JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 29, Number 2, 2022

Pp. 92–105
DOI: 10.1089/cmb.2021.0430

Research Articles

# ProALIGN: Directly Learning Alignments for Protein Structure Prediction via Exploiting Context-Specific Alignment Motifs

LUPENG KONG,[1–3,i] FUSONG JU,[1,2] WEI-MOU ZHENG,[4] JIANWEI ZHU,[5] SHIWEI SUN,[1,2] JINBO XU,[3] and DONGBO BU[1,2]

## ABSTRACT

**Template-based modeling (TBM), including homology modeling and protein threading, is one of the most reliable techniques for protein structure prediction. It predicts protein structure by building an alignment between the query sequence under prediction and the templates with solved structures. However, it is still very challenging to build the optimal sequence-template alignment, especially when only distantly related templates are available. Here we report a novel deep learning approach ProALIGN that can predict much more accurate sequence-template alignment. Like protein sequences consisting of sequence motifs, protein alignments are also composed of frequently occurring alignment motifs with characteristic patterns. Alignment motifs are context-specific as their characteristic patterns are tightly related to sequence contexts of the aligned regions. Inspired by this observation, we represent a protein alignment as a binary matrix (in which 1 denotes an aligned residue pair) and then use a deep convolutional neural network to predict the optimal alignment from the query protein and its template. The trained neural network implicitly but effectively encodes an alignment scoring function, which reduces inaccuracies in the handcrafted scoring functions widely used by the current threading approaches. For a query protein and a template, we apply the neural network to directly infer likelihoods of all possible residue pairs in their entirety, which could effectively consider the correlations among multiple residues. We further construct the alignment with maximum likelihood, and finally build a structure model according to the alignment. Tested on three independent data sets with a total of 6688 protein alignment targets and 80 CASP13 TBM targets, our method achieved much better alignments and 3D structure models than the existing methods, including**

[1]Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China.
[2]University of Chinese Academy of Sciences, Beijing, China.
[3]Toyota Technological Institute, Chicago, Illinois, USA.
[4]Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing, China.
[5]Microsoft Research Asia, Beijing, China.
[i]ORCID ID (https://orcid.org/0000-0002-0963-1293).
An earlier draft of this article was posted online as a preprint at bioRxiv (DOI: 10.1101/2020.12.28.424539).

**HHpred, CNFpred, CEthreader, and DeepThreader. These results clearly demonstrate the effectiveness of exploiting the context-specific alignment motifs by deep learning for protein threading.**

**Keywords:** deep learning and protein threading, protein alignment, protein structure prediction.

## 1. INTRODUCTION

PROTEIN TERTIARY STRUCTURES are essential to understanding protein functions. Experimental protein structure determination by X-ray crystallography, nuclear magnetic resonance, and cryoelectron microscopy is usually expensive and time-consuming, and thus, cannot keep up with the rapid accumulation of protein sequences (Roy et al., 2010). Hence, computational approaches to protein structure prediction purely from sequences are highly desirable.

Template-based modeling (TBM), including homology modeling and protein threading, is one of the widely used methods for protein structure prediction. The rationale underlying TBM is that protein fold types are limited and protein structures are more evolutionarily conserved than protein sequences (Lo Conte et al., 2000), thus enabling predicting the structure of a query protein by referring to the structure of homologous proteins. Specifically, for a query protein, TBM first aligns it to all candidate structures (called templates), computes the optimal alignment that maximizes a predefined scoring function, and finally constructs a 3D structure according to the optimal alignment and corresponding templates (Wu and Zhang, 2007). TBM has been very successful in predicting reasonable 3D models for about two thirds of the proteins without solved structures (Orengo and Thornton, 2005).

The performance of TBM relies on extracting informative sequence and structure features from query proteins and templates. Most of the existing sequence-template alignment approaches make heavy use of position-specific features, such as sequence profile, including position-specific scoring matrix (PSSM) (Altschul et al., 1997) and the profile hidden Markov Model (HMM) (Durbin et al., 1998). For example, HHpred (Söding, 2005) builds an alignment through the comparison of two profile HMMs. As a residue's context (i.e., its sequential neighbors) greatly affects its mutation, context-specific features are more informative than position-specific features. CS-BLAST (Biegert and Söding, 2009) exploits context-specific sequence features, and our previous study (Ma et al., 2013) exploits context-specific structure features, to obtain more accurate alignments than profile-based alignment approaches. Recently, predicted inter-residue contacts and distances have been proven to be effective in improving sequence-template alignments (Zhu et al., 2018; Zheng et al., 2019).

The performance of TBM also relies on a scoring function that can accurately measure alignment quality. A successful scoring function should effectively integrate both sequence and structure features. A simple linear combination of protein features is insufficient as most protein features (such as secondary structure and solvent accessibility) are highly correlated (Peng and Xu, 2009). To handle this, the conditional neural field approach uses a probabilistic nonlinear function to combine protein features, thus effectively reducing both overcounting and undercounting (Ma et al., 2012). DeepThreader (Zhu et al., 2018) and MRFalign (Ma et al., 2014) use an alignment scoring function that contains singleton terms and pairwise terms. Here, the singleton terms quantify how well a query residue can be aligned with a template residue, while pairwise terms quantify how well two query residues can be aligned with two template residues at a given distance. EigenThreader (Buchan and Jones, 2017) and CEthreader (Zheng et al., 2019) make use of predicted inter-residue contact map, while DeepThreader makes use of predicted distance map.

In this work, we present a novel approach called ProALIGN for protein alignment and threading. Unlike many existing approaches that build alignments using a handcrafted scoring function, our approach directly learns and infers protein alignments. Our approach is founded on the observation of alignment motifs: protein sequences consist of sequence motifs with conserved amino acid composition, and protein structures consist of structure motifs with conserved spatial shapes. Similarly, protein alignments also consist of alignment motifs formed by aligned structure motifs. Alignment motifs, which appear frequently in protein alignments, usually exhibit characteristic patterns.

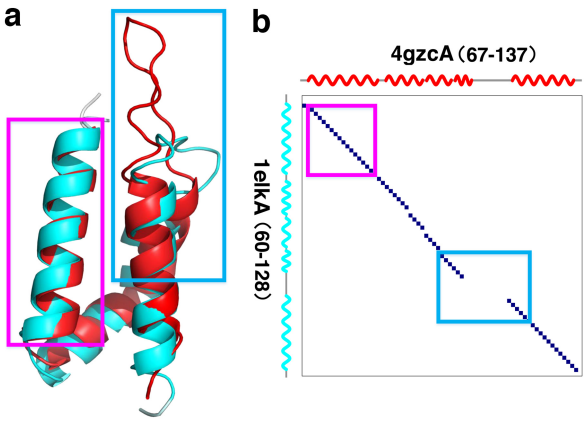


**FIG. 1.** Protein alignment and alignment motifs. **(a)** The structure alignment of proteins 4gzcA (in red) and 1elkA (in cyan). Only parts of these proteins are shown here due to limited space. The rectangle in magenta shows two aligned helices (Y69-V84 of 4gzcA vs. N63-V78 of 1elkA), while the rectangle in blue shows an alignment gap (F99-S114 of 4gzcA vs. T102-S123 of 1elkA). **(b)** The matrix representation of the alignment, where a dot represents an aligned residue pair, highlights alignment motifs with characteristic patterns. For example, the aligned helices show as a diagonal line (the rectangle in magenta), whereas the alignment gap shows as two diagonal lines with a shift between them (the rectangle in blue).

