AlphaPortfolio: Direct Construction Through Deep Reinforcement Learning and Interpretable AI*

Lin William Cong Ke Tang Jingyuan Wang Yang Zhang

First draft: December 2019; current draft: August 2021

Abstract

We directly optimize the objectives of portfolio management via deep reinforcement learning—an alternative to conventional supervised-learning paradigms that routinely entail first-step estimations of return distributions or risk premia. Building upon recent AI breakthroughs, we develop multi-sequence neural-network models tailored to the distinguishing features of financial data such as non-linearity and high dimensionality, while allowing training without labels and interactions with the market environment and state variables. Our AlphaPortfolio yields stellar out-of-sample performances (e.g., Sharpe ratio above two and over 13% risk-adjusted alpha with monthly re-balancing) that are robust under various market conditions economic restrictions (e.g., exclusion of small stocks and short-selling). Moreover, we project AlphaPortfolio onto simpler modeling spaces (e.g., using polynomial-feature-sensitivity) to uncover key drivers of investment performance, including their rotation and nonlinearity. More generally, we highlight the utility of deep reinforcement learning in finance and "economic distillation" for model interpretation.

Keywords: Artificial Intelligence, Asset Pricing, Explainable AI, Machine Learning, Portfolio Theory, Batched/Offline Reinforcement Learning.

*We are grateful to Bing Han, Gavin Feng (discussant), Andrew Karolyi, and Andreas Neuhierl (discussant) for detailed feedback and to Si Cheng for kindly sharing the data on market illiquidity. We also thank David Avramov, Agostino Capponi, Ludwig Chincarini (discussant), John Cochrane, Gianluca De Nard (discussant), Jillian Grennan (discussant), Serhiy Kozak, Stefan Nagel, Markus Pelger, Marcos de Prado, Amin Shams (discussant), Jinfei Sheng, Stathis Tompaidis, Michael Weber, Dacheng Xiu (discussant), Mao Ye, Lu Zhang, Guofu Zhou, Luofeng Zhou, and conference and seminar reviewers and participants at the American Finance Association Annual Meeting, Australasian Banking and Finance conference, Blackrock FMG Webinar, Cheung Kong Graduate School of Business (CKGSB), China International Conference in Finance (Shanghai 2021), 3rd China International Forum on Finance and Policy (CIFFP), Cornell University SC Johnson College of Business, Center for Research in Economics and Statistics (CREST) and École Polytechnique, Econometric Society World Congress (Milan), Financial Markets and Corporate Governance Conference (FMCG), Georgetown University Global Virtual Seminar Series on FinTech "Machine Learning Day," Global Digital Economy Summit for Small and Medium Enterprises (DES2020), Global Quantitative and Macro Investment Conference (Wolfe QES), IIF International Research Conference & Award Summit, 2021 INFORMS Annual Meeting, INQUIRE UK/Europe Webinar, London Quant Group (LQG) webinar, Luohan Academy Webinar, 13th International Risk Management Conference (IRMC), Machine Lawyering Conference: Human Sovereignty and Machine Efficiency in the Law, Midwest Finance Association Annual Meeting, Shanghai Advanced Institute of Finance (SAIF), Southwestern Finance Association Annual Meeting, Annual Meeting of the Swiss Society for Financial Market Research (SGF), Stanford SITE "Macro Finance and Computation"; Toulouse School of Economics, University of Virginia McIntire School of Commerce and Darden School of Business, World Finance Conference, Xi'an Jiaotong University, Yangtze River Delta International Forum "Corporate Finance and Financial Markets," and Zhongnan University of Economics and Law for their comments. Fujie Wang, Hanyao Zhang, and Guanyu Zhou provided excellent research assistance. This research was funded in part by the Ewing Marion Kauffman Foundation, INQUIRE UK, ICPM Research Award, and Cornell Center for Social Sciences. The contents of this article are solely the responsibility of the authors. The paper includes partial results from an earlier paper under the title "AlphaPortfolio for Investment and Economically Interpretable AI." Cong is at Cornell University SC Johnson College of Business and the FinTech Initiative; Tang is at Tsinghua University Institute of Economics; Wang and Zhang are at Beihang University School of Computer Science and Engineering. Corresponding author: Cong at will.cong@cornell.edu

1 Introduction

Portfolio management traditionally entails first minimizing pricing errors or estimating risk premia from historical samples and then combining assets to achieve investment objectives.[1] Such an approach has serious drawbacks due to large estimation errors in the first step and the fact that the objectives in the two steps are not necessarily aligned. Extracting signals that are directly relevant for maximizing the portfolio objective is intuitively appealing yet under-explored. Furthermore, financial or economic data tend to be high-dimensional, noisy, and nonlinear, with complicated interaction effects and fast, non-stationary dynamics, rendering traditional econometric tools ineffective in capturing path dependence and cross-asset linkages. Recent studies have actively adopted machine learning (ML) or neural networks to tackle these challenges under the conventional two-step approach. While some make significant progresses (e.g., Freyberger, Neuhierl, and Weber, 2020; Feng, He, and Polson, 2018), many are not robust under plausible economic restrictions, as discussed in, e.g., Avramov, Cheng, and Metzker (2019).

To overcome these challenges, we take a novel data-driven approach to directly optimize portfolios, utilizing the strength of deep reinforcement learning (RL)—a class of AI models proven to be effective in applications such as computer vision, interactive games, and self-driving (e.g., Mnih, Kavukcuoglu, Silver, Rusu, Veness, Bellemare, Graves, Riedmiller, Fidjeland, and Ostrovski, 2015; Silver, Schrittwieser, Simonyan, Antonoglou, Huang, Guez, Hubert, Baker, Lai, and Bolton, 2017). Our key insight is that given the complexity of financial markets and asset price dynamics, searching using trial-and-error through a flexible modeling space to maximize a portfolio management objective can be more effective than attempting to estimate exogenously specified moments of the return distributions of assets (e.g., expected returns or variance) or to price them accurately regardless of their relevance for constructing the desirable portfolio. To this end, we develop a new AI model tailored for financial markets, letting data dictate which SDF moments or which assets to emphasize in model training in order to directly maximize portfolio performance metrics.

We use RL—a methodology derived from multi-arm bandit problems and approximate solutions for large-scale Markov Decision Processes because historical optimal portfolios are not labeled and trading may interact with the market states. RL either entails online interaction with the environment to generate additional data or exploits stochastic gradient descent for complex model search (e.g., Friedman, 2002) on historical data. Training an RL model mimics how practitioners actually develop strategies by trying tentative strategies out of a vast strategy space and optimize them gradually based on performance feedback de Prado (2018). We are the first to highlight RL's potential advantages over the widely applied su-

[1] Mean-variance optimization based on the investor's preference is one example (Markowitz, 1952). Practitioners and researchers also routinely construct portfolios by sorting based on asset characteristics and simple weight adjustments to manage the portfolio risk.

pervised learning framework and to tailor RL to multi-sequence learning with methodology innovations. Our study also adds to the fast-emerging literature in computer science on offline deep RL by developing a framework for its applications in portfolio management and investment advising.

Despite the efficacy and applicability of AI models, the black-box nature of advanced AI tools may hinder their wide use in finance and economics where interpretation is integral. Like many other models, our deep RL approach is subject to critiques on the complex nature of the algorithm and the lack of transparency. Meanwhile, in a world divided by discrimination and injustice, it is insufficient to attribute all biases in AI to training data either; understanding models as a starting point for improving algorithmic fairness also constitutes a pressing issue.[2] Our second objective is then to understand how various innovations in our model contribute to the performance and to introduce "economic distillation" that lends greater interpretability and transparency to complex AI models by projecting them onto linear modeling or natural language spaces. The polynomial-sensitivity and textual-factor analyses we devise not only provide initial insights into our AI model, but also can be used in other applications in social sciences.

Specifically, we adopt the latest sequence representation extraction models (SREM), such as Transformer Encoder (TE) and Long Short-Term Memory (LSTM), in order to flexibly and effectively represent and extract information from the time series of input features such as firms' fundamentals and market signals, i.e., the states of the environment. We develop a deep-neural-network-based panel data analytics by adding our novel cross-asset attention networks (CAANs) that capture attribute interactions across assets. We then generate a "winner score" to rank assets and trade (the policy and action), and subsequently evaluate how the portfolio performs, i.e., examining the rewards. Our emphasis is not on any specific functional form or tuning parameter of the model, but on the data-driven approach that takes the joint distribution of asset returns as unknown, observes the outcomes of the trading action and its interaction with the environment (e.g., realized Sharpe ratio with or without having the trades impact the market state), tests a range of actions in each state (e.g., various portfolio weights), and then dynamically explores a high-dimensional parameter space to maximize the objective without noisy or potentially misspecified intermediate steps. We show that the combination of SREM and CAAN (to capture the nonlinearity, cross-sectional linkages, and path-dependence in asset returns), and the direct construction through RL both contribute to the performance. We also discuss how AlphaPortfolio allows general management objectives as well as market interactions entailing transaction costs, dynamic budgets, and experience-dependent preferences.

[2]Society increasingly demands transparency and interpretability of algorithmic decisions (Goodman and Flaxman, 2017; Barocas, Hardt, and Narayanan, 2017, and articles 13-15 of European Union's General Data Protection Regulation (GDPR)).

Our deep-RL-based "direct construction" improves portfolio performance out-of-sample (OOS) drastically and the results remain robust after imposing various economic constraints. In an illustrative study of U.S. equities, we use hundreds of firm characteristics and market signals in a 12-month historical window as predictor variables, similar to Freyberger, Neuhierl, and Weber (2020). In the baseline specification, we focus on the subsequent 12-month average OOS Sharpe ratio as the investors' objective, and train a portfolio model (henceforth referred to as "AlphaPortfolio" or "AP") with monthly rebalancing which generates a Sharpe ratio consistently above two on both the full test sample (1990-2016) and the subsamples excluding microcaps (10% or 20% based on market cap). The annualized excess alpha after controlling for various factors (CAPM, Fama-French-Carhart factors, Fama-French-Carhart plus liquidity factors, Fama-French five factors, Fama-French six factors, Stambaugh and Yuan factors, Q4 factors) also consistently exceeds 13.5%. AP's general performance metrics (e.g., turnover and maximum drawdown) are significantly better than those of most known anomalies and machine-learning strategies.[3]

In terms of the specified portfolio management objective, RL-based AP consistently achieves a high OOS Sharpe ratio (reaching 4.7 in the early years of the test sample and above 1.4 throughout the entire test sample), more than doubling the performance of TE-CAAN under the two-step conventional construction. Deep learning (e.g., TE, a cutting-edge AI tool typically used for supervised machine translation to solve the vanishing and exploding gradient problems in Recurrent Neural Networks), obviously contributes to the performance by better handling the high-dimensionality, nonlinearity, path-dependence, etc., of the financial data and asset return dynamics, but CAAN improves OOS performance by a significant margin relative to a plain-vanilla TE model (e.g., a 0.4 improvement in Sharpe ratio) as well. The findings are robust in that they are not driven by short positions alone, ad hoc weighing, high-frequency trading, or particular industry sectors; the outperformance persists under alternative turnover measures, exclusion of unrated and downgraded firms as well as restricted testing samples in recent years or during episodes of different market sentiment, volatility, and liquidity. AP is therefore robust to imposing various economic restrictions that Avramov, Cheng, and Metzker (2019) identify to significantly hamper performances by other machine learning strategies.

Our deep RL approach is fundamentally different from earlier studies and common industry practices in that it combines the power of flexible deep learning models and direct portfolio optimization.[4] RL entails learning how to map situations to actions to maximize a

[3]For example, among successful ML models, Gu, Kelly, and Xiu (2020) achieves an OOS Sharpe ratio of 1.35 with value-weighted long-short portfolios constructed from neural network models and Freyberger, Neuhierl, and Weber (2020) achieves an OOS Sharpe ratio of 1.6 using group Lasso and excluding the bottom 10% small stocks; the maximum drawdowns of common factors such as Fama-French-Carhart four factors over the same time period lie between 40% -60% whereas those of AP lie between 2%-10%.

[4]The idea of directly draw inferences about the optimal portfolio weights from the data was explored in a number of brilliant articles as early as Brandt (1999). The studies all rely on supervised learning (either

numerical reward. Unlike supervised learning in which the learner is told what the correct actions are in the training, RL discovers the best actions for some delayed rewards through trial-and-error search and utilizing feedback from the environment (Sutton and Barto, 2018, p.1). In the context of average OOS monthly Sharpe ratio, the reward is "delayed" in that it is computed over a multi-month window: A portfolio construction in one month could affect the future market environments and thus future portfolio construction once we take into consideration elements such as a manager's portfolio size and transaction costs.[5]

To see how our approach is a more "direct" construction than traditional approaches, let us write down a generic problem researchers tackle using supervised learning: $\min_\theta H(\theta) = (y - f(x,\theta))^2$, where $f(x,\theta)$ is the model we train and θ, x, and y represent the model parameters, input variables, and labeled outputs, respectively. For example, y could be an asset's return, and we are then simply minimizing pricing errors. With known functional form of $\frac{\partial H(x,\theta)}{\partial \theta}$ to minimize H using gradient descents, the two-step paradigm for portfolio construction then takes the estimated function f^* as given to optimize the best portfolio strategy, z, to maximize a reward function R, i.e., $\max_z R(x, z(f^*(x,\theta)))$. For example, if f^* provides the estimates of expected returns and variance-covariance matrix for the assets, and R is investors' next period mean-variance utility, then the optimal construction strategy z is the solution in Markowitz (1952).

Many machine learning models make significant progresses in overcoming the difficulty of estimating $f(x,\theta)$ with scarce yet high-dimensional data, but there is no guarantee that the $H(\theta)$-minimizing f^* also maximizes R in general. A simple case in point is the construction of long-only portfolio. If some assets generate extremely negative returns and would not be included in the portfolio in any case, using historical data to train a model to minimize estimation errors in these assets' higher moments is not necessarily helping improving the portfolio's performance.

A direct portfolio construction formulates the optimization as $\max_\theta R(x, z(x,\theta))$, where $z(x,\theta)$ is a direct investment strategy model of which the output is a portfolio, and the reward $R(\cdot)$, which could be the average Sharpe ratio, average returns after transaction costs, cumulative returns over an investment period, etc., is a general function of the dynamic

regression-based or non-parametric) and Brandt (2010) aptly summarizes their procedures and limitations due to model misspecification and the high-dimensionality and nonlinearity of inputs. In particular, Brandt, Santa-Clara, and Valkanov (2009) employs a linear specification where as the neural-network-based structure here helps capture non-linear and interacting effects more effectively.

[5]Unlike applications in science and engineering (e.g., gaming) for which creating the large training sets needed for RL (e.g., through experimentation) is cost-effective, trying various portfolio constructions in practice entails significant costs and high stakes. We therefore often need either simulations or a model of how our actions impact the market environment. In our baseline empirical tests, we follow the computer science literature to take the impact going forward of trading a particular portfolio to be negligible. It is worth noting that even without the actions' impacting the environment, RL could be useful (e.g., Delarue, Anderson, and Tjandraatmadja, 2020). We later analyze several interesting and complex extensions to illustrate how our RL framework can incorporate rich interactions with the environment and alternative objectives, as well as how AP performance remains robust.

portfolio strategy z and market environment x. Unlike the values of y in supervised learning that are typically explicit or observed, we often do not observe z (and R) directly, not to mention that in reality, z could alter x as well. Therefore, we use RL to "explore" different investment strategies (different θ) under different market environments (different x) to see what rewards we get. This process is data-driven without requiring analytical models of z and is equivalent to a Monte Carlo sampling of the investment path and the reward function $R(\cdot)$. RL uses gradient-based solutions to guide the sampling with the objective of maximizing R and has been found empirically to be more effective than supervised learning in complex environments, as seen in various AI applications, such as AlphaGo and self-driving.

A supervised learning approach to portfolio construction necessarily introduces a separation: $\max_z R(x, z)$ and $\min_\theta H(\theta) = (z^* - z(x, \theta))^2$, i.e., we maximize the reward function with respect to portfolio construction strategy z, which in turn is estimated in a minimization problem requiring historical values of the optimal z^*. But z^* typically cannot be directly observed or accurately approximated or easily computed from the training data. Hence, a supervised learning approach for maximizing R also has to resort to Monte Carlo sampling in some sense. Traditional two-step constructions do not aim to maximize R when they generate the sampling of y. Even when the sampling is guided by maximizing R, a supervised learning formulation can still have a prediction loss gap between z^* and $z(x, \theta)$. Take OOS Sharpe ratio for example. Even if one directly optimizes portfolio weights using supervised learning, one has to supply the "correct" maximal Sharpe ratio in the training, which requires either exhaustively computing possible portfolio constructions to get the historical maximal Sharpe ratios or introducing an intermediate proxy for it that is subject to model misspecification and approximation errors.

In contrast, deep RL can handle unlabeled data and is computationally feasible, making it well-suited for the direct construction task.[6] Moreover, it easily accommodates more general performance metrics as well as dynamic interactions between investment actions and the market environment, allowing portfolio constructions to incorporate price impacts, transaction costs, dynamic budget constraints, clients' long-range goals, etc. (Fischer, 2018). In Section 4.4, we provide multiple illustrations that correspond to real life applications such as fund survival subject to withdrawals and investment manager compensation. As seen through these examples, Deep RL frequently involves models of the environment (so it is not an exact counterfactual) and offline learning using historical data (instead of online interactions with the environment to obtain new data). It also goes beyond portfolio construction using SDF recovery since the portfolio objective is not restricted to Sharpe-ratio maximization only.

We note that deep RL applications do not have to always involve online interactions and

[6]We are not claiming that there cannot be an effective supervised-learning-based approach for direct construction. On the contrary, we believe it is an interesting topic for future research.

offline RL can be particularly useful in social sciences. Unlike in science labs, social scientists typically cannot generate data through online interactions, either because data collection is expensive (e.g., in robotics, trading, educational agents, or healthcare) or dangerous (e.g., in autonomous driving, or healthcare). Moreover, even in domains where online interaction is feasible, we still want to utilize previously collected data instead for example, if the domain is complex and effective generalization requires large datasets. AlphaPortfolio's training is offline RL, but it interacts with the environment to generate new data through the rolling updates in the test sample. In that sense, our RL model is a hybrid and the AP framework is conveniently deployable by practitioners and robo-advisors for trading and investment advising.

Beyond articulating this theoretical advantage of Deep RL for direct portfolio construction, we also aim to better interpret AP. We use gradient-based methods and Lasso to distill the model into a linear model with a small number of input features, while allowing higher-order terms and feature interactions. This novel polynomial sensitivity analysis essentially "projects" a complex model onto a space of linear models. It adds to the advances of explainable AI by combining the strength of surrogate modeling and feature importance analysis. The distilled model informs us of features driving AP's performance. Besides some usual suspects such as Tobin's Q, inventory changes (ivc), changes in shares outstanding (delta_so), etc., also play dominant roles. In addition, we find higher-order terms (e.g, ivc^2) affect AP's behavior but not interaction effects (which could still be important for estimating assets' returns or a pricing kernel). Finally, we observe short-term reversals and identify important features dominant throughout and others rotating in and out. In particular, market trading signals and firms' fundamentals and financials take turns to dominate (correlation of -0.33).

As another illustration of economic distillation through projection, we apply the textual-factor analysis, an analytic combining the strengths of neural network language processing and generative statistical modeling (Cong, Liang, and Zhang, 2018), to understand the behavior of AP based on texts from firms' filings. By projecting it onto a natural language space, we find that AP buys stocks of firms whose 10-K and 10-Q talk about sales, profitability, loss-cutting, etc., whereas it short-sells stocks of firms that prominently mention real estates, mistakes, and corporate events, among others. Economic distillations not only provide initial interpretations of complex models so that we can avoid pitfalls of AI applications when the market environment or policy changes, but also provides a sanity check on coding errors and model fragility. Both the polynomial sensitivity analysis and the textual-factor analysis are new and complement attempts in computer science concerning explainable AI and in economics concerning interpreting ML models.

We organize the remainder of the article as follows. Section 2 provides the background and clarifies our contributions as an interdisciplinary study; Section 3 describes our model and methodology; Section 4 applies the AlphaPortfolio model to U.S. equities; Section 5

introduces economic distillations to interpret the model; Section 6 concludes with a discussion on the general utility of reinforcement learning and implications of interpretable AI in social sciences; the appendices contain foundations of reinforcement learning, a description of variable construction, implementation details of AP using both Transformer and LSTM modules, and economic distillation using textual factors.

2 Related Literature and Contributions

As one of the earliest studies to apply recent breakthroughs in AI to portfolio management, our paper makes three main contributions. (i) We develop an RL-based framework to directly optimize investors' objectives which can accommodate unlabeled financial data and complex environments and overcomes the challenges in the conventional two-step approach. (ii) AP is designed to handle the distinguishing features of financial big data and outperforms most existing strategies (traditional or machine-learning-based), especially after imposing reasonable economic constraints and restrictions. We demonstrate that cutting-edge AI tools typically used for machine translation can be effective and immediately deployable in practice once properly tailored to economic and financial applications. (iii) We provide general, expandable, and intuitive procedures for economically interpretable AI in social sciences that complement endeavors from computer science and machine learning fields.

2.1 Portfolio Theory and Investment Advising

Our paper foremost adds to portfolio theory. The trading strategy aids both institutional investors and retail investors (potentially as a second-generation robo-advisor) and more broadly provides insights on portfolio theory and applications of RL in finance.

Conventional paradigms. Following Markowitz (1952), the typical portfolio construction consists of two steps: (i) estimate population moments using available samples and (ii) optimize over possible combinations of assets, or simply sort and assign ad hoc weights.

Estimating returns accurately is extremely difficult due to the lack of long time series of data (e.g., Merton, 1980), while estimates of variance-covariance are rarely well-behaved (e.g., Green and Hollifield, 1992). This leads to unstable and extremely positive and negative weights in the second-step portfolio construction, resulting in poor OOS performance and implementability in practice. This "error-maximizing" problem of the mean-variance portfolio (e.g., Best and Grauer, 1991) essentially derives from the fact that the second-stage optimization exploits too many small differences in the first-stage estimates without properly considering their estimation errors.

Models explicitly designed to mitigate the estimation errors include the Bayesian approach to estimation errors with diffuse-priors (Barry, 1974; Bawa, Brown, and Klein, 1979), "shrinkage estimators" (e.g., Jobson, 1979; Jorion, 1986), model-based priors (Pástor, 2000; Pástor and Stambaugh, 2000), robust portfolio allocation (Goldfarb and Iyengar, 2003; Garlappi, Uppal, and Wang, 2006), estimation risk, and optimal diversification (e.g., Klein and Bawa, 1976; Kan and Zhou, 2007).[7] But the estimation errors are so problematic that most attempts achieve very moderate success (DeMiguel, Garlappi, and Uppal, 2007).[8] Though ad hoc sorting strategies based on asset characteristics avoid the estimation errors altogether, they are limited in handling high-dimensional features or capturing nonlinear effects.

Instead, we directly optimize the portfolio's performance metric using RL. Our approach is motivated by the possibility that: (i) The relationship between the portfolio weights (as complicated functions of the return distribution) and the predictors could be less noisy than the relationship between the individual moments and the predictors. (ii) Intermediate estimation of the return distribution may introduce additional noise and potential misspecifications. (iii) Given the end goal, a global optimization, albeit imperfect, may work better than two sequential optimizations. A holistic approach such as ours goes beyond linear shrinkage and shares the spirit of Ledoit and Wolf (2012); instead of explicitly applying differential shrinkage just for the variance-covariance matrices (e.g., Ledoit and Wolf, 2017), AP differentially weigh (in addition to SDF moments and assets) inputs in the entire feature space based on their relevance for the objective.

