Supplementary Material 1

David Guevara-Barrientos, Rakesh Kaundal

1 Amino Acid Composition

1.1 Amino Acid Composition (AAC)

The amino acid composition method [1] calculates the frequency of each natural amino acid in the sequence.

$$AAC(t) = \frac{N(t)}{N}, \qquad t \in A$$

Where A is the group of the 20 natural amino acids, $N(t)$ is the number of times amino acid t appears in the sequence, and N is the sequence length.

Vector length 20

Parameters None

1.2 Dipeptide Composition (DPC)

The dipeptide composition method [1] calculates the frequency of each consecutive amino acid pair in the sequence.

$$DPC(t,u) = \frac{N(t,u)}{N-1}, \qquad t,u \in A$$

Where A is the group of the 20 natural amino acids, $N(t,u)$ is the number of times amino acid pair t,u appears in the sequence, and N is the sequence length.

Vector length 400

Parameters None

1.3 Tripeptide Composition (TPC)

The tripeptide composition method [1] calculates the frequency of each consecutive amino acid triplet in the sequence.

$$TPC(t,u,v) = \frac{N(t,u,v)}{N-2}, \qquad t,u,v \in A$$

Where A is the group of the 20 natural amino acids, $N(t,u,v)$ is the number of times amino acid triplet t,u,v appears in the sequence, and N is the sequence length.

Vector length 400

Parameters None

1.4 Composition of k-Spaced Amino Acid Pairs (CKSAAP)

The composition of k-spaced amino acid pairs method [2] calculates the frequency of amino acid pairs separated by k characters.

$$CKSAAP(t,u,k) = \frac{N(t,u)}{N-1}, \qquad t,u \in A, k \text{ between 0 and K}$$

Where A is the group of the 20 natural amino acids, $N(t,u)$ is the number of times amino acid pair t,u, separated by k characters, appears in the sequence, N is the sequence length and K is the maximum number of k. Two consecutive characters are separated by $k = 0$. If $K = 5$, then each possible pair would be calculated for $k = 0, 1, 2, 3, 4$ and 5.

Vector length $(K+1) * 400$

Parameters

- `--gap` K, default 5

1.5 Dipeptide Deviation from Expected Mean (DDE)

The dipeptide deviation from expected mean method [3] calculates the dipeptide composition (D), theoretical mean (M) and theoretical variance (V) and applies the following formulas:

$$
\begin{aligned}
D(t,u) &= \frac{N(r,s)}{N-1} \qquad r,s \in A \\
M(t,u) &= \frac{N(r)}{N} * \frac{N(s)}{N} \\
V(t,u) &= \frac{M(r,s)(1-M(r,s))}{N-1} \\
DDE(t,u) &= \frac{D(r,s)-M(r,s)}{\sqrt{V(r,s)}}
\end{aligned}
$$

Where A is the group of the 20 natural amino acids, $N(t,u)$ is the number of times the amino acid pair t,u appears in the sequence, $N(r)$ and $N(s)$ are the number of times the amino acid r or s appears in the sequence, and N is the sequence length.

Vector length 400

Parameters None

1.6 Amino Acid Pair Antigenicity Scale (AAPAS)

In the amino acid pair antigenicity scale method [4], for each existing amino acid pair, it counts the number of times they appear consecutively in the sequence and multiplies them by their normalized amino acid pair antigenicity scale, which can be understood as the chance each amino acid pair is associated with an epitope.

$$AAP(t,u) = N(t,u) * R(t,u), \qquad t,u \in A$$

Where A is the group of the 20 natural amino acids, $N(t,u)$ is the number of times the amino acid pair t,u appears in the sequence and $R(t,u)$ is the normalized amino acid pair antigenicity scale for the pair t,u.

The values of $R(t,u)$ are in Supplementary Material 2. They were calculated as follows:

$$R_{AAP} = 2\left(\frac{log\left(\frac{f(t,u)^+}{f(t,u)^-}\right) - min}{max - min}\right) - 1, \qquad t,u \in A$$

Where $f(t,u)^+$ and $f(t,u)^-$ are the frequencies of the amino acid pair t,u in the epitopes (obtained from the Bcipep database) [5] and non-epitopes (obtained from the Swiss-Prot database) [6], respectively.

Vector length 400

Parameters None

1.7 Composition Moment Vector (CMV)

The composition moment vector method [7] contains information of the position of each occurence for each amino acid in the sequence in its calculation.

$$CMV(t) = \frac{1}{N(N-1)}\sum_{i=1}^{N} i \text{ if } n_i = t, \text{ 0 if not}, \qquad t \in A$$

Where A is the group of the 20 natural amino acids, n_i is the ith residue in the sequence and N is the sequence length.

Vector length 20

Parameters None

1.8 Enhanced Amino Acid Composition (EAAC)

The enhanced amino acid composition method [8] calculates the frequency of each natural amino acid in a sliding window across the whole sequence.

$$EAAC(t,w_i) = \frac{N(t,w_i)}{W}, \qquad t \in A, \text{for all } i \text{ between 1 and } N - W$$

Where A is the group of the 20 natural amino acids, $N(t, w_i)$ is the number of times amino acid t appears in the sliding window w_i and W is the size of the sliding window. For example, the first sliding window w_1 would go from the first amino acid n_1 to the amino acid n_s, while the second sliding window would go from the second amino acid n_2 to the amino acid n_{s+1}.

Vector length $(N - W + 1) * 20$

Parameters

- `--window` W, default 5

All sequences must have the same length

2 Grouped Amino Acid Composition

2.1 Grouped Amino Acid Composition (GAAC)

The grouped amino acid composition method [8] finds the proportion of each of the five group of proteins in the sequence. These five groups are based on their physicochemical properties, which are aliphatic (AGILMV), aromatic (FWY), positive (HKR), negative (DE) and uncharged (CNPQST) [9].

$$GAAC(g) = \frac{N(g)}{N}, \qquad g \in G$$

Where G are the 5 groups based on the amino acids' physicochemical properties, $N(g)$ is the number of times an amino acid belonging to the group g appears and N is the sequence length.

Vector length 5

Parameters None

2.2 Enhanced Grouped Amino Acid Composition (EGAAC)

The enhanced grouped amino acid composition method [8] finds the proportion of each of the five group of proteins in a sliding window across the peptide sequence. These five groups are based on their physicochemical properties,

which are aliphatic (AGILMV), aromatic (FWY), positive (HKR), negative (DE) and uncharged (CNPQST) [9].

$$EGAAC(g, w_i) = \frac{N(g, w_i)}{W}, \qquad g \in G, \text{for all } i \text{ between 1 and } N - W$$

Where G are the 5 groups based on the amino acids' physicochemical properties, $N(g, w_i)$ is the number of times an amino acid belonging to the group g appears in the sliding window w_i and W is the window size.

