Question Directed Graph Attention Network for Numerical Reasoning over Text

Kunlong Chen† **Weidi Xu**† **Xingyi Cheng***†
Zou Xiaochuan† **Yuyu Zhang**§ **Le Song**†§ **Taifeng Wang**† **Yuan Qi**† **Wei Chu**†
† Ant Group
{kunlong.ckl,weidi.xwd,fanyin.cxy,xiaochuan.zxc,
le.song,taifeng.wang,weichu.cw,yuan.qi}@antgroup.com
§ College of Computing Georgia Institute of Technology
{yuyu}@gatech.edu

Abstract

Numerical reasoning over texts, such as addition, subtraction, sorting and counting, is a challenging machine reading comprehension task, since it requires both natural language understanding and arithmetic computation. To address this challenge, we propose a heterogeneous graph representation for the context of the passage and question needed for such reasoning, and design a question directed graph attention network to drive multi-step numerical reasoning over this context graph. Our model, which combines deep learning and graph reasoning, achieves remarkable results in benchmark datasets such as DROP [1].

1 Introduction

Machine reading comprehension (MRC) aims to develop AI models that can answer questions for text documents. Recently, the performance of MRC in public datasets has been improved dramatically due to the advanced pre-trained models, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and ALBERT (Lan et al., 2019).

However, pre-trained models are not explicitly aware of the concepts of numerical reasoning since numeracy supervision signals are rarely available during pre-training. The representations from these pre-trained models fall short in their ability to support downstream numerical reasoning. Yet such ability is critical for the comprehension of financial news and scientific articles, since basic numerical operations, such as addition, subtraction, sorting and counting, need to be conducted to extract the essential information (Dua et al., 2019).

Recently, Dua et al. (2019) proposed a numerically-aware QANet (NAQANet), which treats the span extractions, counting, and numerical addition/subtraction separately. However, this work is preliminary in the sense that the model neglects the relative magnitude between numbers. To improve this method, Ran et al. (2019) proposed NumNet, which constructs a number comparison graph that encodes the relative magnitude information between numbers on directed edges. Although NumNet achieves superior performance than other numerically-aware models (Hu et al., 2019a; Andor et al., 2019; Geva et al., 2020; Chen et al., 2020), we argue that NumNet is insufficient for sophisticated numerical reasoning, since it lacks two critical ingredients for numerical reasoning:

1. **Number Type and Entity Mention.** The number comparison graph in NumNet is not able to identify different number types, and lacks the information of entities mentioned in the document that connect the number nodes.
2. **Direct Interaction with Question.** The graph reasoning module in NumNet leaves out the direct question representation, which may encounter difficulties in locating important numbers directed by the question as the pivot for numerical reasoning.

The number type and entity information play essential roles in numerical comprehension and reasoning. As per the study in the cognitive system - "this abstract, notation-independent appreciation of numbers develops gradually over the first several years of life ... human infants appreciate numerical quantities at a non-symbolic level: They know approximately how many objects they see before them even though they do not understand number words or Arabic numerals.", the concept of discrete number is gradually developed through the real-life experience (Cantlon et al., 2009). The association among the numbers and entities is a strong regularization for learning the numerical reasoning model: the comparison and addition/subtraction between numbers are typically applied to those with the same type or referring to the same entity. To illustrate it, we show two concrete examples of numerical reasoning over texts in Table 1. In the first example, a question related to the "population" is being asked. There are 5 "*people counting*" numbers and 3 "*date*" numbers. When the type of number is given, the reasoning difficulty is largely reduced if the model learns to extract the "*people counting*" numbers conditioned on this "population" question. In addition, the entities in the graph provide explicit information on the correlation between the passage and

[*] Corresponding author

[1] https://leaderboard.allenai.org/drop/submissions/public. As of September 08, 2020, our models are ranked first in the case of fair comparison using the identical pre-training model.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6759–6768,
November 16–20, 2020. ©2020 Association for Computational Linguistics

Table 1: Two MRC cases requiring numerical reasoning are illustrated. There are entities and numbers of different types. Both are emphasized by different colors: entity, number, percentage, date, ordinal. We explicitly encode the type information into our model and leverage the question representation to conduct the reasoning process.

Question	Passage	Answer
At the battle of Caiboaté how many Spanish and Portuguese were injured or killed?	... In 1754 Spanish and Portuguese military forces were dispatched to force the Guarani to leave the area ... Hostilities resumed in 1756 when an army of 3,000 Spanish, Portuguese, and native auxiliary soldiers under José de Andonaegui and Freire de Andrade was sent to subdue the Guarani rebels. On February 7, 1756 the leader of the Guarani rebels, Sepé Tiaraju, was killed in a skirmish with Spanish and Portuguese troops. ... 1,511 Guarani were killed and 152 taken prisoner, while 4 Spanish and Portuguese were killed and about 30 were wounded...	34
In which quarter did Stephen Gostkowski kick his shortest field goal of the game?	The Cardinals' east coast struggles continued in the second quarter as quarterback Matt Cassel completed a 15-yard touchdown pass to running back Kevin Faulk and an 11-yard touchdown pass to wide receiver Wes Welker, followed by kicker Stephen Gostkowski's 38-yard field goal. In the third quarter, Arizona's deficit continued to climb as Cassel completed a 76-yard touchdown pass to wide receiver Randy Moss, followed by Gostkowski's 35- and 24-yard field goal. In the fourth quarter, New England concluded its domination with Gostkowski's 30-yard	third