Direct derivation of optimal portfolio weights. Brandt (1999) is among the earliest studies that focus directly on the dependence of the portfolio weights on the predictors rather than model the conditional return distribution. The simplest approach entails parametrized portfolio weights as functions of observables (e.g., Brandt and Santa-Clara, 2006; Brandt, Santa-Clara, and Valkanov, 2009), but suffers from potential mispecification of the portfolio weight function which is not necessarily linear in reality. While nonparametric or locally parametric estimators from sample analogues of the FOCs or Euler equations can guard against mispecification (Brandt, 1999), the curse of dimensionality when directly deriving optimal weights using kernel methods or polynomial expansions limits reliable implemen-

[7] Popular among practitioners are Black-Litterman models combining model-based priors with investors' subjective beliefs (Black and Litterman, 1990, 1992) and the "risk parity" approach (e.g., Jurczenko, 2015), which nevertheless suffers from instability and that the variance-covariance matrices are not always positive-definite for easy inversion (e.g., Bailey and de Prado, 2012; de Prado, 2016).

[8] The authors show that various models explicitly developed to deal with the estimation errors fail to beat the naive benchmark (each of the N assets available for investment gets a fraction $1/N$ of the total wealth at rebalancing) in terms of Sharpe ratio, certainty-equivalent return, and turnover, because in practice one has either very short estimation windows or small true Sharpe ratios of the efficient portfolio to start with or small portfolio sizes. Investors typically demand diversified portfolios and even for portfolios with no more than 50 assets, extant models are often estimated using five to ten years of data (instead of the decades of data the authors found necessary for reducing estimation errors sufficiently).

tation with more than two predictors (e.g., Brandt, 2010) unless one specifies additional structures for dimension reduction, such as variants of index regressions (Powell, Stock, and Stoker, 1989; Aït-Sahali and Brandt, 2001). For all these reasons, direct construction of portfolios still remains underexplored.

Our deep learning approach recognizes that high-dimensional and noisy financial data often limit the estimation of asset return distributions and model specification. For example, regularized linear models or PCA rotations in recent studies only partially take care of nonlinearities by including, at the discretion of researchers, some transformations of linear input signals (e.g. Kozak, Nagel, and Santosh, 2020; Bryzgalova, Pelger, and Zhu, 2020; Lettau and Pelger, 2020). Our model picks up nonlinear effects (with activation functions in the neural networks) that Aït-Sahali and Brandt (2001) do not identify. But deep learning's flexible model structure implies a much larger parameter space for us to train the model from unlabeled data, necessitating the need for RL rather than supervised learning used in the literature, as we explained in the introduction.

While recent studies such as Kozak, Nagel, and Santosh (2020) and Bryzgalova, Pelger, and Zhu (2020) complement our paper in demonstrating how ML techniques can extract signals relevant for portfolio design in addition to return prediction, they base penalties on the maximum squared Sharpe ratio implied by the SDF and portfolio construction is a means to an end (recovering robust pricing kernels or maximizing cross-sectional OOS R^2). In our paper, optimizing a portfolio's performance metric (e.g., OOS Sharpe ratio and beyond) directly is the end itself. The general applicability to other investor objectives is not something estimations of pricing kernel automatically achieve. Our paper therefore offers the first ML model for directly optimizing general portfolio management objectives, not to mention that RL allows us to handle unlabeled data and potential market interactions.

RL, investment planning, and robo-advising 2.0. Despite an outburst of media articles and industry reports discussing trends in robo-advising and investment planning, current applications are limited in scope and functionality, and do little on active strategies, etc. (e.g., D'Acunto and Rossi, 2020). Relative to the first-generation robo-advisors which mostly help clients avoid behavioral biases and manage asset allocation and factor exposure through trading ETFs and smart-beta products (Cong, Huang, and Xu, 2020), future robo-advisors likely automate more active strategies and customize the service according to individual investors' preference, tax situation, risk aversion, portfolio constraints, investment horizon, and transaction costs (e.g., Detemple and Murthy, 1997; Liu and Loewenstein, 2002).

These considerations are exactly what the "delayed rewards" feature of RL can easily address. RL can incorporate agents' impacts on the environment, the investors' evolving preference and liquidity needs as well as trading costs and dynamic budget constraints, not to mention the possibility of learning investors' preferences from their investment ac-

tions (Alsabah, Capponi, Ruiz Lacedelli, and Stern, 2019). For example, if the delayed reward is set to be the stable benefits payout for retirees, RL can allow pension investment to better allocate investment across asset classes and even go beyond government securities, investment-grade bonds, and blue-chip stocks, etc., to maximize the fund's objectives. Moreover, economic interpretability offers greater transparency, meeting the demand of investment advisors or robo-advisors to adjust models and convey investment principles to clients.[9] Our AP is therefore useful for goal-based wealth management (Dasa, Ostrova, Radhakrishnanb, and Srivastavb, 2018).

2.2 Machine Learning and AI Applications in Finance

Our paper contributes to an emerging literature that applies machine learning in economics for forecasting macroeconomic outcomes, asset returns, corporate defaults, risk exposures, etc., and for analyzing unstructured data, such as texts.[10] Data in social sciences could differ drastically from data in science and engineering fields. Besides high dimensionality and nonlinearity (e.g., Cochrane, 2011; Harvey, Liu, and Zhu, 2016; Karolyi and Van Nieuwerburgh, 2020) that ML packages from science and engineering help address, financial data are often characterized by low signal-to-noise ratio, significant interaction effects, and non-stationary/fast dynamics.[11]

Existing studies typically focus on supervised learning with several non-mutually exclusive lines of work (de Prado, 2018). Dimension reduction involves either regularization methods, such as Lasso, Ridge, or Elastic Net (e.g., Rapach and Zhou, 2019; Feng, Giglio, and Xiu, 2020), or rotation and clustering techniques, such as PCAs (e.g., Kelly, Pruitt, and Su, 2019; Kozak, Nagel, and Santosh, 2020; Kim, Korajczyk, and Neuhierl, 2019; Chinco, Neuhierl, and Weber, 2019). A second line aims at capturing interactions and nonlinear effects through semi-parametric, distribution-free, or flexible but complex model architectures, such as group Lasso, splines, and ensemble learning (e.g., Freyberger, Neuhierl, and Weber, 2020; Light, Maslov, and Rytchkov, 2017; Rossi, 2018; Moritz and Zimmermann, 2016).

Modern machine learning mostly concerns neural-network-based deep learning that only gained prominence over the past decade. It is quickly adopted by asset pricers for studying

[9]Big data analytics and AI are deemed important components of the next-generation robo-advisors. (EY "The Evolution of Robo-advisors and Advisor 2.0 model" 2018, Scott Becchi, Ugur Hamaloglu, Taroon Aggarwal, Samit Panchal.) For example, Numerai, an AI-based hedge fund with native token Numerarie, already plans for an app Daneel that combines robo-advising and personal assistant. Sharpe Capital is another app that uses ML. Users typically expect to understand what robo-advisors do at a high level.

[10]Cochrane (2011) specifically called for methods beyond cross-sectional regressions and portfolio sorts. Rapach, Strauss, and Zhou (2013); Gu, Kelly, and Xiu (2020) are also among the early contributions in finance.

[11]The non-stationary dynamics relates to the Lucas critique in finance settings. While hotdogs do not change their shape in response to image classification, investors alter their behaviors after others' using ML tools. The low signal-to-noise also necessitates the use of OOS performance instead of in-sample predictability (Martin and Nagel, 2019).

pricing kernels and reducing pricing errors (e.g., Feng, Polson, and Xu, 2018; Chen, Pelger, and Zhu, 2020), but Avramov, Cheng, and Metzker (2019) caution that machine learning's superior performance could be driven by microcaps and value weighing as are traditional anomalies (Hou, Xue, and Zhang, 2020). A fundamental difference between our approach and much of the literature lies in the objective. Most studies focus on estimating risk premia or characterizing the SDF, our paper directly maximizes performance metrics for portfolio construction. Recent finance applications of deep learning (with Feng, He, and Polson, 2018; Cong, Tang, Wang, and Zhang, 2020, etc. as exceptions) do not use multi-sequence modeling to capture both long-range and cross-asset dependences in asset features. Our paper attempts to fill in the gap and also emphasizes hitherto understudied interpretations of complex ML models.[12]

A recent literature in AI has been actively studying data-driven reinforcement learning that utilizes only previously collected offline data, without any additional online interaction (e.g., Fu, Kumar, Nachum, Tucker, and Levine, 2020). A number of papers have illustrated the power of such an approach in enabling data-driven learning of policies for dialogue (Jaques, Ghandeharioun, Shen, Ferguson, Lapedriza, Jones, Gu, and Picard, 2019), robotic manipulation behaviors (Ebert, Finn, Dasari, Xie, Lee, and Levine, 2018; Kalashnikov, Irpan, Pastor, Ibarz, Herzog, Jang, Quillen, Holly, Kalakrishnan, Vanhoucke, et al., 2018), and robotic navigation skills (Kahn, Abbeel, and Levine, 2021). We contribute to this literature involving offline deep RL, also known as batch RL (e.g., Fujimoto, Meger, and Precup, 2019; Kidambi, Rajeswaran, Netrapalli, and Joachims, 2020), by building the first multi-sequence offline RL for portfolio management and demonstrate its advantages over traditional approaches. Overall, our application of deep RL is motivated economically by the challenges observed in portfolio theory and practice as well as the pressing need for greater interpretability (Karolyi and Van Nieuwerburgh, 2020).[13]

2.3 Interpretable AI and Data Science

Economists have recently started to discuss the socioeconomic implications of AI and data science, such as discrimination, data privacy, and macroeconomic outcomes (e.g., Bartlett, Morse, Stanton, and Wallace, 2019; Liu, Sockin, and Xiong, 2020; Farboodi and Veldkamp, 2019). Underlying these issues is that data have massively expanded in volume, velocity, and

[12]Among notable exceptions regarding interpreting complex ML models, Sak, Huang, and Chng (2019) uses characteristic sorts and Stambaugh and Yuan (2017)'s mispricing factors to identify monthly dominant characteristics and ascertain the ex-post source of alpha.

[13]Koray Kavukcuoglu, the director of research at Deepmind, is quoted for saying: "Reinforcement Learning is a very general framework for learning sequential decision making tasks. And Deep Learning, on the other hand, is of course the best set of algorithms we have to learn representations. And combinations of these two different models is the best answer so far we have in terms of learning very good state representations of very challenging tasks that are not just for solving toy domains but actually to solve challenging real world problems" (Garychl, 2018).

variety, and tools for analyzing them have become too complex to inspect or understand. As such, economic interpretability has become critical when applying machine learning or big data analytics in social sciences. Our study is among the earliest to emphasize interpreting deep reinforcement learning models in finance and economics.

Our paper contributes to an emerging literature in computer science and machine learning on model compression or distillation (e.g., Bucilu, Caruana, and Niculescu-Mizil, 2006; Hinton, Vinyals, and Dean, 2015). Unlike their distillations that typically still have a large set of features and are aimed at deployment and computational efficiency, our "economic distillation" advocates using the original complex model for prediction and using the distilled model to identify key drivers. Our objective completely differs and focuses on taking a first step to understand the underlying mechanism from an opaque model.

Our approach also falls into the area of explainable AI (XAI, e.g., Guidotti, Monreale, Ruggieri, Turini, Giannotti, and Pedreschi, 2018; Horel and Giesecke, 2019a). From the functional perspective, *local XAI* tries to understand why the model makes a specific decision for a certain input, whereas *global XAI* tries to elucidate the model logic and rules. From the methodological perspective, XAI entails either *surrogate modeling* or *feature importance extraction*. Surrogate models use decision trees, rule sets, linear or generalized additive models, etc., to proxy the neural network to be interpreted (e.g., Wu, Hughes, Parbhoo, Zazzi, Roth, and Doshi-Velez, 2018; Ribeiro, Singh, and Guestrin, 2016). Feature importance extraction focuses on analyzing contributions of feature inputs to model outcomes, including gradient-based sensitivity analysis (e.g., Sundararajan, Taly, and Yan, 2017; Wang, Zhang, Tang, Wu, and Xiong, 2019), Partial Dependence Plots (PDPs, e.g., Krause, Perer, and Ng, 2016), and significance tests for single-layer neural networks (Horel and Giesecke, 2019b).

We contribute conceptually by introducing projections of AI models onto relatively transparent and interpretable modeling spaces. Methodology-wise, we are the first to expand feature sensitivity analysis and combine it with surrogate modeling to achieve global interpretability. Our polynomial sensitivity analysis thus improves upon conventional gradient-based methods and captures higher-order and interaction effects in economic data.[14] Moreover, we complement our polynomial sensitivity analysis with textual-factor analysis to enhance economic interpretability, which is integral to research in social sciences (Athey, 2018). To our knowledge, we are the first to use texts to improve economic interpretability of big data and AI models.

[14] As two exceptions, Datta and Sen (2018) build on the concept of Shapley value to develop a Quantitative Input Influence method, and demonstrate input influences can be summarized using clustering approaches, and Horel and Giesecke (2019a) develop computationally efficient feature significance test that can not only identify feature interactions of any order but also generate model-free feature importance measures. However, these approaches are not designed for RL-based applications.

3 Model and Methodology

In this section, we focus on the design of AlphaPortfolio. Figure 1 illustrates the overall architecture, which consists of three components. The first component entails using SREMs to extract a representation for each asset from its state history. Next, we introduce a Cross-Asset Attention Network (CAAN) which takes the representations of all assets as inputs to extract representations that capture interrelationships among the assets. The third component is a portfolio generator, which takes the scalar winner score for every asset from CAAN and derives the optimal portfolio weights. Importantly, we embed this AlphaPortfolio strategy into a reinforcement learning framework to train the model parameters to maximize an evaluation criterion, such as the OOS Sharpe ratio. We describe the development of deep sequence modeling in Cong, Tang, Wang, and Zhang (2020) and the basics of reinforcement deep learning in Appendix A.

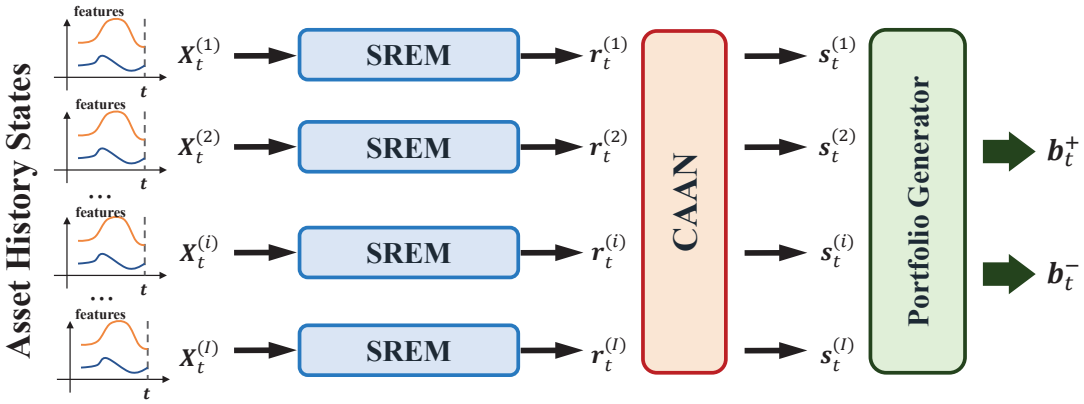


Figure 1: Overall Architecture of AP.

Why one-step portfolio optimization using RL? As we described in the introduction and Section 2, one-step likely works better than the combination of two-step indirect optimizations. Moreover, RL can better handle complex objectives and interactions with the environment, which allow the incorporation of budget constraints, long-term goals, etc.

3.1 Sequence Representations Extraction

The return distribution of an asset has close relationships with its historical states. The historical states of assets are naturally formed as sequence observations. We use vector $\tilde{\boldsymbol{x}}_t^{(i)}$ to denote the state history of asset i at time t, which consists of asset features/firm characteristics, for example, as given in Section 4.1. We name the last K historical holding periods at time t, *i.e.,* the period from time $t-K$ to time t, as a *look-back window* of t. One example is features from the previous 12 months when we construct a portfolio for the 13th month. The historical states of an asset in the look-back window are denoted as a sequence

$\boldsymbol{X}^{(i)} = \left\{\boldsymbol{x}_1^{(i)}, \cdots, \boldsymbol{x}_k^{(i)}, \cdots, \boldsymbol{x}_K^{(i)}\right\}$, where $\boldsymbol{x}_k^{(i)} = \tilde{\boldsymbol{x}}_{t-K+k}^{(i)}$.

For each asset i, SREM learns representation $\boldsymbol{r}^{(i)}$ from its state history $\boldsymbol{X}^{(i)}$ (we omit time t without loss of generality). It is notable that SREM can be any kind of deep sequence models, such as RNN, LSTM, etc. In this paper, we focus on one of the two most cutting-edge deep sequence model (Cong, Tang, Wang, and Zhang, 2020), Transformer Encoder (TE). We discuss the other, LSTM with Historical Attention (LSTM-HA), in Section 5 and Appendix C. Both TE and LSTM-HA are specifically designed to handle sequential information and excel in representing complex information in non-linear time-series modules.

Both variants of recurrent neural networks (RNNs) and the TE-based (or LSTM-based) SREM we propose have been recently used in neural machine translation. Unlike RNNs, TE makes long-range dependencies in sequences easier to learn by reducing network path length and allows for more parallelization by reducing the reliance on the prohibitive sequential nature of inputs.

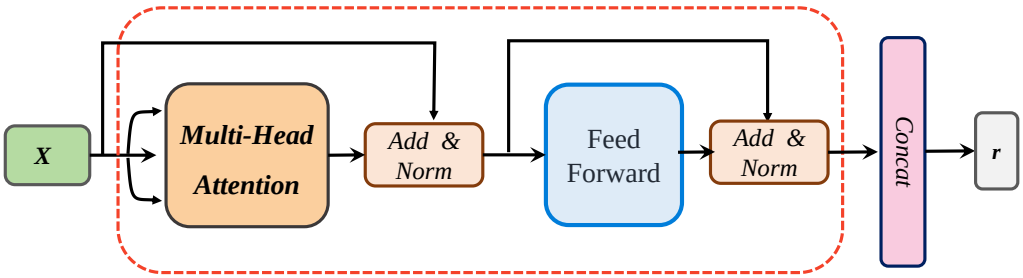


Figure 2: The Architecture of Transformer Encoder.

Figure 2 illustrates the architecture of a plain-vanilla TE. Here the encoder is composed of a stack of several identical layers. Each layer has two sublayers. The first is a multi-head self-attention mechanism, which we adopt and modify for AlphaPortfolio, and the second is a simple position-wise fully connected feed-forward network. In addition, residual connection and layer normalization are employed for each sublayer. We elaborate on implementation details in Appendix C. Overall, TE encodes the sequence input $\boldsymbol{X}^{(i)}$ into vector space as

$$\boldsymbol{Z}^{(i)} = \mathrm{TE}\left(\boldsymbol{X}^{(i)}\right), \tag{1}$$

where $\boldsymbol{Z}^{(i)} = \left\{\boldsymbol{z}_1^{(i)}, \cdots, \boldsymbol{z}_k^{(i)}, \cdots, \boldsymbol{z}_K^{(i)}\right\}$. The $\boldsymbol{z}_k^{(i)}$ is the hidden state encoded at step k, which takes all other steps into consideration. We concatenate all the steps in $\boldsymbol{Z}^{(i)}$ as a representation of the asset: $\boldsymbol{r}^{(i)} = \mathrm{Concat}\left(\boldsymbol{z}_1^{(i)}, \ldots, \boldsymbol{z}_k^{(i)}, \ldots, \boldsymbol{z}_K^{(i)}\right)$, which contains the global dependence among all elements in $\boldsymbol{X}^{(i)}$. In our model, the representation vector for all assets are extracted by the same TE, which means the parameters are shared by all assets. In this manner, the representations extracted by TE are relatively stable and generally applicable

for all assets available rather than for a particular one.

As mentioned earlier, learning long-range dependencies is a key challenge when using vanilla recurrent-based sequence models, *i.e.,* RNN and LSTM. In our design of sequence representation extraction module, LSTM-HA addresses this issue by introducing historical attention mechanism, and the Transformer architecture connects all positions in the sequence, which can effectively extract both short-term and long-term dependencies. We next use LSTM-HA and TE as the first-step SREM respectively and compare their performances. For the i-th stock at time t, the representation extracted by SREM is denoted as $\boldsymbol{r}_t^{(i)}$. It contains both the sequential and global dependences of stock i's historical states from time $t-K+1$ to time t.

3.2 Cross-Asset Attention Network & Winner Score Estimation

Prior attempts applying RL-based models from computer science typically stop at asset representations with a softmax normalization (Jin and El-Saawy, 2016; Deng, Bao, Kong, Ren, and Dai, 2017; Ding, Liu, Bian, Zhang, and Liu, 2018). We propose a CAAN to describe the interrelationships among assets. Note that our design of the CAAN module is inspired in part by the self-attention mechanism in machine translation (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin, 2017).

Figure 3 illustrates the architecture of CAAN. Specifically, given the asset representation $\boldsymbol{r}^{(i)}$ (we omit time t without loss of generality), we calculate a query vector $\boldsymbol{q}^{(i)}$, a key vector $\boldsymbol{k}^{(i)}$ and a value vector $\boldsymbol{v}^{(i)}$ for asset i as

$$\boldsymbol{q}^{(i)} = \boldsymbol{W}^{(Q)}\boldsymbol{r}^{(i)}, \quad \boldsymbol{k}^{(i)} = \boldsymbol{W}^{(K)}\boldsymbol{r}^{(i)}, \quad \boldsymbol{v}^{(i)} = \boldsymbol{W}^{(V)}\boldsymbol{r}^{(i)}, \tag{2}$$

where $\boldsymbol{W}^{(Q)}$, $\boldsymbol{W}^{(K)}$, and $\boldsymbol{W}^{(V)}$ are the matrices of parameters to learn that are asset-independent. The interrelationship of asset j to asset i is modeled as using the $\boldsymbol{q}^{(i)}$ of asset i to query the key $\boldsymbol{k}^{(j)}$ of asset j, *i.e.,* the rescaled inner product between $\boldsymbol{q}^{(i)}$ and $\boldsymbol{k}^{(j)}$:

$$\beta_{ij} = \frac{\boldsymbol{q}^{(i)\top} \cdot \boldsymbol{k}^{(j)}}{\sqrt{d_k}}, \tag{3}$$

where d_k is a rescale parameter to avoid the dot product from becoming too large.[15] Then, we use the normalized interrelationships $\{\beta_{ij}\}$ as weights to sum the values $\{\boldsymbol{v}^{(j)}\}$ of other assets into an attenuation score:

$$\boldsymbol{a}^{(i)} = \sum_{j=1}^{I} \text{SATT}\left(\boldsymbol{q}^{(i)}, \boldsymbol{k}^{(j)}\right) \cdot \boldsymbol{v}^{(j)}, \tag{4}$$

[15] Assume that the components of q and k are independent random variables with mean 0 and variance 1. Then their dot product, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.

where the self-attention function $\mathrm{SATT}(\cdot,\cdot)$ is a softmax normalized interrelationships of β_{ij}, *i.e.*,

$$\mathrm{SATT}\left(\boldsymbol{q}^{(i)},\boldsymbol{k}^{(j)}\right)=\frac{\exp\left(\beta_{ij}\right)}{\sum_{j'=1}^{I}\exp\left(\beta_{ij'}\right)}. \tag{5}$$

Note that the winner score $s^{(i)}$ is calculated according to the attention of all other assets. This way, CAAN accounts for the interrelationships among all assets.

