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

Position Paper:

Quant 4.0: engineering quantitative investment with automated, explainable, and knowledge-driven artificial intelligence

Jian GUO[§‡1,3], Saizhuo WANG[§1,2], Lionel M. NI[2,3], Heung-Yeung SHUM[‡1,2]

[1]*IDEA Research, International Digital Economy Academy, Shenzhen 518045, China*

[2]*The Hong Kong University of Science and Technology, Hong Kong 999077, China*

[3]*The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China*

E-mail: guojian@idea.edu.cn; swangeh@connect.ust.hk; ni@ust.hk; hshum@idea.edu.cn

Received Oct. 20, 2023; Revision accepted Feb. 19, 2024; Crosschecked Aug. 20, 2024

Abstract: Quantitative investment (abbreviated as "quant" in this paper) is an interdisciplinary field combining financial engineering, computer science, mathematics, statistics, etc. Quant has become one of the mainstream investment methodologies over the past decades, and has experienced three generations: quant 1.0, trading by mathematical modeling to discover mis-priced assets in markets; quant 2.0, shifting the quant research pipeline from small "strategy workshops" to large "alpha factories"; quant 3.0, applying deep learning techniques to discover complex nonlinear pricing rules. Despite its advantage in prediction, deep learning relies on extremely large data volume and labor-intensive tuning of "black-box" neural network models. To address these limitations, in this paper, we introduce quant 4.0 and provide an engineering perspective for next-generation quant. Quant 4.0 has three key differentiating components. First, automated artificial intelligence (AI) changes the quant pipeline from traditional hand-crafted modeling to state-of-the-art automated modeling and employs the philosophy of "algorithm produces algorithm, model builds model, and eventually AI creates AI." Second, explainable AI develops new techniques to better understand and interpret investment decisions made by machine learning black boxes, and explains complicated and hidden risk exposures. Third, knowledge-driven AI supplements data-driven AI such as deep learning and incorporates prior knowledge into modeling to improve investment decisions, in particular for quantitative value investing. Putting all these together, we discuss how to build a system that practices the quant 4.0 concept. We also discuss the application of large language models in quantitative finance. Finally, we propose 10 challenging research problems for quant technology, and discuss potential solutions, research directions, and future trends.

Key words: Artificial general intelligence; Artificial intelligence; Automated machine learning; Causality engineering; Deep learning; Feature engineering; Investment engineering; Knowledge graph; Knowledge reasoning; Knowledge representation; Model compression; Neural architecture search; Quant 4.0; Quantitative investment; Risk graph; Explainable artificial intelligence

https://doi.org/10.1631/FITEE.2300720 **CLC number:** TP391

1 Introduction

Quantitative investment (abbreviated as "quant" in this paper) is an important part of the wealth management industry, which is one of the largest sectors of the world's economy. Modern

§ These two authors contributed equally to this work
‡ Corresponding authors
ORCID: Jian GUO, https://orcid.org/0009-0003-5046-2588

quant applies rigorous mathematical and statistical modeling techniques, machine/deep learning techniques, and algorithmic trading techniques to discover asset pricing abnormalities in financial markets and profit from the following arbitrage or investment opportunities. The core of quant trading is quant strategy, which is a systematic function or trading methodology used in financial markets based on predefined rules or trained models for making trading decisions. As shown in Fig. 1, a standard quant strategy contains a series of components, such as an investment instrument, trading frequency, trading mode, strategy type, and data type. For example, a stock hedging strategy hedges market risk by going long in the most favorable stocks and shorting the least favorable ones (or shorting the corresponding index future/option instead).

Quant has been studied for a long time in both academia and industry. Originating a century ago (Bachelier, 1900), many researchers have contributed to the development of this interdisciplinary area (Fig. 2), including many Nobel Prize laureates and Turing award winners. Examples include modern portfolio theory by Markowitz (1952), stochastic calculus for asset pricing by Samuelson (1965), an option pricing model by Black and Scholes (1973), vector autoregression by Sims (1980), the ARCH/GARCH model and co-integration test by Engle (1982) and Engle and Granger (1987), the Fama–French three-factor model by Fama and French (1992), local average treatment effect estimation by Imbens and Angrist (1994), Shapley value in cooperative game theory by Shapley (1953), and deep learning technology by LeCun et al. (2015). On the other hand, the blooming era of quant in industry started in the 1990s, and its evolution can be generally categorized into three generations, denoted as quant 1.0–3.0 (Fig. 3). Quant 1.0 involves a small but elite team applying mathematical and statistical tools to analyze financial markets; we call this the "alpha studio" model. Quant 2.0 involves a large number of investment researchers working on a standardized pipeline to find effective alpha factors (Tulchinsky, 2019) out of the plethora of financial data; we call this the "alpha factory" model. Quant 3.0 moves its attention from factor mining to deep learning modeling, and aims to achieve strong prediction models with simple factors by leveraging the powerful end-to-end learning ability of deep neural networks; we call this the "deep alpha" model.

Although quant 3.0 has demonstrated its success in some strategy scenarios, it still has some primary limitations. First, building a "good" deep neural network is costly because of the heavy work in network design, optimization, deployment, and maintenance. Second, it is a challenge to read understandable messages from a model encoded by deep learning black boxes, making it very unfriendly to investors and quant researchers. Third, the good performance of deep learning relies heavily on extremely large volumes of data, limiting its usage to data-abundant scenarios such as high-frequency trading. In response to these challenges, we propose the idea of quant 4.0, which defines the picture of next-generation quant technology by practicing the "all-in on artificial intelligence (AI)" philosophy. Specifically, the proposed quant 4.0 concept consists of three fundamental pillars:

1. Automated AI aims to build an end-to-end automated pipeline for quant research and trading, to dramatically improve efficiency and sustainability by significantly reducing the cost of labor and time for quant research work including data preprocessing, feature engineering, model construction, and model deployment. In particular, we introduce the state-of-the-art AutoML (He et al., 2021) techniques to automate every module in the strategy development pipeline. In this way, we propose to change traditional hand-crafted modeling to an automated modeling workflow in an "algorithm produces algorithm, model builds model" manner, and eventually move towards a technical philosophy of "AI creates AI."

2. Explainable AI (XAI) attempts to open the black box that encapsulates deep learning models. Pure black-box modeling is unsafe for quant research because people cannot calibrate the risk accurately. It is difficult to know, for example, where returns come from and whether they rely on certain market styles, and what the reason for a specific drawdown is, under black-box modeling. More and more new techniques in the field of XAI could be applied in quant to enhance the transparency of machine learning modeling, and thus we recommend that quant researchers pay more attention to XAI. We must recognize that improving a model's explainability has costs. Therefore, it is important to research how to maximize valuable explanations for investment with

the least loss in performance possible.

3. Knowledge-driven AI differs from data-driven AI, which heavily depends on large volumes of data samples. Data-driven AI is more appropriate for high-frequency trading or stock cross-sectional trading, which has very large sample sizes. Conversely, knowledge-driven AI relies on reasoning from a knowledge base or knowledge graph, and has the potential to solve the quant problem in low-frequency trading such as value investing or global macro investing. Therefore, knowledge-driven AI is an important complement to data-driven AI techniques such as deep learning. In this paper, we introduce knowledge graph technology, which represents knowledge with a network structure composed of entities and relations, and stores knowledge with semantic triples. A knowledge graph of financial behaviors and events could be analyzed, and inferences could be made for investment decisions using symbolic reasoning and neural reasoning techniques.

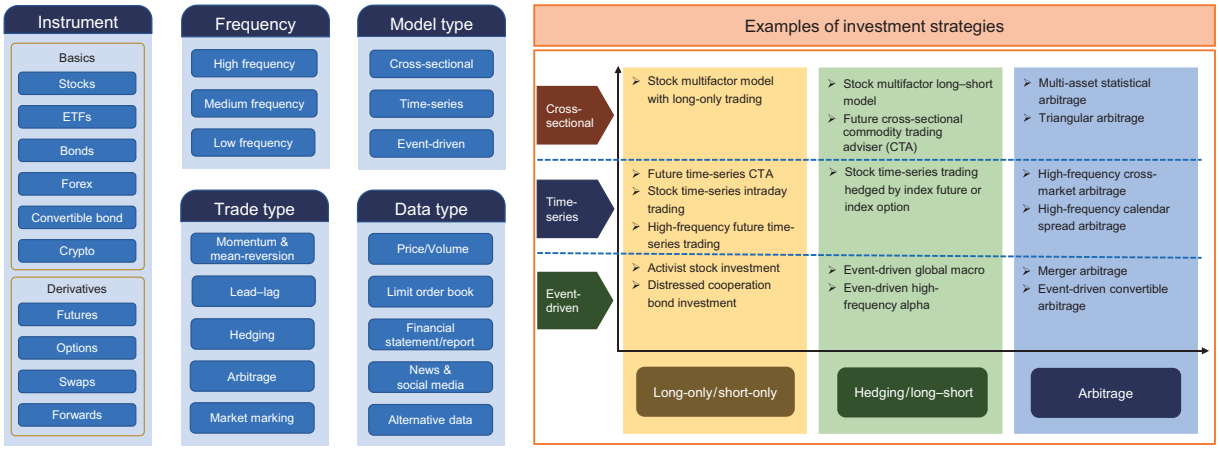


Fig. 1 Quant strategy components and classification of common strategies

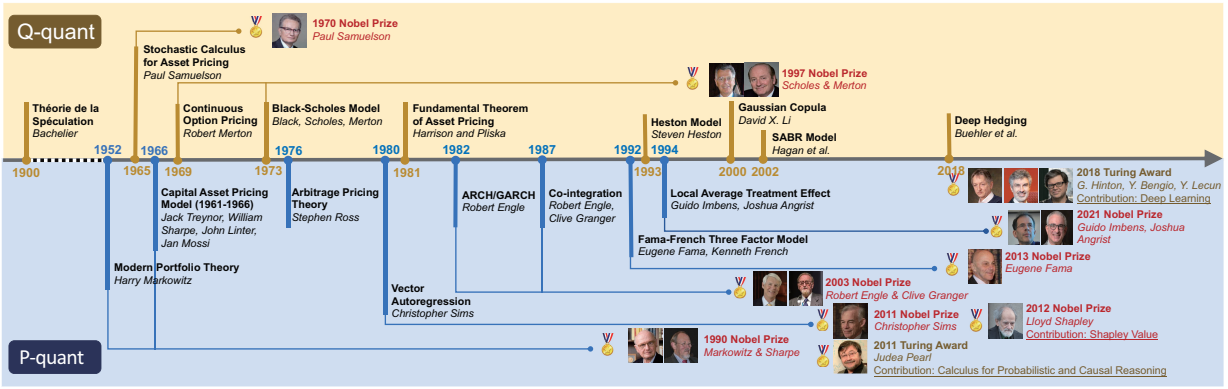


Fig. 2 Main academic contributors and their works that deeply influence the development of quantitative investment (photo credit: Wikipedia)

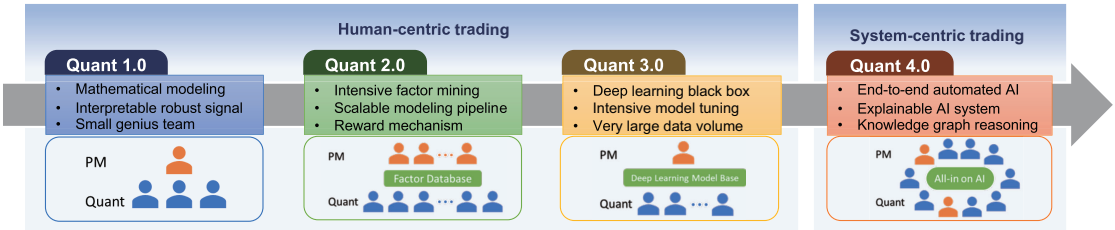


Fig. 3 The development history of quantitative investment in industry, from quant 1.0 to quant 4.0

Moreover, with the rapid development of large language models (LLMs), we study their application in quantitative investment and their potential uses.

2 Automated AI for quant 4.0

Automated AI for quant 4.0 covers the automation of the full quant pipeline. In this section, we will first give an overview of the traditional quant research pipeline and then introduce how to upgrade it to an automated AI pipeline.

2.1 Automating quant research pipeline

Over decades of development, quant research has formed a standard workflow as shown in Fig. 4 (blue part). This workflow consists of several modules, including data pre-processing, factor mining, modeling, portfolio optimization, order execution, and risk analysis. Data pre-processing is intended to standardize the raw data and improve modeling efficiency by resolving issues in raw data, such as missing records, extreme values, outliers, and differences in scales. Factor mining is a task of feature engineering (Zheng and Casari, 2018) that uses financial and economic domain knowledge to design, search for, or extract financial factors (features for downstream modeling) from raw data. Modeling is the task of building statistical or machine learning models that use factors to predict market trends, asset price movements, best trading times, or most/least valuable assets. These models are evaluated through back-test experiments. The final choices of models must consider a number of factors including accuracy, explainability, and robustness, and find the best tradeoff according to the ultimate goal. Portfolio optimization aims to find the optimal asset allocation to simultaneously expect high return and low risk. Although prediction models tell us what or when to buy/sell, portfolio optimization specifies how much to buy/sell. A typical portfolio optimizer attempts to solve a constrained convex quadratic programming problem which is extended from Markowitz's efficient frontier theory (Markowitz, 1952). Order execution is the task of buying or selling orders with optimal prices and minimum market impact. To minimize the market fluctuation introduced by big orders, algorithmic trading provides a series of mathematical tools for order splitting, from the simplest time-weighted average price (TWAP) and volume-weighted average price (VWAP) to the complicated reinforcement learning methods (Nevmyvaka et al., 2006), in which optimal order flow is modeled as a (partially observable) Markov decision process. Risk analysis is the task of discovering and understanding risk exposure to better control unnecessary and harmful risks in quant research and trading (Coleman, 2011). In this step, risks are measured in real time and these messages and analyses are sent back to help quant researchers improve their strategies.

