Deep Learning in Characteristics-Sorted Factor Models[*]

Guanhao Feng[†]
College of Business
City University of Hong Kong

Nicholas G. Polson [‡]
Booth School of Business
University of Chicago

Jianeng Xu[§]
Booth School of Business
University of Chicago

March 24, 2020

Abstract

To study the characteristics-sorted factor model in asset pricing, we develop a bottom-up approach with state-of-the-art deep learning optimization. With an economic objective to minimize pricing errors, we train a non-reduced-form neural network using firm characteristics *[inputs]*, and generate factors *[intermediate features]*, to fit security returns *[outputs]*. Sorting securities on firm characteristics provides a nonlinear activation to create long-short portfolio weights, as a hidden layer, from lag characteristics to realized returns. Our model offers an alternative perspective for dimension reduction on firm characteristics *[inputs]*, rather than factors *[intermediate features]*, and allows for both nonlinearity and interactions on inputs. Our empirical findings are twofold. We find robust statistical and economic evidence in evaluating various portfolios and individual stock returns. Finally, we show highly significant results in factor investing, improvement in dissecting anomalies, as well as the volatility exposures in deep characteristics.

Key Words: Alpha, Characteristics-Sorted Factor Models, Cross-Sectional Return, Deep Learning, Firm Characteristics, Machine Learning, Pricing Errors.

[*]The paper was previously circulated under the title "Deep Learning Factor Alpha" and "Deep Learning in Asset Pricing". We appreciate insightful comments from Will Cong, Serge Darolles, Victor DeMiguel, Li Deng, Jin-Chuan Duan, Shihao Gu (discussant), Bryan Kelly, Soohun Kim (discussant), Markus Pelger, Weidong Tian (discussant), Dacheng Xiu, and Chu Zhang. We are also grateful to helpful comments from seminar and conference participants at Boston University, CUHK, CityU HK, Jinan University, SHUFE, SUSTech, ESSEC Business School, China International Conference in Finance, CQAsia Conference, EcoSta, Informs Annual Conference, PanAgora, SOFIE Annual Conference, Schroders, Wolfe Research Conference, Unigestion Factor Investing Conference, Autumn Seminar of Inquire Europe, Australian Finance&Banking Conference, and New Zealand Finance Meeting. We acknowledge research awards from Unigestion Alternative Risk Premia Research Academy, PanAgora, as well as INQUIRE Europe. Feng acknowledges the ECS grant from Hong Kong Research Grants Council.

[†]Address: 83 Tat Chee Avenue, Kowloon Tong, Hong Kong. E-mail address: gavin.feng@cityu.edu.hk.

[‡]Address: 5807 S Woodlawn Avenue, Chicago, IL 60637, USA. E-mail address: ngp@chicagobooth.edu.

[§]Address: 5807 S Woodlawn Avenue, Chicago, IL 60637, USA. E-mail address: jianeng@uchicago.edu.

1 Introduction

Asset pricing models study why different assets earn different expected returns. According to ICAPM of Merton (1973), a combination of common factors captures the cross section of expected returns, and the regression intercept should be zero. Unlike the regular objective function in statistics and machine learning, the model fitness for asset pricing is not about the explained variation in time series, but the magnitude of intercepts, alphas, in the cross-section. This non-arbitrage restriction on alphas implies that simply adding factors leads to statistical overfitting (increase time series R^2) but does not cause economical overfitting (decrease intercepts).

In empirical asset pricing studies, researchers typically sort securities on firm characteristics and create long-short portfolios as common risk factors to build asset pricing models. For example, the Nobel prize research of Fama and French (1993) add SMB (small-minus-big) and HML (high-minus-low) to CAPM. However, almost all proposed factor models have rejected the zero-alpha hypothesis. We want to approach this puzzle, with a machine learning perspective, as an optimization problem: How does one construct a factor model to minimize pricing errors or alphas?

A rising literature applies machine learning methods in the field of asset pricing. For the stochastic discount factor (SDF) model, Kozak et al. (2019) provide a shrinkage approach to model fitness, and Feng et al. (2019) test new factors through model selections. In dimension reduction through principal components (PCA), Kelly et al. (2019) employ characteristics as instruments, and Lettau and Pelger (2018) provide a regularized estimation for risk premia. We follow their research directions and provide a deep learning framework of the SDF model with dimension reduction.

The goal of our paper is to investigate the underlying mechanism of the characteristics-sorted factor models, which includes sorting securities, generating factors, and fitting the cross section of security returns. We define an economic-driven objective function, pricing errors, for the cutting-edge technology of deep learning optimization. We show the characteristics-sorted factor models can be dissembled as a feed-forwrad neural network:

(1) *Inputs* are firm characteristics. The neural network starts from sorting securities on firm characteristics, which is a nonlinear activation to create long-short portfolio weights.

(2) *Intermediate features* are risk factors. The factors are linear activations (long-short portfolio

weights) on realized returns from the sorting directions.

(3) *Outputs* are security returns. Minimizing an economic objective function is equivalent to minimizing pricing errors for fitting the factor model to portfolio or individual stock returns.

Distinct from the SDF literature on machine learning, to be clear, we focus on training a factor model rather than testing a factor or characteristic. Apart from the PCA literature, our innovation is to apply dimension reduction on firm characteristics [inputs] rather than factors [intermediate features]. We argue the current literature is mostly about intermediate features and outputs (security returns), whereas ours illustrates the complete channel between inputs and outputs. We adopt a non-reduced-form neural network and develop such a bottom-up approach that includes security sorting, factor generation, and fitting the cross-section of security returns. The Fama-French-type characteristics-sorted factor models can be shown as "shallow" learning models. Also, we show our built-in sorting function generalizes the rank-weight approach in Frazzini and Pedersen (2014) for creating their betting-against-beta factor.

A standard asset pricing test is to perform a GRS-type statistical test, from Gibbons et al. (1989), on the proposed factor model, and stop at the hypothesis rejection. However, we approach this procedure as an optimization problem. The innovation is that the deep learning optimization system continues to search for the optimal solution because our bottom-up approach provides the non-reduced-form mechanism. We show a standard "feed-forward" network consisting of an "input layer" of firm characteristics, "hidden layers" of factors, and an "output layer" of security returns (check Figure 2 for the Fama-French example). The factor generation receives training feedback from the objective function through backward propagation, which addresses the question of how much pricing errors can be reduced by optimizing over model parameters.

On the methodological side, we marry state-of-the-art deep learning algorithms with asset pricing factor models. Deep learning is well known for its superior pattern-matching performance, the flexible architecture, and yet, the mysterious "black box." The goal of this paper is to introduce deep learning into asset pricing with a transparent "white box" model. We disassemble the asset pricing mechanism with deep learning concepts: inputs, intermediate features, outputs, and the objective function. We show the routine procedure financial economists have been working for decades is a transparent "white box" model. The "deep" part of the past asset pricing research is to *manually discover* those useful firm characteristics from all economic information.

On the economic side, long-short factors are useful because they reflect compensation for exposure to underlying characteristics and can be evaluated as tradable portfolios. However, many of these characteristics' formulas are highly similar from accounting, trading, macroeconomics, and behavioral perspectives. One unresolved issue is how minor differences in characteristics' formulas affect the corresponding security sorting, long-short factors, and even model fitness. The routine procedure has one long-time overlooked the built-in problem. Most research focuses are between factors [intermediate features] and security returns [outputs], whereas the inputs are characteristics. Our research attempts to fill in this missing piece. Our "white box" deep learning approach investigates the complete channel from characteristics [inputs] to security returns [outputs].

On the empirical side, we study the universe for the largest 3,000 stocks in the U.S. equity market over the last 45 years. Our library consists of 62 published firm characteristics. For statistical evidence, we find the best (highest time series R^2) models for the train assets work well for all other test assets. For economic evidence, in both train and different test assets, we also show substantial and robust reductions on pricing errors and increases in cross-sectional R^2. Using our deep factors, we show a significantly higher Sharpe ratio in factor investing, as well as the improvement for the deep factor model to explain all 62 published anomalies. Finally, to interpret our deep factors in a dimension reduction perspective, we find the volatility charateristics are most important to reduce pricing errrors by controlling Fama-French three factors.

The rest of the paper is organized as follows. We compare and position our study with the relevant literature in section 1.1. Section 2 introduces deep learning concepts into the asset pricing field. Section 3 provides the technical details for our deep learning model implementation. Section 4 illustrates the empirical study design and the main empirical findings. Section 5 adds the summary and discussion for future research.

1.1 Connections with Previous Literature

Our paper builds on several strands of the asset pricing literature. The most related literature are dimension reduction techniques via principal components and its generalizations. For example, Kelly et al. (2019) and Kim et al. (2018) use firm characteristics as instruments to model the time-varying coefficients and estimate PCs. Lettau and Pelger (2018) derive properties of RP-PCA that identify factors with small time series variance that are useful in the cross section. Kozak et al. (2018) show that PCA of anomaly portfolios works as well as a reduced-form factor model in explaining anomaly portfolios. Light et al. (2017) use partial least squares (PLS) to aggregate information on firm characteristics. Our deep learning framework differs from PCA in four main ways:

(1) Our dimension reduction is applied directly on firm characteristics [inputs] rather than factors [intermediate features] or security returns [outputs].

(2) Our dimension reduction also allows for both nonlinearity and interactions on inputs, whereas PCA only extracts linear components.

(3) PCA relies on a balanced data structure, whereas security sorting allows for an unbalanced data structure, such as individual stock returns and characterisitics.

(4) PCA is known by the poor out-of-sample performance due to static component loadings, while sorting securities on lag characteristics can be dynamically updated.

As discussed in the beginning, our approach is closely related to the recent literature on applying machine learning methods the the asset pricing model. Harvey et al. (2016) raise a multiple testing issue to challenge 300 factors discovered in the literature. Feng et al. (2019) develop a regularized two-pass cross-sectional regression to tame the factor zoo, and find only a small number of factors with incremental contribution. Kozak et al. (2019) use a shrinkage estimator on the SDF coefficients for characteristic-based factors with economic interpretation. Kelly et al. (2019) evaluate the contribution of individual characteristics under a nested-model comparison via model fitness improvement. A recent article of DeMiguel et al. (2018) shows the economic rational why many characteristics are needed in investing portfolios.

Our paper adds to the literature on forecasting asset returns with machine learning. Freyberger et al. (2019) apply adaptive group LASSO for selecting firm characteristics and show evidence of nonlinearity. Gu et al. (2018) provide a comprehensive empirical investigation of forecasting performance using multiple machine learning algorithms. Han et al. (2018) employ a forecast combination approach and Bianchi et al. (2018) find that machine learning can forecast bond returns as well. To be clear, our model does not provide a direct forecast for asset returns. However, the prediction literature studies the time series predictive performance between inputs and outputs, and skips the intermediate channel involved with risk factors. We fill in the missing pieces with our bottom-up approach: lag characteristics, realized factors, and realized security returns. The Bayesian predictive regression of Feng and He (2019) uses lag characteristics for the linear conditional factor coefficients, which is a reduced-form approach for approximation.

Alternatively, we provide an out-of-sample evaluation in section 4, which is about the cross section instead of time series. We use one set of test portfolios to train the factor model and test its pricing performance with another set of unseen test assets. This design is a solution to the skepticism of Lewellen et al. (2010), who question the standard protocol of using Fama-French 25 size-B/M portfolios for both factor discovery and model testing.

Finally, we add to the recent development of deep learning in finance and economics. Heaton et al. (2017) introduces deep learning decision models for problems in financial prediction and classification. Feng et al. (2019) provide a seemingly unrelated regression for the deep neural network on stock return prediction. A recent paper of Chen et al. (2019) uses a neural network to estimate the SDF model that explains all asset returns from the conditional moment constraints implied by no arbitrage. This continued progress in deep learning research is promising for both academic research and practical application in finance.

2 Deep Learning and Asset Pricing

Section 2.1 explains why we can treat the asset pricing test via an optimization problem for pricing errors. We demonstrate how a characteristics-sorted factor model can be reformulated within a deep learning architecture in section 2.2. Fama-French-type factor models are shown to be deep learning models, and we discuss implementation is discussed in section 2.3.

2.1 Minimizing pricing errors

From the economic constraint of the beta-pricing model, it follows that the excess asset return can be explained by the risk premia of factors:

$$\mathrm{E}(R_{i,t}) = \alpha_i + \beta_i^{\intercal}\mathrm{E}(f_t) + \gamma_i^{\intercal}\mathrm{E}(g_t), \tag{1}$$

The GRS test suggests a joint test on the vector of time series model intercepts, α, for all test assets. The kernel for the GRS test statistic is a weighted sum for the quadratic alphas, $\alpha^\intercal\Sigma_\alpha^{-1}\alpha$. If a sufficient factor model exists, this weighted sum for pricing errors should be statistically and economically insignificant. However, the long history of rejecting the GRS test shows the literature is still far from the "sufficient" model.

For this reason, we switch to an optimization problem for an alternative perspective. Machine learning and deep learning methods have been criticized for easily over-fitting the data. Adding more factors on the regression's right-hand side increases the time series R^2 but unnecessarily reduces the magnitude of the regression intercept. The asset pricing optimization target is different from the time series model fitness. Therefore, we design such a deep learning framework that pushes the model fitting to the lower bound for pricing errors, which might not be zero.

Denote $\widehat{R}_{i,t}$ as a linear portfolio constructed with factors to mimic the asset return $R_{i,t}$. Because all regressors need to be tradable portfolios in the time series regression, $\widehat{R}_{i,t}$ is formed as a linear combination of portfolios *without an intercept*. The expectation difference, α_i, is the pricing error.

$$\mathrm{E}(R_{i,t} - \widehat{R}_{i,t}) = \alpha_i \tag{2}$$

The tradability for alphas determines the unique objective function in our optimization. The core of our objective function design is $\frac{1}{N}\sum_{i=1}^{N}\alpha_i^2$, an equally weighted version for the GRS test statistic kernel, and measures the average pricing errors. To the best of our knowledge, this paper is the first that focuses on minimizing alphas. We define an economic-driven objective function, minimizing pricing errors, which follows the non-arbitrage restriction from asset pricing models.