As shown in Figure 1, when representing a protein alignment as a matrix with dots denoting aligned residue pairs, aligned helices show as diagonal lines, while alignment gaps show as two diagonal lines with a shift between them. In addition, alignment motifs are context-specific as their characteristic patterns are tightly related to sequence contexts of the aligned regions. These observations enable us to recognize alignment motifs based on sequence contexts.

The deep convolutional network has shown great success in pattern recognition, especially for image processing such as classification and object detection (Cai et al., 2016; Krizhevsky et al., 2017). By treating alignment matrices as images, we utilize a deep convolutional neural network to directly learn protein alignments by integrating both sequential information and predicted inter-residue distances. The trained neural network implicitly but effectively encodes an alignment scoring function, which reduces inaccuracies in the handcrafted scoring function widely used by the current threading approaches. For a query protein and a template, we apply the neural network to infer likelihoods of all possible residue pairs in their entirety, which could effectively consider the correlations among multiple residues. We further construct the optimal alignment with maximum likelihood, and finally build a structure model according to the alignment.

Using three independent benchmark data sets, including 6688 protein alignment targets and 80 CASP13 TBM targets, we show that ProALIGN may produce much more accurate alignments and 3D models than the state-of-the-art approaches including HHpred, CNFpred, CEthreader, and DeepThreader.

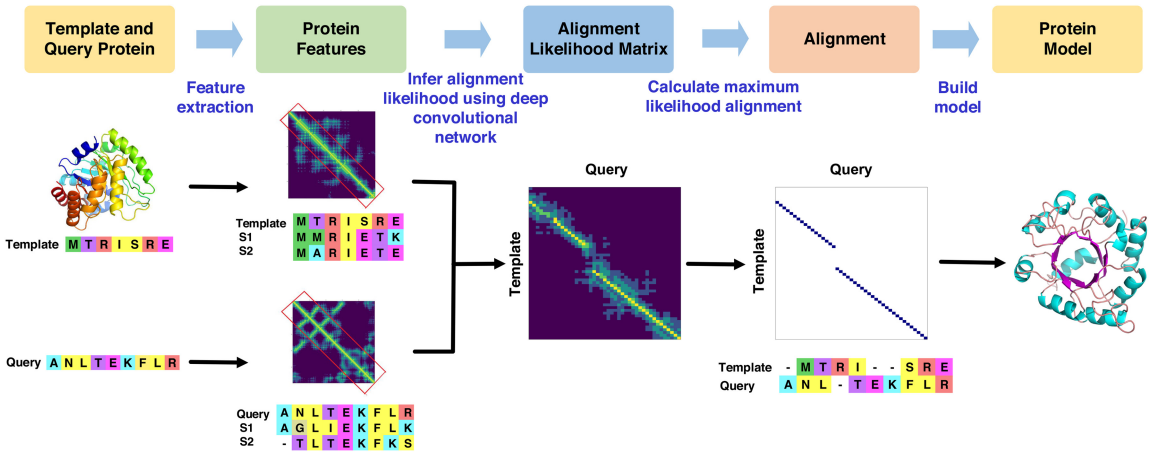


**FIG. 2.** The overall workflow of ProALIGN for TBM. It has four main steps: *(i) feature calculation; (ii) alignment likelihood inference; (iii) alignment generation;* and *(iv) 3D model building*. TBM, template-based modeling.

## 2. METHODS

### 2.1. Overall workflow of ProALIGN

The workflow of our ProALIGN approach for TBM is shown in Figure 2. It consists of the following four main steps: *(i) Feature calculation:* The features to be used include sequence profile, secondary structure, solvent accessibility, and inter-residue distances. *(ii) Alignment likelihood inference:* The input features are fed into a pretrained deep convolutional neural network, which predicts alignment likelihood for each residue pair (one query residue and one template residue). In our approach, alignment likelihood is represented as a matrix form. One entry in the matrix contains the match likelihood value of a residue pair. *(iii) Alignment generation:* Based on the alignment likelihood, we construct the optimal alignment with maximum likelihood. *(iv) Model building:* Build a 3D structure model by running MODELLER (Eswar et al., 2006) on the generated alignment.

Meanwhile, the first two steps are different from the existing approaches and thus are described in detail very soon.

### 2.2. Representation of alignment and formulation of alignment problem

For a query protein with $n$ residues $S = S_1 S_2 \ldots S_n$ and a template with $m$ residues $T = T_1 T_2 \ldots T_m$, we represent an alignment of them as a binary matrix $A$:

$$A = \begin{pmatrix} A_{11} & A_{12} & \ldots & A_{1n} \\ A_{21} & A_{22} & \ldots & A_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ A_{m1} & A_{m2} & \ldots & A_{mn} \end{pmatrix}. \quad (1)$$

Here, $A_{ij} = 1$ if and only if template residue $T_i$ is aligned with query residue $S_j$ (Fig. 3).

Note that a valid alignment $A$ poses two restrictions on the aligned residue pairs: *(i)* A residue aligns with at most one residue. *(ii)* Two aligned residue pairs cannot crossover, that is, suppose query residue $i$ aligns template residue $j$, and query residue $k$ ($k > i$) aligns template residue $l$, then $l$ should be larger than $j$. These restrictions make an alignment matrix to satisfy the following conditions: there is at most a ''1'' in one column and one row, and the aligned residue pairs show a shape of diagonal line rather than a back-diagonal line.

An alignment matrix can be intuitively treated as an image and thus amenable to deep convolutional neural networks, which are good at learning patterns in the alignment matrix. Given a protein pair $T$ and $S$, we use a deep convolutional neural network with parameter $\theta$ to learn and infer their alignment. The probability of an alignment $A$ can be represented as follows:

$$P(A|T, S, \theta) = P(A_{11}, A_{12}, \ldots, A_{ij}, \ldots, A_{mn}|T_1, T_2, \ldots, T_m, S_1, S_2, \ldots, S_n, \theta). \quad (2)$$

Specifically, for a pair of residues $T_i$ and $S_j$, the neural network predicts their likelihood of being aligned (i.e., $A_{ij} = 1$). Our neural network simultaneously estimates the likelihood of all the residue pairs being aligned so that it can take into consideration the correlations among residue pairs.

