Visualization and Analysis of Recurrent Geometric Network oriented to Protein 3D Structure Prediction

Kang Ye
Ruibo Gao (✉ 1319823634@qq.com)

Research Article

Keywords: Visualization, Hidden states, Bidirectional neural network, Recurrent geometric networks, Protein structure prediction

Posted Date: October 31st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2206279/v1

License: © This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
Abstract

Recurrent geometric network (RGN) as a deep learning model have been successfully applied to predict protein 3D structure and achieved better results than conventional methods. However, because of the inter- nal complexity and nonlinear structure of deep neural networks, the underlying processes why the model achieving such performance is challenging and sometimes difficult to explain. In this paper, for protein 3D structure prediction tasks, we analyze these hidden states of recurrent geometric network from a new perspective. Firstly, a method is proposed to explain network position characteristic by searching for the most similar amino acids near the selected amino acids network. We also analyze the update response of new neurons in forward and backward networks to further focus on the position characteristics of hidden state. Then, a superimposed visualization method is put forward to study which and when neurons in bidirectional networks have a greater influence on the prediction results. Finally, we adopt comparison method for torsion angles to analyze how the back- ward neuron affects the final prediction results. The usability and effectiveness of our method are demonstrated through case studies.

1 Introduction

In recent years, deep neural networks have become a central modeling tool for image recognition [1–3], text classification [4] and protein function and structure prediction [5, 6]. While all deep neural networks utilize hidden features, different model structures have shown to be effective for different tasks. Convolutional neural networks (CNNs) learn a task-specific filter-bank to produce spatial feature maps [7], whereas recurrent neural networks (RNNs) can capture representations of hidden time-series. In this work, recurrent geometric network (RGN), as a deep learning model, is used to obtain protein 3D structure, which consists of three stages: calculation, geometry and evaluation [8]. In the RGN model, the computational units are bidirectional LSTMs [9, 10]. For the current task, any existing method cannot directly obtain the protein 3D structure from the amino acid sequence. Especially, on all CASPs [11–13], RGNs outperformed other methods at both short and long multi-domain proteins. These neural networks models all share a common property in that they utilize a hidden feature representation of their input, not pre-specified by the user, which is learned for the task at hand. These hidden representations have proven to be very effective for classification. However, the black-box nature of these learned representations make the models themselves difficult to interpret. So while it is possible for users to produce high-performing systems, it is difficult for them to analyze what the system has learned.

Visualization has been used in several research areas to study the behavior of neural models. In fact, visualization research for neural networks started well before [14, 15]. Over just a handful of years, many different techniques have been introduced to help interpret what neural networks are learning. Xu et al. [16] showed through visualization how the model was able to automatically learn to fix its gaze on salient objects. Hermann et al. [17] visualized inference process of the model. Miao et al. [18] proposed two simplifications to the highly complex long short-term memory (LSTM) acoustic models. Indeed,
these methods have achieved some good results in explaining neural networks. Michiel et al. [19] provided a more in-depth analysis of how the network architecture relates to the nature of the task. The work proved that this method were well suited to capture temporal hierarchies, but only have a limited interpretability, as the individual layer contributions are likely not independent. Karpathy et al. [20] used static visualization techniques to help understand hidden states in language models, which demonstrated that selected cells can model clear events, but there are still some errors estimates. Tang et al. [21] employed visualization techniques to study the behavior of LSTM and gated recurrent unit (GRU), demonstrating LSTM and GRU possess different properties in both information encoding and temporal evolution. Kadar et al. [22] presented novel methods for analyzing the activation patterns of RNNs from a linguistic point of view, which provided an insight into how to improve systems. But there is still more room for developing models with better general-purpose sentence representations. Li et al. [23] explored multiple strategies to interpret meaning composition in neural models. The work demonstrated asymmetries of negation and explained some aspects of the strong performance of LSTMs at these tasks, but did not be performed error analysis. Rauber et al. [24] shown how dimensionality reduction can be used to visualize the relations between learned representations and between neurons in artificial neural networks. The work certified that visualization could provide highly valuable feedback for network designers. Hohman et al. [25] presented a comprehensive, timely survey on visualization and visual analytics in deep learning research, helping researchers to quickly learn key aspects of this young and rapidly growing body of research. Cashman et al. [26] presented a tool for providing insight into the training of recurrent neural networks, showing the user how quickly a model is learning, and how the regime of parameter updates changes over the course of training. The design had some visual layout issues with scale.

