Detecting and adapting to crisis pattern with context based DRL

2020, Milan, 10-15 January 2021

E. Benhamou, D. Saltiel, J. Ohana, J. Atif

HOMA CAPITAL

Dauphine | PSL UNIVERSITÉ PARIS

LAMSADE 1974-2014

MILES

For Alpha

LISIC Laboratoire d'Informatique Signal & Image de la Côte d'Opale

ulco Université Littoral Côte d'Opale

Executive summary

- Show that Deep Reinforcement Learning (DRL) can shed new lights on portfolio allocation

- Advantages:
  - DRL maps directly market conditions to actions
  - No bias or influence from risk assumption
  - Can incorporate more inputs
  - Can detect crisis pattern

Context

- In asset management, there is a gap between mainstream used methods and new machine learning techniques around RL

- DRL has achieved strong results in challenging tasks like autonomous driving, games solving like Atari (Mnih et al. 2013), Go (Silver et al. 2018)

Machine learning in finance

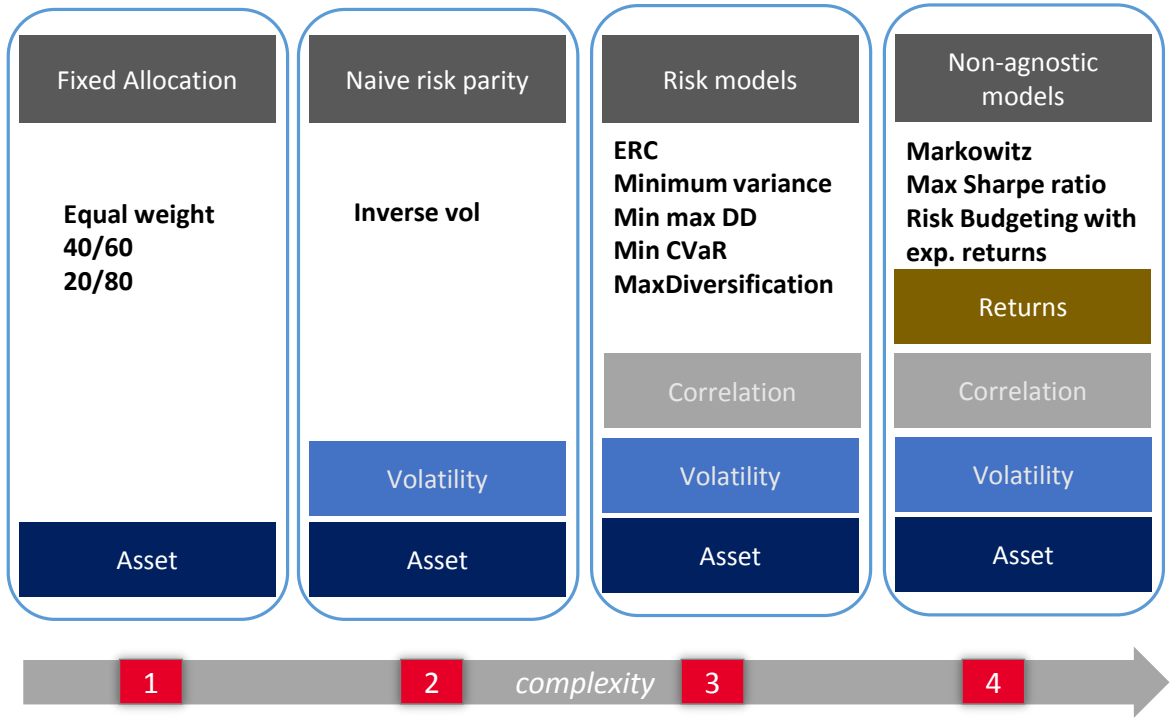
- Surprisingly, ML is still not widely used in Asset Management. This may come from the fact that asset managers have been mostly trained with econometric and financial mathematics background

Related works

- Portfolio: Markowitz 1952, Minimum variance, maximum diversification, maximum decorrelation, risk parity.

- RL has started been used in portfolio allocation with works like Jiang and Liang 2016; Zhengyao et al. 2017; Liang et al. 2018; Yu et al. 2019; Wang and Zhou 2019; Saltiel et al. 2020; Benhamou et al. 2020b; 2020a; 2020c

Traditional methods



1. Portfolio with fixed weights, left to the discretion of the investor
2. Inverse volatility : weigh the assets proportionally to the inverse of their volatility
3. Models based on risk metrics : there is no a priori knowledge of the expected returns
4. By being long-short, risk premia may explain part of the alpha.