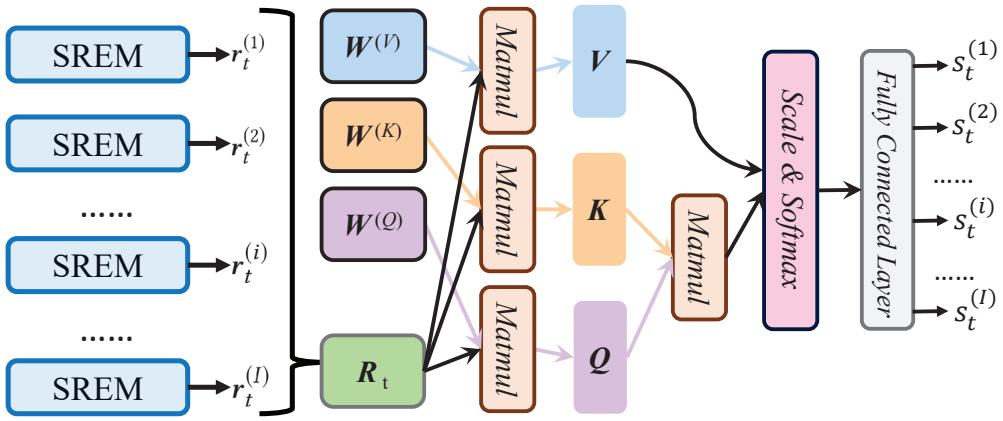


Figure 3: Architecture of Cross-Asset Attention Network (CAAN).

We use a fully connected layer to transform the attention vector $\boldsymbol{a}^{(i)}$ into a winner score as $s^{(i)}=\tanh\left(\boldsymbol{w}^{(s)\top}\cdot\boldsymbol{a}^{(i)}+e^{(s)}\right)$, where $\boldsymbol{w}^{(s)}$ and $e^{(s)}$ are the connection weights and the bias to learn. The winner score $s_t^{(i)}$ indicates the likelihood of asset i being selected into long positions in the t-th holding period. So far, the model embeds little economic meaning because an asset with a higher winner score may not necessarily contribute positively to portfolio performance. It is just a flexible structure (with high-dimensional parameters) for generating portfolios to be trained using RL later.

3.3 Portfolio Generation

Given the winner scores $\left\{s^{(1)},\cdots,s^{(i)},\cdots,s^{(I)}\right\}$ of a total of I assets, AP next constructs a long-short portfolio with long positions in assets with high winner scores and short positions in those with low winner scores. Specifically, we first sort the assets in descending order by their winner scores and obtain the sequence number $o^{(i)}$ for each asset i. Let G denote the preset size of the long and short parts of the portfolio $\boldsymbol{b}^{+}$ and $\boldsymbol{b}^{-}$. If $o^{(i)}\in[1,G]$, asset i then enters the portfolio $\boldsymbol{b}^{+(i)}$, with the investment proportion given by $b^{+(i)}=\frac{\exp\left(s^{(i)}\right)}{\sum_{o^{(i')}\in[1,G]}\exp\left(s^{(i')}\right)}$; if $o^{(i)}\in(I-G,I]$, $b^{-(i)}=\frac{\exp\left(-s^{(i)}\right)}{\sum_{o^{(i')}\in(I-G,I]}\exp\left(-s^{(i')}\right)}$ is the short proportion of asset i.[16]

[16]For example, if $G=50$, then the 50 highest ranked assets would be traded, with a weight given by $b^{+(i)}$.

The remainder assets do not have clear buy/sell signals and are thus not included in the portfolio. For simplicity, we use one vector to record all the information of the two portfolios. That is, we form the vector $\boldsymbol{b}^c$ of length I with $b^{c(i)} = b^{+(i)}$ if $o^{(i)} \in [1, G]$, $b^{c(i)} = b^{-(i)}$ if $o^{(i)} \in (I - G, I]$, or 0 otherwise, $i = 1, \ldots, I$. In what follows, we use $\boldsymbol{b}^c$ and $\{\boldsymbol{b}^+, \boldsymbol{b}^-\}$ interchangeably.

Note that before we fully train the model, because the parameters for TE and CAAN are all randomly initiated, the AlphaPortfolio could perform miserably at the beginning. Before proper training, a high winner score does not mean it is a better asset to invest in. After training, constructing the portfolio based on winner scores can generate portfolios that lead to high performance metrics. We next describe the training process.

3.4 Optimization via Reinforcement Learning

We embed AP into an RL game with continuum agent actions to train the model, where an episode — one complete round of the agent's interacting with the environment in RL (a T-period investment in our context)—is modeled as a state-action-reward trajectory π of an RL agent, *i.e.,* $\pi = \{state_1, action_1, reward_1, \ldots, state_t, action_t, reward_t, \ldots, state_T, action_T, reward_T\}$. The $state_t$ is the historical market state at t, which is expressed as a tensor $\mathcal{X}_t = \{\boldsymbol{X}_t^{(i)}, i = 1, \cdots, I\}$. The $action_t$ is portfolio vector $\boldsymbol{b}_t^c$ given by AP, of which the element $action_t^{(i)}$ indicates the portfolio weight the agent invests asset i at t, then $reward_t$ is the reward of $action_t$.

Let H_π denote the objective of the portfolio manager, with the trajectory of reward $\{reward_1, \ldots, reward_T\}$ as inputs. For example, $action_t$ could be the construction and trading of a portfolio, $reward_t$ is then the return of holding that portfolio, and H_π can be the *Sharpe ratio* computed using the returns in a 12-month window. The objective for portfolio construction can be very general, incorporating transaction costs (defining $reward_t$ as return minus transaction cost) or budget constraints (having budget as a variable of the state) or failures of portfolio managers (e.g., H_π being zero if a particular return in the trajectory is too negative). We explore various versions of H_π in our empirical analyses.

For all episodes, the average reward is $J(\theta) = \mathbb{E}[H_{\pi_\theta}]$. Recall that θ corresponds to the parameters of the proposed AP. The first part comes from the SREM (Sequence Representation Extraction Module). For TE-based SREM, parameters consist of multi-head transformation matrices, attention transformation matrices and weight matrices in feed forward networks. The second part comes from the CAAN module, for which we have a query transformation matrix, a key transformation matrix, and a value transformation matrix. The entries in these matrices are all parameters to be estimated. In addition, Theta also includes the weight matrix and bias that are adopted to transform the attention vector into the winner score.

The objective of the RL model optimization is to find the optimal parameters $\theta^* = \arg\max_\theta J(\theta)$. We use the gradient ascent approach to iteratively optimize θ at round τ as $\theta_\tau = \theta_{\tau-1} + \eta \nabla J(\theta)|_{\theta=\theta_{\tau-1}}$, where η is a learning rate. When we empirically train the model, an episode is defined as one year of investment which contains 12 trading periods, and $\nabla J(\theta)$ is automatically calculated using the deep learning framework we employ.

4 Empirical Performance: A Study of U.S. Equities

4.1 Data Description

We now apply the AlphaPortfolio model to public equities in the United States. Our baseline sample period is July 1965 to June 2016 with 1.76 million month-asset observations. Monthly stock return data are from the Center for Research in Security Prices (CRSP). We follow the literature standard to focus on common stocks of firms incorporated in the United States and trading on Amex, Nasdaq, or NYSE. Firms' balance-sheet data come from the Standard and Poor's Compustat database. To mitigate survivorship bias due to backfilling, we also require that a firm appears in the dataset for at least two years for training the model. For OOS test, we only require a firm to be in the dataset for one year.

Similar to Freyberger, Neuhierl, and Weber (2020), we construct firm characteristics and market signals as raw input features that fall into six categories: price-based signals such as monthly returns, investment-related characteristics such as the change in inventory over total assets, profitability-related characteristics such as return on operating assets, intangibles such as operating accruals, value-related characteristics such as the book-to-market ratio, and trading frictions such as the average daily bid-ask spread. We consider lagged features up to 12 months prior to the month of portfolio construction. Each input variable is treated as available only in the month after it becomes public, a date that lags behind the reporting date to start with. If a variable is not reported at the monthly frequency, we treat it as unchanged from the previous month. Overall, we have 51 times 12 input features for each asset at any time. Appendix B describes the construction of input features.[17] The AP framework allows the inclusion of macroeconomic variables and other alternative data, which likely improves its performance and we leave it for future studies.

[17] Tables 1 and 2 in Freyberger, Neuhierl, and Weber (2020) provide an overview of their input features and their summary statistics while their Section A.1 describes the construction of characteristics and related references. We incorporate time-variation of firm characteristics over the past month for variables beyond past-return-based predictors, and therefore have more input features.

4.2 Empirical Tests and Results

The baseline objective we specify for AP is the OOS Sharpe ratio, which is natural and widely used by academics and practitioners. To train the model, we use data from July 1965 till the end of 1989 and follow the construction of the portfolio outlined in Section 3, with G chosen such that the long and short positions each take 10% of all the stocks available. Although RL is known to allow interactions of actions (trading in our setting) and the environment (e.g., market state variables, price impact, etc.), in the baseline we shut the interaction and focus on the trial-and-error search benefit the RL brings about. Alternative objectives with interactions with the market environment can be accommodated easily, as we illustrate in Section 4.4. Note that RL does not distinguish between training and validation sets. We use historical data to proxy for the environment and the rewards to judge the quality of training and adjust hyper-parameters of AP. In that sense, model selection is embedded in the exploration steps during training. The OOS tests ensure against overfitting and model selection biases in evaluating the AP performance.

To start, we randomly initialize the parameters (over a wide range within the parameter space), randomly draw a month from the training set without replacement, and use inputs from the preceding 12 months (the drawn month included) and then evaluate performance on the subsequent 12 months (e.g., Sharpe ratio computed based on 12 monthly return observations) as the reward to update the parameters. We are not assuming the months are i.i.d. but are essentially drawing a 24-month window without using any window repeatedly. We repeat the step with the remaining months in the training set until we exhaust all the months in the training set. We call this multi-step process an epoch. In our implementation, we use 30 epochs, which is sufficient for the parameters to converge.[18]

After training, we test AP on the sample starting from 1990 with monthly rebalancing. All our results are obtained out-of-sample rather than relying on in-sample predictability adopted in traditional statistical tests. This is crucial to prevent overfitting with low signal-to-noise financial data. Note that the AP model is fine-tuned annually in our test samples (rolling updates), which renders our model a hybrid of offline and online RL once deployed as a live strategy. In other words, after seeing one year's performance, we use the additional data to update the model parameters. Here we use 6 epochs each containing 12 steps. Learning rate is similarly set to 1e-4, then 5e-5 after 2 epochs and 1e-5 after 4 epochs. Even though one could have fine-tuned the model at higher frequencies such as monthly, we use annual frequency to avoid overfitting to monthly variations and high computation costs for updating deep learning models at high frequency — a point also discussed in Gu, Kelly, and

[18] As is typical in the training of AI models, we gradually decrease the learning rate η as we go through more epochs. For example, we use a learning rate of 1e-4 in the first 5 epochs, then 5e-5 in the next 10 epochs, and 1e-5 after that. Such a tuning prevents the parameters from oscillating around optimum points or settling on local optima. By monitoring the flattening of a loss curve (loss as in the negative of reward), one can decide whether the parameters have converged.

Xiu (2020). Updating at a lower frequency tends to reduce the OOS performance because of stale information, which works against getting a superior performance.

Table 1 reports the main results. Columns (1)-(3) display the various moments of the returns of AP as well as metrics such as turnover. AP achieves an OOS Sharpe ratio of 2.0 in the full test data set and even higher when we restrict the training and testing to large and liquid stocks (in Columns (2) and (3) we require the stocks to be in the top 90 or 80 percentiles based on market cap). Apparently, the AP performance is not driven by microcaps and can be implemented without liquidity concerns.[19] If we restrict our attention to the top 90 percentile of stocks in terms of market cap, one thousand dollars invested by AP at the start of the 1990 would become 91,140 dollars by the end of 2016.

Table 1: Out-of-Sample Performance of AlphaPortfolio

In each month, AlphaPortfolio constructs a long-short portfolio of stocks in highest/lowest decile of winner scores. The detailed investment strategy is described in Section 3.3. Parameters are initially obtained from the training periods, then fine-tuned once a year in the OOS periods (rolling update). q_n symbolizes the n^{th} NYSE size percentile. Columns (1)-(3) display portfolio performance metrics, with Return, Std.Dev., and Sharpe ratio all annualized. Columns (4)-(9) further adjust portolio returns by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFC+PS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). Again, (4)-(5) present the alphas for the overall sample whereas (6)-(9) present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. "*," "**," and "***" denote significance at the 10%, 5%, and 1% level, respectively.

	AP Performance				AP Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor	All		$> q_{10}$		$> q_{20}$	
				Models	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
Return(%)	17.00	17.09	18.06	CAPM	13.9***	0.005	12.2***	0.088	14.0***	0.102
Std.Dev.(%)	8.48	7.39	8.19	FFC	14.2***	0.052	13.4***	0.381	14.7***	0.465
Sharpe	2.00	2.31	2.21	FFC+PS	13.7***	0.054	12.3***	0.392	13.3***	0.480
Skewness	1.42	1.73	1.91	FF5	15.3***	0.12	13.8***	0.426	14.7***	0.435
Kurtosis	6.35	5.70	5.97	FF6	15.6***	0.128	14.5***	0.459	15.8***	0.516
Turnover	0.26	0.24	0.26	SY	17.4***	0.037	15.8***	0.332	17.0***	0.394
MDD	0.08	0.02	0.02	Q4	16.0***	0.121	15.0***	0.495	16.2***	0.521

Both high returns and low volatility contribute to the high Sharpe ratio of our algorithm.

[19] As Hou, Xue, and Zhang (2020) point out, 65 percent of known anomalies cannot clear the single test hurdle of $|t| \geq 1.96$ because the original studies overweight microcaps via equal-weighted returns and often with NYSE-Amex-NASDAQ breakpoints in portfolio sorts and cross-sectional regressions, especially those with ordinary least squares, are highly sensitive to microcap outliers. The authors also point out how illiquidity and trading frictions render many anomalies in academic studies infeasible for trading.

Moreover, AP uses a much lower frequency of rebalancing (monthly), turnover, and maximum drawdown relative to other (high-frequency) machine learning strategies or traditional, anomaly-based trading. The performance would be reduced by half if after ranking the assets using winner scores, we value-weight the stocks instead of using the weights AP prescribe. The average holding period is four months, which is easy to implement.

As is standard in the literature, we also control for benchmark factor models in Columns (4)-(9), which include the CAPM, the Fama-French-Carhart 4-factor model (FFC, Carhart 1997), the Fama-French-Carhart 4-factor model plus the Pastor-Stambaugh liquidity factor model (FFC+PS, Pástor and Stambaugh 2003), the Fama-French 5-factor model (FF5, Fama and French 2015), the Fama-French 6-factor model (FF6, Fama and French 2018), the Stambaugh-Yuan 4-factor model (SY, Stambaugh and Yuan 2017), and the Hou-Xue-Zhang 4-factor model (Q4, Hou, Xue, and Zhang 2015). AP has a significant and large annualized α after controlling for various factors. This holds even for recently published latent factor models. For example, controlling for IPCA factors constructed using the 36 characteristics in Kelly, Pruitt, and Su (2019), AP generates 11.316%, 13.476%, and 13.776% α in the full, $> q_{10}$, and $> q_{20}$ samples respectively, all significant at the 1% level.

Note that AP does not pick small and illiquid stocks as many other models do based on back-testing — a somewhat surprising result. We attribute this to the fact that even though small and illiquid stocks tend to commend high expected returns, they also significantly contribute to the volatility of a portfolio. The direct optimization of the Sharpe ratio rather than characteristic sorting effectively avoids small and illiquid stocks in the construction.

Table 2 further demonstrates the efficacy of RL and AI for investment. Panel A compares AP with the "nonparametric" (NP) model and portfolio strategy in Freyberger, Neuhierl, and Weber (2020). AP outperforms most other machine-learning-based strategies in the literature. We pick NP as a benchmark because in addition to the fact that we use similar firm characteristics as in their paper for inputs, NP is likely among the 3-5 best-performing machine learning models in asset pricing. NP achieves a higher Sharpe ratio on its test sample in 1991-2014. Once we exclude illiquid and small stocks, AP outperforms NP significantly, consistent with Avramov, Cheng, and Metzker (2019)'s findings that recent machine learning strategies often derive their performances from microcap and illiquid stocks. The superior performance here does not invalidate other models such as NP, as their focus is on minimizing pricing errors or estimating pricing kernels rather than directly optimizing portfolio performance.

It is worth mentioning that the winner score is not just another estimator of expected returns. RL takes into consideration both the expected returns and other moments of available assets. To see how RL adds to AP's performance, Panel B in Table 2 presents the OOS performance when we follow the traditional two-step approach to first use TE under super-

Table 2: Comparison with Alternative Models Using Out-of-Sample Performance

Panel A compares AP's performance with one benchmark model proposed in Freyberger, Neuhierl, and Weber (2020) in 1991-2014 (their test sample period). *NP* and *AP* denote the nonparametric model in Freyberger, Neuhierl, and Weber (2020) and the AlphaPortfolio, respectively. Panel B presents the results using the full test sample period (1990-2016) when we follow the traditional two-step approach to first use transformer encoder to predict stock returns and then form expected-return-sorted portfolios. Panel C compares the results using the full test-sample period (1990-2016) without (TE) and with CAAN (TE-CAAN) in the AlphaPortfolio. Again, q_n symbolizes the n^{th} NYSE size percentile. Return, Std.Dev., and Sharpe ratio all annualized.

Firms	All		$> q_{10}$		$> q_{20}$	
	(1)	(2)	(3)	(4)	(5)	(6)
Panel A: Comparison with NP Model (1991-2014)						
Model	NP	AP	NP	AP	NP	AP
Return(%)	45.84	15.60	21.12	17.70	15.48	17.90
Std.Dev.(%)	16.66	8.20	13.27	7.60	14.90	8.60
Sharpe	2.75	1.90	1.60	2.33	1.04	2.08
Skewness	3.53	1.20	0.30	1.77	-0.50	1.88
Kurtosis	19.56	6.54	7.80	5.57	13.06	5.46
Turnover	0.69	0.26	0.74	0.24	0.74	0.26
MDD	0.10	0.08	0.27	0.02	0.36	0.08
Panel B: TE-Based Return-Sorted Portfolio						
Weight	Equal	Value	Equal	Value	Equal	Value
Return(%)	8.80	3.20	19.30	5.90	18.10	6.30
Std.Dev.(%)	9.40	8.80	11.70	8.60	9.80	7.90
Sharpe	0.94	0.36	1.65	0.69	1.85	0.80
Skewness	2.46	-1.78	6.02	1.48	4.07	1.00
Kurtosis	19.84	27.55	67.00	10.89	33.21	15.63
Turnover	0.17	0.29	0.18	0.31	0.18	0.28
MDD	0.08	0.15	0.30	0.10	0.02	0.07
Panel C: Ablation Study for CAAN						
Model	TE	TE-CAAN	TE	TE-CAAN	TE	TE-CAAN
Return(%)	13.70	17.00	12.92	17.09	13.65	18.06
Std.Dev.(%)	8.84	8.48	6.66	7.39	6.73	8.19
Sharpe	1.55	2.00	1.94	2.31	2.03	2.21
Skewness	1.84	1.42	3.06	1.73	1.80	1.91
Kurtosis	15.42	6.35	19.69	5.70	6.06	5.97
Turnover	0.38	0.26	0.45	0.24	0.43	0.26
MDD	0.07	0.08	0.03	0.02	0.02	0.02

vised learning to predict stock returns and then form expected-return-sorted portfolios.[20]

[20]Because the links among various assets are typically not explicitly modeled in simple characteristic-based sorting, Table 2 does not involve CAAN when modeling asset returns.

We note that OOS Sharpe ratio can reach 0.8 with market-cap-adjusted weights and close to 2 with equal weights. On the one hand, this demonstrates the TE under the traditional two-step portfolio construction still outperforms many other strategies (machine-learning-based or anomaly/sorting-based, because TE as a flexible deep neural network better captures the nonlinearity and path-dependence information). On the other hand, the performance is dwarfed by the RL-based AP model, highlighting the utility of our one-step RL. For example, the value-weighted portfolio on the full sample, which is more feasible than equal weighting in practice, has an OOS Sharpe ratio of 0.36 as compared to AP's Sharpe ratio of 2. In other words, had we used winner scores as an estimator of expected returns and either equal weights or value weights, the portfolio would significantly underperform AP.

Finally, Panel C in Table 2 demonstrates how our innovation of CAAN further contributes to AP performance. Using TE alone, RL still achieves higher OOS return and Sharpe ratio and lower turnover and maximum drawdown than other ML models (which often perform less well than NP in Panel A to start with) and the two-step approach (as seen in Panel B). Nevertheless, CAAN improves the OOS performance significantly on top of RL, on average increasing Sharpe ratio by an additional 0.33 and annualized returns by almost 4% in the three test samples while lowering the turnover by 40%. Though unreported here, CAAN's impact on performance is even greater when we implement AP using LSTM (Appendix C2).

4.3 Economic Restrictions and Model Robustness

Performance of traditional anomalies and machine learning strategies is often suspected to be primarily driven by microcap stocks, illiquid stocks, extreme market conditions, equal-weighting of the portfolio, etc. We now dispel such concerns and demonstrate AP's robustness. Note that the results reported below after imposing various economic restrictions are only lower bounds on AP's performance, as we do not retrain AP on the subsamples excluding certain stocks or under specific restrictions. Instead, we simply set the portfolio weights of the excluded stocks or non-admissible stocks under the specific restrictions to zero.

Microcaps. Given that microcaps have the highest equal-weighted returns and the largest cross-sectional dispersions in returns and in anomaly variables, anomalies in cross-sectional asset returns could be driven by microcap stocks and costly-to-trade stocks (Novy-Marx and Velikov, 2016; Hou, Xue, and Zhang, 2020). Avramov, Cheng, and Metzker (2019) find that machine techniques face similar issues: Return predictability and the anomalous patterns are concentrated in difficult-to-arbitrage stocks and in times of high limits to arbitrage.

Table 1 reveals that AP's performance is not driven by the bottom 10% and 20% of stocks based on market capitalization. If anything, the performance improves after excluding them, which outperforms almost all other machine learning models in terms of Sharpe ratio,

maximum drawdown, and turnover. This observation implies that AP is more effective in uncovering patterns in large and liquid stocks than in a mixture of large and small cap stocks.

Turnovers, shorting, and transaction costs. Following Koijen, Moskowitz, Pedersen, and Vrugt (2018); Freyberger, Neuhierl, and Weber (2020), we calculate turnover using $Turnover_t = \frac{1}{4}\sum_i \left|w^i_{t-1}(1+r^i_t) - w^i_t\right|$, where the coefficient $\frac{1}{4}$ avoids double counting (a factor of 2) and adjusts for that the long/short strategies have \$2 exposure (another factor of 2). Gu, Kelly, and Xiu (2020) point out that many strategies have turnovers well above 100% monthly, using the alternative turnover measure, $Turnover_{GKX} = \left|w_{i,t+1} - \frac{w_{i,t}(1+r_{i,t+1})}{1+\sum_j w_{j,t} r_{j,t+1}}\right|$.[21] With this definition, AP still has very low turnovers relative to other traditional anomalies or machine learning strategies, whether we calculate the long/short-leg turnover respectively and sum it up, or we directly calculate the turnover of long-short portfolio using this alternative measure.

