Enhancing Arguments Recognition for Financial Mathematical Reasoning over Hybrid Data

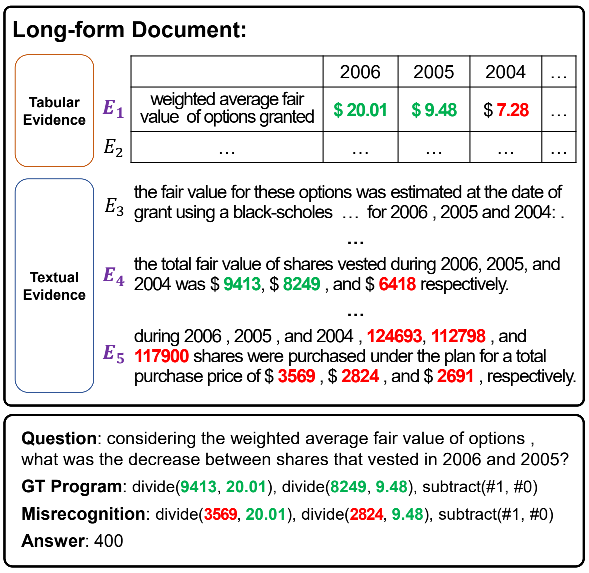
Jinsu Lim, Yechan Hwang, Young-Jun Lee, Ho-Jin Choi
School of Computing, KAIST
{j1n2u, yemintmint, yj2961, hojinc}@kaist.ac.kr

Abstract

Mathematical question answering over long-form documents is challenging across domains like finance or Wikipedia due to the abundance of candidate arguments within evidence, which complicates recognizing proper arguments for mathematical reasoning and poses hard to learning. In this paper, we propose an approach for training a generator to improve argument recognition. Our method enhances the probabilities of proper arguments in a reasoning program generation so that the arguments comprising the ground truth have higher weights. The proposed approach consists of an *argument aggregator* to model the probabilities in each candidate generation and an *argument set loss* to compute the cross-entropy between that probability and the candidates' existence in the ground truth in terms of the argument set. In our experiments, we show performance improvements of 3.62% and 3.98% in execution accuracy and program accuracy, respectively, over the existing FinQANet model based on a financial mathematical QA dataset. Also, we observed that the similarity of argument sets between the generated program and the ground truth improved by about 2.9%, indicating a mitigation of the misrecognition problem.

1 Introduction

Question Answering (QA) with mathematical reasoning on textual data (Saxton et al., 2019; Patel et al., 2021; Lu et al., 2023a; Cobbe et al., 2021; Amini et al., 2019; Dua et al., 2019) and tabular data (Zhu et al., 2021; Chen et al., 2021; Zhao et al., 2022) is an emerging area of research in natural language processing. This emerging field holds significant promise for various applications, including financial analysis, where it can aid businesses in making data-driven investment decisions (Vo et al., 2019) and enable financial analysts to evaluate market trends and risks more effectively (Lanbouri and Achchab, 2015; Wang, 2022).



Long-form Document:

Tabular Evidence

		2006	2005	2004	…
E_1	weighted average fair value of options granted	$ 20.01	$ 9.48	$ 7.28	…
E_2	…	…	…	…	…

Textual Evidence

E_3 the fair value for these options was estimated at the date of grant using a black-scholes … for 2006 , 2005 and 2004: .

…

E_4 the total fair value of shares vested during 2006, 2005, and 2004 was $ 9413, $ 8249 , and $ 6418 respectively.

…

E_5 during 2006 , 2005 , and 2004 , 124693, 112798 , and 117900 shares were purchased under the plan for a total purchase price of $ 3569 , $ 2824 , and $ 2691 , respectively.

Question: considering the weighted average fair value of options , what was the decrease between shares that vested in 2006 and 2005?

GT Program: divide(9413, 20.01), divide(8249, 9.48), subtract(#1, #0)

Misrecognition: divide(3569, 20.01), divide(2824, 9.48), subtract(#1, #0)

Answer: 400

Figure 1: Example of Argument Misrecognition with irrelevant information. *Green* highlights the arguments necessary for the answer, while *red* marks irrelevant arguments that don't contribute to the answer. The retrieved evidence as input context is shown in *purple*.

Recently, datasets such as FinQA (Chen et al., 2021) and ConvFinQA (Chen et al., 2022) have been released, aiming to perform mathematical reasoning over long-form hybrid data containing both tabular and textual information, such as financial statements. These tasks require generating a reasoning program as an arithmetical expression in response to a mathematical question. For instance, Figure 1 shows an example of the FinQA Task. A pipeline in FinQA first uses a retriever to retrieve supporting evidence that contains the necessary arguments from a long-form document. Then, a sequence-to-sequence program generator generates a solution program by combining arguments and operators from the retrieved evidence, as shown in the example of 'divide(9413, 20.01), divide(8249, 9.48), subtract(#1, #0)'. In this scenario, the program generator is required to be able to select the

# steps	% wrong args	% wrong ops	% both
1 step	50.97	21.93	27.10
2 steps	39.21	18.30	42.48
>= 3 steps	20.69	6.90	72.41
Total	41.26	18.03	40.71

Table 1: Percentage of errors in FinQANet attributed to selecting incorrect arguments or operators in the FinQA public test. these scores are based on program accuracy.

necessary arguments from multiple candidate arguments and sort them with operators for mathematical reasoning(Dua et al., 2019).