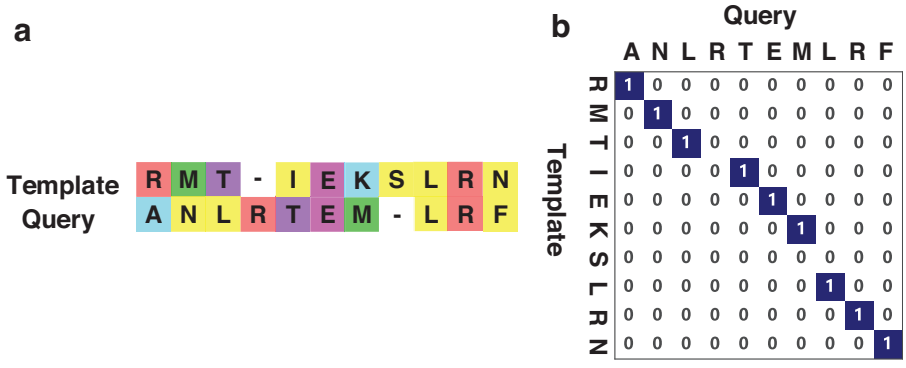


**FIG. 3.** Two equivalent representations of a protein alignment. **(a)** Representation of an alignment as a sequence pair with symbol ''-'' denoting gaps. **(b)** Matrix representation. Intuitively, an alignment matrix can be treated as an image and thus suitable for learning and inference by deep convolutional neural networks.

### 2.3. *Learning protein alignments using deep convolutional neural network*

We design a deep convolutional neural network to learn the relationship between the characteristic pattern of alignment motifs and the residue composition of the corresponding sequence contexts. Here, the residue composition of sequence contexts is represented using a collection of protein features, which construct the input of our neural network.

*2.3.1. Input features.* Our neural network takes sequence features and structure features of proteins as input. Specifically, for both template and query proteins, we first construct multiple sequence alignment through running HHblits (Remmert et al., 2012) against a sequence database, and then build PSSM, which could effectively depict the preference of amino acid types at each position.

The structure features used in this study include secondary structure, solvent accessibility, and inter-residue distances. For a template, secondary structure and solvent accessibility are calculated by running DSSP (Frishman and Argos, 1995) on the template structure. The inter-residue distances are calculated according to the coordinates of $C_\beta$ atom of residues. For a query protein, we predict its secondary structure and solvent accessibility using RaptorX-Property (Wang et al., 2016) and predict inter-residue distance using ProFOLD (Ju et al., 2021). For each residue, we use the distance with neighbor residues within a window. Here, we set the window size as $C = 33$ (the other settings of window size, e.g., 1–41, were also examined and the prediction results are stable when $C \geq 29$).

Thus, for each residue in query or template protein, we obtain a total of 356 features, including 20 PSSM features, 3 secondary structure features, 3 solvent accessibility features, and 330 inter-residue distance features (33 neighboring residues times 10 distance bins). For each residue pair containing a query residue and a template residue, we concatenate their features, thus obtaining a total of $m \times n \times 712$ features. The features of all residue pairs construct template-query joint features (Fig. 4). Our neural network takes these joint features as input.

*2.3.2. Neural network architecture.* To learn the relationship between alignment and the joint protein features, we design a deep neural network with full convolutional architecture (Long et al., 2015). Full convolutional architecture does not assume input length to be fixed, and thus can handle proteins with variable lengths. Using a multilayer convolutional stack structure, a full convolutional network is expected to capture both coarse information and fine information. The shallower convolutional layers have a smaller receptive field and learn detailed features of local areas, while the deeper layers have a larger receptive field and thus can learn more global and abstract information (Fig. 4). Here, we expect that the shallower layers can capture the characteristic pattern of alignment motifs, and the deeper layers can learn the correlation among alignment motifs.

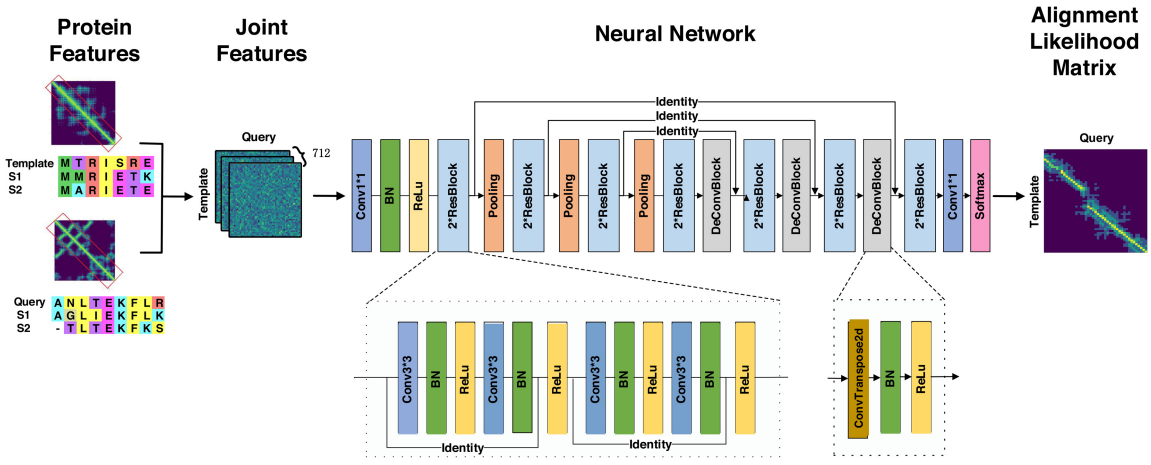


**FIG. 4.** Architecture of the deep neural network used by ProALIGN. The neural network takes sequence features and structure features of proteins as input, and outputs an alignment likelihood matrix in its entirety. The $(i, j)$ entry of the matrix represents the likelihood whether template residue $i$ assigns with query residue $j$. The network has a multilayer convolution stack structure. Using this structure, we expect that the shallower layers can capture the characteristic pattern of alignment motifs, and the deeper layers can learn the correlation among alignment motifs.

We also add several shortcut connections (identity operations) (Long et al., 2015) into the neural network. Using these shortcut connections, we can effectively combine the information captured by shallower and deeper layers. To reduce the risk of gradient vanishing, we use the residual network structure (He et al., 2016). The whole neural network adopts ReLU (Nair and Hinton, 2010) as activation function.

*2.3.3. Loss function.* Each sample of the training set used in this study contains two proteins with their structural alignment (called reference alignment) as label. The objective of training process is to find the optimal network parameter to maximize the probability that the network generates the given reference alignment. For this end, we use the cross-entropy loss function as follows.

$$\mathcal{L}(A) = \sum_{i,j} a_{ij} \log P(A_{ij} = a_{ij}) + (1 - a_{ij}) \log (1 - P(A_{ij} = a_{ij})). \tag{3}$$

Here, $a_{ij} \in \{0, 1\}$ denotes label for residue pair $(i, j)$, that is, whether residue $i$ of template protein aligns with residue $j$ of the query protein in their reference alignment, and $A_{ij}$ denotes network's output for these two residues.

*2.3.4. Training the neural network.* We train the neural network using Adam algorithm (Kingma and Ba, 2014). To prevent overfitting, we adopt weight decay operation with a coefficient of 0.001 during gradient updating.