Compared to time series regressions in asset pricing, another widely used approach is the cross-sectional regression in Equation 3.

$$\mathrm{E}(R_{i,t}) = \beta_0 + \beta_i^\intercal\lambda_f + \gamma_i^\intercal\lambda_g + \alpha_i \tag{3}$$

For example, Fama and MacBeth (1973) provide a two-pass methodology to add a second pass by regressing average returns on estimated betas. This approach has its econometric advantages and particularly allows for non-tradable factors, but its model implied pricing errors, α_i, are regression residuals and no longer tradable. To reserve the economic driven loss function, we choose to work on the time series regression. However, we also provide a robustness check to show the pricing performance using the cross-sectional R^2.

2.2 Characteristics-Sorted Factors in Deep Learning

By following the standard literature, we use excess returns in the study. Our model is to generate additional factors from the deep learning model, f_t, on a benchmark model g_t, which can be CAPM or Fama-French type models. We form the realized return predictor $\widehat{R}_{i,t}$ as a linear combination of f_t and g_t without an intercept. Therefore, the zero mean residual, $\epsilon_{i,t}$, measures the time series variation in forecasting error, and α_i refers to the potential pricing error.

$$\begin{aligned}\widehat{R}_{i,t} &= \beta_i^\intercal f_t + \gamma_i^\intercal g_t, & (4)\\ R_{i,t} - \widehat{R}_{i,t} &= \alpha_i + \epsilon_{i,t}, & (5)\\ f_t &= W_{t-1} r_t, & (6)\\ W_{t-1} &= H(z_{t-1}). & (7)\end{aligned}$$

The additional deep factors, f_t, are long-short portfolios constructed by sorting individual firms on lag firm characteristics z_{t-1}. We use r_t, thousands of individual firm returns at month t, and W_{t-1}, the long-short portfolio weight determined at month $t-1$.

$H(\cdot)$ represents a complex (and hidden) function for z_{t-1} that reflects underlying cross-sectional predictability. The $H(\cdot)$ transformation is the depth for the complete deep neural network. For example, $H(\cdot)$ can be the sorting function as a shallow network, then it transforms z_{t-1} to the long-short directions $\{1, 0, -1\}$. With the long-short directions, researchers multiply the long-short directions by equal or value weights to form W_{t-1}.

With the notation $\{f_t, r_t, W_{t-1}, z_{t-1}\}$, the characteristics-sorted factor model is clear and transparent in the above deep learning architecture. (1) The first inputs are lag characteristics z_{t-1}. (2) By sorting securities in the month $t-1$, we obtain the intermediary features W_{t-1}. (3) By adding the second inputs, individual firm realized returns r_t, we generate f_t. (4) By adding the third inputs, the benchmark model g_t, we fit R_t.

The predictive structure for characteristics-sorted factor investing is due to the lag portfolio construction. The factor f_t is built with long-short portfolio weights at month $t-1$ and individual firm returns at month t. This factor model return fitting is different from those models in Freyberger et al. (2019) and Gu et al. (2018) for predicting returns via firm characteristics, because we use realized returns $\{r_t, g_t\}$ as second and third inputs.

In our framework, f_t is generated while controlling g_t within the deep learning model fitting. This procedure is consistent with the standard protocol that researchers admit new factors for their significance over a benchmark model. The estimated alphas or pricing errors are constructed as

$$\hat{\alpha}_i = \frac{1}{T}\sum_{t=1}^{T}\left(R_{i,t} - \widehat{R}_{i,t}\right) = \frac{1}{T}\sum_{t=1}^{T}\left(R_{i,t} - \widehat{\beta}_i^{\intercal} f_t - \widehat{\gamma}_i^{\intercal} g_t\right). \tag{8}$$

Our optimization objective is to minimize a weighted sum for the time-series variation and cross-sectional pricing errors:

$$\begin{aligned}
\mathcal{L}_\lambda &= \underbrace{\frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\left(R_{i,t} - \widehat{R}_{i,t}\right)^2}_{\text{time-series variation}} + \underbrace{\lambda * \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{T}\sum_{t=1}^{T}(R_{i,t} - \widehat{R}_{i,t})\right)^2}_{\text{pricing errors}} && (9)\\
&= \frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\left(R_{i,t} - \hat{\beta}_i^{\intercal} f_t - \hat{\gamma}_i^{\intercal} g_t\right)^2 + \frac{\lambda}{N}\sum_{i=1}^{N}\hat{\alpha}_i^2 && (10)\\
&= \frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\left(\hat{\epsilon}_{i,t} + \hat{\alpha}_i\right)^2 + \frac{\lambda}{N}\sum_{i=1}^{N}\hat{\alpha}_i^2 && (11)\\
& && (12)\\
&= \underbrace{\frac{1+\lambda}{N}\sum_{i=1}^{N}\hat{\alpha}_i^2}_{\text{pricing errors}} + \underbrace{\frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\hat{\epsilon}_{i,t}^2}_{\text{idiosyncratic error}}, && (13)
\end{aligned}$$

where $\hat{\epsilon}_{i,t} = R_{i,t} - \widehat{R}_{i,t} - \hat{\alpha}_i$, $\sum_{t=1}^{T}$ and $\hat{\epsilon}_{i,t} = 0$.

Here, λ is a tuning parameter that controls the balance between time-series and cross-sectional pricing errors. If λ is too big, we lose the weight for time series variation that supports the factor structure. If λ is too small, the objective function is not helpful in reducing pricing errors. In our empirical study, we perform a validation using a sequence of λ's. Our objective function design follows the RP-PCA of Lettau and Pelger (2018), who also add a penalty to account for cross-sectional pricing errors in average returns. Their regularized estimation is to identify those factors with small time series variation, but help price the cross section.

2.3 Fama-French Examples in Deep Learning

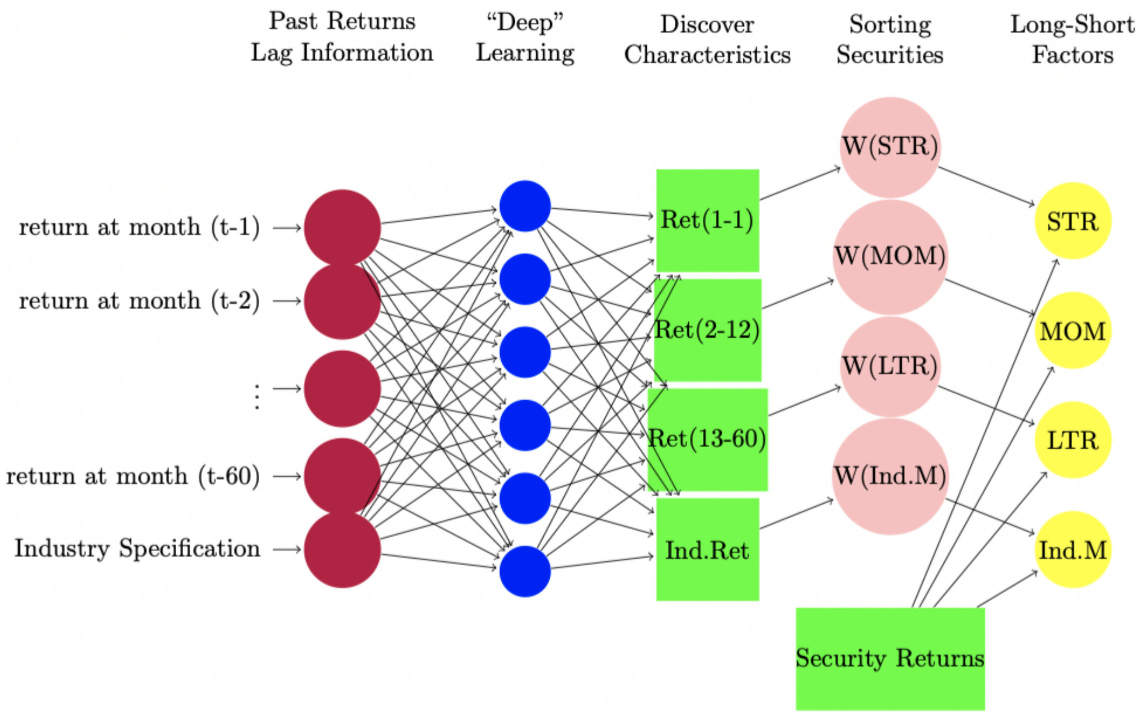
We have seen many asset pricing models proposed to explain the compensation for different risk-taking behaviors. However, the current factor zoo contains too many similar firm characteristics used to proxy for the same risk-taking information. For example, many measures are proposed for the value investing, such as book-to-market ratio, dividend yield, earning-to price ratio, cash flow to price ratio, and so forth. Sorting securities on these "seemingly" related characteristics might be a trial-and-error experiment, which finds the one proxy with the best in-sample performance for the test assets in the test period, but probably does not work for others. A deep learning structure can help pick the best proxy (model selection), combine proxies (dimension reduction), or simply create the best proxy for the objective function.

For example, multiple momentum factors exist: long-term reversal (13-60), short-term reversal (1-1), the Carhart Momentum (2-12), seasonality (1-13), industry momentum, and so forth. All of these similar momentum characteristics are simply sums of past individual security or portfolio returns. Therefore, the raw inputs z_{t-1} in Figure 1 are past returns in purple circles. These momentum characteristics are "calculated" or "combined" from raw inputs and become the new inputs for the actual characteristics sorting. The blue circles in the hidden layers might include many trial-and-error experiments or manual "deep" learning for data mining concerns. The second-to-the-last layer combines the long-short portfolio weights, W, and individual security returns to generate the long-short factors. Figure 1 provides the procedure for calculating characteristics and creating factors. The deep learning philosophy has been adopted in asset pricing for a long time but is manually implemented by researchers.

Figure 2 shows a complete deep learning architecture for a characteristics-sorted factor model, which is the example for the Fama-French five-factor model. Researchers typically start with a formula for the calculation of characteristics used for security sorting as in the blue circles. Then, they sort individual firms on the lag characteristics to determine the long-short portfolio weights as in

Figure 1: **Sorting Securities and Generating Factors**

This figure provides the procedure for calculating characteristics and creating factors. We start with past returns as raw inputs, and then calculate characteristics as the new inputs for security sorting. The last layer is the factor generation on long-short portfolio weights obtained from the previous layer plus individual security returns.



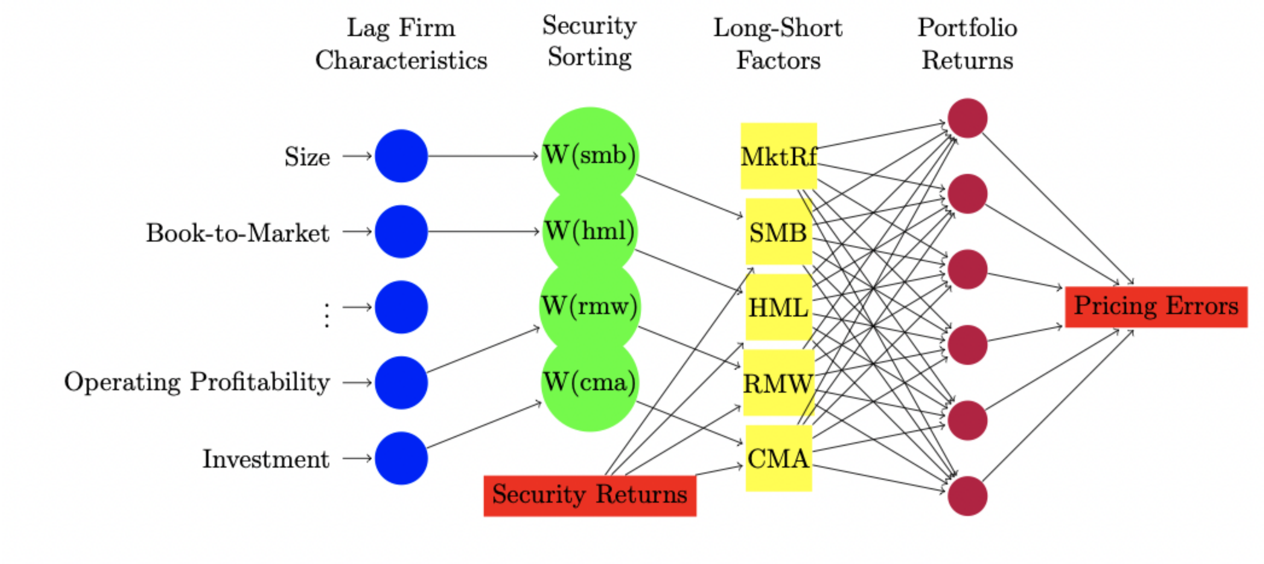
the green circles.[1] Then, in the yellow rectangles, researchers construct factors as long-short portfolios using the portfolio weights from the last layer along with realized security returns. Adding the market factor produces an augmented factor model to explain realized returns of test assets in the purple circles. The last red rectangle is the objective function for pricing errors.

Using our notation, g_t is MktRf if CAPM is the benchmark. The Fama-French five-factor model adds four additional factors, f_t, on the benchmark. The characteristics' size, book-to-market, operating profitability, and investment are z_t. W_t are determined with the bivariate-sorting directions and the lag market equity for value weights. In the standard literature, these four additional factors are tested with significance over CAPM with test assets in purple circles, which can be 25 size-B/M portfolios. In our deep learning diagram, these four additional factors are trained by controlling the

[1]If a firm does not exist or has missing characteristics in some periods, it is not included in the security sorting for those periods. Therefore, security sorting works perfectly for the imbalanced panel data structure with missing values, which is the nature of firm dynamics.

Figure 2: **Fama-French 5-Factor Model in Deep Learning**

This figure provides a deep learning representation of building the Fama-French five-factor model using firm characteristics to calculate the objective function, pricing errors, for portfolio returns. The lag characteristics are inputs. The long-short factors are hidden neurons. The portfolio returns are outputs.



benchmark model CAPM to minimize the objective function, pricing errors.

