软件学报 ISSN 1000-9825, CODEN RUXUEW
2025,36(5):2167−2187 [doi: 10.13328/j.cnki.jos.007206] [CSTR: 32375.14.jos.007206]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

多粒度单元格对比的文本和表格数值问答模型*

琚江舟, 毛云麟, 吴 震, 陈宇飞, 戴新宇, 陈家骏

(计算机软件新技术国家重点实验室 (南京大学), 江苏 南京 210023)

通信作者: 吴震, E-mail: wuz@nju.edu.cn

摘 要: 在文本和表格的数值问答任务中, 模型需要在给定的文本和表格下进行数值推理. 任务目标是生成一个包含多步数值计算的计算程序, 并将计算程序结果作为问题的答案. 为了建模文本和表格, 当前工作通过模板将表格线性化为一系列单元格句子, 再基于文本和单元格句子设计生成器以产生计算程序. 然而, 这种方法面临一个特定问题: 由模板生成的单元格句子间差异微小, 生成器难以区分回答问题所必需的单元格句子 (支撑单元格句子) 和回答问题无关的单元格句子 (干扰单元格句子), 最终导致模型基于干扰单元格句子生成错误的计算程序. 为了解决这个问题, 在生成器上设计一个多粒度单元格语义对比方法, 其主要目的是增加支撑单元格句子和干扰单元格句子表示距离, 进而帮助生成器区分它们. 这个方法由粗粒度单元格语义对比和细粒度单元格语义构成元素对比 (包括行名对比, 列名对比及单元格数值对比) 共同构成. 实验结果验证所提出的多粒度单元格语义对比方法可以使生成器在 FinQA 和 MultiHiertt 数值推理数据集上取得优于基准模型的表现. 在 FinQA 数据集上, 多粒度单元格语义对比方法上最高可以提升答案正确率达到 3.38%; 特别地, 在更为困难的层次化表格数据集 MultiHiertt 中, 该方法使生成器的正确率显著提高了 7.8%. 同大语言模型 GPT-3 结合思维链相比, 基于多粒度单元格语义对比的生成器性能在 FinQA 和 MultiHiertt 上分别表现出 5.44% 和 1.69% 的答案正确率提升. 后续分析实验进一步验证多粒度单元格语义对比方法有助于生成器区分支撑单元格句子和干扰单元格句子.

关键词: 表格和文本学习; 数值问答模型; 多粒度对比学习

中图法分类号: TP18

中文引用格式: 琚江舟, 毛云麟, 吴震, 陈宇飞, 戴新宇, 陈家骏. 多粒度单元格对比的文本和表格数值问答模型. 软件学报, 2025, 36(5): 2167–2187. http://www.jos.org.cn/1000-9825/7206.htm

英文引用格式: Ju JZ, Mao YL, Wu Z, Chen YF, Dai XY, Chen JJ. Text and Table Numerical Question-answering Model Based on Multi-granularity Cell Contrast. Ruan Jian Xue Bao/Journal of Software, 2025, 36(5): 2167–2187 (in Chinese). http://www.jos.org.cn/1000-9825/7206.htm

Text and Table Numerical Question-answering Model Based on Multi-granularity Cell Contrast

JU Jiang-Zhou, MAO Yun-Lin, WU Zhen, CHEN Yu-Fei, DAI Xin-Yu, CHEN Jia-Jun

(State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing 210023, China)

Abstract: In the task of numerical question-answering with texts and tables, the models are required to perform numerical reasoning based on given texts and tables. The goal is to generate a computational program consisting of multi-step numerical calculations, and the program's results are used as the answer to the question. To model the texts and tables, the current work linearizes the table into a series of cell sentences through templates and then designs a generator based on the texts and cell sentences to produce the computational program. However, this approach faces a specific problem: the differences between cell sentences generated by templates are minimal, making it difficult for the generator to distinguish between cell sentences that are essential for answering the question (supporting cell sentences) and those irrelevant to the question (distracting cell sentences). Ultimately, the model generates incorrect computational programs based on distracting cell sentences. To tackle this issue, this study proposes an approach called multi-granularity cell semantic

* 基金项目: 国家自然科学基金 (62376120, 61936012, 62206126)
收稿时间: 2023-12-21; 修改时间: 2024-03-01; 采用时间: 2024-04-02; jos 在线出版时间: 2024-06-20
CNKI 网络首发时间: 2024-07-02

contrast (MGCC) for our generator. The main purpose of this approach is to enhance the representation distances between supporting and distracting cell sentences, thereby helping the generator differentiate between them. Specifically, this contrast mechanism is composed of coarse-grained cell semantic contrasts and fine-grained constituent element contrasts, including contrasts in row names, column names, and cell values. The experimental results validate that the proposed MGCC approach enables the generator to achieve better performance than the benchmark model on the FinQA and MultiHiertt numerical reasoning datasets. On the FinQA dataset, it leads to an improvement of up to 3.38% in answer accuracy. Notably, on the more challenging hierarchical table dataset MultiHiertt, it yields a 7.8% increase in the accuracy of the generator. Compared with GPT-3 combined with chain of chain of thought (CoT), MGCC results in respective improvements of 5.44% and 1.69% on the FinQA and MultiHiertt datasets. The subsequent analytical experiments further verify that the multi-granularity cell semantic contrast approach contributes to the model's improved differentiation between supporting and distracting cell sentences.

Key words: table and text learning; numerical question-answering model; multi-granularity contrastive learning

问答模型在文本和知识图谱等领域已取得显著成功[1−4]. 然而, 在现实场景中, 存在大量文本和表格混合的文档, 如何设计问答模型解决基于文本和表格的数值推理问题, 在很多领域有着广泛的实际需求[5,6]. 如图 1 所示, 基于文本和表格数值问答是指模型在给定的文本和表格下进行数值推理, 生成一个计算程序 (program), 并将计算程序结果作为问题答案. 由于表格数据的复杂性以及文本与表格数据之间的异构特性, 这些因素给当前的问答模型带来了新的挑战, 并吸引了越来越多的关注[5−9].

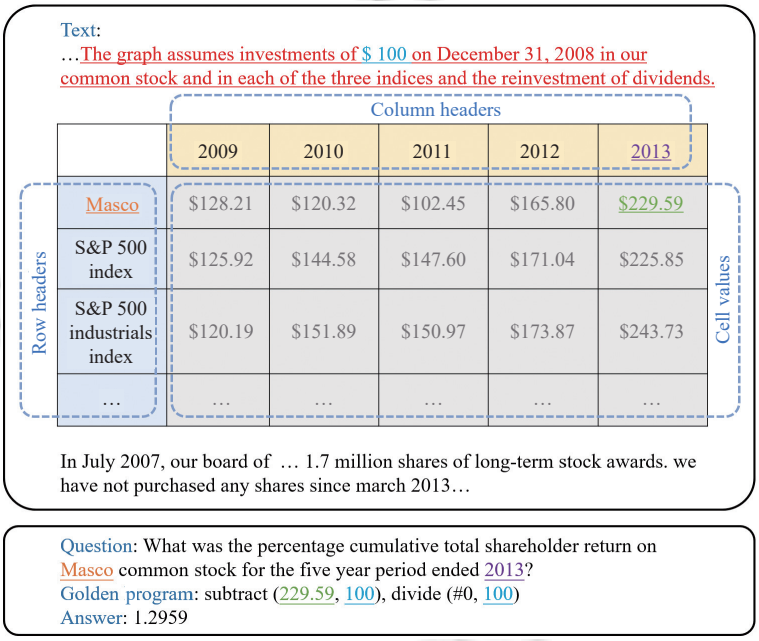


图 1　基于文本和表格数值推理问答任务示例

使用预训练语言模型在文本上进行数值推理已经取得了显著进展[10−15], 这表明了预训练语言模型在数值推理上的潜力. 结合表格特点及充分利用预训练语言模型的这种优势, 当前工作主要通过固定模板将表格线性化为表格文本, 然后设计基于预训练语言模型的问答模型在文本上进行数值推理[5−9], 图 2 展示了整个解决问题过程. 具体而言, 一些工作表明表格单元格语义主要由单元格数值及其对应的行名、列名这 3 个元素构成[16]. 因此当前工作通过由这三者构成的固定模板将表格单元格转化为反映其语义的单元格句子, 进一步可以将整个表格线性化为多个句子构成的表格文本, 图 3 展示了线性化表格过程. 最终, 表格文本和非结构化文本作为回答问题的输入. 但这些输入包含很多无关信息, 且长度超过预训练语言模型的最大输入长度限制. 因此, 一个检索器先被用于从表格文本和非结构化文本中检索相关句子, 然后一个生成器根据检索句子回答问题.

然而, 通过模板"The [row header] of [column header] is [cell value]"转化的这些单元格句子的语义差异很小. 这

是因为模板由行名 (row header)、列名 (column header) 及单元格数值 (cell value) 这三者构成, 而这些行名、列名通常较为简短, 甚至单元格句子可能仅会相差一个行名或者列名, 往往难以提供足够的信息区分它们. 例如, 图 3 中展示的两个经过线性化处理的句子: “The Masco of 2013 is $229.59.”和“The Masco of 2010 is $125.92.” , 其主要区别仅在于列名 (2013 vs. 2010) 的不同. 这种微小的差异会导致一个问题: 生成器难以区分检索器返回的支撑单元格句子 (即回答问题所必需的单元格句子) 和干扰单元格句子 (即与回答问题无关的单元格句子), 最终容易选择干扰单元格数值来生成错误的计算程序.

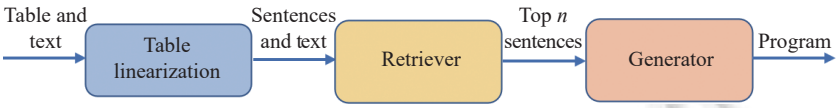


图 2 基于文本和表格数值问答任务流程

	2009	2010	…	2013
Masco	$128.21	$120.32	…	$229.59
S&P 500 index	$125.92	$144.58	…	$225.85
S&P 500 industrials index	$120.19	$151.89	…	$243.73
…	…	…	…	…

Table linearization

Template:
• The [row header] of [column header] is [cell value]
Output:
• The Masco of 2009 is $128.21.
• The Masco of 2010 is $125.92.
• …
• The Masco of 2013 is $229.59.
• …
• The S&P industrials index of 2012 is $120.19.
• The S&P industrials index of 2013 is $151.89.
• …

图 3 表格线性化示意图

为了解决生成器难以区分差异较小的支撑单元格句子和干扰单元格句子的问题, 当前的工作试图通过提升检索器的能力来去除干扰单元格影响. Chen 等人[5]最初提出用模板将单元格转化为代表单元格语义的单元格句子, 然后以表格整行的单元格句子作为基本检索单位, 之后将返回的检索句子作为生成器的输入. Wang 等人[6]认为检索表格的行会不可避免引入同行的干扰单元格, 因此他们提出了专门的单元格检索器来克服这个挑战. Li 等人[7]设计了一个 RankNet 成对损失来提升检索器. 尽管提升检索性能可以降低干扰单元格的负面影响, 但考虑成本和检索的难度, 完美的检索器仍难以实现. 因此, 生成器的输入必然包含检索返回的干扰单元格句子. 为了更好地应对生成器难以区分支撑和干扰单元格句子的挑战, 本文认为可以优化支撑单元格句子和干扰单元格句子的表示, 增加它们之间的距离, 使生成器可以有效地区分它们.

本文探索使用对比学习增加支撑单元格句子和干扰单元格句子表示间的距离. 对比学习可以减少同类别表示间的距离, 增加不同类别表示间的距离[17−19], 这种方法已在多种自然语言任务验证了其有效性[4,20−24]. 鉴于单元格句子整体语义及其语义构成元素这两种粒度都可以用于区分单元格句子, 本文在生成器上提出了一种多粒度单元格语义对比 (multi-granularity cell semantic contrast, MGCC) 学习方法, 以实现增加支撑和干扰单元格句子表示距离的目标. 这个多粒度对比方法包括粗粒度单元格语义对比和细粒度单元格语义构成元素对比. 后者进一步细分为单元格行名对比、单元格列名对比和单元格数值对比. 对于粗粒度单元格语义对比, 本文目的是增大支撑单元格句子的整体语义表示 $S\_support$ 和干扰单元格句子的整体语义表示 $S\_distrub$ 之间的距离. 以问题作为锚点, 本文通过 InfoNCE 损失[24]拉近问题表示和 $S\_support$ 之间的距离, 同时拉远问题表示和 $S\_distrub$ 之间的距离. 但是, 这一过程需要模型学习如何区分单元格句子语义构成元素差异, 例如行名、列名或单元格数值差异. 为了降低学习难度, 本文进一步提出了细粒度单元格语义构成元素对比. 对于细粒度对比, 本文期望增加支撑单元格句子语义构成元素表示 $E\_support$ 和干扰单元格句子语义构成元素表示 $E\_distrub$ 之间的距离. 本文通过 InfoNCE 损失减少问题表示和 $E\_support$ 之间的距离, 增加问题表示和 $E\_distrub$ 之间的距离. 由于单元格句子反映了单元格语义, 因此本文将对比方法称为多粒度单元格语义对比. 粗粒度单元格语义对比和细粒度单元格语义构成元素对比是互补的方法, 用于区分单元格的语义.

