Using Neural Tensor Networks for Open Ended Short Answer Assessment

Dipesh Gautam and Vasile Rus(✉)

The University of Memphis, Memphis, TN 38152, USA
{dgautam,vrus}@memphis.edu

Abstract. In this paper, we present a novel approach to leverage the power of Neural Tensor Networks (NTN) for student answer assessment in intelligent tutoring systems. The approach was evaluated on data collected using a dialogue based intelligent tutoring system (ITS). Particularly, we have experimented with different assessment models that were trained using features generated from knowledge graph embeddings derived with NTN. Our experiments showed that the model trained with the feature vectors generated with NTN, when trained with a combination of domain specific and domain general triplets, performs better than a previously proposed LSTM based approach.

Keywords: Knowledge graph · Neural Tensor Network · Answer assessment · Open ended short answer assessment · Entity vector embedding

1 Introduction

Natural language understanding is the foundation of assessment in conversational ITSs and other educational technologies that elicit freely generated natural language responses. Typically, automatic answer assessment methods measure the extent to which a given student answer or parts of it related or match some target/benchmark concepts. These benchmark or expected concepts are specified by subject matter experts and other experts (e.g., experts in pedagogy or linguistics). If the student answer or parts of it are semantically similar to the target (reference) concepts then the student response is deemed correct; otherwise, it is deemed incorrect. Semantic similarity methods can be categorized as either knowledge based, such as methods that rely on WordNet for computing similarity among concepts, versus corpus based, such as Latent Semantic Analysis (LSA) [10] and Latent Dirichlet Allocation (LDA) [4]. Another category of methods use a combination of knowledge based and corpus based methods [16,20].

There is a major limitation of similarity based assessment methods: they assume the student answer and the reference answer are self contained. Most often, the student responses are elliptical, contain anaphoras, or depend heavily on a broader context such as the instructional task description or prior dialogue

I. I. Bittencourt et al. (Eds.): AIED 2020, LNAI 12163, pp. 191–203, 2020.
https://doi.org/10.1007/978-3-030-52237-7_16

Table 1. An example of student tutor conversation in DeepTutor

Q: *What forces are acting on the puck while the puck is moving on the ice between the two players?*
A1: *The forces acting on the puck are the gravitational force and the normal force from the ice*
A2: *Normal and gravity*
A3: *The* ***downward force from the earth*** *and the normal force from the ice*
E: *The forces acting on the puck are the downward force of gravity and the upward normal force from the ice*

turns (dialogue history) in the case of task-oriented conversational ITSs. For example, in Table 1, student answer A1 is quite self-contained; a semantic similarity approach would lead in this case to a high similarity score to the expected answer (E).

On the other hand, some correct short answers could be elliptical (see answers A2 and A3 in the table) and computing a semantic similarity score between such elliptical answers and the references answer is a challenge simply because the elliptical, shorter answers have many implied parts which a typical semantic similarity approach would fail to account for as such approaches rely mostly on explicitly specified information, i.e., words in this case. The problem with assessing such elliptical answers using a standard semantic similarity approach is that it leads to a low similarity score between the elliptical responses and the expected answers, thus incorrectly assessing elliptical responses even if they are correct.

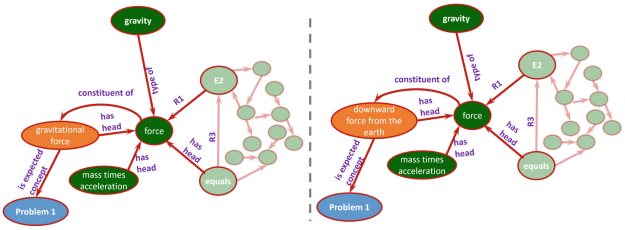


Fig. 1. Portions of knowledge graph that show concepts "gravitational force" (left) and "downward force from the earth" (right)