Vector length $(N - W + 1) * 5$

Parameters

- `--window` W, default 5

All sequences must have the same length

2.3 Composition of k-Spaced Amino Acid Group Pairs (CKSAAGP)

The composition of k-spaced amino acid group pairs method [8] calculates the frequency of amino acid pairs, grouped by their physicochemical properties as in GAAC, separated by k characters.

$$CKSAAGP(g, h, k) = \frac{N(g, h)}{N - 1}, \qquad g, h \in G, k \text{ between 0 and K}$$

Where G are the 5 groups based on the amino acids' physicochemical properties, $N(g, h)$ is the number of times amino acids from the groups g, h, separated by k characters, are paired in the sequence, N is the sequence length and K is the maximum number of k. Two consecutive characters are separated by $k = 0$. If $K = 5$, then each possible pair would be calculated for $k = 0, 1, 2, 3, 4$ and 5.

Vector length $(K + 1) * 25$

Parameters

- `--gap` K, default 5

2.4 Grouped Dipeptide Composition (GDPC)

The grouped dipeptide composition method [8] calculates the frequency of each consecutive amino acid group pair in the sequence.

$$GDPC(g,h) = \frac{N(g,h)}{N-1}, \qquad g,h \in A$$

Where G are the 5 groups based on the amino acids' physicochemical properties, $N(g,h)$ is the number of times amino acids from the groups g,h appear consecutively in the sequence, and N is the sequence length.

Vector length 25

Parameters None

2.5 Grouped Tripeptide Composition (GTPC)

The grouped tripeptide composition method [8] calculates the frequency of each consecutive amino acid group triplet in the sequence.

$$GTPC(g,h,i) = \frac{N(g,h,i)}{N-1}, \qquad g,h,i \in A$$

Where G are the 5 groups based on the amino acids' physicochemical properties, $N(g,h,i)$ is the number of times amino acids from the groups g,h,i appear consecutively in the sequence, and N is the sequence length.

Vector length 125

Parameters None

2.6 Encoding Based on Grouped Weight (EBGW)

For the encoding based on grouped weight method [10], the amino acids are split in 4 groups, based on their hydrophobicity and charge:

Neutral and non-polarity $C1 = A, F, G, I, L, M, P, W, V$

Neutral and polarity $C2 = C, N, S, T, Q, Y$

Acidic $C3 = D, E$

Basic $C4 = H, K, R$

Then, they are combined into the following groups:

- $G1 = C1 + C2$
- $G2 = C1 + C3$
- $G3 = C1 + C4$

And then each amino acid would have an associated value for each group in the following way:

$$G1(n_i) = \begin{cases} 1 & \text{if } n_i \in C1 + C2 \\ 0 & \text{if } n_i \notin C1 + C2 \end{cases}$$

$$G2(n_i) = \begin{cases} 1 & \text{if } n_i \in C1 + C3 \\ 0 & \text{if } n_i \notin C1 + C3 \end{cases}$$

$$G3(n_i) = \begin{cases} 1 & \text{if } n_i \in C1 + C4 \\ 0 & \text{if } n_i \notin C1 + C4 \end{cases}$$

So, if an amino acid belongs to, for example, group $C2$, then it would pre-encode as 1 for the first group $G1$, and 0 for the other ones. If it belongs to group $C1$, it would pre-encode as 1 for every group. This results in three binary sequences H_j, one per group, with N length, being N the sequence length, and j is a number between 1 and 3.

$$H_j(n) = G_j(n_0), G_j(n_1), \ldots, G_j(n_N)$$

The full table associating amino acids with its group value can be found in Supplementary Material 2.

The normalized weight $w(n)$ of a characteristic sequence $H_j(n)$ is the frequency of 1 appearing in it.

$$w(n) = \frac{G_j(n_0) + G_j(n_1) + \ldots + G_j(n_N)}{N}$$

Given a number K, the characteristic sequence $H_j(n)$ can be split into K subsequences. This way, $H_j\lfloor nk/L \rfloor$ represents a subsequence, where $1 \leq k \leq K$, and $\lfloor * \rfloor$ is the largest integer below the result of the division inside. Joining all k values would yield the whole characteristic sequence. Hence, $w_j\lfloor nk/L \rfloor$ is the normalized weight of the subsequence $H_j\lfloor nk/L \rfloor$. This results in the following weight characteristic sequence:

$$W_j = w_j\lfloor n/L \rfloor, w_j\lfloor n2/L \rfloor, \ldots, w_j\lfloor nL/L \rfloor$$

Finally, all three vectors (one per j) are concatenated.

$$EBGW = W_1 + W_2 + W_3$$

Vector length $3K$

Parameters

- `--k` K, default 30

3 Quasi-Sequence-Order

Both quasi-sequence-order and sequence-order-coupling number encodings [11] use the Grantham [12] and the Schneider-Wrede [13] distance matrices.

The l-th rank sequence-order-coupling number is a sum of squares of the distance (according to the distance matrices) between two amino acids that are separated by g characters in the sequence.

$$SOC_l = \sum_{i=1}^{N-l} (l_{i,i+l})^2, \qquad 1 \leq l \leq L$$

Where $l_{i,i+l}$ is the value in a distance matrix between two amino acids at positions i and $i-l$, L is the maximum value of the lag value l, and N is the sequence length.

3.1 Sequence-Order-Coupling Number (SOCN)

This encoding is the concatenation of all SOC_l per distance matrix.

Vector length $2L$

Parameters

- `--lag` L, default 30

3.2 Quasi-Sequence-Order (QSO)

First, the quasi-sequence-order numbers for the amino acids must be calculated as follows:

$$QSO_t = \frac{f_r}{1 + w \sum_{l=1}^{L} SOC_l}, \qquad t \in A$$

Where A is the group of the 20 natural amino acids, f_r is the frequency of each amino acid in the sequence (just as in AAC encoding), and w is a weight factor.

Then, the quasi-sequence-order numbers for the lag values must be calculated as follows:

$$QSO_l = \frac{w * SOC_l - 20}{1 + w \sum_{m=1}^{m} SOC_m}, \qquad 1 \leq l \leq L$$

Where A is the group of the 20 natural amino acids, f_r is the frequency of each amino acid in the sequence (just as in AAC encoding), l is a the lag value, and w is a weight factor.

