Report of Commonsense Machine Reading Comprehension

1st Wentao Zhang
Depart. of Computer Science, ZY College
Shanghai Jiao Tong University
Shanghai, China
zwt1999@sjtu.edu.cn

2nd Zhuoyang Ye
Depart. of Computer Science, ZY College
Shanghai Jiao Tong University
Shanghai, China
yezhuoyang@sjtu.edu.cn

3rd Wen Zhang
Depart. of Automation, SEIEE
Shanghai Jiao Tong University
Shanghai, China
tdjzzw@sjtu.edu.cn

***Abstract*—Commonsense Machine Reading Comprehension (CMRC) is a challenging task which aims to infer the answer to the question with the given document and real world commonsense. In this report, we perform experiment on a CMRC data set called ReCoRD. We tested some famous previous baseline models such as Random Guess, BiGRU-Attention on the data set. We also take the advantage of ALBERT, the state of art pre-trained language model as baseline and try to integrate ALBERT with real world commonsense in explicit way and implicit way. For explicit way, we use Graph Attention Network on the Concept Net knowledge graph and integrate it with ALBERT. For implicit way, we train the ALBERT model with relations in the knowledge graph.**

***Index Terms*—Language Model, Machine Reading Comprehension, Commonsense, Commonsense Question Answering**

I. Introduction

A. Background Introduction

In recent years, commonsense become more popular in the field of Natural Language Processing (NLP). In the research field of machine reading comprehension (MRC), research want the machines to infer the answer to the question from the given document and real world commonsense. Many models and data sets are put forward and achieve remarkable result on various data set. These model integrate real world commonsense from public commonsense data base like ConceptNet [1], NELL [2], Wikidata in either implicit way or explicit way.

Inspired by those models, we propose two approaches: Graph Attention Network [3] graph reasoning and knowledge graph. These two approaches integrate the outside commonsense in explicit way or implicit way respectively. In the explicit way, we utilize the state of the art graph model, Graph Attention Network to integrate the knowledge graph to the pre-trained language model. In the implicit way, we convert the relations in the knowledge graph into example and let the language model predict the relation type and whether two entity are related. We hope the pre-trained language model to learn the commonsense and applied it to the CMRC task.

In this task, we use Reading Comprehension with Commonsense Reasoning Dataset (ReCoRD) [4]. It is consist of queries automatically generated from CNN/Daily Mail news articles. The answer to each query is a text span from a summarizing passage of the corresponding news. The goal of the task is to evaluate a machine's ability of commonsense reasoning in reading comprehension.

B. Task Definition

As we mentioned above, in this course project, we use ReCoRD data set. The task on this dataset can be formally defined as: given a passage $\boldsymbol{p}$ describing an event and a set of text spans (entities) $\boldsymbol{E}$ marked in $\boldsymbol{p}$ and a cloze-style query $Q(\boldsymbol{X})$ with a missing entity $\boldsymbol{X}$, our machine $\boldsymbol{M}$ is expected to act like human, reading the passage $\boldsymbol{p}$ and and then using its hidden commonsense knowledge to choose an entity $\boldsymbol{e} \in \boldsymbol{E}$ that best fit $\boldsymbol{X}$. i.e.

$$e^* = \arg\max_{\boldsymbol{e} \in \boldsymbol{E}} P(Q(\boldsymbol{X})|\boldsymbol{p}, \boldsymbol{M})$$

II. Related Works

Our ideas are inspired by some related works. They are works about commonsense integration, and commonsense reasoning.

A. Pre-trained Language Model

In this course project, we use the state fo the art pre-trained language model ALBERT [5], before that, models like BERT [6], RoBERTa [7], XLNET [8] also have do a good job on many NLP tasks.

B. Commonsense Machine Reading Comprehension

Commonsense Machine Reading Comprehension is a long standing challenging task in NLP field. Many datasets like are generated to test the machine's ability to read and comprehsion. Dataset like WikiHop [9], Narrative QA [10] require machine to generate the answer to the question. SQuAD [11] and ReCoRD [4] are extraction style CMRC data. Cosmos QA [12] and OpenBookQA [13] require the machine to do multiple choice.

C. CMRC Models

Many models that trying to solve the CMRC problem are put forward recently. Models like KT-NET [14], KAR [15] and works like [16] have got some good result on text span extraction CMRC task.