为了验证多粒度单元格语义对比学习 (MGCC) 的有效性, 本文在两个数据集上进行了测试: FinQA[5]和 MultiHiertt[8]. 结果表明, MGCC 能使生成器取得优于基准模型的表现. 具体来说, 在 FinQA 数据集上, MGCC 使生成器的答案正确率最高提高了 3.38%; 而在更具挑战性的层次化表格数据集 MultiHiertt 中, MGCC 实现了 7.8% 的答案正确率提升. 同大语言模型 GPT-3 结合思维链 (CoT) 相比, 基于多粒度单元格语义对比的生成器的性能在 FinQA 和 MultiHiertt 上分别表现出 5.44% 和 1.69% 的答案正确率提升. 进一步的实验分析也验证了 MGCC 有助于生成器区分支撑单元格句子和干扰单元格句子. 本文的主要贡献如下.

1) 在建模文本和表格的数值问答任务中, 存在一个特定的问题: 由于单元格句子的语义差异微小, 生成器难以区分支撑单元格句子和干扰单元格句子, 进而导致其容易基于干扰单元格句子生成错误的计算程序. 为了解决这一难题, 本文提出优化支撑单元格句子表示和干扰单元格句子表示, 增加它们之间的表示距离, 从而帮助生成器对它们进行更有效地区分.

2) 为了增加支撑单元格句子和干扰单元格句子表示之间的距离, 本文设计了多粒度单元格语义对比学习 (MGCC) 方法. 这个多粒度单元格语义对比分为粗粒度单元格语义对比 (针对单元格整体语义) 和细粒度单元格语义构成元素对比. 其核心思想是通过缩小问题与支撑单元格句子语义表示之间的距离, 同时扩大问题与干扰单元格句子语义表示之间的距离, 进而实现增加它们表示之间的距离.

3) 实验结果表明, 本文提出的多粒度单元格语义对比学习 (MGCC) 方法在 FinQA 和 MultiHiertt 数据集上分别提升了生成器的性能, 最高达到 3.38% 和 7.8%. 此外, 分析实验验证了 MGCC 方法可以有效帮助生成器区分支撑单元格句子和干扰单元格句子表示.

1 相关工作

1.1 基于文本和表格的数值推理

在文本和知识图谱等领域, 问答模型已经取得了显著进步[1−4,25]. 基于文本和表格的数值推理任务, 是近几年新出现的一项任务, 它涉及在异构的非结构化文本和半结构化表格中进行数值推理. Chen 等人[5]和 Zhao 等人[8]先后提出了基于文本和表格的数值推理数据集 FinQA 和 MultiHiertt, 这些数据集不仅需要模型输出答案, 同时也要求模型生成计算程序, 使问答过程更具可解释性. 为了处理半结构化的表格数据和利用预训练语言模型强大数值推理能力, Chen 等人[5]最初提出用模板将单元格转化为反映其语义的单元格句子, 然后以表格整行的单元格句子作为基本的检索单位, 设计检索器检索表格的行单元格句子. 这些检索到的句子随后作为生成器的输入来预测计算程序. Wang 等人[6]指出, 以整行的单元格句子作为基本检索单位会不可避免引入同行的干扰单元格, 因此他们提出了专门的单元格检索器来克服这个挑战, 并设计了一个结合多个检索器和生成器的集成模型. 本文同样认为干扰单元格会带来负面影响. 与其他研究不同的是, 本文在生成器中引入对比学习, 以帮助生成器模型区分支撑和干扰单元格. Li 等人[7]设计了一个基于 RankNet 成对损失的检索器和一个基于动态重排的生成器, 在每个生成步骤中对检索到的句子进行动态重排. Sun 等人[9]提出了一个基于数字感知负采样的检索器来优化检索. 这些工作都专注于降低检索器带来的噪音, 进而来帮助生成器生成计算程序. 但因为难以获得完美的检索器, 生成器仍面临难以区分支撑和干扰单元格句子的挑战. 因此, 为了更好地解决这一挑战, 本文提出在生成器中优化单元格句子表示, 以帮助区分支撑和干扰单元格句子.

与本文相关的两个任务分别是基于文本的数值推理任务 (如数学应用题)[10−15]和基于文本和表格的逻辑推理任务[26,27]. 不同于基于文本的数值推理, 本文研究聚焦于文本和表格的数值推理, 因此如何建模复杂的表格数据对基于文本的数值推理模型提出了新的挑战. 基于文本和表格的逻辑推理任务主要是利用文本事实进行逻辑推理, 很少涉及数值推理部分. 因此, 本文研究的任务可以看成是对基于文本和表格逻辑推理研究的重要补充.

1.2 对比学习

对比学习 (contrastive learning) 的核心思想是通过让同类别样本在表示空间中尽可能地彼此靠近, 不同类别样本尽可能地彼此远离[17−19], 这些优化后的表示可以使模型更有效地区分不同类别的特征, 已在多种自然语言处理

任务上都验证了其有效性[3,4,20−23]. 早期的研究主要集中在自监督对比学习, 他们使用数据增强技术构造锚点、正例及负例. 方法是拉近锚点和正例之间的距离, 同时拉远锚点与负例之间的距离. 通过这种自监督的预训练来进行表示学习, 对比学习优化后的表示在很多下游任务上获得了与监督学习类似的效果, 这些结果说明了对比学习可以有效地帮助模型学习每个类别的特征. 因此, 本文采用对比学习来区分不同类型单元格语义表示. 值得注意的是, He 等人[18]指出在自监督对比学习训练中, 负样本的数量非常重要. 他通过实验表明, 随着负样本数量的增加, 学习到的表示对下游任务更加有益. 最近, 对比学习在问答任务上也获得了越来越多的关注. Karpukhin 等人[3]使用对比学习改进了基于稠密向量的检索器, 使用一个简单但有效的对比损失函数来训练检索器, 让与问题相关的段落在表示空间中接近于问题, 而与问题无关的段落远离问题. Yang 等人[23]进一步优化动量对比 (momentum contrast, MoCo), 以获得大量的对比学习负样本来提升问答模型的检索器. 相比 Yang 等人[23]和 Karpukhin 等人[3]的工作, 本文在生成器上利用对比学习优化单元格句子的表示. Caciularu 等人[4]通过一种新颖的对比损失来增强问答模型, 其中模型被鼓励最大化支撑段落与问题之间的相似度, 最小化干扰段落与问题之间的相似度. 对比 Caciularu 等人[4]的工作, 本文研究任务是基于文本和表格进行数值推理. 而对比学习被用于生成器中, 以帮助其区分表格支撑单元格句子表示和干扰单元格句子表示.

2 基于文本和表格的数值问答任务及通用框架

基于文本和表格的数值问答任务是指模型在给定的非结构化文本和半结构化表格下进行数值推理. 为了处理异构的文本和表格, 当前工作首先利用设计的固定模板将单元格转化为单元格句子, 进而将表格线性化为单元格句子构成的表格文本, 图 3 展现了这个过程的一个示例. 但是因为非结构化文本和表格文本包含较多干扰项且超过预训练语言模型最大输入长度, 因此当前研究提出检索器-生成器框架来解决问题[5−9]. 其中检索器被用于从非结构化文本 (包含多个文本句子) 和表格文本 (由单元格句子构成) 中检索相关的句子. 之后生成器将这些检索的相关句子作为输入, 进行数值推理并生成计算程序, 图 2 展示了整个问答模型的工作流程. 本文提出了一种方法, 通过采用对比学习来提升基于检索器-生成器 (encoder-decoder) 框架的问答模型的性能. 因此, 本节将简要介绍基于文本和表格的数值推理任务和当前的模型框架.

2.1 任务定义

文本和表格上的数值推理: 给定一个问题 Q, 问答模型需要对由多个句子组成的文本 T 和一个或多个表格 K 构成的文档进行数值推理. 模型将生成一个计算程序 G 来回答问题 Q, 并返回该程序的计算结果作为答案. 计算程序一般由定义操作数、操作符构成. 在必要的情况下, 还需定义一些特殊的符号, 如图 1 计算程序中的第 i 步推理结果#i.

$$P(G|Q,T,K) = P_{\phi}(G|C,Q)P_{\theta}(C|T,K,Q) \tag{1}$$

在检索器-生成器框架下, 检索器先检索相关句子, 然后生成器根据检索句子和问题 Q 产生计算程序 G. 公式 (1) 描述模型生成计算程序 G 的概率过程. 这里 P_{θ} 代表检索器, P_{ϕ} 代表生成器, C 是检索器选择的句子集合. 检索器 P_{θ} 和生成器 P_{ϕ} 将在第 2.2 节分别介绍.

2.2 数值问答模型框架

2.2.1 表格线性化

为了统一表格和文本, 目前主要通过模板“The [row header] of [column header] is [cell value]”将表格转化为表格文本[5−9]. 图 3 展示了通过模板转化简单表格[5]的过程. 但对于第 3.3 节展示的层次化表格[8], 其特点是其行名和列名由多个层次行名或者列名构成. 本文按从上往下的顺序, 依次命名为第 i 层行名 (列名), 其中 i 从 1 开始. 按照已有工作[8], 对于单元格列名, 本文从第 1 层列名开始, 逐层拼接到最后一层列名, 形成单元格 (组合) 列名; 对于行名, 本文从最后一层行名开始, 逐层拼接到第 1 层行名, 将拼接的行名作为单元格 (组合) 行名. 最后按照上述模板转化层次化表格为表格文本.

2.2.2 检索器

检索器主要是被用于检索相关的句子, 这些相关句子将作为后续生成器的输入. 现有的工作首先利用预训练

语言模型 (如 BERT) 构建一个二分类器, 从非结构化文本和表格文本组成的句子集合 S 中挑选前 top_n 个检索句子. 给定一个问题 Q, 检索到的相关句子集合 C 可以根据公式 (2) 的事实指标集合 I 从 S 中得到:

$$I = \text{argsort}(\{P_\theta(C_i|Q)|i \in N\}[:top_n]) \tag{2}$$

其中, C_i 是第 i 个句子, N 是 S 中每个句子的指标集合, θ 是分类器参数, $P_\theta(C_i|Q)$ 是分类器判定 C_i 为支撑句子 (支撑单元格句子或者回答问题所必需的文本句子) 的概率, argsort 指对概率集合 $\{P_\theta(C_i|Q)|i \in N\}$ 进行降序排序, 并返回与这些概率相对应的句子指标.

2.2.3 生成器

本文旨在通过引入多粒度单元格语义对比机制, 提升生成器区分支撑单元格句子和干扰单元格句子的能力. 因此, 本文首先介绍主流的生成器, 而基于对比机制的生成器将在第 3 节介绍. 主流的生成器采用 encoder-decoder 架构来做数值推理, 其以预训练语言模型为核心的 encoder 编码检索器返回的检索句子集合 C 和问题 Q. 然后由一个 LSMT 及 attention 机制构成的 decoder 来解码计算程序 $G = \{g_i\}_{i=0}^{N}$, 其中 G 包含定义的操作符, 操作数及特殊符号 (例如代表第 i 步计算结果的特殊符号#i) 等构成元素. 这里的 g_i 表示计算程序中的第 i 个构成元素. 生成器一般通过计算程序 G 作为监督学习信号, 从而指导模型优化其参数配置. 公式 (3) 显示了生成器推理过程:

$$P_\phi(G|C, Q) = \prod_{t=0}^{M} P_\phi(g_t|C, Q, g_{<t}) \tag{3}$$

其中, ϕ 是生成器参数, M 是计算程序长度.

