Computers in Biology and Medicine 110 (2019) 144–155

Contents lists available at ScienceDirect

Computers in Biology and Medicine

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

MCP: A multi-component learning machine to predict protein secondary structure

Leila Khalatbari[a], M.R. Kangavari[a], Saeid Hosseini[a,§,*], Hongzhi Yin[c], Ngai-Man Cheung[§]

[a] School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
[§] ST Electronics - SUTD Cyber Security Laboratory, Singapore University of Technology and Design, Singapore
[c] School of Information Technology and Electrical Engineering, University of Queensland, Brisbane, Australia

ARTICLE INFO

Keywords:
Protein secondary structure prediction
Fuzzy k-nearest neighbor
Support vector machine
Ensemble prediction machine

ABSTRACT

The Gene or DNA sequence in every cell does not control genetic properties on its own; Rather, this is done through the translation of DNA into protein and subsequent formation of a certain 3D structure. The biological function of a protein is tightly connected to its specific 3D structure. Prediction of the protein secondary structure is a crucial intermediate step towards elucidating its 3D structure and function. Traditional experimental methods for prediction of protein structure are expensive and time-consuming. Nevertheless, the average accuracy of the suggested solutions has hardly reached beyond 80%. The possible underlying reasons are the ambiguous sequence-structure relation, noise in input protein data, class imbalance, and the high dimensionality of the encoding schemes. Furthermore, we utilize a compound string dissimilarity measure to directly interpret protein sequence content and avoid information loss. In order to improve accuracy, we employ two different classifiers including support vector machine and fuzzy nearest neighbor and collectively aggregate the classification outcomes to infer the final protein structures. We conduct comprehensive experiments to compare our model with the current state-of-the-art approaches. The experimental results demonstrate that given a set of input sequences, our multi-component framework can accurately predict the protein structure. Nevertheless, the effectiveness of our unified model can be further enhanced through framework configuration.

1. Introduction

In this paper, we focus on interpreting the sequential string data which is crucial to many applications including bioinformatics and molecular biology. In the context of computational biology, prediction of protein secondary structure from the input string sequences is the ground to foster protein function determination and design better drugs [15,53,58]. Hence, given protein sequences composed of amino-acid molecules, our aim is to predict the secondary structure that each amino-acid adopts through a classification paradigm. In Fig. 1, the top string chain illustrates a protein sequence. Each letter of this sequence represents an amino-acid molecule. Our aim is to assign each amino-acid molecule to one of three classes of protein secondary structure named as α-helix (H), β-sheet (E) and *coil* (C).

Template-based methods and machine learning strategies are two computational approaches for prediction of protein secondary structure. Template-based methods neither result in higher accuracy compared to machine learning methods nor perform well on non-homologous proteins [31]. Feature extraction from protein sequences is the first step when applying a machine learning approach. However, the extracted features may not reflect all the information a sequence contains and thus can lead to information loss [10,30,35]. Nevertheless, from the biological perspective, protein sequence contains indispensable information to adopt certain structures [31,35,36]. While predicting such structures is an appealing task, challenges abound. First, the relationship between a sequence and its corresponding structure is quite complex [27,58]. Second, the selected features dramatically influence the learner's effectiveness [35]. Additionally, the training data including protein sequences and their related known structures are partially noisy. Lastly, the amino-acid samples are not distributed equally in classes (class imbalance) [1].

To address the challenges, we devise a multi-component predictor to directly process amino-acid sequences into accurate protein structures. While every component participates in the enhancement and correction of the results, the multi-component property of our solution can learn various information from the input sequences. The MCP framework processes the textual context of protein sequences which yields three advantages. First, it avoids information loss through processing of

[*] Corresponding author. , School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran.
E-mail address: saeid.hosseini@uq.net.au (S. Hosseini).

https://doi.org/10.1016/j.compbiomed.2019.04.040
Received 24 September 2018; Received in revised form 23 April 2019; Accepted 29 April 2019

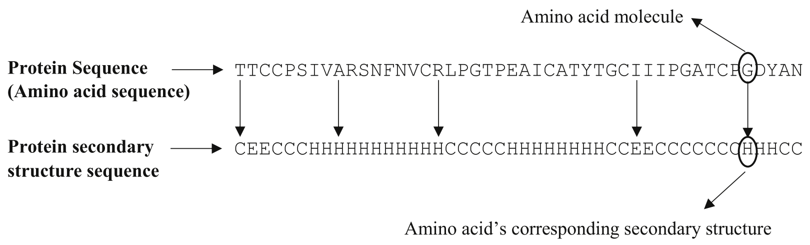

Amino acid molecule
Protein Sequence (Amino acid sequence)
TTCCPSIVARSNFNVCRLPGTPEAICATYTGCIIIPGATCPGDYAN
Protein secondary structure sequence
CEECCCHHHHHHHHHHHCCCCCHHHHHHHHCCEECCCCCCCHHHCC
Amino acid's corresponding secondary structure

Fig. 1. Sequence-structure mapping.

Table 1
An overview of the literature of protein secondary structure prediction.

Category of methods	methods	Reference
Probabilistic and mining	Chou-Fesman	[13]
	GOR	[16]
	Hidden Markov Model (HMM)	[7,8]
	Decision-tree	[20,40]
	Natural Language Processing (NLP)	[31]
	Distance-based learners	[5,18,53]
Machine learning	Neural networks and deep learning	[1,3,4,14,47,51,57,58]
	Support vector machines (SVM)	[32,46,63,64]
	Multi-component approaches	[1,3,4,8,14,25,48,57,61] [6,20,46,51,58,62–64]
Feature extraction	Sequence profiles	[20622545116336]
	pseudo-potentials, amino acid physiochemical properties, AACs, pseudo-AAC, and PseKNC	[9,12,30,61]

the primary sequence data and bypassing the feature extraction procedure. Second, we can eliminate the negative effects of certain feature subsets that can further promote the effectiveness of the learner. Third, we can interpret a latent natural language via employing the dissimilarity measures. The latent language can reveal hidden relationships between protein sequences. Our proposed framework employs two efficient algorithms of Support Vector Machine (SVM) and Fuzzy K-Nearest Neighbor (FKNN) in parallel. The *edit* distance function forms the *edit* kernel for SVM that infers the dissimilarity among input sequences. Furthermore, we embed a compound dissimilarity measure called $\tilde{d}$ into FKNN module. $\tilde{d}$ works based on n-gram scores, LZ scores, and a new parameter called dissimilarity rate (ρ_d). The output of each learner passes through a filtering component to refine the biologically meaningless structures. The corrected output from filtration enters the aggregation pool that can further make a consensus among the decisions of edit-SVM and $\tilde{d}$-FKNN.

Our proposed model is both flexible and modifiable. For instance, one can add more classifiers and expand the ensemble size with different base learners, utilize the dissimilarity measure ($\tilde{d}$) in SVM kernel and subsequently analyze the outcomes more in details, or pass numerical features (i.e. protein sequence profiles) to one learner and the string sequences to another to collectively investigate the resultant differences. Fuzzification of the SVM module can enhance the aggregation process and consequently enrich the final prediction results. A weighted form of the compound dissimilarity measure can lead to accuracy enhancement. A dynamic parameter optimization module which can finally improve the effectiveness of the prediction results.

Our contributions in this study are threefold:

– We devise a flexible *Multi-Component Prediction framework* (*MCP*) which can directly process the latent contexts of the input protein sequences and suggest accurate output secondary structures. Our model can be further generalized to perform on an arbitrary number of classes.
– We employ two different classifiers of edit-SVM and $\tilde{d}$-FKNN and collectively aggregate the filtered classification outcomes to infer the final results.
– We achieve better accuracy in prediction of protein secondary structure using various modules of classification, aggregation and dissimilarity measures.

The remainder of this paper is organized as follows. In section 2, we briefly summarize related work. In section 3, we provide the preliminary definitions, define problems and present an overview of our framework. In section 4, we elucidate the proposed model and explain underlying techniques. We also prove that our model can be generalized to perform on an arbitrary number of classes. In section 5, we examine the effectiveness of the competitor baselines. In section 6, we conclude the paper and discuss future work.

2. Related work

Predicting the structure using a set of string sequences has been well-studied in biology [5,15,15,17,25,31,50]. Moreover, machine learning-based methods such as ensemble models, and deep learning approaches have been recently exploited. In this section, we discuss the related work in two-fold: First, *Statistical and mining methods*, Second, *Machine Learning-based models*. Table 1, demonstrates an overview of the literature.

Probabilistic and mining methods: Primary probabilistic methods [13,16] are based on empirical analytics and mainly compute the tendency of each amino-acid in protein sequence to form a particular secondary structure. GOR [16] extends Chou-Fesman [13] to improve prediction performance through including the context of each amino-acid. Probabilistic models compute the conditional probability for each amino-acid to adopt a secondary structure, given that its neighbors have formed that structure. Since the structure of an amino-acid is correlated with its neighbors, in this paper, we also use a sliding window to incorporate the neighboring context in the prediction process. A more recent probabilistic approach is the Hidden Markov Model (HMM) [7,8]. The HMM graphical models well adapt to one-dimensional sequence processing. In HMM, the states of the graph represent the secondary classes and the output probabilities determine the structures [7]. Chen et al. [8] employ Markov Model (feature extraction) to generate a sequence encoding scheme that is subsequently fed into the SVM to predict the results. In our approach, the fuzzy membership functions provide the probabilities that will regulate final secondary structures.

Tree-based methods are also used in mining approaches [60]. [40] extracts the rules from FS-Tree to leverage sequence-structure mappings. In another tree-approach [20], applies SVM to eliminate the noise and outlier data. He et al. [20] firstly pass the refined data to a decision tree. Subsequently, they apply the extracted rules of the decision tree to predict structures. Moreover, the NLP-based methods [24] consider the textual context of the input sequences to infer the missing structures. Lin et al. [31] exploit n-gram patterns to create a dictionary of synonymous words that can be later used to compute sequence similarities. We similarly employ a compound measure including an n-gram metric to estimate dissimilarities among sequence chunks. The distance-based classifiers have also been employed in structure prediction [18]. K-Nearest Neighbor algorithm (KNN) and its fuzzified versions [5,53] are the most popular distance-based learners. Hence we devise a fuzzified KNN.

