Knowledge-Based Systems 242 (2022) 108356

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Deep metric learning for accurate protein secondary structure prediction

Wei Yang, Yang Liu, Chunjing Xiao *

Henan Key Laboratory of Big Data Analysis and Processing, Henan Engineering Laboratory of Spatial Information Processing, School of Computer and Information Engineering, Henan University, Kaifeng, 475004, China

ARTICLE INFO

Article history:
Received 14 November 2021
Received in revised form 7 January 2022
Accepted 31 January 2022
Available online 7 February 2022

Keywords:
Protein secondary structure prediction
Deep metric learning
Protein language model
Profile feature
Embedding feature

ABSTRACT

Predicting the secondary structure of a protein from its amino acid sequence alone is a challenging prediction task for each residue in bioinformatics. Recent work has mainly used deep models based on the profile feature derived from multiple sequence alignments to make predictions. However, the existing state-of-the-art predictors usually have higher computational costs due to their large model sizes and complex network architectures. Here, we propose a simple yet effective deep centroid model for sequence-to-sequence secondary structure prediction based on deep metric learning. The proposed model adopts a lightweight embedding network with multibranch topology to map each residue in a protein chain into an embedding space. The goal of embedding learning is to maximize the similarity of each residue to its target centroid while minimizing its similarity to nontarget centroids. By assigning secondary structure types based on the learned centroids, we bypass the need for a time-consuming k-nearest neighbor search. Experimental results on six test sets demonstrate that our method achieves state-of-the-art performance with a simple architecture and smaller model size than existing models. Moreover, we also experimentally show that the embedding feature from the pretrained protein language model ProtT5-XL-U50 is superior to the profile feature in terms of prediction accuracy and feature generation speed. Code and datasets are available at https://github.com/fengtuan/DML_SS.

1. Introduction

Protein secondary structure prediction is a fundamental task in protein science [1]. A protein is a polymer composed of 20 amino acid residue types that can perform many molecular functions, such as catalysis, signal transduction, transportation and molecular recognition. In particular, the function that each protein serves is largely determined by its structure, which in turn is closely related to its amino acid sequence. Therefore, knowing the structure of proteins is extremely important for drug design and disease detection. However, due to the development of advanced sequencing technologies, the gap between the number of known protein sequences and the number of determined protein structures is exponentially widening. Therefore, predicting protein structures with computational approaches is the only practical solution to narrow this gap. As an important subproblem in structural bioinformatics, protein secondary structure prediction can help develop an understanding of the relationship between a protein's function and its amino acid sequence [2]. Moreover, accurately predicted secondary structures are useful for structural class prediction [3], template-based or template-free protein tertiary structure prediction [4], and protein sequence alignment [5].

The secondary structure of a protein refers to local folded conformations that form within a polypeptide due to hydrogen bonding and van der Waals forces. Based on hydrogen bonding patterns and geometric constraints in the atomic coordinates of a protein, the DSSP program [6] assigns each residue to one of eight states: H (α-helix), G (3_{10} helix), I (π-helix), E (extended strand), B (isolated β-strand), T (turn), S (bend) and L (loop or other irregular structure). Among the eight states, the α-helix and extended strand are the two most common regular secondary structure elements, while the π-helix occurs extremely infrequently. In particular, the secondary structure is sensitive to single amino acid residue changes [7]. Moreover, these eight secondary structure states are often reduced to three states: H, G and I are reduced to helix (H); B and E to strand (E); and the other states to coil (C). Therefore, protein secondary structure prediction can be divided into coarse-grained 3-state prediction and fine-grained 8-state prediction. Obviously, the latter can provide more detailed structural information and is thus more challenging.

For a given protein, the goal of secondary structure prediction is to assign a secondary structure type to each amino acid residue in it based only on its primary sequence information. In general, the primary sequence of a protein consists of 21 amino acid

* Corresponding author.
E-mail address: chunjingxiao@gmail.com (C. Xiao).

https://doi.org/10.1016/j.knosys.2022.108356
0950-7051/

abbreviation letters (A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y, and X), where X denotes a nonstandard amino acid type. To use machine learning methods for secondary structure prediction, each amino acid letter needs to be represented as a numeric vector. The simplest method is to use a 21-dimensional orthogonal encoding to represent each amino acid, but this method achieves low prediction accuracies. At present, the feature representations widely used in secondary structure prediction are PSSM profile features, HHM profile features and their hybrid. These profile features contain evolutionary information on residues, which are derived by searching a large-scale protein sequence database for multiple sequence alignments. Moreover, the recently proposed methods SeqVec [8] and TAPE [9] can provide a distributed representation of amino acids, but their reported secondary structure prediction accuracy is significantly lower than that of the profile feature. In short, existing methods mainly combine profile features and deep neural networks for protein secondary structure prediction.

Current state-of-the-art secondary structure deep predictors such as SPOD-1D [10], SAINT [11] and OPUS-TASS [12] are all based on the profile features derived from multiple sequence alignments. However, these predictors usually have a high computational cost due to their large models and complex network architectures. For example, the model sizes of SPOD-1D and OPUS-TASS both exceed 1 GB. To capture the long-range dependence between residues, SAINT further introduces multiple attention modules in the deep inception-inside-inception convolutional network. Note that secondary structure elements are determined by the short- and long-range dependence between residues. For example, the α-helix is determined by the short-range dependence, while the extended strand is determined by the long-range dependence. Therefore, the key to accurately predicting the secondary structure is not to design a complex network architecture specifically to capture the long-range dependence between residues, but to better balance the use of short- and long-range dependencies when performing the prediction. For this reason, we believe that a reasonably designed lightweight model can also obtain state-of-the-art secondary structure prediction accuracy.

In addition, we note that pretrained protein language models such as ESM-1B [13] and ProtTrans [14] have successfully learned useful biophysical features from large protein sequence databases. In particular, the embedding features derived from these pretrained models have shown good performance in multiple tasks, such as subcellular localization and contact prediction. Different from the profile feature, the embedding feature already carries the contextual information of the residues, so it does not require a deeper network to capture the long-range dependence between residues. This means that when using the embedding feature for secondary structure prediction, the existing network architectures for the profile feature should not be used directly. Moreover, it should be noted that the quality of the profile feature of a given protein depends on the number of homologous sequences it has. The fewer the number of homologous sequences, the worse the quality of the corresponding profile feature. In fact, more than 90% of proteins have fewer homologous sequences [15]. For these proteins, the profile feature achieves poor secondary structure prediction results. However, the embedding feature does not have this shortcoming and also avoids performing time-consuming multiple sequence alignment operations. In summary, it is necessary to develop a secondary structure prediction model specifically adapted to the embedding feature.

For the above motivations, we propose a novel sequence-to-sequence protein secondary structure prediction method, the deep centroid model, based on deep metric learning. The proposed model is suitable for both the profile feature and embedding feature and adopts a lightweight embedding network with multibranch topology to map each residue in a protein chain into an embedding space. Specifically, each secondary structure category is assigned a centroid vector in the embedding space. The goal of learning is to maximize the similarity of each residue to its target centroid while minimizing its similarity to nontarget centroids. We design three loss functions, hard margin loss, soft margin loss and softmax loss, to train the proposed model. In the training process, the stochastic gradient descent algorithm is used to simultaneously update the embedding network parameters and the centroid vectors.

The main contributions of this study can be summarized as follows. (1) We propose a simple yet effective deep centroid model for protein secondary structure prediction. To the best of our knowledge, this is the first work that employs a deep metric learning technique to predict secondary structure. (2) We perform extensive experiments on six test sets, and the results demonstrate that our method achieves state-of-the-art performance with much fewer parameters than existing state-of-the-art methods. (3) We experimentally show that the embedding feature is superior to the profile feature in terms of prediction accuracy and computational cost. (4) We demonstrate that ensembling multiple individual models, which are obtained by training the proposed model with different initial model parameter values (containing the initial weight of the embedding network and the initial value of the centroid vectors), can further improve the prediction accuracy of the secondary structure.

The rest of the paper is organized as follows: In Section 2, some related works on protein secondary structure prediction and deep metric learning are reviewed. Section 3 proposes a novel sequence-to-sequence secondary structure prediction method based on deep metric learning. Section 4 presents implementation details, an ablation study, and comparison results. Finally, the conclusion is given in Section 5.

2. Related work

2.1. Protein secondary structure prediction

Protein secondary structure prediction has been widely studied with many machine learning methods, such as neural networks [16,17], decision trees [18], support vector machines [19,20], *k*-nearest neighbors [21,22], and hidden Markov models [23,24]. Early works were mainly based on the use of a fixed-size sliding window to predict the secondary structure type of the central amino acid residue. Representative prediction methods include PSIPRED V3.0 [25], JPred4 [26], and DeepCNF [27]. However, since each residue is treated as an independent sample, such methods usually do not capture long-range interactions between residues well. Later, with the increase in available training data, deep model-based sequence-to-sequence prediction dominated this field and achieved state-of-the-art performance. For example, DCRNN [28] used an end-to-end model with a multiscale convolutional neural network and stacked bidirectional gate recurrent units for sequence-to-sequence secondary structure prediction. Busia and Jaitly [29] developed a multiscale chained convolutional architecture with next-step conditioning for improving 8-state prediction performance. SPIDER2 [30] employed a bidirectional Long Short-Term Memory (LSTM) network to capture long-range interactions between residues. Wang et al. [31] proposed deep recurrent encoder–decoder networks to solve the secondary structure prediction problem. In [32], six different deep network architectures were proposed for protein secondary structure prediction. To enable effective processing of local and global interactions between amino acids, Fang et al. [33] designed a deep

inception-inside-inception network model, named MUFOLD-SS. DeepACLSTM [34] applied the cascaded model of an asymmetric convolutional neural network and a bidirectional LSTM network to perform prediction. Indeed, other variants of the combination of convolutional networks and bidirectional LSTM networks have also been proposed [35]. By ensembling different types of neural networks, SPOT-1D [10] showed substantial improvement in secondary structure prediction. More recently, SAINT [11] utilized a self-attention augmented inception-inside-inception network for 8-state secondary structure prediction. To simultaneously predict protein backbone torsion angles and secondary structure, OPUS-TASS [12] adopted a network architecture that integrated a convolutional network module, a bidirectional LSTM module and a transformer module. In addition, some single-sequence prediction methods, such as SPIDER3-Single [36], SPOT-1D-Single [37], and S4PRED [38], have been proposed for orphan sequences, which have no or few homologous sequences; thus, their profile features cannot provide valuable evolutionary information for prediction.

