# Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM Neural Networks

David W. Lu Email: davie.w.lu@gmail.com

Abstract—With the breakthrough of computational power and deep neural networks, many areas that we haven’t explore with various techniques that was researched rigorously in past is feasible. In this paper, we will walk through possible concepts to achieve robo-like trading or advising. In order to accomplish similar level of performance and generality, like a human trader, our agents learn for themselves to create successful strategies that lead to the human-level long-term rewards. The learning model is implemented in Long Short Term Memory (LSTM) recurrent structures with Reinforcement Learning or Evolution Strategies acting as agents The robustness and feasibility of the system is verified on GBPUSD trading.

Keywords—Deep learning, Long Short Term Memory (LSTM), neural network for finance, recurrent reinforcement learning, evolution strategies, robo-advisers, robo-traders

摘要 - 随着计算密度和深度神经网络的突破，许多我们还没有用过去严格研究的各种技术进行探索的领域是可行的。 在本文中，我们将通过可能的概念来实现robo式交易或建议。 为了达到与人类交易者类似的表现和普遍性水平，我们的代理商会自己学习创建成功的战略，从而实现人类长期的回报。 该学习模型在长期短期记忆（LSTM）周期性结构中实施，其中强化学习或进化策略充当代理。系统的稳健性和可行性在GBPUSD交易中得到验证。

## INTRODUCTION

Many of the machine learning or artificial intelligence techniques can be trace back as early as the 1950s. Evolving from the study of pattern recognition and computational learning theory, researchers explore and study the construction of algorithms that can learn and predict on data. With the predictions, researchers came across the idea of of a learning system that can decide something, that adapts its behavior in order to maximize a signal from its environment. This was the creation of a ”hedonistic” learning system.[1] The idea of this learning system may be viewed as Adaptive Optimal Control, nowadays we call it reinforcement learning [2]. In order to accomplish similar level of performance and generality, like a human, we need to construct and learn the knowledge directly from raw inputs, such as vision, without any hand-engineered features, this can be achieved by deep learning of neural networks. Combining the two, some simply refer it to deep reinforcement learning, which could create an artificial agent that is as close as we can to sanely call it true ”artificial intelligence”.

早在20世纪50年代，许多机器学习或人工智能技术就可以追溯到。从模式识别和计算学习理论的研究发展而来，研究人员探索并研究了可以学习和预测数据的算法的构建。通过预测，研究人员发现了一个可以决定某些事情的学习系统的概念，该系统会调整其行为以最大化其环境中的信号。这是创建“享乐主义”的学习系统。[1]这个学习系统的想法可以被看作自适应最优控制，现在我们称之为强化学习[2]。为了达到与人类相似的性能和普遍性水平，我们需要直接从原始输入（如视觉）构建和学习知识，而不需要任何手工设计的特征，这可以通过深入学习神经网络来实现。把两者结合起来，一些人简单地将它称为深层强化学习，它可以创建一个人造代理，尽可能接近我们可以理解为真正的“人工智能”。

In this paper, we’ll focus with direct reinforcement or recurrent reinforcement learning to refer to algorithms that do not have to learn a value function in order to derive a policy. Some researchers called policy gradient algorithms in a Markov Decision Process framework as direct reinforcement to generally refer to any reinforcement learning algorithm that does not require learning a value function. Herein, we will focus on recurrent reinforcement learning. Methods such as dynamic programming[3], TD Learning[4] or Q-Learning[5] had been the focus of most modern researches. These method attempt to learn a value function. Actor-Critic methods[6], which is an intermediate between direct reinforcement and value function methods, in that the ”critic” learns a value function which is then used to update the parameters of the ”actor”.