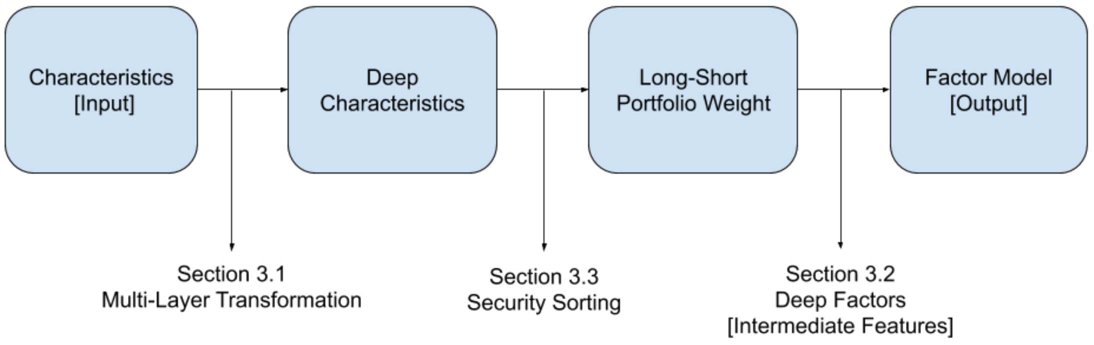
The potential multi-layer transformations and combinations, denoted by $H(\cdot)$, of characteristics are determined before the blue circles. Here, researchers typically determine the formula for anomalies that help pricing in the cross section. A major drawback of this approach is that the characteristics' usefulness is tested statistically ex post, but the feedback for model fitting is never returned to characteristics' construction. With the new technology of backward propagation, our model can be refitted sequentially by the feedback on the change in the objective function.

3 Deep Learning in Characteristics-Sorted Factor Models

In this section, we introduce a bottom-up approach of our deep learning model, which provides a non-reduced-form mechanism. Figure 3 shows a clear roadmap for how we dissemble the characteristics-sorted factor model within deep learning. Section 3.1 illustrates how the dimension reduction on the *[inputs]* firm characteristics performed in the feed-forward neural network via multi-layer transformations. Then, we get the deep characteristics. Section 3.2 calculates the *[intermediate features]* deep factors, whose long-short portfolio weights are calculated in section 3.3. Section 3.4 describes the optimization objective and summarizes the complete deep learning model. In Appendix A, we also provide the optimization details.

In section 3.3, one can instead simply adopt equal or value weights to create the long-short factors. For this reason, we put the weighting scheme of section 3.3 after the factor model in section 3.2 of the below text. However, from an optimization perspective, we suggest adopting our softmax rank-weighting scheme, which is differentiable and provides an economic-driven weighting scheme. We also explain in detail why the neural network optimization requires a differentiable activation function in Appendix A.

Figure 3: **Map for Deep Learning Model Description**


Characteristics [Input]
Deep Characteristics
Long-Short Portfolio Weight
Factor Model [Output]
Section 3.1 Multi-Layer Transformation
Section 3.3 Security Sorting
Section 3.2 Deep Factors [Intermediate Features]

A typical training observation indexed by time t includes five types of data:

$\{R_{i,t}\}_{i=1}^{N}$, excess returns of N test portfolios

$\{r_{j,t}\}_{j=1}^{M}$, excess returns of M individual stocks

$\{z_{k,j,t-1} : 1 \leq k \leq K\}_{j=1}^{M}$, K lagged characteristics of M firms

$\{g_{d,t}\}_{d=1}^{D}$, D benchmark factors.

We use a matrix notation for $\{R_t, r_t, z_{t-1}, g_t\}$, where R_t is an $N \times 1$ vector, r_t is an $M \times 1$ vector, z_{t-1} is a $K \times M$ matrix, and g_t is a $D \times 1$ vector. In section 4.1, we have $M = 3,000$ stocks, $K = 62$ characteristics, and $D = 1$ or 3 for CAPM or Fama-French 3-factor model. Before introducing each part of the deep learning implementation, we provide a summary of parameter notations and dimensions in Table 1.

Table 1: **Deep Learning Mechanism**

This table provides an algorithm summary for the bottom-up approach of our deep learning model. The deep learning network feeds forward from the bottom to the top in the table. The initial input is firm characteristics, and the final outputs are security returns. For each layer, the network takes the output from the immediate lower layer as its new inputs, as well as the additional input if needed. The additional inputs include individual security returns r for deep factors, and the benchmark factor model g for security returns.

	Dimension	Output	Inputs	Operation	Parameters
Security Returns	$N \times 1$	$\hat{R}$	g	$\beta f + \gamma g$	(β, γ)
Deep Factors	$P \times 1$	f	r	Wr	
Rank Weights	$P \times M$	W		$\text{softmax}(y^+) - \text{softmax}(y^-)$	
Deep Characteristics	$K_L \times M$	Y		$F^{[L]}\left(Z^{[L-1]}\right)$	$(A^{[L]}, b^{[L]})$
	$\vdots$	$\vdots$		$\vdots$	$\vdots$
	$K_l \times M$	$Z^{[l]}$		$F^{[l]}\left(Z^{[l-1]}\right)$	$(A^{[l]}, b^{[l]})$
	$\vdots$	$\vdots$		$\vdots$	$\vdots$
Firm Characteristics	$K_0 \times M$	$Z^{[0]}$	z	$Z^{[0]} := z$	

3.1 Deep Characteristics

We introduce how to design an L-layer neural network with the purpose of generating P deep characteristics. This operation is the the "deep" part to induce nonlinearity and interaction within the dimension reduction from K to P characteristics.

All transformations performed in this part are within each individual stock. The data (and intermediate results) of two different stocks are separated and don't interfere with each other. We drop for now the subscript t, bearing in mind that the inputs z are lagged variables. The architecture is as follows, for $j = 1, 2, ..., M$:

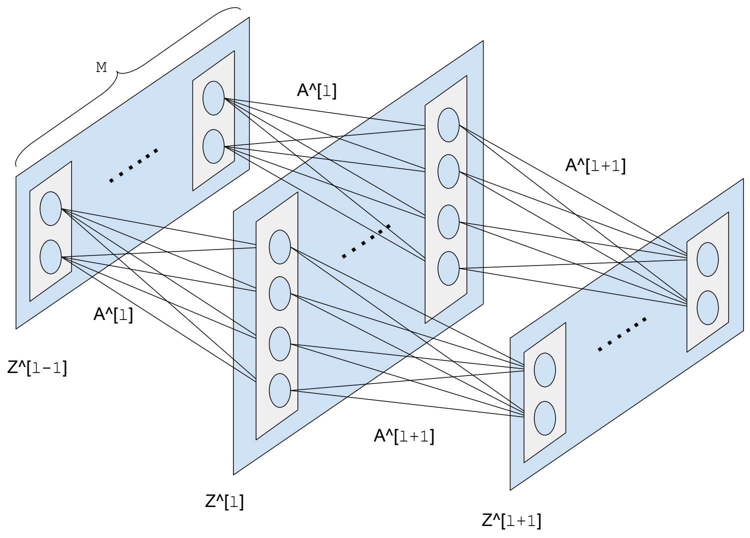
$$Z_{\cdot,j}^{[l]} = F\left(A^{[l]} Z_{\cdot,j}^{[l-1]} + b^{[l]}\right), \text{ for } l = 1, 2, ..., L$$
$$Z_{\cdot,j}^{[0]} := [z_{1,j}, ..., z_{K,j}]^\intercal,$$

where $Z^{[l]}_{\cdot,j}$ is the j-th column of a $K_l \times M$ matrix $Z^{[l]}$. We set $K_0 = K$ and $K_L = P$. F is the univariate activation function, broadcasting to every element of a matrix. The parameters to be trained in this part are deep learning weights A's and biases b's, namely,

$$\left\{(A^{[l]}, b^{[l]}) : A^{[l]} \in \mathbb{R}^{K_l \times K_{l-1}}, b^{[l]} \in \mathbb{R}^{K_l}\right\}_{l=1}^{L}.$$

Figure 4: **Deep Network of $Z^{[l-1]} \to Z^{[l]} \to Z^{[l+1]}$.**

This figure shows how the deep learning network forwards from $Z^{[l-1]}$ to $Z^{[l+1]}$. $K_{l-1} = K_{l+1} = 2$, $K_l = 4$. The lines connecting two layers represent affine transformation, and the circles represent activation function.



The transformations are performed column by column with no communication across different firms. This univariate transformation is perfectly built for the security sorting for different stock universes. Notice the input layer for deep characteristics is a linear function for firm characteristics. The multi-layer structure helps train the parameters for this linear equation. Our deep characteristics are not built for one particular characteristic, but rather for the linear combinations.

With some abuse of notation, we rewrite the architecture for the output Y as our $P \times M$ deep characteristics,

$$
\begin{aligned}
Y &:= Z^{[L]}, \\
Z^{[l]} &= F\left(A^{[l]} Z^{[l-1]} + b^{[l]}\right), \text{ for } l = 1, 2, 3, ..., L \\
Z^{[0]} &:= z.
\end{aligned}
$$

Unlike a standard feed-forward neural network, the l-th layer in our architecture is a neural matrix $Z^{[l]}$. Each row of $Z^{[l]}$ is a $1 \times M$ vector representing the k_l-th intermediary characteristics for M firms, $k_l = 1, 2, ..., K_l$. We explicitly make all the columns (firms) share the same parameters $A^{[l]}$ and $b^{[l]}$, whose dimensions are independent of M. Therefore, the formula for deep characteristics is the same for every firm.

Here, K_l denotes the dimension of the l-th layer because the number of columns is fixed as M for all $Z^{[l]}$'s. Figure 4 illustrates how our deep-learning network operates by showing a sample architecture from the $(l-1)$-th to the $(l+1)$-th layer, where $K_{l-1} = K_{l+1} = 2$ and $K_l = 4$. The Fama-French approach simply drops all hidden layers and uses $Y := z$ for sorting in the latter part. By contrast, $Z^{[0]} := z$ in our deep network goes through multiple layers of affine transformations and nonlinear activations, and ends up with a low-dimensional deep characteristic Y. Here, the layer sizes $\{K_l\}_{l=1}^{L}$, and the number of layers L are architecture parameters chosen by model designers.

3.2 Deep Factors

In this section, we continue with the construction of deep factors based on long-short portfolio weights W (discussed in section 3.3), and then an augmented factor model for asset pricing. To create the long-short factors, we need the individual stock returns and the corresponding weights. The architecture after obtaining W is as follows:

$$\hat{R} := Z^{[L+3]} = h^{[2]}\left(Z^{[L+2]}, g\right) \tag{14}$$

$$f := Z^{[L+2]} = h^{[1]}\left(Z^{[L+1]}, r\right) \tag{15}$$

$$W := Z^{[L+1]} \tag{16}$$

Here, $h^{[1]}$ and $h^{[2]}$ are no longer univariate activation functions. Instead, they are operators specially defined to conduct important transformations, which take two arguments: one from the previous layer and another from additional inputs.

We now describe these operators in detail. $h^{[2]} : \mathbb{R}^P \times \mathbb{R}^D \to \mathbb{R}^N$ is a linear transformation of its two arguments, and the parameters are denoted as $\beta \in \mathbb{R}^{N\times P}$ and $\gamma \in \mathbb{R}^{N\times D}$:

$$h^{[2]}(f, g) = [\beta \ \gamma] \begin{bmatrix} f \\ g \end{bmatrix}. \tag{17}$$

Therefore, g represents the benchmark model, such as Fama-French three factors. $h^{[2]}$ is the augmented factor model by adding our deep factors f along with g. $h^{[1]} : \mathbb{R}^{P\times M} \times \mathbb{R}^M \to \mathbb{R}^P$ defines how we construct deep factors as tradable portfolios. Once given the portfolio weights W and individual stock returns r, it is simply a matrix production:

$$h^{[1]}(W, r) = Wr. \tag{18}$$

The tradability for factor and individual stock returns {f, g, r} is crucial to determine our economic-driven loss function, which follows the non-arbitrage condition.

3.3 Nonlinear Rank Portfolio Weights

The formation of long-short portfolio weights is presented in this section. Two recent papers adopt the rank weights for creating factors. Frazzini and Pedersen (2014) develop their factor, betting against beta, with a "rank weighting." They assign each stock to either the "high" portfolio or the "low" portfolio with a weight proportional to the cross-sectional rank of the stock's estimate beta. Novy-Marx and Velikov (2018) add a further discussion to compare different portfolio weighting schemes: rank (linear) weights versus equal weights.

Following Frazzini and Pedersen (2014), our procedure here generalizes the standard equal weights and introduces more nonlinearity. We define

$$h^{[0]} : \mathbb{R}^M \to [-1, 1]^M$$

to calculate the portfolio weights based on the rankings of deep characteristics. When the argument is a matrix, it broadcasts to all rows. Let y be a $M \times 1$ vector representing some deep characteristic, that is, a row of Y.

$$h^{[0]}(y) = \underbrace{\begin{bmatrix} \text{softmax}(y_1^+) \\ \text{softmax}(y_2^+) \\ \vdots \\ \text{softmax}(y_M^+) \end{bmatrix}}_{\text{long portfolio}} - \underbrace{\begin{bmatrix} \text{softmax}(y_1^-) \\ \text{softmax}(y_2^-) \\ \vdots \\ \text{softmax}(y_M^-) \end{bmatrix}}_{\text{short portfolio}} \tag{19}$$

where $y^+ := y, y^- := -y$ in the simplest case and the nonlinear softmax activation function is an increasing function,

$$\text{softmax}(y_j) = \frac{e^{y_j}}{\sum_{j'=1}^{M} e^{y_{j'}}},$$

and $\sum_{j=1}^{M} \text{softmax}(y_j) = 1$. The first softmax vector in the expression of $h^{[0]}$ represents the weights of stocks in the long portfolio (large y leads to large weight), and the second vector represents the short portfolio (large y leads to small weight). To prevent the exponential operator in softmax from introducing asymmetry and exaggerating the effect of extreme values, we need to first standardize y along the same axis and apply an additional nonlinear transformation before feeding into $h^{[1]}$.