There has been some work on interactive visualization for not only interpreting machine learning models, but focusing on analysis of hidden states for convolutional neural networks. Abadi et al. [27] built a user interface, Tensor-Board, with a machine learning model as the background to research google products and services. Kapoor et al. [28] designed an interactive visual analytic system for practitioners to better understand their machine learning model and its predictions, but the approach could not take correlations or influences between features into account. Liu et al. [29] presented a visual analytics approach for better understanding, diagnosing, and refining deep CNNs. Ming et al. [30] presented a visual analytic method for understanding and diagnosing RNNs for text applications. The work demonstrated the effectiveness of the system and the completeness of design requirements, but the efficiency and quality of co-clustering method result in delays during interaction. Strobelt et al. [31] presented visual analysis tool for recurrent neural networks with a focus on understanding these hidden state dynamics. While the method did not be optimized, it still had a wide range of applications. Strobelt et al. [32] presented a visual analysis tool to explain of the training models through each stage of the
translation process, which showing the utility of the tool through several cases on large-scale models. But there are some improvements in visualization and the linear order of sequences. In this paper, our main contributions are summarized as follows:

1. We propose a method from new perspective to explore the hidden states of the recurrent geometric network, which help us discover hidden information and understand how the internal structure of the model affects the prediction results.

2. We analyze the gradual law of position information in the hidden state by calculating network similarity, and also analyze the update response of new neurons in forward and backward networks to further learn about position characteristics.

3. We study which and when neurons in bidirectional networks have a greater influence on the prediction results by the superimposed visualization method. In addition, we adopt comparison method for torsion angles to analyze how the backward neuron affects the final prediction results.

2 Dataset And Modela

In this section, we will describe three aspects including data set, RGN and bidirectional LSTM architecture.

2.1 Dataset

We use the test sample in the pre-trained model CASP7 https://github.com/aqlaboratory/rgn, in which the sequence information is T0283-T0386, a total of 93 chains. And then, we convert the protein three-dimensional structure file corresponding to each sequence into a torsion angle file, and obtain all the time step hidden states (h state) files corresponding to each sequence from the model. It should be noted that since the RGN author constructed the neural network from the end of the sequence, the predicted torsion angle data and hidden states data all start from the end of the sequence. In order to facilitate the description of the problem, we will follow the sequence of RGN generated data when describing the residue positions later, so the position of each residue is not the true position in its protein sequence. The specific reasons will be explained later. (As shown in the Fig. 1, the RGN author constructed the protein backbone structure from the end to the beginning of the sequence)

2.2 Brief Review of RGN Model and bidirectional LSTM

RGN model takes a sequence of amino acids and PSSMs as input and outputs the 3D structure of the protein. The detailed structural framework of RGN is shown in Fig. 1.

The first stage of RGN is computational units, which integrate information about the amino acid and PSSM with information coming from adjacent units by laying these units in a recurrent bidirectional
LSTM, and then by further stacking units in multiple layers, each unit outputs the torsional angles of the residue. The second stage is geometric units, which take as input the torsional angles for a given residue and the partially completed, and then the geometric units output a new backbone extended by one residue, which is fed into the adjacent downstream unit. The last stage is assessment unit that outputs the completed 3D structure of the protein. During model training, in order to optimize RGN parameters, a third stage minimize the deviations between the predicted and actual structure using the distance-based root mean square deviation ($dRMSD$) metric.

$$dRMSD = \frac{(r_{ac} - r_{pre})^2}{n}$$  \hspace{1cm} (1)

Where, $r_{ac}$ and $r_{pre}$ is the actual value and predicted value, respectively.

$n$ is the length of protein sequence.

The internal function of RGN is realized by a bidirectional LSTM. The traditional RNN can only obtain the features of a shorter range of sequence, while its variant LSTM has a larger range of sequence features, so the prediction result is better. The bidirectional LSTM can learn both the forward and reverse feature of the sequence. This network further improves the prediction effect of the network model. The simple running principle of LSTM is following as

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$c_t = f_t * c_{t-1} + i_t * \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * \tanh(c_t)$$  \hspace{1cm} (2)

Where, $W_f$, $W_i$, $W_c$, $W_o$ are weight matrices, $b_f$, $b_i$, $b_c$, $b_o$ are bias vectors.

$c_t$, $h_t$ are $c$, $h$ memory state for residue $t$, respectively. * is element-wise multiplication.

3 Results And Analysis

In this section, we mainly analyze the hidden layer of RGN from three aspects including the odd-even features of network position, the different effect of forward and backward network and how the backward
network affect the prediction result.