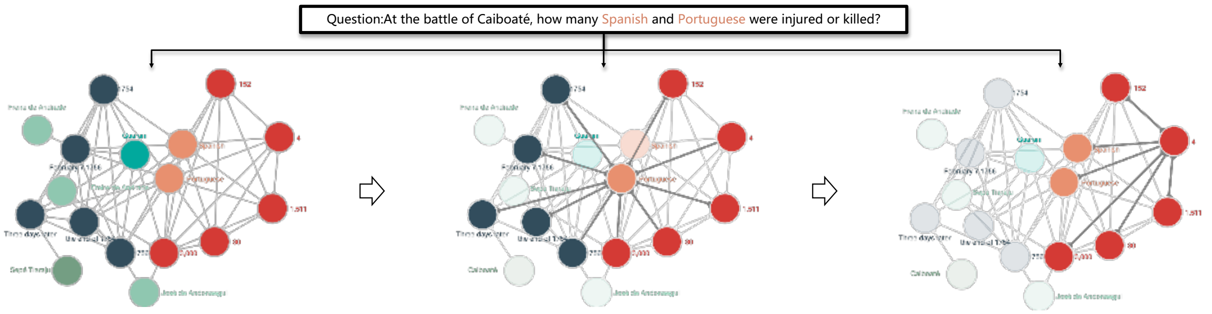

Figure 1: The constructed heterogeneous typed graph of the example in Table 1 is illustrated on the left. The red (dark blue) nodes are the numbers (dates) and the others are entities. The edges encode the relations among the numbers and entities: (1) The numbers with the same number type, e.g., date, are wired together. (2) The graph connects the numbers and the entities that are in the same sentence to indicate their co-occurrence. In the first round, the model pays attention to a sub-graph that contains the *Spanish* and *Portuguese* entities since they are mentioned in the question. In the update, the model learns to distinguish between the numbers and the dates and extracts the numbers related to the question. In the second round, the representations of the numbers are updated by the messages from the entities as well as the question to conduct the reasoning.

the question. The entities in the question may occur in several sentences in the passage, indicating how each number is related to each other through these bridging entities, which helps the QA model better collect and aggregate the information for numerical reasoning. We also observe that when the question entities co-occur in a single sentence (the last sentence in this example), this could be a hint that the answer can be derived from that sentence. The second example illustrates the case in span extraction. Similarly, the model is benefited when the correlations between the numbers and "Stephen Gostkowski" are explicitly provided.

To explicitly integrate the type and entity information into the model, we construct a heterogeneous directed graph where the nodes consist of entities and different types of numbers, and the edges can encode different types of relations. The corresponding graph of the example in Table 1 is illustrated in Figure 1. The graph nodes are composed of entities and numbers from both the question and the passage. The numbers of the same type are densely connected with each other. The co-occurred numbers and entities within a sentence are also connected with each other.

Based on this heterogeneous graph, we propose a question directed graph attention network (QDGAT) for the task of numerical MRC. As the answer-related numbers can be directed by the question, QDGAT incorporates the contextual encoding of the question in the graph reasoning process. More specifically, QDGAT employs a contextual encoder, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), to extract the representations of the numbers and entities in both the question and the passage, serving as the initial embeddings of each node in the graph. With the heterogeneous graph, QDGAT learns to collect information from the graph conditioned on the question for numerical reasoning. Each node is also described by a context-aware representation conditioned on the question, and the representations are updated through a message-passing iteration. After multiple iterations of message passing with graph neural networks, QDGAT gradually aggregates the node information to answer the question. In this sense, QDGAT abstracts the representation of passage and question in a way more consistent with human perception and reasoning, making the model produces a more interpretable reasoning pattern.

We evaluate QDGAT on two benchmark datasets: the DROP dataset (Dua et al., 2019) which requires Discrete Reasoning Over the content of Paragraph, and a subset of the RACE dataset (Lai et al., 2017) that contains the number-related questions. Experimental results indicate that QDGAT achieves remarkable performance on the DROP dataset, currently ranked as top 1 for all released models. And also rank first compared with other models that use the identical pre-training model.

2 Related Work

Machine Reading Comprehension. Benefit from recent improvements of pre-trained deep language models like BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), a considerable progress of MRC have been made on the annotated datasets such as SQuAD (Rajpurkar et al., 2016), RACE (Lai et al., 2017), TriviaQA (Joshi et al., 2017) and so on. To answer complex questions of MRC, a number of neural architectures have been proposed such as Attentive Reader (Hermann et al., 2015), BiDAF (Seo et al., 2017), Gated Attention Reader (Dhingra et al., 2017), R-NET (Wang et al., 2017), QANet (Yu et al., 2018), which achieved excellent results on existing datasets. Some recent works (LCGN (Hu et al., 2019b), NMNs (Gupta et al., 2020), NumNet (Ran et al., 2019)) attaching reasoning capabilities to models shows a promising direction. LCGN uses graph neural networks (GNN) conditioned on the input questions to support rational reasoning. NMNs parse the questions into one of several programs, each of which is responsible for specific reasoning ability.

