Graph Convolutional Networks for Student Answers Assessment

Nisrine Ait Khayi and Vasile Rus

University of Memphis, Institute for Intelligent Systems
Memphis TN, USA
{ntkhynyn,vrus}@memphis.edu

Abstract. Graph Convolutional Networks have achieved impressive results in multiple NLP tasks such as text classification. However, this approach has not been explored yet for the student answer assessment task. In this work, we propose to use Graph Convolutional Networks to automatically assess freely generated student answers within the context of dialogue-based intelligent tutoring systems. We convert this task to a node classification task. First, we build a DTGrade graph where each node represents the concatenation of the student answer and its corresponding reference answer whereas the edges represent the relatedness between nodes. Second, the DTGrade graph is fed to two layers of Graph Convolutional Networks. Finally, the output of the second layer is fed to a softmax layer. The empirical results showed that our model reached the state-of-the-art results by obtaining an accuracy of 73%.

Keywords: Graph Convolutional Networks, Student Answers Assessment, Intelligent Tutoring Systems.

1 Introduction

Student answers assessment or short text grading is a well-defined problem in Natural Language Processing (NLP). It is a an extremely challenging task as students can express the same answer in multiple ways owing to different individual styles and varied cognitive abilities and knowledge levels. Table 1 shows four answers, articulated by four different college students, to a question asked by the state-of-the-art intelligent tutoring system (ITS) DeepTutor [13]. It should be noted that all four student answers in Table 1 are correct answers to the tutor question. As can be seen from the table, some students write full sentences (student answer A4), some others write very short answers (A3), and yet other students write elaborate answers that include additional concepts relative to the reference answer (A1).

Assessing the freely generated student answers in conversational tutoring can be achieved using various approaches. Semantic similarity is a widely adopted and scalable approach in which the student answer is compared to a reference answer produced by an expert. Typically, a normalized semantic similarity score,

Table 1: Table1. Examples of student answers showing the diversity of responses from DeepTutor

Problem description:
While speeding up, a large truck pushes a small compact car.
Tutor question:
How do the magnitudes of forces they exert on each other compare?
Reference answer:
The forces from the truck and car are equal and opposite.
Student answers:
A1. The magnitudes of the forces are equal and opposite to each other due to Newton's third law of motion.
A2. they are equal and opposite in direction
A3. equal and opposite
A4. the truck applies an equal and opposite force to the car.

from 0 to 1 (or from 0 to 5), between the student answer and the expert answer is generated. A high score implies that the student answer is correct, and a low score implies the student answer is incorrect.

More recently deep learning has shown its effectiveness in solving the students answers assessment task [1][2][9][11]. These deep learning models have the advantage of capturing semantic and syntactic information for the text input. Graph Convolutional Networks, in particular, have received a growing attention recently [4][6]. Graph neural networks have been effective at tasks that have rich relational structure and can preserve global structure information of a graph in graph embeddings.

In this paper, we propose a novel approach based on Graph Convolutional Networks [15], for the students answers assessment task. We construct a DTGrade graph where each node consists of the concatenation of a student answer and its corresponding reference answer. We model the graph with a Graph Convolutional Network (GCN) that encodes relevant information about its neighborhood as a real-valued feature vector. The edge between two nodes is built using word frequency and word's document frequency method and an embedding based method. Then, we turn the assessment task into a node classification task.

The rest of the paper is organized as follows: Section 2 presents a review of several prior research works that used Graph Convolutional Networks for different NLP tasks. Section 3 explains the proposed approach. Section 4 summarizes the conducted experiments to evaluate the performance of our approach and the results obtained on the DT-Grade dataset. Finally, we discuss conclusions and highlight future research directions to improve results and overcome the limitations.

