A Deep Bidirectional Long Short-Term Memory Approach Applied to the Protein Secondary Structure Prediction Problem

Leandro Takeshi Hattori*, César Manuel Vargas Benítez§, Heitor Silvério Lopes‡
Bioinformatics and Computational Intelligence Laboratory
Federal University of Technology-Paraná
Curitiba, Brazil
Emails: *lthattori@gmail.com, §cesarbenitez@utfpr.edu.br, ‡hslopes@utfpr.edu.br

***Abstract*—One of the most important open problems in science is the protein secondary structures prediction from the protein sequence of amino acids. This work presents an application of Deep Recurrent Neural Network with Bidirectional Long Short-Term Memory (DBLSTM) cells to this problem. We compare the performance of the proposed approach with the state-of-the-art approaches. Despite the lower complexity of the proposed approach (i.e. Neural Network architecture with fewer neurons), results showed that the DBLSTM could achieve a satisfactory level of accuracy when compared with the state-of-the-art approaches. We also studied the behavior of Gradient Optimizers applied to the DBLSTM. Furthermore, this paper concentrates on well-known quantitative analytical methods applied to evaluate the proposed approach.**

I. Introduction

Nowadays, one of the most important and open challenging problems in Bioinformatics is to obtain a better understanding of the protein folding process. In this process, every protein folds into a unique three-dimensional structure, that determines their specific biological function. The three-dimensional structure is also known as native conformation and it is a function of its secondary structures. A number of diseases, including cancer, Alzheimer's disease, cystic fibrosis, Huntington's disease and diabetes are linked to the result of the aggregation of ill-formed proteins [1], [2]. Notwithstanding, despite a large number of proteins that have been discovered recently (around 84.8 million records in the UniProtKB/TrEMBL repository as in may/2017), only a few of them have their structure known (129,745 in the Protein Data Bank – PDB as in may/2017). Therefore, acquiring knowledge about the structure of proteins is an important issue, since such knowledge can lead to important medical and biochemical advancements and even to the development of new drugs with specific functionality [3], [2]. Here, Computer Science has an important role proposing novel methods for studying the Protein Structure Prediction (PSP). A possible way to infer the full structure of an unknown protein is to determine secondary structures in it. However, the pattern formation rules of secondary structures of proteins are still not known precisely [3].

978-1-5386-3734-0/17/$31.00

In this sense, the Deep Learning (DL) methods have yielded significant results on Bioinformatics [4], [5], including to the protein secondary structures classification problems [6], [7]. Among DL approaches, the Long Short-Term Memory (LSTM) networks have excelled results in sequential/temporal problems. Mainly, because it is capable to find local and global patterns from sequences, fundamental characteristic to find different types of protein secondary structures [6].

The objective of this work is to classify secondary structures of proteins from their primary structure (i.e. linear sequence of amino acids). This is accomplished by using Deep Bidirectional Long Short-Term Memory Networks (DBLSTM).

This paper is organized as follows: Section II presents the background about protein structures and related works. Section III presents the protein datasets and the sequence encoding that were used. Also, Section IV describes in details the DBLSTM network. Furthermore, Section V shows the computational experiments and results. Finally, in Section VI some conclusions and future directions are pointed out.

II. Secondary Structure Classification

The secondary structures (SS) of a protein represent the local conformations of a three-dimensional structure. There are three main secondary structures: α-helices [8], β-sheets [9] and turns [10]. For instance, proteins 1DG2 and 1KFP represent an $\alpha-$helix and an $\beta-sheet$, respectively. Instead of using three classes, [11] group the secondary structures into 8 classes, including others, such as 3_{10}-helix, π-heliX and β-bridge.

As mentioned before, a step toward the three-dimensional structure protein description is the determination of the secondary structures. In this work, we assess the secondary structure classification from its primary structure. However, there is no consensus about the classification task which can be done using different properties of proteins. For instance, early approaches were based on stereochemical principles [12], statistics [13] and Position-specific Scoring Matrices (PSSM) [14]. More recently, [3] used the Kyte and Doolittle (K & D) hydrophobicity scale in order to represent the aminoacids of proteins.