Vector length $2L + 40$

Parameters

- `--lag` L, default 30
- `--weight` w, default 0.1

4 Autocorrelation

The autocorrelation descriptors use the amino acid properties from the AAindex Database [14], found in the `data/AAidx.txt` file. The default indices used (CIDH920105, BHAR880101, CHAM820101, CHAM820102, CHOC760101, BIGC670101, CHAM810101, DAYM780201) were taken from the work by Xiao et al. [15]. All the values in the indices are centralized and standardized for the autocorrelation encodings as follows:

$$P_t = \frac{P_t - \bar{P}}{\sigma}, \qquad t \in A$$

Where A is the group of the 20 natural amino acids, P_t is the value of the property for the amino acid t, and $\bar{P}$ and σ are the average and standard deviation of all the 20 amino acids in the index, respectively.

$$\bar{P} = \frac{\sum_{i=1}^{20} P_i}{20}$$

$$\sigma = \sqrt{\frac{1}{20}\sum_{i=1}^{20}(P_i - \bar{P})^2}$$

4.1 Geary Autocorrelation (Geary)

The Geary autocorrelation [16] is calculated as:

$$Geary(l) = \frac{\frac{1}{2(N-l)}\sum_{i=1}^{N-l}(P_i - P_{i+l})^2}{\frac{1}{N-1}\sum_{i=1}^{N}(P_i - \bar{P}')^2}, \qquad 1 \leq l \leq< L$$

Where l is the lag value, L is the maximum lag value, P_i and P_{i+l} are the centralized and standardized values for the amino acids at positions i and $i + l$, and $\bar{P}'$ is the average property value between all amino acids in the sequence.

$$\bar{P}' = \frac{\sum_{n=1}^{N} P_i}{N}$$

Vector length $L * X$, where X is the count of used indices.

Parameters

- `--lag` L, default 30
- `--indices` *indices*, default 'CIDH920105, BHAR880101, CHAM820101, CHAM820102, CHOC760101, BIGC670101, CHAM810101, DAYM780201' ($X = 8$).

4.2 Moran Autocorrelation (Moran)

The Moran autocorrelation [17] is calculated as:

$$Moran(l) = \frac{\frac{1}{N-l}\sum_{i=1}^{N-l}(P_i - \bar{P}')(P_{i+l} - \bar{P}')}{\frac{1}{N-1}\sum_{i=1}^{N}(P_i - \bar{P}')^2}, \qquad 1 \leq l \leq < L$$

Where l is the lag value, L is the maximum lag value, P_i and P_{i+l} are the centralized and standardized values for the amino acids at positions i and $i + l$, and $\bar{P}'$ is the average property value between all amino acids in the sequence.

$$\bar{P}' = \frac{\sum_{n=1}^{N} P_i}{N}$$

Vector length $L * X$, where X is the count of used indices.

Parameters

- `--lag` L, default 30
- `--indices` *indices*, default 'CIDH920105, BHAR880101, CHAM820101, CHAM820102, CHOC760101, BIGC670101, CHAM810101, DAYM780201' ($X = 8$).

4.3 Normalized Moreau-Broto Autocorrelation (NMB)

The Normalized Moreau-Broto autocorrelation [18] is calculated as:

$$NMB(l) = \frac{\sum_{i=1}^{N-l} P_i * P_{i+l}}{N-l}, \qquad 1 \leq l \leq < L$$

Where l is the lag value, L is the maximum lag value, and P_i and P_{i+l} are the centralized and standardized values for the amino acids at positions i and $i+l$.

Vector length $L * X$, where X is the count of used indices.

Parameters

- `--lag` L, default 30
- `--indices` *indices*, default 'CIDH920105, BHAR880101, CHAM820101, CHAM820102, CHOC760101, BIGC670101, CHAM810101, DAYM780201' ($X = 8$).

5 Composition/Transition/Distribution

The Composition/Transition/Distribution encodings [19, 20] are based on a categorical division of the 20 natural amino acids according to their structural and physicochemical properties. 13 properties were chosen on iFeature [8], and 1 (surface tension) was added [21], as listed in Supplementary Material 2.

5.1 Composition (CTDC)

Calculates the frequency of each division per property.

$$C(d) = \frac{N(d)}{N}$$

Where $N(d)$ is the number of amino acids in the division d found in the sequence, and N is the sequence length.

Vector length 42

Parameters None

5.2 Transition (CTDT)

Calculates the frequency of each transition (division 1 to division 2, division 1 to division 3, etc.) per property between consecutive amino acids.

$$T(d, e) = \frac{N(d, e) + N(e, d)}{N - 1}$$

Where $N(d, e)$ and $N(e, d)$ are the numbers of consecutive amino acids from divisions d and e in both orders (de and ed), and N is the sequence length.

Vector length 42

Parameters None

5.3 Distribution (CTDD)

Calculates where the first, 25%, 50%, 75% and 100% of amino acids in a division occur in a sequence. It is done by highlighting all the amino acids that belong to a certain division in a sequence. Find the position of the first occurence and divide it by N (the sequence length). Then, find the position where the first 25% (rounded down) of the amino acids in that division occurs in the sequence, and divide this position over N. After that, do the same with the other percentages (Figure 1).

Vector length 210

Parameters None

6 Conjoint Triad

For the conjoint triad encodings, the amino acids were classified in 7 classes based on the dipoles and volumes of the side chains [22]: $\{A, G, V\}$, $\{I, L, F, P\}$, $\{Y, M, T, S\}$, $\{H, N, Q, W\}$, $\{R, K\}$, $\{D, E\}$, and $\{C\}$.

Original sequence (N=20)	L	A	T	H	I	F	Q	W	C	R	M	V	K	N	E	Y	P	D	G	S
Sequence by groups of polarity	A	B	B	C	A	A	C	A	A	C	A	A	C	C	C	A	B	C	B	B
Position in sequence	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Position in group A (N=8)	1				2	3		4	5		6	7				8				
Percentage	0%				25%			50%			75%					100%				
Distribution value	**0.05**				**0.25**			**0.40**			**0.55**					**0.80**				
Position in group B (N=5)		1	2														3		4	5
Percentage		0%, 25%	50%														75%			100%
Distribution value		**0.1, 0.1**	**0.15**														**0.85**			**1**
Position in group C (N=7)				1			2			3			4	5	6			7		
Percentage				0%, 25%						50%				75%				100%		
Distribution value				**0.2, 0.2**						**0.5**				**0.7**				**0.9**		

Figure 1: The example sequence has 20 amino acids, where 8 of them belong to polarity group A, 5 to group B and 7 to group C. For group C, the first occurence is at position 4, so the distribution value is 0.2 (4/20). The amino acid at the 25% mark is also position 4 because it is the first ($\lfloor 7 * 0.25 \rfloor = 1$) amino acid of group C, so the distribution value is 0.2. The amino acid at the 50% mark is at position 10 because it is the third ($\lfloor 7 * 0.5 \rfloor = 3$) amino acid of group C, so the distribution value is 0.5 (10/20). The amino acid at the 75% mark is at position 14 because it is the fifth ($\lfloor 7 * 0.75 \rfloor = 5$) amino acid of group C, so the distribution value is 0.7 (14/20). Finally, last amino acid of group C is at position 18, so the distribution value is 0.9 (18/20).

