Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Can large language models effectively process and execute financial trading instructions?[#]

Yu KANG[§1], Xin YANG[§2], Ge WANG[§3], Yuda WANG[§4], Zhanyu WANG[†‡5], Mingwen LIU[†‡6]

[1]*School of Mathematics and Physics, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China*

[2]*School of Mathematics, Sun Yat-sen University, Zhuhai 519082, China*

[3]*School of Engineering, The Hong Kong University of Science and Technology, Hong Kong 999077, China*

[4]*School of Computing and Data Science, The University of Hong Kong, Hong Kong 999077, China*

[5]*School of Electrical and Computer Engineering, Sydney University, Sydney 2006, Australia*

[6]*Likelihood Lab, Guangzhou 510300, China*

[†]E-mail: zwan0839@uni.sydney.edu.au; maxwell@xiaochuang.ai

Received May 1, 2025; Revision accepted Oct. 8, 2025; Crosschecked Oct. 22, 2025

Abstract: The development of large language models (LLMs) has created transformative opportunities for the financial industry, especially in the area of financial trading. However, how to integrate LLMs with trading systems has become a challenge. To address this problem, we propose an intelligent trade order recognition pipeline that enables the conversion of trade orders into a standard format for trade execution. The system improves the ability of human traders to interact with trading platforms while addressing the problem of misinformation acquisition in trade execution. In addition, we create a trade order dataset of 500 pieces of data to simulate the real-world trading scenarios. Moreover, we design several metrics to provide a comprehensive assessment of dataset reliability and the generative power of big models in finance by using five state-of-the-art LLMs on our dataset. The results show that most models generate syntactically valid JavaScript object notation (JSON) at high rates (about 80%–99%) and initiate clarifying questions in nearly all incomplete cases (about 90%–100%). However, end-to-end accuracy remains low (about 6%–14%), and missing information is substantial (about 12%–66%). Models also tend to over-interrogate—roughly 70%–80% of follow-ups are unnecessary—raising interaction costs and potential information-exposure risk. The research also demonstrates the feasibility of integrating our pipeline with the real-world trading systems, paving the way for practical deployment of LLM-based trade automation solutions.

Key words: Large language model; Financial instruction; Evaluation; Dataset construction
https://doi.org/10.1631/FITEE.2500285 **CLC number:** TP391

1 Introduction

Traditional quantitative trading strategies rely on computer programs to collect and analyze financial data. Leveraging statistical models, these programs select optimal investment instruments and execute trades as per preset rules. Although this traditional approach has undeniably increased operational efficiency, modern quantitative systems are often tripped up by input of natural language (Sawhney et al., 2021). These systems are prone to errors in recognition and omissions of information when faced with complex, ambiguous, or incomplete trading instructions. Such issues create a disconnect between human-devised trading strategies and automated execution systems.

‡ Corresponding authors
§ These authors contributed equally to this work
# Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2500285) contains supplementary materials, which are available to authorized users
ORCID: Yu KANG, https://orcid.org/0009-0006-9688-4622; Xin YANG, https://orcid.org/0009-0007-6658-7464; Ge WANG, https://orcid.org/0009-0003-8903-1769; Yuda WANG, https://orcid.org/0009-0006-2439-7841; Zhanyu WANG, https://orcid.org/0000-0002-2079-4931

Moreover, in scenarios where trading platforms require high accuracy and speed in order execution, balancing model precision and execution efficiency is of vital importance (Zhou and Mehra, 2025).

The complexity of financial markets and the diversity of trading systems further compound these challenges (Aghion et al., 2018). There are remarkable differences in the interfaces and rules of various trading systems, leading to poor compatibility of trading orders (Kognole, 2024).

For example, vendor application programming interfaces (APIs) such as Bloomberg EMSX (https://github.com/tkim/emsx_api_doc/blob/master/source/index.rst) and Interactive Brokers TWS/IB API (https://ibkrcampus.com/campus/ibkr-api-page/twsapi-ref/) exhibit divergent field sets and workflow requirements, documented in vendor API manuals and integration guides. This divergence gives rise to cross-platform compatibility issues: A large volume of generated orders requires manual reformatting, thereby seriously undermining automation efficiency (Min and Borch, 2022).

To solve these problems, we propose a method that leverages large language models (LLMs) to generate a high-quality dataset of trading orders. By simulating the real-world trading scenarios, this method can compensate for the scarcity of real-world data while providing a unified standard for compatibility testing across different trading systems.

Existing research has confirmed the effectiveness of LLMs in generating financial domain texts, such as financial news summaries and market analysis reports, offering theoretical support to our work (Dolphin et al., 2024). Furthermore, we have manually aligned and verified the trading-order data generated by LLMs, ensuring their accuracy and consistency, thus enhancing the dataset's credibility. Given the limited availability of real-world natural language trading instruction data online, LLM-generated data have emerged as a necessary alternative. Practice has shown that data generated by LLMs can effectively fill the gap left by the shortage of real-world data (Long et al., 2024).

Therefore, our research addresses these challenges and makes significant contributions as follows:

1. We successfully construct a high-quality dataset comprising 500 diverse trading instructions using LLMs.

2. We develop an intelligent trading instruction recognition and execution system capable of accurately identifying trading requirements, dynamically soliciting missing information, and translating natural language instructions into a standardized format for automated execution.

3. We design and implement a comprehensive set of metrics to evaluate five cutting-edge LLMs in processing trading instructions, offering an in-depth analysis of their capabilities and limitations in the context of automated trading systems.

2 Related works

2.1 LLMs in finance

Natural language processing (NLP) has formed a well-established research framework in the financial sector, with core applications including sentiment analysis (Day and Lee, 2016; Sohangir et al., 2018), fraud detection (Han et al., 2018; Nourbakhsh and Bang, 2019), market forecasting (Chen et al., 2014; Cavalcante et al., 2016), and news headline classification (Sinha and Khandait, 2021). Previous research predominantly employs traditional feature engineering methods such as bag-of-words modeling (Zhang et al., 2010), combined with machine learning algorithms to tackle relevant tasks (Bollen et al., 2011). However, these methods are prone to substantially higher parsing errors compared to text in general domains (Garimella et al., 2021). The advent of deep learning spurred a pivotal breakthrough in financial NLP (Yang et al., 2020). Recurrent neural networks (RNNs) (Schuster and Paliwal, 1997) and their variants processing sequence modeling capabilities outperform traditional time-series models in market volatility forecasting (di Persio and Honchar, 2017; Fischer and Krauss, 2018). However, inherent limitations of RNNs, particularly the lack of symbolic reasoning capability, render them ill-suited to meet the precise logical parsing demands of trade execution (Li ZN et al., 2023). Traditional automated trading systems also face constraints in processing unstructured natural language inputs, as they rely heavily on structured data inputs (Wei et al., 2023; Kim M et al., 2025). Additionally, the programming proficiency required to operate these platforms creates barriers for nontechnical users, driving up learning costs.