To demonstrate properties of the rank-weight scheme, we use the following example. The left panel of Figure (5) shows the final output of $h^{[0]}$ (red line), the portfolio weights W, when

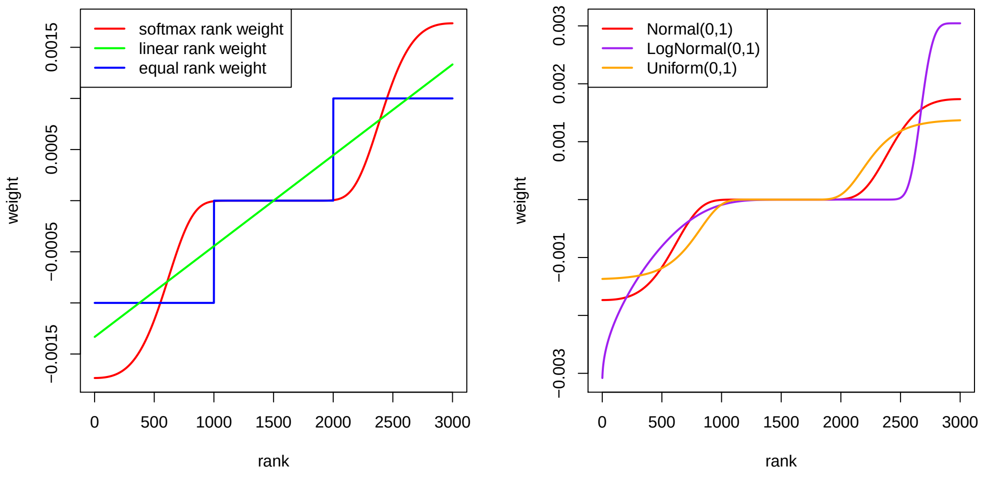
$$y^+ = -50e^{-5y}, y^- = -50e^{5y},$$

and $y = [y_1, y_2, ..., y_{3000}]^\intercal$ is drawn from standard normal distribution $N(0, 1)$. The x-axis shows the cross-sectional ranks of stocks.

For comparison, we also plot the standard equal weights (blue line) with the top and bottom $1/3$ of stocks as well as the rank weights introduced by Frazzini and Pedersen (2014) (green line). Whereas their rank weights are linear in firms' cross-sectional rankings, our weighting scheme adds nonlinearity. In terms of actual holdings, Novy-Marx and Velikov (2018) point out that linear rank-weighted portfolios and equal-weighted portfolios are highly overlapped (83%). The departure of our nonlinear rank-weighted portfolio from these latter two portfolios is obvious: it "tilts" even

Figure 5: **Comparison: Weight vs. Rank**

This figure shows the example of softmax rank weights for 3,000 stocks, $h^{[0]}(y) = \text{softmax}(-50e^{-5y}) - \text{softmax}(-50e^{5y})$. In the right panel, y_j's are distributed as standard normal. The red line is the softmax weight; the blue line is the equal weight (with threshold = 1/3); the green line is the linear rank weights. In the left panel, the red line remains the same. The purple line is the softmax weights when y_j's are standardized samples from LogNormal$(0, 1)$. The orange line is the softmax weights when y_j's are standardized samples from Uniform$(0, 1)$.



more toward stocks with extreme characteristics. This feature, however, can be easily reversed. The flexibility of deep learning allows us to tune portfolio weights as well.

Unlike equal weights and linear rank weights, our softmax weights depend not only on cross-sectional rank information, but also on distributional features such as skewness and finite support, which stay after standardization. For illustration, in the right panel of Figure 5, we plot the softmax weights when characteristics are drawn from the skewed distribution LogNormal$(1, 3)$[2] (purple line) and the bounded distribution Uniform$(0, 1)$[3] (orange line). Interestingly, the distribution of characteristics affects the symmetry and curvature of the weight curve. We see that compared with the standard normal case, uniform characteristics lead to more holdings of stocks with middle ranks and fewer holdings of stocks in the top and bottom. The log-normal distribution breaks the symmetry of weights in the long and short portfolios. In this case, the long portfolio only holds a small proportion of stocks in the right tail, and the short portfolio holds almost all stocks in the lower half but still favors those with smaller characteristics.

[2]For example, all size-related characteristics follow a lognormal distribution.
[3]For example, characteristics such as performance scores follow a bounded distribution.

3.4 Minimizing Loss Function

The function H maps the lag predictors to the portfolio long-short weights,

$$W_{t-1} = H\Big(z_{t-1}\Big),$$

is essentially a composite given by $H(z) = h^{[0]} \circ F^{[L]} \circ \cdots \circ F^{[1]}(z)$. This multi-layer structure is the key idea of interpreting the security sorting as an activation function within a deep learner.

Fixing L, $\{K_l\}_{l=1}^{L}$, which are architecture parameters, our objective function is the mean squared prediction error regularized by mean squared pricing error

$$\mathcal{L}_\lambda(\hat{A}, \hat{b}, \hat{\beta}, \hat{\gamma}) := \frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\Big(R_{i,t} - \widehat{R}_{i,t}\Big)^2 + \frac{\lambda}{N}\sum_{i=1}^{N}\hat{\alpha}_i^2, \tag{20}$$

where

$$\widehat{R}_{i,t} = \hat{\beta}_i^\intercal f_t + \hat{\gamma}_i^\intercal g_t,$$

$$\hat{\alpha}_i = \frac{1}{T}\sum_{t=1}^{T}(R_{i,t} - \widehat{R}_{i,t}),$$

and

$$\hat{\beta} = \Big[\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_N\Big]^\intercal, \; \hat{\gamma} = [\hat{\gamma}_1, \hat{\gamma}_2, ..., \hat{\gamma}_N]^\intercal.$$

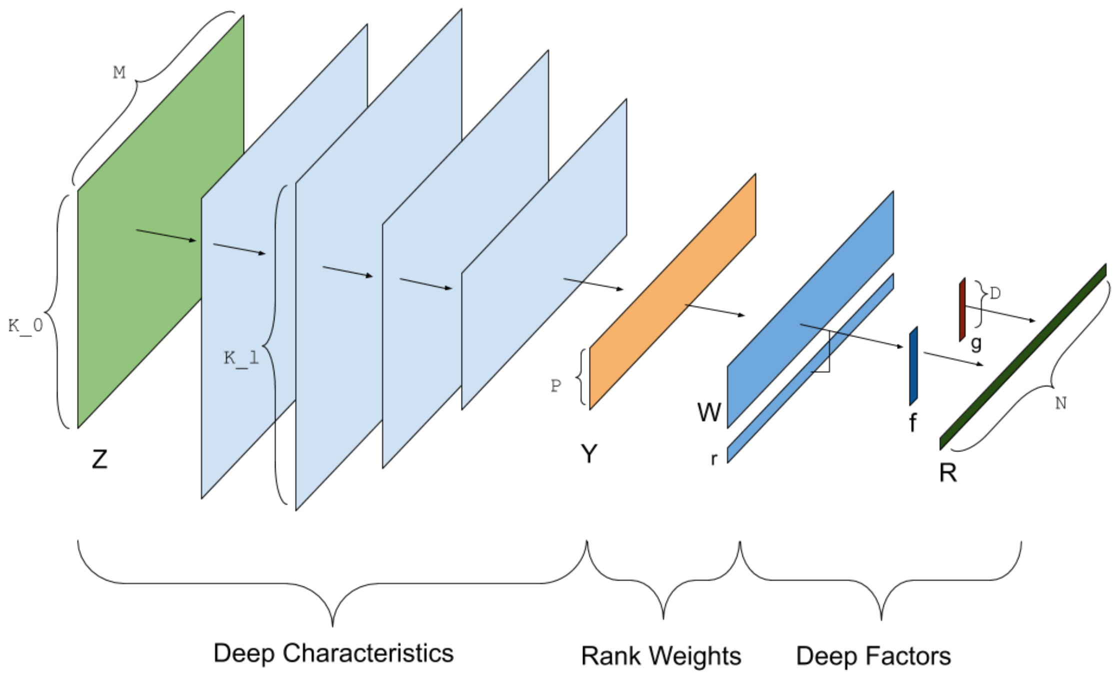
Here, λ is the regularization parameter. To train the deep network is then equivalent to obtaining a joint estimation of $(A, b) := \big\{A^{[l]}, b^{[l]}\big\}_{l=1}^{L}$ and (β, γ). The corresponding estimates are

$$(\hat{A}, \hat{b}, \hat{\beta}, \hat{\gamma}) = \arg\min \mathcal{L}_\lambda.$$

Empirically, we set layer size K_l for the l-th layer all equal to 32 and the number of layers $1 \leq L \leq 6$. For example, a four-layer ($L = 4$) network has layer sizes $K - 32 - 32 - 32 - P$ from $Z^{[0]}$ to Y. We summarize the above deep learning framework in Table 1 and Figure 6.

Figure 6: **Deep Learning Network Architecture**

This figures provides a visualization of deep learning architecture. The firm's characteristics $z$are transformed to deep characteristics Y via the deep network. Then, we "sort" Y to generate factor weight W. The deep factors f and benchmark factors g together are used to price the asset return R.



4 Empirical Findings

We report our empirical findings in this section. Section 4.1 describes the data, including the train-test assets and firm characteristics library. Section 4.2 provides the model fitness for our deep factor model to different test asset returns. We show the improvement for the time series model fitness, the pricing errors, and the cross-sectional model fitness. Section 4.3 shows the interpretation and application of our deep factors and deep characteristics.

4.1 Data

Our monthly data sample is from January 1974 to December 2018. We follow the Fama-French factor in the construction of individual stock filtering and use the largest 3,000 firms for lag market equity.[4] Under our algorithm, we can require an unbalanced cross-section panel for security sorting,

[4]We only include stocks for companies listed on the three main exchanges in the United States: NYSE, AMEX, or NASDAQ. We use those observations for firms with a CRSP share code of 10 or 11. We only include observations for firms listed for more than one year. We exclude observations with negative book equity or negative lag market equity.

but it requires more parameters for training. By covering more than 99.9% for the total market cap, we make it simple with the largest 3,000 stocks universe.

Following the factor library of Green et al. (2017) and Hou et al. (2017), we use 62 firm characteristics for sufficient coverage in these 45 years. The chosen firm characteristics include all main categories: accrual, size, value, momentum, profitability, investment, quality, volatility, and so on. The characteristics library is listed in Table 6 where we show the explained variation on deep characteristics. In the empirical study, we choose to adopt a monthly sorting scheme that has a holding period for one month. Therefore, we drop all annually updated characteristics and adopt their quarterly versions if available.

Notice Fama-French factors use characteristics calculated in the previous December and sort securities every June, though most of their characteristics are available quarterly. Most momentum strategies are sorted every month. When talking about characteristics, we need to be careful about their availability and coverage for the stock universe. Many fundamental quarterly characteristics are only available for a small proportion of stocks for many years, and their anomalies are not robust or driven by small stocks. Hence, our characteristics library uses a minimum coverage of 50% of the stock universe. In addition, for those quarterly fundamental variables, we apply a seasonal adjustment on the characteristics data, which is an average for the quarterly version and the annual version. If the quarterly version is missing, we use the annual version for imputation. For monthly characteristics, most of which are constructed by trading data, there is no coverage issue.

For the in-sample train assets, we train the deep learning model with the monthly bivariate-sorted 3×2 portfolios between size and other characteristics ($3 \times 2 \times 61 = 366$). These bivariate-sorted portfolios are shown to have stable factor loadings in the literature. For the out-of-sample test assets, we try to show the robustness for our trained deep learning model. We provide results for the monthly univariate-sorted 5×1 portfolios ($5 \times 1 \times 62 = 310$), the Fama-French 49-industry portfolios, and individual stocks, such as Dow Jones 30 and S&P 500.

4.2 Empirical Improvement

A critical distinction in applying deep learning in the asset pricing model is that the right-hand-side factors are constructed by individual stock returns, whereas the objective function is evaluated by a set of left-hand-side test portfolios (train asset). As discussed in Lewellen et al. (2010), most factors that price Fama-French 25 size-B/M portfolios do not necessarily show significance on other test portfolios. The individual stocks and different portfolios are "different" assets. This point is key in our out-of-sample model evaluation on "unseen" portfolios and individual stock returns. Without adding deep factors, the out-of-sample analysis is in-sample on the benchmark model. However, the deep factors are generated by the train assets, and then the augmented factor model makes fitting the test assets out-of-sample.

We use three measures to report the empirical results: time series R^2_{TS} for statistical evidence, and cross-sectional pricing errors and cross-sectional R^2_{CS} for economic evidence.

(1) Time Series R^2_{TS}:

$$R^2_{TS} = 1 - \frac{\frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\left(R_{i,t} - \widehat{R}_{i,t}\right)^2}{\frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\left(R_{i,t} - \bar{R}_i\right)^2}, \tag{21}$$

where $\widehat{R}_{i,t} = \hat{\beta}_i f_t + \hat{\gamma}_i g_t$ and $\bar{R}_i = \frac{1}{T}\sum_{t=1}^{T} R_{i,t}$.

(2) Pricing Errors:

$$PE = \frac{1}{N}\sum_{i=1}^{N}\hat{\alpha}_i^2, \tag{22}$$

where $\hat{\alpha}_i = \frac{1}{T}\sum_{t=1}^{T}(R_{i,t} - \widehat{R}_{i,t})$.

(3) Cross-Sectional R^2_{CS}:

$$R^2_{Cs} = 1 - \frac{Q}{Q_0}, \tag{23}$$

where $Q = \min_\lambda (X\lambda - \bar{R})^\intercal (X\lambda - \bar{R})$ and $Q_0 = \min_\lambda (1_N\lambda - \bar{R})^\intercal (1_N\lambda - \bar{R})$. Here $X = [1_N, \hat{\beta}]$ and $\hat{\beta}$ is the multivariate betas for factor loadings. $\bar{R} = [\bar{R}_1, \bar{R}_2, ..., \bar{R}_N]^\intercal$.