Machine learning-based approaches: In the field of sequence-structure mapping, there are three major groups of machine learning models: *neural networks and deep learning paradigms*, support *vector machines* and *multi-component learners*. Neural Networks (NN) are the first generation of machine-learning approaches that are used for protein structure prediction. In practice, employing of a well-designed NN can properly estimate class boundaries. Different versions of recurrent neural networks [4,39,49] greatly suit for processing of the sequence data. For instance, bidirectional recurrent neural networks utilize the information of the whole sequence. While time is a multi-aspect entity [21–23], the long short-term recurrent neural networks can also retain the information over long periods of time [47]. Deep convolutional neural fields involve more sequence information in its learning process and take into account the interdependencies of the adjacent context [58]. The more recent studies both in shallow [1,57] and Deep Neural Networks (DNN) [51], combine the predictions of a number of such networks in an ensemble fashion. Spencer et al. [51] propose an ensemble of three DNNs with a cascade architecture. The model is trained using the restricted Boltzmann machine that works with the real-valued data and the contrastive divergence. Despite the strength of neural networks, the selection of proper parameters - e.g. number of neurons, layers, and activation functions - remains an issue. This can significantly affect the prediction outcome. Moreover, there is a chance that the algorithm falls into local minima. Since the support vector machines can resolve parameter selection and the local minima issues, we employ the SVM component.

SVMs are among the most accurate learners in the literature of protein secondary structure prediction [6,64]. Because of the optimization nature, SVM models perform more accurately than NNs in many applications [6,64]. However SVM kernel should be tuned properly [63,64]. use a dynamic weight allocation function to assign kernel weights and fuse them into a single kernel. Furthermore, a parallel hierarchical grid search is applied to tune the kernel parameters. According to Ref. [63], converting the classification to a regression problem and then employing Support Vector Regression (SVR) can further enhance the prediction accuracy. Therefore, aiming to determine the final protein structure [63], utilizes a non-dominated sorting genetic algorithm to map the SVR outputs to integer values. The SVM models are also employed as ensemble components [32,46]. Nguyen et al. [46] develop an SVM-based cascade architecture where the second layer produces the final prediction results through combining the outputs from the first layer.

Multi-component methods employ a variety of complementary modules to unveil the relationship between the input and output vectors. Some categories of such computational modules are various learners, optimization strategies, distance or dissimilarity measures, and evolutionary algorithms. In practice, the multi-component machines perform more competently than single learners. The reason lies on the fact that each component can overcome a part of the challenge. Several methods reviewed in this section were developed in an ensemble or multi-component manner. The first group of multi-component approaches [8,14,48,57,61] employ complementary modules alongside the learning algorithms to promote the prediction results. Similarly, aiming to foster prediction accuracy, the second category [1,3,4,25,46] exploit multiple classifiers of the same type with various features. The third class of Multi-component approaches [6,20,51,58,62–64] combine classifiers of different types where some are equipped with complementary components. In this work, we devise a multi-component framework to better tackle each of the challenges prediction of the protein secondary structures.

Feature extraction: With the explosion of available unsolved protein sequences in the post-genomic era, exploiting the computational protein structure predictors became inevitable. Almost all of such conventional predictors will effectively learn by consuming numerical feature vectors. The effectiveness of these learners strictly depends on how well the extracted features represent the characteristics and patterns existing in the original sequences. To generate numerical vectors from the protein sequences, various encoding schemes, sequence profiles and amino-acid features are employed. One of the simplest encodings is the orthogonal matrix [20] which takes a vector of 20 elements that contains 19 zeroes and only one element of 1 related to the corresponding amino-acid. The mostly utilized sequence profiles are PSSM [1,4,25,36,51,62,63] and similar matrices. The PSSM profiles incorporate proteins evolutionary information and preserve sequence order information. The most prevalent features used in the literature are pseudo-potentials [61], amino acid physiochemical properties [30], AACs, pseudo-AACs [12] and PseKNC [9]. Pseudo-potentials is a type of context-based score, which estimates the tendency of a residue to adopt a certain secondary structure among a certain neighboring residues. Some of the physio-chemical properties of amino-acids used in effective structure prediction include *dissociation constant values*, *isoelectric point*, *relative molecular mass*, *specific rotation* and the *hydrophobicity scale*. AACs and pseudo-AACs describe proteins using the frequency matrix of amino-acids. While AAC do not involve the sequence order information, pseudo-AACs do not completely lose the sequence order information and include the correlation among residues as well. PseKNC is another discrete vector representation of a sequence (DNA or RNA), which is capable of preserving global or long-range sequence order information. Recently a number of online servers have been developed for generating features from protein sequence. Some of these servers are *iFeatures* [11], *SPiCE* [55], *PyFeat* [44], and *Pse-in-One* [33,34]. The Pse-in-One feature generation server is much more flexible in the sense that it can produce nearly all possible feature vectors for DNA, RNA or protein sequences as well as the user-defined features.

Although the reviewed sequence features of protein are informative, yet the original sequence contains more information and patterns. As a number of methods consider the original sequences of protein as sentences of an unknown language, attempt to directly process sequences and bypass feature extraction in the hope of losing less information [31]. Consequently, in this work, we take advantage of NLP concepts and measures with the aid of both machine learning and template-based methods to gain a deeper understanding of sequence-structure correlation.

3. Problem statement

In this section, we offer primary concepts, notations, and the framework overview.

3.1. Preliminary concepts

We firstly declare biological concepts. The Gene or DNA sequence in every cell does not control genetic properties on its own; rather, this is done through the translation of DNA sequence into protein and then the formation of a specific structure. Hence, the proteins are the functional units of the cells whose functions are tightly connected to their structures.

Definition 1. (protein sequence) A protein sequence P_i, of the length l, is a string composing of l amino-acids (i.e. $P_i[m]$) participating in protein's formation. Each amino-acid molecule in a protein sequence is represented by an alphabetic letter ($P_i[m] \in \sum$).

Definition 2. (protein secondary structure) The secondary structure of a protein is formed by the local compositions of neighboring amino-acids through peptide bonds. During this chemical reaction, the element of water is removed and what is left of the amino-acid molecules is called amino-acid residues. Thus we refer to amino-acid residue as residue from the hereafter. Every residue $P_i[r]$ is assigned with a secondary structure (i.e. $S_i[r]$). There are three classes - in three letters - of protein secondary structure. Therefore, $S_i[r] \in \{H, E, C | H \equiv \alpha - helix, E \equiv \beta - sheets, \text{and } C \equiv coils\}$. Hence, a protein sequence with l characters will correspond to a sequence of secondary structures with similar length of l. Each character in the structure sequence (i.e. $S_i[r]$) is associated with its corresponding amino-acid (i.e. $P_i[m]$) in the primary protein sequence.

Definition 3. (the dissimilarity rate) the dissimilarity rate ρ_d is the number of unique non-identical characters divided by the number of unique identical characters that a pair of sequences share. We disregard the position of characters.The secondary structure of a residue is strongly influenced by the type of its neighboring residues [28]. Therefore, to effectively predict a residue's structure, its neighbors must be involved. We employ the simple but effective sliding window approach to include the adjacent residues in the prediction process.

3.2. Problem definition

Given a set of protein sequences P_i, we aim to model the mapping, denoted by $f: P_i \rightarrow S_i$. Therefore, we infer the similarity between the sequences in P_i via a compound dissimilarity measure. Concurrently, to address the complexity of the function f, we devise our unified framework as a multi-component learning machine.

Problem 1. (sequence similarity inference) Given a set of protein sequences P_i, we aim to use a Compound Dissimilarity measure (CD) to compute the similarity between each pair of sequences.

Problem 2. (multi-component learning) Given our compound dissimilarity measure (CD) and the protein sequences P_i, our goal is to devise a multi-component learning machine to take P_i as input and consume protein sequence dissimilarities.

3.3. Framework overview

Fig. 2 depicts our proposed framework for prediction of protein secondary structure. Since each component contributes to error correction and enhances the prediction accuracy, the multi-component framework can better address the mapping procedure (f). The strength of a learning machine mainly stems from the diversity of its components and the effectiveness of its aggregation method. We employ a pair of diverse classifiers (SVM and FKNN) to form the learning core.

Initially, a *sliding window* of size h chunks the input protein sequences into the fixed-length set of strings (P^h). Including $h-1$ neighbors of the central residue from the sliding window involves the long-range interactions among amino-acids. These interactions are a piece of valuable information for the prediction.

In the *learning phase*, to infer the f mapping, the set of P^h is fed into parallel classifiers of $\tilde{d}$-FKNN and edit-SVM. As both classifiers directly learn from P^h, we can bypass feature extraction. The $\tilde{d}$-FKNN processes P^h using the CD measure ($\tilde{d}$) to find the difference between each pair of sequences ($p_i^h, p_j^h \in P^h$) via LZ, n-gram, and dissimilarity ratios (i.e. ρ_d). LZ complexity score explores sequence order information, repeated patterns, and the degree of randomness [35,53]. The n-gram score captures inner-sequence local similarities to reflect sequence variations during evolution [31]. Finally, the dissimilarity rate reflects the type diversity of the amino-acid molecules in p_i^h. Fusing these scores into a dissimilarity measure can better infer the sequence-structure relations. The *edit* kernel enables SVM to effectively handle string sequence data (Section 6.2). Since $\tilde{d}$-FKNN and edit-SVM learn in parallel, better efficiency is provided. The output of each learning module passes through a *filtering* component to eliminate the biologically meaningless structures. In the *aggregation pool*, five various aggregation rules are accommodated. Each aggregation rule makes a consensus between the decisions of $\tilde{d}$-FKNN and edit-SVM in a different fashion. The fuzzy property of KNN can further enhance the aggregation process. The final secondary structure (i.e. S_i) can be obtained through filtering of the aggregation results.Q. What necessity for sequence processing? the effective measure of dissimilarity in sequence processing has a few advantages over processing of the extracted numerical feature vectors. First, it prevents information loss from the rich protein sequences. Second, the learner's performance varies on different sets of numerical features and it is not always feasible to find the optimal feature set. Additionally, numerical features or encoding schemes (i.e. numerical representations for string protein sequences) may lead to high dimensional feature vectors which can negatively affect the learner's performance [35]. From the biological perspective, every protein sequence contains all essential information for structure adoption [31,35,36]. Hence, we utilize the compound dissimilarity measure $\tilde{d}$ alongside with the *edit* kernel to process the string sequences effectively.

4. Related material

In this section, we elucidate the prediction procedure, datasets, the test and train methods, and finally the evaluation metrics.