Such retrieve-then-generate framework(Brill et al., 2002; Chen et al., 2017; Lee et al., 2019; Li et al., 2023b; Zhang and Moshfeghi, 2022), to generate a correct answer, retrieved evidence must have all necessary arguments. To enable easier and more accurate retrieval of evidence containing all arguments, a coarse-grained region, such as a table row, is chosen as the evidence chunk(Wang et al., 2022). So, it often includes irrelevant information in the retrieved evidence. For instance, in Figure 1 from the FinQA Task, retrieved evidence like E_1, E_4, and E_5 includes numerous irrelevant arguments, marked in red. Including noisy information makes the model challenging and hinders reasoning performance (Wang et al., 2022; Xu et al., 2022; Shi et al., 2023). As shown in table 1, we can show that wrong argument selection is a more significant issue than incorrect operator generation. As the chain of reasoning gets longer, the number of wrong arguments and operators together increases, but the number of wrong arguments only still dominates.

In this paper, to address this problem, we introduce *Arguments Set Loss*, a novel approach aimed at enhancing arguments recognition in question-answering with mathematical reasoning. Our method focuses on fine-tuning the supervision of an existing sequence-to-sequence program generator, with a focus on the accurate generation of arguments. To facilitate this, we have developed an *Argument Aggregator*. This tool aggregates the distribution of arguments at each token step within the program generator, a key aspect in modeling the probability of correctly generating arguments. The aggregator calculates the probability of each candidate argument being generated. The arguments set loss aligns these probabilities with the ground truth program labels. We further refine the process by employing the Arguments Set Loss as an auxiliary loss in the training of the existing sequence-to-sequence generator, placing a greater emphasis on accurate argument selection within the reasoning program.

In summary, this paper makes the following contributions:

1. We highlight the argument misrecognition issue in mathematical QA and propose an *Arguments Set Loss* with an *Arguments Aggregator*.

2. We demonstrate that training models with *Arguments Set Loss* significantly improves performance on the FinQA and ConvFinQA datasets.

3. We empirically validate the effectiveness of the *Arguments Set Loss* in scenarios with noisy evidence, utilizing two distinct datasets and thorough case analyses.

2 Related Works

2.1 QA over Tabular and Textual Data

Recent research has increasingly focused on developing systems capable of answering questions over hybrid data from real sources, such as financial reports and Wikipedia, which combine tables and text (Wenhu Chen, 2021; Chen et al., 2020, 2021; Li et al., 2022; Eisenschlos et al., 2021; Krichene et al., 2021; Li et al., 2021; Zhao et al., 2022; Zhu et al., 2021; Herzig et al., 2021). HybridQA(Chen et al., 2020) is a dataset designed for extractive multi-hop reasoning, integrating both tabular and textual data from Wikipedia. OTT-QA(Wenhu Chen, 2021) is the open-domain setting of HybridQA. It requires retrieving tables and passages as relevant evidence sets from a document collections. TSQA (Li et al., 2021) is a multiple-choice QA dataset based on high-school exams, designed to assess understanding from scenarios and knowledge. Recently, there has been increasing focus on tasks involving mathematical reasoning on financial tables and textual data, such as TAT-QA (Zhu et al., 2021), FinQA (Chen et al., 2021), ConvFinQA (Chen et al., 2022), and MultiHiertt (Zhao et al., 2022). These datasets contribute to developing systems that understand and respond to complex mathematical queries.

2.2 Mathematical Reasoning

One of the key aspects of intelligence is mathematical reasoning(Lu et al., 2023b), especially in understanding NLP tasks that combine numerical information with language. It has been a focus for decades in algorithm development for solving mathematical problems (Newell et al., 1957). Recent research in NLP and machine learning has aimed to enhance the understanding of various question types, including those involving mathematical symbols, expressions, and equations. This work addresses diverse challenges such as solving math word problems (Patel et al., 2021; Lu et al., 2023a), proving theorems (Yang and Deng, 2019; Welleck et al., 2021), and mathQA tasks (Saxton et al., 2019). Researchers employ various methods to address these challenges, including program generation (Chen et al., 2021), span prediction (Dua et al., 2019), and prompting (Li et al., 2023a). There is also a focus on generating textual rationales to explain the reasoning process (Cobbe et al., 2021), enhancing interpretability in reasoning tasks.

3 Preliminary Background

We leverage FinQANet(Chen et al., 2021) as the baseline framework, which consists of an evidence retriever and a program generator. The evidence retriever retrieves relevant evidence set from the long-form document. The generator then generates a program as mathematical reasoning result based on the retrieved evidence.

Retriever FinQANet's retriever retrieves textual snippets from documents composed of free-form text and tables as evidence. The model employs templates to retrieve tabular data using the same model as text. Each cell in a table is transformed into a sentence using the template "column header is value." These sentences are then concatenated, with semicolons serving as separators for each row. For each candidate evidence passage e_i constructed in this way, the retriever model assigns a relevance score r^{e_i} based on its relevance to the question Q.

$$r^{e_i} = cls(Encoder(Q, e_i)) \tag{1}$$

Based on these scores, the retriever model selects the top n passages as retrieved evidence set E^r.

Generator In Figure 2, the first section on seq2seq program generation provides an overview of the Program Generator. The Program Generator generates a predicted program at each step t by predicting tokens for operators or argument indices. The token generation process draws from the following sources: 1. Arguments tokens: these are extracted from the question Q and the retrieved evidence E^r. 2. Predefined special tokens S: These are defined from the domain-specific language, such as the operator or constants. (e.g., add, table_max, const_10) 3. Step memory tokens M: these tokens indicate results from previous reasoning steps (e.g., #0, #1, ...). FinQANet's domain-specific language(DSL) arranges these tokens in a linear sequence to structurally represent reasoning operations and arguments step by step. Each operation function assigns step memory variables sequentially, defining the reasoning program, e.g., "add(arg1, arg2), divide(s0, arg3)". A detailed description of the operations and grammar is provided in Appendix A.

