Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.020771

ARTICLE

Interpreting Randomly Wired Graph Models for Chinese NER

Jie Chen[1], Jiabao Xu[1], Xuefeng Xi[1,*], Zhiming Cui[1] and Victor S. Sheng[2]

[1]The School of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou, China

[2]The School of Computer Science, Texas Tech University, Texas, USA

*Corresponding Author: Xuefeng Xi. Email: xfxi@mail.usts.edu.cn

Received: 10 December 2021 Accepted: 25 February 2022

ABSTRACT

Interpreting deep neural networks is of great importance to understand and verify deep models for natural language processing (NLP) tasks. However, most existing approaches only focus on improving the performance of models but ignore their interpretability. In this work, we propose a Randomly Wired Graph Neural Network (RWGNN) by using graph to model the structure of Neural Network, which could solve two major problems (word-boundary ambiguity and polysemy) of Chinese NER. Besides, we develop a pipeline to explain the RWGNN by using Saliency Map and Adversarial Attacks. Experimental results demonstrate that our approach can identify meaningful and reasonable interpretations for hidden states of RWGNN.

KEYWORDS

Named entity recognition; graph neural network; saliency map; random graph network; interpretation

1 Introduction

Deep learning has achieved a lot of success in many fields [1–4] because of the rapid expansion of data sets and computational power. On the one hand, researchers and industry are excited about deep learning's powerful representation and feature extraction capabilities. On the other hand, they are also concerned about the reliability and trustfulness of models constructed by deep learning and hope to understand what is going on inside the box that people cannot observe. Making deep learning models to be understood by humans is an important research topic that breaks the most fundamental dilemma of deep learning, because for some fields that require a high degree of trustworthiness, performance is not necessary, and trustworthiness is the top priority, such as finance and medical care, Court judgments, etc.

In recent years, deep neural network models have been widely applied to many natural language processing (NLP) applications [3,5,6], but one complaint they often suffer from is their lack of interpretability. The field of computer vision has forged its own path to enhancing deep learning model interpretability, most notably through post-hoc interpretation methods like saliency. For NLP, large neural NLP models, most notably BERT-like models [7–10], have become highly widespread, both in research and industry applications. This increase of model complexity is motivated by a general correlation between model size and test performance. Due to their immense complexity, these models

are generally considered black-box models. A growing concern is therefore if it is responsible to deploy these models. Concerns such as safety, ethics, and accountability are particularly important when machine learning is used for high-stakes decisions, including NLP-focused applications such as translation [11], dialog systems [12], resume screening [13], search [14], etc. [15]. Understanding why it is that the NLP predicting model works is becoming crucially influential. For such insights, post-hoc explanation approaches are valuable, for example, to determine whether a model is doing the "job correctly" before deployment [16,17], to increase human trust into black box systems [15], and to help diagnose model biases [18].

The structure of the neural network has an impact on the performance of models and has evolved from simple chain-like patterns to more sophisticated wiring paths (i.e., skip connection in ResNets [19]). The efficiency of the randomly wired neural network for improving the neural network's performance on image recognition tasks was demonstrated by Xie et al. [10]. Inspired by such the advantage, we further extend the principle of wired neural network to the field of natural language processing (NLP). The purpose of image classification is to distinguish different types of pictures, similar to the goal of named entity recognition, classifying different entities. Meanwhile, the limitation of the RNN [20] which is incapable of dealing with hierarchic sentence has been demonstrated in a study on the Lattice LSTM [21]. Hence, we propose a Randomly Wired Graph Neural Network (RWGNN) to construct a more robust model by exploring its connectivity to improve the accuracy of Chinese NER. The proposed RWGNN can overcome the limitation of the traditional chain-like structure of the neural network by enhancing semantic information and high-level features of a given context. We also use the lexical knowledge to associate characters to capture the local composition. Besides, an attention mechanism is used to capture similar entity attention. The node representation is computed by sequentially aggregating its incoming edges in RWGNN's neighborhood aggregation strategy.

Most existing approaches for interpreting NLP models focus on investigate the causation between input text and output decisions to explore which input tokens are more essential when making the final decisions [22,23]. However, the inner workings of networks should also be studied to answer important questions regarding hidden layers, such as which hidden units are more important for a decision and why they are important. Yuan et al. [24] first employ saliency map and optimization techniques to approximate the detected information of hidden neurons from input sentences. Thus, we use saliency map and adversarial attack-based approach to explain the inner workings of RWGNN.

This paper's primary contributions can be summarized as follows: 1) We propose an interpreting pipeline to explain the GNN-based model-RWGNN; 2) We use two interpreting methods, saliency map and adversarial attacks, to explain the reason why the RWGNN works. 3) We conduct several experiments to visualize the hidden states of RWGNN.

2 Related Work

2.1 Saliency Maps to NLP

Saliency Maps is an important technology that mainly used for model interpretation. By using this technology, human can understand which variables are important for the model. Moreover, Saliency Maps can be understood as the feature map of data. For image classification, it is to use this method to capture some pixels that have an impact on the results in the picture, and to extend it to the NER in natural language processing is to capture some words that have an impact on the results in sentences.

Li et al. [23] used Saliency Maps which method they used called First-Derivative Saliency to quantified the impact of each input on the final results of the model. This quantification can reflect

the negative asymmetry and spatial locality of the model. They used this approach to compare the interpretability and performance of three different models (RNN [25], LSTM and Bi-LSTM [26]) in sentiment analysis tasks. Guan et al. [27] inspired by previous researchers, used Saliency Map as an important tool to reveal Coherency in the model. They used Saliency Map to show the changes in the neural network layer with different input words, showing the changes in the amount of information contained in different neural network layers. This method verifies the information-based measurement method proposed by them, which quantitatively explains how the intermediate layer of the deep NLP model uses the input word information. In the analysis of NLI and sentiment analysis, Han et al. [28] used two complementary interpretation methods: gradient-based Saliency Map and influence function, and proved the feasibility of Saliency Map in deep transformer-based model and lexicon-driven sentiment analysis situation. However, the analysis using gradient-based Saliency Map on NLI which is more complex cannot produce better results compared with the method of influence function.

