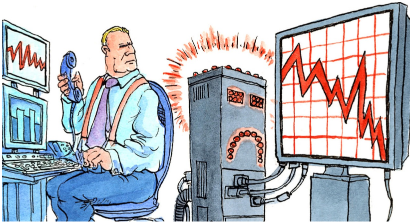
Deep Learning in Asset Pricing

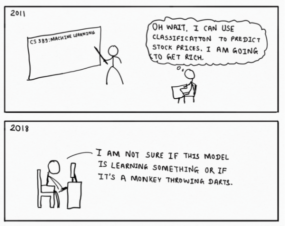
Luyang Chen [†§] Markus Pelger [‡§] Jason Zhu [‡§]

[†]Institute for Computational & Mathematical Engineering, Stanford University
[‡]Department of Management Science & Engineering, Stanford University
[§]Advanced Financial Technologies Laboratory, Stanford

May 28, 2019

Hype: Machine Learning in Finance





- Portfolio Management
- Algorithmic Trading
- Fraud Detection
- Etc ...

Motivation: Asset Pricing

The Challenge of Asset Pricing

- One of the most important questions in finance is:
  Why are asset prices different for different assets?
- No-Arbitrage Pricing Theory:
  Stochastic discount factor (SDF) explains differences in risk and asset prices.
- Fundamental Question:
  What is the SDF?
- Challenges:
  - SDF should depend on all available economic information
  - Functional form of SDF is unknown and likely conplex
  - SDF needs to capture time-variation in economic conditions
  - Risk premium in stock returns has a low signal-to-noise ratio

This paper

Goals of this paper:

General non-linear asset pricing model and optimal portfolio design

$\Rightarrow$ Deep-neural networks applied to all U.S. equity data and large sets of macroeconomic and firm-specific information.

Why is it important?

1. Stochastic discount factor (SDF) generates tradeable portfolio with highest risk-adjusted return (Sharpe-ratio=expected excess return/standard deviation)
2. Arbitrage opportunities
   - Find underpriced assets and earn "alpha"
3. Risk management
   - Understand which information and how it drives the SDF
   - Manage risk exposure of financial assets

Contribution of this paper

- This Paper: Estimate the SDF with deep neural networks
- Crucial innovation: Include no-arbitrage condition in the neural network algorithm and combine four neural networks in a novel way
- Key elements of estimator:
    1. Non-linearity: Feed-forward network captures non-linearities
    2. Time-variation: Recurrent (LSTM) network finds a small set of economic state processes
    3. Pricing all assets: Generative adversarial network identifies the states and portfolios with most unexplained pricing information
    4. Dimension reduction: Regularization through no-arbitrage condition
    5. Signal-to-noise ratio: No-arbitrage conditions improve the risk premium signal

$\Rightarrow$ General model that includes all existing models as a special case

Contribution of this paper

Empirical Contributions

- Empirically outperforms all benchmark models.
- Optimal portfolio has out-of-sample annual Sharpe ratio of 2.6.
- Non-linearities and interaction between firm information matters.
- Most relevant firm characteristics are price trends, profitability, and capital structure variables.

Literature (Partial List)

- Deep-learning for predicting asset prices
  - Feng, Polson and Xu (2019)
  - Gu, Kelly and Xiu (2018)
  - Feng, Polson and Xu (2018)
  - Messmer (2017)

  ⇒ Predicting future asset returns with feed forward network
  - Gu, Kelly and Xiu (2019)
  - Heaton, Polson and Witte (2017)

  ⇒ Fitting asset returns with an autoencoder
- Linear or kernel methods for asset pricing of large data sets
  - Lettau and Pelger (2018): Risk-premium PCA
  - Feng, Giglio and Xu (2017): Risk-premium lasso
  - Freyberger, Neuhierl and Weber (2017): Group lasso
  - Kelly, Pruitt and Su (2018): Instrumented PCA

No-Arbitrage Pricing Theory

- A stochastic discount factor[1] is a stochastic process $\{M_t\}$, such that for any asset i with payoff $x_{i,t+1}$ at time $t+1$, the price of that asset at time t is

$$P_{i,t} = \mathbb{E}_t[M_{t+1} x_{i,t+1}].$$

- Let $R^e_{i,t+1} = R_{i,t+1} - R_f$. **Fundamental no-arbitrage condition**:

$$\mathbb{E}_t[M_{t+1} R^e_{i,t+1}] = 0.$$

- It implies infinitely many **unconditional moments**:

$$\mathbb{E}[M_{t+1} R^e_{i,t+1} \hat{I}_{i,t}] = 0$$

for any $\mathcal{F}_t$-measurable variable $\hat{I}_{i,t}$.

[1] Examples of SDF are included in the appendix [39]-[40].

Model

- Without loss of generality, SDF is the projection on the return space[2]

$$M_{t+1} = 1 - \sum_{i=1}^{N} w_{i,t} R^e_{i,t+1}.$$

⇒ The optimal portfolio $F_{t+1} = \sum_{i=1}^{N} w_{i,t} R^e_{i,t+1}$ has the highest conditional Sharpe ratio.

- The portfolio weights $w_{i,t}$ are a general function of macro-economic information I_t and firm-specific characteristics $I_{i,t}$:

$$w_{i,t} = w(I_t, I_{i,t}).$$

⇒ Need non-linear estimator with many explanatory variables!

⇒ We use neural networks to estimate $w_{i,t}$.

[2] See e.g. Back [2010]. The SDF is an affine transformation of the tangency portfolio.