Traditional methods explained

denote by $w = (w_1, ..., w_l)$ the allocation weights

$\mu = (\mu_1, ..., \mu_l)^T$ be the expected returns

Σ the matrix of variance covariances

r_{min} be the minimum expected return

$$\underset{w}{\text{Minimize}} \quad w^T \Sigma w \tag{1}$$

$$\text{subject to} \quad \mu^T w \geq r_{min}, \sum_{i=1...l} w_i = 1, 1 \geq w \geq 0$$

Example

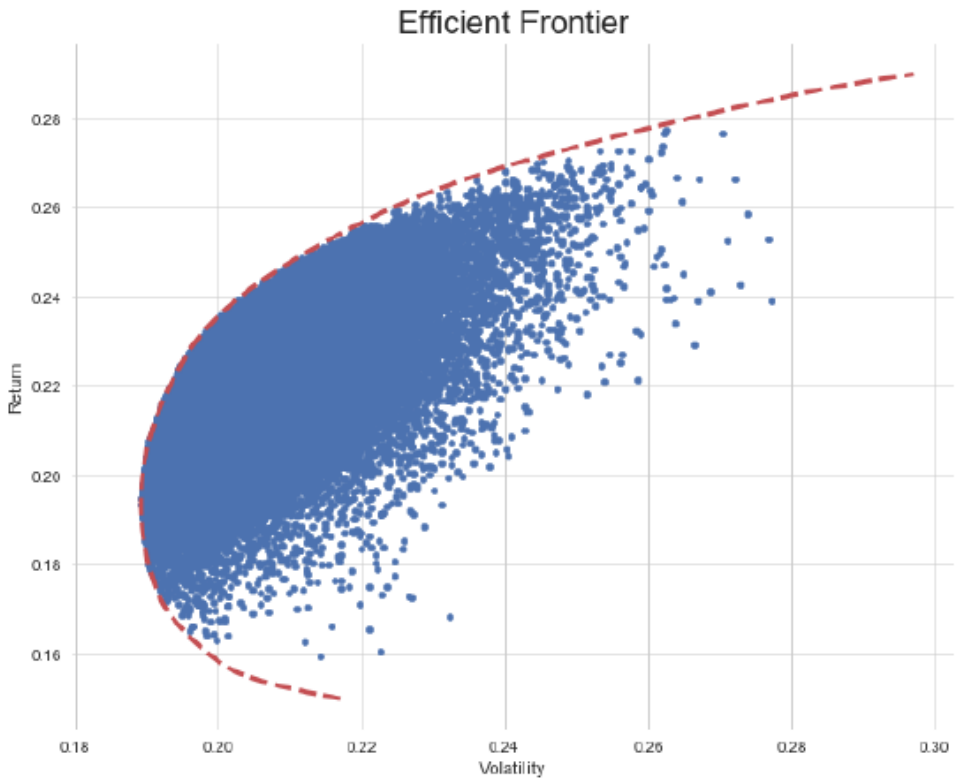


Figure 1: Markowitz efficient frontier for the GAFA: returns taken from 2017 to end of 2019

Minimum variance portfolio

$$\underset{w}{\text{Minimize}} \quad w^T \Sigma w$$

$$\text{subject to} \quad \sum_{i=1 \ldots l} w_i = 1, 1 \geq w \geq 0$$

Maximum diversification portfolio

$$\underset{w}{\text{Maximize}} \quad \frac{w^T \sigma}{\sqrt{w^T \sum w}}$$

$$\text{subject to} \quad \sum_{i=1\ldots l} w_i = 1, 1 \geq w \geq 0$$

$\sigma = (\Sigma_{i,i})_{i=1..l}$ the diagonal elements of the covariance matrix Σ

Maximum decorrelation portfolio

$$\begin{aligned} \underset{w}{\text{Minimize}} \quad & w^T C w \\ \text{subject to} \quad & \sum_{i=1\ldots l} w_i = 1, 1 \geq w \geq 0 \end{aligned}$$