The neural network is trained using the minibatch (Tieleman, 2008) technique. As proteins in a batch always have different sizes, zeroes should be padded to make their sizes equal. However, when a protein has extremely long size, this zero-padding operation might lead to a large memory requirement that exceeds GPU's capacity. To overcome this difficulty, we apply a large batch size for short proteins and a small batch size for long proteins.

## 2.4. Inferring protein alignment using deep convolution neural network

We apply the trained neural network on a query protein $S$ and a template $T$, and obtain an alignment likelihood matrix. The $(i, j)$ entry of the matrix represents the likelihood whether template residue $i$ assigns with query residue.

Note that a valid alignment matrix $A$ poses two restrictions: there is at most a ''1'' in one column and one row, and the arrangement of aligned residue pairs satisfies from top left to bottom right. Here, we denote the set of all valid alignments as $S$. The optimal alignment $A_{opt}$ is expected to be obtained by maximizing the matrix likelihood, at the same time satisfying the definition of valid alignment, that is,

$$\arg\max_{A \in S} \log P(A|S, T) - \lambda \cdot r(A). \tag{4}$$

Here, $r(A)$ represents a regular term to make the alignment compacted. In this study, we set $r(A)$ as the length of actual alignment area, that is, the area from the first aligned residue pair to the last aligned residue pair in built alignment. We expected this regular term to penalize too many mismatch residues in alignment. The optimal setting of weight $\lambda$ is decided using a validation data set (see Supplementary Data, Section 3, for more details). In this study, we apply dynamic programming to solve the optimal alignment that satisfies valid alignment restrictions (see Supplementary Data, Section 2, for more details).

Table 1. Reference-Dependent Alignment Accuracy on Test5.6K Data Set

| | *TMalign* | | *DeepAlign* | | *MMLigner* | |
|---|---|---|---|---|---|---|
| *Method* | *Exact match* | *4-offset* | *Exact match* | *4-offset* | *Exact match* | *4-offset* |
| HHpred | 35.3% | 50.9% | 40.2% | 52.6% | 40.0% | 53.8% |
| CNFpred | 40.5% | 59.7% | 46.5% | 62.3% | 44.8% | 61.9% |
| CEthreader | 37.4% | 60.5% | 41.5% | 62.1% | 41.6% | 62.0% |
| DeepThreader | 46.4% | 64.6% | 52.3% | 67.2% | 50.3% | 65.9% |
| ProALIGN | **52.1%** | **74.7%** | **59.8%** | **77.5%** | **54.6%** | **73.5%** |

Here, ''4-offset'' means that 4-position off the exact match is allowed. The best results are shown in bold.

Table 2. Reference-Dependent Alignment Accuracy on Test1K Data Set

| Method | TMalign | | DeepAlign | | MMLigner | |
|---|---|---|---|---|---|---|
| | *Exact match* | *4-offset* | *Exact match* | *4-offset* | *Exact match* | *4-offset* |
| HHpred | 58.0% | 72.9% | 64.5% | 75.4% | 65.5% | 78.4% |
| CNFpred | 59.7% | 75.0% | 65.7% | 77.6% | 66.6% | 79.3% |
| CEthreader | 57.2% | 76.3% | 61.7% | 77.4% | 63.3% | 79.0% |
| DeepThreader | 64.6% | 79.9% | 71.5% | 82.6% | 71.3% | 83.5% |
| ProALIGN | **69.0%** | **85.3%** | **75.1%** | **87.0%** | **74.2%** | **87.2%** |

Here, "4-offset" means that 4-position off the exact match is allowed. The best results are shown in bold.

## 3. RESULTS AND DISCUSSIONS

### 3.1. Training and test data

*3.1.1. Training data set.* We used the PDB25 data set (constructed from PISCES [Wang and Dunbrack Jr, 2003] in 2015, containing 7952 proteins in which any two proteins share $\leq$25% sequence identity) to construct the training and test data set. To guarantee no overlap between training set and testing set, we first divided the PDB25 proteins into two parts: one part containing 6245 proteins for training and validation, and the other part containing 1707 proteins for testing. From the first part (containing 6245 proteins), we calculate the structure alignment of any two proteins as reference alignment using the structure alignment tool DeepAlign (Wang et al., 2013). After filtering out low-quality alignments (TMscore $<$ 0.40), we obtained a total of 75,874 alignments.

From these alignments, we randomly selected 67,106 alignments as training set and used the remainder 8768 as validation set. In addition, we also randomly selected 1000 alignment pairs (Valid1K) to decide the hyperparameter $\lambda$ used in Eq. (4).

*3.1.2. Test data set.* From the second part of PDB25 proteins (containing 1707 proteins), we generated a total of 5688 alignments for testing (referred to as *Testing5.6K*). In addition, we also evaluated our approach on the 1000 protein alignments (referred to as *Testing1K*) used by DeepThreader. After removing the low-quality alignments (TMscore $<$ 0.40), we acquired a total of 769 alignments with high quality. According to structure similarity, we split these alignments into three groups, that is, easy group with TMscore in (0.80, 1], medium group with TMscore in (0.60. 0.80], and hard group with TMscore in (0.40, 0.60].

To evaluate threading performance, we used the 80 domains released by CASP13 organizer, including 22 TBM Hard domains, 45 TBM Easy domains, and 13 FM (free modeling)/TBM domains. We used PDB40 containing 32,363 proteins as template. It is worth pointing out that the PDB40 was constructed on April 4, 2018, before CASP13, thus avoiding potential misusing of the native structure of CASP13 domains as templates.

### 3.2. Evaluation method

To evaluate alignments, we calculated both reference-dependent and reference-independent alignment accuracy. Here, the reference-dependent accuracy is defined as the percentage of correctly aligned positions judged by the reference alignments, and the reference-independent accuracy of an alignment is defined as quality of the 3D model generated from the alignment. Here, we applied MODELLER to generate the 3D model from an alignment, and use TMscore and GDT (global distance test) score to measure model quality.

Table 3. Reference-Independent Alignment Accuracy Measured by TMscore and GDT on Test5.6K

| Group | No. of alignments | HHpred | | CNFpred | | CEthreader | | DeepThreader | | ProALIGN | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *TMscore* | *GDT* | *TMscore* | *GDT* | *TMscore* | *GDT* | *TMscore* | *GDT* | *TMscore* | *GDT* |
| Test5.6K easy | 216 | 0.798 | 0.673 | 0.810 | 0.690 | 0.803 | 0.679 | 0.806 | 0.682 | **0.821** | **0.700** |
| Test5.6K medium | 1971 | 0.533 | 0.440 | 0.580 | 0.486 | 0.578 | 0.481 | 0.614 | 0.513 | **0.653** | **0.550** |
| Test5.6K hard | 3501 | 0.310 | 0.244 | 0.382 | 0.301 | 0.389 | 0.306 | 0.435 | 0.344 | **0.476** | **0.380** |
| Test5.6K | 5688 | 0.406 | 0.328 | 0.467 | 0.380 | 0.470 | 0.381 | 0.511 | 0.416 | **0.551** | **0.451** |

The best performance is shown in bold.