Equivalent Factor Model Representation

- No-arbitrage condition is equivalent to

$$\mathbb{E}_t[R^e_{i,t+1}] = \frac{\mathrm{cov}_t(R^e_{i,t+1}, F_{t+1})}{\mathrm{var}_t(F_{t+1})} \cdot \mathbb{E}_t[F_{t+1}] \\ = \beta_{i,t}\mathbb{E}_t[F_{t+1}]$$

with factor $F_t = 1 - M_t$.

$\Rightarrow$ Without loss of generality we have a factor representation

$$R^e_{t+1} = \beta_t F_{t+1} + \epsilon_{t+1}.$$

Estimation

- Estimate SDF portfolio weights $w(\cdot)$ to minimize the no-arbitrage moment conditions.
- For a set of conditioning variables $\hat{I}_{i,t} = \hat{g}(I_t, I_{i,t})$, the corresponding loss function is

$$L(w|\hat{g}, I_t, I_{i,t}) = \frac{1}{N}\sum_{i=1}^{N}\frac{T_i}{T}\left\|\frac{1}{T_i}\sum_{t\in T_i} M_{t+1}R^e_{i,t+1}\hat{g}(I_t, I_{i,t})\right\|^2.$$

- How can we choose the conditioning variables $\hat{I}_{i,t}$ as general functions of the macroeconomic and firm-specific information?

$\Rightarrow$ Generative Adversarial Network (GAN)[3] chooses $\hat{g}$!

[3] A brief introduction of GAN by Goodfellow et al. [2014] is included in the appendix [41].

Generative Adversarial Network (GAN)

Formulate GMM as Zero-Sum Game

- Two networks play a zero-sum game:
  1. **SDF Network** (w) creates the SDF M_{t+1}.
  2. **Conditional Network** ($\hat{g}$) generates conditioning variables $\hat{I}_{i,t}$.
- Alternatively update the two networks[a]:
  1. For a given set of conditioning variables $\hat{I}_{i,t}$, **SDF network** is updated to minimize the loss.
  2. For a given estimation of the SDF, **Conditional Network** finds $\hat{I}_{i,t}$ with the largest loss (most mis-pricing).
- Intuition: find the economic states and assets with the most pricing information.

[a] Model calibration details are included in the appendix [46] and [47].

Neural Network Building Blocks

SDF Network and Conditional Network are independent, but share a similar structure.

- Feedforward network[4] captures non-linearities.
- Recurrent network with LSTM cells[5] transforms all macroeconomic time-series into a low dimensional vector of stationary state variables.
  - Time-series data is often non-stationary.
  - Business cycles can affect pricing.
  - Redundant information.

[4] The definition of feedforward network is included in the appendix [42].
[5] The definition of LSTM cells is included in the appendix [44] and [45].

Model Architecture

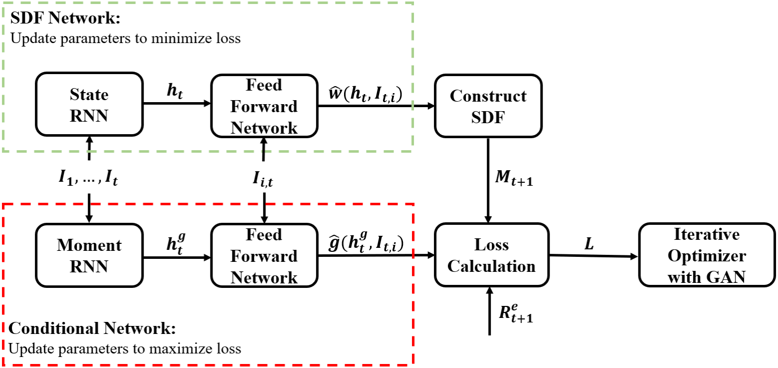


Figure: Model Architecture

Simulation Results - Setup

- Excess returns follow a no-arbitrage model with SDF factor F

$$R^e_{i,t+1} = \beta_{i,t} F_{t+1} + \epsilon_{i,t+1}.$$

- The SDF factor follows $F_t \overset{i.i.d.}{\sim} \mathcal{N}(\mu_F, \sigma_F^2)$ with $\sigma_F^2 = 0.1$ and $SR_F = 1$.
- The idiosyncratic component $\epsilon_{i,t} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_e^2)$ with $\sigma_e^2 = 1$.
- $N = 500$ and $T = 600$. Training/validation/test split is 250,100,250.
- One characteristic and one macroeconomic state process:

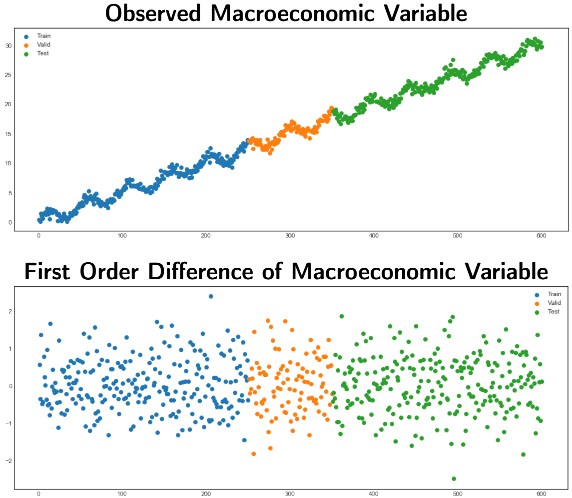
$$\beta_{i,t} = C^{(1)}_{i,t} \cdot b(h_t), \qquad h_t = sin(\pi * t/24) + \epsilon^h_t.$$

$$b(h) = \begin{cases} 1 & \text{if } h > 0 \\ -1 & \text{otherwise.} \end{cases}$$

