

Logiformer: A Two-Branch Graph Transformer Network for Interpretable Logical Reasoning

Fangzhi Xu

School of Computer Science and
Technology, Xi'an Jiaotong University
Xi'an, China
Leo981106@stu.xjtu.edu.cn

Jun Liu*

Shaanxi Province Key Laboratory of
Satellite and Terrestrial Network Tech.
R&D, National Engineering lab for
Big Data Analytics
Xi'an, China
liukeen@xjtu.edu.cn

Qika Lin

School of Computer Science and
Technology, Xi'an Jiaotong University
Xi'an, China
qikalin@foxmail.com

Yudai Pan

School of Computer Science and
Technology, Xi'an Jiaotong University
Xi'an, China
pyd418@foxmail.com

Lingling Zhang

School of Computer Science and
Technology, Xi'an Jiaotong University
Xi'an, China
zhanglingling@xjtu.edu.cn

ABSTRACT

Machine reading comprehension has aroused wide concerns, since it explores the potential of model for text understanding. To further equip the machine with the reasoning capability, the challenging task of logical reasoning is proposed. Previous works on logical reasoning have proposed some strategies to extract the logical units from different aspects. However, there still remains a challenge to model the long distance dependency among the logical units. Also, it is demanding to uncover the logical structures of the text and further fuse the discrete logic to the continuous text embedding. To tackle the above issues, we propose an end-to-end model Logiformer which utilizes a two-branch graph transformer network for logical reasoning of text. Firstly, we introduce different extraction strategies to split the text into two sets of logical units, and construct the logical graph and the syntax graph respectively. The logical graph models the causal relations for the logical branch while the syntax graph captures the co-occurrence relations for the syntax branch. Secondly, to model the long distance dependency, the node sequence from each graph is fed into the fully connected graph transformer structures. The two adjacent matrices are viewed as the attention biases for the graph transformer layers, which map the discrete logical structures to the continuous text embedding space. Thirdly, a dynamic gate mechanism and a question-aware self-attention module are introduced before the answer prediction to update the features. The reasoning process provides the interpretability by employing the logical units, which are consistent

with human cognition. The experimental results show the superiority of our model, which outperforms the state-of-the-art single model on two logical reasoning benchmarks.¹

CCS CONCEPTS

• **Information systems** → *Question answering; Language models; Information extraction.*

KEYWORDS

logical reasoning, machine reading comprehension, graph transformer

ACM Reference Format:

Fangzhi Xu, Jun Liu, Qika Lin, Yudai Pan, and Lingling Zhang. 2022. Logiformer: A Two-Branch Graph Transformer Network for Interpretable Logical Reasoning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*, July 11–15, 2022, Madrid, Spain. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3477495.3532016>

1 INTRODUCTION

Machine reading comprehension [10, 18, 39] has been one of the major focuses in the field of Natural Language Processing (NLP) [4, 9] in recent years. A large number of models have achieved competitive performances in some famous datasets, such as SQuAD[23, 24], RACE[14]. However, these models [7, 27, 36] lack the capability of logical reasoning. To facilitate the machine for human intelligence, the task of logical reasoning MRC [17, 37] was proposed previously. Similar to the traditional MRC, the task of logical reasoning also requires the models to predict the answers depending on the given text inputs. Figure 1 illustrates a logical reasoning example from ReClor dataset [37]. The inputs include the context, question and a set of options. One of the unique characteristics of the text is the rich logical structures. As illustrated in Figure 1, the logical structure of the context can be uncovered in a certain way. We define the split short sentences as logical units (e.g., U1-U6). The logical units contain the independent and complete semantics, which are

¹The code is public in <https://github.com/xufangzhi/Logiformer>.

*The corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR '22, July 11–15, 2022, Madrid, Spain.

© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-8732-3/22/07...\$15.00
<https://doi.org/10.1145/3477495.3532016>

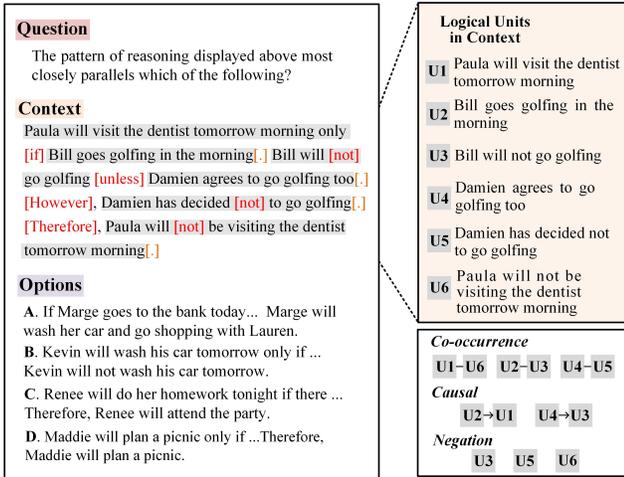


Figure 1: An example of the logical reasoning task and some detailed illustrations.

not kept in the token-level text features. The understanding of the text requires the global semantics of each logical unit, as well as the interactions among them based on some logical relations (e.g., *causal* and *co-occurrence*). Therefore, the main challenges for the task of logical reasoning can be summarized as the following two aspects.

Firstly, it remains a challenge to model the long distance dependency [11] of the extracted logical units. Some previous methods, such as DAGN [12], have proposed to split the text into discourse nodes [32] and constructed a sequential chain graph for reasoning. However, it neglects the natural long-distance dependency among logical units. For example in Figure 1, the first and the last sentences share the same subject (*Paula*) and predicate (*visit the dentist*), though they are distant in the graph space. The chain structure limits the information update. In a word, the simple graph structure built for the logical text would fail to provide the efficient one-hop interaction [25]. Pretrain-based transformer structures [30] have the natural advantage of modeling the long text and show excellent performance on the popular tasks. To enhance the logical perception ability of the language models, previous works have attempted to employ additional segment embedding at the beginning. However, it is still limited to the token-level interactions, which sacrifices the global semantics of logical units. Take the first sentence of the context in Figure 1 as an instance, two extracted units of *Paula will visit the dentist tomorrow morning*(U1) and *Bill goes golfing in the morning*(U2) express the causal relations within this sentence. The token-aware models would stress more on the text semantics and fail to capture such logical information.