In order to generate reasoning program following DSL, FinQANet's Program Generator embeds the given question Q and retrieved evidence E^r using as pre-trained language model $Encoder$, such as RoBERTa, to obtain embeddings h^e.

$$h^e = Encoder(\ Q, [SEP], E^r\ ) \tag{2}$$

and, The embeddings h^s for the predefined special tokens S and h^m for the step memory tokens M are initialized randomly. Feed the embedding h^e, h^s, h^m as input to a recurrent generator based on LSTM with cross-attention and decode the predicted program token $\hat{y}$ by step t.

$$s^t, h^t = RNN(h^e, h^s, h^m, h_{t-1}) \tag{3}$$
$$\hat{y}^t = Argmax(s^t) \tag{4}$$

For training, FinQANet optimizes the loss $\mathcal{L}_{program}$ with the ground truth token s_t and uses the teacher forcing mechanism.

$$\mathcal{L}_{program} = \sum^{t} CE(\hat{s^t}, s^t) \tag{5}$$

4 Methodology

Our objective is to reduce the misrecognition of arguments in input passages containing irrelevant information, specifically focusing on improving the model's understanding of relationships between questions and arguments. To achieve this, we propose the *Arguments Set Loss*, a supervised learning approach that compares entire relevant arguments with the arguments in the generated programs during training, aimed at mitigating the problem of argument misrecognition. We trained the proposed arguments set loss along with the program loss of the baseline model to learn the arguments related to the question along with the sequential pattern of the reasoning program. The overall architecture of the proposed method is shown in Figure 2.

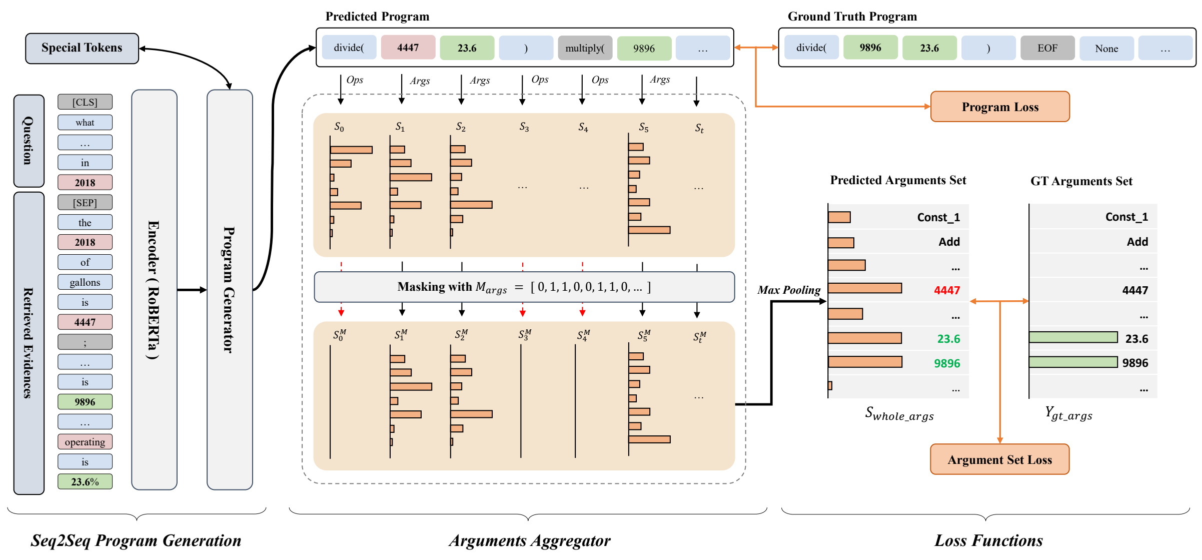

Predicted Program
divide(4447 23.6) multiply(9896 …
Ground Truth Program
divide(9896 23.6) EOF None …
Special Tokens
Question
Retrieved Evidences
[CLS] what … in 2018 [SEP] the 2018 of gallons is 4447 ; … is 9896 … operating is 23.6%
Encoder (RoBERTa)
Program Generator
Ops Args Args Ops Ops Args
Masking with M_{args} = [0, 1, 1, 0, 0, 1, 1, 0, …]
Program Loss
Predicted Arguments Set
GT Arguments Set
Const_1 Add … 4447 … 23.6 9896 …
Max Pooling
S_{whole_args}
Y_{gt_args}
Argument Set Loss
Seq2Seq Program Generation
Arguments Aggregator
Loss Functions

Figure 2: Overall architecture of the proposed framework. The left side of the figure is the Program Generator, which leverages FinQANet(Chen et al., 2021). The right side is the proposed Arguments Set Loss. We train the two objective losses together. Our methodology consists of an Argument Aggregator that generates a global argument distribution of the program generator and an argument set loss that aligns between the predicted argument set and the GT argument set.

4.1 Arguments Aggregator

Our method applies an additional auxiliary loss while maintaining the sequential program generation process in a seq2seq program generator, such as FinQANet's Generator. To this end, we utilize the token distribution of the existing program generator and the structural features of the domain-specific language (DSL) of the reasoning program to generate answers to questions in mathematical reasoning tasks as executable programs. In this domain-specific language, a solution program consists of functions as operators and operands as arguments. The arguments are trained to be generated at fixed positions in the predicted program. Utilizing these features, we have designed an *Arguments Aggregator* to aggregate the global distribution of candidate arguments for a question during training.

Figure 2 shows the process of the *Arguments Aggregator*. To illustrate the process with an example, assume the program generator is generating an incorrect program different from the ground truth program, like "divide(4447, 23.6), multiply(9896, ..." during training. *"4447"* marked in red is an improper argument, while the ones in green are proper arguments. First, at each step t of the program generator, we take the softmax distribution of the token $S = [s_1, s_2, s_3, ..., s_t]$. S is the distribution matrix for arguments and operators. And, to mask the distribution information of the operator in this matrix, an argument mask is created from the DSL.