在本文中，我们将重点介绍直接强化或反复强化学习，特指为了推导出策略而不必学习价值函数的算法。 一些研究人员将马尔科夫决策过程框架中的策略梯度算法称为直接强化，以泛指任何不需要学习价值函数的强化学习算法。 在此，我们将重点关注经常性强化学习。 诸如动态规划[3]，TD学习[4]或Q-Learning [5]等方法已成为大多数现代研究的焦点。 这些方法试图学习一个价值函数。 行为者 - 批评方法[6]是直接强化和价值函数方法之间的中介，因为“critic”学习了一个价值函数，然后用它来更新“actor”的参数。

Why we chose to focus on recurrent reinforcement learning? Though much theoretical progress has been made in the recent years, there had been few public known applications in the financial world. We as start-ups, quantitative hedge funds, client driven investment services, wealth management companies, and most recently robo-advisers, had been focusing on financial decision making problems to trade on its own. Within the reinforcement learning community, much attention is actually given to the question of learning policies versus learning value functions. The value function approach as described earlier had dominated the field throughout the last thirty years. The approach had worked well in many applications, alpha Go, training a Helicopter to name a few. However, the value function approach suffers from several limitations. Q-Learning is in the context of action spaces and discrete state. In many situations, this will suffer the ”curse of dimensionality” When Q-Learning is extended to function approximators, researchers had shown it fail to converge using a simple Markov Decision Process. Brittleness which means small change in the value function can produce large changes in the policy. In the trading signal world, the data can be in presence of large amounts of noise and nonstationarity in the datasets, which could cause severe problems for a value function approach.

为什么我们选择关注经常性强化学习？尽管近年来取得了许多理论上的进步，但金融领域的公共应用还很少。我们作为创业公司，定量对冲基金，客户驱动型投资服务公司，财富管理公司以及最近的机器人顾问，一直专注于财务决策问题，以自己进行交易。在强化学习社区中，实际上对学习政策与学习价值函数的问题给予了很多关注。前面描述的价值函数方法在过去的三十年中一直是该领域的主导。该方法在许多应用中运行良好，alpha Go，培训直升机等等。但是，价值函数方法受到几个限制。 Q-学习是在行动空间和离散状态的背景下进行的。在许多情况下，这将会遭受“维数灾难”。当Q-Learning扩展到函数逼近器时，研究人员已经表明它不能使用简单的马尔可夫决策过程进行收敛。脆弱性意味着价值函数的微小变化会导致政策的巨大变化。在交易信号世界中，数据可能存在大量噪音和数据集中的非平稳性，这可能会导致严重的价值函数方法问题。

Recurrent reinforcement learning can provide immediate feedback to optimize the strategy, ability to produce real valued actions or weights naturally without resorting to the discretization necessary for value function approach. There are other portfolio optimization techniques such as evolution strategies and linear matrix inequilities which relies on predicting the covariance matrix and optimize. For all optimization problems or in reinforcement learning set up, we need an objective, and such objective can be formulated in terms of risk or rewards. Moody et al.[7] shown that how differential forms of the Sharpe Ratio and Downside Deviation Ratio can be formulated to enable efficient on-line learning with recurrent reinforcement learning, Lu[8] had shown using Linear Matrix Inequilities can beat the risk free rate, and Deng et al.[9] had shown max return can be used as objective in recurrent reinforcement learning as well as using deep learning transformations to initialize the features.

经常性强化学习可以提供即时反馈，以优化策略，自然产生真实价值行为或权重的能力，而不诉诸价值函数方法所必需的离散化。还有其他的投资组合优化技术，如演化策略和线性矩阵不等式，它们依赖于预测协方差矩阵和优化。对于所有优化问题或强化学习设置，我们需要一个目标，并且可以根据风险或奖励制定此类目标。穆迪等人[7]表明夏普比率和下降偏差比率的差异形式如何能够通过反复强化学习来实现有效的在线学习，Lu [8]使用线性矩阵不等式可以打败无风险利率，邓等人[9]已经表明最大回报可以用作经常性强化学习的目标，也可以使用深度学习转换来初始化特征。

To extend the recurrent structure, we’ll further discuss in this paper how the back propagation through time method is exploited to unfold the recurrent neural network as a series of time-dependent stacks without feedback. As discuss in [9], gradient vanishing issue is inevitably true in these structures. This was because the unfolded neural networks exhibits extremely deep structures on the feature learning and also the time expansion parts. We introduce the Long Short Term Memory (LSTM) to handle this deficiency. We will discuss the characteristics of LSTM as well as thoughts and techniques such as Dropouts [10] to test. This strategy provides a chance to forecast the final objective and improve the learning efficiency.