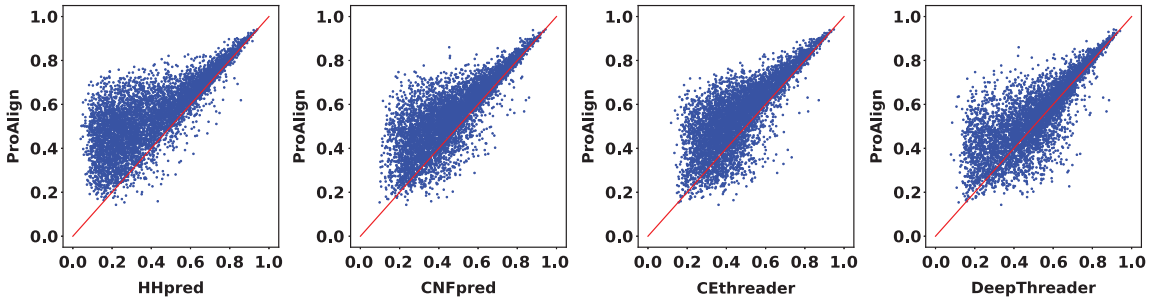


**FIG. 5.** Head-to-head comparison of the quality of models generated using ProALIGN, HHpred, CNpred, CEthreader, and DeepThreader on Test5.6K data set. Here, the model quality is measured using TMscore.

We compared ProALIGN with the state-of-the-art alignment approaches, including HHpred, CNFpred, CEthreader, and DeepThreader. For the sake of fair comparison, we executed these programs with the same MSA (constructed using HHblits with three iterations and E-value set to 0.001 against Uniprot20_2016 [Apweiler et al., 2004]). We run HHpred with mode ''-mact 0.1'' and run CEthreader with ''EigenProfileAlign'' mode.

### 3.3. *Reference-dependent alignment accuracy*

Table 1 shows the reference-dependent alignment accuracy on Test5.6K data set. To reduce the potential bias in generating reference alignments, we evaluated our approach using reference alignments generated using three structure alignment tools, including TMalign (Zhang and Skolnick, 2005), DeepAlign, and MMLigner (Collier et al., 2017).

When using the TMalign reference alignment, ProALIGN shows an alignment accuracy (i.e., exact match) of 52.1%, which outperforms DeepThreader, CEthreader, CNFpred, and HHpred by 5.7%, 14.7%, 11.6%, and 16.8%, respectively. If 4-position off the exact match is allowed in calculating alignment accuracy, ProALIGN shows an alignment accuracy of 74.7%, higher than DeepThreader, CEthreader, CNFpred, and HHpred by 10.1%, 14.2%, 15%, and 23.8%, respectively. Similar observations can be obtained when using DeepAlign and MMLigner reference alignment.

As shown in Table 2, the alignment accuracy of ProALIGN is much higher on Test1K data set (69.0% for exact match, and 85.3% for 4-offset when using TMalign reference alignment). Again, ProALIGN significantly outperforms the existing approaches. These results clearly suggest that ProALIGN can generate more accurate alignments for proteins.

### 3.4. *Reference-independent alignment accuracy*

Furthermore, we assessed ProALIGN and other alignment approaches in terms of reference-independent alignment accuracy. As shown in Table 3, the model built by ProALIGN has an average TMscore of 0.551, which is much better than DeepThreader (0.511), CEthreader (0.470), CNFpred (0.467), and HHpred (0.406). Besides examining the average TMscore of all generated models, we further investigated each model individually using head-to-head analysis. Figure 5 suggests that ProALIGN could generate higher quality models in most cases.

In particular, for the Test5.6K Easy group, all of these alignment approaches could generate high-quality models. For the Test5.6K Medium group, the superiority of ProALIGN is obvious: the average TMscore of the generated model is 0.653, higher than HHpred (0.533), CNFpred (0.580), CEthreader (0.578), and DeepThreader (0.614). For the Test5.6K Hard group, ProALIGN shows an average TMscore of 0.476, and outperforms HHpred, CNFpred, CEthreader, and DeepThreader by 0.166, 0.094, 0.087, and 0.041, respectively.

Table 4. Reference-Independent Alignment Accuracy Measured by TMscore and GDT on Test1K

| *Group* | *No. of alignments* | *HHpred* | | *CNFpred* | | *CEthreader* | | *DeepThreader* | | *ProALIGN* | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *TMscore* | *GDT* | *TMscore* | *GDT* | *TMscore* | *GDT* | *TMscore* | *GDT* | *TMscore* | *GDT* |
| Test1K easy | 121 | 0.822 | 0.752 | 0.831 | 0.763 | 0.826 | 0.754 | 0.838 | 0.769 | **0.846** | **0.776** |
| Test1K medium | 334 | 0.647 | 0.544 | 0.660 | 0.558 | 0.655 | 0.550 | 0.694 | 0.586 | **0.715** | **0.605** |
| Test1K hard | 314 | 0.395 | 0.314 | 0.418 | 0.333 | 0.418 | 0.330 | 0.472 | 0.375 | **0.497** | **0.398** |
| Test1K | 769 | 0.571 | 0.483 | 0.588 | 0.498 | 0.585 | 0.492 | 0.626 | 0.529 | **0.646** | **0.548** |

The best performance is shown in bold.

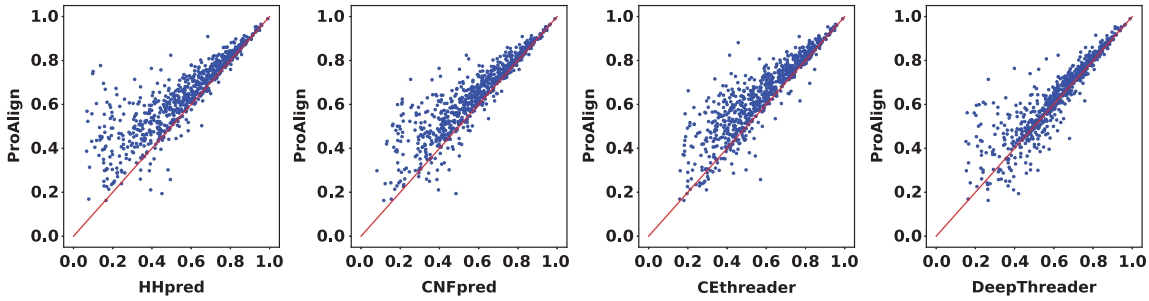


**FIG. 6.** Head-to-head comparison of the quality of models generated using ProALIGN, HHpred, CNpred, CEthreader, and DeepThreader on Test1K data set. Here, the model quality is measured using TMscore.

Evaluation on Test1K data set confirmed the superiority of ProALIGN (Table 4 and Fig. 6). These results together suggested that ProALIGN could generate a 3D model with higher quality than the existing approaches, especially for the medium and hard protein alignments.

### *3.5. Threading performance on CASP13 TBM targets*

When using the threading approach for protein structure prediction, we need to align the query protein with all templates, and then select the most reliable alignment and template to build the 3D model. Thus, a strategy is required to rank alignments.