The Arguments Mask M_{args}=$[M_1, M_2, ..., M_t]$, which takes a value of 1 at each Arguments Position in the DSL, is constructed as follows. Let T_{args} represent the Arguments positions in the DSL. Then, for all t:

$$M_t = \begin{cases} 1 & \textbf{if } t \in T_{args} \\ 0 & \text{otherwise}\,. \end{cases} \tag{6}$$

We obtain the Masked Vector S^M by applying Argument Position Masking to the output S of the Recurrent Generator.

$$S^M = S^T M_{args} \tag{7}$$

This matrix, masked results, is a distribution where the softmax distribution of the operator is masked to 0. To aggregate the softmax distribution of all candidate arguments, we use the maximum value of each argument in the softmax S_t, which represents the probability of each argument appearing in the model-generated result. We employ *MaxPooling* to generate the global argument distribution as follows:

$$S_{whole\_args} = MaxPooling(S^M) \tag{8}$$

In the example, the predicted arguments set is generated as the distribution with the highest generation probabilities of "4447", "23.6" and "9896".

4.2 Arguments Set Loss

To maximize the likelihood of generating proper arguments in the solution program creation, we propose an Arguments Set Loss that aligns the predicted Arguments Set with the GT Arguments Set, promoting proper arguments and depressing improper arguments in the distribution generated through the Arguments Aggregator.

We use the arguments that exist within the ground truth program as the ground truth argument set Y_{gt_args}. Regardless of the frequency of appearance in the program, we construct the labels as "true" for arguments that are present.

To optimize the model, we compute the cross-entropy between the global arguments distribution S_{whole_args} and the ground truth arguments label S_{gt_args}.

$$\mathcal{L}_{argset} = CE(S_{whole\_args}, Y_{gt\_args}) \quad (9)$$

We optimize the total objective loss L_{total}, which is the sum of the $L_{argsset}$ loss and the $L_{program}$ loss for the Program during training.

$$\mathcal{L}_{total} = \mathcal{L}_{program} + \mathcal{L}_{argset} \quad (10)$$

5 Experimental Setup

5.1 Datasets

We utilized datasets based on real-world documents containing noisy and irrelevant information within the evidence. **FinQA**(Chen et al., 2021) is designed for mathematical reasoning over long-form financial documents. The task involves answering mathematical questions by extracting relevant arguments and operators from both tables and free-form text in S&P 500 reports. **ConvFinQA**(Chen et al., 2022) is a dataset focused on conversational long-form mathematical reasoning over financial statements. this dataset requires the model to sequentially identify each relevant argument from conversations and generate reasoning steps accordingly.

5.2 Evaluation Metrics

We use three metrics to assess the mathematical QA and arguments recognition performance.

Program Accuracy: This metric determines whether the program solution is mathematically equivalent to the ground truth program.

Execution Accuracy: This metric evaluates the correctness of the final output of the executed program solution. It checks if the program solution, when run, provides the expected outcome.

Jaccard Similarity: To compare the accuracy of argument recognition with various reasoning paths, we use Jaccard similarity as a measure of the degree of matching between the arguments included in two programs. This metric measures the similarity between a set of arguments in the generated program ($args_{prog}$) and a set of arguments in the ground truth program ($args_{gt}$).

$$Similarity_{jaccard} = \frac{|args_{prog} \cap args_{gt}|}{|args_{prog} \cup args_{gt}|} \quad (11)$$

5.3 Baselines

We compare our approach against representative fine-tuned baselines and Large Language Models (LLMs). **FinQANet**(Chen et al., 2021), , which is the baseline model proposed in the FinQA Task, generates programs sequentially using an encoder-decoder based model. **DyRRen**(Li et al., 2023b): This model applies an evidence-level reranking module to FinQANet to attend to relevant retrieved information at each generation step. **ELASTIC**(Zhang and Moshfeghi, 2022), which is a model with generators separating operators and arguments to mitigate cascade errors. **CounterComp**(Nourbakhsh et al., 2023), which is a Contrastive loss approach to calculate triple loss with a negative sample of similar patterns of operators. To ensure fair comparisons, without the size of pre-trained models, ours is only compared with models based on the RoBERTa encoder sequence-to-sequence model. and, we reference the performance of Large-scale Language Models (LLMs) by considering **ChatGPT**, specifically zero-shot and CoT performances reported in (Li et al., 2023a).

5.4 Implementation Details

In the experiments, we utilized the default retrieval results in the FinQA dataset as input contexts. Our method is implemented based on the original code of FinQANet. The hyperparameter configuration, such as the hidden dimension size and the number of layers, remains the same as FinQANet. For all models, we use the Adam optimizer. The batch size is set to 12 for large models and 20 for base models. We selected the model with the highest Execution Accuracy based on the dev dataset for our final implementation, trained up to 300 epochs. Our method is trained 5 times with different random seeds from scratch and computes the mean and standard deviation of metrics. the QA performances of the baseline models are referenced from the published results in respective papers.

