

Article

Unlocking Everyday Wisdom: Enhancing Machine Comprehension with Script Knowledge Integration

Zhihao Zhou ^{1,†}, Tianwei Yue ^{1,†}, Chen Liang ¹, Xiaoyu Bai ², Dachi Chen ¹, Congrui Hetang ¹ 
and Wenping Wang ^{1,*} 

¹ School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA; zhihaoz@alumni.cmu.edu (Z.Z.); tyue@alumni.cmu.edu (T.Y.); chenlia2@alumni.cmu.edu (C.L.); chendachimail@gmail.com (D.C.); congruihetang@gmail.com (C.H.)

² Center for Theoretical Biological Physics, Rice University, Houston, TX 77005, USA; xybai521@gmail.com

* Correspondence: wenpingw@alumni.cmu.edu

[†] These authors contributed equally to this work.

Abstract: Harnessing commonsense knowledge poses a significant challenge for machine comprehension systems. This paper primarily focuses on incorporating a specific subset of commonsense knowledge, namely, script knowledge. Script knowledge is about sequences of actions that are typically performed by individuals in everyday life. Our experiments were centered around the MCScript dataset, which was the basis of the SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge. As a baseline, we utilized our Three-Way Attentive Networks (TriANs) framework to model the interactions among passages, questions, and answers. Building upon the TriAN, we proposed to: (1) integrate a pre-trained language model to capture script knowledge; (2) introduce multi-layer attention to facilitate multi-hop reasoning; and (3) incorporate positional embeddings to enhance the model's capacity for event-ordering reasoning. In this paper, we present our proposed methods and prove their efficacy in improving script knowledge integration and reasoning.

Keywords: machine comprehension; multi-hop reasoning; script knowledge; commonsense knowledge



Citation: Zhou, Z.; Yue, T.; Liang, C.; Bai, X.; Chen, D.; Hetang, C.; Wang, W. Unlocking Everyday Wisdom: Enhancing Machine Comprehension with Script Knowledge Integration. *Appl. Sci.* **2023**, *13*, 9461. <https://doi.org/10.3390/app13169461>

Academic Editors: Julian Szymanski and Ahmed Rafea

Received: 17 May 2023

Revised: 14 August 2023

Accepted: 16 August 2023

Published: 21 August 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, the rapid development of natural language processing (NLP) technologies has taken place. They have been widely applied into numerous everyday products such as search engines, smart assistants, mobile devices, and more. Very recently, large language models such as ChatGPT and Bard have ushered in a new era of NLP research and application, thereby showcasing their unprecedented performance results across many tasks and their gaining extensive adoption by users worldwide.

Despite the significant achievements, the journey towards developing an NLP system with human-level cognitive abilities is far from over. One of the pressing challenges is the integration of commonsense knowledge into machine comprehension systems—an area which has seen growing attention [1]. Commonsense knowledge refers to an extensive array of “everyday wisdom”, which is considered to be universally known to humans. For instance, in the sentence, “He hit the window with a bat,” it is obvious to a human reader that the “bat” refers to a sporting equipment rather than the animal. However, for machines, leveraging such knowledge poses considerable challenges, primarily because it is rarely explicitly stated in human communication, which comes in addition to its vast scope and diversity. By combining NLP systems with this “everyday wisdom”, we can significantly enhance their capabilities, thus leading to smoother and more human-like interactions.

In this paper, we aim to enhance the integration of a particularly nuanced form of commonsense knowledge: script knowledge. The term “script” refers to typical sequences of activities that describe well-known situations. To illustrate, consider a “someone opening the door” scenario: an individual reaches for his key, and, if it is not found, he rings the

doorbell with the hope that someone inside will open the door. This narrative seems intuitive to a human; so, when given the sentence “He must ring the doorbell”, it is a logical inference that “he forgot his key”. However, a machine lacking this knowledge would struggle to form this connection. Improving script knowledge integration would enable NLP systems to produce better responses to everyday human activities, thus enhancing their practical applicability.

We investigated techniques to augment the script knowledge understanding of an established baseline system known as the TriAN [2]. Our experiments leveraged the MCScript [3] dataset, which is a machine question-answering (QA) dataset that is specifically designed to assess a model’s script knowledge comprehension. We delved deeper into the dataset and the baseline method in subsequent sections. Our main contributions are the following:

- We trained a generative LSTM language model on “scripts” of everyday human activity and used it as a feature encoder.
- We used multiple layers of attention interactions to enable multi-hop reasoning.
- We added positional embeddings to the intermediate features to enhance the temporal sequential reasoning.

2. Background and Related Work

2.1. Machine Reading and Question-Answering Datasets

How can we measure the effectiveness of an NLP system in comprehending a text? Machine reading and question-answering (QA) tasks have become a popular benchmark. Here, a passage of text is provided to the NLP system, which then must answer a series of questions about the text and often given multiple choices for the answer. The accuracy of the system’s responses reflects its text comprehension and reasoning capabilities. A prime example is the Stanford Question Answering Dataset (SQuAD) [4]. It includes ~23 k paragraphs that have been excerpted from top-ranking Wikipedia articles, with over 100 k associated question-answer pairs that span a broad range of topics. The SQuAD data was collected through crowdsourcing. More recent and larger datasets have also been released, with some applying search engines to partially automate the data collection and increase the scale. For instance, the SearchQA dataset [5] contains over 6.9 million snippets for its 140 k questions, while MS MARCO [6] boasts of over 1 million questions and 8.8 million passages.

In addition to the generic ones, there exist machine QA datasets that are designed to assess specific aspects of performance. Our work is centered on enhancing reasoning based on script knowledge, so we used the MCScript [3] dataset as our benchmark. The MCScript, which is a machine QA dataset that has been specifically engineered to evaluate an NLP system’s comprehension of script knowledge, comprises ~140 k questions and 2119 narrative texts that depict 110 everyday activity scenarios. The data was gathered via crowdsourcing. Notably, ~27% of the questions cannot be answered directly from the provided text, thus necessitating the application of script knowledge about the scenarios. This sets the MCScript apart from counterparts such as the SQuAD, which mandates the answers to be present within the given text, thereby potentially allowing the system to generate an answer by simply scanning the text without deeper reasoning. The MCScript dataset has served as the foundation for a script knowledge benchmark [7] at SemEval 2018. An example passage, along with its questions and answer choices, is shown in Figure 1.