6.1 Conjoint Triad (CT)

The conjoint triad method [22] is calculated as:

$$CT_{i,j,k} = \frac{n_{i,j,k} - min\{n_{1,1,1}, n_{1,1,2}, \ldots, n_{7,7,7}\}}{max\{n_{1,1,1}, n_{1,1,2}, \ldots, n_{7,7,7}\}}$$

Where $n_{i,j,k}$ is the number of times three consecutive amino acids belonging to groups i, j and k are seen in the sequence.

Vector length 343

Parameters None

6.2 k-Spaced Conjoint Triad (KSCT)

The k-Spaced conjoint triad method [8] is based on the conjoint triad method, but instead of only evaluating consecutive amino acids, it evaluates triads separated by 0 to K characters. The original CT method is the same as KSCT with $K = 0$.

$$KSCT_{h,i,j} = \frac{n_{h,i,j} - min\{n_{1,1,1}, n_{1,1,2}, \ldots, n_{7,7,7}\}}{max\{n_{1,1,1}, n_{1,1,2}, \ldots, n_{7,7,7}\}}$$

Where $n_{h,i,j}$ is the number of times three consecutive amino acids belonging to groups h, i and j are seen in the sequence. This should be evaluated for $0 \leq k \leq K$, so it is calculated $k+1$ times. For example, for $k = 0$, each triad is formed by the amino acids at positions i, $i+1$ and $i+2$ for $1 \leq i \leq N-2$. For $k = 1$, each triad is formed by the amino acids at positions i, $i+2$, $i+4$ for $1 \leq i \leq N-4$. For $k = 2$, each triad is formed by the amino acids at positions i, $i+3$, $i+6$ for $1 \leq i \leq N-6$.

Vector length $343K$

Parameters

- `--k` K, default 1.

7 Pseudo-Amino Acid Composition

The pseudo-amino acid composition encodings use the hydrophobicity values proposed by Tanford [23], the hydrophilicity values proposed by Hopp and Woods [24] and the side chain mass values are the standard ones. Their initial values are represented by $H_1^0(t)$, $H_2^0(t)$ and $M^0(t)$, where t is each of the 20 natural amino acids. These values are centralized and standardized as follows:

$$P(t) = \frac{P^0(t) - \frac{1}{20}\sum_{i=1}^{20} P^0(i)}{\sqrt{\frac{\sum_{i=1}^{20}[P^0(i) - \frac{1}{20}\sum_{j=1}^{20} P^0(i)]^2}{20}}}, \qquad t \in A$$

Where A is the group of the 20 natural amino acids, and $P(t)$ represents the centralized and standardized value of any of the three properties (H_1, H_2, M) of the amino acid t, so in the end we would have $H_1(t)$, $H_2(t)$ and $M(t)$.

7.1 Pseudo-Amino Acid Composition (PAAC)

For the pseudo-amino acid composition method [11], a correlation function is calculated as:

$$\rho(t,u) = \frac{1}{3}\{[H_1(t) - H_1(u)]^2 + [H_2(t) - H_2(u)]^2[M(t) - M(u)]^2\}, \qquad t,u \in A$$

Where A is the group of the 20 natural amino acids. Then, the sequence-order-correlated factors are computed as follows:

$$f_l = \frac{1}{N-l} \sum_{j=1}^{N-l} \rho(t_j, t_{j+l}), \qquad 1 \leq l \leq L, t \in A$$

Where L is the maximum lag value. Now, the first 20 features (one per amino acid in A) are computed.

$$PAAC_t = \frac{N(t)}{1 + w \sum_{i=1}^{L} f_i}$$

Where $N(t)$ is the number of times the amino acid appears in the sequence, and w is a weighting factor set by default as 0.05, as suggested by Chou et al. [11]. Finally, the last set of features are added to the vector.

$$PAAC_l = \frac{w f_l}{1 + w \sum_{j=1}^{L} f_j}, \qquad 1 \leq l \leq L$$

Vector length $20 + L$

Parameters

- `--lag` L, default 30
- `--weight` w, default 0.05

7.2 Amphiphilic Pseudo-Amino Acid Composition (APAAC)

The amphiphilic pseudo-amino acid composition method [25] only uses the hydrophilicity (H_1) and hydrophobicity (H_2) values. These values are used to define their correlation functions as:

$$H_1(t, u) = H_1(t) H_1(u), \qquad t, u \in A$$
$$H_2(t, u) = H_2(t) H_2(u), \qquad t, u \in A$$

Where A is the group of the 20 natural amino acids. Now, the sequence-order can be found with the following formula:

$$f_l = \frac{1}{N-l} \sum_{j=1}^{N-l} H_1(i, i+l), \qquad 1 \leq l \leq 2L$$

Where L is the maximum lag value. Now, the first 20 features (one per amino acid in A) are computed.

$$APAAC_t = \frac{N(t)}{1 + w \sum_{i=1}^{2} Lf_i}$$

Where $N(t)$ is the number of times the amino acid appears in the sequence, and w is a weighting factor set by default as 0.05, as suggested by Chou et al. [11]. Finally, the last set of features are added to the vector.

$$APAAC_l = \frac{wf_l}{1 + w \sum_{j=1}^{2} Lf_j}, \qquad 1 \leq l \leq L$$

Vector length $20 + 2L$

Parameters

- `--lag` L, default 30
- `--weight` w, default 0.05

8 Binary

8.1 Binary

The binary encoding [26] represents each amino acid in the sequence as a binary string of 20 numbers. For example, amino acid A is "10000000000000000000", C is "01000000000000000000", etc., following the order of "ACDEFGHIKLMNPQRSTVWY".

Vector length $20N$, where N is the sequence length

Parameters None

All sequences must have the same length

8.2 Taylor's Venn Diagram (TVD)

The Taylor's venn diagram method [27] is based on 10 physicochemical groups (hydrophobic, positive, negative, polar, charged, small, tiny, aliphatic, aromatic, proline) where the 20 natural amino acids might belong to. These amino acids are encoded as binary vectors of length 10 (1 per property), getting a 1 if the amino acid belonging to the group that has that property. For example, if the amino acid belongs to the hydrophobic group, it will get 1, and if not, it will get 0.

$$TVD_p(t) = \begin{cases} 1 & \text{if } t \in p \\ 0 & \text{if } t \notin p \end{cases}, \qquad t \in A$$

Where p is a property and A is the set of the 20 natural amino acids. The full table with the binary values is found in Supplementary Material 2.

Vector length N, where N is the sequence length

Parameters None

All sequences must have the same length

9 Pseudo k-Tuple Reduced Amino Acid Composition (PseKRAAC)

The pseudo k-tuple reduced amino acid composition [28] represents proteins as vectors that contain information based on K-tuples of reduced amino acid cluster (RAAC^K) components. These components can depend on a *g-gap* or a *λ-correlation*, a type of reduced amino acid alphabet and a number of clusters (or mode). These types and modes, as well as the groups, are found in Supplementary Material 2.

