DLPAlign: A Deep Learning based Progressive Alignment Method for Multiple Protein Sequences

Mengmeng Kuang *
University of Hong Kong
mmkuang@connect.hku.hk

Yong Liu
Shanghai University
gonic@shu.edu.cn

Lufei Gao
Hong Kong Polytechnic University
gaolufei@hotmail.com

ABSTRACT

This paper proposed a novel and straightforward approach to improve the accuracy of progressive multiple protein sequence alignment method. We trained a decision-making model based on the convolutional neural networks and bi-directional long short term memory networks, and progressively aligned the input protein sequences by calculating different posterior probability matrices.

To evaluate this method, we have implemented a multiple sequence alignment tool called DLPAlign and compared its performance with eleven leading alignment methods on three empirical alignment benchmarks (BAliBASE, OXBench and SABMark). Our results show that DLPAlign can get the best total-column scores on the three benchmarks. When evaluated against the 711 low similarity families with average PID $\leq$ 30%, DLPAlign improved about 2.8% over the second-best MSA software. Besides, we compared the performance of DLPAlign and other alignment tools on a real-life application, namely protein secondary structure prediction on four protein sequences related to SARS-COV-2, and DLPAlign provides the best result in all cases.

CCS CONCEPTS

• **Applied computing** → **Molecular sequence analysis**.

KEYWORDS

Multiple sequence alignment, decision-making model, progressive strategy, posterior probability matrix

ACM Reference Format:
Mengmeng Kuang, Yong Liu, and Lufei Gao. 2020. DLPAlign: A Deep Learning based Progressive Alignment Method for Multiple Protein Sequences. In *CSBio '20: Proceedings of the Eleventh International Conference on Computational Systems-Biology and Bioinformatics (CSBio2020), November 19–21, 2020, Bangkok, Thailand.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3429210.3429221

1 INTRODUCTION

Multiple Sequence Alignment (MSA) can be applied in many cases, such as recovering the history or relationship between protein or amino acid sequences and digging out some structural or functional roles of the sequences [34]. More and more biological modeling methods rely on the assembly of precise MSAs [5], [57]. It is of considerable significance to construct an algorithm to assist the MSA construction.

An MSA can be seen as a table constructed from protein sequences with an appropriate amount of spaces inserted [8], which can be defined as a mathematical problem. Given n sequences $S_i, i = 1, 2, \cdots, n$ as Equation (1),

$$S := \begin{cases} S_1 = (S_{11}, S_{12}, \cdots, S_{1m_1}) \\ S_2 = (S_{21}, S_{22}, \cdots, S_{2m_2}) \\ \vdots \\ S_n = (S_{n1}, S_{n2}, \cdots, S_{nm_n}) \end{cases} \tag{1}$$

an MSA is constructed from this set of sequences by inserting an appropriate amount of gaps needed into each of the S_i sequences of S until the modified sequences, S'_i, all conform to a same length l and no values in the sequences of S of the same column m, consists of only gaps. The mathematical form of an MSA of the above sequence set is shown at Equation (2):

$$S' := \begin{cases} S'_1 = \left(S'_{11}, S'_{12}, \cdots, S'_{1l}\right) \\ S'_2 = \left(S'_{21}, S'_{22}, \cdots, S'_{2l}\right) \\ \vdots \\ S'_n = \left(S'_{n1}, S'_{n2}, \cdots, S'_{nl}\right) \end{cases} \tag{2}$$

Since the early 1980s, the MSA construction problem has been solved by some algorithm-centric approaches [5]: Design algorithms to find the alignment with the most massive sum of column scores. Many excellent algorithms are well applied, such as dynamic programming [23], divide and conquer algorithm [46] and so on. Besides, in the past few decades, many alignment strategies have been proposed, such as progressive strategy [13], non-progressive strategy, consistency-based method [36], iterative refinement [15] etc. The progressive strategy is one of the maturest MSA strategies with large amounts of research validation and highest accuracy from [13]. A typical MSA method using progressive strategy mainly includes five parts: (1) posterior probability matrix calculation [27], (2) distance matrix calculation [4], (3)"Guide Tree" [32] generation by clustering methods, (4) consistency transformation [37], and (5) refinement [53]. In past studies, the main research focus is on the calculation method of the posterior probability matrix, the generation method of the guide tree, and the consistency transformation method, which could be seen from the most popular MSA tools adopting a progressive strategy in the past 10 years, shown as follows: (1) ProbCons [9] uses a pair-hidden Markov model (HMM)

*This is the corresponding author.

CSBio2020, November 19–21, 2020, Bangkok, Thailand

ACM ISBN 978-1-4503-8823-8/20/11…$15.00
https://doi.org/10.1145/3429210.3429221

to calculate the posterior probability matrix, an unweighted probabilistic consistency transformation, using an unweighted pair group method with the arithmetic mean (UPGMA) [45] hierarchical clustering method to generate a guide tree and iterative refinement to construct an MSA. (2) Probalign [39], another popular, highly accurate MSA tool, uses a partition function instead of the ProbCons pair HMM to calculate the posterior probability matrix. (3) MSAProbs [24] combines (1) and (2), using the Root Mean Square (RMS) of pair HMM and the partition function as the calculation method for the posterior probability matrix, and adopts a weighted consistency transformation. (4) GLProbs [55] introduced random HMM, and adaptively uses (i) the partition function, (ii) global pair HMM, (iii) the RMS of global pair HMM and random HMM to calculate the posterior probability matrix using the different average pairwise percentage identity (PID) of each protein family. PID stands for the percentage of the number of homologous positions in the pairwise alignment of two sequences. (5) PnpProbs [56] applies UPGMA and the weighted pair group method with arithmetical mean (WPGMA) [44] adaptively to generate the guide tree in its progressive branch.

Although these MSA tools can achieve relatively high accuracy on the whole, when it comes to a specific protein family, the accuracy of the different tools is often different [33]. Most importantly, after so many years of research, the algorithm-centric method has not significantly improved accuracy. As for the progressive alignment strategy, protein families with low similarity have always been the most challenging part [49].

This paper explores a different approach which adopts classifiers trained from data [41], to tackle the MSA construction problem to improve the quality of MSAs, especially the quality on low similarity protein families. We first determined which specific part in the progressive MSA methods could provide the greatest improvement in accuracy. After that, we transformed the classification of MSA families into the classification of pairwise sequences since pairwise alignments were the smallest building blocks of MSAs [21], and in this process, we obtained large-scale training data (there were 954854 such training data). Deep learning methods were used to train decision-making models to select the most appropriate calculation method of the specific part. We give more details of the decision-making models and the implementation in Sections 2.2 and 3.2. Further, based on the most accurate decision-making model, we build a new progressive MSA tool called DLPAlign.