Large models based on Transformer (Vaswani et al., 2017), such as ChatGPT4 (https://openai.com/chatgpt) and GPT-4 (OpenAI, 2023), reconstruct the paradigm of financial NLP (Li YH et al., 2023; Yu and Hamam, 2024). Studies, such as Sinha et al. (2025), can use fine-tuned LLMs to generate the context required to answer user queries and extract data from modules to efficiently process financial queries. In addition, LLMs enable seamless integration of trading strategy descriptions in natural language with automated execution (Xiao et al., 2024).

Although LLMs have shown unprecedented potential in financial NLP, they still face notable challenges in the real-world financial trading system applications. First, hallucination in LLMs may lead to erroneous trading decisions (Yao, 2025). Financial trading demands high real-time performance and accuracy, yet the inference latency and computational resource requirements of LLMs may fail to meet the timeliness demands of high-frequency trading (Agrawal et al., 2024; Li BL et al., 2024). Furthermore, despite the extensive knowledge base of LLMs, they still lack in-depth expertise in specific financial markets (e.g., the Chinese market) and a thorough understanding of regulatory rules (Li JT et al., 2023)—limiting their applicability in specialized financial scenarios.

2.2 Financial benchmark

With the rapid advancement of LLMs in NLP, datasets generated by such models have emerged as a prominent research focus (Wang et al., 2024). By leveraging LLMs' powerful generative capabilities, scholars have curated extensive multidomain, multitask corpora—for instance, instruction-tuned universal text datasets that substantially enhance model performance in generation, question answering, and related tasks (Sanh et al., 2022). These resources, generated by LLMs, leveraging the models' profound understanding and proficient mastery of human language, supply rich training material that drives progress across NLP.

In the financial sector, dataset creation has also witnessed notable achievements. The financial benchmark (FinBen) (Xie et al., 2024), for instance, offers a comprehensive evaluation framework that comprises 42 datasets and 24 financial tasks. It assesses LLMs across eight dimensions—including information extraction, text analysis, and question answering—and covers common scenarios such as sentiment analysis and risk detection. Other financial corpora focus on news sentiment analysis, market-trend prediction, and similar tasks, furnishing essential data support for NLP research in finance (Sinha et al., 2022). However, most existing resources concentrate on analysis and forecasting of financial information, paying limited attention to the processing of trading instructions (Malo et al., 2014).

Although the number and variety of financial datasets continue to grow (Tatarinov et al., 2025), there remains a pronounced shortage of corpora specifically tailored to trading-instruction scenarios (Huang et al., 2025). Current benchmarks fail to simulate the complex instruction issuance and execution processes inherent in the real-world trading (Li CL et al., 2025), leaving LLMs under-trained for tasks in trade decision-making and strategy execution. As noted in the FinBen report, financial trading demands both real-time responsiveness and precise logical interpretation: Yet existing datasets fall short in terms of instruction complexity, specificity, and domain-specific rigor. This gap highlights the need for targeted data resources and motivates the present study.

3 Dataset

Our dataset is developed via a systematic pipeline to address the scarcity of specialized trading instruction corpora. The process, illustrated in Fig. 1, involves several key steps to transform raw financial reports into a structured, high-quality dataset of 500 diverse trading instructions. The methodology for constructing the dataset is as follows:

1. Data preprocessing. Research reports from Sina Finance are selected as seed data due to their professional and timely nature. These raw data are then preprocessed to remove noise, such as HTML tags, and standardized price expressions. Key information extraction: Sentiment analysis is applied to the cleaned text to identify passages with a clear trading bias, such as buy or sell intentions.

2. Data structuring. The extracted information is organized into a standardized four-part format (topic/background, instruction, thinking, and reasoning). This

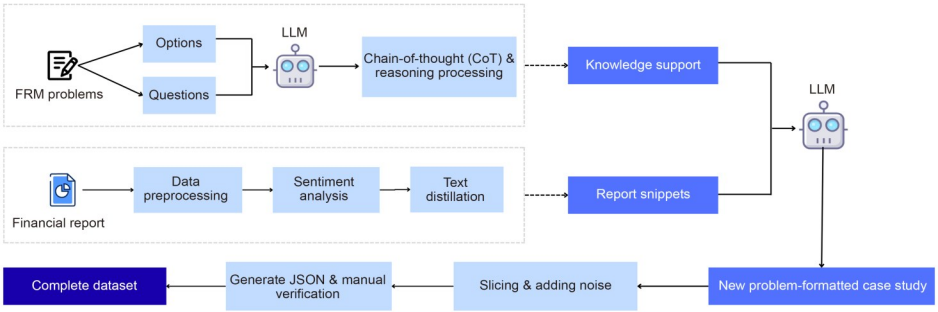

Fig. 1 Data acquisition pipeline for generating trading instructions

format is designed using the financial risk manager (FRM) exam content as a reference for professionalism and consistency.

3. Data augmentation. To simulate the real-world conditions, the structured data are augmented by adding "noise" (e.g., ambiguous phrases and Chinese–English mixing), segmenting structured data into smaller pieces to mimic fragmented instructions.

4. Data annotation. A comprehensive manual alignment is performed on 472 data points to ensure accuracy and credibility. The alignment criteria include key trading elements, such as strategy, ticker symbol, trade type, trade volume, and trade price.

3.1 Data preprocessing

After meticulously analyzing the limitations of existing financial datasets, we observe that while historical trading records offer abundant structured data, they fail to capture the nuances and complexities within trading instructions. Financial analysts' research reports emerge as an ideal alternative data source. These reports encompass crucial information, such as trading recommendations, price forecasts, and risk warnings, all presented in natural language that can imitate the real-world trading scenarios. Users often consult this information to determine the timing and strategy for buying or selling.

Based on these considerations, we select research reports as seed data. They are authoritative and professional, written by experienced analysts and subject to stringent reviews, guaranteeing the accuracy and reliability of the information. Moreover, the timeliness of research reports enables these reports to reflect the latest market dynamics, offering rich sample data for research. By leveraging the natural language descriptions in these reports, we can better simulate various input scenarios that users may encounter in the actual trading. This, in turn, enhances the model's ability to handle complex and diverse trading instructions.

When selecting data sources, we conduct a comprehensive evaluation and ultimately choose Sina Finance as the primary source, driven by two key rationales. First, aligning with our focus on the Chinese market, Sina Finance provides multidimensional real-time market data and professional research reports covering Shanghai and Shenzhen stocks. These reports—authored by renowned domestic and international financial institutions—encompass company analysis, industry trends, investment strategies, and risk warnings, offering rich observational dimensions (see supplementary materials for a raw data sample). Second, Sina Finance's publicly accessible and free data facilitates efficient access to extensive research reports, laying a solid foundation for dataset construction. However, it is important to acknowledge that the data are solely sourced from the Chinese market, so that the research findings may exhibit geographical limitations when generalized to other regions.

Moreover, the raw data (e.g., stock prices and industry classifications) are extracted from publicly available information on Sina Finance (https://finance.sina.com.cn), which aggregates market data disclosed by exchanges and financial institutions. Such factual information is not subject to copyright restrictions under Chinese laws, and our use complies with the principle of fair use for academic research.

After identifying the research reports as seed data, we start to screen the eligible reports in the Chinese market from the reports of Sina Finance. Subsequently, we adopt a systematic approach for key information

extraction from relevant reports.