2.2 Adversarial Attacks to NLP

The main purpose of adversarial attacks is to obtain the weakness of the model. In a sense, it also operates as a test model to verify that the model is both robust and comprehensible. In NLP, the strategy of applying perturbations to make the model generate improper output with high confidence to expose deficiencies is not successful due to the discontinuity of word embedding. Therefore, in NLP, adversarial attacks often use a synonym dictionary to replace a word in the sample to expose the model's weaknesses. There are five main types of adversarial attacks against natural language text, including Model Access, Semantic Application, Target Type, Semantic Granularity and Attacked DNNS.

The white-box attack in the Model Access classification is one of the most serious attacks currently used on machine learning models. White-box adversarial examples are created by calculating derivatives for some character editing operations. There are a variety of classifications under white-box attacks. Direction based HotFlip [29] is a commonly used white-box attack method. It represents swap, insert, delete and other character-level operations as vectors in the input space, and estimates the loss changes of these vectors in the directional derivative. Ebrahimi et al. [30] extended the HotFlip method on this basis. In addition to the above attacks, they added the method of controlled attacks. This function is to modify the loss function so that the algorithm can delete specific words from the output and a targeted attack, which is simply a method to replace specific words.

In addition, they also proposed three types of attacks that can be modified. In addition, Feng et al. [31] used innovative character editing operations to delete, so that only high confidence words were retained in the text, although the results generated by this cannot be understood by human beings. This deletion method changes the method that generally finds large importance words by input perturbation or gradient, but removes the smallest importance token to find the most important input feature.

3 Our Proposed Model: RWGNN

Our Proposed Model: RWGNN Recurrent Neural Networks (RNN) [32–34] have achieved great successes on Chinese Named Entity Recognition (NER). However, such chain-like models are inadequate to capture hierarchical and nested entities due to their poor structures. Because of lacking structural information, the boundaries of entities are hard to be segmented. We propose a block-level Randomly Wired Graph Neural Network (RWGNN) to use a multi-directional wired pattern to extract contextual and structural information. Our experimental results on four NER datasets show

that RWGNN outperforms state-of-the-art baseline models and the general structure of RWGNN models we study is shown in Fig. 1.

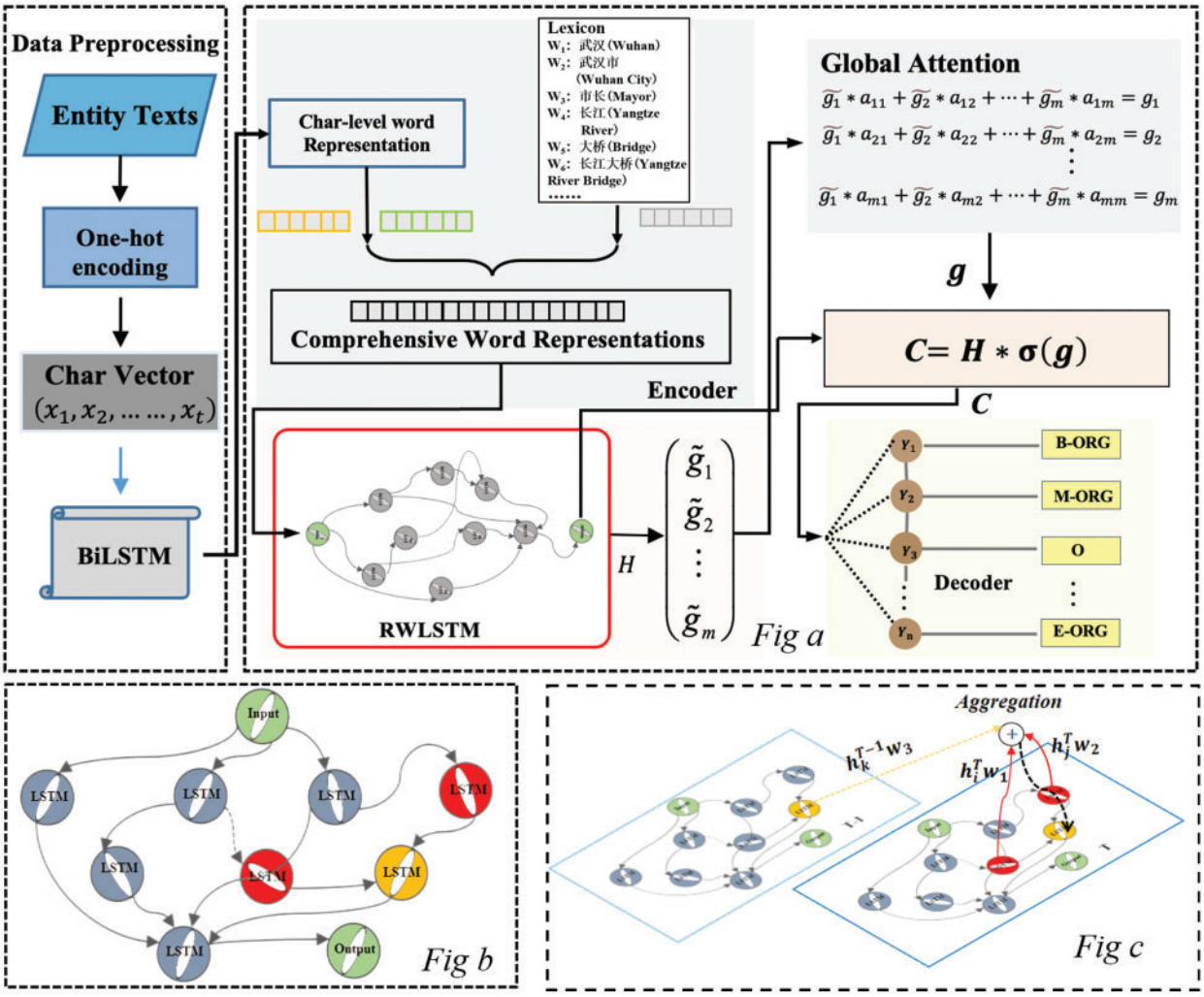

Figure 1: (a): is the overview of RWGNN. (b): Randomly-wired LSTM. (c): The aggreation operation

