RaptorX-Property: a web server for protein structure property prediction

Sheng Wang, Wei Li, Shiwang Liu, and Jinbo Xu

Supplementary Material

This section describes (S1) DeepCNF model, (S2) training methods, (S3) protein features, and (S4) performance metric.

S1. DeepCNF model

As shown in Figure 1 in main text, DeepCNF has two modules: (i) the Conditional Random Fields (CRF) module consisting of the top layer and the label layer, and (ii) the deep convolutional neural network (DCNN) module covering the input to the top layer. When only one hidden layer is used, DeepCNF becomes Conditional Neural Fields (CNF), a probabilistic graphical model described in [1].

Given a protein sequence of length L, let $y = (y_1, \dots, y_L) \in \Sigma^L$ denote its sequence label where y_i is the label at residue i, and Σ is the set of all possible labels. For instance, for protein disorder prediction, $\Sigma = \{0,1\}$ where 0 stands for ordered and 1 for disordered. Let $X = (X_1, \dots, X_L)$ denote the input feature where X_i is a column vector representing the input feature for residue i. DeepCNF calculates the conditional probability of y on the input X with parameter θ as follows,

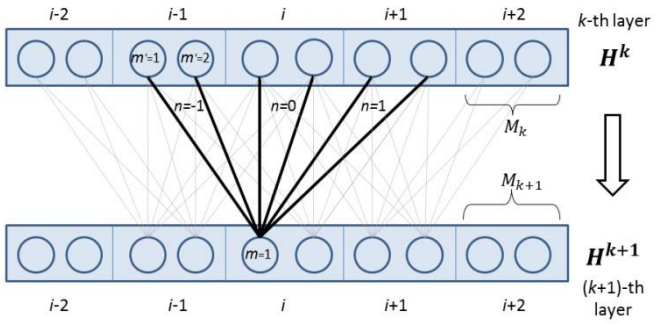
$$P_\theta(y|X) = \exp(\sum_{i=1}^{L}[f_\theta(y, X, i) + g_\theta(y, X, i)])\,/Z(X) \quad (1)$$

Where $f_\theta(y, X, i)$ is the binary potential function specifying correlation among adjacent labels at position i, $g_\theta(y, X, i)$ is the unary potential function modeling relationship between y_i and input features for position i, and $Z(X)$ is the partition function. Formally, $f_\theta()$ and $g_\theta()$ are defined as follows.

$$f_\theta(y, X, i) = \sum_{a,b} T_{a,b}\delta(y_{i-1} = a)\delta(y_i = b) \quad (2)$$

$$g_\theta(y, X, i) = \sum_{a,h} U_{a,h}H_{a,h}(X, i, W)\delta(y_i = a) \quad (3)$$

Where a and b represent two specific labels for prediction, $\delta()$ is an indicator function, $H_{a,h}(X, i, W)$ is a deep neural network function for the h-th neuron at position i of the top layer for label a, and W, U, and T are the model parameters to be trained. Specifically, W is the parameter for the neural network, U is the parameter connecting the top layer to the label layer, and T is for label correlation. The two potential functions can be merged into a single binary potential function $f_\theta(y, X, i) = \sum_{a,b,h} T_{a,b}^h H_{a,b}^h(X, i, W)\delta(y_{i-1} = a)\delta(y_i = b)$. Note that these deep neural network functions for different labels could be shared to $H_{a,h}(X, i, W)$. To control model complexity and avoid over-fitting, we add a L2-norm penalty term as the regularization factor.



SFigure 1. The feed-forward connection between two adjacent layers in the deep convolutional neural network.

SFigure 1 shows two adjacent layers of DCNN. Let M_k be the number of neurons for a single position at the k-th layer. Let $X_i(h)$ be the h-th feature at the input layer for residue i and $H_i^k(h)$ denote the output value of the h-th neuron of position i at layer k. When $k = 1$, H^k is actually the input feature X. Otherwise, H^k is a matrix of dimension $L \times M_k$. Let $2N_k + 1$ be the window size at the k-th layer. Mathematically, $H_i^k(h)$ is defined as follows.

$$H_i^k(h) = X_i(h), \quad \text{if } k = 1$$

$$H_i^{k+1}(h) = \pi\left(\sum_{n=-N_k}^{N_k} \sum_{h'=1}^{M_k} \left[H_{i+n}^k(h') * W_n^k(h,h')\right]\right), \quad \text{if } k < K$$

$$H_h(X,i,W) = H_i^k(h) \quad \text{if } k = K \qquad (4)$$

Meanwhile, $\pi()$ is the activation function, either the sigmoid (i.e., $1/(1+\exp(-x))$) or the tanh (i.e., $(1-\exp(-2x))/(1+\exp(-2x))$) function. W_n^k $(-N_k \le n \le N_k)$ is a 2D weight matrix for the connections between the neurons of position $i+n$ at layer k and the neurons of position i at layer $k+1$. $W_n^k(h,h')$ is shared by all the positions in the same layer, so it is position-independent. Here h' and h index two neurons at the k-th and $(k+1)$-th layers, respectively. See S2.4 about how to calculate the gradient of DCNN by back propagation.