In quant 4.0, we propose an automated quant workflow using state-of-the-art AI technology, as shown in Fig. 4 (orange part). In the following part of this section, we will elaborate on three core modules in the automated pipeline.

1. Automated factor mining (Section 2.2) applies automated feature engineering techniques to search for and evaluate significant financial factors generated from meta factors.

2. Automated modeling (Section 2.3) applies AutoML techniques to discover optimal deep learning models and automatically select the best model configuration and training objective.

3. Automated one-click deployment (Section 2.4) builds an automated workflow to deploy trained large models on trading servers with limited computing power and latency budgets.

2.2 Automated factor mining

Traditionally, financial factors with significant "alpha" are manually explored and developed by quant researchers based on their domain expertise and comprehensive knowledge of financial markets. In quant 4.0, we propose to automate the factor mining process by formulating feature engineering as a search problem and using algorithms to generate factors with satisfactory back-test performance at scale (Fig. 5). In particular, based on the forms of expression, we classify factors as (1) symbolic factors (Kakushadze, 2016), which are symbolic equations or symbolic rules, and (2) machine learning factors, which are expressed by neural networks.

Symbolic factor mining can be regarded as a special case of symbolic regression (La Cava et al., 2021), and this problem consists of four parts: operand space, operator space, search algorithm, and evaluation criterion. The operand space defines which meta factors could be used for factor mining. Typical meta factors include basic price and volume information,

sector categorizations, basic features extracted from limit order books, and market sentiment scores (Rashid et al., 2019; Abdul Karim et al., 2022). The operator space defines which operators could be used in the factor mining process. For example, in cross-sectional stock selection, the operators could be classified as main operators for constructing symbolic factors and post-processing operators for standardizing the factors for different trading environments. Search algorithms aim to search for and find effective or qualified factors as efficiently as possible. Viable algorithms include Monte Carlo algorithms (Jin Y et al., 2020), genetic programming (Chen TX et al., 2021), and gradient-based methods (Biggio et al.,

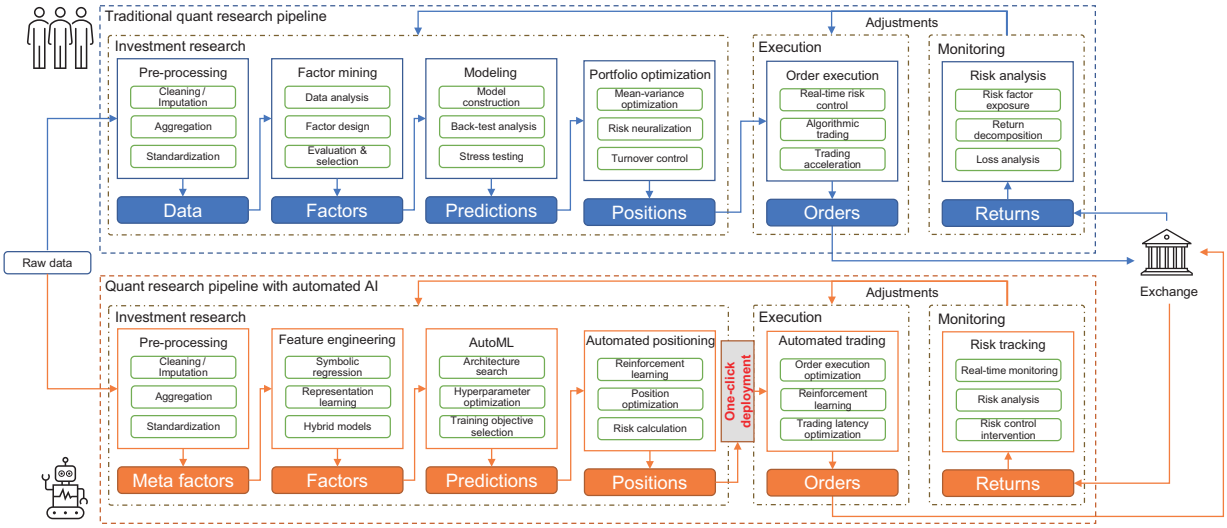


Fig. 4 A prototypical workflow of quantitative investment with comparisons between the current quantitative investment system (manual, upper blue part) and AI investment engineering (automated, lower orange part). References to color refer to the online version of this figure

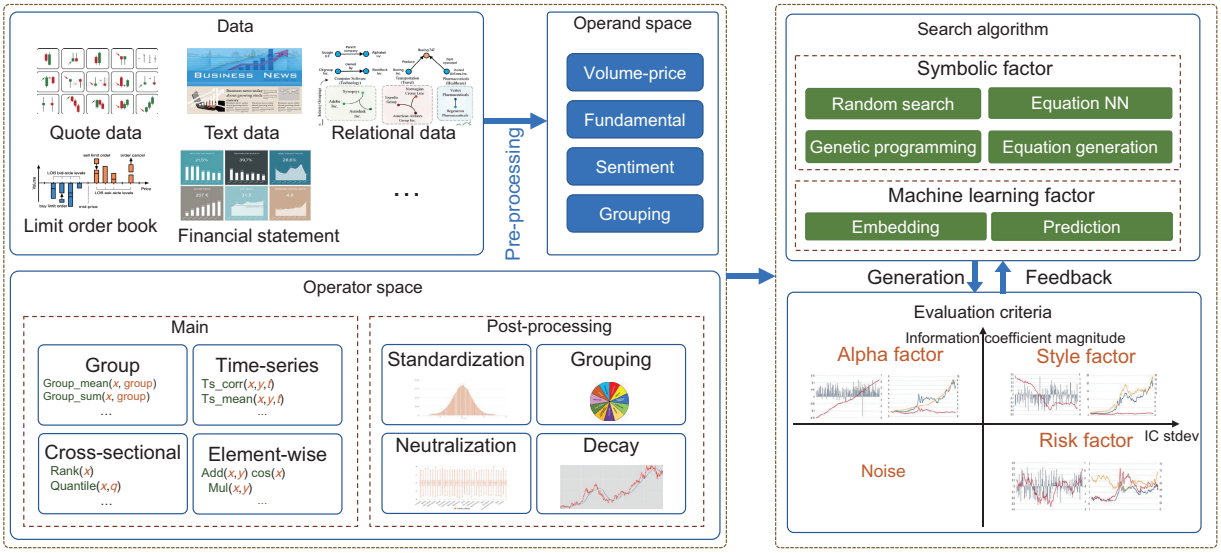


Fig. 5 An example factor mining pipeline. The search space is defined by operators and meta-factors, where meta-factors are extracted from raw data in various forms. The search space is explored by search algorithms that are discrete or continuous. The evaluation module provides feedback to search algorithms based on certain criteria that serve as guidance for the next search iteration. Parts of this figure are cited from Feng et al. (2019), Sawhney et al. (2020), and Wu YF et al. (2021)

2021). Evaluation criteria measure the quality of factors found by search algorithms. Typical evaluation criteria include the information coefficient (IC), information ratio based on information coefficient (ICIR), and risk-adjusted returns. It is also important to keep information diverse among factors by filtering out redundant factors that are highly correlated with others.

Although symbolic factors have their advantages in simplicity and understandability, their representation ability is limited by the richness of operands and operators. Machine learning factors, on the other hand, have more flexibility in representation to fit more complicated nonlinear relationships (Hornik et al., 1989), and thus they have the chance to perform better in market prediction. In particular, mining machine learning factors is equivalent to the process of training neural networks (Thakkar and Chaudhari, 2021), where gradients provide optimal direction for fast search of solutions. Most deep neural networks for stock prediction follow the encoder–decoder architecture (Sutskever et al., 2014), where the encoder maps meta factors to a latent vector representation (embedding), and the decoder transforms this embedding to some outcome such as future return (Wang JY et al., 2019). Not only the final outcome, but also the embedding itself, could be used as a (high-dimensional) machine learning factor (Wang ZC et al., 2021), and further applied to various downstream tasks.

2.3 Automated modeling

Automation of deep learning is a complex problem due to the end-to-end property and network architecture issues in modeling. The configuration of a deep learning model consists of three parts, architecture, hyperparameters, and training objectives, and they jointly determine the final performance of the model. Traditionally, these configurations are tuned manually. In quant 4.0, they are searched for and optimized using various AutoML (He et al., 2021) algorithms. A standard AutoML system needs to answer the following three questions: what to search for (i.e., search space), how to search (i.e., search algorithm), and why to search (i.e., performance evaluation).

Search space is designed from the perspectives of the three configuration components mentioned above. Specifically, network architecture is configured in a hierarchical way with various granularities, ranging from low-level operators, such as convolution kernels, to high-level modules, such as self-attention layers. Hyperparameters, such as learning rate and batch size, control the overall training process. The search space for hyperparameters is simpler than that for architecture since most hyperparameters are continuous (e.g., learning rate) or approximately continuous values (e.g., batch size). Training objectives specify the loss functions and labels used for training models. In addition to classic loss functions such as mean square loss and cross-entropy loss, new loss functions specifically designed for quant tasks can be selected. Labels define the "ground-truth" target that the model aims to fit, such as price raise/fall or future returns in different holding periods.

Given the search space, we could use various search algorithms to find the best model configuration. As the simplest methods, grid and random search algorithms (Bergstra and Bengio, 2012) are straightforward to implement and parallelize. However, they cannot scale well to a high-dimensional search space, because the number of potential configurations grows exponentially with the increase of the number of hyperparameters. An evolutionary algorithm (Real et al., 2017) uses evolution mechanisms to improve model configurations iteratively. It encodes the architecture of neural networks as a population and performs evolution steps on them to improve the model iteratively. Reinforcement learning (Zoph and Le, 2017) models the architecture search problem as a Markov decision process. In each step, a controller chooses an action to sample a new architecture. Then the corresponding model is trained and its performance is used as a reward to update the controller. This loop is iterated until convergence. Bayesian optimization (Falkner et al., 2018) explores the search space using surrogate models to approximate the black-box objective function. Specifically, it initializes a prior distribution using a surrogate function such as a Gaussian process. Then it samples new data points from the prior distribution (with importance), calculates their values using the underlying objective function, and updates the surrogate function accordingly. This process is repeated until the optimal solution is found. Gradient-based methods (Liu HX et al., 2019) are very efficient when the gradient of the objective function exists.

However, for automated modeling, the search space is usually discrete and the gradient cannot be defined directly. One solution is to "soften" the architecture and define an over-parameterized "super-architecture" which covers all possible candidates and is end-to-end differentiable (Liu HX et al., 2019).

The computational cost of an automated model search is a result of two factors—the search algorithm and model evaluation. Model evaluation is usually a bottleneck in the computation because it is very time-consuming to train a deep neural network with a specific configuration to convergence. Several methods have been introduced in previous research to address this issue. First, the neural network training process can be stopped before convergence to reduce the evaluation computation time (Zoph et al., 2018). Second, the model can select fewer samples to accelerate the training process (Klein et al., 2017). Third, warm-start model training can be used to leverage the information from existing selected models (Liu CX et al., 2018) or information can be inherited from an over-parameterized "parent" model (Liu HX et al., 2019) to accelerate the search loop.

2.4 Automated one-click deployment

Model deployment is the task of transferring the developed model from offline research to online trading. However, it is not simply a matter of transferring code and data; it also involves synchronizing data and factor dependency, adapting trading servers and systems, debugging model inference, testing computing latency, and so on. In what follows, we focus on accelerating deep learning inference for high-frequency trading and algorithmic trading scenarios, where we propose an automated one-click deployment solution that uses techniques such as model compilation and model compression.

During the development stage, deep learning model functionality is the top priority for the underlying framework, which strictly maps all the operations to the computation graph. However, such direct mapping introduces much room for optimization at the deployment stage where the computations are fixed. As a result, the model's computation can be simplified and adapted to hardware features without compromising its original semantics. Such optimization, which can be categorized as front-end optimization and back-end optimization, is one of the major topics in deep learning compilers (Li MZ et al., 2021). Model compression (Cheng Y et al., 2018) aims to reduce model size for inference acceleration while minimizing drops in performance. In this way, the compressed model can be regarded as an approximation of the original model. At the micro level, model pruning (Han S et al., 2015) and model quantization (Han S et al., 2016) techniques can be applied to reduce both the number of parameters and the bit size of the individual parameters. Model pruning removes unimportant connections and neurons in neural networks that have little influence on the activation of the neural network. Model quantization converts the parameters from floating-point numbers to low-bit representations. At the macro level, the model can be significantly compressed into a smaller model with simpler architectures via knowledge distillation (Hinton et al., 2015) and low-rank factorization (Zhang XY et al., 2015; Yu et al., 2017).