Fig. 1a is the overview of RWGNN. The encoder ouputs a comprehensive word representation vector h, which is served as the input to the randomly-wired LSTM, and its output is H. Then global attention unit captures the global correlations and outputs g. g and H are two inputs of the gated unit. At last, the CRF decoder generates labels L char-by-char according to the matrix Y. Our proposed RWLSTM is modeling the wiring pattern of neural network, which use a multi-directional wired pattern to extract and enhance contextual and structural information. Fig. 1b represents Randomly-wired LSTM. The edges between nodes are randomly generated. We use the graph structure to model the wiring pattern of the neural network. Take the three colored nodes, input and output node as an example to explain the process of word representations. In the input node, the same word representations are copied three times and passed to three neighbor nodes. Then, the 'LSTM nodes' (yellow node) perform the message exchange process based on the feature of the preceding neighboring node (including two red nodes). Finally, the concatenation of all original output nodes creates

the output of unique green node. Fig. 1c is the aggreation operation. The yellow node in T state is the target node k, two red nodes in T state are neighbor nodes of node k. The yellow in $T-1$ is the preceding state of node k. We perform an aggregation operation to update the current feature of node k.

We use biLSTM to produce word representations which also capture contextual feature information [35]. The representations of pre-trained models have already processed by others. In the course of processing, some information may be lost owing to various constraints, so the information is not sufficient for our task.

However, the word representation generated by char-level features and lexicon is like raw information. We can manipulate these raw data according to the characteristics of our target task while dealing with them, and the information about boundaries and ambiguities will be more appropriate and adequate.

We execute an attention mechanism on the representations (After being calculated by randomly-wired LSTM) to find the global information. The attention mechanism in transformer model [36] is a function that map the important and relevant words from the input sentence and assign higher weights to these words [37]. The equation is as Eq. (1) shows:

$$\text{Attention}\ (Q\ ,\ K\ ,V) = \ \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where Q is a query of target elements, (K) is a set of Keys, and (V) is the value pairs. Due to the process of rwLSTM (enhancing the features), the features have become extremely rich (such as context information and boundary information), but some useless information has also been enlarged, so we use it to enhance the important features and reduce the redundancy.

For training, we minimize the sentence-level negative log-likelihood loss as Eq. (2) shows:

$$L\left(\theta\right) = -\sum_{i=1}^{N}\log\left(p\left(y_i|s_i\right)\right) \tag{2}$$

In contrast to previous works by using graph neural network, like LGN [38], etc., which model text sequences to build a graph, Our RWGNN models the neural network's wiring pattern. The key of the RWGNN model is to use the Watts-Strogatz (WS) [39] model to generate a wired pattern between nodes, which will form a directed acyclic graph (DAGs). Each node in the RWGNN will contain a computable network block, which is the randomly-wired LSTM (RWLSTM) mapped by DAG through the generator. To enhance the extraction and enhancement of context and structure information RWLSTM uses a multidirectional connection mode. In RWGNN, the information exchange between nodes is realized by efficient graphic message transmission architecture.

MSRA/OntoNotes/Resume/Weibo. Table 1 shows the results for the MSRA, Weibo, and Resume datasets, respectively. Weibo, as well as the Resume dataset and the MSRA test set, lacked gold-standard segmentation. For Chinese NER, the top conventional methods included rich handcrafted features, embedding features, radical features, cross-domain features, and semi-supervised data; nonetheless, our RWGNN model outperformed the prevalent techniques and the word-level and character-level methods by a significant margin. On all three datasets, including MSRA, Resume and Weibo, it outperformed all other baselines. The automatic segmentation on the OntoNotes caused errors of word segmentation, resulting in a loss of performance on the downstream NER task. To obviate the necessity for word segmentation, it was possible to use character-level algorithms. The RWGNN model that we propose is a character-level model based on the WS model. In terms of the

F1-score, it outperformed the LGN and lattice LSTM by over 2%, and the F1-score increased by 0.49 percent over the FLAT.

Table 1: Main results (F1) on MSRA/Weibo/Resume/OntoNotes

Models	MSRA	Weibo	Resume	OntoNotes
Word-level LSTM	86.65	47.33	93.58	-
-char-bichar	90.28	52.33	94.24	-
Char-level LSTM	88.81	52.77	93.48	64.30
-bichar-softword	91.87	56.75	94.41	71.89
Lattice LSTM	93.18	58.79	94.46	73.88
LGN	94.19	60.21	95.37	74.89
FLAT	94.12	60.32	95.45	76.45
ERNIE 2.0 large	95.00	-	-	-
RWGNN	95.29	62.32	96.33	76.94

4 Interpretability of RWGNN

As shown in Fig. 2, we propose a pipeline of our interpreting approach. The Part 1 shows the brief general structure of the RWGNN model that we try to investigate. After training, we first build saliency maps for different hidden spatial locations, where saliency scores reflect contributions to the final decision. As the example shown in Part 2, the RWGNN model classifies the test sentence to class c (shown in green). For the hidden layer, the saliency score is computed for each hidden states, and three hidden states are selected (highlighted in red). The Part 3 is adversarial attack, we mask one or more embedding of the word representation, then we observe the change of label predictions.