Table 2, 3, and 4 share the same format. The top-left panel is the in-sample model fitting. We use bivariate-sorted portfolios as train assets for training the deep learning model. Then, we use others as test assets for the cross-sectional out-of-sample evaluation. The benchmark model $\{g_t\}$ includes Fama-French three factors. The first row of every sub-panel indicates the benchmark numbers without any deep factors. The numbers below are *percentage changes*. We have independently trained $7 \times 4 = 28$ deep learning models in every sub-panel, with a different number of added deep factors and a different number of hidden layers. We do not adopt the traditional train-validation-test procedure for model selection. However, we simply underlined the best model (1-layer and 5-factor) by the maximum in-sample pricing error reduction in Table 3.

In Table 2, we see Fama-French three factors explain all characteristics-sorted portfolios, but don't perform well for individual stocks. As these 28 different models are trained by stochastic optimization independently, the R^2_{TS} is unnecessarily increased by adding one more factor. For the top-left panel, we see the recommended model (1-layer and 5-factor) is one of the highest R^2_{TS} ones. We find highly consistent model fitness improvement for the other five panels as well. Notice the augmented factor model increases more than 6% and 9% R^2_{TS} for individual stocks. The R^2_{TS} increases are above 4% for industry portfolios. Even though the Fama-French 3-factor model explains univariate-sorted portfolios and 25 size-B/M portfolios with more than 90% R^2_{TS}, adding our deep factors can still provide a 1% improvement.

The results for average pricing errors are listed in Table 3. Again, we can see the Fama-French 3-factor model explains 25 size-B/M portfolios perfectly, but also works well for bivariate-sorted portfolios and individual stocks. Mechanically, adding deep factors does not necessarily decrease the intercept alphas (some positive numbers are in the table). However, for the recommended model selected by time series R^2_{TS}, we find consistent and large reductions for the pricing errors. The biggest decreases happen for the top two sets of sorted portfolios (around 50%) and industry portfolios (around 14%). For the remaining three cases, we can still find more than 2% drop in pricing errors.

We also plot the model implied pricing errors in Figure 7 for Fama-French 25 portfolios and industry 49 portfolios. The two two figures show pricing errors (distance to the 45 degree line) in Equation 2. We find that Fama-French 3-factor model has positive pricing errors or under price most assets. Adding deep factors only makes a slight improvement in the figure, but many pricing errors are still "visible". This figure shows why the optimization focus is important when the hypothesis is mostly rejected. Though rejecting the hypothesis, we can still improve the model for smaller pricing errors or investing opportunities.

For a robustness check, we also report results for cross-sectional R^2_{CS} in Table 4. We find highly consistent results for improving the asset pricing model fitness on the cross-sectional regression. Fama-French 3 factors do well characteristics sorted portfolios, but badly perform in industry portfolios and individual stocks. The recommended model (1-layer and 5-factor) makes highly positive improvement in both train and test assets. In Figure 7, the bottom two figures show adding deep factors help decrease pricing errors substantially for those outliers.

Table 2 is not about asset pricing but the statistical model improvement. Table 3 is about asset pricing and demonstrates the power of our deep learning model for reducing the pricing errors. Similar to Gu et al. (2018), we also find that a shallow network outperforms a deep network for model fitness in both Table 2 and Table 3. Moreover, we find that a shallow network with too many hidden neurons (factors) does not work well for asset pricing model fitness. Adding too many deep factors to the benchmark model does not help explain either the time series or the cross section.

In our analysis, we simply pick $\lambda = 1$ for fitting the model, which balances the time series variation and cross-sectional variation. For a robustness check, we have include results for different values for tuning parameter λ in Appendix B. As we find, time series R^2_{TS}, pricing errors, and factor investing Sharpe ratios are highly robust for different values of λ.

4.3 Interpreting Deep Factors

4.3.1 Investing Deep Factors

First, we figure out how to use our deep factors and build a factor investing portfolio. We try to show the augmented factor model helps improve the portfolio performance. Kozak et al. (2019) show the portfolio performance for SDF coefficients on factors, which is equivalent to the mean-variance efficient portfolio weights:

$$b = \Sigma_F^{-1}\mu_F,$$

where $F_t = \{f_t, g_t\}$. The efficient portfolio is simply $\{F_t b\}$ before the standardization.

The results for annualized Sharpe ratios are listed in top two panels of Table 5. The train assets are 25 size-B/M portfolios. The top-left panel only takes one factor for the benchmark as CAPM, whereas the top-right one uses Fama-French three factors. In the first row, we can see the market factor alone produces a 45% Sharpe ratio, and adding SMB and HML leads to a 71% ratio. Adding deep factors increases the Sharpe ratios sharply. For recommended model (1-layer and 5-factor) in

Table 5, the highest percentage increase for CAPM is 95.7%, and it is 46.9% for the Fama-French 3-factor model. In absolute terms, adding deep factors on the Fama-French 3-factor model can lead to an annualized Sharpe ratio above 1. Though this factor investing analysis is in-sample, the numbers are higher than those in Figure 3 in Kozak et al. (2019).

For the Sharpe ratio improvement in the nested model, Barillas et al. (2019) shows it is possible to apply a simple squared Sharpe ratio test for the null hypothesis $H_0 : SR_F = SR_g$. The goal of this model diagnostic test is to evaluate the asset pricing model fitness improvement by adding f_t on the benchmark factors g_t. We have included the details for the test in Appendix C. We only include the test significances[5] in the first panel of Table 5. We find the recommended (1-layer and 5-factor) models over both CAPM and FF3 are in the 1% significance level. This is another strong economic evidence to show adding our deep factors help asset pricing models for this significant improvement in the efficient portfolio.

4.3.2 Dissecting Anomalies

Second, we want to check if the augmented factor model is useful for evaluating the factor zoo. In adding deep factors, we expect to see fewer significant anomalies. We have tested two versions of factors: 62 univariate-sorted factors and 61 bivariate-sorted factors. In the literature, we know many "discovered" anomalies that are not robust to different sorting schemes. Fama-French factors are bivariate-sorted factors. The drawback for a long-short portfolio on univariate-sorted characteristics is the high overlap with small stocks. We simply use this anomaly library as out-of-sample assets to evaluate our augmented factor model. Unnecessarily adding factors reduces neither the intercept nor t-statistic.

For the i-th anomaly, we count its alpha as significant if the t-statistic for $\bar{\alpha}_i$ of $\{\alpha_{i,t}\}$ is significant by a simple time series test:

$$\left|\frac{\sqrt{T}\bar{\alpha}_i}{\hat{\sigma}(\alpha_i)}\right| > 1.96,$$

where $\alpha_{i,t} := R_{i,t} - \widehat{R}_{i,t}$, $\bar{\alpha}_i = \frac{1}{T}\sum_{t=1}^{T}\alpha_{i,t}$ and $\hat{\sigma}(\alpha_i) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\alpha_{i,t} - \bar{\alpha}_i)^2}$.

The anomaly testing results for significance counts are listed in the middle and bottom four panels of Table 5. First, the bivariate-sorted factors have higher qualities with 44 and 46 significant

[5]Respectively, $***$ is 1%, $**$ is 5%, and $*$ is 10%.

anomalies by controlling for the benchmark, whereas the univariate-sorted factors only have 30 and 26. Again, we see weak but consistent decreases in significant anomalies for the recommended model (1-layer and 5-factor) in Table 5. For the bottom-right table, adding deep factors reduces the significant anomalies from 46 to 38.

4.3.3 Exploring Deep Characteristics

The final question we want to explore is the sources of those deep characteristics. The scatter plots in Figure 8 are based on our benchmark case (underlined in Table 2) on Fama-French three factors using bivariate-sorted portfolios. We have 540 observations in total for 540 months of data. By controlling SMB and HML factors, the scatter plots demonstrate the nonlinear relationship between deep characteristic and market equity or book-to-market ratio. Notice that, the plots for market equity are converted in the log scale, while we use the original scale in model training.

we have five deep characteristics and each of them carries different relationships. First, if we see linear trends in the first column for log market equity, then we can expect a nonlinear relationship between deep characteristics and market equity. Second, the points with book-to-market ratios are relatively clustered when the ratios are below one. Third, we can obviously find all deep characteristics are driven by other sources given the poor goodness-of-fit by size or value, which implies the generated deep characteristics reflect different singals beyond size and value.

To further explore other sources of deep characteristics, we use a Fama-Macbeth approach to analyze the explained variation through original data. Notice that, though we have the exact formulas for deep characteristics, they are too complicated to interpret through the multi-layer transformation. We can explore the cross-sectional explanatory power using all original characteristics [raw inputs]. In each month t, we run a cross-sectional regression for the deep characteristics on all 62 original characteristics using 3,000 firm observations.

$$\text{deepchar}_{i,t} = a_t + b_{1,t}\text{char}_{1,i,t} + \cdots + b_{62,t}\text{char}_{62,i,t} + \epsilon_{i,t} \tag{24}$$

As the same as the Fama-Macbeth approach, we get 540 regression models in total. We calculate the explained variation and obtain the averages through 540 models. We provide the normalized explained variation in Table 6 to demonstrate the variable importance.

It turns out almost 80% of all deep characteristics variation comes from three volatility characteristics: residual variance w.r.t FF3, residual variance w.r.t. CAPM, and stock total variance. This is the empirical evidence by applying dimension reduction on the input level instead of the intermediate feature level. By controlling Fama-French three factors, the next most important characteristics that reduces the pricing error are related to volatility. Unlike PCA to maximize variation in the data, our new perspective is directly related to the objective function and helps on asset pricing studies.

The volatility factor might be highly nonlinear such that sorting securities on original data is not an optimal case, where our deep learning model provides a better solution. Though we provide five additional deep characteristics, their variable importance are similar and their construction might be due to the same signal "volatility". This might be a model combination to capture various aspects of the signal before forming the long-short portfolios. It is also possible that fitting one additional factor beyond Fama-French three factors is sufficient.

5 Summary and Discussion

In short, our goal is to introduce deep learning into the field of asset pricing. Most people view a deep neural network as a "black box" model. However, we adopt the deep learning framework with a bottom-up approach, which provides a non-reduced-form mechanism for the characteristics-sorted factor model. With an economic objective to minimize pricing errors, we train a deep learning model using firm characteristics *[inputs]*, and generate risk factors *[intermediate features]* to fit the cross section of security returns *[outputs]*. Our algorithm provides deep learning generated factors that reduce the pricing errors and show significant improvement in the efficient portfolio. Another majoring finding is the importance of volatility characteristics to reduce pricing errors by controlling Fama-French three factors. To the best of our knowledge, this paper is the first to provide a unified framework to implement the characteristics-sorted factor model.

We want to emphasize that our paper is not directly related to the literature on predicting asset returns using machine learning. The current prediction literature studies the time series predictive performance between firm characteristics *[inputs]* and security returns *[outputs]*, and skips the intermediate channel involved with risk factors *[intermediate features]*. Our bottom-up approach fills in this missing piece. The Bayesian conditional predictive regression of Feng and He (2019) uses lag characteristics for the dynamics of factor coefficients,

$$\beta_{i,t} = \eta_i + \theta_i z_{i,t}. \tag{25}$$

Recent papers, including Kozak et al. (2019) and Kelly et al. (2019), have similar approaches to incorporate characteristics for asset pricing factor models. For identification reasons, they all assume the time-varying coefficients are linear deterministic functions on characteristics.

Moreover, on the technical side, we design the softmax activation to create the long-short portfolio weights for factor generation. This procedure generalizes the "rank weighting" scheme of Frazzini and Pedersen (2014) and Novy-Marx and Velikov (2018). Though equal- and value-weighted portfolios are widely used procedures, the cross-sectional distribution properties for different characteristics are largely omitted. When evaluating the long-short portfolio for a characteristic, the discussion of the long and short portfolio weights is necessary. Our method provides an alternative view on security sorting as well as factor generation.

Prediction and pattern matching are important applications for machine learning and deep learning. However, our paper shows the flexible optimization framework is also useful to researchers. If the current empirical test procedure always rejects the asset pricing test, stepping out of the comfort zone to look for new technologies is harmless. We have a chance to modify the objective function with an economic goal (minimizing pricing errors). We also have a chance to build up a non-reduced-form neural network to link together different pieces from square one.

References

Barillas, F., R. Kan, C. Robotti, and J. A. Shanken (2019). Model comparison with sharpe ratios. *Journal of Financial and Quantitative Analysis, Forthcoming*.

Bianchi, D., M. Büchner, and A. Tamoni (2018). Bond risk premia with machine learning. Technical report, University of Warwick.

Chen, L., M. Pelger, and J. Zhu (2019). Deep learning in asset pricing. Technical report, Stanford University.

DeMiguel, V., A. Martin-Utrera, F. J. Nogales, and R. Uppal (2018). A portfolio perspective on the multitude of firm characteristics. Technical report, London Business School.

Fama, E. F. and K. R. French (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics 33*(1), 3–56.

Fama, E. F. and J. D. MacBeth (1973). Risk, return, and equilibrium: Empirical tests. *Journal of Political Economy 81*(3), 607–636.

Feng, G., S. Giglio, and D. Xiu (2019). Taming the factor zoo: A test of new factors. *The Journal of Finance, Forthcoming*.

Feng, G. and J. He (2019). Factor investing: Hierarchical ensemble learning. Technical report, City University of Hong Kong.

Feng, G., J. He, and N. Polson (2019). Deep learning for predicting asset returns. Technical report, City University of Hong Kong.

Frazzini, A. and L. H. Pedersen (2014). Betting against beta. *Journal of Financial Economics 111*(1), 1–25.

Freyberger, J., A. Neuhierl, and M. Weber (2019). Dissecting characteristics nonparametrically. *The Review of Financial Studies, Forthcoming*.

Gibbons, M. R., S. A. Ross, and J. Shanken (1989). A test of the efficiency of a given portfolio. *Econometrica: Journal of the Econometric Society*, 1121–1152.