We observe only the macroeconomic time-series $Z_t = \mu_M t + h_t$. All innovations are independent and normally distributed: $C^{(1)}_{i,t} \overset{i.i.d.}{\sim} \mathcal{N}(0, 1)$ and $\epsilon^h_t \overset{i.i.d.}{\sim} \mathcal{N}(0, 0.25)$. [6]

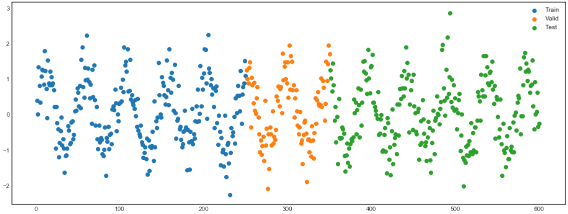
[6] More simulation results are included in the appendix [48]-[50].

Simulation Results - Observed Macroeconomic Variable

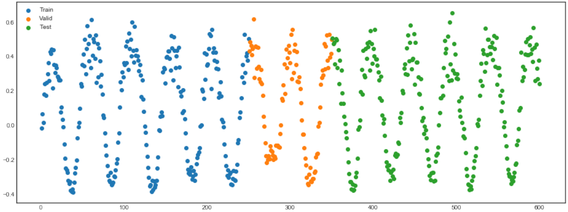


Simulation Results - Fitted Macroeconomic State

True Hidden Macroeconomic State



Fitted Macroeconomic State by LSTM



Simulation Results - Evaluation

Table: Performance of Different SDF Models

	Sharpe Ratio			EV			Cross-sectional R^2		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Population	0.89	0.92	0.86	0.18	0.18	0.17	0.19	0.20	0.15
GAN	0.79	0.77	0.64	0.18	0.18	0.17	0.19	0.20	0.15
FFN	0.05	-0.05	0.06	0.02	0.01	0.02	0.01	0.01	0.02
LS	0.12	-0.05	0.10	0.16	0.16	0.15	0.15	0.18	0.14

Empirical Results - Data

- 50 years of monthly observations: 01/1967 - 12/2016.
- Monthly stock returns for all U.S. securities from CRSP (around 31,000 stocks)
  Use only stocks with with all firm characteristics (around 10,000 stocks)
- 46 firm-specific characteristics for each stock and every month (usual suspects) $\Rightarrow I_{i,t}$
  normalized to cross-sectional quantiles
- 178 macroeconomic variables
  (124 from FRED, 46 cross-sectional median time-series for characteristics, 8 from Goyal-Welch) $\Rightarrow I_t$
- T-bill rates from Kenneth-French website
- Training/validation/test split is 20y/5y/25y

Empirical Results - Benchmark Models

1. LS & EN - Linear factor models:
   The optimal portfolio weights $w_t = I_t\theta$ is linear in characteristics. We minimize loss function

$$\frac{1}{2}\left\|\frac{1}{T}\tilde{R}^{K\top}1 - \frac{1}{T}\tilde{R}^{K\top}\tilde{R}^{K}\theta\right\|_2^2 + \lambda_1\|\theta\|_1 + \frac{1}{2}\lambda_2\|\theta\|_2^2.$$

   $\tilde{R}^K_{t+1} = I_t^\top R^e_{t+1}$ are K portfolios weighted by characteristics I_t.
2. FFN - Deep learning return forecasting (Gu et al. [2018]):
   - Predict conditional expected returns $\mathbb{E}_t[R_{i,t+1}]$
   - Empirical loss function for prediction

$$\frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\left(R_{i,t+1} - g(I_t, I_{i,t})\right)^2$$

   - Use only simple feedforward network for forecasting

Empirical Results - Evaluation

Objects of Interest:

- The SDF factor F_t
- The risk loadings β_t[7]
- The unexplained residual
  $\hat{e}_t = (I_N - \beta_{t-1}(\beta_{t-1}^\top \beta_{t-1})^{-1} \beta_{t-1}^\top) R_t^e$

Performance Measure:

- Sharpe ratio of SDF factor: $SR = \frac{\hat{\mathbb{E}}[F_t]}{\sqrt{\widehat{Var}(F_t)}}$
- Explained variation: $EV = 1 - \frac{\left(\frac{1}{T}\sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} (\hat{\epsilon}_{i,t+1})^2\right)}{\left(\frac{1}{T}\sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} (R_{i,t+1}^e)^2\right)}$
- cross-sectional mean R^2:
  $$\text{XS-}R^2 \;= 1 - \frac{\frac{1}{N}\sum_{i=1}^{N} \frac{T_i}{T}\left(\frac{1}{T_i}\sum_{t\in T_i} \hat{\epsilon}_{i,t+1}\right)^2}{\frac{1}{N}\sum_{i=1}^{N} \frac{T_i}{T}\left(\frac{1}{T_i}\sum_{t\in T_i} \hat{R}_{i,t+1}\right)^2}$$

[7] We estimate loadings by fitting a feedforward network to predict $R_t F_t$.

