NBER WORKING PAPER SERIES

ARTIFICIAL INTELLIGENCE ASSET PRICING MODELS

Bryan T. Kelly
Boris Kuznetsov
Semyon Malamud
Teng Andrea Xu

Working Paper 33351
http://www.nber.org/papers/w33351

NATIONAL BUREAU OF ECONOMIC RESEARCH
1050 Massachusetts Avenue
Cambridge, MA 02138
January 2025, Revised June 2026

Boris Kuznetsov is at the Swiss Finance Institute, EPFL. Semyon Malamud is at the Swiss Finance Institute, EPFL, CEPR, and is a consultant to AQR. Teng Andrea Xu is at EPFL. Semyon Malamud gratefully acknowledges the financial support of the Swiss Finance Institute and the Swiss National Science Foundation, Grant 100018-228042. AQR Capital Management is a global investment management firm that may or may not apply similar investment techniques or methods of analysis as described herein. The views expressed here are those of the authors and not necessarily those of AQR. This work was supported by a Swiss National Supercomputing Centre (CSCS) grant under project ID sm86. The views expressed herein are those of the authors and do not necessarily reflect the views of the National Bureau of Economic Research.

Artificial Intelligence Asset Pricing Models
Bryan T. Kelly, Boris Kuznetsov, Semyon Malamud, and Teng Andrea Xu
NBER Working Paper No. 33351
January 2025, Revised June 2026
JEL No. C45, G10, G11, G14, G17

ABSTRACT

The core statistical technology in artificial intelligence is the large-scale transformer network. We propose a new asset pricing model that implants a transformer in the stochastic discount factor. This structure leverages conditional pricing information via cross-asset information sharing and nonlinearity. We also develop a linear transformer that serves as a simplified surrogate from which we derive an intuitive decomposition of the transformer's asset pricing mechanisms. We find large reductions in pricing errors from our artificial intelligence pricing model (AIPM) relative to previous machine learning models and dissect the sources of these gains.

Bryan T. Kelly
Yale University
and NBER
bryan.kelly@yale.edu

Boris Kuznetsov
École Polytechnique Fédérale
de Lausanne (EPFL)
boris.kuznetsov@epfl.ch

Semyon Malamud
Swiss Finance Institute
semyon.malamud@epfl.ch

Teng Andrea Xu
AQR Capital Management
andrea.xu@aqr.com

1 Introduction

The AI revolution emerged from two complementary large language model (LLM) insights.[1] First, words need to be understood in context. Language models that study words without awareness of the surrounding context cannot faithfully represent a text's meaning and therefore suffer in tasks such as word prediction and translation. The second insight is that LLM performance scales with the number of parameters.[2] The so-called transformer architecture is a watershed discovery that makes it possible to model words in context, support massive parameterization, and maintain computational tractability.

The finance literature has focused primarily on small-scale "own-asset prediction" models—those in which forecasts for asset i depend on conditioning variables that are specific to asset i. Own-asset prediction models abstract from context by failing to use information about the broader universe of assets to describe the risk and return of each individual asset. The recent AI experience suggests these design choices—small parameterizations focused on own-asset prediction—may severely limit the performance of asset pricing models.

This paper introduces the concept of an artificial intelligence pricing model (AIPM). We define an AIPM as a model that embeds a context-aware, large-scale transformer architecture used in state-of-the-art LLMs within the stochastic discount factor (SDF). The critical attributes of an AIPM are *i)* flexible sharing of conditioning information across assets to produce context-aware forecasts, and *ii)* the ability to improve out-of-sample performance through extreme model parameterization. Thus, our first main contribution is an asset pricing model design that leverages the two aforementioned attributes of leading AI models.

The transformer can seem analytically impenetrable. The second contribution of our paper is to derive an intuitive characterization of the transformer's role in the context of asset pricing. To do so, we begin with a simple and interpretable variant of our model that we refer to as the "linear portfolio transformer." This SDF model is easy to describe

[1] See, e.g., Minaee et al. (2024) for an extensive LLM review.

[2] This surprising empirical phenomenon has prompted a search for theoretical foundations. See, e.g., Kaplan et al. (2020), Spigler et al. (2019), Belkin et al. (2019a), Belkin et al. (2019b), Belkin et al. (2020), and Bartlett et al. (2020), as well as Kelly et al. (2024), Didisheim et al. (2024), and Timmermann and Vulicevic (2026) for extensions of these works to the financial domain.

and inspect, and it admits a closed-form regression-based estimator. It uses the same core information sharing apparatus—the "attention" mechanism—as a standard transformer, but removes the nonlinearities present in the fully nonlinear transformer. Most importantly, the analytical tractability of the linear portfolio transformer makes it a powerful surrogate for understanding the impact of cross-asset attention on the SDF portfolio. It also has a number of other surprising model properties that we derive theoretically, including an ability to vary model parameterization without changing the data inputs or deviating from linearity. This property, which we refer to as "linear multi-head attention," is convenient for inspecting the effect of increasing the number of parameters in the linear specification.

Our analytical derivations provide interpretations of the attention-based SDF. The first is related to the representation of a conditional SDF as a linear combination of characteristic-managed "factor" portfolios (e.g. Kozak et al., 2020). In traditional SDF models, the weight that stock i receives in the SDF portfolio ($w_{i,t}$) depends on i's characteristics ($x_{i,t}$). A particularly clear example of this is the Brandt et al. (2009) model: $w_{i,t} = x_{i,t}\lambda$, where λ is the vector of model parameters to be estimated. Our linear portfolio transformer replaces this specification with $w_{i,t} = (\sum_j a_{i,j}x_{j,t})\lambda$. In other words, the optimal weight for each asset i depends not only on its own characteristics, but also on the characteristics of other stocks. How heavily stock i is influenced by stock j is governed by the additional estimable parameter $a_{i,j}$.[3] This cross-sectional information sharing is our asset pricing model's analogue to context awareness for words in an LLM.

The linear portfolio transformer also illustrates that AIPMs can be interpreted as performing a sophisticated version of factor timing. Factor timing models have been proposed by Gupta and Kelly (2019), Haddad et al. (2020), and Ehsani and Linnainmaa (2022), among others. Relatedly, the principal portfolios method of Kelly et al. (2023) explicitly links factor timing to the problem of cross-asset return prediction. These papers consider one factor at a time and are developed in low-parameter settings. In contrast, our portfolio transformers are high-dimensional timing models that optimize the timing of all factors jointly.

The linear portfolio transformer is an intermediate modeling step. While it is a valuable

[3]In addition, the full formulation of our model allows the parameter $a_{i,j}$ to be time varying and depend on observable data.

device for understanding the role of context-awareness in an AIPM, it abstracts from a variety of other model features known to be critical to the success of modern AI models. Although the linear transformer has high parameterization by typical asset pricing standards, there remains significant scope for expanding model parameterization further. We therefore introduce an AIPM that uses a large nonlinear transformer architecture inspired by Vaswani et al. (2017). It includes multi-head attention, softmax transformations, a feed-forward network, residual connections, and, most importantly, the deep learning benefit of stacking multiple transformer blocks together. We discuss the intuitive role of each of these AIPM model components.

Our third main contribution is to document and dissect the empirical performance of AIPMs. We focus on monthly data for US stock returns and use a standard set of 132 stock-level conditioning characteristics from Jensen et al. (2023) (JKP henceforth). As appropriate for machine learning model comparisons, we focus on out-of-sample Sharpe ratios and pricing errors for anomaly portfolios following the formulation of Didisheim et al. (2024) (DKKM henceforth).

We begin with an empirical analysis of the linear portfolio transformer. We document the benefits of cross-asset information sharing by comparing it to a linear SDF in which the attention mechanism is shut down. The benchmark establishes that a standard linear SDF with no cross-asset information achieves an out-of-sample Sharpe ratio of 3.6. The linear portfolio transformer, which introduces the attention mechanism with no other model changes (i.e., maintaining linearity and the same data inputs), raises the Sharpe ratio to 3.9 and shows an alpha t-statistic of 6.8 relative to the attention-free linear model, suggesting that the benefits of cross-asset information sharing are non-trivial even in a simple, linear setting. Next, we use the multi-head capability of the linear portfolio transformer to investigate the benefits of large-scale parameterizations. We find that the out-of-sample Sharpe ratio increases with the number of model parameters, which is consistent with DKKM.

Finally, we analyze the empirical properties of the nonlinear portfolio transformer. To evaluate the role of information sharing in the nonlinear setting, we require a benchmark model that incorporates nonlinearities but does so solely through its use of own-asset information. The recently proposed SDF of DKKM, which is nonlinear but lacks an attention

mechanism, is well-suited for this comparison. In our analysis, DKKM achieves an out-of-sample Sharpe ratio of 3.9. In contrast, through its information-sharing capacity, the nonlinear portfolio transformer achieves a large and significant improvement, reaching a Sharpe ratio of 4.6. We also design an otherwise "apples-to-apples" feed-forward network with the same architecture as the nonlinear portfolio transformer but excluding the attention mechanism to shut down cross-asset information sharing. In this case, we find the same qualitative result that we found when comparing to DKKM. In other words, the excess performance of the transformer is indeed attributable to the information-sharing mechanism.

Next, we investigate how nonlinear transformer performance responds as we scale up parameterization by concatenating more transformer blocks to increase network depth. The Sharpe ratio increases from 3.8 for a single-block transformer (a model with about 100,000 parameters) to 4.6 for a ten-block transformer (a roughly one-million-parameter model). Our analysis suggests that our nonlinear portfolio transformer can be improved further by increasing parameterization, as the slope of out-of-sample performance with respect to the number of blocks has not yet flattened (though due to the computational costs for model training, we leave further parameter expansion to subsequent work). Taken together, our findings suggest a clear hierarchy among models—performance improves progressively as model specifications become richer. Models with the attention mechanism for cross-asset information sharing dominate those without.

Finally, we investigate the economic nature of the cross-asset information sharing learned by the transformer. To make progress on this point, we compare the transformer model to an "information sharing" SDF that relies on observable economic linkages between firms such as customer-supplier relationships, analyst co-coverage, and several others. We find that observable firm linkages and statistically learned attention reflect complementary and mutually beneficial forms of cross-asset information sharing. Observable linkages carry economic information and structure that is not easy for the transformer to learn from returns and stock characteristics alone, while the transformer can detect information-sharing benefits from subtle patterns in return and characteristic data. In summary, the evidence from observable linkages supports our broader conclusion that cross-asset information sharing is important for constructing a better SDF.

This paper contributes to the emerging literature demonstrating that machine learning asset pricing models achieve higher out-of-sample Sharpe ratios and smaller pricing errors than their more parsimonious predecessors. Examples include Gu et al. (2020), Chen et al. (2023), Freyberger et al. (2020), Bryzgalova et al. (2025), Cong et al. (2025), and Fan et al. (2022), among others. Recent work develops the theoretical rationale for performance improvements deriving from richer and more flexible model specifications (Kelly et al., 2024; Didisheim et al., 2024). Our work extends this literature by introducing transformer architectures into asset pricing models, leveraging cross-asset information sharing, and embracing model complexity to further capture predictive phenomena in financial markets.

In relation to machine learning methods for analyzing factor models, our approach aligns with efforts by Connor et al. (2012), Fan et al. (2016), Kelly et al. (2020a), Lettau and Pelger (2020), Kozak et al. (2020), Giglio and Xiu (2021), Giglio et al. (2022), He et al. (2023), and Preite et al. (2022), all of which employ advanced statistical techniques to improve factor model estimation and prediction (see Kelly and Xiu, 2023; Rapach and Zhou, 2020, for comprehensive surveys on these topics). By embedding transformer architectures within the SDF, we extend the boundaries of the factor modeling literature.

By utilizing transformers and their attention mechanisms, we provide evidence that the factor zoo (Harvey et al., 2016; McLean and Pontiff, 2016; Hou et al., 2020; Feng et al., 2020; Jensen et al., 2023; Chen and Zimmermann, 2022) is important for capturing subtle dimensions of cross-asset information flows. Effectively, transformers embed stocks into the space of high-dimensional characteristics and then use this embedding to learn characteristics-based networks for information transmission. In related work, Cong et al. (2021) use a transformer for portfolio choice, and Guijarro-Ordonez et al. (2025) use a transformer in their daily statistical arbitrage model.[4] We build on this literature by emphasizing the benefits of a transformer-based SDF for pricing the cross-section of test asset returns and by exploring the role of cross-asset information sharing achieved with transformers.

The paper proceeds as follows. In Section 2, we introduce the linear portfolio transformer that includes cross-asset attention in an otherwise vanilla conditional SDF, derive its theo-

[4] Gabaix et al. (2025) estimate a transformer model that embeds holdings data in a low-dimensional space that captures demand-related asset and investor characteristics.

retical properties, and present a closed-form regression-based estimator. Section 3 introduces a large-scale AIPM that embeds a deep nonlinear transformer in the SDF. Lastly, in Section 4, we evaluate the empirical properties of these models using monthly US stock data.

2 The Linear Portfolio Transformer

In this section, we introduce the linear portfolio transformer. The linear model is estimable in closed form, providing a particularly transparent environment for dissecting its functionality in intuitive terms. Most conceptual insights drawn from our linear transformer carry over to its nonlinear counterpart in Section 3, which uses similar model components and is based on popular LLM transformers.

2.1 Model Specification

Our setting consists of a universe of N_t risky assets in the economy at time t whose excess returns are given by the vector R_{t+1}. Financial machine learning strives to incorporate large conditioning sets in richly parameterized models with the aim of accurately characterizing the expected out-of-sample behavior of asset prices. To this end, each asset is accompanied by a high-dimensional vector of D characteristics $X_{i,t}$. We use X_t to denote the $N_t \times D$ matrix stacking characteristics for all stocks at time t. We assume that the conditioning variables in X_t span the time t information set. Our framework accommodates a time-varying universe of N_t assets by conditioning assets' behavior on a fixed number D of asset-level characteristics, as in Kelly et al. (2020b).

Kelly and Xiu (2023) discuss the conveniences of representing machine learning asset pricing models in terms of their SDF or, equivalently, in terms of the maximum Sharpe ratio portfolio of risky assets. We follow this approach in defining an AIPM whose conditional SDF representation (Hansen and Richard, 1987) is

$$M_{t+1} = 1 - w(X_t)' R_{t+1}\,, \tag{1}$$

where the vector function $w(X_t)$ maps conditioning information into the SDF's conditional weights on individual risky assets, and $w(X_t)'R_{t+1}$ is the conditionally mean-variance optimal portfolio.

The basic linear portfolio transformer model is:

$$w_t = A_t X_t \lambda = (X_t W X_t') X_t \lambda \,, \tag{2}$$

where W $(D \times D)$ and λ $(D \times 1)$ are the model parameters to be estimated. We refer to the matrix $A_t = X_t W X_t'$ as the "attention" matrix. The following subsections describe and interpret the model's functionality.

2.2 Transformers and Cross-asset Information Sharing

At the core of the portfolio transformer architecture is the so-called attention mechanism, introduced to the machine learning literature in various forms by Vaswani et al. (2017), Cho et al. (2014), and Bahdanau (2014), and most prominently known for its success in large language models like ChatGPT. In the AIPMs we develop, the attention mechanism provides a flexible and tractable framework for introducing cross-asset information sharing.

The literature on financial machine learning has focused primarily on own-asset prediction in which the optimal portfolio weight of asset i depends solely on conditioning variables that are specific to asset i. In an early formulation of this approach, Brandt et al. (2009) specify the optimal portfolio weight for asset i as a linear function of its characteristics, $w_{i,t} = X_{i,t}'\lambda$. This is a special case of (2) in which the matrix A_t is the identity matrix:

$$w_t = X_t \lambda. \tag{3}$$

Likewise, the high-complexity analysis of DKKM centers on the formulation $w_{i,t} = f(X_{i,t})\lambda$, where f is a heavily parameterized neural network. All assets share the same network architecture and model parameters, but each asset's portfolio weight continues to depend, ultimately, only on its own characteristics.

As argued in the introduction, own-asset predictability is a potentially costly restriction

in empirical asset pricing models. However, loosening this restriction can be a difficult task. A simple formulation of cross-asset predictability would be a static version of (2) with $A_t = A$:[5]

$$w_t = AX_t\lambda. \tag{4}$$

The d^{th} characteristic of stock i is thus modified from $X_{i,d,t}$ to $\tilde{X}_{i,d,t}$ by multiplying the d^{th} column of X_t with the i^{th} row of A. In other words, a cross-prediction formulation like (4) uses information sharing to enhance an own-asset predictive characteristic with a weighted average of that same characteristic's value across all assets. Matrix A summarizes the "attention" one should pay to characteristics of all other assets when deciding on the ideal portfolio weight for asset i. The optimal attention matrix A can be learned as part of the training objective (9) along with the weights λ that optimally map the "cross-sectionally smoothed" characteristics into final portfolio weights.

However, the formulation of (4) is unattractive for a few reasons. For one, a static A matrix cannot handle time-varying universes of N_t assets, as is necessary, for example, in the cross-section of individual stocks. More importantly, it ignores potentially valuable information about the similarity among stocks in the conditioning set X_t. We can remedy these shortcomings with a dynamic attention matrix A_t that exploits conditional information sharing. But the question then is how to operationalize conditional attention. Fortunately, the attention mechanism within a transformer is well-suited for this problem. It parameterizes A_t as a function of observables in X_t. In particular, the conditional attention mechanism in (2) shares information based on the *conditional similarity* between the characteristics of asset i and asset j,

$$A_t = X_t W X_t'. \tag{5}$$

In this formulation, information is more actively shared among assets with similar attributes. In fact, if the rows of X_t are variance-standardized, and if W is the identity matrix, then the

[5]This is related to the cross-asset prediction technique of principal portfolios proposed by Kelly et al. (2023) and their multivariate extension by He et al. (2022).