We compared DLPAlign with eleven popular MSA tools on three empirical benchmarks in Section 4. Tables 6, 7 and 8 (in Section 4) demonstrate the alignment accuracies of the MSAs constructed by the tools for families in BAliBASE [48], OXBench [38] and SABMark [52], respectively. Figure 3(a) compares the average TC-score on all families from the three benchmarks between DLPAlign and other eleven popular MSA tools. DLPAlign achieved the highest TC-scores among all the tools on all benchmarks. Note that DLPAlign got better performance on the low or medium similarity protein families (extracted from OXBench, BAliBASE and SABMark benchmarks) that other progressive methods were not good at, which was shown at Figures 3(b) and 3(c).

We think this tool can be used in actual MSA task, so we upload the source code as well as the benchmarks for testing to GitHub (https://github.com/kuangmeng/DLPAlign).

2 METHODS

In this section, we first explain how to determine which part is used to improve the accuracy, then consider which data to use as the training data, and finally introduce the decision-making process of our deep learning methods.

2.1 How to select the best promotion part?

As we mentioned in Section 1, a typical progressive alignment method consists of five main parts: (1) posterior probability matrix calculation, (2) distance matrix calculation, (3) "Guide Tree" generation by clustering methods, (4) consistency transformation and (5) refinement. The most studied are Parts (1), (3) and (4) which are chosen as candidate promotion parts. We refer to the posterior probability matrix calculation as **Part A**, guide tree generation as **Part B**, and consistency transformation as **Part C**. For each part, we extracted several candidate options from previous studies [9], [39], [24], [55], [56], as shown below:

Options for Part A:

(1) Pair-HMM
(2) Partition function
(3) the RMS of pair-HMM and partition function
(4) the RMS of pair-HMM, partition function and random HMM

Options for Part B:

(1) UPGMA
(2) WPGMA

Options for Part C:

(1) Unweighted consistency transformation
(2) Weighted consistency transformation

The critical concern is the upper bounds of the various calculations in different parts of the progressive alignment strategy, and in which part the maximum improvement can be made. For the i-th option of Part A, we implemented a pipeline $\mathcal{P}_A^i$ by implementing the same default methods in other parts. We used the calculation method in each part numbered (1) as the default method for that part. We obtained pipelines $\mathcal{P}_B^i$ and $\mathcal{P}_C^i$ in the same way. we report the results of these pipelines in Section 3.1.

2.2 How to train decision-making models?

In **Part X** of the progressive alignment strategy, for a specific protein family, $\mathcal{F}$, choosing the method with the highest accuracy on which to construct an MSA $\mathcal{M}$ can be expressed as the classification problem $C_{\mathcal{P}_X}^{Aln}$. Classification $C_{\mathcal{P}_X}^{Aln}$ of a protein family is defined as follows:

> $C_{\mathcal{P}_X}^{Aln}$ has n classes $\mathcal{P}_X^1$, $\mathcal{P}_X^2$ ..... $\mathcal{P}_X^n$. A protein family $\mathcal{F}$ is in class $\mathcal{P}_X^i$ if the MSA constructed by the pipeline $\mathcal{P}_X^i$ could get better TC-score than those constructed by others, where n is the number of options in Part X and i is a positive integer not greater than n.

2.2.1 Data augmentation. In the past few years, there have been significant developments in deep learning, which have been applied in Bioinformatics [29], [12], mainly due to the continuous expansion of the data scale. Except BAliBASE, OXBench and SABMark benchmarks, we also gained protein sequences data from SISYPHUS [1], SABmark [52], a part of the extension set of BAliBASE namely

BAliBASE-X, HOMSTRAD [30], and Mattbench [6] which were used for training the classifier.

Table 1 summarizes, for each dataset, the number of families and the total number of sequences.

	Num. of families	Total num. of sequences
BAliBASE	386	11082
BAliBASE-X	147	6031
OXBench	395	3292
SABmark	423	2418
SISYPHUS	126	1772
HOMSTRAD	1030	3454
Mattbench	259	1698
Total	2766	29747

Table 1: The number of families and the total number of sequences in each dataset.

We considered coupling any two sequences in the same protein family as an independent piece of data. We do this is that MSA can be disassembled into multiple pairwise alignments. If a family has n sequences, we can get $C_n^2 = \frac{n\times(n-1)}{2}$ sequence pairs from it. In this way, our data was expanded to 954,854 pairs.

2.2.2 The structure of candidate deep learning models. Because the length of the sequence pairs was not very consistent, we normalized the length before choosing the neural networks. We unified all pairs into a fixed length α (α = 512 in our structure, we choose 512 because there are more than 80% data shorter than 512 and 65% data shorter than 256). When the length of a pair was insufficient, it was filled by gaps at the end to increase the length to the α. When the length of the sequence exceeded α, only the leading fragments of length α were intercepted. When we regard each character as a single word, if we convert it into a one-hot word vector, the size of the vector is a little large, so we first used the word-embedding [20] technique to convert each word into a small size (eight-dimensional vector). Even so, the input scale was still relatively large, so we applied convolutional neural networks (CNNs) [25], which had made a significant breakthrough in computer vision to reduce the dimensionality of the data while retaining its characteristics, as the first two layers of our models. There were order relationships between every character in a protein sequence pair, so we added a recurrent neural network (RNN) [31] layer after the CNNs. The improved versions of the recurrent neural network, long short term memory network (LSTM) [28] and gated recurrent unit network (GRU) [7], and their bi-directional versions (BiLSTM, BiGRU) have many advantages, so they were alternatives. Subsequently, two full connection layers were connected. To reduce overfitting, we added a specific dropout rate to the first full connection layer. This kind of neural network structure is very suitable and widely used for classification tasks [2], [17], [22], [58], [47].

Section 3.2 gives the implementation of different deep learning models, as well as the training and testing processes. The final decision-making model was determined according to the accuracy of different models.

3 IMPLEMENTATION

To test the effectiveness of our methods and select the best promotion part, we have implemented pipelines $\mathcal{P}_A$, $\mathcal{P}_B$ and $\mathcal{P}_C$ by adopting different calculation methods in Part A, Part B and Part C of progressive alignment strategy. In the meantime, different deep learning models were trained for choosing the best decision-making model for the best promotion part.

