# BI-GRU Capsule Networks for Student Answers Assessment

Nisrine Ait Khayi
Department of computer science
University of Memphis/IIS
Memphis TN USA
ntkhynyn@memphis.edu

Vasile Rus
Department of computer science
University of Memphis/IIS
Memphis TN USA
vrus@memphis.edu

## ABSTRACT

Motivated by the good results of capsule networks in text classification and other Natural Language Processing tasks, we present in this paper a Bi-GRU Capsule Networks model to automatically assess freely-generated student answers assessment within the context of dialogue-based intelligent tutoring systems. Our proposed model is composed of several important components: an embedding layer, a Bi-GRU layer, a capsule layer and a SoftMax layer. We have conducted a number of experiments considering a binary classification task: correct or incorrect answers. Our model has reached a highest accuracy of 72.50 when using an Elmo word embedding as detailed in the body of the paper.

## KEYWORDS

Capsule networks, deep learning, intelligent tutoring systems, semantic similarity.

## 1 Introduction

Automatically assessing open-ended short student responses is an extremely challenging task as students can express their responses in numerous ways owing to different individual styles and varied cognitive abilities and knowledge levels. This assessment plays a vital role in improving the tutoring experience. The system provides hints and feedbacks for the struggling students with incorrect answers. 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 (Rus et al., 2013). 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).

The widely adopted and scalable approach to assessing such open-ended student responses is semantic similarity in which a score, usually normalized, is computed between a target student answer and an expert-provided reference answer (Banjade et al., 2016). If the student answer has a high semantic similarity score to the reference answer we infer that the student answer has the same correctness value as the reference answer. A low semantic similarity score implies the student response is incorrect. It should be noted that sometimes the reference answer may denote a common misconception in which case a high-similarity score to such misconception means the student answer also indicates a misconception, i.e., it is incorrect.

**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.*

**Table 1. Example of students' answers**

More broadly, the task of computing the semantic similarity of two texts consists of determining, both quantitatively (e.g., normalized score between 0 and 1) or qualitatively (are the two texts in a paraphrase, elaboration, or entailment relation) the degree of similarity between the two texts. It is a widely used step in many Natural Language Processing (NLP) applications such as text summarization (Wong et al., 2008; Nenkova et al., 2011), question answering (Vo et al., 2015) and machine translation (Corley and Mihalcea, 2005). It should be noted that we can distinguish among semantic similarity tasks and methods at various granularity levels: word-to-word similarity (w2w), phrase-to-phrase(ph2ph), sentence-to-sentence (s2s), paragraph-to-paragraph (p2p), and document-to-document (d2d) similarities.

Several approaches have been proposed to automatically assess the semantic similarity of short, sentence-level texts, which are our focus. For instance, recently, NLP researchers have applied extensively deep learning models, which have the advantage of not needing hand-crafted features and other external resources, that is, just the raw sentences and the corresponding pre-trained word embeddings are needed as input. Despite of this limited-resource approach, many deep learning methods have provided state of art of performance. For example, Pontes and colleagues (2018) proposed a deep learning model that combines convolution and recurrent neural networks to measure the semantic similarity of sentences. This combination of networks has been helpful in capturing the most relevant information of sentences, thus, improving the computation of semantic similarity scores relative to state-of-the-art systems. Other approaches worth-mentioning are: (1) the Non Linear Similarity approach (Tsubaki et al., 2016), where word representations are inferred through the similarity learning of sentences in high-dimensional space with kernel functions, (2) Constituency Tree LSTM (Tai et al., 2015) which is a generalization to LSTMs to tree-structured network topologies, and (3) Skip-thought (Kiros et al. 2015), where an encoder-decoder model is used to reconstruct the surrounded sentences. Then, sentences with common semantic and syntactic properties are mapped to similar vector representations.

Furthermore, Bao et al. (2018) proposed an Attention Siamese Long Short-Term Memory (LSTM) model to measure the semantic textual similarity. An attention mechanism has been used to capture the high-level semantic information. The empirical experiments have demonstrated the effectiveness of the model with an impressive performance.