From the approach point of view, [3] present the application and comparison of Machine Learning and Evolutionary Computation methods to define suitable classifiers for predicting the secondary structure of proteins, such as Gene expression programming (GEP), Sequential Minimal Optimization algorithm (SMO) and RandomForest (collection of tree-structured classifiers). [15] propose a new probabilistic method using Conditional Neural Fields (CNF) for protein 8-class secondary structure prediction. In addition, a Deep Convolutional Neural Fields (DCNF) approach applied to the same problem is presented by [7].

[16] present in details a Generative Stochastic Network (GSN) based approach for predicting local secondary structures and an extension with a Markov chain to sample from a conditional distribution. [17] proposed a template-based approach to enhance 8-class secondary structure prediction. [6] used a Bidirectional Recurrent Neural Network with Long Short-Term Memory cells for the classification task. [18] present the application of Cascaded Convolutional and Recurrent Neural Networks to predict secondary structures.

As mentioned in [15], compared with the protein 3-class secondary structure prediction, the 8-class prediction gains less attention and it is also much more challenging. Therefore, this work aims to classify secondary structures of proteins from their primary structure, considering the 8 classes proposed by [11].

III. Protein Datasets and Sequence Encoding

In this work, two datasets with 8 classes of secondary structure were used: the so-called CB6133 [19] and CB513 datasets [20]. The CB6133 has 6133 protein sequences (instances) and the CB513 has 513 instances which, in turn, were included in the CB6133 dataset. The CB6133 (excluded the CB513 intances) was used in the training (5278 instances) and validation (256 instances) tasks, as in [16]. On the other hand, the CB513 was used for testing the network with 513 instances.

The frequency of each class of secondary structure for both training and testing datasets is shown in Table I. Here, it is important to recall that the datasets are highly unbalanced.

TABLE I
Frequency Distribution

Freq. Train (%)	Freq. Test (%)	Class	
34.535	30.85	H	(α-helix)
21.781	21.25	E	(β-strand)
19.185	21.14	L	(loop)
11.284	11.81	T	(β-turn)
8.258	9.81	S	(bend)
3.911	3.69	G	(3_{10}-helix)
1.029	1.39	B	(β-bridge)
0,018	0.03	I	(π-helix)

The natural encoding of the protein sequence is a string of letters from the alphabet of letters representing the 20 proteinogenic amino acids. However, this encoding is not appropriate for some classification algorithms [3]. Thus, an encoding based on [18] and [14] was used in this work, as shown in Figure 1, where each amino acid (a_i) of the sequence (with n amino acids) has 42 features: 22 profile features are used for representing scores obtained from a Position-Specific Scoring Matrix (PSSM) [21], which are normalized using the logistic *Sigmoid* function. The last 20 features are used for representing the amino acid, using the *one-hot* encoding. In this encoding, only one bit is set and represents the name category of the amino acid.

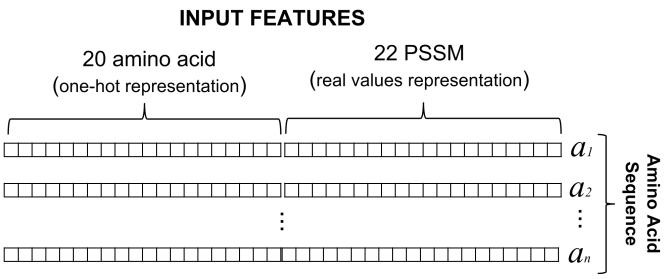

Fig. 1. The encoding scheme.

IV. *The Deep Bidirectional Long Short-Term Memory approach*

Recently, Deep-Learning (DL) gained attention with applications in several domains, such as speech recognition systems (the first major industrial application of DL), natural language understanding, sentiment analysis, language translation, image recognition, particle accelerator data analysis and Bioinformatics. Here, it is important to recall that it was possible with the advent of the Graphical Processing Units (GPUs). DL approaches are representation-learning methods with multiple levels of representation, from the raw input to a slightly more abstract level [22].