T: This past weekend, my family made so much food that there was still plenty of it going into the week. Knowing this, we chose not to make any new food, because we could still eat the leftover food from the weekend. All we would need to do was warm it up on the stove top. The stove is a gas stove, which produces a flame that, in my opinion, cooks food much better than an electric stove. On Monday after getting home from work, rather than making a new meal, my mom took some of the food that we had saved from the weekend and put it into a pan, and put that pan on the stove. She turned the gas on until the burner ignited, and left the heat on until she felt that the food was warm enough. When she ate it, she swore it was as good as it was when it was freshly made.

Q1: Where were they heating the food?

- a. On an electric stove.
- b. On a gas stove.

Q2: How many people were making the food?

- a. Just one.
- b. About 5.

Figure 1. An example passage from the SemEval 2018 dataset, and two corresponding questions.

2.2. Integrating Commonsense Knowledge

2.2.1. Using Knowledge Graphs

Knowledge graphs are broadly used [8–11] as a source of commonsense knowledge. These large graphs consist of nodes representing concepts and edges describing the relationships between them. A representative example is ConceptNet [12], which is a freely available, multi-lingual semantic network. ConceptNet comprises triplets of (subject, relation, object), e.g., (“apple”, “IsA”, “fruit”), (“plane”, “CapableOf”, “fly”), and (“car”, “Causes”, “pollution”). The subject and object are multi-word phrases, and the relation is one of the relation types. ConceptNet contains ~630 k of such triplets and 39 types of relationships. An overview is shown in Figure 2.

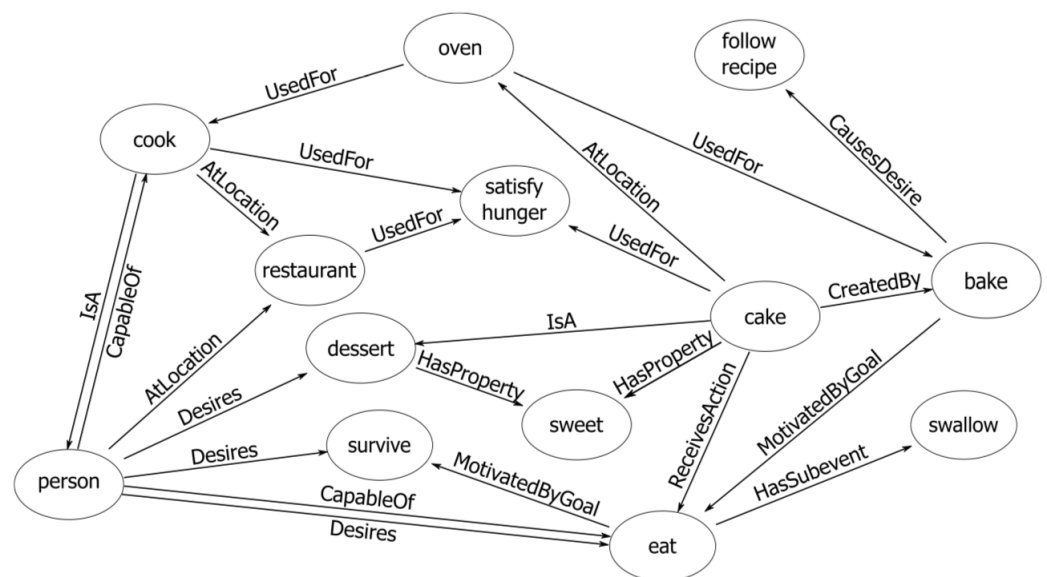


Figure 2. An overview of ConceptNet [12].

ConceptNet is one of the most extensively used commonsense knowledge graphs. For example, ref. [8] generated knowledge embeddings from ConceptNet to enhance the context features for cloze-style QA tasks. Ref. [9] used ConceptNet data to train a network that predicts the relevance score of concepts, thereby assessing the relevance of a candidate answer to a given question in context. Ref. [11] devised two auxiliary training tasks to improve machine reading comprehension, thereby predicting the existence and type of relationships between concepts in provided texts. The ground truth for these sub-tasks has been sourced from ConceptNet.

We also applied ConceptNet in our experiments to acquire the relationship information of concepts, thereby serving as an input for our model. Larger and richer knowledge graphs may bring further improvements, e.g., the Google Knowledge Graph, which is a proprietary knowledge base developed by Google to enhance its search results and other services. It can connect more complex information to answer questions, such as “Who is the president of the country where the White House is located”. Our method is fully compatible with more advanced knowledge graphs, which we leave for future work.

2.2.2. Using Additional Text Corpus

The extra text corpus is also a valuable source of commonsense knowledge. For example, ref. [13] mined object semantic knowledge from Wikipedia articles to generate more plausible text completions. Ref. [14] drew prior statistics from the ProPara [15] dataset (a text corpus focused on scientific procedures, such as photosynthesis) to improve the model’s understanding of entity state changes in scientific processes. In addition, ref. [16] created a new dataset named Common Sense Explanations (CoS-E), which resembles a QA dataset but accompanies each question with a sentence of explanation for its correct answer. A language model (LM) is trained to generate an explanation when given a question–answer pair, with the produced explanation then offered as additional input to the downstream classification model.

Our approach also employs additional text data to acquire commonsense knowledge, especially script knowledge. We trained a generative LM with a story dataset, which we then utilized as a script-knowledge-aware feature encoder. The text in the story dataset combines the ~2100 passages from the MCScript and an additional 500 passages from the MCTest [17] dataset, both being narrative texts, which contains typical sequences of daily activities, a.k.a script knowledge. Compared to the aforementioned related works, our solution is straightforward and can be trained end-to-end.

3. Method

3.1. Baseline Model

The baseline model we re-implemented is called Three-Way Attentive Networks (Tri-ANs) [2]. It models the interactions between the passage and the question, the question and the answer, and the passage and the answer by using a three-way attention mechanism. An overview of the model framework is shown in Figure 3. It consists of an input layer, an attention layer, and an output layer.