PLM	Method	FinQA(dev)		FinQA(Test)	
		Execution Accuracy	Program Accuracy	Execution ccuracy	Program Accuracy
FinBERT	FinQANet	46.64	44.11	50.10	47.52
ChatGPT	ZeroShot(Li et al., 2023a)	-	-	48.56	-
	CoT(Li et al., 2023a)	-	-	63.87	-
RoBERTa-base	FinQANet	56.27	53.49	56.10	54.38
	DyRRen*	59.00	56.62	57.80	55.88
	Ours	**63.19 ± 0.43**	**60.68 ± 0.41**	**59.46 ± 0.44**	**57.35 ± 0.42**
RoBERTa-large	FinQANet	61.22	58.05	61.24	58.86
	FinQANet*	63.87	61.15	61.38	59.28
	Elastic	65.00	61.00	62.16	57.54
	CounterComp	-	-	-	61.18
	DyRRen	66.82	63.87	63.30	61.29
	DyRRen*	63.87	61.27	62.51	60.42
	Ours	**67.50 ± 0.65**	**64.35 ± 0.67**	**64.86 ± 0.41**	**62.84 ± 0.24**
General Crowd Performance				50.68	48.17
Human Expert Performance				91.16	87.49

Table 2: Performance Comparison of Mathematical QA Models on FinQA dev and public test. Ours is the average score trained 5 times, and ‘ ± ’ indicates std. ‘-’ means that no score provided in the paper. ‘ * ’ denotes our implementation trained with evidence retrieved by FinQANet’s retriever.

6 Experiments

6.1 Financial QA Performance Comparison

FinQA Performance Using the above datasets and baselines, we evaluate our model, FinQaNet with Arguments set loss, and demonstrate its effectiveness in Table 2. Our approach exhibits a significant improvement in both Execution Accuracy and Program Accuracy metrics in the dev and public test sets, consistently and significantly outperforming all other baseline models.

Our approach has shown an execution accuracy of 67.50% and a program accuracy of 64.35 % in the dev while achieving 64.86% and 62.84%, respectively, in the test. When compared to *CounterComp*, which trains triplet loss among similar operator pattern examples, our model demonstrated 1.66% outperforming accuracy. This result demonstrates that our arguments-based approach is more robust in noisy retrieved evidence than operator-based approaches. Furthermore, in comparison to *DyRRen*, our approach exhibits a significant difference in both metircs. This result indicates that our method, which explicitly fine-grained supervises the arguments in the retrieved evidence, has a better understanding of noisy retrieved evidence than *DyRRen*, which dynamically assigns weight to evidence requiring argument extraction through a reranking module. DyRRen* is a DyRRen model using FinQANet’s Retriever to compare the performance of the Generator using the same input evidence set. In the comparison between ours and DyRRen*, we observe a 2.42% improvement in program accuracy. This improvement underscores the effectiveness of our approach as much as mitigating noise within evidence through pair-wise retriever for more accurate searches in DyRRen.

Method	ConvFinQA(dev)		ConvFinQA(Test)	
	EA	PA	EA	PA
GPT-2 (Medium)	59.12	57.52	58.19	57.00
T-5 (Large)	58.38	56.71	58.66	57.05
ChatGPT(ZeroShot)	59.86	-	-	-
FinQANet (base)	64.90	63.15	64.95	64.16
Ours (base)	**70.09** ± 0.59	**68.41** ± 0.54	**70.46** ± 0.79	**69.45** ± 0.57
FinQANet (large)	68.32	67.87	68.90	68.24
Ours (large)	**73.94** ± 0.33	**71.78** ± 0.63	**73.93** ± 1.07	**72.80** ± 0.98
General Crowd Performance			46.90	45.52
Human Expert Performance			89.44	86.34

Table 3: Performance Comparison on ConvFinQA dev and private test. Ours is the average score trained 5 times, and ‘ ± ’ indicates std.

Following the comparison with the fine-tuned approach, our method achieved higher execution accuracy compared to the performance of *ChatGPT*. This result demonstrates the efficiency of our approach compared to the performance of *ChatGPT*. *ChatGPT* showed similar performance compared to RoBERTa-based models, explaining the challenging aspects of the FinQA Task.

ConvFinQA Performance The performance of the model in the ConvFinQA task is also presented in Table 3. We also achieved performance improvements over baselines in the ConvFinQA dataset(Chen et al., 2022), which has different linguistic patterns. Compared to the baseline FinQANet, our approach showed consistent performance improvements in both base and large sizes of PLMs. Our, Roberta-large, achieved 73.94% execution accuracy, 71.78% program accuracy in the dev set, 73.93% execution accuracy, and 72.80% program accuracy in the private test. These results demonstrate that the proposed arguments set loss can robustly operate in datasets with conversation history-type linguistic patterns.

6.2 Financial QA Performance Breakdown

These experiments were conducted using default retrieval results from the dataset to ensure fair comparisons based on same search results. Only FinQANet and DyRRen had publicly available code among the baseline models, so we evaluated these two models.

In Table 4, we assess FinQANet* and DyRRen* performance in FinQA across different program lengths and question types. Our approach consistently outperforms other models as program length increases. Particularly, we achieve significant improvements in execution accuracy, reaching 70.18% for one-step programs, 63.32% for two-step programs, and an impressive 33.33% for programs exceeding two steps. DyRRen, which reranks evidence for each reasoning step, primarily improves performance in longer reasoning steps. However, our approach shows performance gains even in shorter reasoning steps and achieves greater improvements in longer cases, highlighting its effectiveness in handling complex reasoning processes.

Moreover, our model excels in analyzing various question types based on evidence types (Table-only, Text-only, Hybrid), particularly in scenarios with tabular data. It achieves remarkable execution and program accuracies of 73.09% and 70.82% in table-only scenarios, showcasing its proficiency in extracting information from structured tables. On the other hand, ours shows a similar performance improvement to other baselines in text-only, where relatively few candidate arguments occur compared to tabular evidence, which produces noisy evidence.

6.3 Arguments Recognition Performance

We conducted experiments to evaluate the impact of the proposed Arguments Set Loss on Arguments Recognition performance. Results in Table 5 are from experiments on the FinQA Dataset, focusing on program arguments recognition. The evaluation was based on Jaccard's Similarity among argument sets. Hard cases, identified by an average argument count of 21 in the trainset, were considered to assess noisy evidence.