Basically, a DL architecture is composed by a stack of modules that are subject to learning with a large number of parameters. Moreover, it is known that large networks rarely present local minima. However, such networks are prone to overfitting. Also, training different architectures is very hard because finding an optimal set of hyper-parameters for each architecture is a difficult task and training each network is computationally expensive.

Dropout is a technique that assesses both problems and it can be interpreted as a way of regularizing a neural network by adding noise to its hidden units [23].

The training of deep networks was proved to be hard due to the variation of the backpropagated gradients at each time step, typically vanishing over several time steps [22]. Regarding this issue, [24] proposed a way to explore the use of rectifying non-linearities instead of using the well-known sigmoid and hyperbolic tangent functions, known as the Rectified Linear Units (ReLU), which is a better model of biological neurons and allows the network to obtain sparse representations. Nowadays, the ReLU is considered to be the most popular non-linear activation function. It is a half-wave rectifier $f(x) = max(x, 0)$ (where x is the input to a neuron). The advantage of the ReLU, compared with other logistic

functions, is that it learns much faster in multi-layer networks, allowing training of a deep supervised network [22].

The purpose of Recurrent Neural Networks (RNN) is to learn long-term dependencies. However, it is known that is difficult to learn to store information for very long [25]. One approach to correct for that is to augment the network with an explicit memory, called the Long Short-Term Memory (LSTM) network [26]. It was proposed for using special hidden units that lead to the natural behaviour of remembering inputs for a long time, known as memory cells. In other words, the gradient can flow for long time, thereby avoiding the vanishing gradient problem. Figure 2 [1] shows a memory cell, which is an accumulator that has a connection to itself at the next time step.

Basically, the LSTM process is composed by three gates (i.e. the forgot gate f_t, update gate i_t and output gate o_t). The variables x_t and h_t represent the input and output of the network at time t. The layers with sigmoid and hyperbolic tangent activation functions are represented by σ and $tanh$.

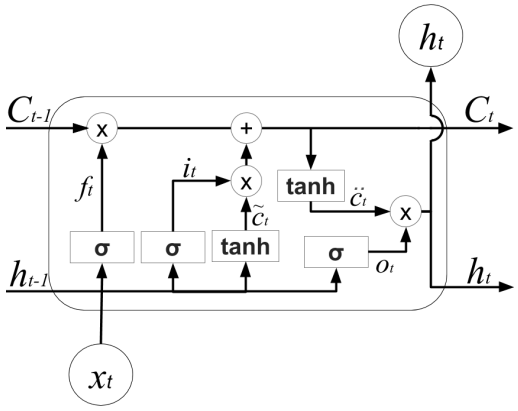

Fig. 2. The internal structure of a Long Short-Term Memory cell (LSTM).

Equations 1–7 present the mechanism of a LSTM.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (1) \qquad i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = tanh(W_C[h_{t-1}, x_t] + b_C) \quad (3) \qquad C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t \quad (4)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (5) \qquad \ddot{C}_t = tanh(C_t) \quad (6)$$

$$h_t = o_t \circ \ddot{C}_t \quad (7)$$

where, f_t, i_t, C_t, o_t and h_t ($\in \Re^k$) are activations of the forgot gate, update gate (or input gate vector), internal long term memory cell state, output gate vector (candidate) and output vector considering k hidden units, respectively. In addition, W_f, W_i, W_C and W_o ($\in \Re^{3q \times k}$) are the weight matrices; and b_f, b_i, b_C and b_o ($\in \Re^k$) are bias terms. Also, the symbol $\circ$ denotes the Hadamard product operator.

In this work, a Deep Bidirectional Long Short-Term Memory (DBLSTM) is used, which is based on [6] and it consists of stacked bidirectional Long Short-Term Memory (BLSTM) elements, as shown in Figure 3. Here, it is important to known that the BLSTM is considered to be the better choice because of its ability to use both previous and future information [27], accessing long-range context in both directions.