Numerical Reasoning in MRC. Numerical reasoning has been studied when solving arithmetic word problems (AWP). However, existing AWP models only worked on small datasets, and the arithmetic expression must be clearly given. Numerical reasoning in MRC is more challenging since the numbers and reasoning rules are extracted from raw text, which requires a more sophisticated model. NAQANet improved the output layer of QANet to predict the answers from the arithmetic computation over numbers. In addition to NAQANet, GenBERT (Geva et al., 2020) injects numerical skills into BERT by generating numerical data. (Chen et al., 2020) provides a semantic parser that points to locations in the text that can be used in further numerical operations. BERT-Calculator (Andor et al., 2019) defines a set of executable programs and learns to choose one to derive numerical answers. NumNet (Ran et al., 2019) uses a numerically-aware graph neural network to encode numbers, which made further progress on the DROP dataset. However, the graph in NumNet contains only numbers and ignores their types and context information which play a key point in numerical reasoning. Our model differs from NumNet in two aspects: (1) We use a heterogeneous graph containing entities and different types of numbers to encode the relations among the entities and numbers, rather than the relations from numerical comparison; (2) We use the question embedding to modulate the attention over graph neighbors and update the representation to achieve reasoning.

3 Method

In this section, we first introduce the machine reading comprehension task requiring numerical reasoning. Then the framework of our model is provided, followed by detailed descriptions about its components.

3.1 Problem Definition

In the MRC task, each data sample consists of a passage P and a related question Q. The goal of an MRC model is to answer the question according to P. Besides predicting the text spans as in the standard MRC tasks, the answer A in the case of numerical reasoning can also be a number derived from arithmetic computations, such as sorting, counting, addition and subtraction.

3.2 Overall Framework

The framework of the proposed model is briefly depicted in Figure 2. The model is composed of three main components, i.e., a representation extractor module, a reasoning module, and a prediction module. The representation extractor is responsible for semantic comprehension. Upon the extractor, a heterogeneous graph with typed numbers and related entities is constructed. To aggregate the information between the numbers and entities, we propose a question directed graph attention network (QDGAT) to make sophisticated reasoning. This graph attention network directly employs the question Q to manage the message passing over the typed graph.

Word Representation Extractor. We employ RoBERTa (Liu et al., 2019) as the base architecture for the representation of textual inputs. The module takes the passage P and the question Q as input and outputs representation vectors for each token:

$$\hat{\mathbf{Q}}, \hat{\mathbf{P}} = \texttt{RoBERTa}(Q, P)\,, \tag{1}$$

where `RoBERTa` denotes the transformer encoder initialized with RoBERTa parameters, $\hat{\mathbf{P}}$ ($\hat{\mathbf{Q}}$) denotes the list of the token vectors of size d_h in the passage (question). It takes the concatenation of `[CLS]`, Q, `[SEP]`, P and `[SEP]` as input, and outputs representations of $\mathbf{Q}$ and $\mathbf{P}$ as $\hat{\mathbf{Q}}$ and $\hat{\mathbf{P}}$.

Graph Construction. This module builds the heterogeneously typed graph from text data. The graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ contains numbers $\mathbf{N}$ and entities $\mathbf{T}$ as the nodes $\mathbf{V} = \{\mathbf{N}, \mathbf{T}\}$, and its edges $\mathbf{E}$ encode the information of the number type and the relationship between the numbers and the entities. The details will be clarified in Section 3.3.

Numerical Reasoning Module. The numerical reasoning module, i.e., QDGAT, is built upon the representation and graph extractor. Based on the graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, the QDGAT network can be formulated as

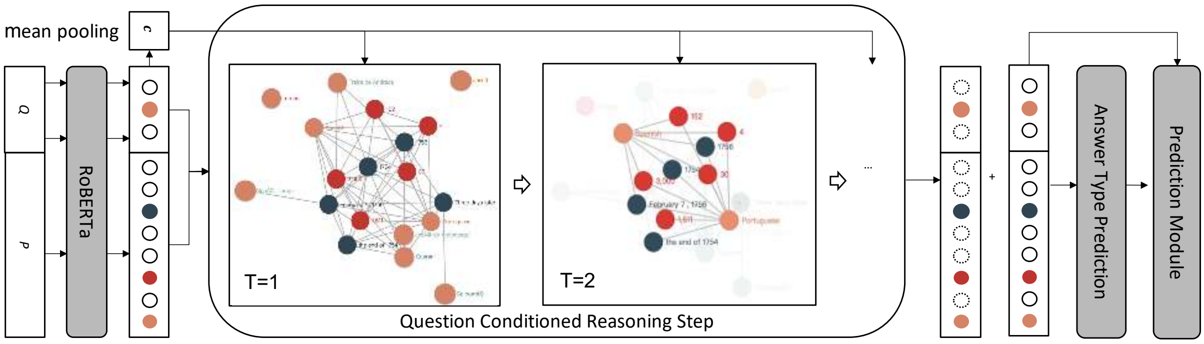

Figure 2: The framework of our model. It consists of a representation extractor (left), a reasoning module (middle) and a prediction module (right). The reasoning module reasons over a heterogeneous directed graph whose nodes are the numbers and the entities. Two kinds of relations are encoded: (1) the numbers of the same type are connected with each other by the type-specific edges, (2) the entities and the numbers are connected when they co-occur in a sentence. The reasoning is conditioned on the question explicitly to guide the message propagation over the graph. In each iteration, each node selectively receives the messages from the neighboring nodes with the question representation to update its representation. The derived representations of these nodes are then combined with the RoBERTa output for the final prediction module. The dashed circle means zero vector.

