Teaching Neural Module Networks to Do Arithmetic

Jiayi Chen , Xiao-Yu Guo , Yuan-Fang Li , and Gholamreza Haffari

Faculty of Information Technology, Monash University, Melbourne, Australia
jche0069@student.monash.edu
{xiaoyu.guo,yuanfang.li,gholamreza.haffari}@monash.edu

Abstract

Answering complex questions that require multi-step multi-type reasoning over raw text is challenging, especially when conducting numerical reasoning. Neural Module Networks (NMNs), follow the *programmer-interpreter* framework and design trainable modules to learn different reasoning skills. However, NMNs only have limited reasoning abilities, and lack numerical reasoning capability. We upgrade NMNs by: (a) bridging the gap between its *interpreter* and the complex questions; (b) introducing addition and subtraction modules that perform numerical reasoning over numbers. On a subset of DROP, experimental results show that our proposed methods enhance NMNs' numerical reasoning skills by 17.7% improvement of F1 score and significantly outperform previous state-of-the-art models.

1 Introduction

Complex Question Answering (CQA) over text is a challenging task in Natural Language Understanding (NLU). Based on the *programmer-interpreter* paradigm, Neural Module Networks (NMNs) (Gupta et al., 2020) learn to first parse complex questions as executable programs composed of various predefined trainable modules, and then execute such programs (implemented by modules) over the given paragraph to predict answers of all kinds. NMNs achieve competitive reasoning performance on a subset of DROP (Dua et al., 2019), and possess remarkable interpretability that is also important for CQA.

However, NMNs' numerical reasoning capability is insufficient: it is incapable of handling arithmetic operations such as addition and subtraction between numbers, which make up nearly 40% questions of the DROP dataset. Moreover, a gap exists between the interpreter and the complex question since there is no interaction between them. Motivated by these, we propose two methods to improve NMNs' numerical reasoning skills.

First, we incorporate the original question in the interpreter, aiming to **directly provide question information in the "execution" process**, especially number-related questions. The intuition behind is that, in the original NMNs, questions participate in the process only through the programmer. This can cause a distance between queries and returns. For example, in Figure 1, the first row shows that the original NMNs found the wrong event (i.e., 'besieged Sinj') solely based on the paragraph information. In contrast, our model NMNs$_{\pm}$ can easily target the correct event (i.e., 'Sinj finally fell') with the help of question information.

Second, we introduce new modules to **support addition and subtraction of up to three numbers**. Endowing NMNs with the ability to support arithmetic can greatly boost its overall performance on DROP and beyond. For instance, in Figure 1, the second row shows that the original NMNs improperly adopt the `find-num` module for the addition question because the module set does not cover such an arithmetic ability. To facilitate the learning of the `add/sub` modules, we extract QA pairs related to addition and subtraction from the original DROP dataset to construct a new dataset for training and evaluation.

Experimental results show that our methods significantly enhance NMNs' numerical reasoning capability. On a subset of DROP, our methods improve F1 score by 17.7% absolute points, and on ADD-SUB questions by 65.7% absolute points. Compared to NumNet (Ran et al., 2019), which is specifically designed for numerical reasoning, our method outperforms it by 2.9% absolute F1 points.

2 Background and Related Work

Semantic Parsing is a widely-adopted approach in the compositional question answering (CQA) task, which involves a number of reasoning steps. In this

Proceedings of the 29th International Conference on Computational Linguistics, pages 1502–1510
October 12–17, 2022.