Wang and colleagues (2018) presented an approach that combines a Bidirectional Long Short-Term Memory Networks (BLSTM) and Convolutional Neural Networks to extract the semantic features of a sentence. Then sentence representations are learned with word-level attention. Finally, an output layer that calculates the similarity score was used. This proposed model was evaluated using the Quora duplicate questions public dataset. The obtained results showed that this model has outperformed many existing approaches, such as Support Vector Machine (SVM), Convolutional Neural Networks (CNN), Bidirectional Long Short Term Memory (BLSTM) and attention based BLSTM, with a highest accuracy of 0.89.

Our approach is very different from these approaches except the fact that uses Deep Learning.

Our task of automatically assessing freely generated student answers within a dialog system context is a special case of the more general semantic similarity task. As shown in Figure 1, given two inputs, the student answer and the reference answer, the assessment model computes the correctness of the student answer. Typically, the reference and student answer are domain specific as tutoring targets specific science topics, e.g., Physics. Furthermore, the answers are generated in the context of problem-solving instructional activities in which students are asked to provide solutions to various problems in the form of short essays, the essays are evaluated and if incorrect and/or incomplete a tutorial dialogue follows in which students provide short answers to tutoring systems' hints. For this work, we don't capture domain specific information. This can be addressed in a future work.

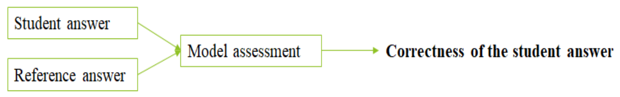

**Figure 1. students' answers assessment problem statement**

Motivated by the good results of deep learning models in similar semantic similarity tasks, we present in this paper a Bi-GRU Capsnet model to assess the students answers generated during student-system dialogue-based interactions. Capsules have the capability to express the semantic meanings in a wider space using a vector instead of a scalar. Thus, these capsules are suitable to express a sentence as a vector (Kim. J et al.,2018). This generated vector captures the instantiation parameters of the input such as the order of the words and their semantic representation. On the other hand, word embeddings also transform words into lower dimensional vectors that preserve the contextual similarity of words. In general, the embedding vectors are fed into various deep learning models. Our model consists of several important components. First, there is an embedding layer that transforms each word of the input to a distributed vector representation. Second, the resulted embedding matrix is fed into a Bidirectional Gated Recurrent Units layer (Bi-GRUs) (Cho et al., 2014) to encode the input text into a fixed length representation. The fixed length representation is then fed into a capsule network. Finally, the capsule network is followed by a fully connected dense layer with SoftMax activation for the classification. We evaluate the performance of our model using the DT-Grade (Banjade et al., 2016) corpus.

The paper is organized as following: paragraph 2 represents the related research work. The next section describes the model architecture and its components. Section 4 depicts the conducted experiments and results. The final section concludes the paper, summarizing the main contribution of this work and the possible directions to improve the current results.

## 2 Related work

Capsule networks have been introduced by Geoffrey Hinton for image classification to overcome the limitations of the Convolution Neural Networks particularly in the pooling layer. These networks are based on so called capsules and are trained using a dynamic routing algorithm (Sabour et al., 2017). Each capsule encodes a particular feature (e.g. local order of words, semantic representations of words) that the network is looking for. The magnitude of a capsule vector defines the probability of the existence of that feature. The layers of capsule networks are connected via computing a learned vector between each pair of capsules. Then, the routing algorithm is used to ensure that the output of the capsule, which is a vector, gets sent to an appropriate parent in the layer above. The capsule computes a "prediction vector" for each possible parent. This prediction vector is calculated by multiplying the capsule 's own output by a weight matrix. A top-down feedback is applied, in case the prediction vector has a large scalar product with the output of a possible parent. This is done to increase the coupling coefficient for that parent and decrease it for other parents. In sum, this iterative routing process decides the credit attribution between the nodes in lower and higher levels. Recently, several NLP researchers have applied Capsule Networks for various tasks such as text classification and sentiment analysis. The obtained results were very impressive and encouraging to further investigate these networks in related tasks.