To address this issue, we propose a knowledge graph based approach by representing the concepts in the student answers and reference answers using embedded vectors that are learned directly from a knowledge graph. The embedded vectors encode indirect relationship between concepts, e.g., they can account for implicit information among concepts in the student answer and the benchmark

answer. To this end, we construct a knowledge graph by extracting concepts and their surface relations from reference answers and then train a Neural Tensor Network (NTN; [22]). The idea here is that once the NTN is trained, these concept vectors encode relationships among entities/concepts in the knowledge graph - the more two entities share same or similar neighbors and relations with those entities, the more similar their vector representations are. For instance, in Fig. 1, the entity "gravitational force" and "downward force from the earth" are more likely to have similar vector embeddings since they share same neighbors (*force* and *problem 1*) and relation types (*constituent of*, *has head* and *is expected concept*).

2 Related Works

Knowledge graphs containing concepts or entities and their relations are important knowledge resources that have been used successfully for various applications such as question answering and information retrieval. However constructing knowledge graphs from unstructured data such as text is challenging. There have been a number prior efforts such as [1,5,9] to extract knowledge graphs from the text. These efforts employed classification approaches to classify whether an entity participates in a particular relation or not. The output of those methods is in the form of triplets specifying two entities and the corresponding relations among those entities (entity 1, relations, entity 2). The OpenIE tool [1] is an example of such an information extraction software system that outputs triplets from a given input text[1].

Often such knowledge graphs do not specify all possible relations between entities and in general they lack reasoning capabilities to infer the unspecified relations. That is, there are many true relations among entities in a given knowledge graphs that are not explicitly encoded in the graph. The task of explicitly inferring those missing relations is called the knowledge completion task. Several attempts have been made to complete knowledge graphs with missing relations among their entities.

One such approach to the task of knowledge completion in knowledge graphs relies on relational learning and was proposed by Nickel and colleagues [18]. They used a tensor model representation of relational data and developed RESCAL, an approach that employs tensor factorization to factorize the tensor obtained from relational data. This approach is comparable to LSA with two dimensional matrices representing relation between entities. However, in the RESCAL approach the representation of relations with three dimensional (3-D) matrices make it possible to have multiple relationships between entity pairs.

Socher and colleagues [22] proposed a neural network approach to represent relations with neural network. They developed a method to represent entities as vectors and relations as neural tensor networks (NTN), a variant of neural networks which combines a feed forward model with a bi-linear tensor product. The parameters of such NTN encode the latent relationship between the entities.

[1] https://nlp.stanford.edu/.

One of the important aspects of NTN that attracted our attention towards using the model in answer assessment is that it learns entity embeddings for each concepts as a vector that inherently encodes the relationship with the other entities. Such embeddings of concepts could help infer implied relationships and concepts in knowledge graphs corresponding to student answers. Our work relies on classification method for which concepts in answers are represented by embedding vectors learned while training Neural Tensor Network similar to that proposed by Socher and colleagues [22].

To our knowledge, our work is the first attempt to use knowledge graphs and a knowledge completion mechanism for automated answer assessment. In the past decade, automated assessment systems [6,11,21] were developed for texts of various sizes and generated with different purposes in mind. For instance, SAT-style argumentative essays have a well-defined structure and are 3–5 paragraphs in length on average. On the other hand, in problem-solving conversational tutoring systems students generate short answers in the form of dialogue turns while working with the tutoring system to solve a given problem. Unlike the essay grading, which focuses more on style, coherence, and organization of ideas, the short answer assessment task focuses more on assessing the correctness of the student response. Ziai and colleague [23] pointed the need of publicly available good quality dataset that could arguably enable comparison of such systems that are designed for different purposes. To this end, we focus here on the latter task of short answer assessment and compare result with previous works such as [13,14] that were proposed for same problem as this work. In the past, Latent Semantic Analysis, for instance, was used [7,17] for short answer assessment. However, LSA is an algebraic method that relies on word co-occurrence analysis of large collections of naturally occurring texts and it cannot account for linguistic phenomena such as anaphora resolution which is quite frequent in tutorial dialogues as explained next. While analyzing tutorial dialogues in a dialogue based tutorial system, Niraula and colleague [19] found that a significant portion of student answers contain pronoun that refer entities in the previous utterances. Methods to address such problems were proposed at different times such as [2,3,12,13]. In their methods, they assume that the question and the problem description provide import contextual cues for elliptic answers. In our case, when generating knowledge graphs, pronouns are solved to their corresponding referents.