$(i,j)^{\text{th}}$ element of A_t reports the correlation across characteristics for the asset pair (i,j). If W is not the identity matrix but is instead a $D \times D$ matrix of free parameters, then the model has the flexibility to learn which conditioning characteristics represent the most fruitful pathways of cross-asset prediction.[6]

Through algebraic manipulation of (2), we see

$$\begin{aligned} w_t &= (X_t W X_t') X_t \lambda \\ &= X_t(\lambda' \otimes W)\,\mathrm{vec}(X_t' X_t) \\ &= \left(\mathrm{vec}(X_t' X_t)' \otimes X_t\right) \mathrm{vec}\left(\lambda' \otimes W\right) \\ &= \check{X}_t \check{\lambda}. \end{aligned} \tag{6}$$

The portfolio weights in (6) are linear in $\check{X}_t = \mathrm{vec}(X_t' X_t)' \otimes X_t$, which is an $N_t \times D^3$ matrix of three-way interactions among characteristics of all assets, with up to D^3 distinct linear parameters $\check{\lambda}$. This brings the linear portfolio transformer full circle back to the simple SDF model of (3). The linearity of portfolio weights in an extremely large number of nonlinear functions of X_t is reminiscent of the high complexity SDF formulation of DKKM. But unlike DKKM, the linear transformer achieves this complexity by exploiting cross-asset predictability in the form of cross-asset characteristic interactions.

2.3 Transformers and Factor Timing

It is common in the literature to use the set of D characteristics in X_t to construct characteristic-managed portfolios, or "factors," whose returns are defined by the $D \times 1$ vector $F_{t+1} = X_t' R_{t+1}$. A basic linear portfolio specification, $w_t = X_t \lambda$ as in (3), can thus be viewed as a portfolio of factors:

$$w_t' R_{t+1} = F_{t+1}' \lambda. \tag{7}$$

[6]The information entering the attention function A_t can differ from the core information that maps into portfolio weights in (4) without loss of generality. For example, we may wish to consider $w_t = (Y_t W Y_t') X_t \lambda$ for some Y_t with first dimension N_t and $Y_t \neq X_t$. For simplicity of exposition, we work from a single matrix of conditioning variables X_t throughout.

Recent literature on "factor timing" has documented robust evidence of time-series predictability in anomaly factor returns.[7] The literature also documents portfolio gains from strategies that form dynamic combinations of factors to exploit time variation in factor expected returns. One example of this approach is factor momentum, which builds a portfolio that combines factors in proportion to their recent average returns.

In light of this, equations (6) and (7) together give another perspective on how the portfolio transformer approaches cross-asset prediction. It introduces an enormous number of ways to time each factor in F_t using each element of $X_t'X_t$, as captured by the term $\check{X}_t$. The trained portfolio transformer coefficient vector $\text{vec}\left(\lambda' \otimes W\right)$ selects the best combination of these factor timing strategies.

Kelly et al. (2023) develop an asset pricing theory for simple timing models and establish a rigorous connection with the problem of cross-asset return prediction. Their analysis deals with one factor at a time and in low-complexity settings. In contrast, the portfolio transformer pushes cross-asset information sharing to extreme levels of complexity and conditionality while jointly optimizing the timing of all factors.

2.4 Multiple Heads

In the machine learning literature, the structure in (5) is referred to as an attention "head." Transformers typically possess multiple attention heads. In analogy, we define the H-head linear portfolio transformer as

$$\begin{aligned} w_t &= \underbrace{(X_t W_1 X_t')X_t\lambda_1}_{\text{head \# 1}} + \cdots + \underbrace{(X_t W_H X_t')X_t\lambda_H}_{\text{head \# H}} \\ &= \left(\text{vec}(X_t'X_t)' \otimes X_t\right) \text{vec}\left(\sum_{h=1}^{H} \lambda_h' \otimes W_h\right) = \check{X}_t\check{\lambda}. \end{aligned} \tag{8}$$

Why incorporate a variety of heads in the portfolio transformer? Equation (8) shows that multi-head attention allows for multiple pathways of cross-asset predictability. Increasing the number of heads increases the model's parameterization and thus its flexibility without

[7] See Gupta and Kelly (2019), Haddad et al. (2020), Kelly et al. (2023), and Ehsani and Linnainmaa (2022).

changing the regressors, $\check{X}_t$, which are the same regardless of the number of heads. Relatedly, we can think of the multi-head formulation as performing model averaging—(8) is an ensemble of H different single-head models for the optimal portfolio.[8]

2.5 Estimation and Identification

A natural objective function for training an AIPM with SDF weight $w\left(X_t;\Theta_{\mathcal{T}}\right)$ is:

$$\min_{\Theta_{\mathcal{T}}} E_t\left[\left(1-w\left(X_t;\Theta_{\mathcal{T}}\right)' R_{t+1}\right)^2\right]+g(\Theta_{\mathcal{T}};z), \tag{9}$$

where $w\left(X_t;\Theta_{\mathcal{T}}\right)$ is the SDF weight function with parameters $\Theta_{\mathcal{T}}$, and $g(\Theta_{\mathcal{T}};z)$ is a shrinkage penalty term with a shrinkage parameter z. This objective is built around the equivalence between an SDF and the mean-variance efficient portfolio. Kelly and Xiu (2023) refer to this penalized least squares problem as maximum Sharpe ratio regression (MSRR) since its solution is the (penalized) conditionally efficient portfolio. DKKM prove that the estimator in (9) approximates the conditionally mean-variance efficient portfolio, and they derive technical conditions under which this solution consistently recovers the true optimal portfolio.[9] Similarly, (9) maps to the standard asset pricing Euler condition that the true SDF M_{t+1} conditionally prices all assets with zero error:

$$E_t[M_{t+1}R_{t+1}]=0, \quad \text{where} \quad M_{t+1}=1-w\left(X_t;\Theta_{\mathcal{T}}\right)' R_{t+1}. \tag{10}$$

When $g(\Theta_{\mathcal{T}};z)=0$, the Euler equation (10) is the first-order condition of the optimization problem in (9). It is also the vector of SDF pricing errors for all risky assets. In other words, the objective (9) can be viewed as a portfolio optimization problem or, equivalently, as a pricing error minimization problem.

The estimation problem in (9) is presented in general terms and also accommodates the nonlinear specification introduced in Section 3. In the case of the linear portfolio transformer

[8] One technical caveat regarding multi-head attention is that the parameters in this formulation are not uniquely identified. We discuss our approach to identification in the following section.

[9] Britten-Jones (1999) establishes the link between an unconditional version of (9) and the unconditional maximum Sharpe ratio portfolio.

with a ridge penalty function, the objective is

$$\min_{\check{\lambda}}\{E_t\left[\left(1-\check{\lambda}'\check{X}_t'R_{t+1}\right)^2\right]+z\check{\lambda}'\check{\lambda}\}\,, \tag{11}$$

which delivers a convenient closed-form expression for the linear portfolio transformer estimator:

$$\hat{\check{\lambda}}=\left(\sum_t \check{X}_t'R_{t+1}R_{t+1}'\check{X}_t+zI\right)^{-1}\left(\sum_t \check{X}_t'R_{t+1}\right). \tag{12}$$

This formula begins to shed light on the issue of identification in the multi-head transformer. The number of unique parameters in an H-head model is $P=H(D^2+D)$, as there are D^2 parameters for each W_h and D parameters for each λ_h. The dimension of the regression parameter $\check{\lambda}$ in (8) is D^3. If the number of heads is large enough that $H(D^2+D)\geq D^3$, MSRR can recover up to D^3 unique parameters. We refer to a model with D^3 unique parameters as "saturated" because it is the richest unique parameterization that can be achieved with a linear transformer using D conditioning variables.

If the number of heads is small enough that $H(D^2+D)<D^3$, then there are cross-parameter restrictions on the elements of $\check{\lambda}$ and the estimator in (12) must be modified to impose these constraints. In Appendix A, we provide an algorithm for recovering a unique set of $H(D^2+D)<D^3$ restricted parameters, which amounts to a form of constrained least squares. By varying the number of attention heads, we can investigate the effect of model complexity on the performance of the linear transformer.

3 The Nonlinear Portfolio Transformer

The linear portfolio transformer offers a window into the role of attention for an AIPM. However, it abstracts from a variety of nonlinearities typically employed in the transformers that underlie LLMs and other leading AI models. In this section, we develop a deep nonlinear portfolio transformer. Its architecture is an asset pricing adaptation of well-known transformer models, such as the one introduced in Vaswani et al. (2017).

First, we give a complete statement of the nonlinear portfolio transformer. Then, we discuss the intuition behind its various nonlinearities and how they aid model performance.

3.1 Model Specification

The nonlinear portfolio weight function is a K-block transformer architecture. Each block is composed of two sublayers. The first sublayer applies a multi-head attention unit defined as

$$\mathcal{A}(Y) = \sum_{h=1}^{H} \sigma\left(YW_hY'\right) YV_h \tag{13}$$

to the generic $N_t \times D$ matrix input Y. W_h and V_h are $D \times D$ matrices of parameters. The function $\sigma : \mathbb{R}^{N_t \times N_t} \to \mathbb{R}^{N_t \times N_t}$ is a row-wise softmax operator applied to the $N_t \times N_t$ matrix YW_hY'.

After the softmax operation, the original Y is added back in a so-called "residual connection:"

$$\mathcal{A}^{\mathcal{R}}(Y) = \mathcal{A}(Y) + Y. \tag{14}$$

The second sublayer is a fully connected feed-forward network with one hidden layer of d_f neurons:

$$\mathcal{F}(Y) = \max\left[0, Y\mathcal{W}_1 + \iota b_1'\right]\mathcal{W}_2 + \iota b_2', \tag{15}$$

where the parameter matrix $\mathcal{W}_1$ is $D \times d_f$, b_1 is $d_f \times 1$, $\mathcal{W}_2$ is $d_f \times D$, b_2 is $D \times 1$, and ι's are conforming vectors of ones.[10] The output of the feed-forward step is, therefore, the same dimension as the input, $N_t \times D$. Again, we include a residual connection after the feed-forward network:

$$\mathcal{F}^{\mathcal{R}}(Y) \;=\; \mathcal{F}(Y) + Y\,. \tag{16}$$

[10]The max operator in (15) is applied row-wise. More specifically, the network can be understood as operating on asset-by-asset observations, so it takes the $D \times 1$ input y_i (the i^{th} row of Y) and produces a $D \times 1$ output. These outputs are then stacked into the $N_t \times D$ matrix $\mathcal{F}(Y)$.

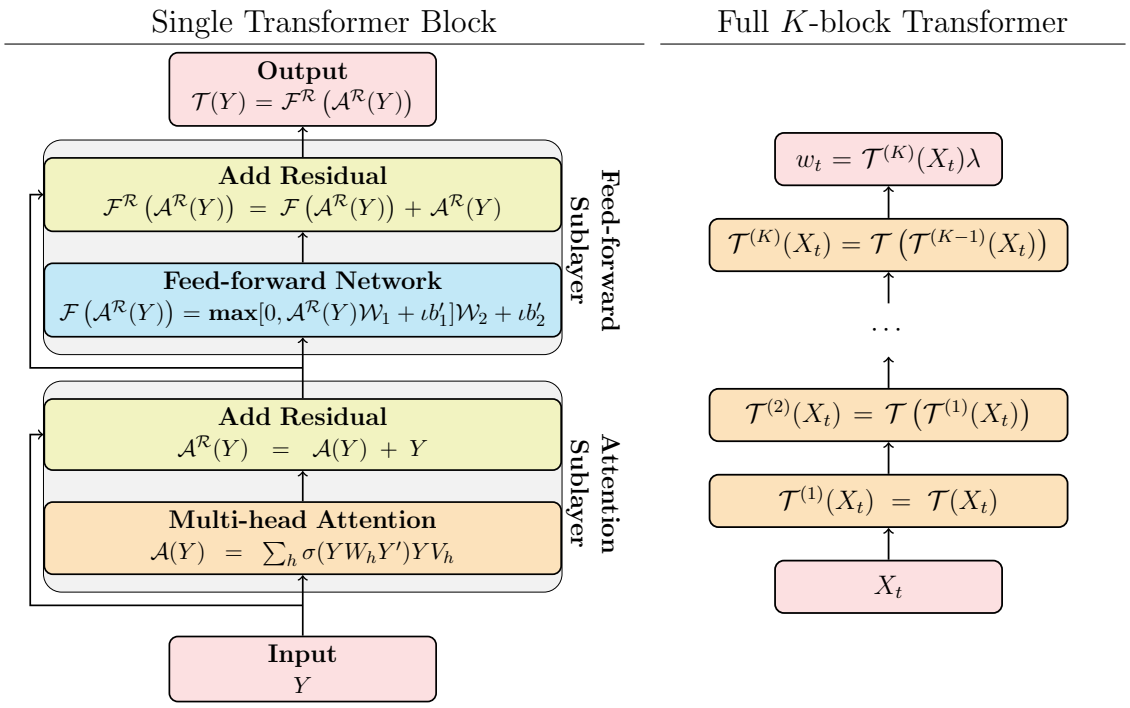

Figure 1: **Illustration of Nonlinear Portfolio Transformer**

The right figure shows the full architecture of a K-block portfolio transformer. The left figure shows the structural details within a transformer block.

A complete transformer block $\mathcal{T}$ is a composition of these two sublayers:

$$\mathcal{T}(Y) = \mathcal{F}^{\mathcal{R}}\big(\mathcal{A}^{\mathcal{R}}(Y)\big). \tag{17}$$

Given the transformer block in (17), we can state the full recursive definition of the portfolio transformer. The recursion is initialized with an identity layer whose input and output are the raw $N_t \times D$ conditioning matrix X_t:

$$\mathcal{T}^{(0)}(X_t) = X_t. \tag{18}$$

The input to each additional transformer block $k = 1, \ldots, K$ is the output from the previous block:

$$\mathcal{T}^{(k)}(X_t) \;=\; \mathcal{T}\left(\mathcal{T}^{(k-1)}(X_t)\right). \tag{19}$$

The recursion culminates in an $N_t \times D$ output matrix of reconstituted asset characteristics, $\mathcal{T}^{(K)}(X_t)$. The final linear layer maps these characteristics into the conditional SDF portfolio weights with a final $D \times 1$ parameter vector λ:

$$w_t = \mathcal{T}^{(K)}(X_t)\lambda. \tag{20}$$

Figure 1 illustrates the flow of input data Y through the network. Transformer block k is equipped with block-specific parameters,

$$\theta^{(k)} = \left((W_h^{(k)}, V_h^{(k)})_{h=1}^H, \mathcal{W}_1^{(k)}, b_1^{(k)}, \mathcal{W}_2^{(k)}, b_2^{(k)}\right), \ k = 1, \ldots, K\,,$$

and the complete set of AIPM parameters is then given by

$$\Theta_{\mathcal{T}} = \left\{(\theta^{(k)})_{k=1}^K, \lambda\right\}, \tag{21}$$

for a total of $K(2D^2H + 2Dd_f + d_f + D) + D$ parameters. In the subsequent analysis, we use $d_f = 256$ and $H = 1$.[11]

3.2 The Role of Nonlinearities

DKKM argue that the out-of-sample performance of an SDF improves with the number of parameters. This is a counterpart to the virtue of complexity Kelly et al. (2024) (KMZ henceforth) in the context of asset pricing models. Heavy parameterizations of AI models enhance their capacity to approximate unknown data-generating processes. This phenomenon is well known in machine learning domains like image and language modeling.

The nonlinear portfolio transformer achieves extremely high complexity through the repeated composition of heavily parameterized functions. However, the nonlinear design choices in the transformer architecture are by no means arbitrary. They are the result of continual research refinements discovered largely in the language modeling domain. In this section, we attempt to provide intuition for each of the portfolio transformer's nonlinearities

[11] Our experiments suggest the gains from increasing H are small; hence, we focus on $H = 1$ to reduce computational costs.

and why they are likely to be beneficial in the asset pricing context. A key facet of our empirical analysis explores the sensitivity of AIPM model performance as a function of model complexity.

3.2.1 Softmax Attention

The first nonlinearity to appear in the full portfolio transformer is the softmax function, $\sigma(YW_hY')$, in equation (13). This function works row-wise (i.e., asset-by-asset) on the attention matrix $A_t = YW_hY'$, converting each row to a probability distribution (each element is non-negative and rows sum to one):

$$\sigma(A_t)_{i,j} = \frac{\exp(A_{t,i,j})}{\sum_l \exp(A_{t,i,l})}.$$

While the general attention mechanism gathers information about asset i from the characteristics of all other assets, the softmax operation selectively focuses attention on relatively few related assets while ignoring the rest. This beneficial property of softmax is described in the following lemma.

Lemma 1 (Selective Softmax Attention) *Consider the effect of multiplying W_h by a constant α in $\sigma(X_tW_hX_t')$. As $\alpha \to +\infty$, we have*

$$\lim_{\alpha\to+\infty} \sigma(\alpha X_tW_hX_t')_{i,j} = \frac{1}{|\chi_t(i)|}\mathbf{1}_{j\in\chi_t(i)}. \tag{22}$$

where $\chi_t(i)$ is the attention-worthy set of assets for gathering information about asset i:

$$\chi_t(i) = \{j \in \{1,\dots,N_t\} : \ X_{i,t}'W_hX_{j,t} \ \in \ \arg\max_{j_1} X_{i,t}'W_hX_{j_1,t}\}. \tag{23}$$

Recent theoretical results (Tarzanagh et al., 2023) suggest that transformers tend to operate in the asymptotic regime described by Lemma 1, achieving a binary partition of assets (or, more commonly, language tokens) into those that are relevant for predicting the behavior of asset i and those that are not.

3.2.2 Feed-forward Network

The second nonlinearity of the portfolio transformer is denoted $\mathcal{F}$ in equation (15). In each transformer block k, $\mathcal{F}$ receives the output from the preceding attention sublayer, $\mathcal{A}^{\mathcal{R}}\left(\mathcal{T}^{(k-1)}(X_t)\right)$, which, like the original conditioning data, contains D characteristics for each of the N_t assets. The characteristics in $\mathcal{A}^{\mathcal{R}}\left(\mathcal{T}^{(k-1)}(X_t)\right)$ have been reconfigured to incorporate attention-based information sharing and other nonlinearities in previous layers. $\mathcal{F}$ further transforms these with a shallow, fully connected feed-forward neural network and introduces a large number of additional parameters every time the feed-forward layer is applied.