Method	FinQANet*		DyRRen*		Ours	
	EA	PA	EA	PA	EA	PA
Program Steps						
1 step	67.28	65.14	67.58	65.90	**70.18**	**68.50**
2 steps	58.92	56.97	60.88	58.19	**63.32**	**60.39**
> 2 steps	27.38	25.0	30.95	28.57	**33.33**	**30.95**
Question Type						
Table-only	67.99	65.86	70.25	67.56	**73.09**	**70.82**
Text-only	55.83	54.06	56.18	**55.12**	**56.89**	54.77
Hybrid	41.77	39.24	39.24	37.97	**43.67**	**41.77**

Table 4: Performance Breakdown in the FinQA Dataset. We separate the Question Types based on the ground truth program step length and whether the Evidence is included in a table or free-form text.

	Overall	# of candidate ≤ 21	# of candidate > 21
FinQANet *	0.7661	0.7743	0.7530
DyRRen*	0.7776	0.7751	0.7813
Ours	**0.7951**	**0.7918**	**0.8004**

Table 5: Performance table for Arguments Recognition, a significant performance improvement compared to the baseline model, especially in cases with many candidate arguments.

Table 5 compares our method with FinQANet* and DyRRen*. Our approach outperforms both, with an overall Jaccard similarity of 0.7951, surpassing FinQANet by 0.029 and DyRRen by 0.0175. Particularly in challenging scenarios with more than 21 candidate arguments, our model achieved a Jaccard similarity of 0.8004, an improvement of 0.0474 over FinQANet. These results highlight the effectiveness of our model in enhancing arguments recognition, especially in noisy environments, and affirm its ability to handle diverse argument sets within the FinQA Dataset.

7 Further Study

In this section, we assess the impact of noisy type on QA performance by conducting experiments under scenarios with varying numbers of candidate options. Through this analysis, we observe that mathematical reasoning faces challenges with noisy evidence, and our model exhibits better performance on more noisy examples compared to the baseline. We also present a case study in Appendix B for a clearer understanding of the

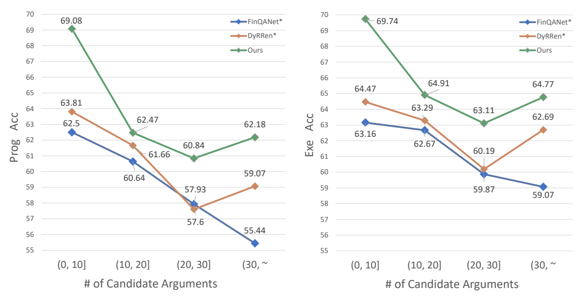


Figure 3: Mathematical Reasoning performance based on the number of argument candidates in input passages of the program generator. As the number of arguments candidates increases, there is a higher likelihood of including irrelevant arguments unrelated to the question.

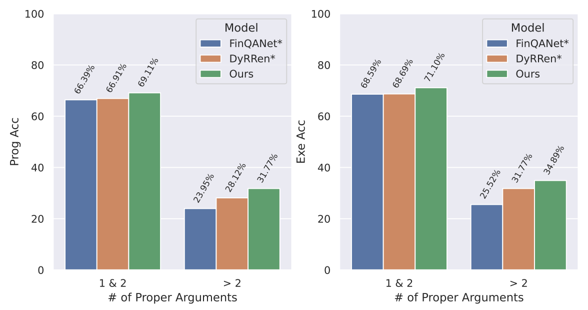


Figure 4: Performance based on different sizes of the proper argument in ground truth programs. Ours shows significant performance gains in the recognition of multiple arguments compared to the baseline.

improvements in argument recognition achieved through the proposed method

7.1 Performance on Argument Candidate Size

Figure 3 depicts the performance of mathematical QA tasks concerning the size of argument candidates in retrieved evidence. This analysis highlights the impact of argument set loss on performance amidst noisy retrieved evidence. Our results demonstrate that our method outperforms others across various ranges of argument candidates. Notably, our model achieves a program accuracy of 60.84% and an execution accuracy of 63.11% when dealing with more than 20 candidate sizes.

Significantly, our model shows that performance improvements become more pronounced as the number of argument candidates increases. While DyRRen also exhibits performance improvement with more than 30 arguments, it shows limited enhancement in certain ranges. In contrast, our model consistently shows enhanced performance when evidence contains numerous arguments. This enhancement in QA performance is attributed to the effective focus on appropriate arguments in noisy evidence using argument set loss. Therefore, argument set loss plays a pivotal role in enhancing QA performance, particularly in the presence of noisy evidence. Moreover, our model shows performance improvement even when the number of candidate arguments is small. This is especially important in cases where the evidence tokens are similar or when multiple arguments need to be extracted. In these scenarios, our model's effectiveness in recognizing and extracting relevant arguments contributes to its overall superior performance.

7.2 Performance on Proper Arguments Size

Figure 4 compares the performance when recognizing multiple proper arguments within the ground truth program. We examined our ability to identify these arguments without distraction from the noisy evidence. FinQANet achieves a program accuracy of 66.39% for 1-2 arguments and 23.95% for more than 2 arguments. DyRRen shows slight improvements, with 66.91% and 28.12% in the same categories. Ours achieves a program accuracy of 69.11% for 1-2 arguments and 31.77% for more than 2 arguments. The execution accuracy follows a similar trend. Our model demonstrates a significant performance difference of 7.82% for more than 2 arguments. These results indicate the effectiveness of our approach in QA performance when recognizing multiple arguments. It can be seen that explicit supervision loss at the argument level contributed to a significant performance improvement.