follows:

$$\mathbf{M}^Q = \mathbf{W}^M \hat{\mathbf{Q}}\,, \tag{2}$$

$$\mathbf{M}^P = \mathbf{W}^M \hat{\mathbf{P}}\,, \tag{3}$$

$$\mathbf{c} = \mathbf{W}^c \texttt{MEAN}(\hat{\mathbf{Q}})\,, \tag{4}$$

$$\mathbf{U} = \texttt{QDGAT}(\mathcal{G}; \mathbf{M}^P, \mathbf{M}^Q, \mathbf{c})\,, \tag{5}$$

where $\mathbf{W}^M \in \mathbb{R}^{d_h \times d_h}$ is a shared projection matrix to obtain the input of QDGAT, `MEAN` denotes the mean pooling, $\mathbf{W}^c \in \mathbb{R}^{d_h \times d_h}$ projects the averaged vector of the representations in the question to derive $\mathbf{c}$. $\mathbf{c}$ is the question language embedding used to direct the reasoning in `QDGAT`. QDGAT then reasons over the representations ($\mathbf{M}^P$, $\mathbf{M}^Q$) and the graph $\mathcal{G}$ conditioned on the question command $\mathbf{c}$.

Prediction Module The prediction module takes the output of graph reasoning network $\mathbf{U}$ for final prediction. At present, the types of answers are generally divided into three categories in NAQANet and NumNet+: (a) span extraction, (b) count, (c) arithmetic expression. We implemented separate modules for these answer types and all of them take the output of graph network $\mathbf{U}$ and question embedding $\mathbf{c}$ as input. They are specified as follows:

- Span extraction: There are three span extraction tasks, i.e., single passage span, multiple passage spans, single question span. The probability for single span extraction is derived by the product of the probabilities of the start and end positions in either question or passage. For multiple spans extraction, the probability is constructed referring to (Efrat et al., 2019).
- Count: This problem is regarded as a 10-class classification problem (0-9), which covers about 97% counting problems in the DROP dataset.
- Arithmetic expression: The answer is derived by an arithmetic computation. In the DROP dataset, only addition and subtraction operations are involved. We achieved this by classifying each number into one of $(-1, 0, +1)$, which is then used as the coefficient of the number in the numerical expression to arrive at the final answer.

We used a unique classification network to classify the data sample into one of five fine-grained types (T). And each type solver employs a unique output layer to calculate the conditional answer probability $p(A|T)$.

3.3 Graph Construction with Typed Number and Entities

Here, we illustrate how to construct the heterogeneous graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ in our model. NumNet solely concerns the numerical comparisons between numbers by using the directed edges. The graph used in our model differs from NumNet significantly: Rather than modeling the numerical comparison, our graph instead exploits two sources of information, i.e., the type of numbers and the related entities. As illustrated in Figure 2, the nodes of graph $\mathbf{V}$ consists of both entities $\mathbf{T}$ and numbers $\mathbf{N}$, both of which are recognized by an external name entity recognition (NER) system [2].

Specifically, the NER software labels each token in the text into one of 21 pre-defined categories. The tokens labeled as `NUMBER`, `PERCENT`, `MONEY`, `TIME`, `DATE`, `DURATION`, `ORDINAL` are regarded as the numbers. Since DROP dataset contains a lot of samples related to American football games, we also used heuristic rules to extract the numbers of `YARD` type in the data samples. Besides, we leveraged a number extractor, i.e., word2num [3], to extract the remaining

[2] We used Standford CoreNLP toolkit (Manning et al., 2014).

[3] https://pypi.org/project/word2number/

numbers, which are labeled as NUMBER. All these tokens construct the number set $\mathbf{N}$ with 8 number types ($\mathcal{V}_N = (\texttt{NUMBER}, \texttt{PERCENT}, \texttt{MONEY}, \texttt{TIME}, \texttt{DATE}, \texttt{DURATION}, \texttt{ORDINAL}, \texttt{YARD})$). As for other recognized tokens, we map them into the label ENTITY to build the entity set $\mathbf{T}$ whose type set $\mathcal{V}_T$ is ENTITY. In the following, we use $t(v) \in \mathcal{V}_N \cup \mathcal{V}_T$ to indicate the type of the node. The type information can directly inform the model to find the numbers related to the question and thus reduces the reasoning difficulty.

The edges $\mathbf{E}$ encode the relationship among the numbers and the entities, which correspond to two situations.

- *The edge between the numbers*: An edge $e_{i,j}$ exists between two numbers v_i and v_j if and only if these two numbers are of the same type in $\mathcal{V}_N$. And its relation $r_{i,j} = r_{j,i}$ corresponds to the number type.
- *The edge between the entity and the number*: An edge $e_{i,j}$ exists between an entity v_i and a number v_j if and only if v_i and v_j co-occur in the same sentence. In this situation, the relation $r_{i,j} = r_{j,i}$ is ENT+DIGIT.

The edges in the first situation cluster the same typed numbers together, which provides an evident clue to help to reason over the numbers. In the second situation, we assume that an entity is relevant to a number when they appear closely. This kind of edges roughly indicates the correlations between the numbers and the entities in most cases. On the other hand, the relative magnitude relations in Numnet+ are not considered in our graph since early experiments with these relations did not improve results. Overall, the graph has 9 relations $\mathcal{R}$, i.e., 8 relations for number types and 1 relation for ENT+DIGIT.

