An Inference Method for Professional Texts with Computational Expressions under Few-shot Scenarios

Leiwen Yang[1], Wei Zheng[1], Feng Yuan[2], and Yuqing Sun[1]

[1] School of Software, Shandong University, China
[2] Shandong Shanda Oumasoft Co.Ltd, China
leoiwan@foxmail.com 357971254@qq.com sdyuanf@sina.com sun_yuqing@sdu.edu.cn

Abstract. We propose an inference method for complex professional texts with computational expressions. We use the expert rules to locate and rewrite the expressions. We adopt the pre-trained language model as the initial model and select the high quality samples from historical dataset for pre-training the model. The positive samples in the prompt are retrieved from the high-quality data that are similar to the inferred instance and the negative ones are generated by expert rules. For the target task, we fine-tune the model with the given few samples. Experimental results show that the performance of this method outperforms baselines and is applicable to low-resource scenarios.

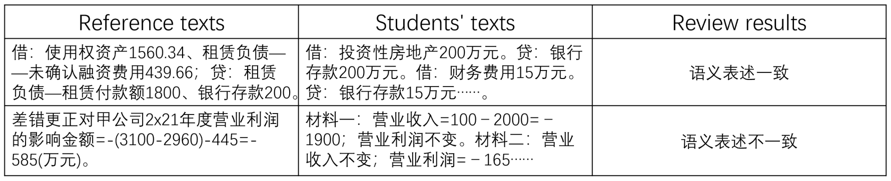
Keywords: Short Text Inference, Computational Expression, Few-shot Scenario

1 Introduction

Short text review is the practical application of text inference, which evaluates students' texts based on the given reference. In many specialty exams, there often exist computational expressions in text, such as the example in Figure 1, where the numbers and symbols contain the special meanings, that exceed the understanding capabilities of the traditional language models. Thus it is a challenge to infer this kind of professional texts with computational expressions.

Previous research has focused on inferring texts based on the human-labelled datasets by using the pre-trained language models [1, 2]. For example, Gao et al. compute the similarities between text features and manually set a similarity threshold for text inference [3]. Herasymovych et al. use reinforcement learning to learn the similarity threshold [4]. Sarkar et al. concatenate, align, or apply the mutual attention on features for inference and classification [5]. Compared to the above neural networks, tree-based classifiers offer better interpretability [6].

With the emergence of large language models (LLM for short), there is an increasing trend of applying these models to text inference tasks. Users can directly infer texts without any task-specific model training [7]. But this kind of approaches do not work well on the professional texts with computational expressions [8].



Reference texts	Students' texts	Review results
借：使用权资产1560.34、租赁负债——未确认融资费用439.66；贷：租赁负债—租赁付款额1800、银行存款200。	借：投资性房地产200万元。贷：银行存款200万元。借：财务费用15万元。贷：银行存款15万元……。	语义表述一致
差错更正对甲公司2x21年度营业利润的影响金额=-(3100-2960)-445=-585(万元)。	材料一：营业收入=100－2000=－1900；营业利润不变。材料二：营业收入不变；营业利润=－165……	语义表述不一致

Fig. 1: Examples of Professional Texts with Computational Expressions.

To solve the computational expression problem, some approaches adopt the chain of thought [9] to assist the model step-by-step reasoning. For example, Kojima et al. incorporate the prompt with prefix "let's think step by step" for LLMs [10]. Wang et al. [11] generate multiple computational paths to derive the results and aggregate them through voting, which improves the model accuracy. Lightman et al. [12] propose an automatic verification method for correcting the reasoning results. Sanh et al. also demonstrate that the quality of prompts significantly influences the model performance [13]. The computational expression and text inference are often studied independently, the above methods focus on the reasoning of computational expressions, and do not pay attention to text inference.

In this paper, we propose a method for inferring complex professional texts with computational expressions. We adopt the pre-trained language model as the initial model and select the high quality samples from historical dataset for pre-training the initial model. The positive samples in the prompt are retrieved from the high-quality data that are similar to the inferred instance and the negative ones are generated by expert rules. For the target task, we fine-tune the pre-trained model with the given few samples.

Experimental results on the real educational examination review data show that our method outperforms other methods in terms of performance. Even under low-resource scenarios, it achieves satisfactory results, meeting the requirements of practical applications.

2 Method

The proposed method consists of three main parts: rule guided data augmentation (2.1), pre-training (2.2) and fine-tuning (2.3) the inference model, as shown in Figure 2. Details are given below.

2.1 Rules Guided Data Augmentation

Recognize expressions from text. Computational expressions consist of numbers and arithmetic operators, which have special meanings that are distinct from usual texts and have significant impacts on the text semantics. So we adopt the expert rules to recognize these computational expressions in texts. For example, the rule $(?:\ d*\ .?\ d+[+-/])+\ d\ .?\ d+$ locates the expressions containing numbers, decimal points, and basic arithmetic operators.

