Report for AFAC

Yidan Sun - Nankai University

Abstract

This document aims to demonstrate the basic approach to task completion, elucidate the principles of PiVe and MindMap, and provide an explanation of the original paper. Specific technical implementation details can be found in another submitted document.

1 Introduction

In recent years, the financial industry has witnessed a significant transformation driven by advancements in artificial intelligence (AI) and machine learning (ML). Among these developments, large language models (LLMs) have emerged as powerful tools capable of understanding and generating human-like text based on vast amounts of data. These models, epitomized by OpenAI's GPT-3 and its successors, have demonstrated remarkable proficiency in tasks ranging from natural language processing to data analysis and content creation. Leveraging these capabilities, this paper presents the development and implementation of an intelligent agent designed to automatically generate stock and industry research reports.

The aim of this study is to design and implement an intelligent agent that utilizes large language models to automatically generate stock and industry research reports. The specific objectives include significantly reducing the time required to produce comprehensive and accurate financial reports, thereby enhancing the efficiency of the report generation process. Additionally, the study seeks to minimize human biases in the analysis and presentation of financial data, ensuring the objectivity of the reports. Furthermore, the study aims to achieve scalability in the report generation process, enabling the coverage of a broader range of companies and industries without a proportional increase in resource allocation.

The contributions of this project are as follows:

1. Utilizing the PiVe model to enhance the accuracy of entity and relationship extraction in large language models.
2. Implementing the MindMap model to achieve retrieval-augmented generation using knowledge graphs.
3. Employing MetaGPT to develop an intelligent agent capable of automatically generating research reports.

2 Implementation Outline

Our intelligent report generation agent operates in two modes: individual stock report generation and industry report generation. For the report generation method, we provide several options: you can choose to enhance entity and relationship extraction using the PiVe algorithm, and you can also opt to use MindMap for Retrieval-Augmented Generation (RAG) based on knowledge graphs. The outline is shown in Figure 1

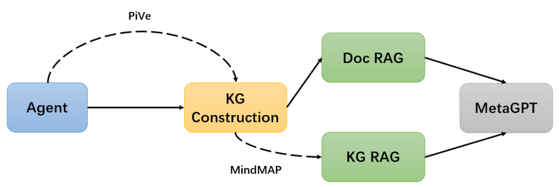


Figure 1: outline of Agent

In terms of dataset construction, we did not use news APIs but opted to obtain news from websites through web scraping (https://www.eastmoney.com/). Given the time-sensitive nature of stock market information, we only selected news from the past month. When constructing reference data for individual stock reports, we scraped news related to a single stock. For industry report reference data, we scraped news about a specific industry as well as news related to 10 leading stocks within that industry.

3 Task 1: Constructing Knowledge Graph

For constructing financial knowledge graphs, the key lies in the quality of entity and relationship extraction. The PiVe model (Han et al., 2023), coincidentally, focuses on enhancing the graph-based generation capabilities of LLMs. Specifically, PiVe involves the use of an external validation module (a smaller language model) to integrate feedback into prompts. PiVe iteratively improves prompts by leveraging the validation module and refining prompts through corrective instructions, significantly enhancing the quality of generated semantic graphs returned to the LLM.

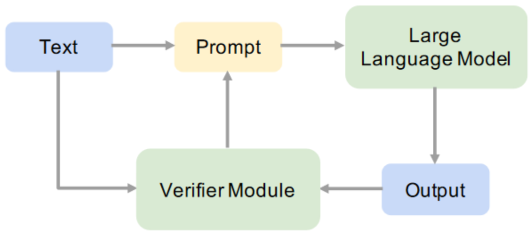


Figure 2: outline of PiVe

The PiVe model first employs an online large language model (such as ChatGPT 3.5) to directly extract entity-relation-entity triplets from sentences. Subsequently, the univerifier module is utilized to reason over the generated results and supplement missing triplets.