3 Methods

Our assessment system is based on a multi-class classifier that classifies a student answer into one of the four assessment labels: (i) correct, (ii) correct but incomplete, (iii) incorrect and (iv) contradictory. For this, we extract entities and relations from student answers and reference answers and obtain embedding vector of these entities. In the following sections, we discuss in detail the steps of knowledge graph construction, entity embeddings, and assessment models.

3.1 Entity Relations Extraction

In order to construct the knowledge graph, a large collection of entities and relations triplets are needed. These triplets could then be used to learn latent (implied) relationships and thus discover missing, valid links between the entities. In our work, we use two categories of such entity relations: (i) semantic relations obtained from WordNet [22] and (ii) surface relations defined and extracted from the domain dataset, i.e., the DT-Grade dataset (see later).

While extracting surface relation triplets, we assume that there are a finite number of problems that are authored for training with a given intelligent tutoring system. An entity could be a token, a text chunk, or an unique identification number of the problem. The token entities are obtained by tokenizing the reference answers. From those tokens, we keep only content words such as nouns, verbs, adverbs and adjectives as entities. The text chunks are obtained from dependency parse trees. We used SpaCy [8] for text parsing. Besides that, other kind of entities and binary relationships are extracted using OLLIE [15], a state-of-art tool for information extraction.

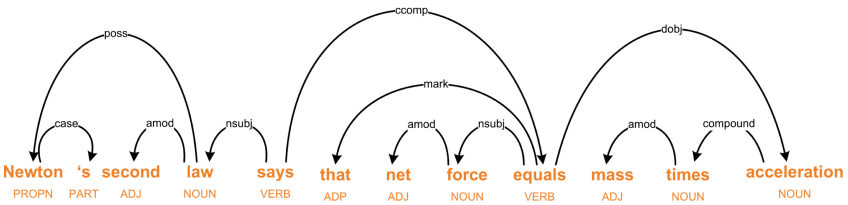


Fig. 2. Example of a sentence parsed with SpaCy dependency parser

In addition to extracting phrases, the dependency parse tree provides a way to obtain syntactic relations between entities. For instance, from Fig. 2 we can obtain several possible relations between entities. We define the following five relation types:

1. **is concept of:** if an entity is an expected concept of a problem. A problem is an abstract entity that represents problem's unique identification number ("*Problem 1*" is an abstract entity; Fig. 1).
2. **is constituent of:** if an entity is constituent of another entity; i.e., if an entity is a part of another entity ("*force*" is constituent of "*gravitational force*"; Fig. 1)
3. **has head text:** if a noun phrase's head word is another entity according to the dependency parse tree.
4. **has ancestor text:** if an entity's ancestor is another entity according to the dependency parse tree.
5. **has child text:** if an entity's child is another entity according to the dependency parse tree.

3.2 Knowledge Graph Embedding

A collection of entity-relation triplets forms a knowledge graph. Such graphs are usually extracted from explicit information in texts. Many valid relationships among the entities in the graph are not explicitly mentioned in those graphs. This is known as the knowledge incompleteness property. Among several approaches proposed previously, we used Neural Tensor Network (NTN) proposed by Socher and colleagues [22], which learns the connection strength between entity pairs, hence discover missing links. The NTN architecture consists of a bilinear tensor layer as well as feed forward layer which makes NTN powerful by harnessing the power of both bilinear and feed forward networks. Here, we present a high level architecture (Fig. 3) of a typical Neural Tensor Network and the scoring function (see Eq. 1) that is originally used in the original paper by Socher et al. Several NTN units (equal to the number of relation types) trained in unison produces a knowledge graph embedding. Since the errors from each unit (i.e error for each relation type) are aggregated while training, the weights of each cell affect each other during training. In other words, the whole knowledge graph represented by neural tensor network gets updated. While after training, the weights of these NTN embed the relation between entities, the connection strength of two entities in the knowledge graph is given by the score function shown in Eq. 1.