Green, J., J. R. Hand, and X. F. Zhang (2017). The characteristics that provide independent information about average us monthly stock returns. *The Review of Financial Studies 30*(12), 4389–4436.

Gu, S., B. T. Kelly, and D. Xiu (2018). Empirical asset pricing via machine learning. Technical report, The University of Chicago.

Han, Y., A. He, D. Rapach, and G. Zhou (2018). What firm characteristics drive us stock returns? Technical report, Washington University in St. Louis.

Harvey, C. R., Y. Liu, and H. Zhu (2016). ... and the cross-section of expected returns. *The Review of Financial Studies 29*(1), 5–68.

Heaton, J., N. Polson, and J. H. Witte (2017). Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry 33*(1), 3–12.

Hou, K., C. Xue, and L. Zhang (2017). Replicating anomalies. Technical report, National Bureau of Economic Research.

Kelly, B., S. Pruitt, and Y. Su (2019). Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics, Forthcoming*.

Kiefer, J. and J. Wolfowitz (1952). Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 462–466.

Kim, S., R. A. Korajczyk, and A. Neuhierl (2018). Arbitrage portfolios in large panels. Technical report, Georgia Institute of Technology.

Kozak, S., S. Nagel, and S. Santosh (2018). Interpreting factor models. *The Journal of Finance 73*(3), 1183–1223.

Kozak, S., S. Nagel, and S. Santosh (2019). Shrinking the cross section. *Journal of Financial Economics, Forthcoming*.

Lettau, M. and M. Pelger (2018). Estimating latent asset-pricing factors. Technical report, National Bureau of Economic Research.

Lewellen, J., S. Nagel, and J. Shanken (2010). A skeptical appraisal of asset pricing tests. *Journal of Financial Economics 96*(2), 175–194.

Light, N., D. Maslov, and O. Rytchkov (2017). Aggregation of information about the cross section of stock returns: A latent variable approach. *The Review of Financial Studies 30*(4), 1339–1381.

Merton, R. C. (1973). An intertemporal capital asset pricing model. *Econometrica: Journal of the Econometric Society*, 867–887.

Novy-Marx, R. and M. Velikov (2018). Betting against betting against beta. Technical report, University of British Columbia.

Robbins, H. and S. Monro (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 400–407.

Table 2: **Statistical Evidence: Time Series R^2_{TS}**

This table provides the results for time series R^2_{TS}, which is listed in equation 21. The top-left panel shows results on train assets for the deep learning model. Other panels shows results on test assets for evaluating the out-of-sample model evaluation. The benchmark model includes Fama-French three factors, which are listed in the first row. We have independently trained $7 \times 4 = 28$ deep learning models in all sub-panels, with a different number of added deep factors and a different number of hidden layers. The numbers shown are *percentage changes* over the benchmark model. We have underlined the best model selected by the maximum pricing error reduction.

# Layers	$L=1$	$L=2$	$L=3$	$L=4$	$L=1$	$L=2$	$L=3$	$L=4$
# Factors	Bivariate-Sorted Portfolios				Univariate-Sorted Portfolios			
FF3	0.92	0.92	0.92	0.92	0.90	0.90	0.90	0.90
1	1.3%	1.5%	1.4%	1.1%	0.4%	0.4%	0.4%	0.3%
2	1.2%	1.1%	1.6%	1.5%	0.7%	0.5%	0.5%	0.5%
3	2.0%	1.7%	1.8%	1.7%	0.9%	0.7%	0.6%	0.5%
4	1.9%	2.2%	1.9%	1.6%	0.7%	0.8%	0.7%	0.5%
5	2.4%	2.7%	1.7%	1.9%	1.2%	1.4%	0.7%	0.6%
6	1.9%	2.5%	2.1%	1.9%	1.3%	1.1%	0.8%	0.6%
7	2.1%	1.9%	2.4%	2.2%	1.4%	0.8%	1.0%	0.9%
	Fama &French 25 Portfolios				Industry 49 Portfolios			
FF3	0.91	0.91	0.91	0.91	0.55	0.55	0.55	0.55
1	0.2%	0.2%	0.2%	0.1%	0.7%	0.9%	0.8%	0.7%
2	0.3%	0.2%	0.3%	0.2%	1.7%	1.2%	1.0%	0.8%
3	0.7%	0.4%	0.3%	0.3%	2.3%	2.1%	1.4%	1.2%
4	0.4%	0.5%	0.4%	0.2%	2.2%	2.2%	1.7%	1.4%
5	0.8%	1.2%	0.4%	0.4%	4.1%	4.8%	2.3%	1.5%
6	1.0%	0.7%	0.7%	0.4%	4.3%	3.3%	2.6%	1.9%
7	1.2%	0.6%	0.8%	0.6%	5.1%	3.1%	3.0%	2.9%
	Dow Jones 30				S&P 500			
FF3	0.37	0.37	0.37	0.37	0.32	0.32	0.32	0.32
1	1.5%	1.2%	1.6%	1.6%	1.7%	2.2%	2.0%	1.7%
2	2.4%	2.1%	2.2%	1.7%	3.9%	3.3%	2.9%	3.1%
3	3.5%	4.1%	2.7%	2.7%	5.4%	5.0%	4.6%	3.9%
4	3.6%	4.0%	2.7%	3.7%	6.3%	6.0%	5.3%	5.1%
5	6.3%	6.9%	4.0%	3.8%	9.2%	9.5%	6.0%	6.1%
6	6.4%	5.0%	5.2%	5.0%	9.8%	8.3%	7.7%	7.1%
7	7.2%	6.7%	6.2%	6.0%	11.4%	9.2%	8.9%	8.4%

Table 3: **Economic Evidence: Pricing Errors** $\frac{1}{N}\sum_{i=1}^{N}\hat{\alpha}_i^2$

This table provides the results for pricing errors, which are listed in equation 22. The top-left panel shows results on train assets for the deep learning model. Other panels show results on test assets for evaluating the out-of-sample model evaluation. The benchmark model includes Fama-French three factors, which is listed in the first row (10^{-5}). We have independently trained $7 \times 4 = 28$ deep learning models in all sub-panels, with a different number of added deep factors and a different number of hidden layers. The numbers shown are *percentage changes* over the benchmark model. We have underlined the best model selected by the maximum pricing error reduction.

# Layers	$L=1$	$L=2$	$L=3$	$L=4$	$L=1$	$L=2$	$L=3$	$L=4$
# Factors	Bivariate-Sorted Portfolios				Univariate-Sorted Portfolios			
FF3	0.47	0.47	0.47	0.47	0.32	0.32	0.32	0.32
1	-16.3%	-17.0%	-12.4%	-23.2%	-10.8%	-11.9%	-8.8%	-16.6%
2	-15.5%	-8.1%	-29.0%	-19.1%	-14.6%	-3.5%	-21.6%	-13.2%
3	-28.5%	-21.0%	-24.2%	-29.1%	-21.2%	-16.9%	-18.4%	-22.8%
4	-27.8%	-32.6%	-19.1%	-16.0%	-23.3%	-26.7%	-13.8%	-10.1%
5	$\underline{-51.7\%}$	-39.2%	-30.6%	-22.9%	$\underline{-49.2\%}$	-36.8%	-23.3%	-13.8%
6	-33.4%	-30.6%	-12.1%	-28.6%	-32.5%	-28.1%	-8.8%	-22.6%
7	-32.2%	-21.0%	-36.6%	-31.9%	-39.5%	-13.4%	-31.2%	-28.7%
	Fama &French 25 Portfolios				Industry 49 Portfolios			
FF3	1.59	1.59	1.59	1.59	2.03	2.03	2.03	2.03
1	3.7%	3.6%	1.4%	3.3%	3.4%	1.5%	1.0%	2.2%
2	-3.2%	2.1%	4.7%	1.8%	-5.6%	1.9%	1.8%	-1.2%
3	2.2%	1.5%	5.5%	4.4%	-2.1%	-0.4%	4.9%	3.3%
4	2.0%	-2.1%	1.8%	4.2%	0.0%	-12.5%	-0.1%	3.1%
5	$\underline{-3.0\%}$	-8.0%	6.8%	3.0%	$\underline{-14.2\%}$	-16.3%	4.7%	1.9%
6	-4.2%	-2.7%	0.2%	4.8%	-8.5%	-8.6%	0.4%	5.6%
7	-16.3%	6.3%	4.5%	3.8%	-27.0%	3.8%	0.1%	-0.3%
	Dow Jones 30				S&P 500			
FF3	2.01	2.01	2.01	2.01	4.25	4.25	4.25	4.25
1	2.7%	-3.4%	-2.3%	-6.7%	1.7%	2.1%	0.0%	1.3%
2	-6.1%	-1.4%	-3.8%	1.0%	-1.1%	1.9%	2.4%	-1.1%
3	-4.8%	-7.0%	-1.5%	-3.7%	0.6%	-0.5%	1.9%	2.6%
4	8.8%	-3.0%	-2.0%	-4.5%	0.5%	-6.3%	-0.4%	1.7%
5	$\underline{-2.1\%}$	-7.6%	-1.5%	0.0%	$\underline{-2.2\%}$	-5.7%	3.5%	0.8%
6	-8.5%	-11.0%	3.0%	2.4%	0.8%	-4.6%	-0.7%	4.5%
7	3.5%	1.7%	2.2%	-7.8%	-9.1%	2.3%	2.7%	0.4%

Table 4: Economic Evidence: Cross-sectional R^2_{CS}

This table provides the results for cross-sectional R^2_{CS}, which is listed in equation 23. The top-left panel shows results on train assets for the deep learning model. Other panels shows results on test assets for evaluating the out-of-sample model evaluation. The benchmark model includes Fama-French three factors, which are listed in the first row. We have independently trained $7 \times 4 = 28$ deep learning models in all sub-panels, with a different number of added deep factors and a different number of hidden layers. The numbers shown are *percentage changes* over the benchmark model. We have underlined the best model selected by the maximum pricing error reduction.

# Layers	$L = 1$	$L = 2$	$L = 3$	$L = 4$	$L = 1$	$L = 2$	$L = 3$	$L = 4$
# Factors	Bivariate-Sorted Portfolios				Univariate-Sorted Portfolios			
FF3	0.52	0.52	0.52	0.52	0.25	0.25	0.25	0.25
1	8.9%	12.5%	16.6%	16.8%	30.8%	39.4%	50.6%	40.4%
2	19.1%	27.8%	26.5%	31.3%	58.1%	79.9%	96.2%	74.2%
3	25.5%	25.9%	23.7%	18.4%	71.9%	68.5%	77.3%	68.3%
4	26.7%	31.9%	31.4%	25.8%	99.3%	77.3%	55.1%	79.3%
5	$\underline{47.3\%}$	33.5%	41.9%	34.2%	$\underline{135.2\%}$	102.4%	81.4%	90.0%
6	32.0%	36.7%	29.1%	29.4%	95.5%	121.2%	77.2%	104.7%
7	37.9%	38.4%	34.2%	37.6%	102.9%	106.7%	97.9%	112.1%
	Fama &French 25 Portfolios				Industry 49 Portfolios			
FF3	0.63	0.63	0.63	0.63	0.10	0.10	0.10	0.10
1	25.3%	24.9%	28.5%	24.6%	68.3%	41.4%	45.8%	11.5%
2	27.6%	35.9%	33.7%	24.6%	245.6%	4.0%	96.4%	36.3%
3	30.8%	44.2%	28.1%	30.3%	183.0%	156.7%	187.3%	224.3%
4	35.5%	30.1%	44.4%	33.2%	117.8%	208.6%	187.0%	179.1%
5	$\underline{36.5\%}$	41.7%	32.8%	38.3%	$\underline{199.3\%}$	239.8%	266.2%	232.5%
6	36.9%	36.9%	31.4%	37.2%	91.5%	227.1%	149.0%	192.5%
7	37.1%	40.2%	49.4%	41.5%	287.2%	153.6%	224.9%	308.2%
	Dow Jones 30				S&P 500			
FF3	0.11	0.11	0.11	0.11	0.09	0.09	0.09	0.09
1	108.3%	57.4%	68.1%	0.3%	20.4%	14.3%	9.1%	22.1%
2	59.9%	3.2%	84.0%	101.8%	24.4%	39.7%	39.2%	53.0%
3	279.1%	276.4%	80.2%	115.8%	21.7%	28.9%	65.7%	38.5%
4	177.1%	77.9%	273.5%	156.3%	33.2%	60.0%	37.9%	30.2%
5	$\underline{382.7\%}$	248.5%	266.1%	344.6%	$\underline{39.8\%}$	72.4%	54.2%	27.3%
6	350.6%	232.0%	385.9%	310.5%	56.0%	79.6%	23.7%	53.7%
7	146.3%	453.6%	582.9%	142.9%	53.9%	72.1%	54.7%	48.1%

Table 5: **Interpreting Deep Factors**

This table provides the results for the annualized Sharpe ratio for factor investing (a portfolio for all factors), as well as the alpha t-statistics significance for the factor zoo (univariate- and bivariate-sorted factors). The left panel uses CAPM as the benchmark, and the right panel includes Fama-French three factors. The benchmark model results are listed in the first row. We have independently trained $7 \times 4 = 28$ deep learning models in all sub-panels, with a different number of added deep factors and a different number of hidden layers. The numbers shown are *percentage changes* over the benchmark model. We also conduct a squared Sharpe ratio test of Barillas et al. (2019) to show the significances of nested asset pricing model improvement. Respectively, $***$ is 1%, $**$ is 5%, and $*$ is 10%. We have underlined a few cases that are consistent in both train assets and test assets.