III. Methodology

In this part, we will talk about baselines and the two new method we have tried on this data set. We take random guess, BiGRU-Attention and ALBERT [5] as the baselines. And among the two new approach we proposed, the first is Graph Attention network Multi-Hop Reasoning (**GATMHR**) to integrate the commonsense in a explicit way. The second is Relation Pre-Training (**RePT**), we pre-train the ALBRET model with the relation from the knowledge base in an unsupervised way and then train the model on the data set.

A. Baselines

1) ***Random Guess****:* In this baseline, we randomly choose some entity $\boldsymbol{e}$ from the given entity set $\boldsymbol{E}$ and write to the output as the result.

2) ***BiGRU-Attention****:* We also take BiGRU-Attention as baseline. The model architecture can be shown in Figure 1.

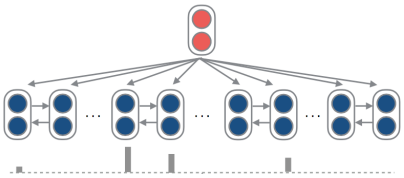

Fig. 1. BiGRU-Attention

In this baseline, we firstly convert the given passage $\boldsymbol{p}$ and query $\boldsymbol{Q}$ into sets of integer tokens $\boldsymbol{C} = \{c_1, c_2, c_3, \cdots, c_n\}$, $\boldsymbol{K} = \{k_1, k_2, k_3, \cdots, k_n\}$. Then we concatenate the tokenized text and query together to get the token representation of the example $\boldsymbol{S} = \{c_1, c_2, c_3, \cdots, c_n, k_1, k_2, k_3, \cdots, k_n\}$. Then we make use of pre-trained 300-dimension GLoVe [17] word vector to initialize the word embedding. We look up the example in the embedding and get a vectorized representation of each word in example, formally: $\boldsymbol{S} = \{\boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c_3}, \cdots, \boldsymbol{c_n}, \boldsymbol{k_1}, \boldsymbol{k_2}, \boldsymbol{k_3}, \cdots, \boldsymbol{k_n}\}$. Then we use a bidirectional GRU to encode these vectors:

$$\boldsymbol{h}_{f_n} = \overrightarrow{\boldsymbol{GRU}}\ [\{\boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c_3}, \cdots, \boldsymbol{c_n}, \boldsymbol{k_1}, \boldsymbol{k_2}, \boldsymbol{k_3}, \cdots, \boldsymbol{k_n}\}]$$

$$\boldsymbol{h}_{b_n} = \overleftarrow{\boldsymbol{GRU}}\ [\{\boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c_3}, \cdots, \boldsymbol{c_n}, \boldsymbol{k_1}, \boldsymbol{k_2}, \boldsymbol{k_3}, \cdots, \boldsymbol{k_n}\}]$$

Where $\boldsymbol{h}_{f_n}$ and $\boldsymbol{h}_{b_n}$ are forward and backward output of GRU. Then we apply attention mechanism to the output hidden vector $\boldsymbol{h}_{f_n}$ and $\boldsymbol{h}_{b_n}$ with hidden state of each time step :

$$\hat{\boldsymbol{h}} = \text{softmax}\left(\frac{\boldsymbol{h}\boldsymbol{H}^{\boldsymbol{T}}}{\sqrt{d}}\right)\boldsymbol{H}$$

where $\hat{\boldsymbol{h}}$ is the final hidden vector of dimension $d = 512$ and $\boldsymbol{H} = \{\boldsymbol{h_1}, \boldsymbol{h_2}, \cdots, \boldsymbol{h_n}\}$ where $\boldsymbol{h}_i$ is the hidden state of each time step. Then we use the concatenation of $\hat{\boldsymbol{h}_{f_n}}$ and $\hat{\boldsymbol{h}_{b_n}}$ as the final representation of the text and query. We denote this vector as $\hat{\boldsymbol{h}_c}$

Then we input this vector into two layer neural network to get the distribution of the start position and end position of the answer.

$$\boldsymbol{A} = \boldsymbol{W_2}(\text{ReLU}(\boldsymbol{W_1}\hat{\boldsymbol{h}_c} + \boldsymbol{b_1})) + \boldsymbol{b_2}$$

B. Explicit Approach

In this part, we discuss about ALBERT encoder we use and the explicit way we use to integrate the commonsense knowledge from ConceptNet. The architecture of the model is shown in Figure 2.