Zhao and colleagues (2018) used capsule networks with dynamic routing algorithm for text classification. To boost performance, they have applied three different strategies to stabilize the dynamic routing process by decreasing some noise capsules. First, an Orphan category has been added to the network to capture the background information such as stop words and the words that are unrelated to specific categories. Second, a Leaky-SoftMax approach has been used to update the connection strength between the parent capsules and their children. Third, the connection strength has been amended using the probability of the existence of the child capsules. To evaluate the performance of the proposed approach, they have conducted several experiments using six different datasets. The obtained results demonstrated the effectiveness of capsule networks over many baseline methods. Our approach is similar in the sense that we model the student answer assessment task as a text classification task. However, the architecture of our proposed model is different. In fact, Zhao and colleagues' model consists of a convolutional layer after the embedding layer and our proposed model consists of a BI-GRU layer instead.

Kim and colleagues (2018) have applied capsule networks for text classification. They have used a simple dynamic routing algorithm to boost the efficiency of the model. Their proposed model consists of the following components: (1) an embedding layer, (2) a feature map that use convolutions, (3) a convolutional capsule layer, and (4) a text capsule layer. The authors have conducted several experiments using different datasets. The results demonstrated the potential of the application of the capsule networks in the text classification. This approach is similar to our work in the sense of considering the student answer assessment task as a text classification task. The main difference is using a BI-GRU layer instead of convolutions after the embedding layer.

Capsule networks have been applied successfully in other NLP tasks. Zhang and colleagues (2018) proposed a relation extraction approach based on capsule networks with attention mechanism. Wang and colleagues (2018) presented an attention-based BI-GRU-CapsNet model to detect hypernymy relationship between compound entities. Xia and colleagues (2018) proposed two capsule-based architectures to solve the zero-shot intent detection problem: the INTENT-CAPSNET that extracts semantic features from utterances and aggregate them to discriminate existing intents, and INTENTCAPSNET to discriminate emerging intents via knowledge transfer from existing intents.

Based on these successes of capsule networks on related tasks, we have explored their potential for assessing student answers. To the best of our knowledge, this is the first attempt at using capsule networks for assessing student generated answers in conversational intelligent tutoring systems.

## 3 Bi-GRU Capsnet Model

Our proposed model (figure 2) consists four major components: (1) an embedding layer that transforms each word to a distributed vector with a dimension $d$, (2) a bidirectional- GRU encoder, (3) a capsule network that generates semantic representations of the student and reference answers using a dynamic routing algorithm, (4) a SoftMax layer that computes the probabilities of the correctness classes.

### 3.1 Embedding layer

Given a student answer $X$ and a reference answer $X'$, we tokenize them into a sequence of words: $X = [w_1, \dots, w_n]$ and $X' = [w'_1, \dots, w'_m]$. Afterwards, each token is converted into a d-dimensional vector through the embedding layer. In this work, we consider the following word embeddings approaches: Glove, Word2vec and ELMo.

- Glove embedding has been proposed by Pennington et al. (2014). It is a "count b-based" model where the word co-occurrence count matrix is pre-processed by normalizing the counts and log-smoothing operation. This matrix is then factorized to get lower dimensional representations.
- Word2vec embedding has been proposed by Mikolov and colleagues (2013). Two models have been proposed: CBOW and skip-gram. CBOW computes the probability of a target word given the context surrounding words within a window. Skip-gram is the opposite of CBOW model where the probability of the surrounding words is computed given the target word.
- ELMo (Peters et al. ,2018) method produces word embeddings for each context where the word is used, thus allowing different representations for the same word. The mechanism of ELMo is based on the representation obtained from a bidirectional language model(biLM). It consists of two language models (LM): forward LM and backward LM. The use of ELMo embedding has boosted the performance of several deep learning models.