S2. Training methods

Suppose we have T protein sequences for training, and each sequence t with length L_t. We study the behavior of three different objective functions, namely Log-likelihood, Labelwise accuracy, and the AUC function.

S2.1 Log-likelihood

The log-likelihood is a widely-used objective function for training CRF [2]. Mathematically, the log-likelihood is defined as follows:

$$\text{Log\_likelihood} = \sum_{t=1}^{T} \log P_\theta(y^t|X^t) \qquad (5)$$

Where $P_\theta(y|X)$ is defined in equation (1).

S2.2 Labelwise accuracy

Gross *et. al.* [3] proposed an objective function that could directly maximize the labelwise accuracy defined as

$$\text{Labelwise} = \sum_{t=1}^{T} \sum_{i=1}^{L_t} \delta\left(P_\theta(y_i^{(\tau)}) > max_{y_i \neq y_i^\tau} P_\theta(y_i)\right) \qquad (6)$$

Where $y_i^{(\tau)}$ denotes the real label at position i, $P_\theta(y_i^{(\tau)})$ is the predicted probability of the real label at position i being (τ). It could be represented by the marginal probability

$$P_\theta\left(y_i^{(\tau)}|X^t\right) = \frac{1}{Z(X)} \cdot \sum_{y_{1:L^t}} \left[\delta(y_i = (\tau)) \cdot exp\left(\mathbb{F}_{1:L^t}(X^t, y, \theta)\right)\right] \qquad (7)$$

where $\mathbb{F}_{l_1:l_2}(X,y,\theta) = \sum_{i=l_1}^{l_2} \mathbb{f}_\theta(y,X,i)$.

To obtain a smooth approximation to this objective function, [3] replaces the indicator function with a sigmoidal function $Q_\lambda(x) = 1/(1+\exp(-\lambda x))$ where the parameter λ is set to 15 by default. Then it becomes the following form:

$$\text{Labelwise} \approx \sum_{t=1}^{T} \sum_{i=1}^{L_t} Q_\lambda\left(P_\theta\left(y_i^{(\tau)}|X^t\right) - P_\theta(\tilde{y}_i^{(\tau)}|X^t)\right) \qquad (8)$$

Where $\tilde{y}_i^{(\tau)}$ denotes the label other than $y_i^{(\tau)}$ that has the maximum posterior probability at position i.

S2.3 The AUC function

Definition. The AUC of a predictor function P_θ on label τ is defined as:

$$AUC(P_\theta, \tau) = \mathbb{P}\left(P_\theta(y_i^\tau) > P_\theta(y_j^\tau) \,\middle|\, i \in D^\tau, j \in D^{!\tau}\right) \quad (9)$$

where $\mathbb{P}()$ is the probability over all pairs of positive and negative examples, D^τ is a set of positive examples with true label τ, and $D^{!\tau}$ is a set of negative examples with true label not being τ. Note that the union of D^τ and $D^{!\tau}$ contains all the training sequence positions, i.e., $D^\tau = \bigcup_{t=1}^{T} \bigcup_{i=1}^{L_t} \delta_{i,t}^\tau$ where $\delta_{i,t}^\tau$ is an indicator function. If the true label of the i-th position from sequence t equals to τ, then $\delta_{i,t}^\tau$ is equal to 1; otherwise 0. Again, $P_\theta(y_i^\tau)$ could be represented by the marginal probability $P_\theta(y_i^\tau | X^t)$ from the training sequence t.