Risk parity portfolio

$$\underset{w}{\text{Minimize}} \quad \frac{1}{2} w^T \Sigma w - \frac{1}{n} \sum_{i=1}^{l} \ln(w_i)$$

$$\text{subject to} \quad \sum_{i=1 \ldots l} w_i = 1, 1 \geq w \geq 0$$

Reinforcement learning

$$\begin{aligned} \underset{\pi(.)}{\text{Maximize}} \quad & \mathbb{E}[R_T] \\ \text{subject to} \quad & a_t = \pi(s_t) \end{aligned}$$

Mathematical formulation

- MDP setting

A Markov decision process is defined as a tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, p, r)$ where:

- $\mathcal{X}$ is the state space,
- $\mathcal{A}$ is the action space,
- $p(y|x,a)$ is the transition probability such that $p(y|x,a) = \mathbb{P}(x_{t+1} = y | x_t = x, a_t = a)$,
- $r(x,a,y)$ is the reward of transition (x, a, y).

Regular observations

- Regular observations:
  - Past returns $r_t = \frac{p_t^k}{p_{t-1}^k} - 1$ where p_t^k is the price at time t of the asset k
  - Empirical standard deviations $\sigma_t^k = \sqrt{\frac{1}{d}\sum_{u=t-d+1}^{t}(r_u - \mu)^2}$ useful to detect regime changes

→ three dimensional tensor $A_t = [A_t^1, A_t^2]$

$$\text{with} \quad A_t^1 = \begin{pmatrix} r_{t-i_j}^1 & \dots & r_t^1 \\ \dots & \dots & \dots \\ r_{t-i_j}^m & \dots & r_t^m \end{pmatrix}, \; A_t^2 = \begin{pmatrix} \sigma_{t-i_j}^1 & \dots & \sigma_t^1 \\ \dots & \dots & \dots \\ \sigma_{t-i_j}^m & \dots & \sigma_t^m \end{pmatrix}$$

Contextual observation

- Risk aversion index
- Correlation between equities and bonds
- Citi economic surprise index