We preprocess and normalize the raw data. This process includes removing HTML tags, special characters, and redundant line breaks to obtain clean text; normalizing price expressions, such as $112 per share and $112, into a uniform format; applying segmenting and HanLP (https://hanlp.com) lexical annotation techniques to extract keywords, including verbs, nouns, and numbers. These steps ensure the data quality and consistency of the subsequent analysis.

Then, we use sentiment analysis techniques to screen the text initially. Sentiment analysis has been widely validated in financial text research to effectively identify text segments relevant to trading decisions. We analyze the sentiment tendencies of the preprocessed texts for each company or stock, and correlate the sentiment expressions with potential trading intentions: Negative sentiments may suggest a sell tendency, whereas positive sentiments may point to a buy intention. To achieve this, we adopt a lightweight hybrid approach. On one hand, domain-specific sentiment lexicons containing financial terms (e.g., "bullish," "bearish," "undervalued," and "overweight") are used to capture explicit directional signals. On the other hand, a supervised machine learning classifier is trained on a small set of annotated financial sentences to recognize implicit sentiment cues that may not be covered by lexicons. This combination allows the pipeline to filter out neutral descriptions while retaining segments with actionable trading intentions in a reliable yet efficient manner.

In this process, we select only those passages that, through sentiment analysis, are determined to contain a clear trading bias. These passages are extracted as part of the prompt input to our model. For example, when a passage exhibits a negative sentiment evaluation of a specific stock, we extract it so that the model can generate appropriate trading orders based on this information. This targeted extraction approach ensures that our dataset contains only information that is directly relevant to the trading decision, thus improving the effectiveness of subsequent data generation.

3.2 Data structuring

After completing data collection and key information extraction, we face the challenge of how to organize this information into structured and standardized trading order data. We find that there is a lack of specialized datasets directly related to trading orders in the current research field, and most of existing financial datasets focus on market data, such as stock prices, which are difficult to directly apply to the research purpose of trading order processing.

To construct high-quality trading order data, we adopt a two-track approach. On one hand, we use the aforementioned trading-related snippets extracted from research reports as the base material; on the other hand, to ensure professionalism and consistency, we select the financial risk manager (FRM) (https://www.garp.org/frm) exam content as a reference standard, as it offers rich practical trading order knowledge and scenarios.

As shown in Fig. 1, our data construction pipeline operates through two parallel streams that converge into a final LLM processing stage. The first stream processes FRM exam problems by parsing them into options and questions, which are then fed into an LLM for chain-of-thought (CoT) reasoning and processing to generate knowledge support. The second stream handles financial reports through sequential data preprocessing, sentiment analysis, and text distillation to produce report snippets that also serve as knowledge support. Both knowledge sources are then integrated and fed into a final LLM to generate new problem-formatted case studies. The generated case studies undergo data augmentation through slicing and noise addition to better reflect the fragmented and noisy nature of real financial communications. Finally, the augmented data are formatted into JavaScript object notation (JSON) records with manual verification to ensure quality and consistency, resulting in our complete dataset.

3.3 Data augmentation

After constructing data entries with comprehensive trading information, we implement a further processing methodology to enhance the complexity and representativeness of the dataset. This involves data augmentation through noise injection (Kim GI and Chung, 2024) (specific examples are shown in Table 1) and data segmentation via slicing, simulating the diverse variations encountered in the real-world applications to improve the fidelity of the dataset to actual market conditions (Fons et al., 2020).

Table 1 Examples of noise injection

Noise type	Original data	Data after adding noise
Colloquial filler	听说上半年茅台酒收入稳健增长，我想买100股看看。	emmm…那个，听说上半年茅台酒收入稳健增长，我想买100股看看。
Vague phrase	东方财富金融信息做的挺专业的，我计划在20块左右买点。	东方财富金融信息看起来做的挺专业的，我可能会在20块左右买点。
Punctuation noise	五粮液最近的业绩表明渠道改革初显成效啊，我决定就按行情搞它200股。	五粮液最近的业绩。表明渠道改革初显成效，我决定！就按行情搞它200股。
Chinese–English mixing	伊利股份的股价短期内在下跌，我决定按市场价买入两百股。	伊利股份的股价短期内在下跌，I’ve decided to buy 200 shares at the market price.

3.3.1 Noise injection

Noise types are strictly limited to four categories: colloquial fillers, vague phrases, punctuation or sentence-breaking noise, and Chinese–English mixing.

1. Colloquial fillers. Colloquial fillers refer to hesitation words such as “en,” “nàge” (that), or “emmm,” aiming to examine the robustness of the model in handling hesitant tones in real conversations, without misinterpreting key information due to filler words.

2. Vague phrases. Vague phrases include expressions such as “keneng” (possible), “kanqilai” (look like), or “dagai” (probably), simulating uncertain or tentative inputs and enabling the model to either trigger clarification or adopt a conservative strategy.

3. Punctuation noise. Punctuation or sentence-breaking noise denotes randomly adding, deleting, or misplacing punctuation, as well as artificially breaking sentences. Such anomalies often cause rule-based parsers to fail, so that robustness under abnormal punctuation is essential.

4. Chinese–English mixing. Chinese–English mixing refers to inserting English words, tickers, abbreviations, or Pinyin in sentences, which is common in financial contexts and tests the model’s ability to parse commands in mixed-language environments.

To ensure controllability, noise injection follows an independent Bernoulli sampling mechanism with upper limits: colloquial fillers 8%–12% (high frequency, low semantic impact), vague phrases 6%–10% (less frequent but significant for uncertainty), punctuation noise 5%–10% (reflecting common errors in corpora), and Chinese–English mixing 3%–7% (less frequent but strongly disruptive). For each sample, at most two noise types are applied.

3.3.2 Data segmentation

The purpose of slicing is to construct inputs that are closer to real interaction scenarios, evaluating the understanding and reasoning capabilities of large models under fragmented information conditions.

In real transactions, instructions are often not delivered in full at once but may be truncated in instant messaging, voice transcription, or market updates, or spread across multiple messages or conversation rounds. To simulate such partially received and context-dispersed inputs, we slice the original text sequentially into nonoverlapping segments of 10–15 words, with breakpoints aligned as closely as possible to natural semantic boundaries. These segments preserve sufficient key information for judgment while exposing the model to a fragmented context. For example, when a sliced segment also contains injected noise, such as a filler word or a misplaced punctuation mark, the model must still extract the trading intent from incomplete and noisy input, closely mirroring real interactive scenarios.

The examples of data processing methods can be found in Table 2, and the examples of the dataset that is used as input to the LLMs for evaluations are in Table 3.

3.4 Data annotation

To ensure high-quality annotations for financial transaction understanding, we implement a comprehensive annotation process with rigorous inter-annotator agreement (IAA) evaluation. Building upon our initial manual alignment of 472 transaction data points from 500 artificially constructed samples, we establish a systematic annotation framework involving domain expertise and quality control mechanisms.

Table 2 Example of data processing

Data type	Example
Original data	The Skyworth figure in my hand has risen a lot. I decided to take advantage of the good market price and sell all 300 shares in my hand.
Noise data	Oh, Skyworth's stock? It's emmm…risen so much! Selling maybe, uh, all 300 shares, capitalizing on the high price.
Sliced data	The Skyworth figure in my hand has risen a lot.