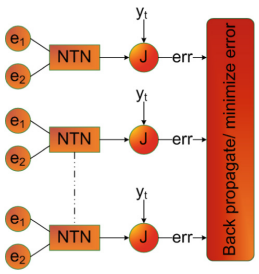


Fig. 3. High level architecture of knowledge graph embedding derived using NTN.

$$g(e_1, R, e_2) = U_R^T f\left(e_1^T W_R^{[1:k]} e_2 + V_R \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + b_R\right) \tag{1}$$

where $e_1, e_2 \in \mathbb{R}^d$ are d dimensional vectors of entities, $f = tanh$, is a non-linear activation function, $W_R^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ is a tensor and the bilinear tensor product $e_1^T W_R^{[1:k]} e_2$ results in a vector $h \in \mathbb{R}^k$, where each entry is computed by one slice $i = 1, ..., k$ of tensor: $h_i = e_1^T W_R^i e_2$. The other parameters for relation R are the standard form of a neural network: $V_R \in \mathbb{R}^{k \times 2d}$ and $U \in \mathbb{R}^k, b_R \in \mathbb{R}^k$.

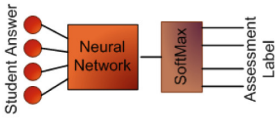
To train such NTN, the entity relation triplet such as *"(net force, has head text, equal)"* are labeled as true relation and negative examples such as *"(net force, has head text, friction)"*, created by corrupting one of the entities in each of the positive relation triplets are labeled as false. Then, such negative and

positive triplets with corresponding binary labels are used to train the NTN. While training, the network updates its weights as well as the entity vector to obtain better representation of each of the entity after each epoch. The vectors produced as bi-product are useful in our answer assessment method.

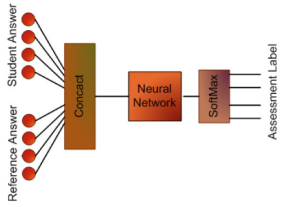
3.3 Classifier Using Entity Embedding

Using the entity embeddings obtained after training with NTN, we construct vectors by extracting entities from an answer instance and averaging the entity vectors to get a single vector for the answer instance. We obtain such vectors for both the student answer as well as the reference answer. While computing the average of vector entities, out of vocabulary entities in student answers need to be handled. We address this problem by replacing such out of vocabulary entities with the vectors of potential synonyms or one of its constituents, if the case. If none exists, we simply use the "*NONE*" word vector.

Once the vectors of the student and reference answers are obtained, we feed them onto a classifier. Indeed, our assessment model is a classifier that categorizes the student answer into one of the classes that represent the assessment labels. We used two types of classifiers based on neural networks. The first type is a simple neural network with one input (Fig. 4a), the vector of the student answer. The second type concatenates the reference answer vector and the student answer vector (Fig. 4b). The advantage of the classifier with two input vectors is its ability to learn by comparing the student answers with standard reference answers during training. In other words, such a classifier learns to distinguish between a good answer that is semantically close to the reference answer and incomplete or incorrect answers which are not semantically close to the reference answer. Additionally, the reference answers are generally self contained and complete, hence they can provide contextual cues to the student answer, when used together.