As performed by CEthreader, we ranked alignments according to the ratio of query contacts that are aligned onto template contacts, that is, $CMO_d(A) = \|CM_d^Q \cap CM_d^T \|/\| CM_d^Q)\|$. Here, $CMO_d^Q$ and $CMO_d^T$ represent the number of residue contacts with distance less than threshold $d$ for query protein $Q$ and template $T$, respectively. The average of the ratios at four different distance thresholds $CMO(A) = \frac{1}{4}(CMO_8(A) + CMO_{10}(A) + CMO_{12}(A) + CMO_{14}(A))$ was finally used to rank alignments. For each target, we evaluated two predicted models: (i) top1 model: the model built based on the alignment ranked first, and (ii) top5 model: the best model among the models built based on top5 alignments.

Table 5 shows quality (measured using TMscore and GDT) of the top1 and top5 models predicted for the CASP13 TBM targets. Our method achieved a higher average TMscore (0.740/0.754) than HHpred (0.665/0.709), CNFpred (0.674/0.705), CEthreader (0.687/0.721), and DeepThreader (0.698/0.725). The superiority of ProALIGN is more obvious for the TBM Hard and FM/TBM targets. In the case of TBM Hard targets, ProALIGN generated top1/top5 model with an average TMscore of 0.688/0.703, which is significantly higher than HHpred (0.594/0.658), CNFpred (0.606/0.637), CEthreader (0.625/0.664), and DeepThreader (0.626/0.659). We can observe similar results when using GDT as quality measure.

In addition to comparing the average TMscore of predicted models, we further examined the number of high-quality models. Head-to-head comparison of these approaches (Fig. 7) illustrated that ProALIGN generated more high-quality models than other approaches. For instance, ProALIGN outperformed CEthreader on 59 targets, while CEthreader outperformed ProALIGN on 20 targets. Moreover, on 17 targets, ProALIGN generated models with a significantly higher quality (the difference of TMscore $> 0.10$). In addition, ProALIGN generated high-quality models (TMscore $> 0.40$) on 76 targets. These results clearly demonstrate the performance of alignment generation and alignment ranking used by ProALIGN.

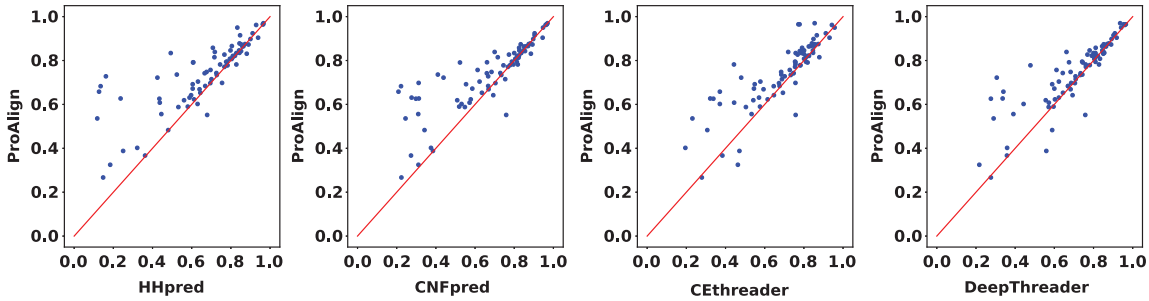


**FIG. 7.** Head-to-head comparison of quality (measured using TMscore) of the top1 model predicted by HHpred, CNFpred, CEthreader, and DeepThreader on 80 CASP13 TBM targets.

Table 5. Threading Performance on 80 CASP13 Domain Targets

| Method | ALL TMscore | ALL GDT | FM/TBM TMscore | FM/TBM GDT | TBM hard TMscore | TBM hard GDT | TBM easy TMscore | TBM easy GDT |
|---|---|---|---|---|---|---|---|---|
| HHpred | 0.665/0.709 | 0.600/0.646 | 0.414/0.460 | 0.389/0.439 | 0.594/0.658 | 0.491/0.550 | 0.773/0.807 | 0.715/0.753 |
| CNFpred | 0.674/0.705 | 0.614/0.645 | 0.421/0.464 | 0.406/0.451 | 0.606/0.637 | 0.504/0.534 | 0.781/0.807 | 0.727/0.756 |
| CEthreader | 0.687/0.721 | 0.619/0.655 | 0.461/0.526 | 0.455/0.510 | 0.625/0.664 | 0.517/0.552 | 0.782/0.805 | 0.716/0.746 |
| DeepThreader | 0.698/0.725 | 0.636/0.663 | 0.503/0.532 | 0.475/0.512 | 0.626/0.659 | 0.528/0.555 | 0.790/0.813 | 0.735/0.760 |
| ProALIGN | **0.740/0.754** | **0.674/0.691** | **0.570/0.584** | **0.553/0.561** | **0.686/0.708** | **0.571/0.601** | **0.815/0.826** | **0.758/0.773** |

Here, we show quality (measured using TMscore and GDT) of the top1/top5 predicted models.
The best performance is shown in bold.
ALL, all targets; FM, free modeling; TBM, template-based modeling.

### 3.6. Analysis of feature contributions to protein alignment

As mentioned above, ProALIGN uses a collection of protein features, including PSSM, secondary structure, solvent accessibility, and inter-residue distances. To access the contribution of these features to protein alignment, we fed various feature combinations to the deep convolutional neural network used by ProALIGN.

Table 6 demonstrates that when only PSSM is used, ProALIGN showed an average TMscore of 0.518. The addition of secondary structure and solvent accessibility leads to increase of alignment accuracy (0.532). In contrast, the addition of inter-residue distance leads to a more significant performance increase (0.545). These two types of features are complementary to a certain degree, especially for the hard group; thus, when using all of these features, alignment accuracy further increases to 0.554.

### 3.7. The contribution of deep convolutional neural network to alignment likelihood inference

ProALIGN uses a deep convolutional neural network to infer alignment likelihood of all residue pairs in their entirety; thus, in principle, this network is expected to consider correlations among residue pairs and exploit the characteristic pattern of alignment motifs. To examine this issue, we compared ProALIGN with another approach (denoted as IndivInferrer) that infers alignment likelihood for each residue pair individually using a fully connected neural network.

Table 7 suggests that ProALIGN considerably outperformed IndivInferrer in terms of alignment accuracy (TMscore 0.640 vs. 0.602). We further performed a case study using 1qd1A as query protein and 3dfeA as template. As shown in Figure 8, compared with ProALIGN, IndivInferrer reported more aligned regions with overestimated likelihood. For example, IndivInferrer assigned the aligned regions (R80–R135 of 1qd1A vs. R60–R100 of 3dfeA, blue rectangle in Figure 8b) with a likelihood exceeding 0.5. These incorrect aligned regions precluded IndivInferrer from finding the optimal alignment. In fact, the alignment generated using IndivInferrer deviates greatly from the reference alignment.

In contrast, ProALIGN overestimated alignment likelihood on only a few regions. The alignment generated by ProALIGN is close to the reference alignment, and the model thus built is significantly better than that built using IndivInferrer (TMscore: 0.753 vs. 0.299).