3 XAI for quant 4.0

XAI (Murdoch et al., 2019), as an attractive research direction for decades, is critical to ensuring the trustworthiness and robustness of AI models. For quant, improvement in the explainability of AI can make the decision-making process more transparent and easier to analyze, provide insights to researchers and investors, and uncover potential risk exposures. In this section, we discuss how to leverage XAI in quant 4.0 and connect these techniques to real quant scenarios using stock alpha investment as an example. We decompose XAI tasks into three dimensions, stock, time, and factors, and show how XAI can be applied to enhance the interpretability of quant models.

3.1 Explanation on stock

Explanations can be provided for individual stocks to illustrate their sensitivity to different factors at different times and their relationships with each other. This can be achieved through various XAI tasks, such as stock similarity analysis, lead-lag effect identification, and sector trend evaluation.

1. Stock similarity: The ubiquity of correlations between stocks and the commonalities shared among correlated stocks present an opportunity for leveraging XAI in quant modeling. By incorporating the relationships between financial instruments, we can improve the accuracy of our analysis and predictions

beyond traditional methods that treat stocks individually. Furthermore, analyzing the similarities between stock embeddings allows us to gain insight into what the model has learned. However, the challenge lies in selecting an appropriate similarity metric that is both flexible and effective. Metric learning (Kulis, 2013; Kaya and Bilge, 2019) and graph structure learning (Zhu et al., 2021) are areas of research that are relevant to solving this challenge. Developing a good similarity metric among stock embeddings can enable creation of a graph structure by computing an adjusted adjacency matrix based on pairwise similarity.

2. Lead-lag effect: In a lead-lag effect (Hou, 2007) (Fig. 6a), the trend of a stock is followed by some other stocks with a lag in time. Investors can profit by precisely identifying lead-lag effects on the market (Li YL et al., 2022). However, identification of lead-lag effects is difficult, because duplicated trends appear frequently in financial markets but only few of them are actually caused by lead-lag effects. Strict identification of lead-lag effects needs to be conducted via causal inference, which requires counterfactual explanations (Bottou et al., 2013). Nevertheless, counterfactual reasoning is usually infeasible in real-world financial markets because the analysis is based solely on historical data.

3. Sector trends: Sectors are groupings of stocks based on common characteristics, such as industry or market capitalization. The trend of individual stocks can be influenced by their sector, making it important to identify the contributions of sectors to individual stocks. One way to accomplish this is by treating a stock's sector membership as a categorical feature and using feature importance algorithms to determine its importance. Additionally, investors can gain insight into the sensitivity of sectors to different types of features by examining the interactions between sector memberships and other ordinary features. This can help in making more informed investment decisions and understanding how the model makes predictions. In the context of XAI, interpreting the role of sectors can provide valuable explanations for the model's output.

3.2 Explanation on time

Explanations can be computed on individual time points to illustrate a stock's situation and factors at that specific cross-section, and explanations across multiple cross-sections can be further combined to provide insights for market features in a time interval.

1. Extreme market: In stock markets, there are extreme conditions where nearly all stocks on the market experience severe price drops. Under such circumstances, it is hard for quant strategies to obtain excess return because the prices of all stocks drop together, and there is little room for arbitrage. Therefore, in extreme markets, it becomes crucial to identify the stocks that are less affected by the market downturn and trade them to earn excess returns. To achieve this, we can decompose stock returns from two perspectives: those contributed by market trends and those contributed by stock-specific features. The decomposition can be computed by categorizing factors into market factors and stock-specific factors. Then, the importance of these two types of factors can be computed via feature importance algorithms. We need to select the stocks where the importance of stock-specific factors

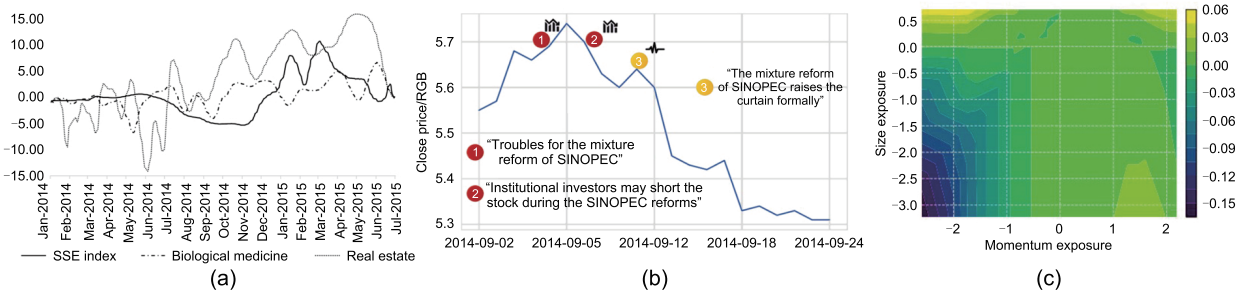


Fig. 6 XAI examples in quant: (a) lead-lag effect in Chinese stock market; (b) influence of breaking news across time; (c) interaction between size and momentum factors. (a) is reprinted from Guo K et al. (2017), Copyright 2017, with permission from Elsevier. (b) is reprinted from Hu et al. (2018), Copyright 2018, with permission from ACM. (c) is cited from Wang J et al. (2021)

outweighs that of market factors.

2. Calendar effect: Calendar effects (Sakalauskas and Kriksciuniene, 2009) are market patterns related to the calendar, such as days of the week, months of the year, and event-related periods such as the U.S. presidential cycle. They are caused by market participants' expectations of future development and can strongly influence market trends. Therefore, identifying and using calendar effects is important in quant for adjusting investment strategies. Feature importance algorithms can help identify these effects by computing the importance of calendar factors, such as categorical features for weekdays and days of the month. If model predictions heavily rely on these features, it may indicate potential calendar effects.

3. Style transition: Style factors are used in multiple factor models such as BARRA (MSCI, 1996) to describe the intrinsic features of stocks such as size, volatility, and growth. In such models, exposure of stocks to these style factors contributes to their returns, and the return contribution per unit exposure, also called factor return, differs across style factors. Moreover, the return of each factor changes over time because of the transitions in the market's preference for different styles. If such transitions can be accurately recognized, investors can adjust their strategies accordingly to focus on stocks with large exposures to the dominant style factors. To detect style transitions, we can regard style exposures as factors and compute their contribution to stock returns using feature importance algorithms. We can then observe the distribution of factor contributions across time and detect shifts in this distribution as signals for style transitions.

4. Event influence: Breaking events usually have a great influence on stock markets (Fig. 6b). Investors need to have a good understanding of the influences of breaking events to reduce the negative impacts or profit from the events. Usually, an event is associated with two pieces of information: occurrence time and specific contents. Event contents can be encoded as relevant features using natural language processing techniques (Hu et al., 2018), and the effect of an event can be computed as its importance concerning market trends after its occurrence. In addition, we may compute causal explanations to show the causal effect of the event.

3.3 Explanation on factors

Explanations can be computed on each factor to illustrate the sensitivity of different stocks to the factor at different times. The explanations can be further combined to show the interactive effects among factors for specific stocks.

1. Factor type: Factors can be categorized in various ways, such as by data source (e.g., volume-price factors, sentiment factors, and fundamental factors), financial features (e.g., momentum factors, mean-reversion factors, and lead-lag factors), and time scales (e.g., tick-, minute-, and day-level factors). Feature importance algorithms can be used to compute the contribution of different types of factors to portfolio returns, helping investors gain a better understanding of AI-generated investment strategies.

2. Factor interaction: Deep learning models are good at capturing the complicated associations between factors, allowing weak factors to be combined to form strong factors. Such interactions reflect intriguing patterns among factors and provide new insights into finding new factors, and feature crossing techniques (Luo et al., 2019; Tsang et al., 2020) can be used to reveal such interactions.

3. Factor hierarchy: The semantic similarity among factors can be depicted hierarchically, using techniques such as hierarchical clustering (Müllner, 2011) to create factor evolution graphs. These graphs show factor relations by grouping together factors with higher similarities in lower-order neighborhoods.

4 Knowledge-driven AI for quant 4.0

Knowledge-driven AI is an important complementary technology to data-driven AI, especially in low-frequency investment scenarios such as value investing and global macro investment. A knowledge-based system consists of two parts, a knowledge base and a knowledge reasoning engine (Hayes-Roth et al., 1983), which correspond to the problems of knowledge representation and knowledge reasoning, respectively.

4.1 Knowledge representation

The goal of knowledge representation is to encode human knowledge in machine-readable forms.

It is the foundation of knowledge-driven AI and it determines the modeling method for downstream knowledge reasoning tasks. The semantic network, first implemented by computer scientists in 1956 (Lehmann, 1992), is an early-stage knowledge base concept (Sowa, 1992). Since then, logic-based knowledge representations such as fuzzy logic (McNeill, 1993), expert systems (Jackson, 1998), and frame-based languages (Minsky, 1974) have been extensively studied and achieved great success. With the emergence of big data, various knowledge standards, such as the resource description framework (RDF) and Web ontology language (OWL), have been proposed to meet the growing demand for knowledge exchange. These techniques are further extended to satisfy large-scale applications in practice, leading to the development of knowledge graph techniques, which have become one of the mainstream methods for building knowledge-based systems in data-driven scenarios.

A knowledge graph typically consists of two parts: ontology and instances. The ontology serves as the knowledge graph schema, specifying the types and semantic meanings of the entities and relationships (Kendall, 2019). Meanwhile, instances denote the relationships between entities, usually represented as semantic triples that contain a subject, predicate, and object. In practice, knowledge graphs are often very large, containing millions or even billions of entities. As a result, knowledge acquisition techniques are necessary for automatically constructing the knowledge graph. This is typically achieved through knowledge extraction and knowledge graph completion (Ji et al., 2022).

For quant, a financial behavioral knowledge graph should capture information from three key aspects: (1) fundamental information about financial entities, (2) financial events taking place between these entities, and (3) the causal relationships between entities and events. This knowledge graph can represent a range of entities, including financial entities, concepts, and events, with relationships categorized as relationships between entities, relationships between events, and relationships between events and entities. Knowledge constituting a financial behavioral knowledge graph can be acquired from various sources, and the most challenging part is extracting useful structural knowledge from unstructured data. Specifically, the challenges include accurately understanding and extracting information from raw data, discovering facts from contradictory information, extrapolation from incomplete data, and aligning data with different update frequencies. Fig. 7 provides a fictitious example of a financial knowledge graph, where various entities such as public companies, securities, sectors, individuals, and events are characterized, along with relationships such as supply chain, capital chain, and behavioral relations. The information used to build this knowledge graph is drawn from diverse sources such as news reports, litigation documents, financial statements, research reports, and sectors, and extracted using natural

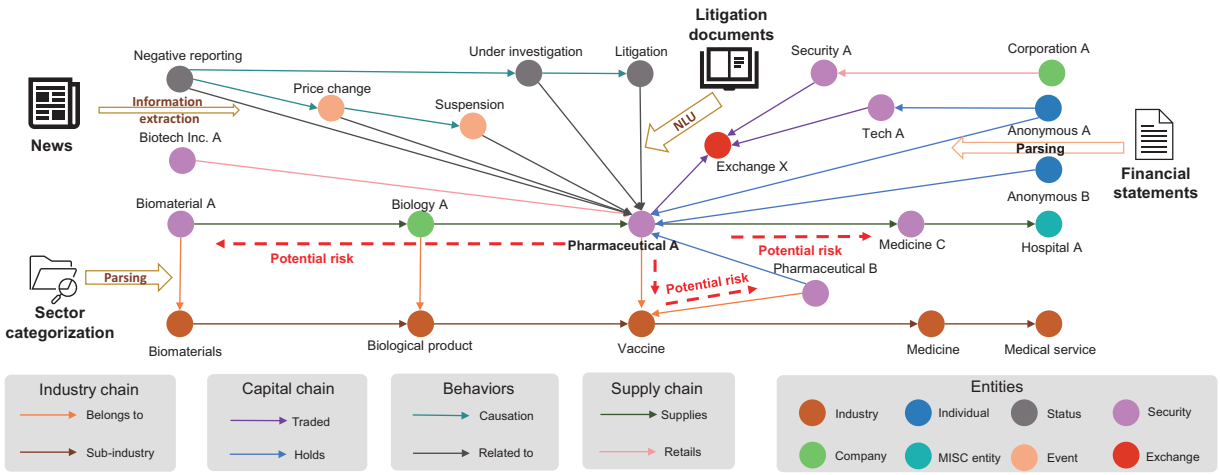


Fig. 7 An example of financial knowledge graph that contains behavioral information. All the financial entities and events are fictitious, only for illustration purposes. References to color refer to the online version of this figure

language processing techniques, including natural language understanding and information extraction.

Moreover, we can leverage the power of LLMs to perform knowledge extraction given some properly edited prompts. In this way, no specific models are needed while maintaining good extraction accuracy.

4.2 Knowledge reasoning

Knowledge reasoning involves analyzing, inferring, proving, and making decisions based on existing knowledge and data. Different methods can be used for reasoning, including symbolic logic methods, neural methods, and neuro-symbolic methods.