For the *g-gap* type of calculation, it represents the sequence-order information of subsequences of length K separated by g residues. Thus, it counts the number of times a combination of groups in the selected RAAC appears (Figure 2).

For the *λ-correlation* type of calculation, it represents the sequence-order information of groups of amino acids separated by λ residues between amino

Figure 2: In this example, for type 1, mode 3, $K = 2$, there are 9 possible combinations between the three groups in mode 3 ($mode^K$, so $3^2 = 9$). The sequence "GIALPMN" has each amino acid mapped to groups 2, 1, 2, 1, 2, 1, 3, respectively. If the g value is set to 0, it evaluates pairs without interleaving residues in between, so the combination (2, 1) appears 3 times, (1, 2) appears 2 times and (1, 3) appears once, while the other 6 possible groups have 0 occurences. These are the values in the resulting vector. If the g value is set to 1, it evaluates pairs interleaving one **residue** between pairs, so the found combinations are (2, 1), (2, 1) and (2, 1), which makes it 3 occurences for (2, 1) and 0 for the other combinations.

Figure 3: In this example, for type 1, mode 3, $K = 2$, there are 9 possible combinations between the three groups in mode 3 ($mode^K$, so $3^2 = 9$). The sequence "GIALPMN", has each amino acid mapped to groups 2, 1, 2, 1, 2, 1, 3, respectively. If the λ value is set to 1, it evaluates consecutive pairs, so the combination (2, 1) appears 3 times, (1, 2) appears 2 times and (1, 3) appears once, while the other 6 possible groups have 0 occurences. These are the values in the resulting vector. If the λ value is set to 2, it evaluates pairs interleaving one residue between **amino acids**, so the found combinations are (2, 2), (1, 1), (2, 2), (1, 1) and (2, 3), which makes it 2 occurences for (2, 2), 2 occurences for (1, 1), 1 occurence for (2, 3) and 0 for the other combinations.

acids. Thus, it counts the number of times a combination of groups in the selected RAAC appears (Figure 3).

Vector length $mode^K$

Parameters

- `--type` *type*, required
- `--raactype` *mode*, required.
- `--subtype` *subtype*, required. `g-gap` or `lambda-correlation`
- `--ktuple` K, default 2. Can be 1, 2 or 3.
- `--gapLambda` *gapLambda*, required. Value for g or λ, depending on the subtype.

10 Secondary Structure with PSIPRED or SPINE-X

These encodings use the generated `.ss2` files from PSIPRED [29] or the `.spxOut` files from SPINE-X [30]. There must be one file per input sequence.

10.1 Secondary Structure Elements Binary (SSEB)

The secondary structure elements binary method [8] represents each amino acid, depending on the type of secondary structure element where they were classified in, as a vector of 3 binary digits. The elements are helix (001), sheet (010) and coil (100).

Vector length $3N$, where N is the sequence length

Parameters

- `--path`, path where `.ss2` and `.spXout` files are located. One per input sequence.

All sequences must have the same length

10.2 Secondary Structure Elements Content (SSEC)

The secondary structure elements content method [8] calculates the frequency of each element type (helix, sheet, coil) found in the peptide sequence.

$$SSEC(e) = \frac{N(e)}{N}, \qquad e \in Helix, Sheet, Coil$$

Where $N(s)$ is the number of times the element e appears in the sequence, and N is the sequence length

Vector length 3

Parameters

- `--path`, path where `.ss2` and `.spXout` files are located. One per input sequence.

10.3 Secondary Structure Probabilities Bigram (SSPB)

Each amino acid in the sequence gets a probability of it having one of the three structural elements (helix, coil, sheet). The secondary structure probabilities bigram [31] sums the multiplication of the probabilities for each of the combinations between structural elements among the pairs of amino acids separated by n residues. This parameter n was added by us, originally it was 1.

$$SSPB(e, f) = \frac{1}{N} \sum_{i=1}^{N-n} P_i(e) * P_{i+n}(f), \qquad e, f \in \{helix, coil, sheet\}$$

Where $P_i(e)$ and $P_{i+n}(f)$ are the probabilities of the amino acids at positions i and $i+n$ in the sequence having the elements e and f, respectively, and N is the sequence length.

Vector length 9

Parameters

- `--path`, path where `.ss2` and `.spXout` files are located. One per input sequence.
- `--n` n, default 1

10.4 Secondary Structure Probabilities Auto-Covariance (SSPAC)

Each amino acid in the sequence gets a probability of it having one of the three structural elements (helix, coil, sheet). The secondary structure probabilities auto-covariance method [31] sums the multiplication of the probabilities for each structural element among the pairs of amino acids separated by n residues, where n ranges from 1 to N.

$$SSPAC(n, e) = \frac{1}{L} \sum_{i=1}^{L-n} P_i(e) * P_{i+n}(e), \qquad 1 \leq n \leq N, e \in helix, coil, sheet$$

Where $P_i(e)$ and $P_{i+n}(e)$ are the probabilities of the amino acids at positions i and $i + n$ in the sequence having the element e, N is the maximum value for the separation between residues, and L is the sequence length.

Vector length $3N$

Parameters

- `--path`, path where `.ss2` and `.spXout` files are located. One per input sequence.
- `--n` N, default 10

11 Secondary Structure with SPINE-X

These encodings use the generated `.spxOut` files from SPINE-X [30]. There must be one file per input sequence.

11.1 Torsional Angles (TA)

The torsion angles method [8] adds the *phi* and *psi* values per amino acid to the vector.

Vector length $2N$, where N is the sequence length.

Parameters

- `--path`, path where `.spXout` files are located. One per input sequence.

All sequences must have the same length

11.2 Torsional Angles Composition (TAC)

The torsional angles composition [31] converts the *phi* and *psi* values per amino acid from degrees to radians, calculates the sine and cosine of these two angles, divides these values by the length of the sequence, and adds the 4 final values to the vector.

$$TAC(f, a) = \frac{1}{N} \sum_{i=1}^{N} f(\frac{a_i \pi}{180}), \qquad f \in \{sin, cos\}, a \in \{phi, psi\}$$

Where a_i is the *phi* or *psi* value for the amino acid at position i in the sequence, and N is the sequence length.

Vector length 4

Parameters

- `--path`, path where `.spXout` files are located. One per input sequence.

11.3 Torsional Angles Bigram (TAB)

The torsional angles bigram [31] converts the *phi* and *psi* values per amino acid from degrees to radians, and calculates the sine and cosine of these two angles, so each amino acid has 4 associated values. Then, each type of value is multiplied as pairs in the sequence separated by n residues, and finally divided by the sequence length. This parameter n was added by us, originally it was 1.

$$TAB(f, g, a, n) = \frac{1}{N} \sum_{i=1}^{N-n} f(\frac{a_i \pi}{180}) * g(\frac{a_{i+n} \pi}{180}), \qquad f, g \in \{sin, cos\}, a \in \{phi, psi\}$$

Where a_i and a_{i+n} are the *phi* or *psi* values for the amino acid at position i and $i + n$ in the sequence, and N is the sequence length.