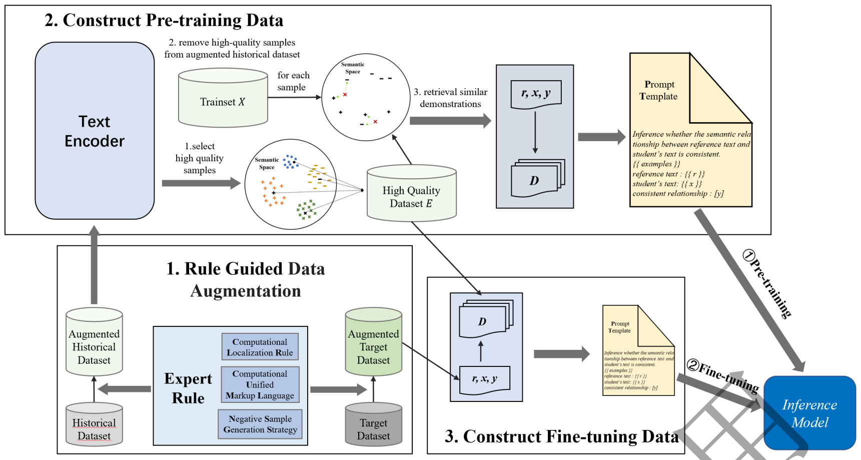

2. Construct Pre-training Data
Text Encoder
2. remove high-quality samples from augmented historical dataset
Trainset X
for each sample
Semantic Space
3. retrieval similar demonstrations
r, x, y
D
Prompt Template
Inference whether the semantic relationship between reference text and student's text is consistent.
{{ examples }}
reference text : {{ r }}
student's text: {{ x }}
consistent relationship : [y]
1.select high quality samples
Semantic Space
High Quality Dataset E
①Pre-training
1. Rule Guided Data Augmentation
Augmented Historical Dataset
Historical Dataset
Expert Rule
Computational Localization Rule
Computational Unified Markup Language
Negative Sample Generation Strategy
Augmented Target Dataset
Target Dataset
D
r, x, y
Prompt Template
3. Construct Fine-tuning Data
②Fine-tuning
Inference Model

Fig. 2: The Proposed Method

Rewrite the expressions with the unified markup language. We design the unified computation markup language (UML for short) to rewrite the recognized texts. For example, the special tokens [s_i_t] and [e_i_t] are added to both sides of an expression to emphasize the integer arithmetics, while the special tokens [s_f_t] and [e_f_t] are used to emphasize float the arithmetics.

Since in the student's texts, the same calculation process may be expressed by different forms, another function of the unified computational markup language is to align them with each others. For example, these expressions $600 * 0.01 = 6$, $600*1\% = 6$ and $600/100 = 6$ are with the same meanings. The unified expression can reduce the difficulty for inferring the meaning of expressions.

Construct the negative samples for a given correct expression. Computational expressions have a decisive impact on the semantics of the text. To make the model understand the meaning of different calculation processes, we design negative sample strategy to generate multiple negative samples for each correct computational expressions. For each positive samples, we rewrite the calculation process of an expression by changing some values and symbols. For example, $600 * 0.01 = 6$ can be rewritten as $600 * 0.01 = 7$ and $600 + 0.01 = 6$. By this, the diversity of computing process is enriched.

2.2 Pre-training the Text Inference Model with Historical Data

We adopt the historical data for data augmentation, as discussed in Sec.2.1. Then we construct the prompts with these data.

Prompt Construction. We design the prompt template $\mathcal{T}(x, D, r, y)$ to construction the model inputs, where x means a student's text, D means demonstrations that are similar to x. r and y mean the reference text and review results corresponding to x, respectively.

The construction steps are given as follows:

1. The high quality samples E: for the augmented dataset, the samples are grouped by labels. After encoding texts, we use $k-means$ to cluster sample in each group. We select the center sample as the high quality sample and add it to E.
2. The training samples X: we remove the high quality samples from the augmented samples. The rest of the samples constitute the training samples X.
3. The similar demonstrations D: for each $x \in X$, we select the closest sample in each group from E to form the similar demonstrations D [14] .

Pre-training model. We use the above constructed data to pre-train the text inference model. Our objective is to maximize the conditional probability $p(y|x, r)$, where p is the result probability predicted by the model. The loss function is the cross entropy loss.

$$\mathcal{L} = -y \log p + (1 - y) \log (1 - p) \quad (1)$$

2.3 Fine-tuning the Text Inference Model with a Few Samples in Target Task

For the target task, we also adopt a few labeled samples in target task for data augmentation, as discussed in Sec.2.1 For each sample in the augmented dataset, relevant demonstrations are retrieved from the high quality samples from the pre-training phase, as discussed in Sec.2.2, to construct context inputs. During the fine-tuning process, the loss function is similar to that of the pre-training process.

3 Experiments

3.1 Dataset and Implementation Details

We use the data of the 2022 accountant examination (AR21 for short) for experiments, which includes 21 questions. Each question contains 1 to 5 knowledge points, with the reference texts for the knowledge points. The first 14 questions are used as the historical data, while the remaining 7 questions are used as the target dataset under the low-resource scenarios. The statistics of the dataset are shown in Table 1.