Question type	Paragraph	Question	Answers
date compare	In the Morean War, the Republic of Venice besieged Sinj in October 1684 and then again March and April 1685, but both times without success. … With the help of the local population of Poljica as well as the Morlachs, the fortress of Sinj finally fell to the Venetian army on 30 September 1686. On 1 September 1687 the siege of Herceg Novi started, and ended with a Venetian victory on 30 September. …	Which happened first, the fell of Sinj or the siege of Herceg Novi?	Original NMNs: besieged Sinj $NMN_{\pm}$: Sinj finally fell Ground-truth: the fell of Sinj / Sinj finally fell
	Original NMNs' program: (span(compare-date-lt(find find)))	$NMN_{\pm}$: (span(compare-date-lt(find find)))	
addition / subtraction (2 numbers)	… In the first quarter, the Niners struck first as kicker Joe Nedney got a 47-yard field goal. In the second quarter, the Saints took the lead with QB Drew Brees completing a 5-yard and a 33-yard TD pass to WR Lance Moore. San Francisco would answer with Nedney's 49-yard field goal, yet New Orleans replied with Brees' 47-yard TD pass to WR Robert Meachem. …	How many yards was Nedney's combined field goal yards in the first and second quarters?	Original NMNs: 47 $NMN_{\pm}$: 96 Ground-truth: 96
	Original NMNs' program: (find-num (filter (find))	$NMN_{\pm}$: (addition(find-num(find))(find-num(find)))	
addition / subtraction (3 numbers)	The Greek census 2011 recorded 9,903,268 Greek citizens (91.56%), 480,824 Albanian citizens (4.44%), 75,915 Bulgarian citizens (0.7%), 46,523 Romanian citizenship (0.43%), 34,177 Pakistani citizens (0.32%), 27,400 Georgia (country) citizens (0.25%) and 247,090 people had other or unidentified citizenship (2.3%). …	How many more people were Greek citizens compared to Albanian and Bulgarian citizens combined?	Original NMNs: 9903268 $NMN_{\pm}$: 9346529 Ground-truth: 9346529
	Original NMNs' program: (find-num (find-max-num (find)))	$NMN_{\pm}$: (subtraction (find-num(find)) (addition(find-num(find))(find-num(find))))	

Figure 1: Three examples in the DROP dataset and the predictions by original NMNs and our improved model NMNs$_{\pm}$. The relevant tokens and their corresponding modules are highlighted.

approach, a *programmer* maps natural-language questions into machine-readable representations (logical forms), which are executed by an *interpreter* to yield the final answer. For instance, WNSMN (Saha et al., 2021) uses a generalized framework of dependency parsing inspired by the Stanford dependency parse tree (Chen and Manning, 2014) to parse queries into noisy heuristic programs. Neural Module Networks (Gupta et al., 2020) extend semantic parsing by making interpreter a learnable function with specified modules and executing the logical forms from the programmer in a step-wise manner.

Neural Module Networks initially is proposed to overcome the Visual Question Answering (VQA) problem (Andreas et al., 2016), where questions are often compositional. Gupta et al. (2020) employs the programmer-interpreter framework with attention (Vaswani et al., 2017) to tackle the CQA task. Specifically, the programmer parses each question into an executable program. The interpreter takes the program as input and perform various symbolic reasoning functions. The modules are defined in a differentiable way, aiming to maintain the uncertainty about each intermediate decision output and propagate them through layers. For instance, the predicted program of the first example in Figure 1 is `span(compare-date-lt(find, find))`. The interpreter would first calls the `find` module twice to find events queried by the question (e.g., 'the fell of Sinj') and outputs appropriate paragraph attention. The `compare-date-lt` module can further locate the dates (e.g., '30 September 1686') to compute their relation. By demonstrating the intermediate reasoning steps in this manner, NMNs perform interpretable problem-solving.

Numerical Reasoning is a necessary ability for models to handle the CQA task (Geva et al., 2020). Dua et al. (2019) modify the output layer of QANet (Yu et al., 2018) and propose a number-aware model NAQANet to deal with numerical questions. NumNet (Ran et al., 2019) leverage Graph Neural Network to capture relations between numbers. Similarly, QDGAT (Chen et al., 2020a) distinguish number types more precisely by adding the connection with entities and obtained better performance. Nerd (Chen et al., 2020b) search possible programs exhaustively based on answers and employed these programs as weak supervision. Another similar work (Guo et al., 2021) proposes a question-aware interpreter but uses a totally different approach to measure the alignment between the question and the context paragraph. Though these approaches can achieve the high performance on DROP dataset, it is incomprehensible for the reasoning procedure.

3 Model

In this section, we tend to illustrate our proposed methods. Basically, we will show the incorporation of questions in Section 3.1. In Section 3.2, the newly extended module: addition and subtraction will be described.

3.1 The Incorporation of Questions

Taking one module `compare-date` as a case study: it performs comparisons between two references queried by the question. A key reasoning step inside, is the `find-date` module that obtains appropriate a date token distribution D related to each reference: `find-date`$(P) \rightarrow D$. It is worth noting that there is no interaction with the question, which could contain essential information (e.g., entities) that is useful to correctly answer the question. Therefore, we revise the `find-date` module as follows: `find-date`$(P,Q) \rightarrow D$:

$$\mathbf{S}^{date}_{i,d_j} = [\alpha\mathbf{P};(1-\alpha)\mathbf{Q}]_i \mathbf{W}_{date}\mathbf{P}_{d_j}, \quad (1)$$

$$\mathbf{A}^{date}_{i:} = softmax(\mathbf{S}^{date}_{i:}), \quad (2)$$

$$D = \textstyle\sum_i [\alpha P;(1-\alpha)Q]_i \cdot \mathbf{A}^{date}_{i:} \quad (3)$$

where **P** and **Q** represent the contextualized embeddings of the paragraph and question, and $\mathbf{P}_{d_j}$ of the j^{th} date tokens in the paragraph, $\mathbf{W}_{date}$ is a trainable parameter, P,Q are the expected attention distribution of the paragraph and the question respectively.