2 Related Work

Graph Convolutional Networks (GCN) have yielded great results in multiple NLP tasks. For instance, Sahu and colleagues [14] proposed a novel inter-sentence relation extraction model that builds a labelled edge Graph Convolutional Network on a document-level graph. The experimental results showed that the model has achieved a comparable performance to state-of-the-art neural models on the inter-sentence relation extraction task. Working on the same task, Zhang and colleagues [17] proposed a novel model for the relation extraction task. Their model consists of the following components: 1) an instance encoder based on convolutional neural networks (CNN) to encode the instance semantics into a vector, 2) a relational knowledge learning component that employs graph convolutional networks to learn explicit relational knowledge, and 3) a knowledge-aware attention component to select the most informative instance that matches the relevant relation. The experimental results showed that this model outperforms several baselines such as CNN. GCNs have been applied successfully as well for the semantic role labeling task that can be described as the task of discovering in texts who did what to whom. To this end, Marcheggiani and colleagues [12] have proposed a model that consists of the following components: 1) word embeddings, 2) a BiLSTM encoder that takes as input the embedding representation of each word, 3) a syntax-based GCN encoder that re-encodes the BiLSTM representation based on the predicted syntactic structure of the sentence, and 4) a classifier to predict the role associated with each word. The empirical results showed that this based GCN model has achieved the state-of-the-art results. GCNs have been explored successfully in text classification. For this purpose, Yao and colleagues[16] proposed to use Graph Convolutional Networks for text classification. They built a single text graph for the whole corpus based on word co-occurrence and document word relations then learnt a Text Graph Convolutional Network for the corpus. The proposed model has been evaluated using multiple benchmarks. The experimental results showed that GCN outperforms several baselines such as Bi-Directional LSTM and LSTM. In this work, we don't consider a heterogenous graph where nodes present words and documents. The nodes represent documents only as a concatenation between student answers and reference answers. Based on these successes of Graph Convolutional Networks on NLP related tasks, we have explored their potential for assessing student answers. To the best of our knowledge, this is the first attempt at using GCNs for assessing student generated answers in conversational intelligent tutoring systems.

3 Proposed Method

Our proposed method consists of building first a graph from the DTGrade. The built graph is fed into two GCN layers. Finally, we apply a classifier to predict the class of each text node.

3.1 DT-Grade Graph

We build a text graph from the DT-Grade dataset based on the citation relation approach [15]. We consider each document, whose content is the combination of the student answer and its corresponding reference answer, as a node. Thus, the classification of a pair of student answer and reference answer turns to a node classification task. The number of the nodes in the text graph is 900 which is the number of instances in the DT-Grade dataset. Formally given a graph G=(V,E)

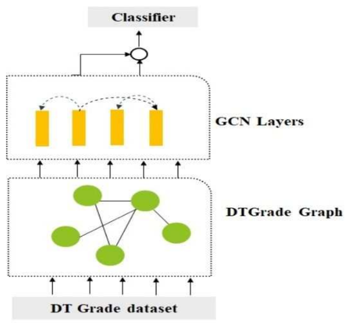

Fig. 1: The model architecture consists of : 1- building a DTGraph 2- feeding it to two GCN layers 3- and finally applying a classifier

where V and E are sets of nodes and edges. The weight of the edge between two nodes is calculated using two methods: a TF-IDF method and an embedding based method. In the first one, we compute the term frequency-inverse document frequency (TF-IDF) between two text nodes. We add an edge between two nodes if the weight is above a threshold of 0.9. The second method is based on word2vec embeddings. First, word2vec is used to learn a vector representation for each word in the text representing each node. Then, we compute the Word Mover's Distance (WMD) to measure the similarity between two texts representing two nodes in the graph. Texts that share many words should have smaller distances than texts with very dissimilar words. WMD has been introduced to measure the distance between two text documents that takes into account the alignments between words. In this paper, we consider the text associated with each node as a short document. The WMD algorithm finds the values of an auxiliary 'transport' matrix T, such that Tij describes how much d_i^a should be transported to . The WMD learns T to minimize:

$$D(x_i, x_j) = min_{T>=0} \sum_{i,j=1}^{n} T_{ij}||x_i - x_j||_2^p \tag{1}$$

Subject to : $\sum_{i,j=1}^{n} T_{ij} = d_i^a, \sum_{i,j=1}^{n} T_{ij} = d_i^b$
where: d_i^a and d_i^b are the n-dimensional normalized bag-of-vectors for the two nodes' texts, $x_i \in R^d$ is the embedding vector of the i^{th} word and p is usually set to 1 or 2. The resulted graph is fed afterwards into a two-layers GCN, as explained next.

3.2 Graph Convolutional Networks (GCN)

GCN is a recent class of multilayer neural networks that operate on graphs [8][15]. For every node in the graph, GCN encodes relevant information about its neighborhood as a real-valued feature vector. Formally given a graph G=(V,E) where V and E are sets of nodes and edges. Every node is assumed to connect with itself, i.e., $(v, v) \in E$ for any v . Let $X \in R^{n \times m}$ be a matrix containing all n nodes with their features, where m is the dimension of the feature vectors, each row $x_i \in R$ is the feature vector for v. We consider A an adjacency matrix of the graph G and its degree matrix D where $D_{ii} = \sum_j A_{ij}$.When using GCN with multiple layers, the information about larger neighbors is captured. Following the recommendation of Kipf et al [15] that multiple layers yield better performance, we consider multiple layers of GCN. The new k-dimensional node feature matrix of layer $L^{(j+1)}$is computed as following:

$$L^{(j+1)} = p(\tilde{A} L^{(j)} W_j) \tag{2}$$

Where $\tilde{A} = D^{-1/2} A D^{-1/2}$ is the normalized symmetric adjacency matrix and W_j is a weight matrix and p is an activation matrix and $L^{(0)} = X$.