2.2. Deep metric learning

Deep metric learning has been successfully applied to many vision tasks, including person re-identification [39,40], face recognition [41], and image retrieval [42], and has achieved promising results. The goal of deep metric learning is to learn a deep embedding so that samples with the same labels remain close in the embedding space, while samples with different labels remain far apart from each other. To learn these embeddings, various loss functions, such as contrastive loss [43], triplet loss [44], hierarchical triplet loss [45], margin-based loss [46], N-pair loss [47], angular loss [48], ranked list loss [49], multi-similarity loss [50], tuple margin loss [51], and circle loss [52], have been previously proposed. It should be noted that the large number of redundant sample pairs in a minibatch usually slows down convergence speed and reduces the model capability. Therefore, the key to the success of these loss functions is to calculate the loss based on information pairs. To mine the information pairs, sampling strategies such as semi-hard triplet mining, negative sample mining, distance weighted sampling and pair mining are often used. Moreover, Proxy-NCA [53] proposed using proxies to address the sampling problem. Subsequently, ProxyNCA++ [54] further improved Proxy-NCA by incorporating several enhancements. Existing deep metric learning methods are mainly based on pretrained network models and an additional embedding layer for embedding learning. However, in this study, our embedding network is obtained by training from scratch with randomly initialized parameters. In particular, the proposed method avoids a time-consuming k-nearest neighbor search by performing secondary structure type assignment based on the learned centroids.

3. The proposed method

In this section, we propose a novel sequence-to-sequence protein secondary structure prediction method, the deep centroid model, based on metric learning. The schematic overview of the proposed model is given in Fig. 1. For the kth secondary structure category, let its corresponding centroid in a deep embedding space be $\mathbf{c}^{(k)} \in \mathbb{R}^d$, where d represents the dimension of the embedding space, and $k \in [1, 2, \ldots, c]$, where c is 8 for fine-grained secondary structure prediction and 3 for coarse-grained prediction. Our goal is to learn a nonlinear embedding, which is a deep convolutional network parameterized by Θ, to maximize the similarity between each residue and its target centroid and minimize the similarity between it and all nontarget centroids. In this study, the similarity of two vectors in the embedding space is defined as the dot product between them. For the purpose of training stability, the embedding space is limited to a unit sphere. This means that for any vector $\mathbf{x}$ in the embedding space, we have $\|\mathbf{x}\|_2 = 1$. In particular, unlike the centroid in the nearest centroid classifier, which is obtained by calculating the sample mean of a specific class in the training data, the centroid in our model is the parameter vector that needs to be learned. In the training phase, random gradient descent is adopted to jointly optimize the parameters of the embedding network and all the centroid vectors. In the evaluation stage, each amino acid residue is classified as the class with the most similar centroid. In the following, we will first introduce the designed deep embedding network and then describe the loss function used to train the proposed model.

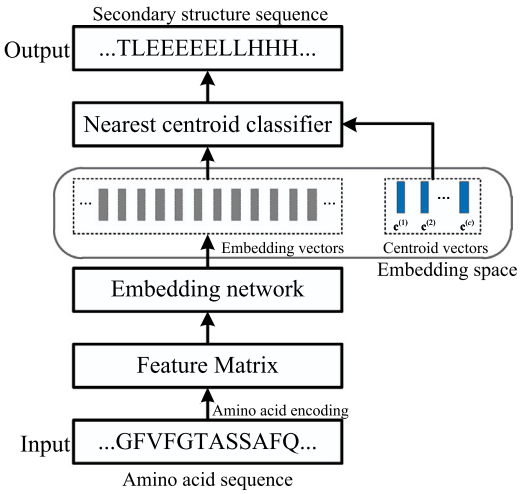

Fig. 1. The proposed deep centroid model for protein secondary structure prediction.

3.1. Deep embedding network

Here we first introduce our designed basic building block. Then, the inception unit is described, which is created by a combination of the basic building block. Finally, we provide the overall architecture of deep embedding network for protein secondary structure prediction.

The basic building block is shown in Fig. 2. Note that $\mathbf{X} \in \mathbb{R}^{N \times M_{\text{in}} \times L}$ is a three-dimensional input tensor, where N is the batch size, M_{in} is the number of input channels, and L is the largest protein chain length in a minibatch. In the building block, the input tensor $\mathbf{X}$ is passed through the first convolutional block, which has a one-dimensional convolution layer (Conv1D) with M filters of kernel size 1, followed by a batch normalization layer (BN) and Hardswish activation function $f(x) = x \cdot \frac{\min(\max(0, x+3), 6)}{6}$ [55]. Here we constrain M to be an even value. Then, the output of the first convolutional block is divided into two parts evenly along the channel dimension by the channel split operator. The content of the first part remains unchanged for feature reuse. For the second part, dropout is applied first to reduce overfitting, and then its output is fed into the second convolutional block. Note that in this convolutional block, the kernel size is 3 and the output channel size is $M/2$. Finally, the depth concatenation operation is performed to merge the output of the two branches, which is followed by another dropout.

Protein secondary structure is determined by both local and nonlocal interactions between amino acid residues. Therefore, to better capture such complex relationships, the embedding network should adopt a multibranch topology to enrich the feature

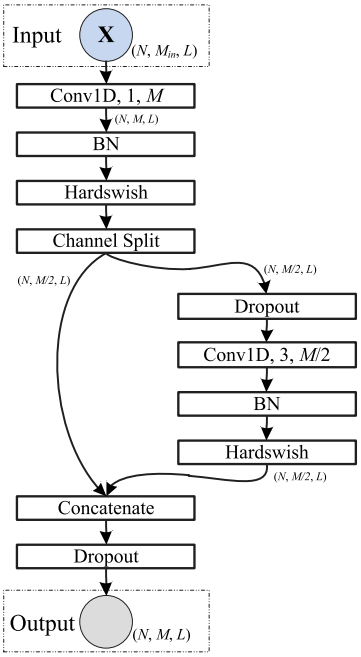

Fig. 2. Illustration of the basic building block.

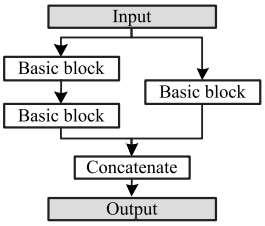

Fig. 3. The Inception unit.

space by fusing information with diverse receptive field sizes. To this end, we further build an inception unit based on the designed basic building block. As illustrated in Fig. 3, the inception unit contains two branches. One branch consists of two consecutively stacked basic building blocks, and the other branch contains only one basic building block. The final output is the depth concatenation of two branches. In practice, we find that two branches are sufficient, and further increasing the number of branches does not further improve the prediction accuracy of the secondary structure.

Fig. 4 shows the overall architecture of the proposed embedding network. It includes n consecutively stacked inception units, followed by a convolutional layer, a ReLU nonlinear activation layer, a reshape layer, a fully connected layer (FC), and a normalization operator. The n consecutively stacked inception units are used to capture the local and nonlocal interactions between amino acid residues. In particular, the first inception unit converts the channel dimension from C_{in} to C, and the subsequent units keep this channel size unchanged, where C_{in} is the encoding length of the amino acid residue. Obviously, the channel size C controls the capacity of our network model. The larger C is, the more parameters the network requires. For the convolutional layer, 96 filters with a kernel size of 9 are used to reduce the channel dimension and increase the receptive field size of the network. The FC determines the size of the final embedding dimension d. For each amino acid residue embedding vector $\mathbf{x}$ in the output of the FC, we further use the normalization operator to project it onto the d-dimensional unit sphere. Specifically, we

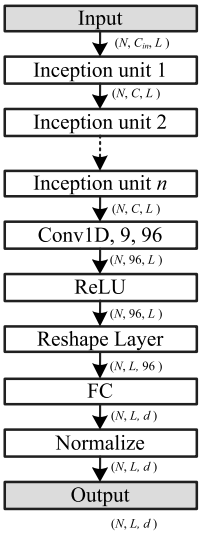

Fig. 4. The framework of our deep embedding network.

first calculate the mean of $\mathbf{x}$, subtract it from each component of $\mathbf{x}$ to obtain the vector $\tilde{\mathbf{x}}$, and finally output $\frac{\tilde{\mathbf{x}}}{\max(\|\tilde{\mathbf{x}}\|_2, \epsilon)}$, where ϵ is the small positive constant to avoid division by zero. In addition, for each centroid vector, we also applied the same normalization operation.

3.2. The loss function

Let a minibatch consist of N protein chains, with a maximum protein chain length of L. Note that different protein chains usually have different numbers of residues. To ensure that all protein chains in a minibatch have input features of the same size, we set the feature vector corresponding to the padding position to a zero vector. In this work, the padding positions were all located at the right end of the shorter protein chain. Moreover, let $\mathbf{Z} \in \mathbb{R}^{N\times L\times d}$ be the output tensor of a minibatch after performing the deep embedding, and its corresponding label matrix and binary mask matrix are $\mathbf{Y} \in \mathbb{R}^{N\times L}$ and $\mathbf{M} \in \mathbb{R}^{N\times L}$, respectively, where $Y_{ij} \in [1, 2, \ldots, c]$ and the entry $M_{ij} = 0$ indicates a padded position (i.e., the jth residue in the ith chain does not exist.).

To train the proposed deep centroid model, we designed three loss functions, hard-margin loss, soft-margin loss and softmax loss. Inspired by the large margin nearest neighbor classification [56], we define the hard margin loss as:

$$\zeta_{\text{HML}} = \frac{1}{\sum_{i=1}^{N}\sum_{j=1}^{L} M_{ij}} \sum_{i=1}^{N}\sum_{j=1}^{L} M_{ij} \left(\sum_{k\neq Y_{ij}} \left[s_{ijk} + \gamma - s_{ijY_{ij}} \right]_+ \right), \tag{1}$$

where the operator $[\bullet]_+ = \max(\bullet, 0)$ is the standard hinge function, s_{ijk} denotes the dot product between $\mathbf{Z}_{i,j,:}$ and $\mathbf{c}^{(k)}$, and γ is a positive hyperparameter that controls the similarity margin in the embedding space. Note that when the similarity of a residue to its target centroid is γ more than its similarity to all other centroids, the hinge function has a negative argument and hence incurs zero loss. In addition, it can be observed that the padded residue positions make no contribution to the overall loss.