Table 3 Examples of data for the input of the LLMs

Number	Example
1	If the share price of Moutai drops to 1800, I'll take the opportunity to buy more. I plan to purchase 200 shares of it.
2	Longi Corporation, I bet it can fly. I have to take the "golden pit" of 100 yuan first!
3	Goodix's fingerprint recognition chip is really advanced. I'll buy some when the stock price stabilizes.

Our annotation team comprises four graduate student annotators, each possessing basic financial knowledge, organized into two groups of two members each. Each group is assigned half of the total dataset samples. Additionally, a financial domain expert with more than 5 years of trading experience and chartered financial analyst (CFA) certification served as the final arbiter for complex cases and business-related disputes.

3.4.1 Annotation process

Within each group, both annotators independently label all assigned samples using predefined criteria covering five essential trading elements: trading strategy (market vs. limit orders), ticker symbol, trade type (buy/sell), trade volume, and trade price (Table 4). Following independent annotation, group members cross-validate their results to identify discrepancies and assess consistency.

When a consensus cannot be reached through group discussion, cases are escalated to our CFA-certified expert for final adjudication, particularly for business logic and financial interpretation issues. This hierarchical resolution mechanism ensures both technical accuracy and domain validity. All annotations follow strict JSON formatting standards, with null values represented as null for missing information (e.g., price in market orders or absent user inputs), and UTF-8 encoding for all string fields. This standardization facilitates consistent parsing and analysis across different LLM responses.

Table 4 Examples of manual alignment results

User input	JSON output
If Montai's stock price can fall to 1800, I will take the opportunity to stock up and plan to buy 200 shares of it.	{"strategy": "limit order", "symbol": "60 0519", "order_type": "buy," "price": 1800.0, "quantity": 200}
Looking at Vanke stocks to go up, I intend to sell 300 shares first, the bottom position in the hand will not move, and see whether there is room to rise behind.	{"strategy": null, "symbol": "00 0002", "order_type": "sell", "price": null, "quantity": 300}
I intend to buy 100 shares of Guizhou Moutai while the current stock price is reasonable.	{"strategy": "market order", "symbol": "60 0519", "order_type": "buy," "price": null, "quantity": 100}

Our comprehensive annotation process addresses the observed challenge where LLMs frequently struggle to distinguish between market and limit orders. A market order executes immediately at the best available price, while a limit order sets a specific price threshold that may or may not be filled. This distinction remains crucial for accurate financial instruction interpretation and automated trading system development.

3.4.2 Evaluation

We employ Cohen's Kappa coefficient to quantify agreement between annotators within each group. The observed Kappa scores are 0.89 and 0.92 for the two groups, respectively, both exceeding the recommended threshold of 0.8 and indicating substantial agreement. For cases with disagreement, annotators engage in structured discussion based on the established financial trading standards and data formatting guidelines.

This dataset is constructed to simulate the complexity of real transactions, reflecting ambiguous expressions, information gaps, and domain-specific requirements in the actual transaction scenarios. The structured design provides a crucial testing platform for evaluating core LLM capabilities: generating stable and structured outputs, accurately mapping semantic intentions, identifying missing information, and conducting targeted inquiries.

4 Experiments

This study aims to evaluate the effectiveness of LLMs in processing financial trading instructions, focusing on the following core research questions:

1. Can LLMs generate text data in accurate financial trading formats?

2. For user instructions containing colloquial expressions or financial terminology, can LLMs correctly understand and map the information into the required financial trading format?

3. When user instructions lack necessary information, can LLMs conduct multiple follow-ups and finally generate complete and accurate financial trading format data?

4.1 Flowmatch pipeline

To address these questions, we design a phased flowmatch evaluation framework, dividing user model interactions into two core stages and a subsequent information supplementation process:

1. Format stability verification. This stage focuses on the model's basic understanding of financial trading formats. The model is required to first generate JSON data conforming to predefined specifications, with key evaluations on the integrity of the output structure (e.g., whether mandatory fields such as transaction type, asset symbol, and price parameters are included) and syntactic compliance (e.g., correct key names, data type matching, and JSON syntax). This stage verifies whether the model can stably output a basic format framework parsable by financial systems.

2. Instruction-format mapping accuracy assessment. After confirming format stability, we further examine the model's semantic understanding of user instructions. User instructions containing financial terminology (e.g., "limit order" and "stop loss") or natural language expressions (e.g., "buy Apple Incorporated stock on dips") are input into the model to evaluate its ability to accurately extract key information (e.g., asset symbol, trading direction, and price conditions) and map the information to the standard format fields established at stage 1 without omission or error.

3. Information complementation and follow-up ability test. If the JSON data generated in the first two interactions are incomplete due to missing information in user instructions (e.g., lacking critical parameters such as "stop price" or "expiry date"), the model enters the follow-up stage. This stage evaluates whether the model can identify missing fields, reasonably ask users for supplementary information in natural language, and incrementally map the user's subsequent responses into the existing format. Qualified performance should ensure that follow-ups adhere to financial business logic (e.g., confirming transaction type before asking for quantity), and the complemented data are neither redundant nor missing, ultimately forming executable complete trading instructions.

4.2 Evaluation metrics

To comprehensively measure model performance from shallow to deep levels, we design a multilevel metric system.

4.2.1 Shallow format metrics

1. Generation rate. The proportion of valid JSON outputs conforming to grammatical specifications among the total test cases is calculated as

$$\frac{\text{Number of valid JSON outputs}}{\text{Total number of input cases}}\times 100\%.$$

2. Missing rate. The proportion of missing mandatory fields in the output data is calculated as the average missing rate by field:

$$\frac{\text{Number of missing fields}}{\text{Total number of fields required}}\times 100\%.$$

3. Correctness. The proportion of correct field values in the output data is calculated as

$$\frac{\text{Number of correct field values}}{\text{Total number of field values filled}}\times 100\%.$$

4. Accuracy. The proportion of completely valid outputs that are format-correct, field-complete, and value-accurate integrates shallow metrics:

$$\frac{\text{Number of fully correct JSON outputs}}{\text{Total number of input cases}}\times 100\%.$$

4.2.2 Field-level metrics: precision, recall, and F1

1. Counting (micro-average): We evaluate the fields such as strategy, order_type, symbol, price,

and quantity. Before comparison, we normalize case/spacing and synonyms; numeric values are compared after parsing to floats.

2. True positives (TP): Number of predicted fields whose key exists in the ground truth and whose value exactly matches the ground truth after normalization.

3. False positives (FP): Number of predicted fields that are unnecessary (key not required) or have an incorrect value.

4. False negatives (FN): Number of required fields that are missing from the prediction or are left incorrect.

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \times 100\%,$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \times 100\%,$$

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}} \times 100\%.$$

5. Reporting levels: We compute the above at three granularities: overall (all fields), strategy (only strategy), and order type (only order_type).