3.1 Find the best promotion part.

To evaluate the advantages and disadvantages of several methods in Part A and to what extent they could be improved, we got four different pipelines by using different calculation methods of the posterior probability matrix in the GLProbs' code and implemented the calculation in Parts B and C by default, naming them $\mathcal{P}_A^i, i = 1, 2, 3, 4$, which respectively represents the different options of Part A mentioned above. We implemented $\mathcal{P}_B^i, i = 1, 2$, where i denoted the different clustering methods for guide tree generation in Part B, and $\mathcal{P}_C^i, i = 1, 2$, where i denoted the different calculation of consistency transformation in Part C in the same way.

To measure the accuracies of MSAs constructed by different pipelines, the total-column score (TC-score), which was first introduced in BAliBASE [50], is the most popular measurement in many alignment benchmark tests. TC-score represents the percentage of the correctly aligned columns in alignments comparing with the references. Qscore (http://www.drive5.com/qscore) is an essential tool for analyzing the quality of MSAs in this paper. We chose the famous BAliBASE, OXBench, and SABmark benchmarks as the evaluation materials.

Table 2 summarizes for each pipeline $\mathcal{P}_A^i$, $\mathcal{P}_B^i$ or $\mathcal{P}_C^i$ and each benchmark database the average TC-scores of the alignments constructed by the pipelines for the families in the database.

Table 2 shows that if a particular decision is used in Part A to assist in selecting different calculation methods, the theoretical maximum promotion proportion can be obtained. So next, we chose the right decision-making method for pipeline $\mathcal{P}_A$.

3.2 Determine the best decision-making model.

We implemented the neural network structures mentioned in Section 2.2.2 and named them CNN, CNN-RNN, CNN-LSTM, CNN-BiLSTM, CNN-GRU and CNN-BiGRU, according to the different recurrent neural network layers used.

We divided the collected pairs data into two subsets: (1) 80% was randomly selected for model training, and (2) the remaining 20% was used for final testing. In the training process, a five-fold cross-validation was performed. This kind of validation method proved to be the most efficient [19]. In the process of training, we also set early stopping to further reduce overfitting [54].

Table 3 reveals the macro average (averaging the unweighted mean per label) [51] of the precision, recall and f_1-score of the four labels in the 20% test data. If $\mathcal{P}r_i$ is the precision and $\mathcal{R}_i$ is the recall rate of class i, where i = 1, 2, 3 or 4 in this paper, the macro

[1]Upper Bound: The highest TC-score could be obtained when each family chooses the pipeline which could get the best TC-score in this part.
[2]Max. Improvement: The proportion that the upper bound can improve compared with the best-existed result of this part.

	BAliBASE	OXBench	SABMark
$\mathcal{P}_A^1$	62.17	80.95	39.22
$\mathcal{P}_A^2$	61.06	81.73	39.00
$\mathcal{P}_A^3$	64.42	81.57	40.11
$\mathcal{P}_A^4$	64.27	81.56	40.62
Upper Bound [1]	66.50	82.93	42.84
Max. Improvement [2]	**3.23%**	**1.47%**	**5.47%**
$\mathcal{P}_B^1$	62.17	80.95	39.22
$\mathcal{P}_B^2$	62.42	80.93	39.20
Upper Bound	62.74	80.96	39.28
Max. Improvement	0.51%	0.01%	0.15%
$\mathcal{P}_C^1$	62.17	80.95	39.22
$\mathcal{P}_C^2$	62.95	81.00	39.09
Upper Bound	63.65	81.21	39.69
Max. Improvement	1.11%	0.26%	1.20%

Table 2: Average TC-score of each tool on the three empirical benchmarks

average precision ($\mathcal{P}r_{macro}$) and recall ($\mathcal{R}_{macro}$) can be calculated by Formula (3).

$$\begin{aligned} \mathcal{P}r_{macro} &= \frac{1}{N}\sum_{i=1}^{N} Pr_i \\ \mathcal{R}_{\text{macro}} &= \frac{1}{N}\sum_{i=1}^{N} R_i \end{aligned} \tag{3}$$

where N = 4, which means the category number is 4 in this paper.

The macro average f_1-score ($\mathcal{F}_{macro}$) equals to the harmonic average of $\mathcal{P}r_{macro}$ and $\mathcal{R}_{macro}$. According to the f_1-score in Table 3, we decided to use CNN-BiLSTM as our decision-making model.

Models	$\mathcal{P}r_{macro}$	$\mathcal{R}_{macro}$	$\mathcal{F}_{macro}$
CNN	87.13	86.69	86.91
CNN-RNN	86.90	88.13	87.51
CNN-LSTM	87.13	87.71	87.41
CNN-BiLSTM	87.70	88.68	**88.19**
CNN-GRU	87.93	86.71	87.32
CNN-BiGRU	88.23	87.89	88.06

Table 3: The macro average precision, recall and f_1-score on the test data

Table 4 illustrates, in more detail, the precision, recall and f_1-score in the four different categories of the CNN-BiLSTM model we finally selected. Although there is a difference in the precision or recall rate among the different categories, the f_1-scores of each category is generally good; they were all over 85%, indicating that our model can handle all categories well.

The structure and some details of the model we used are shown in Figure 1.

Category	Precision	Recall	F_1-score
$\mathcal{P}_A^1$	82.97	96.90	89.39
$\mathcal{P}_A^2$	79.46	92.83	85.62
$\mathcal{P}_A^3$	95.44	77.81	85.73
$\mathcal{P}_A^4$	92.93	87.18	89.96
Macro Avg.	87.70	88.68	88.19

Table 4: The precision, recall and f_1-score on the test data in different categories

4 RESULTS

Given the high accuracy of our decision-making model (CNN-BiLSTM) on the “posterior probability matrix calculation" part, we integrated it into the existed high-accuracy progressive alignment method GLProbs, to construct a new alignment tool, which we named DLPAlign.

The “posterior probability matrix calculation" part in DLPAlign is shown in Figure 2. Each pair x, y in $\mathcal{F}$ is inputed to the decision-making model to get a label (say $label_{x,y}$). These labels represent the specific calculation method of posterior probability matrix that should be used. Which calculation method is chosen for the protein family $\mathcal{F}$ is determined by the dominate proportion of labels that all of its pairs get after passing through the decision-making model. Because we already know the percentage of each correct label of $\mathcal{C}^{Aln}_{\mathcal{P}^i_A}$, where i = 1, 2, 3 or 4, the dominate proportion of predicted labels can be calculated using Formula (4).

$$\text{dominate\_proportion} = \text{argmax}_i \left(\frac{PLP_i}{TLP_i}\right) \tag{4}$$

where PLP_i means the percentage of the i^{th} predicted label, TLP_i means the proportion of the i^{th} true label and i = 1, 2, 3 or 4.