In Equation 1, we concatenate the paragraph embeddings **P** and question embeddings **Q** that output from a pre-trained BERT (Devlin et al., 2019) model to construct the context representation. A hyper-parameter α is used to adjust their contributions, whose value is empirically determined (Appendix A.1). The context representation is provided to compute the improved similarity matrix $\mathbf{S}_{date}$. We concatenate the paragraph and question attention inputs in the same way to calculate the final expected distribution over the date tokens D (Eq. 3). Now the interpreter is equipped with question information to make the prediction.

3.2 Addition and Subtraction Modules

In the NMNs' modelling paradigm, for addition/subtraction operations, the programmer takes as input two number distributions and produces an output number distribution over all possible result values: `add/sub`$(N_1, N_2) \rightarrow RL$. N_1 and N_2 represent the probability distributions of the 1^{st} and 2^{nd} operands over all numbers that are extracted from the paragraph and collected into a sorted operand list OL. The positive and negative values of these numbers are exhaustively combined in pairs, from which the possible results of addition/subtraction operations are compiled into a sorted result list RL. For each input number distribution $N_i, i = 1, 2$, a matrix $\mathbf{C}_i \in \mathbb{R}^{m \times n}$ is constructed, where m is the total number of possible results, and n is the maximum number of unique combinations. Each value $\mathbf{C}_i[j,k]$ is found by looking up the probability value in $N_i[k]$ where $OL[k]$ is the i^{th} operand in any pair that produces result $RL[j]$. The probability that the j^{th} number in N_i is the correct operand of the k^{th} pair. We compute the marginalized joint probability by summing over the product of C_i as the expected distribution over result list RL. For the addition module, it is:

$$p(prediction = RL[j]) = \sum_{k_1,k_2=1}^{n} \mathbb{1}_{(OL[k_1]+OL[k_2]=RL[j])} \mathbf{C}_1[j,k_1] * \mathbf{C}_2[j,k_2]$$

For instance, assume the sorted operand list OL from a paragraph is [1, 5, 7, 11] and $N_1 = [0.1, 0.4, 0.2, 0.3]$. Different combinations are formed, e.g., (+n_1, +n_2) for addition and (+n_1, -n_2) for subtraction, and all possible results of the combinations are compiled into two result lists, one for addition and one for subtraction. For subtraction in this case, $RL = [0, 2, 4, 6, 10]$. The value of $\mathbf{C}_1$[2,1] is 0.4, which is found from N_1[1] because the result 4 can be calculated from (+5, -1); and $\mathbf{C}_1[2,3] = 0.3$ which equals to N_1[3] as 4 is the result of (+11, -7) as well. $\mathbf{C}_2$ is computed in the same way to further obtain final distribution over RL.

We compose `add/sub` modules in programs to perform 3-number arithmetic. The key to our approach is to construct and distinguish appropriate C_i and RL in different reasoning steps. In the second arithmetic step, we should combine the operand list from the paragraph and the result list from the previous step to obtain a new result list RL', `add/sub`$(RL, N) \rightarrow RL'$. Due to the changes in operands and results, the modules should refer to a different $\mathbf{C}'_i \in \mathbb{R}^{m' \times n'}$ in the computation. We extend 2-number `add/sub` modules to recognize the participation of the third number by conditional statement, in order to differentiate the operand and result lists the interpreter should refer to in different steps. Taking the last example in Figure 1, the `addition` module would first compute the distribution over result list for 'Albanian and Bulgarian citizens'. The `subtraction` module can identify itself in the second step calculation and take the correct input to construct the new matrix $\mathbf{C}'_i$. The expected distribution over new result list RL' now represent the difference of 'Greek citizens' and the previous result.

Instead of introducing specific modules for multi-number arithmetic such as '3-num-add', the structure of NMNs allows us to recursively execute basic operations several times in a compositional program. This design is in accord with the reasoning process of the CQA task, and natural for NMNs to perform complex computations.

4 Experiments

Dataset. We construct our own train/dev/test sets based on the DROP dataset (Dua et al., 2019), which requires numerical reasoning skills.