Despite the intricate structure of the feed-forward layer, this is perhaps the best-understood and least novel aspect of the transformer. Its role is to refine the information about each asset by parsing out the most predictive nonlinear representations of the inputs. Note that, because $\mathcal{F}$ operates row-wise, it transforms the "characteristics" of each asset i independently of the other assets. Thus, the nonlinearity in this layer acts on an asset's own features alone. Therefore, it does not directly contribute to the transformer's cross-asset information sharing.

The feed-forward component is the transformer's closest analogue to the neural network architecture employed by DKKM. Their model can be represented as

$$w_t = S(X_t)\lambda\,, \tag{24}$$

where S is a neural network mapping applied to own-asset features (precluding cross-asset prediction). The model achieves its complexity by expanding the original D features to a much larger set of $P \gg D$ nonlinear transformations of those features. The DKKM model provides another natural benchmark for the AIPM by separating the effects of nonlinearities from the effects of cross-asset prediction. As we show in the empirical analysis, the combination of both nonlinearity and cross-asset prediction delivers the strongest model, outperforming models with only own-asset nonlinearity or only cross-asset prediction in a linear transformer.[12]

[12]This is consistent with evidence of nonlinear transformer performance in the language domain (Dong

3.2.3 Residual Connections

Both the attention and feed-forward sublayers undergo a residual connection step before passing on their output. Residual (or "skip") connections are common in deep learning models (Orhan and Pitkow, 2017) and are critical to the performance of transformers (Dong et al., 2021). The literature primarily attributes their benefit to stabilizing the optimization by helping to counteract so-called vanishing and exploding gradients during training.

4 Empirical Findings

4.1 Data

In this section, we dissect the empirical performance of AIPMs. To make the conclusions from this analysis as easy to digest as possible, we perform our analysis in a conventional setting with conventional data. We focus on the SDF portfolio problem using monthly data for US stocks over the period 1963 to 2022 from JKP. This open-source dataset compiles an extensive collection of stock characteristics from the finance literature.[13] The universe includes NYSE/AMEX/NASDAQ stocks with CRSP share codes 10–12.

To leverage this data in a machine learning environment, it is helpful to have a stock-month panel with few missing values and consistency in the set of characteristics that are available over time. To this end, we follow the sample filters of DKKM. First, we restrict the raw dataset of 153 characteristics to a smaller group of 132 characteristics that have fewer than 30% missing values over the full sample. Next, we drop "nano" stocks whose market capitalization is below the 1^{st} percentile of the NYSE sample. Then, we exclude stock-month observations with missing values for more than a third of the characteristics. Finally, we cross-sectionally rank-standardize each characteristic into the $[-0.5, 0.5]$ interval and replace any missing characteristic values with the cross-sectional median value of zero. After this filtering procedure, the resulting $N_t \times D$ matrix of characteristics each month constitutes

et al., 2021; Tarzanagh et al., 2023).

[13] JKP characteristics for individual stocks can be downloaded at `https://wrds-www.wharton.upenn.edu/pages/get-data/contributed-data-forms/global-factor-data/`. Detailed documentation and further factor portfolio data are available at `jkpfactors.com`.

the conditioning set X_t used in our empirical analysis.

4.2 Performance Metrics and Benchmark Models

We use two primary metrics to evaluate the out-of-sample performance of an asset pricing model. The first is the out-of-sample Sharpe ratio of the SDF portfolio. This is a natural model comparison statistic because the true SDF coincides with the mean-variance efficient portfolio.

The second metric is the out-of-sample Hansen and Jagannathan (1997) distance or HJD. As discussed in DKKM, the out-of-sample HJD statistic has attractive model comparison properties. It averages an SDF's squared pricing errors across all test assets with a fixed weighting matrix defined as the out-of-sample inverse covariance matrix of test assets. With this weighting matrix, the HJD can be interpreted as the out-of-sample pricing error of the portfolio of test assets that is most mispriced by the model. And because the weighting matrix is the same for all models, the HJD can be directly compared across models (unlike other alpha or GMM-based statistics). The set of test assets we study is the 132 JKP anomaly factors, though none of our conclusions are sensitive to this choice.[14] We also report a robust HJD statistic in which the test asset out-of-sample covariance matrix (denoted $\hat{\Sigma}$) is replaced with its ridge-regularized counterpart (denoted $\tilde{\Sigma}(z)$) according to

$$\tilde{\Sigma}(z) = (1-z)\hat{\Sigma} + zI_N. \tag{25}$$

This ensures that our model comparison is not driven by instability in $\hat{\Sigma}^{-1}$. In our empirical analysis of the robust HJD statistic, we fix $z = 0.5$, but our conclusions are similar to other choices of $z \in [0, 1]$.

To infer the statistical significance of differences between models, we report pairwise SDF performance comparisons. In particular, for two models, A and B, we run an alpha regression

[14] Throughout our analysis we use JKP anomaly factors constructed as $F_{t+1} = X_t' R_{t+1}$, where X_t are the JKP characteristics. These differ somewhat from the portfolio sort-based factors on the JKP website. This choice maintains consistency between the test asset factors and the notion of "factors" implicit in the BSV, DKKM, and transformer models. None of our empirical analyses are sensitive to using the sort-based factors from the JKP website.

$$R_{A,t} = \alpha + \beta R_{B,t} + \epsilon_t \,, \tag{26}$$

where the out-of-sample SDF returns on either side of the regression are rescaled to have 15% annualized volatility (to aid comparability of alphas across models). We report the annualized alpha estimate as well as its t-statistic.

We compare AIPMs to a variety of benchmark models. The first four benchmarks are standard small linear factor models from the literature. We also compare against a moderately large linear model and two large nonlinear models:

FF6: This is a six-factor model that includes the five factors of Fama and French (2015) plus the UMD momentum factor.

SY: This is a four-factor model that includes the "mispricing" factors of Stambaugh and Yuan (2017).

HXZ: This is a five-factor model that includes the q-factor model specification of Hou et al. (2015) augmented to include the expected growth factor of Hou et al. (2021).

DHS: This is a three-factor model that includes the long-horizon and short-horizon behavioral factors of Daniel et al. (2020).

BSV: This model uses SDF weights that are linear in stock characteristics, $w_t = X_t\lambda$, following Brandt et al. (2009). Because the number of characteristics D in the JKP data is relatively large, we estimate λ using a ridge penalty as in (11).[15]

DKKM: This is the shallow neural network SDF of DKKM based on random features. It is nonlinear and high-dimensional, but it precludes cross-asset predictability. The SDF weight specification is $w_t = S(X_t)\lambda$ where $S_{i,t}(k) = \text{ReLU}(\gamma_k' X_{i,t})$ for all assets i with $k = 1, \ldots, P$. Our empirical analysis sets $P = 25{,}000$.[16]

[15] We use the same ridge penalty selection as for the linear transformer (Section 4.3).

[16] DKKM use sin and cos activation functions in their construction of random features, while we use the more common ReLU activation. Our results are insensitive to the particular choice of nonlinear activation.

MLP: This neural network SDF uses a multi-layer perceptron (MLP). Unlike DKKM, the MLP allows for network depth in the nonlinear specification, but it continues to use only own-asset predictability.[17] The corresponding network function, $w^{MLP}(X_t; \Theta^{MLP})$, uses over 300,000 parameters and minimizes the MSRR objective (9) (without a penalty term) with the same training algorithm as the nonlinear transformer (Section 4.3).

4.3 Training

We train all models in 60-month rolling training windows, with the first training window using signals from January 1963 through December 1967 (and corresponding returns data from February 1963 to January 1968). At the end of the training window, we construct one-month-ahead out-of-sample SDF portfolio returns for each model, then we roll the training sample forward one month and retrain. The first out-of-sample return observation is in February 1968, and the last is in December 2022.

Our linear attention model is the fully saturated specification of (11). We consider a grid of shrinkage parameters $z = 10^i$, $i \in \{-10, \dots, 3\}$. We select the ridge penalty that optimizes the MSRR objective with leave-one-out cross-validation. We report statistics of model performance over the 1968–2022 test sample. With $D = 132$ characteristics, the linear attention model is equivalent to a regression with $D^3 \approx 2.3$ million parameters. As we explain in Appendix A, this can be interpreted as a multi-head attention model with approximately 131 ($\approx D^3/(D^2 + D)$) heads.

For the nonlinear transformer, we randomly initialize all elements of the self-attention matrices Q_h, K_h, and V_h as $\mathcal{N}(0, 1/D)$, for each $h = 1, \dots, H$. The reduced-form attention matrix is then given by $W_h = Q_h K_h'$. We initialize feed-forward network weights $\mathcal{W}_1^{(k)}$ as $\mathcal{N}\left(0, \frac{1}{d_f}\right)$ and $\mathcal{W}_2^{(k)}$ as $\mathcal{N}\left(0, \frac{1}{D}\right)$, and we initialize feed-forward biases at 0. We initialize the final layer output weights λ as $\mathcal{N}\left(0, \frac{1}{D}\right)$. We train to minimize the MSRR objective (9) using the Adam gradient descent algorithm (Kingma and Ba, 2014). Because the initial network weights are randomly generated, model estimates are sensitive to the random seed.

[17] We consider 2 hidden layers, each layer of width 512, and ReLU activations, mimicking the feed-forward layer of our nonlinear transformer model. Our data experiments suggest that the performance is increasing in the network width and saturates around a width of 512. By contrast, performance is U-shaped in the network depth and saturates at a depth of 2, making our benchmark choice conservative.

To minimize this dependency, we repeat the training ten times for different random seeds and then average the outcomes.[18]

For the four simple benchmark models, we estimate the SDF as the sample tangency portfolio of factors in the training sample, without ridge shrinkage.[19] For the BSV, we select the ridge shrinkage parameter using the full sample, since the BSV serves only as a benchmark model. For the DKKM model, we follow Didisheim et al. (2024) and use the ridgeless specification.

Some of the simple benchmark models are unavailable for small portions of our February 1968 to December 2022 sample. In particular, out-of-sample SDF returns for FF6 are available from July 1968 to December 2022, HXZ from January 1972 to December 2022, SY from February 1968 to December 2016, and DHS from July 1977 to December 2018. Individual benchmark model statistics use that model's full available sample, and pairwise model comparisons use the intersection of the models' samples.

4.4 Model Performance Comparison

This subsection compares attention-based models with benchmarks. Table 1 reports the performance of asset pricing models in terms of out-of-sample Sharpe ratio and pricing errors. Figure 2 plots the cumulative out-of-sample SDF returns of all models. Figure 3 reports correlations between out-of-sample SDF returns for all models, and Table 2 reports alphas and their t-statistics in pairwise comparisons of SDF returns for all models over the full sample (Table 3 repeats this analysis in the post-2002 subsample).

4.4.1 Setting the Benchmark: Traditional Factor Model Performance

Panel A of Table 1 reports the performance of asset pricing models in the full out-of-sample period from 1968 to 2022. The first four columns show that low-dimensional benchmarks from the literature deliver similar performance, with out-of-sample SDF Sharpe ratios rang-

[18] With ten seeds, the model converges in the sense that additional seeds have no impact on the behavior of the model. Further details of transformer model training and return computations are provided in Appendix B.

[19] Because the number of parameters in these benchmarks is small, the benefits of ridge shrinkage are negligible. We verify this is the case in our data. Allowing for ridge shrinkage results in no discernible improvement in SDF performance for the low-dimensional benchmark models.

Table 1: **Portfolio Transformer Performance**

The table reports annualized out-of-sample Sharpe ratios, HJD pricing errors, and robust HJD pricing errors (with $z = 0.5$) for each model over the full sample (Panel A) and the post-2002 sample (Panel B).

	FF6	HXZ	SY	DHS	BSV	DKKM	Lin. Attn.	MLP	Trans.
Panel A: 1968–2022									
Sharpe Ratio	1.01	1.77	1.37	1.21	3.60	3.87	3.89	4.31	4.57
Pricing Error	0.55	0.42	0.53	0.61	0.15	0.13	0.14	0.13	0.09
Pricing Error (robust)	0.60	0.38	0.56	0.71	0.04	0.04	0.04	0.06	0.03
Panel B: Post-2002									
Sharpe Ratio	0.74	0.95	0.73	0.46	2.03	2.41	2.25	3.07	3.37
Pricing Error	0.75	0.73	0.88	0.87	0.52	0.44	0.48	0.42	0.34
Pricing Error (robust)	0.44	0.43	0.61	0.60	0.17	0.17	0.17	0.20	0.12

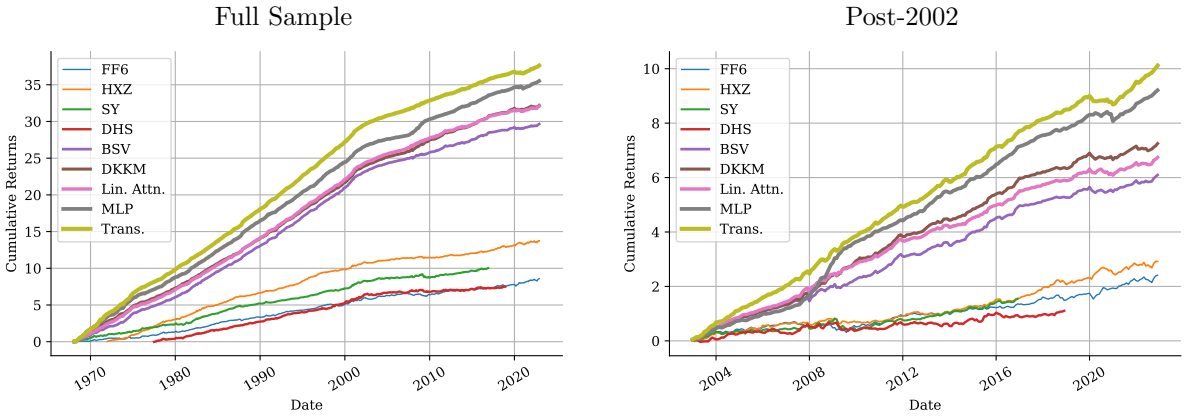

Figure 2: **Cumulative SDF Returns**

This figure shows the cumulative sum of SDF returns for the linear attention and nonlinear transformer models and benchmark models. The left panel shows the full out-of-sample period beginning in 1968, and the right panel begins in 2003.

ing from 1.0 on the low end (FF6) to 1.8 on the high end (HXZ). In terms of pricing errors, the best performer is HXZ with an HJD pricing error of 0.42, while the largest pricing error is 0.61 from DHS. The ranking of benchmark models in terms of pricing errors is preserved when we use the robust HJD, whose covariance matrix is regularized with 50% shrinkage to the identity matrix.

A notable feature of the cumulative returns in Figure 2 is the flattening of all models around 2002. The scale of the plot is dictated by the performance of machine learning models, but on close inspection, the post-2002 flattening is evident for low-dimensional benchmarks as well. Motivated by this, Panel B of Table 1 reports out-of-sample model performance in

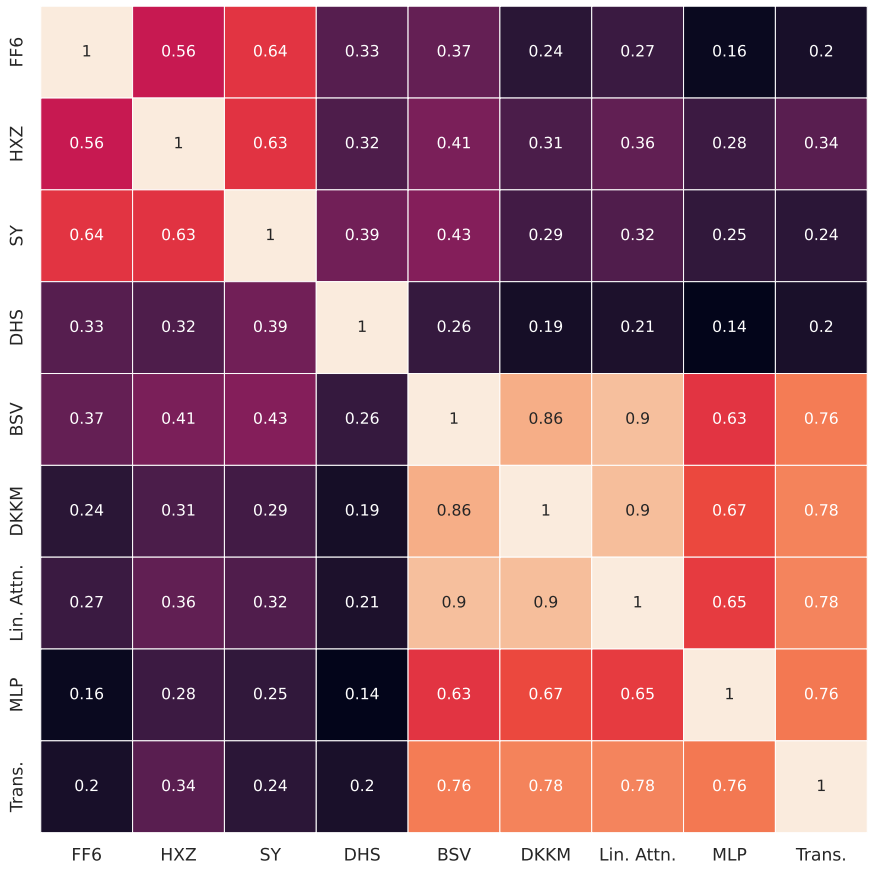

Figure 3: **Pairwise Correlations**

This figure shows pairwise correlations of SDF returns across all models.

the post-2002 subsample. In this latter period, the performance of all simple benchmarks is notably worse, and the ranking of simple benchmarks changes. The best model in terms of the Sharpe ratio is still HXZ (0.95), but the worst is now DHS (0.46). In the latter sample, HXZ achieves the smallest pricing error (0.73), while SY has the largest (0.88). Figure 3 shows that the FF6, HXZ, and SY models are roughly 60% correlated with each other, while DHS is less correlated.