Since it is hard to calculate the derivatives of equation (9), we use the following Wilcoxon-Mann-Whitney statistic [4], which is an unbiased estimator of $AUC(P_\theta, \tau)$:

$$AUC^{WMW}(P_\theta, \tau) = \frac{\sum_{i \in D^\tau} \sum_{j \in D^{!\tau}} \delta\left(P_\theta(y_i^\tau | X) > P_\theta(y_j^\tau | X)\right)}{|D^\tau| \cdot |D^{!\tau}|} \quad (10)$$

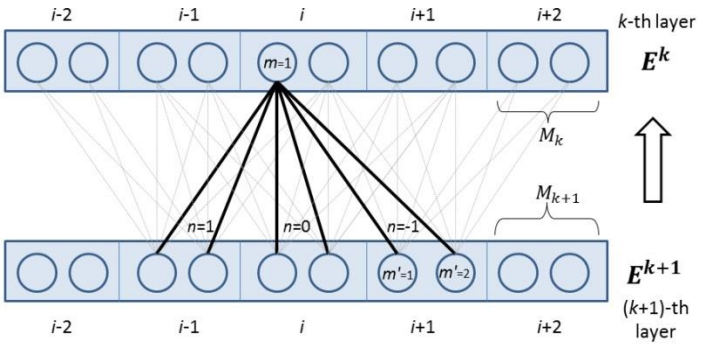
Finally, by summing over all labels, the overall AUC objective function is $\sum_\tau AUC^{WMW}(P_\theta, \tau)$.

Approximation. For a large dataset, the computational cost of AUC by equation (10) is high. Recently, Calders *et. al.* [5] proposed a polynomial approximation of AUC which can be computed in linear time. The key idea is to approximate the indicator function $\delta(x > 0)$, where x represents $(P_\theta(y_i^\tau | X) - P_\theta(y_j^\tau | X))$, by a polynomial Chebyshev approximation. That is, we approximate $\delta(x > 0)$ by $\sum_{\mu=0}^{d} c_\mu x^\mu$ where d is the degree and c_μ the coefficient of the polynomial [5]. Let $n_1 = |D^\tau|$ and $n_0 = |D^{!\tau}|$. Using the polynomial Chebyshev approximation, we can approximate equation (10) as follows.

$$AUC^{WMW}(P_\theta, \tau) \approx \frac{1}{n_0 n_1} \sum_{\mu=0}^{d} \sum_{l=0}^{\mu} \gamma_{\mu l}\, s\left(P_\theta{}^l, D^\tau\right) v\left(P_\theta{}^{u-l}, D^{!\tau}\right) \quad (11)$$

where $\gamma_{\mu l} = c_\mu \binom{\mu}{l} (-1)^{\mu - l}$, $s(P^l, D^\tau) = \sum_{i \in D^\tau} P(y_i^\tau)^l$ and $v(P^l, D^{!\tau}) = \sum_{j \in D^{!\tau}} P(y_j^\tau)^l$. Note that we have $s(P^l, D^\tau) = \sum_{t=1}^{T} \sum_{i=1}^{L_t} \delta_{i,t}^\tau P(y_i^\tau)^l$ and a similar structure for $v(P^l, D^{!\tau})$.

S2.4 Calculation of gradient of DeepCNF by back-propagation



SFigure 2. Illustration of calculating the gradient of deep convolutional neural network from layer $k+1$ to layer k.

As shown in SFigure 2, we can calculate the neuron error values as well as the gradients of DeepCNF model at the k-th layer by back-propagation as follows.

$$E_i^k(h) = \eta(H_i^k(h)) * \sum_u \left[E_i(u) * U_{a,h}\right] \quad \text{if } k = K$$

$$E_i^k(h) = \eta(H_i^k(h)) * \sum_{n=-N_k}^{N_k} \sum_{h'=1}^{M_{k+1}} \left[E_{i+n}^{k+1}(h') * W_n^k(h', h)\right] \quad \text{if } k < K \qquad (12)$$

Where η is the derivative of the activation function π. In particular, it is $\eta(x) = (1 - x) * x$ and $\eta(x) = 1 - x * x$ for the sigmoid and tanh function, respectively. $\boldsymbol{E^k}$ is the neuron error value matrix at the k-th layer, with dimension $L \times M_k$. Finally, the gradient of the parameter W at the k-th layer is:

$$\nabla_{W_n^k(h,h')} = \sum_{i=1}^{L} \left[E_i^{k+1}(h) * H_{i+n}^k(h')\right] \qquad (13)$$

S3. Protein features

Given a protein sequence, we use the same feature set for the prediction of SS3/SS8, ACC, and DISO. The feature set could be divided into residue-related feature and evolution-related feature. Since our server has two prediction modes depending on if sequence profile is used or not. When sequence profile is used, each residue has residue- and evolution-related features. Otherwise, only residue-related features are used.

