**王柏霳**

0926-137427‧Brownwang0426@gmail.com

|  |  |  |
| --- | --- | --- |
| **教育程度** | **國立台灣大學**，法學碩士 五月，2015  <https://goo.gl/S1Y8RV> | |
| **國立台灣大學**，法學學士 七月，2011  <https://goo.gl/9XEQha> | |
| 榮譽： | 最佳優良讀書小組獎，國立台灣大學，2011 |
| **高雄市市立高級中學**  七月，2006 | |
| **相關經歷** | **歐盟法律研究中心**，國立台灣大學 2012–2013  行政助理  負責中心期刊以及文章之發行，協助舉辦多起國際研討會，其間包含「Criminal Law International Symposium, Preemptive Deterrence of Corruption Series」（由立法院贊助）、「The International Academic Symposium on The Values of Peace, Freedom and Equality in Europe and Asia」以及「Human Right and Holocaust」。 | |
|  | **國家科學委員會研究計畫**（指導教授：陳自強） 2012–2014  研究助理  負責翻譯聯合國統一契約文件以及歐洲聯盟DCFR文件，設計報告內容，並觀察日本近年債法改正之近況以及未來動向。 | |
|  | **歐盟法與生命對話課程**，國立台灣大學  2012–2014  教學助理  設計課程活動，幫助學生課業上之問題，協調學生以及授課教師。 | |
|  | **律師**，經兆國際法律事務所  2016–2018  受雇律師  負責民刑事行政訴訟，非訟英文契約草擬等。 | |
| **考取證照** | 律師（中華民國），名次62 2012  <https://goo.gl/oFV5Dv> | |
|  | TOEIC，總分975 2014  <https://www.dropbox.com/s/xhj2e8vmke7fww3/IMG_3613.JPG?dl=0> | |
| **公開著作** | 論履行請求權之排除，從英美法、日本債法改正以及賽局理論之觀點  網址：  <http://handle.ncl.edu.tw/11296/hng27s> | |
| **其他技能** | 電腦：Python、C、C++ | |
|  | 語言：中文、英文以及日文（閱讀） | |
| **個人資料** | 興趣：人工智慧、賽局理論、電腦程式、數學、機器學習、自然語言學習。  個人Linkedln：<https://www.linkedin.com/in/brown-%E7%8E%8B%E6%9F%8F%E9%9A%86-68bb4938/>  個人Github：  <https://github.com/Brownwang0426/Deep_Neural_Game_Theory> | |
| **程式說明** | 代表性著作程式: Deep Neural Game Theory深度網路賽局理論  在傳統的深度學習底下，是希望藉由Back Propagation去調整人工神經網路本身的Synapse，使得輸入端Input Neurons因此所獲得的輸出端Output Neurons，可以切進目標數值Target Neurons。  這是Hinton於1980年代發明Back Propagation最原初的目的。  這是藉由已經知道的輸入端，去推測應有的目標數值。  但是當輸入端有部分資訊是遺漏的，機器是否能夠自行【推論】應有的輸入端的數值，則是有疑問。  比方說以最微小的蚊子為例，當我們的手在他前面揮舞，蚊子接受到此一輸入端的資訊時，為何他可以自行推論出，大腦輸出飛離現場的資訊到肌肉，進而牽動自己的翅膀而飛離現場，對自己會是最佳的策略、好讓自己生存?  傳統的機器需要靠人力在輸入端輸入手部揮動的資訊，在目標數值設定飛走的資訊。所以機器以後看到手部揮動的資訊，他就會輸出飛走的資訊。  但是這個過程當中，我們已經隱存地將人類既有的觀念放到機器裡面了，就是【飛走事實上可以讓自己存活】，但是機器無須知道飛不飛走會不會對生存有所影響的資訊，也無須對此做出推理。當這個機器蚊子下次碰到了電蚊拍，他就不會飛走了，他也就死了。既便是他把這資訊存放在他的靈魂(或者是雲端)裡面，他下次也不會知道如何使用這些資訊，他還是會再死一次。除非人類手動再去重複以上的輸入方法，否則這機器蚊子永遠不會知道電蚊拍對他的威脅。當然，如果人類之於上帝，而蚊子之於人類，上帝應該會累死。  這樣可以說是機器擁有推理、決策能力了嗎?恐怕不是  為了解決這個問題，本文將Back Propagation的概念更進一步昇華、應用，本文是先將已經被訓練好的類神經網路，將其中部分的Input Neurons挖空，透過Back Propagation強迫這個訓練好的類神經網路去找出部分空缺的Input Neurons的最佳解答，以符合預先設定的期望的目標數值Target Neurons。但是不改變整個被訓練好的類神經網路的Synapse。  