Empirical Results - Cross Section of Individual Stock Returns

Table: Performance of Different SDF Models

	SR			EV			Cross-Sectional R^2		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
LS	1.80	0.58	0.42	0.09	0.03	0.03	0.15	0.00	0.14
EN	1.37	1.15	0.50	0.12	0.05	0.04	0.17	0.02	0.19
FFN	0.45	0.42	0.44	0.11	0.04	0.04	0.14	-0.00	0.15
GAN	2.68	1.43	0.75	0.20	0.09	0.08	0.12	0.01	0.23

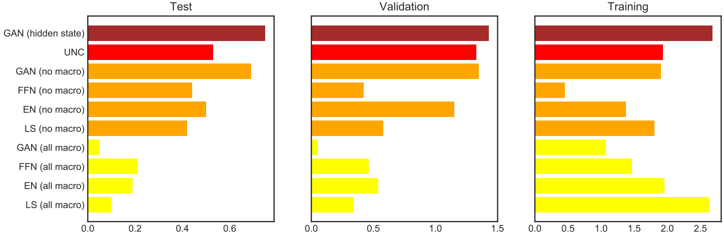
Empirical Results - Cross Section of Individual Stock Returns

Table: SDF Factor Portfolio Performance[8]

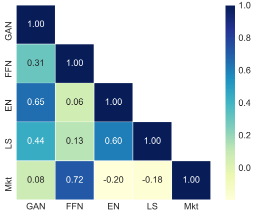
	SR			Max Loss			Max Drawdown		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
FF-3	0.27	-0.09	0.19	-2.45	-2.85	-4.31	7	10	10
FF-5	0.48	0.40	0.22	-2.62	-2.33	-4.90	4	3	7
LS	1.80	0.58	0.42	-1.96	-1.87	-4.99	1	3	4
EN	1.37	1.15	0.50	-2.22	-1.81	-6.18	1	3	5
FFN	0.45	0.42	0.44	-3.30	-4.61	-3.37	6	3	5
GAN	2.68	1.43	0.75	0.38	-0.28	-5.76	0	1	5

[8] Turnover as a measure of transaction costs is included in the appendix [51].

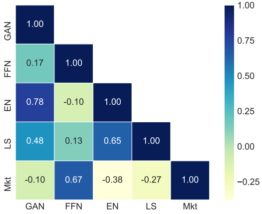
Performance of Models with Different Macroeconomic Variables



Empirical Results - SDF Factors and Market Factor



(a) Whole Time Horizon



(b) Test Period

Figure: Correlation between SDF Factors for Different Models

$\Rightarrow$ GAN SDF factor has a small correlation with the market factor.

Empirical Results - SDF Factor and Fama-French Factors

Table: GAN-SDF Factor and Fama-French 5 Factors

	Mkt-RF	SMB	HML	RMW	CMA	intercept
Regression Coefficients	0.00	0.00	-0.04	0.08***	0.04	0.76***
	(0.02)	(0.02)	(0.03)	(0.03)	(0.04)	(0.06)
Correlations	-0.10	-0.09	0.01	0.17	0.05	-

$\Rightarrow$ Fama-French factors do not span SDF.

Empirical Results - Size Effect

Table: Performance of Different SDF Models with Size Thresholds

	SR			EV			Cross-Sectional R^2		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Size $\geq$ 0.001% of total market cap									
LS	1.44	0.31	0.13	0.07	0.05	0.03	0.14	0.03	0.10
EN	0.93	0.56	0.15	0.11	0.09	0.06	0.17	0.05	0.14
FFN	0.42	0.20	0.30	0.11	0.10	0.05	0.19	0.08	0.18
GAN	2.32	1.09	0.41	0.23	0.22	0.14	0.20	0.13	0.26
Size $\geq$ 0.01% of total market cap									
LS	0.32	-0.11	-0.06	0.05	0.07	0.04	0.13	0.05	0.09
EN	0.37	0.26	0.23	0.09	0.12	0.07	0.17	0.08	0.14
FFN	0.32	0.17	0.24	0.13	0.22	0.09	0.22	0.15	0.26
GAN	0.97	0.54	0.26	0.28	0.34	0.18	0.27	0.23	0.32

Empirical Results - Predictive Performance

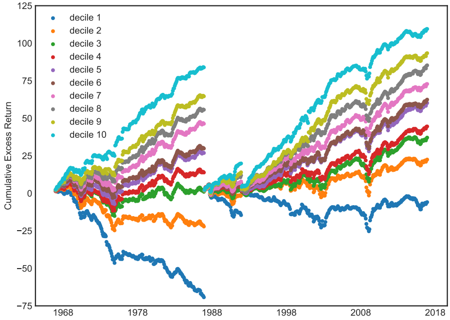
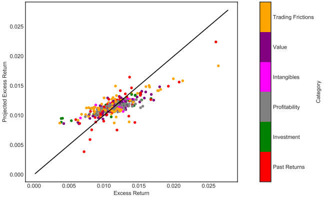


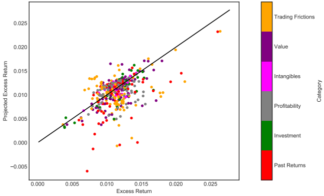
Figure: Cumulative Excess Returns of Decile Sorted Portfolios by GAN

$\Rightarrow$ Risk loadings predicts future stock returns.