3.3 The Classifier

The output of the second GCN layer is fed into a softmax layer as following:

$$Z = softmax(\tilde{A} ReLU(\tilde{A} X W_0) W_j) \tag{3}$$

Where $\tilde{A} = D^{-1/2} A D^{-1/2}$ is the normalized symmetric adjacency matrix, W_j,W_0 are weight parameters and $softmax(x_i) = exp(x_i) \div \sum_i exp(x_i)$. $\tilde{A} X W_0$ contains the first layer document embeddings and
$(\tilde{A} ReLU(\tilde{A} X W_0) W_j)$ contains the second layer document embeddings.

4 Experiments

Our experiments were conducted in the context of student generated answers in response to hints (in the form of questions) in conversational intelligent tutoring systems. To this end, we have used a previously annotated dataset as described next

4.1 DT-Grade dataset

The DT-Grade dataset [3] was created by extracting student responses from logged tutorials interactions between 36 junior level college students and a state of the art ITS. During the interactions, each student solved 9 conceptual physics problems – they had to provide the correct answer and a full justification based on Physics principles. Their answer was evaluated and if the answer was incorrect or incomplete, e.g., a full justification was not provided, a dialogue followed in which the ITS helped the student discover the solution through personalized scaffolding in the form of hints that varied in their degree of information/help provided. Each annotation instance in the DT-Grade dataset consists of the following attributes: (1) problem description (describes the scenario or context), (2) tutor question, (3) student answer (as typed by the students, i.e., without correcting spelling and grammatical errors) and (4) reference answers. In addition, the data includes the correctness class of each student answer. Each student response was categorized by human experts into one of the following four classes: (1) Correct: Answer is correct; (2) Correct-but- incomplete: The response provided by the student is correct, but something is missing; (3) Incorrect: Student answer is incorrect; and (4) Contradictory: The student answer is contradicting with the answer. In this work, we consider only two classes: correct and incorrect. The correct answers are those labeled as "correct" in the DT-Grade dataset. All the other instances are considered "incorrect". As a result, we obtained the following class distribution shown in Table 2 below.

Table 2: The distribution of classes in training (800 instances) and testing data (100 instances)

Dataset	**Correct(%)**	**Incorrect(%)**
Training	41	59
Testing	41.59	58.41

4.2 Experimental Setting

Several experiments have been conducted with different parameters settings to evaluate the performance of our proposed method. To this end, we trained and evaluated a two-layer GCN using the DTGrade dataset. In all experiments, we trained our model for a maximum of 1000 epochs (training iterations) using the Categorical Cross Entropy loss function and Adam optimizer [10] with a learning rate of 0.01. We stopped the training when the validation loss does not decrease for 100 consecutive epochs, as suggested in prior works [15]. To avoid overfitting, we applied a dropout rate=0.5. For the graph convolution layer, we used a hidden layer size of 16 units with L2 regularization and ReLU activation. We selected randomly 600 instances for training, 100 instances for validation, and 200 instances as an independent test set. In the first set of experiments, we have used the TF-IDF approach to compute the weight of the DTGrade graph edges. Then, we repeated the experiment with the following filters: 1) local pool filter [15] which is considered as a baseline filter for Graph Convolutional Networks, 2) Chebyshev polynomial filter [7] and 3) ARMA filter [5]. In a second set of experiments, we have used the word2vec embedding with 300 dimension and WMD distance (see section 3.1) to compute the weight of the edges. We report the accuracy of the model using the three filters.

4.3 Results and Analysis

Table 3 summarizes the results of using GCN with different parameters settings. Several observations can be made. First, the use of the TF-IDF method to compute the weights between the edges outperform the word2vec based method in all experiments. The highest accuracy obtained with TF-IDF was 73% versus 70% of the word2vec method. The performance degradation when using the embedding based approach may due to adding some edges between nodes that are not very related closes. This explains the incorrect assessment of many short students' responses. Added to this, the word2vec embedding based approach may not propagate label information to the whole graph well in comparison with the TF-IDF approach. Second, the empirical results show that ARMA filter outperforms the other polynomial filters regardless the method used for weighting the edges in the DTGraph. This is attributed to the implementation strategy of the ARMA filter that allows better handling of the graph variations. The results depicted in table 3 show also that Graph Convolutional Networks outperform the previous deep learning models: Transformer [2], Bi-GRU Capsnet[1], LSTM and Bi-GRU by obtaining the state of the-art results on the DTGrade dataset. Graph neural networks have been effective at tasks thought to have rich relational structure and can preserve global structure information of a graph in graph embeddings.