(a) Student answer as input



(b) Student answer and reference answer as input

Fig. 4. Student answer classifier

Compared to one input classifier, training and predicting with the two input classifier is different when there are multiple possible reference answers (usually, paraphrases of each other) for same problem. For training, those reference

answers, paired with corresponding student answers produce a larger number of training examples, an advantage over the one input classifier. However, while predicting, multiple pairs with same true label but different predicted labels could be possible for a single instance (student answer). In such situations, a majority vote strategy is used to select the predicted assessment label; i.e., the assessment label that is predicted most frequently for a student answer is selected as the final predicted label.

4 Experiments and Results

We performed experiments with two different types of classifiers using entity vectors learned with NTN trained with both semantic (domain general) and surface (domain specific) relation triplets. The two types of classifiers, one input and two inputs trained with different entity vectors obtained from various triplet sources, are shown in Table 2. The domain general triplets are obtained from WordNet relations (prefixed with "WN") whereas the domain specific triplets are obtained from DT-Grade dataset (prefixed with "DT"). We also performed experiments by augmenting the domain general triplets with domain specific triplets (prefixed with "Aug"). For augmentation, we combined the domain general entities and relation obtained from WordNet with entities and triplets obtained from the DT-Grade dataset. In the following sections, we first describe datasets and then present the results obtained in various experimental setups.

Table 2. Experimental models

One input classifier	Two input classifier	Triplet source
WN1IP	WN2IP	WordNet
DT1IP	DT2IP	DT-Grade
Aug1IP	Aug2IP	Augmented (WordNet & DT-Grade combined)

4.1 Dataset

Tutorial Dataset. We used the DT-Grade dataset [3] which contains instances in the form of student answer - ideal answer pairs extracted from logged tutorial interaction of 40 junior level college students and a state-of-the-art intelligent tutoring system. The instructional tasks were conceptual physics problems. The dataset consists of 900 instances. The student responses were labeled with the following four assessment labels (shown in Table 3).

Table 3. DT-Grade dataset

Labels	Description	Distribution
Correct	Covers all the expected concepts	367 (40.77%)
Incomplete	Covers some of the expected concepts	211 (23.44%)
Contradictory	Semantically opposite or contrast to expected answer	84 (9.33%)
Incorrect	Does not include any of the expected concepts	238 (26.44%)
	Total	900

Knowledge Graph Dataset. We used the WordNet knowledge graph dataset described by Socher and colleagues [22]. We preprocess the WordNet triplets to combine the different senses for same word into a single entity for training our neural tensor network. Though the different senses are combined, the relations that those different senses previously participated in was kept unchanged and treated as a separate training instance. This makes the model simple yet enabling the encoding of the relations in the embedding. There are 11 relations categories obtained from WordNet, 33,163 entities, and 109,165 relationship triplets. These categories characterize the semantic relations between the entities in the knowledge graph. Additionally, we created an entity relation triplets dataset from the reference answers in the DT-Grade dataset. The entities we created are of two types: (i) the question itself is the entity, i.e. there are 900 such entities and (ii) the content words, phrases, head words, parents and children obtained by parsing the reference answers using the SpaCy [8] dependency parser. Encoding question as an entity provides contextual information such as the relation *"is concept of"* (see Sect. 3.1) to the knowledge graph. After obtaining the entities, we identified 5 syntactic relations among the entities obtained from reference answers, with 1,263 entities and 22,941 relation triplets. We used these two categories of knowledge graph datasets separately as well as augmenting the syntactic knowledge graph dataset by combining with the semantic knowledge graph dataset.

4.2 Results

The results of 10-folds cross validation training-testing process is summarized in Table 4. We report the performance in terms of accuracy and F1 measure. The result shows that Aug2IP performed best with an average accuracy of 0.644, which is 2.2% better than *LSTM, the previously best performing model (0.622) [13]. Also its F1 score (= 0.642) is 2.2% and Kappa (= 0.482) is 3.2% better, respectively, than that of *LSTM. The *LSTM used the problem description, tutor question, student answer, and reference answer as input, however, and relied on one-hot-encoding inputs for entities to discover general semantic and domain specific linguistic relationships. In fact, our two inputs classifier when used with domain specific vectors (DT2IP & Aug2IP) performed better. This suggests that the NTN model could learn vectors better than the word2vec used

in previous approach. The result aligns with our expectation that the knowledge graph inferred with NTN can encode the latent relations between entities.