The soft-margin loss replaces the standard hinge function in the hard-margin loss with the softplus function. In particular, the softplus function is a smooth approximation of the hinge function. Therefore, the soft-margin loss can be formulated as follows:

$$\zeta_{\text{SML}} = \frac{1}{\sum_{i=1}^{N}\sum_{j=1}^{L} M_{ij}} \sum_{i=1}^{N}\sum_{j=1}^{L} M_{ij} \left(\sum_{k\neq Y_{ij}} u\left(s_{ijk} - s_{ijY_{ij}} \right) \right), \tag{2}$$

where $u(x) = \log(1 + \exp(\beta x))/\beta$ denotes the softplus function, and β is a hyperparameter that can be adjusted.

To introduce the softmax loss, we convert the similarity measure into a probability. For the residue at the jth position in the ith chain, its probability of selecting the kth centroid as the reference point can be defined as:

$$p_{ijk} = \frac{\exp(\tau s_{ijk})}{\sum_{l=1}^{c} \exp(\tau s_{ijl})}, \tag{3}$$

where τ is a scaling parameter. Obviously, we expect to maximize the probability $p_{ijY_{ij}}$. In particular, maximizing $p_{ijY_{ij}}$ also means that nontarget centroids have less chance to be selected as the reference point. To this end, we define the softmax loss as:

$$\zeta_{\text{SoftmaxLoss}} = -\frac{1}{\sum_{i=1}^{N}\sum_{j=1}^{L} M_{ij}} \sum_{i=1}^{N}\sum_{j=1}^{L} M_{ij} \log(p_{ijY_{ij}}). \tag{4}$$

The pseudocode of training the proposed model based on the softmax loss is given in Algorithm 1. In particular, we adopt a validation-based early stopping strategy to avoid overtraining the proposed model.

Algorithm 1 Training of the proposed model

Hyperparameters Setting: the batch size, N; the number of classes, c; the embedding dimension, d; the channel size, C; the number of inception units, n; the dropout ratio, p; the scaling parameter, τ; the learning rate, η;
Input: the training set,$\boldsymbol{T}$; the validation set,$\boldsymbol{V}$;
Output: the learned embedding network parameters and centroid vectors;

```
Randomly initialize the embedding network parameters;
Randomly initialize the c centroid vectors and normalize them into unit vectors;
for each training epoch do
    Shuffling the training set T;
    for each training iteration do
        Sample a minibatch from the shuffled training set to generate a feature tensor X, a label matrix Y, and a binary mask matrix M;
        Feedforward the tensor X into the embedding network to obtain the embedding tensor Z;
        Calculate the softmax loss according to the Eqs. (3) and (4);
        Backpropagation to compute gradient and update the centroid vectors and network parameters;
    end for
    Calculate an evaluation accuracy based on the current model parameters and the validation set V;
    if early stopping criterion is satisfied then
        return The model parameters with the best evaluation accuracy;
    end if
end for
```

4. Experiments

4.1. Datasets

To evaluate the proposed method, we conduct experiments on six publicly available test sets: CASP12, CASP13, CASP14, CB513, TEST2016, and TEST2018. Table 1 summarizes the detailed dataset statistics. The first three datasets are derived from the Protein Structure Prediction Center.[1] The Critical Assessment of Structure Prediction (CASP) on the website holds a community-wide competition every two years to determine state-of-the-art techniques in modeling protein structure from amino acid sequences. Since 1994, the CASP has held 14 sessions, and its released target protein structure prediction lists have been widely used as benchmark test sets for protein structure prediction. In this study, we use the three recently released target lists, CASP12, CASP13, and CASP14. For each target list, PDB IDs are first downloaded from the website, and then the corresponding primary and secondary structure sequences are extracted from the FASTA format file ss.txt[2] provided by the RCSB PDB according to the PDB IDs. Finally, the obtained sequences are further filtered by deleting protein chains with > 25% sequence identity. CB513 is a well-known benchmark dataset constructed by Cuff and Barton that has been used to evaluate many protein secondary structure predictors. In addition, TEST2016 and TEST2018 are constructed by [10] and can be obtained from the website.[3] TEST2016 is composed of 1213 proteins deposited on the PDB database from June 2015 to February 2017. TEST2018 consists of 250 high-quality proteins with resolution < 2.5 Åand R-free < 0.25 that were deposited on the PDB database from January 2018 to July 2018. In particular, the maximum protein chain length present in both datasets did not exceed 700.

Table 1
Statistics for the six test sets. #Chains = Number of protein chains; #Residues = Number of residues; #Max = Maximum chain length; #Min = Minimum chain length.

Dataset	#Chains	#Residues	#Max	#Min
CASP12	47	13718	1494	75
CASP13	41	12217	756	41
CASP14	33	9049	2194	19
CB513	513	84119	754	20
TEST2016	1213	287877	699	30
TEST2018	250	56654	615	31

For the first four test sets, CASP12, CASP13, CASP14, and CB513, we use a large nonhomologous dataset, cullpdb_pc25_res2.5_R1.0_d200416_chains13482, downloaded from the PISCES server [57] to construct their corresponding training set and validation set. This dataset contains 13,482 protein chains with resolution < 2.5 Å, R-value < 1.0, and pairwise percent sequence identity < 25%. The secondary structure sequence of each chain is extracted from the ss.txt file. In particular, if the file dose not contain corresponding structural information for the chain, it will be deleted. Moreover, the protein chains with less than 50 or more than 800 amino acids are also removed. The final dataset contains 12,510 protein chains. To ensure that this dataset does not have homology with each test set, it is further filtered by deleting sequences with greater than 25% sequence identity with each test set. After filtering, the numbers of protein chains corresponding to the four test sets CASP12, CASP13, CASP14, and CB513 are 12 376, 12 453, 12 504, and 12 012, respectively. For each of these filtered protein chains, we randomly select 512 protein chains as the validation set and the remaining protein chains are taken as the training set. Furthermore, for the latter two test sets, TEST2016 and TEST2018, we use their default training set (10,029 protein chains) and validation set (983 protein chains). It should be noted that the redundancy between the two test sets and their corresponding training and validation sets was removed by using BlastClust with a 25% sequence identity cutoff.

[1] https://www.predictioncenter.org/.

[2] https://cdn.rcsb.org/etl/kabschSander/ss.txt.gz.

[3] https://servers.sparks-lab.org/downloads/SPOT-1D-dataset.tar.gz.

4.2. Amino acid encoding

To represent each amino acid residue in a given protein chain as a numeric vector, we use two amino acid encodings: profile-based hybrid encoding and embedding encoding. As in [10,11], our profile-based hybrid encoding consists of seven physicochemical properties, PSSM profile features and HHM profile features. The seven physicochemical properties of each amino acid include sheet probability, helix probability, isoelectric point, hydrophobicity, van der Waals volume, polarizability and graph shape index. These property values can be considered position-independent features, which can be obtained from [58]. For a protein chain of length L, its PSSM profile feature and HHM profile feature are matrices of size $L \times 20$ and $L \times 30$, respectively. In this study, each PSSM profile matrix is generated by performing the PSI-BLAST program against the Uniref50 database (updated in May 2020) with E-value = 0.001 and two iterations. Each HHM profile matrix is generated by four iterations of HHblits v3.3.0 with default parameters against the database uniprot20_2013_03, which can be downloaded from http://wwwuser.gwdg.de/~compbiol/data/hhsuite/databases/hhsuite_dbs/old-releases/. The two profile features are position-dependent, and they often contain evolutionary information derived from sequence homologs. In particular, this evolutionary information can provide useful information for accurately predicting protein secondary structure. However, it should be noted that for some orphan sequences or low-homology sequences, their profile features can only provide limited information for prediction because there is no available evolutionary information. This means that the secondary structure of low-homology proteins is more difficult to predict accurately. In summary, the feature size of each amino acid residue in the hybrid encoding is 57. To prevent the data scale from affecting the network training, we perform data normalization on the features of the hybrid encoding. Specifically, we first calculate the mean and standard deviation of each feature based on the training set and then transform the features of all data into a distribution with a mean of 0 and a standard deviation of 1.

Based on the massive protein sequence databases BFD, UniRef50 and UniRef100, recent works, including ESM-1b [13] and ProTrans [14], have successfully learned useful information from sequence data, which can be transferred to protein function or structure prediction tasks, by using unsupervised learning to train advanced language models such as BERT [59] and T5 [60]. In particular, the output embedding of the pretrained language model contains the biophysical properties of the protein sequence. In this study, we use the output of the encoder of the pretrained model ProtT5-XL-U50 [14] as the embedding feature. ProtT5-XL-U50 is a transformer-based language model with 3 billion parameters, which is first trained on the BFD database and then fine-tuned on the UniRef50 database. During pretraining, it uses the AdaFactor optimizer with an "inverse square root" learning rate schedule for optimization. To obtain the embedding feature of a given protein chain, we use its primary sequence as the input of ProtT5-XL-U50 and then run the model in half-precision to obtain the output of the encoder. In the output embedding, the feature dimension corresponding to each residue is 1024. Obviously, the embedding feature is also position-dependent since it captures the contextual information of each residue.

4.3. Evaluation metrics

To evaluate the performance of the proposed method, we adopt two widely used measurements, per-residue accuracy and segment overlap measure (SOV) [61]. Let S be a set of the secondary structure conformational states, where $S = \{H, G, I, E, B, T, S, L\}$ for 8-state prediction and $S = \{H, E, C\}$ for 3-state prediction. The per-residue prediction accuracy of state s is defined as:

$$Q_s = 100 \times \frac{n_s}{N_s}, s \in S, \tag{5}$$

where n_s is the total number of residues that are correctly predicted to be state s, N_s is the total number of residues that are of state s. The overall per-residue prediction accuracy is defined as:

$$Q_{|S|} = 100 \times \frac{\sum_{s \in S} n_s}{\sum_{s \in S} N_s}, \tag{6}$$

where $|S|$ denotes the number of states in the set S.