Input Layer: A training example consists of a passage $\{P_i\}_{i=1}^{|P|}$, a question $\{Q_i\}_{i=1}^{|Q|}$, an answer $\{A_i\}_{i=1}^{|A|}$, and a label $y^* \in \{0, 1\}$ as input. P_i is the representation of the i -th word in the passage, which is the same for the question and the answer. For the representation, we used the GloVe word vector [18] for the passage, the question, and the answer ($E_{P_i}^{GLOVE}$, $E_{Q_i}^{GLOVE}$, and $E_{A_i}^{GLOVE}$), as well as used the following features:

- POS Embeddings ($E_{P_i}^{POS}$ and $E_{Q_i}^{POS}$): Randomly initialized 12-dimensional vectors trained to encode pre-labeled part-of-speech tags (whether the word is a noun, verb, etc.) for passage and question texts.
- NER Embeddings ($E_{P_i}^{NER}$): Randomly initialized 8-dimensional vectors trained to encode pre-labeled name–entity recognition tags (whether the word belongs to categories such as people, company, date, etc) for the passage texts.

- REL Embeddings ($\mathbf{E}_{P_i}^{REL}$): Randomly initialized 10-dimensional vectors trained to encode a relationship between a word in the passage and any word in the question/answer. Such relationship comes from querying ConceptNet for an edge between the words. If multiple relationships exist, a random one is chosen.
- Handcrafted Features: Handcrafted features include logarithmic term frequency ($\mathbf{E}_{P_i}^{TF}$) and co-occurrence features ($\mathbf{E}_{P_i}^{CO}$). Logarithmic term frequency features come from English Wikipedia. Co-occurrence features are binary, thus being true if a passage word is found in the question/answer.

$$\mathbf{W}_{P_i} = [\mathbf{E}_{P_i}^{GLOVE}, \mathbf{E}_{P_i}^{POS}, \mathbf{E}_{P_i}^{NER}, \mathbf{E}_{P_i}^{REL}, \mathbf{E}_{P_i}^{CO}, \mathbf{E}_{P_i}^{TF}] \quad (1)$$

$$\mathbf{W}_{Q_i} = [\mathbf{E}_{Q_i}^{GLOVE}, \mathbf{E}_{Q_i}^{POS}] \quad (2)$$

$$\mathbf{W}_{A_i} = [\mathbf{E}_{A_i}^{GLOVE}] \quad (3)$$

Attention Layer: We used word-level attention to model interactions between the passage, the question, and the answer. The model first calculates context vectors for the passage by attending to the question. Then, it calculates context vectors for the answer by attending to the question and the passage. The attention function is represented in (4) and (5). f is a non-linear activation function and is set to *ReLU*. From the attention function, we can compute question-aware passage representations $\mathbf{W}_{P_i}^q$, passage-aware answer representations $\mathbf{W}_{A_i}^p$, and question-aware answer representations $\mathbf{W}_{A_i}^q$ in (6)–(8).

$$Att_{seq}(\mathbf{u}, \{\mathbf{v}_i\}_{i=1}^n) = \sum_{i=1}^n \alpha_i \mathbf{v}_i \quad (4)$$

$$\alpha_i = softmax_i(f(\mathbf{W}_1 \mathbf{u})^T f(\mathbf{W}_1 \mathbf{v}_i)) \quad (5)$$

$$\mathbf{W}_{P_i}^q = Att_{seq}(\mathbf{W}_{P_i}, \{\mathbf{W}_{Q_i}\}_{i=1}^{|Q|}) \quad (6)$$

$$\mathbf{W}_{A_i}^p = Att_{seq}(\mathbf{W}_{A_i}, \{\mathbf{W}_{P_i}\}_{i=1}^{|P|}) \quad (7)$$

$$\mathbf{W}_{A_i}^q = Att_{seq}(\mathbf{W}_{A_i}, \{\mathbf{W}_{Q_i}\}_{i=1}^{|Q|}) \quad (8)$$

After that process, three groups of context embeddings are appended to the original embeddings to form the final embeddings. Then, three BiLSTMs are applied to the concatenated embeddings to model the temporal dependency, as shown in (9)–(11).

$$\mathbf{h}^p = BiLSTM([\mathbf{W}_{P_i}, \mathbf{W}_{P_i}^q]_{i=1}^{|P|}) \quad (9)$$

$$\mathbf{h}^q = BiLSTM([\mathbf{W}_{Q_i}]_{i=1}^{|Q|}) \quad (10)$$

$$\mathbf{h}^a = BiLSTM([\mathbf{W}_{A_i}, \mathbf{W}_{A_i}^p, \mathbf{W}_{A_i}^q]_{i=1}^{|A|}) \quad (11)$$

Output Layer: The question sequence and answer sequence representation— $\mathbf{h}^q$ and $\mathbf{h}^a$, respectively, are summarized into fixed-length vectors $\mathbf{q}$ and $\mathbf{a}$ with self-attention. The self-attention function is defined as in (12) and (13). To represent the passage, we used the sequence attention function defined before to summarize out the passage representation $\mathbf{p}$ by attending $\mathbf{q}$ to $\mathbf{h}^p$. These operations are shown in (14)–(16).

$$Att_{self}(\{\mathbf{u}_i\}_{i=1}^n) = \sum_{i=1}^n \alpha_i \mathbf{u}_i \quad (12)$$

$$\alpha_i = softmax_i(\mathbf{W}_2^T \mathbf{u}_i) \quad (13)$$

$$\mathbf{q} = Att_{self}(\{\mathbf{h}_i^q\}_{i=1}^{|Q|}) \quad (14)$$

$$\mathbf{a} = Att_{self}(\{\mathbf{h}_i^a\}_{i=1}^{|A|}) \quad (15)$$

$$\mathbf{p} = Att_{seq}(\mathbf{q}, \{\mathbf{h}_i^p\}_{i=1}^{|P|}) \quad (16)$$

Then, the question and answer vectors are summarized to single vectors using self attention. The passage vectors are summarized to a single vector via bi-linear attention on the question vectors. Finally, we used a bi-linear function to summarize the 3 sequence representations and applied a sigmoid activation function to obtain the final probability score on whether the choice was the correct answer for the question with respect to the passage, which is shown in (17).