The proportion of the i^{th} true label is shown as Table 5.

TLP_1	TLP_2	TLP_3	TLP_4
21.94%	16.28%	21.31%	40.47%

Table 5: The proportion of the i^{th} true label.

Depending on the final family category, we use (1) the pair HMM, (2) the partition function, (3) the RMS of pair HMM and the partition function or (4) the RMS of pair HMM, the partition function and random HMM to accomplish the calculation of the posterior probability matrix.

4.1 Comparing the accuracy of DLPAlign with other MSA tools

To determine the accuracy of DLPAlign implemented by the CNN-BiLSTM decision-making model and comparing it with other MSA tools, three empirical benchmarks were selected - BAliBASE 3.0, OXBench 1.3 and SABmark 1.65 - and the newest versions of eleven

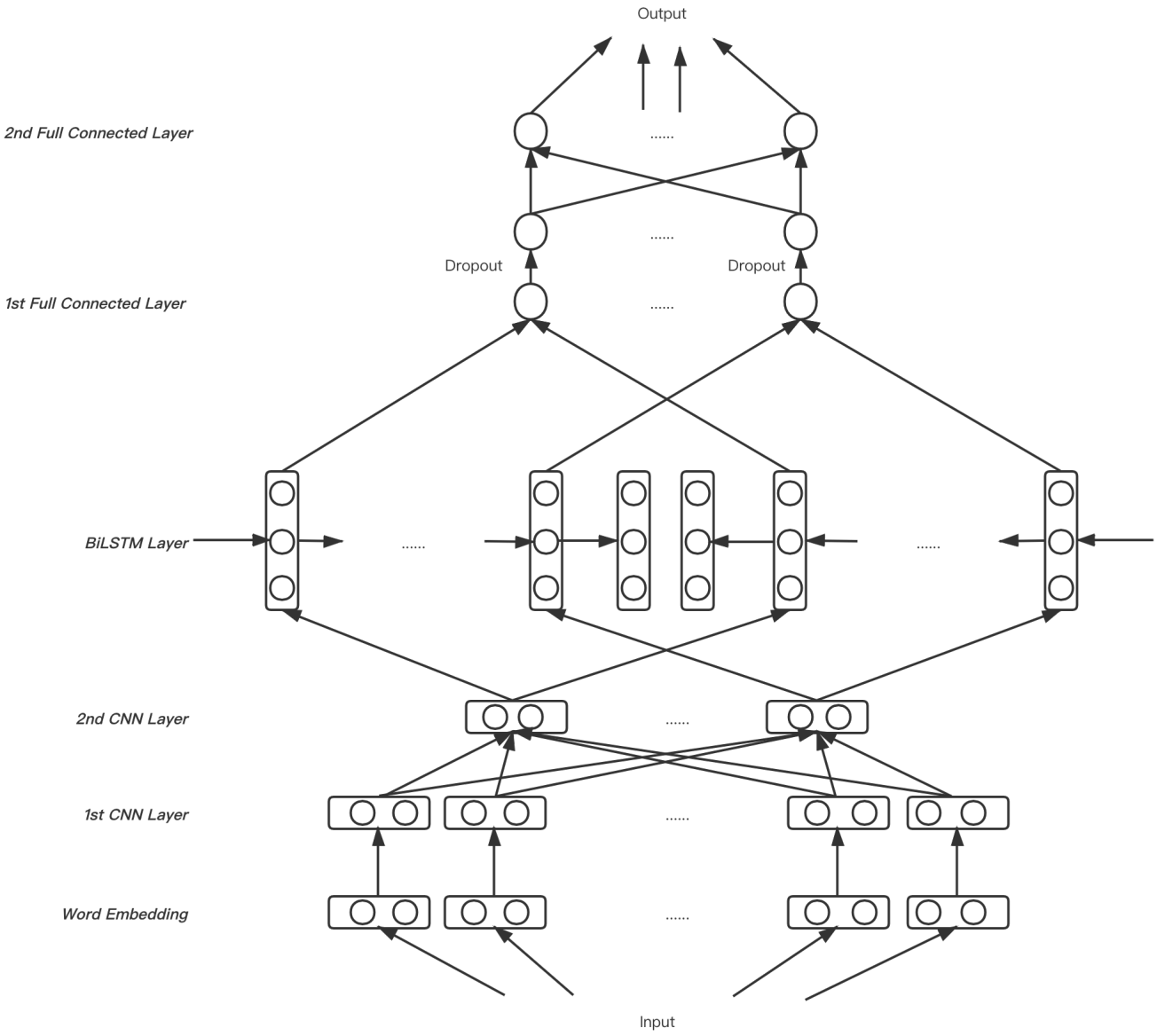


Figure 1: The neural network structure of our decision-making model. ***Firstly the input is transformed through the Word embedding layer into a 512 × 8 matrix. Then the matrix passes through two CNN layers with filter sizes of 6 and 3 respectively (each CNN layer is followed by a max-pooling layer of size 2). Next, the output of the previous layer goes through the Bi-directional LSTM layer with a hidden size of 64. Finally, two full connection layers are connected and a 0.5 dropout to the first full connection layer is set.***

popular MSA tools were chosen for comparison: QuickProbs [16], PnpProbs, GLProbs, MSAProbs, Probalign, ProbCons, PicXAA [40], MAFFT [18], MUSCLE [11], ClustalΩ [42] and T-Coffee [35]. Of these eleven MSA tools, PicXAA adopted the non-progressive strategy, PnpProbs used both the non-progressive strategy and the progressive strategy, and the others used the progressive strategy. The TC-score mentioned in Section 3.1 was the leading indicator in the comparison.

Figure 3(a) shows the average TC-scores on all families from BAliBASE, OXBench and SABMark of DLPAlign as well as eleven MSA tools. DLPAlign can get about 1.55% improvement over second-best result. Figures 3(b) and 3(c) compare the average TC-scores