Table 2 reports alphas and their t-statistics in pairwise comparisons of out-of-sample SDF returns for all models (the result in each position of the table corresponds to regressing the column SDF on the row SDF). The last row reports alphas from the regression of column models on all other models. While simple benchmarks all perform somewhat similarly to one another, Table 2 indicates that their performance differences are often statistically significant

Table 2: **Pairwise Model Comparison, Full Sample**

The table reports alphas and their t-statistics in pairwise comparisons of out-of-sample SDF returns for the sample from July 1977 to December 2016. For each pair of models, we report the alpha from regressing out-of-sample SDF returns from the "column" model on the SDF of the "row" model: $R_{\text{column},t} = \alpha + \beta R_{\text{row},t} + \epsilon_t$. The SDF returns on both sides of the regression are rescaled to have 15% annualized volatility to aid comparability of alphas across models. We report alphas as annualized percentages along with t-statistics in parentheses. The last row reports the alpha from regressing the column model SDF on all other models jointly. Asterisks indicate statistical significance at the 99% confidence level.

	FF6	HXZ	SY	DHS	BSV	DKKM	Lin. Attn.	MLP	Trans.
FF6		1.5*	0.8*	1.0*	4.4*	5.2*	5.0*	6.1*	6.4*
		(8.8)	(5.0)	(5.3)	(23.5)	(25.8)	(25.3)	(29.9)	(32.0)
HXZ	0.1		0.2	0.8*	3.9*	4.7*	4.5*	5.7*	5.9*
	(0.7)		(1.0)	(3.7)	(19.9)	(22.5)	(21.9)	(26.6)	(28.5)
SY	0.3	1.2*		0.9*	4.3*	5.0*	4.9*	5.9*	6.3*
	(2.0)	(7.4)		(4.4)	(22.4)	(24.8)	(24.2)	(28.9)	(31.0)
DHS	0.9*	1.8*	1.1*		4.6*	5.3*	5.1*	6.2*	6.5*
	(4.5)	(9.1)	(5.7)		(22.8)	(25.5)	(24.9)	(29.6)	(31.5)
BSV	-0.8*	-0.1	-0.5	0.2		1.3*	0.9*	3.0*	2.6*
	(-2.8)	(-0.5)	(-1.9)	(0.6)		(8.0)	(6.8)	(13.3)	(15.6)
DKKM	-0.2	0.3	-0.1	0.5	0.3		0.4*	2.6*	2.4*
	(-0.5)	(1.0)	(-0.3)	(1.5)	(1.7)		(3.1)	(11.0)	(12.1)
Lin. Attn.	-0.4	0.1	-0.2	0.4	0.1	0.7*		2.7*	2.3*
	(-1.4)	(0.5)	(-0.7)	(1.2)	(0.6)	(4.8)		(11.5)	(12.9)
MLP	-0.1	0.5	-0.1	0.6	0.7*	1.2*	1.1*		1.9*
	(-0.4)	(1.4)	(-0.2)	(1.8)	(2.8)	(4.7)	(4.3)		(8.4)
Trans.	-0.5	-0.3	-0.5	0.2	-0.7*	0.2	-0.2	1.2*	
	(-1.4)	(-1.0)	(-1.4)	(0.5)	(-3.4)	(0.9)	(-1.1)	(4.9)	
Others	0.0	0.2	-0.2	0.5	-0.5*	0.3	0.0	1.1*	1.3*
	(0.0)	(0.7)	(-0.8)	(1.5)	(-3.7)	(1.9)	(0.3)	(4.7)	(7.7)

at the 1% level, and HXZ appears dominant among simple factor models over the full sample. However, these differences between low-dimensional models become smaller in magnitude and are generally insignificant in the post-2002 sample (Table 3).

These four simple benchmark models summarize the behavior of asset pricing models that i) use small conditioning sets, ii) impose a specific structure with a small number of estimated parameters, and iii) restrict factors to use only own-asset predictive characteristics.

Table 3: **Pairwise Model Comparison, Post-2002 Sample**

The table repeats the analysis of Table 2 for the sample beginning in 2003. Asterisks indicate statistical significance at the 99% confidence level.

	FF6	HXZ	SY	DHS	BSV	DKKM	Lin. Attn.	MLP	Trans.
FF6		0.7	0.4	0.3	2.4*	3.5*	3.0*	4.4*	4.4*
		(2.3)	(1.4)	(0.9)	(8.4)	(10.6)	(9.6)	(13.2)	(14.3)
HXZ	0.3		0.0	0.2	2.3*	3.3*	2.9*	4.4*	4.2*
	(1.0)		(0.1)	(0.6)	(7.8)	(10.3)	(9.2)	(12.9)	(14.1)
SY	0.4	0.5		0.3	2.4*	3.4*	3.0*	4.4*	4.4*
	(1.5)	(2.1)		(0.8)	(8.4)	(10.7)	(9.6)	(13.3)	(14.8)
DHS	0.8	1.0*	0.7		2.7*	3.6*	3.2*	4.6*	4.7*
	(2.4)	(3.1)	(2.3)		(8.7)	(11.1)	(10.0)	(13.6)	(14.4)
BSV	-0.7	-0.4	-0.7	-0.6		1.3*	0.8*	3.0*	2.4*
	(-2.0)	(-1.1)	(-1.9)	(-1.7)		(5.9)	(4.3)	(8.9)	(10.9)
DKKM	-0.2	-0.3	-0.5	-0.5	-0.2		0.0	2.3*	1.8*
	(-0.6)	(-0.7)	(-1.2)	(-1.1)	(-0.8)		(0.1)	(6.6)	(6.9)
Lin. Attn.	-0.5	-0.2	-0.4	-0.5	-0.1	0.7*		2.7*	2.1*
	(-1.2)	(-0.4)	(-1.1)	(-1.2)	(-0.5)	(3.8)		(7.7)	(8.4)
MLP	-0.0	0.1	-0.4	0.1	0.4	0.9	0.7		1.8*
	(-0.0)	(0.3)	(-0.9)	(0.3)	(0.9)	(2.3)	(1.8)		(4.8)
Trans.	-1.2	-1.3*	-1.6*	-0.8	-1.2*	-0.1	-0.5	1.4*	
	(-2.6)	(-2.9)	(-3.6)	(-1.7)	(-4.3)	(-0.3)	(-1.9)	(3.7)	
Others	0.2	0.1	-0.6	0.2	-0.5	0.2	-0.0	1.1*	1.4*
	(0.4)	(0.2)	(-1.7)	(0.4)	(-2.5)	(0.7)	(-0.2)	(2.9)	(6.0)

4.4.2 Expanding the Conditioning Set

The BSV model maintains the basic linear structure of standard low-dimensional benchmarks but increases the dimensionality of the SDF to 132 factors. While the functional form is basic, this is still a relatively high-dimensional model with a complexity ratio (P/T) of 2.2, given our 60-month training window. By increasing the set of conditioning variables to include more anomaly patterns from the literature, this linear model shows a large improvement in performance. Its out-of-sample Sharpe ratio is more than double that of the best simple model, and its pricing error drops by two-thirds. It has a large and significant alpha versus

all four simple benchmark models. Like all models, its performance is weaker in the latter sample, but its relative outperformance of simple benchmarks is preserved.

4.4.3 Introducing Nonlinearities

DKKM uses the same large conditioning set as BSV, but more flexibly leverages this conditioning information in a nonlinear specification. The thousands of DKKM factors are long-short portfolios based on stock characteristics that have been transformed through a wide and shallow neural network. While the nonlinearities in DKKM allow for interactive effects among predictor variables within the same stock, DKKM maintains the separation of signals across stocks and thus rules out cross-stock information sharing.

Extracting nonlinear information from characteristics with a shallow network improves model performance by about 10% over the linear BSV specification (a 7.5% rise in Sharpe ratio and a 13.3% reduction in pricing error). The benefits of nonlinearity are perhaps best illustrated by the fact that DKKM has a large and significant alpha versus BSV, while the alpha of BSV relative to DKKM is essentially zero (both economically and statistically).

4.4.4 Information Sharing Via Attention

Linear attention introduces the possibility of cross-asset information sharing in the SDF. It does so in a particularly tractable way that premultiplies the BSV model's linear SDF weights with a dynamic attention matrix. The linear attention model achieves an out-of-sample Sharpe ratio of 3.89 and a pricing error of 0.14 in the full sample. This is a significant improvement over the BSV model.

The linear attention model can also be interpreted as a nonlinear SDF relying on triple interactions among characteristics for all stocks. As discussed in Section 2.4, the parameterization of the linear transformer model varies depending on the number of attention heads. Figure 4 reports the performance of linear attention as we vary the model from a single attention head to the maximum possible number of heads. The figure shows a benefit from using more heads, all else equal, with out-of-sample performance eventually flattening after about 20 heads.

The linear attention model only marginally improves over the nonlinear DKKM model,

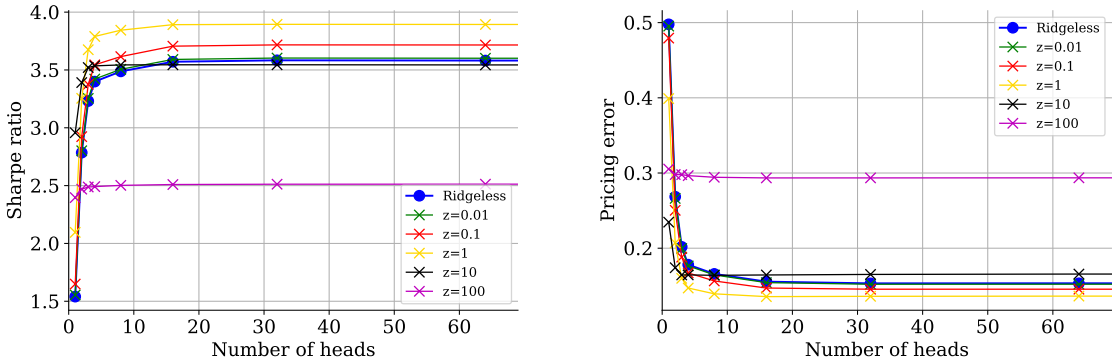

Figure 4: **Complexity and Linear Attention Model Performance**

This figure reports out-of-sample Sharpe ratio (left panel) and pricing error (right panel) as a function of the number of heads in the linear attention model with $D = 132$ characteristics. The simplest model we consider uses a single attention head, and the most complex model is fully saturated with D^3 unique parameters (132 heads). Different curves illustrate how the degree of ridge shrinkage affects model performance.

despite DKKM's narrower reliance on its own-asset predictive information. In the latter part of the sample, linear attention continues to outperform the BSV specification, but it underperforms the shallow neural network of DKKM in terms of both Sharpe ratio and pricing error. Perhaps more surprisingly, the linear attention model is highly correlated (90%) with BSV and DKKM. Evidently, the simple linear attention specification unlocks only modest (but in some cases significant) benefits of cross-asset information sharing.

4.4.5 Interactive Effects of Information Sharing and Nonlinearity

We now turn to the full nonlinear transformer specification. This uses multiple sequential layers of attention blocks and nonlinearity to more flexibly incorporate predictive information from the conditioning variables. However, the transformer's depth and nonlinearity can aid the model's recovery of own-asset effects as well as cross-asset information sharing. Thus, to drill more specifically into the cross-asset information mechanism, we must benchmark the transformer against a multi-layer neural network that has similar nonlinearities and depth, but that shuts down cross-asset prediction. The MLP provides the necessary benchmark. With it, we can evaluate the information-sharing role of the transformer while controlling for deep nonlinearities in own-asset prediction.

The performance of the MLP model shows that pure depth (even without cross-asset

prediction) is a powerful modeling device for asset returns. The out-of-sample MLP Sharpe ratio of 4.31 is a large improvement compared to the shallow DKKM model (Sharpe ratio of 3.87), though their pricing errors are about the same. In the more recent sample, the gains from depth are more pronounced with an MLP Sharpe ratio of 3.07 versus 2.41 for DKKM. In both samples, the MLP has a large and highly significant alpha versus DKKM. Thus, for the transformer to excel in cross-asset information sharing, it must exceed the high bar set by the MLP.

By leveraging deep cross-asset information sharing, the transformer raises the out-of-sample Sharpe ratio to 4.57. It reduces anomaly pricing errors by about 30%, from 0.13 for MLP to 0.09. In the post-2002 sample, the transformer Sharpe ratio is 3.37 versus 3.07 for MLP, and the pricing error is 0.34 versus 0.42 for MLP. These improvements are both economically and statistically significant, as shown in Table 3.

Note that while both DKKM and MLP have very large parameterizations (roughly 25,000 and 300,000 parameters, respectively), they have fewer parameters than our largest transformer models (which have roughly one million parameters). However, the underperformance of DKKM and MLP persists even when we increase the number of neurons in each model so that they both reach one million parameters. Their respective Sharpe ratios are essentially unchanged with more parameters, and their return series are more than 90% correlated with the versions reported above. This is because both models are already large enough that they have converged to their respective limiting kernel models (see Jacot et al., 2018; Kelly et al., 2026).[20] In other words, the gains in performance from the transformer come through an improved architecture that admits cross-asset information sharing, rather than from simply increasing the number of parameters in more basic neural networks.

Interestingly, the MLP itself also has significant alpha versus the transformer model. Why is it that these models appear additive to one another? The answer lies in the "limits to learning" (Chen et al., 2025) that accompanies highly parameterized models in environments with limited data (which is the case for machine learning asset pricing models of monthly

[20]This is visualized in the "virtue of complexity" curves reported by DKKM. Their figures show that the model flattens at its highest possible out-of-sample Sharpe ratio by the time the model reaches $25,000$ parameters. Increasing the dimension of this model to one million parameters has essentially zero effect on its behavior, reflecting convergence to its kernel limit.

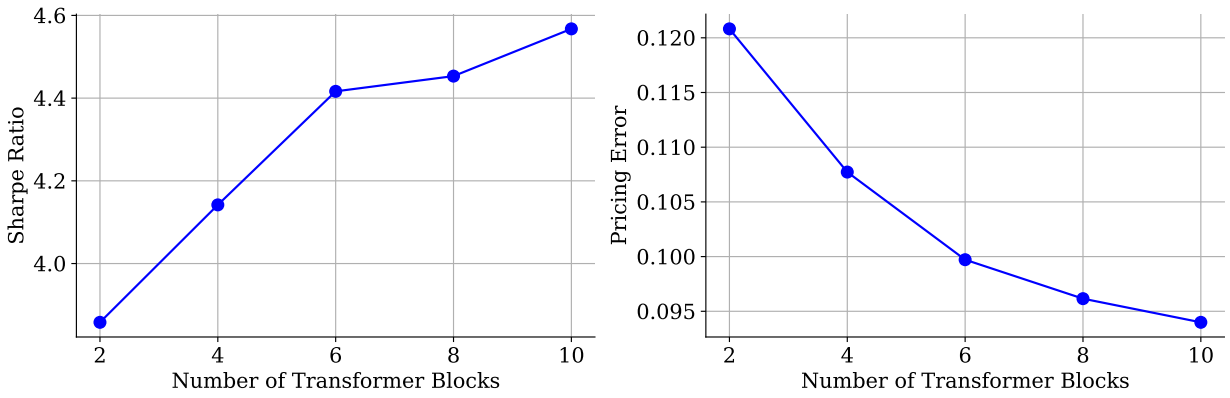

Figure 5: **Depth and Transformer Model Performance**

This figure reports out-of-sample Sharpe ratio (left panel) and pricing error (right panel) as a function of the number of blocks in the nonlinear transformer model. The simplest model we consider uses a single transformer block, and the most complex model uses 10 blocks.

stock returns). Limits to learning occur because the machine learning models are so heavily parameterized relative to the number of available data points that the law of large numbers breaks down. Machine learning models dominate simple models because they are typically more realistic approximations of the data-generating process. But limits to learning imply that these models remain incapable of converging on the true "infeasible best" model. As a result, machine learning models that use different functional forms will have a tendency to learn different (yet correlated and noisy) versions of the true model. As a further result, different machine learning models have a tendency to be additive to one another. Kelly and Malamud (2025) refer to this phenomenon as the "virtue of ensemble complexity." Limits to learning and ensemble complexity are discussed in detail in Kelly and Malamud (2025). Baba Yara et al. (2025) make a related point regarding limits to factor model spanning.

In Figure 5, we analyze how the out-of-sample performance of the portfolio transformer model varies with model depth up to 10 transformer blocks. In the case of transformer models, there may be additional benefits to even deeper specifications, but due to the computational costs of training deep transformers, we leave further exploration for future research. The findings of Figure 5 align with empirical evidence from the literature on large language models (e.g. Kaplan et al., 2020), indicating that additional transformer blocks enhance a model's ability to effectively represent language. It is interesting that the same benefits of

Table 4: **Tangency Portfolio Weights Across Models**

The table reports the estimated ex post optimal portfolio weights for the tangency portfolio of all factors subject to a non-negativity constraint. The top row reports tangency weights for the full sample, and the bottom row reports them for the post-2002 sample.

	FF6	HXZ	SY	DHS	BSV	DKKM	Lin. Attn.	MLP	Trans.
Full sample	0	0	0	0.03	0	0	0	0.37	0.60
Post-2002	0	0	0	0	0	0	0	0.44	0.56

transformer depth emerge in the asset pricing context.

While BSV, DKKM, and linear attention are roughly 90% correlated with one another, MLP and the transformer are more differentiated from that group and from each other (the correlation of MLP and transformer is 76%). Rather than relying solely on pairwise comparisons and alpha tests, we can understand the relative performance and differentiation/diversification of a given SDF versus the set of all models by estimating the ex-post mean-variance combination of models. We do this by fixing the sample volatility of all SDFs to 15% to put models on equal risk footing and imposing a no-shorting constraint. The resulting portfolio weights are reported in Table 4. In the full sample, the transformer commands 60% of the tangency portfolio, followed by 37% for MLP and 3% for DHS (though DHS is insignificant based on its 95% bootstrap confidence interval). The Sharpe ratio of the ex-post tangency portfolio is 5.66, while the Sharpe ratio of the transformer model alone is 5.46.[21] Thus, the gains from supplementing the transformer with other model specifications are relatively small. The tangency weights are roughly the same in the post-2002 sample. In summary, asset pricing models evidently benefit from cross-asset information sharing when the attention mechanism is incorporated in a sufficiently deep network.