Secondly, it is intractable to bridge the gap between the discrete logical structures and the continuous text embedding space. Take a closer observation of the logical units in Figure 1, the units are not completely separated. In this work, we pay much attention to the two explicit relationships (*causal* and *co-occurrence*). We summarize them into the logical branch and syntax branch respectively. From the logical branch, connectives (e.g., *if*, *unless*, *Therefore*) play a significant role in covering the logical relations. For example in

Figure 1, the logical units U1 and U2 are connected with the connective *if* while U3 and U4 share the connective *unless*. From the syntax branch, the logical units are not independent but have some repeating occurrences. For example, the units of *Bill goes golfing in the morning* and *Bill will not go golfing* belong to the co-occurrence information, which have strong correlations. Thus, these syntactic relationships help make the corresponding logical units closer in structure. The above connection structures from two branches are discrete, which is incompatible to the continuous text representation. Some early works simply feed the text to the Pretrained Language Models (PLMs) [5, 33] and rely on the context to learn the logical semantics. But it includes much noise in the text embedding and lacks the potential of interpretability. Previously, LReasoner [31] proposes a method to transform the logical expressions to text based on the templates and feeds the extended text into PLM. However, it still embeds the logic in an implicit form and fails to make up for the weakness of PLMs in logical reasoning. Some works like FocalReasoner [21] uncovers the logical structure in only one aspect (e.g., capture the co-occurrence between units). It leads to the weak capability of the model to capture logical relationships.

In light of the above challenges, we propose a novel model named **Logiformer** which is an end-to-end architecture with graph transformer for interpretable logical reasoning of text. By employing the fully connected graph transformer structure to enhance the direct interactions, we tackle the issue of long distance dependency among the logical units. To encode the discrete logical structures to the continuous text embedding space, we apply the attention biases from both the logical and syntax branches. The whole reasoning is on the basis of logical units and the built graphs, which are consistent with the human cognition. The explicit relations among units and the weighted attention maps provide the interpretability for the logical reasoning. In details, firstly, Logiformer split the text into logical units and construct the logical graph based on the causal relations for the logical branch. For the syntactic branch, the split nodes and a syntax graph are also obtained. Secondly, we feed the node sequences and two graph topology to the fully connected graph transformers respectively. The respective adjacent matrices are viewed as the attention biases to encode the logical structures to each graph transformer. Thirdly, we combine the updated features from two branches with a dynamic gate mechanism. With the additional token-level embedding, we can map the features to the same space. By means of the question-aware self-attention, the final feature can be utilized to predict the answers.

The main contributions are listed as follows:

- A two-branch graph transformer network named Logiformer is proposed to model the long distance dependency of the logical units and encode the discrete logical structure to the continuous text embedding. As far as we know, we are the first to tackle both issues in the logical reasoning task.
- In light of drawbacks of chain-type text graphs, we take the fully connected structures into consideration, containing the awareness of both logic and syntax simultaneously. Two graphs are constructed based on the extracted logical units and their topology is utilized as the attention biases.
- The extraction of the logical units and the explicit relations are consistent with the human cognition. The uncovered

logical structures and the weighted attention maps of the logical units provide the excellent interpretability for the logical reasoning process.

- Extensive experiments show that Logiformer outperforms the state-of-the-art (SOTA) results with single model on two logical reasoning datasets. Furthermore, ablation studies prove the effectiveness of each module in our model.

2 RELATED WORK

In this section, we will introduce the current researches on MRC and logical reasoning.

2.1 Machine Reading Comprehension

Recent years have witnessed the rapid growth of MRC[18], where the model is required to infer the answers based on the given context and a question. A variety of datasets have been proposed to check the performances of MRC models. Among them, SQuAD[23, 24] focuses on the span extractions on the factual questions. HotpotQA[34] and OpenBookQA[20] require the multi-hop reasoning capability of the models. A couple of multiple choice datasets like RACE[14] cover the examinations for middle or high school students. Some representative models achieve great success on these datasets. RetroReader [41] applies a two-state strategy to solve the questions. But it mainly investigates the overall interactions of the context and question, which fails to deal with the complex logic within the text. SG-Net [40] integrates the syntax information into the self-attention module to improve the performance, but it does not show the potential on tackling the logical information. Generally speaking, the datasets mentioned above rely much on the token-level matching, which can be well tackled with large-scale pretraining models like BERT[6] and GPT-3[2]. To make the models closer to the human intelligence, it is necessary to introduce more challenging tasks requiring logical reasoning. Previously, the task of Natural Language Inference(NLI)[1, 28] is proposed to motivate the models to infer the relations(i.e., Contradiction, Entailment and Neutral) between two sentences. Nevertheless, it is limited by the fixed inputs and outputs and fails to extend the task to more complex settings.

2.2 Logical Reasoning

To improve the reasoning ability of the models, several datasets on multiple choice have been proposed previously. ReClor[37], which is extracted from standardized graduate admission examinations and law school admission test, has aroused wide concerns. For better evaluation, it separates the biased examples into EASY set and the challenging ones into HARD set. LogiQA [17] is also one of the representatives, which also aims to improve the logical reasoning capability. It is sourced from expert-written questions and covers multiple types of deductive reasoning. Experiments show that previous SOTA models on traditional MRC perform bad on the two datasets. Under such circumstances, some of the recent works attempt to enhance logical reasoning from different perspectives. DAGN[12] proposes a reasoning network based on the discourse units extracted from the text. But it simply forms a chain-type discourse network and weakens the relations between two distant units. FocalReasoner [21] stresses that fact units in the form of subject-verb-object are significant for logical reasoning. It constructs a supergraph on top of the fact units and updates the node

Table 1: The set of logical units from the example context (split by connectives and punctuations).

Symbol	Logical Units
U_1	Paula will visit the dentist tomorrow morning
U_2	Bill goes golfing in the morning
U_3	Bill will not go golfing
U_4	Damien agrees to go golfing too
U_5	Damien has decided not to go golfing

features relying on Graph Neural Network. However, it ignores the relation connectives from the text and lacked the logical modeling. LReasoner [31] focuses on capturing symbolic logic from the text and puts forward a context extension framework based on logical equivalence laws. However, it relies heavily on the language models for token-level embedding and neglects the sentence-level interactions.

3 METHODOLOGY

This section will introduce the proposed end-to-end model Logiformer. The architecture of Logiformer is shown in Figure 2. The left part of the model is an example of the logical reasoning task. The understanding of text will be divided into two branches: logical branch (upper) and syntactic branch (lower). This architecture mainly includes the following three parts: a) graph construction from the text; b) logical-aware and syntax-aware graph transformers for feature updates; c) the decoder including a dynamic gate mechanism and a question-aware self-attention module.

3.1 Task Formulation

Given a dataset \mathcal{D} for logical reasoning, which consists of N examples totally. The inference of the i^{th} question can be formulated as follows:

$$\hat{a} = \arg \max_{a_{i,j} \in A_i} p(a_{i,j} | c_i, q_i, A_i; \theta), \quad (1)$$

where c_i, q_i, A_i represent the context, question sentence and candidate set respectively. The number of options in A_i is $n, j \in [0, n-1]$ and $a_{i,j} \in A_i$ represents the j^{th} option. \hat{a} denotes the predicted option. θ denotes the trainable parameters.