Table 6. Reference-Independent Alignment Accuracy of ProALIGN When Using Various Combinations of Features As Input

| Group | No. of alignments | PSSM TMscore | PSSM GDT | PSSM+SA+SS TMscore | PSSM+SA+SS GDT | PSSM+Dist TMscore | PSSM+Dist GDT | PSSM+SA+SS+Dist TMscore | PSSM+SA+SS+Dist GDT |
|---|---|---|---|---|---|---|---|---|---|
| Test1K easy | 121 | 0.821 | 0.753 | 0.835 | 0.766 | 0.836 | 0.768 | 0.840 | 0.772 |
| Test1K medium | 334 | 0.673 | 0.567 | 0.688 | 0.580 | 0.702 | 0.593 | 0.709 | 0.600 |
| Test1K hard | 314 | 0.446 | 0.352 | 0.466 | 0.370 | 0.476 | 0.379 | 0.490 | 0.392 |
| Test1K | 769 | 0.604 | 0.508 | 0.620 | 0.524 | 0.631 | 0.533 | 0.640 | 0.542 |

Data set: Test1K.
PSSM, position-specific scoring matrix; SA, solvent accessibility; SS, secondary structure.

Table 7. Reference-Independent Alignment Accuracy of ProALIGN and IndivInferrer That Infers Alignment Likelihood for Each Residue Pair Individually

| Group | No. of alignments | *IndivInferrer* TMscore | GDT | *ProALIGN* TMscore | GDT |
|---|---|---|---|---|---|
| Test1K easy | 121 | 0.831 | 0.760 | 0.840 | 0.772 |
| Test1K medium | 334 | 0.670 | 0.562 | 0.709 | 0.600 |
| Test1K hard | 314 | 0.441 | 0.345 | 0.490 | 0.392 |
| Test1K | 769 | 0.602 | 0.505 | 0.640 | 0.542 |

Data set: Test1K.

We also observed that although ProALIGN reports multiple aligned regions with high likelihood, these regions essentially compose two alignments and both alignments can be used to build high-quality models (Fig. 9). In particular, using the first alignment (S1-R105 of 1qd1A vs. S1-G111 of 3dfeA), we obtained a predicted model with TMscore of 0.753, while using the second alignment (A172-C285 of 1qd1A vs. S1-G111 of 3dfeA), we could also obtain a perfect model with TMscore of 0.593.

Taken together, these results clearly demonstrate the advantage of deep convolutional network in considering correlations among residue pairs and the superiority of ProALIGN over IndivInferrer in inferring alignment likelihood.

## 4. CONCLUSION

The results presented in this study have clearly highlighted the special features of directly learning and inferring protein alignment through exploiting context-specific alignment motifs. The abilities of our approach to protein structure prediction have been demonstrated using 6688 protein alignments and 80 CASP13 TBM targets as examples. Compared with the state-of-the-art threading approaches, ProALIGN could achieve much more accurate alignments and predicted structure models.

In principle, deep convolutional networks have large receptive fields when using a deeper structure; however, the ability to detect correlation among long-range residues is limited in practice (Luo et al., 2016). The application of graph convolutional networks (Kipf and Welling, 2016) is promising in circumventing this limitation. In future studies, this network will be incorporated into ProALIGN for the potential to model long-range correlations more accurately.

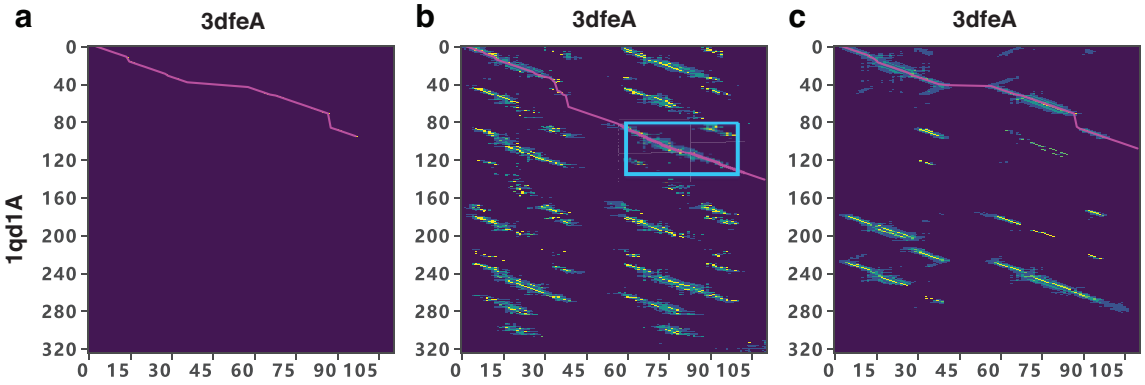


**FIG. 8.** Comparison of ProALIGN and IndivInferrer in terms of accuracy of inferred alignment likelihood. **(a)** The reference alignment of proteins 1qd1A and 3dfeA. **(b)** Alignment likelihood inferred using IndivInferrer approach, which predicts alignment likelihood for each residue pair individually. The aligned regions (E80-Q135 of 1qd1A vs. G60-M100 of 3dfeA, blue rectangle) are assigned with high likelihood exceeding 0.5. However, this aligned region is incorrect, and the derived alignment deviates greatly from the reference alignment. **(c)** Alignment likelihoods inferred using ProALIGN.

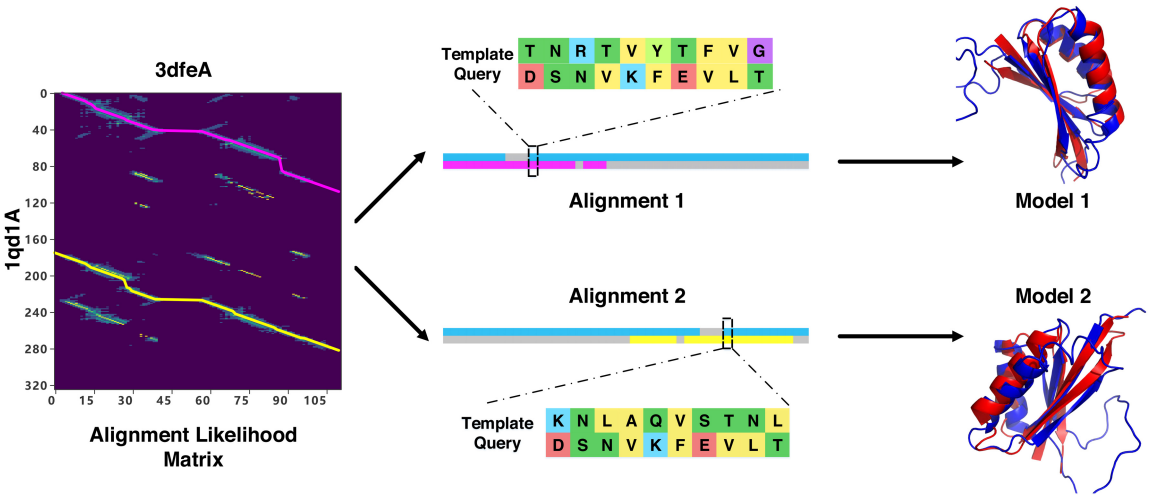


**FIG. 9.** Two constructed alignments according to the alignment likelihoods inferred by ProALIGN for proteins 1qd1A and 3dfeA. Using the first alignment (magenta line, S1-R105 of 1qd1A vs. S1-G111 of 3dfeA), we obtained a predicted model with TMscore of 0.753, while using the second alignment (yellow line, A172-C285 of 1qd1A vs. S1-G111 of 3dfeA), we also obtained a perfect model with TMscore of 0.593.