为了扩展循环结构，我们将在本文中进一步讨论如何利用时间反向传播方法将循环神经网络展开为一系列没有反馈的时间相关叠加。 正如[9]中所讨论的那样，渐变消失问题在这些结构中是不可避免的。 这是因为展开的神经网络在特征学习上表现出非常深的结构，并且还有时间扩展部分。 我们引入长期短期记忆（LSTM）来处理这个缺陷。 我们将讨论LSTM的特点以及诸如Dropouts [10]的想法和技巧来测试。 这个策略提供了预测最终目标并提高学习效率的机会。

The recurrent reinforcement learner requires to optimize the objective through gradient ascent. In this paper, we will also explore literature in Evolution Strategies [11] and NelderMead method [12] as to search for the gradient or so called direct search or derivative-free methods.

经常性强化学习者需要通过梯度上升来优化目标。 在本文中，我们还将探索Evolution 策略[11]和NelderMead方法[12]中的文献，以搜索梯度或所谓的直接搜索或无导数方法。

Finally, the Trading Systems will be tested among S&P 500, EURUSD, and commodity futures market. The remaining parts of this part are organized as follows. Section II, we will walk through how we construct the trading agents, Section III reveals how we construct the recurrent layers in plain recurrent and LSTM. Additionally, how dropouts can affect the training and reduce gradient vanishing issues. Section IV, we will talk about gradient ascent, evolution strategies and HelderMead Methods. Section V, we will detail the test results and comparison of methods listed in Section II to IV. Section VI concludes his paper and provides thoughts on future directions.

最后，交易系统将在标准普尔500指数，欧元兑美元和商品期货市场之间进行测试。 这部分的其余部分组织如下。 第二部分，我们将介绍我们如何构建交易代理人，第三部分揭示了我们如何构建简单循环和LSTM的复发层。 此外，辍学者如何影响训练并减少渐变消失问题。 第四部分，我们将讨论梯度上升，演化策略和HelderMead方法。 第五部分，我们将详细介绍第二部分至第四部分所列方法的测试结果和比较。 第六节总结他的论文，并提供未来方向的想法。

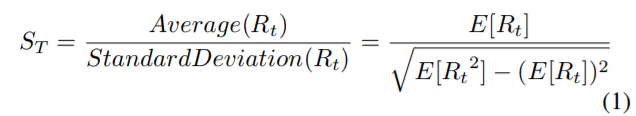
## RECURRENT REINFORCEMENT LEARNING

To demonstrate the feasibility of agents that trades, we consider agents that trade fixed position size on a single security. The methods described here can be generalized to more sophisticated agents that trades or optimize a portfolio, trades varying quantities of a security, allocate assets continuously or manage multiple asset portfolios. We’ll discuss this further in separate sessions. See [13] for some initial discussions.

为了证明交易代理商的可行性，我们考虑代理商将单一证券的固定头寸交易。 这里描述的方法可以推广到更复杂的代理商，这些代理商可以交易或优化投资组合，交易不同数量的证券，持续分配资产或管理多个资产组合。 我们将在单独的会议中进一步讨论。 参见[13]进行一些初步讨论。

Intuitively, we find a objective function so that the agent knows what we are trying to maximize or minimize. Most modern fund managers attempt to maximize risk-adjusted return using the Sharpe Ratio, as suggested by modern portfolio theory. The Sharpe Ratio is define as follows[14]:

直觉上，我们找到一个目标函数，以便代理知道我们想要最大化或最小化的内容。 正如现代投资组合理论所指出的，大多数现代基金经理试图使用夏普比率来最大化风险调整回报。 夏普比率定义如下[14]：



where Rt is the return on investment for trading period t and E[.] denotes the expectation. In modern portfolio theory, higher Sharpe Ratio rewards investment strategies that rely on less volatile trends to make a profit. As discussed earlier, there are other functions or ratios we can use, however for ease of demonstration purposes we will use Sharpe Ratio and Downside Deviation Ratio in this article.