Table 4. Performance of models

Model	Avg acc	F1	Kappa
*LSTM [13]	0.622	0.620	0.450
DT2IP	**0.626**	**0.624**	0.450
Aug2IP	**0.644**	**0.642**	0.482
WN2IP	0.564	0.569	0.334
DT1IP	0.569	0.569	0.350
Aug1IP	0.604	0.604	0.409
WN1IP	0.551	0.565	0.302

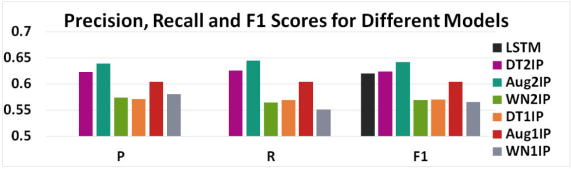


Fig. 5. Comparison of precision, recall and f1 score for different models

Besides performing better than previous model, the result suggests that when trained with vectors created from the same dataset, the classifiers that takes both student answer and reference answer as input perform better compared to models that only take student answer as input. For instance, DT2IP has average accuracy of 0.626 which is 5.7% higher than of DT1IP. Similarly Aug2IP has average accuracy of 0.644 which is 4% higher than Aug1IP. Whereas the performance of WN2IP is higher than that of WN1IP, it is a small improvement (1.3%) when compared to other classifiers.

Table 4 further shows that the classifier when trained on domain specific vectors (prefixed with DT) perform better than when trained on domain general vectors prefixed with WN). Moreover, when the domain specific triplets were augmented with domain general triplets, the performance boosted up significantly (3.5% improvement for Aug1IP than DT1IP, and 1.8% for Aug2IP than DT2IP).

Figure 5 shows the average precision, recall, and F1 score of various models we experimented with. As seen from the figure, our two input classifier trained with augmented vectors performed best in terms of precision (0.639), recall (0.644) and F1 score. Compared to domain specific vectors (DT1IP and DT2IP) the

domain general vectors(WN1IP and WN2IP) performed worse. The reason could be because a significant number of the entities extracted from student answers and reference answers from DT-Grade dataset were not present in the domain general vocabulary. And lack of such entities resulted in inaccurate representation of entities. Because such entities need either semantically similar entities or synonym words instead of relying on NONE entity in the vocabulary.

5 Conclusions

In this paper, we proposed several knowledge graph based models to assess freely generated student responses with a focus on short responses generated in tutorial dialogues. The improved performance in terms of accuracy and F1 measures of the propose models suggests knowledge graph based models yield better vectorial representation of student answer and reference answer texts. In addition, the two input classifier always performed better than the one input classifier when trained with the same set of vectors. This is expected, since the two input classifier uses the reference answer as input as well. More importantly, when the two input classifier is trained with augmented vectors, they performed best. This suggest that the relation triplets obtained from actual tutorial data helps to encode highly predictive features when training with NTN. In summary, the NTN model learns entity vectors that help to represent concepts and relations some of which are not explicitly mentioned and therefore benefit methods for answer assessment such as the one we propose here.

Our method has several areas where further improvement is possible. One of those areas is to define more relations among entities that are specific to a target domain, e.g., physics. In this work, we have limited ourselves to general, syntactically-derived relations. In the future, we plan to integrate methods that can automatically discovering domain specific relations from free text, for instance.

Acknowledgements. This research was partially sponsored by the National Science Foundation under award The Learner Data Institute (award #1934745), NSF award Investigating and Scaffolding Students' Mental Models during Computer Programming Tasks (award # 1822816) and an award from the Department of Defense (U.S. Army Combat Capabilities Development Command - Soldier Center). The opinions, findings, and results are solely the authors' and do not reflect those of the funding agencies.