[21] As in the analysis of Table 2, the sample is restricted to the intersection of the available data for benchmark models, in this case, July 1977 through December 2016. In this restricted sample, Sharpe ratios differ from the full sample results reported earlier. For example, the nonlinear transformer Sharpe ratio is 5.46 compared to 4.57 in the full period. The main discrepancy arises from the exclusion of the period 2017 to 2022, during which all models perform worse.

4.5 Further Explorations of AIPM Behavior

4.5.1 Information Sharing Through the Lens of Principal Portfolios

The principal portfolio formulation of Kelly et al. (2023) provides a rigorous framework for understanding the extent of cross-asset information sharing and cross-prediction in a trading strategy. They consider a single signal $S_t \in \mathbb{R}^{N_t}$ that predicts the cross-section of returns and analyze a class of strategies $w_t = AS_t$ that exploit return predictability. Their A is a "prediction matrix" that can be thought of as a static attention matrix. They note that typical asset pricing analyses focus on the own-asset predictive effects of signals (take, for example, a momentum strategy in which each asset is bought or sold in proportion to its recent average return). They show that any strategy that relies on a symmetric matrix A can be fully explained by a strategy that uses "own-asset" predictive information.

Kelly et al. (2023) also show that one can identify the distinctive cross-asset predictive effects of a strategy by analyzing the anti-symmetric component of A. In particular, the matrix A can be decomposed as

$$A = A^s + A^a, \quad \text{where } A^s = 0.5(A + A') \quad \text{and } A^a = 0.5(A - A')$$

with the symmetric component of A denoted A^s and the anti-symmetric component denoted A^a. A strategy that relies purely on the anti-symmetric cross-prediction properties of the signal S_t will be unexplained by a symmetric (own-prediction) strategy based on S_t.

We can adapt the formalism of Kelly et al. (2023) to the linear attention modeling framework as follows:[22]

$$\begin{aligned}
w_t &= A_{1,t}(X_t\lambda_1) + \cdots + A_{H,t}(X_t\lambda_H) \\
&= (A^s_{1,t} + A^a_{1,t})(X_t\lambda_1) + \cdots + (A^s_{H,t} + A^a_{H,t})(X_t\lambda_H) \\
&= \underbrace{A^s_{1,t}(X_t\lambda_1) + \cdots + A^s_{H,t}(X_t\lambda_H)}_{\text{symmetric component}} + \underbrace{A^a_{1,t}(X_t\lambda_1) + \cdots + A^a_{H,t}(X_t\lambda_H)}_{\text{anti-symmetric component}} \\
&= w^s_t + w^a_t.
\end{aligned}$$

[22] The principal portfolios are an essentially linear framework. Thus, due to various forms of nonlinearity in the full nonlinear transformer formulation, this decomposition does not apply directly, though extending the symmetry decomposition of Kelly et al. (2023) is an interesting topic for future research.

Table 5: **Symmetry Decomposition of Linear Attention Model**

The table (and the associated cumulative return plot) reports the performance of symmetric and anti-symmetric components of the linear attention SDF over the full sample.

	Lin. Attn. Total	Anti-symmetric Component	Symmetric Component
Sharpe Ratio	3.89	3.10	3.23
Pricing Error	0.14	0.26	0.21

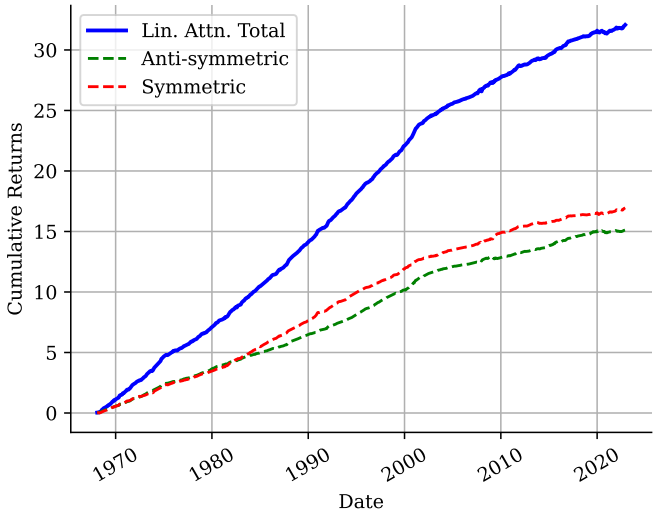

Table 5 reports the performance of the linear attention model through the lens of this symmetry decomposition. We find that both symmetric and anti-symmetric sub-strategies exhibit similar performance, with the symmetric part slightly outperforming. This is consistent with the general focus in the literature on own-asset prediction as a first-order asset pricing phenomenon. The interesting fact is that the anti-symmetric component (i.e., the pure cross-prediction effect) is quite distinct—its out-of-sample returns are only 32% correlated with the symmetric component. Thus, the high return on the overall linear attention SDF derives from its ability to complement the usual "own-prediction" effects of conditioning characteristics with remarkably potent cross-asset predictive effects.

The symmetry decomposition is convenient to apply to the linear attention model. However, the multi-layer nonlinear transformer builds cross-attention recursively, propagating cross-asset information through all the transformer blocks. This makes the extraction of anti-symmetric cross-prediction effects much more cumbersome, and we leave this for future research. Table 2 shows that the performance of the linear attention model is subsumed by

Table 6: **Comparison of AIPM and Factor Timing Strategies**

The table reports the distribution of alphas of timing strategies (in annualized percent) versus the nonlinear transformer model. For each methodology—PP, PAP, and HKS—we report the mean, median, 75th percentile, and 95th percentile of alphas across 138 individual characteristic-based timing strategies, as well as the alpha of the equal-weighted ensemble strategy. Both the dependent and independent variables are standardized to 15% annualized volatility. The sample period is 1968–2019, and the analysis uses the exact strategies constructed by and reported in Kelly et al. (2023).

Factor	Mean	Median	P75	P95	Ensemble
		PP			
α	1.4	1.3	1.8	3.0	1.4
t-stat	0.4	0.3	0.5	0.8	0.4
		PAP			
α	1.7	1.6	2.4	4.3	1.9
t-stat	0.4	0.4	0.6	1.1	0.5
		HKS			
α	2.5	2.2	3.7	6.0	2.8
t-stat	0.6	0.6	1.0	1.6	0.7

the nonlinear transformer, providing indicative evidence that a substantial component of the nonlinear transformer's performance is due to cross-asset information sharing as well.

4.5.2 Comparison With Factor Timing Strategies

In Section 2.3, we discuss the fact that transformer portfolios perform a version of factor timing. In this section, we directly compare AIPM to state-of-the-art factor timing strategies. We focus on the three most potent timing strategies reported in the factor timing comparison of Kelly et al. (2023). These include their principal portfolios ("PP"), principal alpha portfolios ("PAP"), as well as the Haddad et al. (2020) PCA-based factor timing strategy ("HKS"). For each method, Kelly et al. (2023) construct 138 different factor timing strategies. They also report ensemble strategies using all 138 individual factor timing strategies for each method. We therefore make comparisons with a total of 138×3 timing portfolios plus another 3 ensemble portfolios.

Table 6 reports comparisons of timing strategies with the nonlinear portfolio transformer model. We compute time-series regression alphas and associated t-statistics in which the left-hand side return is a candidate factor timing strategy and the right-hand side is the

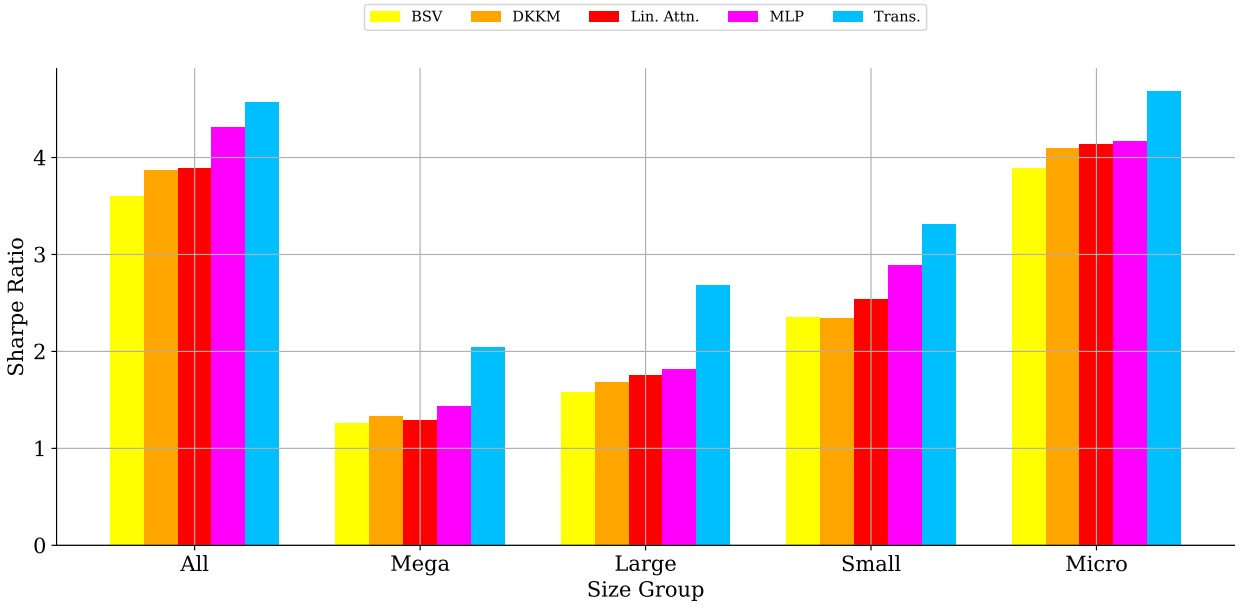

Figure 6: **Sharpe Ratios By Size Group**

This figure reports Sharpe ratios for each model over the full sample for size-based subsamples. The models include BSV, DKKM, linear attention, MLP, and the nonlinear transformer.

AIPM return. The first four columns report the distribution (mean, median, 75th percentile, and 95th percentile) of alphas and of t-statistics across the 138 strategies for each timing method (alpha and t-statistic percentiles are computed independently). The last column shows the alpha and t-statistics for the ensemble timing strategies. Evidently, factor timing strategies are generally well-priced by the transformer model. Even the 95$^{\text{th}}$ percentile of all timing strategies is statistically insignificant (despite the alphas being calculated over a long sample ranging from 1968–2019).[23] The qualitative upshot of Table 6 is that the transformer model appears to achieve not just a high degree of unconditional efficiency, but also high conditional efficiency. This is based on the prevailing view of factor timing strategies as fairly efficient conditional strategies in the empirical literature, and the fact that these strategies are subsumed by the AIPM.[24]

4.5.3 Stock Size Effects and Other Test Assets

Each model that we have studied thus far uses a single specification that applies uniformly to all stocks, large and small, liquid and illiquid. Given the very high Sharpe ratios of all the machine learning specifications that we study, it is likely that much of the performance of these models is generated from the relatively illiquid subset of stocks in our universe.

To investigate the role of cross-asset information sharing in more detail, we first study how SDF Sharpe ratios are affected by restricting the universe of stocks to various JKP size categories: "mega" stocks are those above the 80$^{\text{th}}$ percentile of the NYSE size distribution each month, "large" includes percentiles 50 to 80, "small" includes 20 to 50, and "micro" includes stocks below the 20$^{\text{th}}$ but above the 1$^{\text{st}}$ NYSE size percentile. (Recall that our analysis throughout the paper already excludes "nano" stocks that fall below the first percentile of the NYSE size distribution.) Because there is no natural notion of size groups for the standard simple benchmark models, they are excluded from the analysis in this section.

We retrain each model using stocks in each size group and then report the out-of-sample Sharpe ratios for the models within that group in Figure 6. Three results stand out. First, the nonlinear transformer model dominates all other models in every size group. In fact, the relative increase in performance from the transformer is largest among mega and large-cap stocks. Second, there is a clear size effect for all models—the Sharpe ratio increases monotonically as we move from mega stocks to micro stocks. Third, the pooled sample of all stocks has a higher Sharpe ratio than any subgroup, reflecting the benefits of diversification among a broader universe of stocks.

Next, we investigate size effects by evaluating pricing errors when test assets are JKP factors constructed using only stocks within each size group (the SDF itself is trained using the "all" stock sample). Pricing errors by size group are reported in Figure 7. The main conclusion from this analysis is that pricing errors are generally similar across size groups. On average across competing models, there is a small decrease in pricing errors for mega stocks relative to other size groups. But regardless of the set of test assets, the nonlinear

[23] In contrast, the AIPM has a large and highly significant alpha versus *every* timing strategy we analyzed, which is unsurprising given that the best timing strategies have at most a Sharpe ratio of around 1.5.

[24] We are grateful to the Editor for pointing this out.

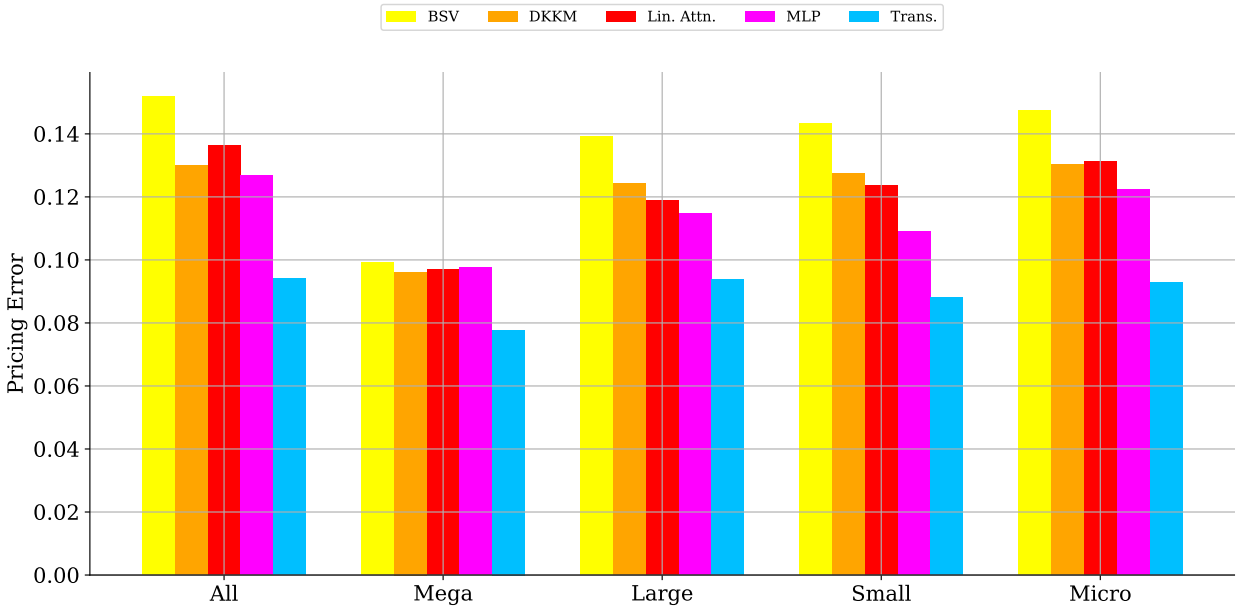

Figure 7: **Pricing Errors By Size Group**

This figure reports HJD pricing errors for JKP factors constructed using only stocks within each size subgroup (using the model trained from the "all" stock sample). The models include BSV, DKKM, linear attention, MLP, and the nonlinear transformer.

Table 7: **Tangency Portfolio Weights Across Models By Size Group**

The table reports the estimated ex-post optimal weights in the tangency portfolio formed from the model SDFs, subject to a nonnegativity constraint, over the full sample. Each model is re-estimated on data restricted to stocks in a given size group. The ex-post tangency portfolio then combines the resulting size-group-specific SDFs. The models include BSV, DKKM, linear attention, MLP, and the nonlinear transformer.

	Mega	Large	Small	Micro
BSV	0.09	0	0	0
DKKM	0	0	0	0
Lin. Attn.	0	0	0	0.23
MLP	0.01	0	0.20	0.23
Trans.	0.91	1.00	0.80	0.54
Tangency SR	1.84	2.70	3.25	4.59
Trans. SR	1.84	2.70	3.23	4.42

transformer delivers smaller pricing errors than its competitors.

The distinctive benefits of the transformer model versus other machine learning specifications are summarized in the ex-post tangency portfolio weights reported in Table 7. For large and mega stocks, the transformer is the only model that receives substantial weight in

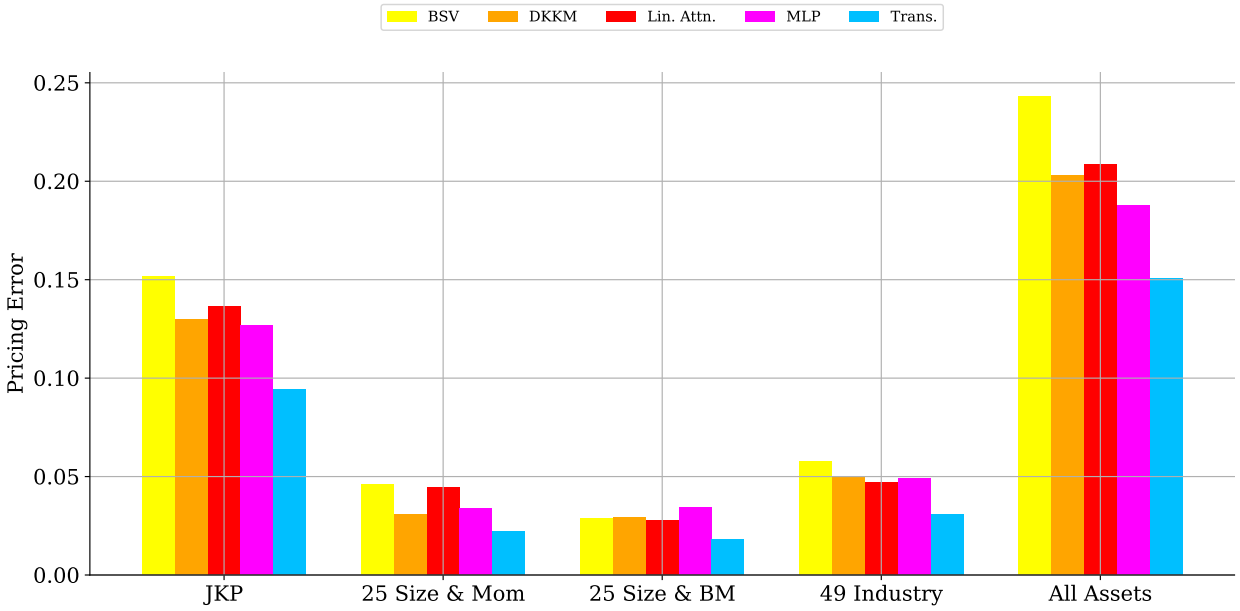

Figure 8: **Fama-French Portfolios as Test Assets**

This figure reports HJD pricing errors based on additional sets of test assets, including the JKP factors, 25 size and momentum-sorted portfolios, 25 size and book-to-market-sorted portfolios, 49 industry portfolios, and all sets of test assets combined. Except for the JKP factors, test assets are from Ken French's website. The models include BSV, DKKM, linear attention, MLP, and the nonlinear transformer.

the tangency portfolio of all candidate SDFs. In the full cross-section, the transformer continues to be responsible for the dominant share of the tangency portfolio, but other machine learning models, such as MLP, also earn meaningful weight, driven by their usefulness in the micro-cap universe.