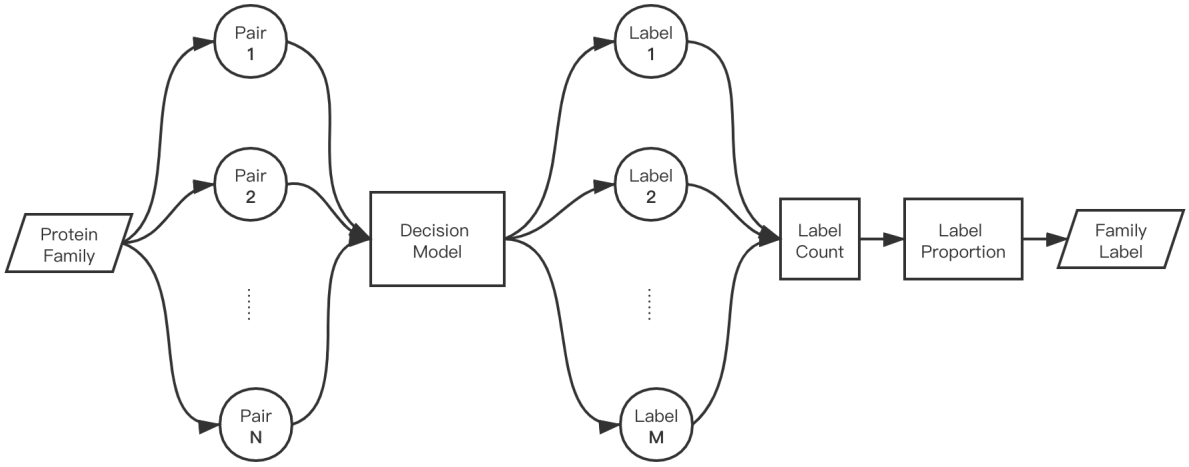


Figure 2: The process of splitting a protein family into pairs, using decision-making model to determine the label of each pair, and finally calculating the dominate proportion of the labels to get the label of the protein family.

on low similarity (with PID ≤ 30% and there are totally 711 such families) and medium similarity (with 30% < PID ≤ 60% and there are totally 352 such families) families in BAliBASE, OXBench and SABmark. It can be seen from the figures that the improvement of DLPAlign is pronounced, especially in the low similarity families set which can be improved by 2.8%.

	All (386)	RV11 (38)	RV12 (44)
DLPAlign	**65.47**	**47.67**	**87.32**
QuickProbs	**65.41**	**46.93**	87.03
PnpProbs	62.46	45.15	87.25
GLProbs	62.09	44.68	**87.38**
MSAProbs	64.51	44.40	87.03
ProbCons	61.89	40.89	84.14
PicXAA	59.97	46.64	86.60
MAFFT	50.08	28.23	75.57
Muscle	53.17	32.06	58.90
ClustalΩ	56.20	36.22	79.38
ProbAlign	60.68	45.69	86.69
T-Coffee	59.16	39.27	84.16
Improved %	0.09	1.58	-0.07

Table 6: Average TC-scores for BAliBASE

The alignments in BAliBASE were organized into reference sets that were designed to represent real multiple alignment problems. Table 6 shows the average TC-score of the whole benchmark with 386 families and the accuracy of its two divergent reference sets (say, RV11 and RV12). DLPAlign could also handle RV11, which is a very divergent subset, obtaining a 1.58% higher TC-score than the second-best MSA tool.

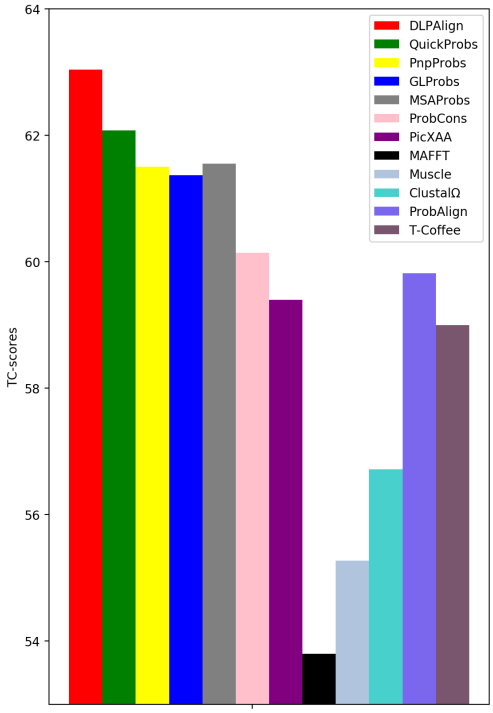
Table 7 shows the results of DLPAlign, as well as other MSA tools, for OXBench. In addition to the complete set of families, the table shows the alignment accuracy for families with average PID of less than and more than 30%.

	All (395)	0 - 30% (63)	30% - 100% (332)
DLPAlign	**82.52**	**44.16**	**89.80**
QuickProbs	81.77	41.50	88.35
PnpProbs	**82.06**	**43.96**	89.54
GLProbs	81.93	43.34	**89.55**
MSAProbs	81.50	42.81	89.08
ProbCons	80.68	41.50	88.35
PicXAA	81.14	39.78	89.32
MAFFT	78.15	35.93	86.40
Muscle	80.67	40.95	88.21
ClustalΩ	79.99	37.39	88.08
ProbAlign	81.68	41.06	89.39
T-Coffee	80.18	40.04	87.80
Improved %	0.56	0.45	0.28

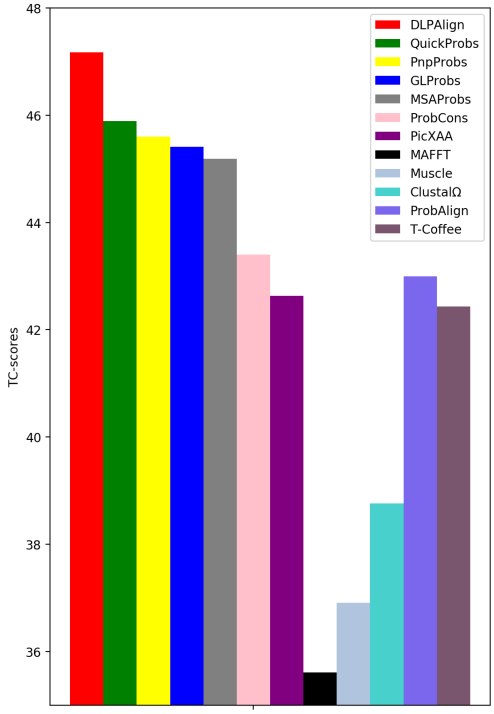
Table 7: Average TC-scores for OXBench

Note that OXBench did not divide the whole database into different subsets. We made the division here because we thought the two parts after separation could respectively represent divergent families and high similarity families. It can be seen that no matter which divergent set or high similarity set is used, DLPAlign can always produce some improvement.