Table 3: Table 3. performance of GCN using binary encoding with different filters

Model	**Accuracy**
GCN (TF-IDF+localpool filter)	68
GCN (TF-IDF+ Chebyshev filter)	72
GCN (TF-IDF+ARMA filter)	**73(+0.5)**
GCN (word2vec+WMD+ localpool filter)	62.5
GCN ((word2vec+WMD+chebyshev filter)	70
GCN ((word2vec+WMD+ARMA filter)	70
Transformer Encoder+Elmo	71
Bi-GRU+Glove	56.25
LSTM + Glove	60
Bi-GRU Capsnet+ Elmo	72.5

5 Conclusion

Motivated by good results of applying the Graph Convolutional Networks (GCN) in the NLP, we propose to use a GCN based model to assess the correctness of student answers in conversational intelligent tutoring systems. This is the first time such model is applied for this task. The results demonstrated the effectiveness of the proposed model by yielding state of the-art results on the DT-Grade dataset. A highest accuracy of 73% has been achieved when using the TF-IDF and the ARMA filter. As a future direction, we are planning to explore more novel deep learning models that perform well on a small size of dataset such as ours.

References

1. Ait Khayi,N, Rus, V.: Bi-GRU Capsnet for student answers assessment.In: The 2019 KDD Workshop on Deep Learning for Education (DL4Ed) in conjunction with the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2019). Anchorage, Alaska, USA (2019)
2. Ait Khayi,N, Rus, V.: Attention based Transformer for student answers assessment. In:The Flairs-33rd International Conference. (2020)
3. Banjade, R., Maharjan, N., Niraula, N. B., Gautam, D., Samei, B., Rus, V.:Evaluation dataset (DT-Grade) and word weighting approach towards constructed short answers assessment in tutorial dialogue context. the 11th Workshop on Innovative Use of NLP for Building Educational Applications (pp. 182-187).(2016)
4. Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., ... ,Gulcehre, C.:Relational inductive biases, deep learning, and graph network. arXiv preprint arXiv:1806.01261.(2018)
5. Bianchi, F. M., Grattarola, D., Alippi, C., Livi, L.: Graph neural networks with convolutional ARMA filters. arXiv preprint arXiv:1901.01343 (2019)
6. Cai, H.; Zheng, V. W.; and Chang, K.:A comprehensive survey of graph embedding problems, techniques and applications.In:IEEE Transactions on Knowledge and Data Engineering 30(9):1616– 1637.(2018)
7. Defferrard, M., Bresson, X., Vandergheynst, P. : Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in neural information processing systems (pp. 3844-3852) (2016)
8. Duvenaud,D.,Maclaurin,D., Aguilera-Iparraguirre,J., Bombarell,R., Hirzel,T., Aspuru-Guzik,A., Ryan P.: Convolutional networks on graphs for learning molecular fingerprints. In: NIPS.(2015)
9. Gong, T., Yao, X.: An Attention-based Deep Model for Automatic Short Answer Score. International Journal of Computer Science and Software Engineering, 8(6), 127-132 (2019)
10. Kingma, D. P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. (2014)
11. Maharjan, N., Gautam, D., Rus, V. : Assessing free student answers in tutorial dialogues using LSTM models. In: International Conference on Artificial Intelligence in Education (pp. 193-198). (2018)
12. Marcheggiani, D.,Titov, I.: Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In: EMNLP.(2017)
13. Rus, V., D'Mello, S. K., Hu, X., Graesser, A. C.: Recent advances in intelligent tutoring systems with conversational dialogue. AI Magazine, 34(3), 42-54(2013)
14. Sahu, S.K., Christopoulou, F., Miwa, M.,Ananiadou, S. : Inter-sentence Relation Extraction with Document-level Graph Convolutional Neural Network. In: ACL.(2019)
15. Kipf,T., Welling,M. : Semi supervised classification with graph convolutional networks. In ICLR.(2017)
16. Yao, L., Mao, C., Luo, Y. :Graph convolutional networks for text classification. In : the AAAI Conference on Artificial Intelligence (Vol. 33, pp. 7370-7377). (2019)
17. Zhang, N., Deng, S., Sun, Z., Wang, G., Chen, X.D., Zhang, W., Chen, H.:Long-tail Relation Extraction via Knowledge Graph Embeddings and Graph Convolution Networks. In:NAACL-HLT.(2019)