4.1. Step-by-step prediction procedure

Every effective sequence-based prediction model considers five steps:

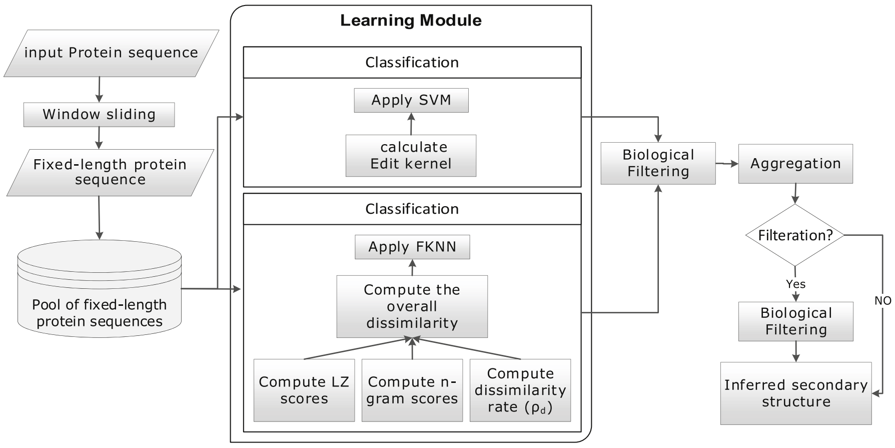


Fig. 2. The framework of our proposed approach.

I) To select a valid test and train benchmarks via multiple data sets to verify experimental results.
II) To properly encode the input biological sequence and feed it into a predictor. The encoded format must preserve as much information as possible in comparison with the original data and also have a high correlation with the target label. It is important to mention that the input representation greatly affects the performance of the predictors. Hence the representation of protein sequences for a predictor is a challenge and must be thoughtfully addressed.
III) To design a powerful prediction engine to effectively address the challenges of secondary structure prediction. The more robust, accurate and efficient the engine is, the more effective it will function.
IV) To select an appropriate evaluation framework including proper and comprehensive evaluation measures and methods to adapt to the conditions of the problem. Given the classification problem of structure prediction, the measures of accuracy, specificity, sensitivity, MCC, and SOV are used frequently and can properly describe the effectiveness of a competitor model. Moreover, the cross-validation is utilized to tune parameters in train and test phases.
V) To establish a public user-friendly web-server which can receive the user's input protein data and predict the secondary structures using the proposed method(s). Throughout this paper, we elaborate on each step in details.

4.2. Test, train, and dataset description

We employ four publicly available data sets of RS126, CASP10, CASP11 and CASP12 (more recent) to validate the effectiveness of our approach. To accomplish a comprehensive validation we first employ RS126 for training and testing via 10-fold cross-validation and report the results. Then we utilize CASP10, CASP11 and CASP12 (parameter tunig is conducted on an entirely different dataset) to further confirm our achievements. Table 2 reports some statistics about the datasets.

The RS126 dataset contains globular non-homologous proteins. The *coil* with a portion of 45% is the most common structure in the dataset, while the other 23% and 32% of the residues are respectively categorized as *β-sheet* and *α-helix*. CASP datasets contain proteins which are regarded as hard targets meaning that it is difficult to detect their homologous structure templates from known protein structures. These datasets were employed to select the best prediction models in the worldwide annual competition of CASP (http://predictioncenter.org). Only proteins with and no missing and available crystal structure were selected from the CASP datasets.

4.3. Evaluation measures

To comprehensively evaluate our approach, we employed 5 common evaluation measures for both classifiers and protein structure predictors. The measures include accuracy (Q_3), precision, recall, specificity, and Matthews Correlation Coefficient (MCC) that can be computed by confusion matrix elements (i.e. TP, FP, TN, and FN). TP, FP, TN, and FN respectively stand for True Positive, False Positive, True Negative and False Negative. TP_j is the number of residues belonging to class j that have been predicted as having the class j. FP_j is the number of residues not belonging to class j that have been predicted as having the class j. TN_j is the number of residues not belonging to class j that have been predicted as having other class labels. FN_j is the number of residues belonging to class j, but they are predicted as other classes.

Table 2
Statistics of datasets.

Data set	No of proteins	No of residues	Year of creation
RS126 [50]	126	32465	1993
CASP10 [41]	73	16568	2012
CASP11 [42]	64	13517	2014
CASP12 [43]	24	5032	2016

4.4. Availability

The link to download our codes is available at (https://sites.google.com/view/protein-structure-prediction):

5. Methodology

In this section, we explain the components of our proposed framework.

5.1. Pre-processing

Conventional machine learning-based methods generate numerical feature vectors from the sequences of primary structures. The vectors are subsequently fed into a learning machine for structure prediction. Although such numerical features comprise either protein evolutionary information or biochemical properties, in comparison with the original sequence, they result in information loss. Moreover, some numerical encoding schemes can even create high dimensional vectors. Note that various collection of numerical features can influence the performance of a learner. In a nutshell, all aforementioned issues with the numerical features can reduce the effectiveness of the learners. Hence, we bypass feature extraction and directly process the input protein sequences. Nevertheless, for pre-processing, we only apply a sliding window (size $h = 17$) [51,63] on input sequences. The neighbors of a residue in a protein sequence contribute strongly to its secondary structure. We predict the secondary structure of each residue at the center of the window, that directly involves $(h-1)/2$ number of neighbors on both sides. Assuming that there are M residues in the protein dataset, sliding the window will then construct M sequences (i.e. P_i^h), where each P_i^h will position one of the residues at the center. Note that, large h will deviate the dissimilarity values between the pair of sequences. The small value of h (i.e. $h < 5$) is not preferred either. Because the dissimilarity measure includes computation of n-gram scores and small n is biologically meaningless. Logically the value for n must be smaller than h.

5.2. The compound dissimilarity measure

The $\tilde{d}$-FKNN component performs classification based on the compound dissimilarity measure $\tilde{d}$ that is computed using three parameters including LZ-scores, n-gram scores, and the ρ_d dissimilarity rate. Adding each parameter to $\tilde{d}$ can further improve the accuracy (Section 6.2).

The LZ complexity measure reflects the degree of repeated patterns or the level of randomness in a sequence and can include the position information [35,53]. Assume $p_i^h[m: n]$ is a fragment of a protein sequence starting at position m and ending at position n where $1 < m < n < h$. Thus we can show that $p_i^h = p_i^h[1: m_1]p_i^h[m_1 + 1: m_2]...p_i^h[m_{k-1} + 1: m_k]...p_i^h[m_{z-1} + 1: h]$. The LZ complexity of the protein sequence $c(p_i^h)$ is the number of fragments in the exhaustive history that represents the decomposition of the sequence. Each fragment obtained from the decomposition process must be unique except in the last step, at which it is permitted to copy a previously generated fragment.

For example, for the protein sequence $p_i^h = TTCCPSTCIVPSA$ the exhaustive history is *T.TC.C.P.S.TCI.V.PSA* where '.' separates the fragments generated at each step. Hence $c(p_i^h)$ is 8 which is the number of fragments in the exhaustive history. The initial LZ score between two protein sequences $(\zeta\left(p_i^h, p_j^h\right))$ is defined in Eq. (1) [35].

$$\zeta\left(p_i^h, p_j^h\right) = c\left(p_i^h p_j^h\right) - c(p_i^h) \tag{1}$$

Where $p_i^h p_j^h$ is the concatenation of two protein sequences of p_i^h and p_j^h. The more similar the two sequences are, the less the value of $\zeta(.)$ will be. The final LZ score (the LZ dissimilarity of two sequences) is attained from the normalization of Eq. (1) as formulated in Eq. (2). We utilize the normalized LZ score in our work [35].

$$LZ = \frac{max\left\{\zeta\left(p_i^h, p_j^h\right), \zeta\left(p_j^h, p_i h\right)\right\}}{max\left\{c(p_i^h), c\left(p_j^h\right)\right\}} \tag{2}$$

Algorithm 1 generates a unique [29] exhaustive history for a sequence.

Algorithm 1. create the exhaustive history of a protein sequence
Input:p_i^h (a protein sequence with length h).
Output: $\eta(p_i^h)$ (set of fragments related to the exhaustive history of protein sequence).

```
1: $\eta(p_i^h) \leftarrow \varnothing,\ i = 1,\ j = 1$
2: $\eta(p_i^h) \leftarrow p_i^h[i],\ i + =1,\ j + =1$
3: **if** $j \geq h$ **then**
4: $\eta(p_i^h) \leftarrow p_i^h[i: j]$ and terminate
5: **end if**
6: **if** $p_i^h[i: j] \notin \eta(p_i^h)$ **then**
7: $\eta(p_i^h) \leftarrow p_i^h[i: j], i = j + 1, j + =1$, **go to** 3
8: **else**
9: $j+ = 1$, **go to** 3
10: **end if**
```

LZ complexity for a sequence considers the character distribution rather than the characters themselves. Hence, this property leads to the same complexity for sequences of the same distributions, but with different characters. For example, the exhaustive history of the sequences two sequences of $p_i^h = APAFSVSGG$ and $p_j^h = THTDKRKLL$ can be respectively represented by $\eta(p_i^h) = A.\ P.\ AF.\ S.\ V.\ SG.\ G$ and $\eta\left(p_j^h\right) = T.\ H.\ TD.\ K.\ R.\ KL.\ L$. However, the LZ complexity for both strings will be identically set to 7. Nevertheless, each amino-acid brings along distinct properties to form a certain secondary structure. Therefore, to include sensitivity to the type of amino-acid molecules, we employ a new parameter called dissimilarity rate ρ_d. Let π_i^m and π_j^m be the respective list of unique amin-acids composing p_i^h and p_j^h while $m_i \in \pi_i^m$ and $m_j \in \pi_j^m$. Here, ρ_d is retrieved by substituting of Eq. (4) and Eq. (5) into Eq. (3).

$$\rho_d = \frac{1 + |\delta_{i,j}|}{1 + |\mu_{i,j}|} \tag{3}$$

$$\delta_{i,j} = \{m_i \cup\ m_j | m_i \in \pi_i^m and\ m_j \in \pi_j^m\} \backslash \{m_i | m_i \in \pi_i^m and\ m_i \in \pi_j^m\} \tag{4}$$

$$\mu_{i,j} = \{m_i | m_i \in \pi_i^m and\ m_i \in \pi_j^m\} \tag{5}$$

Local similarities among protein sequences can identify conserved structures during proteins' evolution [31]. To incorporate the local similarities into our measure, we employ n-gram score $|\Gamma i, j^n|$ between each pair of sequences (p_i^h, p_j^h). The larger the n, the more strictly the similarity will be computed. Let Γ_i^n and Γ_j^n be the respective sets of n-gram patterns associated with the protein sequences of p_i^h and p_j^h where $\gamma_i^n \in \Gamma_i^n$ and $\gamma_j^n \in \Gamma_j^n$. We compute the n-gram score ($\Gamma_{i,j}^n$) using Eq. (6).

$$\Gamma_{i,j}^n = \{\gamma_i^n | \gamma_i^n \in \Gamma_i^n\ and\ \gamma_i^n \in \Gamma_j^n\} \tag{6}$$

Algorithm 2 shows how we obtain the n-gram patterns of a sequence.