	# Layers	$L=1$	$L=2$	$L=3$	$L=4$	$L=1$	$L=2$	$L=3$	$L=4$
	# Factors	g = CAPM				g = Fama & French 3 Factors			
Sharpe Ratio	FF3	0.45	0.45	0.45	0.45	0.71	0.71	0.71	0.71
	1	4.9%	6.2%	4.6%	1.8%	4.2%	3.3%	1.6%	8.4%
	2	41.5%**	13.7%	11.9%	29.0%**	8.9%	1.8%	13.1%*	8.7%
	3	27.2%	43.5%**	25.2%	19.4%	16.7%*	7.1%	9.3%	14.0%
	4	79.6%***	81.2%***	9.7%	26.4%	20.6%**	18.7%*	5.5%	5.0%
	5	76.2%***	30.6%	80.5%***	10.5%	46.9%***	22.4%*	31.1%***	13.8%
	6	95.7%***	81.2%***	78.5%***	81.8%***	32.7%**	17.3%	14.9%	29.8%**
	7	82.0%***	60.8%**	43.7%	35.2%	51.5%***	24.9%	28.8%*	29.8%**
# Significance (Univariate)	FF3	30	30	30	30	26	26	26	26
	1	5	6	5	5	-1	-1	-1	-2
	2	-5	2	5	0	1	1	-2	-1
	3	-2	-4	6	0	0	-1	-1	-2
	4	-4	-6	6	2	-2	-2	-1	-1
	5	-3	-2	0	5	-4	0	-1	-2
	6	-6	-4	1	-1	-3	-1	0	-2
	7	-1	-2	1	-3	-2	0	-2	-2
# Significance (Bivariate)	FF3	44	44	44	44	46	46	46	46
	1	1	1	1	1	-1	-1	-2	-4
	2	-1	0	1	-1	-5	-1	-4	-3
	3	1	3	2	0	-6	-2	-2	-5
	4	0	-5	2	0	-2	-5	-2	-1
	5	-4	1	0	1	-8	-7	-5	-1
	6	-7	-2	1	-2	-4	-5	0	-2
	7	0	1	1	0	-7	-1	-6	-6

Figure 7: **Model Implied Return v.s. Average Return**

This figures provides a visualization of pricing errors for Fama & French 25 value-weighted portfolios and Industry 49 value-weighted portfolios. The top two figures use the time series model implied returns, and the bottom two use the cross-sectional model implied returns. We also plot the Fama-French three-factor model implied returns as the benchmark. The pricing errors are the distance between the scatter plotted points and the 45 degree line. The positive ones are below the 45 degree line, and the negative ones are above.

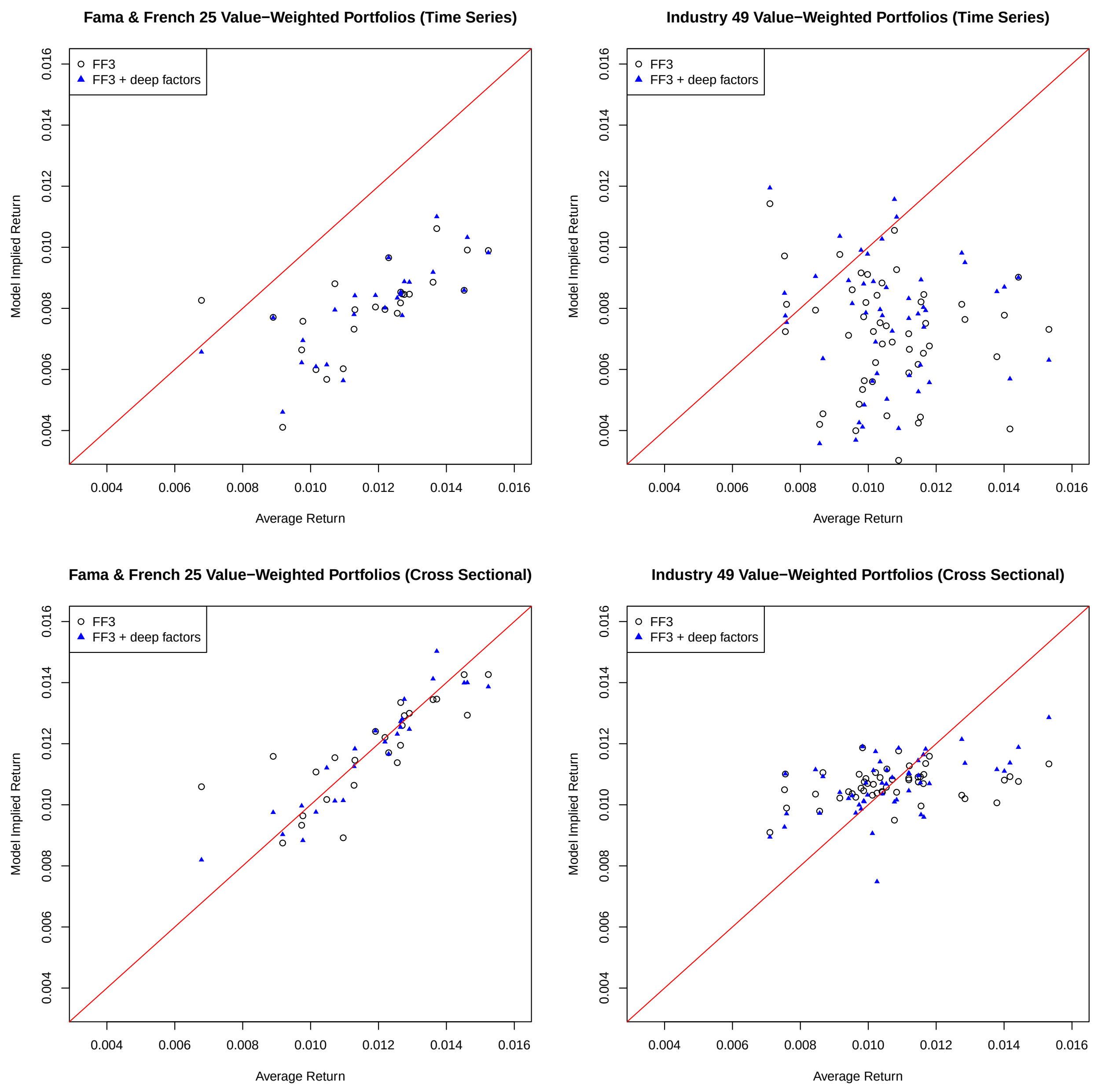


Figure 8: Deep Characteristics on Size and Value

This figure demonstrates the nonlinear relationship by plotting deep characteristics on market equity (size) and book-to-market (value). The listed results are based on the benchmark case (underlined in Table 2) on Fama-French three factors using bivariate-sorted portfolios, where SMB and HML are included in model training. We have 540 observations in total for 540 months of data. Notice that, the plots for market equity are converted in the log scale, while we use the original scale in model training.

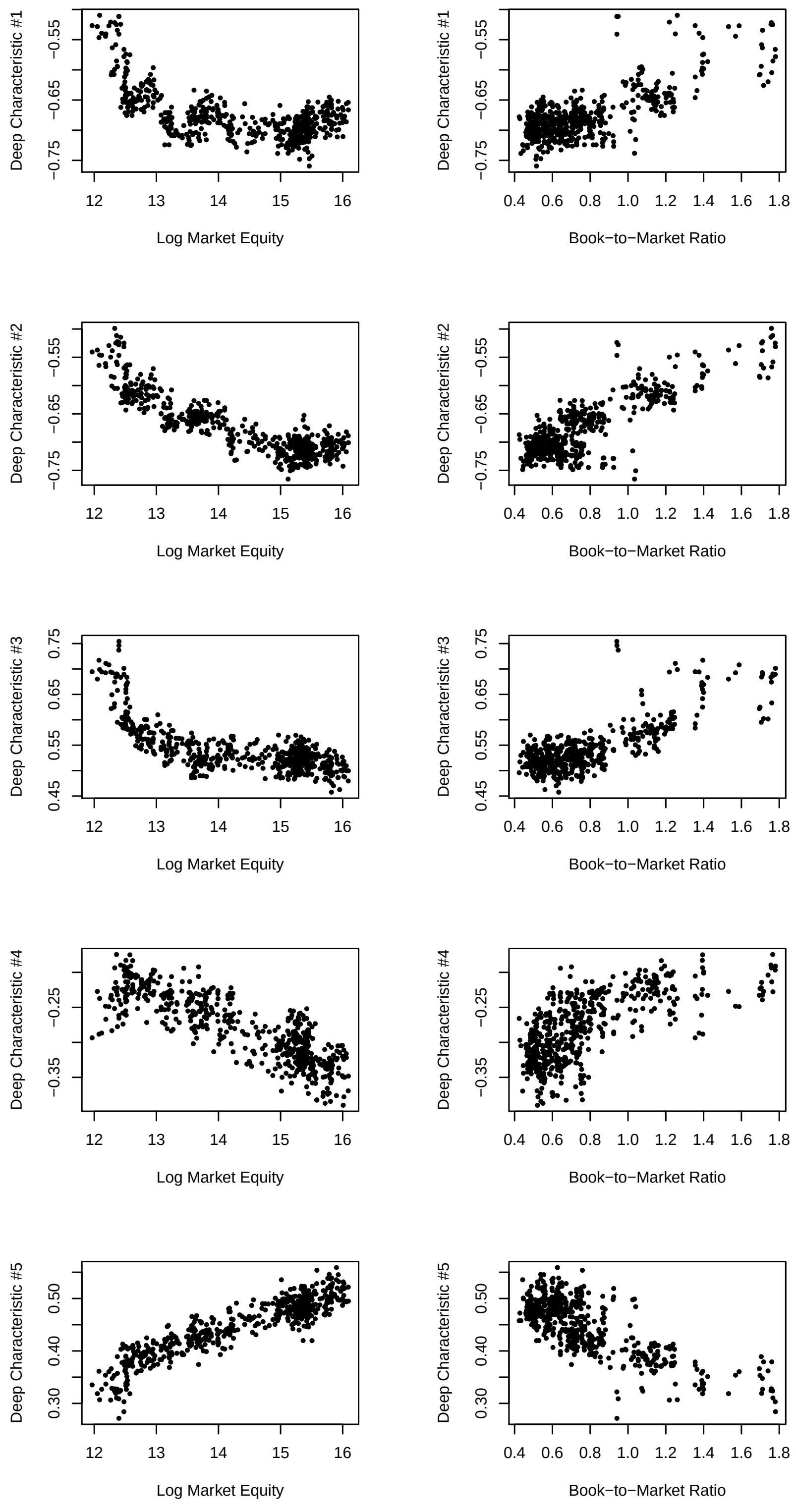


Table 6: Normalized Explained Variation for Deep Characteristics

This table provides the normalized explained variation of the deep characteristics through 62 raw characteristics. The listed results come from our benchmark case (underlined in Table 2) on Fama-French three factors using bivariate-sorted portfolios. In each month, we run a cross-sectional regression on each deep characteristic using 3,000 firm observations. To aggregate the coefficients through 540 months, we obtain the average exposure on deep characteristics through 62 raw characteristics.

Variable	Char #1	Char #2	Char #3	Char #4	Char #5
Lag market equity	0.22%	0.19%	0.23%	0.13%	0.14%
Employee growth rate	0.15%	0.13%	0.13%	0.13%	0.14%
Industry-adjusted change in profit margin	0.25%	0.21%	0.17%	0.63%	0.18%
Industry-adjusted change in asset turnover	0.21%	0.17%	0.19%	0.18%	0.19%
Industry-adjusted book to market	0.37%	0.29%	0.42%	0.27%	0.31%
Industry Concentration	0.09%	0.07%	0.10%	0.08%	0.09%
Number of earnings increases	0.11%	0.09%	0.10%	0.39%	0.08%
Dollar trading volume	0.35%	1.72%	0.32%	1.26%	0.56%
Share turnover	0.24%	0.38%	0.26%	0.48%	0.28%
Industry momentum	0.19%	0.13%	0.15%	0.13%	0.15%
Maximum daily return	0.20%	0.24%	0.21%	0.21%	0.22%
Volatility of liquidity (dollar trading volume)	0.21%	0.20%	0.21%	0.23%	0.19%
Volatility of liquidity (share turnover)	0.35%	0.21%	0.27%	0.24%	0.24%
Illiquidity	0.12%	0.12%	0.13%	0.11%	0.11%
Zero trading days	0.28%	0.31%	0.12%	0.55%	0.13%
Price delay	0.08%	0.08%	0.08%	0.08%	0.08%
Working capital accruals	0.16%	0.14%	0.14%	0.11%	0.10%
Asset growth	0.53%	0.51%	0.45%	0.39%	0.44%
Book-to-market	0.42%	0.38%	0.67%	0.37%	0.51%
Cash flow to price ratio	0.97%	0.66%	0.58%	0.65%	0.53%
Earnings to price	1.33%	0.81%	0.65%	0.95%	0.59%
Payout yield	0.28%	0.21%	0.20%	0.19%	0.19%
Net income	0.20%	0.18%	0.17%	0.15%	0.16%
Dividend to price	0.15%	0.16%	0.13%	0.17%	0.14%
Unexpected quarterly earnings	0.15%	0.13%	0.14%	0.12%	0.13%
Revenue surprise	0.17%	0.13%	0.16%	0.21%	0.17%
Cash holdings	0.84%	1.05%	0.38%	0.17%	0.18%

Variable	Char #1	Char #2	Char #3	Char #4	Char #5
Change in shares outstanding	0.12%	0.11%	0.13%	0.11%	0.12%
Cash flow to debt	0.27%	0.36%	0.29%	0.22%	0.25%
Percent accruals	0.13%	0.10%	0.11%	0.13%	0.17%
Gross profitability	0.39%	0.33%	0.99%	0.28%	0.42%
Leverage	1.33%	0.49%	1.57%	1.00%	0.58%
R&D to market capitalization	0.12%	0.12%	0.13%	0.12%	0.12%
Sales growth	0.21%	0.27%	0.22%	0.17%	0.20%
Sales to price	0.18%	0.24%	0.54%	0.18%	0.18%
Capital expenditures and inventory	0.20%	0.20%	0.20%	0.19%	0.21%
R&D to sales	0.38%	0.49%	0.34%	0.30%	0.33%
Financial statements score	0.13%	0.14%	0.15%	0.14%	0.16%
Growth in long-term debt	0.32%	0.27%	0.22%	0.23%	0.24%
Return on assets	0.54%	0.74%	0.53%	0.31%	0.40%
Depreciation / PP&E	0.14%	0.13%	0.11%	0.10%	0.11%
Growth in common shareholder equity	0.27%	0.16%	0.12%	0.13%	0.16%
Growth in long term net operating assets	0.15%	0.15%	0.15%	0.13%	0.15%
Change in profit margin	0.21%	0.17%	0.14%	0.13%	0.13%
Change in asset turnover	0.22%	0.25%	0.20%	0.24%	0.21%
Change in tax expense	0.13%	0.13%	0.11%	0.09%	0.11%
Net Operating Assets	0.30%	0.32%	0.56%	0.16%	0.17%
Return on net operating assets	0.37%	0.79%	0.91%	0.23%	0.24%
Profit margin	0.45%	0.65%	0.41%	0.36%	0.35%
Asset turnover	0.68%	0.39%	0.61%	0.41%	0.94%
Bid-ask spread	0.35%	0.65%	0.27%	0.37%	0.35%
Residual variance w.r.t. FF3	**24.06%**	**24.51%**	**25.27%**	**24.65%**	**25.80%**
Residual variance w.r.t. CAPM	**42.01%**	**41.85%**	**43.18%**	**43.86%**	**43.79%**
Stock variance	**15.73%**	**15.13%**	**13.94%**	**15.51%**	**16.08%**
Market Beta	0.29%	0.12%	0.17%	0.10%	0.10%
1-month momentum	0.17%	0.17%	0.17%	0.16%	0.18%
12-month momentum	0.48%	0.51%	0.47%	0.43%	0.48%
60-month momentum	0.14%	0.12%	0.19%	0.13%	0.13%
36-month momentum	0.17%	0.23%	0.13%	0.15%	0.15%
6-month momentum	0.19%	0.20%	0.18%	0.19%	0.19%
Seasonality	0.57%	0.62%	0.54%	0.52%	0.58%

Appendix A Optimization Details

This section shows how we minimize our objective function to train the deep learner. The common techniques include stochastic gradient descent (SGD), dropout, and ensemble learning. In the model training, we only apply SGD.

