MLST550 Clonal Complex: Uncoupled Quorum Sensing Modulates the Interplay of Virulence and Resistance.* Cold Spring Harbor Laboratory.

Chakravarty, B., Gu, Z., Chirala, S. S., Wakil, S. J., and Quiocho, F. A. (2004). Human fatty acid synthase: Structure and substrate selectivity of the thioesterase domain. *Proc. Natl. Acad. Sci. U.S.A.* 101, 15567–15572. doi: 10.1073/pnas.0406901101

Che, Y., Ju, Y., Xuan, P., Long, R., and Xing, F. (2016). Identification of multi-functional enzyme with multi-label classifier. *PLoS ONE* 11:e0153503. doi: 10.1371/journal.pone.0153503

Cheng, W., and Hüllermeier, E. (2009). Combining instance-based learning and logistic regression for multilabel classification. *Mach. Learning* 76, 211–225. doi: 10.1007/s10994-009-5127-5

Chou, T.-C., and Ho, K.-C. (2006). Preface. *Biosens. Bioelectr.* 22, 459–460. doi: 10.1016/j.bios.2006.08.034

Cornish-Bowden, A. (2014). Current iubmb recommendations on enzyme nomenclature and kinetics. *Perspect. Sci.* 1, 74–87. doi: 10.1016/j.pisc.2014.02.006

Dai, H., Umarov, R., Kuwahara, H., Li, Y., Song, L., and Gao, X. (2017). Sequence2vec: a novel embedding approach for modeling transcription factor binding affinity landscape. *Bioinformatics* 33, 3575–3583. doi: 10.1093/bioinformatics/btx480

De Ferrari, L., Aitken, S., van Hemert, J., and Goryanin, I. (2012). Enzml: multi-label prediction of enzyme classes using interpro signatures. *BMC Bioinformatics* 13:61. doi: 10.1186/1471-2105-13-61

des Jardins, M., Karp, P. D., and Krummenacker, M. (1997). Prediction of enzyme classification from protein sequence without the use of sequence similarity. *Proc Int Conf Intell Syst . . ..*

Dobson, P. D., and Doig, A. J. (2005). Predicting enzyme class from protein structure without alignments. *J Mol Biol.* 345, 187–99. doi: 10.1016/j.jmb.2004.10.024

Dong, Q., Zhou, S., and Guan, J. (2009). A new taxonomy-based protein fold recognition approach based on autocross-covariance transformation. *Bioinformatics* 25, 2655–2662. doi: 10.1093/bioinformatics/btp500

Eddy, S. R. (2011). Accelerated profile hmm searches. *PLoS Comput Biol.* 7:e1002195. doi: 10.1371/journal.pcbi.1002195

Elisseeff, A., and Weston, J. (2002). "A kernel method for multi-labelled classification," in *Advances in Neural Information Processing Systems 14*, eds T. G.Dietterich, S. Becker, Z. Ghahramani (Vancouver, BC: MIT Press), 681–687.

Finn, R. D., Coggill, P., Eberhardt, R. Y., and Eddy, S. R. (2016). The pfam protein families database: towards a more sustainable future. *Nucleic Acids Res.* 44, D279–D285. doi: 10.1093/nar/gkv1344

Fu, Y., Foden, J. A., Khayter, C., Maeder, M. L., Reyon, D., Joung, J. K., et al. (2013). High-frequency off-target mutagenesis induced by crispr-cas nucleases in human cells. *Nat. Biotechnol.* 31:822.

Hoffmann, B., Beck, M., Sunder-Plassmann, G., Borsini, W., Ricci, R., Mehta, A., et al. (2007). Nature and prevalence of pain in fabry disease and its response to enzyme replacement therapy–a retrospective analysis from the fabry outcome survey. *Clin. J. Pain* 23, 535–542. doi: 10.1097/AJP.0b013e318074c986

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature* 521, 436–444. doi: 10.1038/nature14539

Li, Y., Ding, L., and Gao, X. (2018c). On the decision boundary of deep neural networks. *arXiv [Preprint] arXiv:1808.05385.*

Li, Y., Han, R., Bi, C., Li, M., Wang, S., and Gao, X. (2018a). DeepSimulator: a deep simulator for Nanopore sequencing. *Bioinformatics* 34, 2899–2908. doi: 10.1093/bioinformatics/bty223

Li, Y., Li, Z., Ding, L., Pan, Y., Huang, C., Hu, Y., et al. (2018b). SupportNet: solving catastrophic forgetting in class incremental learning with support data. *arXiv [Preprint] arXiv:1806.02942.*

Li, Y., Wang, S., Umarov, R., Xie, B., Fan, M., Li, L., et al. (2018d). DEEPre: sequence-based enzyme EC number prediction by deep learning. *Bioinformatics* 34, 760–769. doi: 10.1093/bioinformatics/btx680

Li, Y., Xu, F., Zhang, F., Xu, P., Zhang, M., Fan, M., et al. (2018e). Dlbi: deep learning guided bayesian inference for structure reconstruction of super-resolution fluorescence microscopy. *Bioinformatics* 34, i284–i294. doi: 10.1093/bioinformatics/bty241