$$y = \sigma(\mathbf{p}^T \mathbf{W}_3 \mathbf{a} + \mathbf{q}^T \mathbf{W}_4 \mathbf{a}) \quad (17)$$

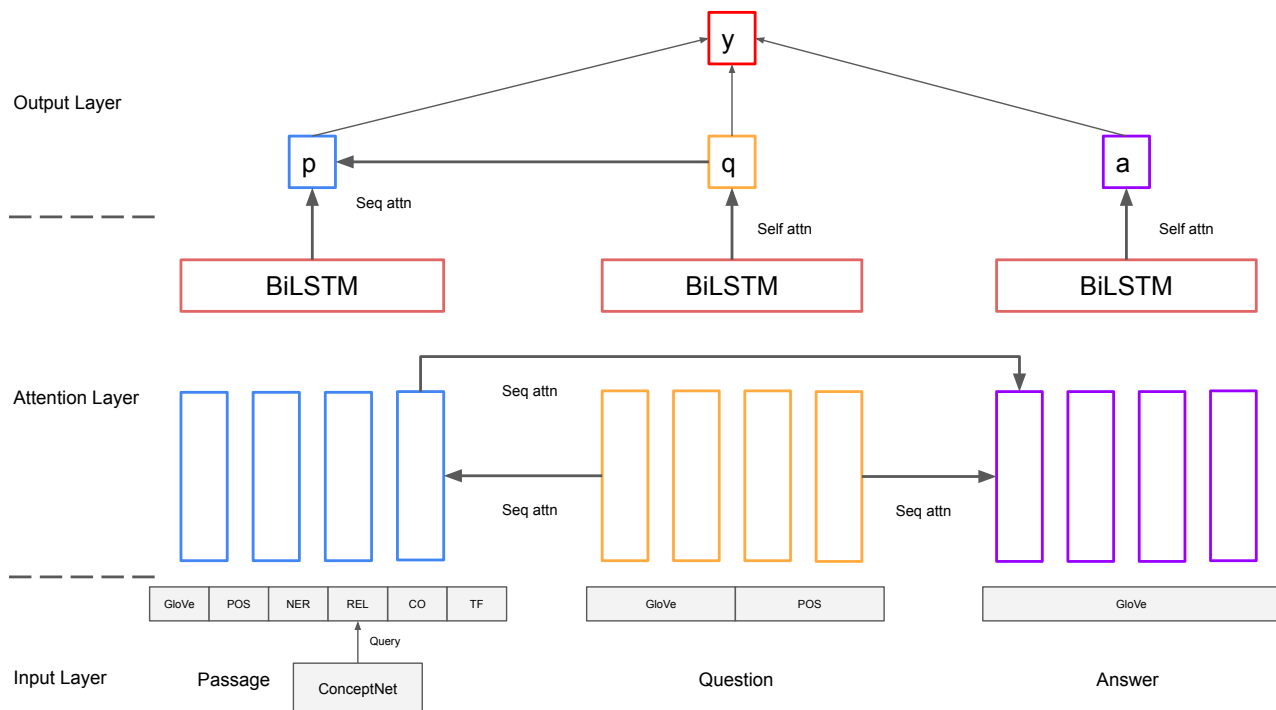


Figure 3. Baseline model framework [2].

3.2. Error Analysis

In order to improve upon the TriANs, it is important to understand its limitations. In this section, we performed error analysis on the original TriANs model.

3.2.1. Qualitative Analysis

We inspected all the errors made by machine and grouped common problems into the following categories. The main problem clusters are the following:

- **Failure to build temporal sequence:** From a human perspective, temporal sequences often imply causal relationships. For instance, if event A happens before event B, it is possible that A caused B. This implication becomes stronger if A occurs immediately before B, thereby suggesting that A is the direct cause of B. However, the model frequently identified only indirect causes, thus indicating its struggle to interpret these direct temporal–causal links accurately. This can be mitigated via the better integration of script knowledge.

- Failure to choose based on elimination: This term refers to instances where the incorrect options are explicitly contradicted by evidence in the passage, even though there is no clear evidence supporting the correct choice. In these cases, it might be challenging to identify the correct answer directly. However, the correct response should still have been deduced by a process of elimination.
- Lack of commonsense knowledge: Despite the model leveraging ConceptNet to incorporate commonsense knowledge, it appears this approach is not sufficient. The model could still make non-sensical mistakes.
- Lack of quantification ability: Sometimes to answer the question, it is required to perform some simple computation or estimation on the value. The model was observed to struggle with these operations.
- Weakness in co-reference resolution: The system struggled with a co-reference resolution when the reference was ambiguous. Resolving these ambiguities often requires simple reasoning, which is grounded in both the context and commonsense knowledge. This can also be improved by better commonsense integration.
- Ground truth answers may be wrong: Some ground truth answers seemed to be wrong from the human perspective, which contributes to a small portion of the errors.

3.2.2. Quantitative Analysis

Besides qualitative analysis, we also performed quantitative analysis. In Figure 4, we plotted the distribution of confidence of the model on the choice in 4 cases, as explained in the caption of the figure. From the “pred_t_in_t” and “pred_f_in_f” sub-plots, we can see that the model was usually confident with its correct predictions. In the other two sub-plots, where the model made incorrect predictions, the confidence scores were much lower. This lack of confidence indicates that there was no sufficient information for the model to make confident choices. The addition of more background knowledge may resolve some ambiguity and provide improvements.

We then performed some analysis on the distribution of errors for 10 question scenarios, as shown before in Figure 5. We calculated the error rate of the model for 10 scenarios in Figure 6. From the figure, the “when” category resulted in being the worst. This indicates that the current model struggles with reasoning about temporal sequences, thus suggesting room for improvement through the integration of script knowledge.