Symbolic reasoning can be performed deterministically or probabilistically. In deterministic symbolic reasoning, inference rules are applied to given facts recursively until the desired conclusion is reached. On the other hand, probabilistic symbolic reasoning (Ng and Subrahmanian, 1992; Richardson and Domingos, 2006) represents logic rules more flexibly by modeling the distribution of fact triples based on existing facts and rules. Neural reasoning methods employ deep learning models to learn decision rules with nonlinear associations. The model can be trained using a knowledge graph structure described by semantic triples and the attributes of entities and relationships. During the inference process, the trained model predicts the fact of the input semantic triple. To do this, entities and relationships are first represented by entities in embedding spaces (Bordes et al., 2013; Trouillon et al., 2016; Xiao et al., 2016). Then, the possibility of semantic triples holding true can be computed directly from these representations (Bordes et al., 2013), or leveraging the structural information of the knowledge graph (Guo LB et al., 2019). To further enhance reasoning capability, symbolic and neural reasoning methods can be combined through neurosymbolic reasoning. This can be achieved by either injecting logic structures such as ontological schemas into the embedding framework (Wang YZ et al., 2021), or using neural knowledge representations for logic reasoning (Rocktäschel and Riedel, 2017) and rule learning (Wang WY and Cohen, 2016).

For quant, the knowledge representations can be incorporated into existing factors as external information and used as input to deep learning models for better predictions. Fig. 8 demonstrates a typical pipeline of knowledge reasoning in quant. In this pipeline, events and relationships between stocks are represented as semantic triples, with entities and relationships embedded as vectors. Neural reasoning is then performed on these semantic triples to compute the embeddings of events, relationships, and the entire knowledge graph. After training on historical data, these knowledge representations can be used to generate trading decisions in investment strategies.

There have been several other studies investigating the use of knowledge reasoning in quantitative finance. For instance, Ding X et al. (2016) proposed a method to improve event embeddings by incorporating relational and categorical knowledge.

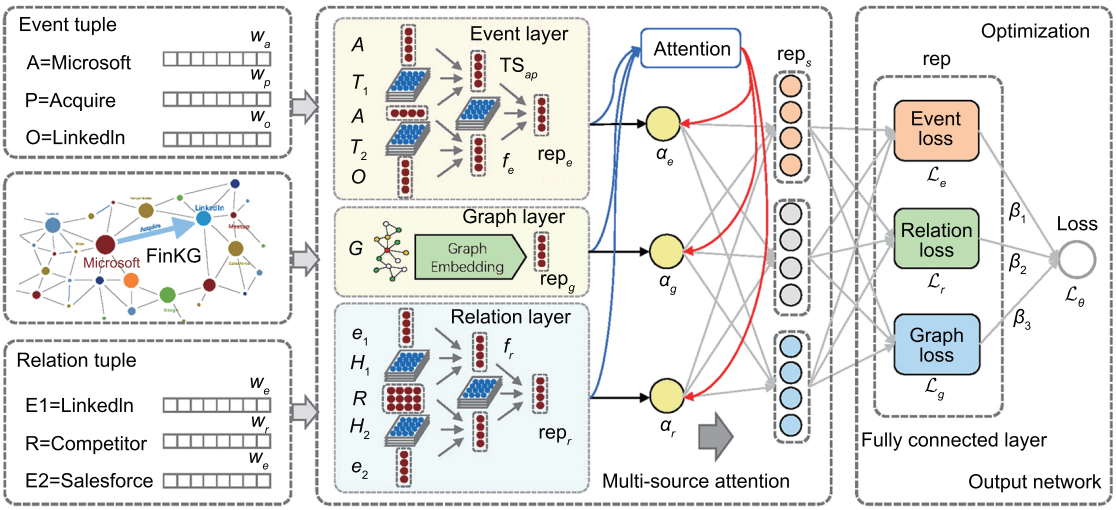


Fig. 8 Knowledge graph reasoning for stock prediction. Reprinted from Cheng DW et al. (2020), Copyright 2020, with permission from ACM

They retrieved external information about the entities in the semantic triple representing an event from a knowledge graph and used it to compute the event embedding. Similarly, Deng et al. (2019) extracted events from news texts and used entity linking techniques (Sil and Yates, 2013) to align the extracted information with the knowledge graph. They generated event embeddings using TransE and combined them with volume-price data in a temporal convolutional network (Bai et al., 2018) to predict stock prices. Feng et al. (2019) leveraged fundamental information, such as sector categorizations and supply chains, to build a knowledge graph, and used graph convolution to compute the embedding of each stock. They then used these embeddings to predict stock returns by minimizing the stock ranking loss. In another study, Long et al. (2020) used node2vec (Grover and Leskovec, 2016) to generate stock embeddings based on a knowledge graph and computed the similarity between stocks using these embeddings. They enhanced the original factors by using the factors from the top-K nearest neighbors for each stock. Finally, there are other studies (Ang and Lim, 2021; Xu et al., 2021) that also use knowledge graphs to generate better stock embeddings or to perform event-driven investment.

5 Building quant 4.0: engineering and architecture

Sections 2–4 introduced the three components of quant 4.0 from the algorithmic perspective. In this section, we review quant 4.0 from a system point of view and study how to put all these components together in one system. Fig. 9 illustrates the architecture of our proposed quant 4.0 system framework, including the offline system for quant research and the online system for quant trading.

5.1 System for offline research

The offline quant research system aims to improve the efficiency of quant research. It contains several layers (hardware layer, raw data layer, meta factor layer, factor layer, and model layer) and modules (high-performance computing clusters, data system, cache system, data pre-processing, automated factor mining, knowledge-based system, large-scale data analytics, AutoML, and risk simulation).

The underlying hardware platform for offline research is a high-performance computing cluster consisting of many computing nodes that are mounted with shared storage and interconnected in a high-bandwidth network. The combination of multiple nodes aggregates the computing power distributed in individual nodes to support large-scale quant computing tasks.

The data layer collects an extensive amount of financial data and provides data management and query service for upper-layer applications. Because financial data are heterogeneous and multimodal, the data layer incorporates different types of database systems such as SQL (Codd, 1970), time-series (Namiot, 2015), NoSQL (Gessert et al., 2017), and graph databases (Kaliyar, 2015). Additionally, to accelerate data access, a highly efficient distributed storage system is required for the large volume of financial data. For high-frequency tick data, in-memory databases (Tan et al., 2015) can be used as data caches to store frequently accessed data and reduce data transfer time between hard disks and memory.

The factor mining system consists of a meta factor layer, a factor layer, and a factor base that connects them. The meta factor layer pre-processes raw data with various modalities into meta factors with unified formats and appropriate values to meet the demand of factor mining for unified input formats. The factor layer builds an automated factor mining engine based on the automated factor mining pipeline. It is backed by some key techniques in distributed execution and computing acceleration. The factor base is an integrated platform for the storage, computation, dependency management, back test, tracking, and analysis of all factors. The generated data are committed to the factor base in various forms, including computed values, symbolic expressions, and model configurations for machine learning factors.

In parallel with the factor mining system, there is a knowledge-based system that leverages a distributed graph computing platform to provide knowledge-driven AI capabilities. The platform serves as a knowledge base to store a large financial behavior graph, which serves as the foundation for downstream financial knowledge reasoning and decision-making through an inference engine. Unlike traditional domain knowledge bases, the financial behavior knowledge graph in quant 4.0 has the

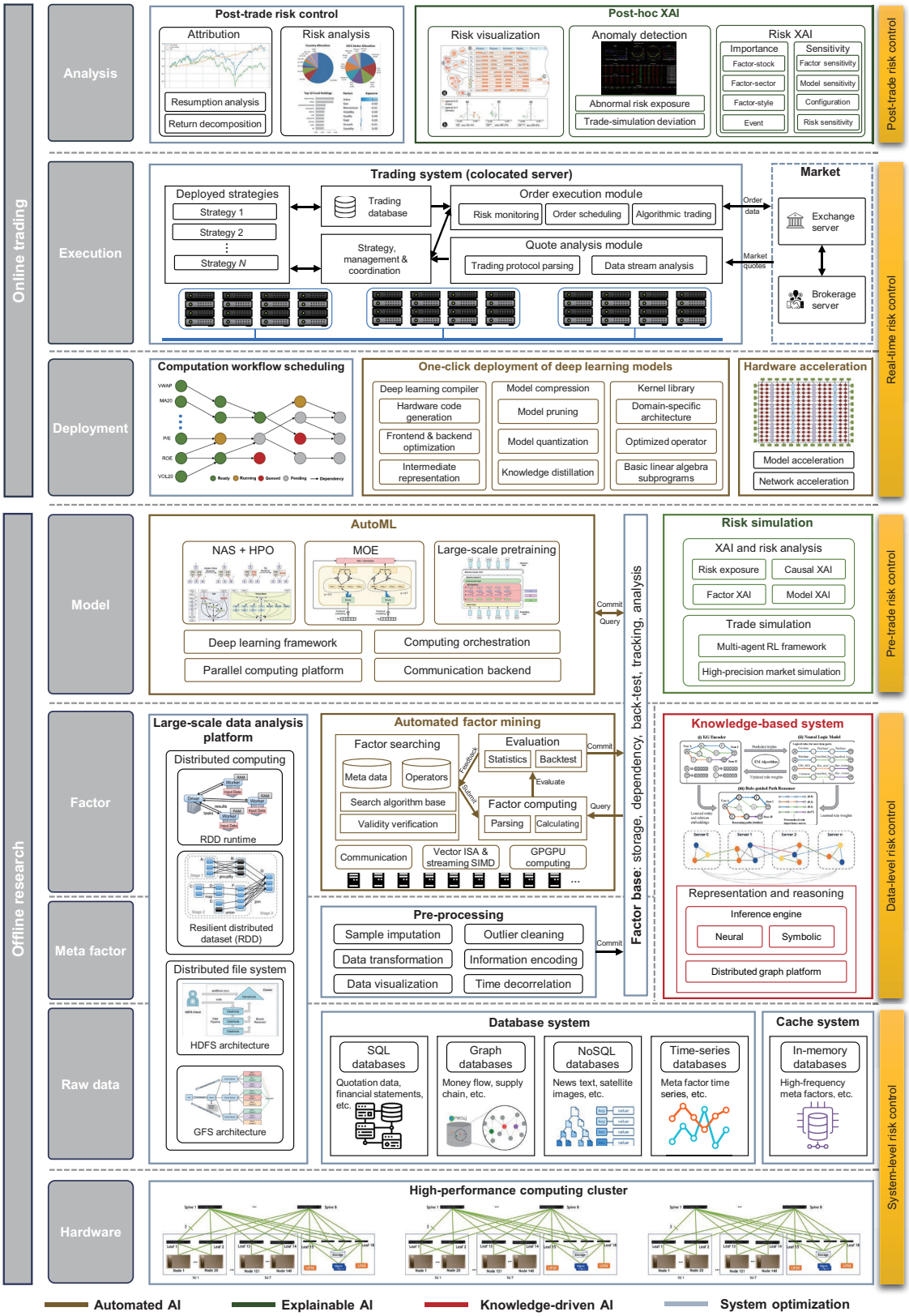


Fig. 9 Architecture of an example quant 4.0 engineering platform for investment research and trading. Parts of this figure are cited from Ghemawat et al. (2003), Shvachko et al. (2010), Zaharia et al. (2010, 2012), Bender et al. (2018), Real et al. (2019), Wang QW et al. (2021), and Fedus et al. (2022). References to color refer to the online version of this figure

potential to grow to a massive scale with billions of nodes and edges. Therefore, a flexible and scalable architecture (Lu et al., 2014) is necessary to efficiently store and manage large-scale dynamic graph data.

The modeling system is in charge of the automatic generation of machine learning models, and the corresponding risk evaluation and back-test simulation procedures before they are deployed in real-world environments. The AutoML module implements automated model generation algorithms on large-scale distributed deep learning systems. By using deep learning frameworks and computing orchestration tools on parallel computing platforms with communication backends, the AutoML module implements algorithms such as neural architecture search (NAS)/HPO (Jin HF et al., 2019), mixture-of-experts (Fedus et al., 2022), and large-scale pre-training (Shoeybi et al., 2020). The risk and simulation module involves XAI algorithms and market simulators to identify and analyze the potential risk exposures of the models before they are deployed to real-world trading environments.

5.2 System for online trading

The online trading system focuses on deploying investment strategies for real trading and executing post-trade analysis, and its major goal is to achieve low trading latency and high execution efficiency.

The deployment layer involves a computation scheduling module, an automated deployment module for deep learning models, and a hardware acceleration module. The computation scheduling module arranges a reasonable and efficient computation order for factors based on their intrinsic data dependencies, and forms a directed acyclic graph (DAG). The automated deployment module for deep learning models implements the one-click model deployment algorithms discussed in Section 2.4, as well as kernel libraries that provide implementations of common functions that are highly optimized based on hardware features. The hardware acceleration module aims to improve the computation efficiency of data processing and model inference using special hardware technology such as field-programmable gate array acceleration and high-level synthesis (Cong et al., 2022).

The execution layer converts trading decisions to actual orders that are executed in exchanges, and its goal is to reduce the trading latency as much as possible. The latency can be decomposed into two parts: transmission latency is the delay of signal communication between trading servers and market servers, and computation latency is the delay between received quotes and sent orders. To reduce transmission latency, the trading system is usually deployed on servers that are co-located with market servers. To reduce computation latency, a trading system must be optimized in a full-strategy pipeline from data collection to order execution through various software and hardware acceleration techniques.