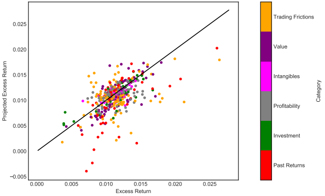
Empirical Results - Predictive Performance



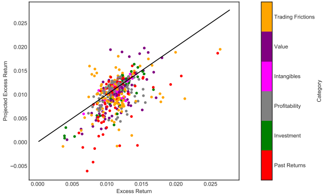
(a) GAN



(b) FFN



(c) EN



(d) LS

Figure: Projected Excess Return of Decile Sorted Portfolios

Empirical Results - Predictive Performance

Table: Time Series Pricing Errors for β-Sorted Portfolios

	Average Returns		Fama-French 3				Fama-French 5			
Decile	Whole	Test	Whole		Test		Whole		Test	
			α	t	α	t	α	t	α	t
1	-0.12	-0.02	-0.21	-12.77	-0.13	-5.01	-0.20	-11.99	-0.12	-4.35
2	-0.00	0.05	-0.09	-8.79	-0.05	-3.22	-0.09	-8.29	-0.05	-2.68
3	0.04	0.08	-0.04	-5.18	-0.02	-1.40	-0.04	-4.87	-0.01	-1.05
4	0.07	0.09	-0.02	-2.30	-0.00	-0.35	-0.02	-2.86	-0.01	-0.54
5	0.10	0.12	0.01	2.08	0.03	2.46	0.01	1.36	0.03	2.17
6	0.11	0.12	0.02	2.75	0.03	2.85	0.01	1.51	0.02	2.20
7	0.14	0.15	0.05	6.61	0.05	4.39	0.04	5.16	0.04	3.41
8	0.18	0.18	0.08	9.32	0.08	5.83	0.07	8.05	0.07	4.86
9	0.22	0.21	0.11	9.16	0.11	5.71	0.11	8.58	0.11	5.39
10	0.37	0.37	0.24	10.03	0.25	6.27	0.25	10.43	0.27	6.59
10-1	0.48	0.39	0.45	18.50	0.38	10.14	0.46	18.13	0.39	9.96
GRS Asset Pricing Test			GRS	p	GRS	p	GRS	p	GRS	
			39.72	0.00	11.25	0.00	37.64	0.00	10.75	0.00

$\Rightarrow$ Standard factor models cannot explain cross-sectional returns.

Empirical Results - Performance on Portfolios

Table: Explained Variation and Pricing Errors for Short-Term Reversal Sorted Portfolios[9]

ST_REV	EN	FFN	GAN	EN	FFN	GAN
Decile	Explained Variation			Alpha		
1	0.84	0.74	0.77	-0.18	-0.21	-0.13
2	0.86	0.81	0.82	0.00	-0.05	0.00
3	0.80	0.82	0.84	0.13	0.04	0.06
4	0.69	0.80	0.82	0.16	0.03	0.03
5	0.58	0.68	0.71	0.13	-0.03	-0.04
6	0.43	0.66	0.75	0.22	0.05	0.01
7	0.23	0.64	0.77	0.20	0.03	-0.02
8	-0.07	0.49	0.67	0.23	0.03	-0.05
9	-0.25	0.29	0.58	0.30	0.09	-0.01
10	-0.24	-0.04	0.35	0.47	0.38	0.18
	Explained Variation			Cross-Sectional R^2		
All	0.43	0.58	0.70	0.45	0.79	0.94

[9]Results for Momentum sorted portfolios are included in the appendix [52].

Empirical Results - Performance on Portfolios

Table: Explained Variation and Pricing Errors for Size Sorted Portfolios[10]

LME	EN	FFN	GAN	EN	FFN	GAN
Decile	Explained Variation			Alpha		
1	0.80	0.75	0.79	0.09	-0.00	0.10
2	0.89	0.89	0.90	-0.11	-0.09	-0.06
3	0.91	0.80	0.91	-0.07	0.02	-0.02
4	0.90	0.77	0.91	-0.05	0.04	-0.01
5	0.90	0.78	0.91	0.01	0.10	0.04
6	0.88	0.80	0.91	0.03	0.09	0.02
7	0.84	0.81	0.89	0.04	0.05	-0.01
8	0.84	0.85	0.88	0.06	0.03	-0.02
9	0.77	0.81	0.82	0.06	-0.01	-0.04
10	0.32	0.28	0.49	-0.04	-0.15	-0.10
	Explained Variation			Cross-Sectional R^2		
All	0.83	0.78	0.86	0.96	0.95	0.97

[10] Results for Book-to-Market Ratio sorted portfolios are included in the appendix [53].

Empirical Results - Performance on Portfolios

Table: Explained Variation and Pricing Errors for Decile Sorted Portfolios

	Explained Variation			Cross-Sectional R^2		
Charact.	EN	FFN	GAN	EN	FFN	GAN
ST_REV	0.43	0.58	0.70	0.45	0.79	0.94
SUV	0.42	0.75	0.83	0.64	0.97	0.99
r12_2	0.26	0.27	0.54	0.66	0.71	0.93
NOA	0.58	0.69	0.78	0.94	0.96	0.95
SGA2S	0.52	0.63	0.73	0.93	0.95	0.96
LME	0.83	0.78	0.86	0.96	0.95	0.97
RNA	0.50	0.48	0.69	0.93	0.87	0.96
...	...	...	...	...	...	...
CF2P	0.46	0.47	0.66	0.90	0.89	0.99
BEME	0.70	0.75	0.82	0.97	0.94	0.98
Variance	0.48	0.27	0.61	0.74	0.72	0.89
...	...	...	...	...	...	...