3 基于多粒度单元格语义对比的生成器

文本和表格的数值推理问答中, 模型需要基于单元格句子进行数值推理, 然后生成包含单元格数值的计算程序. 然而, 生成器区分支撑和干扰单元格句子是有挑战的, 主要理由是: (1) 由模板构造的单元格句子基本上由简短的行名和列名组成, 但这些行名和列名在形式上差异很小, 难以提供足够的信息区分它们的语义; (2) 单元格句子可能仅相差一个行名或者列名, 这进一步减少了它们之间的语义差异[6]; (3) 这些简短的行名或者列名可能和问题有重合词汇, 也会导致生成器难以区分它们的语义; (4) 由于标注成本昂贵, 导致训练数据是有限的. 同时, 模板句子与预训练语言模型的训练语料之间存在差异. 这些都可能会使得生成器难以学习到高质量的单元格句子语义表示, 进而导致生成器容易基于干扰单元格数值生成错误的计算程序. 对比学习的一个核心思想是拉近同类表示的距离, 拉远不同类别表示的距离, 优化的表示可以进一步提升模型的性能[20−23]. 受益于对比学习的思想, 本文期望用对比学习方式来优化支撑和干扰单元格句子语义表示, 拉远它们的语义表示距离, 进而帮助生成器区分它们. 除了由粗粒度角度出发考虑单元格句子整体语义差别, 同时本文也从细粒度角度出发利用单元格句子语义构成元素 (行名, 列名和单元格数值) 来告诉生成器模型更具体的差异. 因此, 本文提出了一种多粒度单元格语义对比 (MGCC) 方法, 来帮助生成器区分支撑和干扰单元格句子.

后文图 4 展示了基于多粒度单元格语义对比的生成器架构. 首先生成器 encoder 对问题和检索返回的句子进行编码. 然后, 本文提出的多粒度单元格语义对比用于优化 encoder 的输出. 基于这些优化后的输出, decoder 可以更好地区分支撑单元格句子和干扰单元格句子表示. 具体地, 本文将在第 3.1 节介绍图 4 中 encoder 的输入. 图 4 中的 SemContrast 是粗粒度单元格语义对比, 该部分将在第 3.2 节介绍. 图 4 中的 RowContrast, ColContrast 和 ValContrast 分别代表细粒度单元格语义构成元素的行名对比, 列名对比和单元格数值对比, 本文将在第 3.3 节介绍这些内容. 此外, 这里提到的 decoder 与第 2.2.3 节介绍的 decoder 是一致的.

3.1 多粒度表格单元格语义对比的生成器输入设计

为了获得多粒度单元格语义对比的正负例表示, 本文对生成器的输入做了改进. 给定一个问题 Q, 通过检索, 本文可以获得前 top_n 个检索句子 $C = < C_1, C_2, \ldots, C_{top_n} >$, 这里 C_i 是第 i 个检索句子. 不同于之前的生成器直接拼接 Q 和 C, 这里本文在每个检索句子 C_i 后面拼接一个特殊单词 [SEP], 其可以作为每个 C_i 的表示[4]. 具体输入为 [CLS] Q [SEP] C_1 [SEP] C_2 [SEP] … C_{top_n} [SEP] . 之后本文将拼接结果作为 encoder 输入, 得到表示 H^c .

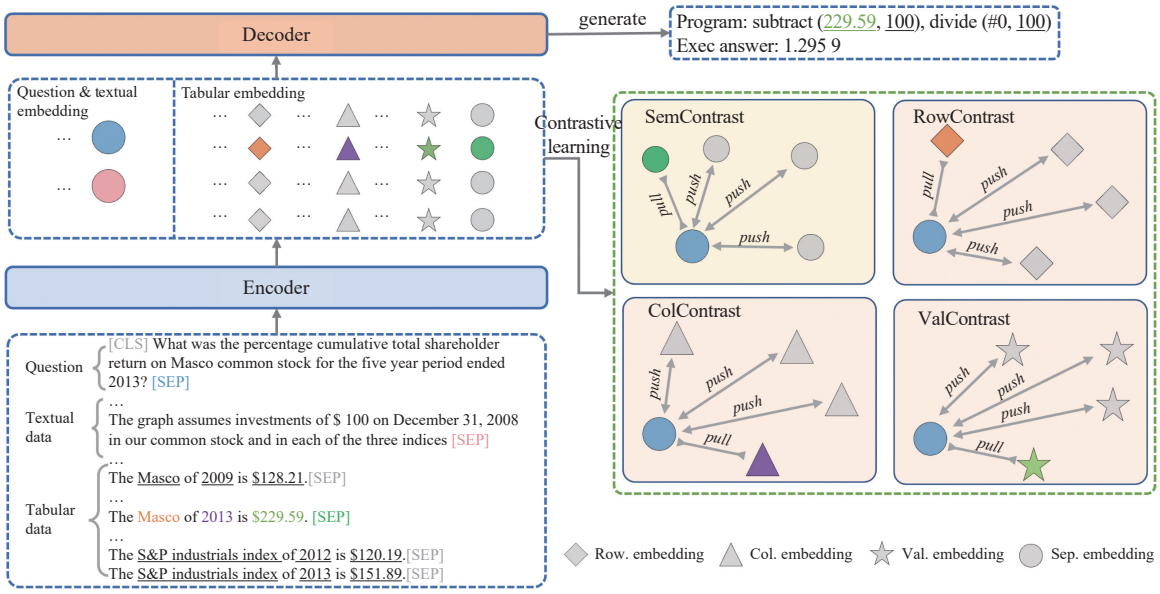


图 4 基于多粒度单元格语义对比的生成器框架

3.2 粗粒度单元格语义对比

为了使生成器可以区分支撑单元格句子和干扰单元格句子, 本文提出了一种粗粒度单元格语义对比学习方法. 此方法拉远支撑单元格句子整体语义表示 *S_support* 和干扰单元格句子整体语义表示 *S_distrub* 之间的距离. 因为支撑单元格和干扰单元格由问题确定, 所以不同于直接拉近 *S_support* 之间的距离, 拉远 *S_support* 和 *S_distrub* 的距离. 这里本文以问题 *Q* 作为锚点, 以 *S_support* 作为正例, 以 *S_distrub* 作为负例. 然后拉近问题 *Q* 表示与 *S_support* 之间的距离, 拉远问题 *Q* 表示与 *S_distrub* 之间的距离, 进而实现区分支撑和干扰单元格句子. 因为单元格句子整体语义也反映单元格整体语义, 所以本文称粗粒度对比方法为粗粒度单元格语义对比. 这里本文使用 InfoNCE[24]损失函数达成上述目的.

锚点表示: 本文以问题经过分词器后子词的平均表示作为问题的表示; 或者类似于 Caciularu 等人[4]的做法, 本文用 *Q* 后的特殊单词 [SEP] 表示作为问题的表示.

正例表示: 本文将支撑单元格句子的所有子词的平均表示或者支撑单元格句子紧随的 [SEP] 作为单元格句子整体语义表示, 随机采样固定数量 *K*1 的支撑单元格句子整体语义表示作为正例表示.

负例表示: 本文用干扰单元格所有子词的平均表示或者干扰单元格句子紧随的 [SEP] 表示作为干扰单元格句子整体语义表示, 随机采样固定数量 *K*2 的干扰单元格句子整体语义表示作为负例表示.

损失计算: 本文使用 InfoNCE 损失拉近问题表示和 *S_support* 之间的距离, 同时拉远问题表示与 *S_distrub* 之间的距离. 具体见公式 (4):

$$\mathcal{L}_{cg} = -\sum_{u^{\dagger}\in U_{cg}^{\dagger}} \log\left(\frac{e^{\mathrm{sim}(u^{\dagger},q)/\tau}}{\sum_{u\in(U_{cg}'\vee u^{\dagger})} e^{\mathrm{sim}(u,q)/\tau}}\right) \tag{4}$$

其中, $U_{cg}^{\dagger}$ 为正例表示的集合, U_{cg}' 是负例表示的结合, τ 是温度系数, τ 越小, 代表模型更加关注如何将锚点和更困难负例区分开, $\mathrm{sim}(\cdot,\cdot)$ 是相似度函数, 其计算过程见公式 (5):

$$\mathrm{sim}(u,q) = \frac{\tilde{u}^{\mathrm{T}}\tilde{q}}{\|\tilde{u}\|\|\tilde{q}\|} \tag{5}$$

其中, $\|\cdot\|$ 表示求 ℓ_2 范数. 为了避免问答任务的学习目标和对比学习冲突, 和其他工作一样[4], 本文通过映射函数 f_1 和 f_2, 将来自于 H^c 的原始锚点表示 q 和正负例表示 u 映射到一个辅助空间进行对比学习. 公式 (6) 反映了这个

过程. 在第 4.8.2 节, 本文探究了不同映射函数的影响.

$$\tilde{u} = f_1(u),\ \tilde{q} = f_2(q) \tag{6}$$

3.3 细粒度单元格语义构成元素对比

因为单元格句子语义差异来自于其构成元素 (行名, 列名, 单元格数值), 所以本文从细粒度单元格句子语义构成元素出发, 通过区分单元格句子语义构成元素来使模型区分支撑单元格句子和干扰单元格句子. 最终本文提出了细粒度单元格语义构成元素对比方法, 其由 3 部分构成包括: 细粒度单元格行名对比, 细粒度单元格列名对比和细粒度单元格数值对比. 因为单元格语义构成元素和单元格句子语义构成元素是一致的, 因此本文将细粒度对比方法称为细粒度单元格语义构成元素对比. 细粒度单元格语义构成元素对比可以明确地告诉模型单元格句子语义的具体差异 (行名, 列名, 单元格数值), 而粗粒度对比强调单元格句子语义整体差异, 这两者可以被看成是相互补充的方法来区分单元格句子.

3.3.1 细粒度单元格行名对比

通过区分单元格行名可以区分不同单元格句子语义, 因此本文提出细粒度单元格行名对比方法. 本文用问题作为锚点, 拉近问题 Q 表示与支撑单元格句子行名表示之间的距离, 同时拉远问题 Q 表示与干扰单元格句子行名表示之间的距离.

锚点表示: 本文以问题经过分词器后子词的平均表示作为问题的表示; 或者本文用 Q 后的特殊单词 [SEP] 表示作为问题的表示.

正例表示: 对于简单表格, 本文随机采样固定数量 $K1$ 的支撑单元格行名, 将它们作为正例行名. 对于层次化表格, 为了更有效区分支撑和干扰单元格句子行名表示, 本文从支撑单元格的层次化行名和由层次化行名拼接而成的组合行名中随机采样固定数量 $K1$ 的行名作为正例行名. 正例行名对应的子词平均表示被作为正例行名表示.

下面本文通过图 5 介绍层次化正例行名和组合正例行名. 在图 5 所示的层次化表格中, 对于下划线标记的支撑单元格 946, 其第 1 层正例行名是“change in plan asserts”, 第 2 层正例行名是“actual return on plan asserts”. 其组合正例行名是“actual return on plan asserts change in plan asserts”.