The analysis layer monitors the execution of investment strategies and performs analysis for further adjustments. It involves a post-trade risk control module that is responsible for ordinary performance monitoring and a post-hoc XAI module for explaining strategy behavior from the perspective of AI. The post-trade risk control module performs return and risk attribution, with a goal of analyzing a strategy's performance and revealing the intrinsic risk structure hidden by machine learning black boxes. Furthermore, the post-hoc XAI module provides thorough risk analysis by analyzing all strategy components in terms of importance and sensitivity, together with appropriate visualizations.

6 LLMs for quantitative investment

LLMs such as GPT-4 (OpenAI, 2023) and LLaMA (Touvron et al., 2023) have shown remarkable intelligent capability in many areas, and we believe they have great potential in quantitative finance (Wang SZ et al., 2023; Zhang HH et al., 2023). In particular, LLM techniques such as supervised fine-tuning (SFT) (Wei et al., 2022), prompt engineering (Liu PF et al., 2023), and retrieval-augmented generation (RAG) (Karpukhin et al., 2020; Lewis et al., 2020) (Fig. 10a) can help incorporate finance domain knowledge and quant into LLM agents (Sumers et al., 2023; Weng, 2023) (Fig. 10b) to tackle complex problems.

LLM techniques are also widely applied in the financial domain (Liu XY et al., 2023; Wu SJ et al., 2023; Xie et al., 2023). They are usually trained on a financial corpus and evaluated on some general financial-related tasks such as question answering and analysis report writing, rather than quant-specific tasks such as alpha mining and predictive

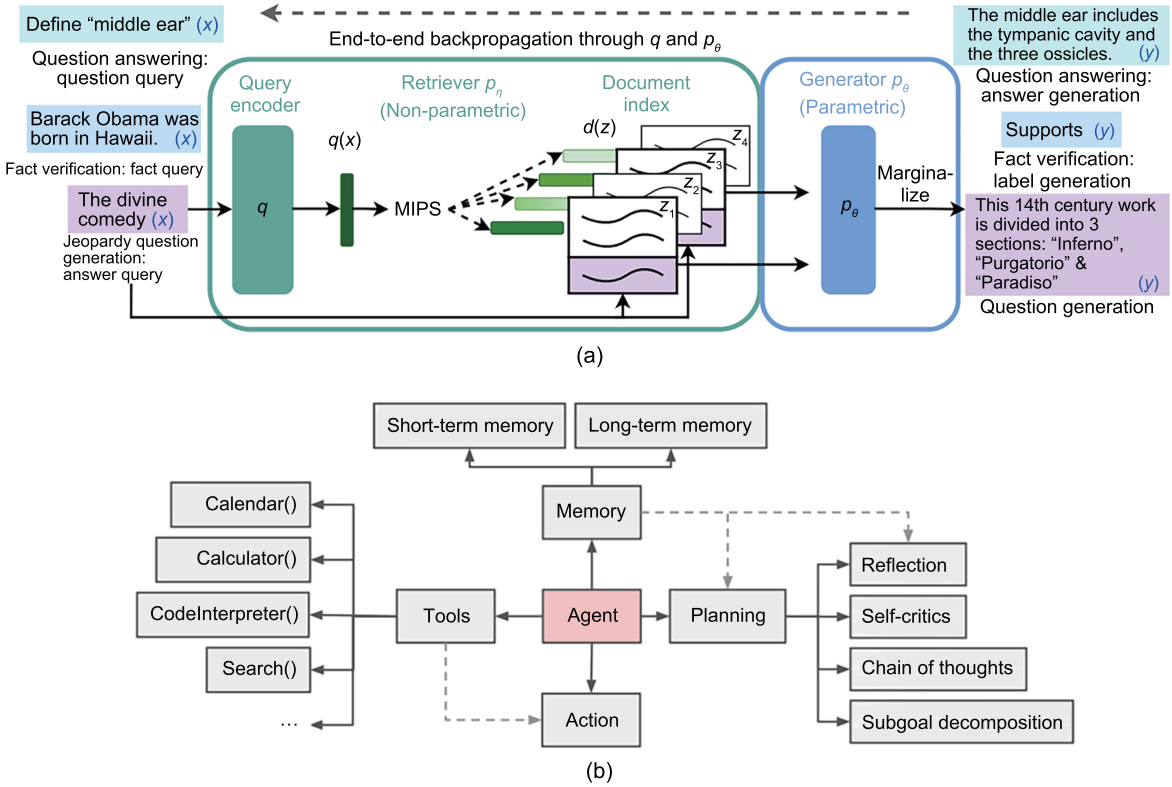


Fig. 10 LLM-relevant techniques: (a) RAG pipeline (Karpukhin et al., 2020); (b) architecture of an LLM-based autonomous agent (Weng, 2023) (LLM: large language model; RAG: retrieval-augmented generation)

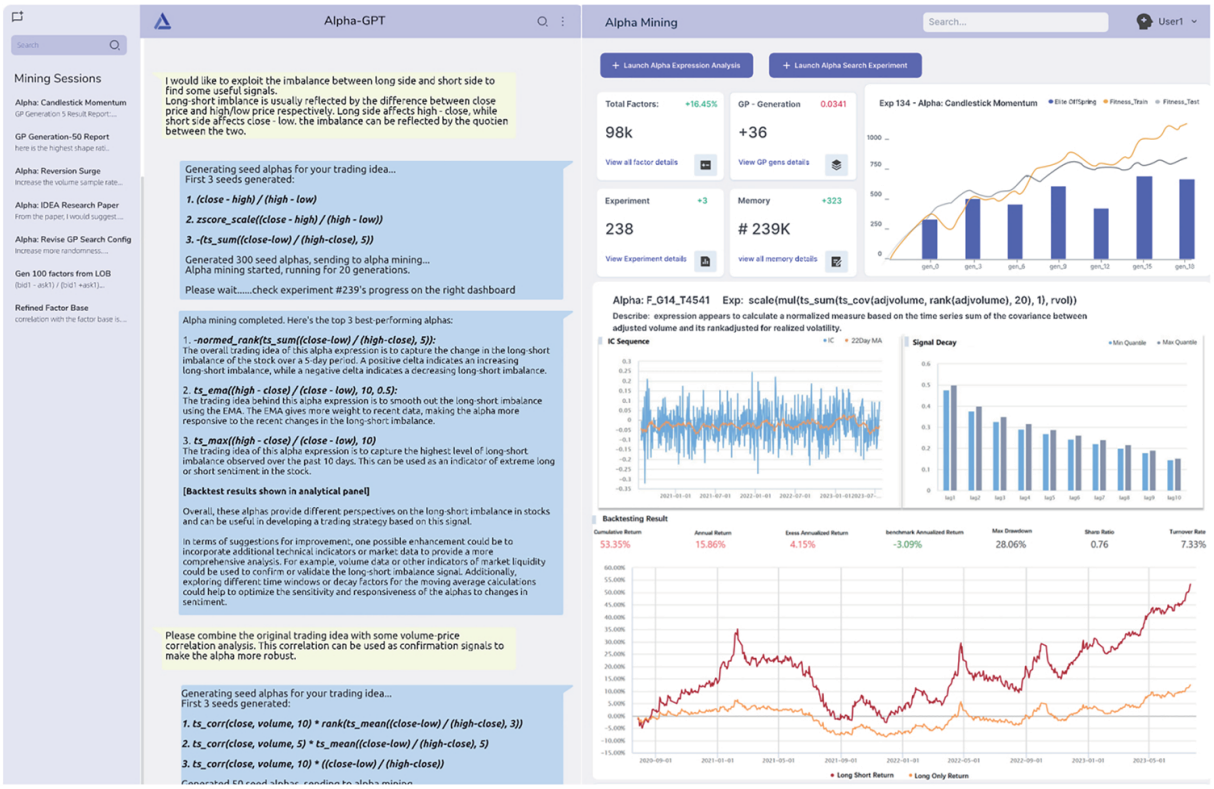
modeling. In this section we introduce different roles that LLMs can play in quant research, including as a quant copilot, financial analyst, and feature engineer.

6.1 Quant copilot

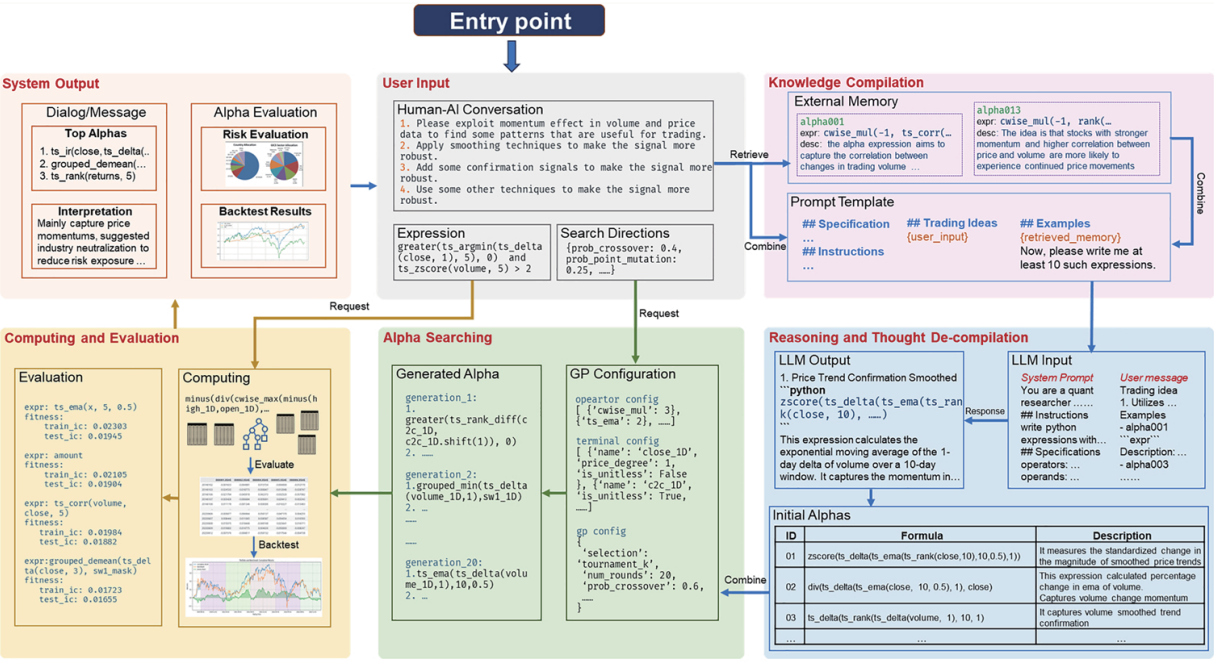
An LLM can act as a quant copilot to communicate with the quant research pipeline and human quants can improve their research efficiency using this agent. For example, Wang SZ et al. (2023) proposed Alpha-GPT (Fig. 11), an LLM-based agent that attempts to understand the ideas or intent of quants and generate creative, insightful, and effective alphas. Human quants can play with Alpha-GPT using natural language iteratively to discover “good” alphas. This agent system is implemented with a domain-specific prompt engineering framework that leverages the power of LLMs. Specifically, the agent turns the user input into a comprehensive prompt according to predefined templates, which include original input, user intent, relevant examples, and other specifications. Then the agent calls the underlying alpha mining system to discover the most valuable alphas according to the user intent. In this example, an LLM acts as a mediator that smooths the communication between the human and quant pipeline, liberating quants from the tedious work of translating their intuitions into computer-readable programs and configurations.

6.2 Financial analyst

An LLM can act as a financial analyst to generate judgments, ranks, or scores that can help build useful alphas. For example, Zhang HH et al. (2023) used an LLM (i.e., ChatGPT) to evaluate whether the sentiment of input news is positive or negative based on domain-optimized prompts (Fig. 12). If the sentiment is mostly positive, the related stocks are considered good to buy, and vice versa. In this way, the LLM behaves as a human financial analyst, providing insights that can then be used to inform trading decisions. Similarly, Lopez-Lira and Tang (2023) studied how ChatGPT could predict stock returns by prompting it to classify sentiments into



(a)



(b)

Fig. 11 The Web-based user interface of Alpha-GPT (a) and the internal workflow of Alpha-GPT (b) (Wang SZ et al., 2023)

Prompt (English): Forget all previous instructions. You are now a financial expert giving investment advice. I'll give you a news summary, and you need to answer whether this news is GOOD NEWS or BAD NEWS for the listed company. Please choose only one option from GOOD NEWS, BAD NEWS, NOT SURE, and do not provide any additional responses. The news content is: "The company continues to promote the upgrading of its traditional filter business product structure, demonstrating its resilience in a fiercely competitive market. Looking ahead, we are optimistic about the company's solid foundation in the mobile optics business and expect forward-looking layouts such as HUD and AR to open up a second growth curve." The listed company is Zhejiang Crystal Optoelectronics Technology Co., Ltd.

Response: GOOD NEWS

Elaboration (English): Based on the news, the company is continuously pushing for the upgrade of its traditional filter film business and demonstrating resilience in a highly competitive industry. Furthermore, looking ahead, the company is optimistic about solidifying its foundation in the mobile optical business and expects future growth through forward-looking initiatives such as HUD and AR. Based on this information, I believe this news is GOOD NEWS for the listed company.

Fig. 12 Prompts and LLM responses when using an LLM as an information analyst (Zhang HH et al., 2023)

three categories, with the results aiding in trading decisions. Steinert and Altmann (2023) also used LLMs for sentiment classification, but went a step further by having the LLM directly produce sentiment scores rather than choosing from predefined categories.