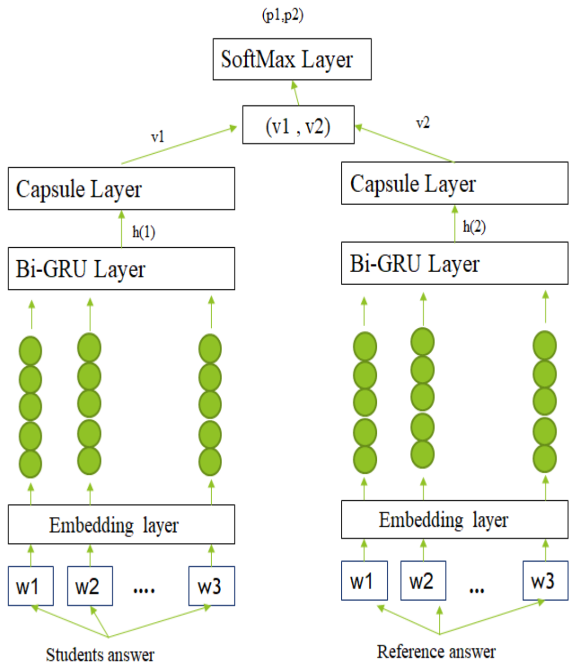

**Figure 2. Bi-Gru-Capsnet model architecture**

### 3.3.2 Bi-GRU layer

A GRU model is a variant of the Recurrent Neural Network (RNN). GRU has two gates: un update gate z and a reset gate r. The update gate determines how much memory of previous cell to keep alive, and the rest gate determines how to combine the input of new cell with the previous memory. For each position $t$, GRU computes $h_t$ with input $x_t$ and previous state $h_{t-1}$, as:

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \quad (1)$$

$$u_t = \sigma(W_u x_t + U_u h_{t-1}) \quad (2)$$

$$\widetilde{h_t} = \tanh\big(W_c x_t + U(r_t \cdot h_t - 1)\big) \quad (3)$$

$$h_t = (1 - u_t) \cdot h_{t-1} + u_t \cdot \widetilde{h_t} \quad (4)$$

Where $h_t$ , $r_t$ and $u_t$ are $d$-dimensional hidden state, reset gate, and update gate. $W_r$ , $W_u$ , $W_c$ and $U_r$ , $U_u$ and $U$ are the parameters of the GRU model. $\sigma$ is the sigmoid function, and . is the element-wise production.

The outputs vectors $h_t$ and $h_t{}'$ are fed into the capsule layer.

## 3.3 Capsule layer

The assumption behind capsule networks is that there are capsules (group of neurons) that tell whether certain entities are present in an image. A capsule as shown in figure 3 has an activation vector that represents the instantiation parameters of an entity and whose length represents the probability of the existence of that entity.

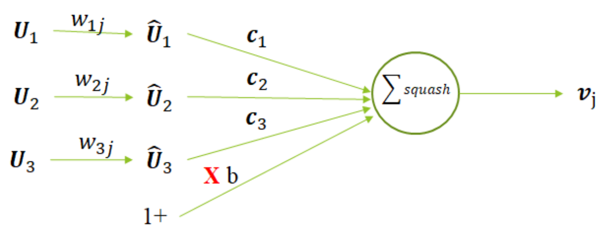

**Figure 3. Capsule structure**

Given the input vectors $u_1$, $u_2$ and $u_3$ from the previous layers, a learned transformation matrix $W_{ij}$ is applied to generate the predictors vectors $\widehat{u_\iota}$ as following:

$$\hat{u}_{j|i} = W_{ij} u_i \quad (5)$$

Then, in the higher layer, a capsule $s_j$ is computed by the linear combination of all the prediction vectors with weights $c_{ij}$ as following:

$$s_j = \sum_i c_{ij} u_{j|i} \quad (6)$$

where $c_{ij}$ are coupling coefficients computed the dynamic routing algorithm described in figure 4.

