DEEP REINFORCEMENT LEARNING APPLIED TO GENERATE TRADING SIGNALS FOR FINANCIAL MARKETS

Borja Gómez Solórzano[1]

[1]Quant at Darwinex

MSTC final work presentation January 2019

Outline

1. Motivation
   - Deep Learning
   - Reinforcement Learning
   - Direct Reinforcement
2. The model
   - The Simple Trader Agent
   - The optimization problem
3. Results
   - The model implementation
   - SPX500 Examples

Outline

Deep Learning

- It is an emerging technique based on neuron conectivities.
- Used for feature learning.
- Difficult to understand the output.

Deep Learning

- It is an emerging technique based on neuron conectivities.
- Used for feature learning.
- Difficult to understand the output.

Deep Learning

- It is an emerging technique based on neuron conectivities.
- Used for feature learning.
- Difficult to understand the output.

Outline

Reinforcement Learning

- Learning comes from the interactions with the environment.
- Solves dynamic control problems like trading decision.
- The agent "understands" the Deep Feature Learning.

Reinforcement Learning

- Learning comes from the interactions with the environment.
- Solves dynamic control problems like trading decision.
- The agent "understands" the Deep Feature Learning.

Reinforcement Learning

- Learning comes from the interactions with the environment.
- Solves dynamic control problems like trading decision.
- The agent “understands” the Deep Feature Learning.

Outline

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Direct Reinforcement

The policy is learned directly from observation to action.

Advantages:

- Avoids the Bellman's *curse of dimensionality*.
- Allows to implement recurrences (RRL)
- Can learn stochastic policies.

Disadvantages:

- Usually converge to a local instead of a global optimum.
- High variance on the results.

Outline

The algorithm

Recurrent Reinforcement Learning

Input: $\mathcal{O}_t$
Output: π_θ policy.
Initialize θ randomly
for each episode **do**
    for each time t **do**
        apply deep learning $g_t(o_t, \theta)$
        take action a_{t+1} from $f(g_t, a_t)$
    end for
    $u(a;\theta) = U_T(R_1, R_2, ...R_T)$
    π_θ: improved from a local approximation of the u gradient
end for
return u, π_θ

The recurrent model

Trader action

$$F_t = F(O_{t-1}^{(n)}, F_{t-1}^{(m)}; \theta_t)$$

Rewards

$$R_t = F_{t-1} r_t - c|F_t - F_{t-1}|$$

Utility, Sharpe ratio

$$S_t = \frac{\mu(R_t)}{\sigma(R_t)}$$

The recurrent model

Trader action

$$F_t = F(O_{t-1}^{(n)}, F_{t-1}^{(m)}; \theta_t)$$

Rewards

$$R_t = F_{t-1} r_t - c|F_t - F_{t-1}|$$

Utility, Sharpe ratio

$$S_t = \frac{\mu(R_t)}{\sigma(R_t)}$$

The recurrent model

Trader action

$$F_t = F(O_{t-1}^{(n)}, F_{t-1}^{(m)}; \theta_t)$$

Rewards

$$R_t = F_{t-1} r_t - c|F_t - F_{t-1}|$$

Utility, Sharpe ratio

$$S_t = \frac{\mu(R_t)}{\sigma(R_t)}$$

Outline

1. Motivation
   - Deep Learning
   - Reinforcement Learning
   - Direct Reinforcement
2. The model
   - The Simple Trader Agent
   - The optimization problem
3. Results
   - The model implementation
   - SPX500 Examples

The optimization

The maximization problem

$$\begin{aligned}
\underset{\theta=\{W,W_a\}}{\text{maximize}} \quad & U_T(R_1, ..., R_T) \\
\text{subject to} \quad & R_t = F_{t-1} - c|F_t - F_{t-1}| \\
& F_t = f(< W, g_t > +b + W_a F_{t-1}) \\
& g_t = \varphi(o_t)
\end{aligned}$$

Vanishing gradient

Problem

- The model has a deep timed-based structure.
- The magnitude of the transition weight has a large impact on the learning process.

Solution

- Evolutionary algorithms, for instance particle swarm optimization.
- LSTM networks.

Vanishing gradient

Problem

- The model has a deep timed-based structure.
- The magnitude of the transition weight has a large impact on the learning process.