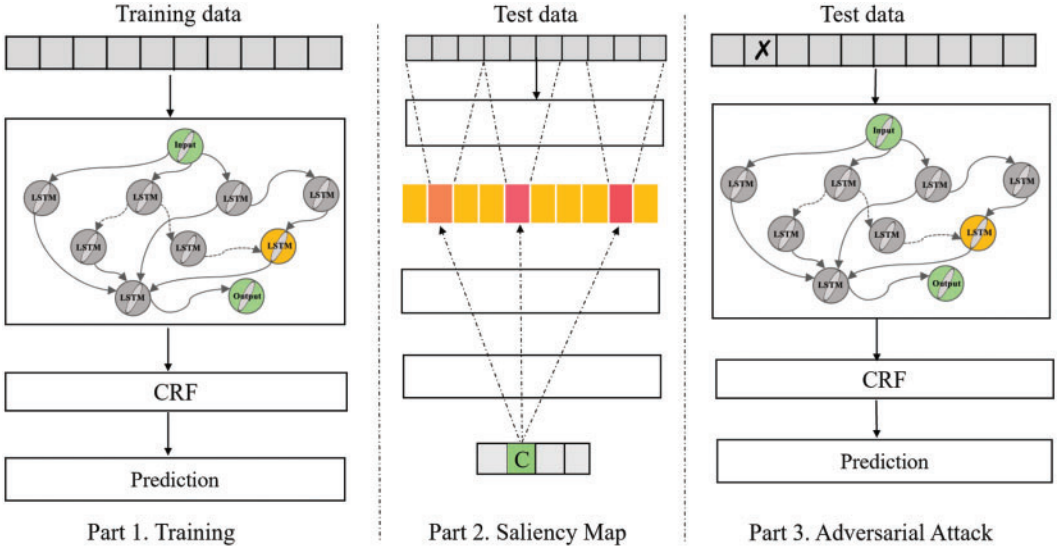

Figure 2: Illustration of the overall pipeline of our interpreting approach

4.1 Saliency Map Visualizations

We look at three different ways for assessing saliency, including Vanilla Gradient [40], Integrated Gradients [41] and SmoothGrad [42]. We utilize the model's output in the loss calculation since our goal is to understand why the model made its prediction (rather than the ground-truth response). By using the L2 norm, we can reduce each token's gradient (which is the same dimension as the token embedding) to a single value for each technique.

- *Vanilla Gradient* [40]: This method depicts the loss gradient for each token.
- *Integrated Gradients* [41]: This method establishes a baseline, which is an information-free input (we use a sequence of all zero embeddings). The gradient along the path from this baseline to the original input is integrated to establish word symbolic importance [10].
- *SmoothGrad* [42]: This method takes the gradient of multiple noisy copies of the input and averages it. Every embedding is given a small amount of Gaussian noise, and the average gradient value is calculated.

4.2 Adversarial Attacks

We explore two adversarial attacks: word replacement to change the model's prediction (HotFlip [29]) and word removal to preserve the model's prognosis (Input Reduction).

Untargeted & Targeted: To modify the model's prediction, HotFlip [29] utilizes the gradient to swap out words from the input. This provides a nearly counterfactual solution to the question of which words should be switched to produce a given prediction. According to the goal of attacks, it can usually be divided into two main modes, namely untarget attack and target attack. Target Attack disturbs input x by adding a minimum amount of perturbation δ, thus forcing the model to misclassify the disturbed samples into misclassification labels. In other words, for a multi-classification network, the attacker will induce the network to classify the input samples into a specified class label. Untarget Adversatial Attack will increase the disturbance δ on the input sample x, so that the model will produce wrong classification. We can regard this attack mode as a special case of target attack, which only needs to generate confrontation samples to deceive the original network.

Input Reduction: This strategy removes as many words from the input as possible without affecting the model's prediction [4]. Iteratively eliminating the word with the smallest gradient value is how input reduction works. Fig. 3 displays four examples of NER input reduction, where we achieved using AllenNLP interpret.

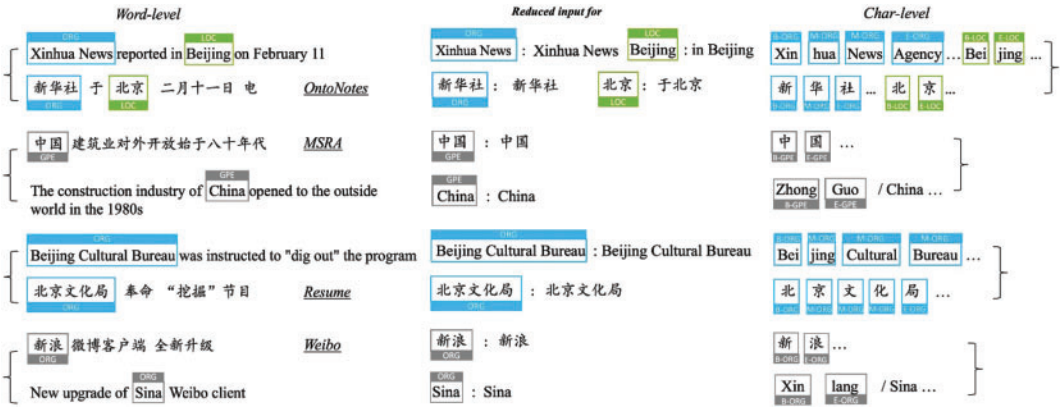

Figure 3: AllenNLP interpret was used to create an interpretation for Chinese NER

5 Experiments

In this section, we will first describe our experimental datasets shown in Table 2. Then, we experimentally show the interpretations of RWGNN by using four different angles. We also detail the hyperparameter configuration of our proposed model RWGNN.