Lin, J., and Wong, K.-C. (2018). Off-target predictions in crispr-cas9 gene editing using deep learning. *Bioinformatics* 34, i656–i663. doi: 10.1093/bioinformatics/bty554

Lin, W.-Z., Fang, J.-A., Xiao, X., and Chou, K.-C. (2011). idna-prot: Identification of dna binding proteins using random forest with grey model. *PLoS ONE* 6:e24756. doi: 10.1371/journal.pone.0024756

Pattanayak, V., Lin, S., Guilinger, J. P., Ma, E., Doudna, J. A., and Liu, D. R. (2013). High-throughput profiling of off-target dna cleavage reveals rna-programmed cas9 nuclease specificity. *Nat. Biotechnol.* 31:839. doi: 10.1038/nbt.2673

Quester, S., and Schomburg, D. (2011). Enzymedetector: an integrated enzyme function prediction tool and database. *BMC Bioinform.* 12:376. doi: 10.1186/1471-2105-12-376

Roy, A., Yang, J., and Zhang, Y. (2012). Cofactor: an accurate comparative algorithm for structure-based protein function annotation. *Nucleic Acids Res.* 40, W471–W477. doi: 10.1093/nar/gks372

Saier, M. H. Jr, Reddy, V. S., Tsu, B. V., Ahmed, M. S., Li, C., and Moreno-Hagelsieb, G. (2015). The transporter classification database (tcdb): recent advances. *Nucleic Acids Res.* 44, D372–D379. doi: 10.1093/nar/gkv1103

Shen, H. B., and Chou, K. C. (2007). Ezypred: a top-down approach for predicting enzyme functional classes and subclasses. *Biochem. Biophys. Res. Commun.* 364, 53–59. doi: 10.1016/j.bbrc.2007.09.098

Soudry, D., Hoffer, E., and Srebro, N. (2017). The implicit bias of gradient descent on separable data. *arXiv [Preprint] arXiv:1710.10345.*

Spyromitros, E., Tsoumakas, G., and Vlahavas, I. (2008). "An empirical study of lazy multilabel classification algorithms," in *Artificial Intelligence: Theories, Models and Applications*, eds J. Darzentas, G. A. Vouros, S. Vosinakis, and A. Arnellos (Berlin, Heidelberg: Springer), 401–406.

Tian, W., Arakaki, A. K., and Skolnick, J. (2004). Eficaz: a comprehensive approach for accurate genome-scale enzyme function inference. *Nucleic Acids Res.* 32, 6226–6239.

Umarov, R., Kuwahara, H., Li, Y., Gao, X., and Solovyev, V. (2018). PromID: human promoter prediction by deep learning. *arXiv [Preprint] arXiv:1810.01414.*

Xia, Z., Li, Y., Zhang, B., Li, Z., Hu, Y., Chen, W., et al. (2018). DeeReCT-PolyA: a robust and generic deep learning method for PAS identification. *Bioinformatics.* doi: 10.1093/bioinformatics/bty991. [Epub ahead of print].

Yang, J., Yan, R., Roy, A., Xu, D., Poisson, J., and Zhang, Y. (2015). The i-tasser suite: protein structure and function prediction. *Nat Methods* 12, 7–8. doi: 10.1038/nmeth.3213

Zhang, M.-L., and Zhou, Z.-H. (2007). Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recogn.* 40, 2038–2048. doi: 10.1016/j.patcog.2006.612.019

Zhang, M. L., and Zhou, Z. H. (2006). Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Trans. Know. Data Eng.* 18, 1338–1351. doi: 10.1109/Tkde.2006.162

Zhang, S., Li, X., Lin, Q., and Wong, K. C. (2018). Synergizing CRISPR/Cas9 off-target predictions for ensemble insights and practical applications. *Bioinformatics.* doi: 10.1093/bioinformatics/bty748. [Epub ahead of print].

Zhou, X. B., Chen, C., Li, Z. C., and Zou, X. Y. (2007). Using chou's amphiphilic pseudo-amino acid composition and support vector machine for prediction of enzyme subfamily classes. *J. Theor. Biol.* 248, 546–551. doi: 10.1016/j.jtbi.2007.06.001

Zhu, Y.-G., Johnson, T. A., Su, J.-Q., Qiao, M., Guo, G.-X., Stedtfeld, R. D., et al. (2013). Diverse and abundant antibiotic resistance genes in chinese swine farms. *Proc. Natl. Acad. Sci. U.S.A.* 110, 3435–3440. doi: 10.1073/pnas.1222743110

Zou, H.-L., and Xiao, X. (2016). Classifying multifunctional enzymes by incorporating three different models into chou's general pseudo amino acid composition. *J. Membr. Biol.* 249, 551–557. doi: 10.1007/s00232-016-9904-3

Zou, Q., Chen, W., Huang, Y., Liu, X., and Jiang, Y. (2013). Identifying multi-functional enzyme by hierarchical multi-label classifier. *J. Comput. Theor. Nanosci.* 10, 1038–1043. doi: 10.1166/jctn.2013.2804

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.