Action

- Portfolio weights: $(p_t^1, \ldots, p_t^l)$ modelled by a softmax layer

Reward

- Final net profit: $\frac{P_{t_T}}{P_{t_0}} - 1$

- Sharpe ratio: μ / σ

- Sortino ratio: $\mu / \tilde{\sigma}$ where

$\tilde{\sigma}$ is the downside standard deviation

A complex network

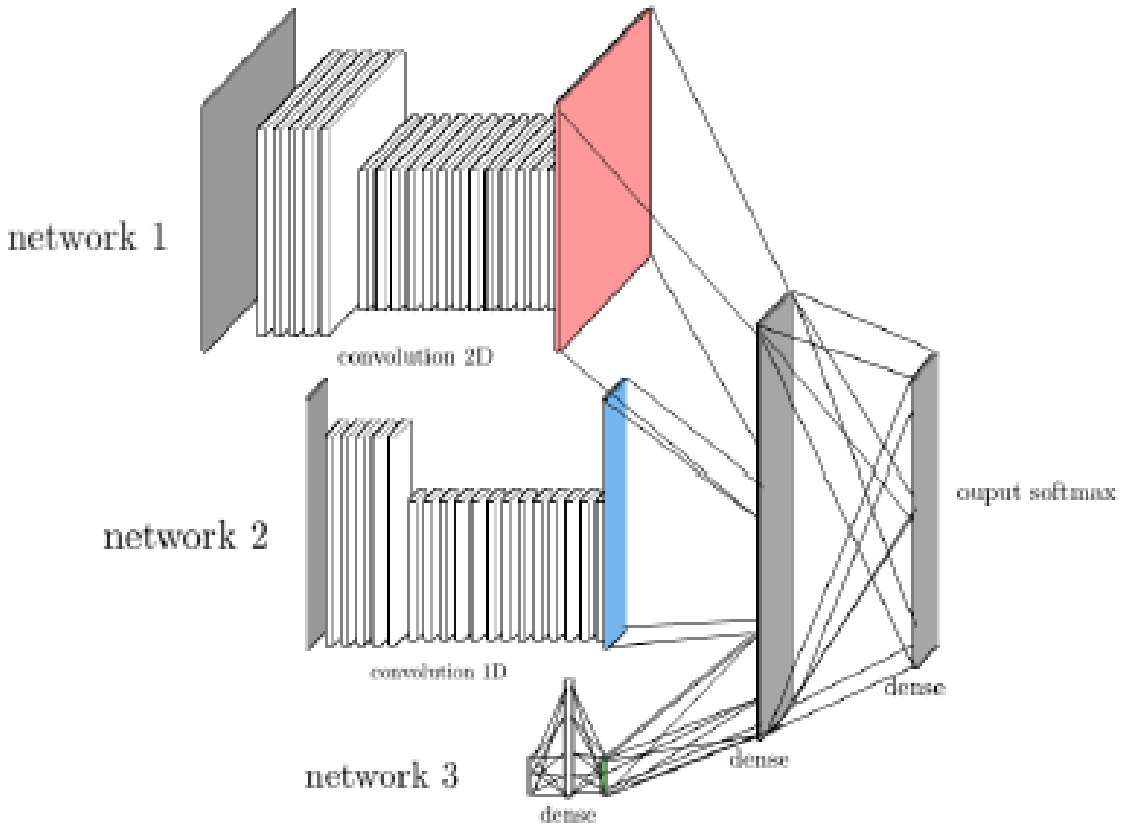


Fig. 3. Possible DRL network architecture

Particularities of our network

- Multi inputs
- Multi outputs

- Compared to traditional portfolio method can incorporate leverage separately from normal portfolio weights

Training of the network

- Adversarial policy gradient: noise on data as we have a single experiment and wants to have different scenarios

- Replay buffer as the reward is only at the final period

Algorithmic in details

Algorithm 1 Adversarial Policy Gradient

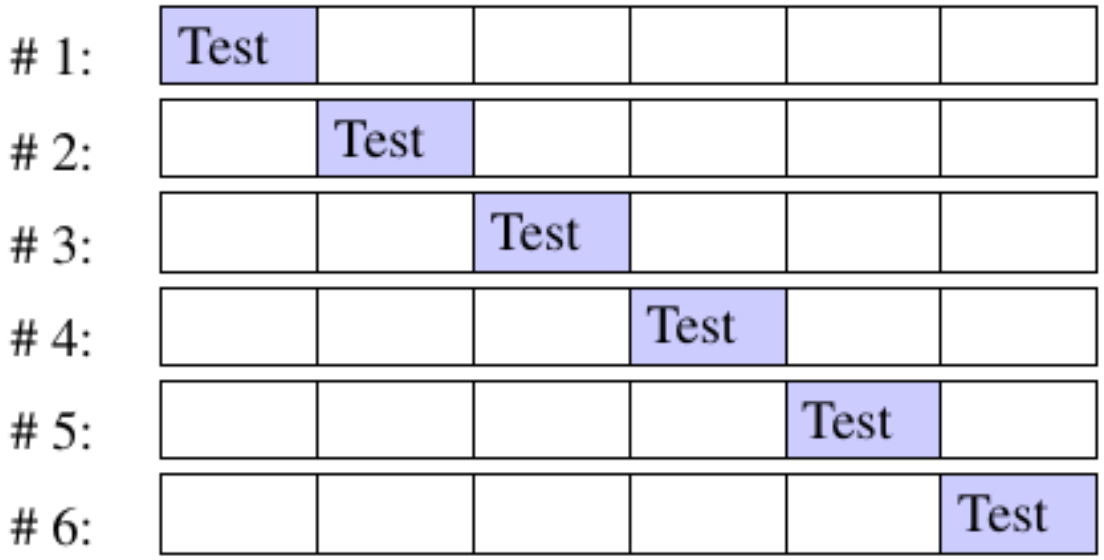
Input: initial policy parameters θ, empty replay buffer $\mathcal{D}$
repeat
  reset replay buffer
  while not terminal **do**
    Observe observation o and select action $a = \pi_\theta(o)$ with probability p and random action with probability $1 - p$,
    Execute a in the environment
    Observe next observation o', reward r, and done signal d to indicate whether o' is terminal
    apply noise to next observation o'
    store (o, a, o') in replay buffer $\mathcal{D}$
    if Terminal **then**
      for however many updates in $\mathcal{D}$ **do**
        compute final reward R
      end for
      update network parameter with Adam gradient ascent $\vec{\theta} \longrightarrow \vec{\theta} + \lambda \nabla_{\vec{\theta}} J_{[0,t]}(\pi_{\vec{\theta}})$
    end if
  end while
until convergence

Other parameters

- Learning rate of 0.01
- Adversarial Gaussian noise with a standard deviation of 20bps
- 500 maximum iterations with early stop if no improvement over the last 50 iterations

Walk forward analysis

- Standard k-fold cross validation does not work in finance as it uses futures information in train set