Since the current methods mainly focus on the token-level representation of the text with the help of PLMs, they will naturally ignore some global semantics for each sentence or phrase. To capture the global feature within each sentence, we first obtain the text fragments, which are split by connectives or punctuations. We define the text fragment that reflect the complete semantic of an event or argument as the logical unit with the symbol U . Take the context in Figure 2 as an instance, we split the text by both connectives and punctuations and obtain a set of logical units shown in Table 1.

Considering that there exist explicit causal relations between units, we further introduce the conditional connective ' \rightarrow '. And in some cases, it is required to reverse logical units for the negation expression, we also employ the operation ' \neg '. Combining the logical units and causal connections in the form of conjunction, we can derive the logical expression of the text:

$$(U_2 \rightarrow U_1) \wedge (U_4 \rightarrow \neg U_3) \wedge \neg U_5. \quad (2)$$

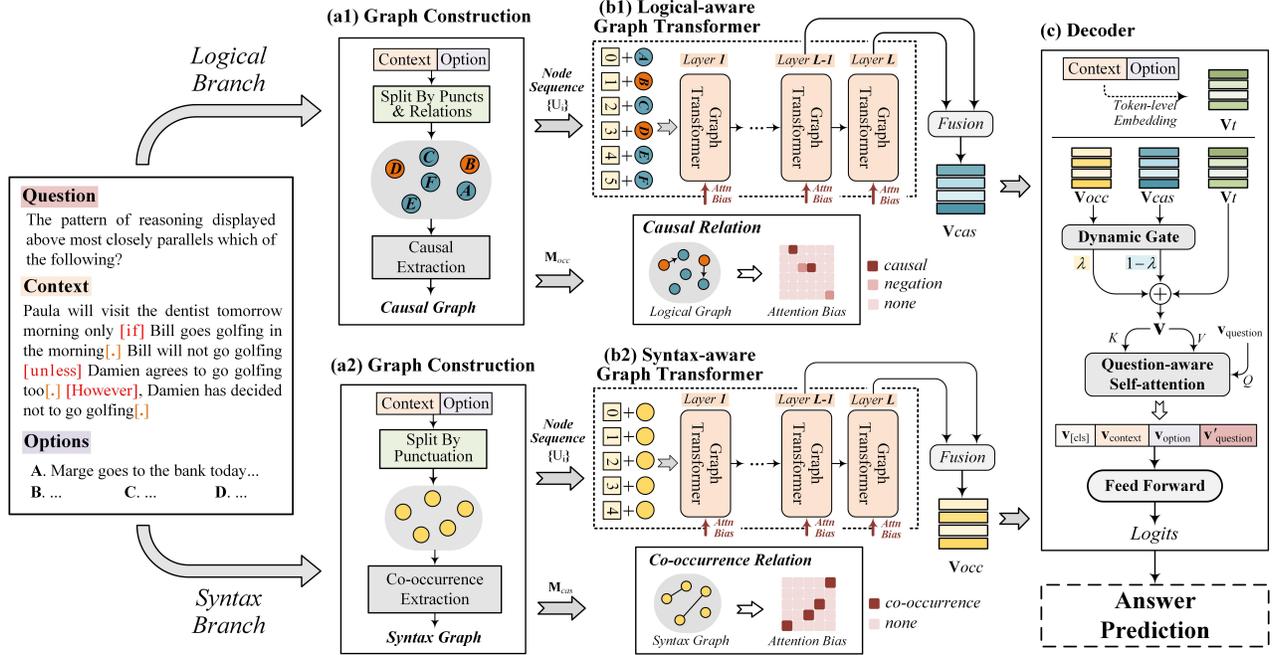


Figure 2: The architecture of Logiformer. The left part is an input example of the dataset. The graph construction modules (a1,a2) split the text into logical units and build two graphs from two branches respectively. The graph transformer structures (b1,b2) update the text features combined with the logical and syntactic relations. Finally, the decoder module (c) is utilized to conduct the feature fusion and predict the answers.

Obviously, there exist two key components in the logical expression: i) logical units U_k ; ii) logical connectives, i.e., \rightarrow and \neg . The former one focuses on the syntactic information, while the latter one is more related to logical structure of the context.

3.2 Graph Construction

Given the i^{th} inputs, Logiformer first concatenates the context c_i with each option $a_{i,j}$ respectively to form the input sequences. According to the previous analysis, Logiformer will tackle the inputs from two branches (i.e., logical branch and syntax branch) and build two graphs (i.e., logic graph and syntax graph) respectively.

3.2.1 Logical Graph. For the logical branch, Logiformer mainly concentrates on the causal relations. Considering that the causal relation often appears with explicit logical words such as *if*, *unless*, *because*, we can leverage the explicit logical words as the basis of split. Therefore, we include 100 commonly used logical words according to PDTB 2.0 [22].

Combining the explicit logical words and punctuations, we can separate the text sequence into logical units. Each unit serves as a node for future updates. Especially, we pick out the nodes pairs connected by the explicit causal relation words and name them as *condition node* (orange nodes) and *result node* (blue nodes). Meanwhile, we classify the common nodes which do not contain causal relations into *result node* (blue nodes). Thus, we obtain the node set from the perspective of logic.

According to the extracted causal node pairs, we can create directed connection from each condition node p to result node q .

This kind of connection is reflected in the adjacent matrix $M_{cas} \in \mathbb{R}^{K_{cas} \times K_{cas}}$ of the logical graph as $M_{cas}[p-1, q-1] = 1$.

Also, to avoid the semantic reverse brought by the negation, we mark the nodes with the explicit negation words (e.g., *not*, *no*). The node k with negation semantics are expressed in the adjacent matrix as $M_{cas}[k-1, k-1] = -1$.

Therefore, the logical graph has the perception of the causal relations and negations. And the obtained adjacent matrix $M_{cas} \in \mathbb{R}^{K_{cas} \times K_{cas}}$ of the logical graph is asymmetric.

3.2.2 Syntax Graph. The main purpose of the syntactic understanding is to capture the inner relations between the logical units U_k . Noticing that some logical units share the common words or phrases in Figure 2, e.g., *Bill*, *Damien* and *go golfing*. It illustrates that the text has a strong characteristics of co-occurrence. Also, co-occurrence usually exists between two complete sentences. Therefore, we consider to split the text sequence only by punctuations and obtain a set of sentence nodes with no original connection. It is required to extract the co-occurrence between the sentence nodes. As each node consists of its related tokens, we propose a simple strategy to capture the co-occurrence, shown in Algorithm 1.