Table 2: The details of four datasets

Dataset	Type	Train	Dev	Test
MSRA	Sent.	46.4 k	-	4.4 k
	Char.	2169.9 k	-	172.6 k
OntoNotes	Sent.	15.7 k	4.3 k	4.3 k
	Char.	491.9 k	200.5 k	208.1 k
Resume	Sent.	3.8 k	0.46 k	0.48 k
	Char.	124.1 k	13.9 k	15.1 k
Weibo	Sent.	1.4 k	0.27 k	0.27 k
	Char.	73.8 k	14.5 k	14.8 k

5.1 Datasets

Four Chinese NER datasets were used in our paper. (1) OntoNotes 4.0 [43]: OntoNotes is a multilingual News corpus which has been manually annotated and contains a variety of text annotations, including Chinese named entity labels. Segmentation that satisfies technical standards is provided. Only Chinese documents (about 16 k phrases) are used, and the data is analyzed in the same way as Che et al. [44] (2) MSRA [45]: MSRA is a News dataset that includes three different types of named entities: LOC, PER, and ORG. The training set has Chinese word segmentation, while the test set does not. (3) Weibo NER [46]: It is a collection of annotated NER messages culled from Sina Weibo, a Chinese blogging platform. For both named entity and nominal mention, the corpus includes PER, ORG, GPE, and LOC. (4) Resume NER [21]: It is built out of Sina Finance resumes which have been annotated with eight various categories of named entities. The gold-standard Chinese segmentation is lacking from both the Weibo and Resume datasets. Table 2 shows the details of the four datasets.

5.2 Hyper-Parameter Setting

With a learning rate of $2e-5$ for OntoNotes and MSRA, and $2e-4$ for the tiny datasets from Weibo and Resume, we implemented Adam [47] as an optimizer. We preferred dropout at a rate of 0.5 for embeddings and 0.2 for aggregation module outputs to reduce overfitting. The embedding and state sizes were both set at 50. The number of multi-head attention heads was adjusted to ten. For minor datasets like Resume and Weibo, the head dimensions were set to 10, and for MSRA and OntoNotes, they were set to 20. As performance indicators, the conventional Precision (P), F1-score (F1), and Recall (R) were utilized.

5.3 AllenNLP Interpret

Many research codebases bury high-level parameters under implementation details, are challenging to run and debug, and are difficult enough to extend that they are more likely to be rewritten. AllenNLP[1] [48] is a deep semantic natural language processing platform designed to address these problems and to significantly lower barriers to high quality NLP research in 2018. Based on AllenNLP, Wallace et al. [18] develop a extensive framework which is called "AllenNLP Interpret" to interpreting NLP models in 2019. This toolkit includes interpretation primitives (such as input gradients) for all NLP models and tasks, as well as a library of front-end visualization components. For BERT's masked language modeling purpose, we might utilize AllenNLP interpret to produce a saliency map by utilizing Vanilla Gradient [40]. We use it to demonstrate RWGNN in our paper by using an input reduction task, as shown in Fig. 3. The words "∓" or without any further words are adequate to forecast Organization and Location tags under word-level and char-level labels, respectively, according to input reduction. The results show that the model can capture as much word information as possible in the training step, and make label predictions based on as little information as possible in the inference step.

5.4 Visual Interpretation of Hidden States

In this work, we investigate the interpretation of RWGNN model for name entity recognition tasks in NLP. The general structure of RWGNN models we study is shown in Fig. 1 and explained in Section 3. Intuitively, we wish to investigate the hidden states of a GNN-based model so that we can answer three questions; those are, which hidden states are more important to decisions? what is detected by these hidden states from input sentences? and what is the meaning of the detected information? However, there are two main challenges for answering these questions; those are, how to explore what is detected by hidden states? and how to interpret the detected information? Existing approaches in computer vision cannot be directly applied since word representations are discrete from each other and cannot be abstracted as images. We combine the idea of saliency map and optimization to answer the question of what is detected by hidden states. Based on the property of word representations and graph neural network, we propose to approximately interpret the meaning of detected information, which contains the word-boundary feature. Then we use regularization terms to locate the word-boundary feature. Generally speaking, the interpretation procedure consists of three main steps. First, we employ gradient-based approaches to estimate the contributions of different states in a hidden layer. Based on the magnitude of contributions, the hidden states are sorted, and those with high contribution are selected to be interpreted in the following steps. Second, to obtain what is detected by different states, we iteratively update a randomly initialized input. Finally, the initialized input is a sequence of numerical vectors but such abstract values are hard to interpret. Based on the property of word representations that words with semantically similar meanings are embedded to nearby points, we design regularization terms to encourage different vectors to be similar to each other. Then we explore the word-boundary feature in term of cosine similarity to approximately represent the meaning of the target hidden states.

We visualize three hidden states, which contains word-boundary features, as shown in Fig. 4. These three hidden was processing same representation of text sequence"青岛省长假借助青海湖旺季... ", but the boundary information of the same Chinese sentence detected by these three hidden states are not completely consistent. The dark blue and yellow hidden states believes that the boundary exists between the character '省' (province) and '长' (governor), and the light blue indicate the boundary

[1] https://github.com/allenai/allennlp.

exists between the character '海' (hai) and '长' (zhang). Thus, the final and correct word boundary is detected by dark blue and yellow hidden states.