We further demonstrate the robustness of the transformer model by analyzing pricing errors for additional sets of test assets. In Figure 8 we report HJD pricing errors for the original test assets (132 JKP factors), as well as for 25 portfolios formed on size and momentum, 25 portfolios formed on size and book-to-market, and for 49 industry portfolios (all from Ken French's website). We also report pricing errors when we combine all of these with the 132 JKP factors used in our main analysis. The main conclusion is that the transformer achieves lower pricing errors for every group of test assets. The figure also shows that the JKP factors are more demanding test assets to price, as reflected in their comparatively large pricing error, which motivates our choice to focus on these test assets in our main analysis.

4.5.4 Comparison With Observable Linkages

The transformer relies on statistical learning to infer the best avenues for information sharing across stocks. Relatedly, the literature documents certain manifestations of cross-stock predictability through observable economic linkages between stocks. Examples include supply chain linkages between customer and supplier firms (Cohen and Frazzini, 2008), stocks with an equity research analyst in common (Ali and Hirshleifer, 2020), and technological similarity measured from firms' patents (Lee et al., 2019), among others. In this section, we use data on observable economic linkages to better understand the nature of cross-stock information that is learned by the AIPM's attention mechanism.

Our linkage data are from Liu and Moskowitz (2025), who compile and analyze a variety of observable cross-stock linkages.[25] These include analyst co-coverage, shared headquarters city, customer and supplier linkages, shared lead arrangers, shared institutional ownership, and patent similarity. We also use data on industry membership (four-digit SIC), return correlation, market cap group membership, and liquidity (turnover) group membership based on JKP data (groups are defined based on a stock's modal NYSE quintile for a given variable within the year). Table 8 describes the available linkage types and the available time samples. All linkages are updated at the annual frequency.

Each linkage is recorded as an $N_t \times N_t$ data matrix each month t, denoted $\tilde{L}_{k,t}$, with different linkages indexed by the subscript $k = 1, \ldots, K$. For categorical linkages (e.g., analyst co-coverage and size group membership), $\tilde{L}_{k,t}$ is an adjacency matrix whose (i, j) element takes a value of 1 if stocks have the shared attribute or 0 otherwise. For continuous linkages like patent text similarity and return correlation, $\tilde{L}_{k,t}$ is a weighted adjacency matrix where edge weights reflect the magnitude of the underlying linkage measure. To align linkages with our transformer attention matrix, all linkage data matrices are normalized to be right stochastic (so their rows are positive and sum to one) via the row-wise softmax function, $\sigma(\cdot)$. We denote the transformed linkages by dropping the tilde: $L_{k,t} = \sigma(\tilde{L}_{k,t})$.

We compare linkages to the single-layer, single-head attention formulation of our nonlin-

[25] We are grateful to Yukun Liu for sharing these data.

Table 8: **Observable Stock Linkages**

This table shows the types of observable stock linkage variables and their sample availability. Analyst co-coverage, shared headquarters city, customer and supplier linkages, shared lead arrangers, shared institutional ownership, and patent similarity are from Liu and Moskowitz (2025). We also construct variables for industry membership (four-digit SIC), return correlation, market cap group membership, and liquidity (turnover) group membership (groups are defined based on a stock's modal NYSE quintile for a given variable within the year). All linkages are updated at the annual frequency.

Variable	**Coverage**	**Definition**
Analyst	1990–2019	Share coverage by at least one analyst during year
City	1970–2018	Same headquarters city
Customer	1976–2014	i customer of j during year
Leader	1991–2017	Share at least one lead arranger in DealScan in the previous four years
Liquidity	Full sample	Same turnover quintile
Ownership	1980–2016	At least one institutional owner with more than 1% share in both i and j during year
Patents	1970–2018	Patent textual similarity
Correlation	Full sample	Return correlation (using daily data during year)
SIC-4	Full sample	Same industry
Size	Full sample	Same market capitalization quintile
Supplier	1976–2014	i supplier of j during year

ear transformer model, which takes the form

$$w_t = A_t X_t \lambda,$$

where A_t is the $N_t \times N_t$ nonlinear attention matrix of the transformer model (due to its own softmax transformation, this matrix is also right stochastic).[26] A_t describes how information is shared across assets. The i^{th} row of A_t is the weight that asset i places on all other assets (as learned by the attention mechanism) to "update" and improve upon the information in own-asset characteristics.

We investigate which types of economic relationships among stocks are captured by the transformer's attention mechanism using a regression of the form

$$\text{vec}(A_t) = b_0 + b_1 \,\text{vec}(L_{1,t}) + \ldots + b_K \,\text{vec}(L_{K,t}) + \epsilon_t, \quad t = 1, \ldots, T. \tag{27}$$

[26]The one-layer nonlinear transformer is a conservative benchmark for the observable linkage comparison because our deeper transformers perform at least as well as the one-layer model.

Table 9: **Transformer Attention and Stock Linkages**

The univariate columns report the coefficient and R^2 from separate regressions of the attention matrix A_t on each individual linkage matrix. The multivariate column reports regression coefficients with the R^2 in the last row. p-values are based on DSP with 1,000 permutations to construct the null distribution. Dependent and independent variables are standardized to unit variance, and coefficients are scaled by 100 for readability. We use *** to denote $p \leq 0.001$, ** to denote $p \leq 0.01$, and * to denote $p \leq 0.05$.

	Univariate		Multivariate
Linkage	β	R^2	β
Size	22.36***	0.0539	2.83***
Liquidity	16.18	0.0250	-0.62***
Ownership	9.54***	0.0079	-1.67
Patents	3.35***	0.0009	0.35***
SIC-4	2.99***	0.0007	0.30***
Analyst	2.91***	0.0007	0.19
City	1.98	0.0003	0.11
Leader	1.66***	0.0002	-0.17
Customer	0.27**	0.0000	-0.01
Supplier	0.27**	0.0000	0.01
Correlation	0.18***	0.0000	0.03***
R^2			0.1836

We estimate this regression using OLS. The network structure of the data implies a prominent dependence structure among observations, and this induces a bias in OLS standard errors for regression coefficients (Krackhardt, 1988; Dekker et al., 2007). To solve this problem, we use the double semi-partialing method of Dekker et al. (2007) to construct robust p-values for our coefficient estimates.[27]

We first study the association between transformer attention and each linkage in univariate versions of regression (27). Univariate regression coefficients are reported in the first

[27]The essence of this procedure is to randomly permute the data matrices and re-estimate the regression to generate a null distribution of regression coefficients. Because the permutation-based distribution accounts for network dependencies (as well as accounting for dependence among regressors through a "partialing" step), it is suitable for calculating robust p-values. The details of our implementation are described in Algorithm 3 of the Appendix. The data arrays used in this analysis are monthly time series of high-dimensional matrices. To help alleviate memory constraints when running regressions, we subsample three months per year over the 1970–2022 period and randomly draw 1,000 stocks within each selected month. For each predictor, we randomly permute the dyadic data matrices via node relabeling and re-estimate the regression 1,000 times to construct a null distribution of coefficients.

column of Table 9 and the R^2 is reported in the second column. p-values are computed based on Algorithm 3. In univariate regressions, 9 of 11 linkage matrices have significant explanatory power for the transformer attention matrix, indicating that the cross-asset information sharing pathways detected by the AIPM overlap with observable stock linkages documented in the literature.

The last column of Table 9 reports the multivariate regression, in which five linkage types remain significant determinants of transformer attention. These are shared size group, shared liquidity group, patent similarity, shared SIC industry, and return correlation. These results indicate that transformer attention indeed captures information in the stock linkages that have been studied in the literature. Collectively, observable linkages explain 18% of the variation in transformer attention.

Next, we investigate the extent to which cross-asset information sharing via observable linkages impacts portfolio performance, and how this compares with the AIPM. For our nonlinear transformer model, portfolio weights depend on cross-asset attention via the functional form:

$$w_t = A_t X_t \lambda \quad \text{and} \quad A_t = \sigma(X_t W X_t').$$

Meanwhile, the benchmark linear model BSV restricts A_t to be the identity matrix,

$$A_t = I,$$

meaning that there is no cross-asset attention.

Our linkage-based model is a middle ground between these two specifications. We specify cross-asset attention based on stock linkage data as follows:

$$A_t = I + b_1 L_{t,1} + \cdots + b_K L_{t,K}.$$

This model nests the BSV model (when $b_1 = \ldots = b_K = 0$) and offers an alternative attention mechanism to the one used by the transformer. We train the observable linkages model following the same protocol used to train all other models in the paper, and track this

Table 10: **Comparison of Linear (BSV), Linkages, and Transformer SDF Models**

Panel A reports out-of-sample SDF Sharpe ratios. Panel B performs pairwise model comparisons following the same structure as Table 2. SDF returns are rescaled to have 15% annualized volatility to aid comparability of alphas across models. We report alphas as annualized percentages along with t-statistics in parentheses. We use * to denote $p \leq 0.001$.

	Panel A: Sharpe Ratio		
Model	Full Sample	Pre-2002	Post-2002
BSV	3.61	5.22	2.14
Linkages	3.90	5.46	2.60
Transformer	4.78	6.62	3.46

	Panel B: Alpha (column vs. row)		
Model	BSV	Linkages	Transformer
BSV	-	36.10*	27.87*
t-stat.		(13.05)	(15.70)
Linkages	29.95*	-	40.73*
t-stat.	(10.38)		(15.17)
Transformer	-4.15	20.34*	-
t-stat.	(-1.98)	(6.69)	

new model's out-of-sample performance.

In Panel A of Table 10, we compare the performance of the nonlinear transformer, the observable linkages model, and the BSV model. We report performance over the full sample, as well as over the pre-2002 and post-2002 subsamples. We see that the portfolio performance afforded by cross-asset information sharing via observable linkages (Sharpe ratio 3.9) improves over the BSV model that excludes cross-asset information sharing (Sharpe ratio 3.6). The nonlinear transformer model, however, outperforms the linkage model by a fairly large margin (Sharpe ratio 4.8).[28]

In Panel B of Table 10, we also perform pairwise model comparisons of the BSV model, the observable linkages model, and our main nonlinear transformer model. SDF models are all normalized to 15% annualized volatility (as in the rest of the paper), and annualized alphas (and t-statistics) are computed from a time series regression of the column portfolio on the row portfolio. We see that the BSV model is spanned by the transformer (its alpha

[28]BSV and transformer performance statistics differ slightly from those in Table 1 of the main draft because the sample for the linkages model begins slightly later (1967 versus 1963).

is negative and marginally significant). The linkage model and transformer both have large and significant alpha versus one another (their pairwise correlation is 59%).

This mutual additivity of the linkages model and the transformer model is evidence of two effects. The first is that observed linkages and learned statistical attention reflect complementary forms of cross-asset information sharing. Some useful avenues of information sharing can be inferred through observable linkages. Observable linkages carry economic information that is not entirely learnable from returns and stock characteristics alone. But the observable avenues are not exhaustive. The transformer, on the other hand, detects information-sharing benefits from subtle relationships in return and characteristic data beyond what is captured in observable linkages. Neither of these avenues subsumes the other. Second, complementary benefits of observable linkages and statistical attention are further evidence of the "limits to learning" effect discussed in Section 4.4.5. In summary, the competitive performance of the linkages model supports our broader conclusion that cross-asset information sharing is important for constructing a better SDF.

4.5.5 Robustness to Estimation Windows

In this section, we test the robustness of our findings to different training windows and different test windows. In Figure 9, we evaluate the sensitivity of our results to using different rolling training windows of 60 (as in our main analysis), 120, or 360 months. In this analysis, the test sample is 1993–2022 to accommodate the 360-month initial training window (all other aspects of the performance comparison are unchanged). Regardless of the window used for model training, the transformer achieves a higher out-of-sample Sharpe ratio (shown in the top panel) and smaller pricing errors (bottom panel).

Next, we hold the rolling training window fixed at 60 months, and compare models in various testing windows that correspond to decade-long subsamples. These results are reported in Figure 10. The transformer model matches or outperforms BSV and DKKM in every decade. It outperforms MLP in all decades, but one (the 2010s), and it outperforms linear attention in all decades but one (the 1970s).

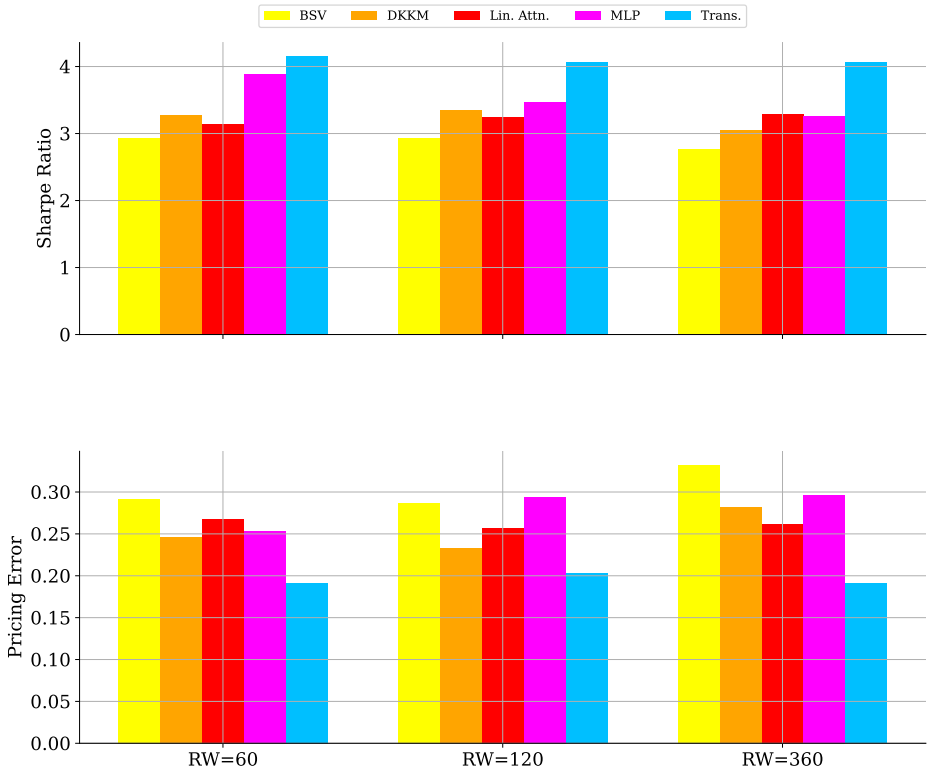

Figure 9: **Alternative Training Windows**

This figure reports HJD pricing errors for SDF models trained with rolling windows of 60, 120, or 360 months. The test sample is the same in all cases: 1993–2022. All other aspects of the analysis are the same as in the main analysis above (same set of JKP test assets, same stock universe, etc.).

4.5.6 Robustness to Point-in-time Data

Many of the characteristics in JKP were published in only the last decade or two. In this section, we perform robustness analyses to investigate the sensitivity of our findings to using "point-in-time" data. Specifically, in each month t, we only use a given JKP characteristic in our analysis if the original study was published by month t. There are two points to note. First, one can compare with traditional factor models such as FF6 or HXZ by cross-referencing our analysis here with earlier exhibits. However, this is a highly conservative comparison due to the fact that for most of the sample, these traditional models themselves use data that would not satisfy the point-in-time standard that we apply here. Second, it is worth noting that our main analysis (non-point-in-time) does not necessarily favor the transformer model in the sense that the machine learning benchmarks for comparison (e.g., DKKM and MLP) have equal access to non-point-in-time data.