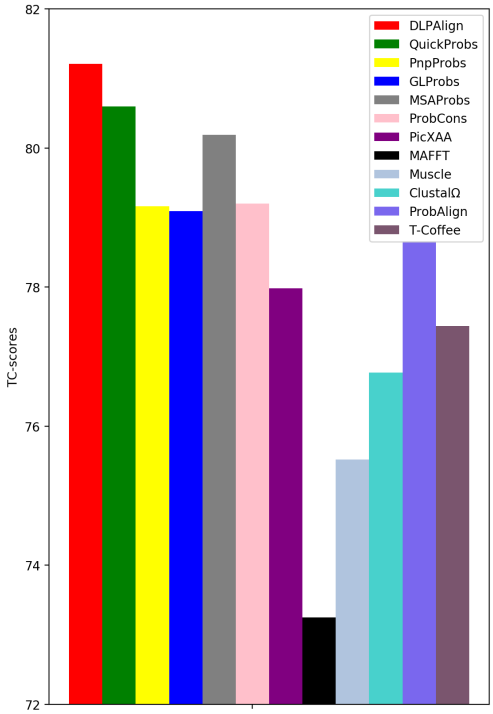
Table 8 summaries the accuracy for SABMark benchmark, which was divided into two subset Twilight Zone and Superfamilies, depending on the SCOP classification [26]. These subsets together covered the entire known fold space using sequences with very low to low, and low to intermediate similarity, respectively. DLPAlign improved both subsets and the whole benchmark. For Twilight



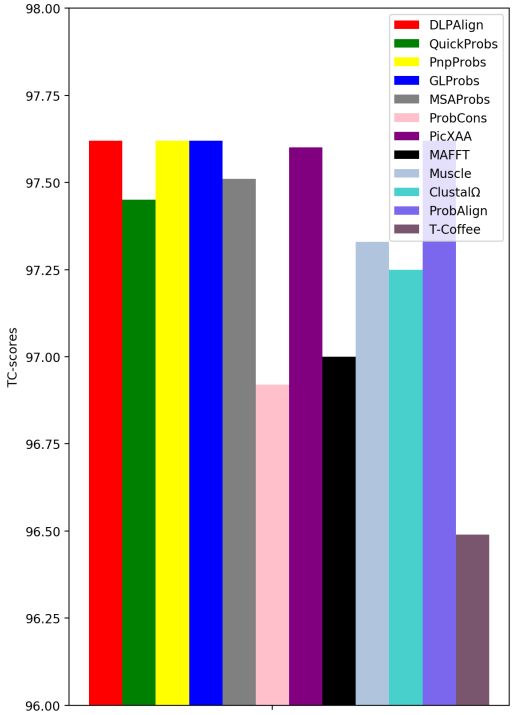
(a) On all 1204 families of BAliBASE, OXBench and SABmark, DLPAlign gets the best average TC-score which is 63.04 and improves about 1.55% over the second-best MSA tool QuickProbs.



(b) Over 711 families in BAliBASE, OXBench and SABmark with PID $\leq$ 30%, DLPAlign gets the best average TC-score 47.17 and improves about 2.8% over the second best one which is 45.89.



(c) Over 352 families in BAliBASE, OXBench and SABmark with PID between 30% and 60%, DLPAlign gets the best average TC-score 81.21 and improves about 0.8% over the second best one which is 80.60.



(d) Over 141 families in BAliBASE, OXBench and SABmark with PID > 60%, all the MSA tools could achieve more than 96.80 on the average TC-score.

Figure 3: Comparison of average TC-scores over different similarity protein families in BAliBASE, OXBench and SABmark

	All (423)	Superfamily (315)	Twilight Zone (108)
DLPAlign	**42.59**	**48.37**	**25.71**
QuickProbs	40.65	46.56	23.40
PnpProbs	**41.48**	**47.03**	**24.63**
GLProbs	41.22	46.95	23.86
MSAProbs	40.04	45.81	22.60
ProbCons	39.17	44.64	22.57
PicXAA	38.44	44.45	20.33
MAFFT	33.00	39.10	14.72
Muscle	33.47	39.13	16.96
ClustalΩ	35.47	41.43	18.10
ProbAlign	38.63	44.11	22.64
T-Coffee	39.07	44.15	24.25
Improved %	2.68	2.85	4.38

Table 8: Average TC-scores for SABMark

	Average running time (sec.)		
	BAliBASE	OXBench	SABMark
DLPAlign	38.10	0.84	0.40
QuickProbs	6.10	0.11	0.06
PnpProbs	13.30	0.27	0.21
GLProbs	13.63	0.23	0.15
MSAProbs	9.49	0.14	0.07
ProbCons	33.98	0.49	0.24
PicXAA	29.82	0.49	0.35
MAFFT	0.33	0.15	0.14
Muscle	1.69	0.04	0.08
ClustalΩ	0.97	0.03	0.04
ProbAlign	21.92	0.27	0.12
T-Coffee	339.6355	6.1013	2.8393

Table 9: Average running time (in seconds) of three benchmarks by DLPAlign and other MSA tools

Zone, except for DLPAlign, none of the MSA tools could get a TC-score of more than 25%. In this subset, DLPAlign's TC-score was 4.38% higher than the second-best MSA tool, PnpProbs.

4.2 Comparing the efficiency of DLPAlign with other MSA tools

All the tools were run on an HP desktop computer with four Intel Cores i5-3570 (3.40 GHz) and a main memory of size 19.4 GB. Table 9 shows the efficiency results.

It should be noted that ProbCons, Probalign, MSAProbs, GLProbs and PnpProbs all used the standard progressive alignment steps mentioned in Section 1, so there was not much difference in their running times. The running time of DLPAlign comprised mainly the running time of the decision-making model and the standard progressive alignment. As Table 9 shows, the time taken by the decision-making model was affected by the benchmark size (BAliBASE was largest benchmark while SABMark was the smallest), but in general, the time was acceptable.

4.3 A real-life application: Protein secondary structure prediction

Protein secondary structure prediction is an appropriate application of multiple sequence alignment [43].

We picked some protein sequences related to SARS-COV-2 which could be found at Protein Data Bank (PDB) [3] with these PDB ID: 6YI3, 6VYO, 6W9C and 6W61 [1]. We then used the following process to evaluate the performance of DLPAlign with the other five highly accurate MSA tools.

> Given a protein sequence S, we use Jpred 4 [10] to search protein sequences similar to it. Then, we construct an MSA $\mathcal{M}$ for these sequences and S, and use the secondary structure prediction tool provided on Jpred 4 to predict the secondary structure of S. Finally, comparing the predictions with reference secondary structures offered by Jpred 4.

Table 10 summaries the number of wrongly aligned residues for each MSA tool. The table shows that DLPAlign always got the fewest wrong aligned number of the leading MSA tools.