Gupta et al. (2020) extracted a subset of questions from DROP that is supported by the model's reasoning capability. This subset contains approximately 20,000/500/2,000 QA pairs for train/dev/test. To train the `add/sub` modules, we augment the NMNs' subset with more than 5,000 new questions from DROP. These questions were heuristically identified based on first n-grams and regular expressions (Appendix A.2). Statistics of this newly constructed dataset can be found in Table 1. Note that the ADD-SUB questions include both 2-/and 3-number arithmetic and **all experiments in this paper are conducted on this new dataset**. Model performance is evaluated with the same F1 and EM (Exact Match) scores as Gupta et al. (2020).

Question types	train	dev	test
Full	25,165	623	2,547
DATE-COMPARE (13.9%)	3,505	91	333
DATE-DIFFERENCE (12.2%)	3,055	75	313
NUMBER-COMPARE (12.1%)	2,642	157	632
EXTRACT-NUMBER (12.8%)	3,349	57	222
COUNT (17.3%)	4,527	73	288
EXTRACT-ARGUMENT (13.1%)	3,467	51	208
ADD-SUB (18.6%)	**4,689**	**124**	**553**
2-numbers	4,440	106	505
3-numbers	259	24	66

Table 1: Question types distribution on the expanded DROP subset used in the follow experiments.

Result. In Table 2, we list the overall performance of the original NMNs, NumNet and our proposed method NMNs$_{\pm}$.

Method	F1	EM
original NMNs (Gupta et al., 2020)	57.5	54.9
NumNet (Ran et al., 2019)	72.3	69.4
NMNs$_{\pm}$ (ours)	**75.2**	**72.6**
w/o add/sub	61.4	58.1
w/o qi	74.3	71.7

Table 2: Performance comparison between different models on **our test set**. Constrained by the page limit, case study and analysis are in Appendix A.5.

In Table 2, row "w/o add-sub" is the model variant with question attention only, and row "w/o qi" only has the `add/sub` modules only. Compared to the original NMNs, two proposed methods both improve model performance and the `add/sub` modules contributes more. Our full NMNs$_{\pm}$ model, with both components added, achieves 75.2% F1 and 72.6% EM scores, obtaining significant deltas of 17.7% absolute points compared to the original NMNs for both F1 and EM. Additionally, NMNs$_{\pm}$ outperforms NumNet by 2.9% and 3.2% absoule points in F1 and EM.

It can be unfair since the original NMNs will perform poorly on the newly added ADD-SUB questions. Therefore, we list the model performance on different question types in Table 3. Our model achieves higher scores across almost all question types comparing to the original NMNs, attesting to the effectiveness of our proposed techniques. And it turns out that adding ADD-SUB question types and more training data does not improve the results of the original DROP split. This might due to the performance degradation of the programmer after adding these new ADD-SUB programs. When comparing to NumNet, though our model fail on 2-number ADD-SUB questions, we achieve 5.4% F1 improvement on 3-number ADD-SUB questions, thus results in a comparable performance. Note that the 2-number data is nearly **18 times** the 3-number data, which shows our model or NMNs relies less on large scale datasets.

Question type	NMNs	NMNs$_{\pm}$	NumNet
DATE-COMPARE	79.2	**84.9**	72.0
DATE-DIFFERENCE	69.0	73.3	**74.1**
NUMBER-COMPARE	89.6	**90.3**	89.9
EXTRACT-NUMBER	86.4	**89.1**	85.6
COUNT	54.2	**60.2**	52.4
EXTRACT-ARGUMENT	73.4	**75.3**	66.1
ADD-SUB	0.7	66.4	**67.6**
2-numbers	0.8	67.9	**71.5**
3-numbers	0.3	**41.2**	35.8

Table 3: F1 comparison on different question types.

Additional ablation studies for the `add/sub` modules (A.3) and a qualitative analysis (A.4) can be found in the appendix.

5 Conclusion

In this work, we extend NMNs' numerical reasoning capability to 2-/and 3-number addition and subtraction, and incorporate the influence of question information to the interpreter on number related questions. Experimental results show that our methods significantly enhance NMNs' numerical reasoning ability, with an increase of 17.7% absolute F1 points on a newly constructed DROP subset that includes arithmetic questions. Moreover, our approach also outperforms NumNet, a SOTA numerical reasoning model, by 2.9% F1 points.

Acknowledgements

This work is partially funded by the DARPA CCU program (HR001121S0024).

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48. IEEE Computer Society.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP*, pages 740–750.