References

1. Angeli, G., Premkumar, M.J.J., Manning, C.D.: Leveraging linguistic structure for open domain information extraction. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), vol. 1, pp. 344–354 (2015)

2. Bailey, S., Meurers, D.: Diagnosing meaning errors in short answers to reading comprehension questions. In: Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications, pp. 107–115. Association for Computational Linguistics (2008)
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. In: Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 182–187 (2016)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
5. Chan, Y.S., Roth, D.: Exploiting background knowledge for relation extraction. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 152–160. Association for Computational Linguistics (2010)
6. Dikli, S.: An overview of automated scoring of essays. J. Technol. Learn. Assess. **5**(1) (2006)
7. Graesser, A.C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Tutoring Research Group Tutoring Research Group, Person, N.: Using latent semantic analysis to evaluate the contributions of students in AutoTutor. Interact. Learn. Environ. **8**(2), 129–147 (2000)
8. Honnibal, M., Montani, I.: spaCy 2: natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing (2017, To appear)
9. Jiang, J., Zhai, C.: A systematic exploration of the feature space for relation extraction. In: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference, pp. 113–120 (2007)
10. Landauer, T.K.: Latent Semantic Analysis. American Cancer Society (2006). https://doi.org/10.1002/0470018860.s00561. https://onlinelibrary.wiley.com/doi/abs/10.1002/0470018860.s00561
11. Leacock, C., Chodorow, M.: C-rater: automated scoring of short-answer questions. Comput. Humanit. **37**(4), 389–405 (2003). https://doi.org/10.1023/A:1025779619903
12. Maharjan, N., Banjade, R., Rus, V.: Automated assessment of open-ended student answers in tutorial dialogues using Gaussian mixture models. In: The Thirtieth International Flairs Conference (2017)
13. Maharjan, N., Gautam, D., Rus, V.: Assessing free student answers in tutorial dialogues using LSTM models. In: Penstein Rosé, C., et al. (eds.) AIED 2018. LNCS (LNAI), vol. 10948, pp. 193–198. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93846-2_35
14. Maharjan, N., Rus, V.: A concept map based assessment of free student answers in tutorial dialogues. In: Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., Luckin, R. (eds.) AIED 2019. LNCS (LNAI), vol. 11625, pp. 244–257. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23204-7_21
15. Schmitz, M., Bart, R., Soderland, S., Etzioni, O.: Open language learning for information extraction. In: Proceedings of Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL) (2012)

16. Mohler, M., Bunescu, R., Mihalcea, R.: Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 752–762. Association for Computational Linguistics (2011)
17. Mohler, M., Mihalcea, R.: Text-to-text semantic similarity for automatic short answer grading. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, pp. 567–575. Association for Computational Linguistics (2009)
18. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: ICML, vol. 11, pp. 809–816 (2011)
19. Niraula, N.B., Rus, V., Banjade, R., Stefanescu, D., Baggett, W., Morgan, B.: The DARE corpus: a resource for anaphora resolution in dialogue based intelligent tutoring systems. In: LREC, pp. 3199–3203 (2014)
20. Pérez, D., Gliozzo, A.M., Strapparava, C., Alfonseca, E., Rodriguez, P., Magnini, B.: Automatic assessment of students' free-text answers underpinned by the combination of a BLEU-inspired algorithm and latent semantic analysis. In: FLAIRS Conference, pp. 358–363 (2005)
21. Shermis, M.D., Burstein, J.C.: Automated Essay Scoring: A Cross-Disciplinary Perspective. Routledge, Abingdon (2003)
22. Socher, R., Chen, D., Manning, C.D., Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In: Advances in Neural Information Processing Systems, pp. 926–934 (2013)
23. Ziai, R., Ott, N., Meurers, D.: Short answer assessment: establishing links between research strands. In: Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, pp. 190–200. Association for Computational Linguistics (2012)