**TraderBot -Specification**  
Himel Patel  
Last Updated: 8 October 2017

# Overview

The daily trader bot is a piece of software which attempts to trade the currency markets. The bot will focus on only one market at first. This market will be the most liquid currency pair to date, EURUSD. The trader bot will focus on short timeframe trades, exactly how short will be defined in this spec. The reasoning for focusing only on short timeframe trades is because in this domain the markets are driven less by macro fundamentals and more by speculation and technical analysis. I believe this form of trading will be performed better by the bot than long term fundamental trades.

It is common for bots to be programmed with a rules based approach. This will not be the case here, instead machine learning will be used. To be more precise, the type of machine learning used will be reinforcement learning. The thinking behind this is that trading can be treated similar to a game which is where reinforcement learning has proven to be most successful.

The bot will trade historically data making decisions on when to enter into short or long trade and when to exit these trades. There will be strict constraints on a trade’s duration to force the learning process to focus on short timeframe trading. The rewards in the reinforcement learning process will be related to the profitability on a trade, being either positive or negative, and the accumulated capital balance.

The motivation behind this project is to recoup losses from a car incident I had resulting in me needing to pay NZD$1600 to cover the insurance expenses. Therefore, the goal of the bot is not to focus on long term sustainable profit making. But instead making a series of short term trades over several months to recoup these losses and make a little bit extra to compensate me for the time spent working on this project. Therefore, it will be considered a great success if this bot can make NZD$3200 over several months by trading the markets in real-time.

This software contains more than just the trading bot but also an environment to train the bot and code for deployment, so live trading can be done.

# Non Goals

The current version of software will not support the following features:

* The ability to have multiple trades open (limited by the broker simulator)
* The bot does not have the option to have multiple take profit targets i.e. essential staging an entry or exit.

# Software Component Diagram

Neural Network

Agent

Data Provider

Trade Manager

Reward Manager

NN Trainer

Broker Simulator

Replay Memory Holder

Program Manager

Trade History Recorder

Trade Viewer

# Neural Network Architecture

## Data Input Format

* Window of previous prices
  + Gaussian Normalized? - <https://visualstudiomagazine.com/articles/2014/01/01/how-to-standardize-data-for-neural-networks.aspx>
  + Window Size?
* Currently in a position? E.g. 0 = No, -1 = Short, 1 = Long
* Position unrealised profit/loss
* Position entry point (in Normalized format?)

## Layers

* Convolutional layer
  + No striding as the position may have helpful information?
  + Batch Normalization
  + Softmax

## Data Output Format

* Action {nothing, buy, sell, exit}
* Trade Size {small, medium, large} (TBD – the exact size of each)

# Reinforcement Learning

## Learning Parameters

## Reward Function

## Trading Parameters/Rules

The following parameters and rules are for steering the trading bot in a particular direction. These will likely change over time. All tested rules and parameters will be kept as a historic reference (or put in an excel sheet if it gets too large).

Initial Ideas:

* Maximum time in a trade: 8 hours
* Trade sizes: small = US$50 per 50 pips, medium = US$200 per 50 pips, large = US$500 per 50 pips
* Starting balance US$2500

# Development for Production

The following section outlines further development required after the bot is built and the training process has yielded adequate results from back-testing.

## Broker Connect

A CFD broker must be chosen which offers an API for automated trading. This API will most likely need a wrapper to create the same interface as the broker simulator. It is also important that this API is available for a demo account so that the bot can be tested with real-time data (not just back testing historical data). This will allow for any obvious bugs/issues relating to real-time trading to be discovered before real capital is risked. Possible candidates are:

* IB Brokers (seems a bit too professional e.g. account fees)
* OANDA (rest API)
* IG (rest API)
* FXCM (Forex connect)
* PepperStone

# Broker Simulator

It makes more sense for the time being to use an existing trading simulator library in the interest of saving time and to begin the learning process. Once this is underway, a custom broker simulator can be implemented which will hopeful be faster than the current library backtrader.

The broker simulator accesses the provided minute bid data and posts this data to registered clients i.e. trading agent. It also provides a way for the agent to enter a position and tracks the position information.

## Interfaces

* For trading agent to get current price and execute orders
* For any custom rules to be enforced i.e. trades must not be open for more than x hours
* For a historic module to get information when positions are closed and account balance

## Order/Trade Related Data

* Trade Direction
* Entry Price
* Position Size – {small, medium, large}
* Unrealized Profit/Loss
* Time in Trade

## General Data

* Account Balance
* Is Position Open
* Close Bid/Ask