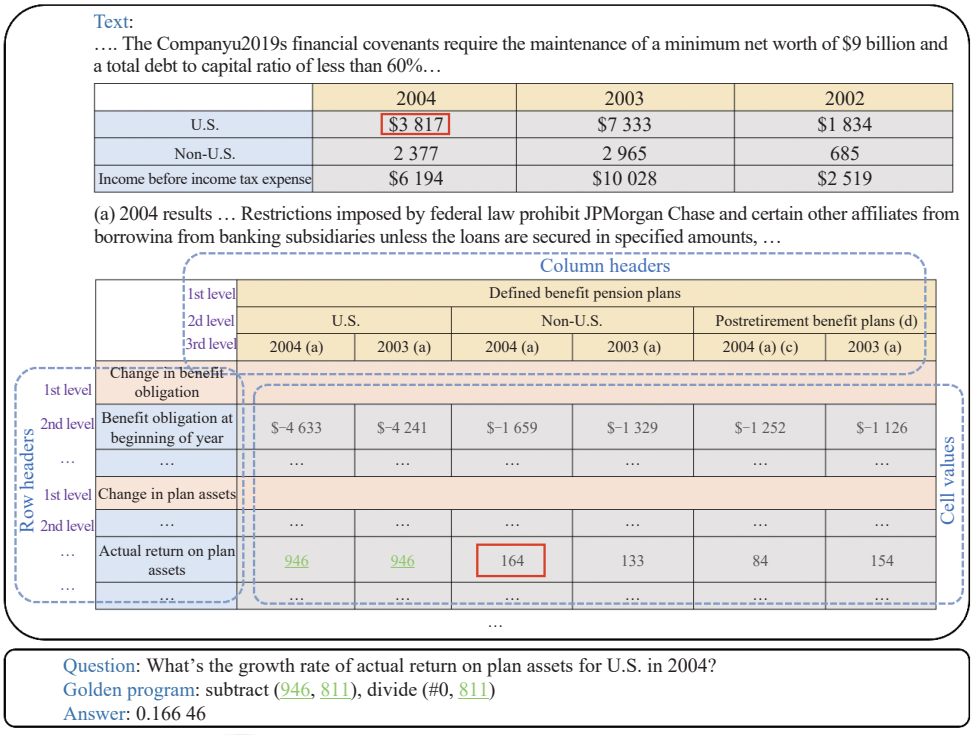


图 5 MultiHiertt 数据集问答任务示例

负例表示: 对于简单表格, 本文随机采样固定数量 $K2$ 的干扰单元格行名, 将它们作为负例行名. 对于层次化表格, 其层次化负例行名由层次化正例行名确定. 具体而言, 对于层次化正例行名, 其层次化负例行名来自于相同表格的干扰单元格且与层次化正例行名的层次相同, 本文随机采样固定数量 $K2$ 的这种层次化行名作为负例行名. 对于组合负例行名, 本文从干扰单元格句子的组合行名和简单表格的干扰单元格行名中随机采样固定数量 $K2$ 的行名作为负例行名. 负例行名对应的子词平均表示作为负例行名表示.

同样基于图 5, 本文用方框标记检索返回的干扰单元格. 以支撑单元格 946 为例, 本文选择干扰单元格 164 作为负例. 因此, 支撑单元格 964 的第 1、2 层正例行名对应的负例行名分别为“change in plan asserts” (第 1 层负例行名, 来自单元格 164) 和“actual return on plan asserts” (第 2 层负例行名, 来自单元格 164). 支撑单元格 964 组合正例行名对应的负例包括组合负例行名“actual return on plan asserts change in plan asserts” (来自单元格 164) 和来自于简单表格的负例行名“U.S.”.

3.3.2 细粒度单元格列名对比

单元格列名可以被用于区分不同单元格句子语义, 其是单元格句子重要组成部分, 因此这里本文提出细粒度单元格列对比. 本文用问题 Q 作为锚点, 拉近问题 Q 表示与支撑单元格句子列名表示之间的距离, 拉远问题 Q 表示与干扰单元格列名表示之间的距离.

锚点表示: 本文以问题经过分词器后子词的平均表示作为问题的表示; 或者用 Q 后的特殊单词 [SEP] 表示作为问题的表示.

正例表示: 对于简单表格, 本文随机采样固定数量 $K1$ 的支撑单元格列名, 将它们作为正例列名. 对于层次化表格, 本文从支撑单元格的层次化列名和由层次化列名拼接而成的组合列名中随机采样固定数量 $K1$ 的列名作为正例列名. 正例列名对应的子词平均表示被作为正例列名表示.

在图 5 中, 支撑单元格 946 对应的第 1、2、3 层正例列名分别是“defined benefit pension plans” (第 1 层正例列名), “U.S.” (第 2 层正例列名) 和“2004(a)” (第 3 层正例列名). 其组合正例列名是“defined benefit pension plans U.S. 2004(a)”.

负例表示: 对于简单表格, 本文随机采样固定数量 $K2$ 的干扰单元格列名, 将它们作为负例行名. 对于层次化表格, 其层次化负例列名由层次化正例列名确定. 具体而言, 对于层次化正例列名, 其层次化负例列名来自于相同表格的干扰单元格且与层次化正例列名的层次相同, 本文随机采样固定数量 $K2$ 的这种层次化列名作为负例列名. 对于组合负例列名, 本文从干扰单元格句子的组合列名和简单表格的干扰单元格列名中随机采样固定数量 $K2$ 的列名作为负例列名. 负例列名对应的子词平均表示作为负例列名表示.

在图 5 中, 本文以支撑单元格 946 为例, 并选择干扰单元格 164 作为负例. 对于支撑单元格 946 其第 1、2、3 层正例列名对应的负例列名分别是“defined benefit pension plans” (第 1 层负例列名, 来自单元格 164), “Non-U.S.” (第 2 层负例列名, 来自单元格 164), “2004(a)” (第 3 层负例列名, 来自单元格 164). 此外, 支撑单元格 964 组合正例列名对应的负例包括组合负例列名“defined benefit pension plans Non-U.S. 2004(a)” (来自单元格 164) 和来自简单表格的负例列名“2004”.

3.3.3 细粒度单元格数值对比

单元格列名和行名决定了单元格是否为支撑单元格, 但单元格数值的改变基本不会影响其是否为支撑单元格. 尽管如此, 生成器生成的计算程序包含支撑单元格数值. 因此, 本文期望单元格数值对比可以直接帮助区分干扰单元格数值和支撑单元数值, 最终帮助 decoder 解码出正确的计算程序. 因此本文提出了细粒度单元格数值对比.

锚点表示: 本文以问题经过分词器后子词的平均表示作为问题的表示; 或者用 Q 后的特殊单词 [SEP] 表示作为问题的表示.

正例表示: 本文随机采样固定数量 $K1$ 的支撑单元格数值作为正例, 用支撑单元格数值表示作为正例表示.

负例表示: 随机采样固定数量 $K2$ 的干扰单元格数值作为负例, 干扰单元格数值表示作为负例表示.

3.3.4 细粒度单元格语义构成元素对比损失

对于细粒度单元格语义构成元素对比损失 $\mathcal{L}_{fg}$, 其由细粒度行名对比损失 $\mathcal{L}_{\text{row}}$, 细粒度列名对比损失 $\mathcal{L}_{\text{col}}$, 细

粒度单元格数值对比损失 $\mathcal{L}_{\text{value}}$ 这 3 部分组成, 具体见公式 (7):

$$\mathcal{L}_{fg}=\mathcal{L}_{\text{row}}+\mathcal{L}_{\text{col}}+\mathcal{L}_{\text{value}} \tag{7}$$

对任意 $\mathcal{L}_t\in\{\mathcal{L}_{\text{row}},\mathcal{L}_{\text{col}},\mathcal{L}_{\text{value}}\}$, 本文使用 InfoNCE 损失拉近问题表示与支撑单元格句子语义构成元素表示之间的距离, 同时拉远问题表示与干扰单元格句子语义构成元素表示之间的距离, 具体见公式 (8):

$$\mathcal{L}_t=-\sum_{u^{\dagger}\in U_t^{\dagger}}\log\left(\frac{\mathrm{e}^{\text{sim}(u^{\dagger},q)/\tau}}{\sum_{u\in(U_t'\vee u^{\dagger})}\mathrm{e}^{\text{sim}(u,q)/\tau}}\right) \tag{8}$$

其中, $U_t^{\dagger}$ 为行名 (列名或者单元格数值) 对比正例表示的集合, U_t' 是行名 (列名或者单元格数值) 对比负例表示的集合, 其中 τ 是温度系数, $\text{sim}(\cdot,\cdot)$ 与公式 (5) 类似.

3.4 基于多粒度单元格语义对比的生成器训练和推理

在训练阶段, 生成器同时进行计算程序解码训练和多粒度单元格语义对比训练, 其训练损失如公式 (9) 所示:

$$\mathcal{L}=\lambda\times\mathcal{L}_g+(1-\lambda)\times(\mathcal{L}_{cg}+\mathcal{L}_{fg}) \tag{9}$$

其中, $\mathcal{L}_g,\mathcal{L}_{cg},\mathcal{L}_{fg}$ 分别表示计算程序解码损失、粗粒度单元格语义对比损失、细粒度单元格语义构成元素对比损失; λ 是权重, 控制解码损失和多粒度单元格语义对比损失的重要程度. 由于多粒度单元格语义对比仅在训练阶段优化生成器参数, 在训练完成后, 其不参与推理过程. 因此在推理阶段, 生成器仅需自回归地生成计算程序, 多粒度单元格语义对比方法不会带来额外的推理开销.

4 实 验

在实验部分, 本文采用文本和表格的数值问答数据集 FinQA[5]和 MultiHiertt[8], 分别基于 FinQANet 和 MT2Net 的生成器来验证多粒度单元格语义对比 (MGCC) 方法的有效性.

4.1 数据集介绍

FinQA 数据集包含 8 281 个问题, 每个问题对应的文档含有一个简单表格和文本. 如图 1 所示, 这个数据集不仅标注了问题, 计算程序 (Program) 及答案, 还标注了支撑单元格 (图 1 中下划线标注的单元格) 和文本中的支撑事实 (图 1 中下划线标记的文本, 支撑事实是回答问题所必需的文本). 该数据集所有问题都要求模型生成计算程序, 并根据此程序返回答案. 数据集被划分为 train, valid, test 集, 其占比分别是 75%, 10%, 15%, 具体大小见表 1.

表 1 FinQA 和 MultiHiertt 数据集 train, valid, test 分布

数据集	train	valid	test
FinQA	6 251	883	1 147
MultiHiertt	7 830	1 044	1 566

MultiHiertt 数据集有 10 440 个问题, 每个问题对应的文档包含多个层次化表格和文本, 具体的样本分布信息见表 1. 与 FinQA 数据集不同, 这个数据集的问题答案包括由计算程序返回答案和答案抽取两种类型, 其中答案抽取指的是答案为某个单元格数值或连续的文本片段. 在 MultiHiertt 数据集中, 计算程序问题类型占整个数据集 80% 以上的比例. 该数据集同样标注支撑单元格, 支撑事实和计算程序或者答案片段, 图 5 展示了 MultiHiertt 数据集一个示例, 这个示例其答案由计算程序返回. 在这个示例中, 下划线标注了支撑单元格, 方框标记了干扰单元格. 这里的计算程序通过引入第 i 步推理步结果*#i*, 使其包含推理结构 (提供了每一步推理逻辑). 这种方式相比单纯的计算表达式, 更接近人类的计算顺序, 因为传统的计算表达式操作符的优先级可能与人类的计算顺序不一致.

因为本文的方法是针对表格数据的, 进一步, 本文探究 FinQA 和 MultiHiertt 中问题支撑事实类型分布. FinQA 和 MultiHiertt 数据集中问题的支撑事实包括仅依赖文本, 仅依赖表格, 以及同时依赖文本和表格这 3 种主要类型. 表 2 中 Table (=1) 表示问题仅依赖单个表格, Table (>1) 表示问题仅依赖多个表格, Text and Table (=1) 表示支撑事实包括文本和一个表格, Text and Table (>1) 表示支撑事实包括文本和多个表格. 从表 2 的分布可以看出

FinQA 数据集中 76% 的问题需要表格数据作为其支撑事实. 对于 MultiHiertt 数据集, 高达 89.76% 的问题需要表格信息来生成计算程序. 更值得注意的是, MultiHiertt 数据集有 31.13% 问题的支撑事实包括多个表格, 其涉及更为复杂的跨表格推理. 上述分析表明, 这两个数据集能够有效地证明本文提出方法的有效性.

表 2 FinQA 和 MultiHiertt 数据集支撑事实类型分布 (%)

数据集	Text	Table (=1)	Table (>1)	Text and Table (=1)	Text and Table (>1)
FinQA	23.42	62.43	0.00	14.15	0.00
MultiHiertt	10.24	33.09	7.93	25.54	23.20

4.2 基准模型

在基准模型中, 除非特别指明, 检索器一般使用 BERT-base (110M 参数)[28]为基础的分类器, 判断来自于非结构化文本和表格文本的句子是否为支撑事实. FinQA 数据集默认使用 FinQANet 的 BERT-base 的分类器, 以行单元格句子和文本句子为检索单位. 然后使用分类器返回的 top 3 的检索句子作为生成器的输入, top 3 检索句子的召回率是 89.66%[5]. MultiHiertt 数据集默认采用 MT2Net 的 BERT-base 分类器作为检索器, 其以单元格句子和文本句子为检索单位, 使用分类器返回的 top 10 的检索句子作为生成器的输入, top 10 的检索句子的召回率是 80.8%[8]. 下面本文分别介绍采用的基准模型, 按基准模型的生成器类型大致可以分为 3 种类型.