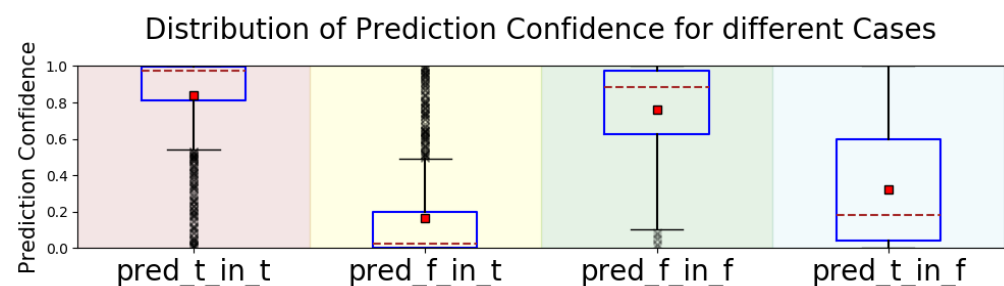


Figure 4. Distribution of prediction confidence in 4 cases. “pred_t_in_t” stands for true positives. “pred_f_in_t” stands for false negatives. “pred_f_in_f” stands for true negatives. “pred_t_in_f” stands for false positives. The blue box represents the data points between the lower/upper quartiles. The small red square is the mean. The red dashed line represents the median. The short black solid line is the boundary for outliers. The small black crosses are outliers.)

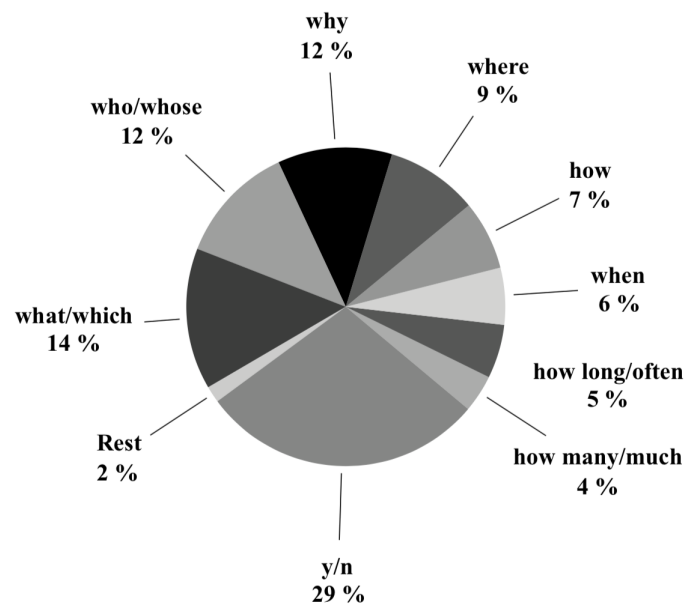


Figure 5. Distribution of question types in MCScript dataset [3].

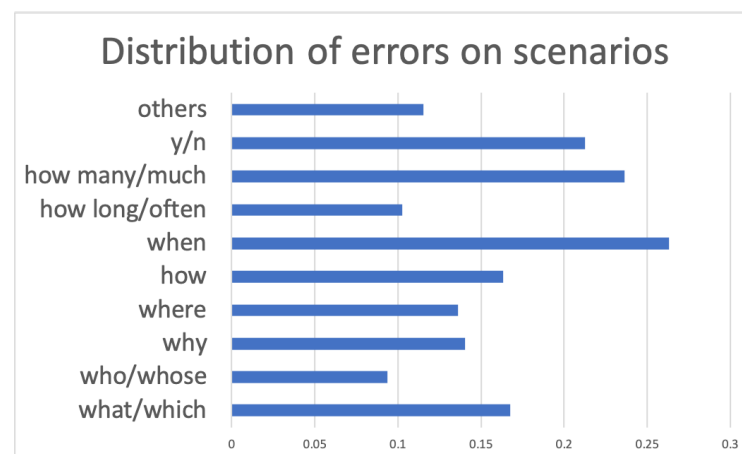


Figure 6. Distribution of error cases for 10 question scenarios.

3.3. Proposed Method

3.3.1. Methodology Breakdown

To enhance the script knowledge integration for the baseline model, we proposed several techniques that are explained below.

A: Integrating a Pre-Trained Language Model

We pre-trained a generative language model (LM) with a dataset of narrative passages. The LM was an LSTM that predicts the next word when given the predicted words in an auto-regressive manner. The dataset was made by combining the passages from the MCScripts and MCTest [17], which totals ~2600 passages. This forms an extended script knowledge base that mitigates overfitting. We then used the pre-trained LM to produce additional feature embeddings for the input text, and we fine-tuned it jointly with the entire model. The auto-regressive generation task naturally made the model more aware of “what happens next” in a series of events, which embodies the script knowledge.

B: Multi-hop Attention

The attention mechanism has been proven to be an effective method to help the model focus on the most relevant information [19–22]. It serves this purpose in the baseline

method by enabling the model to give more weights to the most helpful parts of the passage for answering the question. However, single-hop attention is insufficient to perform more complex hierarchical reasoning [23,24]. Therefore, we proposed multi-hop attention. Take another look at the example text in Figure 1. For the first question, the model should first attend to heating the food in the passage, which is warm it up. Then, the model should scan up or down to find where the food is heated, which is on the stove top. This two-step process is an example of multi-hop reasoning. The first hop is to locate the key word; the second hop is to locate information before or after the key word, which is closely related depending on the question. Generally, the more indirect the relation between the questioned item and the true answer, the more hops of attention are needed.

C: Positional Embedding

The attention operation used in the baseline method is invariant to the order of words. However, temporal order is an important aspect of script knowledge. To explicitly represent such sequential information in the process of multi-hop attention, we added positional embedding for each word, which aimed to enhance the model's reasoning regarding event orderings.

3.3.2. Model Architecture

The overall architecture of our model is shown in Figure 7. The framework can be divided into the following steps.