4.2.3 Interaction ability metrics

1. Follow-up rate. The proportion of cases where the model initiates follow-ups when user instructions lack critical information, reflecting its awareness of information complementation, is calculated as

$$\frac{\text{Number of cases with follow-ups}}{\text{Total number of missing field cases}} \times 100\%.$$

2. Missed follow-up rate. The proportion of cases where the model fails to identify necessary missing information, leading to incomplete final outputs, is calculated as

$$\frac{\text{Number of cases without follow-ups}}{\text{Total number of missing field cases}} \times 100\%.$$

3. Extra follow-up rate. The proportion of cases where the model asks for redundant information (e.g., inquiring about already provided fields), measuring the precision of its follow-up logic, is calculated as

$$\frac{\text{Number of redundant follow-up cases}}{\text{Total number of follow-up cases}} \times 100\%.$$

These metrics progress from format compliance to semantic accuracy and interaction intelligence, comprehensively covering the core capability dimensions of financial trading instruction processing and providing a multidimensional analytical framework for quantifying the practicality of LLMs in this domain.

4.3 Experimental setup

We select six off-the-shelf LLMs: Yi-large (Young et al., 2025), GPT-4o-2024-05-13 (OpenAI, 2024b), GPT-4o-mini-2024-07-18 (OpenAI, 2024a), Qwen-max-0428 (Qwen Team, 2024), TraderGPT (https://tradergpt.xiaochuangai.com), and DeepSeek-V3 (DeepSeek-AI, 2025). Each model was accessed via its API to evaluate its performance on (1) identifying missing schema fields in natural-language trading instructions, (2) generating targeted follow-up questions to elicit those fields, and (3) producing a complete, executable JSON-schema trading plan.

4.4 Experimental analysis

Tables 5 and 6 show that most models convert the vast majority of natural-language trading instructions into syntactically valid JSON: the generation rate ranges from 80.17% (TraderGPT) to 98.90% (DeepSeek-V3), with five of six models at or above 87%. However, semantic and structural completeness remains the bottleneck: missing rates span 11.95% (GPT-4o-2024-05-13) to 66.20% (TraderGPT), and end-to-end accuracy remains low from 6.10% to 14.17%, with DeepSeek-V3 and Yi-large tied at the top (14.17%). These figures indicate that, although Transformer architectures reliably emit the target schema, extracting every required field (e.g., order_type, symbol, price, and quantity) with full correctness still eludes most models.

Table 7 assesses the financial inquiry capabilities of LLMs through three metrics: follow-up rate (higher is better, reflecting the proactiveness of initiating necessary follow-ups), missed follow-up rate (lower is better, reflecting the extent of key information omission), and extra follow-up rate (lower is better, reflecting the redundancy of unnecessary follow-ups). Among all models, DeepSeek-V3 performs the best in all metrics: it achieves a follow-up rate of 99.60% (nearly perfect proactiveness in initiating necessary follow-ups), a missed follow-up rate of only 1.23% (hardly omits key financial information), and an extra follow-up rate of 70.12% (has the least redundant follow-ups). This

Table 5 Format generation evaluation of rates and exactness

Model name (release)	Type	Generation rate (%) ↑	Missing rate (%) ↓	Correctness (%) ↑	Accuracy (%) ↑
DeepSeek-V3	Overall	**98.90±0.20**	32.34±0.62	79.62±1.02	**14.17±0.36**
	Strategy	99.32±0.70	40.68±0.81	80.00±1.18	71.19±0.92
	Order type	86.44±0.61	13.56±0.37	91.18±0.94	89.83±0.69
GPT-4o-2024-05-13	Overall	89.83±0.59	**11.95±0.41**	63.88±0.96	10.00±0.35
	Strategy	91.51±0.61	8.49±0.31	60.82±0.84	60.38±0.58
	Order type	98.11±0.47	1.89±0.19	89.42±0.81	89.62±0.57
GPT-4o-mini-2024-07-18	Overall	87.29±0.68	14.08±0.49	65.97±1.01	9.82±0.38
	Strategy	92.23±0.77	7.77±0.29	55.79±0.93	55.34±0.67
	Order type	94.17±0.58	5.83±0.27	91.75±0.88	91.26±0.61
Qwen-max-0428	Overall	92.37±0.55	19.42±0.46	73.21±1.04	13.33±0.41
	Strategy	94.50±0.60	5.50±0.21	63.11±0.83	63.30±0.62
	Order type	95.41±0.52	4.59±0.22	90.38±0.74	89.91±0.61
Yi-large	Overall	98.31±0.41	31.75±0.68	**80.16±1.00**	**14.17±0.19**
	Strategy	95.17±0.82	44.83±0.88	82.81±1.19	70.69±0.81
	Order type	87.07±0.66	12.93±0.39	91.09±0.88	89.66±0.65
TraderGPT	Overall	80.17±0.88	66.20±1.02	26.76±1.31	6.10±0.30
	Strategy	82.33±0.92	48.00±0.98	28.41±1.28	40.55±0.78
	Order type	65.22±0.85	30.15±0.90	33.67±1.20	45.18±0.82

Bold indicates the best performing model; underline indicates the second-best performing model. All values are mean±standard deviation over multiple runs. ↑ indicates that a higher value of the metric is better, while ↓ indicates that a lower value is better

Table 6 Format generation evaluation of F1, precision, and recall

Model name (release)	Type	F1 (%) ↑	Precision (%) ↑	Recall (%) ↑
DeepSeek-V3	Overall	**78.52±0.84**	81.54±0.92	78.71±0.73
	Strategy	75.68±0.88	80.00±0.84	71.79±0.82
	Order type	90.73±0.79	91.18±0.86	90.29±0.81
GPT-4o-2024-05-13	Overall	72.19±0.75	67.64±0.72	77.41±0.70
	Strategy	68.60±0.77	60.82±0.69	78.67±0.69
	Order type	91.18±0.70	89.42±0.62	93.00±0.59
GPT-4o-mini-2024-07-18	Overall	71.90±0.82	68.81±0.88	77.19±0.79
	Strategy	62.72±0.81	55.79±0.78	71.62±0.71
	Order type	92.23±0.73	91.75±0.71	92.71±0.66
Qwen-max-0428	Overall	76.03±0.82	72.55±0.73	**79.86±0.78**
	Strategy	72.22±0.78	63.11±0.64	84.42±0.70
	Order type	91.26±0.69	90.38±0.63	92.16±0.62
Yi-large	Overall	76.95±0.83	**82.35±0.79**	75.81±0.80
	Strategy	75.18±0.90	82.81±0.88	68.83±0.90
	Order type	90.64±0.79	91.09±0.77	90.20±0.68
TraderGPT	Overall	65.32±0.76	42.18±0.72	48.75±0.79
	Strategy	63.25±0.82	40.91±0.75	46.12±0.80
	Order type	69.02±0.81	46.38±0.78	52.25±0.83

Bold indicates the best performing model; underline indicates the second-best performing model. All values are mean±standard deviation over multiple runs. ↑ indicates that a higher value of the metric is better, while ↓ indicates that a lower value is better

indicates that DeepSeek-V3 can not only proactively and fully supplement the core information required for financial transactions but also precisely grasp the boundary of "no need to follow up," perfectly matching the dual requirements of "information accuracy" and "interaction efficiency" in financial scenarios.