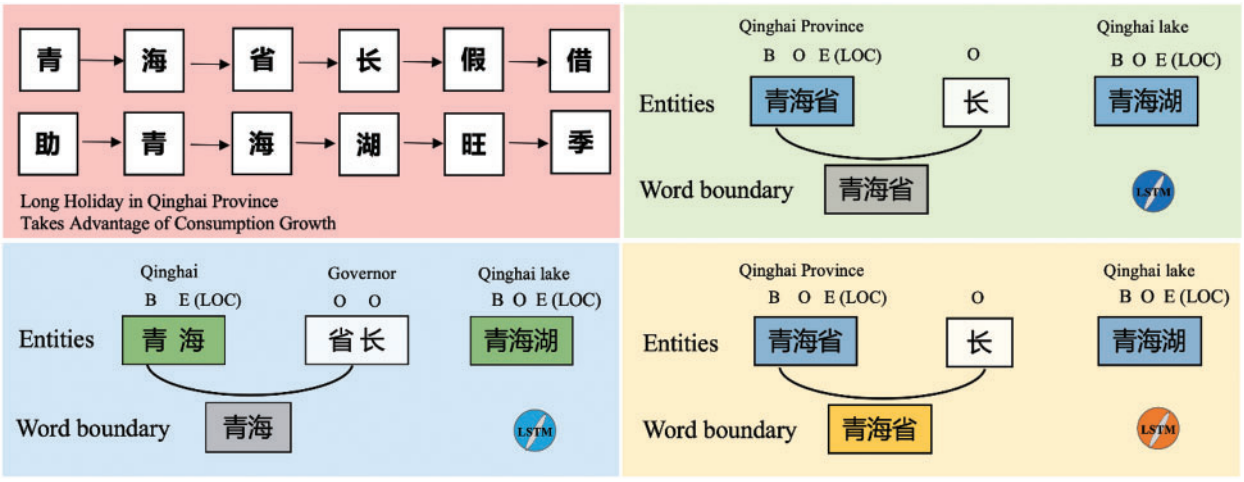

Figure 4: Visual interpretation of hidden states for word ambiguity example

5.5 Saliency Maps for Hidden States

Since there are a large number of neurons in hidden layers, it is not possible to interpret each neuron. Hence, we employ saliency map techniques to select hidden states with high contributions for further interpretation. The saliency map acts like a heatmap, where saliency scores are estimated by the first order derivatives and reflects the contribution of different neurons. While most of existing approaches build saliency maps to explore the contribution of individual words or characters in input sentences, we study the importance of different hidden states instead. Formally, for an input sentence X, the model predicts that it belongs to label class c and produces a label class score S_c. Let a_{ij} represents the activation vector of the hidden state i of step j. Also let A_j denotes the activations of step j, which is a matrix, where each column corresponds to one hidden state. The relationship between the score Sc and A_j is highly non-linear due to the nonlinear functions in deep neural networks. Inspired by the strategy in recent works [23,40], we compute the first-order Taylor expansion [49] as a linear function to approximate the relationship as Eq. (3) shows:

$$S_c \approx Tr\left(w(A_j)^T A_j\right) + b \tag{3}$$

where Tr $(\cdot)$ denotes the trace of a matrix and $w(A_j)$ is the gradient of class score S_c with respect to the layer j. Such gradient can be obtained by using the first order derivative of S_c with respect to the layer A_j as Eq. (5) shows:

$$w\left(A_j\right) = \frac{\partial S_c}{\partial A_j} \tag{4}$$

For the hidden state i in the step j, the gradient of S_c with respect to this hidden state is the i^{th} column of $w(A_j)$, denoted as $w(A_j)_i$. Then the saliency score of this location $Scorec(X)_{i;j}$ is calculated using linear approximation as Eq. (5) shows:

$$Score_c(X)_{i,j} = w\left(A_j\right)_i \cdot a_{ij} \tag{5}$$

where · refers to the dot product of vectors. It is noteworthy that we do not directly use gradients as saliency scores. The reason is that gradients only reflect the sensitivity of the class score when there is a small change in the corresponding spatial location. The employed linear approximation incorporates the activation values to measure how much one hidden state contributes to the final class score. In addition, after training, the weights and parameters in the model are fixed so that the gradient of S_c with respect to a specific spatial location is fixed and does not depend on the input. By using the linear approximation, the saliency score becomes input-dependent.

As shown in Table 3, we use three different saliency methods, including Gradient, SmoothGrad and Integrated Gradients, to visualize the tokens recognized by RWGNN. These tokens not only contain entities but also words contained in dictionary information. The red token represents that the current token is very important for the entire text, while the blue tokens represent relatively low attribution. Compared with Gradient, SmoothGrad and Integrated Gradients have many more restrictive regulations, so the number of tokens spotted by them is also less than that of Gradient method.

Table 3: The two examples intrepreted by three different saliency methods, including gradient, smoothgrad and integrated gradients. Color legend: Lower attribution, Higher attribution

Dataset	Text	Visualization: tokens
Gradient	印度河流经印度	印度河; 流经 ; 河流 ; 印度
	India river flow through India	India River; flow through; River; India
	青海省长假借助青海湖旺季	青海; 省长; 青海省 ;青海湖
	Long holiday in Qinghai Province takes advantage of Qinghai Lake peak season	Qinghai; Governor; Qinghai Province; Qinghai Lake
SmoothGrad	印度河流经印度	印度河; 河流 ; 印度
	India river flow through India	India River; River; India
	青海省长假借助青海湖旺季	青海; 青海省; 青海湖
	Long holiday in Qinghai Province takes advantage of Qinghai Lake peak season	Qinghai; Qinghai Province; Qinghai Lake
Integrated Gradients	印度河流经印度	印度; 印度河
	India river flow through India	India; India River
	青海省长假借助青海湖旺季	青海; 青海湖
	Long holiday in Qinghai Province takes advantage of Qinghai Lake peak season	Qinghai; Qinghai Lake