Step 1: The First Hop

The first hop is the same as with the attention applied before the BiLSTMs in the baseline model. We denote the input of the first hop as $\mathbf{W}_{P_{i1}}$, $\mathbf{W}_{Q_{i1}}$, and $\mathbf{W}_{A_{i1}}$, in Equations (18)–(20), respectively. Using the same attention function with that in the baseline model, we obtain $\mathbf{W}_{P_{i1}}^q$, $\mathbf{W}_{A_{i1}}^p$, and $\mathbf{W}_{A_{i1}}^q$, respectively.

$$\mathbf{W}_{P_{i1}} = [\mathbf{E}_{P_i}^{GLOVE}] \quad (18)$$

$$\mathbf{W}_{Q_{i1}} = [\mathbf{E}_{Q_i}^{GLOVE}] \quad (19)$$

$$\mathbf{W}_{A_{i1}} = [\mathbf{E}_{A_i}^{GLOVE}] \quad (20)$$

Step 2: The Second Hop

The input of the second hop is a concatenation of the input of the first hop, the output embeddings from first hop, and the positional embeddings, as shown in Equations (21)–(23). Then, we applied the same attention mechanism as in the first hop and obtained $\mathbf{W}_{P_{i2}}^q$, $\mathbf{W}_{A_{i2}}^p$, and $\mathbf{W}_{A_{i2}}^q$, respectively.

$$\mathbf{W}_{P_{i2}} = [\mathbf{W}_{P_{i1}}; \mathbf{W}_{P_{i1}}^q; \mathbf{E}_{P_i}^{PE}] \quad (21)$$

$$\mathbf{W}_{Q_{i2}} = [\mathbf{W}_{Q_{i1}}; \mathbf{E}_{Q_i}^{PE}] \quad (22)$$

$$\mathbf{W}_{A_{i2}} = [\mathbf{W}_{A_{i1}}; \mathbf{W}_{A_{i1}}^p; \mathbf{W}_{A_{i1}}^q; \mathbf{E}_{A_i}^{PE}] \quad (23)$$

Step 3: The Output Layer

In the third step, we concatenated the output embeddings from the previous step with additional features, including the embeddings generated from the pre-trained

language model, as shown in Equations (24)–(26). $\mathbf{W}_{P_{i3}}$, $\mathbf{W}_{Q_{i3}}$, and $\mathbf{W}_{A_{i3}}$ are the corresponding inputs of BiLSTMs.

$$\mathbf{W}_{P_{i3}} = [\mathbf{W}_{P_{i2}}; \mathbf{W}_{P_{i2}}^q; \mathbf{E}_{P_{i2}}^{LMF}; \mathbf{E}_{P_{i2}}^{POS}; \mathbf{E}_{P_{i2}}^{NER}; \mathbf{E}_{P_{i2}}^{REL}; \mathbf{E}_{P_{i2}}^{CO}; \mathbf{E}_{P_{i2}}^{TF}] \quad (24)$$

$$\mathbf{W}_{Q_{i3}} = [\mathbf{W}_{Q_{i2}}; \mathbf{E}_{Q_{i2}}^{POS}] \quad (25)$$

$$\mathbf{W}_{A_{i3}} = [\mathbf{W}_{A_{i2}}; \mathbf{W}_{A_{i2}}^p; \mathbf{W}_{A_{i2}}^q] \quad (26)$$

Finally, we acquired the prediction through the same operations as in the baseline.

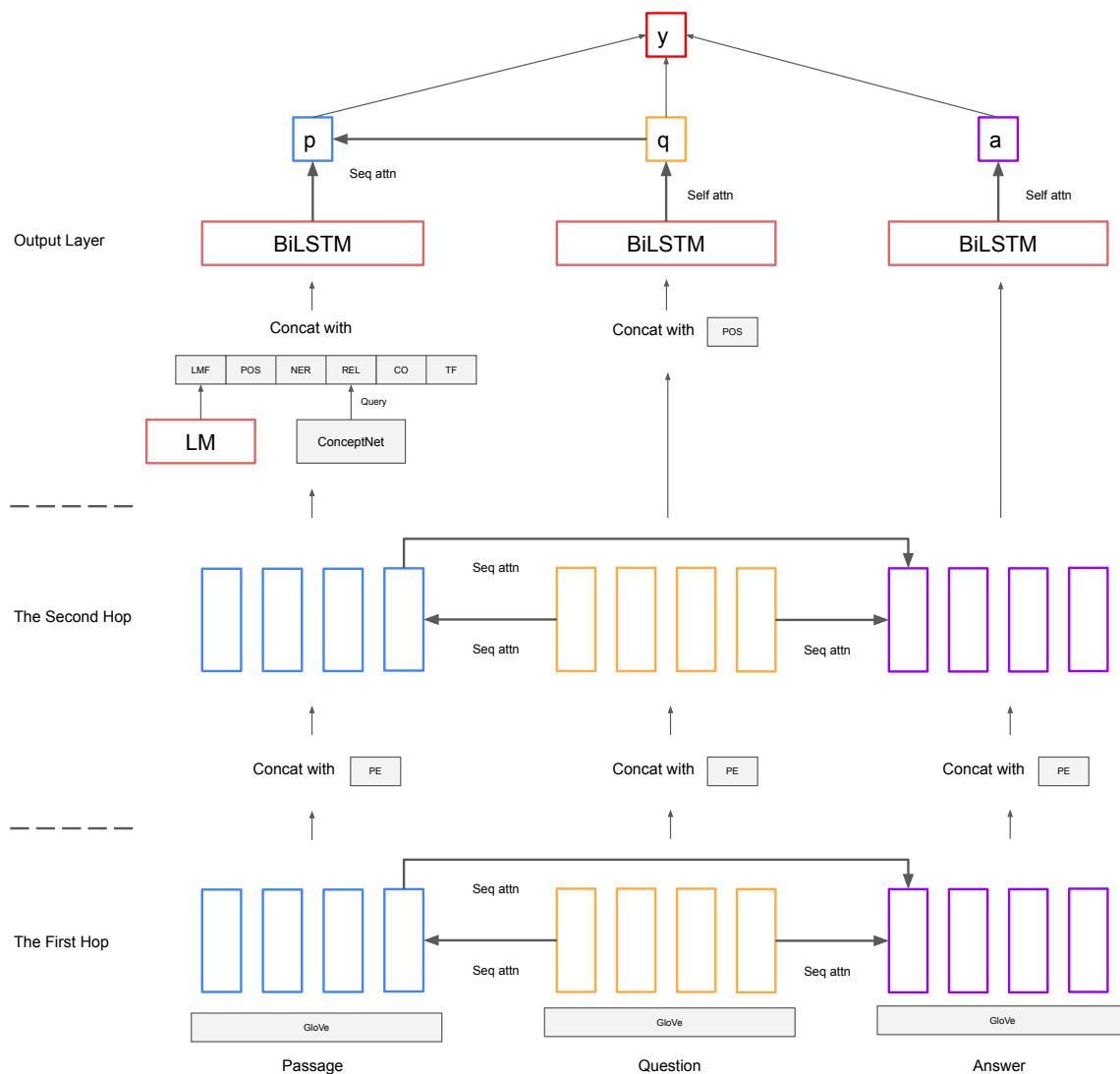


Figure 7. The proposed model architecture.