Another metric, segment overlap measure (SOV), is calculated using secondary structure segments rather than individual residues, and is thus a structurally more meaningful metric. To calculate SOV, all observed and predicted secondary structure sequences need to be divided into segments. In particular, the divided segments need to satisfy two constraints: (1) Only one conformational state can occur in a segment. (2) Segments adjacent to each other in position cannot be in the same conformational state. For a given protein chain, let $O = \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_m\}$ be the set of its observed segments, and $P = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ be the set of its predicted segments. For each state s, let $\tilde{\Omega}_s$ be the set of all the overlapping pairs of segments $(\mathbf{o}, \mathbf{p})$, where $\mathbf{o} \in O$ and $\mathbf{p} \in P$ are both in state s and have at least one residue overlapping. If $\mathbf{o} \in O$ is a segment in state s, and there are no overlapping segments in the set P that are in the same state, then the set consisting of all these $\mathbf{o}$'s is called $\tilde{\Psi}_s$. Moreover, let Ω_s and Ψ_s be the union of all $\tilde{\Omega}_s$ and $\tilde{\Psi}_s$ derived from a dataset to be evaluated, respectively. Then, the overall SOV can be defined as follows:

$$SOV_{|S|} = 100 \times \frac{\sum_{s \in S} \sum_{(\mathbf{o},\mathbf{p}) \in \Omega_s} \left[\frac{\text{minov}(\mathbf{o},\mathbf{p}) + \Delta(\mathbf{o},\mathbf{p})}{\text{maxov}(\mathbf{o},\mathbf{p})} \times l(\mathbf{o}) \right]}{\sum_{s \in S} \left[\sum_{(\mathbf{o},\mathbf{p}) \in \Omega_s} l(\mathbf{o}) + \sum_{\mathbf{o} \in \Psi_s} l(\mathbf{o}) \right]}, \tag{7}$$

where $l(\mathbf{o})$ is the length of segment $\mathbf{o}$, minov$(\mathbf{o}, \mathbf{p})$ is the length of the actual overlap between $\mathbf{o}$ and $\mathbf{p}$, maxov$(\mathbf{o}, \mathbf{p})$ is the length of the total span of $\mathbf{o}$ and $\mathbf{p}$, and $\Delta(\mathbf{o}, \mathbf{p})$ is defined as the minimum of four integers maxov$(\mathbf{o}, \mathbf{p})$ − minov$(\mathbf{o}, \mathbf{p})$, minov$(\mathbf{o}, \mathbf{p})$, $\lfloor l(\mathbf{o}) \rfloor$, and $\lfloor l(\mathbf{p}) \rfloor$.

4.4. Implementation details

We utilize the PyTorch framework throughout all the experiments. During training, the minibatch size is set to 32. In particular, we only use random sampling to construct a minibatch and do not use any sample mining strategy. The AdamW optimizer with a weight decay of 0.001 is utilized for model optimization. The learning rate is set to 0.001 and is kept constant. Moreover, the early stopping criterion is satisfied when the overall per-residue accuracy on the validation set does not increase for 8 consecutive epochs. We performed our experiments with an NVIDIA GTX TITAN X GPU, Intel i7-8700K CPU, and 32 GB memory.

Additionally, it should be noted that the two amino acid encodings have significantly different feature dimensions, and the embedding encoding captures the contextual information of each residue. To this end, we use different parameter settings to construct the embedding network. Specifically, for profile-based hybrid encoding, a deeper embedding network is used to capture the long-range interaction between amino acid residues. The number of inception units n is set to 5, and the channel size C is set to 448. For the embedding encoding, we set n to 2 and C to 1024. In practice, we found that a deeper network does not improve the prediction accuracy of the secondary structure. However, for the embedding dimension d, we adopt the same setting for the two encodings and set it to 32. In the ablation study, we will analyze the influence of these parameters on the prediction performance of the proposed method. Moreover, the dropout

Table 2
The eight-class prediction results of the proposed method under different loss functions on the CB513 test set.

Loss function	Hyperparameter	Hybrid feature		Embedding feature	
		Q_8	SOV_8	Q_8	SOV_8
Hard margin loss	$\gamma = 0.01$	73.74	71.41	74.98	72.77
	$\gamma = 0.04$	74	71.33	75.38	73.04
	$\gamma = 0.07$	74	71.52	75.19	72.81
	$\gamma = 0.1$	73.91	71.42	75.46	72.99
	$\gamma = 0.13$	74.17	71.67	75.37	73
	$\gamma = 0.16$	73.88	71.6	75.24	72.63
	$\gamma = 0.19$	73.94	71.22	75.3	72.84
	$\gamma = 0.22$	74.13	71.58	75.33	72.67
Soft-margin loss	$\beta = 5$	74.11	71.35	75.2	72.8
	$\beta = 10$	73.93	71.55	75.29	72.56
	$\beta = 15$	74.08	71.53	74.99	72.66
	$\beta = 20$	74.14	71.83	75.14	72.69
	$\beta = 25$	73.51	70.98	75.21	72.45
	$\beta = 30$	74.1	71.7	75.18	72.63
	$\beta = 35$	73.93	71.13	75.16	72.75
	$\beta = 40$	73.64	71.1	75.09	72.79
Softmax loss	$\tau = 2$	74.34	71.59	75.34	72.86
	$\tau = 6$	74.29	71.95	75.48	73.14
	$\tau = 10$	74.27	71.48	75.57	72.87
	$\tau = 14$	74.14	71.46	75.56	72.98
	$\tau = 18$	74.49	71.7	75.54	73.22
	$\tau = 22$	74.43	71.87	75.52	73.14
	$\tau = 26$	74.52	72.16	75.5	73.24
	$\tau = 30$	74.14	71.58	75.58	73.29

ratio in the basic building block is set to 0.2 in all experiments. Considering the high dimensionality of the embedding feature, we applied dropout at a ratio of 0.5 during training. For the profile-based feature, we did not use dropout. In particular, for the convenience of comparison, we named the proposed method DML_SS. Our source code, datasets and models are available at https://github.com/fengtuan/DML_SS.

4.5. Ablation study

To evaluate the proposed method, we conduct extensive experiments on test set CB513 and its corresponding validation set to analyze the impact of loss functions, network hyperparameters and amino acid encoding strategies.

4.5.1. Impact of loss functions

In this study, we implement three loss functions, hard margin loss, soft-margin loss and softmax loss, to train the proposed model. It should be noted that each loss function has a hyperparameter. To explore the impact of the loss functions on the performance of the proposed model, we perform comparative experiments on the CB513 dataset based on both the profile-based hybrid feature and embedding feature. The eight-class prediction results of the proposed method under the three loss functions are shown in Table 2. Here, we use the overall metrics Q_8 and SOV_8 as evaluation criteria. It can be observed that the softmax loss outperforms the hard margin loss and soft-margin loss in most cases. Note that the softmax loss can also provide predictive probabilities. Therefore, we choose it as the default loss function. In particular, considering that the optimal hyperparameters corresponding to the two encoding features are different, we simply set the scaling parameter τ to 18 in the subsequent experiments.

In addition, considering that jointly optimizing multiple loss functions can improve the performance of deep metric learning methods, we further analyze the impact of combined losses on the prediction accuracy of the proposed model. To train the proposed model using multiple loss functions, we defined the combined loss as: $\zeta = \mu\zeta_{\text{SoftmaxLoss}} + (1-\mu)\zeta_{\text{Triplet}}$, where $\mu \in [0, 1]$ is a balanced hyperparameter, $\zeta_{\text{SoftmaxLoss}}$ is our softmax loss, and ζ_{Triplet} is the weighed triplet loss defined in [62], which adopts the cosine similarity distance to measure the distance between two residues. In particular, compared to the original triplet loss, the weighted triplet loss not only avoids introducing an additional margin hyperparameter but also achieves better performance in person re-identification [62]. Note that when $\mu = 1$, the combined loss degenerates into our softmax loss. The experimental results of the proposed method under the combined loss with different hyperparameter values are given in Table 3. As can be seen from the table, the combined loss slightly outperforms the softmax loss on the profile-based hybrid feature only when $\mu = 0.7$ or $\mu = 0.9$. For all other cases, the combined loss is consistently inferior to the softmax loss. This is obviously different from person re-identification, where training strategy based on combined losses typically achieve consistent performance gain. There are two possible reasons for this situation: (1) The task types of the two are different. The protein secondary structure prediction task contains few categories (3 or 8) and each category contains many samples. Person re-identification tasks contain many categories (usually more than 600) and each category contains few samples (usually less than 50). (2) For protein secondary structure prediction, each minibatch is constructed by randomly sampling of protein chains. In our study, each minibatch contains about 5000 to 8000 residues, and the distribution of classes in a minibatch is imbalanced. However, for person re-identification, each minibatch is constructed by first randomly

Table 3
The eight-class prediction results of the proposed method under the combined loss on the CB513 test set.

Hyperparameter	Hybrid feature		Embedding feature	
	Q_8	SOV_8	Q_8	SOV_8
$\mu = 0.1$	73.3	70.13	75.07	72.68
$\mu = 0.2$	73.76	71.2	75.16	72.69
$\mu = 0.3$	73.84	71.19	75.22	72.4
$\mu = 0.4$	74.4	72.01	75.35	72.88
$\mu = 0.5$	74.05	71.7	75.38	72.89
$\mu = 0.6$	73.97	71.56	75.3	72.67
$\mu = 0.7$	74.58	72.21	75.32	72.83
$\mu = 0.8$	74.04	71.46	75.29	72.92
$\mu = 0.9$	74.49	72.11	75.38	72.99
$\mu = 1$	74.49	71.7	75.54	73.22

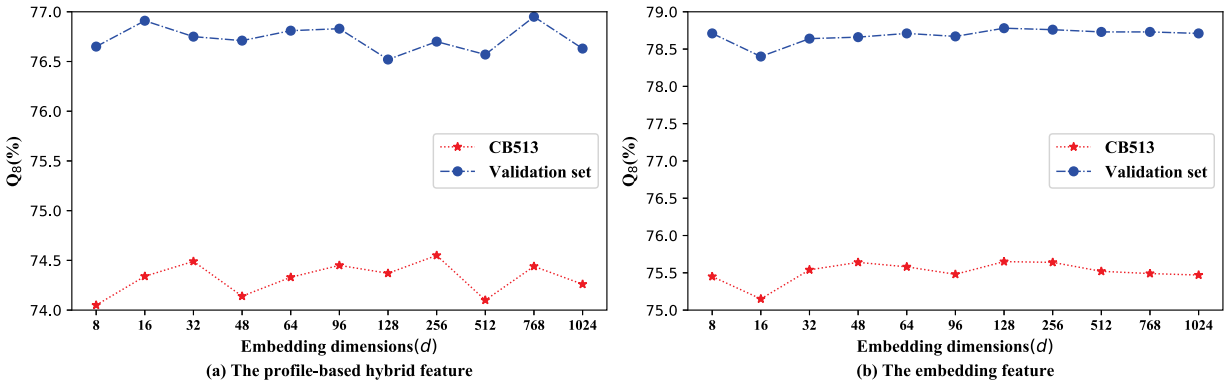

Fig. 5. The eight-class prediction results of the proposed method under varying embedding dimensions on the two feature representations.