Key-field errors and risk. Beyond aggregate scores, we summarize a key-field error rate (KFER) that

flags any case where side, quantity, or price is missing or incorrectly mapped after normalization. KFER isolates execution-critical defects. We also tag dangerous cases—e.g., side flips, order-of-magnitude quantity mistakes, and price scale/threshold inversions—that could plausibly cause unintended fills or outsized exposure. Stronger models exhibit fewer such hazards, and the focused follow-ups noticeably reduce them; residual risks cluster around price normalization and quantity units, while side flips are rarer but the highest severity. Concrete KFER and hazardous-case shares will be reported after runs are finalized.

5 Ablation studies

5.1 Experimental setup

We perform ablation experiments to measure the individual impact of two dataset construction modules: instruction slicing and noise injection. In each ablation setting, we remove one module while keeping the rest of the pipeline identical. Evaluation is based on three inquiry metrics—follow-up rate, missed follow-up rate, and extra follow-up rate—on three representative production LLMs: DeepSeek-V3, GPT-4o-2024-05-13, and Qwen-max-0428.

The instruction slicing module fragments each instruction into 10–15 word segments, simulating incomplete queries from chat or voice transcription. The noise injection module perturbs instructions with numeric typos, synonym substitutions, and light formatting noise to mimic the real-world imperfections such as speech-to-text errors and human entry mistakes.

5.2 Experimental analysis

As shown in Tables 8–10 for the instruction slicing module, fragmenting orders into 10–15 word segments simulates incomplete chat or voice queries. Removing slicing keeps follow-up rates high (≈96%–99%) but consistently increases the extra follow-up rate

Table 7 Inquiry capability evaluation

Model name (release)	Follow-up rate (%) ↑	Missed follow-up rate (%) ↓	Extra follow-up rate (%) ↓
Yi-large	98.51±0.35	9.49±0.22	76.12±1.00
GPT-4o-2024-05-13	97.04±0.58	6.96±0.47	75.78±0.95
GPT-4o-mini-2024-07-18	94.54±0.60	7.46±0.40	74.63±0.90
Qwen-max-0428	96.00±0.40	4.00±0.30	74.09±0.88
TraderGPT	90.13±0.85	10.66±0.55	80.22±1.10
DeepSeek-V3	**99.60±0.72**	**1.23±0.18**	**70.12±0.80**

Bold indicates the best performing model; underline indicates the second-best performing model. All values are mean±standard deviation over multiple runs. ↑ indicates that a higher value of the metric is better, while ↓ indicates that a lower value is better

Table 8 Inquiry capability ablation of both slicing and noise enabled

Model name (release)	Follow-up rate (%) ↑	Missed follow-up rate (%) ↓	Extra follow-up rate (%) ↓
DeepSeek-V3	**99.60±0.72**	**1.23±0.18**	**70.12±0.80**
GPT-4o-2024-05-13	97.04±0.58	6.96±0.47	75.78±0.95
Qwen-max-0428	96.00±0.40	4.00±0.30	74.09±0.88

Bold indicates the best performing model. All values are mean±standard deviation over multiple runs. ↑ indicates that a higher value of the metric is better, while ↓ indicates that a lower value is better

Table 9 Inquiry capability ablation without noise (slicing only)

Model name (release)	Follow-up rate (%) ↑	Missed follow-up rate (%) ↓	Extra follow-up rate (%) ↓
DeepSeek-V3	**99.30±0.70**	**1.40±0.20**	**72.90±0.90**
GPT-4o-2024-05-13	96.70±0.60	7.20±0.50	78.50±0.95
Qwen-max-0428	95.80±0.45	4.20±0.35	76.80±0.90

Bold indicates the best performing model. All values are mean±standard deviation over multiple runs. ↑ indicates that a higher value of the metric is better, while ↓ indicates that a lower value is better

Table 10 Inquiry capability ablation without slicing (noise only)

Model name (release)	Follow-up rate (%) ↑	Missed follow-up rate (%) ↓	Extra follow-up rate (%) ↓
DeepSeek-V3	**99.40±0.65**	**1.35±0.20**	**77.50±1.00**
GPT-4o-2024-05-13	97.20±0.60	6.80±0.50	82.10±1.10
Qwen-max-0428	96.10±0.45	3.90±0.35	80.00±1.00

Bold indicates the best performing model. All values are mean±standard deviation over multiple runs. ↑ indicates that a higher value of the metric is better, while ↓ indicates that a lower value is better

by about 6 percentage points (PPs)–8 PPs for most models, indicating more redundant questions. This shows that slicing pressures models with partial context, improving their ability to request only truly missing fields, thus enhancing the robustness in interactive trading scenarios.

For the noise injection module, perturbing input text with numeric typos, synonym changes, and formatting noise mimics the real-world imperfections in order entry. Removing noise changes the follow-up rate only marginally (≤0.3 PPs) but increases both missed follow-up rate and extra follow-up rate for all models (e.g., DeepSeek-V3: extra follow-up rate +2.78 PPs). This demonstrates that noise injection boosts resilience to imperfect inputs, enriches data diversity, and reduces overreaction to minor distortions.

5.3 Summary

Instruction slicing primarily strengthens robustness under the fragmented context, while noise injection increases input diversity and improves precision under noisy conditions. Together, these modules simulate realistic trading dialog challenges, ensuring the benchmarking captures both structural and conversational complexity.

6 System experiments

To validate the practical applicability of our intelligent trading system, we construct a simulation environment synchronized with real market data (see Fig. S2 in the supplementary materials). This environment strictly follows the actual price feeds from A-share and Chinese futures markets, supporting both market orders and limit orders, while ensuring experimental safety by avoiding the actual capital flows and trade settlements.

6.1 Experimental design

Considering the trading hour constraints of the A-share market, all experiments are conducted during official trading sessions: 9:30–11:30 AM and 1:00–3:00 PM. This design ensures consistency between experimental data and real market conditions, enhancing the credibility of our results.

We sample 30 inputs from the preprocessed dataset to simulate real-life scenarios. For agent configuration, both the user agent and the trading agent use GPT-4o-2024-05-13 as the core reasoning engine. Besides, the system establishes a 20-min expected execution window on limit order trading. If market prices fail to reach the user's desired limit price within this timeframe, the system automatically converts to market order execution, ensuring the final trade completion.

One major challenge in practical deployment is dealing with incomplete user inputs. To address this, our system adopts a schema-aware completion mechanism directly within the input processing workflow. In our designed JSON schema, the system proactively detects missing fields, such as strategy and price, and interactively requests users to supply the required information. The schema-aware mechanism enables the agents to identify gaps based on a predefined schema template, ensuring robust downstream processing and preventing transaction failures due to incomplete information. This strategy enhances the adaptability and fault tolerance of agent-user interactions.

To enhance system robustness, we design a multi-layered price information assurance mechanism. First, we implement a real-time data synchronization mechanism: The trading agent proactively pushes the latest market price information before each interaction round, ensuring that the user agent makes decisions based on accurate market conditions. Second, we establish a price reasonableness validation mechanism: Recognizing that users may have strategic limit order

requirements (such as buying low or selling high), the system sets relative price deviation thresholds (5% of the current market price). When the user agent provides limit prices exceeding this range, the system performs secondary confirmation to avoid unintended trades due to input errors or understanding deviations. Finally, we introduce a trade timeout retry mechanism: When limit orders fail to execute within the preset 20-min window, the system automatically retrieves updated market information and converts to market order execution, ensuring trade completion and improving the overall success rate.