### 3.1 Odd-even Features of Hide State Based on Local Similarity Calculation

Our main work explores the hidden information in the state of RGN network, and its impact on the prediction results. For \( h \) state, we use a local search method to calculate the similarity of amino acid networks. Because the \( h \) state is a short-term memory, memory may be quickly lost at a long distance, leading to having the lower similarity between amino acid pairs.

Firstly, we analyze the hidden mode of the predicted torsion angles. Taking the protein sequence T0283 as an example, we visualize the three torsion angles corresponding to each amino acid in a sequence as a heat map, which is shown in Fig. 2. In Fig. 2, we can observe that the more similar the two torsion angles are, the closer the color is. For example, the predicted torsion angle 16 is more similar to that 18, but the predicted torsion angle 17 has more similarity to that 19. Therefore it can be seen that the prediction torsion angles of odd-numbered positions or even-numbered positions seem to be more similar. In fact, the properties of the predicted results reflect the same properties that the network has.

Then, since RGN network uses the last hidden layer to predict the results, we also use the last 1600 hidden states when calculating the similarity of the hidden states. Fig. 3 shows the similarity of the hidden state near the 16th amino acid. In this paper, we use Euclidean distance to calculate the similarity of two hidden state vectors. In Fig. 3, we take the 16th residue as the center point and use the distance formula to calculate the network similarity between the 20 residues near the center point and the center point. It can be seen from Fig. 3 that as the distance from the center position increases, the similarity of the hidden states gradually decreases. However, it is strange that the most similar to the center position 16 is the position 14 or 18 rather than the position 15 or 17. Not only that, the similarity value with an even distance from the center position is smaller than the odd position.

In order to eliminate unexpected situations, we further test and verify the effect. We first calculate the network similarity between each residue in the sequence and its nearby residues, and then superimpose the network similarity of the same distance from the center point. The superposition method is shown in Fig. 4.

Assume the residue \( i_{th} \in [1, N] \) is a center point, \( k \in N \) is the maximum distance from the center point, \( N \) is the length of the sequence, for the arbitrary residue \( j_{th} \in [1, N] \), \( \hat{d}_{i-th} - j-th = n \in (0, k] \) denotes the distance between the center point \( i_{th} \) and the residue \( j_{th} \), then the superimposed distances \( dis_n \) are calculated as follows:
where, \( Sim(.) \) is the network similarity of the hidden state, which is got by Euclidean distance.

When we take the leftmost end \( H \) of the sequence as the center point, the distance of nearby residues is displayed in position 1. If the residue \( L \) is the center point, there are two \( dis_1 \) distances in position 2. If the third position of the sequence is the center point, there are two \( dis_1 \) and two \( dis_2 \) distances. The case where the last residue of the sequence is the center is similar to the first case as shown at position \( N \).

Fig. 5 shows the superimposed effect after using the method in Fig. 4. As the distance from the center position increases, the similarity of the hidden states gradually decreases. Obviously, the histogram of the odd position is higher than the histogram of the even position, which indicates that all the residues in the sequence have learned the parity characteristics. In the abscissa 0-20, the height of all histograms increases with the increase of distance, which indicates that the similarity to the hidden states of the center position is gradually decreasing, that is, the information stored in the \( h \) state is gradually lost. When the abscissa is around 20-38, we find that the similarity has increased slightly, indicating that the ability of \( h \) state to memorize information has decreased, and it has not been able to fully memorize information.

From Fig. 3 and Fig. 5, we can see that there is an obvious difference of hidden states in odd and even position. In order to analyze the reason for a large gap between the similarities of the odd position and even position in the hidden states, we further explain the situation with Fig. 6. We use the hidden states networks corresponding to the 14\(_{th}\) residue of Fig. 3 to subtract the hidden states network corresponding to the 16\(_{th}\) residue of Fig. 3 to subtract the hidden states network corresponding to the 16\(_{th}\) residue. The difference value between the two hidden state networks is called the response states, which can reflect the sensitivity of the neural networks to the current input. The greater the absolute value of the response states, the greater the sensitivity of the network to the current input, which indirectly indicates the size of the network’s effect. Similarly, we employ the hidden states networks of the 16\(_{th}\) residue of Fig. 3 to subtract the hidden states networks of the 15\(_{th}\) residue.