Different from the lightweight prompt engineering methods in Zhang HH et al. (2023), Lopez-Lira and Tang (2023), and Steinert and Altmann (2023), BloombergGPT (Wu SJ et al., 2023) implants financial domain knowledge through a much heavier pretraining process. Specifically, it trains a brand-new LLM on a large-scale financial corpus and evaluates it on sentiment analysis tasks that are carried out using prompts. The LLM demonstrates superior results over other general-purpose LLMs on these finance-related tasks.

6.3 Feature engineer

An LLM can also act as an "feature engineer" that compiles the input data into latent "features" (embeddings) for downstream quant prediction tasks. Actually, before the LLM era, the encoder/decoder architecture was popular in natural language modeling. For instance, the encoder-only Transformer language model BERT (Devlin et al., 2019) is widely used as an upstream feature engineer, and its outputs are usually used as feature inputs for many downstream NLP tasks, such as question answering (Rajpurkar et al., 2016) and sentiment classification (Socher et al., 2013). There are also a number of references discussing how to embed natural language features in financial investment scenarios (Ding X et al., 2016; Hu et al., 2018; Du and Tanaka-Ishii, 2020; Li W et al., 2020; Ying et al., 2020; Xu et al., 2021). These works aim to build effective quant prediction models mainly from financial text data such as news, reports, or user-generated content (UGC) from social media using various language models. The evolution of LLMs has further broadened this approach, enhancing their ability to produce more informative embeddings. With the rapid growth of model size, the reasoning power of LLMs has been improved dramatically. Ding YJ et al. (2023) (Fig. 13) used LLaMA-7B (Touvron et al., 2023) as the upstream feature engineer that embeds financial news and predicts stock returns in downstream tasks. Specifically, it generates daily global market embeddings from all news headlines. Then, this global embedding is aligned with each individual stock via an attention mechanism to generate stock embeddings. These global embeddings are then tailored to individual stocks through an attention mechanism, creating embeddings for each individual stock. These stock embeddings are then used by downstream models that predict future stock returns.

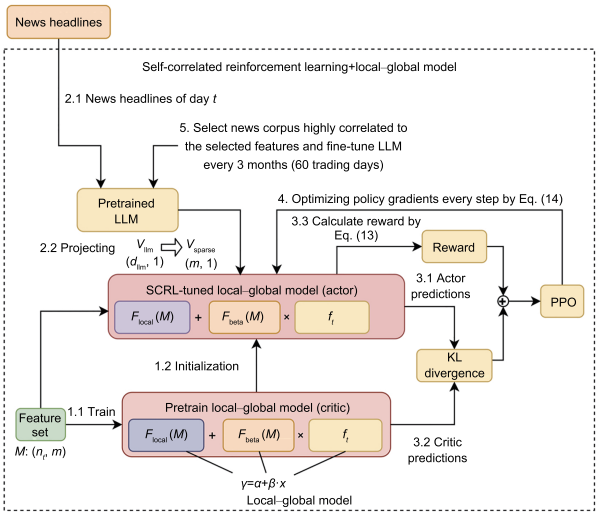


Fig. 13 LLM-generated embeddings of news headlines for stock prediction (Ding YJ et al., 2023)

7 Discussion of 10 challenges in quant technology

In this section, we summarize 10 challenges in the development of next-generation quant technology. As shown in Fig. 14, these challenges include computing and data infrastructure, investment modeling, risk modeling, market simulation, and cognitive AI technology, and provide new research directions for researchers interested in AI technology and quant. We believe that some problems might be solved in the next couple of years with the rapid development of AI technology, while others may remain challenging for a long time.

7.1 Exponentially growing demand for computing power

With the rapid development of GPU technology and parallel computing technology, AI models have been scaling up year by year. The rapid growth in model size not only improves model performance, but has also led to a paradigm shift that is deeply affecting the technical roadmap of AI research. Accordingly, as an important domain going all-in on AI, quant 4.0 relies heavily on ultra-large-scale computing power and related engineering technology. Quant 4.0 investment research requires super-computing infrastructure as a fundamental support for large-scale factor mining, large-scale modeling, and back-testing and evaluation. We note that quant 4.0 demands much more computing power than what we could ever imagine, owing to multiple demanding workloads including large-scale automated factor mining, large-scale deep learning, rolling training, model ensemble, feature selection, and NAS/HPO. Given the expensive and limited computing power, we attempt to provide a few suggestions on future directions, including research on faster algorithms, online learning, and large-scale pretraining.

7.2 Alternative data technology

Alternative data, a concept opposite to conventional financial data, provides a much broader space for quant research. Although the concept of alternative data has appeared for a decade and is getting more and more popular in quant industry, it is still in the early stage of industry application. On one hand, data acquisition is difficult owing to intellectual property and privacy protection issues. To this end, we suggest research on certificate techniques for data asset ownership, data encryption techniques, and federated learning techniques for the credit assignment, safe transfer, and utilization of alternative data from multiple parties. On the other hand, data aggregation is also difficult owing to the heterogeneity in data. Hence, it is important to develop better data mining techniques for tackling such heterogeneity and differentiating signals from noise.

7.3 Financial knowledge engineering

As we introduced in Section 4, knowledge-driven AI will play an important role in future quantitative finance. Researching new knowledge representation methods, building complete and reliable knowledge bases, and developing new knowledge reasoning and decision algorithms are crucial problems in knowledge engineering for financial investment. Given the limitations of popular knowledge techniques, it is necessary for researchers to explore more efficient and more sophisticated methods to better represent and leverage different types of knowledge. Before the

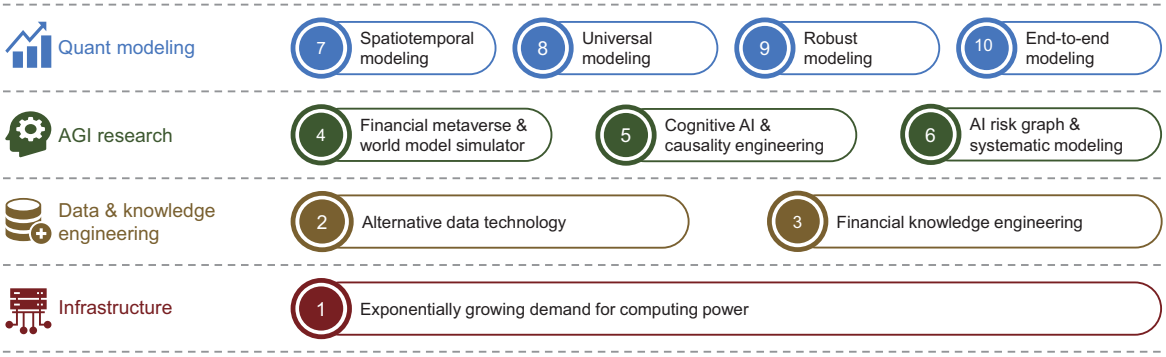


Fig. 14 The 10 challenges in quantitative investment technology

wide application of knowledge engineering in quant, a number of technical difficulties need to be solved in the future, including development of more sophisticated knowledge representation methods, a more advanced knowledge management system, and next-generation knowledge reasoning algorithms.

7.4 Financial metaverse and world model simulator

In quant research, it is very important to understand the underlying logic and micro-structure of financial markets. Unfortunately, empirical studies using historical data usually result in biased conclusions because they do not provide experimental access to all relevant information. The financial metaverse aims to build a simulated financial market parallel to real-world financial markets and use it as an experimental environment to simulate situations in real markets. Technical challenges in building such a financial metaverse include fine-grained data collection and demand for computing power for high-precision simulation.

7.5 Cognitive AI and causality engineering

Cognitive AI has been regarded as the future direction of artificial general intelligence (AGI) (Alattas et al., 2021) by many domain experts. Different from perceptive AI, which is applied mainly in high-frequency and high-breadth trading tasks, cognitive AI gives quant opportunities to touch those high-capacity but low-frequency investment strategies, including value investing, which holds a position over months or even over years. Meanwhile, causal inference (Yao et al., 2021) and causal machine learning (Pearl and Mackenzie, 2018; Schölkopf et al., 2021) have been regarded as one potential technical route toward AGI (Bengio, 2022). For investment tasks, understanding the true causal relationships among numerous factors is extremely challenging, due to the difficulty in enumerating all possible confounding variables (VanderWeele and Shpitser, 2013). Therefore, we propose a potential solution through causality engineering, which aims to build and maintain a large-scale causal diagram database that stores and manages all known/inferred causal relationships among variables.

7.6 AI risk graph and systematic modeling

The rapid growth of various types of financial big data creates opportunities to model and analyze financial risk systematically, ranging from macroeconomy to micro-market. We propose the concept of AI risk graph, a special financial knowledge graph for modeling financial risks at different hierarchical dimensions. An AI risk graph should be able to represent the causal dependency and risk transfer among various economic entities. Such a graph could be used to systematically model various types of risks at different levels by computing the conditional probability of risk for a specified object in certain conditions. In addition to current risk models such as BARRA (MSCI, 1996), complex risk measurement for investment is required to exploit nonlinear risks.

7.7 Spatiotemporal modeling

Traditionally, stock strategies are developed either along the time axis (called time-series modeling or temporal modeling) or along the stock axis (called cross-sectional modeling or spatial modeling). Specifically, cross-sectional modeling compares only the relative strengths of investment signals within the same cross-section at certain time points, and time-series modeling treats each stock individually. A technically difficult but practically feasible idea is to merge cross-sectional modeling and time-series modeling in a unified framework, to use the advantages of both sides. In this way, spatiotemporal data mining techniques (Wang SZ et al., 2022) can be applied to model financial data.

7.8 Universal modeling

Because data volume and computing power are growing rapidly, the large-scale pretrained model has become one of the mainstream AI paradigms in practice. Successful examples (Brown et al., 2020; Chen M et al., 2021; Radford et al., 2021; Ramesh et al., 2021) have demonstrated the effectiveness of large-scale pretrained models on a number of application domains. Many successful pretraining models have exhibited universal power to help improve many downstream tasks with different task training and data sources. We expect that this phenomenon could be reproduced in quant applications. The pretraining-fine-tuning paradigm in

AI has demonstrated its success in natural language processing (Brown et al., 2020) and computer vision (Dosovitskiy et al., 2021; Wang WH et al., 2022). By pretraining a large model on as much data in a self-supervised manner, one can obtain a model extracting the common information across various tasks and use it to achieve superior performances on a series of downstream tasks compared to task-specific training. We think that the pretraining-finetuning paradigm can be transferred to quant scenarios for a number of reasons, because many finance prediction tasks have sufficiently large volumes of data and may share common patterns. However, there are still several difficulties in applying it to quant, such as the low signal-to-noise ratio and potential leakage of future information.

7.9 Robust modeling

Robustness is the biggest issue in quant modeling. Typical challenges for obtaining robust machine learning models for quant data include the extremely low signal-to-noise ratio and the non-independent and identically distributed nature of quant data. To address these problems, we suggest that researchers consider the following research directions: (1) causal effect modeling (Guo RC et al., 2021) to explore the causal effect between factors and financial decisions; (2) continual learning techniques (de Lange et al., 2022) to improve out-of-distribution generalization (Liu JS et al., 2021) and thus to increase model robustness; (3) model ensemble techniques such as bootstrap aggregating (Breiman, 1996a), boosting (Schapire, 1990; Breiman, 1998), stacking (Wolpert, 1992; Breiman, 1996b), and Bayesian model averaging (Hoeting et al., 1999) to improve prediction stability by combining multiple single models; (4) ensemble model diversification such as model randomization to expand single model diversity and increase the robustness of the ensemble model; (5) multi-modal learning for incorporating diversified data from different sources and patterns to improve model robustness.

7.10 End-to-end modeling

As we introduced in Section 2.1, the traditional quant research pipeline consists of a number of steps, and each step has its own optimization direction. Hence, the optimization goals of these steps are somewhat different. For example, a "good" factor with high information coefficient does not necessarily contribute positively to the final model output in a complicated nonlinear relationship. Therefore, it is natural to consider if there exists an end-to-end model with comparable performance. However, it is difficult to build a consistently optimal model due to the lack of clear supervision signal, difference in frequency between different steps (seconds for trading decisions and minutes to days for meta factors), low signal-to-noise ratio, and high demands for computing power. We recommend that researchers who are interested in the problem begin with an existing baseline model in the reinforcement learning framework (Wang JY et al., 2019; Wang ZC et al., 2021) and pay more attention to fusing data, factors, decisions, and executions in multiple time granularities.

Contributors

Jian GUO designed the research. Jian GUO and Saizhuo WANG conducted the main study and drafted the paper. Lionel M. NI and Heung-Yeung SHUM provided advisory support throughout the research process and helped in refining and elevating the conceptual framework of the study. All the authors revised and finalized the paper.

Conflict of interest

Heung-Yeung SHUM is an editorial board member of *Frontiers of Information Technology & Electronic Engineering*, and he was not involved in the peer review process of this paper. All the authors declare that they have no conflict of interest.