The features of the protein sequence, following the encoding presented in Section III, are connected to the input of the first layer at each time step. Each input of the *forward* and *backward* LSTM blocks receive the features and the reversed features, respectively. On the other hand, the output layer has 8 neurons that represent the 8 classes of secondary structure. Then, a *softmax* activation function returns a value that represent the predicted secondary structure (SS), following the 8-class SS classification (See Section III).

Finally, it is important to mention that the ReLU is used in the input of each layer. Also, the *Dropout* technique is used in the output of each layer during the training task.

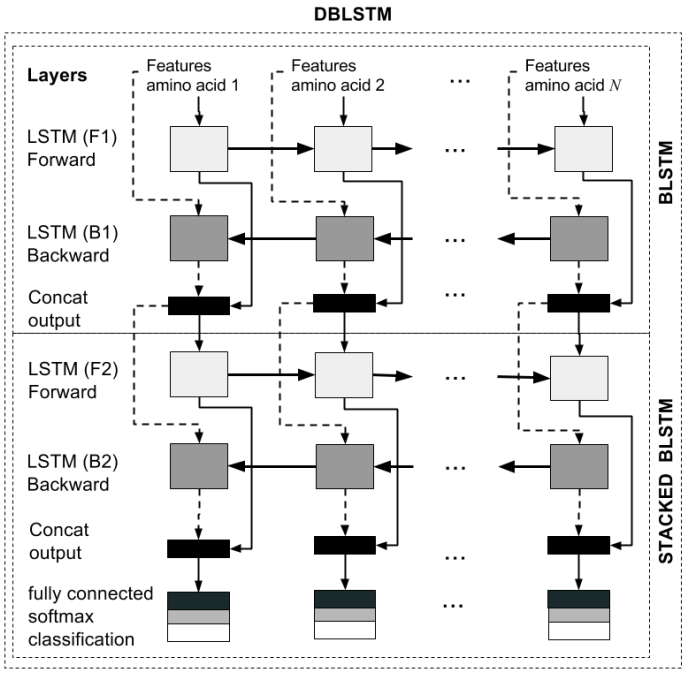

Fig. 3. The Deep Bidirectional Long Short-Term Memory network.

A. Cost Function

The cost function used in this work is the categorical cross-entropy between the predicted ($p_{i,j}$) and target ($t_{i,j}$) protein secondary structure classes, as presented in Equation 8. It is the loss function (L_i) for multi-class problems with softmax output.

$$L_i = -\sum_j t_{i,j} \log(p_{i,j}) \quad (8)$$

B. Gradient optimization approaches

Gradient optimization approaches are used in order to optimize the gradient descendent of the network and to control the value of the learning rate (η) which, in turn, determines the size of the steps that are taken towards the minimum error

[1] Based on colah.github.io/posts/2015-08-Understanding-LSTMs/

based on the cost function (See Section IV-A). In other words, η defines the velocity at which the network reach the minimum during the training process.

In this work, the following approaches were evaluated: Stocastic Gradient Descendet (SGD) [28], Momentum [29], AdaGrad [30], RmsProp [31], AdaDelta [32] and Adam [33]. Results are presented in Section V-A.

V. Computational Experiments and results

All experiments done in this work were run in a computer with an Intel Core i7 processor at 3.30GHz, a GPU Nvidia Titan X and a minimal installation of Ubuntu 14.04 LTS [2]. The software was developed using the Python programming language and the Lasagne framework [3].

The main objectives of this work are to evaluate the performance of Gradient optimization approaches (see Section V-A) and to compare the proposed approach with results obtained by other researchers (see Section V-B).