8 Conclusion

In this paper, we address the challenge of misrecognizing arguments in mathematical Question-Answering (QA), primarily caused by noisy evidence retrieval. To mitigate the issue of learning argument recognition from such noisy evidence, we propose an *Arguments Set Loss* as an auxiliary loss during training. This approach enhances the extraction and recognition of proper arguments from input passages. Additionally, we introduce the *Arguments Aggregator*, a novel reasoning program that leverages the structural features of domain-specific language to aggregate information about arguments during the training of seq2seq generators.

Our experiments on the FinQA and ConvFinQA datasets show a substantial improvement in mathematical QA performance, particularly in argument recognition for generating reasoning programs. We further analyzed the model's performance on argument recognition and compared its effectiveness across different types of noisy evidence. Our findings indicate that the Arguments Set Loss helps the model focus more effectively on recognizing arguments, even when dealing with irrelevant or noisy information. This enhancement is especially notable when generating complex reasoning programs. In future work, we plan to refine our approach to better manage operators, aiming to improve the accuracy and efficiency of mathematical QA systems.

Limitations

Our study primarily focuses on the generator in a QA pipeline, without addressing the retriever. we chose to concentrate on improving the generator's ability to recognize arguments, even when the retrieval results contain irrelevant information. Although refining the retriever or chunking mechanisms could further mitigate the impact of irrelevant data, these aspects were beyond the scope of our study.

The results of our experiments were compared to those validated on the FinQA leaderboard[1]. However, certain datasets designed for more complex table structures, such as MultiHiertt (Zhao et al., 2022), require not only mathematical reasoning but also extractive qa through span prediction. As a result, we did not include them in our experiments. In future work, we plan to explore whether the Arguments Set Loss can improve performance in extractive qa tasks alongside the retriever. While our method is effective, it is currently limited to mathematical reasoning tasks where operators and arguments are represented in list form. It faces challenges when applied to graphs and specific mathematical expressions. In future research, we aim to extend its applicability to a wider range of mathematical structures.

Acknowledgment

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [No.2022-0-00641, XVoice: Multi-Modal Voice Meta Learning]

[1]https://codalab.lisn.upsaclay.fr/competitions/4138

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. 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)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Eric Brill, Susan Dumais, and Michele Banko. 2002. An analysis of the AskMSR question-answering system. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 257–264. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *Findings of EMNLP 2020*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 2022. ConvFinQA: Exploring the chain of numerical reasoning in conversational finance question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6292, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

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, Volume 1*

(Long and Short Papers), pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Julian Eisenschlos, Maharshi Gor, Thomas Müller, and William Cohen. 2021. MATE: Multi-view attention for table transformer efficiency. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7606–7619, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Eisenschlos. 2021. Open domain question answering over tables via dense retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 512–519, Online. Association for Computational Linguistics.

Syrine Krichene, Thomas Müller, and Julian Eisenschlos. 2021. DoT: An efficient double transformer for NLP tasks with tables. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3273–3283, Online. Association for Computational Linguistics.

Zineb Lanbouri and Said Achchab. 2015. A hybrid deep belief network approach for financial distress prediction. In *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)*, pages 1–6. IEEE.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Moxin Li, Fuli Feng, Hanwang Zhang, Xiangnan He, Fengbin Zhu, and Tat-Seng Chua. 2022. Learning to imagine: Integrating counterfactual thinking in neural discrete reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 57–69, Dublin, Ireland. Association for Computational Linguistics.

Xianzhi Li, Xiaodan Zhu, Zhiqiang Ma, Xiaomo Liu, and Sameena Shah. 2023a. Are chatgpt and gpt-4 general-purpose solvers for financial text analytics? an examination on several typical tasks. *arXiv preprint arXiv:2305.05862*.

Xiao Li, Yawei Sun, and Gong Cheng. 2021. Tsqa: Tabular scenario based question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13297–13305.

Xiao Li, Yin Zhu, Sichen Liu, Jiangzhou Ju, Yuzhong Qu, and Gong Cheng. 2023b. Dyrren: A dynamic retriever-reranker-generator model for numerical reasoning over tabular and textual data. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023a. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *International Conference on Learning Representations (ICLR)*.

Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. 2023b. A survey of deep learning for mathematical reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14605–14631, Toronto, Canada. Association for Computational Linguistics.

A. Newell, J. C. Shaw, and H. A. Simon. 1957. Empirical explorations of the logic theory machine: A case study in heuristic. In *Papers Presented at the February 26-28, 1957, Western Joint Computer Conference: Techniques for Reliability*, IRE-AIEE-ACM '57 (Western), page 218–230, New York, NY, USA. Association for Computing Machinery.

Armineh Nourbakhsh, Sameena Shah, and Carolyn Rosé. 2023. Using counterfactual contrast to improve compositional generalization for multi-step quantitative reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14930–14943, Toronto, Canada. Association for Computational Linguistics.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Nhi NY Vo, Xuezhong He, Shaowu Liu, and Guandong Xu. 2019. Deep learning for decision making and the optimization of socially responsible investments and portfolio. *Decision Support Systems*, 124:113097.

Bai Wang. 2022. A financial risk identification model based on artificial intelligence. *Wireless Networks*, pages 1–9.

Yingyao Wang, Junwei Bao, Chaoqun Duan, Youzheng Wu, Xiaodong He, and Tiejun Zhao. 2022. MuGER2: Multi-granularity evidence retrieval and reasoning for hybrid question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6687–6697, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. 2021. Naturalproofs: Mathematical theorem proving in natural language. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Eva Schlinger William Wang William Cohen Wenhu Chen, Ming-wei Chang. 2021. Open question answering over tables and text. *Proceedings of ICLR 2021*.