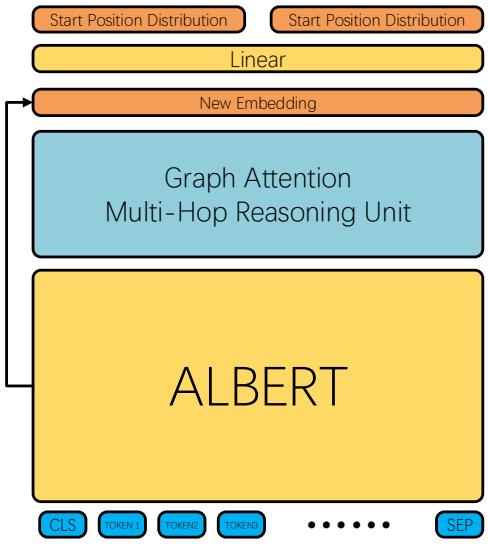

Fig. 2. Explicit Model Architecture

1) ConceptNet: For the source of outside commonsense, we use English part of the Concept Net database. The data is given in relations. The tuple (e_i, r_j, e_k) means e_i has relation r_j with e_k. This can also be seen as an edge in the graph. We use these relations to build the GAT. There are 2809102 relations and 764957 nodes in the English Concept Net.

2) Data Preparation: The Concept Net is too big and some nodes are not related with the task. So we match the entity given in the task with the Concept Net, 59% of the entities are exactly matched in the Concept Net. For those matched entities, we perform BFS to collect those nodes that can be reached within three relation hop. In this way, we sample a sub graph with 1015868 and 288176 relation.

For the text and query, we tokenize the query and text with Sentence Piece [18] tokenizer. We concatenate the query and text tokens together as the text input of model.

3) ALBERT: A Lite BERT (ALBERT) is the state of the art large scale pre-trained model. It improve BERT [6] by making use of new pre-training task and layer parameter sharing. ALBERT is trained in a unsupervised way on Wikipedia and Book Corpus, which are full of real world commonsense, so it is a powerful feature extractor for multi-task learning and commonsense reasoning. Furthermore, it can be treated as knowledge base. There is work like [19] that discuss if it is feasible to use pre-trained language model as the knowledge base.

Google has made the code and pre-trained model public so far. In this course work, we use the ALBERT-base model and try to improve the performance of the model.

4) Bi-Attention Reasoning: We use an explicit reasoning method in the course work. The output of the ALBERT encoder is a matrix $\boldsymbol{M}$ of size $L \times H$ where L is the length of the input sequence, in this report $L = 512$, and H is the size hidden state. In this course work $H = 768$. Inspired by the BiDAF [20] model, we decide to use Bi-Attention mechanism as the approach for reasoning.

After encoding, we cut the matrix into text $\boldsymbol{M_p}$ and query $\boldsymbol{M_q}$. We compute query aware passage $\hat{\boldsymbol{M}}_{\boldsymbol{p}}$ and passage aware query $\hat{\boldsymbol{M}}_{\boldsymbol{q}}$. Where

$$\hat{\boldsymbol{M}}_{\boldsymbol{p}} = \text{softmax}\left(\frac{\boldsymbol{M_q M_p^T}}{\sqrt{H}}\right)\boldsymbol{M_p}$$

$$\hat{\boldsymbol{M}}_{\boldsymbol{q}} = \text{softmax}\left(\frac{\boldsymbol{M_p M_q^T}}{\sqrt{H}}\right)\boldsymbol{M_q}$$

Then we concatenate $\hat{\boldsymbol{M}}_{\boldsymbol{p}}$, $\hat{\boldsymbol{M}}_{\boldsymbol{q}}$ the hidden representation of input $\hat{\boldsymbol{M}}$.

5) Graph Attention Network: Graph Neural Network (GNN) is widely used in commonsense reasoning. The state of the art GNN model is GAT. The embedding of the node in the knowledge graph is randomly initialized. Assume the embedding of the node are $\{\boldsymbol{h_1}, \boldsymbol{h_2}, \cdots, \boldsymbol{h_n}\}$. All these hidden vectors are of 32 dimension. We firstly perform a linear transformation with a weight matrix $\boldsymbol{W} \in \mathbb{R}^{32\times32}$ on the hidden vectors correspond to the entities which are matched in advance from node embedding. Then concatenate every node's hidden vector with their neighbors' hidden vectors and normalized relations type together, and use a trainable vector $\boldsymbol{a} \in \mathbb{R}^{65}$ as the attention factor, then apply Leaky ReLU function on it. Finally we use the softmax function to calculate the importance score. Formally:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}(\boldsymbol{a}^T[\boldsymbol{We_i}||\boldsymbol{We_j}||r_{ij}])\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}(\boldsymbol{a}^T[\boldsymbol{We_i}||\boldsymbol{We_k}||r_{ik}])\right)}$$

Where $\boldsymbol{e}_i$ is the embedding of entities, r_{ij} is the relation type of entity i and entity j and $\mathcal{N}_i$ is the neighbor set of entity i in knowledge graph, note that $i \in \mathcal{N}_i$.