The initial model's weights are sourced from the transformers library by Huggingface[3]. This study uses the Qwen-1_8B as the initial model for our method,

[3] https://huggingface.co

dataset	question	number/question	total
train	14	8897	124549
test	7	8243	57707

Table 1: The Statistics of Dataset.

with the hyper-parameters set to the default from finetune.py[4]. For retrieving the similar demonstrations, the bge-large-zh-v1.5 is used as the embedding model. During k-means clustering, k is set to 50.

We use accuracy and macro-f1 as the evaluation metric. The metric value is the average of three experiments on multiple problems.

As the comparison methods, we select the traditional text inference methods, categorizing BERT's CLS output, referred to here as **Supervised**. Additionally, **NLIPT** [15] is used as a comparative method with hyper-parameters following the original paper's settings. Furthermore, **COT** [10] is used as another comparative method. Since the COT does not require fine-tuning the model, it cannot be directly applied under low-resource settings. The experiments with the COT involve using 1, 3, and 5 demonstrations per input.

3.2 Results and Analysis

The experimental results on AE21 is shown in Table 2. The low-resource scenarios are each question random select 20,50,100 samples, respectively.

As the amount of training data increases, the performance of all methods generally improves. In the 20 samples/question low-resource scenario, our method outperforms the NLIPT, indicating its effectiveness in the low-resource settings. In the 100samples/question scenario, our method approaches the performance of supervised training with all training data. Notably, the simple COT method shows the much lower performance compared to other methods. This method may cause the LLM to focus only on the calculation process and ignore other texts. For such tasks, LLM should focus on both the expressions and the texts.

According to the results of the ablation experiments, each component of our method proves the positive effect to performance. In particular, historical data has an important impact on method performance.

Based on the experimental results in the 0 samples/question column of Table 2, our method demonstrates the zero-shot capability although it shows a considerable gap compared to the fine-tuned results. On the one hand, there is no labelled sample to indicate the target task domain and the label spaces. On the other hand, the training corpus of the large language models does not include the task domain.

[4] https://github.com/QwenLM/Qwen

samples/question	-		0		20		50		100	
	acc	f1	acc	f1	acc	f1	acc	f1	acc	f1
Supervised	99.26	99.26	44.08	30.87	77.57	77.38	88.05	88.05	94.90	94.90
NLIPT	-	-	-	-	94.02	94.01	94.26	94.26	96.65	96.65
Ours	-	-	55.36	43.31	**96.94**	**96.94**	**97.29**	**97.29**	**98.19**	**98.19**
w/o UML	-	-	-	-	95.94	95.94	97.27	97.27	97.37	97.37
w/o PT	-	-	-	-	80.77	80.69	88.57	88.57	94.46	94.46
w/o PT+UML	-	-	57.33	41.29	80.77	80.69	89.72	89.72	94.83	94.83
COT demonstration			-		1		3		5	
	-	-	-	-	51.90	34.17	49.30	33.17	49.25	33.01

Table 2: Experimental Results on AE21.

4 Conclusion

This paper presents a specialty text inference method tailored for mixed computational expressions. We adopt the pre-trained language model as the initial model and select the high quality samples from historical dataset for pre-training the model. The positive samples in the prompt are retrieved from the high-quality data that are similar to the inferred instance and the negative ones are generated by expert rules. For the target task, we fine-tune the pre-trained model with the given few samples. Experimental results demonstrate that this method performs well in the low-resource scenarios and outperforms baselines. For the future work, we are planning to conduct the interpretability analyses on evaluation results.

References

1. Jacob Devlin Kenton, Chang Ming-Wei, and Lee Kristina Toutanova et al. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.
2. Yinhan Liu, Myle Ott, and Naman Goyal et al. Roberta: A robustly optimized bert pretraining approach. *arXiv:1907.11692*, 2019.
3. Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP*, 2021.
4. Mykola Herasymovych, Karl Märka, and Oliver Lukason. Using reinforcement learning to optimize the acceptance threshold of a credit scoring model. *ASC*, 84:105697, 2019.
5. Soumyendu Sarkar. Effectiveness of deep networks in nlp using bidaf as an example architecture. *arXiv:2109.00074*, 2021.
6. Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *methods*, 5(13):25, 2018.
7. Ningyu Zhang, Luoqiu Li, and Xiang Chen et al. Differentiable prompt makes pre-trained language models better few-shot learners. In *ICLR*, 2021.
8. Xiaofei Sun, Xiaoya Li, and Jiwei Li et al. Text classification via large language models. In *EMNLP*, 2023.

9. Jason Wei, Xuezhi Wang, and Dale Schuurmans et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.
10. Takeshi Kojima, Shixiang (Shane) Gu, and Machel Reid et al. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.
11. Xuezhi Wang, Jason Wei, and Dale Schuurmans et al. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2022.
12. Hunter Lightman, Vineet Kosaraju, and Yuri Burda et al. Let's verify step by step. In *ICLR*, 2023.
13. Victor Sanh, Albert Webson, and Colin Raffel et al. Multitask prompted training enables zero-shot task generalization. In *ICLR*, 2021.
14. Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.
15. Leiwen Yang, Tao Yang, and Feng Yuan et al. Professional text review under limited sampling constraints. In *ChineseCSCW*, 2023.