**Routing Algorithm**

```
1: procedure ROUTING ($\hat{u}_{j|i}, r, l$ )
2: for all capsule $i$ in layer $l$ and capsule $j$ in layer $l+1$:
        $b_{ij} \leftarrow 0$
3: for $r$ iterations do
4: for all capsule $i$ in layer $l$ :
        $c_i \leftarrow softmax(b_i)$                SoftMax computes Eq.3
5: for all capsule $j$ in layer $(l+1)$: $s_j \leftarrow \sum_i c_{ij} \hat{u}_{j|i}$
6: for all capsule $j$ in layer $(l+1)$:
        $v_j \leftarrow squash(s_j)$                squash computes Eq.1
7: for all capsule $i$ in layer $l$ and capsule $j$ in layer $(l+1)$ :
        $b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j$
   return $\boldsymbol{v_j}$
```

**Figure 4. the routing algorithm**

As stated before, the output of a capsule represents the probability that the input has the entity that the capsule describes. So, the range of the activation vector should be in the [0,1] interval. For this purpose, a squash function is applied to generate the final output vector $v_j$ as following:

$$\boldsymbol{v_j} = \frac{\|s_j\|^2}{1+\|s_j\|} \frac{s_j}{\|s_j\|^2} \quad (7)$$

The final outputs of the capsule layers for the given students' answers are activation vectors $v_1$ and $v_2$.

Afterwards, we concatenate these vectors as $[v_1, v_2]$ and we feed this concatenation into a SoftMax layer that computes the probability for each correctness class.

## 4 Experiments and Results

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 The DT-Grade Dataset

The DT-Grade dataset consists of 900 instances of student answers extracted from logged tutorial interactions between 40 junior level college students and the state-of-the-art intelligent tutoring system DeepTutor. Each annotation example (See table 1) consists of the following attributes: (1) problem description, (2) tutor question, (3) student answer and (4) reference answers. In addition, the data includes the correctness class of each student answer. There are four classes: correct, correct but incomplete, incorrect and contradictory.

In this work, we consider two classes: correct and incorrect. The correct answers are those labeled as “correct“ in the DT-Grade dataset. All the other instances are considered as belonging to the “incorrect” class.

| |
|---|
| **Problem Description:** |
| A car windshield collides with a mosquito, squashing it. |
| **Question:** |
| How does Newton's third law apply to this situation? |
| **Student Answer:** |
| the windshield will apply a force to the mosquito equal the force applied by the mosquito to the windshield |
| **Reference answer** |
| 1: Since the windshield exerts a force on the mosquito, which we can call action, the mosquito exerts an equal and opposite force on the windshield, called the reaction. |

**Table 3. Annotation example of the DT-Grade dataset**

### 4.2 Results

Several experiments have been conducted with different capsnet neural networks (see table 4) varying the embedding representations and the number of capsules to evaluate the performance of our proposed model using the DT-Grade dataset.

| Model | Accuracy % | F1 Measure |
|---|---|---|
| Bi-GRU-capsnet (Glove,10) | 61 | 0.61 |
| Bi-GRU-capsnet (Glove,15) | 60.62 | 0.55 |
| Bi-GRU-capsnet (Glove,20) | 58.75 | 0.6 |
| Bi-GRU-capsnet (Word2vec,10) | 55 | 0.59 |
| Bi-GRU-capsnet (Word2vec,15) | 56.25 | 0.57 |
| Bi-GRU-capsnet (Word2vec,20) | 52.25 | 0.47 |
| Bi-GRU-capsnet (Elmo,20) | 69.37 | 0.68 |
| Bi-GRU-capsnet (Elmo,15) | 66.25 | 0.66 |
| Bi-GRU-capsnet (Elmo,10) | **72.5** | **0.7** |
| Bi-GRU (Glove) | 56.25 | 0.56 |
| LSTM (Glove) | 60 | 0.6 |