其中Rt是交易期t的投资回报率，E [.]表示期望值。 在现代投资组合理论中，较高的夏普比率奖励依赖较不稳定趋势获利的投资策略。 如前所述，我们可以使用其他功能或比率，但为了便于演示，我们将在本文中使用夏普比率和下偏差比率。

Next step we need to define how a agent would trade. The trader would take a long, neutral or a short position. A long position is entering a purchase of some quantity of a security, while a short position is triggered by selling the security. Herein, we will follow mostly the notation in [7][15] for the ease of explaining and reconciliation. Let’s define Ft ∈ [−1, 0.1] represents the trading positions at time t. A long position when Ft > 0. In this case, the trader buys the security at price Pt, and hopes that the prices goes up on period t+1. A short position is when Ft < 0. In this case, the trader short sale (borrow to sell) the security at price Pt, and hopes that the prices goes down on period t+1 so that trader can buy it back to return the security that it borrowed. Intuitively, one can use a Tanh function to represent this set up since it’s goes from -1 to 1.

下一步，我们需要定义代理商将如何交易。 交易者需要持有多头，中性或空头头寸。 多头头寸进入购买某种数量的证券，而空头头寸则通过出售证券来触发。 在这里，我们将主要遵循[7] [15]中的表示法，以便于解释和和解。 我们定义Ft∈[-1,0.1]表示t时刻的交易头寸。 Ft> 0时的多头头寸。在这种情况下，交易者以价格Pt购买证券，并且希望价格在时期t + 1上升。 当Ft <0时，空头头寸。在这种情况下，交易者以价格Pt卖出（借入卖出）证券，并且希望价格在t + 1期间下跌，以便交易者可以买回它返回 它借来的安全性。 直观地说，可以使用Tanh函数来表示这个设置，因为它从-1到1。

We define the trader function in a simple form of:



where xt = [rt−m+1, ...rt] and the return rt = pt −pt−1 ,Note that the trader function can also add in a bias term b and the last trading decision with parameter u to add into the regression. The latest trading decision with parameter can discourage the agent to frequently change the trading positions and to avoid huge transaction costs. We can then rewrote the equation to:

其中xt = [rt-m + 1，... rt]和回报rt = pt -pt-1,请注意，交易者函数还可以添加一个偏倚项b和最后一个带参数u的交易决策添加到回归。 具有参数的最新交易决策可能会阻止代理商频繁更改交易头寸并避免巨额交易成本。 然后，我们可以将方程重写为：



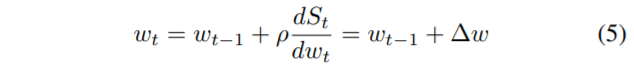
Adding the number of shares as s with transaction cost c, we can then write the return at time t as:

加入股票数量s和交易成本c，我们可以得出在t时刻的回报：



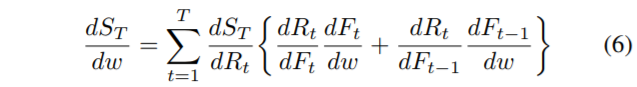
With the above elements set up, we can now try to maximize the Sharpe Ratio using Gradient Ascent or other methods which we’ll discuss further in Section IV to the find the optimal weights for the agent to use. Again let’s think through given trading system model Ft, the goal is to adjust the parameter or weights w in order to maximize ST . We can write the weights as follows:

通过设置上述要素，我们现在可以尝试使用梯度上升或其他方法来最大化夏普比率，我们将在第四节中进一步讨论寻找代理使用的最佳权重。 再次让我们考虑通过给定的交易系统模型Ft，目标是调整参数或权重W以最大化ST。 我们可以按如下方式编写权重：