Our evaluation framework encompasses three key metrics:

1. Success rate. This measures the system's ability to complete valid transactions.

2. Average interaction. This reflects collaboration efficiency between the user agent and the trading agent.

3. Execution time. This evaluates the overall system responsiveness.

6.2 Results

Experimental results demonstrate that the system successfully completed all 30 trading attempts, achieving a 100% success rate. The conversation example can be seen in Fig. 2. Each transaction requires an average of 2.9 interaction rounds, with the total experimental duration of approximately 5892 s.

Among the 30 tests, 26 trades (86.7%) are executed directly through limit orders, while 4 trades are automatically converted to market orders after failing to reach expected prices within the 20-min window. This result indicates that the system can fulfill user expectations within reasonable timeframes in most cases, while the market order fallback mechanism effectively guarantees the final trade execution.

Notably, the execution time per trade reflects the system's need for thorough deliberation and validation when processing complex trading decisions. This duration primarily includes market data analysis, the formulation of trading strategies, risk assessment, and the confirmation of multiround interactions. Compared to traditional manual trading processes, the system achieves relatively efficient automated trade execution while maintaining a high success rate.

7 Discussion

From the perspective of engineering application, the results of this study not only have theoretical significance but also provide direct references for system deployment.

In terms of potential benefits, the improvement of the LLMs' accuracy in parsing and the enhancement of information integrity mean that more trading instructions can be directly processed without the need for manual review. This will reduce the frequency of manual intervention, lower labor costs, and improve the overall processing efficiency and response speed, enabling the trading system to maintain accuracy while having higher throughput capacity.

In terms of risk scenarios, experiments also reveal that when the LLMs process input with high noise or ambiguous expressions, "hallucination" problems may still

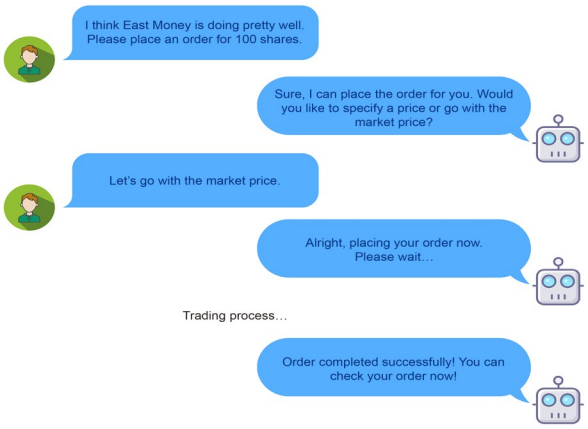

Fig. 2 Agent conversation. This is an example of how agents communicate and finish the trading process

occur. This could potentially lead to incorrect trading directions, quantities, or prices in the automatically executed scenarios, resulting in capital losses or compliance risks. Therefore, in the actual deployment, necessary protective measures should be set based on these experimental observations, such as adding an artificial review step for high-value or abnormal trading instructions, introducing rule verification and confidence thresholds for key fields, and setting limits on amounts or frequencies, to enjoy the efficiency benefits brought by automation while minimizing potential risks.

8 Conclusions

Our findings highlight three key implications for the field. First, while LLMs offer a powerful tool for automating trade order conversion, their outputs necessitate rigorous validation to mitigate risks from misinformation or over-interrogation. Second, the proposed intelligent pipeline—combining structured data generation, proactive information supplementation, and alignment with financial logic—provides a viable pathway to bridge human–AI collaboration in trading systems. Third, the dataset and metrics introduced in this work address a critical gap in benchmarking LLMs for domain-specific financial tasks, enabling future research to refine model performance in the real-world scenarios. In the future, we will improve the quality and robustness of datasets continuously and evaluate LLMs in finance from more diverse aspects.

Contributors

Yu KANG designed the research, managed the project, performed the system simulation, and prepared the dataset. Xin YANG conducted the investigation, and collected and processed the data. Ge WANG and Yuda WANG carried out the experiments and data analysis. Yu KANG, Xin YANG, Ge WANG, and Yuda WANG drafted the paper. Zhanyu WANG and Mingwen LIU revised the paper. Mingwen LIU provided the project administration and funding acquisition. All the authors contributed to the revision and finalization of the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding authors upon reasonable request.

References

Aghion P, Bergeaud A, Boppart T, et al., 2018. Firm dynamics and growth measurement in France. *J Eur Econ Assoc*, 16(4): 933-956. https://doi.org/10.1093/jeea/jvy031

Agrawal A, Kedia N, Panwar A, et al., 2024. Taming throughput-latency tradeoff in LLM inference with Sarathi-serve. Proc 18th USENIX Symp on Operating Systems Design and Implementation, p.117-134.

Bollen J, Mao HN, Zeng XJ, 2011. Twitter mood predicts the stock market. *J Comput Sci*, 2(1):1-8. https://doi.org/10.1016/j.jocs.2010.12.007

Cavalcante RC, Brasileiro RC, Souza VLF, et al., 2016. Computational intelligence and financial markets: a survey and future directions. *Expert Syst Appl*, 55:194-211. https://doi.org/10.1016/j.eswa.2016.02.006

Chen YS, Cheng CH, Tsai WL, 2014. Modeling fitting-function-based fuzzy time series patterns for evolving stock index forecasting. *Appl Intell*, 41(2):327-347. https://doi.org/10.1007/s10489-014-0520-6

Day MY, Lee CC, 2016. Deep learning for financial sentiment analysis on finance news providers. Proc IEEE/ACM Int Conf on Advances in Social Networks Analysis and Mining, p.1127-1134. https://doi.org/10.1109/ASONAM.2016.7752381

DeepSeek-AI, 2025. DeepSeek-V3 technical report. https://arxiv.org/abs/2412.19437

di Persio L, Honchar O, 2017. Recurrent neural networks approach to the financial forecast of Google assets. *Int J Math Comput Simul*, 11:7-13.