Then we aggregate the neighbor with the parameter α_{ij}

$$\hat{\boldsymbol{e_i}} = \text{ReLU}\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\boldsymbol{We}_j\right)$$

6) Graph Attention Multi-hop Reasoning: For those entities that are matched to the knowledge graph, we enrich it with the knowledge from the knowledge graph. In the Bi-Attention Reasoning part, we have got $\hat{\boldsymbol{M}}_{\boldsymbol{p}} = \{\boldsymbol{w_1}, \boldsymbol{w_2}, \cdots \boldsymbol{w_n}\}$ where $\boldsymbol{w}_i$ is the vector representation of each word. Some of these word are entities matched to the node in the knowledge graph. Assume that $\{\boldsymbol{w_i}, \boldsymbol{w_{i+1}}, \boldsymbol{w_{i+2}}, \cdots, \boldsymbol{w_{i+j}}\}$ form an entity. We enrich the entity with the corresponding node vector in the GAT by element-wise product. That is, for any $\boldsymbol{w_m} \in \{\boldsymbol{w_i}, \boldsymbol{w_{i+1}}, \boldsymbol{w_{i+2}}, \cdots, \boldsymbol{w_{i+j}}\}$

$$\hat{\boldsymbol{w_m}} = \boldsymbol{w_m} \circ \boldsymbol{W_e}\hat{\boldsymbol{e_m}}$$

Where $\boldsymbol{W_e} \in \mathbb{R}^{32\times H}$ is a linear transformation weight. Then we replace the entity back to $\hat{\boldsymbol{M}}_{\boldsymbol{p}}$, we got the graph knowledge 'enriched' text representation $\boldsymbol{M_{p_e}}$. We use $\boldsymbol{M_{p_e}}$ and $\boldsymbol{M_q}$ as the input of Bi-Attention Reasoning part. We repeat for this 'enriching' and reasoning for K times to achieve the purpose of multi-hop reasoning.

7) Joint Training and Loss Function: Multi-task training is useful in many NLP task. The commonsense machine reading comprehension(CMRC) task is hard but the Named Entity Recognition (NER) task is relatively easier.

After reasoning, the multi-hop reasoning part output 'reasoned' representation of query and passage $\hat{\boldsymbol{M}}_{\boldsymbol{r}} \in \mathbb{R}^{L\times H}$. We apply the output transformation weight $\boldsymbol{W_o} \in \mathbb{R}^{H\times2}$ and reshape to get the distribution of the start position and the end position of the answer. At the same time, we also apply a transformation weight $\boldsymbol{W}_{\text{NER}} \in \mathbb{R}^{H\times3}$, because we are given the entities in the passage and All tokens in the passage can be classified into 3 types: B-The beginning of some entity, I-The entity token, but not the beginning one, O-Not belong to any entity. So the joint loss is

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{start}} + \mathcal{L}_{\text{end}} + \lambda\mathcal{L}_{\text{NER}}$$

Where λ is a fixed hyper-parameter. The experiments shows that this loss can help model convergence.

C. Implicit Approach

In this part, we discuss about the implicit way to pre-train ALBERT with the relation from Concept Net. Works mentioned above have shown that language models can be used as the knowledge base. So we want to implicitly encode the commonsense into the language model. So we propose two pre-training task: Is Related Prediction Pre-training and Relation Prediction Pre-training

1) Is Related Prediction Pre-train Task: The first task is relatively easier. For every entity $\boldsymbol{e_i}$ in the knowledge graph, we take out all relation begin with $\boldsymbol{e_i}$. This forms set $\boldsymbol{R}$

$$\boldsymbol{R} = \{(\boldsymbol{e_i}, \boldsymbol{r_{ij}}, \boldsymbol{e_j}), (\boldsymbol{e_i}, \boldsymbol{r_{ij+1}}, \hat{\boldsymbol{e_{j+1}}})\cdots(\boldsymbol{e_i}, \boldsymbol{r_{ij+n}}, \boldsymbol{e_{j+n}})\}$$

With probability 15%, we replace some related entities with some entities that is not related. Then these entities are converted into a string like"[CLS][$\boldsymbol{e_i}$][r_{i1}][$\boldsymbol{e_1}$][CLS][$\boldsymbol{e_i}$][r_{i2}][$\boldsymbol{e_2}$]...[CLS][$\boldsymbol{e_i}$][r_{in}][$\boldsymbol{e_n}$]" as the input of the ALBERT model. The model is required to predict if the two entities are related. We hope the model learn the commonsense after being trained under this task.