Kunlong Chen, Weidi Xu, Xingyi Cheng, Zou Xiaochuan, Yuyu Zhang, Le Song, Taifeng Wang, Yuan Qi, and Wei Chu. 2020a. Question directed graph attention network for numerical reasoning over text. In *Proceedings of EMNLP*, pages 6759–6768.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. 2020b. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *Proceedings of ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

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 NAACL-HLT*, pages 2368–2378.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 946–958. Association for Computational Linguistics.

Xiaoyu Guo, Yuan-Fang Li, and Gholamreza Haffari. 2021. Improving numerical reasoning skills in the modular approach for complex question answering on text. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2713–2718. Association for Computational Linguistics.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *Proceedings of ICLR*.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. Numnet: Machine reading comprehension with numerical reasoning. In *Proceedings of EMNLP-IJCNLP 2019*, pages 2474–2484.

Amrita Saha, Shafiq R. Joty, and Steven C. H. Hoi. 2021. Weakly supervised neuro-symbolic module networks for numerical reasoning. *CoRR*, abs/2101.11802.

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 *Proccedings of NIPS*, pages 5998–6008.

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. In *Proceedings of ICLR*.

A Appendix

A.1 Hyper-parameter setting for compare-date modules

As mentioned above, we use a hyper-parameter α to represent question's and paragraph's weights for the combined context representation. We determine the final coefficient through a series of control parameter comparison experiments: use the same data to train and validate the model with different α. The model achieves the best performance (84.9 F1) for DATE-COMPARE questions when α was set to 0.4 (40% for paragraph attention and 60% for question attention), which increase 5.7 absolute points compared to the original NMNs model. The experiment verifies the importance of question information in the numerical reasoning process.

A.2 Data extraction

In this research, we expand the DROP subset for original NMNs to cover addition and subtraction questions. Subtraction questions can be easily targeted by their first n-gram, such as 'how many more', 'how many yards difference'. For three number subtraction, we need to further specified the format by regular expression, such as 'how many more EVENT-A and EVENT-B than EVENT-C?' or 'how many more EVENT-A compared to EVENT-B and EVENT-C?'. For addition, it is hard to identify how many numbers should participate in the calculation from some of the questions (e.g. 'how many total yards did Roethlisberger get in the game?'). Therefore, we use regular expression to distinguish two or three numbers addition and follow the patterns such as 'how many total...', 'how many ... combined'.

A.3 Addition and subtraction modules training

To discuss the contribution of individual `addition` and `subtraction` module for NMNs, we conduct an ablation experiment by training and testing the model on different datasets as shown in Table 4. The five rows represent the model trained on various datasets: addition questions only, subtraction questions only, addition and the original NMNs subset, subtraction and original NMNs subset and our full subset. The columns indicate the model performance results when they test on addition/subtraction questions only and the full DROP subset. As can be seen from the result, the model with subtraction ability only perform greater than with addition ability only.

Datasets	addsub dataset		full dataset	
	F1	EM	F1	EM
add	41.2	41.2	46.0	43.8
sub	45.7	45.7	51.3	49.2
add+origin	51.5	51.5	69.2	64.2
sub+origin	55.1	55.1	72.6	69.8
add+sub+origin	66.4	66.4	74.3	71.7

Table 4: Ablation experiment result for addition and subtraction modules

A.4 Qualitative analysis

Figure 2 shows some incorrect prediction cases from the original NMNs and the answer from our improved model NMNs$_\pm$. From the examples, we can clearly identify how the proposed techniques improve the numerical reasoning process:

- In the first example, the original NMNs match wrong tokens 'dissolved the Constituent Assembly' given the question 'Which event happened first, the Constituent Assembly being elected, or the elimination of hierarchy in the army?', thus located a wrong date 'January 1918'. After enhancing the interpreter's awareness of the question, NMNs$_\pm$ can precisely target the spans 'a Constituent Assembly was elected' in the paragraph and further provide the correct prediction.
- The following two examples are wrongly answered by the original NMNs because of incorrect program predictions. The second question was initially categorized into a COUNT question which called the `count` module to calculate the number of attended paragraph spans. The same situation occurs in the third question, because the original NMNs lack the modules that can correctly expresses the reasoning behind the question. The prediction results prove that our NMNs$_\pm$ model handle simple arithmetic operations such as addition and subtraction which meets the task requirement.

A.5 Prediction analysis

The wrong prediction cases study for the original NMNs over DROP is the main motivation of our proposed methods. We conclude the error factors of five numerical question types in detail: DATE-COMPARE, COUNT, DATE-DIFFERENCE, NUMBER-COMPARE and EXTRACT-NUMBER.