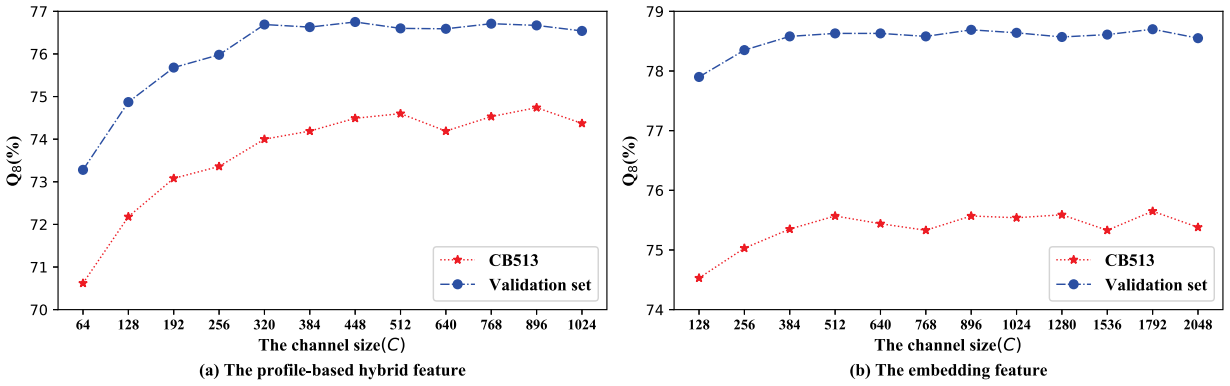

Fig. 6. The eight-class prediction results of the proposed method under different channel sizes on the two feature representations.

sampling P categories and then sampling K images from each category. Obviously, it is relatively difficult for the former to mine informative sample pairs compared with the latter. In short, how to combine multiple losses to improve the prediction accuracy of the secondary structure needs further study in the future.

4.5.2. Impact of network hyperparameters

The designed embedding network has three hyperparameters: the embedding dimension (d), the channel size (C), and the number of inception units (n). In this section, we simultaneously study their impact on the performance of the proposed method under two different feature representations (the profile-based hybrid feature and the embedding feature). In particular, we only change the parameter of interest while keeping other optimization parameters at their default values.

Impact of d. We investigate the performance of the proposed method with varying embedding dimensions {8, 16, 32, 48, 64, 96, 128, 256, 512, 768, 1024}. As shown in Fig. 5, the embedding dimension does not significantly influence the eight-class prediction accuracy. We can see that the Q_8 accuracies on the validation set and CB513 test set only fluctuate within 0.5%. It should be noted that increasing the embedding dimension does not guarantee the improvement of prediction accuracy. This is different from fine-grained image retrieval, which usually requires a higher embedding dimension to obtain good retrieval performance [50]. One possible reason is that the number of secondary structure classes is low and the lower embedding dimension is sufficient to provide good class discrimination.

Impact of C. The channel size C determines the width of the network and thus controls the size of our embedding network. The results of different width values are shown in Fig. 6. Considering that the two features have significantly different dimensions, we use different channel size sets. It can be observed from the figure that for the profile-based hybrid feature, the Q_8 accuracy on the CB513 test set increases as the network width increases when C is not greater than 512. However, when $C > 512$, further increasing the network width cannot guarantee the improvement of prediction performance. Moreover, we can see that the embedding feature also exhibits similar performance.

Impact of n. The number of inception units n controls the depth of our embedding network. The results of the proposed method with varying n values are shown in Fig. 7. As shown in the figure, on the CB513 test set, the accuracy of the proposed method based on the profile-based hybrid feature reaches a maximum at $n = 5$. For the embedding feature, the proposed method does not show significant performance improvement when $n > 2$. One possible reason is that the embedding feature is derived from the pretrained deep language model and has already captured the contextual information of each residue. Therefore, it does not require a deeper embedding network to capture nonlocal interactions between residues.

It should be noted that the default parameter settings of the proposed method are determined by our experience and are not necessarily optimal. In particular, we did not determine the model parameters based on the best performance on the validation set. This is because the accuracies on the validation set have a certain degree of fluctuation and thus cannot provide a good basis for parameter settings.

4.5.3. Impact of amino acid encoding strategies

Amino acid encoding plays an extremely important role in accurately predicting protein secondary structure. To investigate the impact of amino acid encoding strategies on the performance of the proposed method, we conduct a comparative experiment on the CB513 test set. Specifically, our experiment involved seven amino acid encodings: HHM, PSSM, HHM+phys,

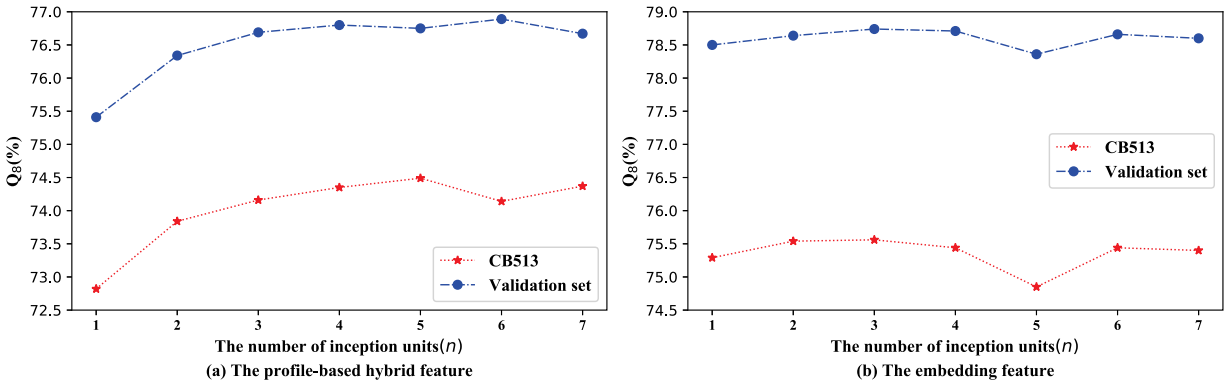

Fig. 7. The eight-class prediction results of the proposed method under varying n values on the two feature representations.

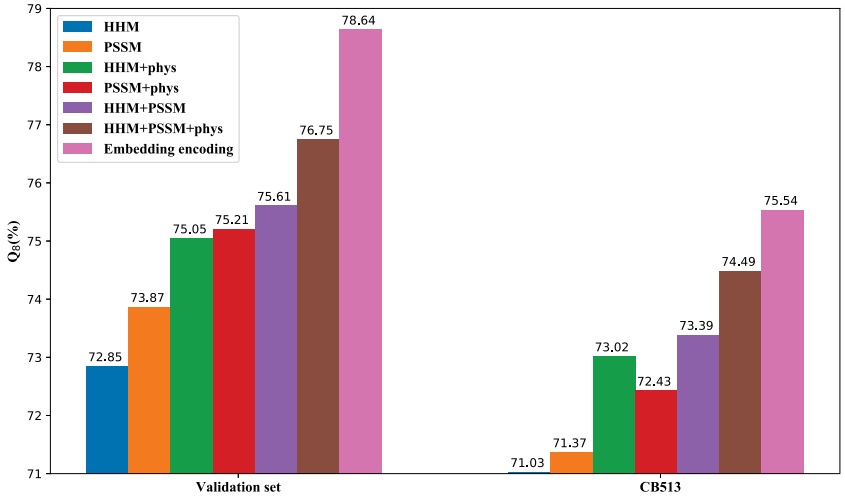

Fig. 8. The 8-state per-residue accuracies (Q_8) of the proposed method under different encoding strategies.

PSSM+phys, HHM+PSSM, HHM+PSSM+phys, and embedding encoding. The first six encodings are different combinations derived from the three basic encodings, HHM, PSSM, and phys (containing 7-dimensional physicochemical features). Among them, HHM+PSSM+phys represents the profile-based hybrid encoding. Considering that phys is position-independent, we do not include it as its own encoding. The per-residue prediction accuracies of the proposed method under different amino acid encodings for 8-state and 3-state predictions are illustrated in Figs. 8 and 9, respectively. For comparison purposes, we also present the prediction results on the validation set. Interestingly, for the three basic encodings, HHM, PSSM, and phys, we find that combining any two or all three of them can further improve the prediction accuracy of the secondary structure. This is because they contain complementary information. In particular, the multiple sequence alignments (MSAs) that construct the PSSM matrix are mainly composed of more homologous sequences, while the MSAs that generate the HHM matrix usually contain remote homologous sequences. Moreover, it can be observed that PSSM consistently outperforms HHM on the validation set but not on the CB513 test set. The 8-state accuracy of PSSM on the CB513 test set is higher than that of HHM, but the latter shows better performance in the 3-state prediction.