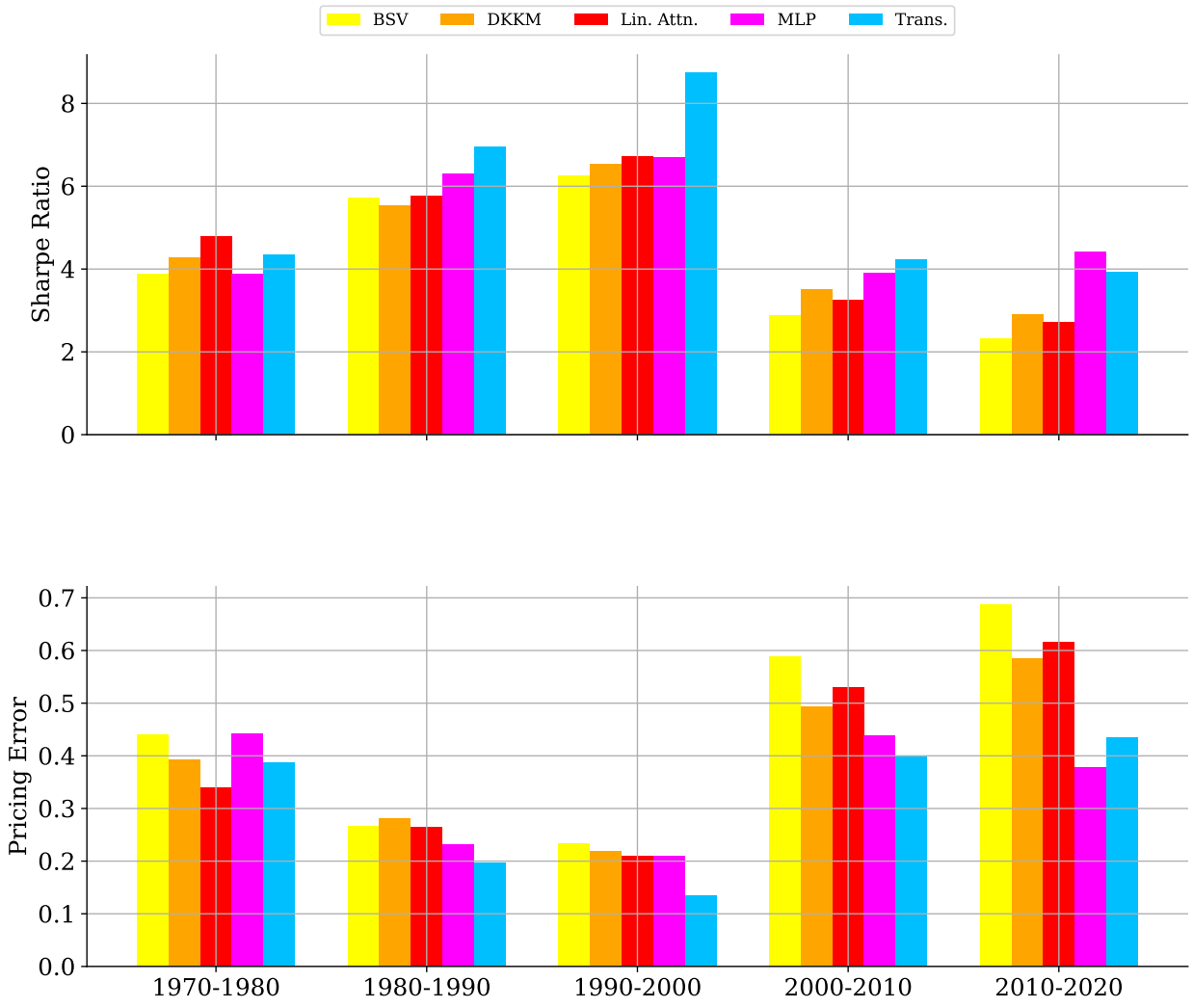

Figure 10: **Model Comparison by Decade**

This figure compares models in terms of Sharpe ratio (top panel) and HJD pricing error (bottom panel) by decade.

Table 11: **Point-in-time Portfolio Transformer Performance**

The table repeats the analysis of Table 1 using point-in-time JKP characteristics.

	BSV	**DKKM**	**Lin. Attn.**	**MLP**	**Trans.**
Sharpe Ratio	2.61	3.19	2.91	3.68	3.79
Pricing Error	0.28	0.22	0.26	0.20	0.19

Publication dates used to determine point-in-time data are also from JKP. We retrain our models each month using the set of published characteristics at that time. We began training in 1985 when we had 10 characteristics available. When computing pricing errors,

the test assets are also the subset of JKP factors that are available point-in-time. Table 11 reports the same statistics as Table 1 but restricted to the point-in-time analysis. Overall, the patterns in point-in-time data are in line with the findings from our main analysis. The superior performance of the nonlinear transformer model persists, though the margin of improvement is somewhat smaller than when using all historically available data.

5 Conclusion

This paper introduces the concept of an artificial intelligence pricing model (AIPM) by embedding transformer architectures in the stochastic discount factor. Leveraging the principles that have propelled advances in natural language processing—context awareness and extensive parameterization—we demonstrate how transformers can significantly enhance asset pricing models through cross-asset information sharing and nonlinearity.

Our first contribution is the development of the linear portfolio transformer, an interpretable model that incorporates the attention mechanism to facilitate information sharing across assets while maintaining analytical tractability. While more simplistic than the full nonlinear transformer, the linear attention model is a useful surrogate for understanding how attention enhances the SDF by capturing conditional relationships among assets. We show analytically that the attention-based SDF can be viewed as a dynamic combination of characteristic-managed factor portfolios.

Building on this foundation, our second contribution is to embed a full nonlinear transformer architecture in the SDF. It incorporates advanced features such as multi-head attention, softmax transformations, and stacking multiple transformer blocks, further exploiting the benefits of context awareness and model complexity. We describe the role played by each component in enhancing the model's predictive capabilities.

Empirically, we evaluate the performance of AIPMs using monthly US stock return data and a comprehensive set of 132 stock-level conditioning characteristics. The linear portfolio transformer demonstrates that even in a linear setting, cross-asset information sharing leads to substantially higher out-of-sample Sharpe ratios and lower pricing errors than attention-free models. The nonlinear portfolio transformer further amplifies these gains, outperforming

existing machine learning models by achieving higher Sharpe ratios, lower pricing errors, and significant alpha over benchmarks lacking cross-asset attention. Our findings reveal a clear hierarchy among models: as we incorporate more complex specifications and attention mechanisms, performance consistently improves.

References

Ali, Usman, and David Hirshleifer, 2020, Shared analyst coverage: Unifying momentum spillover effects, *Journal of Financial Economics* 136, 649–675.

Baba Yara, Fahiz, Brian Boyer, and Carter Davis, 2025, The limits of factor model spanning, *Working Paper* .

Bahdanau, Dzmitry, 2014, Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv:1409.0473* .

Bartlett, Peter L, Philip M Long, Gábor Lugosi, and Alexander Tsigler, 2020, Benign overfitting in linear regression, *Proceedings of the National Academy of Sciences* 117, 30063–30070.

Belkin, Mikhail, Daniel Hsu, Siyuan Ma, and Soumik Mandal, 2019a, Reconciling modern machine-learning practice and the classical bias–variance trade-off, *Proceedings of the National Academy of Sciences* 116, 15849–15854.

Belkin, Mikhail, Daniel Hsu, and Ji Xu, 2020, Two models of double descent for weak features, *SIAM Journal on Mathematics of Data Science* 2, 1167–1180.

Belkin, Mikhail, Alexander Rakhlin, and Alexandre B Tsybakov, 2019b, Does data interpolation contradict statistical optimality?, in *The 22nd International Conference on Artificial Intelligence and Statistics*, 1611–1619, PMLR.

Bolya, Daniel, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, and Judy Hoffman, 2022, Hydra attention: Efficient attention with many heads, in *European Conference on Computer Vision*, 35–49, Springer.

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

Britten-Jones, Mark, 1999, The sampling error in estimates of mean-variance efficient portfolio weights, *The Journal of Finance* 54, 655–671.

Bryzgalova, Svetlana, Markus Pelger, and Jason Zhu, 2025, Forest through the trees: Building cross-sections of stock returns, *The Journal of Finance* 80, 2447–2506.

Chen, Andrew Y, and Tom Zimmermann, 2022, Open source cross-sectional asset pricing, *Critical Finance Review* 11, 207–264.

Chen, Luyang, Markus Pelger, and Jason Zhu, 2023, Deep learning in asset pricing, *Management Science* .

Chen, Zhimin, Bryan Kelly, and Semyon Malamud, 2025, Limits to (machine) learning, *Working Paper* .

Cho, Kyunghyun, Bart Van Merriënboer, Çağlar Gulçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, 2014, Learning phrase representations using rnn encoder–decoder for statistical machine translation, in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1724–1734.

Cohen, Lauren, and Andrea Frazzini, 2008, Economic links and predictable returns, *The Journal of Finance* 63, 1977–2011.

Cong, Lin William, Guanhao Feng, Jingyu He, and Xin He, 2025, Growing the efficient frontier on panel trees, *Journal of Financial Economics* 167, 104024.

Cong, Lin William, Ke Tang, Jingyuan Wang, and Yang Zhang, 2021, Alphaportfolio: Direct construction through deep reinforcement learning and interpretable ai, *SSRN Electronic Journal. https://doi. org/10.2139/ssrn* 3554486.

Connor, Gregory, Matthias Hagmann, and Oliver Linton, 2012, Efficient semiparametric estimation of the fama–french model and extensions, *Econometrica* 80, 713–754.

Daniel, Kent, David Hirshleifer, and Lin Sun, 2020, Short-and long-horizon behavioral factors, *The review of financial studies* 33, 1673–1736.

Dekker, David, David Krackhardt, and Tom AB Snijders, 2007, Sensitivity of mrqap tests to collinearity and autocorrelation conditions, *Psychometrika* 72, 563–581.

Didisheim, Antoine, Shikun Barry Ke, Bryan T Kelly, and Semyon Malamud, 2024, Apt or aipt: The surprising dominance of large factor models, Technical report, National Bureau of Economic Research.

Dong, Yihe, Jean-Baptiste Cordonnier, and Andreas Loukas, 2021, Attention is not all you need: Pure attention loses rank doubly exponentially with depth, in *International Conference on Machine Learning*, 2793–2803, PMLR.

Ehsani, Sina, and Juhani T Linnainmaa, 2022, Factor momentum and the momentum factor, *The Journal of Finance* 77, 1877–1919.

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

Fan, Jianqing, Zheng Tracy Ke, Yuan Liao, and Andreas Neuhierl, 2022, Structural deep learning in conditional asset pricing, *Available at SSRN 4117882* .

Fan, Jianqing, Yuan Liao, and Weichen Wang, 2016, Projected principal component analysis in factor models, *Annals of statistics* 44, 219.

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

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

Gabaix, Xavier, Ralph SJ Koijen, Robert J Richmond, and Motohiro Yogo, 2025, Asset embeddings, Technical report, National Bureau of Economic Research.

Giglio, Stefano, Bryan Kelly, and Dacheng Xiu, 2022, Factor models, machine learning, and asset pricing, *Annual Review of Financial Economics* 14, 337–368.

Giglio, Stefano, and Dacheng Xiu, 2021, Asset pricing with omitted factors, *Journal of Political Economy* 129, 1947–1990.

Gu, Shihao, Bryan Kelly, and Dacheng Xiu, 2020, Autoencoder asset pricing models, *Journal of Econometrics* .

Guijarro-Ordonez, Jorge, Markus Pelger, and Greg Zanotti, 2025, Deep learning statistical arbitrage, *Management Science* .

Gupta, Tarun, and Bryan Kelly, 2019, Factor momentum everywhere, *The Journal of Portfolio Management* 45, 13–36.

Haddad, Valentin, Serhiy Kozak, and Shrihari Santosh, 2020, Factor timing, *The Review of Financial Studies* 33, 1980–2018.

Hansen, Lars Peter, and Ravi Jagannathan, 1997, Assessing specification errors in stochastic discount factor models, *The Journal of Finance* 52, 557–590.

Hansen, Lars Peter, and Scott F Richard, 1987, The role of conditioning information in deducing testable restrictions implied by dynamic asset pricing models, *Econometrica: Journal of the Econometric Society* 587–613.

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

He, Ai, Dashan Huang, Jiaen Li, and Guofu Zhou, 2023, Shrinking factor dimension: A reduced-rank approach, *Management science* 69, 5501–5522.

He, Songrun, Ming Yuan, and Guofu Zhou, 2022, Principal portfolios: The multi-signal case, *Available at SSRN 4245333* .

Hou, Kewei, Haitao Mo, Chen Xue, and Lu Zhang, 2021, An augmented q-factor model with expected growth, *Review of Finance* 25, 1–41.

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

Hou, Kewei, Chen Xue, and Lu Zhang, 2020, Replicating anomalies, *The Review of Financial Studies* 33, 2019–2133.

Jacot, Arthur, Franck Gabriel, and Clément Hongler, 2018, Neural tangent kernel: Convergence and generalization in neural networks, *Advances in neural information processing systems* 31.

Jensen, Theis Ingerslev, Bryan Kelly, and Lasse Heje Pedersen, 2023, Is there a replication crisis in finance?, *The Journal of Finance* 78, 2465–2518.

Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei, 2020, Scaling laws for neural language models, *arXiv preprint arXiv:2001.08361* .

Kelly, Bryan, Boris Kuznetsov, Semyon Malamud, , and Yuan Zhang, 2026, Large and deep factor models, *arXiv preprint arXiv:2402.06635* .

Kelly, Bryan, Semyon Malamud, and Lasse Heje Pedersen, 2023, Principal portfolios, *The Journal of Finance* 78, 347–387.

Kelly, Bryan, Semyon Malamud, and Kangying Zhou, 2024, The virtue of complexity in return prediction, *The Journal of Finance* 79, 459–503.

Kelly, Bryan, Seth Pruitt, and Yinan Su, 2020a, Characteristics are covariances: A unified model of risk and return, *Journal of Financial Economics* .

Kelly, Bryan, Seth Pruitt, and Yinan Su, 2020b, Instrumented principal component analysis, *Working paper* .

Kelly, Bryan, and Dacheng Xiu, 2023, Financial machine learning, *Foundations and Trends® in Finance* 13, 205–363.

Kelly, Bryan T, and Semyon Malamud, 2025, Understanding the virtue of complexity, *Yale Working Paper* .

Kingma, Diederik P, and Jimmy Ba, 2014, Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980* .

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

Krackhardt, David, 1988, Predicting with networks: Nonparametric multiple regression analysis of dyadic data, *Social networks* 10, 359–381.

Lee, Charles MC, Stephen Teng Sun, Rongfei Wang, and Ran Zhang, 2019, Technological links and predictable returns, *Journal of Financial Economics* 132, 76–96.

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.

Liu, Yukun, and Toby Moskowitz, 2025, Economic correlations, *Yale Working Paper* .

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

Minaee, Shervin, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao, 2024, Large language models: A survey, *arXiv preprint arXiv:2402.06196* .

Orhan, Emin A, and Xaq Pitkow, 2017, Skip connections eliminate singularities, *arXiv preprint arXiv:1701.09175* .

Preite, Massimo Dello, Raman Uppal, Paolo Zaffaroni, and Irina Zviadadze, 2022, What is missing in asset-pricing factor models?

Rapach, David E, and Guofu Zhou, 2020, Time-series and cross-sectional stock return forecasting: New machine learning methods, *Machine learning for asset management: New developments and financial applications* 1–33.

Spigler, Stefano, Mario Geiger, Stéphane d'Ascoli, Levent Sagun, Giulio Biroli, and Matthieu Wyart, 2019, A jamming transition from under-to over-parametrization affects generalization in deep learning, *Journal of Physics A: Mathematical and Theoretical* 52, 474001.

Stambaugh, Robert F, and Yu Yuan, 2017, Mispricing factors, *The review of financial studies* 30, 1270–1315.

Tarzanagh, Davoud Ataee, Yingcong Li, Christos Thrampoulidis, and Samet Oymak, 2023, Transformers as support vector machines, Preliminary version at NeurIPS M3L Workshop, 2023.

Timmermann, Allan, and Luka Vulicevic, 2026, Compute, complexity, and the scaling laws of return predictability, *Available at SSRN 6105327* .

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, 2017, Attention is all you need, *Advances in neural information processing systems* 30.

A Theory of Linear Transformers

Let

$$L^{linear}(X_t) \;=\; X_t W' X_t' \tag{28}$$

be a linear attention matrix. Then, we have:

$$\begin{aligned}
\tilde{F}^{linear}_{k,t+1} &= X_t(k)'\, L^{linear}(X_t)\, R_{t+1} \\
&= \; X_t(k)'\, (X_t W' X_t') R_{t+1} \\
&= \; (X_t(k)'[X_t(1), \cdots, X_t(D)]) W' ([X_t(1), \cdots, X_t(D)]' R_{t+1}) \\
&= \; [X_t(k)' X_t(1), \cdots, X_t(k)' X_t(D)]\, W' \begin{pmatrix} F_{1,t+1} \\ \vdots \\ F_{D,t+1} \end{pmatrix} \\
&= \; \sum_{k_1=1}^{D} \underbrace{\Theta_t(k_1)}_{timing\ \ variables} \quad F_{k_1,t+1},
\end{aligned} \tag{29}$$

where we have defined

$$F_{k,t+1} \;=\; X_t(k)' R_{t+1}\,. \tag{30}$$

The formula (29) uncovers an important algebraic identity: Although the attention-based factor $\tilde{F}^{linear}_{k,t+1}$ is formally based on $X_t(k)$, it is actually a *factor-timing portfolio,* combining all D factors together, with timing variables that depend on $X_t(k)$.[29] That is, it is not attention that is helping the factor characteristics. It is the factor characteristics that help attention; attention is the primary channel of alpha generation.

A.1 Selecting Attention Heads with OLS and MSRR Objectives

We consider a generalization of the basic linear transformer from (8). To this end, we assume that there are two types of features, $X_t \in \mathbb{R}^{N_t \times D}$ and $Z_t \in \mathbb{R}^{N_t \times d}$. The first type of features

[29] Namely, $\Theta_t(k_1) = \sum_{k_2=1}^{D} X_t(k)' X_t(k_2) W(k_1, k_2)$.

is used to construct attention matrices. The second one is used to construct features. Then, we define

$$\begin{aligned} \text{LMHA}(X_t) &= N_t^{-1} \underbrace{(X_t W_1 X_t') Z_t \lambda_1}_{head\ \#1} + \cdots + N_t^{-1} \underbrace{(X_t W_H X_t') Z_t \lambda_H}_{head\ \#H} \\ &= N_t^{-1} \left(\text{vec}(X_t' Z_t)' \otimes X_t \right) \text{vec} \left(\sum_{h=1}^{H} \lambda_h' \otimes W_h \right) = S_t \check{\lambda} \end{aligned} \tag{31}$$

to be the Linear Multi-Head Attention (LMHA) portfolio, where we have

$$S_t = N_t^{-1} \text{vec}(X_t' Z_t)' \otimes X_t \,. \tag{32}$$

We consider a ridge-penalized version of the portfolio optimization problem:

$$MSRR: \min_{\{W_h, \lambda_h\}_{h=1}^H} \left\{ \frac{1}{T} \sum_{t=1}^{T} (1 - R_{t+1}' \text{LMHA}(X_t))^2 + z \|\check{\lambda}\|^2 \right\}. \tag{33}$$

Our next key observation is that the nonlinearity of the optimal LMHA problem arises from its finite head structure. Indeed, a simple dimension counting implies that the set of $\check{\lambda} \in \mathbb{R}^{D^2 d}$ that can be represented as an H-head model (31) forms a manifold of dimension $(D^2 + d)H$. If the number of heads is sufficiently large (e.g., $H \geq \min(D^2, d)$), this set coincides with $\mathbb{R}^{D^2 d}$ and, hence, we can simply optimize (33) directly over all $\check{\lambda} \in \mathbb{R}^{D^2 d}$; the number of heads H plays no role for such "infinite heads" models.[30] We refer to the corresponding vector $\check{\lambda}$ solving (33) as $\check{\lambda}_{MSRR}(H)$. We abuse notation and use $\check{\lambda}_{MSRR}(\infty)$ to denote $\check{\lambda}_{MSRR}(H)$ for all $H \geq \min(D^2, d)$. The following is true.