(1) 不使用检索器, 文档直接作为生成器输入

Longformer[29]模型通过修改 Transformer[30]的 attention 机制, 可以处理输入长度高达 32k 字符, 因此不需要检索器检索相关事实, 所有非结构化文本和表格文本直接输入模型生成计算程序.

(2) 使用检索器, 生成器直接生成答案

Retriever+Direct Generation[5]采用主流的问答模型架构, 唯一差别在于生成器仅生成答案, 而不解码计算程序.

Retriever+TAPAS[8]由检索器和生成器构成, 其中在生成器部分使用表格预训练模型直接生成答案.

(3) 使用检索器, 生成器生成计算程序或者计算表达式

TF-IDF+Single Op[5]是一个简单的启发式方法, 先用 TF-IDF 检索 top 2 的句子. 然后获得每个句子中第 1 个数值, 并将这两个数值相除作为问题答案.

Retriever+Seq2Seq[5]采用检索器和生成器的架构. 相比较于 FinQANet, 其差别在于生成器的 encoder 是双向 LSTM, decoder 为没有额外 attention 机制的 LSTM.

Retriever+NeRd[31]先使用检索器检索相关事实, 其基于指针网络的神经符号生成器生成一个嵌套格式计算表达式.

Retriever+TAGOP[32]其通过序列标注识别计算表达式操作数, 通过分类识别操作符, 只能处理操作符简单的问题.

Retriever+NumNet[33]提出一个额外的模块建模数值之间的关系, 同时通过分类判别每个数值的符号, 这种方式只能处理操作符简单的问题.

FinQANet[5]由一个检索器和生成器构成. 其生成器的 decoder 包含一个特殊设计的 attention 机制, 与第 2.2.3 节介绍生成器一致.

FinQANet(FinBERT)[34]与 FinQANet 的差别是生成器的 encoder 换为 FinBERT[35], 这是一个基于金融领域语料的预训练语言模型.

DyRRen[7]采用检索器和生成器的架构, 其首先提出了一个增强的检索器, 在生成器中提出一种重排机制, 使生成器每一步动态地对检索排序, 结合事实排序的结果生成计算程序.

MT2Net[8]这个模型主要是针对 MultiHiertt 数据集设计的, 其包含检索器和生成器两个部分. 因为 MultiHiertt 有 2 种答案类型, 因此在生成器部分有 2 个子模块, 即计算程序生成子模块, 答案抽取子模块. 对于计算程序子模块, 其结构与第 2.2.3 节介绍生成器一致. 对于答案抽取模块, 其采用 T5 分别生成答案的开始和结束位置来获得连续答案片段. 此外, 生成器还嵌入了一个答案类型预测层, 该层的功能是判定哪一子模块应负责输出答案.

Llama 2-13B[36]是一个具有 130 亿参数的开源自回归大语言模型 (large language model, LLM), 它在众多任务中展示出了强劲的性能. 研究发现, 通过在测试问题之前附加相关的问题示例以及相应回答问题的推理链 (chain of thought, CoT)[35], 可以指导模型构建测试问题的推理链来生成答案, 进一步提升大语言模型的表现.

GPT-3.5-Turbo[37]是一个闭源的商业自回归大语言模型 (LLM), 其在很多任务上展现了强大的性能, 其也可以通过 CoT 进一步提升其性能.

Human Expert/General Crowd Performance[5,8]显示了专家/非专家在测试上的表现.

4.3 评价指标

对于 FinQA 数据集, 本文通过评价生成的计算程序正确率 (program accuracy) 和答案正确率 (answer accuracy) 来衡量问答模型的性能. 因为 MultiHiertt 数据集的部分问题不涉及计算程序, 这里参考相关工作[8], 用答案正确率 (answer accuracy) 和答案单词的 $F1$ 重合度指标来评价模型性能.

需要特别说明, 由于数据集 MultiHiertt 的测试集不公开答案, 需要通过在线榜单获取模型在测试集上性能结果. 因此在 MultiHiertt 数据集, 本文报告的是两种类型问题综合结果. 同时需要强调的是, 在 MutliHiertt 数据集中, MGCC 仅用于提升生成器计算程序生成子模块, 没有作用于答案抽取子模块. 对于答案抽取部分的问题, 本文采用 MT2Net 提供的答案抽取子模块来生成答案. 最终, 本文将 MGCC 提升的计算程序生成子模块的结果与 MT2Net 的答案抽取子模块结果结合 (MT2Net 介绍见第 4.2 节), 作为测试集的答案, 并在线提交到测评网站, 获得在线榜单成绩 (https://codalab.lisn.upsaclay.fr/competitions/6738).

4.4 实验设置

本文通过 FinQA 和 MultiHiertt 两个数据集来验证多粒度单元格语义对比 (MGCC) 方法的有效性. 在这两个数据集上, 本文通过验证集选取模型参数, 并在测试集上报告结果. 在主试验和消融实验中, 本文采用 5 个随机种子 (分别为 0、21、42、777 和 9999) 进行实验, 并取这些实验结果的平均值作为模型最终结果, 同时报告 5 次实验结果的方差. 对于分析实验, 考虑到计算资源的限制, 本文报告随机种子 42 时的模型结果.

对于 FinQA 数据集, 本文基于 FinQANet 来验证 MGCC 在生成器上的有效性. 具体而言, 检索器保持与 FinQANet 一致, 检索器返回的 top 3 检索句子的召回率是 89.66%[5]. 然后本文使用生成器 encoder 分别为 BERT-base (110M 参数)[28]的 FinQANet(BERT-base) 和 RoBERTa-large (355M 参数)[38]的 FinQANet(RoBERTa-large) 来验证 MGCC 有效性. 对于 FinQA 数据集, 问题表示和单元格句子整体语义表示采用 [SEP] 的表示.

对于 MultiHiertt 数据集, 本文采用 MT2Net 生成器的计算程序生成子模块验证 MGCC 有效性. 具体地, 检索器保持与 MT2Net 一致, 检索器返回的 top 10 的检索句子的召回率是 80.8%[8]. 对于计算程序生成子模块, 采用 encoder 为 RoBETa-large 的计算程序生成子模块验证 MGCC 有效性. 对于 MultiHiertt 数据集, 问题表示和单元格句子整体语义表示采用对应的经过分词器后所有子词的平均表示.

FinQANet(BERT-base)+MGCC: 使用最大序列长度为 512 的 BERT-base (110M 参数) 作为生成器的 encoder, decoder 与 Chen 等人[5]提出 FinQANet 一致. 多粒度对比损失的温度系数 τ 为 0.8, 训练损失权重 λ 为 0.8. 优化器采用 AdamW[39], 其学习率是 2.0E–5, 同时采用线性预热方案在 10% 的数据上对学习率进行预热. 训练的 batch size 设定为 32, 一共训练 300 个 epochs. 在这个模型中, 本文随机采样的正例数目 $K1$ 和负例数目 $K2$ 为 10, 这里 $K1$ 和 $K2$ 代表最多采样个数.

FinQANet(RoBERTa-large)+MGC: 这里 RoBERTa-large (355M 参数) 被作为 FinQANet 生成器的 encoder, decoder 与 Chen 等人[5]提出 FinQANet 相同. 多粒度对比损失的温度系数 τ 为 0.6, 训练损失权重 λ 为 0.8. AdamW 被用于优化模型参数, 学习率是 1.5E–5, 也采用线性预热方案在 10% 的数据上进行学习率预热. 训练的 batch size 为 32, 训练的 epochs 是 300. 这个模型下, 随机采样的正例数目 $K1$ 和负例数目 $K2$ 为 30, 这里 $K1$ 和 $K2$ 代表最多采样个数.

MT2Net(RoBERTa-large)+MGCC: 使用 RoBERTa-large 作为 MT2Net 生成器的计算程序生成子模块的 encoder, decoder 与 Chen 等人[5]提出 FinQANet 一致. 多粒度对比损失的温度系数 τ 为 0.8, 训练损失权重 λ 为 1. AdamW 被

用于优化模型参数, 学习率是 2.0E–5, 采用线性预热方案在前 100 步数据上进行学习率预热. 训练的 batch size 为 16, 训练的 epochs 是 60. 本文随机采样的正例数目 $K1$ 为 20, 负例数目 $K2$ 为 10, 这里 $K1$ 和 $K2$ 代表最多采样个数.

4.5 主要实验结果

本节主要展示基准模型与引入多粒度单元格语义对比 (MGCC) 的模型在 FinQA 和 MultiHiertt 数据集上性能表现.

表 3 展示了在 FinQA 数据集上, 基准模型和 FinQANet+MGCC 的对比结果. 这里报告的所有比较基准结果, 除了 DyRRen 结果来自于 Li 等人[7], 大模型结果来自 Srivastava 等人[40], 其他的结果来自于 Chen 等人[5].

表 3 问答模型在 FinQA test 集上结果 (%)

Model	Answer accuracy	Program accuracy
TF-IDF+Single Op	1.01	0.90
Retriever+Direct Generation	0.30	—
Retriever+Seq2Seq	19.71	18.38
LongFormer(base)	21.90	20.48
Retriever+NeRd(BERT-base)	48.57	46.76
FinQANet(FinBERT)	50.10	47.52
DyRRen(BERT-base)	59.37	57.54
DyRRen(RoBERTa-large)	63.30	61.29
Llama 2-13B	1.80	—
Llama 2-13B +CoT	25.34	—
GPT-3.5-Turbo	40.47	—
GPT-3.5-Turbo+CoT	59.18	—
FinQANet(BERT-base)	50.00	48.00
FinQANet(BERT-base)+MGCC	51.28±0.4	49.43±0.4
FinQANet(RoBERTa-large)	61.24	58.86
FinQANet(RoBERTa-large)+MGCC	**64.62±0.1**	**62.47±0.2**
Human Expert Performance	91.16	87.49
General Crowd Performance	50.68	48.17

本文从表 3 可以得出以下结论.

(1) 验证 MGCC 的 FinQANet 在性能上优于大多数基准模型. IF-IDF+Single Op 性能较弱, 这表明纯粹依赖启发式规则难以解决复杂的数值推理问题. 通过对比 FinQANet 和 Retriever+Direct Generation, 说明使用计算程序作为监督信号在生成器的训练中起着关键作用. 这是因为计算程序给生成器提供了更为明确的学习信号, 减少了学习空间和复杂度. 将 FinQANet(BERT-base) 和 Retriever+Seq2Seq 比较表明, 使用知识丰富的预训练语言模型来编码问题对回答数值推理问题是有用的. 通过对比 FinQANet(BERT-base) 和 Retriever+NeRd(BERT-base), 说明解码包含推理结构的计算程序是有用的, 这让模型在小规模的数据集上可以快速学习到如何解决问题. 比较 FinQANet (BERT-base) 和 LongFormer(base), 尽管两者有大致的参数规模, 但 FinQANet 在答案正确率上的表现超过了 LongFormer(base) 达到 29.38%. 这强调了在数据存在大量噪音时, 应该优先使用检索器筛选出相关的事实, 再由一个生成器生成计算程序. FinQANet(FinBERT) 表现较差, 可能是因为 FinBERT 预训练的语料较少.

(2) 对于不同参数规模的 FinQANet, 本文都能验证多粒度单元格语义对比方法 (MGCC) 有效性. 特别地, FinQANet(RoBERTa-large)+MGCC 相比 FinQANet(RoBERTa-large), 在答案正确率上获得了 3.38% 的大幅提升. 同时相比 FinQANet(BERT-base), FinQANet(BERT-base)+MGCC 在答案正确率上获得了 1.28% 提升. 因此在两种参数规模的 FinQANet 上都验证了 MGCC 有效性. 至于在更大的参数规模上, MGCC 表现更好. 可能是因为模型参数规模的增加, FinQANet 建模表格和文本能力得到增强, 从而更有效地利用 MGCC 区分单元格语义的能力.

(3) FinQANet(RoBERT-large)+MGCC 超过了所有比较的基准模型. 比较于 FinQANet, DyRRen 在检索器和生成器上都做了进一步的优化, 但是 FinQANet(RoBERTa-large)+MGCC 表现超过 DyRRen(RoBERTa-large). 说明在

RoBERTa-large 作为生成器 encoder 时, 相比 DyRRen 生成器中设计的重排机制, 通过 MGCC 区分单元格句子语义来帮助生成器选择支撑单元格更有效.