The difference between the two networks can be used to analyze which neurons update causes the similarity to change.
Fig. 6 shows the comparison of hidden state in forward and backward network. It can be seen from Fig. 6 that the response value of the forward network is much smaller than that of the backward network. For Fig. 6(a), the red response value is larger than the blue response value, and there is almost no new neuron update. However, in the backward network of Fig. 6(b), the blue response value not only has a larger response value, but also has many new updates neuron. Therefore, we come to a conclusion that the neuron update of the backward network makes the good similarity between the even position 14 and the center point 16, and result in the poor similarity between the odd position 15 and 16. From the effect of the forward and backward networks, the forward network seems to be less effective and slow to respond.

3.2 Different effects Analysis of forward and backward networks

In this section, we mainly analyze the different effects of the forward and backward networks. We take the sequence T0299 as an example, and calculate the network response between residues 17 and 18. As shown in Fig. 7, it can be seen that the response of the backward network is much greater than that of the forward network, which is very common in the entire sequence. In order to further illustrate the situation of Fig. 7, we calculate the network response between all adjacent residues of a protein chain, and add up the absolute value of the response corresponding to the same neuron. As shown in Fig. 8, the forward network hardly responds, and the response of backward network is very obvious. According to the results shown in the figures, if the forward network is almost ineffective, then building a bidirectional LSTM network will lose its meaning, so we propose two hypotheses to explore the cause of the above problem as following.

- When predicting protein 3D structure based on the amino acid sequence, the backward sequence holds more information than the forward sequence, which causes the backward network to have a larger response in order to learn more information.
- It may be a problem with the constructed bidirectional network, which causes the forward network to not learn the information of the forward sequence well, so the response is small.

To verify the two hypotheses proposed, we first use a simple character language model as an example to describe. As shown in Fig. 9, for given the sequence ABCDEFGHI, when we predict word B from the forward and backward directions, respectively, the forward network needs to learn the information of A, and the backward network learn the information of IHGFEDC. However, because the front information of B is short and the complexity is low, the learning of the forward network is obviously easier, and the learning of the backward network is much more difficult. When the sequence length is longer,
the learning of backward network will becomes more difficult. When we predict E, the front and back information of E is ABCD and IHGF, respectively. At this time, since the length of the forward and backward information is the same, the effects of the forward and backward network should be roughly the same.

Then we calculate the responses of the residue 178-179 and 88-89 at the end and middle of the T0299 sequence, respectively. It can be seen from Fig. 10(a) that at the end of the sequence, the responses of the forward and backward network are equally intense. Therefore, we can infer that the forward network at this time has learned the characteristics of the forward sequence well. Simultaneously, we can overturn the hypothesis 1, which means that the feature information of the forward sequence is not less than that of the backward sequence. As shown in Fig. 10(b), when the length of the front and back information of the residue 88-89 is equal, the response of forward network is still much smaller than that of the backward network. Combining with the conclusions obtained in Fig. 10(a), at this time, we speculate that there may be a problem with the bidirectional network architecture.

In order to analyze where the forward network work in the sequence, we add up the absolute values of all forward neurons at each time step. As shown in Fig. 11(a), the forward network seems to have a smaller effect in the front and middle of the sequence, but suddenly has a larger effect at the end of the sequence. For the backward network, the same method is used to understand how it works. As shown in Fig. 11(b), the way the backward network works is completely different from the forward network. First of all, the value in Fig. 11(b) is larger and more complex, which shows that the backward network has learned more complex features. Secondly, Fig. 11(b) does not suddenly increase or decrease at the beginning or end of the sequence, and the characteristics it expresses have nothing to do with the length of the sequence. Finally, Fig. 11(b) has similar parity characteristics to Fig. 5, and we further find that the characteristics expressed by the two figures are completely different. As shown in Fig. 12, the feature of T0299 is that the value of odd digits is smaller and the value of even digits is larger, while the feature of T0295 in Fig. 11(b) is that the odd digits are larger and the even digits are smaller. This shows that the feature in Fig. 11(b) is related to the sequence number, while the feature in Fig. 5 is that the similarity value of the odd position in all sequences is greater than the even position, and the feature here has nothing to do with the sequence number.

From Fig. 11(a) and Fig. 11(b), we can infer that there seems to be a strange force hindering the forward network from learning features. When the end of the sequence is reached, this obstacle is reduced, allowing the forward network to learn features again. A very important message is that the authors of RGN construct the protein backbone from the end to the beginning of the amino acid sequence. Therefore, both the amino acid torsion angle data and the hidden states data start from the last residue of the protein sequence. As shown in Fig. 13, the Bidirectional neural network learns the forward and backward information of the sequence, and constructs the protein backbone from the G at the end of the protein sequence. Therefore, the beginning part in Fig. 11(a) corresponds to part1 in Fig. 13, and the end part corresponds to part3. The same goes for other images.
We compare the blue and yellow parts in Fig. 13. The yellow part representing the backward network starts learning from a simple word, and the learning content gradually becomes more complicated. However, the blue part representing the forward network starts learning from the most complex word string, and the learning content becomes gradually simpler. As we described in Fig. 9. Although both the forward network and the backward network are learning the characteristics of the sequence, the learning method of the forward network obviously does not conform to the general law, so the forward network is more difficult to learn. At this point we can explain the end of Fig. 11(a). In the beginning of the figure, because the features that the forward network needs to learn are too complex, it cannot perform well. When the end of the figure is reached, the features that the forward network needs to learn are too short, so the features of the sequence can be learned well.