Residue-related features. (a) amino acid identity represented as a binary vector of 20 elements; (b) amino acid physic-chemical properties (7 values from Table 1 in [6]); (c) propensity of being at endpoints of a secondary structure segment (11 values from Table 1 in [7]); (d) correlated contact potential (40 values from Table 3 in [8]); and (e) reduced AAindex (5 values from Table 2 in [9]). These features may allow for a richer representation of amino acids [10, 11].

Evolution-related features. We use PSSM (position specific scoring matrix) generated by PSI-BLAST [12] to encode the evolutionary information of the sequence under prediction. We also use the HHM profile generated by HHpred [13], which is complementary to PSSM to some degree.

S4. Performance metric

For SS3/SS8 and ACC prediction, the performance of a method is measured by QX accuracy where X is the number of labels. Specifically, QX is defined as the percentage of residues for which the predicted X labels are correct. For instance, for SS3 and ACC prediction we use Q3 accuracy whereas for SS8 we use Q8.

For DISO prediction, we use some complex measurements based on the confusion matrix consisting of TP, TN, FP, and FN, for the disorder state under consideration. Specifically, TP (true positives) and TN (true negatives) are the numbers of correctly predicted disordered residues and ordered residues, respectively; whereas FP (false positives) and FN (false negatives) are the numbers of misclassified residues, respectively. Then we use balanced accuracy (bAcc), sensitivity (Sens), specificity (Spec), precision (Prec), Matthews correlation coefficient (Mcc), and area under the ROC curve (AUC) as the measurements. Specifically, sensitivity and specificity are defined as $\mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$ and $\mathrm{TN}/(\mathrm{TN} + \mathrm{FP})$, respectively. Precision is defined as $\mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$. Balanced accuracy is the average of sensitivity and specificity. Mcc is defined as $(\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN})/\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TN} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FN})}$.

==========

REFERENCES

1. Ma, J.; Peng, J.; Wang, S.; Xu, J., A conditional neural fields model for protein threading. *Bioinformatics* **2012,** 28, (12), i59-i66.
2. Lafferty, J.; McCallum, A.; Pereira, F. C., Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001. Morgan Kaufmann 2001 (2001).* **2001**.
3. Gross, S. S.; Russakovsky, O.; Do, C. B.; Batzoglou, S. In *Training conditional random fields for maximum labelwise accuracy*, Advances in Neural Information Processing Systems, 2006; 2006; pp 529-536.
4. Hanley, J. A.; McNeil, B. J., The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* **1982,** 143, (1), 29-36.
5. Calders, T.; Jaroszewicz, S., Efficient AUC optimization for classification. In *Knowledge Discovery in Databases: PKDD 2007*, Springer: 2007; pp 42-53.
6. Meiler, J.; Müller, M.; Zeidler, A.; Schmäschke, F., Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks. *Molecular modeling annual* **2001,** 7, (9), 360-369.
7. Duan, M.; Huang, M.; Ma, C.; Li, L.; Zhou, Y., Position‐specific residue preference features around the ends of helices and strands and a novel strategy for the prediction of secondary structures. *Protein Science* **2008,** 17, (9), 1505-1512.
8. Tan, Y. H.; Huang, H.; Kihara, D., Statistical potential‐based amino acid similarity matrices for aligning distantly related protein sequences. *Proteins: Structure, Function, and Bioinformatics* **2006,** 64, (3), 587-600.
9. Atchley, W. R.; Zhao, J.; Fernandes, A. D.; Drüke, T., Solving the protein sequence metric problem. *Proceedings of the National Academy of Sciences of the United States of America* **2005,** 102, (18), 6395-6400.
10. Walsh, I.; Martin, A. J.; Di Domenico, T.; Tosatto, S. C., ESpritz: accurate and fast prediction of protein disorder. *Bioinformatics* **2012,** 28, (4), 503-509.
11. Ma, J.; Wang, S., AcconPred: Predicting Solvent Accessibility and Contact Number Simultaneously by a Multitask Learning Framework under the Conditional Neural Fields Model. *BioMed Research International* **2015,** 2015.
12. Altschul, S. F.; Madden, T. L.; Schäffer, A. A.; Zhang, J.; Zhang, Z.; Miller, W.; Lipman, D. J., Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic acids research* **1997,** 25, (17), 3389-3402.
13. Söding, J.; Biegert, A.; Lupas, A. N., The HHpred interactive server for protein homology detection and structure prediction. *Nucleic acids research* **2005,** 33, (suppl 2), W244-W248.