In summary, our approach ProALIGN should greatly facilitate understanding protein tertiary structures and functions.

## ACKNOWLEDGMENT

We greatly appreciate Shuai Cheng Li for fruitful discussions on this study.

## AUTHOR DISCLOSURE STATEMENT

The authors declare they have no conflicting financial interests.

## FUNDING INFORMATION

This work is supported by the National Natural Science Foundation of China grant 2020YFA0907000 to D.B. and the National Natural Science Foundation of China grants 62072435, 31770775, and 82130055 to S.S. and D.B. This work is also supported by the National Institutes of Health grant R01GM089753 to J.X. and the National Science Foundation grant DBI1564955 to J.X. The funders had no role in the study design, data collection and analysis, decision to publish, or preparation of the article.

## SUPPLEMENTARY MATERIAL

Supplementary Data

## REFERENCES

Altschul, S.F., Madden, T.L., Schäffer, A.A., et al. 1997. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res.* 25, 3389–3402.

Apweiler, R., Bairoch, A., Wu, C.H., et al. 2004. UniProt: The universal protein knowledgebase. *Nucleic Acids Res.* 32, D115–D119.

Biegert, A., and Söding, J. 2009. Sequence context-specific profiles for homology searching. *Proc. Natl. Acad. Sci*. 106, 3770–3775.

Buchan, D.W., and Jones, D.T. 2017. EigenTHREADER: Analogous protein fold recognition by efficient contact map threading. *Bioinformatics*. 33, 2684–2690.

Cai, Z., Fan, Q., Feris, R.S., et al. 2016. A unified multi-scale deep convolutional neural network for fast object detection, 354–370. In European Conference on Computer Vision. Amsterdam, The Netherlands.

Collier, J.H., Allison, L., Lesk, A.M., et al. 2017. Statistical inference of protein structural alignments using information and compression. *Bioinformatics*. 33, 1005–1013.

Durbin, R., Eddy, S.R., Krogh, A., et al. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, England.

Eswar, N., Webb, B., Marti-Renom, M.A., et al. 2006. Comparative protein structure modeling using Modeller. *Curr. Protoc. Bioinformatics*. 15, 5–6.

Frishman, D., and Argos, P. 1995. Knowledge-based protein secondary structure assignment. *Proteins.* 23, 566–579.

He, K., Zhang, X., Ren, S., et al. 2016. Deep residual learning for image recognition, 770–778. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, Nevada.

Ju, F., Zhu, J., Shao, B., et al. 2021. Copulanet: Learning residue co-evolution directly from multiple sequence alignment for protein structure prediction. *Nat. Commun.* 12, 1–19.

Kingma, D.P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Kipf, T.N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Krizhevsky, A., Sutskever, I., and Hinton, G.E. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM*. 60, 84–90.

Lo Conte, L., Ailey, B., Hubbard, T.J., et al. 2000. SCOP: A structural classification of proteins database. *Nucleic Acids Res.* 28, 257–259.

Long, J., Shelhamer, E., and Darrell, T. 2015. Fully convolutional networks for semantic segmentation, 3431–3440. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, Massachusetts, USA.

Luo, W., Li, Y., Urtasun, R., et al. 2016. Understanding the effective receptive field in deep convolutional neural networks, 4898–4906. In Advances in Neural Information Processing Systems 29. Barcelona Spain.

Ma, J., Peng, J., Wang, S., et al. 2012. A conditional neural fields model for protein threading. *Bioinformatics*. 28, i59–i66.

Ma, J., Wang, S., Wang, Z., et al. 2014. MRFalign: Protein homology detection through alignment of Markov random fields. *PLoS Comput. Biol.* 10, e1003500.

Ma, J., Wang, S., Zhao, F., et al. 2013. Protein threading using context-specific alignment potential. *Bioinformatics*. 29, i257–i265.

Nair, V., and Hinton, G.E. 2010. Rectified linear units improve restricted boltzmann machines. In International Conference on Machine Learning. Haifa, Israel.

Orengo, C.A., and Thornton, J.M. 2005. Protein families and their evolution—A structural perspective. *Annu. Rev. Biochem.* 74, 867–900.

Peng, J., and Xu, J. 2009. Boosting protein threading accuracy, 31–45. In Annual International Conference on Research in Computational Molecular Biology. Tucson, Arizona, USA.

Remmert, M., Biegert, A., Hauser, A., et al. 2012. HHblits: Lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nat. Methods*. 9, 173–175.

Roy, A., Kucukural, A., and Zhang, Y. 2010. I-TASSER: A unified platform for automated protein structure and function prediction. *Nat. Protoc.* 5, 725–738.

Söding, J. 2005. Protein homology detection by HMM–HMM comparison. *Bioinformatics*. 21, 951–960.

Tieleman, T. 2008. Training restricted Boltzmann machines using approximations to the likelihood gradient, 1064–1071. In International Conference on Machine Learning. Helsinki, Finland.

Wang, G., and Dunbrack Jr, R.L. 2003. PISCES: A protein sequence culling server. *Bioinformatics* 19, 1589–1591.

Wang, S., Li, W., Liu, S., et al. 2016. RaptorX-Property: A web server for protein structure property prediction. *Nucleic Acids Res.* 44, 430–435.

Wang, S., Ma, J., Peng, J., et al. 2013. Protein structure alignment beyond spatial proximity. *Sci. Rep.* 3, 1448.

Wu, S., and Zhang, Y. 2007. LOMETS: A local meta-threading-server for protein structure prediction. *Nucleic Acids Res.* 35, 3375–3382.

Zhang, Y., and Skolnick, J. 2005. TM-align: A protein structure alignment algorithm based on the TM-score. *Nucleic Acids Res.* 33, 2302–2309.

Zheng, W., Wuyun, Q., Li, Y., et al. 2019. Detecting distant-homology protein structures by aligning deep neural-network based contact maps. *PLoS Comput. Biol.* 15, e1007411.

Zhu, J., Wang, S., Bu, D., et al. 2018. Protein threading using residue co-variation and deep learning. *Bioinformatics*. 34, i263–i273.

Address correspondence to:
*Prof. Jinbo Xu*
*Toyota Technological Institute*
*Chicago, IL 60637*
*USA*

*E-mail:* jinbo.xu@gmail.com

*Dr. Dongbo Bu*
*Key Lab of Intelligent Information Processing*
*Institute of Computing Technology*
*Chinese Academy of Sciences*
*Beijing 100190*
*China*

*E-mail:* dbu@ict.ac.cn