Empirical Results - Characteristic Importance

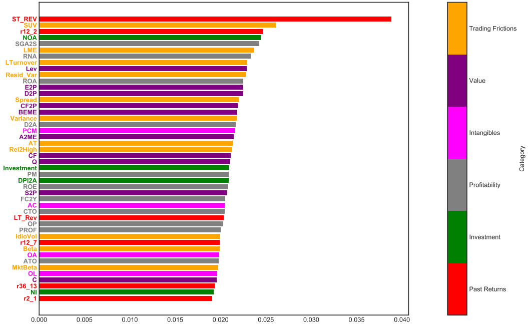
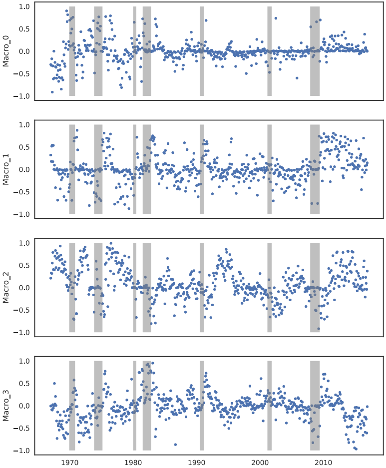


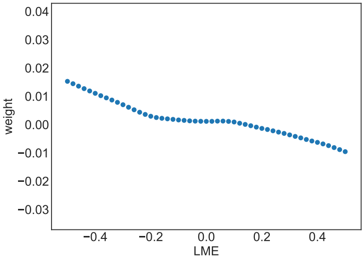
Figure: Characteristic Importance[11] by GAN

[11] Our sensitivity analysis is similar to Sirignano et al. [2016]. See the appendix [54].

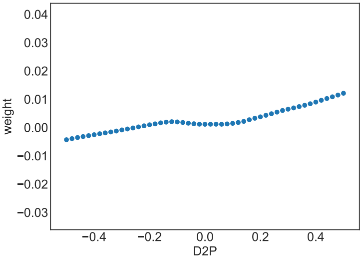
Empirical Results - Macroeconomic Hidden States



Empirical Results - SDF Weights



(a) Size

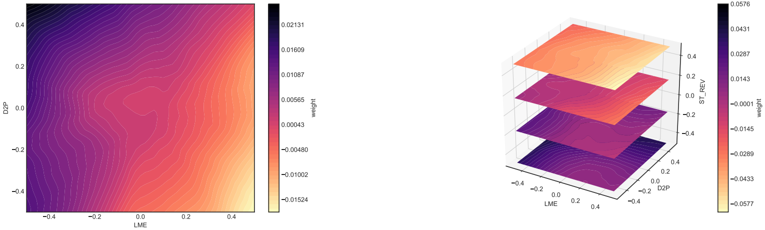


(b) Dividend Yield

Figure: Weight as a Function of Size and Dividend Yield

$\Rightarrow$ Size and dividend yield have close to linear effect!

Empirical Results - SDF Weights



(a) Size and Dividend Yield

(b) Size, Dividend Yield and Short-Term Reversal

Figure: Weight as a Function of Multiple Variables

$\Rightarrow$ Complex interaction between multiple variables!

Conclusion

- Linear models perform well because when considering characteristics in isolation, the models are approximately linear.
- Non-linearities matter for the interaction.
- Most relevant variables are price trends and liquidity.
- Macroeconomic data has a low dimensional factor structure.
- Pricing all individual stocks leads to better pricing models on portfolios.
- SDF structure stable over time.
- Mean-variance efficient portfolio implied by pricing kernel highly profitable in a risk-adjusted sense.

Appendix - Deep Learning in Asset Pricing
SDF Example - CAPM

In CAPM, there is only one factor $R^{M,e} = R^M - R^f$. Find the SDF M_{t+1}, which has the form of $M_{t+1} = a + bR^{M,e}_{t+1}$. With no-arbitrage condition $\mathbb{E}_t[M_{t+1}R^e_{t+1}] = 0$, we have

$$\frac{a}{b} = -\frac{\mathbb{E}_t[(R^{M,e}_{t+1})^2]}{\mathbb{E}_t[R^{M,e}_{t+1}]}$$

Notice that the SDF is negatively correlated with the market factor.

Appendix - Deep Learning in Asset Pricing

SDF Example - Geometric Brownian Motion

Assume the stock price follows geometric Brownian Motion

$$\frac{dS_t}{S_t} = \mu dt + \sigma dW_t$$

Find the pricing kernel π_t, which has the form of

$$\frac{d\pi_t}{\pi_t} = adt + bdW_t$$

- For risk-free rate r,

$$\mathbb{E}_t\left[\frac{\pi_{t+dt}}{\pi_t}(1+rdt)\right] = 1 \quad \Rightarrow \quad a = -r.$$

- For stock return $\mu dt + \sigma dW_t$,

$$\mathbb{E}_t\left[\frac{\pi_{t+dt}}{\pi_t}(1+\mu dt + \sigma dW_t)\right] = 1 \quad \Rightarrow \quad b = (r-\mu)/\sigma$$

Appendix - Deep Learning in Asset Pricing
Generative Adversarial Network by Goodfellow et al. [2014]

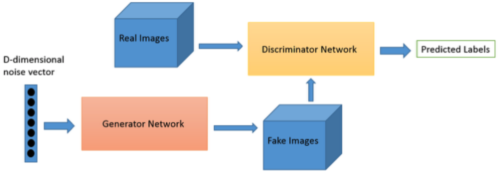


Figure: GAN Structure by Goodfellow et al. [2014]

1. The generator takes random numbers and returns an image.
2. This generated image is fed into the discriminator alongside a stream of images taken from the actual data set.
3. The discriminator takes in both real and fake images and returns probabilities, with 1 representing a prediction of authenticity and 0 representing fake.