5.6 Analysis of Adversarial Attacks

Researchers mainly uses black-box and white-box adversaries in the NLP field, Ebrahimi et al. [30] demonstrate that white-box adversaries significantly outperform black-box adversaries especially in targeted scenarios on character-level neural machine translation task. Thus, we follow this conclusion and we use two type methods of white-box adversaries–targeted and untargeted attacks. We implement these two adversarial attacks to test the robustness of RWGNN. We use these two attacks to manipulate the word sequence by applying four different operations–Flip, Insert, Delete and Swap. Besides, we use Cosine similarity to measure the similarity between original text and attacked text, which not only include sentence but also the predicted entities. Table 4 shows Cosine similarity of original and attacked text under the untargeted and targeted attacks. We randomly pick 100 text sequences from

four datasets and the results are the average of them. If the value is closer to 0, it means that the similarity of the two objects is higher, otherwise the similarity is lower. Unlike delete and swap, insert and flip have the advantage of making changes to one-letter words, so we expect them to perform better. We see this for the white-box attacks which can pick the best change to every word using the gradients. Targeted attack is more challenging for not only mute a word and entity but also replace it with another one. The evaluation metric for targeted attack is same to untargeted attack. targeted attacks are relatively simple, since an adversary can perturb the input to move it to the other side of a decision boundary.

Table 4: Cosine similarity of original and attacked text under the untargeted and targeted attacks

Attack	Flip		Insert		Delete		Swap	
	Targeted	Untargeted	Targeted	Untargeted	Targeted	Untargeted	Targeted	Untargeted
ON	0.451	0.214	0.420	0.301	0.621	0.356	0.412	0.317
MSRA	0.376	0.247	0.353	0.317	0.541	0.435	0.395	0.294
Weibo	0.357	0.218	0.421	0.263	0.741	0.414	0.547	0.358
Resume	0.476	0.258	0.564	0.367	0.732	0.574	0.631	0.491

Under the targeted attack, the cosine similarity is higher than that of the untargeted attack, that is, the attack on the targeted entity will make the original text and the attacked text have a more obvious difference. It also proves that the entities recognized by RWGNN are extremely important for the entire context. Specifically, the operation of deleting entities has the greatest impact on the similarity between texts compared to the other three operations.

6 Conclusion

Investigating hidden states in graph neural networks are of great importance to understand their working mechanisms. However, most current approaches focus on models for images tasks. It is challenging to understand the meaning of hidden states in NLP models, since word representations are discrete and cannot be abstracted. In this work, we propose a pipeline to interpret deep NLP models. We combine the saliency map and adversarial attacks techniques to explore the information detected by the most important hidden states in NLP models. It is also shown that our method helps explain how the decision and why the decision is made.

Funding Statement: The authors are grateful to the anonymous reviewers for their valuable comments. This research has been supported by the National Science Foundation of China (NSFC) under Grants 61876217 and 62176175; the Innovative Team of Jiangsu Province under Grant XYDXX-086 and Jiangsu Postgraduate Research and Innovation Plan (KYCX20_2762).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. et al. (2016). Deep learning for visual understanding: A review. *Neurocomputing, 187,* 27–48. DOI 10.1016/j.neucom.2015.09.116.
2. Miotto, R., Wang, F., Wang, S., Jiang, X., Dudley, J. T. (2018). Deep learning for healthcare: Review, opportunities and challenges. *Briefings in Bioinformatics, 19(6),* 1236–1246. DOI 10.1093/bib/bbx044.
3. Young, T., Hazarika, D., Poria, S., Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine, 13(3),* 55–75. DOI 10.1109/MCI.2018.2840738.
4. Wu, Z., Jiang, B., Karimi, H. R. (2019). A logarithmic descent direction algorithm for the quadratic knapsack problem. *Applied Mathematics and Computation, 369,* 124854. DOI 10.1016/j.amc.2019.124854.
5. Xi, X., Zhou, P., Zhou, G. (2020). Global encoding for long Chinese text summarization. *ACM Transactions on Asian and Low-Resource Language Information Processing, 19(6),* 1–17. DOI 10.1145/3407911.
6. Wu, Z., Karimi, H. R., Dang, C. (2019). An approximation algorithm for graph partitioning via deterministic annealing neural network. *Neural Networks, 117,* 191–200. DOI 10.1016/j.neunet.2019.05.010.
7. Liu, W., Zhou, P., Zhao, Z., Wang, Z., Deng, H. et al. (2020). Fastbert: A self-distilling bert with adaptive inference time. arXiv preprint arXiv:2004.02178.
8. Ulčar, M., Robnik-Šikonja, M. (2020). Finest bert and crosloengual bert. *International Conference on Text, Speech, and Dialogue*, pp. 104–111, Brno, Czech Republic.
9. Gao, Z., Feng, A., Song, X., Wu, X. (2019). Target-dependent sentiment classification with bert. *IEEE Access, 7,* 154290–154299. DOI 10.1109/Access.6287639.
10. Xie, S., Kirillov, A., Girshick, R., He, K. (2019). Exploring randomly wired neural networks for image recognition. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Korea, pp. 1284–1293.
11. McCann, B., Bradbury, J., Xiong, C., Socher, R. (2017). Learned in translation: Contextualized word vectors. arXiv preprint arXiv:1708.00107.12.
12. Zhang, Z., Takanobu, R., Zhu, Q., Huang, M., Zhu, X. (2020). Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences,* 63, 2011–2027. DOI 10.1007/s11431-020-1692-3.
13. Singh, A., Rose, C., Visweswariah, K., Chenthamarakshan, V. (2010). PROSPECT: A system for screening candidates for recruitment. *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pp. 659–668. Canada.
14. Balush, I., Vysotska, V., Albota, S. (2021). Recommendation system development based on intelligent search nlp and machine learning methods. *CEUR Workshop Proceedings*, vol. 2917, pp. 584–617. Lviv-Shatsk.
15. Doshi-Velez, F., Kim, B. (2017). Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.
16. Ribeiro, M. T., Singh, S., Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. San Francisco.
17. Lundberg, S. M., Lee, S. I. (2017). A unified approach to interpreting model predictions. *Proceedings of the 31th International Conference on Neural Information Processing Systems*, pp. 4768–4777. Long Beach.
18. Wallace, E., Tuyls, J., Wang, J., Subramanian, S., Gardner, M. et al. (2019). Allennlp interpret: A framework for explaining predictions of NLP models. arXiv preprint arXiv:1909.09251.
19. He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778. Las Vegas.
20. Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D Nonlinear Phenomena, 404,* 132306. DOI 10.1016/j.physd.2019.132306.