(4) 在 FinQA 数据集上, 与结合 CoT 的大型语言模型 GPT-3.5-Turbo 和 Llama 2-13B 相比, 基于 RoBERTa-large 的 FinQANet 结合 MGCC 方法分别获得 5.44% 和 39.28% 答案正确率提升. 这表明对于大型语言模型来说, 结合文本和表格的数值问答任务对于大语言模型也是有挑战性的. 通过对预训练语言模型进行微调, 不仅可以超越大语言模型的性能, 还能在实际应用中降低推理成本.

表 4 展示了基准模型和 MT2Net+MGCC 在 MultiHiertt 测试集上的结果, 这些基准模型结果来自 Zhao 等人[8].

表 4 问答模型在 MultiHiertt test 集上结果 (%)

Model	Answer accuracy	*F*1
Longformer	2.86	6.23
Retriever+TAPAS	7.67	10.04
Retriever+NumNet	10.77	12.02
Retriever+TAGOP(RoBERTa-large)	17.81	19.35
Fetriever+Seq2Seq	24.58	26.30
FinQANet(RoBERTa-large)	31.72	33.60
Llama 2-13B	1.54	—
Llama 2-13B+CoT	30.66	—
GPT-3.5-Turbo	25.88	—
GPT-3.5-Turbo+CoT	42.33	—
MT2Net(RoBERTa-large)	36.22	38.43
MT2Net(RoBERTa-large)+MGCC	**44.02±0.4**	**44.41±0.4**
Human Expert Performance	83.12	87.03

从表 4 本文可以得到下列结论.

(1) 在 MultiHiertt 数据集上, 验证 MGCC 的 MT2Net(RoBERTa-large) 在所有对比的非大语言基准模型中取得了领先性能. 相比于仅生成答案的 Retriever+TAPAS 和直接输入文档的 Longformer, MT2Net(RoBERTa-large) 领先幅度是巨大的. MT2Net(RoBERTa-large) 在两种答案评价指标上都超过 FinQANet(RoBERTa-large) 模型, 这主要得益于 MT2Net(RoBERTa-large) 在生成器中设计了一个额外答案抽取子模块, 专门处理答案抽取问题类型. 对比于处理简单表达式类型的 NumNet 和 TAGOP, MT2Net(RoBERTa-large) 也获得了大幅性能领先, 这说明 MT2Net (RoBERTa-large) 设计的生成器更为有效.

(2) 比较于 MT2Net(RoBERTa-large), MT2Net(RoBERTa-large)+MGCC 获得了 7.8% 答案正确率的显著提升. 相比 FinQA 数据集, MGCC 在 MultiHiertt 数据集上获得了更大的提升. 本文认为, 这种提升的原因主要与 MultiHiertt 数据集的特殊性有关. MultiHiertt 数据集中的问题需要层次化表格来回答, 这些转化后的支撑单元格句子和干扰单元格句子是更加难以区分的. 这主要是由于多个简短行名拼接是难以学习的; 干扰单元格多个层次化行名或者列名更容易和问题中词汇重合, 使生成器难以区分它们; 以及问题中支撑单元格句子和干扰单元格句子的层次化行名 (列名) 的重复, 这种问题占比大约是 15%. 因此, 在 MT2Net 中, 区分支撑单元格句子和干扰单元格句子成为一个重大的挑战. MGCC 的应用, 在这一挑战上带来了更大的收益.

(3) 对比于大语言模型, MT2Net(RoBERTa-large)+MGCC 也展现了性能优势. 在答案正确率上, 其超过了 GPT-3.5-Turbo+CoT 和 Llama 2-13B+CoT, 分别提升了 1.69% 和 13.36%.

4.6 消融实验

表 5 展示了不同参数规模的 FinQANet 结合不同对比类型, 在 FinQA 测试集上的结果. 这里 SemContrast 表示粗粒度单元格语义对比, RowContrast 表示细粒度单元格行名对比, ColContrast 表示细粒度单元格列名对比, ValContrast 表示细粒度单元格数值对比. 总体而言, 从表 5 中可以归纳出粗粒度单元格语义对比和 3 种细粒度单元格语义构成元素对比都是有效的.

表 5 MGCC 在 FinQA test 集上消融实验 (%)

Model	Answer accuracy	Program accuracy
FinQANet(BERT-base)	50.00	48.00
FinQANet(BERT-base)+MGCC	**51.28±0.4**	**49.43±0.4**
FinQANet(BERT-base)+SemContrast	50.11±0.6	48.37±0.3
FinQANet(BERT-base)+RowContrast	50.24±0.4	48.3±0.3
FinQANet(BERT-base)+ColContrast	50.75±1	48.96±1.2
FinQANet(BERT-base)+ValContrast	50.32±0.6	48.34±0.5
FinQANet(RoBERTa-large)	61.24	58.86
FinQANet(RoBERTa-large)+MGCC	**64.62±0.1**	**62.47±0.2**
FinQANet(RoBERTa-large)+SemContrast	63.86±0.6	61.67±0.7
FinQANet(RoBERTa-large)+RowContrast	63.41±0.3	61.32±0.2
FinQANet(RoBERTa-large)+ColContrast	63.59±0.1	61.26±0.1
FinQANet(RoBERTa-large)+ValContrast	64.34±0.3	62.07±0.3

(1) 不同参数规模的 FinQANet 结合多粒度单元格语义对比 (MGCC) 都比仅结合单一类型对比要带来更多的性能提升. FinQANet +MGCC 在生成器 encoder 为 BERT-base 时, 多粒度单元格语义对比带来的性能提升好于单一类型的细粒度对比, 两者带来的提升在答案正确率上最高相差 1.17%. 当生成器 encoder 是 RoBERTa-large 时, 比较于单一类型对比, 多粒度单元格语义对比也会带来更多的提升, 最高可以带给 FinQANet 模型 1.21% 答案正确率的提升.

(2) 在不同参数规模的 FinQANet 下, 粗粒度单元格语义对比大部分时候带来的性能提升要好于细粒度单元格语义构成元素对比. 这和本文动机是一致的, 单元格语义决定着模型是否选择单元格数值作为计算程序一部分, 更好的区分干扰单元格和支撑单元格整体语义可以提升模型性能. 同时, 本文观察到 FinQANet(BERT-base) 结合粗粒度单元格语义对比与 3 种细粒度单元格语义构成元素对比时, 模型性能提升幅度较小. 这一现象可能归因到能力较弱的编码器 (encoder), 其学习区分正负样本的能力存在限制, 需要设计更为精细化的正负样本选择机制.

(3) 在不同参数规模的 FinQANet 下, 细粒度单元格语义构成元素对比都可以带来模型性能提升, 但带来最大提升的细粒对比类型不同. 在生成器 encoder 为 BERT-base 时, 带来最大提升的是细粒度单元格列名对比; 在生成器 encoder 为 RoBERTa-large 时, 带来的最大提升是细粒度单元格数值对比.

表 6 展示了 MT2Net(RoBERTa-large) 结合不同对比类型在 MultiHiertt 测试集结果. 相关的标记和表 5 类似.

表 6 MGCC 在 MultiHiertt test 集上消融实验 (%)

Model	Answer accuracy	*F*1
MT2Net(RoBERTa-large)	36.22	38.43
MT2Net(RoBERTa-large)+MGCC	44.02±0.4	44.41±0.4
MT2Net(RoBERTa-large)+SemContrast	43.70±0.1	44.09±0.1
MT2Net(RoBERTa-large)+RowContrast	43.56±0.2	43.92±0.2
MT2Net(RoBERTa-large)+ColContrast	**44.09±0.1**	**44.49±0.1**
MT2Net(RoBERTa-large)+ValContrast	43.70±0.5	44.09±0.5

通过表 6, 可以看到粗粒度单元格语义对比和 3 种细粒度单元格语义构成元素对比都是有效的. 具体地, 可以看到:

(1) 大部分情况下, 多粒度单元格语义对比 (MGCC) 都好于细粒度单元格语义构成元素对比. 特别地, 细粒度单元格列对比优于 MGCC. 本文认为, 这是由于 MultiHiertt 数据集中表格的行名层次最多有 3 层, 但列名层次最多有 7 层, 因此如何区分复杂列名变得更为关键, 这同时也说明了细粒度单元格语义构成元素对比的必要性.

(2) 在大多数情况下, 粗粒度单元格语义对比好于细粒度单元格语义构成元素对比. 此外, 值得注意的是, 不同于 FinQA 数据集, 在这里单元格数值对比并没有超越粗粒度单元格语义对比. 本文认为, 这主要是因为层次化表格中单元格句子的语义较为复杂, 通过整体单元格语义进行区分效果更好.

结合表 5 和表 6, 本文可以得到下列结论: (1) 在两个数据集上, MGCC、粗粒度单元格语义对比及细粒度单元格语义构成元素对比都是有效的; (2) MGCC 优于粗粒度单元格语义对比和细粒度单元格语义构成元素对比; (3) 在大多数情况下, 粗粒度单元格语义对比优于细粒度单元格语义构成元素对比; (4) 随着表格类型差异, 细粒度单元格语义构成元素对比效果存在变化. 更进一步, 粗粒度单元格语义对比和细粒度单元格语义构成元素对比可以看成在对比学习框架中使用不同类型的正负样本, 而这些不同类型正负样本提升 FinQANet 性能是有差异的, 因此在对比学习框架, 选择合适的正负样本是重要的.

本文旨在通过消融实验验证粗粒度单元格语义对比和细粒度单元格语义构成元素对比的有效性. 其他组合类型的消融实验都是基于已验证的对比学习方法, 因此其有效性在一定程度上得到保障, 同时考虑到计算资源的限制, 本文将不进行额外的验证.

4.7 多粒度单元格语义对比有效性分析

4.7.1 多粒度单元格语义对比下的 FinQANet 生成器预测结果错误分析

为了进一步探究多粒度表格单元格语义对比 (MGCC) 方法的有效性, 本研究从问题的支撑事实类型出发, 具体由操作符错误 (operator errors) 和操作数错误 (number errors) 两个方面进一步分析 MGCC 带来的提升. 在表 7 中, “操作符错误”指预测的计算程序中操作符集合 (这里期望通过操作符集合是否一致近似判断操作符错误) 和标注计算程序中的操作符集合不一致, 而“操作数错误”是预测的计算程序数值集合和标注的计算程序数值集合不一致. 表 7 中括号外数值表示在当前这种划分下的模型预测结果错误比例, 而括号内的数值是 FinQANet(RoBERTa-large)+MGCC 相比 FinQANet(RoBERTa-large) 减少的错误比例. 根据表 7 的数据, 本文可以得到下列结论.

表 7 使用 MGCC 下 FinQANet(RoBERTa-large) 生成器在测试集预测结果错误占比 (%)

错误类型	Text	Table	Text and Table	Error sum
Operator errors	2.87 (−0.01)	4.70 (−1.49)	12.11 (−2.48)	19.68 (−3.97)
Number errors	5.14 (−0.00)	10.20 (−1.74)	21.79 (−3.05)	37.13 (−4.79)

(1) MGCC 带来了更为显著的操作数错误比例下降. MGCC 带来的操作符错误比例下降了 3.97%, 而操作数错误比例下降了 4.79%. 这主要因为 MGCC 优化了表格单元格语义和构成计算程序的支撑单元格数值表示. 同时, 通过优化单元格语义, 降低了操作符学习的难度, 从而降低了操作符错误的占比.

(2) MGCC 带来的操作数错误比例下降主要集中在包含表格支撑事实类型问题上. 在支撑事实包含文本和表格时, MGCC 带来的操作数错误比例下降是 3.05%, 支撑事实是表格时, 其带来的错误占比下降是 1.74%, 但在支撑事实为文本时, MGCC 没有带来明显的下降. 支撑事实是文本和表格时, MGCC 带来更大的错误占比下降, 其原因可能为这部分错误占比大, 有较大的提升空间.

4.7.2 在 t-SNE 下多粒度单元格语义对比的锚点, 正例及负例可视化

为了探究多粒度单元格对比 (MGCC) 对于生成器区分支撑与干扰单元格句子的影响, 本研究通过 t-SNE 技术对 FinQANet(RoBERTa-large) 在有无 MGCC 情形下的锚点、正负例表示进行了空间可视化. 分析包括了 MGCC 中的 4 种对比类型 (粗粒度单元格语义 (a2)、行名 (b2)、列名 (c2)、数值名 (d2) 对比) 下的表示聚类情况. 其中, 蓝色点代表锚点表示, 绿色点代表正例表示, 红色点代表负例表示, (a1), (b1), (c1), (d1) 表示无 MGCC 下相同子词表示. 结果显示, 使用 MGCC 时, 锚点、正例和负例表示在空间上表现出更明显的聚类, 验证了 MGCC 能有效提升模型区分能力.