Lemma 2 *Suppose that $H \geq \min(D^2, d)$. Let $S = (S_t)_{t=1}^T \in \mathbb{R}^{(\sum_t N_t) \times D^2 d}$. Then, the solution $\check{\lambda}_{MSRR}(H)$ to problem (33) is given by*

$$\check{\lambda}_{MSRR}(\infty) = (T^{-1} S' R R' S + z I_{D^2 d \times D^2 d})^{-1} T^{-1} S' R. \tag{34}$$

Lemma 2 shows that the optimal LMHA $\check{\lambda}$ can be found in closed-form. However, such "infinite head" models lose the attractive interpretability of the finite-head models and make

[30] They are also known as "hydra attention" models in the literature; see, Bolya et al. (2022).

it difficult to understand how exactly the model learns the cross-predictability patterns among the different stocks. Most importantly, the infinite head $\check{\lambda}$ does not admit a unique decomposition into the single-attention-head components. For example, any decomposition $\lambda_h = \lambda_h^1 + \lambda_h^2$ leads to a different decomposition of $\check{\lambda}$. However, it is possible to characterize the optimal multi-head attention policy $\check{\lambda}_{MSRR}(H)$ directly in terms of $\check{\lambda}_{MSRR}(\infty)$. Furthermore, the decomposition we construct is unique, with the attention heads being pairwise orthogonal and ordered according to their importance.

To derive such an orthogonal multi-head decomposition, we start by noting that $\check{\lambda}(\infty) \in \mathbb{R}^{D^2 d}$ can be "reshaped" as a $(D^2) \times d$ matrix. We abuse notation and use $\check{\lambda}$ to denote this reshaped vector. We can then use the singular value decomposition and write

$$\check{\lambda}_{MSRR}(\infty) = \sum_{h=1}^{d} W_h \otimes \lambda_h,\ W_h \ \in \ \mathbb{R}^{D \times D},\ \lambda_h \in \mathbb{R}^d, \tag{35}$$

where $W_h \in \mathbb{R}^{D \times D}$ are D^2-dimensional eigenvectors of the matrix $\check{\lambda}(\infty)\check{\lambda}(\infty)' \in \mathbb{R}^{(D^2)\times(D^2)}$, while $\lambda_h \in \mathbb{R}^d$ are the d-dimensional eigenvectors of the matrix $\check{\lambda}(\infty)'\check{\lambda}(\infty) \in \mathbb{R}^{d \times d}$, with the singular values ordered in the decreasing order. We use

$$\check{\lambda}_{MSRR}(\infty; H) = \sum_{h=1}^{H} W_h \otimes \lambda_h \tag{36}$$

to denote the approximation of $\check{\lambda}(\infty)$ with the top H attention heads based on the singular value decomposition. By direct calculation and standard properties of the singular value decomposition, we have

$$\check{\lambda}_{MSRR}(H) = \arg\min_{\check{\lambda}(H)} \|(\check{\lambda}_{MSRR}(\infty) - \check{\lambda}(H))\|^2 . \tag{37}$$

As we now show, the true optimal H-head model solves an intuitive optimization problem. Indeed, by the properties of ordinary least squares regression, we get

$$\|1 - R'S\check{\lambda}\|^2 = \|1 - R'S\check{\lambda}_{MSRR}\|^2 \ + \ \|R'S(\check{\lambda}_{MSRR} - \check{\lambda})\|^2 . \tag{38}$$

These identities immediately imply the following result.

Proposition 1 *For any* $H < \min(D^2, d)$,

$$\check{\lambda}_{MSRR}(H) = \arg\min_{\check{\lambda}(H)} \|R'S(\check{\lambda}_{MSRR}(\infty) - \check{\lambda}(H))\|^2 \tag{39}$$

where the minimum is over all H*-heads* $\check{\lambda}(H)$*. Thus, if* $S'RR'S = I$*, then,* $\check{\lambda}_{MSRR}(\infty; H)$ *is the solution to* (39) *.*

Proposition 1 establishes a striking connection between finite-head LMHA models and singular value decomposition. For example, W_1 is the "top" attention head, capturing the largest amount of cross-predictability. We refer to W_1 as the *principal attention head*. This attention head is closely related to principal portfolios introduced in Kelly et al. (2023). Namely, Kelly et al. (2023) show how to find the optimal attention matrix L for signals s_t, building a portfolio Ls_t using a singular value decomposition. Here, we show how to find the optimal L of the form $L = X_t W X_t'$ with signals $s_t = Z_t\lambda$.

While for generic S the problem (39) does not admit a closed-form solution, we believe the structure of these solutions is similar to that described in Proposition 1. The rotationally symmetric case of Proposition 1 implies that the problem in (39) has a tremendous number of local extrema: Any collection of H different singular values corresponds to a local extremum, while the global minimum is unique, given by the H largest singular values.

A.2 Computing Optimal Attention Heads

Define the timing variables $\Xi_t = N_t^{-1} X_t' Z_t \in \mathbb{R}^{D\times d}$, so that the signal matrix (32) can be written as $S_t = \text{vec}(\Xi_t)' \otimes X_t$. Any LMHA policy then admits a representation $\check{\lambda} = \sum_{t=1}^{T} q_t \otimes \Xi_t$ for some coefficient vectors $q_t \in \mathbb{R}^D$. We write $\pi_t = S_t\check{\lambda}$ for the resulting portfolio weights, and $\pi_t(H)$ for those of its H-head truncation $\check{\lambda}(\infty; H)$. The following is true.

Theorem 2 (Computing the Multi-Head Decomposition) *Let*

$$\check{\lambda} = \sum_t q_t \otimes \Xi_t \,. \tag{40}$$

be an LMHA model. Define $\Sigma^q \in \mathbb{R}^{T\times T}$ *via* $\Sigma^q(t_1,t_2) = q'_{t_1}q_{t_2}$*, and recall that* $\Xi_t \in \mathbb{R}^{D\times d}$*. Then,*

$$\check{\lambda}'\check{\lambda} = \sum_{t_1,t_2} \Sigma^q(t_1,t_2)\Xi'_{t_1}\Xi_{t_2} \ \in \mathbb{R}^{d\times d}\,. \tag{41}$$

Let

$$\check{\lambda}'\check{\lambda} = \sum_{h=1}^{d} \lambda_h\lambda'_h \tag{42}$$

be its spectral (eigenvalue) decomposition. Then, the optimal multi-head attention is given by

$$\check{\lambda}(\infty;H) = \sum_{h=1}^{H} W_h \otimes \lambda_h, \tag{43}$$

with the optimal attention heads given by

$$W_h = \check{\lambda}\lambda_h\,. \tag{44}$$

Defining $\hat{b}(H) \in \mathbb{R}^{d\times H}$ *to be the first* H *singular vectors* $[\lambda_1,\cdots,\lambda_H]$ *and*

$$\tilde{\Xi}_t(H) = \Xi_t(\hat{b}(H)\hat{b}(H)'), \tag{45}$$

we get that

$$\pi_t(H)'R_{t+1} = \sum_{\tau}(F'_{t+1}q_\tau) \ \underbrace{(\Xi'_t\tilde{\Xi}_\tau(H))}_{timing\ \ attention} \tag{46}$$

B Model Optimization

B.1 Linear Attention

Algorithm 1 Calculating return on the linear attention portfolio at time $t+1$

Require: Covariances $\Xi_\tau \in \mathbb{R}^{D\times d}$ and linear factors $F_{\tau+1} \in \mathbb{R}^D$ for $\tau = t-N_{rol}+1, ..., t-1$.

1: Compute matrices $\Sigma^F = (F'_{\tau_1+1}F_{\tau_2+1})_{\tau_1,\tau_2=1}^{N_{rol}} \in \mathbb{R}^{N_{rol}\times N_{rol}}$ and $\Sigma^\Xi = (\Xi'_{\tau_1}\Xi_{\tau_2})_{\tau_1,\tau_2=1}^{N_{rol}} \in \mathbb{R}^{N_{rol}\times N_{rol}}$.

2: Find the vector of eigenvalues $\Lambda \in \mathbb{R}^{N_{rol}}$ and the matrix of eigenvectors $v \in \mathbb{R}^{N_{rol}\times N_{rol}}$ of $\Sigma^F \circ \Sigma^\Xi \in \mathbb{R}^{N_{rol}\times N_{rol}}$.

3: Divide each eigenvector v_k, $k = 1, ..., N_{rol}$ by the square root of the corresponding eigenvalue $\sqrt{\lambda_k}$. This lets us recover the normalised eigenvectors V_k, $k = 1, ..., N_{rol}$, of the parent matrix $\tilde{S}'\tilde{S} = V\,\mathrm{diag}(\Lambda)V'$ using the formula $V_k = \sum_{\tau=1}^{N_{rol}} v_{k,\tau}F_{\tau+1} \otimes \Xi_\tau$. Notice that $v_{k,\tau}$s are scalars.

4: Compute vector $Q = \Lambda \circ (v'\mathbf{1}_{N_{rol}\times 1}) \in \mathbb{R}^{N_{rol}\times 1}$.

5: For each $z_k \in z \in \mathbb{R}^{N_z}$, compute vectors $\tilde{Q}_k = \left((z_k\mathbf{1}_{N_{rol}\times 1} + \Lambda)^{-1} - z_k^{-1}\right) \circ Q$. Then, stack $\tilde{Q}_k \in \mathbb{R}^{N_{rol}\times 1}$ by columns, getting $\tilde{Q} \in \mathbb{R}^{N_{rol}\times N_z}$. This way, we set the ground for fitting variables of interest (M^*, $W^*(M)$ and $b^*(z, M)$) simultaneously for a grid of z_k.

6: Now, denote $v_\tau \in \mathbb{R}^{N_{rol}}$ the τ-th *row* of the normalised eigenvector matrix $v \in \mathbb{R}^{N_{rol}\times N_{rol}}$, there are N_{rol} such v_τs. Define the outer products $\tilde{v}_\tau := F_{\tau+1}v'_\tau \in \mathbb{R}^{D\times N_{rol}}$.

7: **for** $\tau \in t-N_{rol}, ..., t-1$ **do**

8: Compute $h_{k,\tau} = z_k^{-1}F_{\tau+1} \in \mathbb{R}^D$ for the grid of z_k and stack $h_{k,\tau}$ by columns into $h_\tau \in \mathbb{R}^{D\times N_z}$, then compute $q_\tau := h_\tau + \tilde{v}_\tau\tilde{Q} \in \mathbb{R}^{D\times N_z}$.

9: Normalise q_τ by N_{rol}, i.e., do $q_\tau = \frac{q_\tau}{N_{rol}}$. This normalises the matrix M^* (we cannot normalise M^* directly because we do not compute M^* explicitly).

10: **end for**

11: **for** $z_k \in z \in \mathbb{R}^{N_z}$ **do**

12: Now that we have $q_{k,\tau} \in \mathbb{R}^D$ for all $z_k \in z$ and all $\tau = t-N_{rol}, ..., t-1$, i.e., we have a 3-dimensional tensor $q \in \mathbb{R}^{D\times N_z\times N_{rol}}$, we split it across z dimension into N_z vectors $q_k \in \mathbb{R}^{D\times N_{rol}}$. Compute $\Sigma_k^q = q'_k q_k \in \mathbb{R}^{N_{rol}\times N_{rol}}$.

13: Compute the inner product of M^*s without explicitly computing M^*: $W_k^{*'}W_k^* = \sum_{\tau_1,\tau_2=t-N_{rol}}^{t-1} \Sigma_{k,\tau_1\tau_2}^q\, \Xi_{\tau_1}^{D\times d'}\, \Xi_{\tau_2}^{D\times d}$. Here, $\Xi_{\tau_1}^{D\times d'}\Xi_{\tau_2}^{D\times d} \in \mathbb{R}^{d\times d}$ and $\Sigma_{k,\tau_1\tau_2}^q$ is a scalar.

14: Compute eigenvector matrix $m_k \in \mathbb{R}^{d\times d}$ of $W_k^{*'}W_k^* \in \mathbb{R}^{d\times d}$. Each column of m corresponds to some attention head.

15: **end for**

16: **for** $h \in [1, 2, 3, 4, 8, 16, 32...]$ **and** $z_k \in z \in \mathbb{R}^{N_z}$ **do**

17: Denote $m_{k,h} \in \mathbb{R}^{d,h}$ the matrix of eigenvectors m_k with h columns that correspond to the largest eigenvalues of $W_k^{*'}W_k^*$. Compute $\tilde{\Xi}_{\tau,k,h} = \Xi_\tau^{D\times d}m_{k,h}m'_{k,h} \in \mathbb{R}^{D\times d}$.

18: Compute the $t+1$ out-of-sample return $R_{t+1,k,h}^{pf}$ on the Linear MSRR attention portfolio $R_{t+1,k,h}^{pf} = R_{t+1}\tilde{S}_{t+1}M_{k,h}^* = \sum_{\tau=t-N_{rol}}^{t-1}(\Xi'_t\tilde{\Xi}_{\tau,k,h})(F'_{t+1}q_{k,\tau})$, where $\Xi'_t\tilde{\Xi}_{\tau,k,h}$ and $F'_{t+1}q_{k,\tau}$ are scalars.

19: **end for**

20: We end up with $N_z \cdot N_h$ out-of-sample returns on linear attention portfolio $R_{t+1,k,h}^{pf}$ for a grid of penalty parameters z_k and number of attention heads h.

B.2 Nonlinear Portfolio Transformer Training

Algorithm 2 Nonlinear transformer training

Require: Panel data of stock characteristics $X_t \in \mathbb{R}^{N_t \times D}$, returns $R_{t+1} \in \mathbb{R}^{N_t}$, rolling window period τ, a learning rate η, number of epochs e, and a nonlinear transformer model $\mathcal{T}^{(K)}$, which is composed of K transformer-blocks. Each transformer-block $\mathcal{T}^{(k)}$ is composed of a multi-head attention unit $\mathcal{A}$, a feed-forward network $\mathcal{F}$, and an objective function $\in \{\text{MSRR}, \text{MSE}\}$.

Denote by $\Theta_{\mathcal{T}}$ the set of learnable parameters.
for epoch $\in e$ **do**
  for month $\in \tau$ **do**
    if loss == MSRR **then**
      $\mathcal{L}_\Theta \leftarrow (1 - \lambda' \mathcal{T}^{(K)}(X_t)' R_{t+1})^2$.
    else
      $\mathcal{L}_\Theta \leftarrow \|R_{t+1} - \mathcal{T}^{(K)}(X_t)\lambda\|^2$.
    end if
    $\Theta_{s+1} \leftarrow \Theta_s - \eta \nabla \mathcal{L}$.
  end for
end for

B.3 Network Matrix Regression with Double Semi-Partialing (DSP)

Algorithm 3 Double Semi-Partialing (DSP)

Require: For each period $t \in \{1, \ldots, T\}$, let $A_t \in \mathbb{R}^{N_t \times N_t}$ denote the learned attention matrix, where N_t is the number of firms at time t. For each linkage type $k \in \{1, \ldots, K\}$, let $L_{t-1,k} \in \mathbb{R}^{N_t \times N_t}$ denote a linkage matrix capturing pairwise firm relationships observed at time $t-1$. We assume a common number of firms across periods (achieved by subsampling) and write N for the length of each $\mathrm{vec}(A_t)$.

Let $y = [\mathrm{vec}(A_1), \ldots, \mathrm{vec}(A_T)]^\top \in \mathbb{R}^{TN}$.
Let $X_k = [\mathrm{vec}(L_{0,k}), \ldots, \mathrm{vec}(L_{T-1,k})]^\top \in \mathbb{R}^{TN}$.
Let $X = [X_1, \ldots, X_K] \in \mathbb{R}^{TN \times K}$.
Let $\rho(\cdot)$ be a permutation function.
for $k = 1, \ldots, K$ **do**
    Let $Z_k = [\mathbf{1}_{TN},\ X_1, \ldots, X_{k-1}, X_{k+1}, \ldots, X_K] \in \mathbb{R}^{TN \times K}$.
    Regress X_k on Z_k and compute the residualized focal predictor

$$\hat{\eta}_k = (Z_k^\top Z_k)^{-1} Z_k^\top X_k, \qquad R_k = X_k - Z_k \hat{\eta}_k.$$

    Regress y on $[Z_k,\ R_k]$ and store the observed coefficient $\hat{\beta}_k$ and observed test statistic s_k for the column R_k.
    for $i = 1, \ldots, B$ **do**
        Let $\widetilde{R}_{i,k} = \rho_i(R_k)$
        Regress y on $[Z_k,\ \widetilde{R}_{i,k}]$ and store the permuted coefficient $\widetilde{\beta}_{i,k}$ and permuted test statistic $\widetilde{s}_{i,k}$ for the column $\widetilde{R}_{i,k}$.
    end for
    Compute the two-sided permutation p-value:

$$p_k = \frac{1}{B} \sum_{i=1}^{B} \mathbf{1}(|\widetilde{s}_{i,k}| \geq |s_k|)\,.$$

end for
Report $\{\hat{\beta}_k,\ s_k,\ p_k\}_{k=1}^K$.