Dolphin R, Dursun J, Chow J, et al., 2024. Extracting structured insights from financial news: an augmented LLM driven approach. https://arxiv.org/abs/2407.15788

Fischer T, Krauss C, 2018. Deep learning with long short-term memory networks for financial market predictions. *Eur J Oper Res*, 270(2):654-669. https://doi.org/10.1016/j.ejor.2017.11.054

Fons E, Dawson P, Zeng XJ, et al., 2020. Evaluating data augmentation for financial time series classification. https://arxiv.org/abs/2010.15111

Garimella A, Chiticariu L, Li YY, 2021. Domain-aware dependency parsing for questions. Proc Findings of the Association for Computational Linguistics, p.4562-4568. https://doi.org/10.18653/v1/2021.findings-acl.400

Han JG, Barman U, Hayes J, et al., 2018. NextGen AML: distributed deep learning based language technologies to augment anti money laundering investigation. Proc ACL System Demonstrations, p.37-42. https://doi.org/10.18653/v1/P18-4007

Huang JM, Xiao MX, Li D, et al., 2025. Open-FinLLMs: open multimodal large language models for financial applications. https://arxiv.org/abs/2408.11878

Kim GI, Chung K, 2024. Extraction of features for time series classification using noise injection. *Sensors*, 24(19):6402. https://doi.org/10.3390/s24196402

Kim M, Yoo S, Park S, 2025. A rule-based stock trading recommendation system using sentiment analysis and technical indicators. *Electronics*, 14(4):773. https://doi.org/10.3390/electronics14040773

Kognole S, 2024. Fix protocol: the backbone of financial trading. *Int J Comput Sci Inf Technol*, 16(4):97-114. https://doi.org/10.5121/IJCSIT.2024.16408

Li BL, Jiang YK, Gadepally V, et al., 2024. LLM inference

serving: survey of recent advances and opportunities. Proc IEEE High Performance Extreme Computing Conf, p.1-8. https://doi.org/10.1109/HPEC62836.2024.10938426

Li CL, Shi Y, Luo YY, et al., 2025. Will LLMs be professional at fund investment? DeepFund: a live arena perspective. https://arxiv.org/abs/2503.18313

Li JT, Bian YX, Wang GX, et al., 2023. CFGPT: Chinese financial assistant with large language model. https://arxiv.org/abs/2309.10654

Li YH, Wang SF, Ding H, et al., 2023. Large language models in finance: a survey. Proc 4th ACM Int Conf on AI in Finance, p.374-382. https://doi.org/10.1145/3604237.3626869

Li ZN, Huang YP, Li ZY, et al., 2023. Neuro-symbolic learning yielding logical constraints. Proc 37th Int Conf on Neural Information Processing Systems, Article 946.

Long L, Wang R, Xiao RX, et al., 2024. On LLMs-driven synthetic data generation, curation, and evaluation: a survey. Proc Findings of the Association for Computational Linguistics, p.11065-11082. https://doi.org/10.18653/v1/2024.findings-acl.658

Malo P, Sinha A, Korhonen P, et al., 2014. Good debt or bad debt: detecting semantic orientations in economic texts. *J Assoc Inform Sci Technol*, 65(4):782-796. https://doi.org/10.1002/asi.23062

Min BH, Borch C, 2022. Systemic failures and organizational risk management in algorithmic trading: normal accidents and high reliability in financial markets. *Soc Stud Sci*, 52(2):277-302. https://doi.org/10.1177/03063127211048515

Nourbakhsh A, Bang G, 2019. A framework for anomaly detection using language modeling, and its applications to finance. https://arxiv.org/abs/1908.09156

OpenAI, 2023. GPT-4 technical report. https://arxiv.org/abs/2303.08774

OpenAI, 2024a. GPT-4o mini: Advancing Cost-Effective Intelligence. https://openi.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence [Accessed on Sept. 1, 2025].

OpenAI, 2024b. Hello GPT-4o. https://openai.com/index/hello-gpt-4o [Accessed on Sept. 1, 2025].

Qwen Team, 2024. Introducing Qwen1.5. https://qwenlm.github.io/blog/qwen1.5 [Accessed on Sept. 1, 2025].

Sanh V, Webson A, Raffel C, et al., 2022. Multitask prompted training enables zero-shot task generalization. Proc Int Conf on Learning Representations.

Sawhney R, Wadhwa A, Agarwal S, et al., 2021. Quantitative day trading from natural language using reinforcement learning. Proc Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, p.4018-4030. https://aclanthology.org/2021.naacl-main.316

Schuster M, Paliwal KK, 1997. Bidirectional recurrent neural networks. *IEEE Trans Signal Process*, 45(11):2673-2681. https://doi.org/10.1109/78.650093

Sinha A, Khandait T, 2021. Impact of news on the commodity market: dataset and results. Proc Future of Information and Communication Conf on Advances in Information and Communication, p.589-601. https://doi.org/10.1007/978-3-030-73103-8_41

Sinha A, Kedas S, Kumar R, et al., 2022. SEntFiN 1.0: entity-aware sentiment analysis for financial news. *J Assoc Inform Sci Technol*, 73(9):1314-1335. https://doi.org/10.1002/asi.24634

Sinha A, Agarwal C, Malo P, 2025. FinBloom: knowledge grounding large language model with real-time financial data. https://arxiv.org/abs/2502.18471

Sohangir S, Wang DD, Pomeranets A, et al., 2018. Big data: deep learning for financial sentiment analysis. *J Big Data*, 5(1):3. https://doi.org/10.1186/s40537-017-0111-6

Tatarinov N, Sukhani S, Shah A, et al., 2025. Language modeling for the future of finance: a survey into metrics, tasks, and data opportunities. https://doi.org/10.48550/arXiv.2504.07274

Vaswani A, Shazeer N, Parmar N, et al., 2017. Attention is all you need. Proc 31st Int Conf on Neural Information Processing Systems, p.6000-6010.

Wang K, Zhu JH, Ren MJ, et al., 2024. A survey on data synthesis and augmentation for large language models. https://doi.org/10.48550/arXiv.2410.12896

Wei L, Wang ZY, Xu J, et al., 2023. A lightweight sentiment analysis framework for a micro-intelligent terminal. *Sensors*, 23(2):741. https://doi.org/10.3390/s23020741

Xiao YJ, Sun E, Luo D, et al., 2024. TradingAgents: multi-agents LLM financial trading framework. https://doi.org/10.48550/arXiv.2412.20138

Xie Q, Han W, Chen Z, et al., 2024. FinBen: a holistic financial benchmark for large language models. https://doi.org/10.48550/arXiv.2402.12659

Yang Y, Uy MCS, Huang A, 2020. FinBERT: a pretrained language model for financial communications. https://arxiv.org/abs/2006.08097

Yao MH, 2025. Robustness of big language modeling in finance. *ITM Web Conf*, 70:02003. https://doi.org/10.1051/itmconf/20257002003

Young A, Chen B, Li C, et al., 2025. Yi: open foundation models by 01.AI. https://arxiv.org/abs/2403.04652

Yu W, Hamam H, 2024. Architecting an enterprise financial management model: leveraging multi-head attention mechanism-Transformer for user information transformation. *Peer J Comput Sci*, 10:e1928. https://doi.org/10.7717/peerj-cs.1928

Zhang Y, Jin R, Zhou ZH, 2010. Understanding bag-of-words model: a statistical framework. *Int J Mach Learn Cyber*, 1(1):43-52. https://doi.org/10.1007/s13042-010-0001-0

Zhou ZY, Mehra R, 2025. An end-to-end LLM enhanced trading system. https://doi.org/10.48550/arXiv.2502.01574

List of supplementary materials

1 Raw research report examples

2 User interaction example

3 UI interface

4 Prompt

Fig. S1 Execution workflow example

Fig. S2 User interface of the intelligent agent trading system

Fig. S3 Core task prompt defining the trading agent's role and required data fields

Fig. S4 Example trading instruction prompt with context and stop-loss order

Fig. S5 Financial reasoning prompt for step-by-step analysis

Fig. S6 Extended trading instruction prompt variant