Solution walk forward

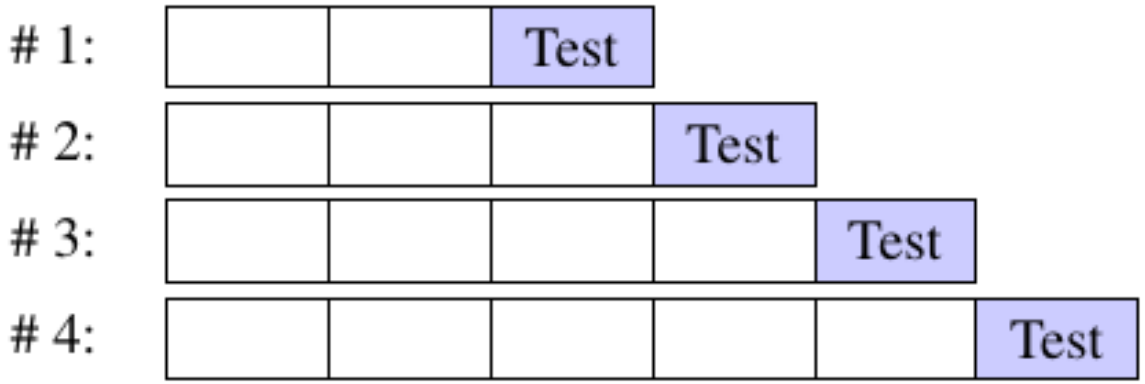


Figure : anchored walk forward

Experiments

- Data from 01/05/2000 to 19/06/2020
- Risky asset = MSCI world index
- Hedging strategies 4 SGCIB proprietary hedging strategies

Hedging strategies

- Directional hedges - react to small negative return in equities,
- Gap risk hedges - perform well in sudden market crashes,
- Proxy hedges - tend to perform in some market configurations, like for example when highly indebted stocks under-perform other stocks,
- Duration hedges - invest in bond market, a classical diversifier to equity risk in finance.

Evaluation metrics

- Annualized return
- Annualized daily based Sharpe ratio
- Sortino ratio (ratio of annualized return over the downside standard deviation)
- Maximum daily drawdown (max DD)

Baseline

- Pure risky asset
- Markowitz
- Follow the winner
- Follow the loser

Results in numbers

TABLE I
PERFORMANCE RESULTS

	Portfolio 1	Portfolio 2	Portfolio 3	Dynamic Markovitz	Deep RL Net_profit	Deep RL Sharpe	Naive winner
Net Performance	-6.3%	-2.1%	3.9%	0.7%	**8.8%**	8.6%	3.9%
Std dev	6.1%	6.5%	7.3%	4.3%	4.5%	4.2%	7.3%
Sharpe ratio	na	na	0.53	0.17	1.95	**2.08**	0.53