Question type	Paragraph	Question	Answers
Date compare	On 12 November 1917, a Constituent Assembly was elected. In these elections, 26 mandatory delegates were proposed by the Bolshevik Central Committee and 58 were proposed by the Socialist Revolutionaries. … The Bolsheviks dissolved the Constituent Assembly in January 1918, when it came into conflict with the Soviets. On 16 December 1917, the government ventured to eliminate hierarchy in the army, removing all titles, ranks, and uniform decorations.	Which event happened first, the Constituent Assembly being elected, or the elimination of hierarchy in the army?	Original NMNs: eliminate hierarchy in the army $NMN_{\pm}$: the Constituent Assembly being elected Ground-truth: Constituent Assembly being elected
	Original NMNs' program: (span(compare-date-gt(find find))) $NMN_{\pm}$: (span(compare-date-gt(find find)))		
Count	… The Ravens would later add 16 more points on three Billy Cundiff field goals and a fumble forced by Ray Lewis and recovered by Haloti Ngata and then run into the end zone (Cundiff also missed two 51-yard field goals). …	How many field goals did Billy Cundiff kick (both successful and unsuccessful)?	Original NMNs: 2 $NMN_{\pm}$: 5 Ground-truth: 5
	Original NMNs' program: (count (filter (find)) $NMN_{\pm}$: (addition(find-num(find))(find-num(find)))		
Find number	… In the first quarter, the Niners struck first as kicker Joe Nedney got a 47-yard field goal. In the second quarter, the Saints took the lead with QB Drew Brees completing a 5-yard and a 33-yard TD pass to WR Lance Moore. San Francisco would answer with Nedney's 49-yard field goal, yet New Orleans replied with Brees' 47-yard TD pass to WR Robert Meachem. …	How many yards was Nedney's combined field goal yards in the first and second quarters?	Original NMNs: 47 $NMN_{\pm}$: 96 Ground-truth: 96
	Original NMNs' program: (find-num (filter (find)) $NMN_{\pm}$: (addition(find-num(find))(find-num(find)))		

Figure 2: Qualitative analysis. The highlighted spans are corresponding to the modules in the program for each question.

Deficiency	**Paragraph**	**Question**	**Interpretation**
Match incorrect start and end of the span as answer (53.4%)	… In **1438** Svitrigaila withdrew to Moldavia. The reign of Sigismund Kestutaitis was brief — he was assassinated in 1440. Svitrigaila returned from exile in **1442** and ruled Lutsk until his death a decade later. …	What event happened later, Svitrigaila withdrew to Moldavia or Svitrigaila returned from exile?	the final output is not exactly the same with the ground-truth, but F1 score is not 0
Match wrong dates for the entities (20.8%)	The Siege of Vienna in **1529** was the first attempt by the Ottoman Empire, ... Thereafter, 150 years of bitter military tension and reciprocal attacks ensued, culminating in the Battle of Vienna of **1683**, which marked the start of the 15-year-long Great Turkish War. The inability of the Ottomans to capture Vienna in 1529 turned the tide against almost a century of conquest throughout eastern and central Europe. …	Which happened first, the Siege of Vienna or the Great Turkish War?	reasons include dates are too close, or match to the date for a similar entity
Date comparison based on natural language inference or phrases instead of symbolic comparison (13.4%)	The Russians advance into the Polish-Lithuanian Commonwealth **led to** the kingdom of Sweden invading Poland in 1655 under King Charles X. …	What happened first, the Russian advance into Poland-Lithuania or the Swedish invasion of Poland?	the module cannot understand the relation expressed by words or phrases, such as 'lead to', 'before'.
Wrong comparison result due to incomplete date format (12.4%)	… An agricultural worker had been shot during a local strike on **9 August 1917** at Ypaja and a Civil Guard member was killed in a local political crisis at Malmi on **24 September**. …	Which happened first, the shooting of a worker during a strike, or the killing of a Civil Guard member in Malmi?	the dates with omitted year are represented in format (DD, MM, -1), which lead to wrong comparison result. We proposed a forward matching mechanism in the data pre-processing step to tackle this problem.

Figure 3: Root causes for the wrong prediction in DATE-COMPARE questions. The related events mentioned in the question are highlighted in blue and red, and their relevant dates are in the same color with underline.