The motivation to test several Gradient optimization approaches is to identify the most suitable one for optimizing the DBLSTM architecture for the problem. The performance of our proposed approach and the methods from literature are measured according to their overall accuracy (A_{CC}) and other statistical parameters obtained from confusion matrices which are more suitable, regarding the highly unbalanced datasets as shown in Table I: weighted accuracy (wA_{CC}), Sensibility (S_e) and Specificity (S_p).

$$A_{CC} = \frac{TP}{(TP + FP)} \quad (9) \qquad wA_{CC} = \frac{\sum_i f_i . A_{CC_i}}{N} \quad (10)$$

$$S_e = \frac{T_P}{T_P + F_N} \quad (11) \qquad S_p = \frac{T_N}{F_P + T_N} \quad (12)$$

where,

T_P, F_P, T_N, F_N are the four possible outcomes that can be computed: the true positive, false positive, true negative and false negative, respectively.

Also, in Equation 10, the variables f_i and A_{CC_i} are the frequency (see the 2nd and 3rd columns of Table I) and the accuracy obtained for each ith class, respectively. N represents the number of classes.

A. Evaluation of Gradient Optimization approaches

As mentioned in Section IV-B, the performance of the Stocastic Gradient Descendet (SGD), Momentum, AdaGrad, RmsProp, AdaDelta and Adam approaches is studied in order to determine the most suitable approach to the problem. There is no specific procedure for adjusting parameters of the gradient optimization approaches. In this work, we decided to use the default values for the parameters that are available in the Lasagne Framework.

A visual comparison between the gradient optimization approaches was done using a ROC (Receiver Operating Characteristics) plot, which is commonly used in decision making in Machine Learning [3] and shows the difference between methods in a clear manner. For instance, Figure 4 presents the ROC plot for the approaches evaluated in this work for the following secondary structures classes: E (β-strand), L (loop) and T (β-turn). In a ROC plot, the x and y axes are defined as $(1 - S_p)$ and S_e, respectively. The best prediction would be the closer to the top left corner. Some of the optimizers achieved almost the same performance

In Figures 4(a), 4(b) and 4(c), it is observed that some of the optimizers have achieved almost the same performance. However, the results strongly suggests that the RmsProp is the top gradient optimizer for the problem and, therefore, the results obtained using this approach were used in the comparison with other approaches (see Section V-B).

B. Comparison with other Deep Learning approaches

In this section the results of our experiments are presented, as well as a comparison with the best results found in literatura. Table II presents the results obtained by our proposed approach (DBLSTM) and other researchers [6], [18]. In this table, first column represents the Deep network. The second column shows the overall accuracy (A_{CC}) obtained by our approach and the best results found in literature. The third column identifies the obtained weighted accuracy (wA_{CC}) Here, it is important to recall that the wA_{CC} is more suitable, regarding the highly unbalanced testing dataset, and it is not reported by the other researchers. Then, it is calculated from their reported results (note that [6] reports only the A_{CC}).

Two BLSTM networks were used by [6]. The DCRNN, DCNF, GSN, CNF and SSpro8 networks were used by [18]. In Table II, it is observed that our deep network achieved superior results than both BLSTM networks, indicating that the stacked BLSTM of our network brings higher levels of abstraction. In addition, our network is superior to SSprop8, CNF and GSN. Also, our model achieved slightly lower overall accuracies than DCRNN and DCNF (1,4% 0,3% of difference, respectively).

TABLE II
Comparison with other methods on the testing sets of CB513

Methods	A_{CC} (%)	wA_{CC} (%)
DBLSTM	68.0	66.0
BLSTM *small* [6]	67.1	*
BLSTM *large* [6]	67.4	*
DCRNN [18]	69.4	67.5
DCNF [18]	68.3	67.6
GSN [18]	66.4	63.6
CNF [18]	63.3	62.9
SSpro8 [18]	51.1	50.8

Regarding that the testing dataset is unbalanced, we also compare the classification accuracies of individual secondary structure classes obtained by our approach (DBLSTM) and the previously cited works (see Figure 5). Comparing our approach against the other methods, it is observed that the DBLSTM performs higher on the S (bend) class which, in turn, is a low frequency class. Also, the DBLSTM achieved slightly lower accuracies on lower frequency classes G (3_{10}-helix) and B (β-bridge), when compared with DCNF and DCRNN, respectively. Moreover, β-bridges (class I) had not