With low turnovers, AP's performance stands out even after incorporating transaction costs (but without retraining the model). For example, setting a cost at 0.1%, AP still yields OOS Sharpe ratios of 2.01, 2.27, and 2.16 for the full sample and the $size > q_{10}$ and $size > q_{20}$ subsamples, respectively. The turnover and maximum drawdown do not differ much from the baseline model.

Table 3: Out-of-Sample Performance of Long-Only AP

This table presents the OOS performance of a long-only AlphaPortfolio taking positions only in the highest decile of winner scores, with Return, Std.Dev., and Sharpe ratio all annualized. Portfolio returns are further adjusted by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFC+PS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). The first columns present the alphas for the overall sample. The remaining four columns present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5%, and 1% level, respectively.

	AP Performance				AP Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor Models	All		$> q_{10}$		$> q_{20}$	
					α(%)	R^2	α(%)	R^2	α(%)	R^2
Return(%)	22.41	37.60	44.27	CAPM	12.88***	0.477	25.80***	0.508	31.79***	0.485
Std.Dev.(%)	19.28	25.08	27.63	FFC	12.54***	0.791	25.79***	0.798	30.97***	0.733
Sharpe	1.16	1.50	1.60	FFC+PS	11.02***	0.795	22.20***	0.806	28.20***	0.738
Skewness	0.67	1.03	1.55	FF5	9.91***	0.731	21.62***	0.755	27.84***	0.696
Kurtosis	5.64	4.72	7.55	FF6	12.10***	0.791	24.12***	0.802	30.31***	0.734
Turnover	0.38	0.44	0.48	SY	15.38***	0.720	26.29***	0.739	31.68***	0.681
MDD	0.24	0.27	0.25	Q4	14.51***	0.698	26.83***	0.714	33.01***	0.656

[21] Note that to have the proper normalization for long-short portfolios in our setting, we have added 1 in the denominator.

While many existing strategies (whether anomaly-based or machine-learning-based) for long-short portfolio construction heavily depend on the short positions, AP's long and short positions both contribute significantly to the performance. If anything, the long positions play a more dominant role. For the long-short AlphaPortfolio, they generate returns of 37.7%, 40.3%, and 41.1% for the full sample, size$> q10$ sample, and size$> q20$ sample respectively, as compared to the 4.8%, 6.1%, and 5% from short positions.

For robustness, we also train a long-only AlphaPortfolio, and the results are reported in Table 3. The OOS Sharpe ratio and excess alphas are lower than the long-short AlphaPortfolio but higher than most other long-only strategies and anomalies.

Performance attenuation over time. Given our high-dimensional inputs involving known anomalies, there is a natural concern of data mining. Moreover, many anomalies have attenuated since the early 2000s (e.g., Chordia, Subrahmanyam, and Tong, 2014; McLean and Pontiff, 2016; Linnainmaa and Roberts, 2018; Han, He, Rapach, and Zhou, 2018) because the U.S. equity market witnessed several structural changes since the 2000s, such as the introduction of decimalization. Could AP be trading on anomalies that were known only ex post or have been traded away in recent years or on seeming anomalies from data snooping? To answer this, we restrict the test to a subsample of more recent years (2001-2016). As we report in Table 11 Columns (4)-(6), AP's performance remains robust. Controlling for IPCA factors, AP also generates 12.372%, 14.808%, and 14.076% α in the full, $> q_{10}$, and $> q_{20}$ samples from 2001 to 2016, all significant at the 1% level.

We also plot AP's Sharpe ratio and excess α relative to seven benchmark factor models over time. We look at non-overlapping 3-year windows and the trends are depicted in Figure 4. Overall, the Sharpe ratio is particularly high at the start of the test sample but does not exhibit particular trends post 2000s. Excess alphas fluctuate but show no sign of attenuation.

Industry attribution and weights. We perform industry/style attribution tests and report the results in Table 5. After regressing AP's monthly OOS returns (1990-2016) on the 12 industries according to Fama and French, the intercept is both economically and statistically significant.[22] We also do not see significant loadings on most industry portfolios except for durables (positive), energy (negative), and retails (negative). Overall, AP picks up patterns beyond those associated with industries or styles and does not heavily weigh particular industries.

It is also worth noting that our findings are not driven by the issue of equal weighing versus value weighing. The Pearson coefficient of assets' investment proportion, and its market capitalization shows that under 15% of the time, the two factors are significantly negatively correlated. In other words, our portfolio is close to value-weighted, which is more

[22]The definition and data are at https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.

Table 4: Out-of-Sample Performance in Recent Years

This table reports alphas for portfolios of long/short stocks in the highest/lowest decile of winner scores from 2001 to 2016. Return, Std.Dev., and Sharpe ratio are all annualized. Portfolio returns are further adjusted by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFCPS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). The first columns present the alphas for the overall sample. The remaining four columns present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5% and 1% level, respectively.

	AP Performance				AP Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor Models	All		$> q_{10}$		$> q_{20}$	
					$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
Return(%)	18.10	16.10	16.60	CAPM	16.5***	0.007	13.6***	0.136	13.8***	0.176
Std.Dev.(%)	9.20	7.90	8.90	FFC	16.3***	0.078	12.9***	0.497	13.3***	0.594
Sharpe	1.97	2.04	1.87	FFC+PS	15.7***	0.080	11.7***	0.506	11.7***	0.606
Skewness	1.67	1.53	1.61	FF5	18.0***	0.151	13.8***	0.426	14.6***	0.432
Kurtosis	5.95	4.23	3.59	FF6	17.8***	0.174	13.3***	0.560	14.0***	0.620
Turnover	0.25	0.23	0.25	SY	18.9***	0.065	15.3***	0.428	16.5***	0.502
MDD	0.05	0.03	0.04	Q4	16.9***	0.121	13.7***	0.532	14.6***	0.551

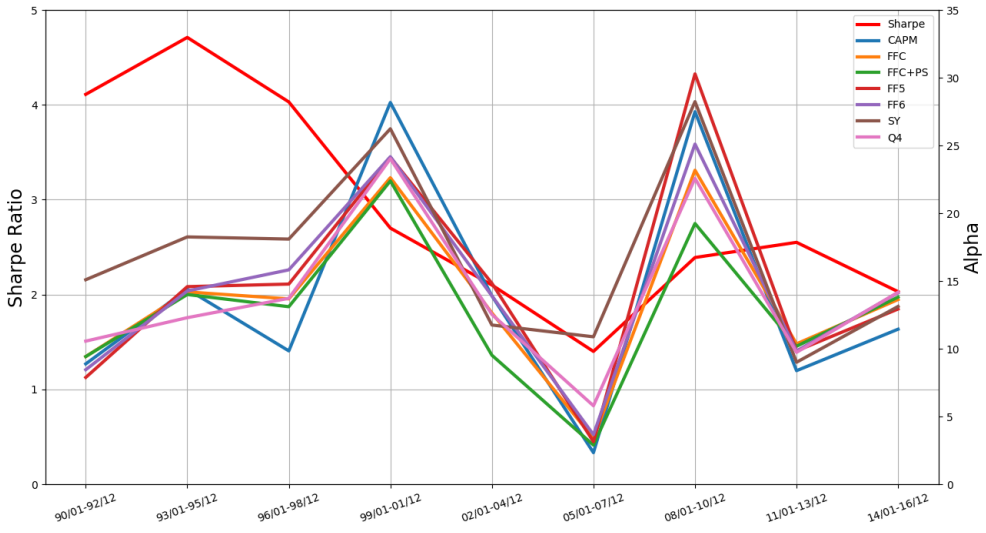


Figure 4: Trends of AlphaPortfolio Performance.

Table 5: Industry Attribution Test for AlphaPortfolio

This table presents the results from regressing the AlphaPortfolio return on various industry portfolio returns. The 12 industries are Fama-French industries based on four-digit SIC codes: 1 NoDur (Consumer Nondurables) – Food, Tobacco, Textiles, Apparel, Leather, Toys; 2 Durbl (Consumer Durables) – Cars, TVs, Furniture, Household Appliances; 3 Manuf (Manufacturing) – Machinery, Trucks, Planes, Off Furn, Paper, Com Printing; 4 Enrgy – Oil, Gas, and Coal Extraction and Products; 5 Chems – Chemicals and Allied Products; 6 BusEq (Business Equipment) – Computers, Software, and Electronic Equipment; 7 Telcm – Telephone and Television Transmission; 8 Utils – Utilities; 9 Shops – Wholesale, Retail, and Some Services, Laundries, Repair Shops; 10 Hlth – Healthcare, Medical Equipment, and Drugs; 11 Money – Finance; 12 Other – Mines, Constr, BldMt, Trans, Hotels, Bus Serv, Entertainment. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5%, and 1% level, respectively.

		Industry												
	Intercept	NoDur	Durbl	Manuf	Enrgy	Chems	BusEq	Telcm	Utils	Shops	Hlth	Money	Other	R^2
All														
Coefficient	1.438***	0.096	0.035	0.169**	-0.133***	-0.059	0.055	0.019	-0.026	-0.125**	-0.074*	-0.033	-0.004	0.119
Std.Err.	0.141	0.071	0.039	0.084	0.037	0.067	0.034	0.043	0.045	0.061	0.043	0.048	0.078	
$> q_{10}$														
Coefficient	1.494***	0.058	0.073**	0.027	-0.060*	-0.107*	0.047	0.020	-0.082**	-0.119**	-0.012	0.043	0.048	0.135
Std.Err.	0.125	0.064	0.035	0.075	0.033	0.060	0.030	0.039	0.040	0.054	0.039	0.043	0.070	
$> q_{20}$														
Coefficient	1.584***	0.078	0.076**	0.041	-0.082**	-0.134**	0.080**	0.006	-0.078*	-0.167***	-0.034	0.049	0.104	0.190
Std.Err.	0.133	0.068	0.037	0.080	0.035	0.063	0.032	0.041	0.042	0.057	0.041	0.046	0.074	

feasible to construct in practice than many extant anomalies relying on equal weights.

In fact, we use the softmax function to generate portfolio weights from winner scores for two reasons. First, the optimization of deep neural networks relies on gradient-based backpropagation algorithm (Rumelhart, Hinton, and Williams, 1986). Our softmax function is differentiable. In contrast, value weights bear no gradient relationship to winner scores, implying that constructing value-weighted portfolios blocks the backpropagation procedure for model training. Second, our softmax-generated portfolio outperforms in OOS Sharpe ratio, risk-adjusted returns, portfolio turnover, maximum drawdown, etc. One can alternatively take long positions in the top 10% and short positions in the bottom 10% based on the winner score using market capitalization instead of the softmax transformation to determine portfolio weights. But the resulting portfolio has an OOS Sharpe ratio below 1.5 and risk-adjusted return below 5%, not to mention that the turnover and maximum drawdown are all higher than those of AP.

Unrated or downgraded firms. To rule out the possibility that our results are driven by unrated or downgraded firms that are hard to trade on, we next follow Avramov, Cheng, and Metzker (2019) to test our model on a subsample including only rated firms, i.e., firms with data on S&P's long-term issuer credit rating. Such rated firms tend to be large and liquid, with better disclosure, more analyst coverage, and smaller idiosyncratic volatility, which makes trading them cheaper and more feasible. Within the rated firms, we further exclude

firms with credit rating downgrades which are typically associated with greater trading and arbitrage frictions (e.g., Avramov, Chordia, Jostova, and Philipov, 2013). As in Avramov, Cheng, and Metzker (2019), we exclude stock-month observations from 12 months before to 12 months after the downgrade events. The results are reported in Table C.1. As seen in Panel A, even though the α values are smaller, they are still significant and greater than most known anomalies. The Sharpe ratios shown in Panel B exhibit similar patterns.

The significant reductions in excess alpha and Sharpe ratio are natural and mostly an artifact that AP is not reestimated after imposing the economic restrictions or re-trained on samples excluding unrated and downgraded firms. We simply set the portfolio weights for excluded stocks to zero when performing OOS tests. So the excess returns reported here are all lower bounds on AP's performance under the various economic restrictions.

Market conditions. Prior studies document that traditional anomalies are more salient during high investor sentiment, high market volatility, and low market liquidity (e.g., Stambaugh, Yu, and Yuan, 2012; Nagel, 2012). Many machine learning strategies are also shown to have insignificant α in times of low VIX or low sentiment (Avramov, Cheng, and Metzker, 2019). If the OOS tests of AP only perform well during high investor sentiment and high market volatility, AP may not be implementable in practice due to limits to arbitrage.

To investigate this issue, we examine the AP performance in subperiods of different states of investor sentiment, market volatility (implied and realized), and liquidity. Investor sentiment ($SENT$) is defined as the monthly Baker and Wurgler (2007) investor sentiment; market volatility (VIX) is defined as the monthly VIX index of implied volatitity of S&P 500 index options; realized market volatility ($MKTVOL$) is defined as the standard deviation of daily CRSP value-weighted index return in a month; and market illiquidity ($MKTILLIQ$) is defined as the value-weighted average of stock-level Amihud illiquidity for all NYSE/AMEX stocks in a month.[23] We divide the full sample into two subperiods using the median breakpoints of $SENT$, VIX, $MKTVOL$, and $MKTILLIQ$ over the whole sample period. We report the analyses in Table 7. For brevity, we only present baseline samples and FF6-adjusted α of the portfolios.

Overall, unlike most other machine learning strategies that yield no significant profits during low volatility/sentiment or high liquidity or when unrated or downgraded firms are excluded (Avramov, Cheng, and Metzker, 2019), AP continues exhibiting a high Sharpe ratio and significant α even after controlling for various factors, regardless of the market conditions. AP is particularly more profitable during high investor sentiment or market volatility or liquidity, and the alpha is both economically and statistically significant under various market conditions. If anything, the Sharpe ratio seems better during low sentiment

[23]Investor sentiment index is available at Jeffrey Wurgler's website http://people.stern.nyu.edu/jwurgler/ while monthly VIX index is available at CBOE website http://www.cboe.com/products/vix-index-volatility/vix-options-and-futures/vix-index/vix-historical-data.

Table 6: Out-of-Sample Performance of Portfolios Excluding Unrated (and Downgraded) Stocks

This table reports alphas for portfolios of long/short stocks in the highest/lowest decile of winner scores from 1990 to 2016, where (1)-(3) exclude *unrated* stocks from the portfolio and (4)-(6) exclude both *unrated* and recently *downgraded* stocks. In Panel A, portfolio returns are adjusted by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFC+PS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). Within (1)-(3) and within (4)-(6), the first columns present the alphas for the whole sample. The remaining four columns present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. Panel B presents other performance metrics in a similar fashion, with Return, Std.Dev., and Sharpe ratio all annualized. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5%, and 1% level, respectively.

Panel A: Out-of-Sample Alpha Under Various Factor Models

	Excluding Unrated Firms						Excluding Unrated & Downgraded					
	(1)		(2)		(3)		(4)		(5)		(6)	
Firms	All		Size $> q_{10}$		Size $> q_{20}$		All		Size $> q_{10}$		Size $> q_{20}$	
	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
CAPM	3.4***	0.000	4.6**	0.051	5.3***	0.067	4.7***	0.000	3.8***	0.070	4.3***	0.076
FFC	3.1***	0.061	5.2***	0.380	5.5***	0.504	4.4***	0.047	4.2***	0.305	4.4***	0.437
FFC+PS	2.4*	0.070	4.6***	0.384	4.7***	0.511	3.4***	0.060	3.6***	0.309	3.4***	0.448
FF5	3.6**	0.146	4.3**	0.251	4.0***	0.375	5.0***	0.100	3.7***	0.219	2.9***	0.338
FF6	3.7***	0.147	5.6***	0.391	5.3***	0.517	5.0***	0.100	4.7***	0.308	4.0***	0.446
SY	5.3***	0.182	7.9***	0.393	7.4***	0.494	6.6***	0.149	6.9***	0.333	6.2***	0.444
Q4	3.5***	0.109	5.8***	0.294	5.6***	0.394	5.0***	0.082	5.1***	0.257	4.5***	0.350

Panel B: Other Portfolio Performance Metrics

	Excluding Unrated Firms			Excluding Unrated & Downgraded		
	(1)	(2)	(3)	(4)	(5)	(6)
Firms	All	Size $> q_{10}$	Size $> q_{20}$	All	Size $> q_{10}$	Size $> q_{20}$
Return(%)	6.18	8.30	8.99	7.45	7.54	8.06
Std.Dev.(%)	5.93	7.89	7.03	6.42	7.02	7.00
Sharpe	1.04	1.05	1.28	1.16	1.07	1.15
Skewness	-0.24	2.23	1.69	-0.89	1.18	1.18
Kurtosis	4.52	11.83	10.16	7.96	7.92	8.68
MDD	0.06	0.08	0.08	0.05	0.08	0.08

Table 7: Out-of-Sample Performance of AP Under Various Market Conditions

This table reports annualized alphas (adjusted by Fama-French 6-factor model) and Sharpe ratios of the AlphaPortfolio in high and low sentiment periods (Panel A), low and high VIX periods (Panel B), low and high realized volatility periods (Panel C), and low and high illiquidity periods (Panel D). Sentiment ($SENT$) is measured from the raw version of monthly Baker and Wurgler (2007) sentiment index that excludes the NYSE turnover variable; market volatility (VIX) is defined as the monthly VIX index of implied volatility of S&P 500 index options; realized market volatility ($MKTVOL$) is defined as the standard deviation of daily CRSP value-weighted index return in a month; market illiquidity ($MKTILLIQ$) is defined as the value-weighted average of stock-level Amihud illiquidity for all NYSE/AMEX stocks in a month. The panels report the results in the full sample as well as subsamples that exclude the bottom 10% and 20% microcaps. q_n symbolizes the n^{th} NYSE size percentile. Newey-West adjusted t-stats are shown in brackets with "*," "**," and "***" denoting 10%, 5%, and 1% significance levels, respectively.

	(1)	(2)	(3)	(4)	(5)	(6)
Sample	All		Size $> q_{10}$		Size $> q_{20}$	
Variable	Low	High	Low	High	Low	High
Panel A: AP Performance Under Various SENT Periods						
FF-6 α	19.273***	13.351***	14.132***	13.746***	14.756***	16.072***
t-stat	(5.975)	(7.914)	(8.284)	(9.441)	(8.980)	(9.751)
Sharpe	1.734	1.69	2.106	1.796	2.123	1.677
Panel B: AP Performance Under Various VIX Periods						
FF-6 α	10.248***	19.776***	10.812***	18.660***	10.272***	21.084***
t-stat	(5.060)	(7.421)	(8.862)	(10.201)	(8.598)	(11.107)
Sharpe	1.719	1.713	2.371	1.951	2.331	1.932
Panel C: AP Performance Under Various MKTVOL Periods						
FF-6 α	10.385***	17.167***	10.687***	17.750***	11.038***	19.626***
t-stat	(30.636)	(7.577)	(6.686)	(11.088)	(6.765)	(11.896)
Sharpe	1.654	1.668	2.429	1.855	2.461	1.806
Panel D: AP Performance Under Various MKTILLIQ Periods						
FF-6 α	11.9635***	19.385***	14.107***	13.048***	16.096***	12.541***
t-stat	(5.616)	(6.971)	(9.084)	(9.029)	(10.209)	(8.038)
Sharpe	1.447	1.874	1.971	1.885	1.880	1.877

and low market volatility. Our deep RL signals deliver meaningful risk-adjusted performance over the entire test sample period as well as in various market states.

4.4 Market Interactions and Flexible Management Objectives

Thus far, AP involves a deterministic gradient and implicit in our baseline RL setup is that our portfolio manager is a price-taker. This common assumption in many asset pricing studies implies a limitation on the scale of AP. Depending on the size of the portfolio,

transaction costs and price impacts could differ significantly. More generally, an RL learner's action could alter the market state. While it is too complicated to model the interactions between actions and market environments completely, RL is known for excelling at incorporating such interactions, and the AP framework is flexible enough to accommodate various scenarios in portfolio management. We provide several illustrative examples, in which we retrain the model allowing the investment actions to interact with the market environment specified by us or in the literature.

In the first example, we consider the simple situation that AP's portfolio size is kept constant over time. To account for transaction fees and price impacts, we introduce a transaction cost of 10-100 basis points of the transaction amount. Note that in training the model, we allow the action (trading) to affect the state (return of assets after fees). The results are reported in Table 8. The cost consideration brings down the strategies turnover, but OOS Sharpe ratio and other performance metrics remain comparable with the baseline.

Table 8: Out-of-Sample Performance of AlphaPortfolio Considering Trading Costs

This table reports performance for portfolios of long/short stocks in the highest/lowest decile of winner scores from 1990 to 2016, where (1)-(3) set transaction cost rate as 0.001 and (4)-(6) set transaction cost rate as 0.01. Within (1)-(3) and within (4)-(6), the first columns present the performance for the whole sample. The remaining four columns present performance for subsamples excluding microcap firms in the smallest decile and quintile, respectively. q_n symbolizes the n^{th} NYSE size percentile. Return, Std.Dev, and Sharpe are all annualized.

	Transaction Cost Rate 0.001			Transaction Cost Rate 0.01		
	(1)	(2)	(3)	(4)	(5)	(6)
Firms	All	Size $> q_{10}$	Size $> q_{20}$	All	Size $> q_{10}$	Size $> q_{20}$
Return(%)	15.73	14.04	14.92	16.59	16.33	15.44
Std.Dev.(%)	7.77	5.90	7.66	9.85	6.96	7.82
Sharpe	2.02	2.38	1.95	1.68	2.35	1.97
Skewness	2.04	1.24	2.53	2.08	1.49	2.36
Kurtosis	11.26	4.54	11.83	10.73	4.33	10.28
Turnover	0.20	0.23	0.25	0.16	0.19	0.20
MDD	0.03	0.02	0.02	0.06	0.02	0.02

In the second example, in addition to considering price impacts and transaction costs, we incorporate dynamic budget consideration. All returns are reinvested which implies that the AP strategy dynamically changes with the portfolio size, which affects the trading costs and returns. This portfolio size limit is part of the market environment (it affects investment opportunities in practice as well), and interacts with the trading actions. In order to make the trading costs under varying scale of the portfolio realistic, we can adopt the model and parameter estimates from comprehensive studies such as Frazzini, Israel, and Moskowitz (2018). For illustration, we take their trading cost model with input variables, "Beta*IndexRet*buysell," "Time trend," "Log of ME," "Fraction of daily volume," and

Table 9: Out-of-Sample Performance of AP Considering Portfolio Scale and Price Impact

This table reports performance for portfolios of long stocks in the highest slice of winner-scores from 1990 to 2016 with a budget constraint. Portfolio returns are further adjusted by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFCPS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). The first columns present the alphas for the overall sample. The remaining four columns present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5% and 1% level, respectively.

	AP Performance				AP Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor Models	All		$> q_{10}$		$> q_{20}$	
					$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
Return(%)	36.70	43.57	38.36	CAPM	26.66***	0.347	32.17***	0.362	26.89***	0.401
Std.Dev.(%)	24.36	28.38	27.20	FFC	27.58***	0.626	30.97***	0.630	25.74***	0.655
Sharpe	1.51	1.54	1.41	FFC+PS	26.82***	0.626	28.64***	0.634	22.75***	0.661
Skewness	0.81	2.68	2.59	FF5	26.72***	0.570	27.81***	0.587	21.73***	0.217
Kurtosis	3.65	19.81	17.57	FF6	29.56***	0.633	30.48***	0.628	24.30***	0.658
Turnover	0.17	0.21	0.24	SY	33.72***	0.565	33.80***	0.557	27.09***	0.587
MDD	0.22	0.28	0.27	Q4	32.16***	0.600	32.77***	0.541	26.76***	0.556

"sqrt(Fraction of daily volume)." We examine the long-only AP with an initial budget of $10 million USD and retrain the model. The results are reported in Table 9. Compared with Table 3, it still achieves an OOS Sharpe ratio over 1.5 and other performance metrics comparable with the baseline AP. It is worth mentioning that the turnover is almost half of the baseline AP once we take transaction cost and budget dynamics into consideration.