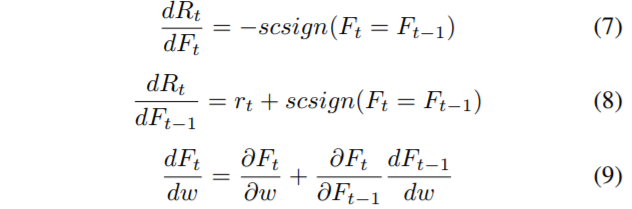


where wt is any weight of the network at time t, St is the measure we’d like to maximize or minimize, and ρ is an adjustable learning rate.

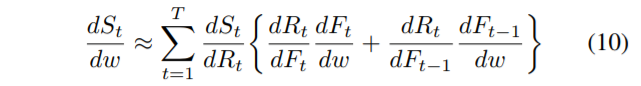
Examining the derivatives of ST or gradient with respect to the weight w over a sequence of T periods is:



The trader can then be optimized in batch mode by repeatingly compute the value of ST on forward passes through the data with:



Due to the inherent recurrence, the quantitites dFt/dw are total derivatives that depend upon the entire sequence of previous time periods. In other words, dFt/dw is recurrent and depends on all previous values. Though it does slow down the gradient ascent but due to modern computational power and range of samples, it does not present insuperable burden. To correctly compute and optimize these total derivatives, we can deploy a similar bootstrap method as in Back-Propagation Through Time(BPTT)[16]. Alternatively, one can use a simple on-line stochastic optimization by consider only the term in (6) that depends on the most recent realized return Rt during a forward pass through the data. The equation in (6) becomes:

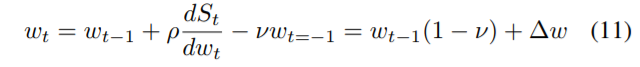
由于固有的复发，定量dFt / dw是取决于以前时间段的整个序列的总导数。 换句话说，dFt / dw是经常性的并且取决于以前的所有值。 尽管它确实减缓了梯度上升，但是由于现代计算能力和样本范围的原因，它并不具有无法承受的负担。 为了正确地计算和优化这些总体导数，我们可以采用与Back-Propagation Through Time（BPTT）[16]相似的引导方法。 或者，可以使用简单的在线随机优化，只考虑（6）中的术语，该术语取决于在向前传递数据期间最近实现的回报Rt。 （6）中的等式变为：

Such an algorithm performs a stochastic optimization or effectively making the algorithm a stochastic gradient ascent. As we previously mentioned, there are other methods to maximize the objective function. We’ll discuss that further in section IV.

We also tested the weight decay variant of the gradient ascent learning algorithm as described in [15] to verify the performance. Using the weight decay, (5) becomes:

这种算法执行随机优化或有效地使算法成为随机梯度上升。 正如我们前面提到的，还有其他方法可以使目标函数最大化。 我们将在第四节进一步讨论。

我们还测试了[15]中描述的梯度上升学习算法的权重衰减变体，以验证性能。 使用权重衰减，（5）变成：



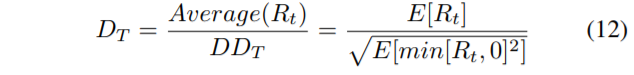
where ν is the co-efficient of weight decay. Adding the weight decay improves neutral network performance based on the fact that smaller weights will have less tendency to over- fit the noise in the data. Similar to the findings in [15], the weight decay will not help single layer neural networks since it’s theoretically for the purpose to simplify the rule learned by the neural network and prevent the neural network from memorizing noise in the data. The next section will introduce the deep learning transformation and dropouts to better fine tune the performance.