Assume the total number of the nodes to be K_{occ} . The input for the algorithm is the sentence node U_k , corpus C_s containing redundant stop words and hyper-parameter δ . The output is an adjacent matrix $M_{occ} \in \mathbb{R}^{K_{occ} \times K_{occ}}$, which reflects the co-occurrence relations between nodes. As for any two nodes, we transform them into two token sets Set_k and Set_j separately, without order and duplicate elements (Line 3 & Line 5 in Algorithm 1). We define $len(Set)$ to be the number of tokens in a set. Further, let the token overlap ratio

Algorithm 1: Co-occurrence Extraction

Input: Sentence Nodes U_k ($k \in \{1, 2, \dots, K_{occ}\}$), hyper-parameter δ , stop words corpus C_s
Output: Co-occurrence Matrix M_{occ}

- 1 Initialize co-occurrence matrix $M_{occ} \in \mathbb{R}^{K_{occ} \times K_{occ}}$ to zero.
- 2 **for** $k = 1, 2, \dots, K_{occ}$ **do**
- 3 Include all the tokens of U_k into Set_k . Meanwhile, exclude stop words from Set_k based on C_s
- 4 **for** $j = k + 1, k + 2, \dots, K_{occ}$ **do**
- 5 Include all the tokens of U_j into Set_j and exclude stop words from Set_j based on C_s
- 6 /* Ensure the two sets not empty */
- 7 **if** $len(Set_k) > 0$ and $len(Set_j) > 0$ **then**
- 8 $B \leftarrow \min\{len(Set_k), len(Set_j)\}$
- 9 $overlap \leftarrow len(Set_k \& Set_j) / B$
- 10 **if** $overlap > \delta$ **then**
- 11 $M_{occ}[k - 1, j - 1] \leftarrow 1$
- 12 $M_{occ}[j - 1, k - 1] \leftarrow 1$
- 13 **end**
- 14 **end**
- 15 **end**
- 16 **return** Co-occurrence Matrix M_{occ}

of two sets be the co-occurrence metric (Line 7 & Line 8). Thus, we can determine the co-occurrence relation between U_k and U_j if the overlap score exceeds the threshold.

So far, the connections within the sentence nodes have been explored based on the co-occurrence relations. Thus, the syntax graph is constructed reflected by the obtained adjacent matrix $M_{occ} \in \mathbb{R}^{K_{occ} \times K_{occ}}$.

3.3 Graph Transformer

Some previous works [8, 38] point out the drawbacks of graph neural network, such as the issue of over-smooth [15]. Therefore, we take the novel architecture of graph transformer [3, 35] into account. After the extraction of nodes and the construction of two graphs, we feed them into the logical-aware and syntax-aware graph transformer structures respectively.

3.3.1 Logical-aware Graph Transformer. The simple illustration of the logical-aware graph transformer is shown in Figure 3. First of all, it is necessary to get the original feature embedding for each node. Given the concatenated input sequence of the i^{th} question:

$$\text{Input}(c_i, a_{i,j}) = [\text{CLS}]c_i[\text{SEP}]a_{i,j}[\text{SEP}], \quad (3)$$

we employ the RoBERTa model [19] as the encoder for the token-level features. For the token sequence $\{t_1^{(k)}, t_2^{(k)}, \dots, t_T^{(k)}\}$ with the length T of each node U_k , the obtained token embedding is represented as $\{v_{t_1}^{(k)}, v_{t_2}^{(k)}, \dots, v_{t_T}^{(k)}\}$. We take the average embedding of T tokens as the original feature for node U_k :

$$v_k = \frac{1}{M} \sum_{i=1}^M v_i^{(k)}. \quad (4)$$

To keep the original order information of nodes in the text, positional embedding is added to the node representation.

$$V_i = V_o + \text{PosEmbed}(V_o), \quad (5)$$

where $V_o = [v_1; v_2; \dots; v_{K_{cas}}]$, $V_o \in \mathbb{R}^{K_{cas} \times d}$, d is the dimension of the hidden state, and K_{cas} is the number of nodes. $\text{PosEmbed}(\cdot)$ provides a d -dimensional embedding for each node in the input sequence.

We feed the node representation V_i into the logical-aware graph transformer. Firstly, V_i is projected to three matrices Q, K and V of the self-attention module:

$$\begin{aligned} Q &= V_i \cdot W^Q, \\ K &= V_i \cdot W^K, \\ V &= V_i \cdot W^V, \end{aligned} \quad (6)$$

where $W^Q, W^K, W^V \in \mathbb{R}^{d \times d_k}$ are projection matrices, and the obtained matrices $Q, K, V \in \mathbb{R}^{K_{cas} \times d_k}$. Then, we compute the attention based on the query, key and value matrices.

$$A = \frac{QK^T}{\sqrt{d_k}}, \quad (7)$$

$$\text{Att}(Q, K, V) = \text{softmax}(A) \cdot V,$$

where $A \in \mathbb{R}^{K_{cas} \times K_{cas}}$ is a weight matrix for node pairs. From the equations, the transformer structure provides a fully connected setting to all nodes, which ignores the inner causal relations. Therefore, Logiformer employs the obtained topology information $M_{cas} \in \mathbb{R}^{K_{cas} \times K_{cas}}$ of the logical graph as an attention bias. The representation of the weight matrix A is adjusted as follows:

$$A' = \frac{QK^T}{\sqrt{d_k}} + M_{cas}. \quad (8)$$

To improve the robustness and capability of the attention module, we apply the multi-head attention mechanism with the head number H :

$$\text{Att}_{MH}(Q, K, V) = [\text{Head}_1; \dots; \text{Head}_H] \cdot W^H, \quad (9)$$

where $W^H \in \mathbb{R}^{(H \cdot d_k) \times d_k}$ is the linear projection matrix, $\text{Head}_i = \text{Att}_i(Q, K, V)$, the input query, key and value matrices are obtained by the linear projections of $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d_k}$ respectively. For simplicity, we assume $d = d_k$ and omit the bias term of the linear projection.

Repeating the multi-head attention for L layers, we take out the hidden states of the last two layers. To enhance the robustness of the model, we make a fusion of them as the updated node features:

$$V_{cas} = V_{cas}^{(L-1)} + V_{cas}^{(L)}, \quad (10)$$

where $V_{cas} \in \mathbb{R}^{K_{cas} \times d}$, and $V_{cas}^{(L-1)}, V_{cas}^{(L)} \in \mathbb{R}^{K_{cas} \times d}$ represents the hidden states of the last two layers respectively. Note that there are lots of ways of feature fusion, we only present the simple addition for illustration.