The new technology for deep learning that allows us to train such a complex bottom-up system is, the structure of the deep learner makes its objective function differentiable with respect to its parameters. The first-order derivative information is directly available by carefully applying the backward-chain rule. The TensorFlow library performs automatic derivative calculation for practitioners, allowing us to train the model using SGD.[6] Let the superscript (t) denote the t-th iterate. SGD updates the parameters by

$$\begin{bmatrix} \hat{A}^{(t+1)} \\ \hat{b}^{(t+1)} \\ \hat{\beta}^{(t+1)} \\ \hat{\gamma}^{(t+1)} \end{bmatrix} \longleftarrow \begin{bmatrix} \hat{A}^{(t)} \\ \hat{b}^{(t)} \\ \hat{\beta}^{(t)} \\ \hat{\gamma}^{(t)} \end{bmatrix} - \eta^{(t+1)} \nabla \mathcal{L}_{\lambda}^{(t)} \tag{26}$$

until convergence, where η is the step size, and the gradient is evaluated at $(\hat{A}^{(t)}, \hat{b}^{(t)}, \hat{\beta}^{(t)}, \hat{\gamma}^{(t)})$. At each iterate, the loss $\mathcal{L}_{\lambda}^{(t)}$ only involves a random subset of data, $\mathcal{B} \subset \{1, 2, ..., T\}$, called mini-batch,

$$\mathcal{L}_{\lambda}^{(t)}(A, b, \beta, \gamma) = \frac{1}{N|\mathcal{B}|} \sum_{t \in \mathcal{B}} \sum_{i=1}^{N} \left(R_{i,t} - \widehat{R}_{i,t} \right)^2 + \frac{\lambda}{N} \sum_{i=1}^{N} \left(\frac{1}{|\mathcal{B}|} \sum_{t \in \mathcal{B}} (R_{i,t} - \widehat{R}_{i,t}) \right)^2, \tag{27}$$

where $|\mathcal{B}| < T$, and in practice we set $|\mathcal{B}| = 120$; namely, we use a batch of 120 months for training. This mini-batch setting on the time dimension is reasonable for the asset pricing factor model, which we usually assume with no serial correlation.

Also, we set the number of epochs (roughly the number of times SGD explores the whole training set) to be 300, because the objective function has stopped decreasing significantly. Adding too many epochs for model training can cause over-fitting. In our study, we consider 300 epochs a reasonable number. Figure 9 gives examples of the objective function decreasing during training. The loss curve is almost flat on the right tale at the log scale. Another possibility for improving the

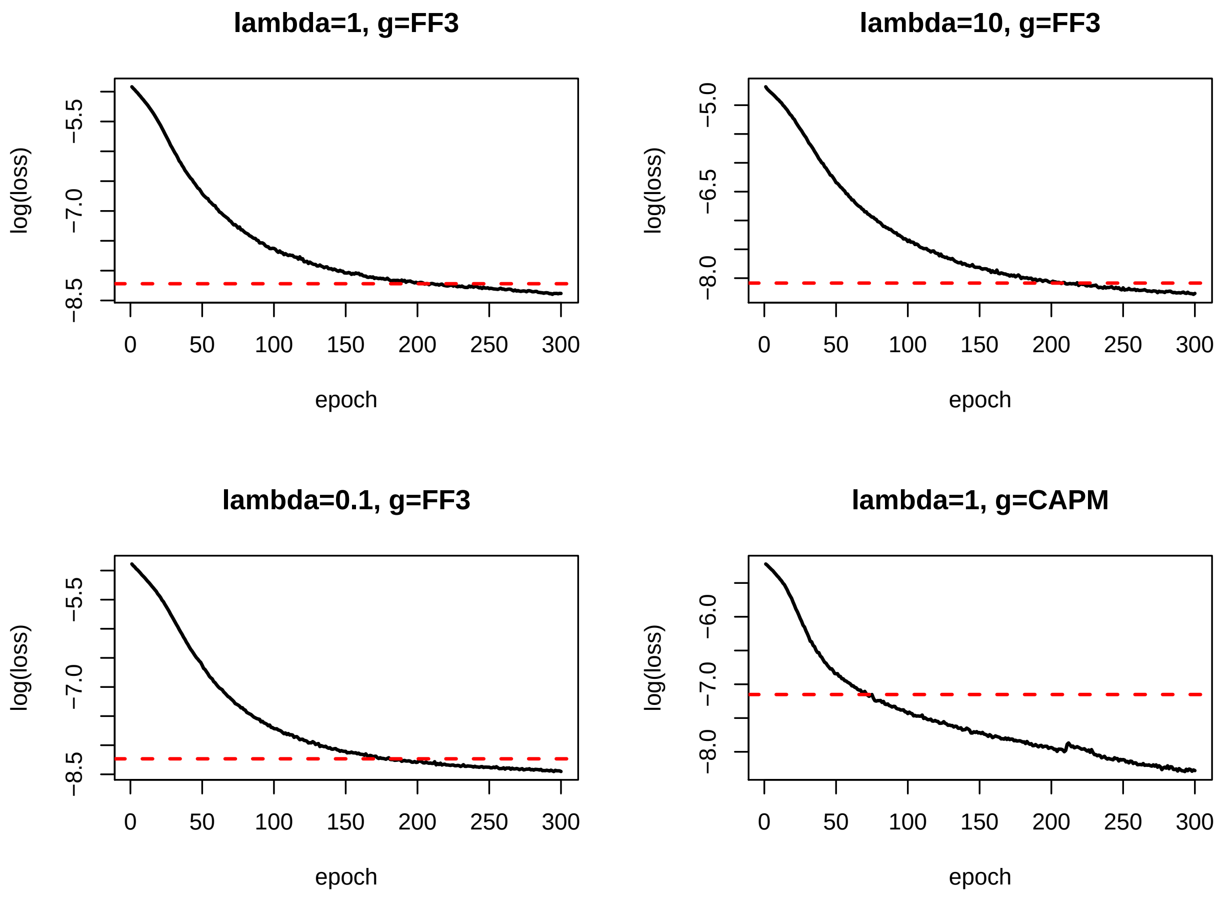
[6] See Robbins and Monro (1951), Kiefer and Wolfowitz (1952).

results is to add an ensemble to average out the predictive variance.

The below loss curves are from our benchmark case in the empirical results. All four models have the same architecture $(L = 1, P = 5)$ but with different benchmark models g or regularization parameter λ. We plot the objective functions versus the number of epochs. In all four cases, the objective functions with respect to the training data decrease as SGD goes on and almost converges. The red dashed lines represent the loss level for the Fama-French 3-factor model or CAPM. We find a slight improvement over the Fama-French 3-factor model but a dramatic improvement over CAPM.

Figure 9: **Objective Function of the Training Data vs. # Epochs**

This figure provides an example of the objective function with respect to the training data decreasing as SGD iterates. All four models have the same architecture $(L = 1, P = 5)$ but with different models g or λ. The red dashed lines represent the loss level for the Fama-French 3-factor model or CAPM.



Appendix B Robustness Check for Different Tuning Parameters

This table provides the robust results for our deep learning model training using different tuning parameters, λ, in equation 9. The main empirical results use the one with $\lambda = 1$, and the two panels below show $\lambda = 0.1$ and $\lambda = 10$ for consistent results. The top-left panel shows results on train assets for the deep learning model. The benchmark model includes Fama-French three factors, which are listed in the first row. We have independently trained $7 \times 4 = 28$ deep learning models in all sub-panels, with a different number of added deep factors and a different number of hidden layers. The numbers shown are *percentage changes* over the benchmark model.

	# Layers	$L = 1$	$L = 2$	$L = 3$	$L = 4$	$L = 1$	$L = 2$	$L = 3$	$L = 4$
	# Factors	$\lambda = 0.1$				$\lambda = 10$			
Time Series R^2	FF3	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92
	1	1.5%	1.3%	1.5%	1.6%	1.5%	1.2%	1.6%	1.2%
	2	1.8%	1.5%	1.6%	1.4%	1.3%	1.6%	1.6%	1.6%
	3	1.9%	2.1%	1.9%	1.6%	1.9%	2.1%	1.9%	2.1%
	4	2.4%	2.0%	1.6%	1.9%	2.9%	1.9%	2.0%	1.7%
	5	1.7%	2.1%	2.0%	1.9%	2.4%	2.1%	1.7%	1.6%
	6	2.6%	2.4%	1.8%	1.9%	2.2%	1.9%	2.1%	1.6%
	7	2.5%	2.7%	1.7%	2.4%	2.4%	2.1%	2.3%	2.0%
Pricing Errors	FF3	0.47	0.47	0.47	0.47	0.47	0.47	0.47	0.47
	1	-23.5%	-16.8%	-28.1%	-29.5%	-12.7%	-12.7%	-16.3%	-20.5%
	2	-28.3%	-26.9%	-22.6%	-21.8%	-18.2%	-17.2%	-19.6%	-12.5%
	3	-44.4%	-32.6%	-21.3%	-30.4%	-21.6%	-32.1%	-14.4%	-17.1%
	4	-50.0%	-30.3%	-12.7%	-24.1%	-48.3%	-8.9%	-17.9%	-22.0%
	5	-32.2%	-31.6%	-26.1%	-17.4%	-31.8%	-17.5%	-33.5%	-21.9%
	6	-34.3%	-24.1%	-27.3%	-32.9%	-30.4%	-9.0%	-41.9%	-9.3%
	7	-54.4%	-28.0%	-27.8%	-30.8%	-16.2%	-26.2%	-42.8%	-19.7%
Sharpe Ratio	FF3	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.71
	1	7.8%	3.0%	9.9%	13.6%	1.6%	2.5%	3.3%	6.6%
	2	15.8%	10.3%	11.2%	9.4%	4.1%	4.8%	12.5%	1.9%
	3	39.8%	19.5%	11.4%	18.1%	13.7%	18.9%	9.9%	8.4%
	4	42.4%	13.2%	13.1%	9.8%	35.3%	9.3%	10.9%	14.8%
	5	17.8%	33.8%	19.7%	15.8%	21.7%	5.8%	29.6%	29.7%
	6	26.6%	10.5%	39.5%	32.6%	26.4%	7.4%	52.1%	3.7%
	7	109.1%	15.1%	36.7%	18.3%	64.5%	24.8%	38.0%	20.9%

Appendix C Squared Sharpe Ratio Test in Barillas et al. (2019)

Since the benchmark model g is nested in the augmented model F, the Sharpe ratio of F is greater or equal to that of g. The improvement in the squared Sharpe ratio is a quadratic form as shown by Equation (2) in Barillas et al. (2019),

$$SR_F^2 - SR_g^2 = \alpha_f^\intercal \Sigma_\epsilon^{-1} \alpha_f$$

where α_f is the $N \times 1$ intercept vector from the pricing model

$$f_t = \alpha_f + \tilde{\beta} g_t + \epsilon_t,\ t = 1, 2..., T$$

and Σ_ϵ is covariance matrix of ϵ_t. Therefore, the simple test of equality in the Sharpe ratios is actually the GRS test, with the tradable deep factors f_t serving as left-hand-side test assets on the right-hand-side benchmark g_t.

Under the null hypothesis $H_0 : SR_F = SR_g$, i.e. $\alpha_f^\intercal \Sigma_\epsilon^{-1} \alpha_f = 0$, the GRS test statistic is proportional to the difference in squared sample Sharpe ratios divided by one plus the squared sample Sharpe ratio of g,

$$\begin{aligned}
\left(\frac{T}{P}\right)\left(\frac{T-P-D}{T-D-1}\right)\frac{\widehat{SR}_F^2 - \widehat{SR}_g^2}{1+\widehat{SR}_g^2} &\sim F(P, T-P-D) \\
\widehat{SR}_F^2 - \widehat{SR}_g^2 &= \hat{\alpha}_f^\intercal \hat{\Sigma}_\epsilon^{-1} \hat{\alpha}_f \\
\widehat{SR}_g^2 &= \bar{g}^\intercal \hat{\Sigma}_g^{-1} \bar{g},
\end{aligned}$$

where $\bar{g}, \hat{\Sigma}_g$ are the sample mean and covariance matrix of benchmark factors g_t.