Appendix - Deep Learning in Asset Pricing
Feedforward Network

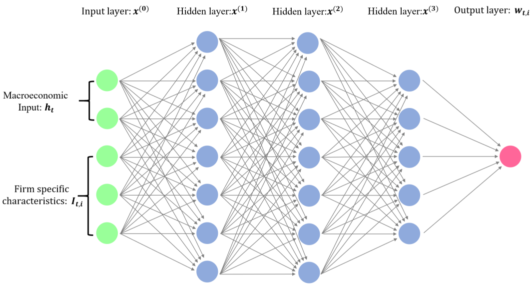


Figure: Feedforward Network with 3 Hidden Layers

$$x^{(l)} = \mathrm{ReLU}\left(W^{(l-1)\top} x^{(l-1)} + w_0^{(l-1)}\right)$$
$$y = W^{(L)\top} x^{(L)} + w_0^{(L)}$$

Appendix - Deep Learning in Asset Pricing
Feedforward Network with Dropout

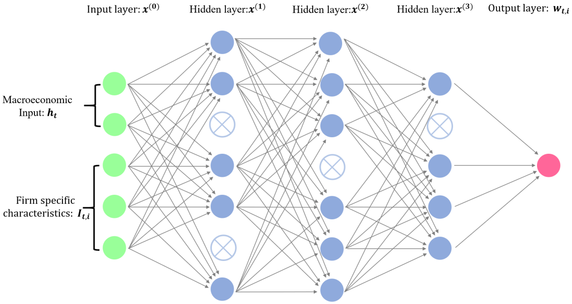


Figure: Feedforward Network with 3 Hidden Layers and Dropout

Appendix - Deep Learning in Asset Pricing
Long-Short-Term-Memory Cell (LSTM)

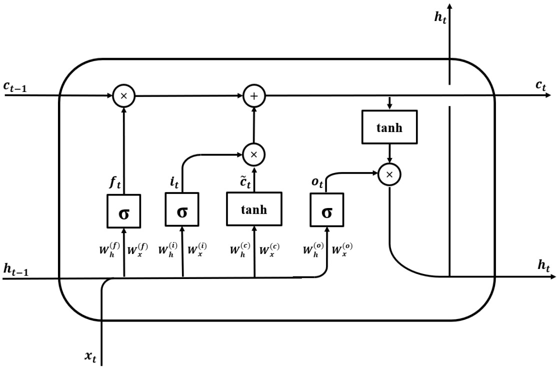


Figure: Long-Short-Term-Memory Cell (LSTM)

Appendix - Deep Learning in Asset Pricing
LSTM Cell Structure

At each step, a new memory cell $\tilde{c}_t$ is created with current input x_t and previous hidden state h_{t-1}

$$\tilde{c}_t = \tanh(W_h^{(c)} h_{t-1} + W_x^{(c)} x_t + w_0^{(c)}).$$

The input and forget gate control the memory cell, while the output gate controls the amount of information stored in the hidden state:

$$\begin{aligned} \text{input}_t =& \sigma(W_h^{(i)} h_{t-1} + W_x^{(i)} x_t + w_0^{(i)}) \\ \text{forget}_t =& \sigma(W_h^{(f)} h_{t-1} + W_x^{(f)} x_t + w_0^{(f)}) \\ \text{out}_t =& \sigma(W_h^{(o)} h_{t-1} + W_x^{(o)} x_t + w_0^{(o)}). \end{aligned}$$

The final memory cell and hidden state are given by

$$\begin{aligned} c_t =& \text{forget}_t \circ c_{t-1} + \text{input}_t \circ \tilde{c}_t \\ h_t =& \text{out}_t \circ \tanh(c_t). \end{aligned}$$

Appendix - Deep Learning in Asset Pricing
Hyper-Parameter Search

Table: Selection of Hyperparameters for GAN

Notation	Hyperparameters	Candidates	Optimal
HL	Number of layers in SDF Network	2, 3 or 4	2
HU	Number of hidden units in SDF Network	64	64
SMV	Number of hidden states in SDF Network	4 or 8	4
CSMV	Number of hidden states in Conditional Network	16 or 32	32
CHL	Number of layers in Conditional Network	0 or 1	0
CHU	Number of hidden units in Conditional Network	4, 8, 16 or 32	8
LR	Initial learning rate	0.001, 0.0005, 0.0002 or 0.0001	0.001
DR	Dropout	0.95	0.95

1. For each combination of hyperparameters (384 models) we fit the GAN model.
2. We select the five best combinations of hyperparameters on the validation data set.
3. For each of the five combinations we fit 9 models with the same hyperparameters but different initialization.
4. We select the ensemble model with the best performance on the validation data set.

Appendix - Deep Learning in Asset Pricing

Other Implementation Details

- For training deep neural networks, the vanilla stochastic gradient descend method has proven to be not an efficient method. A better approach is to use optimization methods that introduce an adaptive learning rate (e.g. Adam).
- Regularization is crucial and prevents the model from over-fitting on the training sample. Although l_1/l_2 regularization might also be used in training other neural networks, Dropout is preferable and generally results in better performances.
- We use ensemble averaging to create a group of models that provide a significantly more robust estimation. Let's denote $\hat{w}^{(j)}$ to be the optimal portfolio weights given by the j^{th} model. The ensemble model is a weighted average of the outputs from models with the same architecture but different starting values for the optimization and gives more robust estimates: $\hat{\omega} = \frac{1}{9}\sum_{j=1}^{9}\hat{\omega}^{(j)}$.