其中ν是权重衰减的系数。 增加权重衰减可以改善中性网络的性能，因为权重越小，数据中过度拟合噪声的倾向越小。 与[15]中的发现类似，权重衰减对单层神经网络无帮助，因为它在理论上是为了简化神经网络学习的规则，并防止神经网络记住数据中的噪声。 下一节将介绍深度学习转换和退出以更好地调整性能。

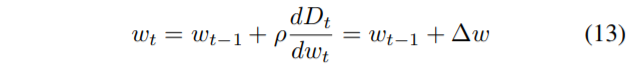
Although Sharpe ratio is the most widely used risk-adjusted metric, it provides rankings that it is counter-intuitive investors’ sense of risk because the use of variance or as risk measure does not distinguish between upside and downside risk, therefore penalize both large positive or negative returns. To most investors, the risk refers to returns in a portfolio that decreases its profitability. In this paper, we will experiment both signals with recurrent neural network and downside deviation ratio to protect downside risk.

虽然夏普比率是最广泛使用的风险调整指标，但它提供的排名是反直觉投资者的风险意识，因为使用方差或作为风险指标并没有区分上行风险和下行风险，因此惩罚了两者 巨大的正面或负面回报。 对大多数投资者而言，风险是指投资组合中的收益降低其盈利能力。 在本文中，我们将用循环神经网络和下降偏差比率对两个信号进行实验，以保护下行风险。

Similar to equation (1), we can define downside deviation ratio as follows:



Equation (5) becomes：



Computationally, it will be easier if 0 here is described as a very small number. We will check the performance of downside deviation ratio and Sharpe ratio in Section V.

## LSTM FOR INFORMATIVE FEATURE LEARNING

To further this research, we attempt to find efficient algorithms that takes the decision objective into account when estimating either the covariance matrix [17] or the features [9]. As an example of the former, the Directed Principal Component Analysis [17] is stated for estimating the covariance matrix with the decision objective in mind. Such method is useful for portfolio estimations and predictions. The latter, which we attempt to use a deep neural network transformation or a fuzzy learning method to help understand the signals we feed into the recurrent reinforcement learning structure[9]. Herein, we will explore using Long Short Term Memory.

为了进一步研究，我们试图找到在估计协方差矩阵[17]或特征[9]时考虑决策目标的有效算法。 作为前者的一个例子，定向主成分分析[17]是为了估计具有决策目标的协方差矩阵而陈述的。 这种方法对投资组合估算和预测很有用。 后者，我们尝试使用深度神经网络变换或模糊学习方法来帮助理解我们馈入到循环强化学习结构中的信号[9]。 在此，我们将探讨使用长期短期记忆。

We implemented LSTM(Long Short Term Memory) [18] to understand and dynamically sense the market condition and use it for informative feature learning. In theory, the appeals of Recurrent Neural Networks is the idea that they might be able to connect previous information to the present task we are aiming to achieve. Unfortunately, in practice, it is possible for the gap between the relevant information and point where it is needed become very large. As the gap grows, RNNs become unable to learn to connect the information [19]. LSTM was first introduced in 1997 [18] to resolve the difficulties to model long sequences. The fundamental issue was that gradients propagated over many stages tend to either vanish or explode. In a traditional recurrent neural network, during the gradient back-propagation phase, the gradient signal can end up being multiplied by humongous times perhaps as many times as of the timesteps by the weight matrix associated with the connections between the neurons of the recurrent hidden layer. In other words, the magnitude of weights in the transition matrix can have a large impact on the learning process. If the weights in this matrix are small it leads to vanishing gradients where the gradient signal gets so small that learning either becomes very slow or stops working altogether. In defiance of this, if the weights in this matrix are large where the gradient signal is large, where we often refer this as exploding gradients.