Algorithm 2. generate the n-gram patterns of a protein sequence
Input: p_i^h (a protein sequence with length h).
Output: $\Gamma_{i,j}^n$ (the set of n-gram patterns ($n < h$) related to the input protein sequence).

```
1: $\Gamma_{i,j}^n \leftarrow \varnothing,\ i = 0$
2: **while** $i + n \leq h$ **do**
3: $\Gamma_{i,j}^n \leftarrow p_i^h[i + 1: i + n], i+ = 1$
4: **end while**
```

We fuse LZ, n-gram ($|\Gamma i, j^n|$) and the dissimilarity scores (ρ_d) into Eq. (7) to create our final compound dissimilarity measure ($\tilde{d}$).

$$\tilde{d} = \frac{max\left\{\zeta\left(p_i^h, p_j^h\right), \zeta\left(p_j^h, p_i h\right)\right\} \times (1 + |\mu_{i,j}|)}{max\left\{c(p_i^h), c\left(p_j^h\right)\right\} \times (1 + |\delta_{i,j}|) \times (1 + |\Gamma_{i,j}^n|)} \tag{7}$$

5.3. Fuzzy KNN algorithm

Unlike rule-based methods and decision trees [52], the KNN can implement irregular decision boundaries. However, the effectiveness of KNN method [19,54] is influenced by distance measure. We use dissimilarity measure ($\tilde{d}$) in the membership function ($U_c(.)$) of the fuzzy KNN ($\tilde{d}$-FKNN) to find the nearest neighbors. FKNN does not return one secondary structure for each input residue ($p_i^h[r]$). Rather, it produces the likelihood for the input residue to adopt each secondary structure. Eq. (8) [26] computes the fuzzy membership values of a test residue ($p_i^h[r]$) in each secondary class.

$$U_c(p_i^h[r]) = \frac{\sum_{j=1}^{k} I_{cj}\left(\frac{1}{\left|\left|p_i^h - p_j^h\right|\right|^{\frac{2}{m-1}}}\right)}{\sum_{j=1}^{k}\left(\frac{1}{\left|\left|p_i^h - p_j^h\right|\right|^{\frac{2}{m-1}}}\right)} \tag{8}$$

$U_c(p_i^h[r])$ is the likelihood for the test residue $p_i^h[r]$ to adopt class c. k is the number of neighbors, $\|.\|$ denotes the degree of dissimilarity between p_i^h and its neighbor (p_j^h), and m is the degree of fuzziness. Finally, U_{cj} is the initial fuzzy membership value of the neighbor p_j^h for class c. The likelihood for $p_i^h[r]$ to adopt any a is dependent on the membership values of the neighbors and the inverse dissimilarity between $p_i^h[r]$ and the neighbors ($p_j^h[r]$). The inverse dissimilarity ($\frac{1}{\left|\left|p_i^h - p_j^h\right|\right|}$) determines how each neighbor can enforce the membership of the input residue in class c. As far the neighbor from the input residue, the less weight will be assigned to it [42]. Eq. (9) [26] computes the initial membership values for the neighbors ($I_c\left(p_j^h\right)$). The class of the training residue with known structure (p_j^h) is s.

$$I_c\left(p_j^h[r]\right) = \begin{cases} 0.51 + \left(\frac{n_j}{k'}\right) \times 0.49 & \text{for } c = s \\ \left(\frac{n_j}{k'}\right) \times 0.49 & \text{for } c \neq s \end{cases} \tag{9}$$

Where $I_c\left(p_j^h[r]\right)$ is the corresponding membership value for training residue $p_j^h[r]$ in class c, k' is the number of neighboring residues of p_j^h, and n_j is the number of neighboring training residues of p_j^h that belong to class s. $I_c(.)$ assigns three class values of fuzzy membership ($c \in \{H, E, C\}$) to each training residue. If the label of $p_j^h[r]$ and its neighbors is c, $I_c\left(p_j^h[r]\right)$ will get the full membership of 1 for class c. If $p_j^h[r] \neq c$ or its neighbors do not belong to class c, $p_j^h[r]$ will obtain the membership value for class c. In other words, the function $I_c(.)$ aims to fuzzify the class membership of the labeled residues (with known structure) which lie in the intersecting region of three classes in the sample space. As the value of $U_c(p_i^h[r])$ is computed using $I_c\left(p_j^h[r]\right)$, the class of unlabeled residues ($p_i^h[r]$) located in the intersecting region of classes will be less influenced by the labeled residues ($p_j^h[r]$) lying in the intersecting region [26]. Assignment of the initial fuzzy membership values to the training residues in fuzzy KNN can be considered a training phase, which leads to performance enhancement compared to the crisp KNN. Note that the fuzzy membership can assert prediction confidence and associate an input data with a class.

5.4. Edit-SVM algorithm

The SVM method finds the global optimum solution and discovers the boundary with the maximum margin. It can search for high dimensional feature spaces to find the separating hyperplane - instead of a non-linear boundary [19]. Using kernel tricks, the transformation to higher dimensions does not need to be directly calculated and thus the efficiency of the algorithm remains high. As a component in our framework, SVM can efficiently outperform both neural network and individual learners in prediction task [6,64]. The SVM module processes the protein language and predicts the structure of the residue at the center of sliding window. We also employ a kernel to process string data. Like *RBF* kernel, the well-performing *edit* kernel [2] (Eq. (10)) uses the *edit distance edit*(*x*,*y*) to further map the original feature space to an infinite dimensional space [54,64].

$$K(x, y) = e^{\gamma \cdot edit(x,y)} \tag{10}$$

Given two strings of x and y from the set of Σ alphabet, we use *Levenshtein* edit distance $edit(x, y)$ to transform x to y by a minimum number of operators [56].

5.5. Biological filtering

Considering a protein sequence of p_i^h, composed of n amino-acids and three classes of secondary structures, the prediction vector will have 3^n different states. However, not all of the states are biologically meaningful (e.g. *H*). In order to rectify the prediction output, we apply a set of biological transformations [1,5] on the secondary structure sequences: $EHE \rightarrow EEE,\ HEH \rightarrow HHH,\ HCH \rightarrow HHH,\ ECE \rightarrow EEE$ and $HEEH \rightarrow HHHH$.

In some methods with more single false predictions (not contiguous mis-predictions), the filtration process can foster accuracy, but for other cases, it may result in a minor improvement.

5.6. Aggregation rules

Aggregation is an influential module in the effectiveness of a multi-component learner [6]. In our work, five various aggregation rules are proposed to produce the final secondary structures. The aggregators take advantage of the output from $\tilde{d}$-FKNN and edit-SVM classifiers. $\tilde{d}$-FKNN generates three fuzzy membership values which provide a confidence level for each decided class and enhance the aggregation process. The accuracy of our method using each aggregation rule is evaluated in section 6. Let $p_i^h[r]$ be the r^{th} residue of the protein sequence p_i^h. Also, assume Δ^1_{FKNN} and Δ^2_{FKNN} to be the first and second decisions (associated with the class with maximum and mid membership values) made by $\tilde{d}$-FKNN classifier. Suppose $s_i[r]$ to be the predicted secondary structure for the residue $p_i^h[r]$.

Aggregation 1:

```
$\forall\ p_i^h[r],\ \ if\ \{\Delta^1_{FKNN} = \Delta_{SVM}\}\ then$
$s_i[r] = \Delta^1_{FKNN}$
else
$s_i[r] = \Delta^2_{FKNN}$
endif
```

According to aggregation 1, if both $\tilde{d}$-FKNN and edit-SVM vote to a class, it will be the final prediction result. Otherwise, based on the fuzzy levels of confidence, the second decision of $\tilde{d}$-FKNN (Δ^2_{FKNN}) will most likely be the next choice. Aggregations 2, 4 and 5 work based on the weighted decisions of the classifiers. We introduce two strategies for weight assignment. The first strategy assigns a weight proportional to the accuracy of the classifier on a validation set. Accordingly, the more accurate the classifier, the higher its priority will be. Let ω_1 and ω_2 be the weights for either $\tilde{d}$-FKNN or edit-SVM, where $\omega_1 \leq \omega_2$ and $\omega_1 + \omega_2 = 1$. Accordingly, Eqs. (11) and (12) are used to compute the weights.

$$\omega_1 = \frac{min\{\alpha_{fknn}, \alpha_{svm}\}}{\alpha_{fknn} + \alpha_{svm}} \tag{11}$$

$$\omega_2 = 1 - \omega_1 \tag{12}$$

The interval of [0,1] is divided into two sub-intervals proportional to the weight of each classifier. If $\omega_1 = 0.6$ and $\omega_2 = 0.4$, the sub-intervals of $i_1 = [0–0.6]$ and $i_2 = (0.6–1]$ will be dedicated to the first and the second classifiers respectively. Finally, we generate a random number r in the interval of [0,1]. If the value of r lies in i_1, the first classifier will decide the final class. Otherwise, the second classifier will determine the final prediction. The length of the dedicated sub-interval to a certain classifier is the probability that the decision of that classifier is returned as the final decision. We call this strategy as *Roulette Wheel 1* which is particularly more applicable when the accuracy of the classifiers are not level. *Roulette Wheel 2* differs from the *Roulette Wheel 1* if a sub-interval is dedicated to each classifier. Instead of assigning the sub-intervals with the weights of the classifiers, *Roulette Wheel 2* considers a step size and examines the resultant accuracy of the prediction machine. Here we appoint different breakpoints over the interval of [0,1] and the selected breakpoint assigns a sub-interval to each classifier. For example, if the step size is set to 0.1, the breakpoints {0.1,0.2,0.3,..,0.9} will be examined and the sub-intervals of the classifiers can be appointed in pairs as $i_1 = [0,0.1]$ and $i_2 = (0.1,1]$, $i_1 = [0,0.2]$ and $i_2(0.2,1]$, $i_1 = [0,0.3]$ and $i_2(0.3,1]$,..., and $i_1 = [0,0.9]$ and $i_2 = (0.9,1]$. In this work, the accuracy of the two classifiers is nearly even. Hence, we employ the weight assignment strategy of *Roulette Wheel 2* in aggregation rules of 2, 4 and 5. Fig. 3 depicts the change in prediction accuracy when the value of breakpoint increases.