2) Relation Prediction Pre-train Task: Another pre-training task is Relation Prediction. We let the model predict the masked relations. Similarly, for relation set $\boldsymbol{R}$ of every entity, we mask the relation and form the training string like "[$\boldsymbol{e_i}$][MASK][$\boldsymbol{e_1}$][SEP][$\boldsymbol{e_i}$][MASK][$\boldsymbol{e_2}$]...[SEP][$\boldsymbol{e_i}$][MASK][$\boldsymbol{e_n}$]". We want the model learn how the entity related by predicting the masked relations.

3) Loss Function: Similar to the explicit way, we also use the same joint loss function in this task.

IV. Experiment and Results

In this part we discuss the experiment details and show the result.

A. Experiment

1) Random Guess: For the random guess experiment, we randomly choose the entities given in the passage.

2) Bi-GRU Attention: In this experiment, we use 300-dimension pre-trained GLoVe word vector. If there is word that is out of vocabulary, we use zero vector instead. Also, we use zero vectors to pad all the vectors to the same length. We use Adam [21] optimizer with learning rate 1e-4 and epsilon 0.99. The hidden size of GRU is set to 512.

3) ALBERT: We train ALBERT with ReCoRD dataset without any modification on the model. We use LAMB [22] optimizer with learning rate 5e-5 and batch size 8 and $\lambda = 1$

4) ALBERT + Bi-Attention Multi-Hop Reasoning: To test the effectiveness of the Bi-Attention Reasoning, we add the Bi-Attention mechanism without GAT. The model architecture is shown in Figure 3.

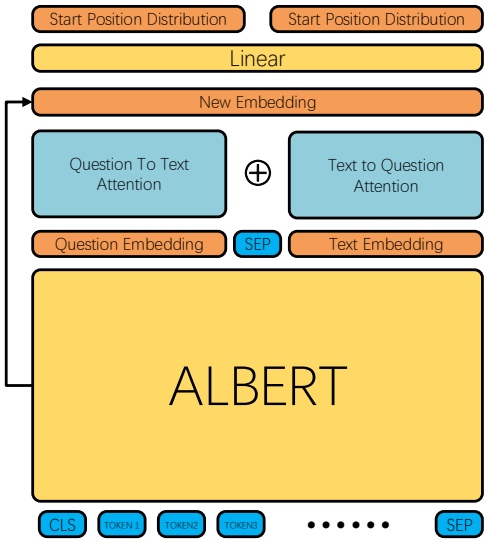

Fig. 3. The model architecture of ALBERT + Bi-Attention

We train this model with LAMB optimizer and learning rate 5e-5, batch size 6, $K = 1$, and $\lambda = 1$

5) ALBERT + Graph Attention Multi-Hop Reasoning: In this experiment, we test the model we mentioned above with LAMB optimizer with learning rate 5e-5, batch size 1, $K = 3$ and $\lambda = 1$.

6) ALBERT + Relation Pre-training: Firstly we create pre-training data for the two relation pre-training task, and train for 2 epochs on each pre-training task with LAMB optimizer and learning rate 5e-5. Then we train the pre-trained ALBERT model on ReCoRD data set for 4 epochs with the same optimizer, learning rate and $\lambda = 1$.

Model	EM	F1
Random Guess	18.44	18.97
Bi-GRU+Attention + GLoVe	3.11	5.33
ALBERT	43.05	45.22
ALBERT + NER	56.76	58.56
ALBERT + NER + Bi-Attention	**58.07**	**59.75**
ALBERT + NER + GATMHR	54.42	56.13
ALBERT + NER + Relation Pretraining	46.24	48.54

TABLE I
THE TEST RESULT ON THE DEV SET OF RECORD

B. Result

The results are shown in the table I

From the result, we can find that the Bi-GRU Attention model performs worse than random entities guess, this is feasible and will be discussed in the next section. We can also find that NER loss do help the model find the correct entity. And the Bi-Attention mechanism improve the result by about 2 point. However, both Graph Attention Network Multi-Hop Reasoning and Relation Pre-training do not improve the performance, possible reasons also will be discussed in the next part.