3.4 Question Directed Graph Attention Network

Here, we present the details of the QDGAT function. Based on the heterogeneous graph $\mathcal{G}$, our QDGAT makes context-aware numerical reasoning conditioned on the question, which collects the relational information through multiple iterations of message passing between the numbers and the entities. It dynamically determines which objects to interact with through the edges in the graph, and sends messages through the graph to propagate the relational information. To achieve this, we augment the reasoning module with the contextualized question representation. For instance in the example in Table 1, the task is to find how many *Spanish* and *Portuguese* were injured or killed. The entities and the numbers are explicitly marked and are modeled in a heterogeneous graph, as shown in Figure 1. Our model is able to extract the related entities, i.e., the *Spanish* and *Portuguese*, conditioned on $\mathbf{c}$. Among the numbers related to these two entities, a number of them are of date type, while the others are about people. However, only the numbers related to people should be concerned as requested by the question. Then the model reasons over these numbers to derive the expression for the answer calculation.

Module Input. The graph neural network takes the representations from the extractor as the input. Each node is represented by the corresponding vector in $\mathbf{M}^P$ and $\mathbf{M}^Q$. Formally, when v_i is in the passage, the input of node v_i is the $\mathbf{v}_i = \mathbf{M}^P[\mathbf{I}^P(v_i)]$, where $\mathbf{I}^P$ returns the index of v_i in $\mathbf{M}^P$ [4]. The collected vectors from the question and the passage construct the input of reasoning module $\mathbf{v}^0$.

Question Directed Node Embedding Update. At each iteration $t \in \{1, ...T\}$, a question directed layer integrates the question information with the current node embedding representations. This step is to mimic the reasoning step of detecting relevant nodes. More specifically, the question, represented by $\mathbf{c}$, is used to direct the information propagation between the nodes (i.e., the numbers and the entities). Each node collects the information from the neighbors with the question command. The role of numbers and entities is not only dependent on the input itself, but also the neighbors and the relations between them. Therefore, we adopt the self-attention layer (Vaswani et al., 2017) to dynamically aggregate the information. The representation is first converted into three spaces denoting the query, key and value, conditioned on $\mathbf{c}$:

$$\mathbf{m}^t = \mathbf{W}^t_{dc} g(\mathbf{W}_{fc}\mathbf{c}), \tag{6}$$

$$\mathbf{x}^t_q = \mathbf{W}_{qv}[\mathbf{v}^t : \mathbf{v}^0] \odot \mathbf{W}_{qc}\mathbf{m}^t, \tag{7}$$

$$\mathbf{x}^t_k = \mathbf{W}_{kv}[\mathbf{v}^t : \mathbf{v}^0] \odot \mathbf{W}_{kc}\mathbf{m}^t, \tag{8}$$

$$\mathbf{x}^t_v = \mathbf{W}_{vv}[\mathbf{v}^t : \mathbf{v}^0] \odot \mathbf{W}_{vc}\mathbf{m}^t, \tag{9}$$

where $\mathbf{m}^t$ denotes the command vector extracted dynamically from the $\mathbf{c}$ with $\mathbf{W}^t_{dc}$ and $\mathbf{W}_{fc} \in \mathrm{R}^{d_h \times 2d_h}$, g denotes the ELU activation function (Clevert et al., 2016), $\mathbf{W}_{qv}$, $\mathbf{W}_{kv}$ and $\mathbf{W}_{vv}$ are of size $d_h \times 2d_h$, $\mathbf{W}_{qc}$, $\mathbf{W}_{kc}$ and $\mathbf{W}_{vc}$ are of size $d_h \times d_h$, $[a : b]$ means the concatenation of a and b, and $\odot$ means the element-wise multiplication. These equations include the input $\mathbf{v}^0$ to maintain the original information.

Directed Graph Attention. At each iteration, this graph attention layer for each node aggregates information from the neighbors of the node. This step is to mimic the reasoning step of selecting the relevant relations to operate on. More specifically, we compute the relatedness between the node i and j, which is measured by summarizing all relations:

$$a^t_{i,j} = f(\sum_{r \in \mathcal{R}_{i,j}} \mathbf{W}^r_a[\mathbf{x}^t_{q,i} : \mathbf{x}^t_{k,j}]), \tag{10}$$

where $\mathcal{R}_{i,j}$ means the relations between the two nodes, $a_{i,j}$ denotes the attention score of the node i for the node j, $\mathbf{W}^k_a$ is the vector to map the representations into a scalar for the relation r and f denotes the leakyReLU activation function (Xu et al., 2015).

This attention score is used in the message propagation to collect the right amount of information from each neighboring node. In the propagation function, the

[4] When v_i corresponds to several tokens, the average of these vectors is used.

calculation of the node interaction is as follows:

$$\alpha_{i,j}^t = \frac{\exp(a_{i,j}^t)}{\sum_{j' \in \mathcal{N}_i} \exp(a_{i,j'}^t)} \,, \tag{11}$$

$$\hat{\mathbf{x}}_i^t = \sum_{j \in \mathcal{N}_i} \alpha_{i,j} \mathbf{x}_{v,j} \,, \tag{12}$$

$$\mathbf{v}_i^{t+1} = \mathbf{W}_u [\mathbf{v}_i^t ; \hat{\mathbf{x}}_i^t] \,, \tag{13}$$

where $\mathcal{N}_i$ contains the adjacent nodes of the node i in the $\mathcal{G}$ and $\mathbf{W}_u$ is in $\mathbb{R}^{d_h \times 2d_h}$. With the weight $\alpha_{i,j}$ obtained, the values of neighboring nodes are summarized to derive a new representation $\hat{\mathbf{x}}$. Finally, the new representation of $\mathbf{v}$ is computed by mapping the concatenation of $\mathbf{v}^0$ and $\hat{\mathbf{x}}$.