The accuracy is the average of 15 generated random numbers (r) for each breakpoint. The best accuracy (breakpoint 0.75) assigns the interval of [0,75) to $\tilde{d}$-FKNN, and [0.75,1] to edit-SVM. Aggregation 2 is described below:

Aggregation 2:

```
$\forall\ p_i^h[r],\ \ if\ \ \Delta^1_{FKNN} = \Delta_{SVM}\ \ then$
$s_i[r] = \Delta^1_{FKNN}$
else
if $r \leq \omega_{FKNN}$
$s_i[r] = \Delta^1_{FKNN}$
else
$s_i[r] = \Delta_{SVM}$
endif
endif
```

Where Δ^1_{FKNN} and Δ_{SVM} do not equate, the *Roulette Wheel 2* determines the final prediction based on Δ^1_{FKNN} and Δ_{SVM}.

Aggregation 3:

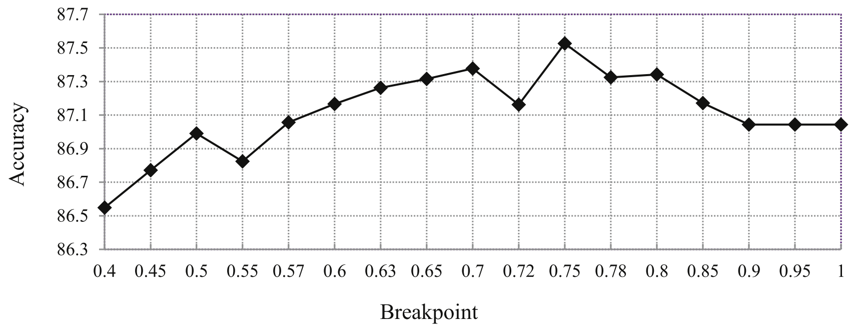


Fig. 3. Impact of interval breakpoints on accuracy.

Table 3
The accuracy of MCP and its components on RS126 dataset.

Method	Q3
FKNN + LZ	76.38
FKNN + LZ + ρ_d	80.42
FKNN + LZ + ρ_d + n-gram ($\tilde{d}$)	81.96
Edit-SVM	82.2701
MCP1	83.0906
MCP2	**85.4072**
MCP3	75.487
MCP4	80.8661
MCP5	**87.2853**

```
$\forall\ p_i^h[r],\ if\ \Delta_{FKNN}^1 = \Delta_{SVM}$
then
$s_i[r] = \Delta_{FKNN}^1$
else
$s_i[r] = \Delta_{FKNN}^3$
endif
```

Aggregation 3 claims that when Δ_{FKNN}^1 differs from Δ_{SVM}, the last fuzzy decision of KNN (i.e. Δ_{FKNN}^3) will predict the final structure. Intuitively, selecting the last fuzzy decision is not necessarily the best choice. We opt for the last fuzzy decision which complies with the result of edit-SVM and the sequence-structure relation.

Aggregation 4:

```
$\forall\ p_i^h[r],\ if\ \Delta_{FKNN}^1 = \Delta_{SVM}\ then$
$s_i[r] = \Delta_{FKNN}^1$
else
if $r \le \omega_{FKNN}$
$s_i[r] = \beta(\Delta_{FKNN}^1)$
else
$s_i[r] = \beta(\Delta_{SVM})$
endif
endif
```

Aggregation 5 differs from 2 as it receives the output from $\tilde{d}$-FKNN ($\beta(\Delta_{FKNN})$) and edit-SVM ($\beta(\Delta_{SVM})$). Due to relying on early fuzzy decisions, Aggregation 2 and 5 provide the best results.

Proof. ***Performing classification using MCP on an arbitrary number of classes*** Let the number of classes be an arbitrary value l. There will thus be a unified final result for Δ_{SVM} which is the voting between the decisions of $\frac{l(l-1)}{2}$ SVM models (one-versus-one strategy [59]). Also, there will be l decisions associated with FKNN as Δ_{FKNN}^1, Δ_{FKNN}^2, ..., Δ_{FKNN}^l where Δ_{FKNN}^1 and Δ_{FKNN}^l correspond to the secondary class with the maximum and minimum values of the fuzzy membership function. In aggregation rules of 1, 2, and 5, we take advantage of the decision of SVM (Δ_{SVM}) and the first decision of FKNN (Δ_{FKNN}^1). Therefore, MCP can perform classification on an arbitrary number of classes.

6. Experiment

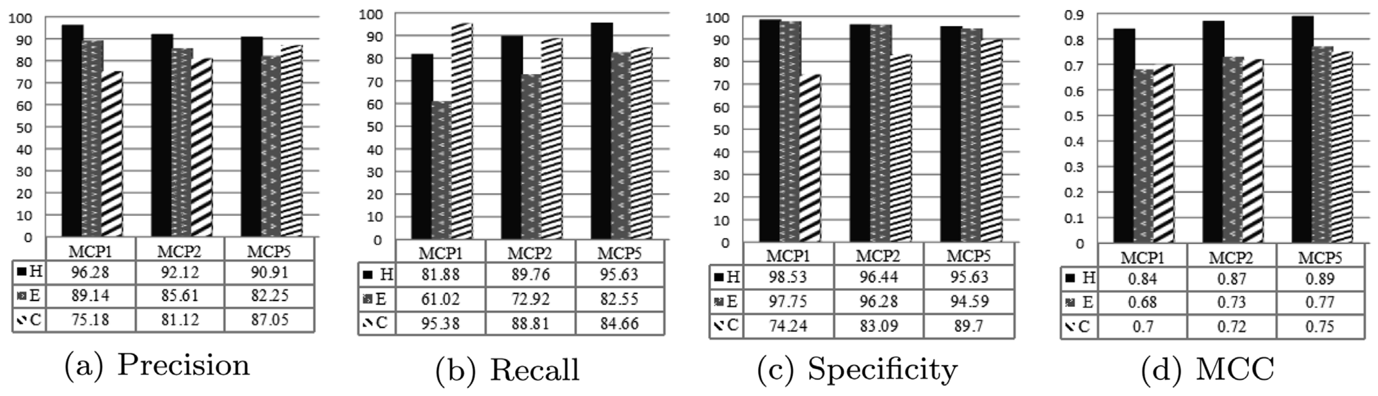
We conducted comprehensive experiments on real-world RS126 benchmark dataset to evaluate the effectiveness of our framework against state-of-the-art models in protein structure prediction. We also considered four widely-used evaluation measures of accuracy, recall, specificity, and Matthews Correlation Coefficient (MCC) to compare the classification baselines.

6.1. Framework evaluation

In this section, the effectiveness of our framework and its components are comprehensively investigated on the introduced datasets. We used Q_3 to gradually evaluate the performance of our proposed framework. Table 3 compares the effectiveness of various aggregation rules. It also shows that each component beside the primary ensemble can enhance the overall performance of the framework. As it's evident from the table, extending the FKNN model with the dissimilarity rate (ρ_d) can improve the accuracy. Also, the multi-component variations (i.e. MCP1 to MCP5) remarkably perform better than the single classifiers (e.g. $\tilde{d}$-FKNN and edit-SVM). Our multi-component framework gains the best effectiveness as it employs a new dissimilarity measure alongside with different aggregation rules. Aggregation rules of 1, 2 and 5 perform better than other variations. Because they utilize the first fuzzy and edit-SVM decisions beside filtration process. Note that we exclude fuzzy rules of 3 and 4 as they gain lower accuracies.

Fig. 4 compares various versions of our framework (MCP) on RS126 dataset. According to Fig. 4(a), MCP1 has the highest precision in predicting H and E structures. However, it shows a low precision for class C. Also, compared to MCP1, the MCP2 model demonstrates a better precision in class C. Moreover, since the transformations of $HCH \rightarrow HHH$ and $ECE \rightarrow EEE$ reduce the number of false-positives, the extra filtration in MCP5 significantly promotes the precision for the C structures. Despite the fact that MCP5 gains a lower precision for H and E structures as shown in Fig. 4(b), MCP5 notably shows a better recall compared to MCP1 and MCP2 in the prediction of H and E structures (see Fig. 5).

As depicted in Fig. 4(b), MCP1 is sensitive to the class imbalance. Since the higher portion of structures respectively belongs to class C, H, and E, MCP1 tends to predict each sample as C, H, and E respectively. This imbalanced tendency leads to lower false negative predictions for each of the classes based on the number of data samples. Hence, MCP1 exhibits the maximum and minimum recall values for the classes of C and E. Nevertheless, while MCP2 partially resolves this sensitivity, MCP5 remarkably eliminates the effect of the class imbalance. As illustrated in Fig. 4(c), all the variations of our model demonstrate a very high and level values of specificity for H and E. For class C, specificity is significantly improved via MCP2 and MCP5. Except for MCP1 in class C, the values of specificity for all variations of our method and particularly in every class of the secondary structure is remarkably higher compared to the values of other metrics. Since the rate of true-negative compared



(a) Precision	MCP1	MCP2	MCP5
■ H	96.28	92.12	90.91
▨ E	89.14	85.61	82.25
▧ C	75.18	81.12	87.05

(b) Recall	MCP1	MCP2	MCP5
■ H	81.88	89.76	95.63
▨ E	61.02	72.92	82.55
▧ C	95.38	88.81	84.66

(c) Specificity	MCP1	MCP2	MCP5
■ H	98.53	96.44	95.63
▨ E	97.75	96.28	94.59
▧ C	74.24	83.09	89.7

(d) MCC	MCP1	MCP2	MCP5
■ H	0.84	0.87	0.89
▨ E	0.68	0.73	0.77
▧ C	0.7	0.72	0.75

(a) Precision (b) Recall (c) Specificity (d) MCC

Fig. 4. Effectiveness comparison between MCP1, MCP2, and MCP5 on RS126.