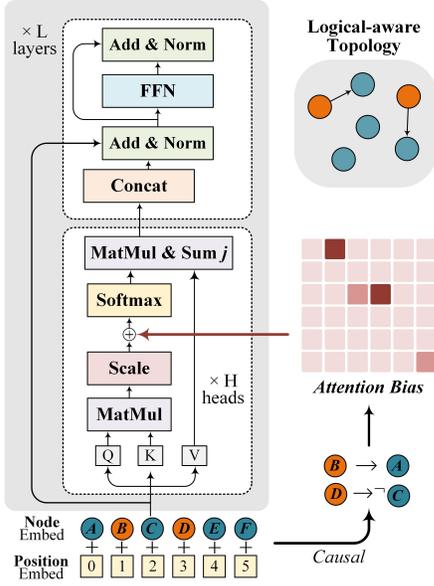


Figure 3: The illustration of logical-aware graph transformer. The inputs are the node sequence as well as the topology and the outputs are omitted.

3.3.2 Syntax-aware Graph Transformer. The general idea of the syntax-aware graph transformer is similar to that of the logical-aware graph transformer. The initialization of node features are also obtained by averaging the embedding of each token. And the positional information is kept to provide the perception of original text orders. After obtaining the weight matrix from the self-attention module, we apply the adjacent matrix M_{occ} to provide the attention bias for the weights. The final feature of syntax-branch node sequence is also obtained through the fusion of last two layers, represented as $V_{occ} \in \mathbb{R}^{K_{occ} \times d}$.

3.4 Decoder

So far, we have obtained the token-level representation $V_t \in \mathbb{R}^{N \times d}$, syntax branch node representation $V_{occ} \in \mathbb{R}^{K_{occ} \times d}$ and logical branch node representation $V_{cas} \in \mathbb{R}^{K_{cas} \times d}$. To make comprehensive use of the advantages of the three features, it is required to ensure dimension consistency. Therefore, we broadcast the feature of each node to all the tokens it contains. The updated features are transformed to $V_t, V'_{occ}, V'_{cas} \in \mathbb{R}^{N \times d}$.

The above three features with the same dimension do not necessarily make equal contributions to the final prediction. It will be beneficial to automatically assign the importance. Thus, Logiformer will dynamically generate the fusion weights for V'_{occ} and V'_{cas} . The gate parameter λ is expressed as follows:

$$\lambda = \text{softmax}([V'_{occ}; V'_{cas}]W_g + b_g), \quad (11)$$

where $W_g \in \mathbb{R}^{2d \times 1}$ and $b_g \in \mathbb{R}^{N \times 1}$ are the weight term and bias term for the linear projection. V'_{occ} and V'_{cas} are concatenated at the last dimension. The obtained gate parameter $\lambda \in \mathbb{R}^{N \times 1}$ is a vector. That is to say, each token will be provided with one specific weight.

Table 2: Detailed Splits of ReClor and LogiQA.

Dataset	#Train	#Valid	#Test	#Reason Type
ReClor	4,638	500	1,000	17
LogiQA	7,376	651	651	5

The final feature V can be represented in the following expression:

$$V = LN(V_t + \lambda \cdot V'_{occ} + (1 - \lambda) \cdot V'_{cas}), \quad (12)$$

where $LN(\cdot)$ denotes the layer normalization operation. Since we do not employ the global node in the graph transformer, the global feature will not be updated. To this end, Logiformer integrates the local token-level features and gets the updated global information:

$$V_{cls} = LN(V_{t,cls} + \frac{1}{N-1} \sum_{i=1}^{N-1} (V'_{occ,i} + V'_{cas,i})), \quad (13)$$

where $V_{t,cls}$ is the first token of the original token-level embedding. $V'_{occ,i}$ and $V'_{cas,i}$ represent the i^{th} token embedding of the syntactic branch and logical branch feature respectively. We utilize the global feature V_{cls} to replace the first token feature (i.e., [cls] feature) of V . V can be expressed as the concatenation of V_{cls} , $V_{context}$ and V_{option} , that is $V = [V_{cls}; V_{context}; V_{option}]$.

To conduct the reasoning, the feature of the question $V_{question}$ is also of great significance. Logiformer applies a simple self-attention module for the global feature V and question $V_{question}$. The updated question embedding is expressed as:

$$V'_{question} = \text{softmax}\left(\frac{V_{question}V^T}{\sqrt{d}}\right) \cdot V. \quad (14)$$

For simplicity, the linear projections for the self-attention are omitted. At last, we concatenate the V_{cls} , $V_{context}$, V_{option} and $V'_{question}$ to get the final feature $V_{final} \in \mathbb{R}^{N \times d}$:

$$V_{final} = [V_{cls}; V_{context}; V_{option}; V'_{question}]. \quad (15)$$

For each option in one example, we can get one specific final feature. They are fed into the feed forward network to obtain the scores, and we take the highest one as the predicted answer.

4 EXPERIMENTS

In this section, extensive experiments are conducted to compare our model with SOTA single model methods in both ReClor and LogiQA datasets. Ablation studies are followed to verify the effectiveness of the proposed modules.

4.1 Datasets and Baselines

4.1.1 Datasets. In this paper, we conduct the experiments on two logical reasoning datasets ReClor [37] and LogiQA [17]. ReClor consists of 6,138 examples sourced from some standardized tests, while LogiQA includes totally 8,678 questions collected from National Civil Servants Examinations of China. The detailed splits of both datasets are included in Table 2. It can be concluded that ReClor is more diverse in the number of logical reasoning types, while LogiQA contains more examples. Both of them are challenging for the task of logical reasoning.

Table 3: The tuned hyper-parameters with search scopes.

Name of Parameter	Search Scope	Best
training batchsize	{1,2,4,8}	2
#epoch	{9,10,11,12,13}	12
#head in graph transformer	{4,5,6,7,8}	5
#layer in graph transformer	{4,5,6,7,8}	5
max sequence length	{128,256,512}	256
learning rate for RoBERTa	{4e-6, 5e-6, 6e-6, 5e-5}	5e-6

4.1.2 **Baselines.** To prove the superiority of our model, we mainly employ the following baselines, including the SOTA method of single model.

- **Random:** The results are based on the random predictions.
- **Human Performance**[17, 37]: For ReClor, human performance is defined as the average score of different graduate students in a university on the test split. For LogiQA, the result is the average score of three post-graduate students on 500 randomly selected instances from the test split.
- **DAGN** [12]: It proposed a discourse-aware network, which took RoBERTa-Large[19] as the token encoder and employed GNN for the feature update.
- **FocalReasoner** [21]: It focused on the fact units extracted from the text and built a supergraph for the reasoning. Similar to DAGN, it also leveraged RoBERTa-Large and GNN [26] for the token embedding and node update respectively.
- **LReasoner** [31]: It captured the symbolic logic from the text and further extended them into natural language based on several logical equivalence laws. For the fair comparison, we take the results of the single model with RoBERTa encoder into consideration.