We denote the node embedding update and the graph attention layers as a function:

$$\mathbf{v}^{t+1} = \texttt{QDGAT-single}(\mathcal{G}, \mathbf{v}^t, \mathbf{c}) \,. \tag{14}$$

From the process of this reasoning step, we can see that the module receives the information from the question, which directly manages the message propagation among the numbers and the entities.

Module Output We perform T iterations of the reasoning step of `QDGAT-single` to perform `QDGAT` in Equation 5. The output of the last layer $\mathbf{v}^T$ is obtained for the numbers and entities in $\mathbf{U}$. For other tokens, the representation vectors from the extractor are used. Formally, the calculation of the output $\mathbf{U}$ is implemented as follows:

$$\mathbf{U}_i = \begin{cases} \mathbf{M}_i + \mathbf{v}_{\mathbf{J}(i)}^T, & \text{if } i\text{-th token} \in \mathbf{V} \\ \mathbf{M}_i, & \text{otherwise} \end{cases} \tag{15}$$

where $\mathbf{J}(i)$ denotes the index of token i in the graph nodes, $\mathbf{M}$ denotes the combination of $\mathbf{M}^P$ and $\mathbf{M}^Q$ for simplicity. $\mathbf{U}$ is then used in the prediction module for the five answer types mentioned above.

4 Experiments

4.1 Dataset and Evaluation Metrics

We performed experiments on the DROP dataset (Dua et al., 2019), which was recently released for research on numerical machine reading comprehension (MRC). DROP is constructed by crowd-sourcing question-answer pairs on passages from Wikipedia, which contains 77,409 / 9,536 / 9,622 samples in the original training / development / testing split. Following the previous work (Dua et al., 2019), we used Exact Match (EM) and F1 score as the evaluation metrics.

4.2 Baselines

We choose publicly available methods (including non-published ones on the dataset leaderboard) as our baselines:

- **Semantic parsing models:** Syn Dep, OpenIE and SRL (Dua et al., 2019). All these models are enhanced versions of KDG (Krishnamurthy et al., 2017) with different sentence representations.
- **Traditional MRC models:** (1) BiDAF, a model that uses a bi-directional attention flow network to obtain a query-aware context representation; (2) QANet, a model that combines convolution and self-attention models to answer the questions; (3) BERT (Devlin et al., 2019), a pre-trained deep Transformer (Vaswani et al., 2017) model that has improved results on many NLP tasks.
- **MRC models with numerical reasoning module:** (1) NAQANet (Dua et al., 2019), a model that adapts the output layer of QANet to numeric reasoning; (2) ALBERT-Calculator (Andor et al., 2019), a model based on ALBERT-xxlarge (Lan et al., 2020) that picks one of executable programs from a predefined set to derive numerical answers. (3) NumNet, a model that embeds numerical properties into the distributed representation by using a GNN on the number graph; (4) NumNet+ [5], an enhanced version of NumNet, which uses a pre-trained RoBERTa model and supports multi-span answers.

4.3 Experiment Settings

We use the large RoBERTa model as the contextual encoder, with 24 layers, 16 attention heads, and 1024 embedding dimensions. This indicates that the hidden size d_h is 1024. The model was trained end-to-end for 5 epochs using Adam optimizer (Kingma and Ba, 2015) with a batch size of 16. For the hyperparameters of RoBERTa, the learning rate is 5e-5 and the L2 weight decay is 1e-6. For the other parts, the learning rate is 1e-4 and the L2 weight decay is 5e-5. We perform $T = 4$ iterations of the graph reasoning step, which performs best in our experiments. We adopt the standard data preprocessing following previous work (Ran et al., 2019).

4.4 Main Results

The overall experimental results are reported in Table 2, where the performance of baseline methods is obtained from previous work (Dua et al., 2019; Seo et al., 2017; Ran et al., 2019; Andor et al., 2019) and the public leaderboard.[6]

The first three methods in Table 2 are based on either semantic parsing or information extraction, and perform poorly on the numerical MRC task. Traditional MRC methods BiDAF and QANet, which has no numerical reasoning modules, achieve slightly better performance but still far from satisfying. Methods that are customized for numerical reasoning, including NAQANet and NumNet, have achieved significantly better performance in terms of EM and F1 score. Compared to traditional MRC methods, these methods can handle different answer types, e.g., span extraction, counting, and addition/subtraction of numbers.

Our method QDGAT outperforms all the existing methods, achieving 86.38 F1 score and 83.23 EM on

[5] https://github.com/llamazing/numnet_plus

[6] https://leaderboard.allenai.org/drop/submissions/public

Table 2: Overall results on the development and test set of DROP. For QDGAT$_p$, we used more careful data preprocessing and a RoBERTa pre-trained on the SQuaD dataset. † denotes that the result is taken from the public leaderboard. Better results are in bold.