In the third example shown in Table 10, we adopt a portfolio performance metric balancing a fund's return and survival when training the model. In other words, we set the management objective to be the expected cumulative return over a year of a long-short portfolio subject to a transaction cost rate is set as 10 basis points and a fund failure if the portfolio ever incurs a 50% loss during the 12-month window. Given that whether a fund is alive is part of the market state, trading actions obviously affect the long-run environment because a fund is closed down if the action results in a big loss. This portfolio management objective captures that many funds are prone to redemption and discontinued funding once the loss exceeds a certain threshold. In practice, the lock-in period could differ depending on whether the fund is a hedge fund or mutual fund and whether it is open- or close-ended, which can be adjusted in the AP framework. Quite intuitively, without lowering portfolio volatility being explicit in the objective, AP generates higher risk-adjusted returns at the expense of lower Sharpe ratios. Nevertheless, both the OOS excess alpha and Sharpe ratio are still higher than most other known strategies rebalancing at a monthly frequency or lower. The RL approach is also effective in the sense that in the OOS test, the fund never

Table 10: Out-of-Sample Performance for an AP Fund

This table reports performance for a long-short portfolio taking long positions in assets with the highest 10% winner scores from 1990 to 2016, and short positions in assets with the lowest 10% winner scores. The objective is set as the cumulative return over a 12-month window and transaction cost rate is set as 0.1%. During training, AP will stop trading upon incurring a 50% loss in the 12-month window. Portfolio returns are further adjusted by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFCPS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). The first columns present the alphas for the overall sample. The remaining four columns present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. Returns and Sharpe ratios are annualized. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5% and 1% level, respectively.

	AP Performance				AP Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor Models	All		$> q_{10}$		$> q_{20}$	
					$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
Return(%)	26.61	24.82	24.01	CAPM	22.12***	0.047	18.95***	0.150	18.20***	0.168
Std.Dev.(%)	15.35	15.68	14.59	FFC	23.80***	0.243	19.76***	0.433	18.92***	0.487
Sharpe	1.73	1.58	1.65	FFC+PS	24.35***	0.243	18.65***	0.436	17.27***	0.493
Skewness	2.26	3.71	3.01	FF5	26.56***	0.358	22.26***	0.489	21.53***	0.548
Kurtosis	11.98	28.57	17.14	FF6	27.64***	0.381	23.01***	0.500	22.08***	0.555
Turnover	0.19	0.22	0.23	SY	30.14***	0.227	24.26***	0.384	22.95***	0.429
MDD	0.09	0.06	0.06	Q4	28.54***	0.366	23.88***	0.480	22.99***	0.530

incurs a loss of 50% or higher. In other words, even though in the training the fund fails from time to time, a trained AP fund model survives throughout the entire test period.

In the fourth example, we adopt a portfolio performance metric incorporating the delegated nature of investment management. We set the management objective to be the expected compensation over multiple years for managing a long-only portfolio. The compensation is comprised of a management fee of 0.5% of the asset under management (AUM) and a carried interest of 10% for excess return. The management fee and carry vary quite a bit in practice but the compensation structure is realistic (e.g., Ma, Tang, and Gomez, 2019). For a simple illustration, we set the career length of the manager to be four years and the return benchmark to be zero. For a fund with an initial AUM of $10 million USD with all after-fee proceeds reinvested. A fund manager earns on average a cumulative compensation of $2,094,850 USD over four years, with a standard deviation of $556,630 USD.[24] Table 11 reports other metrics of the AP performance. The lack of concern for volatility means AP focuses on high cumulative returns at the expense of Sharpe ratio. AP also seems to pick small and illiquid stocks, the exclusion of which reduces the Sharpe ratio by around a quarter in magnitude.

Our goal is not to provide an exhaustive list of AP objectives or market interactions.

[24]We have 27 years in the test sample and we extrapolate the last three years into a four-year cycle.

Table 11: Out-of-Sample Performance for Compensation-based Objective

This table reports performance for portfolios of long/short stocks in the highest/lowest decile of winner-scores from 1990 to 2016. The objective is set as manager's compensation. During each step at training, AP considers a 4-year trading period. Each year the compensation is calculated as $AUM * 0.5\% + Positive_return * 0.1$. Portfolio returns are further adjusted by the CAPM, Fama-French-Carhart 4-factor model (FFC), Fama-French-Carhart 4-factor and Pastor-Stambaugh liquidity factor model (FFCPS), Fama-French 5-factor model (FF5), Fama-French 6-factor model (FF6), Stambaugh-Yuan 4-factor model (SY), and Hou-Xue-Zhang 4-factor model (Q4). The first columns present the alphas for the overall sample. The remaining four columns present alphas for subsamples excluding microcap firms in the smallest decile and quintile, respectively. q_n symbolizes the n^{th} NYSE size percentile. "*," "**," and "***" denote significance at the 10%, 5% and 1% level, respectively.

	AP Performance				AP Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor Models	All		$> q_{10}$		$> q_{20}$	
					$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
Return(%)	42.85	44.69	44.94	CAPM	38.96***	0.008	37.41***	0.089	37.11***	0.075
Std.Dev.(%)	23.92	29.76	36.48	FFC	40.98***	0.104	37.88***	0.205	37.37 ***	0.145
Sharpe	1.79	1.50	1.23	FFC+PS	40.89***	0.104	37.70***	0.205	36.58***	0.146
Skewness	3.24	8.67	11.0	FF5	43.70***	0.166	41.09 ***	0.240	40.91***	0.175
Kurtosis	19.01	112,25	157.99	FF6	44.95***	0.179	41.90***	0.244	41.48***	0.176
Turnover	0.18	0.19	0.20	SY	47.93***	0.091	43.73***	0.189	42.66***	0.139
MDD	0.07	0.04	0.06	Q4	46.04***	0.162	43.41***	0.229	43.19***	0.161

Instead, our illustrations serve to demonstrate the flexibility and versatility of the AP framework. Depending on the specific case in investment advising or trading, the objective and market environment can be correspondingly specified and incorporated into the training process. Note that to accommodate under supervised learning these market interactions and flexible management objectives, one has to label the data using a theoretical model or using proxies based on historical observations, which may be unavailable, computationally expensive, or yield poor performance. Take the last example, without theoretical guidance or gross approximation, it is unclear what the historically "correct" performance is when we train the model using supervised learning. RL, in contrast, is a framework proven in the AI field to perform well in solving such problems.

5 Economic Distillation of AlphaPortfolio

We have to recognize that unlike genetics and physical laws, business environments and financial markets are evolving constantly. Policies and consumer preferences change all the time. We cannot always take machine learning packages and big data analytics off the shelf and apply them blindly to problems in economics and finance just because it seems to predict well in backtesting. Moreover, the use of big data and AI may present prejudice

against groups of individuals.[25] The perils are particularly worrisome because of the fast and nonstationary dynamics in social sciences. A necessary step to address the concern is to understand complex AI and machine learning models.

To this end, we introduce an "economic distillation" procedure. In this part, we describe the polynomial feature sensitivity analysis. The main idea is to project AP onto a modeling space that is simpler and more transparent. The distilled model "represents" or mimics the complex AI model and therefore can reveal important properties of the original model and help with its economic interpretation.

For illustration, we project AP onto a modeling space of linear regressions using Algorithm 1. We first express the function of historical features of a stock to its score in the TE-CAAN system. We then examine the marginal contribution of each feature and inspect its comparative statics when other features change. The procedure allows us to identify the variables (or their higher-order terms or interaction terms) that matter the most in the model. Next, we use these variables and their higher-order and interaction terms as input variables to estimate a Lasso regression model. We set penalty parameters such that 50-60 inputs are selected, which is comparable to the number of input time series in AP.

To complement the analysis, in Appendix D, we also regress AP's winner scores onto each firms' corresponding textual loadings on various topics discussed in the firms' filings. The projection of the model onto the natural language space using textual factors (Cong, Liang, and Zhang, 2018; Cong, Liang, Yang, and Zhang, 2019) helps enhance our understanding of how AlphaPortfolio behaves.

5.1 Polynomial Sensitivity Analysis

For economic distillation, we adopt the gradient-based characteristic importance method to determine which characteristics AP mostly depends on. We use $s = \mathcal{F}(\boldsymbol{X})$ to denote a combined network of TE and CAAN, which maps asset's state history $\boldsymbol{X}$ to its winner score s. x_q is used to denote an element of $\boldsymbol{X}$, which is the value of feature q. Given the state history $\boldsymbol{X}$ of an asset, the sensitivity of s to x_q can be calculated as

$$\delta_{x_q}(\boldsymbol{X}) = \lim_{\Delta x_q \to 0} \frac{\mathcal{F}(\boldsymbol{X}) - \mathcal{F}\left(x_q + \Delta x_q, \boldsymbol{X}_{\neg x_q}\right)}{x_q - (x_q + \Delta x_q)} = \frac{\partial \mathcal{F}(\boldsymbol{X})}{\partial x_q}, \tag{6}$$

[25] For example, COMPAS that guides criminal sentencing in the United States could introduce racial biases, according to a report by ProPublica; Algorithm PredPol designed to predict crime locations leads police to unfairly target certain neighborhoods; Joy Buolamwini points out that gender-recognition AIs from IBM, Microsoft, and Chinese company Megvii increase the risk of false identification of women and minorities; even Google searches could propagate biases against women CEOs and job seekers. Such phenomena have consequential socioeconomic impacts on labor market dynamics, wealth inequality, etc. While people often attribute these issues to training data, algorithms and models may also have embedded stories and ideology due to designers' negligence or cultural insensitivity. Algorithms thus may (unintentionally) perpetuate initial random errors and induce undesirable behaviors by catering to users' addiction and bigotry. In a sense, it is equally important to understand a black-box model as it is to carefully process and sample data.

where $\boldsymbol{X}_{\neg x_q}$ denotes the element $\boldsymbol{X}$ except x_q.

In our implementation of AP using PyTorch, the gradient follows from the autograd module in the deep learning package. For all possible stock states in a market, the average influence of the stock state feature x_q to the winner score s is:

$$\mathbb{E}\left[\delta_{x_q}\right] = \int_{D_{\boldsymbol{X}}} \Pr(\boldsymbol{X})\delta_{x_q}(\boldsymbol{X})\ \mathrm{d}_\sigma, \tag{7}$$

where $\Pr(\boldsymbol{X})$ is the probability density function of $\boldsymbol{X}$, and $\int_{D_{\boldsymbol{X}}} \cdot\ \mathrm{d}_\sigma$ is an integral over all possible values of $\boldsymbol{X}$. According to the Law of Large Numbers, given a dataset that contains historical states of I stocks in N holding periods, the $E\left[\delta_{x_q}\right]$ is approximated as

$$\mathbb{E}\left[\delta_{x_q}\right] = \frac{1}{I \times N} \sum_{n=1}^{N} \sum_{i=1}^{I} \delta_{x_q}\left(\boldsymbol{X}_n^{(i)} \middle| \mathcal{X}_n^{(\neg i)}\right), \tag{8}$$

where $\boldsymbol{X}_n^{(i)}$ is the historical state of the i-th stock at the n-th holding period, and $\mathcal{X}_n^{(\neg i)}$ denotes the historical states of other stocks that are concurrent with the historical state of the i-th stock.

We use $\mathbb{E}\left[\left|\delta_{x_q}\right|\right]$ to measure the overall influence of an asset feature x_q on the winner score. We then generate polynomial terms with the most important features. For each month in the OOS periods, we can distill the AP model by regressing winner scores to selected terms using Lasso. Results in Table 12 show that even the distilled linear model achieves significant performance in OOS tests. Here $poly = 1$ is essentially a linear regression. One can include higher-degree polynomial terms in the distillation exercises, and we stop at degree-2 for parsimony. Note that the distillation utilizes knowledge from the trained AlphaPortfolio model and underperforms the original model. Hence, it is not the case that we should or can effectively deploy the distilled model for trading in place of AlphaPortfolio.

Algorithm 1: pseudo-code for economic distillation approach one

Input: Model parameters $\boldsymbol{\theta}$ of AlphaPortfolio trained on $\mathcal{D}_{train}$; Test data set $\mathcal{D}_{test} = \{\mathcal{X}_1, \ldots, \mathcal{X}_N\}$ which consists of N trading periods; Hyperparameters $\{K, \alpha, p\}$;

Output: Evaluation metrics of the test period;

for $n = 1$ to N **do**
  for each $\boldsymbol{X}^{(i)}$ in $\mathcal{X}_n$, generate winner score $\boldsymbol{s}_n = \left\{s^{(1)}, \ldots, s^{(I)}\right\}$ from AP;
  Calculate gradients of $s^{(i)}$ to each input raw characteristic q as $\delta_{x_q}(\boldsymbol{X})$;
  Select characteristics with top $K\%$ of $\mathbb{E}\left[\left|\delta_{x_q}\right|\right]$;
  Generate p-degree polynomial terms with selected characteristics $\boldsymbol{q}_n^{selected}$;
  Select important terms with Lasso regression(Penalty factor $= \alpha$);
  Regress $\boldsymbol{s}_n$ to selected terms, obtain corresponding coefficient;
  Re-generate winner score $\boldsymbol{s}'_n$ using selected terms and coefficients;
  Using $\boldsymbol{s}'_n$ to calculate rate of return r_n of this trading period;
end for
Calculate evaluation metrics given $R = \{r_1, \ldots, r_N\}$;

Table 12: Out-of-Sample Performance of Distillation for Algorithm 1 with Polynomial of Degree Two

In each month of the OOS periods, the AlphaPortfolio is distilled into a linear model, and winner scores on the test sample are regenerated. Portfolios are formed according to the distilled winner scores following the same strategy as in Table 1. q_n symbolizes the n^{th} NYSE size percentile. Return, Std.Dev., and Sharpe ratio all annualized.

Algo1_Poly2 Performance					Algo1_Poly2 Excess Alpha					
	(1)	(2)	(3)		(4)	(5)	(6)	(7)	(8)	(9)
Firms	All	$> q_{10}$	$> q_{20}$	Factor Models	All		$> q_{10}$		$> q_{20}$	
					$\alpha(\%)$	R^2	$\alpha(\%)$	R^2	$\alpha(\%)$	R^2
Return(%)	16.80	22.20	19.20	CAPM	14.1***	0.000	17.9***	0.054	15.0***	0.095
Std.Dev.(%)	16.40	12.40	9.40	FFC	15.8***	0.024	18.2***	0.196	15.9***	0.411
Sharpe	1.02	1.79	2.04	FFC+PS	15.4***	0.024	17.5***	0.197	14.2***	0.428
Skewness	1.89	6.21	2.21	FF5	16.8***	0.044	19.5***	0.252	17.0***	0.447
Kurtosis	7.35	65.62	8.87	FF6	17.8***	0.062	20.3***	0.272	18.0***	0.503
Turnover	0.60	0.40	0.40	SY	21.7***	0.053	22.0***	0.188	19.1***	0.361
MDD	0.08	0.03	0.03	Q4	17.2***	0.042	20.6***	0.263	18.2***	0.505

5.2 Important Features and Dynamic Patterns

We then carry out both panel or Fama-Macbeth-type regressions to combine each monthly distilled linear model. We can do this for polynomials of different degrees. For parsimony, we report in Table 13 the top 50 dominant features as functions of firm characteristics and

market signals (e.g., price and daily volume), together with the corresponding t-values. The sign and magnitudes of the t-values allow a glimpse into how each selected feature affects the portfolio construction.

To investigate the time-varying effects of dominant characteristics, we plot heatmaps in Figure 5 for the top 15 terms in degree-2 polynomial functions to highlight their rankings in the OOS periods. We also outline basic statistics for both degree-2 and degree-1 polynomials in Table 14. For distillation with degree-2 polynomials across Ranks 1, 2, and 3, the top contributing variables are ivc (82.4%), ipm^2 (50.3%), Q (36.7%), ivc^2 (22.2%), delta_so (21.6%), and C (10.5%).[26] For degree-1 polynomials across Rank 1, 2, and 3, at the top again are ivc (97.8%), Idol_vol (43.2%), and ipm (26.9%). The other five important characteristics (free_cf, Ret_max, ret, delta_so, and C) account for 13.9% to 19.1% each.

From Figure 5 and Tables 13 and 14, we find that a small set of stock features determines the performance of our algorithm. For example, the inventory change (ivc) plays a key role in our algorithm with a probability of more than 80% included in the top contributing factors in both degree-1 and 2 polynomials. Thomas and Zhang (2002) first document that ivc can negatively predict stocks' future returns, which is consistent with earning management of firms. Given that ivc still plays an important role post 2002, the anomaly has not been traded away. Short-term previous return (ret_11 and ret_10) are strongly negatively significant, especially for portfolios with large stocks, implying a short-term reversal, which is consistent with Avramov, Cheng, and Metzker (2019). Note that the signs of certain firm characteristics are different for different lags, which potentially reflect the path-dependent nature of AP.[27]

Other factors including Tobin's Q, pretax profit margin (ipm), ratio of cash and short-term investment to total asset (C), idiosyncratic volatility (Idol_vol) etc. are also prominent. Among them, idiosyncratic volatility (Idol_vol), max per daily return in a month (Ret_max), etc., are arbitrage constraints and market signals related to trading; growth in external financing (fcf), operating income before depreciation and tax (ipm), etc., are financial signals related to firms' fundamentals. Trading signals affect stock returns through mispricing channel while financial signals do so likely through risk channels (Livdan, Sapriza, and Zhang, 2009). The patterns not only imply that future studies could focus on time-varying relevance of a small set of economic mechanisms and variables, but also tell researchers which of the features' nonlinear effects to consider.

To further analyze the rotation patterns of dominant features, we compute the Pearson correlation coefficients both pair-wise and between trading (Idol_vol, Idol_vol^2, Ret_max, ret) and financial (Q, C, C^2, delta_so, ivc, ipm^2, ivc^2, investment, free_cf) signals. Table 15 reveals that inventory changes (ivc) and pretax profit margin to the second power (ipm^2)

[26] Percentages in brackets denote the fraction to be top contributing variables across all months.

[27] Although not reported here, we also perform the analysis in Table 8 focusing on the top 10 features with 12-month lags. While some variables still exhibit sign changes that indicate path dependence, the signs on ivc, idol_vol, Q, ret, Ret_max consistently follow what theory would prescribe.

Table 13: T-statistics of Selected Features for Algorithm One with Polynomial of Degree Two: Fama-Macbetch Regressions

This table presents the results where Fama-Macbetch regressions are employed to interpret Algorithm 1. Polynomial degree is set to two for all terms in the regression. The suffix "_n" denotes the sequence number of features starting from 12 months ago, *i.e.,* pe_7 denotes P/E ratio at the time of five $(12 - 7 = 5)$ months lag. For details of each characteristic, please refer to Appendix B. q_n symbolizes the n^{th} NYSE size percentile. This table presents t-statistics in the top half in magnitude.

All		size $> q_{10}$		size $> q_{20}$	
pe_7^2	19.05	Q_9	-31.01	Q_9	-23.62
s2p_11^2	17.8	Q_9^2	17.03	Idol_vol_1	14.29
Std_turnover_7^2	-9.4	investment_0	-11.74	Idol_vol_0	13.42
Std_turnover_2^2	9.24	Std_volume_1	-10.6	Q_9^2	12.56
Std_volume_11^2	8.93	Idol_vol_0	10.12	Std_volume_1	-12.29
Std_turnover_0^2	8.02	free_cf_6	-9.58	free_cf_6	-10.13
ldp_0^2	7.94	Idol_vol_4^2	8.44	investment_7	-9.96
Std_volume_5^2	7.89	Idol_vol_1	8.4	investment_0	-8.71
roa_0	7.79	Idol_vol_1^2	8.01	delta_so_1	-8.6
pe_1^2	7.78	Std_volume_1^2	7.62	Idol_vol_4	8.16
Std_turnover_4^2	7.71	ret_11^2	7.47	lev_7	-7.91
roa_0^2	7.63	free_cf_9	-7.34	ret_10^2	7.88
Std_volume_0^2	7.41	ret_5^2	7.31	Std_volume_10	-7.6
Std_turnover_6^2	7.4	delta_so_1	-7.29	Idol_vol_2	7.57
Std_turnover_1^2	7.3	Idol_vol_0^2	7.24	Std_volume_1^2	7.52
Std_turnover_3^2	-7.25	ret_10^2	7.01	delta_so_11	-7.41
Std_turnover_5^2	7.16	Ret_max_7	6.96	Idol_vol_4^2	7.35
pe_7	6.29	delta_so_11	-6.92	ret_9^2	7.27
Beta_daily_3^2	6.16	ldp_6	-6.82	nop_8	-7.13
Std_volume_4^2	6.09	Ret_max_10^2	6.81	Idol_vol_1^2	7.08
Std_turnover_9^2	5.86	s2p_4^2	6.58	beme_9	6.99
roa_11^2	5.71	ret_10	-6.58	pe_7	6.86
Beta_daily_8^2	5.63	s2p_0^2	6.52	ret_2^2	6.85
roa_11	5.48	ret_5	-6.49	ret_10	-6.84
o2p_11^2	-5.34	Ret_max_3^2	6.45	delta_so_8^2	6.79
roe_11^2	5.2	ret_9^2	6.37	Ret_max_9^2	6.78
roa_6^2	5.18	C_2	6.28	ret_11^2	6.75
s2p_10^2	5	ret_2^2	6.28	C_2^2	6.65
ret_11^2	4.97	pe_7	6.25	beme_8	6.65
nop_0^2	4.85	investment_7	-6.16	free_cf_9	-6.65
roa_6	4.74	ret_6^2	6.1	ivc_0^2	6.63
s2p_11 ivc_10	4.52	sat_6	6.08	ret_5^2	6.48
roe_5^2	4.51	ret_11	-6.06	free_cf_8	-6.44
ldp_6^2	4.5	ret_3	-5.95	Idol_vol_0^2	6.38
Turnover_11^2	4.49	ivc_0^2	5.91	Ret_max_10^2	6.38
lev_5^2	4.48	ldp_6^2	5.91	Ret_max_3^2	6.25
roa_3	4.4	sat_6^2	5.78	sat_6^2	6.19
std_4^2	4.4	Ret_max_9^2	5.71	ret_3^2	6.18
delta_so_11^2	4.37	Idol_vol_2	5.69	me_2^2	6.1
Std_turnover_10^2	4.29	ret_2	-5.66	ret_6^2	6
Beta_daily_11^2	4.23	ret_3^2	5.43	Idol_vol_11	6
s2p_11 ivc_5	4.18	s2p_4	-5.36	ldp_6	-6
roc_8	4.01	ol_11	5.15	ret_11	-5.89
Std_volume_9^2	3.9	Turnover_9	-5.15	sga2s_9	5.73
s2p_11 noa_11	3.87	Idol_vol_4	5.1	shrout_9^2	5.72
ivc_11^2	3.74	e2p_10	5.03	s2p_0	5.67
noa_11 roa_6	3.54	free_cf_1	-5.02	rna_2^2	5.58
delta_shrout_11^2	3.5	nop_8	-4.95	s2p_4^2	5.55
Std_turnover_11^2	3.45	ivc_11^2	4.94	delta_so_5	-5.54

Figure 5: The following three heatmaps illustrate how the ranking of dominant features change over time. The most dominant features are inventory change (ivc), idiosyncratic volatility (Idol_vol), change in shares outstanding (delta_so), Tobin's Q (Q), cash and short-term investments to total assets (C), maximum daily return in the month (Ret_max), Pretax profit margin to the second power(ipmˆ2), ivc to the second power (ivcˆ2), investment, cash flow to book value of equity (free_cf), sale-to-price (s2p), C to the second power (Cˆ2), standard deviation of daily turnover (Std_volume), Idol_vol to the second power (Idol_volˆ2) and monthly return (ret). Appendix B provides detailed description of the features.