4.2 Implementation Details

All of the experiments are conducted with a single GPU of Tesla V100. For the fair comparison, the RoBERTa-large model [19] is utilized as the encoder for text during the experiments and the hidden size is set to 1024. During the training process, the epoch number is fixed to 12 and the batchsize is set to 2 for both ReClor and LogiQA datasets. We take Adam [13] with linearly-decayed learning rate and warm up and select peak learning rate as 5e-6. We select the model with best accuracy on the validation split to conduct the test. The details of important hyper-parameters and their search scopes are attached in Table 3.

4.3 Comparison Results

Logiformer is evaluated on two logical reasoning datasets. The main results on the validation split and test split of ReClor dataset are shown in Table 4. And the results on LogiQA dataset are shown in Table 5. The test split of ReClor is organized into easy fold and hard fold, presented as ‘Test-E’ and ‘Test-H’ respectively in the table.

For the fair comparison, we consider the results with single model and with the encoder of RoBERTa for all the baselines. Compared with the SOTA results on two logical reasoning benchmarks, our proposed model Logiformer shows excellent improvements.

On the ReClor dataset, we witness the improvements of 2.20% and 1.10% on the validation and test split over previous SOTA model LReasoner. Since LReasoner does not make the results on Test-E

Table 4: Experimental results on ReClor dataset. The percentage signs (%) of accuracy values are omitted. The optimal and sub-optimal results are marked in bold and underline respectively (same for the following tables).

Model	Valid	Test	Test-E	Test-H
Random	25.00	25.00	25.00	25.00
Human Performance[37]	-	63.00	57.10	67.20
BERT-Large [37]	53.80	49.80	72.00	32.30
XLNet-Large[37]	62.00	56.00	75.70	40.50
RoBERTa-Large [37]	62.60	55.60	75.50	40.00
DAGN [12]	65.80	58.30	75.91	44.46
FocalReasoner [21]	<u>66.80</u>	58.90	77.05	44.64
LReasoner [31]	66.20	<u>62.40</u>	-	-
Logiformer	68.40	63.50	79.09	51.25

Table 5: Experimental results on LogiQA dataset.

Model	Valid	Test
Random	25.00	25.00
Human Performance[17]	-	86.00
BERT-Large [17]	34.10	31.03
RoBERTa-Large [17]	35.02	35.33
DAGN [12]	36.87	39.32
FocalReasoner [21]	<u>41.01</u>	<u>40.25</u>
Logiformer	42.24	42.55

and Test-H splits public, we omit the comparison. Compared with FocalReasoner on the validation split, Logiformer shows strong generalization capability with 1.6% and 4.6% improvements on the validation and test split. Especially on the Test-H split, 6.61% improvement proves our superiority for the more difficult logical reasoning questions. The most important observation is that Logiformer is the first single model with RoBERTa encoder to beat the human performance by 0.50% on the ReClor dataset. Although the machine still falls behind humans on more challenging questions, our proposed method is positively narrowing the gaps.

On the LogiQA dataset, Logiformer outperforms the previous SOTA model Logiformer by 1.13% and 2.30% on the validation and test split respectively. It proves the excellent generalization capability of Logiformer. However, we also discover the huge gap between humans and the machine. In view that the context in LogiQA dataset is organized in a more structural form, humans are easier to capture the inner logic. The deep learning based models are good at capturing the semantic changes and lack the perception of fixed logic.

4.4 Ablation Studies

Considering that the architecture of Logiformer is mainly divided into three parts: a) graph construction, b) graph transformer and c) decoder, the ablation studies are also laid out from these three aspects. The experimental results are shown in Table 6.

Firstly, in the part of graph construction, we build syntax graph and logical graph based on the different node extraction strategies. We ablate the effects of the two graphs in turn. That is to say, we only consider one of the branches each time. From the results, the logical graph contributes more to the performance on the ReClor dataset, which improves 4.80% and 3.60% on the validation and test

Table 6: Ablation Studies. The improvements on the accuracy are marked in red.

Model	ReClor						LogiQA			
	Valid	Δ	Test	Δ	Test-E	Test-H	Valid	Δ	Test	Δ
Logiformer	68.40	-	63.50	-	79.09	51.25	42.24	-	42.55	-
a) Graph Construction										
w/o syntax graph	66.40	-2.00	61.20	-2.30	77.50	48.39	38.56	-3.68	38.71	-3.84
w/o logical graph	63.60	-4.80	59.90	-3.60	75.00	48.04	38.25	-3.99	37.63	-4.92
b) Graph Transformer										
w/o co-occurrence bias	66.80	-1.60	62.80	-0.70	77.05	51.61	41.94	-0.30	42.55	-
w/o causal bias	65.20	-3.20	63.30	-0.20	76.82	52.68	39.94	-2.30	41.47	-1.08
w/o both of attention biases	66.20	-2.20	61.60	-1.90	75.23	50.89	41.63	-0.61	39.94	-2.61
c) Decoder										
w/o dynamic gates	67.00	-1.40	61.90	-1.60	76.14	50.71	41.32	-0.92	42.55	-
w/o question-aware attention	66.60	-1.80	60.40	-3.10	76.36	47.86	41.63	-0.61	42.09	-0.46

Table 7: The details of ReClor Test Split on different question types. NA: Necessary Assumption, S:Strengthen, W:Weaken, I:Implication, CMP:Conclusion/Main Point, MSS:Most Strongly Supported, ER:Explain or Resolve, P:Principle, D:Dispute, R:Role, IF:Identify a Flaw, O:Others.

Model	NA	S	W	I	CMP	MSS	ER	P	D	R	IF	O
Logiformer	74.56	64.89	55.75	45.65	75.00	66.07	61.90	69.23	70.00	75.00	58.12	60.27
w/o syntax graph	70.18	59.57	55.75	45.65	66.67	57.14	67.86	56.92	56.67	50.00	62.39	57.53
Δ	-4.38	-5.32	-	-	-8.33	-8.93	+5.96	-12.31	-13.33	-25.00	+4.27	-2.74
w/o logical graph	68.42	61.70	51.33	41.30	66.67	51.79	59.52	55.38	43.33	59.38	63.25	65.75
Δ	-6.14	-3.19	-4.42	-4.34	-8.33	-14.28	-2.38	-13.85	-26.67	-15.62	+5.13	+5.48

split respectively. The syntax graph also shows 2.00% and 2.30% improvements on ReClor. As is mentioned above, we are the first to model the causal relations within the context in the task of logical reasoning. The effectiveness of logical graph also verifies our proposed method.

Secondly, we explore the impact of two attention biases on the model performance. Thus, we ablate the effects of one or both of attention bias matrices. From the results, co-occurrence bias and causal bias have different effects on the two splits of ReClor dataset, where the former one contributes more to the test split and the latter one is more helpful to the validation split. Meanwhile, positive effects are witnessed by applying both of the attention biases to the graph transformer, leading to 1.90% and 2.61% on the test split of ReClor and LogiQA respectively. Combining the ablation results for graph construction module, the fully connected structure of the logical units itself also has a positive role in the model performance.