5 CONCLUSION

The significant contributions of this paper are (i) using convolutional neural networks and bi-directional long short term networks to train a decision-making model to determine which specific calculation method to use in the posterior probability matrix calculation of progressive alignment approaches and (ii) releasing a new progressive multiple protein sequence alignment tool based on this deep learning model named DLPAlign. As hindsight, DLPAlign, based on the decision-making model, can get better accuracy on alignment of protein families compared with existed leading MSA methods and perform especially excellent on low similarity families.

We would also like to point out that we have not optimized DLPAlign for efficiency. The efficiency of DLPAlign slows down with the increase of the number of sequences, because the input of the decision-making model is sequence pair. The more sequences, the more sequence pairs will be obtained; thus, the number of times the decision-making model runs is increased.

One way to improve efficiency is to reduce the number of times the decision-making model runs. In this study, we regard each sequence pair as a sentence to classify. If we can take the combination of more than two sequences as the input of the decision-making model, then the whole decision-making model will run less, which will significantly improve the efficiency. It is also one of the improvements of DLPAlign in the future.

Besides, if we use the protein family simulation tool such as INDELible [14] to obtain enough protein family data, we can also consider the entire protein family or the temporary MSA built by the fast MSA tool as the training data of the decision-making model, so that the decision-making model will only run once, the efficiency can be further improved.

[1] They are all available only from April 2020, and relevant references have not been published.

PDB ID (Length)	DLPAlign	QuickProbs	PnpProbs	GLProbs	MSAProbs	PicXAA
6YI3 (140)	**10**	16	12	12	15	16
6VYO (128)	**5**	**5**	**5**	**5**	**5**	7
6W9C (317)	**20**	31	24	22	23	32
6W61 (299)	**13**	17	**13**	**13**	15	20

Table 10: Number of wrongly aligned residues in the predicted secondary structures of proteins

6 ACKNOWLEDGMENT

This research was guided by Dr. Hing-fung Ting of the University of Hong Kong. We thank Dr. Ting and Dr. Bin Yan from the University of Hong Kong for many helpful discussions and suggestions.

REFERENCES

[1] Antonina Andreeva, Andreas Prlić, Tim JP Hubbard, and Alexey G Murzin. 2007. SISYPHUS—structural alignments for proteins with non-trivial relationships. *Nucleic acids research* 35, suppl_1 (2007), D253–D259.

[2] Soo Hyun Bae, Inkyu Choi, and Nam Soo Kim. 2016. Acoustic scene classification using parallel combination of LSTM and CNN. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016)*. 11–15.

[3] Protein Data Bank. 1971. Protein data bank. *Nature New Biol* 233 (1971), 223.

[4] Felix Bast. 2013. Sequence similarity search, multiple sequence alignment, model selection, distance matrix and phylogeny reconstruction. *Nature Protocol Exchange* (2013).

[5] Maria Chatzou, Cedrik Magis, Jia-Ming Chang, Carsten Kemena, Giovanni Bussotti, Ionas Erb, and Cedric Notredame. 2016. Multiple sequence alignment modeling: methods and applications. *Briefings in bioinformatics* 17, 6 (2016), 1009–1023.

[6] Noah Daniels, Anoop Kumar, Lenore Cowen, and Matt Menke. 2011. Touring protein space with Matt. *IEEE/ACM transactions on computational biology and bioinformatics* 9, 1 (2011), 286–293.

[7] Rahul Dey and Fathi M Salemt. 2017. Gate-variants of gated recurrent unit (GRU) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. IEEE, 1597–1600.

[8] Chuong B Do and Kazutaka Katoh. 2009. Protein multiple sequence alignment. In *Functional Proteomics*. Springer, 379–413.

[9] Chuong B Do, Mahathi SP Mahabhashyam, Michael Brudno, and Serafim Batzoglou. 2005. ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome research* 15, 2 (2005), 330–340.

[10] Alexey Drozdetskiy, Christian Cole, James Procter, and Geoffrey J Barton. 2015. JPred4: a protein secondary structure prediction server. *Nucleic acids research* 43, W1 (2015), W389–W394.

[11] Robert C Edgar. 2004. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research* 32, 5 (2004), 1792–1797.

[12] Worrawat Engchuan and Jonathan H Chan. 2015. Pathway activity transformation for multi-class classification of lung cancer datasets. *Neurocomputing* 165 (2015), 81–89.

[13] Da-Fei Feng and Russell F Doolittle. 1987. Progressive sequence alignment as a prerequisitetto correct phylogenetic trees. *Journal of molecular evolution* 25, 4 (1987), 351–360.

[14] William Fletcher and Ziheng Yang. 2009. INDELible: a flexible simulator of biological sequence evolution. *Molecular biology and evolution* 26, 8 (2009), 1879–1888.

[15] Osamu Gotoh. 1996. Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *Journal of molecular biology* 264, 4 (1996), 823–838.

[16] Adam Gudyś and Sebastian Deorowicz. 2017. QuickProbs 2: towards rapid construction of high-quality alignments of large protein families. *Scientific reports* 7, 1 (2017), 1–12.

[17] Chinnappa Guggilla, Tristan Miller, and Iryna Gurevych. 2016. CNN-and LSTM-based claim classification in online user comments. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2740–2751.

[18] Kazutaka Katoh and Daron M Standley. 2013. MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Molecular biology and evolution* 30, 4 (2013), 772–780.

[19] Ron Kohavi et al. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, Vol. 14. Montreal, Canada, 1137–1145.

[20] Siwei Lai, Kang Liu, Shizhu He, and Jun Zhao. 2016. How to generate a good word embedding. *IEEE Intelligent Systems* 31, 6 (2016), 5–14.

[21] Giddy Landan and Dan Graur. 2009. Characterization of pairwise and multiple sequence alignment errors. *Gene* 441, 1-2 (2009), 141–147.

[22] Qingshan Liu, Feng Zhou, Renlong Hang, and Xiaotong Yuan. 2017. Bidirectional-convolutional LSTM based spectral-spatial feature learning for hyperspectral image classification. *Remote Sensing* 9, 12 (2017), 1330.

[23] Weiguo Liu, Bertil Schmidt, Gerrit Voss, and Wolfgang Muller-Wittig. 2007. Streaming algorithms for biological sequence alignment on GPUs. *IEEE transactions on parallel and distributed systems* 18, 9 (2007), 1270–1281.

[24] Yongchao Liu, Bertil Schmidt, and Douglas L Maskell. 2010. MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities. *Bioinformatics* 26, 16 (2010), 1958–1964.

[25] Shih-Chung B Lo, Heang-Ping Chan, Jyh-Shyan Lin, Huai Li, Matthew T Freedman, and Seong K Mun. 1995. Artificial convolution neural network for medical image pattern recognition. *Neural networks* 8, 7-8 (1995), 1201–1214.