Jialiang Xu, Mengyu Zhou, Xinyi He, Shi Han, and Dongmei Zhang. 2022. Towards robust numerical question answering: Diagnosing numerical capabilities of NLP systems. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7950–7966, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kaiyu Yang and Jia Deng. 2019. Learning to prove theorems via interacting with proof assistants. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6984–6994. PMLR.

Jiaxin Zhang and Yashar Moshfeghi. 2022. Elastic: Numerical reasoning with adaptive symbolic compiler. In *Advances in Neural Information Processing Systems*, volume 35, pages 12647–12661. Curran Associates, Inc.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6588–6600, Dublin, Ireland. Association for Computational Linguistics.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287, Online. Association for Computational Linguistics.

A Domain Specific Language on the FinQA and ConvFinQA

Following the domain-specific language defined by (Chen et al., 2021), we designed an Argument Aggregator. the reasoning program is defined using a domain-specific language, which can be structured in either a nested or a sequential format. In both the FinQA and ConvFinQA datasets, a reasoning program is used to represent the reasoning process for solving mathematical questions and answers. the reasoning program is defined using a domain-specific language, which can be structured in either a nested or a sequential format. In the nested format, operations are organized hierarchically, such as *"divide(add(arg1, arg2), arg3)"*. In contrast, the sequential format arranges operations linearly, with each operation executed in sequence. Unlike the nested format, the sequential format requires step memory variables to reference the result of previous reasoning steps. These variables are assigned sequentially, starting from 0 and denoted as "#0, #1, ...". They track intermediate outputs from one step to the next, as shown in *"add(arg1, arg2), divide(#0, arg3)"*. we adpoted the sequential format following the setting of FinQANet.

Table 6 outlines the operators defined for the FinQA task, along with examples. The operators consist of 6 arithmetic operators and 4 table operators. Arithmetic operators take 2 arguments, arg1 and arg2, and operate on numerical data or step memory variables within the evidence. Table operators use column names as arguments. When generating programs as sequences, a special token, "none", is padded to keep the same sequence as the arithmetic operators. As a result, each reasoning step generates 4 tokens, including the closing token ")". In our proposed method, we utilize this characteristic to mask the vectors of arguments from the softmax matrix generated within the program.

Operator	Example of sequence
add	add(, arg1, arg2,)
subtract	subtract(, arg1, arg2,)
multiply	multiply(, arg1, arg2,)
divide	divide(, arg1, arg2,)
exp	exp(, arg1, arg2,)
greater	greater(, arg1, arg2,)
table-sum	table_sum(, col name, none,)
table-average	table_average(, col name, none,)
table-max	table_max(, col name, none,)
table-min	table_min(, col name, none,)

Table 6: The operators defined by (Chen et al., 2021)

B Case Study

To clearly understand our method's improvement in mathematical QA, we present a case study of success cases and failure cases on the FinQA dataset. As shown in Figure 5, we sampled examples from the FinQA test for Ours and FinQANet with RoBERTa-large.

Success Case The first example demonstrates a case where our generator successfully reasons from a multitude of irrelevant arguments. The input evidence contained 58 candidate arguments due to the numerous columns in the table. The example requires recognizing and calculating the liability at the end of 2004 and 2005 to calculate the net change during 2005. In this case, tabular evidence is challenging to recognize proper argument because the values in all cells are linearized. e.g., *"liability as of January 1 2004 is $ 2239; liability as of December 31 2004 is $ 665; ... "*. The baseline model, FinQA, misrecognized 2239 next to the correct answer argument in this input and generated an incorrect answer program. On the other hand, our model successfully recognized the intended arguments. Our approach demonstrates enhanced performance in scenarios with many candidate arguments.

Error Case The second example represents a failure to recognize arguments pertinent to the question. The FinQA task involves reasoning based on an understanding of financial terminology (Chen et al., 2021). Despite applying our proposed loss function, our model struggled to recognize arguments in cases requiring nuanced financial reasoning. For instance, when considering the condition "1-3 years," the baseline model was misled by prioritizing the "total" aspect, whereas our proposed method correctly aligned with the "1-3" condition. However, in some instances, our model failed to recognize the context, erroneously associating certain parameters with the "total" condition. These failure cases underscore the challenges posed by intricate argument dependencies and nuanced financial reasoning, showcasing the areas where our model demonstrates improvement over baseline models in argument extraction tasks.

✓ ***Example of arguments recognition from large tabular evidence.***

Question: what is the net change in the balance of employee separations liability during 2005?

Evidence:

	liability as of january 1 2004	liability as of december 31 2004	…	2005 cash payments	liability as of december 31 2005	…
employee separations	$ 2239	$ 665	..	$ -448	$ 301	
..	…	…	..	…	…	…
total	$ 3689	$ 1096	…	$ - 773	$ 479	

Program: Ground truth: **subtract(301, 665)** FinQANet: **subtract(301, 2239)** Ours: **subtract(301, 665)**

✗ ***Failure to recognize due to lack of understanding of financial terms.***

Question: considering the contractual obligations in which payments due by 1-3 years , what is the percentage of the operating leases in relation to the total obligations?

Evidence:

contractual obligations	payments due by period total	payments due by period less than 1 year	payments due by period 1-3 years	payments due by period 3-5 years	payments due by period more than 5 years
operating leases	$ 44,048	$ 7,957	$ 13,789	$ 11,061	$11,241
purchase obligations	51471	47966	2265	1240	0
total	$ 95,519	$ 55,923	$ 16,054	$ 12,301	$11,241

Program: Ground truth: **divide(13789, 16054)** FinQANet: **divide(44048, 95519)** Ours: **subtract(13789, 95519)**

Figure 5: Two cases showing predicted reasoning program from the Ours and FinQANet (RoBERTa-large). Arguments that match the ground truth are highlighted in *green*, while incorrect arguments are indicated in *red*.