4. Experiments

4.1. Hyper-Parameters and Training

We followed the same setting of hyper-parameters for the training as was set in [2]. We used GloVe word vectors. We used LSTMs with a hidden size of 96, a dropout rate of 0.4 after the embeddings and LSTM outputs, a clipping gradient of 10, and batch size of 32. We used Adamax as the optimizer and initially set the learning rate to 2×10^{-3} . We decayed it by 0.5 after 10 and 15 epochs.

For the language model pre-training, we used a one-layer LSTM with a hidden size of 256. The maximum sequence length for training was set to 60 words. We used a variational dropout [25,26] to prevent overfitting. Our setup is a simple proof-of-concept; more advanced LM architecture implementation and larger text corpus use are fully compatible.

4.2. Experimental Results

4.2.1. Language Model Samples

We demonstrate a few sample text generated by the LM in Figure 8. Our language model was able to generate plausible consecutive events when given an initial sentence, thereby suggesting that it is able to learn and carry script knowledge.

I went into my bedroom and flipped the light switch: I took the ladder to the light switch. I then cut open the light bulb and turned on the light bulb. I then got a little light bulb ...

I think it is time to do the laundry: I make sure to clean the floor. I will find a nice flat sheet. I put the comforter in the back of the bed. I then put the clothes in the washing machine. I put them in the dryer machine ...

I was going to visit some friends in Florida: I had to go to the airport. I got out my car and started looking in the car. I went to the store and found a seat. The next day I decided to go to the airport. I got out and walked in a parking lot ...

Figure 8. Examples of the generative results of our LSTM language model. The sentences in bold font are the inputs; the subsequent content was completed by the model.

4.2.2. Question-Answering Accuracy

As the dataset we were using was from the SemEval contest, the test set did not come with ground truth labels. Therefore, we evaluated our system against the validation set. As shown in Table 1, our proposed method improved upon the TriANs with respect to the validation accuracy. The best model, which combined all of the three proposed techniques, achieved an 84.22% accuracy, which was 0.8% higher than the baseline model. Upon comparing variants 1, 3, and 4, as well as the proposed model, we notice that all of the introduced techniques contributed to the performance gain. We also studied the interaction between the LM feature and the handcrafted features by comparing the baseline, variant 1, variant 2, and variant 3. While both helped the performance, using the LM features alone seems better than combining the two.

Table 1. Validation accuracy of model variants for the MCScript dataset. We ablated the effect of handcrafted features (Handcrafted Feat.), LM features (LM Feat.), multi-hop attention (Multi-hop), and positional embeddings (PE). We repeated the experiments for 3 times and reported the average and standard deviation numbers.

Variant	Handcrafted Feat.	LM Feat.	Multi-Hop	PE	Accuracy
TriAN (baseline)	✓				83.42% ± 0.03%
Variant 1					82.42% ± 0.04%
Variant 2	✓	✓			83.49% ± 0.04%
Variant 3		✓			84.05% ± 0.06%
Variant 4		✓	✓		84.13% ± 0.05%
Ours		✓	✓	✓	84.22% ± 0.04%

5. Discussion

5.1. Results Analysis

We conducted an ablation study on the components of our method. Comparing the baseline with variant 2 shows that adding the LM features afforded a +0.07% increase in accuracy. Completely replacing the handcrafted features with the LM features (variant 2 vs. 3) performed even better, thus showing a +0.63% increase in accuracy over the baseline. This shows that data-driven features that are learned from a large text corpus are a superior replacement for the handcrafted ones such as word frequency. Variant 3 vs. 4 showed that multi-hop attention could afford a +0.08% increase in accuracy, and adding positional embeddings (variant 4 vs. the proposed method) further afforded a +0.09% increase in accuracy. The most significant gain came from integrating the LM. Our interpretation is that training this LM on a corpus of narrative text in an auto-regressive manner effectively makes it summarize a script knowledge base, which, in return, enables our model to better comprehend the event sequences. The improvements from multi-hop attention show its effectiveness at reasoning indirect connections between the concepts. The further gains from the positional embeddings prove that it is helpful to make word embeddings that are aware of their positions in the passage, which enhances the model's awareness of temporal sequences and causal relationships.

5.2. Future Work

Our framework is compatible with many potential improvements. We list a few here as the future work.

A language model with a higher capacity: Extending upon our LM integration, we can obtain pre-trained BERT [27] embeddings and fine-tune the embeddings to our dataset. As BERT is a more capable model that is trained on a much larger text corpus; the embeddings will implicitly carry a large body of commonsense knowledge, including script knowledge, which are expected to perform better than our current language model features.

A better representation for events: In this study, we encoded the input texts in a per-word manner. A potential alternative is memory networks [28,29] that encode each event at the sentence level. With memory networks, each sentence will have a single embedding so that the attention could be applied at the event level rather than at the word level. That may allow the model to focus more on semantics rather than syntax. Similarly, we would also like to explore training the language model at the sentence level by using the hidden representations to predict which sentence should directly follow the previous sentence.

A stronger knowledge graph: In this study, our REL feature encoded the limited 39 types of relationships from the ConceptNet. It was also bounded by the entities that exist in the ConceptNet. Our framework is compatible with larger knowledge graphs that contain more complex relationships, e.g., the Google Knowledge Graph.

6. Conclusions

In this paper, we presented methods to improve script knowledge integration and reasoning, which included the following: 1. Integrating a generative language model that learns script knowledge from a large text corpus. 2. Using multi-hop attention to support multi-hop reasoning. 3. Using positional embeddings to enhance the reasoning about event sequences and causal relationships. The experiments on the MCScript dataset demonstrate the effectiveness of our framework.