References

Abdul Karim Z, Muhamad Fahmi FSR, Abdul Karim B, et al., 2022. Market sentiments and firm-level equity returns: panel evidence of Malaysia. *Econ Res-Ekon Istraž*, 35(1):5253-5272. https://doi.org/10.1080/1331677X.2021.2025126

Alattas K, Alkaabi A, Alsaud AB, 2021. An overview of artificial general intelligence: recent developments and future challenges. *J Comput Sci*, 17(4):364-370. https://doi.org/10.3844/jcssp.2021.364.370

Ang G, Lim EP, 2021. Learning knowledge-enriched company embeddings for investment management. Proc 2nd ACM Int Conf on AI in Finance, Article 25. https://doi.org/10.1145/3490354.3494390

Bachelier L, 1900. Théorie de la spéculation. *Ann Sci L'cole Norm Supér*, 17:21-86 (in French). https://doi.org/10.24033/asens.476

Bai SJ, Kolter JZ, Koltun V, 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. https://arxiv.org/abs/1803.01271

Bender G, Kindermans PJ, Zoph B, et al., 2018. Understanding and simplifying one-shot architecture search. Proc 35th Int Conf on Machine Learning, p.549-558.

Bengio Y, 2022. GFlowNets and System 2 Deep Learning. https://www.microsoft.com/en-us/research/video/gflownets-and-system-2-deep-learning/ [Accessed on Nov. 10, 2022].

Bergstra J, Bengio Y, 2012. Random search for hyper-parameter optimization. *J Mach Learn Res*, 13(10):281-305.

Biggio L, Bendinelli T, Neitz A, et al., 2021. Neural symbolic regression that scales. Proc 38th Int Conf on Machine Learning, p.936-945.

Black F, Scholes M, 1973. The pricing of options and corporate liabilities. *J Polit Econ*, 81(3):637-654. https://doi.org/10.1086/260062

Bordes A, Usunier N, Garcia-Durán A, et al., 2013. Translating embeddings for modeling multi-relational data. Proc 26th Int Conf on Neural Information Processing Systems, p.2787-2795.

Bottou L, Peters J, Quiñonero-Candela J, et al., 2013. Counterfactual reasoning and learning systems: the example of computational advertising. *J Mach Learn Res*, 14(1):3207-3260.

Breiman L, 1996a. Stacked regressions. *Mach Learn*, 24(1):49-64. https://doi.org/10.1007/BF00117832

Breiman L, 1996b. Bagging predictors. *Mach Learn*, 24(2):123-140. https://doi.org/10.1007/BF00058655

Breiman L, 1998. Arcing classifier (with discussion and a rejoinder by the author). *Ann Statist*, 26(3):801-849. https://doi.org/10.1214/aos/1024691079

Brown TB, Mann B, Ryder N, et al., 2020. Language models are few-shot learners. Proc 34th Int Conf on Neural Information Processing Systems.

Chen M, Tworek J, Jun H, et al., 2021. Evaluating large language models trained on code. https://arxiv.org/abs/2107.03374

Chen TX, Chen W, Du LY, 2021. An empirical study of financial factor mining based on gene expression programming. Proc 4th Int Conf on Advanced Electronic Materials, Computers and Software Engineering, p.1113-1117. https://doi.org/10.1109/AEMCSE51986.2021.00228

Cheng DW, Yang FZ, Wang XY, et al., 2020. Knowledge graph-based event embedding framework for financial quantitative investments. Proc 43rd Int ACM SIGIR Conf on Research and Development in Information Retrieval, p.2221-2230. https://doi.org/10.1145/3397271.3401427

Cheng Y, Wang D, Zhou P, et al., 2018. Model compression and acceleration for deep neural networks: the principles, progress, and challenges. *IEEE Signal Process Mag*, 35(1):126-136. https://doi.org/10.1109/MSP.2017.2765695

Codd EF, 1970. A relational model of data for large shared data banks. *Commun ACM*, 13(6):377-387. https://doi.org/10.1145/362384.362685

Coleman T, 2011. A Practical Guide to Risk Management. https://papers.ssrn.com/abstract=2586032 [Accessed on Nov. 10, 2022].

Cong J, Lau J, Liu G, et al., 2022. FPGA HLS today: successes, challenges, and opportunities. *ACM Trans Reconfig Technol Syst*, 15(4):51. https://doi.org/10.1145/3530775

de Lange M, Aljundi R, Masana M, et al., 2022. A continual learning survey: defying forgetting in classification tasks. *IEEE Trans Patt Anal Mach Intell*, 44(7):3366-3385. https://doi.org/10.1109/TPAMI.2021.3057446

Deng SM, Zhang NY, Zhang W, et al., 2019. Knowledge-driven stock trend prediction and explanation via temporal convolutional network. Proc Companion World Wide Web Conf, p.678-685. https://doi.org/10.1145/3308560.3317701

Devlin J, Chang MW, Lee K, et al., 2019. BERT: pre-training of deep bidirectional transformers for language understanding. Proc Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, p.4171-4186. https://doi.org/10.18653/v1/N19-1423

Ding X, Zhang Y, Liu T, et al., 2016. Knowledge-driven event embedding for stock prediction. Proc COLING, the 26th Int Conf on Computational Linguistics: Technical Papers, p.2133-2142.

Ding YJ, Jia S, Ma TY, et al., 2023. Integrating stock features and global information via large language models for enhanced stock return prediction. https://arxiv.org/abs/2310.05627

Dosovitskiy A, Beyer L, Kolesnikov A, et al., 2021. An image is worth 16×16 words: Transformers for image recognition at scale. Proc 9th Int Conf on Learning Representations.

Du X, Tanaka-Ishii K, 2020. Stock embeddings acquired from news articles and price history, and an application to portfolio optimization. Proc 58th Annual Meeting of the Association for Computational Linguistics, p.3353-3363. https://doi.org/10.18653/v1/2020.acl-main.307

Engle RF, 1982. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, 50(4):987-1007. https://doi.org/10.2307/1912773

Engle RF, Granger CWJ, 1987. Co-integration and error correction: representation, estimation, and testing. *Econometrica*, 55(2):251-276. https://doi.org/10.2307/1913236

Falkner S, Klein A, Hutter F, 2018. BOHB: robust and efficient hyperparameter optimization at scale. Proc 35th Int Conf on Machine Learning, p.1436-1445.

Fama EF, French KR, 1992. The cross-section of expected stock returns. *J Finance*, 47(2):427-465. https://doi.org/10.1111/j.1540-6261.1992.tb04398.x

Fedus W, Zoph B, Shazeer N, 2022. Switch Transformers: scaling to trillion parameter models with simple and efficient sparsity. *J Mach Learn Res*, 23(1):120.

Feng FL, He XN, Wang X, et al., 2019. Temporal relational ranking for stock prediction. *ACM Trans Inform Syst*, 37(2):27. https://doi.org/10.1145/3309547

Gessert F, Wingerath W, Friedrich S, et al., 2017. NoSQL database systems: a survey and decision guidance. *Comput Sci Res Dev*, 32(3-4):353-365. https://doi.org/10.1007/s00450-016-0334-3

Ghemawat S, Gobioff H, Leung ST, 2003. The Google File System. Proc 19th ACM Symp on Operating Systems Principles, p.29-43. https://doi.org/10.1145/945445.945450

Grover A, Leskovec J, 2016. node2vec: scalable feature learning for networks. Proc 22nd ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, p.855-864. https://doi.org/10.1145/2939672.2939754

Guo K, Sun Y, Qian X, 2017. Can investor sentiment be used to predict the stock price? Dynamic analysis based on China stock market. *Phys A Statist Mech Appl*, 469:390-396. https://doi.org/10.1016/j.physa.2016.11.114

Guo LB, Sun ZQ, Hu W, 2019. Learning to exploit long-term relational dependencies in knowledge graphs. Proc 36th Int Conf on Machine Learning, p.2505-2514.

Guo RC, Cheng L, Li JD, et al., 2021. A survey of learning causality with data: problems and methods. *ACM Comput Surv*, 53(4):75. https://doi.org/10.1145/3397269

Han S, Pool J, Tran J, et al., 2015. Learning both weights and connections for efficient neural network. Proc 28th Int Conf on Neural Information Processing Systems, p.1135-1143.

Han S, Mao HZ, Dally WJ, 2016. Deep compression: compressing deep neural network with pruning, trained quantization and Huffman coding. Proc 4th Int Conf on Learning Representations.

Hayes-Roth F, Waterman DA, Lenat DB, 1983. Building Expert Systems. Addison-Wesley Longman Publishing Co., Boston, USA.

He X, Zhao KY, Chu XW, 2021. AutoML: a survey of the state-of-the-art. *Knowl-Based Syst*, 212:106622. https://doi.org/10.1016/j.knosys.2020.106622

Hinton G, Vinyals O, Dean J, 2015. Distilling the knowledge in a neural network. https://arxiv.org/abs/1503.02531

Hoeting JA, Madigan D, Raftery AE, et al., 1999. Bayesian model averaging: a tutorial. *Statist Sci*, 14(4):382-401.

Hornik K, Stinchcombe M, White H, 1989. Multilayer feedforward networks are universal approximators. *Neur Netw*, 2(5):359-366. https://doi.org/10.1016/0893-6080(89)90020-8

Hou K, 2007. Industry information diffusion and the lead-lag effect in stock returns. *Rev Financ Stud*, 20(4):1113-1138. https://doi.org/10.1093/revfin/hhm003

Hu ZN, Liu WQ, Bian J, et al., 2018. Listening to chaotic whispers: a deep learning framework for news-oriented stock trend prediction. Proc 11th ACM Int Conf on Web Search and Data Mining, p.261-269. https://doi.org/10.1145/3159652.3159690

Imbens GW, Angrist JD, 1994. Identification and estimation of local average treatment effects. *Econometrica*, 62(2):467-475. https://doi.org/10.2307/2951620

Jackson P, 1998. Introduction to Expert Systems. Addison-Wesley, Boston, USA.

Ji SX, Pan SR, Cambria E, et al., 2022. A survey on knowledge graphs: representation, acquisition, and applications. *IEEE Trans Neur Netw Learn Syst*, 33(2):494-514. https://doi.org/10.1109/TNNLS.2021.3070843

Jin HF, Song QQ, Hu X, 2019. Auto-Keras: an efficient neural architecture search system. Proc 25th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining, p.1946-1956.

Jin Y, Fu WL, Kang J, et al., 2020. Bayesian symbolic regression. https://arxiv.org/abs/1910.08892

Kakushadze Z, 2016. 101 formulaic alphas. https://arxiv.org/abs/1601.00991

Kaliyar RK, 2015. Graph databases: a survey. Proc Int Conf on Computing, Communication & Automation, p.785-790. https://doi.org/10.1109/CCAA.2015.7148480

Karpukhin V, Oguz B, Min S, et al., 2020. Dense passage retrieval for open-domain question answering. Proc Conf on Empirical Methods in Natural Language Processing, p.6769-6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

Kaya M, Bilge HS, 2019. Deep metric learning: a survey. *Symmetry*, 11(9):1066. https://doi.org/10.3390/sym11091066

Kendall EF, McGuinness DL, Ding Y, 2019. Ontology Engineering. Morgan & Claypool Publishers, San Rafael, California, USA.

Klein A, Falkner S, Bartels S, et al., 2017. Fast Bayesian optimization of machine learning hyperparameters on large datasets. Proc 20th Int Conf on Artificial Intelligence and Statistics, p.528-536.

Kulis B, 2013. Metric learning: a survey. *Found Trends Mach Learn*, 5(4):287-364. https://doi.org/10.1561/2200000019

La Cava WG, Orzechowski P, Burlacu B, et al., 2021. Contemporary symbolic regression methods and their relative performance. Proc 1st Neural Information Processing Systems Track on Datasets and Benchmarks.

LeCun Y, Bengio Y, Hinton G, 2015. Deep learning. *Nature*, 521(7553):436-444. https://doi.org/10.1038/nature14539

Lehmann F, 1992. Semantic networks. *Comput Math Appl*, 23(2-5):1-50. https://doi.org/10.1016/0898-1221(92)90135-5

Lewis P, Perez E, Piktus A, et al., 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. Proc 34th Int Conf on Neural Information Processing Systems, Article 793.

Li MZ, Liu Y, Liu XY, et al., 2021. The deep learning compiler: a comprehensive survey. *IEEE Trans Parall Distrib Syst*, 32(3):708-727. https://doi.org/10.1109/TPDS.2020.3030548

Li W, Bao RH, Harimoto K, et al., 2020. Modeling the stock relation with graph network for overnight stock movement prediction. Proc 29th Int Joint Conf on Artificial Intelligence, p.4541-4547. https://doi.org/10.24963/ijcai.2020/626

Li YL, Wang TC, Sun BQ, et al., 2022. Detecting the lead–lag effect in stock markets: definition, patterns, and investment strategies. *Finance Innov*, 8(1):51. https://doi.org/10.1186/s40854-022-00356-3

Liu CX, Zoph B, Neumann M, et al., 2018. Progressive neural architecture search. Proc 15th European Conf on Computer Vision, p.19-35. https://doi.org/10.1007/978-3-030-01246-5_2

Liu HX, Simonyan K, Yang YM, 2019. DARTS: differentiable architecture search. Proc 7th Int Conf on Learning Representations.