Solution

- Evolutionary algorithms, for instance particle swarm optimization.
- LSTM networks.

Vanishing gradient

Problem

- The model has a deep timed-based structure.
- The magnitude of the transition weight has a large impact on the learning process.

Solution

- Evolutionary algorithms, for instance particle swarm optimization.
- LSTM networks.

Vanishing gradient

Problem

- The model has a deep timed-based structure.
- The magnitude of the transition weight has a large impact on the learning process.

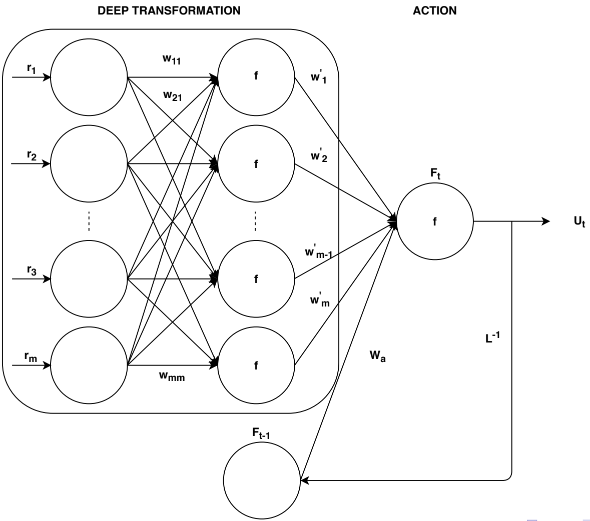
Solution

- Evolutionary algorithms, for instance particle swarm optimization.
- LSTM networks.

Outline

1. Motivation
   - Deep Learning
   - Reinforcement Learning
   - Direct Reinforcement
2. The model
   - The Simple Trader Agent
   - The optimization problem
3. Results
   - The model implementation
   - SPX500 Examples

Two layer model



Side data model

- Adding new features reduce bias (but increase variance).
- Helps the trader to take "better" decisions.
- Highest / Lowest price, moving averages, ...

Side data model

- Adding new features reduce bias (but increase variance).
- Helps the trader to take “better” decisions.
- Highest / Lowest price, moving averages, ...

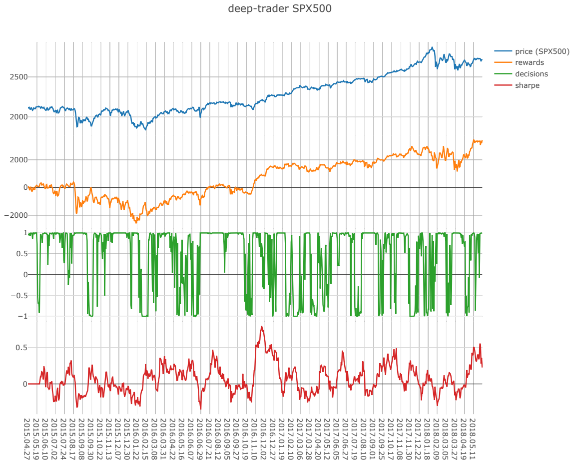
Side data model

- Adding new features reduce bias (but increase variance).
- Helps the trader to take “better” decisions.
- Highest / Lowest price, moving averages, ...

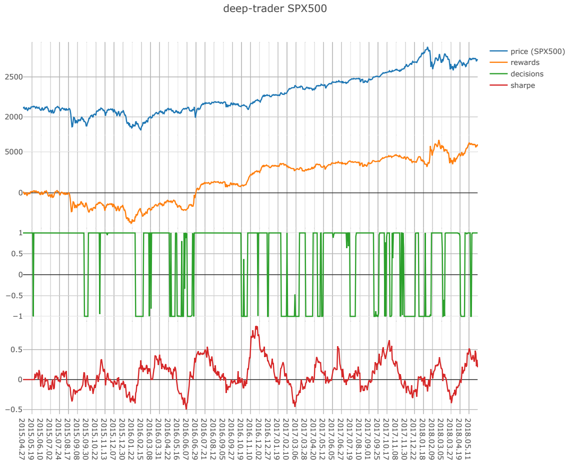
Outline

1. Motivation
   - Deep Learning
   - Reinforcement Learning
   - Direct Reinforcement
2. The model
   - The Simple Trader Agent
   - The optimization problem
3. Results
   - The model implementation
   - SPX500 Examples