Vector length 10

Parameters

- `--path`, path where `.spXout` files are located. One per input sequence.
- `--n` n, default 1.

11.4 Torsional Angles Autocovariance (TAAC)

The torsional angles auto-covariance method [31] converts the *phi* and *psi* values per amino acid from degrees to radians, and calculates the sine and cosine of these two angles, so each amino acid has 4 associated values. Then, it sums the multiplication of each type of value among the pairs of amino acids separated by n residues, where n ranges from 1 to N.

$$TAAC(f, a, n) = \frac{1}{L}\sum_{i=1}^{L-n} f(\frac{a_i\pi}{180}) * f(\frac{a_{i+n}\pi}{180}), \qquad f \in \{sin, cos\}, a \in \{phi, psi\}, 1 \leq n \leq N$$

Where a_i and a_{i+n} are the *phi* or *psi* values for the amino acid at position i and $i + n$ in the sequence, N is the maximum value for the separation between residues, and L is the sequence length.

Vector length $4N$

Parameters

- `--path`, path where `.spXout` files are located. One per input sequence.
- `--n` N, default 10.

11.5 Accessible Surface Area (ASA)

The accessible surface area method [8] reads the ASA values per amino acid and adds them to the vector.

Vector length N, where N is the sequence length.

Parameters

- `--path`, path where `.spXout` files are located. One per input sequence.

All sequences must have the same length

12 Disorder

The disorder-based methods use the generated `.dis` files generated by VSL2 [32]. There must be one file per input sequence.

12.1 Disorder

The disorder method [33] reads the probability values per amino acid and adds them to the vector.

Vector length N, where N is the sequence length.

Parameters

- `--path`, path where `.dis` files are located. One per input sequence.

All sequences must have the same length

12.2 Disorder Content (DisorderC)

The disorder content method [8] calculates the frequency of ordered and disordered residues in the sequence.

$$DisorderC(d) = \frac{N(d)}{N}, \qquad d \in order, disorder$$

Where $N(d)$ is the number of ordered or disordered residues in the sequence, and N is the sequence length.

Vector length 2

Parameters

- `--path`, path where `.dis` files are located. One per input sequence.

12.3 Disorder Binary (DisorderB)

The disorder binary method [8] encodes each amino acid as a binary vector of length 2. If the residue is ordered, then it is encoded as [1, 0], and if it is disordered, it is encoded as [0, 1].

Vector length $2N$, where N is the sequence length.

Parameters

- `--path`, path where `.dis` files are located. One per input sequence.

All sequences must have the same length

13 k-Nearest Neighbors

The k-nearest neighbors (KNN) methods require two additional files: a training file in FASTA format that will contain a training set, and a label file, which will contain the class each sequence corresponds to. The KNN method uses the similarity score between every two sequences in the training file as distance.

The K values depend on the total number of samples provided in the training file, finding the amount of sequences in $1\%, 2\%, 3\%, \ldots, K\%$ of the training file. If the training file has 10 sequences, then from 1% to 10% the value will be 1, from 11% to 20% the value will be 2, and so on.

13.1 k-Nearest Neighbors - Peptides (KNNpeptide)

The k-nearest neighbor for peptides method [34] indicates how many of the sequences per each class in the neighboring $K\%$ from the training file are close to the input sequence according to the similarity score $s(a, b)$, which is calculated as:

$$d(t,u) = \begin{cases} BLOSUM62(t,u) & \text{if } BLOSUM62(t,u) > 0 \\ 0 & \text{if } BLOSUM62(t,u) \leq 0 \end{cases}, \qquad t, u \in A$$

$$s(a,b) = \sum_{i=1}^{N} d(a_i, b_i)$$

Where A is the set of the 20 natural amino acids, $BLOSUM62(t,u)$ is the value for the amino acid pair (t,u) in the BLOSUM62 matrix, and a_i and b_i are the amino acids at position i in the sequences a and b.

Vector length KC, where C is number of classes.

Parameters

- `--train`, path where the fasta training file is located
- `--labels`, path where the label file is located. All sequences in the training file must be in the labels file.
- `--k` K, default 30.

All sequences must have the same length

13.2 k-Nearest Neighbors - Proteins (KNNproteins)

The k-nearest neighbor for proteins [8] indicates how many of the sequences per each class in the neighboring k% from the training file are close to the input sequence, according to the similarity score $s(a,b)$, which is calculated as:

$$s(a,b) = \frac{2 * NW(a,b)}{T + N}$$

Where $NW(a,b)$ is the number of equal characters in the resulting Needleman-Wunsch alignment [35] between sequences a and b, T is the number of training sequences, and N is the sequence length.

Vector length KC, where C is number of classes.

Parameters

- `--train`, path where the fasta training file is located
- `--labels`, path where the label file is located. All sequences in the training file must be in the labels file.
- `--k` K, default 30.

All sequences must have the same length

14 Position-Specific Scoring Matrix (PSSM)

The position-specific scoring matrix-based methods use the generated `.pssm` by `blastpgp` in legacy BLAST [36] and `psiblast` in BLAST+ [37] against the uniref50 database [38].

14.1 Position-Specific Scoring Matrix (PSSM)

The PSSM method [33] inserts all 20 values per sequence amino acid in the vector.

Vector length $20N$, where N is the sequence length.

Parameters

- `--path`, path where the `.pssm` files are located. One per input sequence.

All sequences must have the same length

14.2 PSSM Amino Acid Composition (PSSMAAC)

The PSSM amino acid composition method [39] calculates the average score for each of the 20 natural amino acids along the whole sequence.

$$PSSMAAC(t) = \frac{1}{N} \sum_{i=1}^{N} s_{i,t}, \qquad t \in A$$

Where A is the set of the 20 natural amino acids, $s_{i,t}$ is the score in the PSSM matrix for the amino acid t at position i in the sequence, and N is the sequence length.

Vector length 20

Parameters

- `--path`, path where the `.pssm` files are located. One per input sequence.

14.3 Bigram PSSM (BiPSSM)

The bigram PSSM method [31] sums the product between the PSSM values of two residues in the sequence separated by n characters for two amino acid types and divides that sum by the sequence length. This parameter n was added by us, originally it was 1.

$$BiPSSM(t,u) = \frac{1}{N}\sum_{i=1}^{N-n} s_{i,t} * s_{i+n,u}, \qquad t,u \in A$$

Where A is the set of the 20 natural amino acids, $s_{i,t}$ and $s_{i+n,u}$ are the scores in the PSSM matrix for the amino acids t and u at positions i and $i+n$ respectively, and N is the sequence length.