Appendix - Deep Learning in Asset Pricing
Simulation Setup

- Excess returns follow a no-arbitrage model with SDF factor F

$$R^e_{i,t+1} = \beta_{i,t} F_{t+1} + \epsilon_{i,t+1}.$$

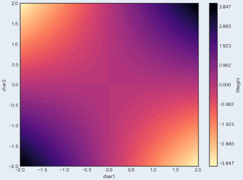
- The SDF factor follows $F_t \overset{i.i.d.}{\sim} \mathcal{N}(\mu_F, \sigma_F^2)$ with $\sigma_F^2 = 0.1$ and $SR_F = 1$.
- The risk loadings β

$$\beta_{i,t} = C^{(1)}_{i,t} \cdot C^{(2)}_{i,t} \qquad \text{with} \qquad C^{(1)}_{i,t}, C^{(2)}_{i,t} \overset{i.i.d.}{\sim} \mathcal{N}(0,1).$$

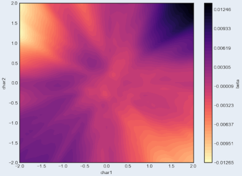
- The idiosyncratic component $\epsilon_{i,t} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_e^2)$ with $\sigma_e^2 = 1$.
- $N = 500$ and $T = 600$. Training/validation/test split is 250,100,250.

Appendix - Deep Learning in Asset Pricing
Simulation Results

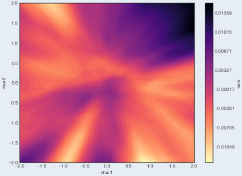
Loadings β with 2 characteristics



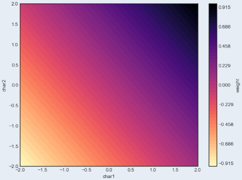
(a) Population Model



(b) GAN



(c) FFN



(d) LS

Appendix - Deep Learning in Asset Pricing

Simulation Results

Table: Performance of Different SDF Models

	Sharpe Ratio			EV			Cross-sectional R^2		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Population	0.96	1.09	0.94	0.16	0.15	0.17	0.17	0.15	0.17
GAN	0.98	1.11	0.94	0.12	0.11	0.13	0.10	0.09	0.07
FFN	0.94	1.04	0.89	0.05	0.04	0.05	-0.30	-0.09	-0.33
LS	0.07	-0.10	0.01	0.00	0.00	0.00	0.00	0.01	0.01

Appendix - Deep Learning in Asset Pricing
Turnover

Table: Turnover by Models

	Long Position			Short Position		
Model	Train	Valid	Test	Train	Valid	Test
LS	0.25	0.22	0.24	0.64	0.55	0.61
EN	0.36	0.35	0.35	0.83	0.83	0.84
FFN	0.69	0.63	0.65	1.38	1.29	1.27
GAN	0.47	0.40	0.40	1.05	1.04	1.02

Turnover for positions with positive and negative weighs for the SDF factor portfolio.

Appendix - Deep Learning in Asset Pricing
Performance on Portfolios

Table: Explained Variation and Pricing Errors for Momentum Sorted Portfolios

r12_2	EN	FFN	GAN	EN	FFN	GAN
Decile	Explained Variation			Alpha		
1	0.04	-0.06	0.33	0.37	0.39	0.11
2	0.12	0.10	0.52	0.25	0.18	-0.01
3	0.19	0.25	0.66	0.14	0.05	-0.06
4	0.28	0.34	0.73	0.15	0.08	-0.02
5	0.37	0.46	0.80	0.19	0.09	0.02
6	0.45	0.58	0.78	0.02	-0.03	-0.09
7	0.62	0.69	0.68	0.01	0.01	-0.05
8	0.58	0.71	0.64	-0.03	-0.04	-0.09
9	0.55	0.70	0.58	0.08	0.04	-0.03
10	0.51	0.53	0.53	0.24	0.29	0.19
	Explained Variation			Cross-Sectional R^2		
All	0.26	0.27	0.54	0.66	0.71	0.93

Appendix - Deep Learning in Asset Pricing
Performance on Portfolios

Table: Explained Variation and Pricing Errors for Book-to-Market Ratio Sorted Portfolios

BEME	EN	FFN	GAN	EN	FFN	GAN
Decile	Explained Variation			Alpha		
1	0.38	0.66	0.70	0.03	-0.12	-0.08
2	0.48	0.73	0.78	0.10	-0.05	-0.04
3	0.71	0.84	0.86	0.07	-0.03	-0.01
4	0.76	0.88	0.89	0.00	-0.07	-0.07
5	0.82	0.87	0.88	0.05	0.02	0.01
6	0.77	0.82	0.88	0.06	0.04	0.02
7	0.81	0.81	0.87	0.03	0.08	0.03
8	0.71	0.59	0.78	0.03	0.12	0.06
9	0.80	0.72	0.80	-0.02	0.11	0.07
10	0.68	0.73	0.79	-0.05	-0.00	0.00
	Explained Variation			Cross-Sectional R^2		
All	0.70	0.75	0.82	0.97	0.94	0.98

Appendix - Deep Learning in Asset Pricing
Economic Significance of Variables

- we define the sensitivity of a particular variable as the average absolute derivative of the weight w with respect to this variable:

$$\text{Sensitivity}(x_j) = \frac{1}{C} \sum_{i=1}^{N} \sum_{t=1}^{T} \left| \frac{\partial w(I_t, I_{t,i})}{\partial x_j} \right|,$$

  where C a normalization constant.
- A sensitivity of value z for a given variable means that the weight w will approximately change (in magnitude) by $z\Delta$ for a small change of Δ in this variable.

References I

Kerry Back. *Asset pricing and portfolio choice theory*. Oxford University Press, 2010.

Luyang Chen, Markus Pelger, and Jason Zhu. Deep learning in asset pricing. *Available at SSRN 3350138*, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Shihao Gu, Bryan T Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. Working Paper 25398, National Bureau of Economic Research, 2018.

Justin Sirignano, Apaar Sadhwani, and Kay Giesecke. Deep learning for mortgage risk. *Working paper*, 2016.