Method	Dev		Test	
	EM	F1	EM	F1
Syn Dep	9.38	11.64	8.51	10.84
OpenIE	8.80	11.31	8.53	10.77
SRL	9.28	11.72	8.98	11.45
BiDAF	26.06	28.85	24.75	27.49
QANet	27.50	30.44	25.50	28.36
BERT	30.10	33.36	29.45	32.70
NAQANet	46.20	49.24	44.07	47.01
ALBERT-Calculator	80.22	83.98	79.85	83.56
NumNet	64.92	68.31	64.56	67.97
NumNet+ (RoBERTa)	81.07†	84.42†	81.52†	84.84†
NumNet+ (ensemble)	82.63†	85.59†	83.14†	86.16†
QDGAT (RoBERTa)	**82.74**	**85.85**	**83.23**	**86.38**
QDGAT$_p$ (RoBERTa)	**84.07**	**87.05**	**84.53**	**87.57**
QDGAT$_p$ (ensemble)	**85.31**	**88.10**	**85.46**	**88.38**
Human			94.09	96.42

the test set, which narrows the human performance gap to less than 11 points. NumNet+ is the most relevant one to our method, which also leverages a graph neural network as well as the RoBERTa contextual encoder. Compared to NumNet+, QDGAT incorporates the number types and entity mentions into the graph attention network, and directs the graph reasoning process with the question. In this way, our method can better capture the relations between numbers and entities, and also reduce the learning difficulty due to the interaction with the question during the graph reasoning. Experimental results demonstrate the effectiveness of QDGAT, which outperforms NumNet+ by 1.23 in terms of EM and 1.37 in terms of F1 score. Ensembling three of our models with different random seeds and learning rates further improves the performance.

4.5 Ablation Analysis

To examine the impact of different components of QDGAT, we conduct ablation studies and compare the performance in Table 3. QDGAT$_{\mathrm{NH}}$ removes the number type and entity from the graph, and QDGAT$_{\mathrm{NQ}}$ removes question direction from QDGAT and instead uses a normal graph convolution message passing mechanism. NumNet+ serves as a baseline for reference, since it has no question attention, no entities and no number types in the graph. We observe that QDGAT$_{\mathrm{NQ}}$, which has no question directed attention, performs worse. This justifies that the reasoning with graph neural network is more effective when conditioned on the input question. We also observe that QDGAT$_{\mathrm{NH}}$ performs significantly worse, which demonstrates the importance of incorporating the information of number types and entity mentions in the reasoning graph. This is consistent with our intuition that numbers with the same type or connected to the same entity are more relevant to each other.

Table 3: Ablation study results on the development set of DROP. QDGAT$_{\mathrm{NH}}$ removes the number type and entity from the graph, and QDGAT$_{\mathrm{NQ}}$ removes question direction from QDGAT. Better results are in bold.

Method	EM	F1
NumNet+	81.07	84.42
QDGAT$_{\mathrm{NH}}$	81.98	84.94
QDGAT$_{\mathrm{NQ}}$	82.04	85.01
QDGAT	**82.74**	**85.85**

Table 4: Decomposed performance on different answer types in the development set of DROP. Better results are in bold.

Method	Number		Date		Span	
	EM	F1	EM	F1	EM	F1
NumNet+	82.89	83.13	56.67	63.91	82.00	86.84
QDGAT	**86.00**	**86.23**	**60.27**	**67.48**	**84.05**	**88.53**

Table 4 decomposes the QA performance on different answer types in the development set of DROP. As reported in the table, QDGAT works better on the questions relating to numbers and dates, which requires more specific numerical reasoning compared with the span extraction. The remarkable improvement indicates that the proposed method effectively benefits the reasoning module to comprehend the numerical problems. Notably, the performance in span extraction can still be improved by our method. The span extraction in DROP heavily relies on the ability to comprehend the relation between the number and the entity (c.f. the second example in Table 1).

4.6 Performance on RACENum

To investigate the generalization capability of QDGAT in numerical reasoning, we examine whether the pre-trained model on DROP is transferable. We compare QDGAT with NumNet+ on RACE (Lai et al., 2017), a dataset collected from the English exams for middle and high school Chinese students. We extracted a special part of examples from RACE, where the questions start with "how many", referred to as RACENum. RACENum is then divided into middle school exam (RACENum-M) and high school exam (RACENum-H) categories. The RACENum-M and RACENum-H datasets contain 633 and 611 questions accordingly. Since the original RACE dataset is in the multiple-choice form, we converted them into the DROP data format. The accuracy of NumNet+, QDGAT and its ablation variants on RACENum are summarized in Table 6, which is consistent with the performance comparison on the DROP dataset.

The overall low scores are attributed to the lack of training on the in-domain data. QDGAT achieves

Table 5: The cases from the DROP dataset. The predictions from the QDGAT and NumNet+ are illustrated. The differences between the output of these two models demonstrate the properties of the proposed model. The last two columns indicate the arithmetic expression, obtained by assigning a sign (plus, minus or zero) for each extracted numbers (we omitted the zero sign numbers). Then the answer was derived by summing up the signed numbers.