For verifier module, we use T5-Large ((Raffel et al., 2020)), and Flan-T5-XXL ((Chung et al., 2024)) as the backbone models for dataset-specific verifier module, and unified verifier module, respectively. T5 models follow the encoder-decoder architecture and treat all NLP tasks as unified text-to-text transduction tasks. Flan-T5 is instruction-fine-tuned version of T5 which was trained on 1,836 NLP tasks initialized from fine-tuned T5 checkpoint. For T5-large, we fine-tune all parameters for separate verifier modules per each dataset. While for Flan-T5-XXL, we use LoRA as a parameterefficient fine-tuning method, to train a unified verifier module which can follow the instruction.

4 Task 2: MindMap for KG-RAG

In the original document, a KG-RAG method was proposed in the references; however, through practical implementation, we found that this method did not perform well in real-world tasks. Therefore, we identified a new KG-RAG approach, namely, MindMap ((Wen et al., 2023)).

The goal of this work is to build a plug-and-play prompting approach to elicit the graph-of-thoughts reasoning capability in LLMs. We call our method MindMap because it enables LLMs to comprehend graphical inputs to build their own mind map that supports evidence-grounded generation. A conceptual demonstration of MindMap is in Figure 3. Specifically, MindMap sparks the graph of thoughts of LLMs that (1) consolidates the retrieved facts from KGs and the implicit knowledge from LLMs, (2) discovers new patterns in input KGs, and (3) reasons over the mind map to yield final outputs. We conducted experiments on three datasets to illustrate that MindMap outperforms a series of prompting approaches by a large margin. This work underscores how LLM can learn to conduct synergistic inference with KG. By integrating both implicit and explicit knowledge, LLMs can achieve transparent and dependable inference, adapting to different levels of correctness in additional KG information.

4.1 Evidence Graph Mining

We first use LLM to identify key entities from the question query Q and save them in the graph G

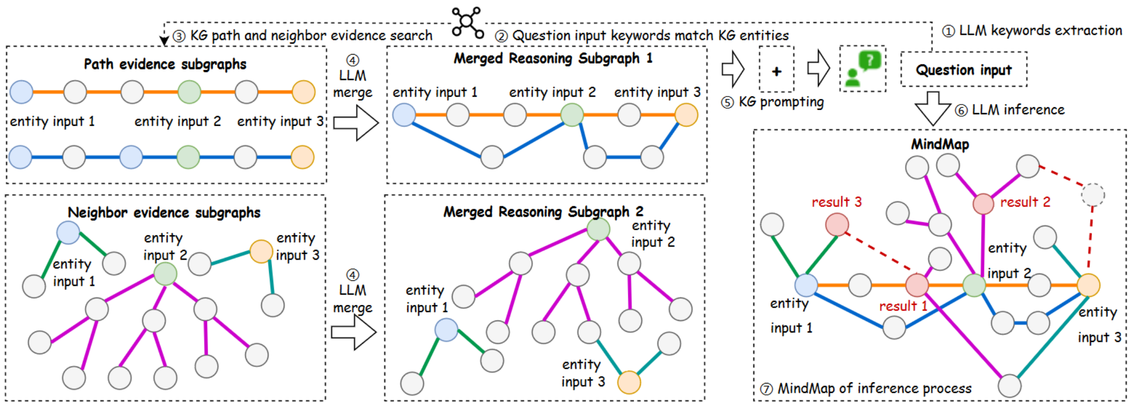
We use two approaches to build the evidence subgraph set G_q from the source knowledge graph:

1. Path-based exploration traces intermediary paths within G to connect important entities from the query. We form path segments by exploring connected nodes from a chosen node in V_0^q for at most k hops. The process continues until all segments are connected, creating a set of subgraphs stored in G^q_{path}.

2. Neighbor-based exploration adds related knowledge by expanding each node n in N_q by 1-hop to its neighbors, adding triples $\{(n, e, n')\}$ to G^q_{nei}.