Thirdly, we focus on the effectiveness of two important parts in the decoder. For the proposed dynamic gate mechanism, we set each element of the gate parameter vector $\lambda \in \mathbb{R}^{N \times 1}$ to 0.5 to ablate the effect of gates. The results show that dynamic gate mechanism contributes 1.6% improvement to the test split of ReClor, but does not have effects on that of LogiQA. It may result from the characteristics of LogiQA dataset, which require the equal contribution of syntax and logical information. For the question-aware attention, we remove the self-attention module and use the original token-level representation of the question to form the final vector. The ablation results illustrate that the update of the question feature contributes a lot to the model performance, especially for the ReClor dataset. Considering that the question types are various on

the ReClor dataset, the awareness of the question sentence is of great help.

Additionally, we present the detailed results of ReClor test split on different question types and also list the corresponding ablation results of two graphs in Figure 7. The majority of the types witness the significant improvements, especially for *Principle*, *Dispute* and *Role*. It illustrates that Logiformer has the advantages of inferring the hidden fact or truth within the context. A few types, such as *Explain or Resolve* and *Identify a Flaw*, show a downward trend. We blame this issue to the lack of modeling on negation. For example, the type of *Identify a Flaw* requires the model to figure out the most weakness one from the options, which is sentimentally opposite to the most of the types. The feature distribution obtained from the current language model is insufficient to clearly distinguish the implicit opposite semantics. Therefore, the modeling of sentimentally negative questions is worth exploring in the future work.

4.5 Supplementary Analysis

During the experiments, hyper-parameters are utilized in many places. Limited by the space, we only select one of them to conduct the analysis, which is the overlap threshold δ for the extraction of co-occurrence. The results of different values of δ are shown in Figure 4.

Results illustrate that the best performance is achieved when δ is equal to 0.5. When the hyper-parameter δ drops, it means that more co-occurrence pairs will be extracted, leading to much extra noise. When δ reaches 0.7, the number of co-occurrence relations is limited, the performance of the model still maintains at a high level. It further proves the robustness of Logiformer.

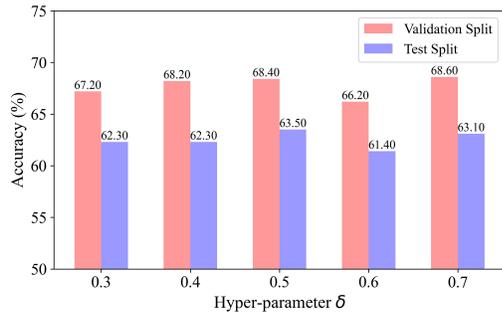


Figure 4: The model performances on the ReClor dataset under different δ .

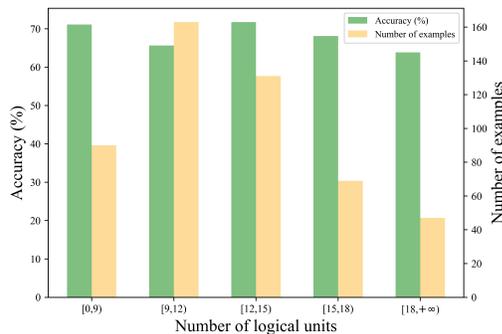


Figure 5: The model performances on under different numbers of logical units.

Also in Logiformer, logical units are important parts. Thus, we will study the influence of the number of logical units on the accuracy. We conduct the analysis on the validation split of ReClor dataset. The number of logical units are obtained based on the split of relation connectives and punctuations. The results are presented in Figure 5. The green bar represents the accuracy and the yellow bar is the number of examples. From the statistics, the examples with 9-12 logical units account for the largest proportion. Most importantly, the accuracy remains stable for the different number of logical units. On one hand, it proves the effectiveness of the split of logical units. On another hand, we attribute the results to the employment of the graph transformer. Fully connected structure tackles the issue of long distance dependency, which reduces the impact of the increase in the number of logical units.

5 CASE STUDY

In Figure 6, we present a successful case on the validation split of the ReClor dataset to illustrate the logical reasoning process in Logiformer. The basis of the reasoning process is the two constructed graphs from the logical branch and the syntax branch. In this case, Logiformer extracts 11 logical units (named from A to K) based on punctuations and relation connectives for the logical branch. The split results are consistent with our expectation, and among them 4 pairs of causal units ($C - D$, $E - F$, $H - I$, $J - K$) are detected. For the syntax branch, the concatenated text is split into 7 sentence nodes (named from a to g). Among them, 3 logical units (a, d, f) are detected as the co-occurrence relations. The topology of the

two graphs provides the explicit understanding of the text, which is a key point to the interpretability of Logiformer. In addition, we present the attention maps in the final layer of the graph transformers from both branches (blue one for logical branch, orange one for syntax branch). The data in the attention matrices is mapped to the range of $[0,1]$ for better illustration. Darker color indicates the stronger correlations between two logical units. The weighted attention maps well reflect the relations and provide a boarder view for interpretability.

Meanwhile, we observe a drawback in this case. *he gets married* and *his wedding* are two similar expressions in semantic, indicating the similar meanings. However, they have no overlap of words and are not detected as the co-occurrence relation. This detail is worthy of studying in the future, which is beneficial to the fine-grained understanding of the logical text.

6 CONCLUSION AND FUTURE WORK

We propose a two-branch graph transformer network for logical reasoning of text, which is named as Logiformer. Firstly, we introduce two different strategies to construct the logical graph and syntax graph respectively. Especially for the logical graph, we are the first to model both causal relations and negations in the logical reasoning task. Secondly, we feed the extracted node sequences to the fully connected graph transformer for each graph. The topology of the graph is utilized to form the attention bias for the self-attention layers. Thirdly, a dynamic gate mechanism is applied to make a fusion of the features from two branches. To improve the awareness of different question types, the question feature is updated based on the self-attention module. Finally, the concatenated text sequence is passed through the feed forward layer and obtains the answer prediction. The whole reasoning process provides the interpretability, reflected by logical units with explicit relations and the visualization of the attention maps.

In the future, we will explore the role of question to further improve the interpretability [29]. Also, we are interested in extending the logical expressions based on contrastive learning, like [16].

ACKNOWLEDGMENTS

This work was supported by National Key Research and Development Program of China (2020AAA0108800), National Natural Science Foundation of China (62137002, 61937001, 62176209, 62176207, 61877050, 62106190, 62192781 and 62050194), Innovative Research Group of the National Natural Science Foundation of China (61721002), Innovation Research Team of Ministry of Education (IRT_17R86), The National Social Science Fund of China(18XXW005), Consulting research project of Chinese academy of engineering “The Online and Offline Mixed Educational Service System for ‘The Belt and Road’ Training in MOOC China”, China Postdoctoral Science Foundation (2020M683493), Project of China Knowledge Centre for Engineering Science and Technology, “LENOVO-XJTU” Intelligent Industry Joint Laboratory Project, and the Fundamental Research Funds for the Central Universities (xzy022021048, xpt012021005, xhj032021013-02).