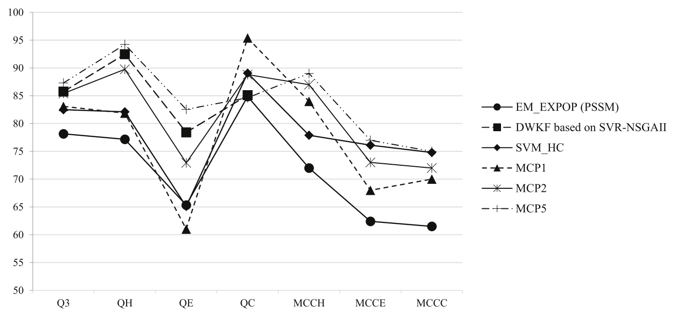


Fig. 5. Comparison of the accuracy and MCC between the best baselines on RS126.

to false-positive is large, our proposed method reaches higher effectiveness. As the MCC metric concurrently considers all the elements of the confusion matrix, it can reflect reliable evaluation outcomes. The MCC is further useful for the class imbalance issue, where the size of the classes is different. Fig. 4(d) shows high values of MCC for class *H* and even values for *E* and *C* classes. With regard to the classification task, MCP5 and then MCP2 achieve the best performance.

Tables 4–6 compare MCP2 and MCP5 versus the initial classifier (MCP1) on RS126 dataset. For the class *H* (Table 4) MCP1 - unlike MCP5 - owns the lowest values of recall and MCC, and the highest values of precision and specificity. The same scenario holds for class *E* (Table 5). The behavior of the models is quite different towards class *C* (Table 6). Nevertheless, MCP5 acquires the highest values for precision, specificity, and MCC metrics. In total, MCP2 achieves moderate values for all the metrics in three classes. However, since MCP5 gains the highest MCC and Q_3 values for all three classes, it clearly outperforms other variations of our proposed framework.

Genuinely, there are 8 categories of protein secondary structures. The number of categories can be reduced to three by applying various reduction rules (e.g. DSSP [1]) on similar properties. According to Tables 4–6, the measures exhibit higher values for class *H* compared to other classes. The reason is that the diversity of the protein categories is less in class *H* and it sufficiently includes one-third of the dataset samples.

In order to further emphasize the capability of our model, we conducted additional experiments on the three recent and challenging datasets of CASP10, CASP11, CASP12. To train the Edit-SVM module, the kernel, and soft-margin parameters have been determined using the independent validation set of RS126 and then we conducted the train and test on each of the CASP datasets separately using 10-fold cross validation. Therefore, an independent portion of data is used for testing the method.

As is evident from Table 7, the Edit-SVM module is more competent than the $\tilde{d}$-FKNN component. However, the effectiveness of the $\tilde{d}$-FKNN module is greatly better than other known versions of distance-based algorithms (refer to Table 11). With the increase in the number of instances from CASP12 to CASP10 the accuracy of both modules increases since the size of the training portion grows. As expected, the MCP5 manifests higher accuracy than MCP2 while MCP2 shows better effectiveness compared to MCP1 (due to their competency in the corresponding aggregation rule) (see Table 8).

6.2. Comparison and discussion

We now compare our framework against the competitors listed below:

- *Muti-Component Predictor (MCP)*: While MCP1 utilizes Aggregation

Table 4
Effectiveness on class H of RS126.

Method	Precision	Recall	Specificity	MCC
MCP1	**89.13**	<u>61.02</u>	**97.74**	<u>0.68</u>
MCP2	85.61	72.92	96.28	0.73
MCP5	<u>82.25</u>	**82.54**	<u>94.59</u>	**0.77**

Table 5
Effectiveness on class E of RS126.

Method	Precision	Recall	Specificity	MCC
MCP1	**96.27**	<u>81.87</u>	**98.53**	<u>0.84</u>
MCP2	92.12	89.76	96.43	0.87
MCP5	<u>90.91</u>	**94.22**	<u>95.63</u>	**89**

Table 6
Effectiveness on class C of RS126.

Method	Precision	Recall	Specificity	MCC
MCP1	<u>75.18</u>	**95.38**	<u>74.23</u>	<u>0.7</u>
MCP2	81.12	88.81	83.09	0.72
MCP5	**87.05**	<u>84.65</u>	**89.69**	**0.75**

Table 7
of MCPs and its components on CASP datasets.

method	CASP10	CASP11	CASP12
Edit-SVM	86.14	85.67	85.47
FKNN + LZ + ρ_d + n-gram($\tilde{d}$)	82.45	81.44	80.78
MCP1	87.01	86.61	86.22
MCP2	87.92	87.86	87.36
MCP5	89.04	88.5	87.95

Table 8
Effectiveness of structure prediction on RS126-all versions of baselines.

Method Name	Q_3	Q_H	Q_E	Q_C	MCC_H	MCC_E	MCC_C
EM_EXPOP(SC) [6]	65.2	61.54	40.76	78.8	0.473	0.387	0.438
EM_LOGOP(SC) [6]	65.19	61.59	40.78	78.73	0.474	0.387	0.438
EM_LINOP(SC) [6]	65.19	61.59	40.78	78.73	0.474	0.386	0.438
EM_EXPOP(PSSM) [6]	78.14	77.18	65.34	84.86	0.72	0.624	0.615
EM_LOGOP(PSSM) [6]	78.12	77.2	65.28	84.83	0.72	0.624	0.615
EM_LINOP(PSSM) [6]	78.14	77.18	65.34	84.86	0.72	0.624	0.615
SVM-PHGS(DWKF) [64]	84.6	91.2	72.1	84.3	NA	NA	NA
SVR-NSGAII(DWKF) [63]	85.75	92.47	78.41	85.11	NA	NA	NA
SVM_HC [30]	82.5	82.1	65.09	89.07	0.779	0.761	0.748
MCP1	83.09	81.87	61.02	**95.4**	0.84	0.68	0.7
MCP2	85.41	89.76	72.92	88.81	0.87	0.73	0.72
MCP5	**87.3**	**94.2**	**82.5**	84.65	**0.89**	**0.77**	**0.75**

Rule 1 (Section 5), each variation of MCP employs related rules.

- *Support Vector Regression, Non-Dominated Sorting Genetic Algorithm II* (SVR-NSGAII): This model [63] takes sequence profiles as input. It employs the SVR for classification and NSGAII to map the real values to integers and optimize the kernel parameters.
- *Support Vector Machine, Parallel Hierarchical Grid Search (SVM-PHGS)*: This model [64] takes the sequence profiles as training input. It uses SVM and PHGS to optimize the kernel parameters.
- *Ensemble Method*(EM) [6]: is a method that combines multiple individual learners of multi-layer perceptron (MLP), RBF neural network, and four SVM classifiers. The method applies numerous combination rules to generate the final prediction output. EM also investigates the effect of two input data types including Position-Specific Scoring Matrix (PSSM) [45] and the amino-acid coding scheme.
- *Support Vector Machine, Hybrid Coding for Protein (SVM-HC)* [30]: employs the geometrical similarities and where missing, the SVM module leverages the final protein structure. SVM-HC extracts a 6-bit code from the physiochemical properties of both amino-acids and the tendency factors.
- *Group Template Pattern Classifiers (GTPCs)*: GTPCs [37] primarily, divides the training set into groups of proteins with similar length and then trains a model for each group to capture longer range interactions of amino-acids. The base learner of this model is SVM. At the time of classification, each input goes to the model trained with proteins of similar length.
- *Deep Inception-Inside-Inception Networks (MUFold-SS)*: MUFold-SS [15] employs base inception modules to build a deep network. Each block of the network consists of an inception-inside-inception unit. These units are the same base inception modules containing another inception module in each layer (inception-inside-inception).
- *Data Partition and Semi-Random Subspace Method (PSRSM)*: PSRSM [38] initially divides the training data into subsets of similar length. Consequently, on each subset, random sub-spaces are generated by the semi-random subspace method. Finally, an ensemble of SVMs is trained and combined using majority voting.
- *Two Dimensional Deep Convolutional Neural Network (2DCNN)*: 2DCNN [36] propose a two-dimensional deep convolutional neural network including 6 convolutional and 5 max-pooling layers. Their network can better incorporate the sequence order information as well as the amino-acid interactions.

We use RS126, CASP10, CASP11, and CASP12 to compare our approach with other rivals proposing various types of methods. As Tables 9 and 10 demonstrate, our solution outperforms other rivals (Section 2) including ensemble machines, neural networks, SVMs, and decision trees. The contributing factors to this enhancement are as follows: *attempting to interpret the protein language rather than using numerical sequence representation*, *enabling each base learner (SVM and KNN) to process protein language accurately by incorporating proper kernels and measures* ($\tilde{d}$), the *designation of an effective multi-component architecture*, *utilizing fuzzy decisions* and *employing flexible and effectual aggregation pool*. Table 11 compares the effectiveness of our $\tilde{d}$-FKNN component with three distance-based classifiers. Previously we demonstrated the accuracy of our fuzzy KNN when gradually added each dissimilarity coefficients to it. Table 11 confirms the impact of each dissimilarity coefficient in the prediction enhancement of KNN and fuzzy KNN. It also indicates that taking the string protein sequences for the input (our $\tilde{d}$-FKNN component) is more beneficial compared to consuming of the numerical encoding schemes as utilized in the distance-based classifiers of [18].

Table 9
The accuracy of our approach versus other competitors on RS126.

Method's Name	Q_3	Q_H	Q_E	Q_C
Ensemble NN (Imbalanced Training Set) [1]	73.33	68.07	52.78	73.02
Ensemble NN (Over-Sampling) [1]	73.75	71.72	68.9	77.44
Ensemble NN (Under-Sampling) [1]	73.02	69.76	77.48	73.2
Ensemble NN (Under and Over-Sampling) [1]	73.73	71.05	69.38	77.38
Ensemble NN (Tree-based) [1]	73.51	75.78	67.85	74.84
NN with SMV voting scheme [1]	74.66	74.85	72.78	75.32
NN with GWMV voting scheme [1]	74.9	72.61	70.25	78.56
NN with WMV voting scheme [1]	74.64	72.43	69.76	78.43
PLM-PBC-HPP [57]	69	67.1	62.7	73.4
ELM-HPP01 [57]	67.9	62.5	61.9	71.6
Single-Stage One-against-all [46]	69.7	54.1	79.3	59
Single-Stage One-against-one [46]	67.6	54.5	79.8	58.3
Single-Stage DAG [46]	67.5	54.2	80	58.3
Single-Stage Crammer and Singer [46]	70.2	55.8	78	56.5
Two-Stage One-against-all [46]	66.5	61.2	78.5	63.9
Two-Stage One-against-one [46]	66.5	57.5	81.2	65.4
Two-Stage DAG [46]	66.8	57.4	80.9	65.5
Single-Stage Vapnik and Weston [46]	70.4	55.7	78.2	57.1
Two-Stage Vapnik and Weston [46]	66.1	57.8	81.9	67
Multi-SVM Ensemble [46]	74.98	75.37	66.43	79.26
Two-Stage Crammer and Singer [46]	66.8	57.9	81	66.8
DT [20]	NA	70.4	78.4	67.1
SVM-DT [20]	NA	72.8	79.6	69.3
MCP1	83.09	81.87	61.02	**95.38**
MCP2	85.41	89.76	72.92	88.81
MCP5	**87.29**	**94.22**	**82.54**	84.65

Table 10
The comparison of the MCC metric on RS126.