Vector length 400

Parameters

- `--path`, path where the `.pssm` files are located. One per input sequence.
- `--n` n, default 1

14.4 PSSM Autocovariance (PSSMAC)

The PSSM autocovariance method [40] calculates the autocovariance between two residues separated by n characters for a specific amino acid type.

$$\bar{s}_t = \frac{1}{N} sum_{i=1}^{N} s_{i,t}, \qquad t \in A$$

$$PSSMAC(t,n) = \sum_{i=1}^{N-n} \frac{(s_{i,t} - \bar{s}_t) * (s_{i+n,t} - \bar{s}_t)}{N-n}, \qquad t \in A$$

Where A is the set of the 20 natural amino acids, $s_{i,t}$ and $s_{i+n,t}$ are the scores in the PSSM matrix for the amino acid t at positions i and $i+n$, and N is the sequence length.

Vector length 400

Parameters

- `--path`, path where the `.pssm` files are located. One per input sequence.
- `--n` n, default 1

14.5 Pseudo-PSSM (PPSSM)

The pseudo-PSSM method [41] finds the average for every amino acid type in the PSSM matrix, and calculates the correlation between residues separated by n characters per each amino acid type. First, all values in the PSSM matrix must be standardized by using the following formula:

$$s_{i,t} = \frac{s^0_{i,t} - \frac{1}{20}\sum_{j=1}^{20} s^0_{i,j}}{\sqrt{\frac{1}{20}\sum_{k=1}^{20}(s^0_{i,k} - \frac{1}{20}\sum_{j=1}^{20} s^0_{i,j})^2}}, \qquad t \in A$$

Where A is the set of the 20 natural amino acids, $s^0_{i,t}$ is the initial score in the PSSM matrix for the amino acid t at the row i, and $s^0_{i,j}$ and $s^0_{i,k}$ are the initial scores in the PSSM matrix for the row i, columns j and k.

$$\bar{s}_t = \frac{1}{N}\sum_{i=1}^{N} s_{i,t}, \qquad t \in A$$

$$\rho_t(n) = \frac{1}{N-n}\sum_{i=1}^{N-n}(s_{i,t} - s_{i+n,t})^2, \qquad t \in A$$

Where $s_{i,t}$ and $s_{i+n,t}$ are the standardized scores in the PSSM matrix for the amino acid t at rows i and $i+n$, and N is the sequence length.

The PPSSM vector is the concatenation of the 20 values for $\bar{s}_t$ and the 20 values of $\rho_t(n)$.

Vector length 40

Parameters

- `--path`, path where the `.pssm` files are located. One per input sequence.
- `--n` n, default 1

15 Other Encodings

15.1 Amino Acid Index (AAI)

The amino acid index method [42] uses the amino acid properties from the AAindex Database [14]. This database has 544 different indices, where 531 have no "NA" values for any of the 20 amino acids. The features are the values for each amino acid in the sequence found in each one of the indices.

Vector length $531N$, where N is the sequence length.

Parameters None

All sequences must have the same length

15.2 BLOSUM62

The BLOSUM62 method [43] uses the BLOSUM62 matrix to get the features, which are all the values for the 20 amino acids in each respective row. This means that for every amino acid in the sequence there will be 20 features.

Vector length $20N$, where N is the sequence length.

Parameters None

All sequences must have the same length

15.3 Z-Scale (ZS)

The z-srancale method [44] uses the z-scale table [45], where each amino acid type has 5 z-values. This means that for every amino acid in the sequence there will be 5 features.

Vector length 205, where N is the sequence length.

Parameters None

All sequences must have the same length

References

1. Manoj Bhasin and Gajendra P.S. Raghava. Classification of Nuclear Receptors Based on Amino Acid Composition and Dipeptide Composition. *Journal of Biological Chemistry*, 279(22):23262–23266, may 2004.

2. Ke Chen, Lukasz A. Kurgan, and Jishou Ruan. Prediction of protein structural class using novel evolutionary collocation-based sequence representation. *Journal of Computational Chemistry*, 29(10):1596–1604, jul 2008.

3. Vijayakumar Saravanan and Namasivayam Gautham. Harnessing Computational Biology for Exact Linear B-Cell Epitope Prediction: A Novel Amino Acid Composition-Based Feature Descriptor. *OMICS: A Journal of Integrative Biology*, 19(10):648–658, oct 2015.

4. J. Chen, H. Liu, J. Yang, and K.-C. Chou. Prediction of linear B-cell epitopes using amino acid pair antigenicity scale. *Amino Acids*, 33(3):423–428, sep 2007.

5. Sudipto Saha, Manoj Bhasin, and Gajendra PS Raghava. Bcipep: A database of B-cell epitopes. *BMC Genomics*, 6(1):79, dec 2005.

6. A. Bairoch. The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Research*, 28(1):45–48, jan 2000.

7. Jishou Ruan, Kui Wang, Jie Yang, Lukasz A. Kurgan, and Krzysztof Cios. Highly accurate and consistent method for prediction of helix and strand content from primary protein sequences. *Artificial Intelligence in Medicine*, 35(1-2):19–35, sep 2005.

8. Zhen Chen, Pei Zhao, Fuyi Li, André Leier, Tatiana T Marquez-Lago, Yanan Wang, Geoffrey I Webb, A Ian Smith, Roger J Daly, Kuo-Chen Chou, and Jiangning Song. iFeature: a Python package and web server for features extraction and selection from protein and peptide sequences. *Bioinformatics*, 34(14):2499–2502, jul 2018.

9. Tzong-Yi Lee, Zong-Qing Lin, Sheng-Jen Hsieh, Neil Arvin Bretaña, and Cheng-Tsung Lu. Exploiting maximal dependence decomposition to identify conserved motifs from a group of aligned signal sequences. *Bioinformatics*, 27(13):1780–1787, jul 2011.

10. Zhen-Hui Zhang, Zheng-Hua Wang, Zhen-Rong Zhang, and Yong-Xian Wang. A novel method for apoptosis protein subcellular localization prediction combining encoding based on grouped weight and support vector machine. *FEBS Letters*, 580(26):6169–6174, nov 2006.

11. K C Chou. Prediction of protein cellular attributes using pseudo-amino acid composition. *Proteins*, 43(3):246–55, may 2001.

12. R. Grantham. Amino Acid Difference Formula to Help Explain Protein Evolution. *Science*, 185(4154):862–864, sep 1974.

13. G. Schneider and P. Wrede. The rational design of amino acid sequences by artificial neural networks and simulated molecular evolution: de novo design of an idealized leader peptidase cleavage site. *Biophysical Journal*, 66(2):335–344, feb 1994.

14. Shuichi Kawashima, Piotr Pokarowski, Maria Pokarowska, Andrzej Kolinski, Toshiaki Katayama, and Minoru Kanehisa. AAindex: amino acid index database, progress report 2008. *Nucleic acids research*, 36(Database issue):D202–5, jan 2008.