**Table4. The performance results of various models.**

A first set of experiments have been conducted using the pretrained Glove embeddings with 100 dimension and three different values of the number of capsules. Based on the literature, we have started with a value of 10 and added two other values: 15 and 20. This has been done to test the impact of different expressiveness levels of the capsule network layer on the performance. A second set of experiments have been conducted using word2vec embeddings with 100 dimensions while using the same different values of the number of capsules. Another set of experiments have been run using ELMo embeddings with 300 dimensions, which are the state of art of embeddings, while using the same values of the number of capsules. To compare the performance of our model with existing ones, we have empirically experimented the following baseline deep learning models: (1) An LSTM (Long Short-Term Memory) neural network that consists of a Glove embedding and 240 cells. (2) A Bi-GRU network that consists of Glove embedding with 50 units.

During the experiments, we used 80% of data set for training and 20% for testing. The distribution of classes, as shown in Table 5, in training and testing is imbalanced. To overcome this problem, we adjusted the class weights in the model during the training.

| Dataset | Correct (%) | Incorrect (%) |
|---|---|---|
| Training | 41 | 59 |
| Testing | 41.58 | 58.41 |

**Table5. the distribution of classes in training and testing data**

Table5 represents the distribution of classes in the training and test dataset. As mentioned previously, we have considered data with correct label as correct and anything else as incorrect.

**Hyperparameter**s. In all the experiments, we used a Bi-GRU layer with 50 units. Several numbers of units have been tested and this value has led to higher accuracy. We also added a 0.2 Dropout to the Bi-GRU layer to prevent over-fitting. For the capsule layer, we used 3 iterations for the routing algorithm and 16 for the capsule dimension. We also added a 0.2 Dropout to prevent over-fitting.
For optimization, we use the Adam optimizer (Kimgma and Ba,2014) with a learning rate of 0.0001. The gradients are clipped to 0.5 to prevent exploding gradients. We trained our model for 100 epochs to obtain the results. The increase of epochs, particularly when using the ELMo embedding, showed an increase in the overall accuracy and F1-measure values.

Table 4 shows the results on the DT-Grade dataset. Our Bi-GRU capsnet model outperforms the baselines deep learning models, particularly the Bi-Gru and LSTM models. The results show that our model reaches the highest accuracy of 72.5 and 0.7 of F1-measure when using the ELMo embedding particularly. This is not a surprising result. Several research works have demonstrated that ELMo embeddings boost the performance of deep learning models in various NLP tasks. However, the accuracy and F1 score decreased significantly when using the word2vec embedding approach. 72.5 accuracy is considered a very good result for the DT-Grade dataset due to its small size in comparison with larger NLP datasets.

## 5. Conclusions

In this paper, we proposed a Bi-GRU-Capsnet model to assess the correctness of the students answers within the intelligent tutoring system DeepTutor. We have chosen this deep learning model to get benefits from its no requirements of hands -crafted features and external resources. Added to this, Capsule networks have the capability to express the semantic meanings in a wider space using a vector that captures the instantiation parameters of the input such as the order of words and their semantic representation . The experimental results show that our model reached the state of art of performance on the DT-Grade dataset. Particularly, our model reached the highest accuracy when using the ELMo embeddings. In the future, we plan to investigate more deep learning models to improve the current results.

## ACKNOWLEDGMENTS

To be added later – removed during the blind review process.

## REFERENCES

[1] Ani Nenkova, Kathleen McKeown, et al. 2011. Automatic summarization. Foundations and Trends R in Information Retrieval 5(2–3):103–233.

[2] Banjade, R., Maharjan, N., Niraula, N.B., Gautam, D., Samei, B., Rus, V.2016. Evaluation dataset (DT-grade) and word weighting approach towards constructed short answers assessment in tutorial dialogue context. In: BEA@ NAACL-HLT. pp. 182–187 (2016)

[3] Bao, W., Bao, W., Du, J., Yang, Y., & Zhao, X. (2018). Attentive Siamese LSTM Network for Semantic Textual Similarity Measure. 2018 International Conference on Asian Language Processing (IALP), 312-317.