Method's Name	MCC_H	MCC_E	MCC_C
Ensemble NN (Imbalanced Training Set) [1]	0.65	0.54	0.51
Ensemble NN (Over-Sampling) [1]	0.66	0.57	0.51
Ensemble NN (Under-Sampling) [1]	0.64	0.56	0.49
Ensemble NN (Under-Sampling and Over-Sampling) [1]	0.65	0.56	0.51
Ensemble NN (Tree-based) [1]	0.64	0.55	0.52
NN with SMV voting scheme [1]	0.67	0.58	0.53
NN with GWMV voting scheme [1]	0.67	0.58	0.53
NN with WMV voting scheme [1]	0.67	0.58	0.53
MCP1	0.84	0.68	0.7
MCP2	0.87	0.73	0.72
MCP5	**0.89**	**0.77**	**0.75**

Table 11
The Q_3 metric for the distance-based methods on RS126.

Method's Name	Q_3
KNN [18]	49.85
Fuzzy KNN [18]	53.08
Minimum Distance [18]	59.22
FKNN + LZ scores (our model component)	76.38
FKNN + LZ + ρ_d (our model component)	80.42
FKNN + LZ + ρ_d + n-gram (our model component)	**81.96**

Table 12
The comparison of the Q_3 metric using the CASP datasets.

Method's Name	CASP10	CASP11	CASP12
GTPCs [37]	83.07	81.98	82.35
MUFold-SS [15]	85.03	83.4	80.8
PSRSM [38]	85.51	85.89	85.55
2DCNN [36]	83.56	81.16	80.30
MCP1	87.01	86.61	86.22
MCP2	87.92	87.86	87.36
MCP5	**89.04**	**88.5**	**87.95**

While EM_EXPOP(PSSM) has the least performance across all classes, our MCP5 model gains the highest values for all the measures in *E* and *H* classes. Unlike MCP5, all other competitors gain a very low accuracy in class *E*.

Table 12 compares the effectiveness of the versions of our MCP framework against four recent approaches. Across all three datasets, all versions of MCP can beat the rivals. Even the single component of Edit-SVM displays better accuracy compared to all the rivals from Table 12. Although PSRSM [38] employs a greater number of ensemble components (SVMs), it cannot beat our two-component framework. The reason might rely on our effective aggregation pool and processing the original sequences of proteins using two effective measures for string (language) interpretation (d and edit kernel). Our approach performs better than the deep neural network-based models of 2DCNN [36] and MUFold-SS [15]. In addition to the higher accuracy of our MCP method, its training is not as costly as deep networks. Furthermore, our model is highly interpretable and traceable which means the procedure of learning is transparent and comprehensible. However, the learning procedure in deep networks is considered as a black box with little comprehensibility. Third, the optimization nature of SVM can avoid local-minima which is not guaranteed in neural networks.

To investigate the impact of both protein language interpretation and the compound structure of our framework on the improvement of the prediction disjointedly, consider two pairs of comparisons. First, consider the PSRSM method [36] from Table 12 and the Edit-SVM component of our framework from Table 7. On CASP10 dataset, a single component of our MCP model (Edit-SVM) beats the whole ensemble framework of PSRSM which employs several numbers of SVMs using PSSM data. On CASP11 and CASP12, the accuracy of PSRSM is only a neglectable amount (below 0.1%) higher.

To discuss the influence of our multi-component architecture on the enhancement of the prediction, compare the accuracy of PSRSM against MCP1, MCP2 and MCP3. On all three datasets, all versions of our framework beat PSRSM.

6.3. Conclusion

In this paper, we propose a unified multi-component framework to effectively predict the protein secondary structure. We divide the prediction task into four subtasks: Pre-processing, classification (Edit-SVM and $\tilde{d}$-FKNN), filtration, and aggregation. The fuzzy component uses LZ score, n-gram score, and the dissimilarity rate and fuses them into a unified dissimilarity metric. The biologically filtered output will then be fed into the aggregation pool to better infer the prediction results. To eliminate data loss we disregard Feature extraction task. Moreover, our proposed model can exploit latent natural language to better explain the hidden sequence-structure relationships. The experimental results demonstrate the strength of the string dissimilarity measures and admit the competence of our proposed method against other rivals.

Our multi-component framework is easily modifiable. One can add more classifiers and expand the ensemble size with different base learners. Fuzzification of Edit-SVM module can also enhance the aggregation process. A weighted dissimilarity measure in Fuzzy KNN can better recognize dissimilarity components and result in higher accuracy. We leave such modifications as future work. From another perspective, practical web-servers have significantly contributed to the advancement of the algorithms in medical and chemistry domains. As another future task, we will similarly provide a user-friendly web-server to interactively reflect our proposed prediction model.

References

[1] M. Alirezaee, A. Dehzangi, E. Mansoori, Ensemble of neural networks to solve class imbalance problem of protein secondary structure prediction, International Journal of Artificial Intelligence & Applications 3 (6) (2012) 9.
[2] E. Aygun, B.J. Oommen, Z. Cataltepe, Peptide classification using optimal and information theoretic syntactic modeling, Pattern Recogn. 43 (11) (2010) 3891–3899.
[3] S. Babaei, A. Geranmayeh, S.A. Seyyedsalehi, Protein secondary structure prediction using modular reciprocal bidirectional recurrent neural networks, Comput. Methods Progr. Biomed. 100 (3) (2010) 237–247.
[4] S. Babaei, A. Geranmayeh, S.A. Seyyedsalehi, Towards designing modular recurrent neural networks in learning protein secondary structures, Expert Syst. Appl. 39 (6) (2012) 6263–6274.
[5] R. Bondugula, O. Duzlevski, D. Xu, Profiles and fuzzy k-nearest neighbor algorithm for protein secondary structure prediction, Proceedings of the 3rd Asia-Pacific Bioinformatics Conference, World Scientific, 2005, pp. 85–94.
[6] H. Bouziane, B. Messabih, A. Chouarfia, Effect of simple ensemble methods on protein secondary structure prediction, Soft Computing 19 (6) (2015) 1663–1678.
[7] C. Bystroff, A. Krogh, Hidden markov models for prediction of protein features, Protein Structure Prediction, Springer, 2008, pp. 173–198.
[8] C. Chen, Y. Tian, X. Zou, P. Cai, J. Mo, Prediction of protein secondary structure content using support vector machine, Talanta 71 (5) (2007) 2069–2073.
[9] W. Chen, H. Lin, K.-C. Chou, Pseudo nucleotide composition or PseKNC: an effective formulation for analyzing genomic sequences, Mol. Biosyst. 11 (10) (2015) 2620–2634.
[10] Y. Chen, Long sequence feature extraction based on deep learning neural network for protein secondary structure prediction, Information Technology and Mechatronics Engineering Conference (ITOEC), IEEE, 2017, pp. 843–847 2017 IEEE 3rd.
[11] Z. Chen, P. Zhao, F. Li, e. a. Leier, iFeature: A python package and web server for features extraction and selection from protein and peptide sequences, Bioinformatics 34 (14) (2018) 2499–2502.
[12] K.-C. Chou, Prediction of protein cellular attributes using pseudo-amino acid composition, Proteins: Structure, Function, and Bioinformatics 43 (3) (2001) 246–255.
[13] P.Y. Chou, G.D. Fasman, Prediction of protein conformation, Biochemistry 13 (2) (1974) 222–245.
[14] P.M. Dinubhai, H.B. Shah, Protein secondary structure prediction using neural network: a comparative study, Int J Enhanc Res Manag Comput Appl 3 (4) (2014) 18–23.
[15] C. Fang, Y. Shang, D. Xu, S.S. MUFOLD, New deep inception-inside-inception networks for protein secondary structure prediction, Proteins: Structure, Function, and Bioinformatics 86 (5) (2018) 592–598.
[16] J. Garnier, J.-F. Gibrat, B. Robson, Gor method for predicting protein secondary structure from amino acid sequence, Methods in Enzymology, vol. 266, Elsevier, 1996, pp. 540–553.
[17] J. Garnier, D.J. Osguthorpe, B. Robson, Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins, J. Mol. Biol. 120 (1) (1978) 97–120.
[18] A. Ghosh, B. Parai, Protein secondary structure prediction using distance based classifiers, Int. J. Approx. Reason. 47 (1) (2008) 37–44.
[19] J. Han, J. Pei, M. Kamber, Data Mining: Concepts and Techniques, Elsevier, 2011.
[20] J. He, H.-J. Hu, R. Harrison, P.C. Tai, Y. Pan, Rule generation for protein secondary structure prediction with support vector machines and decision tree, IEEE Trans. NanoBioscience 5 (1) (2006) 46–53.
[21] S. Hosseini, H. Yin, M. Zhang, Y. Elovici, and X. Zhou. Mining Subgraphs from Propagation Networks through Temporal Dynamic Analysis.
[22] S. Hosseini, H. Yin, M. Zhang, X. Zhou, S. Sadiq, Jointly modeling heterogeneous temporal properties in location recommendation, International Conference on Database Systems for Advanced Applications, Springer, 2017, pp. 490–506.
[23] S. Hosseini, H. Yin, X. Zhou, S. Sadiq, M.R. Kangavari, N.-M. Cheung, Leveraging Multi-Aspect Time-Related Influence in Location Recommendation, World Wide Web, 2017, pp. 1–28.
[24] W. Hua, D.T. Huynh, S. Hosseini, J. Lu, X. Zhou, Information extraction from microblogs: a survey, Int. J. Software and Informatics 6 (4) (2012) 495–522.
[25] A.K. Johal, R. Singh, Protein secondary structure prediction using improved support vector machine and neural networks, International Journal of Engineering and Computer Science 3 (1) (2014) 3593–3597.
[26] J.M. Keller, M.R. Gray, J.A. Givens, A fuzzy k-nearest neighbor algorithm, IEEE transactions on systems, man, and cybernetics (4) (1985) 580–585.
[27] E. Krissinel, On the relationship between sequence and structure similarities in proteomics, Bioinformatics 23 (6) (2007) 717–723.
[28] S.Y. Lee, J.Y. Lee, K.S. Jung, K.H. Ryu, A 9-state hidden markov model using protein secondary structure information for protein fold recognition, Comput. Biol. Med. 39 (6) (2009) 527–534.
[29] A. Lempel, J. Ziv, On the complexity of finite sequences, IEEE Trans. Inf. Theory 22 (1) (1976) 75–81.