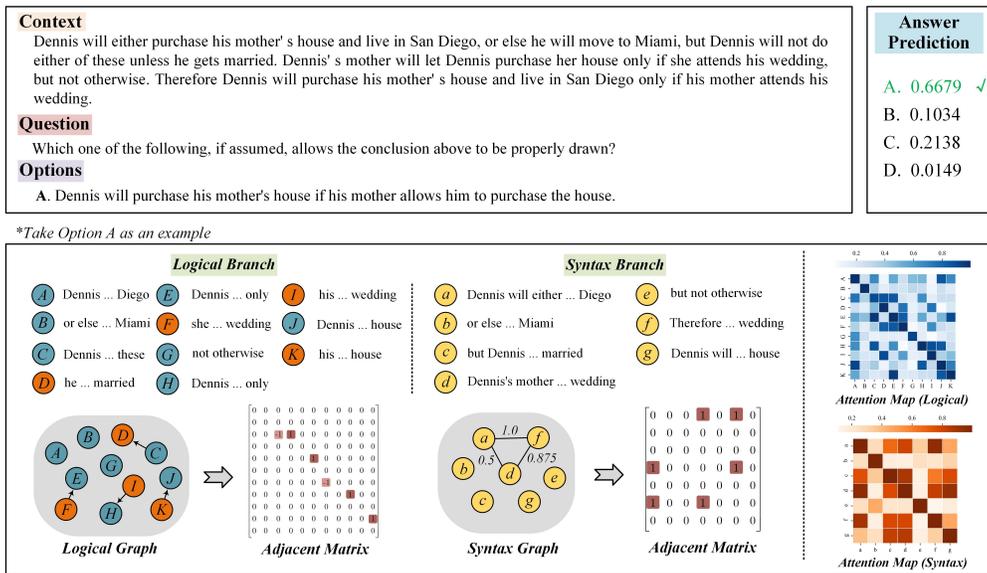


Figure 6: The illustration of an successful case. The interpretability of Logiformer lies in the logical units in text with explicit relations and the visualization of the weighted attention maps.

REFERENCES

- [1] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* (2015).
- [2] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).
- [3] Deng Cai and Wai Lam. 2020. Graph transformer for graph-to-sequence learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, 7464–7471.
- [4] Gobinda G Chowdhury. 2003. Natural language processing. *Annual review of information science and technology* 37, 1 (2003), 51–89.
- [5] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020).
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
- [7] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. 2016. Gated-attention readers for text comprehension. *arXiv preprint arXiv:1606.01549* (2016).
- [8] Vijay Prakash Dwivedi and Xavier Bresson. 2020. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699* (2020).
- [9] Julia Hirschberg and Christopher D Manning. 2015. Advances in natural language processing. *Science* 349, 6245 (2015), 261–266.
- [10] Lynette Hirschman and Robert Gaizauskas. 2001. Natural language question answering: the view from here. *natural language engineering* 7, 4 (2001), 275–300.
- [11] Charles F Hockett. 1947. Problems of morphemic analysis. *Language* 23, 4 (1947), 321–343.
- [12] Yinya Huang, Meng Fang, Yu Cao, Liwei Wang, and Xiaodan Liang. 2021. DAGN: Discourse-Aware Graph Network for Logical Reasoning. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 5848–5855.
- [13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- [14] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683* (2017).
- [15] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*.
- [16] Qika Lin, Jun Liu, Lingling Zhang, Yudai Pan, Xin Hu, Fangzhi Xu, and Hongwei Zeng. 2021. Contrastive Graph Representations for Logical Formulas Embedding. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2021).
- [17] Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124* (2020).
- [18] Shanshan Liu, Xin Zhang, Sheng Zhang, Hui Wang, and Weiming Zhang. 2019. Neural machine reading comprehension: Methods and trends. *Applied Sciences* 9, 18 (2019), 3698.
- [19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).
- [20] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789* (2018).
- [21] Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. 2021. Fact-driven Logical Reasoning. *arXiv preprint arXiv:2105.10334* (2021).
- [22] Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Arvind K Joshi, and Bonnie L Webber. 2008. The Penn Discourse TreeBank 2.0. In *LREC*. Citeseer.
- [23] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. *arXiv preprint arXiv:1806.03822* (2018).
- [24] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).
- [25] Guillaume Salha, Romain Hennequin, and Michalis Vazirgiannis. 2020. Simple and effective graph autoencoders with one-hop linear models. *arXiv preprint arXiv:2001.07614* (2020).
- [26] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks (TNN)* 20, 1 (2008), 61–80.
- [27] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603* (2016).
- [28] Shane Storks, Qiaozhi Gao, and Joyce Y Chai. 2019. Recent advances in natural language inference: A survey of benchmarks, resources, and approaches. *arXiv preprint arXiv:1904.01172* (2019).
- [29] Moganarangan Thayaparan, Marco Valentino, and André Freitas. 2020. A survey on explainability in machine reading comprehension. *arXiv preprint arXiv:2010.00389* (2020).
- [30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems (NIPS)*, 5998–6008.
- [31] Siyuan Wang, Wanjun Zhong, Duyu Tang, Zhongyu Wei, Zhihao Fan, Daxin Jiang, Ming Zhou, and Nan Duan. 2021. Logic-Driven Context Extension and Data Augmentation for Logical Reasoning of Text. *arXiv preprint arXiv:2105.03659*

- (2021).
- [32] Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Discourse-aware neural extractive text summarization. *arXiv preprint arXiv:1910.14142* (2019).
- [33] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems (NIPS)* 32 (2019).
- [34] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600* (2018).
- [35] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do Transformers Really Perform Bad for Graph Representation? *arXiv preprint arXiv:2106.05234* (2021).
- [36] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541* (2018).
- [37] Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2019. ReClor: A Reading Comprehension Dataset Requiring Logical Reasoning. In *International Conference on Learning Representations (ICLR)*.
- [38] Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. 2020. Graph-bert: Only attention is needed for learning graph representations. *arXiv preprint arXiv:2001.05140* (2020).
- [39] Xin Zhang, An Yang, Sujian Li, and Yizhong Wang. 2019. Machine reading comprehension: a literature review. *arXiv preprint arXiv:1907.01686* (2019).
- [40] Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020. SG-Net: Syntax-guided machine reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 34. 9636–9643.
- [41] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2021. Retrospective Reader for Machine Reading Comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 35. 14506–14514.