Note that the embedding encoding achieves the best performance on both the CB513 test set and the validation set. Particularly, on the CB513 test set, the Q_8 and Q_3 accuracies of the embedding encoding are 1.05% and 0.87% higher than those of the profile-based hybrid encoding, respectively. In addition, compared with profile-based hybrid encoding, embedding encoding also has the following advantages: (1) Feature generation requires less storage space. Generating the embedding feature requires the use of the pretrained protein language model ProtT5-XL-U50, the size of which is 11.3 GB. In this study, the protein sequence databases uniprot20_2013_03 and UniRef50 were used to generate HHM and PSSM profile features, respectively. The size of uniprot20_2013_03 after decompression is 17.3 GB, and the size of the BLAST database created based on UniRef50 is 54.7 GB. If the most recent databases are used, feature generation will require more storage space, as the size of these databases is still growing. (2) Feature generation requires less time. Based on the CB513 test set, we made a simple comparison of the time required to generate the three features. The dataset contains 513 protein chains. In our experiments, the times taken to generate the embedding features, PSSM profile features and HHM profile features for all protein chains were 116.8 s, 13 789 s and 11 615 s, respectively. Therefore, profile-based hybrid encoding is approximately 220 times slower than embedding encoding. Finally, it is worth mentioning that the 3-state prediction accuracy of the embedding encoding on the validation set approaches to the theoretical limit of protein secondary structure prediction (88%–90%) [5].

4.6. Comparison with the existing deep predictors

In this section, we compare the proposed method with some existing state-of-the-art predictors on the six test sets, CASP12, CASP13, CASP14, CB513, TEST2016, and TEST2018. Among the comparison predictors, SPIDER3 [30] is a 3-state predictor based

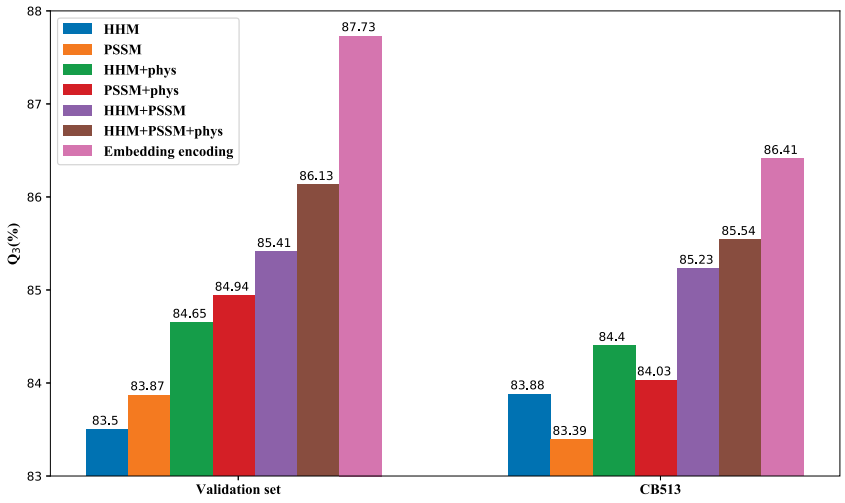

Fig. 9. The 3-state per-residue accuracies (Q_3) of the proposed method under different encoding strategies.

Table 4
Comparison of eight-class secondary structure prediction results on the six independent test sets CASP12, CASP13, CASP14, CB513, TEST2016, and TEST2018. Best results are shown in boldface. The symbols "*" and "**" indicate that the reported results come from [10] and [11], respectively. The symbol "-" indicates that the result is not available. The symbol "embed" means that the results are based on the embedding feature.

Methods	CASP12		CASP13		CASP14		CB513		TEST2016		TEST2018	
	Q_8	SOV_8	Q_8	SOV_8	Q_8	SOV_8	Q_8	SOV_8	Q_8	SOV_8	Q_8	SOV_8
CNN_BIGRU	73.26	62.31	70.68	57.17	68.17	53.22	71.21	67.65	73.91	70.92	72.78	68.75
DeepACLSTM	73.93	62.69	71.17	59.55	68.69	54.27	71.84	68.76	75.19	73.67	73.42	71.32
DCRNN	72.95	61.03	70.93	57.06	67.85	53.77	71.34	68.33	72.19	68.63	70.6	65.82
DeepCNN	73.81	60.79	71.92	55.38	67.91	51.78	72.44	69.51	74.54	71.56	72.75	69.18
MUFold-SS	74.83	**65.46**	72.05	58.9	68.83	52.12	73	69.58	76.03	73.67	74.29	71
NetSurfP-2.0*	–	–	–	–	–	–	–	–	–	-	73.81	71.14
SPOD-1D*	–	–	–	–	–	–	–	–	76.03	73.88	74.26	71.45
SAINT**	–	–	–	–	–	–	–	–	76.23	–	74.48	–
DML_SS	74.87	64.74	72.53	58.23	69.47	54.48	74.49	71.7	76.62	74.6	74.82	72.23
DML_SS^{embed}	**76.2**	62.76	**74.17**	**60.1**	**70.22**	**62.23**	**75.54**	**73.22**	**78.03**	**75.9**	**76.48**	**73.44**

on the bidirectional LSTM model. DeepCNN [29] and MUFold-SS [33] are predictors composed of only the convolutional network. SAINT [11] improved MUFold-SS by combining the self-attention mechanism with the deep inception-inside-inception network. CNN_BIGRU [32], DeepACLSTM [34], DCRNN [28], and NetSurfP-2.0 [35] are cascaded hybrid models of bidirectional recurrent neural networks and convolutional networks. In addition, SPOT-1D [10] is an ensemble of 9 models with different parameter settings, including 3 LSTM models, 3 LSTM-RESNET models, and 3 RESNET-LSTM models. It should be noted that all of these predictors are proposed for profile-based features. Therefore, we only present their results on profile-based hybrid encoding. For a fair comparison, all predictors should use the same datasets and input features. To this end, we reproduce five methods, CNN_BIGRU, DeepACLSTM, DCRNN, DeepCNN, and MUFold-SS, based on our training/validation/test settings. For the four complex predictors, NetSurfP-2.0, SPOD-1D, SAINT, and SPIDER3, we use the results directly reported in the literature.

The eight-class prediction results of the proposed method and eight existing predictors on the six test sets are given in Table 4. From the table, it can be seen that the proposed method DML_SS consistently outperforms the eight state-of-the-art predictors on the four datasets CASP14, CB513, TEST2016, and TEST2018 in terms of both Q_8 and SOV_8. On the remaining two datasets CASP12 and CASP13, the Q_8 accuracy of DML_SS is also better than that of all other methods. Note that the model sizes of CNN_BIGRU, DeepACLSTM, DCRNN, DeepCNN, MUFold-SS, NetSurfP-2.0, and SAINT are 16.2 MB, 23 MB, 18.2 MB, 9.4 MB, 17.8 MB, 108 MB, and 140.9 MB, respectively. However, the model size of DML_SS is only 8.3 MB. This means that DML_SS achieves better performance with fewer parameters. In addition, regarding the two amino acid encodings, we can see that embedding encoding is significantly better than profile-based hybrid encoding in most cases. The only exception is that on the CASP2 dataset, the SOV_8 accuracy of the embedding feature is 1.98% lower than that of the hybrid feature based on evolutionary information, although the Q_8 accuracy of the former is 1.33% higher than that of the latter.

Table 5 gives three-class prediction results on the six test sets CASP12, CASP13, CASP14, CB513, TEST2016, and TEST2018. It can be observed from the table that DML_SS outperforms the five predictors CNN_BIGRU, DeepACLSTM, DCRNN, DeepCNN, and SPIDER-3 in most cases. The performance of DML_SS on the three large test sets, CB513, TEST2016, and TEST2018 is consistently better than MUFold-SS, while the latter's performance on the three small test sets, CASP12, CASP13, and CASP14 is better than DML_SS. It should be noted that for the results of MUFold-SS reported in this study, the optimizer used during training is AdamW instead of the original Adam optimizer. In practice, we found that the former can improve the prediction performance of MUFold-SS. On the two datasets TEST2016 and TEST2018, the SOV_3 accuracies of DML_SS are higher than those of SPOD-1D, but its Q_3 accuracies are lower than those of the latter. Moreover, we can see that compared with the profile-based hybrid feature, the embedding feature has achieved better classification performance on all six test sets.

Table 5
Comparison of three-class secondary structure prediction results on the six independent test sets CASP12, CASP13, CASP14, CB513, TEST2016, and TEST2018. Best results are shown in boldface. The symbol "*" indicates that the reported results come from [10]. The symbol "–" indicates that the result is not available. The symbol "embed" means that the results are based on the embedding feature.

Methods	CASP12		CASP13		CASP14		CB513		TEST2016		TEST2018	
	Q_3	SOV_3	Q_3	SOV_3	Q_3	SOV_3	Q_3	SOV_3	Q_3	SOV_3	Q_3	SOV_3
CNN_BIGRU	83.84	73.45	83.11	69.3	79.43	63.29	84.52	80.22	85.04	81.61	84.17	79.41
DeepACLSTM	83.9	72.47	82.93	68.34	78.51	61	84.51	80.48	85.62	82.6	84.66	80.05
DCRNN	83.91	71.61	82.3	69.47	79.08	60.05	84.28	79.89	83.72	78.39	82.75	75.1
DeepCNN	84.79	71.98	82.66	63.82	78.86	59.45	84.47	78.84	85.14	79.31	84.16	76.83
MUFold-SS	84.84	**73.69**	83.65	69.39	79.95	60.12	85.51	80.38	85.97	81.98	84.63	79.53
NetSurfP-2.0*	–	–	–	–	–	–	–	–	–	-	85.31	78.58
SPOD-1D*	–	–	–	–	–	–	–	–	86.67	79.52	85.66	78.77
SPIDER-3*	–	–	–	–	–	–	–	–	84.66	75.62	83.84	73.89
DML_SS	84.66	72.13	83.33	67.97	79	60.37	85.54	81.49	86.1	82.72	84.83	80.5
DML_SS^{embed}	**86.08**	73.46	**84.95**	**71.07**	**80.75**	**65.81**	**86.41**	**82.39**	**87.41**	**84.51**	**86.82**	**82.43**

Table 6
The eight-class secondary structure prediction results on the TEST2018 test set.

Methods	Q_L	Q_B	Q_E	Q_G	Q_I	Q_H	Q_S	Q_T	Q_8	SOV_8
KNN	67.79	11.94	84.79	41.85	0	93.34	32.9	57.47	74.55	72.23
DML_SS	65.23	14.37	86.25	37.89	0	94.38	34.49	58.33	74.82	72.23
KNN^{embed}	71.16	15.86	87.31	40.84	0	94.21	34.3	58.61	76.32	73.88
DML_SS^{embed}	71.62	17.54	87.91	37.26	0	94.32	35.92	57.54	76.48	73.44

Table 7
The three-class secondary structure prediction results on the TEST2018 test set.