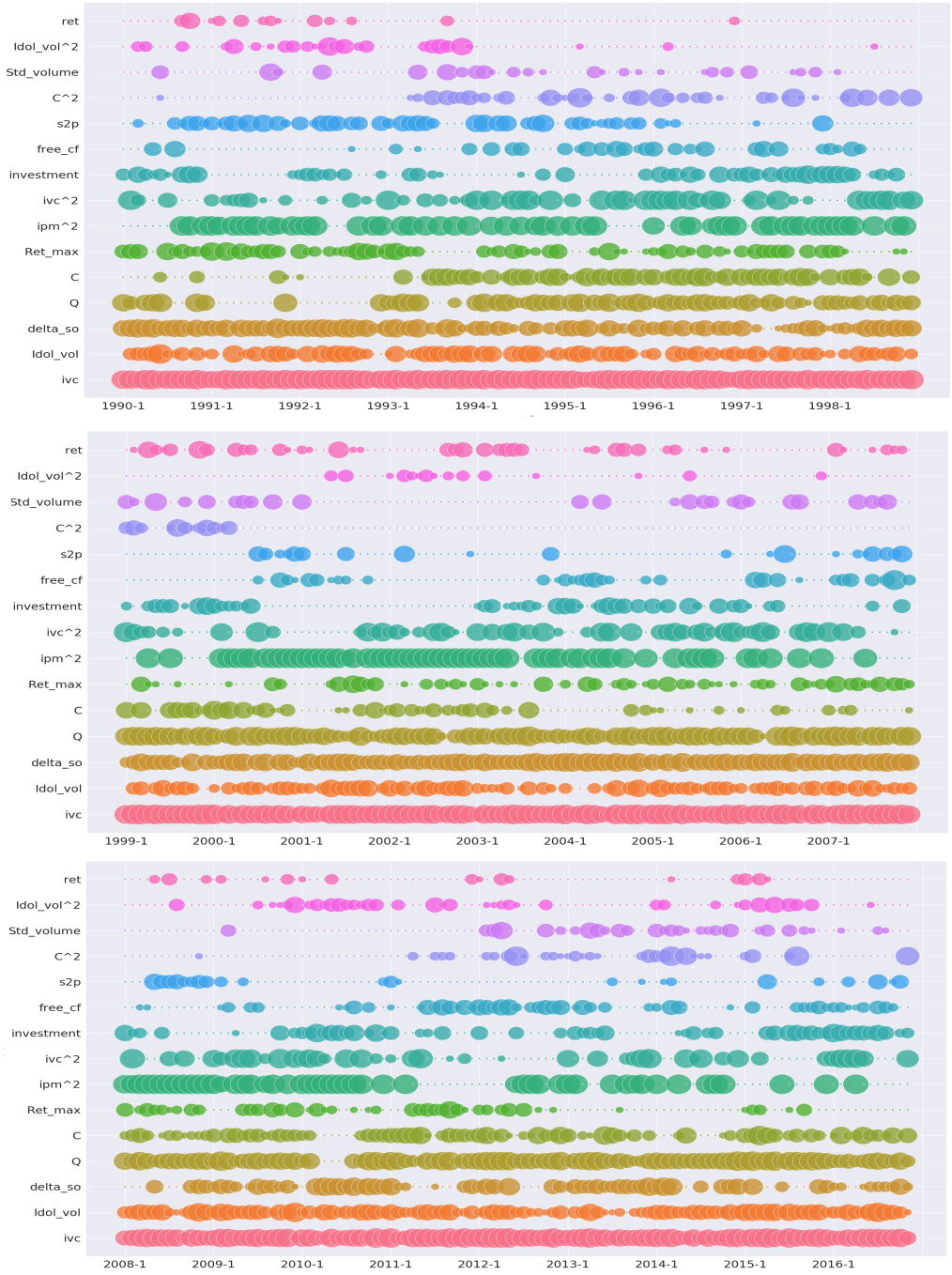


Table 14: Dominant Features after Economic Distillation

This table presents the probability of features ranked in top three across different months in the distillation Algorithm 1. In each month of the OOS periods, the AlphaPortfolio is distilled into a second degree polynomial function (Panel A) and a one degree polynomial function (Panel B). The features are calculated by summing up the absolute value of historical 12 months' regression coefficients. Features are hence ranked by their importance in each month.

Panel A: Polynomial Degree Two					
Rank 1		Rank 2		Rank 3	
ipmˆ2	45.7%	ivc	36.7%	ivc	24.7 %
ivc	21.0 %	Q	16.7 %	Q	17.9 %
ivcˆ2	9.9 %	ivcˆ2	8.3 %	delta_so	14.8%
sga2sˆ2	5.2%	delta_so	6.5 %	C	8.3%
investmentˆ2	3.7%	pmˆ2	6.2%	Idol_vol	5.6%
Q	2.2%	investmentˆ2	3.7%	ivcˆ2	4.0%
Panel B: Polynomial Degree One					
Rank 1		Rank 2		Rank 3	
ivc	63.3%	ivc	25.3%	Idol_vol	18.8%
ipm	21.9%	Idol_vol	19.8%	delta_so	13.3%
Idol_vol	4.6%	free_cf	8.3 %	C	13.3%
investment	1.9%	ret	7.7 %	ret	9.3%
free_cf	1.5%	Ret_max	5.2 %	ivc	9.3%
sga2s	1.2%	C	5.2 %	Ret_max	9.0%

take turns to play important roles in AlphaPortfolio construction and so do corporate liquidity (C, Cˆ2) and changes in shares outstanding (delta_so) among others. Moreover, features related to trading and those related to financials also take turns to dominate. The rotation patterns are highly significant based on the p-values.

5.3 Distillation on the Training Set

Algorithm 2 shown next describes an alternative distillation exercise. While Algorithm 1 mimics AlphaPortfolio concerning OOS tests, Algorithm 2 distills what AlphaPortfolio has learned from the training set. The former gives information on how a model behaves on a test set while the latter describes what the model learns from a training set. We find similar results using Algorithm 1 and omit the details. Both algorithms can be extended to capture persistent latent variables in future work.

Table 15: Pearson Correlation Coefficients and Feature Rotations

This table presents the Pearson correlations among prominent features constructed from firm characteristics and market signals. Panel A contains the pair of dominant features with the most negative Pearson correlations. Panel B contains the correlation between two subsets of the most dominant features: trading and financial. Trading subset contains {Idol_vol, Idol_volˆ2, Ret_max, ret}, and financial subset contains {Q, C, Cˆ2, delta_so, ivc, ipmˆ2, ivcˆ2, investment, free_cf}.

Term 1	Term 2	Correlation	P-value
Panel A: Pairwise Correlation of Dominant Features			
ivc	ipmˆ2	-0.38	1.55×10^{-12}
delta_so	Cˆ2	-0.33	1.40×10^{-12}
C	delta_so	-0.31	2.09×10^{-8}
Cˆ2	ret	-0.26	1.47×10^{-6}
investment	s2p	-0.25	4.20×10^{-6}
Idol_vol	ipmˆ2	-0.24	1.22×10^{-5}
ipmˆ2	Cˆ2	-0.23	3.79×10^{-5}
Ret_max	Cˆ2	-0.22	8.43×10^{-5}
C	ret	-0.21	1.80×10^{-4}
delta_so	Idol_volˆ2	-0.2	2.90×10^{-4}
Panel B: Trading Signals VS. Financial Signals			
Trading_related	Financial_related	-0.33	8.00×10^{-10}

Algorithm 2: pseudo-code for economic distillation approach two

Input: Model parameters $\boldsymbol{\theta}$ of AlphaPortfolio trained on $\mathcal{D}_{train}$; Training set $\mathcal{D}_{train}$; Test data set $\mathcal{D}_{test} = \{\mathcal{X}_1, \ldots, \mathcal{X}_N\}$; Hyperparameters $\{K, \alpha, p\}$;

Output: Evaluation metrics of test period;

Generate winner score $\boldsymbol{S}$ of all trading periods in $\mathcal{D}_{train}$ using AP;
Calculate gradient-based sentivity for each raw feature q as $\mathbb{E}\left[\delta_{x_q}\right]$;
Select characteristics with top $K\%$ of $\mathbb{E}\left[\left|\delta_{x_q}\right|\right]$;
Generate p-degree polynomial terms with selected characteristics $\boldsymbol{q}^{selected}$;
Select important terms with Lasso regression(Penalty factor = α);
Regress $\boldsymbol{S}$ to selected terms, obtain corresponding coefficients;
for $n = 1$ to N **do**
    Use $\mathcal{X}_n$ to construct selected terms;
    Regenerate winner score $\boldsymbol{s}'_n$ using selected terms and coefficients;
    Using $\boldsymbol{s}'_n$ to calculate rate of return r_n of this trading period;
end for
Calculate evaluation metrics given $R = \{r_1, \ldots, r_N\}$;

5.4 Economic Distillations: The Case of LSTM-HA

Overall, economic distillation provides us a basis to better understand and interpret our machine learning model. It informs us of the key input features and the way they matter

(interaction or higher-order, etc.) so that we can adjust the model accordingly when the economic environment changes.[28] It also serves as a sanity check of the complex machine learning model in the sense that if something in the distilled model appears strange, such as a stale price playing a dominant role when it should have been subsumed in other features, it is likely that the complex model contains a misspecification or coding error.

Appendix C2 reports the performance of an AP model based on LSTM. On the positive side, the Bi-LSTM-CAAN model achieves an OOS Sharpe ratio of around 3 after excluding small-cap stocks. This confirms the superior performance of the AP framework and even outperforms the TE-based AP. On the negative side, our economic distillation in Appendix C2 reveals that LSTM-HA does not have stable utilization of input features and suffers from exploding gradient issues, suggesting that the trained model goes to some extremes and such RNN-like models may not be the most suitable. This is consistent with that computer scientists deal with vanishing and exploding gradients only in the training sample, but such issues may resurface in the test samples or actual deployment. Appendix C2 thus provides a case in point of how economic distillation can help assess model stability and functionality, especially on test sets, which computer science studies typically leave out.

6 Conclusion

We propose deep reinforcement learning (RL)-based portfolio management, an alternative that improves over traditional indirect portfolio-construction frameworks. We develop a multi-sequence learning model building on the latest AI innovations in order to effectively capture the high-dimensional, nonlinear, noisy, interacting, and path-dependent nature of financial data and market environments, and then optimizing it using RL. While our key contribution lies in the conceptual framework, the resulting AlphaPortfolio yields superb OOS performances under various economic and trading restrictions as well as management objectives, as seen in the U.S. equity market, making it deployable by practitioners for trading and investment advising.

Our framework and empirical findings have broader implications on the utility of RL in social sciences and the importance of economically interpretable AI. Unlike supervised learning that requires knowledge of the environment by way of examples of desirable behaviour, RL represents a new approach for goal-directed learning in a complex environment or action space. Deep RL is routinely used in and has seen great commercial success with applications for speech recognition, natural language processing, computer vision, interactive games, etc. (Amazon Alexa, Apple Siri, AlphaGo, and Google Android are leading examples). Moreover,

[28] Interaction terms do not contribute to AP's performance significantly. But this is not at odds with studies that emphasize interaction effects because those studies focus on estimating SDF or minimizing estimation errors in assets' return moments whereas we focus on portfolio performance.

most models using regressions, SVM, and neural networks have RL-based implementations. Portfolio management is just one of the potential applications of RL to tackle complex social science problems with well-specified objectives, but limited pre-existing knowledge or labeled data for deriving full solutions.

Moreover, our "economic distillation" not only reveals key firm characteristics (including their rotation and nonlinearity) that drive AlphaPortfolio's performance, but also provides a concrete backbone for and an incremental step towards interpretations of machine learning and AI applications in business practice and social sciences. Our polynomial sensitivity analysis innovates on current practices in computer science and allows great flexibility. For example, one can put in third-order and fourth-order terms of a feature if one deems them important. Textual-factor analysis derives from topic modeling and word embedding and constitutes one of the many possibilities of using natural languages to better explain model behaviors. Both procedures are projections of complex models into transparent and interpretable spaces. Other applications of our economic distillation approach constitute interesting future research.

References

Aït-Sahali, Yacine, and Michael W Brandt, 2001, Variable selection for portfolio choice, *The Journal of Finance* 56, 1297–1351.

Alsabah, Humoud, Agostino Capponi, Octavio Ruiz Lacedelli, and Matt Stern, 2019, Robo-advising: Learning investors risk preferences via portfolio choices, *Journal of Financial Econometrics*.

Athey, Susan, 2018, The impact of machine learning on economics, in *The economics of artificial intelligence: An agenda* (University of Chicago Press).

Avramov, Doron, Si Cheng, and Lior Metzker, 2019, Machine learning versus economic restrictions: Evidence from stock return predictability, *Available at SSRN 3450322*.

Avramov, Doron, Tarun Chordia, Gergana Jostova, and Alexander Philipov, 2013, Anomalies and financial distress, *Journal of Financial Economics* 108, 139–159.

Bailey, David H, and Marcos López de Prado, 2012, *Journal of Investment Strategies (Risk Journals)*1.

Baker, Malcolm, and Jeffrey Wurgler, 2007, Investor sentiment in the stock market, *Journal of Economic Perspectives* 21, 129–152.

Barocas, Solon, Moritz Hardt, and Arvind Narayanan, 2017, Fairness in machine learning, *NIPS Tutorial*.

Barry, Christopher B, 1974, Portfolio analysis under uncertain means, variances, and covariances, *The Journal of Finance* 29, 515–522.

Bartlett, Robert, Adair Morse, Richard Stanton, and Nancy Wallace, 2019, Consumer-lending discrimination in the fintech era, Discussion paper, National Bureau of Economic Research.

Bawa, Vijay S, Stephen J Brown, and Roger W Klein, 1979, Estimation risk and optimal portfolio choice, *NORTH-HOLLAND PUBL. CO., N. Y., 190 pp.*

Best, Michael J, and Robert R Grauer, 1991, On the sensitivity of mean-variance-efficient portfolios to changes in asset means: some analytical and computational results, *The Review of Financial Studies* 4, 315–342.

Black, Fischer, and Robert Litterman, 1990, Asset allocation: combining investor views with market equilibrium, Discussion paper, Discussion paper, Goldman, Sachs & Co.

——— , 1992, Global portfolio optimization, *Financial Analysts Journal* 48, 28–43.

Brandt, Michael W, 1999, Estimating portfolio and consumption choice: A conditional euler equations approach, *The Journal of Finance* 54, 1609–1645.

——— , 2010, Portfolio choice problems, in *Handbook of financial econometrics: Tools and techniques* . pp. 269–336 (Elsevier).

——— , and Pedro Santa-Clara, 2006, Dynamic portfolio selection by augmenting the asset space, *The Journal of Finance* 61, 2187–2217.

——— , and Rossen Valkanov, 2009, Parametric portfolio policies: Exploiting characteristics in the cross-section of equity returns, *The Review of Financial Studies* 22, 3411–3447.

Bryzgalova, Svetlana, Markus Pelger, and Jason Zhu, 2020, Forest through the trees: Building cross-sections of stock returns, *Working Paper*.

Bucilu, Cristian, Rich Caruana, and Alexandru Niculescu-Mizil, 2006, Model compression, in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* pp. 535–541. ACM.

Carhart, Mark M, 1997, On persistence in mutual fund performance, *The Journal of Finance* 52, 57–82.

Chen, Luyang, Markus Pelger, and Jason Zhu, 2020, Deep learning in asset pricing, Discussion paper, Working paper.

Chinco, Alexander M, Andreas Neuhierl, and Michael Weber, 2019, Estimating the anomaly base rate, Discussion paper, National Bureau of Economic Research.

Chordia, Tarun, Avanidhar Subrahmanyam, and Qing Tong, 2014, Have capital market anomalies attenuated in the recent era of high liquidity and trading activity?, *Journal of Accounting and Economics* 58, 41–58.

Cochrane, John H, 2011, Presidential address: Discount rates, *The Journal of Finance* 66, 1047–1108.

Cong, Lin William, Shiyang Huang, and Douglas Xu, 2020, Rise of factor investing: security design and asset pricing implications, .

Cong, Lin William, Tengyuan Liang, Baozhong Yang, and Xiao Zhang, 2019, Analyzing textual information at scale, *Economic Information to Facilitate Decision Making* Forthcoming, edited by Kashi Balachandran.

Cong, Lin William, Tengyuan Liang, and Xiao Zhang, 2018, Textual factors: A scalable, interpretable, and data-driven approach to analyzing unstructured information, Discussion paper, resubmission requested.

Cong, Lin William, Ke Tang, Jingyuan Wang, and Yang Zhang, 2020, Deep sequence modeling: Development and applications in asset pricing, *Journal of Financial Data Science* Forthcoming.

D'Acunto, Francesco, and Alberto G Rossi, 2020, Robo-advising, .

Dasa, Sanjiv R, Daniel Ostrova, Anand Radhakrishnanb, and Deep Srivastavb, 2018, A new approach to goals-based wealth management, *Journal Of Investment Management* 16, 1–27.

Datta, Anupam, and Shayak Sen, 2018, global cluster explanations for machine learning models, .

de Prado, Marcos López, 2016, Building diversified portfolios that outperform out of sample, *The Journal of Portfolio Management* 42, 59–69.

——— , 2018, *Advances in financial machine learning* (John Wiley & Sons).

Delarue, Arthur, Ross Anderson, and Christian Tjandraatmadja, 2020, Reinforcement learning with combinatorial actions: An application to vehicle routing, *arXiv preprint arXiv:2010.12001*.

DeMiguel, Victor, Lorenzo Garlappi, and Raman Uppal, 2007, Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy?, *The Review of Financial Studies* 22, 1915–1953.

Deng, Yue, Feng Bao, Youyong Kong, Zhiquan Ren, and Qionghai Dai, 2017, Deep direct reinforcement learning for financial signal representation and trading, *IEEE TNNLS* 28, 653–664.

Detemple, Jerome, and Shashidhar Murthy, 1997, Equilibrium asset prices and no-arbitrage with portfolio constraints, *The Review of Financial Studies* 10, 1133–1174.

Ding, Yi, Weiqing Liu, Jiang Bian, Daoqiang Zhang, and Tie-Yan Liu, 2018, Investor-imitator: A framework for trading knowledge extraction, in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* pp. 1310–1319. ACM.

Ebert, Frederik, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine, 2018, Visual foresight: Model-based deep reinforcement learning for vision-based robotic control, *arXiv preprint arXiv:1812.00568*.

Fama, Eugene F, and Kenneth R French, 2015, A five-factor asset pricing model, *Journal of Financial Economics* 116, 1–22.

——— , 2018, Choosing factors, *Journal of Financial Economics* 128, 234–252.

Farboodi, Maryam, and Laura Veldkamp, 2019, Long run growth of financial data technology, Discussion paper, .

Feng, Guanhao, Stefano Giglio, and Dacheng Xiu, 2020, Taming the factor zoo: A test of new factors, *The Journal of Finance* 75, 1327–1370.

Feng, Guanhao, Jingyu He, and Nicholas G Polson, 2018, Deep learning for predicting asset returns, *arXiv preprint arXiv:1804.09314*.

Feng, Guanhao, Nick Polson, and Jianeng Xu, 2018, Deep learning factor alpha, *Available at SSRN 3243683*.

Fischer, Thomas G, 2018, Reinforcement learning in financial markets-a survey, Discussion paper, FAU Discussion Papers in Economics.

Frazzini, Andrea, Ronen Israel, and Tobias J Moskowitz, 2018, Trading costs, *Available at SSRN 3229719*.

Freyberger, Joachim, Andreas Neuhierl, and Michael Weber, 2020, Dissecting characteristics nonparametrically, *The Review of Financial Studies* 33, 2326–2377.

Friedman, Jerome H., 2002, Stochastic gradient boosting, *Computational Statistics & Data Analysis* 38, 367378.

Fu, Justin, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine, 2020, D4rl: Datasets for deep data-driven reinforcement learning, *arXiv preprint arXiv:2004.07219*.

Fujimoto, Scott, David Meger, and Doina Precup, 2019, Off-policy deep reinforcement learning without exploration, in *International Conference on Machine Learning* pp. 2052–2062. PMLR.

Garlappi, Lorenzo, Raman Uppal, and Tan Wang, 2006, Portfolio selection with parameter and model uncertainty: A multi-prior approach, *The Review of Financial Studies* 20, 41–81.

Garychl, 2018, Applications of reinforcement learning in real world, *Medium* Aug 1, 2018.

Goldfarb, Donald, and Garud Iyengar, 2003, Robust portfolio selection problems, *Mathematics of Operations Research* 28, 1–38.

Goodman, Bryce, and Seth Flaxman, 2017, European union regulations on algorithmic decision-making and a right to explanation, *AI Magazine* 38, 50–57.

Green, Richard C, and Burton Hollifield, 1992, When will mean-variance efficient portfolios be well diversified?, *The Journal of Finance* 47, 1785–1809.

Gu, Shihao, Bryan Kelly, and Dacheng Xiu, 2020, Empirical asset pricing via machine learning, *The Review of Financial Studies* 33, 2223–2273.

Guidotti, Riccardo, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi, 2018, A survey of methods for explaining black box models, *ACM computing surveys (CSUR)* 51, 1–42.

Han, Yufeng, Ai He, David Rapach, and Guofu Zhou, 2018, What firm characteristics drive us stock returns?, *Available at SSRN 3185335*.

Harvey, Campbell R, Yan Liu, and Heqing Zhu, 2016, ...and the cross-section of expected returns, *The Review of Financial Studies* 29, 5–68.

Heaven, Douglas, 2019, Why deep-learning ais are so easy to fool, *Nature (News Feature)* October 9.

Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean, 2015, Distilling the knowledge in a neural network, *arXiv preprint arXiv:1503.02531*.

Horel, Enguerrand, and Kay Giesecke, 2019a, Computationally efficient feature significance and importance for machine learning models, *arXiv preprint arXiv:1905.09849*.

——— , 2019b, Towards explainable ai: Significance tests for neural networks, *arXiv preprint arXiv:1902.06021*.

Hou, Kewei, Chen Xue, and Lu Zhang, 2015, Digesting anomalies: An investment approach, *The Review of Financial Studies* 28, 650–705.

——— , 2020, Replicating anomalies, *The Review of Financial Studies* 33, 20192133.

Jaques, Natasha, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard, 2019, Way off-policy batch deep reinforcement learning of implicit human preferences in dialog, *arXiv preprint arXiv:1907.00456*.

Jin, Olivier, and Hamza El-Saawy, 2016, Portfolio management using reinforcement learning, Discussion paper, Stanford University.

Jobson, JD, 1979, Improved estimation for markowitz portfolios using james-stein type estimators, in *Proceedings of the American Statistical Association, Business and Economics Statistics Section* vol. 71 pp. 279–284.

Jorion, Philippe, 1986, Bayes-stein estimation for portfolio analysis, *Journal of Financial and Quantitative Analysis* 21, 279–292.

Jurczenko, Emmanuel, 2015, *Risk-based and factor investing* (Elsevier).

Kahn, Gregory, Pieter Abbeel, and Sergey Levine, 2021, Badgr: An autonomous self-supervised learning-based navigation system, *IEEE Robotics and Automation Letters* 6, 1312–1319.