21. Zhang, Y., Yang, J. (2018). Chinese NER using lattice LSTM. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 1554–1564. Melbourne.
22. Lei, T., Barzilay, R., Jaakkola, T. (2016). Rationalizing neural predictions. arXiv preprint arXiv:1606.04155.
23. Li, J., Chen, X., Hovy, E., Jurafsky, D. (2016). Visualizing and understanding neural models in NLP. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 681–691. San Diego.
24. Yuan, H., Chen, Y., Hu, X., Ji, S. (2019). Interpreting deep models for text analysis via optimization and regularization methods. *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, pp. 5717–5724. Hawaii.
25. Zaremba, W., Sutskever, I., Vinyals, O. (2014). Recurrent neural network regularization. arXiv preprint arXiv:1409.2329.
26. Huang, Z., Xu, W., Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991.
27. Guan, C., Wang, X., Zhang, Q., Chen, R., He, D. et al. (2019). Towards a deep and unified understanding of deep neural models in NLp. *Proceedings of the 36th International Conference on Machine Learning*, pp. 2454–2463. Long Beach.
28. Han, X., Wallace, B. C., Tsvetkov, Y. (2020). Explaining black box predictions and unveiling data artifacts through influence functions. arXiv preprint arXiv:2005.06676.
29. Ebrahimi, J., Rao, A., Lowd, D., Dou, D. (2017). Hotflip: White-box adversarial examples for text classification. arXiv preprint arXiv:1712.06751.
30. Ebrahimi, J., Lowd, D., Dou, D. (2018). On adversarial examples for character-level neural machine translation. arXiv preprint arXiv:1806.09030.
31. Feng, S., Wallace, E., Grissom II, A., Iyyer, M., Rodriguez, P. et al. (2018). Pathologies of neural models make interpretations difficult. arXiv preprint arXiv:1804.07781.
32. Dong, X., Chowdhury, S., Qian, L., Guan, Y., Yang, J. et al. (2017). Transfer bi-directional LSTM RNN for named entity recognition in Chinese electronic medical records. *IEEE 19th International Conference on e-Health Networking*, pp. 1–4. Dalian, China.
33. Chowdhury, S., Dong, X., Qian, L., Li, X., Guan, Y. et al. (2018). A multitask bi-directional rnn model for named entity recognition on Chinese electronic medical records. *BMC Bioinformatics, 19(17),* 75–84. DOI 10.1186/s12859-018-2467-9.
34. Li, J., Zhao, S., Yang, J., Huang, Z., Liu, B. et al. (2020). WCP-RNN: A novel RNN-based approach for bio-ner in Chinese emrs. *The Journal of Supercomputing, 76(3),* 1450–1467. DOI 10.1007/s11227-017-2229-x.
35. Zhong, Q., Tang, Y. (2020). An attention-based BILSTM-CRF for Chinese named entity recognition. *2020 IEEE 5th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, pp. 550–555. Chengdu, China.
36. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. (2017). Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, New York, pp. 6000–6010.
37. Lin, J., Sun, X., Ma, S., Su, Q. (2018). Global encoding for abstractive summarization. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 163–169. Melbourne.
38. Gui, T., Zou, Y., Zhang, Q., Peng, M., Fu, J. et al. (2019). A lexicon-based graph neural network for Chinese NER. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1040–1050. Hong Kong.
39. Watts, D. J., Strogatz, S. H. (1998). Collective dynamics of a small-world networks. *Nature, 393(6684),* 440–442. DOI 10.1038/30918.

40. Simonyan, K., Vedaldi, A., Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034.
41. Sundararajan, M., Taly, A., Yan, Q. (2017). Axiomatic attribution for deep networks. *Proceedings of the 34th International Conference on Machine Learning*, pp. 3319–3328. Sydney.
42. Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M. (2017). Smoothgrad: Removing noise by adding noise. arXiv preprint arXiv:1706.03825, 2017.
43. Weischedel, R., Pradhan, S., Ramshaw, L., Palmer, M., Xue, N. et al. (2011). *Ontonotes release 4.0.* LDC2011T03, Philadelphia, Penn: Linguistic Data Consortium.
44. Che, W., Wang, M., Manning, C., Liu, T. (2013). Named entity recognition with bilingual constraints. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 52–62. Atlanta.
45. Zhang, S., Qin, Y., Wen, J., Wang, X. (2006). Word segmentation and named entity recognition for SIGHAN bakeoff3. *Proceedings of the 5th SIGHAN Workshop on Chinese Language Processing*, pp. 158–161. Sydney.
46. Peng, N., Dredze, M. (2015). Named entity recognition for Chinese social media with jointly trained embeddings. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pp. 548–554. Lisbon.
47. Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
48. Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P. et al. (2018). Allennlp: A deep semantic natural language processing platform. *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pp. 1–6. Melbourne, Australia.
49. André, R., Luciani, X., Moreau, E. (2020). Joint eigenvalue decomposition algorithms based on first order taylor expansion. *IEEE Transactions on Signal Processing, 68,* 1716–1727. DOI 10.1109/TSP.78.