Methods	Q_C	Q_E	Q_H	Q_3	SOV_3
KNN	82.2	80.11	90.45	84.82	80.76
DML_SS	80.97	81.56	90.89	84.83	80.5
KNN^{embed}	84.67	83.64	90.72	86.71	82.23
DML_SS^{embed}	84.98	85.26	89.71	86.82	82.43

4.7. Comparison with k-nearest neighbor classification

Each residue in a given protein chain can be mapped into a d-dimensional numeric vector through our embedding network. Therefore, after performing this mapping on all protein chains in a dataset, we can use the k-nearest neighbor (KNN) algorithm to predict secondary structure based on the mapped training dataset. In our experiments, the cosine similarity distance is adopted to perform the k-nearest neighbor search, and the parameter k is set to 81. For comparison with the proposed method, the embedding dimension d is also set to 32. The eight-class and three-class prediction results of the two methods on the TEST2018 test set are shown in Tables 6 and 7, respectively. From these two tables, we can observe that although the two methods have different prediction results in single-state accuracy measures, they are comparable to each other in terms of their overall per-residue accuracy and SOV metric. Moreover, it should be noted that the k-nearest neighbor algorithm requires storing a large amount of training data and performing a time-consuming k-nearest neighbor search. However, the proposed method DML_SS only needs to store c learned centroids and uses the nearest centroid classifier for classification. Therefore, DML_SS is significantly faster than KNN in terms of prediction speed.

4.8. Ensemble prediction

The generalization ability of an ensemble model composed of multiple individual models is often much better than that of its individual models. In this section, we further improve the prediction accuracy of the secondary structure based on this feature of the ensemble model. Specifically, our ensemble model consists of five individual models, which are obtained by training the proposed method DML_SS with different random seeds {0, 10000, 20000, 30000, 40000}. The output of the ensemble model is the mean of the output of all base models. The prediction results of the proposed method and its ensemble under two different feature representations on the TEST2016 and TEST2018 test sets are given in Table 8. It can be seen from the table that the ensemble prediction performance of 8-state and 3-state secondary structures under the two feature representations is consistently better than that of the individual model. In particular, we find that ensembling more than five models does not further improve the prediction accuracy. Moreover, it should be noted that the random seed only affects the initial weight of the embedding network and the initial value of the centroid vectors. In fact, we also try to obtain distinct models by changing the network width and projection dimension. However, our results suggest that the performance of the ensemble model has essentially not changed.

5. Conclusion

In this paper, a simple and effective deep centroid model, named DML_SS, is proposed for both 3-state and 8-state protein secondary structure prediction based on deep metric learning. In the proposed DML_SS, each residue in a protein chain is mapped into an embedding space using a lightweight network with multibranch topology. The goal of the embedding learning is to maximize the similarity of each residue to its target centroid while minimizing its similarity to nontarget centroids. The three designed loss functions, hard margin loss, soft margin loss, and softmax loss, can be used to simultaneously optimize the embedding network and the centroid vectors. After obtaining the embedding representation of each residue, we can adopt the nearest centroid classifier or k nearest neighbor algorithm for secondary structure type assignment based on the cosine similarity distance. As an extension of DML_SS, we also propose an ensemble of multiple individual models to further improve the prediction accuracy of the secondary structure. The proposed method is evaluated on six publicly available test sets, CASP12, CASP13, CASP14, CB513, TEST2016, and TEST2018, and experimental results show that it achieves state-of-the-art performance with a simple architecture and smaller model size.

In addition, we note that protein language models trained on massive protein sequence data with self-supervised techniques

Table 8
The prediction results of the proposed method and its ensemble on the TEST2016 and TEST2018 test sets.

Methods	TEST2016				TEST2018			
	Q_8	SOV_8	Q_3	SOV_3	Q_8	SOV_8	Q_3	SOV_3
DML_SS	76.62	74.6	86.1	82.72	74.82	72.23	84.83	80.5
Ensemble DML_SS	77.29	75.48	86.64	83.73	75.36	73.17	85.44	81.16
DML_SS^{embed}	78.03	75.9	87.41	84.51	76.48	73.44	86.82	82.43
Ensemble DML_SS^{embed}	78.41	76.44	87.66	84.85	76.79	74.14	86.98	82.57

are revolutionizing biological modeling based on sequence information. The embedding feature from pretrained language models not only contains knowledge of intrinsic biological properties but also avoids expensive multiple sequence alignment operations against massive protein sequence databases. To our knowledge, this is the first work to design a secondary structure predictor specifically for the embedding feature. Extensive comparative experiments have shown that the embedding feature from ProtT5-XL-U50 significantly outperforms the profile-based hybrid feature in achieving higher prediction accuracy. We believe that this advantage will be further extended as the potential of protein language models is further explored.

In summary, the main advantages of the proposed model include: (1) It achieves state-of-the-art performance for protein secondary structure prediction; (2) It has a simple network architecture and a small model size; (3) It demonstrates for the first time that deep metric learning can perform accurate and fast protein secondary structure prediction; (4) It is applicable to both the profile-based hybrid feature and the embedding feature derived from pretrained protein language models; (5) Based on the deep embedding representation of residues, both nearest centroid classifier and *k*-nearest neighbor algorithm can be used for secondary structure type assignment. Moreover, the limitations of the proposed model include: (1) Unable to provide domain-level protein secondary structure prediction results; (2) Unable to provide information about the reliability of its prediction. These limitations will be addressed in our future studies. In future work, we intend to: (1) integrate with existing domain prediction techniques, such as FUpred [63] and DNN-Dom [64], to implement domain-level protein secondary structure prediction; (2) integrate with existing uncertainty quantification techniques [65–70] to estimate uncertainty in protein secondary structure prediction.

CRediT authorship contribution statement

Wei Yang: Conceptualization, Methodology, Software, Data curation, Writing – original draft. **Yang Liu:** Visualization, Reviewing and editing. **Chunjing Xiao:** Validation, Investigation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors acknowledge the three anonymous reviewers for their constructive comments. This work was supported by the National Science Foundation of China (Grant nos. 61806074 and 62176087).

References

[1] Q. Jiang, X. Jin, S.J. Lee, S. Yao, Protein secondary structure prediction: A survey of the state of the art, J. Mol. Graph Model. 76 (2017) 379–402.
[2] D.Y. Paul, Z. Bing Bing, Y.Z. Albert, Machine learning techniques for protein secondary structure prediction:An overview and evaluation, Curr. Bioinf. 3 (2) (2008) 74–86.
[3] L. Zhang, L. Kong, X. Han, J. Lv, Structural class prediction of protein using novel feature extraction method from chaos game representation of predicted secondary structure, J. Theor. Biol. 400 (2016) 1–10.
[4] R. Pearce, Y. Zhang, Deep learning techniques have significantly impacted protein structure prediction and protein design, Curr. Opin. Struct. Biol. 68 (2021) 194–207.
[5] Y. Yang, J. Gao, J. Wang, R. Heffernan, J. Hanson, K. Paliwal, Y. Zhou, Sixty-five years of the long march in protein secondary structure prediction: the final stretch? Brief Bioinform. 19 (3) (2018) 482–494.
[6] W. Kabsch, C. Sander, Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features, Biopolymers 22 (12) (1983) 2577–2637.
[7] P. Baldi, S. Brunak, P. Frasconi, G. Soda, G. Pollastri, Exploiting the past and the future in protein secondary structure prediction, Bioinformatics 15 (11) (1999) 937–946.
[8] M. Heinzinger, A. Elnaggar, Y. Wang, C. Dallago, D. Nechaev, F. Matthes, B. Rost, Modeling aspects of the language of life through transfer-learning protein sequences, BMC Bioinformatics 20 (1) (2019) 723.
[9] R. Rao, N. Bhattacharya, N. Thomas, Y. Duan, X. Chen, J. Canny, P. Abbeel, Y.S. Song, Evaluating protein transfer learning with TAPE, Adv. Neural Inf. Process. Syst. 32 (2019) 9689–9701.
[10] J. Hanson, K. Paliwal, T. Litfin, Y. Yang, Y. Zhou, Improving prediction of protein secondary structure, backbone angles, solvent accessibility and contact numbers by using predicted contact maps and an ensemble of recurrent and residual convolutional neural networks, Bioinformatics 35 (14) (2019) 2403–2410.
[11] M.R. Uddin, S. Mahbub, M.S. Rahman, M.S. Bayzid, SAINT: self-attention augmented inception-inside-inception network improves protein secondary structure prediction, Bioinformatics (2020).
[12] G. Xu, Q. Wang, J. Ma, OPUS-TASS: a protein backbone torsion angles and secondary structure predictor based on ensemble neural networks, Bioinformatics 36 (20) (2020) 5021–5026.
[13] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C.L. Zitnick, J. Ma, R. Fergus, Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences, Proc. Natl. Acad. Sci. 118 (15) (2021) e2016239118.
[14] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, W. Yu, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, D. Bhowmik, B. Rost, ProtTrans: TOwards cracking the language of lifes code through self-supervised deep learning and high performance computing, IEEE Trans. Pattern Anal. Mach. Intell. 14 (8) (2021).
[15] S. Ovchinnikov, H. Park, N. Varghese, P.S. Huang, G.A. Pavlopoulos, D.E. Kim, H. Kamisetty, N.C. Kyrpides, D. Baker, Protein structure determination using metagenome sequence data, Science 355 (6322) (2017) 294–298.
[16] D.T. Jones, Protein secondary structure prediction based on position-specific scoring matrices, J. Mol. Biol. 292 (2) (1999) 195–202.
[17] N. Qian, T.J. Sejnowski, Predicting the secondary structure of globular proteins using neural network models, J. Mol. Biol. 202 (4) (1988) 865–884.
[18] J. Selbig, T. Mevissen, T. Lengauer, Decision tree-based formation of consensus protein secondary structure prediction, Bioinformatics 15 (12) (1999) 1039–1046.
[19] J. Guo, H. Chen, Z. Sun, Y. Lin, A novel method for protein secondary structure prediction using dual-layer SVM and profiles, Proteins 54 (4) (2004) 738–743.
[20] B. Yang, Q. Wu, Z. Ying, H. Sui, Predicting protein secondary structure using a mixed-modal SVM method in a compound pyramid model, Knowl.-Based Syst. 24 (2) (2011) 304–313.
[21] S. Salzberg, S. Cost, Predicting protein secondary structure with a nearest-neighbor algorithm, J. Mol. Biol. 227 (2) (1992) 371–374.
[22] W. Yang, K. Wang, W. Zuo, Prediction of protein secondary structure using large margin nearest neighbour classification, Int. J. Bioinf. Res. Appl. 9 (2) (2013) 207–219.