(1) 使用 MGCC 可以很好区分开支撑单元格句子和干扰单元格句子表示. 在没有使用 MGCC 下, 从图 6(a1), (b1), (d1), 粗粒度单元格语义对比, 行对比, 列对比正负例表示混合在一起, 模型难以区分它们. 在使用 MGCC 后, 从图 6(a2), (b2), (d2) 观察, 模型可以很好地区分 MGCC 的粗粒度单元格语义对比、细粒度单元格行对比、细粒度单元格列对比的正负例表示. 即模型可以很好地区分支撑单元格句子语义和干扰单元格句子语义.

(2) 使用 MGCC 后, 问题表示和正例表示距离更近. 对比图 6(a1), (b1), (c1), 本文可以发现未使用 MGCC 下, 问题表示和正负例表示之间的距离远近是不定的. 在使用 MGCC 下, 问题表示和正例表示有更近的距离. 图 6(d1)

显示了在不使用 MGCC 情况下, 模型可以区分干扰单元格数值表示和支撑单元格数值表示. 这可能是因为生成器需要预测支撑单元格数值以用于计算程序生成, 因此其必须学习区分支撑单元格数值和干扰单元格数值表示. 尽管如此, MGCC 使支撑单元格数值表示的类聚地更紧.

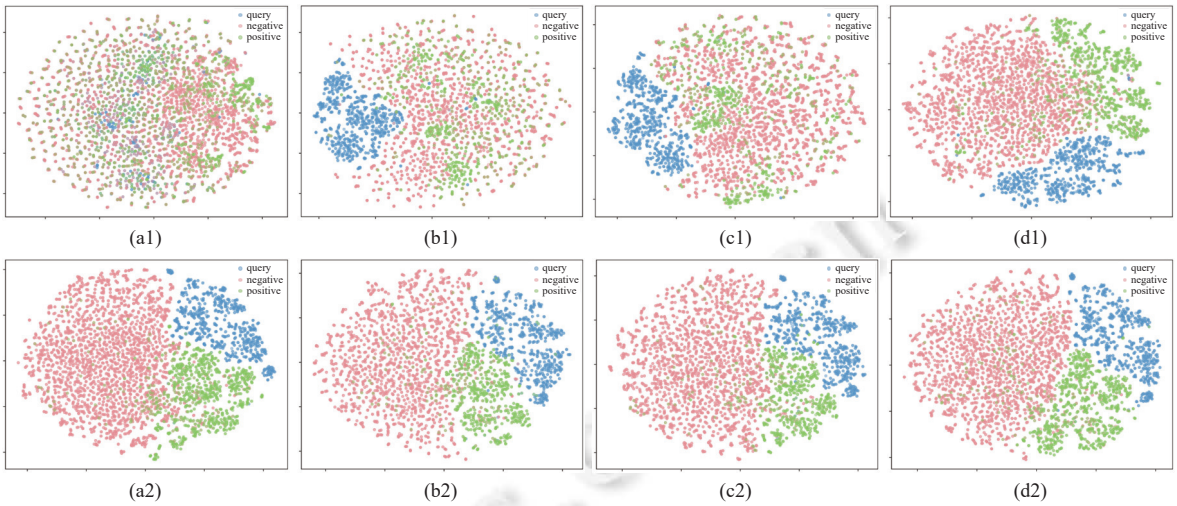


图 6 t-SNE 下多粒度单元格语义对比的锚点, 正例及负例可视化

4.7.3 多粒度单元格语义对比下计算程序生成过程展示

后文图 7 详细描述了在引入多粒度单元格语义对比 (MGCC) 机制前后, 问答模型预测计算程序的主要步骤, 即每步决策时的 top 3 候选操作符或操作数及对应的决策概率值 (精确到小数点后 5 位).

通过可视化概率决策过程, 本文可以做些简单的解释性分析: (1) 分析每步决策依据. 例如, 在第 1 步决策中, 可以清楚地观察到模型之所以选择“subtract”作为操作符, 而忽略其他选项, 这是因为模型给“subtract”分配了最高的概率值 (1.0), 而将“divide”和“greater”分别赋予了 0.0 的概率. (2) 分析错误原因. 在第 2 步决策时, FinQA(RoBERTa-large) 由于分配给干扰单元格 (55.2) 最高的概率 0.999, 而当前步的支撑单元格 (65.7) 仅分配了 0.000 98 概率, 因此导致了其决策错误. 分析这些错误决策的概率不仅有助于理解错误发生的原因, 也为模型后续优化提供了方向. (3) 比较 MGCC 机制的引入前后问答模型决策依据的变化. 引入了 MGCC 后, 在第 2 步中, FinQA(RoBERTa-large) +MGCC 正确地选择了支撑单元格 65.7, 对 FinQANet(RoBERTa-large) 易混淆的干扰单元格 55.2 分配了接近 0.0 的概率, 说明了 MGCC 可以帮助生成器有效区分支撑和干扰单元格句子.

4.8 多粒度单元格语义对比超参数探究

4.8.1 温度系数对 FinQANet 生成器性能探究

相关研究表明[41], 温度系数是对比学习的一个重要超参数, 它对模型的学习方式和性能有显著影响. 通常, 为了给负样本更多的惩罚, 对比学习的温度系数不会大于 1.0. 温度系数越小, 经过 Softmax 后概率分布会变得更陡峭, 使模型更加关注困难样本学习. 然而, 如果温度系数过小, 会导致对比学习仅关注少数困难样本, 忽略大多数样本正负例的区分. 这里本文探究了温度系数在{0.1, 0.2, 0.4, 0.6, 0.8, 1.0}这 6 种设置下, 不同参数规模的生成器在 FinQA 测试集上的效果. 这里 Ans Acc 表示答案正确率, Prog Acc 表示计算程序正确率.

从图 8 可以观察 FinQANet(BERT-base)+MGCC 和 FinQANet(RoBERTa-large)+MGCC 大致都有相似的趋势: FinQANet 生成器性能随温度系数增加而增加, 进而达到一个峰值, 这之后随着温度系数的增大下降. 例如, FinQANet(RoBERTa-large)+MGCC 在温度系数为 0.6 时, 达到最高的 65.21% 答案正确率. 最高点和最低点性能相差 2.00%. 对于 FinQANet(BERT-base)+MGCC 在温度系数为 0.4 时, 答案正确率达到最高, 为 52.66%. 其最高点和最低点相差 2.01%. 结合实验结果, 选择一个合适的温度系数对于 MGCC 是重要的. 正确的温度系数能够平衡模型对困难样本的关注和对大多数样本正负例的区分能力, 从而优化模型的整体性能.

Text:
Concentration of credit risk financial instruments that potentially subject us to concentrations of credit risk consist of cash and cash equivalents, …

	2010	2009	2008
Balance at beginning of year	$55.2	$65.7	$14.7
Additions charged to expense	$23.6	$27.3	$36.5
...	...	...	...
Balance at end of year	$50.9	$55.2	$65.7

In 2008, subsequent to the allied acquisition, we recorded a provision for doubtful accounts of $ 14.2 million to adjust the allowance…

Question: In 2008 what was the change in the the allowance for doubtful accounts?
Golden program: subtract (65.7, 14.7)
Answer: 51

Method	FinQANet(RoBERTa-large)	FinQANet(RoBERTa-large)+MGCC
Generated program	*subtract* (55.2, 14.7) ✗	*subtract* (65.7, 14.7) ✓
Candidate prob. Step 1	*subtract*: 1.0, *divide*: 0.0, *greater*: 0.0 ✓	*subtract*: 1.0, *divide*: 0.0, *greater*: 0.0 ✓
Candidate prob. Step 2	*55.2*: 0.999, *65.7*: 0.00098, *14.2*: 2E−5 ✗	*65.7*: 1.0, *14.7*: 0.0, *14.2*: 0.0 ✓
Candidate prob. Step 3	*14.7*: 0.99953, *65.7*: 0.00047, *14.2*: 0.0 ✓	*14.7*: 1.0, *65.7*: 0.0, 55.2: 0.0 ✓

图 7 计算程序生成过程展示

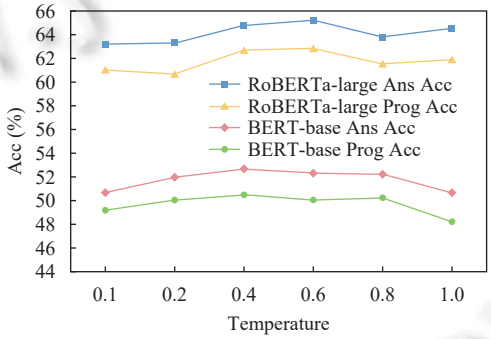


图 8 温度系数对 FinQANet 生成器性能影响

4.8.2 不同映射函数对 FinQANet 生成器性能探究

相关研究表明将原始锚点, 正负例向量映射到辅助空间进行对比损失学习是有益的[4,18]. 因此本文探究了 4 种映射类型: 两种非线性变换 (Sigmoid 和 ReLU)、线性变换, 以及不进行任何映射 (None). 为了优化计算资源消耗, 并结合第 4.8.1 节的模型性能随温度系数变化的讨论, 所有实验的温度系数设定为 1.0. 表 8 展示了在不同映射类型下, FinQANet 生成器的程序准确率 (program accuracy) 和答案准确率 (answer accuracy).

表 8 不同映射函数对 FinQANet 生成器性能影响 (%)

方法	Sigmoid	ReLU	Linear	None
FinQANet(BERT-base)+MGCC	(48.21, 50.65)	(47.86, 49.69)	(47.95, 49.69)	(48.91, 51.00)
FinQANet(RoBERTa-large)+MGCC	(61.90, 64.52)	(60.33, 62.69)	(62.30, 64.98)	(62.42, 64.34)

从表 8 可以得到下列结论: (1) 在两种参数规模下, 基于 Sigmoid 的非线性映射函数比 ReLU 表现更优. (2) 对于非线性和线性映射, FinQANet(BERT-base)+MGCC 在非线性映射函数上获得更好性能; FinQANet(RoBERT-

large)+MGCC 在线性映射函数上带来生成器性能好于非线性映射函数. (3) 不使用映射函数在两种参数规模上都使生成器性能达到最优. 然而, 结论 (2) 和 (3) 与 Chen 等人[17]工作有些出入. 可能的原因是, 在本文的实验中, MGCC 学习目标是区分干扰和支撑单元格句子语义, 而生成器也是要区分干扰和支撑单元格句子语义, 这两者学习目标有很大相似性. 而在 Chen 等人[17]工作中, 对比学习的目标和下游任务的目标差异较大, 非线性映射有助于剔除与对比损失无关的特征. 尽管不使用映射函数转化原始锚点, 正负例向量到辅助空间获得了最好性能, 但其优势相较于使用 Sigmoid 非线性映射函数并不显著. 因此本工作仍采用主流的非线性 Sigmoid 映射函数将原始锚点, 正负例映射到辅助空间进行对比损失学习.

5 总结与展望

本文探讨了基于文本和表格的数值推理任务, 这个任务需要模型基于异构文本和表格进行数值推理, 生成计算程序, 并以计算程序结果作为答案. 为了充分利用预训练语言模型强大的推理能力, 当前工作使用模板将表格线性化为表格文本, 但这会导致问答模型生成器难以区分支撑单元格句子和干扰单元格句子. 为了解决这个问题, 本文提出了一种多粒度单元格语义对比 (MGCC) 方法, 其旨在优化支撑和干扰单元格句子语义表示, 使生成器可以更有效区分他们. 实验结果验证了 MGCC 的有效性, 进一步的实验分析也验证了 MGCC 确实可以帮助生成器区分支撑和干扰单元格句子语义表示.