我们实现了LSTM（长期短期记忆）[18]来理解和动态感知市场状况，并将其用于信息特征学习。从理论上讲，Recurrent Neural Networks的吸引力在于他们可能能够将以前的信息与我们期望实现的目前任务联系起来。不幸的是，在实践中，相关信息与所需信息之间的差距可能变得非常大。随着差距的扩大，RNN变得无法学会连接信息[19]。 LSTM在1997年首次引入[18]来解决长序列模型的困难。根本的问题是，在许多阶段传播的梯度趋于消失或爆炸。在传统的递归神经网络中，在梯度反向传播阶段期间，梯度信号最终可能乘以与时间步长相同的很多倍，这是由与重复隐藏层的神经元之间的连接相关联的权重矩阵。换句话说，转换矩阵中权重的大小会对学习过程产生重大影响。如果矩阵中的权重较小，则会导致梯度信号变得很小的梯度消失，导致学习变得非常缓慢或完全停止工作。无视这一点，如果这个矩阵中的权重在梯度信号很大的情况下很大，我们经常将其称为爆炸梯度。

Previously, we talked about the issues based on a recurrent neural network. These issues are the main motivation behind the LSTM model which introduces a new structure called a memory cell. A memory cell is built with four main elements: an input gate, a neuron with a self-recurrent connection, a forget gate and an output gate. The self-recurrent connection ensure the state of a memory cell can remain constant from one timestep to another. The input gate allows incoming signal to alter the state of the memory cell or block it. The output gate can allow the state of the memory cell to have an effect on other neurons or prevent it. Finally, the forget gate can modulate the memory cells self-recurrent connection, allowing the cell to remember or forget its previous state, as needed. We may wonder why does a LSTM have a forget gate when their purpose is to link distant occurrences to a final output. When we are analyzing a time series and comes to the end of it, for example, you may have no reason to believe that the next time instance has any relationship to the previous, and therefore the memory cell should be set to zero before the next instance. In Figure 1, we can see how gates work, with straight lines representing closed gates and open circles representing open gates. The lines and circles running horizontal up the hidden layer are the forget gates.

以前，我们讨论了基于循环神经网络的问题。这些问题是引入称为存储单元的新结构的LSTM模型背后的主要动机。存储器单元由四个主要元件构成：输入门，具有自回归连接的神经元，忘记门和输出门。自回归连接可确保存储单元的状态从一个时间步到另一个时间步保持不变。输入门允许输入信号改变存储单元的状态或阻止它。输出门可以允许存储单元的状态对其他神经元产生影响或阻止它。最后，忘记门可以调制存储器单元的自回复连接，从而允许单元根据需要记住或忘记其以前的状态。我们可能会想知道为什么LSTM的目的是将远处的事件与最终的输出连接起来时会有一个忘记的门。例如，当我们分析一个时间序列并结束它时，您可能没有理由相信下一个时间点与前一个时间点有任何关系，因此在下一个时间点之前应将存储单元设置为零实例。在图1中，我们可以看到门是如何工作的，直线表示关闭的门和开放的圆表示打开的门。在隐藏层上水平运行的线条和圆圈是忘记门。

With the concept in mind, let’s walk through the mathematical expressions. We’ll try to use the notations as simple as possible here to explain. Please see [20] for further readings and detailed discussions. Note that the notation used in this section is not the same representation if any overlap as in Section II. Here xt is the input vector at time t, ht is the hidden layer vector, the W are input weight matrices and U are the recurrent weight matrices and b are bias vectors. Functions σ, m, and n are point-wise non-linear activation functions. Logistic sigmoid  is used for activation functions of the gates or σ and hyperbolic tangent tanh is used as the block input and out activation functions (m, n). Finally, the point-wise multiplication of two vectors is denoted with ,We can write the expressions as follows:

记住这个概念，让我们来看看数学表达式。 我们将尝试尽可能简单地使用符号来解释。 请参阅[20]以获得进一步阅读和详细讨论。 请注意，本节中使用的表示法与第II节中的重叠表示法不同。 这里xt是时间t处的输入向量，ht是隐藏层向量，W是输入权矩阵，U是递归权矩阵，b是偏向量。 函数σ，m和n是点式非线性激活函数。 Logistic Sigmoi()用于门或σ的激活函数，双曲正切tanh用作块输入和输出激活函数（m，n）。 最后，两个向量的逐点相乘表示为.

  我们可以写下如下表达式：