Question & Answer	Passage	NumNet+	QDGAT
Q: How many less in age percentage in teenagers than adult? **A:** 1.3	The age distribution, in Aigle is; 933 children or 10.7% of the population are between 0 and 9 years old and 1,137 teenagers or 13.0% are between 10 and 19. Of the adult population, 1,255 people or 14.3% of the population are between 20 and 29 years old...	19-13.0-10=-4	14.3-13.0 =1.3
Q: How many yards did Kasay kick? **A:** 94	... Carolina scored first in the second quarter with kicker John Kasay hitting a 45-yard field goal . The Falcons took the lead with QB Joey Harrington completing a 69-yard TD pass to WR Roddy White . The Panthers followed up with QB Jake Delhomme completing a 13-yard TD pass to RB DeShaun Foster ... In the fourth quarter , the Panthers scored again , with Kasay kicking a 49-yard field goal . The Falcons ' Andersen nailed a 25-yard field goal to end the scoring ...	+45=45	45+49 =94
Q: How many months after Mengistu Haile Mariam was made head of state did Ethiopia close the U.S. military mission and the communications centre? **A:** 2	... A sign that order had been restored among the Derg was the announcement of Mengistu Haile Mariam as head of state on 02/1977. However, the country remained in chaos as the military attempted to suppress its civilian opponents in a period known as the Red Terror ... Ethiopia closed the U.S. military mission and the communications centre in 04/1977. In 06/1977, Mengistu accused Somalia of infiltrating SNA soldiers into the Somali area to fight alongside the WSLF. Despite considerable evidence to the contrary...	Count: 3	+4-2=2

Table 6: The accuracy on the unsupervised RACENum dataset.

Method	RACE-M	RACE-H	Avg.
NumNet+	46.98	31.59	39.29
QDGAT$_{\mathrm{NH}}$	50.88	35.30	43.09
QDGAT$_{\mathrm{NQ}}$	49.67	**35.84**	42.76
QDGAT	**52.53**	34.86	**43.70**

43.7 points on RACENum on average, which is approximately 4.5 points higher than NumNet+. Both QDGAT$_{\mathrm{NQ}}$ and QDGAT$_{\mathrm{NH}}$ still outperform NumNet+ by a 2–3 points margin. We further confirmed that ablating either the entity information or question attention from the heterogeneous graph weakens the power of QDGAT to learn numeracy and the capability of understanding numbers in either digits or word form. Compared with QDGAT, ablating the question directed attention, i.e., QDGAT$_{\mathrm{NQ}}$, leads to about a 1 point drop. For QDGAT$_{\mathrm{NH}}$ that removes the number type and entity mentions from the graph, it performs consistently worse than QDGAT, demonstrating the impact of the heterogeneous graph for numerical reasoning.

4.7 Case Study

We show several examples to provide insights into how our model works. Table 5 compares the different model prediction results from NumNet+ and QDGAT:

- The first example shows the importance of number types. NumNet+ treats all numbers as the same type, which fails to capture that the question only cares about percentage and incorrectly predicts "*19*" (type age) as part of the result. In contrast, QDGAT extracts the relevant numbers and derives the correct answer.
- The second example highlights the importance of entity mentions. NumNet+ fails to extract "*49-yard*", but QDGAT easily captures this number since "*49-yard*" and "*45-yard*" are connected to the same entity "*Kassy*" on the heterogeneous graph which is generated from the passage.
- The third example shows the importance of question conditioning. Solving this example requires to extract the two dates related to two events mentioned in the question. Without direct interaction between the question, the model tends to recognize this example as a counting problem since the question starts with "how many". However, when combined with question directed attention, correct numbers can be filtered out.

5 Conclusion

In this work, we propose a novel method named QDGAT for numerical reasoning in the machine reading comprehension task. Our method not only builds a more compact graph containing different types of numbers, entities, and relations, which can be a general method for other sophisticated reasoning tasks but also conditions the reasoning directly on the question language embedding, which modulates the attention over graph neighbors and change messages being passed iteratively to achieve reasoning. The experimental results verify the effectiveness of our method. In the future, we plan to extend our model to learn the heterogeneous graph automatically, which assures more flexibility for numerical reasoning. We would also explore to learn the types of numbers and entities together the reasoning modules using variational autoencoder techniques (Kingma and Welling, 2014), which may help the NER system better adapt to the numerical reasoning task.

References

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving BERT a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5946–5951. Association for Computational Linguistics.

Jessica F Cantlon, Melissa E Libertus, Philippe Pinel, Stanislas Dehaene, Elizabeth M Brannon, and Kevin A Pelphrey. 2009. The neural development of an abstract concept of number. *Journal of cognitive neuroscience*, 21(11):2217–2229.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. 2020. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Minnesota.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2017. Gated-attention readers for text comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1832–1846.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Avia Efrat, Elad Segal, and Mor Shoham. 2019. Tag-based multi-span extraction in reading comprehension. *CoRR*, abs/1909.13375.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *ACL*.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *International Conference on Learning Representations*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019a. A multi-type multi-span network for reading comprehension that requires discrete reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1596–1606, Hong Kong, China. Association for Computational Linguistics.

Ronghang Hu, Anna Rohrbach, Trevor Darrell, and Kate Saenko. 2019b. Language-conditioned graph networks for relational reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10294–10303.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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. *CoRR*, abs/1907.11692.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, System Demonstrations*, pages 55–60.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. Numnet: Machine reading comprehension with numerical reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2474–2484.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hananneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. 2015. Empirical evaluation of rectified activations in convolutional network. *CoRR*, abs/1505.00853.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution. In *International Conference on Learning Representations*.