[26] Loredana Lo Conte, Bart Ailey, Tim JP Hubbard, Steven E Brenner, Alexey G Murzin, and Cyrus Chothia. 2000. SCOP: a structural classification of proteins database. *Nucleic acids research* 28, 1 (2000), 257–259.

[27] Ari Löytynoja and Michel C Milinkovitch. 2003. A hidden Markov model for progressive multiple alignment. *Bioinformatics* 19, 12 (2003), 1505–1513.

[28] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long short term memory networks for anomaly detection in time series. In *Proceedings*, Vol. 89. Presses universitaires de Louvain.

[29] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. 2017. Deep learning in bioinformatics. *Briefings in bioinformatics* 18, 5 (2017), 851–869.

[30] Kenji Mizuguchi, Charlotte M Deane, Tom L Blundell, and John P Overington. 1998. HOMSTRAD: a database of protein structure alignments for homologous families. *Protein science* 7, 11 (1998), 2469–2471.

[31] Lichao Mou, Pedram Ghamisi, and Xiao Xiang Zhu. 2017. Deep recurrent neural networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing* 55, 7 (2017), 3639–3655.

[32] S Nelesen, Kevin Liu, Donggao Zhao, C Randal Linder, and Tandy Warnow. 2008. The effect of the guide tree on multiple sequence alignments and subsequent phylogenetic analyses. In *Biocomputing 2008*. World Scientific, 25–36.

[33] Cédric Notredame. 2002. Recent progress in multiple sequence alignment: a survey. *Pharmacogenomics* 3, 1 (2002), 131–144.

[34] Cédric Notredame. 2007. Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput Biol* 3, 8 (2007), e123.

[35] Cédric Notredame, Desmond G Higgins, and Jaap Heringa. 2000. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology* 302, 1 (2000), 205–217.

[36] Jason S Papadopoulos and Richa Agarwala. 2007. COBALT: constraint-based alignment tool for multiple protein sequences. *Bioinformatics* 23, 9 (2007), 1073–1079.

[37] Benedict Paten, Javier Herrero, Kathryn Beal, and Ewan Birney. 2009. Sequence progressive alignment, a framework for practical large-scale probabilistic consistency alignment. *Bioinformatics* 25, 3 (2009), 295–301.

[38] GPS Raghava, Stephen MJ Searle, Patrick C Audley, Jonathan D Barber, and Geoffrey J Barton. 2003. OXBench: a benchmark for evaluation of protein multiple sequence alignment accuracy. *BMC bioinformatics* 4, 1 (2003), 47.

[39] Usman Roshan and Dennis R Livesay. 2006. Probalign: multiple sequence alignment using partition function posterior probabilities. *Bioinformatics* 22, 22 (2006), 2715–2721.

[40] Sayed Mohammad Ebrahim Sahraeian and Byung-Jun Yoon. 2010. PicXAA: greedy probabilistic construction of maximum expected accuracy alignment of multiple sequences. *Nucleic acids research* 38, 15 (2010), 4917–4928.

[41] Nigam H Shah and Jessica D Tenenbaum. 2012. FOCUS on translational bioinformatics: The coming age of data-driven medicine: translational bioinformatics' next frontier. *Journal of the American Medical Informatics Association: JAMIA* 19, e1 (2012), e2.

[42] Fabian Sievers and Desmond G Higgins. 2014. Clustal Omega, accurate alignment of very large numbers of sequences. In *Multiple sequence alignment methods*. Springer, 105–116.

[43] VA Simossis and J Heringa. 2004. Integrating protein secondary structure prediction and multiple sequence alignment. *Current Protein and Peptide Science* 5, 4

(2004), 249–266.
[44] PHA Sneath and RR Sokal. 1973. Numerical Taxonomy WH Freeman and Co. *San Francisco* (1973), 1–573.
[45] Peter HA Sneath and Robert R Sokal. 1962. Numerical taxonomy. *Nature* 193, 4818 (1962), 855–860.
[46] Jens Stoye, Vincent Moulton, and Andreas WM Dress. 1997. DCA: an efficient implementation of the divide-and-conquer approach to simultaneous multiple sequence alignment. *Bioinformatics* 13, 6 (1997), 625–626.
[47] Prissadang Suta, Pornchai Mongkolnam, and Wichai Eamsinvattana. 2015. Controlled and collaborative presentation via tablet PCs for classroom and meeting room uses. In *2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. IEEE, 102–107.
[48] Julie D Thompson, Patrice Koehl, Raymond Ripp, and Olivier Poch. 2005. BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins: Structure, Function, and Bioinformatics* 61, 1 (2005), 127–136.
[49] Julie D Thompson, Benjamin Linard, Odile Lecompte, and Olivier Poch. 2011. A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. *PloS one* 6, 3 (2011), e18093.
[50] Julie D. Thompson, Frédéric Plewniak, and Olivier Poch. 1999. BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics (Oxford, England)* 15, 1 (1999), 87–88.
[51] Vincent Van Asch. 2013. Macro-and micro-averaged evaluation measures [[basic draft]]. *Belgium: CLiPS* 49 (2013).
[52] Ivo Van Walle, Ignace Lasters, and Lode Wyns. 2004. SABmark—a benchmark for sequence alignment that covers the entire known fold space. *Bioinformatics* 21, 7 (2004), 1267–1268.
[53] MG Williams, H Shirai, J Shi, HG Nagendra, J Mueller, K Mizuguchi, RN Miguel, SC Lovell, CA Innis, CM Deane, et al. 2001. Sequence-structure homology recognition by iterative alignment refinement and comparative modeling. *Proteins: Structure, Function, and Bioinformatics* 45, S5 (2001), 92–97.
[54] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. 2007. On early stopping in gradient descent learning. *Constructive Approximation* 26, 2 (2007), 289–315.
[55] Yongtao Ye, David Wai-lok Cheung, Yadong Wang, Siu-Ming Yiu, Qing Zhang, Tak-Wah Lam, and Hing-Fung Ting. 2015. GLProbs: Aligning multiple sequences adaptively. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 12, 1 (2015), 67–78.
[56] Yongtao Ye, Tak-Wah Lam, and Hing-Fung Ting. 2016. PnpProbs: a better multiple sequence alignment tool by better handling of guide trees. *BMC bioinformatics* 17, 8 (2016), 285.
[57] Wenyu Zhang, Jiajia Chen, Yang Yang, Yifei Tang, Jing Shang, and Bairong Shen. 2011. A practical comparison of de novo genome assembly software tools for next-generation sequencing technologies. *PloS one* 6, 3 (2011), e17915.
[58] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630* (2015).