Deficiency	Paragraph	Question	Interpretation
Error caused by other modules (25%) • Find module cannot match the correct spans in the paragraph • Filter module cannot identify the target in the passage	… In the third quarter, Cincinnati continued to struggle as Patriots RB Sammy Morris getting a 7-yard TD run. The Bengals' only response would be kicker Shayne Graham nailing a 40-yard field goal. In the fourth quarter, New England increased its lead with Gostkowski kicking a 36-yard field goal. Cincinnati's final response was Graham kicking a 48-yard field goal. …	How many field goals were there in the second half?	the filter module is uncapable to identify 'the second half' means the third and fourth quarters, then it wrongly target on 'the second quarter'
Require numerical operation such as addition, subtraction or conditional decision (17.5%)	… In the first quarter, the Redskins drew first blood when the kicker Shaun Suisham nailed a 49-yard field goal for the only score of the quarter. … The Giants would get on the board with kicker Lawrence Tynes getting a 35-yard field goal. …	How many more field goals did Shaun Suisham make compared to Lawrence Tynes?	the subtraction operation is required in reasoning process, instead of direct counting
Wrong program prediction (16.5%)	… The Cowboys would only kick field goals in this game, as Dan Bailey was 4 for 4 on field goals. Dallas lead 12-10 with under 2 minutes to go. …	How many field goals did the Cowboys make?	predicted program: (count(filter(find()))); correct program: (find-num(find()))
Mistakes in counting the key tokens in the question (15%) • unable to count 0 when the tokens not exist in the paragraph • should not predict the answer based on counting tokens	… In the first quarter, Denver trailed early as QB Josh McCown completed a 15-yard TD pass to WR Tim Dwight. The Broncos replied with RB Travis Henry getting a 4-yard TD run. …	How many field goals did Janikowski kick in the first quarter?	'field goal' or 'Janikowski' is not mentioned in the filtered paragraph, but the module didn't predict 0 as answer
Unable to count multiple times in one span or identify phrases (15%)	… In the first quarter, the Bengals opened the scoring with two Shayne Graham field goals. ... In windy conditions, Phil Dawson hit a pair of 29-yard field goals, and Chris Jennings had a 10-yard touchdown run to put the Browns up 13. …	How many field goals were kicked in the game?	The count module only count once for 'two field goals' or 'a pair of field goals'
Cannot identify true or false in counting (7.5%)	… Houston had a chance to tie the game with one second left in regulation, but Brown's 42-yard field goal attempt **sailed wide left.** …	How many field goals were scored during the game?	The count module should not count once for 'the field goal sailed wide left'
Other various error due to wrong mean value calculation (3.5%)			

Figure 4: Root causes for the wrong prediction in COUNT questions. The inputs to the find module and their targets in the paragraph are highlighted in red. The blue spans are related to the filter module.

Deficiency	Paragraph	Question	Interpretation
Unable to identify the tokens as date	... It was waged from 1593 to 1606 but in Europe it is sometimes called the Fifteen Years War, reckoning from the 1591-92 Turkish campaign that captured Bihac. …	How many years did the Turkish campaign that captured Bihac last?	the date parser cannot interpret 1591-92 into two dates
Computation error, should be the date difference plus one	After twelve years of peace following the Indian Wars of 1622-1632, another Anglo-Powhatan War began on March 18, 1644, as a last effort by the remnants of the Powhatan Confederacy, still under Opechancanough, to dislodge the English settlers of the Virginia Colony. …	How many years did the Indian Wars last?	the prediction is calculated by the year difference between 1622 and 1632, but the answer is 11 years
Cannot understand the relation between dates through semantic expression	… His wife died in 1583, and on 7 November 1590 he was married in the same church to Jaél de Peigne, a French Hugenot. She was naturalised in June 1601. After Henry's death she remarried on 19 April 1617 George Downham, Bishop of Derry, and died c.1632. …	How many years was it after her first marriage did Jael de Peigne marry for the second time?	the time-diff module cannot identify the second one or the last one of the related dates
Wrong program prediction	… The Yongle Emperor began the preparation for relocating the imperial capital to Beiping in 1403, a process that lasted throughout his entire reign. … In 1420, the reconstruction of Beiping City was completed, and the Ming Dynasty officially relocated the imperial capital to Beiping and renamed the city to Beijing. …	How many years did the relocation of the capital to Beiping take?	predicted program: (count (find)) correct program: (time-diff (find, find))
Match wrong dates for the entities	… The remains of about 70 men, women, and adolescents were found in the path of a planned expressway near Lima in 2007. Forensic evidence suggests that the natives were killed by European weapons, probably during the uprising in 1536. …	How many years after the uprising in 1536 where many natives were killed did Archaeologists find their remains near Lima?	the find date module did not target on the correct date 1536 for the first event

Figure 5: Root causes for the wrong prediction in date-difference questions. The related events are highlighted in blue, which is the input of the find module. The dates grounding correctly predicted in the compare-date modules are highlighted in red color. The answer predicted by NMNs should be the difference of these two dates.