未来工作可以从以下 3 个方面进行深入探索: (1) 在消融实验中, 本文发现 MGCC 相比仅使用粗粒度单元格语义对比或者细粒度单元格语义构成元素对比, 没有大幅度领先. 因此, 如何更有效地结合粗粒度和细粒度对比机制是值得进一步研究的问题. (2) 表格单元格语义差异小是表格数据存在的天然特征, 探索如何将这种多粒度单元格语义对比扩展到预训练语言模型中, 使其能区分差异细微的单元格语义也是一个有前途方向. (3) 在消融实验中, 本文发现于对于能力较弱的编码器 (encoder), 其学习区分对比学习正负样本能力受到限制. 因此探索针对不同能力模型 (如能力较强或较弱的模型), 如何选择恰当的正负样本以优化学习效果和模型性能是一个值得研究方向.

References:

[1] You L, Zhou YQ, Huang XJ, Wu LD. A maximum entropy model based confidence scoring algorithm for QA. Ruan Jian Xue Bao/Journal of Software, 2005, 16(8): 1407–1414 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1407.htm [doi: 10.1360/jos161407]

[2] Qiao SJ, Yang GP, Yu Y, Han N, Qin X, Qu LL, Ran LQ, Li H. QA-KGNet: Language model-driven knowledge graph question-answering model. Ruan Jian Xue Bao/Journal of Software, 2023, 34(10): 4584–4600 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6882.htm [doi: 10.13328/j.cnki.jos.006882]

[3] Karpukhin V, Oguz B, Min S, Lewis P, Wu L, Edunov S, Chen DQ, Yih WT. Dense passage retrieval for open-domain question answering. In: Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, 2020. 6769–6781. [doi: 10.18653/v1/2020.emnlp-main.550]

[4] Caciularu A, Dagan I, Goldberger J, Cohan A. Long context question answering via supervised contrastive learning. In: Proc. of the 2022 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Seattle: Association for Computational Linguistics, 2022. 2872–2879. [doi: 10.18653/v1/2022.naacl-main.207]

[5] Chen ZY, Chen WH, Smiley C, Shah S, Borova I, Langdon D, Moussa R, Beane M, Huang TH, Routledge B, Wang WY. FinQA: A dataset of numerical reasoning over financial data. In: Proc. of the 2021 Conf. on Empirical Methods in Natural Language Processing. Online and Punta Cana: Association for Computational Linguistics, 2021. 3697–3711. [doi: 10.18653/v1/2021.emnlp-main.300]

[6] Wang B, Ju JZ, Mao YL, Dai XY, Huang SJ, Chen JJ. A numerical reasoning question answering system with fine-grained retriever and the ensemble of multiple generators for FinQA. arXiv:2206.08506, 2022.

[7] Li X, Zhu Y, Liu SC, Ju JZ, Qu YZ, Cheng G. DyRRen: A dynamic retriever-reranker-generator model for numerical reasoning over tabular and textual data. arXiv:2211.12668, 2022.

[8] Zhao YL, Li YX, Li CY, Zhang R. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In: Proc. of the 60th Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers). Dublin: Association for Computational Linguistics, 2022. 6588–6600. [doi: 10.18653/v1/2022.acl-long.454]

[9] Sun JS, Zhang H, Lin C, Su XD, Gong YY, Guo J. APOLLO: An optimized training approach for long-form numerical reasoning.

arXiv:2212.07249, 2024.

[10] Wang B, Ju JZ, Fan Y, Dai XY, Huang SJ, Chen JJ. Structure-unified M-tree coding solver for math word problem. In: Proc. of the 2022 Conf. on Empirical Methods in Natural Language Processing. Abu Dhabi: Association for Computational Linguistics, 2022. 8122–8132. [doi: 10.18653/v1/2022.emnlp-main.556]

[11] Jie ZM, Li JR, Lu W. Learning to reason deductively: Math word problem solving as complex relation extraction. In: Proc. of the 60th Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers). Dublin: Association for Computational Linguistics, 2022. 5944–5955. [doi: 10.18653/v1/2022.acl-long.410]

[12] Liang ZW, Zhang JP, Wang L, Qin W, Lan YS, Shao J, Zhang XL. MWP-BERT: Numeracy-augmented pre-training for math word problem solving. In: Proc. of the 2022 Findings of the Association for Computational Linguistics: NAACL 2022. Seattle: Association for Computational Linguistics, 2022. 997–1009. [doi: 10.18653/v1/2022.findings-naacl.74]

[13] Wang Y, Liu XJ, Shi SM. Deep neural solver for math word problems. In: Proc. of the 2017 Conf. on Empirical Methods in Natural Language Processing. Copenhagen: Association for Computational Linguistics, 2017. 845–854. [doi: 10.18653/v1/D17-1088]

[14] Amini A, Gabriel S, Lin P, Koncel-Kedziorski R, Choi Y, Hajishirzi H. MathQA: Towards interpretable math word problem solving with operation-based formalisms. arXiv:1905.13319, 2019.

[15] Koncel-Kedziorski R, Roy S, Amini A, Kushman N, Hajishirzi H. MAWPS: A math word problem repository. In: Proc. of the 2016 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego: Association for Computational Linguistics, 2016. 1152–1157. [doi: 10.18653/v1/N16-1136]

[16] Chen XL, Chiticariu L, Danilevsky M, Evfimievski A, Sen P. A rectangle mining method for understanding the semantics of financial tables. In: Proc. of the 14th IAPR Int'l Conf. on Document Analysis and Recognition (ICDAR). Kyoto: IEEE, 2017. 268–273. [doi: 10.1109/ICDAR.2017.52]

[17] Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. In: Proc. of the 37th Int'l Conf. on Machine Learning. JMLR.org, 2020. 1597–1607.

[18] He KM, Fan HQ, Wu YX, Xie SN, Girshick R. Momentum contrast for unsupervised visual representation learning. In: Proc. of the 2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR). Seattle: IEEE, 2020. 9726–9735. [doi: 10.1109/CVPR42600.2020.00975]

[19] Gao TY, Yao XC, Chen DQ. SimCSE: Simple contrastive learning of sentence embeddings. In: Proc. of the 2021 Conf. on Empirical Methods in Natural Language Processing. Punta Cana: Association for Computational Linguistics, 2021. 6894–6910. [doi: 10.18653/v1/2021.emnlp-main.552]

[20] Luo Y, Guo F, Liu ZH, Zhang Y. Mere contrastive learning for cross-domain sentiment analysis. In: Proc. of the 29th Int'l Conf. on Computational Linguistics. Gyeongju: Int'l Committee on Computational Linguistics, 2022. 7099–7111.

[21] Yue ZR, Kratzwald B, Feuerriegel S. Contrastive domain adaptation for question answering using limited text corpora. In: Proc. of the 2021 Conf. on Empirical Methods in Natural Language Processing. Punta Cana: Association for Computational Linguistics, 2021. 9575–9593. [doi: 10.18653/v1/2021.emnlp-main.754]

[22] You CY, Chen N, Zou YX. Self-supervised contrastive cross-modality representation learning for spoken question answering. In: Proc. of the 2021 Findings of the Association for Computational Linguistics: EMNLP 2021. Punta Cana: Association for Computational Linguistics, 2021. 28–39. [doi: 10.18653/v1/2021.findings-emnlp.3]

[23] Yang N, Wei FR, Jiao BX, Jiang DX, Yang LJ. xMoCo: Cross momentum contrastive learning for open-domain question answering. In: Proc. of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th Int'l Joint Conf. on Natural Language Processing (Vol. 1: Long Papers). Association for Computational Linguistics, 2021. 6120–6129. [doi: 10.18653/v1/2021.acl-long.477]

[24] van den Oord A, Li YZ, Vinyals O. Representation learning with contrastive predictive coding. arXiv:1807.03748, 2019.

[25] Chen YH, Jia YH, Tan CY, Chen WL, Zhang M. Method for complex question answering based on global and local features of knowledge graph. Ruan Jian Xue Bao/Journal of Software, 2023, 34(12): 5614–5628 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6799.htm [doi: 10.13328/j.cnki.jos.006799]

[26] Chen WH, Zha HW, Chen ZY, Xiong WH, Wang H, Wang W. HybridQA: A dataset of multi-hop question answering over tabular and textual data. arXiv:2004.07347, 2021.

[27] Li X, Sun YW, Cheng G. TSQA: Tabular scenario based question answering. In: Proc. of the 35th AAAI Conf. on Artificial Intelligence. AAAI, 13297–13305. [doi: 10.1609/aaai.v35i15.17570]

[28] Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional Transformers for language understanding. In: Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1: Long and Short Papers). Minneapolis: Association for Computational Linguistics, 2019. 4171–4186. [doi: 10.18653/v1/N19-1423]

[29] Beltagy I, Peters ME, Cohan A. Longformer: The long-document Transformer. arXiv:2004.05150, 2020.

[30] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 6000–6010.

[31] Chen XY, Liang C, Yu AW, Zhou D, Song D, Le QV. NEURAL symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In: Proc. of the 8th Int'l Conf. on Learning Representations. Addis Ababa, 2020.

[32] Zhu FB, Lei WQ, Huang YC, Wang C, Zhang S, Lv JC, Feng FL, Chua TS. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In: Proc. of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th Int'l Joint Conf. on Natural Language Processing (Vol. 1: Long Papers). Association for Computational Linguistics, 2021. 3277–3287. [doi: 10.18653/v1/2021.acl-long.254]

[33] Ran Q, Lin YK, Li P, Zhou J, Liu ZY. NumNet: Machine reading comprehension with numerical reasoning. In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int'l Joint Conf. on Natural Language Processing (EMNLP-IJCNLP). Hong Kong: Association for Computational Linguistics, 2019. 2474–2484. [doi: 10.18653/v1/D19-1251]

[34] Araci D. FinBERT: Financial sentiment analysis with pre-trained language models. arXiv:1908.10063, 2019.

[35] Wei J, Wang XZ, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E, Le Q, Zhou D. Chain-of-thought prompting elicits reasoning in large language models. arXiv:2201.11903, 2023.

[36] Touvron H, Martin L, Stone K, *et al*. Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288, 2023.

[37] Introducing ChatGPT. 2022. https://openai.com/blog/chatgpt

[38] Liu YH, Ott M, Goyal N, Du JF, Joshi M, Chen DQ, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692, 2019.

[39] Loshchilov I, Hutter F. Decoupled weight decay regularization. arXiv:1711.05101, 2019.

[40] Srivastava P, Malik M, Gupta V, Ganu T, Roth D. Evaluating LLMs' mathematical reasoning in financial document question answering. arXiv:2402.11194, 2024.

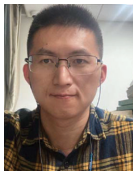
[41] Wang F, Liu HP. Understanding the behaviour of contrastive loss. In: Proc. of the 2021 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR). Nashville: IEEE, 2021. 2495–2504. [doi: 10.1109/CVPR46437.2021.00252]

附中文参考文献:

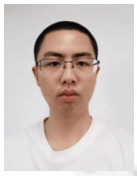
[1] 游斓, 周雅倩, 黄萱菁, 吴立德. 基于最大熵模型的 QA 系统置信度评分算法. 软件学报, 2005, 16(8): 1407–1414. http://www.jos.org.cn/1000-9825/16/1407.htm [doi: 10.1360/jos161407]

[2] 乔少杰, 杨国平, 于泳, 韩楠, 覃晓, 屈露露, 冉黎琼, 李贺. QA-KGNet: 一种语言模型驱动的知识图谱问答模型. 软件学报, 2023, 34(10): 4584–4600. http://www.jos.org.cn/1000-9825/6882.htm [doi: 10.13328/j.cnki.jos.006882]

[25] 陈跃鹤, 贾永辉, 谈川源, 陈文亮, 张民. 基于知识图谱全局和局部特征的复杂问答方法. 软件学报, 2023, 34(12): 5614–5628. http://www.jos.org.cn/1000-9825/6799.htm [doi: 10.13328/j.cnki.jos.006799]



琚江舟(1991－), 男, 博士生, 主要研究领域为问答系统.



陈宇飞(2000－), 男, 硕士生, 主要研究领域为问答系统.

毛云麟(1999－), 男, 硕士生, 主要研究领域为问答系统.

戴新宇(1979－), 男, 博士, 教授, 博士生导师, CCF 专业会员, 主要研究领域为自然语言处理, 人机对话交互, 人工智能应用.

吴震(1993－), 男, 博士, 助理教授, 主要研究领域为情感分析, 观点挖掘, 情感生成, 迁移学习.

陈家骏(1963－), 男, 博士, 教授, 博士生导师, CCF 专业会员, 主要研究领域为自然语言处理, 机器翻译, 程序设计语言.