[30] Z. Li, J. Wang, S. Zhang, Q. Zhang, W. Wu, A new hybrid coding for protein secondary structure prediction based on primary structure similarity, Gene 618 (2017) 8–13.

[31] H.-N. Lin, T.-Y. Sung, S.-Y. Ho, W.-L. Hsu, Improving protein secondary structure prediction based on short subsequences with local structure similarity, Bmc Genomics, vol. 11, 2010, p. S4 BioMed Central.

[32] L. Lin, S. Yang, R. Zuo, Protein secondary structure prediction based on multi-svm ensemble, Intelligent Control and Information Processing (ICICIP), 2010 International Conference on, IEEE, 2010, pp. 356–358.

[33] B. Liu, F. Liu, X. Wang, J. Chen, L. Fang, K.-C. Chou, Pse-in-One: a web server for generating various modes of pseudo components of DNA, RNA, and protein sequences, Nucleic Acids Res. 43 (W1) (2015) W65–W71.

[34] B. Liu, H. Wu, K.-C. Chou, Pse-in-one 2.0: an improved package of web servers for generating various modes of pseudo components of dna, rna, and protein sequences, Nat. Sci. 9 (04) (2017) 67.

[35] T. Liu, X. Zheng, J. Wang, Prediction of protein structural class using a complexity-based distance measure, Amino Acids 38 (3) (2010) 721–728.

[36] Y. Liu, J. Cheng, Y. Ma, Y. Chen, Protein secondary structure prediction based on two dimensional deep convolutional neural networks, 2017 3rd IEEE International Conference on Computer and Communications (ICCC), IEEE, 2017, pp. 1995–1999.

[37] Y. Liu, Y. Ma, J. Cheng, A novel group template pattern classifiers (GTPCs) method in protein secondary structure prediction, 2017 3rd IEEE International Conference on Computer and Communications (ICCC), IEEE, 2017, pp. 2713–2717.

[38] Y. Ma, Y. Liu, J. Cheng, Protein secondary structure prediction based on data partition and semi-random subspace method, Sci. Rep. 8 (2018).

[39] F. Masulli, S. Mitra, Natural computing methods in bioinformatics: a survey, Inf. Fusion 10 (3) (2009) 211–216.

[40] N. Mossos, D.F. Mejia-Carmona, I. Tischer, Fs-tree, Sequential association rules and first applications to protein secondary structure analysis, Advances in Computational Biology, Springer, 2014, pp. 189–198.

[41] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, A. Tramontano, Critical assessment of methods of protein structure prediction (CASP) : round x, Proteins: Structure, Function, and Bioinformatics 82 (2014) 1–6.

[42] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, A. Tramontano, Critical assessment of methods of protein structure prediction: progress and new directions in round xi, Proteins: Structure, Function, and Bioinformatics 84 (2016) 4–14.

[43] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, A. Tramontano, Critical assessment of methods of protein structure prediction (CASP): round xii, Proteins: Structure, Function, and Bioinformatics 86 (2018) 7–15.

[44] R. Muhammod, S. Ahmed, D. Md Farid, S. Shatabda, A. Sharma, A. Dehzangi PyFeat, A python-based effective feature generation tool for DNA, RNA, and protein sequences, Bioinformatics (2019).

[45] P.C. Ng, S. Henikoff, Predicting deleterious amino acid substitutions, Genome Res. 11 (5) (2001) 863–874.

[46] M.N. Nguyen, J.C. Rajapakse, Multi-class support vector machines for protein secondary structure prediction, Genome Informatics 14 (2003) 218–227.

[47] K. Paliwal, J. Lyons, R. Heffernan, A short review of deep learning neural networks in protein structure prediction problems, Advanced Techniques in Biology & Medicine (2015) 1–2.

[48] M.S. Patel, H.S. Mazumdar, Knowledge base and neural network approach for protein secondary structure prediction, J. Theor. Biol. 361 (2014) 182–189.

[49] G. Pollastri, A.J.M. Martin, C. Mooney, A. Vullo, Accurate prediction of protein secondary structure and solvent accessibility by consensus combiners of sequence and structure information, BMC Bioinf. 8 (1) (2007) 201.

[50] B. Rost, C. Sander, Prediction of protein secondary structure at better than 70accuracy, J. Mol. Biol. 232 (2) (1993) 584–599.

[51] M. Spencer, J. Eickholt, J. Cheng, A deep learning network approach to ab initio protein secondary structure prediction, IEEE ACM Trans. Comput. Biol. Bioinform 12 (1) (2015) 103–112.

[52] P.-N. Tan, Introduction to Data Mining, Pearson Education India, 2006.

[53] Y.T. Tan, B.A. Rosdi, Fpga-based Hardware Accelerator for the Prediction of Protein Secondary Class via Fuzzy K-Nearest Neighbors with Lempel-Ziv Complexity Based Distance Measure, (2015).

[54] S. Theodoridis, K. Koutroumbas, Pattern Recognition, second ed., Elsevier, San Diego, 2003.

[55] B.A. van den Berg, M.J.T. Reinders, J.A. Roubos, D. de Ridder SPiCE, A web-based tool for sequence-based protein classification and exploration, BMC Bioinf. 15 (1) (2014) 93.

[56] M.P.J. Van der Loo, The stringdist package for approximate string matching, The R Journal 6 (1) (2014) 111–122.

[57] G. Wang, Y. Zhao, D. Wang, A protein secondary structure prediction framework based on the extreme learning machine, Neurocomputing 72 (1–3) (2008) 262–268.

[58] S. Wang, J. Peng, J. Ma, J. Xu, Protein secondary structure prediction using deep convolutional neural fields, Sci. Rep. 6 (2016) 18962.

[59] J.J. Ward, L.J. McGuffin, B.F. Buxton, D.T. Jones, Secondary structure prediction with support vector machines, Bioinformatics 19 (13) (2003) 1650–1655.

[60] X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, S.Y. Philip, Top 10 algorithms in data mining, Knowl. Inf. Syst. 14 (1) (2008) 1–37.

[61] A. Yaseen, Y. Li, Context-based features enhance protein secondary structure prediction accuracy, J. Chem. Inf. Model. 54 (3) (2014) 992–1002.

[62] M. Zamani, S.C. Kremer, A multi-stage protein secondary structure prediction system using machine learning and information theory, 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, 2015, pp. 1304–1309.

[63] M.H. Zangooei, S. Jalili, Protein secondary structure prediction using dwkf based on svr-nsgaii, Neurocomputing 94 (2012) 87–101.

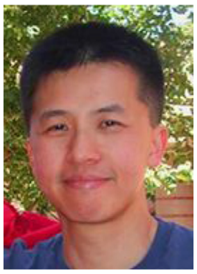
[64] M.H. Zangooei, S. Jalili, Pssp with dynamic weighted kernel fusion based on svm-phgs, Knowl. Based Syst. 27 (2012) 424–442.

Leila Khalatbari is currently a Ph.D. candidate in Artificial Intelligence at Sharif University of Technology, Tehran, Iran. Since 2014, she has been a researcher in the computational cognitive models laboratory at Iran University of Science and Technology. She received her B.S. degree in computer software engineering from Qazvin. She recently completed her M.S. degree in Artificial Intelligence. Her research interests include machine learning, data analysis, data mining, and bioinformatics.

Mohammad Reza Kangavari received B.S. degree in mathematics and computer science from the Sharif University of Technology in 1982, the M.S. degree in computer science from Salford University in 1989, and the Ph.D. degree in computer science from the University of Manchester in 1994. He is currently a lecturer in the Computer Engineering Department, Iran University of Science and Technology, Tehran, Iran. His research interests include Intelligent Systems, Human-Computer-Interaction, Cognitive Computing, Data Science, Machine Learning, and Wireless Sensor Networks.

Saeid Hosseini completed the Ph.D. degree in Computer Science at the University of Queensland, Australia. He obtained his M.Sc. from the Queensland University Of Technology in 2012. He received the Australian Postgraduate Award in 2015. He is currently a post-doc researcher. His research interests mainly focus on diffusion networks, graph mining, crowdsourcing, recommendation systems, spatiotemporal databases, and social network analytics. Dr. Hosseini is an invited reviewer in reputed proceedings including ICDM and TKDE. He has also been a program committee member in CSS (2017) and DASFAA (2017 and 2018).

Hongzhi Yin works as a Lecturer and an ARC DECRA Fellow with The University of Queensland, Australia. He received his doctoral degree from Peking University in July 2014 under the supervision of Prof. Bin Cui. His research interests include user behavior modeling, user profiling, recommender system, especially spatial-temporal recommendation, user linkage across social networks, network embedding, knowledge graph mining, and construction, topic discovery and event detection, deep learning. He has been serving as conference organizers, conference PC member for PVLDB, SIGIR, ICDE, IJCAI, ICDM, CIKM, DASFAA, ASONAM, MDM, WISE, PAKDD and reviewer of more than 10 reputed journals such as VLDB Journal, TKDE, TOIS, TKDD, and etc.



Ngai-Man Cheung received the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CA, in 2008. He is currently an Assistant Professor with the Singapore University of Technology and Design (SUTD). From 2009 to 2011, he was a postdoctoral researcher with the Image, Video and Multimedia Systems group at Stanford University, Stanford, CA. He has also held research positions with Texas Instruments Research Center Japan, Nokia Research Center, IBM T. J. Watson Research Center, HP Labs Japan, Hong Kong University of Science and Technology (HKUST), and Mitsubishi Electric Research Labs (MERL). His work has resulted in 11 U.S. patents granted with several pending. His research interests include signal, image, and video processing, computer vision and machine learning.