[2]Available in: www.ubuntu.com

[3]Available in: http://lasagne.readthedocs.io/en/latest/user/tutorial.html

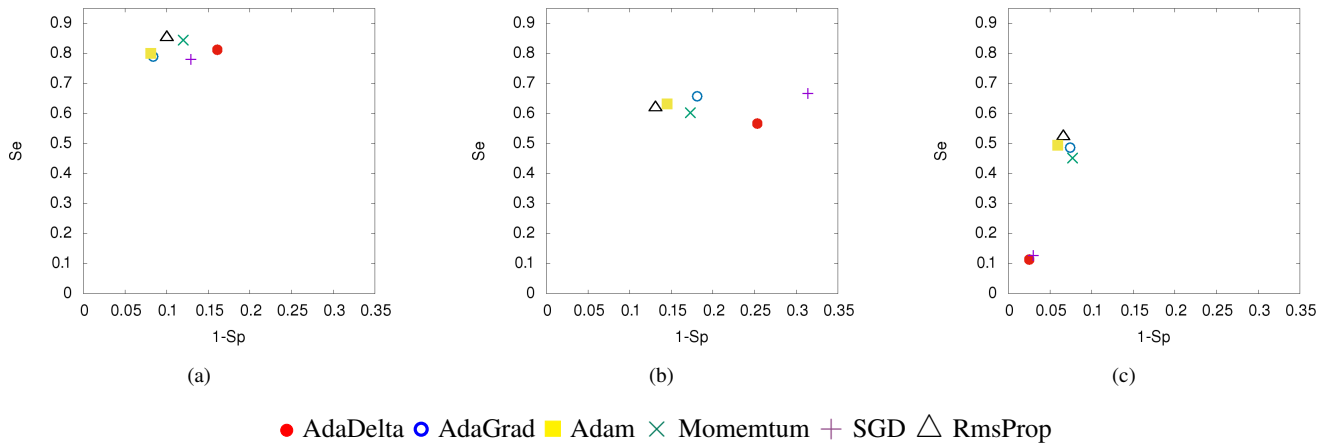

Fig. 4. Comparative analysis for classes: (a) E (β-strand) (b) L (loop) and (c) T (β-turn)

been classified correctly due to the very low frequency of the class (0.03%).

Considering high frequency classes H (α-helix), E (β-strand) and L (loop), our approach (DBLSTM), DCRNN, DCNF, GSN and CNF perform better than SSpro8. Also, it is possible to observe that CNF is the best among other methods for high frequency classes (H, E and L), considering the overall accuracy. On the other hand, it performs very poorly for low frequency classes (T, S, G, B and I). Furthermore, the DCRNN is the best for the class B. Finally, it is observed that SSpro8 is the worst in this comparison.

Here, it is important to recall that a fairer comparison can be done using ROC plots. However, the other works do not present the sensitivity (S_e) and specificity (S_p) measures.

Unlike the other previously cited works, we also reports the sensitivity (S_e) and Specificity (S_p) for each classes in Table III. In this table, it is observed that the sensitivity is inversely proportional to the frequency of the classes. The best results, according to these metrics, are shown in bold in the table.

TABLE III
SENSITIVITY (S_e) AND SPECIFICITY (S_p) PER SECONDARY STRUCTURE CLASS ON CB513

	H	E	L	T	S	G	B	I
S_e	**0.922**	**0.854**	**0.620**	0,524	0,127	0,245	0,005	0,000
S_p	**0.918**	**0.900**	**0.869**	0,934	0,988	0,988	1,000	1,000

VI. CONCLUSIONS

The prediction of protein structures is an important open research problem in Bioinformatics, which can lead to the development of highly specialised drugs for disease treatments. In this work, the performance of a Deep Bidirectional Long Short-Term Memory (DBLSTM) was analyzed in this paper, under the Protein Secondary Structure Prediction Problem.

The results obtained by our approach were competitive with the solutions found in literature. Also, with analysis according to the sensitivity, specificity and frequency of the classes, it is possible to verify that the DBLSTM approach can achieve better results if training with data augmentation is used.