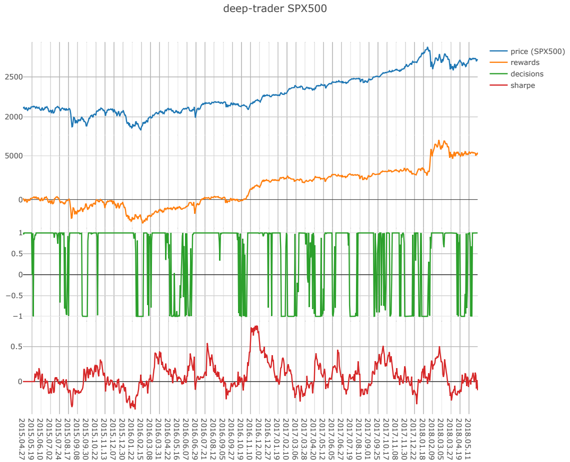
One layer model



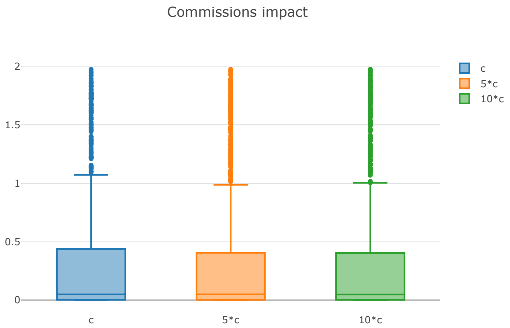
Two layer model



Side data model



Commission analysis



Conclusions

- The simple agent trader is a trend follower algorithm.
- The side data augmented model improves the results of the simpler case with one hidden layer.
- The two hidden layer model improves the results of the model with one hidden layer.
- Future work lines
  - To implement a LSTM model, trade multiple assets, new utility functions,...

Conclusions

- The simple agent trader is a trend follower algorithm.
- The side data augmented model improves the results of the simpler case with one hidden layer.
- The two hidden layer model improves the results of the model with one hidden layer.
- Future work lines
  - To implement a LSTM model, trade multiple assets, new utility functions,...

Conclusions

- The simple agent trader is a trend follower algorithm.
- The side data augmented model improves the results of the simpler case with one hidden layer.
- The two hidden layer model improves the results of the model with one hidden layer.
- Future work lines
  - To implement a LSTM model, trade multiple assets, new utility functions,...

Conclusions

- The simple agent trader is a trend follower algorithm.
- The side data augmented model improves the results of the simpler case with one hidden layer.
- The two hidden layer model improves the results of the model with one hidden layer.
- Future work lines
  - To implement a LSTM model, trade multiple assets, new utility functions,...

Conclusions

- The simple agent trader is a trend follower algorithm.
- The side data augmented model improves the results of the simpler case with one hidden layer.
- The two hidden layer model improves the results of the model with one hidden layer.
- Future work lines
    - To implement a LSTM model, trade multiple assets, new utility functions,...

Bibliography I

- John Moody and Matthew Saffell,
  Reinforcement Learning for Trading,
  Advances in neural information processing systems 11. NIPS Conference, Denver, Colorado, USA, November 30 - December 5, 1998, pages 917-923

- John Moody, Lizhong Wu, Yuansong Liao & Matthew Saffell,
  Performance functions and Reinforcement Learning for trading systems and portfolios,
  Journal of Forecasting, Volume 17, pages 441-470, 1998.

Bibliography II

Yue Deng, Feng Bao, Youyong Kong, Zhiquan Ren, and Qionghai Dai,
Deep Direct Reinforcement Learning for Financial Signal Representation and Trading
IEEE Transactions on Neural Networks and Learning Systems, 2017, volume 28, pages 653-664.

David W. Lu
Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM Neural Networks.
2017.

Bibliography III

John Moody, Yufeng Liu, Matthew Saffell and Kyoungju Youn
Stochastic Direct Reinforcement: Application to Simple Games with Recurrence.
Appears in *Artificial Multiagent Learning, Sean Luke et al. editors, AAAI Press, Menlo Park, 2004.*

Diederik P. Kingma, Jimmy Lei Ba.
Adam: A method for stochastic optimization.
Published as a conference paper at ICLR 2015.

Thank You!
QA