[23] K. Asai, S. Hayamizu, K. Handa, Prediction of protein secondary structure by the hidden Markov model, Comput. Appl. Biosci. 9 (1993) 141–146.
[24] Z. Aydin, Y. Altunbasak, M. Borodovsky, Protein secondary structure prediction for a single-sequence using hidden semi-Markov models, BMC Bioinformatics 7 (2006) 178.
[25] D.W. Buchan, S. Ward, A.E. Lobley, T. Nugent, K. Bryson, D.T. Jones, Protein annotation and modelling servers at university college London, Nucleic Acids Res. (2010) gkq427.
[26] A. Drozdetskiy, C. Cole, J. Procter, G.J. Barton, JPred4: a protein secondary structure prediction server, Nucleic Acids Res. (2015).
[27] S. Wang, J. Peng, J. Ma, J. Xu, Protein secondary structure prediction using deep convolutional neural fields, Sci. Rep. 6 (2016) 18962.
[28] Z. Li, Y. Yu, Protein secondary structure prediction using cascaded convolutional and recurrent neural networks, in: IJCAI'16 Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, AAAI Press, pp. 2560–2567.
[29] A. Busia, N. Jaitly, Next-step conditioned deep convolutional neural networks improve protein secondary structure prediction, 2017, arXiv, arXiv:1702.03865.
[30] R. Heffernan, Y. Yang, K. Paliwal, Y. Zhou, Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility, Bioinformatics (Oxford, England) 33 (18) (2017) 2842–2849.
[31] Y. Wang, H. Mao, Z. Yi, Protein secondary structure prediction by using deep learning method, Knowl.-Based Syst. 118 (2017) 115–123.
[32] I. Drori, I. Dwivedi, P. Shrestha, J. Wan, Y. Wang, Y. He, A. Mazza, H. Krogh-Freeman, D. Leggas, K. Sandridge, L. Nan, K.A. Thakoor, C. Joshi, S. Goenka, C. Keasar, I. Pe'er, High quality prediction of protein Q8 secondary structure by diverse neural network architectures, 2018, arXiv preprint arXiv:1811.07143.
[33] C. Fang, Y. Shang, D. Xu, MUFOLD-SS: New deep inception-inside-inception networks for protein secondary structure prediction, Proteins 86 (5) (2018) 592–598.
[34] Y. Guo, W. Li, B. Wang, H. Liu, D. Zhou, DeepACLSTM: deep asymmetric convolutional long short-term memory neural models for protein secondary structure prediction, BMC Bioinformatics 20 (1) (2019) 1–12.
[35] M.S. Klausen, M.C. Jespersen, H.B. Nielsen, K.K. Jensen, V.I. Jurtz, C.K. Sonderby, M.O.A. Sommer, O. Winther, M. Nielsen, B. Petersen, NetSurfP-2.0: Improved prediction of protein structural features by integrated deep learning, Proteins 87 (6) (2019) 520–527.
[36] R. Heffernan, K. Paliwal, J. Lyons, J. Singh, Y. Yang, Y. Zhou, Single-sequence-based prediction of protein secondary structures and solvent accessibility by deep whole-sequence learning, J. Comput. Chem. 39 (26) (2018) 2210–2216.
[37] J. Singh, T. Litfin, K. Paliwal, J. Singh, A.K. Hanumanthappa, Y. Zhou, SPOT-1D-single: Improving the single-sequence-based prediction of protein secondary structure, backbone angles, solvent accessibility and half-sphere exposures using a large training set and ensembled deep learning, Bioinformatics (2021).
[38] L. Moffat, D.T. Jones, Increasing the accuracy of single sequence prediction methods using a deep semi-supervised learning framework, Bioinformatics (2021).
[39] A. Hermans, L. Beyer, B. Leibe, In defense of the triplet loss for person re-identification, 2017, arXiv abs/1703.07737.
[40] D. Yi, Z. Lei, S. Liao, S.Z. Li, Deep metric learning for person re-identification, in: 2014 22nd International Conference on Pattern Recognition, pp. 34–39.
[41] F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: A unified embedding for face recognition and clustering, in: 2015 IEEE Conference on Computer Vision And Pattern Recognition, CVPR, pp. 815–823.
[42] H.O. Song, S. Jegelka, V. Rathod, K. Murphy, Deep metric learning via facility location, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 2206–2214.
[43] S. Chopra, R. Hadsell, Y. LeCun, Learning a similarity metric discriminatively, with application to face verification, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, CVPR'05, pp. 539–546, http://dx.doi.org/10.1109/CVPR.2005.202.
[44] E. Hoffer, N. Ailon, Deep metric learning using triplet network, in: International Workshop on Similarity-Based Pattern Recognition, Springer, pp. 84–92.
[45] W. Ge, Deep metric learning with hierarchical triplet loss, in: Proceedings of the European Conference on Computer Vision, ECCV, pp. 269–285.
[46] C.-Y. Wu, R. Manmatha, A.J. Smola, P. Krahenbuhl, Sampling matters in deep embedding learning, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 2840–2848.
[47] K. Sohn, Improved deep metric learning with multi-class n-pair loss objective, in: Adv. Neural Inf. Process. Syst., pp. 1857–1865.
[48] J. Wang, F. Zhou, S. Wen, X. Liu, Y. Lin, Deep metric learning with angular loss, in: 2017 IEEE International Conference on Computer Vision, ICCV, pp. 2612–2620.
[49] X. Wang, Y. Hua, E. Kodirov, G. Hu, R. Garnier, N.M. Robertson, Ranked list loss for deep metric learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5207–5216.
[50] X. Wang, X. Han, W. Huang, D. Dong, M.R. Scott, Multi-similarity loss with general pair weighting for deep metric learning, in: Proceedings of the IEEE Conference on Computer Vision And Pattern Recognition, pp. 5022–5030.
[51] B. Yu, D. Tao, Deep metric learning with tuplet margin loss, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6490–6499.
[52] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, Y. Wei, Circle loss: A unified perspective of pair similarity optimization, in: Proceedings of the IEEE/CVF Conference on Computer Vision And Pattern Recognition, pp. 6398–6407.
[53] Y. Movshovitz-Attias, A. Toshev, T.K. Leung, S. Ioffe, S. Singh, No fuss distance metric learning using proxies, in: 2017 IEEE International Conference on Computer Vision, ICCV, pp. 360–368.
[54] E.W. Teh, T. DeVries, G.W. Taylor, ProxyNCA++: Revisiting and revitalizing proxy neighborhood component analysis, in: European Conference on Computer Vision, pp. 448–464.
[55] A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, Q. Le, Searching for MobileNetV3, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV, pp. 1314–1324.
[56] K.Q. Weinberger, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, J. Mach. Learn. Res. 10 (2009) 207–244.
[57] G. Wang, R.L. Dunbrack, PISCES: a protein sequence culling server, Bioinformatics 19 (12) (2003) 1589–1591.
[58] J. Meiler, M. Müller, A. Zeidler, F. Schmäschke, Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks, J. Mol. Model. 7 (2001) 360–369.
[59] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint arXiv:1810.04805.
[60] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P.J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, J. Mach. Learn. Res. 21 (140) (2020) 1–67.
[61] A. Zemla, Č. Venclovas, K. Fidelis, B. Rost, A modified definition of sov, a segment based measure for protein secondary structure prediction assessment, Proteins Struct. 34 (1999).
[62] M. Ye, J. Shen, G. Lin, T. Xiang, L. Shao, S.C.H. Hoi, Deep learning for person re-identification: A survey and outlook, IEEE Trans. Pattern Anal. Mach. Intell. (2021) 1.
[63] W. Zheng, X. Zhou, Q. Wuyun, R. Pearce, Y. Li, Y. Zhang, FUpred: detecting protein domains through deep-learning-based contact map prediction, Bioinformatics 36 (12) (2020) 3749–3757.
[64] Q. Shi, W. Chen, S. Huang, F. Jin, Y. Dong, Y. Wang, Z. Xue, DNN-dom: predicting protein domain boundary from sequence alone by deep neural network, Bioinformatics 35 (24) (2019) 5128–5136.
[65] M. Abdar, M.A. Fahami, S. Chakrabarti, A. Khosravi, P. Pławiak, U.R. Acharya, R. Tadeusiewicz, S. Nahavandi, BARF: A new direct and cross-based binary residual feature fusion with uncertainty-aware module for medical image classification, Inform. Sci. 577 (2021) 353–378.
[66] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U.R. Acharya, V. Makarenkov, S. Nahavandi, A review of uncertainty quantification in deep learning: Techniques, applications and challenges, Inf. Fusion 76 (2021) 243–297.
[67] M. Abdar, S. Salari, S. Qahremani, H.-K. Lam, F. Karray, S. Hussain, A. Khosravi, U.R. Acharya, S. Nahavandi, UncertaintyFuseNet: Robust uncertainty-aware hierarchical feature fusion with ensemble Monte Carlo dropout for COVID-19 detectio, 2021, arXiv preprint arXiv:2105.08590.
[68] M. Abdar, M. Samami, S. Dehghani Mahmoodabad, T. Doan, B. Mazoure, R. Hashemifesharaki, L. Liu, A. Khosravi, U.R. Acharya, V. Makarenkov, S. Nahavandi, Uncertainty quantification in skin cancer classification using three-way decision-based Bayesian deep learning, Comput. Biol. Med. 135 (2021) 104418.
[69] Y. Qin, Z. Liu, C. Liu, Y. Li, X. Zeng, C. Ye, Super-resolved q-space deep learning with uncertainty quantification, Med. Image Anal. 67 (2021) 101885.
[70] D. Seuss, Bridging the gap between explainable AI and uncertainty quantification to enhance trustability, 2021, arXiv preprint arXiv:2105.11828.