4.2 Evidence Graph Aggregation

In this phase, LLM is instructed to consolidate various evidence sub-graphs G_q^* into a unified reasoning graph G_m. This reasoning graph G_m, upon completion, serves as external augmented graph input for Step III. We first extract at least k path-based and k neighbor-based evidence sub-graphs from the previous section, each representing potential connections between query entities. These



① LLM keywords extraction
② Question input keywords match KG entities
③ KG path and neighbor evidence search
④ LLM merge
⑤ KG prompting
⑥ LLM inference
⑦ MindMap of inference process
Path evidence subgraphs
Neighbor evidence subgraphs
Merged Reasoning Subgraph 1
Merged Reasoning Subgraph 2
Question input
MindMap
entity input 1
entity input 2
entity input 3
result 1
result 2
result 3

Figure 3: framework of MindMap

sub-graphs are then formatted into entity chains. This design offers two advantages: (a) It simplifies sub-graphs into a concise and consistent format, capturing essential information. (b) It leverages LLM's natural language understanding and generation capabilities to unify semantically similar entities and resolve potential ambiguities.

4.3 LLM Reasoning with Mind Map

In this step, LLMs are prompted with two reasoning graphs G^m_{path} and G^m_{nei} in Step II to produce the final outputs. To generate a mind map and find the final results, we provide LLMs with a prompt consisting of five components: system instruction, question, evidence graph G_m, graph-of-thought instruction, and examples. The graph-of-thought instruction utilizes Langchain technology to guide LLMs in understanding and enhancing the input, constructing their own reasoning mind map, and indexing the knowledge sources within the mind map.

5 Research Report Generation

MetaGPT is a meta-programming framework for multi-agent collaboration based on LLMs. It is highly convenient and flexible, with well-defined functions like role definition and message sharing, making it a useful platform for developing LLM-based multi-agent system.

In this task, we designed two roles: an industry analyst named David for industry analysis and an equity analyst named Thomas for individual stock analysis. Both roles inherit from the base class Role. Each role is associated with corresponding Actions, as shown in Figure 4, and each Action also inherits from the base class Action.

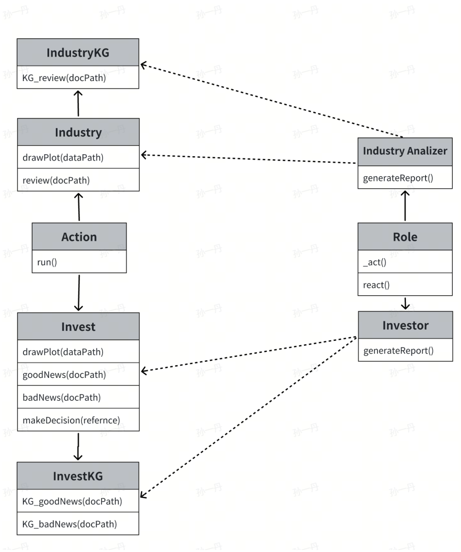


Figure 4: UML of MetaGPT

In each Action class, there is an asynchronous function designed to generate images reflecting the actual stock price movements. Additionally, there are two asynchronous functions used to analyze the positive and negative news related to the stock. These stock-related actions are ultimately invoked by the Investor role, which is responsible for generating individual stock reports. Similarly, for the Actions related to industry analysis, there is an asynchronous function designed to generate line charts of the prices of selected stocks. Additionally, this Action includes a function called review, which can analyze and comment on the market and top-tier stocks within the market based on the collected information, determining whether to continue holding these stocks.

For the KG-based RAG method, we also integrate it into the corresponding Actions. In InvestorKG, the KGgoodNews function inputs documents and questions into MindMap. MindMap generates an index graph based on the content of the documents, converts it into natural language, and returns the answer to the corresponding Action. The same applies to IndustryKG.

6 Conclusion

In summary, we first utilized the PiVe model to correct the collected triples, thereby improving the quality of the generated knowledge graph. Subsequently, we employed MindMap to perform RAG from the constructed knowledge graph. Finally, the information returned by MindMap was used as part of the responses from the Investor/IndustryAnalyst and returned to the user.

This report primarily discusses the methods and principles used. For specific implementation details, please refer to the submitted supplementary document.

References

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Jiuzhou Han, Nigel Collier, Wray Buntine, and Ehsan Shareghi. 2023. Pive: Prompting with iterative verification improving graph-based generative capability of llms. *arXiv preprint arXiv:2305.12392*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Yilin Wen, Zifeng Wang, and Jimeng Sun. 2023. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. *arXiv preprint arXiv:2308.09729*.