3.3 The influence of backward network on prediction results

Section 3.1 and 3.2 mainly illustrate the importance of the backward network, and in this section, we analyze that how the neurons of the backward network affect the final prediction results.

In protein sequence T0299, we observe that the difference between $\phi$ or $\psi$ of the residues 1 and 2 is small, and the difference of $\omega$ is $+28.997$. In the same way, for residues 2 and 3, the difference of $\omega$ is $-33.993$. Fig. 14 shows the network responses of adjacent residues. It can be seen from Fig. 14 that the values of the red strips and blue strips are completely opposite in some places. In order to verify this characteristic, for the current sequence, we search for all adjacent residues whose the difference of torsion angle $\phi$ or $\psi$ is within 3 degrees, and difference of $\omega$ is greater than 10 degrees. Fig. 15 shows the average network responses of all residues meeting the above requirements. Fig. 15(a) and Fig. 15(b) are the average network responses corresponding to the forward and backward network, respectively. Compared to Fig. 15(b), the response value of the forward network is small and has no regularity. However, in the backward network, the red and blue strips are opposite, which shows that the backward network is more closely related to the prediction result, and the neurons of the backward network adjust the increase and decrease of the predicted torsion angle by controlling the positive or negative of its own value. We further observe that the red strips representing the increase of $\omega$ have both positive and negative values in the backward network. So the participation of all neurons makes the values of $\omega$ increase.

4 Discussion And Conclusion

In this work, we mainly study the hidden state of RGN from a new perspective. We first analyze the law of position information, and found that the hidden state of RGN has odd-even properties by calculating network similarity. In addition, we study which neurons in the forward and backward network have a greater influence on the prediction results by the superimposed visualization method. Finally, we adopt comparison method for torsion angles to find the internal backward network of RGN seems to play a
major role in the prediction result, and they influence the increase or decrease of the prediction result by controlling the positive or negative value.

However, due to the lack of biological background, we cannot understand why the network can learn these characteristics. In addition, we have not tested it in the traditional bidirectional RNN neural network model, so the above content cannot be extended to all bidirectional network models for the time being. In the future, we hope to verify our findings in other networks and propose better ways to interpret the network.

References


**Figures**

**Figure 1**

The framework of RGN network.

**Figure 2**
The Heat map of the prediction torsion angles. In torsion angle of Fig. 2, the negative values are filled with blue and green, and the yellow and red denote positive values.

Figure 3

Hidden states similarity histogram near the 16th amino acid in T0283. The horizontal coordinate is the position of each amino acid, and the vertical coordinate represents the hidden state similarity value of nearby amino acids and the center 16th amino acid. The smaller the value, the more similar it is.
Figure 4

The superimposing process of same distance in hidden states.

Figure 5

In sequence T0299, the stacking effect when the distance from the center point is 0-80.
Figure 6

Response comparison of forward and backward networks.

Figure 7
Network response between residues 17 and 18 in T0299. The left half of the figure (0-800) is forward network, and the right half (801-1600) is backward network.

**Figure 8**

The superimposed response of whole chain in T0299.

**Figure 9**

Character learning mechanism is based on bidirectional RNN.
Figure 10

Network response of in different positions. Fig.(a) is network response of residue178-179 at the end of T0295 sequence, and Fig.(b) denotes network response of residue 88-89 in the middle of T0299 sequence.
Figure 11

The superimposed effect of neurons about different protein at each time step. Fig.(a) is the superimposed effect of proteinT0299 in forward network, and Fig.(b) is the superimposed effect of proteinT0295 in backward network.
Figure 12

The superimposed effect of proteinT0299 in backward network.
Figure 13

The process of building a protein backbone by the bidirectional network. The G at the end of the protein sequence is starting point that the network constructs the protein backbone.
Network response comparison between residues 1-2 and 2-3 in T0299.

Figure 15

The superimposed effect of the forward or backward network in T0299.