Kalashnikov, Dmitry, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al., 2018, Scalable deep reinforcement learning for vision-based robotic manipulation, in *Conference on Robot Learning* pp. 651–673. PMLR.

Kan, Raymond, and Guofu Zhou, 2007, Optimal portfolio choice with parameter uncertainty, *Journal of Financial and Quantitative Analysis* 42, 621–656.

Karolyi, Andrew, and Stijn Van Nieuwerburgh, 2020, New methods for the cross-section of returns, *The Review of Financial Studies* Forthcoming.

Kelly, Bryan T, Seth Pruitt, and Yinan Su, 2019, Characteristics are covariances: A unified model of risk and return, *Journal of Financial Economics* 134, 501–524.

Kidambi, Rahul, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims, 2020, Morel: Model-based offline reinforcement learning, *arXiv preprint arXiv:2005.05951*.

Kim, Soohun, Robert A Korajczyk, and Andreas Neuhierl, 2019, Arbitrage portfolios, *Available at SSRN*.

Klein, Roger W, and Vijay S Bawa, 1976, The effect of estimation risk on optimal portfolio choice, *Journal of Financial Economics* 3, 215–231.

Koijen, Ralph SJ, Tobias J Moskowitz, Lasse Heje Pedersen, and Evert B Vrugt, 2018, Carry, *Journal of Financial Economics* 127, 197–225.

Kozak, Serhiy, Stefan Nagel, and Shrihari Santosh, 2020, Shrinking the cross-section, *Journal of Financial Economics* 135, 271–292.

Krause, Josua, Adam Perer, and Kenney Ng, 2016, Interacting with predictions: Visual inspection of black-box machine learning models, in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* pp. 5686–5697.

Ledoit, Olivier, and Michael Wolf, 2012, Nonlinear shrinkage estimation of large-dimensional covariance matrices, *The Annals of Statistics* 40, 1024–1060.

——— , 2017, Nonlinear shrinkage of the covariance matrix for portfolio selection: Markowitz meets goldilocks, *The Review of Financial Studies* 30, 4349–4388.

Lettau, Martin, and Markus Pelger, 2020, Factors that fit the time series and cross-section of stock returns, *The Review of Financial Studies* 33, 2274–2325.

Light, Nathaniel, Denys Maslov, and Oleg Rytchkov, 2017, Aggregation of information about the cross section of stock returns: A latent variable approach, *The Review of Financial Studies* 30, 1339–1381.

Linnainmaa, Juhani T, and Michael R Roberts, 2018, The history of the cross-section of stock returns, *The Review of Financial Studies* 31, 2606–2649.

Liu, Hong, and Mark Loewenstein, 2002, Optimal portfolio selection with transaction costs and finite horizons, *The Review of Financial Studies* 15, 805–835.

Liu, John Zhuang, Michael Sockin, and Wei Xiong, 2020, Data privacy and temptation, *Working Paper*.

Livdan, Dmitry, Horacio Sapriza, and Lu Zhang, 2009, Financially constrained stock returns, *The Journal of Finance* 64, 1827–1862.

Ma, Linlin, Yuehua Tang, and Juan-Pedro Gomez, 2019, Portfolio manager compensation in the us mutual fund industry, *The Journal of Finance* 74, 587–638.

Markowitz, Harry, 1952, Portfolio selection, *The Journal of Finance* 7, 77–91.

Martin, Ian, and Stefan Nagel, 2019, Market efficiency in the age of big data, *Working Paper*.

McLean, R David, and Jeffrey Pontiff, 2016, Does academic research destroy stock return predictability?, *The Journal of Finance* 71, 5–32.

Merton, Robert C, 1980, On estimating the expected return on the market: An exploratory investigation, *Journal of Financial Economics* 8, 323–361.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, and Georg Ostrovski, 2015, Human-level control through deep reinforcement learning, *Nature* 518, 529–533.

Moritz, Benjamin, and Tom Zimmermann, 2016, Tree-based conditional portfolio sorts: The relation between past and future stock returns, *Available at SSRN 2740751*.

Nagel, Stefan, 2012, Evaporating liquidity, *The Review of Financial Studies* 25, 2005–2039.

Novy-Marx, Robert, and Mihail Velikov, 2016, A taxonomy of anomalies and their trading costs, *The Review of Financial Studies* 29, 104–147.

Pástor, L'uboš, 2000, Portfolio selection and asset pricing models, *The Journal of Finance* 55, 179–223.

——— , and Robert F Stambaugh, 2000, Comparing asset pricing models: an investment perspective, *Journal of Financial Economics* 56, 335–381.

——— , 2003, Liquidity risk and expected stock returns, *Journal of Political Economy* 111, 642–685.

Powell, James L, James H Stock, and Thomas M Stoker, 1989, Semiparametric estimation of index coefficients, *Econometrica: Journal of the Econometric Society* pp. 1403–1430.

Rapach, David, and Guofu Zhou, 2019, Time-series and cross-sectional stock return forecasting: New machine learning methods, *Available at SSRN 3428095*.

Rapach, David E, Jack K Strauss, and Guofu Zhou, 2013, International stock return predictability: what is the role of the united states?, *The Journal of Finance* 68, 1633–1662.

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin, 2016, Why should i trust you?: Explaining the predictions of any classifier, in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* pp. 1135–1144. ACM.

Rossi, Alberto G, 2018, Predicting stock market returns with machine learning, Discussion paper, Working paper.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams, 1986, Learning representations by back-propagating errors, *Nature* 323, 533.

Sak, Halis, Tao Huang, and Michael T Chng, 2019, Exploring the factor zoo with a machine-learning portfolio, *Working Paper*.

Silver, David, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, and Adrian Bolton, 2017, Mastering the game of go without human knowledge, *Nature* 550, 354–359.

Stambaugh, Robert F, Jianfeng Yu, and Yu Yuan, 2012, The short of it: Investor sentiment and anomalies, *Journal of Financial Economics* 104, 288–302.

Stambaugh, Robert F, and Yu Yuan, 2017, Mispricing factors, *The Review of Financial Studies* 30, 1270–1315.

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan, 2017, Axiomatic attribution for deep networks, in *Proceedings of the 34th International Conference on Machine Learning-Volume 70* pp. 3319–3328. JMLR. org.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le, 2014, Sequence to sequence learning with neural networks, *NIPS'14* pp. 3104–3112.

Sutton, Richard S, and Andrew G Barto, 2018, *Reinforcement learning: An introduction* (MIT press).

Thomas, Jacob K, and Huai Zhang, 2002, Inventory changes and future returns, *Review of Accounting Studies* 7, 163–187.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, 2017, Attention is all you need, in Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, ed.: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA* pp. 5998–6008.

Wang, Jingyuan, Yang Zhang, Ke Tang, Junjie Wu, and Zhang Xiong, 2019, Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks, in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* pp. 1900–1908.

Wu, Mike, Michael C Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez, 2018, Beyond sparsity: Tree regularization of deep models for interpretability, in *Thirty-Second AAAI Conference on Artificial Intelligence*.

Appendix A. Basics of Reinforcement Learning

Articles and books on the basics of neural networks and their applications are widely available. In this appendix, we briefly introduce the basics of reinforcement learning (as opposed to supervised or unsupervised learning).

Reinforcement learning (RL) is "learning what to do — how to map situations to actions— so as to maximize a numerical reward signal" and its defining features are "trial" (Sutton and Barto, 2008). RL is one of the three basic machine learning paradigms, alongside supervised learning and unsupervised learning. RL typically solves a reward maximization problem in a Markov-decision-process (MDP) setting in which an agent makes the best decision given its information set under a stochastically-evolving environment.

In RL, an agent must be able to learn about the state of its environment and take actions that potentially affect the state going forward. Below we denote the set of possible states to be S, and the set of possible actions to be A. Beyond the agent and the environment, the other four main elements of a reinforcement learning system are: a *policy*, a *reward signal*, a *value function*, and, optionally, a *model* of the environment.

1. *Policy*: A policy defines the agent's way of behaving at a given time. It is similar to the (behavioral) *strategy* concept in sequential games and determines behavior. In general, a policy is a mapping from perceived states of the environment to (possibly stochastic) actions to be taken when in those states.

2. *Reward Signal*: A reward signal $R \in \mathbb{R}$ defines the goal of a reinforcement learning problem. In each time step, the environment sends the agent a reward (usually a number) based on the current and subsequent states, and the actions taken.[29] The agent's sole objective is to maximize the total discounted reward it receives over the life time. The reward essentially drives the policy; if an action selected by the policy results in low expected reward, then the policy is modified to select some other action in that situation in future.

3. *Value*: Whereas the reward signal indicates the immediate payoff, a value function specifies the maximal total rewards an agent expects to accumulate in the long run, starting from the current state.[30] Whereas rewards determine the immediate, intrinsic desirability of environmental states, values reveal the long-term desirability of states after taking into account the states that are likely to follow and the rewards available in those states. For example, a state might always yield a low immediate reward but still have a high value because it is regularly followed by other states that yield high rewards.

[29] In a sense, this is similar to the state-dependent utility function in economics.

[30] This is similar to the multi-period utility function used in intertemporal choice models in economics, where future utility is subject to a discount compared with contemporaneous one.

4. *Model*: A model approximates an agent's dynamic interaction with the environment. Specifically, a model (discrete-time, just for illustration) specifies $P\left((s_{i+1}, R_{i+i})|s_i, a_i\right)$, where subscripts denote time.

Formally, a policy can be described as $\pi(a|s)$, which represents the probability of taking action $a \in A$ given state $s \in S$. The value function of a state s under a policy π, denoted $v_\pi(s)$, is the expected payoff when starting in s and following π thereafter. Denote γ as the discount rate for rewards, A_t for the action in period t, and $R_t \equiv R_t(s_{t-1}, s_t)$ for the reward from period $t-1$ to period t. For MDPs, we can write the value of state s under policy π, $v_\pi(s)$, by:

$$v_\pi(s) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^t R_{t+k+1}|S_t = s] = \mathbb{E}_\pi[G_t|S_t = s], \text{ for all } s \in S. \tag{9}$$

Uppercase S_t corresponds to a random variable, and lower case s_t indicates a realized value of S_t. Here $G_t = \sum_{k=0}^{\infty} \gamma^t R_{t+k+1}$ is the total discounted reward after period t. The agent's problem is $\max_\pi v_\pi(s)$

By iterated law of expectations:

$$\begin{aligned} v_\pi(s_t) =& \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s_t] \\ =& \mathbb{E}_\pi[R_{t+1} + \mathbb{E}_\pi[\sum_{k=1}^{\infty} \gamma^k R_{t+k+1}|S_t = s_t, S_{t+1} = s_{t+1}]|S_t = s_t] \\ =& \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s_t]. \end{aligned}$$

Note S_{t+1} is a random variable given $S_t = s_t$ when the model is stochastic. Therefore, the agent in RL simply solves a Bellman equation. More specifically, if the game is finite, i.e., there are some final states whose values are determined by the model, such as winning a Go game or successfully getting out of a maze. We can use these final states in a "backward induction" to get the value function in other states. In this sense, we have reduced the solution of a more complex problem (the value function of an "earlier" state which is far away from the end) to those of simpler problems (the value function of a "later" state which is closer to the end) with similar structure, which is exactly the thought behind finite horizon dynamic programming.

Similarly, we can define the value of choosing action a in state s under policy π, denoted by $q_\pi(s, a)$, by:

$$q_\pi(s, a) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^t R_{t+k+1}|S_t = s, A_t = a] = \mathbb{E}_\pi[G_t|S_t = s, A_t = a], \text{ for all } s \in S, a \in A(s), \tag{10}$$

where $A(s) \in A$ denotes all the possible actions under state s. It is obvious that $v(s) = \max_a q_\pi(s,a)$. And again,

$$\begin{aligned} q_\pi(s_t, a_t) =& \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s_t, A_t = a_t] \\ =& \mathbb{E}_\pi[R_{t+1} + \gamma \max_{a_{t+1}} q_\pi(S_{t+1}, a_{t+1}) | S_t = s_t, A_t = a_t], \end{aligned}$$

which is also in the form of Bellman equation. Solving this Bellman equation for $q(s,a)$ is the process called value-based learning, or Q-learning where we learn a value function that maps each state-action pair to a value. Q-learning works well when you have a finite (and small enough for computation) set of actions.

Another type of RL algorithm is policy-based learning, where we can deal with optimization over a continuum of possible policies. For instance, with a self-driving car, at each state you can have an infinite number of potential actions (turning the wheel at 15, 17.2, 19.4 degrees).[31] Outputting a Q-value for each possible action for example under Q-learning would be infeasible. In policy-based learning, we directly optimize the parameters in a policy function.

Specifically, we define our policy that has a parameter vector $\theta \in \mathbb{R}^d$, i.e.,

$$\pi_\theta(a|s) = \mathbb{P}\left[A_t = a | S_t = s, \theta\right].$$

Now our policy becomes parameterized. Similar to loss functions in machine learning, we can define a scalar performance measure $J(\theta)$ of the policy with respect to the policy parameter θ. These methods seek to *maximize* performance, so their updates approximate gradient *ascent* in J:

$$\theta_{t+1} = \theta_t + \alpha \nabla \widehat{J(\theta_t)}.$$

Now the question becomes how we estimate the gradient of J over θ? First, we need to introduce the concept of *trajectory*: $\tau = (s_1, a_1, s_2, a_2, \ldots, s_H, a_H)$, in which the agent starts at state s_1, chooses action a_1, and gets to state a_2, and so on. The total discounted reward, which is the natural target function of RL problem, is then

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{H} R(s_t, a_t)\right] = \sum_\tau \mathbb{P}_{\pi_\theta}(\tau) R(\tau). \tag{11}$$

Here $R(s_t, a_t)$ indicates that the reward in a period is a function of its current state s_t and action a_t. We then can randomly sample a number of trajectories based on current parameters θ, get an estimation of gradient using the samples and implement gradient ascent

[31] Example adopted from https://www.freecodecamp.org/news/an-introduction-to-policy-gradients-with-cartpole-and-doom-495b5ef2207f/

algorithms to maximize J.[32]

In our particular application of RL in portfolio construction, even though the high dimensionality can be handled by deep Q-learning, we have continuous action space and need to take the policy-based approach.

It should be clear that reinforcement learning is similar in spirit to multi-armed bandit, optimal experimentation, and dynamic programming. We next provide a simple application for illustration.

An application to cart-pole balancing. In this task, an inverted pendulum is mounted on a pivot point on a cart. The cart itself is restricted to linear movement, achieved by applying horizontal forces. Due to the system's inherent instability, continuous cart movement is needed to keep the pendulum upright. The observation consists of the cart position x, pole angle ω, the cart velocity $\dot{x}$, and the pole velocity $\dot{\omega}$. Therefore, the state is the four-element tuple, $s = (x, \omega, \dot{x}, \dot{\omega})$. The 1D action space consists of the horizontal force applied to the cart body. The reward function is given by

$$r(s, a) = 10 - (1 - \cos(\omega)) - 10^{-5}\|a\|_2^2,$$

which states the angle deviated from upright should be close to 90° in order to get high reward. In addition, the force a should not be too large in order to maintain the stability of the system. The game terminates when $|x| > 2.4$ or $|\omega| > 0.2$, i.e., the pole irreversibly falls down. (This comes from the survey study of Duan et al., 2016, `https://arxiv.org/pdf/1604.06778.pdf`, for complete physical/environmental parameters, see `https://github.com/rll/rllab`.)

We can use (artificial) neural networks (NN) to generate the i-th action a_i given state variables, where the parameters θ are the weights in neural network.

$$\pi_\theta(a_i|s) = \begin{cases} 1 & \text{if } a_i = f_\theta(s), \\ 0 & \text{else,} \end{cases}$$

where $f_\theta(\cdot)$ is the artificial neural network with weights θ. In Figure A.1, we have input $\boldsymbol{s} = (x_1, x_2, x_3, x_4)$. To transform the input $\boldsymbol{x}$ to a, the first step involves linear transformation:

$$\begin{aligned} u_1 &= \theta_{10} + \theta_{11}s_1 + \theta_{12}s_2 + \theta_{13}s_3 + \theta_{14}s_4, \\ u_2 &= \theta_{20} + \theta_{21}s_1 + \theta_{22}s_2 + \theta_{23}s_3 + \theta_{24}s_4, \end{aligned}$$

with all θ_{ij} as model parameters just like β in linear models. After that, we apply a nonlinear

[32]For the exact way to compute policy gradient, please refer to Sutton and Barto 2008, Ch 13. Or for a simpler version, see this excellent blog post.

function called activation function $g(\cdot)$ on (u_1, u_2).[33] We get:

$$y_1 = g(u_1), \qquad y_2 = g(u_2).$$

And we get the output $\boldsymbol{y} = (y_1, y_2)$. Then, the final action of $f_\theta(s)$ could be given as

$$a = g\left(\theta_{y0} + \theta_{y1} y_1 + \theta_{y2} y_2\right).$$

Note that the parameters θ_{ij} are the ones we use policy derivative algorithms to optimize in order to generate the optimal horizontal force and get the max reward.

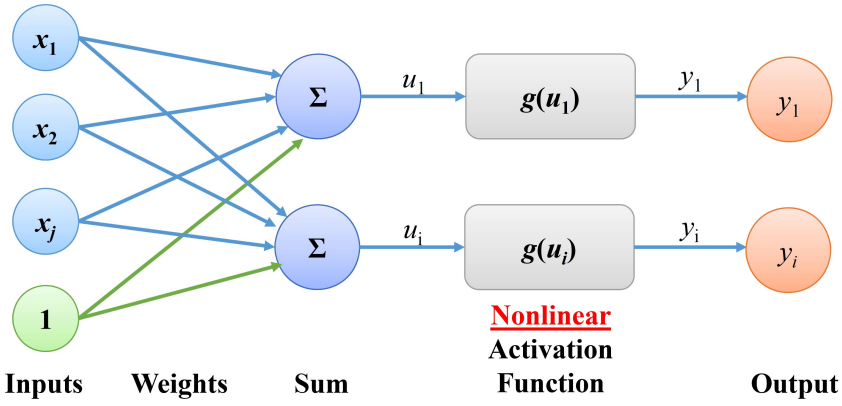


Figure A.1: A Single-Layer Artificial Neural Network

In general, after specifying the environment, reward, policy, and parameters, we can use policy derivative to approximate the optimal policy. The agent seeks θ that maximizes the reward function using gradient ascent on sampled state-action trajectories.

[33] Common activation functions include sigmoid, rectified linear unit (ReLU), hyperbolic tangent, etc.

Appendix B. Input Feature Construction

This section details the construction of the 51 variables we use as input features. We obtain the raw data from three WRDS databases: CRSP, CRSP Compustat Merged, and Financial Ratio Firm Level. Characteristics with highlights can be obtained from Financial Ratio Firm Level database.

A2ME: We define assets-to-market cap as total assets over market capitalization.

$$A2ME = \frac{AT}{(SHROUT * PRC)} \tag{12}$$

OA: We define operating accruals as change in non-cash working capital minus depreciation scaled by lagged total asset.

$$OA = \frac{\Delta\left((ACT + CHE - LCT - DLC - TXP) - DP\right)}{AT_{t-1}} \tag{13}$$

AOA: We define AOA as absolute value of operation accruals.

AT: Total asset.

BEME: Ratio of book value of equity to market equity.

Beta_daily: Beta_daily is the sum of the regression coefficients of daily excess returns on the market excess return and one lag of the market excess return.

C: Ratio of cash and short-term investments to total assets.

$$C = CHE/AT \tag{14}$$

C2D: Cash flow to price is the ratio of income and extraordinary items and depreciation and amortization to total liabilities.

$$C2D = (IB + DP)/LT \tag{15}$$

CTO: We define capital turnover as ratio of net sales to lagged total assets.

$$CTO = SALE/AT_{t-1} \tag{16}$$

Dept2P: Debt to price is the ratio of long-term debt and debt in current liability to the market capitalization.

$$Dept2P = \frac{(DLTT + DLC)}{(SHROUT * PRC)} \tag{17}$$

Δceq: The percentage change in the book value of equity.

$$\Delta ceq = (CEQ_t - CEQ_{t-1})/CEQ_{t-1} \tag{18}$$

Δ(ΔGm - ΔSales:) The difference in the percentage change in gross margin and the percentage change in sales.

$$\Delta(\Delta Gm - \Delta Sales) = \frac{SALE_t - COGS_t}{SALE_{t-1} - COGS_{t-1}} - \frac{SALE_t}{SALE_{t-1}} \tag{19}$$

ΔSo: Log change in the split adjusted shares outstanding.

$$\Delta So = \log\left(CSHO_t * AJEX_t\right) - \log\left(CSHO_{t-1} * AJEX_{t-1}\right) \tag{20}$$

Δshrout: Percentage change in shares outstanding.

$$\Delta shrout = \left(SHROUT_t - SHROUT_{t-1}\right)/SHROUT_{t-1} \tag{21}$$

ΔPI2A: The change in property, plants, and equipment over lagged total assets.

$$\Delta P12A = \frac{\Delta\left(PPENT + INVT\right)}{AT_{t-1}} \tag{22}$$

E2P: We define earnings to price as the ratio of income before extraordinary items to the market capitalization.

$$E2P = IB/\left(SHROUT * PRC\right) \tag{23}$$

EPS: We define earnings per share as the ratio of income before extraordinary items to shares outstanding.

$$EPS = IB/SHROUT \tag{24}$$

Free CF: Cash flow to book value of equity.

$$FreeCF = \frac{NI + DP + WCAPCH + CAPX}{BE} \tag{25}$$

Idol vol: Idiosyncratic volatility is the standard deviation of the residuals from a regress of excess returns on the Fama and French three-factor model.

Investment: We define investment as the percentage year-on-year growth rate in total assets.

$$Investment = \left(AT_t - AT_{t-1}\right)/AT_{t-1} \tag{26}$$

IPM: Pretax profit margin, $EBT/Revenue$.

IVC: We define IVC as change in inventories over the average total assets of t and $t-1$.

$$IVC = \frac{2 * \left(INVT_t - INVT_{t-1}\right)}{AT_t + AT_{t-1}} \tag{27}$$

Lev: Leverage is the ratio of long-term debt and debt in current liabilities to the sum of long-term debt, debt in current liabilities, and stockholder's equity.

$$Lev = \left(DLTT + DLC\right)/SEQ \tag{28}$$

LDP: We define the dividend-price ratio as annual dividends over price.

$$LDP = \frac{\sum\left(RET - RETX\right)}{PRC} \tag{29}$$

ME: Size is the market capitalization.

Turnover: Turnover is volume over shares outstanding.

$$Turnover = VOL/SHROUT \tag{30}$$

NOA: Net operating assets are the difference between operating assets minus operating liabilities scaled by lagged total assets.

$$NOA = \frac{left\,(AT - CHE - IVAO) - (AT - DLC - DLTT - MIB - PSTK - CEQ)}{AT_{t-1}} \tag{31}$$

NOP: Net payout ratio is common dividends plus purchase of common and preferred stock minus the sale of common and preferred stock over the market capitalization.

$$NOP = \frac{(DVC + PRSTKC - SSTK)}{ME} \tag{32}$$

O2P: Payout ratio is common dividends plus purchase of common and preferred stock minus the change in value of the net number of preferred stocks outstanding over the market capitalization.

$$O2P = \frac{DVC + PRSTKC - (PSTKRV_t - PSTKRV_{t-1})}{ME} \tag{33}$$

OL: Operating leverage is the sum of cost of goods sold and selling, general, and administrative expenses over total assets.

$$OL = (COGS + XSGA)/AT \tag{34}$$

PCM: The price-to-cost margin is the difference between net sales and costs of goods sold divided by net sales.

$$PCM = (SALE - COGS)/SALE \tag{35}$$

PM: The profit margin (operating income/sales).