Author Contributions: Conceptualization, Z.Z.; Methodology, Z.Z. and T.Y.; Software, C.L. and D.C.; Investigation, T.Y. and C.H.; Resources, X.B.; Data curation, X.B.; Writing—original draft, Z.Z.; Writing—review & editing, W.W. and C.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sap, M.; Shwartz, V.; Bosselut, A.; Choi, Y.; Roth, D. Commonsense Reasoning for Natural Language Processing. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts, Online, 5–10 July 2020; pp. 27–33. [\[CrossRef\]](#)
2. Wang, L.; Sun, M.; Zhao, W.; Shen, K.; Liu, J. Yuanfudao at SemEval-2018 Task 11: Three-way Attention and Relational Knowledge for Commonsense Machine Comprehension. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 5–6 June 2018; pp. 758–762. [\[CrossRef\]](#)
3. Ostermann, S.; Modi, A.; Roth, M.; Thater, S.; Pinkal, M. MCScript: A Novel Dataset for Assessing Machine Comprehension Using Script Knowledge. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018.
4. Rajpurkar, P.; Zhang, J.; Lopyrev, K.; Liang, P. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 2383–2392. [\[CrossRef\]](#)
5. Dunn, M.; Sagun, L.; Higgins, M.; Guney, V.U.; Cirik, V.; Cho, K. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv* **2017**, arXiv:1704.05179.
6. Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; Deng, L. Ms marco: A human generated machine reading comprehension dataset. *arXiv* **2016**, arXiv:1611.09268.
7. Ostermann, S.; Roth, M.; Modi, A.; Thater, S.; Pinkal, M. SemEval-2018 Task 11: Machine Comprehension Using Commonsense Knowledge. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 5–6 June 2018; pp. 747–757. [\[CrossRef\]](#)
8. Mihaylov, T.; Frank, A. Knowledgeable Reader: Enhancing Cloze-Style Reading Comprehension with External Commonsense Knowledge. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15–20 July 2018; pp. 821–832. [\[CrossRef\]](#)
9. Zhong, W.; Tang, D.; Duan, N.; Zhou, M.; Wang, J.; Yin, J. Improving question answering by commonsense-based pre-training. In Proceedings of the Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, 9–14 October 2019; Proceedings, Part I 8; Springer: Berlin/Heidelberg, Germany, 2019; pp. 16–28.
10. Chen, T.; Wang, X.; Yue, T.; Bai, X.; Le, C.X.; Wang, W. Enhancing Abstractive Summarization with Extracted Knowledge Graphs and Multi-Source Transformers. *Appl. Sci.* **2023**, *13*, 7753. [\[CrossRef\]](#)
11. Xia, J.; Wu, C.; Yan, M. Incorporating Relation Knowledge into Commonsense Reading Comprehension with Multi-Task Learning. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, New York, NY, USA, 3–7 November 2019; pp. 2393–2396. [\[CrossRef\]](#)
12. Speer, R.; Havasi, C. Representing General Relational Knowledge in ConceptNet 5. In Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), Istanbul, Turkey, 23–25 May 2012; pp. 3679–3686.
13. Lin, H.; Sun, L.; Han, X. Reasoning with Heterogeneous Knowledge for Commonsense Machine Comprehension. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; Association for Computational Linguistics, Stroudsburg, PA, USA; pp. 2032–2043. [\[CrossRef\]](#)
14. Tandon, N.; Dalvi, B.; Grus, J.; Yih, W.t.; Bosselut, A.; Clark, P. Reasoning about Actions and State Changes by Injecting Commonsense Knowledge. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 57–66. [\[CrossRef\]](#)
15. Dalvi, B.; Huang, L.; Tandon, N.; Yih, W.t.; Clark, P. Tracking State Changes in Procedural Text: A Challenge Dataset and Models for Process Paragraph Comprehension. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 1595–1604. [\[CrossRef\]](#)
16. Rajani, N.F.; McCann, B.; Xiong, C.; Socher, R. Explain Yourself! Leveraging Language Models for Commonsense Reasoning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 4932–4942. [\[CrossRef\]](#)
17. Richardson, M.; Burges, C.J.; Renshaw, E. MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; Association for Computational Linguistics, Stroudsburg, PA, USA, 2013; pp. 193–203.
18. Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) Doha, Qatar, 25–29 October 2014; pp. 1532–1543. [\[CrossRef\]](#)
19. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Guyon, I.,

- Luxburg, U.V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2017; Volume 30.
20. Song, G.; Leng, B.; Liu, Y.; Hetang, C.; Cai, S. Region-Based Quality Estimation Network for Large-Scale Person Re-Identification. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18, New Orleans, LA, USA, 2–7 February 2018; AAAI Press: Washington, DC, USA, 2018.
 21. Hetang, C. Impression Network for Video Object Detection. In Proceedings of the 2023 IEEE 3rd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China, 26–28 May 2023; Volume 3, pp. 735–743. [\[CrossRef\]](#)
 22. Galassi, A.; Lippi, M.; Torrioni, P. Attention in Natural Language Processing. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 4291–4308. [\[CrossRef\]](#) [\[PubMed\]](#)
 23. He, X.; Zhang, T.; Zhang, G. MultiHop attention for knowledge diagnosis of mathematics examination. *Appl. Intell.* **2023**, *53*, 10636–10646. [\[CrossRef\]](#)
 24. Wang, T.; Huang, R.; Wang, H.; Zhi, H.; Liu, H. Multi-Hop Knowledge Graph Question Answer Method Based on Relation Knowledge Enhancement. *Electronics* **2023**, *12*, 1905. [\[CrossRef\]](#)
 25. Gal, Y.; Ghahramani, Z. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. In Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16, Barcelona, Spain, 5–10 December 2016; Curran Associates Inc.: Red Hook, NY, USA, 2016; pp. 1027–1035.
 26. Jeon, I.; Park, Y.; Kim, G. Neural variational dropout processes. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
 27. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186. [\[CrossRef\]](#)
 28. Weston, J.; Chopra, S.; Bordes, A. Memory networks. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
 29. Abdelrahman, G.; Wang, Q. Deep graph memory networks for forgetting-robust knowledge tracing. *IEEE Trans. Knowl. Data Eng.* **2022**, *35*, 7844–7855. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.