Considering that only the overall accuracy rate may be misleading, we can conclude that the ROC plot analysis is a better way to analyze the classification performance of the approaches in this classification (or prediction) problem.

Parallel processing with GPUs is also essential to allow us to obtain high quality results in a reasonable computing time. As the size of the proteins and the network increase, the computational complexity will increase.

It is possible to highlight that the use of a self-adjusting strategy for tuning parameters of the gradient optimization approaches can be explored. This could be achieved using Evolutionary Computation approaches.

Future work will also include the use of hybrid techniques incorporating other types of deep neural networks, as well as hierarchical classification methods, memory network, Turing machines and clock-wise recurrent network. For instance, a Convolutional Neural Network (CNN) can be inserted in the first layer of our network. Also, better results can be achieved by increasing the number of neurons of the hidden layers and increasing the number of stacked BLSTM layers.

In a broader sense, the proposed approach presented in this paper is very promising for the research areas of Deep Learning and Bioinformatics.

ACKNOWLEDGMENT

Authors would like to thank the Brazilian National Research Council (CNPq) for the research grants to H.S. Lopes and L.T. Hattori. Also, we thank Nvidia for the GPU TITAN X donation.

REFERENCES

[1] L. Luheshi and C. Dobson, "Bridging the gap: From protein misfolding to protein misfolding diseases," *Federation of European Biochemical Societies Letters*, vol. 583, no. 16, pp. 2581–2586, 2009.

[2] M. E. M. Noble, J. A. Endicott, and L. N. Johnson, "Protein kinase inhibitors: Insights into drug design from structure," *Science*, vol. 303, no. 5665, pp. 1800–1805, 2004.

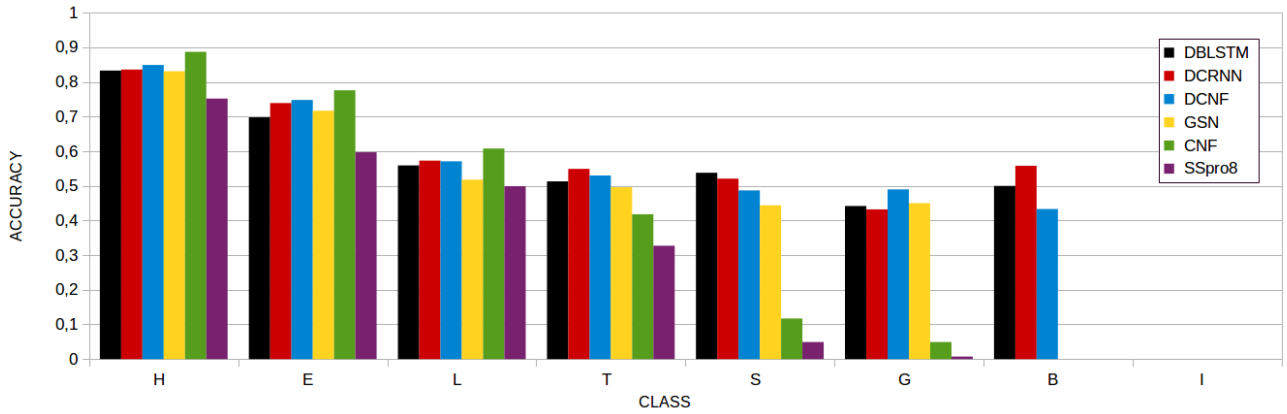

Fig. 5. A comparison of the classification accuracy of individual secondary structure classes on CB513 obtained by DBLSTM (black) and other approaches from literature.

[3] C. M. V. Benítez, C. Chidambaram, F. Hembecker, and H. S. Lopes, *A Comparative Study of Machine Learning and Evolutionary Computation Approaches for Protein Secondary Structure Classification*. INTECH Open Access Publisher, 2011.

[4] J. Hanson, Y. Yang, K. Paliwal, and Y. Zhou, "Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks," *Bioinformatics*, vol. 33, no. 5, p. 685, 2017.