V. Discussion and Analysis

1) Bi-GRU Attention: This model performs worst in our experiment, even worse than the random guess. However, this is reasonable because in the random guess baseline, we randomly choose entities in the given entity set. But the Bi-GRU Attention model not know these entities and it have to complete the regression task instead of classification task. The former is much harder than latter. So it is natural that this model can not beat random guess.

Furthermore, this model has few pre-trained or pre-known commonsense knowledge. So it is understandable for it to perform so bad in this task.

2) ALBERT: Although this model is considered as the baseline, ALBERT itself is a very powerful language model with rich pre-trained commonsense knowledge, so it is natural for ALBERT to reach such a high score.

3) NER Joint Loss: In the experiment, the NER joint loss help the model to improve the performance and converge faster. By training with entity supervised, the model can learn to fit on the entities much easier. And the experiment result shows that such supervised training increase the performance significantly.

4) Bi-Attention Multi-Hop Reasoning: Inspired by BiDAF model and some other related works, we design this reasoning model. By using Bi-Attention mechanism, we get the query aware passage and passage aware query. Thank to the powerful feature extractor ALBERT, this mechanism improve the performance of the model.

5) Graph Attention Multi-Hop Reasoning: In our experiment, the Graph Attention Multi-Hop Reasoning do not improve the performance. I think the reason can be as follows

1. The batch size is too small because we need to process the whole large graph.

2. The embedding of node is randomly initialized, and it is not sufficiently updated or the information is not sufficiently diffused in the network. So the knowledge enriched to the output of ALBERT may just noise.
3. The commonsense needed not only related to the given entities, it also related other words in the passage and query, we only introduce the knowledge of given entities from knowledge graph.
4. The training cost too much time for leak of computing power, so we can not choose some good hyper parameters.

For those reasons above, I think the result is acceptable.

6) Relation Pre-training: After relation pre-training, the performance of our model drop significantly for about 10 point. I think reason for this is that the input of our pre-training task is not a real sentence. So the learning process of pre-training destroy the previous language model, so the result witness a significant drop.

VI. Conclusion

In this course project, we test some classic model in machine reading comprehension and try the best large scale pre-trained language model ALBERT. And we propose the explicit and implicit approaches to integrate the commonsense knowledge from Concept Net data base. And some of approaches we put forward improve the performance of ALBERT significantly.

References

References

[1] R. Speer, J. Chin, and C. Havasi, "Conceptnet 5.5: An open multilingual graph of general knowledge," *CoRR*, vol. abs/1612.03975, 2016. [Online]. Available: http://arxiv.org/abs/1612.03975

[2] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling, "Never-ending learning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*, 2015.

[3] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017.

[4] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. V. Durme, "Record: Bridging the gap between human and machine commonsense reading comprehension," 2018.

[5] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," 2019.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.

[7] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[8] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," 2019, cite arxiv:1906.08237Comment: Pretrained models and code are available at https://github.com/zihangdai/xlnet. [Online]. Available: http://arxiv.org/abs/1906.08237

[9] J. Welbl, P. Stenetorp, and S. Riedel, "Constructing datasets for multi-hop reading comprehension across documents," 2017.

[10] T. Kočiský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette, "The narrativeqa reading comprehension challenge," 2017.

[11] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," 2018.

[12] L. Huang, R. L. Bras, C. Bhagavatula, and Y. Choi, "Cosmos qa: Machine reading comprehension with contextual commonsense reasoning," 2019.

[13] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 2381–2391. [Online]. Available: https://www.aclweb.org/anthology/D18-1260

[14] A. Yang, Q. Wang, J. Liu, K. Liu, Y. Lyu, H. Wu, Q. She, and S. Li, "Enhancing pre-trained language representations with rich knowledge for machine reading comprehension," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 2346–2357. [Online]. Available: https://www.aclweb.org/anthology/P19-1226

[15] C. Wang and H. Jiang, "Explicit utilization of general knowledge in machine reading comprehension," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 2263–2272. [Online]. Available: https://www.aclweb.org/anthology/P19-1219

[16] D. Weissenborn, T. Kočiský, and C. Dyer, "Dynamic integration of background knowledge in neural nlu systems," 2017.

[17] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[18] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," 2018.

[19] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, "Language models as knowledge bases?" 2019.

[20] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," 2016.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[22] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, "Large batch optimization for deep learning: Training bert in 76 minutes," 2019.