Deficiency	Paragraph	Question	Interpretation
Did not match the correct number for entities when there were multiple existence	… On 21 June 1916, two troops of the 10th, totaling 92 troopers, attacked Mexican Federal Army troops in an engagement in the Battle of Carrizal, Chihuahua. **12** US troops were killed and 23 taken prisoner; **45** Federales were casualties, including the Mexican general Gomez. …	Which group experienced more casualties, US troops or Federales?	'US troops' was mentioned multiple times in the paragraph, the find-num module wrongly targets on another number
Need multiple answers to the question	… In the city, the age distribution of the population shows 21.8% under the age of 18, 13.1% from 18 to 24, **31.7%** from 25 to 44, 20.1% from 45 to 64, and **13.2%** who were 65 years of age or older. The median age was 34 years. For every 100 females, there were 87.1 males. For every 100 females age 18 and over, there were 83.5 males. …	Which age groups had a bigger population than those 65 years of age or older but lower than those 25 to 44?	there are two age groups conform to the condition: '45 to 64', 'under the age of 18'
Interpret the comparison adjective in the question wrongly	… In terms of ancestry, 28.1% were German, **19.8%** were Irish, 12.2% were English, 9.9% were Italian, **6.8%** were Polish, and 6.2% were American.	Which group had the least ancestry, Irish or Polish?	The compare-num module cannot identify the meaning of 'the least'
Wrong program prediction	… The racial makeup of the county was 81.2% white, **12.7%** black or African American, **2.4%** Asian, 0.3% American Indian, 0.1% Pacific islander, 0.9% from other races, and 2.5% from two or more races. Those of Hispanic or Latino origin made up 3.5% of the population. …	Which group made up more of the population than the Asians but less than black or African American?	the predicted program is (span (compare-num-gt (find, find))), it did not consider the less than condition in the question
Not matching the correct start and end of a span to answer the question			

Figure 6: Root causes for the wrong prediction in number-compare questions. Similar to figure 1, the input of the find module is highlighted in blue and red, and their related numbers are underlined. The paragraph span predicted as the answer is the one associated to a smaller/larger-valued number according to the questions asking.

Deficiency	Paragraph	Question	Interpretation
Find module return an inaccurate result for find-num module	… In the second half the Ravens scored 4 consecutive touchdowns. First a Le'Ron McClain 3-yard run. Then **2** by Willis McGahee: first an 8-yard run, then a 19-yard run. In the fourth quarter, the Ravens capped off their huge victory when Troy Smith ran in a TD from 15 yards …	How many yards was the shortest touchdown run in the game?	there is no key words around the ground-truth answer 2, so the module predict '3-yard run' as answer
Filter module fix to the wrong spans (e.g. based on natural language inference)	… In the first quarter, Tennessee drew first blood as QB Vince Young completed a 16-yard TD pass to WR Roydell Williams for the only score of the period. In the second quarter, the Chiefs tied the game with QB Brodie Croyle completing a 10-yard TD pass to WR Samie Parker. Afterwards, the Titans responded with kicker Rob Bironas managing to get a 37-yard field goal. Kansas City would take the lead prior to halftime with Croyle completing a 9-yard TD pass to FB Kris Wilson …	How many yards was the shortest touchdown of the first half?	the filter module is uncapable to identify 'the first half' means the first and second quarters, then it wrongly target on 'the first quarter'
Require arithmetic operation	… In the first quarter, the Niners struck first as kicker Joe Nedney got a **47-yard field goal**. In the second quarter, the Saints took the lead with QB Drew Brees completing a 5-yard and a 33-yard TD pass to WR Lance Moore. San Francisco would answer with Nedney's **49-yard field goal**, yet New Orleans replied with Brees' 47-yard TD pass to WR Robert Meachem …	How many yards was Nedney's combined field goal yards in the first half?	predicted program: (find-num (filter (find))) correct program: (addition(find-num(filter (find))) (find-num(filter (find))))

Figure 7: Root causes for the wrong prediction in extract-number questions. The inputs to the find module and their targets in the paragraph are highlighted in red. The blue spans are related to the filter module. The find-num module finally extracts the number associated with this paragraph attention as the answer.