Usage of contextual information

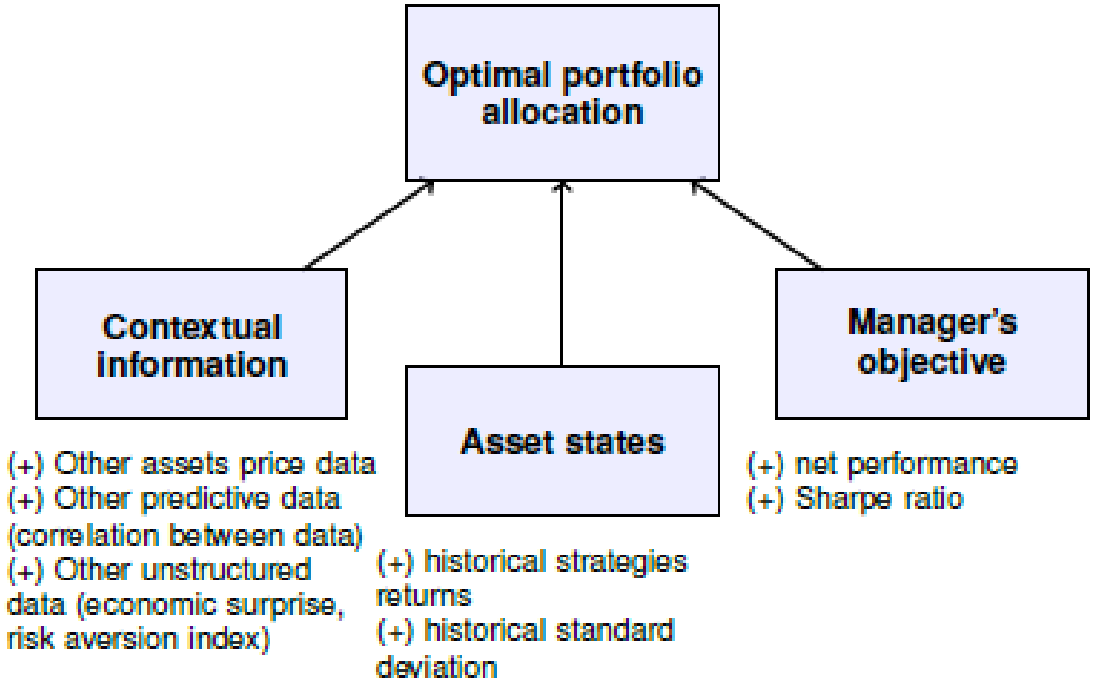


Fig. 1. Portfolio allocation problem

Impact of context

TABLE II
RESULTS OF THE VARIOUS MODELS

Reward	Adversarial training?	Network	Previous weight?	Context?	Annual return	Sharpe
NetProfit	No	Conv2D	No	Yes	8.8%	1.95
Sharpe	Yes	Conv2D	No	Yes	8.6%	2.08
NetProfit	No	Conv2D	Yes	Yes	8.5%	2.03
Sharpe	No	Conv2D	No	Yes	8.4%	2.01
NetProfit	Yes	Conv2D	No	Yes	8.0%	1.35
NetProfit	Yes	Conv2D	No	No	7.7%	1.94
Sharpe	No	Conv2D	No	No	6.4%	1.31
NetProfit	No	LSTM	No	Yes	6.2%	1.49
NetProfit	No	Conv2D	No	No	5.4%	0.97
Sharpe	Yes	LSTM	No	Yes	5.4%	1.23
NetProfit	Yes	LSTM	No	Yes	5.1%	0.93
NetProfit	Yes	Conv2D	Yes	Yes	4.3%	0.63
Sharpe	Yes	Conv2D	No	No	4.2%	0.69
NetProfit	No	LSTM	Yes	Yes	3.8%	0.52
Sharpe	No	Conv2D	Yes	No	3.8%	0.52
NetProfit	No	Conv2D	Yes	Yes	3.8%	0.52
Sharpe	Yes	LSTM	Yes	Yes	3.8%	0.52
Sharpe	Yes	Conv2D	Yes	Yes	3.7%	0.51
NetProfit	Yes	Conv2D	Yes	No	3.7%	0.51
NetProfit	No	LSTM	Yes	No	3.6%	0.49
NetProfit	Yes	LSTM	Yes	Yes	3.5%	0.48
NetProfit	Yes	LSTM	No	No	3.4%	1.24
Sharpe	No	LSTM	Yes	Yes	3.4%	0.48
NetProfit	No	LSTM	No	No	3.4%	0.47
Sharpe	Yes	Conv2D	Yes	No	3.4%	0.51
NetProfit	Yes	LSTM	Yes	No	2.3%	0.97
Sharpe	Yes	LSTM	No	No	2.3%	0.32
Sharpe	No	LSTM	No	Yes	1.5%	0.22
Sharpe	No	Conv2D	Yes	No	0.9%	0.13
Sharpe	Yes	LSTM	Yes	No	-5.1%	na
Sharpe	No	LSTM	No	No	-5.1%	na
Sharpe	No	LSTM	Yes	No	-5.1%	na

Hyper parameters used

TABLE IV
HYPER PARAMETERS USED

hyper-parameters	value	description
batch size	50	Size of mini-batch during training
regularization coefficient	1e-8	L_2 regularization coefficient applied to network training
learning rate	0.01	Step size parameter in Adam
standard deviation period	20 days	period for standard deviation in asset states
commission	10 bps	commission rate
stride	2,1	stride used in convolution networks
conv number 1	5,10	number of convolutions in sub-network 1
conv number 2	2	number of convolutions in sub-network 2
lag period 1	[60, 20, 4, 3, 2, 1, 0]	lag period for asset states
lag period 2	[60, 20, 4, 3, 2, 1, 0]	lag period for contextual states
noise	0.002	adversarial Gaussian standard deviation

Future work

- Test more contextual data
- Impact of more layers and other neural network design choice