[4] Cho, K., Merrienboer, B.V., Gülçehre, Ç., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *EMNLP*.

[5] Courtney Corley and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In Proceedings of the ACL workshop on empirical modeling of semantic equivalence and entailment. Association for Computational Linguistics, pages 13–18.

[6] Elvys Linhares Pontes, Stéphane Huet, Andréa Carneiro Linhares, Juan-Manuel Torres-Moreno (2018). Predicting the Semantic Textual Similarity with Siamese CNN and LSTM

[7] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation.” in EMNLP, vol. 14, 2014, pp. 1532–154

[8] Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. Association for Computational Linguistics, pages 985–992.

[9] Kim, J., Jang, S., Choi, S., & Park, E.L. (2018). Text Classification using Capsules. CoRR, abs/1808.03976.

[10] Kingma, D.P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.

[11] KIROS R., ZHU Y., SALAKHUTDINOV R., ZEMEL R. S., TORRALBA A., URTASUN R. & FIDLER S. (2015). Skip-thought vectors. In Proceedings of the 28th International Conference on Neural Information Processing Systems, NIPS’15, p. 3294–3302, Cambridge, MA, USA: MIT Press.

[12] Ngoc Phuoc An Vo, Simone Magnolini, and Octavian Popescu. 2015. Fbk-hlt: An application of semantic textual similarity for answer selection in community question answering. In Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval. volume 15, pages 231–235.

[13] Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L.S. (2018). Deep contextualized word representations. NAACL-HLT.

[14] Sabour, S., Frosst, N., & Hinton, G.E. (2017). Dynamic Routing Between Capsules. NIPS.

[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Advances in neural information processing systems, 2013, pp. 3111–3119.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” arXiv preprint arXiv:1301.3781, 2013.

[17] TAI K. S., SOCHER R. & MANNING C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. CoRR, abs/1503.00075.

[18] TSUBAKI M., DUH K., SHIMBO M. & MATSUMOTO Y. (2016). Non-linear similarity learning for compositionality. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA., p. 2828–2834.

[19] Vasile Rus, Mihai Lintean, Rajendra Banjade, Nobal Niraula, and Dan Stefanescu. 2013b. SEMILAR: The Semantic Similarity Toolkit. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, August 4-9, 2013, Sofia, Bulgaria.

[20] Vasile Rus, Nobal B. Niraula, and Rajendra Banjade. 2015. DeepTutor: an effective, online intelligent tutoring system that promotes deep learning. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15). AAAI Press 4294-4295.

[21] Vasile Rus, Rajendra Banjade, and Mihai Lintean (2014). On Paraphrase Identification Corpora, The 9th International Conference on Language Resources and Evaluation, Reykjavik, May 26-31, 2014, Iceland.

[22] Vasile Rus, Sidney D’Mello, Xiangen Hu, & Arthur C. Graesser. 2013. Recent ad-vances in conversational intelligent tutoring systems. AI maga-zine, 34(3), 42-54.

[23] Wang, Q., Ruan, T., Zhou, Y., Xu, C., Gao, D., & He, P. (2018). An Attention-based BI-GRU-CapsNet Model for Hypernymy Detection between Compound Entities. 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 1031-1035.

[24] Wang, Yue & Di, Xiaoqiang & Li, Jinqing & Yang, Huamin & Bi, Lin. (2018). Sentence Similarity Learning Method based on Attention Hybrid Model. Journal of Physics: Conference Series. 1069. 012119. 10.1088/1742-6596/1069/1/012119.

[25] Xia, C., Zhang, C., Yan, X., Chang, Y., & Yu, P.S. (2018). Zero-shot User Intent Detection via Capsule Neural Networks. EMNLP.

[26] Zhang, N., Deng, S., Sun, Z., Chen, X., Zhang, W., & Chen, H. (2018). Attention-Based Capsule Networks with Dynamic Routing for Relation Extraction. EMNLP.