Prof: We define profitability as gross profitability divided by the book value of equity.

$$Porf = GP/BE \tag{36}$$

Q: Tobin's Q is total assets plus the market value of equity minus cash and short-term investments minus deferred taxes scaled by total assets.

$$Q = \frac{(AT + ME/1000 - CEQ - TXDB)}{AT} \tag{37}$$

Ret: Return in the month.

Ret_max: Maximum daily return in the month.

RNA: The return on net operating assets.

ROA: Return-on-assets.

ROC: ROC is the ratio of market value of equity plus long-term debt minus total assets to

cash and short-term investments.

$$ROC = \frac{(DLTT + ME/1000 - AT)}{CHE} \tag{38}$$

ROE: Return-on-equity.

ROIC: Return on invested capital.

S2C: Sales-to-cash is the ratio of net sales to cash and short-term investments.

$$S2C = SALE/CHE \tag{39}$$

Sale_g: Sales growth is the percentage growth annual rate in annual sales.

$$Sale_g = SALE_t/SALE_{t-1} - 1 \tag{40}$$

SAT: We define asset turnover as the ratio of sales to total assets.

$$SAT = SALE/AT \tag{41}$$

S2P: Sale-to-price is the ratio of net sales to the market capitalization.

SGA2S: SG&A to sales is the ratio of selling, general and administrative expenses to net sales.

$$SGA2S = XGSA/SALE \tag{42}$$

Spread: The bid-ask spread is the average daily bid-ask spread in the month.

Std_turnover: Standard deviation of daily turnover in the month.

Std_vol: Standard deviation of daily trading volume in the month.

Tan: Tangibility.

$$Tan = \frac{0.715 * RECT + 0.547 * INVT + 0.535 * PPENT + CHE}{AT} \tag{43}$$

Total_vol: Standard deviation of daily return in the month.

Appendix C. TE-CAAN Implementation and a Bi-LSTM-HA Edition of AlphaPortfolio

C1. Multihead Attention and Implementation Details of TE-CAAN

To better understand the model and connect that to empirical studies, we should explain how AlphaPortfolio builds on existing Transformer models and details of our implementation. Readers familiar with Transformer models can safely skip this subsection.

First, it is useful to discuss the use of multi-head attention in existing Transformer models, which we inherit. Scaled dot-product attention in plain-vanilla Transformer models (shown in FigureC.1) constitutes a basic unit of multi-head attention. It replaces recurrence with self-attention. Unlike traditional attention methods, self-attention performs attention on a single sequence. The value of each position is calculated by all the positions in the sequence.

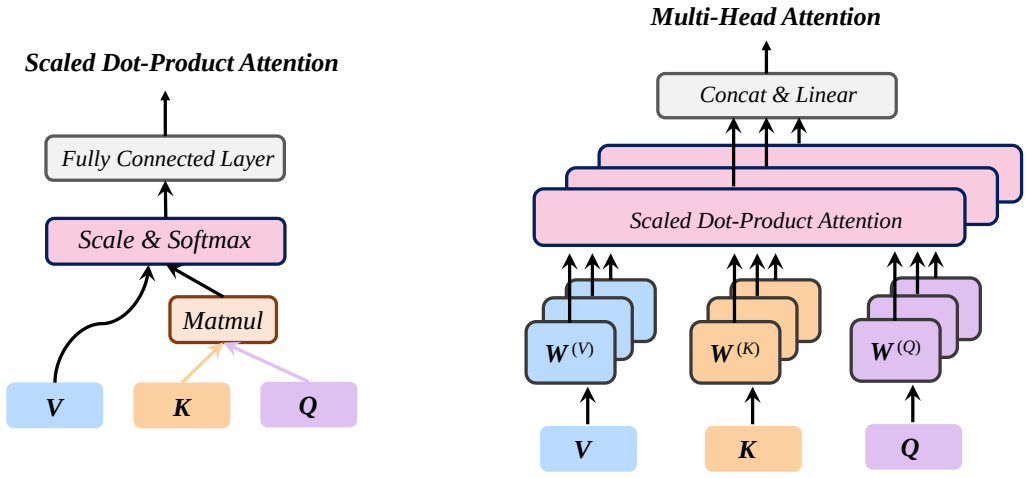


Figure C.1: Scaled Dot-Product Attention (left) and Multi-Head Attention (right).

The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. In practice, query, key and value matrices can be, respectively, packaged into Q, K and V. So we can compute the attention function on a set of queries simultaneously. The scale factor, $\frac{1}{\sqrt{d_k}}$, is to avoid the dot-product getting too large.[34]

$$Attention(Q, K, V) = softmax\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{44}$$

[34] Assume that the components of q and k are independent random variables with mean 0 and variance 1. Then their dot prodcut, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.

Multi-head attention (shown in Figure C.1) can be regarded as applying scaled dot-product attention in an h different feature space and finally concatenate the results.

$$MultiHead(Q, K, V) = Concat(head_1, \cdots, head_h)W^O \tag{45}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{46}$$

In addition to attention sublayers, each encoder contains a fully connected feed-forward network which is applied to each position separately and identically. In fact, we can consider this part as convolutions with kernel size 1. It consists of two linear transformations with a ReLU activation in-between.

It should be clear that our TE-CAAN retains features of the original design to the extent possible and utilizes residual connection and layer normalization. However, we isolate the encoder and differ in implementation details. Our model is based on PyTorch version 1.0.1 on four NVIDIA 1080Ti. To achieve better performance and take full advantage of computing resources, we adopt PyTorch's advanced API to automate data parallelism for the TE.

Because the amount of data in neural network translation tasks differ significantly from that in our task, we do not follow the original parameter setting with a stack of six encoder blocks. We instead find that one TE block achieves fast convergence and already produces exceptional results. Also, we reduce the dimension of embedding from 512 to 256 and reduce the dimension of feed forward from 2048 to 1024 for computational efficiency. The number of heads in multi-head attention is set to four. Once again, our innovation lies in the deep reinforcement learning approach for direct optimization, which drives the results, not these fine specifications of the TE model. Importantly, we add CAAN as which in itself is an innovation on top of TE.

Table C.1: Hyperparameters of TE-CAAN-Based AP

Hyper-parameter	Choice	Hyper-parameter	Choice
Embedding dimension	256	Optimizer	SGD
Feed-forward network	1021	Learning rate	0.0001
Number of multi-head	4	Dropout ratio	0.2
Number of TE layer	1	Training epochs	30

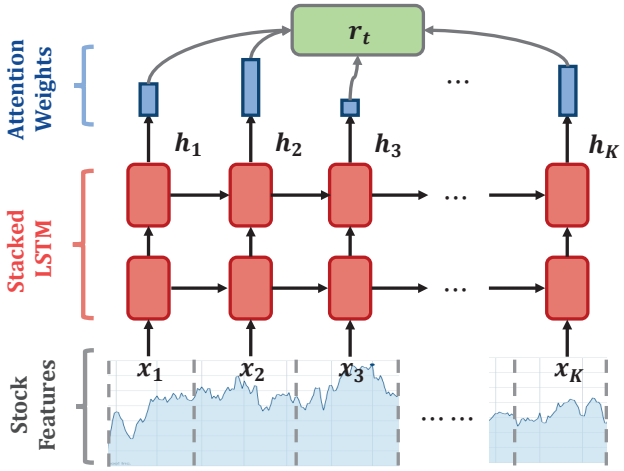


Figure C.2: The Architecture of LSTM-HA.

C2. AP based on Bi-directional LSTM with Historical Attention

For SREM, instead of TE, we could alternatively use a Long Short-Term Memory with Historical Attention (LSTM-HA) model to learn the representation from stock's historical features. As illustrated in Figure C.2, we start by utilizing an LSTM network to recursively encode $\boldsymbol{X}^{(i)}$ into a vector:

$$\boldsymbol{h}_k = \text{LSTM}\left(\boldsymbol{h}_{k-1}, \boldsymbol{x}_k^{(i)}\right), \ k \in [1, K], \tag{47}$$

where $\boldsymbol{h}_k$ is the hidden state encoded by LSTM at step k. The $\boldsymbol{h}_K$ at the last step is usually used as a representation of the sequence. It contains the sequential dependence among elements in $\boldsymbol{X}^{(i)}$.

While $\boldsymbol{h}_K$ can fully exploit the sequential dependence of elements in $\boldsymbol{X}^{(i)}$, the global and long-range dependence are not effectively modeled. Therefore, we adopt a historical state attention mechanism to enhance $\boldsymbol{h}_K$ using all middle states $\boldsymbol{h}_k$. Specifically, following the standard attention architecture (Sutskever, Vinyals, and Le, 2014), the historical state attention enhanced representation denoted by $\boldsymbol{r}^{(i)}$, is calculated as

$$\boldsymbol{r}^{(i)} = \sum_{k=1}^{K} \text{ATT}\left(\boldsymbol{h}_K, \boldsymbol{h}_k\right) \boldsymbol{h}_k, \tag{48}$$

where $\text{ATT}(\cdot,\cdot)$ is an attention function defined as

$$\begin{aligned} \text{ATT}\left(\boldsymbol{h}_K, \boldsymbol{h}_k\right) &= \frac{\exp\left(\alpha_k\right)}{\sum_{k'=1}^{K}\exp\left(\alpha_{k'}\right)}, \\ \alpha_k &= \boldsymbol{w}^\top \cdot \tanh\left(\boldsymbol{W}^{(1)}\boldsymbol{h}_k + \boldsymbol{W}^{(2)}\boldsymbol{h}_K\right). \end{aligned} \tag{49}$$

Here, $\boldsymbol{w}$, $\boldsymbol{W}^{(1)}$, and $\boldsymbol{W}^{(2)}$ are the parameters to learn.

For an LSTM-implementation of our AP, we replace TE with Bi-directional LSTM with attention mechanism. Table C.2 reports the results. In terms of OOS metrics such as the Sharpe ratio, the Bi-LSTM-CAAN model even outperforms our original AP.

Table C.2: Out-of-Sample Performance of Bi-LSTM-CAAN-Based AP

This table presents the OOS performance for Bi-LSTM-CAAN-based AP. For each month in the OOS periods (1990-2016), AP constructs a long-short portfolio, which goes long the 10% of stocks with the highest winner scores and shorts the 10% of stocks with the lowest winner scores. The investment proportions are calculated according to Section 3.3. Return, Std.Dev., and Sharpe ratio are all annualized.

	(1)	(2)	(3)
Firms	All	size $> q_{10}$	size $> q_{20}$
Return(%)	16.90	15.48	15.07
Std.Dev.(%)	7.70	5.06	5.08
Sharpe	2.19	3.06	2.97
Skewness	1.63	0.86	1.06
Kurtosis	6.85	2.43	4.88
Turnover	0.46	0.39	0.40
MDD	0.03	0.01	0.02

However, our economic distillation reveals that LSTM does not have stable utilization of input features or economic interpretability using textual factors. This is consistent with that LSTM-HA deals with vanishing and exploding gradients only in the training sample and with that AI may face instability of performance (Heaven, 2019).

Specifically, from Table C.3, Bi-LSTM-CAAN-based AP tend to select characteristics of the first/last position ("_0" and "_11") in the input sequence, a pattern robust when we change the number of months to generate lagged inputs. This is indicative of exploding gradient issues, which means the trained model go to some extremes or gradient-based interpretation methods are not that suitable for such RNN-like models in our case.

While gradient exploding problem can be solved by gradient clip during training (back-

propagation), when we have a well-trained model and use it to test OOS, researchers typically do not know whether in the test sample there is a problem of gradient explosion. In other words, the detection of gradient explosions in computer science focuses on the training stage and would not flag such technical issues of a model from test samples. Our economic distillation therefore helps to detect modeling issues out-of-sample that the traditional CS approach neglects.

Table C.3: Fama-Macbetch T-test Values of Selected Terms (Bi-LSTM-Based AP)

This table presents the results of using Fama-Macbetch method to interpret Algorithm 1. Polynomial degree is set as one and for all terms with suffixed like "_*no*", *no* indicates the sequence number of input features, *i.e., pe_*7 denotes P/E ratio at the time of five $(12 - 7 = 5)$ months ago. For details of each characteristic, please refer to Appendix B. q indicates the size percentile of *NYSE* firms. In this table, we present the top 50 significant terms.

All		size $> q_{10}$		size $> q_{20}$	
pe_0	85.04	tan_11	-83.87	tan_11	-84.56
ivc_11	-82.17	ivc_11	-78.49	ivc_11	-81.81
C_11	78.34	pe_0	68.77	pe_0	68.10
Q_11	-70.57	C_11	68.14	C_11	62.64
tan_11	-65.63	Q_11	-52.07	Q_11	-50.44
ivc_0	-58.49	Idol_vol_0	50.99	e2p_11	50.42
Idol_vol_0	51.74	ivc_0	-50.08	Idol_vol_0	48.96
Turnover_0	-44.84	Turnover_0	-48.06	ret_11	-46.00
delta_so_11	-43.99	e2p_11	45.04	Turnover_0	-45.15
Idol_vol_11	38.00	ret_11	-44.48	ivc_0	-44.60
Turnover_11	-34.92	delta_so_11	-39.49	delta_so_11	-38.08
Ret_max_11	-33.85	Beta_daily_0	-37.94	Turnover_11	-37.19
s2p_11	33.25	Turnover_11	-36.8	Beta_daily_0	-35.24
Beta_daily_0	-31.15	beme_11	31.72	beme_11	28.06
delta_shrout_11	26.15	Idol_vol_11	27.07	s2p_11	26.95
s2p_0	23.97	s2p_11	25.91	investment_11	-26.54
ret_11	-23.82	cto_0	-24.41	Idol_vol_11	24.57
beme_11	23.29	Std_volume_0	-20.67	oa_11	24.23
oa_11	23.03	investment_11	-20.65	cto_0	-22.75
roa_11	-22.62	Beta_daily_11	20.31	Beta_daily_11	22.07
roa_0	-22.36	sat_11	20.29	Std_volume_0	-20.98
investment_11	-19.87	oa_11	20.17	sat_11	20.78
Std_volume_0	-19.86	s2p_0	20.03	s2p_0	20.68
pe_11	-18.91	delta_shrout_11	19.97	delta_shrout_11	19.08
sat_11	18.49	roa_11	-19.52	noa_11	19.05
at_11	17.02	pe_11	-17.76	shrout_11	17.93
cto_0	-16.31	shrout_11	17.58	roic_0	17.45
c2d_11	16.28	Ret_max_11	-16.09	me_11	16.33
shrout_11	15.09	sat_0	16.05	nop_0	-15.85
beme_0	13.93	me_11	15.49	roa_11	-14.59
investment_0	-13.84	nop_0	-15.46	pe_11	-14.16
sga2s_11	13.61	c2d_11	14.39	c2d_11	13.81
roic_11	13.42	roic_0	13.9	sat_0	13.39
me_11	13.20	noa_11	12.89	Ret_max_11	-13.31
aoa_11	-12.21	delta_pi2a_11	12.86	delta_so_0	-12.92
delta_pi2a_0	12.09	sga2s_11	12.45	sga2s_11	12.38
Beta_daily_11	12.07	delta_so_0	-11.76	vol_11	-11.13
roic_0	12.07	sale_g_0	11.62	a2me_11	-10.69
shrout_0	11.07	roic_11	11.23	sale_g_0	10.66
delta_pi2a_11	10.89	at_11	11.12	roic_11	10.62
sat_0	10.52	a2me_11	-10.35	investment_0	-10.38
e2p_11	10.21	beme_0	10.29	at_11	9.91
delta_so_0	-9.56	C_0	10.14	nop_11	9.41
C_0	9.03	nop_11	10.02	shrout_0	9.29
sale_g_0	8.93	vol_11	-9.83	aoa_11	-9.03
sale_g_11	8.53	shrout_0	8.82	beme_0	8.65
std_0	-8.23	sale_g_11	8.09	C_0	8.40
a2me_11	-8.07	investment_0	-7.53	std_11	-6.95
Spread_11	-6.79	std_11	-6.9	free_cf_11	6.84

Appendix D. Economic Distillation via Textual Factor Analysis

The concept of projecting a complex model onto simpler spaces can be extended to natural languages too. The main idea behind a textual factor analysis is that texts are written in natural languages and if we find correlations of AP holdings with the topics discussed in text documents, we may be able to develop a narrative or a better understanding of the model.

To do this, we use the general framework Cong, Liang, and Zhang (2018) introduce for analyzing large-scale text-based data. The methodology combines the strengths of neural network language models and generative statistical modeling. It generates textual factors by (i) representing texts using vector word embedding, (ii) clustering words using locality-sensitive hashing, and (iii) identifying spanning vector clusters through topic modeling. Arguably, one can use other text analytics but the data-driven approach in Cong, Liang, and Zhang (2018) captures complex linguistic structures while ensuring computational scalability and economic interpretability.[35]

Specifically, we use Google's word2vec for the word embedding step and follow Cong, Liang, and Zhang (2018) to generate textual factors from firm-level documents. They are essentially topics and themes with relative frequency on a set of words and phrases that span the textual space. We then regress each firm's document on the textual factors to obtain the loadings on each topic.

We obtain text data from Company Filings at SEC Edgar (`https://www.sec.gov/edgar/`). The U.S. Securities and Exchange Commission (SEC) approved a rule requiring publicly-listed firms to file their securities documents via the Electronic Data Gathering, Analysis and Retrieval (EDGAR) system since 1993. We illustrate the approach with Management Discussion and Analysis (MD&A) sections of both the quarterly report (10-Q) and the annual report (10-K). Other forms of text data we can utilize are Risk Factor Discussion in 10-K reports and analyst reports.

[35]The difficulties in analyzing textual data are three-fold: first, language structures are intricate and complex, and representing or summarizing them using simple frequency/count-based approaches is highly reductive and may lose important informational content; second, textual data are high-dimensional and processing a large corpus of documents is computationally demanding; third, there lacks a framework relating textual data to sparse regression analysis traditionally used in social sciences while maintaining interpretability. In the paper, the authors also discuss applications of textual factors in (i) prediction and inference, (ii) interpreting existing models and variables, and (iii) constructing new metrics and explanatory variables, with illustrations using topics in economics such as macroeconomic forecasting and factor asset pricing.

Mathematically, let K denote the number of textual factors, where K is endogenously specified and can potentially depend on the data (we use 200 for simplicity). Denote the set of textual factors by the triplet $(S_i, F_i \in \mathbb{R}^{|S_i|}, d_i \in \mathbb{R}_{\geq 0})$, where S_i denotes the support of word-cluster $i = 1, \cdots, K$, F_i is a real-valued vector representing the textual factor indicating the relative frequencies of words of factor, and the factor importance d_i. Given the textual factors and a firm's document D (represented by a document-term vector $N^{(D)} \in \mathbb{R}^V$, where V is the size of the vocabulary in the texts), the loading of the textual factor i is simply the projection

$$x_i^{(D)} := \frac{\langle N_{S_i}^{(D)}, F_i \rangle}{\langle F_i, F_i \rangle}, \tag{50}$$

and the document D can be represented quantitatively as $(x_1^{(D)}, \ldots x_K^{(D)}) \in \mathbb{R}^K$.

To understand the meaning of these loadings (textual βs), think about how a company continuously discusses and discloses information on profitability, social responsibility, innovativeness, etc., through MD&A. The $x_k^{(D)}$ we obtain allows us to assign a number that measures how much the company loads on that topic—a metric we can use in simple sparse regression framework.

The final step is to regress each stock's winner score in each month from AP onto the contemporaneous textual βs. We iterate the process a few times to reduce the number of textual factors based on their interpretability, importance in the MD&A data, and significance in the AP construction. Specifically, after each iteration, we discard word clusters that are in coherent or are infrequently mentioned in the firms' filing or have insignificant correlations with the winner score. Table D.1 contains examples of the most loaded topics/textual factors when constructing AP. A positive coefficient indicates when discussions on the topic dominate the firm's text data, the firm's stock more likely receives a long position; a negative coefficient indicates the opposite. The word lists are the corresponding words within each textual factor. The stocks AP buys typically mention issues such as loss-cutting, sales, and actions as well as profitability, cash, and investment that are related to C, C^2, investment, ipm, and ipm^2 from the polynomial sensitivity analysis; the stocks AP short-sells are the ones heavily discussing real estates, corporate events, and acknowledging mistakes as well as uncertainty and inventory, which relate to C^2, delta_so, ivc, ivc^2, Idol_vol, and Idol_vol^2. Further analysis can be conducted. For example, one can relate the negative loading on corporate events textual factor to theories explaining why stock returns may be negative on average for firms going through certain corporate events.

Table D.1: Winner Scores' Loadings on Textual Factors

This table contains examples of the most loaded topics/textual factors when firms' winner scores are regressed on over 30 contemporaneous textual factor loadings selected based on factor importance and domain expertise. The regression coefficients are reported under the textual topics, where a positive (negative) number indicates a long (short) position on stocks with the topic prominently showing up in the firm's filings. The remainder columns list words within each textual factor. We do not stem words because we use word vectors trained directly by Google word2vec in the word embedding step.

Topics	**Most Frequent Words in the Textual Factor**					
Loss-cutting	deferral	curtailment	divestiture	diminution	cutback
(0.1500)	shutdown	abolishment	retrenchment	imposition	reductions
	decrease	reclassification	amortization	resell	resale
	recycled	afford	salable		
Profitability	profit	ebit	quarterly	earnings	profitably
(0.1358)	profits	pretax	revenue	revenues	viable
	profitable	writedown	business	unprofitable	net_loss
	yoy	gross_margin	net_income	efficiency	operational
Sales	sales	purchases	growth	retail	earnings
(0.0428)	profit	income	profits	comps	resales
	shipments	sale	fiscal	revenue	
Cash/Invest	subsidizing	unaffordable	reimburse	reinvestment	subsidy
(0.0381)	invest	paying	afford	cash_trapped	tariffs
Actions	secures	closing	unveils	receives	exploring
(0.0339)	acquire	introduces	signs_agreement	deploys	stopped
	completion	donates	acquires	announces	delayed
Inventory	hoarding	replenishing	stockpiling	overcharging	supplying
(-0.0209)	accumulating	distributing	producing	restock	stockpile
	rationing	buying	storing	restocking	inventory
	transfers	seasonal	restocked	shipping	stocked
Mistakes	forgiveness	confess	forgives	admit	forgiving
(-0.0536)	contrition	forgiven	wrongs	excuses	atonement
	apologize	punish	repay	mistake	amends
	clarifications	sorry	forgave	repent	redress
Uncertainty	volatility	speculator	irrationals	traders	fluctuation
(-0.1411)	speculative	risky	turmoil	instability	uncertainty
	turbulance	changing	evolving	unpredictable	hedge
Corporate	recapitalization	divestitures	mergers	divestiture	unbundling
Events	buyout	acquire	transaction	acquisitive	restructure
(-0.1444)	acquisitions	acquired	divestiture	merger	amalgamated
	takeovers	synergies	expansions	takeover	takeover
Real Estate	bungalows	dwellings	acres	houses	carport
(-0.2362)	residences	cottages	barns	cabins	outbuilding
	homes	buildings	condos	lofts	mansion
	condominiums	townhome	real_estate	foreclosure	backyard
	apartment	farmhouse	cottage	bedroom	villa
	rented	residence	duplex	ranch	motorhome

This textual factor analysis only constitutes an initial step towards developing a narrative for how AP behaves. Our simple choice of text data and the application of textual factors are not meant to be optimal and definitive but serve as an illustration of interpreting AI models with texts. More comprehensive analysis in combination with economic theory constitutes future research. For parsimony, we have only included representative textual factors. More implementation details can be found in Cong, Liang, and Zhang (2018).