[5] M. D. Wang and H. R. Hassanzadeh, "Deeperbind: Enhancing prediction of sequence specificities of dna binding proteins," *bioRxiv*, p. 099754, 2017.

[6] S. K. Sønderby and O. Winther, "Protein secondary structure prediction with long short term memory networks," arXiv preprint arXiv:1412.7828, 2015.

[7] S. Wang, J. Peng, J. Ma, and J. Xu, "Protein secondary structure prediction using deep convolutional neural fields," *Scientific Report*, vol. 6, p. 18962, 2016.

[8] L. Pauling, R. Corey, and H. Branson, "Configurations of polypeptide chains with favored orientations of the polypeptide around single bonds: two pleated sheets," *Proceedings of the National Academy of Sciences of the USA*, vol. 37, no. 11, pp. 729–740, 1951.

[9] L. Pauling, R. B. Corey, and H. R. Branson, "The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain," *Proceedings of the National Academy of Sciences of the USA*, vol. 37, pp. 205–211, 1951.

[10] P. N. Lewis, F. A. Momany, and H. A. Scheraga, "Chain reversals in proteins," vol. 303, no. 2, pp. 211–229, 1973.

[11] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features," vol. 22, no. 12, pp. 2577–2637, 1983.

[12] V. I. Lim, "Algorithms for prediction of alpha helices and structural regions in globular proteins," *Journal of Molecular Biology*, vol. 88, pp. 873–894, 1974.

[13] J. Garnier, D. Osguthorpe, and B.Robson, "Analysis and implications of simple methods for predicting the secondary structure of globular proteins." *Journal of Molecular Biology*, vol. 120, pp. 97–120, 1978.

[14] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *Journal of Molecular Biology*, vol. 292, no. 2, pp. 195–202, 1999.

[15] Z. Wang, F. Zhao, J. Peng, and J. Xu, "Protein 8-class secondary structure prediction using conditional neural fields," *Proteomics*, vol. 11, no. 19, pp. 3786–3792, 2011.

[16] J. Zhou and O. G. Troyanskaya, "Deep supervised and convolutional generative stochastic network for protein secondary structure prediction," in *Proceedings of the 31st International Conference on International Conference on Machine Learning*, ser. ICML'14, vol. 32. Journal of Machine Learning Research, 2014, pp. I–745–I–753.

[17] A. Yaseen and Y. Li, "Template-based c8-scorpion: a protein 8-state secondary structure prediction method using structural information and context-based features," *BMC Bioinformatics*, vol. 15, no. Suppl 8, pp. S3–S3, Jul 2014.

[18] Z. Li and Y. Yu, "Protein secondary prediction using cascaded convolutional and recurrent neural networks," in *International Joint Conference on Artificial Intelligence*. International Joint Conference on Artificial Intelligence, 2016, pp. 1–8.

[19] G. Wang and R. L. Dunbrack, "PISCES: a protein sequence culling server," *Bioinformatics*, vol. 19, no. 12, pp. 1589–1591, 2003.

[20] J. A. Cuff and G. J. Barton, "Evaluation and improvement of multiple sequence methods for protein secondary structure prediction," *Proteins: Structure, Function, and Bioinformatics*, vol. 34, no. 4, pp. 508–519, 1999.

[21] M. Gribskov, A. D. McLachlan, and D. Eisenberg, "Profile analysis: detection of distantly related proteins," *Proceedings of the National Academy of Sciences*, vol. 84, no. 13, pp. 4355–4358, 1987. [Online]. Available: http://www.pnas.org/content/84/13/4355.abstract

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[23] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[24] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks." in *Aistats*, vol. 15, no. 106, 2011, p. 275.

[25] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 273–278.

[28] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[29] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[30] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[31] T. Tieleman and G. Hinton, "Neural networks for machine learning, lecture 6.5 – rmsprop," 2012.

[32] M. D. Zeiler, "DADELTA: An adaptive learning rate method." 2012.

[33] D. Kingma and J. Ba, "Adam: A method for stochastic optimization." 2014.