15. Nan Xiao, Dong-Sheng Cao, Min-Feng Zhu, and Qing-Song Xu. protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences. *Bioinformatics (Oxford, England)*, 31(11):1857–9, jun 2015.

16. Robert R. Sokal and Barbara A. Thomson. Population structure inferred by local spatial autocorrelation: An example from an Amerindian tribal population. *American Journal of Physical Anthropology*, 129(1):121–131, jan 2006.

17. Z P Feng and C T Zhang. Prediction of membrane protein types based on the hydrophobic index of amino acids. *Journal of protein chemistry*, 19(4):269–75, may 2000.

18. David S. Horne. Prediction of protein helix content from an autocorrelation analysis of sequence hydrophobicities. *Biopolymers*, 27(3):451–477, mar 1988.

19. I. Dubchak, I. Muchnik, S. R. Holbrook, and S. H. Kim. Prediction of protein folding class using global description of amino acid sequence.

Proceedings of the National Academy of Sciences, 92(19):8700–8704, sep 1995.

20. Inna Dubchak, Ilya Muchnik, Christopher Mayor, Igor Dralyuk, and Sung-Hou Kim. Recognition of a protein fold in the context of the SCOP classification. *Proteins: Structure, Function, and Genetics*, 35(4):401–407, jun 1999.

21. C.Z. Cai. SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Research*, 31(13):3692–3697, jul 2003.

22. Juwen Shen, Jian Zhang, Xiaomin Luo, Weiliang Zhu, Kunqian Yu, Kaixian Chen, Yixue Li, and Hualiang Jiang. Predicting protein-protein interactions based only on sequences information. *Proceedings of the National Academy of Sciences of the United States of America*, 104(11):4337–41, mar 2007.

23. Charles. Tanford. Contribution of Hydrophobic Interactions to the Stability of the Globular Conformation of Proteins. *Journal of the American Chemical Society*, 84(22):4240–4247, nov 1962.

24. T. P. Hopp and K. R. Woods. Prediction of protein antigenic determinants from amino acid sequences. *Proceedings of the National Academy of Sciences*, 78(6):3824–3828, jun 1981.

25. K.-C. Chou. Using amphiphilic pseudo amino acid composition to predict enzyme subfamily classes. *Bioinformatics*, 21(1):10–19, jan 2005.

26. Zhen Chen, Yong-Zi Chen, Xiao-Feng Wang, Chuan Wang, Ren-Xiang Yan, and Ziding Zhang. Prediction of Ubiquitination Sites by Using the Composition of k-Spaced Amino Acid Pairs. *PLoS ONE*, 6(7):e22930, jul 2011.

27. William Ramsay Taylor. The classification of amino acid conservation. *Journal of Theoretical Biology*, 119(2):205–218, mar 1986.

28. Yongchun Zuo, Yuan Li, Yingli Chen, Guangpeng Li, Zhenhe Yan, and Lei Yang. PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition. *Bioinformatics*, 33(1):122–124, jan 2017.

29. David T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology*, 292(2):195–202, 1999.

30. Eshel Faraggi, Tuo Zhang, Yuedong Yang, Lukasz Kurgan, and Yaoqi Zhou. SPINE X: Improving protein secondary structure prediction by multistep learning coupled with prediction of solvent accessible surface area and backbone torsion angles. *Journal of Computational Chemistry*, 33(3):259–267, jan 2012.

31. Farshid Rayhan, Sajid Ahmed, Swakkhar Shatabda, Dewan Md Farid, Zaynab Mousavian, Abdollah Dehzangi, and M. Sohel Rahman. iDTI-ESBoost: Identification of Drug Target Interaction Using Evolutionary and Structural Features with Boosting. *Scientific Reports*, 7(1):17731, dec 2017.

32. Kang Peng, Predrag Radivojac, Slobodan Vucetic, A Keith Dunker, and Zoran Obradovic. Length-dependent prediction of protein intrinsic disorder. *BMC Bioinformatics*, 7(1):208, dec 2006.

33. Yudong Cai, Tao Huang, Lele Hu, Xiaohe Shi, Lu Xie, and Yixue Li. Prediction of lysine ubiquitination with mRMR feature selection and analysis. *Amino Acids*, 42(4):1387–1395, apr 2012.

34. Xiang Chen, Jian Ding Qiu, Shao Ping Shi, Sheng Bao Suo, Shu Yun Huang, and Ru Ping Liang. Incorporating key position and amino acid residue features to identify general and species-specific Ubiquitin conjugation sites. *Bioinformatics*, 29(13):1614–1622, 2013.

35. Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, mar 1970.

36. S. Altschul. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, sep 1997.

37. Christiam Camacho, George Coulouris, Vahram Avagyan, Ning Ma, Jason Papadopoulos, Kevin Bealer, and Thomas L Madden. BLAST+: architecture and applications. *BMC Bioinformatics*, 10(1):421, dec 2009.

38. B. E. Suzek, Y. Wang, H. Huang, P. B. McGarvey, and C. H. Wu. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, mar 2015.

39. Taigang Liu, Xiaoqi Zheng, and Jun Wang. Prediction of protein structural class for low-similarity sequences using support vector machine and PSI-BLAST profile. *Biochimie*, 92(10):1330–1334, oct 2010.

40. Qiwen Dong, Shuigeng Zhou, and Jihong Guan. A new taxonomy-based protein fold recognition approach based on autocross-covariance transformation. *Bioinformatics*, 25(20):2655–2662, oct 2009.

41. Hong Bin Shen and Kuo Chen Chou. Nuc-PLoc: A new web-server for predicting protein subnuclear localization by fusing PseAA composition and PsePSSM. *Protein Engineering, Design and Selection*, 20(11):561–567, 2007.

42. Chun-Wei Tung and Shinn-Ying Ho. Computational identification of ubiquitylation sites from protein sequences. *BMC Bioinformatics*, 9(1):310, dec 2008.

43. Tzong-Yi Lee, Shu-An Chen, Hsin-Yi Hung, and Yu-Yen Ou. Incorporating Distant Sequence Features and Radial Basis Function Networks to Identify Ubiquitin Conjugation Sites. *PLoS ONE*, 6(3):e17331, mar 2011.

44. Yong-Zi Chen, Zhen Chen, Yu-Ai Gong, and Guoguang Ying. SUMOhydro: A Novel Method for the Prediction of Sumoylation Sites Based on Hydrophobic Properties. *PLoS ONE*, 7(6):e39195, jun 2012.

45. Maria Sandberg, Lennart Eriksson, Jörgen Jonsson, Michael Sjöström, and Svante Wold. New Chemical Descriptors Relevant for the Design of Biologically Active Peptides. A Multivariate Characterization of 87 Amino Acids. *Journal of Medicinal Chemistry*, 41(14):2481–2491, jul 1998.