Liu JS, Shen ZY, He Y, et al., 2021. Towards out-of-distribution generalization: a survey. https://arxiv.org/abs/2108.13624

Liu PF, Yuan WZ, Fu JL, et al., 2023. Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. *ACM Comput Surv*, 55(9):195. https://doi.org/10.1145/3560815

Liu XY, Wang GX, Yang HY, et al., 2023. FinGPT: democratizing Internet-scale data for financial large language models. https://arxiv.org/abs/2307.10485

Long JW, Chen ZP, He WB, et al., 2020. An integrated framework of deep learning and knowledge graph for prediction of stock price trend: an application in Chinese stock exchange market. *Appl Soft Comput*, 91:106205. https://doi.org/10.1016/j.asoc.2020.106205

Lopez-Lira A, Tang YH, 2023. Can ChatGPT forecast stock price movements? Return predictability and large language models. https://arxiv.org/abs/2304.07619

Lu Y, Cheng J, Yan D, et al., 2014. Large-scale distributed graph computing systems: an experimental evaluation. *Proc VLDB Endow*, 8(3):281-292. https://doi.org/10.14778/2735508.2735517

Luo YF, Wang MS, Zhou H, et al., 2019. AutoCross: automatic feature crossing for tabular data in real-world applications. Proc 25th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining, p.1936-1945. https://doi.org/10.1145/3292500.3330679

Markowitz H, 1952. Portfolio selection. *J Finance*, 7(1):77-91. https://doi.org/10.2307/2975974

McNeill D, 1993. Fuzzy Logic. Simon & Schuster, New York, USA.

Minsky M, 1974. A Framework for Representing Knowledge. MIT-AI Laboratory Memo 306.

MSCI, 1996. Barra's Risk Models. https://www.msci.com/research-paper/barra-s-risk-models/014972229 [Accessed on Nov. 10, 2022].

Müllner D, 2011. Modern hierarchical, agglomerative clustering algorithms. https://arxiv.org/abs/1109.2378

Murdoch WJ, Singh C, Kumbier K, et al., 2019. Definitions, methods, and applications in interpretable machine learning. *Proc Natl Acad Sci USA*, 116(44):22071-22080. https://doi.org/10.1073/pnas.1900654116

Namiot D, 2015. Time series databases. Proc XVII Int Conf on Data Analytics and Management in Data Intensive Domains, p.132-137.

Nevmyvaka Y, Feng Y, Kearns M, 2006. Reinforcement learning for optimized trade execution. Proc 23rd Int Conf on Machine Learning, p.673-680. https://doi.org/10.1145/1143844.1143929

Ng R, Subrahmanian VS, 1992. Probabilistic logic programming. *Inform Comput*, 101(2):150-201. https://doi.org/10.1016/0890-5401(92)90061-J

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

Pearl J, Mackenzie D, 2018. The Book of Why: the New Science of Cause and Effect. Basic Books, Inc., New York, USA.

Radford A, Kim JW, Hallacy C, et al., 2021. Learning transferable visual models from natural language supervision. Proc 38th Int Conf on Machine Learning, p.8748-8763.

Rajpurkar P, Zhang J, Lopyrev K, et al., 2016. SQuAD: 100,000+ questions for machine comprehension of text. Proc Conf on Empirical Methods in Natural Language Processing, p.2383-2392. https://doi.org/10.18653/v1/D16-1264

Ramesh A, Pavlov M, Goh G, et al., 2021. Zero-shot text-to-image generation. Proc 38th Int Conf on Machine Learning, p.8821-8831.

Rashid A, Fayyaz M, Karim M, 2019. Investor sentiment, momentum, and stock returns: an examination for direct and indirect effects. *Econ Res-Ekon Istraž*, 32(1):2638-2656. https://doi.org/10.1080/1331677X.2019.1650652

Real E, Moore S, Selle A, et al., 2017. Large-scale evolution of image classifiers. Proc 34th Int Conf on Machine Learning, p.2902-2911.

Real E, Aggarwal A, Huang YP, et al., 2019. Regularized evolution for image classifier architecture search. Proc 33rd AAAI Conf on Artificial Intelligence, p.4780-4789. https://doi.org/10.1609/aaai.v33i01.33014780

Richardson M, Domingos P, 2006. Markov logic networks. *Mach Learn*, 62(1-2):107-136. https://doi.org/10.1007/s10994-006-5833-1

Rocktäschel T, Riedel S, 2017. End-to-end differentiable proving. Proc 31st Int Conf on Neural Information Processing Systems, p.3791-3803.

Sakalauskas V, Kriksciuniene D, 2009. Research of the calendar effects in stock returns. Proc Int Conf on Business Information Systems, p.69-78. https://doi.org/10.1007/978-3-642-03424-4_9

Samuelson PA, 1965. Proof that properly anticipated prices fluctuate randomly. *IMR*, 6(2):41.

Sawhney R, Agarwal S, Wadhwa A, et al., 2020. Spatiotemporal hypergraph convolution network for stock movement forecasting. Proc IEEE Int Conf on Data Mining, p.482-491. https://doi.org/10.1109/ICDM50108.2020.00057

Schapire RE, 1990. The strength of weak learnability. *Mach Learn*, 5(2):197-227. https://doi.org/10.1007/BF00116037

Schölkopf B, Locatello F, Bauer S, et al., 2021. Toward causal representation learning. *Proc IEEE*, 109(5):612-634. https://doi.org/10.1109/JPROC.2021.3058954

Shapley LS, 1953. A value for n-person games. In: Kuhn HW, Tucker AW (Eds.), Contributions to the Theory of Games (AM-28), Volume II. Princeton University Press, Princeton, USA. https://doi.org/doi:10.1515/9781400881970-018

Shoeybi M, Patwary M, Puri R, et al., 2020. Megatron-LM: training multi-billion parameter language models using model parallelism. https://arxiv.org/abs/1909.08053

Shvachko K, Kuang HR, Radia S, et al., 2010. The Hadoop Distributed File System. Proc IEEE 26th Symp on Mass Storage Systems and Technologies, p.1-10. https://doi.org/10.1109/MSST.2010.5496972

Sil A, Yates A, 2013. Re-ranking for joint named-entity recognition and linking. Proc 22nd ACM Int Conf on Information & Knowledge Management, p.2369-2374. https://doi.org/10.1145/2505515.2505601

Sims CA, 1980. Macroeconomics and reality. *Econometrica*, 48(1):1-48. https://doi.org/10.2307/1912017

Socher R, Chen DQ, Manning CD, et al., 2013. Reasoning with neural tensor networks for knowledge base completion. Proc 26th Int Conf on Neural Information Processing Systems, p.926-934.

Sowa JF, 1992. Semantic Networks. http://www.jfsowa.com/pubs/semnet.htm [Accessed on Nov. 16, 2022].

Steinert R, Altmann S, 2023. Linking microblogging sentiments to stock price movement: an application of GPT-4. https://arxiv.org/abs/2308.16771

Sumers TR, Yao SY, Narasimhan K, et al., 2023. Cognitive architectures for language agents. https://arxiv.org/abs/2309.02427

Sutskever I, Vinyals O, Le QV, 2014. Sequence to sequence learning with neural networks. Proc 27th Int Conf on Neural Information Processing Systems, p.3104-3112.

Tan KL, Cai QC, Ooi BC, et al., 2015. In-memory databases: challenges and opportunities from software and hardware perspectives. *ACM SIGMOD Rec*, 44(2):35-40. https://doi.org/10.1145/2814710.2814717

Thakkar A, Chaudhari K, 2021. A comprehensive survey on deep neural networks for stock market: the need, challenges, and future directions. *Expert Syst Appl*, 177:114800. https://doi.org/10.1016/j.eswa.2021.114800

Touvron H, Martin L, Stone K, et al., 2023. Llama 2: open foundation and fine-tuned chat models. https://arxiv.org/abs/2307.09288

Trouillon T, Welbl J, Riedel S, et al., 2016. Complex embeddings for simple link prediction. Proc 33rd Int Conf on Machine Learning, p.2071-2080.

Tsang M, Cheng DH, Liu HP, et al., 2020. Feature interaction interpretability: a case for explaining ad-recommendation systems via neural interaction detection. Proc 8th Int Conf on Learning Representations.

Tulchinsky I, 2019. Introduction to alpha design. In: Tulchinsky I (Ed.), Finding Alphas: a Quantitative Approach to Building Trading Strategies. Wiley, Chichester, UK. https://doi.org/10.1002/9781119571278.ch1

VanderWeele TJ, Shpitser I, 2013. On the definition of a confounder. *Ann Statist*, 41(1):196-220. https://doi.org/10.1214/12-AOS1058

Wang J, Zhang H, Bonne G, 2021. Machine Learning Factors: Capturing Non Linearities in Linear Factor Models. https://www.msci.com/www/research-report/machine-learning-factors/02410413451 [Accessed on Nov. 16, 2022].

Wang JY, Zhang Y, Tang K, et al., 2019. AlphaStock: a buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks. Proc 25th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining, p.1900-1908. https://doi.org/10.1145/3292500.3330647

Wang QW, Xu ZH, Chen ZT, et al., 2021. Visual analysis of discrimination in machine learning. *IEEE Trans Vis Comput Graph*, 27(2):1470-1480. https://doi.org/10.1109/TVCG.2020.3030471

Wang SZ, Cao JN, Yu PS, 2022. Deep learning for spatio-temporal data mining: a survey. *IEEE Trans Knowl Data Eng*, 34(8):3681-3700. https://doi.org/10.1109/TKDE.2020.3025580

Wang SZ, Yuan H, Zhou L, et al., 2023. Alpha-GPT: human-AI interactive alpha mining for quantitative investment. https://arxiv.org/abs/2308.00016

Wang WH, Bao HB, Dong L, et al., 2022. Image as a foreign language: BEiT pretraining for all vision and vision-language tasks. https://arxiv.org/abs/2208.10442

Wang WY, Cohen WW, 2016. Learning first-order logic embeddings via matrix factorization. Proc 25th Int Joint Conf on Artificial Intelligence, p.2132-2138.

Wang YZ, Wang HZ, He JW, et al., 2021. TAGAT: Type-Aware Graph Attention neTworks for reasoning over knowledge graphs. *Knowl-Based Syst*, 233:107500. https://doi.org/10.1016/j.knosys.2021.107500

Wang ZC, Huang BW, Tu SK, et al., 2021. DeepTrader: a deep reinforcement learning approach for risk-return balanced portfolio management with market conditions embedding. Proc 35th AAAI Conf on Artificial Intelligence, p.643-650. https://doi.org/10.1609/aaai.v35i1.16144

Wei J, Bosma M, Zhao V, et al., 2022. Finetuned language models are zero-shot learners. Proc 10th Int Conf on Learning Representations.

Weng LL, 2023. LLM Powered Autonomous Agents. https://lilianweng.github.io/posts/2023-06-23-agent/ [Accessed on July 29, 2023].

Wolpert DH, 1992. Stacked generalization. *Neur Netw*, 5(2):241-259. https://doi.org/10.1016/S0893-6080(05)80023-1

Wu SJ, Irsoy O, Lu S, et al., 2023. BloombergGPT: a large language model for finance. https://arxiv.org/abs/2303.17564

Wu YF, Mahfouz M, Magazzeni D, et al., 2021. How robust are limit order book representations under data perturbation? https://arxiv.org/abs/2110.04752

Xiao H, Huang ML, Zhu XY, 2016. From one point to a manifold: knowledge graph embedding for precise link prediction. Proc 25th Int Joint Conf on Artificial Intelligence, p.1315-1321.

Xie QQ, Han WG, Zhang X, et al., 2023. PIXIU: a comprehensive benchmark, instruction dataset and large language model for finance. Proc 37th Conf on Neural Information Processing Systems.

Xu WT, Liu WQ, Xu C, et al., 2021. REST: relational event-driven stock trend forecasting. Proc Web Conf, p.1-10. https://doi.org/10.1145/3442381.3450032

Yao LY, Chu ZX, Li S, et al., 2021. A survey on causal inference. *ACM Trans Knowl Disc Data*, 15(5):74. https://doi.org/10.1145/3444944

Ying XT, Xu C, Gao JL, et al., 2020. Time-aware graph relational attention network for stock recommendation. Proc 29th ACM Int Conf on Information & Knowledge Management, p.2281-2284. https://doi.org/10.1145/3340531.3412160

Yu XY, Liu TL, Wang XC, et al., 2017. On compressing deep models by low rank and sparse decomposition. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.67-76. https://doi.org/10.1109/CVPR.2017.15

Zaharia M, Chowdhury M, Franklin MJ, et al., 2010. Spark: Cluster Computing with Working Sets. https://www.usenix.org/conference/hotcloud-10/spark-cluster-computing-working-sets [Accessed on Nov. 11, 2022].

Zaharia M, Chowdhury M, Das T, et al., 2012. Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. Proc 9th USENIX Symp on Networked Systems Design and Implementation.

Zhang HH, Hua FR, Xu CJ, et al., 2023. Unveiling the potential of sentiment: can large language models predict Chinese stock price movements? https://arxiv.org/abs/2306.14222

Zhang XY, Zou JH, Ming X, et al., 2015. Efficient and accurate approximations of nonlinear convolutional networks. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.1984-1992. https://doi.org/10.1109/CVPR.2015.7298809

Zheng A, Casari A, 2018. Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists. O'Reilly, Boston, USA.

Zhu YQ, Xu WZ, Zhang JH, et al., 2021. A survey on graph structure learning: progress and opportunities. https://arxiv.org/abs/2103.03036

Zoph B, Le QV, 2017. Neural architecture search with reinforcement learning. Proc 5th Int Conf on Learning Representations.

Zoph B, Vasudevan V, Shlens J, et al., 2018. Learning transferable architectures for scalable image recognition. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.8697-8710. https://doi.org/10.1109/CVPR.2018.00907