例如先訓練機器簡單的加法，例如1+1=2 ，1+2=3 ，A(Input Neurons)+B(Input Neurons)=C(Target Neurons) 。訓練完畢之後，告訴機器B=2 ，C=10  看類神經網路是否可以透過Back Propagation去找出A=8，使得整體輸出端可以切進目標數值，也就是C=10。  看似沒什麼，但是如果輸入的數值是，不飛走(Input Neurons)+手部揮舞(Input Neurons)=死亡(Target Neurons)，飛走(Input Neurons)+手部揮舞(Input Neurons)=生存(Target Neurons)的話，訓練完畢之後，我們可以告訴機器當人類手部揮舞時，要採取怎樣的行動才可以生存(使輸出端切進生存)，而這樣卻神奇地可以使機器推論出【飛走事實上可以讓自己存活】的最佳策略。  換而言之，Hinton的機器是透過調整神經連結值而使輸出值切進目標數值，而這邊的機器則是走了逆向、相反的方向－這邊反而是調整已經訓練過的機器的輸入值，並使最後的輸出值切進目標數值，使機器擁有決策推理能力，可以思考怎樣的行為(也就是輸入端)，可以符合自己的最佳利益，但是整體的神經連結值完全並無變動。這種機器可以說是擁有自我意識，理解採取怎樣的行為會對自己是最有利的，訓練的過程可以是機器一邊嘗試錯誤、一邊學習（每死一次就會變得聰明一次），或者是由人類專家將各個行為與外界環境的交互結果的資訊先輸入機器中。  那這個跟賽局理論有何關聯呢?  在賽局理論底下，每個玩家都是根據對方玩家的行動做出反應，以謀求自己利益的最大化，這個跟上面蚊子根據人類手部的行為作出反應因此飛走，以謀求自己利益的最大化是一樣的。既然上面的機器可以模擬蚊子的思考模式，推測出飛走會是最佳的策略。想當然爾，這個機器也可以模擬兩個玩家的想法，模擬兩個玩家的策略模式。  舉例而言，以最傳統的賽局理論的同時賽局(Simultaneous Game)為例:  “simultaneous game”的图片搜索结果【Figure 1】  如果我們把兩個玩家的策略(T, M, B)以及(L, C, R)當作是兩個輸入端，這一個人工神經網路的目標數值端則是因應上圖而對應的(0.3, 0.4)等等，將這些數值輸入訓練人工神經網路，當此一神經網路訓練完畢之後，我們可以套用前面的A(Input Neurons) + B(Input Neurons) = C(Target Neurons) 的訓練方法，只是這次A 與B 所希望達成的C 是相反，A 是希望達到(1,0) ，B 是希望達到(0,1)，訓練輸入端的方式也就是，先讓玩家column先走(先讓玩家column的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(1, 0)的輸入值)，再讓玩家row再走(再讓玩家row的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(0, 1)的輸入值)，最後重複以上動作。  最終機器可以自動找出納許均衡Nash Equilibrium，也就是(T, L)以及(M, C)。  想看點更暴力的嗎? 沒問題。  以上只是傳統的一般的深度網路神經Deep Feedforward Neural Network與賽局理論同時賽局Simultaneous Game的結合，但是當然這個觀念，也是套用到遞迴網路神經Recurrent Neural Network(包括LSTM、Neural Turing Machine等)上面，使遞迴網路神經Recurrent Neural Network可以與賽局理論結合，但是，這是怎樣的一個觀念呢? 詳見下述。  在賽局理論底下，除了以上的同時賽局(Simultaneous Game)以外，也包括了序列賽局(Sequential Game)，也就是大家熟悉的樹狀圖，如下所示:  “sequential game”的图片搜索结果 【Figure 2】  在這序列賽局底下，各個玩家是如何揣摩對手的想法，以做出對自己最有利的決策呢?  就是採取了反向推論Back Deduction的做法，在傳統的賽局理論底下，玩家P1在第三層會去揣摩玩家P2在第四層次的想法，並做出反應，以謀求自己利益的最大化，相同地，玩家P2在第二層會去揣摩玩家P1在第三層的以上的想法，並做出反應，以謀求自己利益的最大化，最後，玩家P1在第一層會去揣摩玩家P2在第二層的以上的想法，並做出反應，以謀求自己利益的最大化。  那遞迴網路神經Recurrent Neural Network可以如何與賽局理論中的序列賽局Sequential Game結合呢?  在人工智慧底下，除了以上一般的深度網路神經Deep Feedforward Neural Network外，也包括遞迴網路神經Recurrent Neural Network(包括LSTM、Neural Turing Machine等，統稱RNN)，其形狀如大家熟悉的，長得如下:  Input Neurons  Hidden Neurons  Output Neurons  同樣地，在剛剛我們所說的傳統的賽局理論的同時賽局(Simultaneous Game)與傳統的一般的深度網路神經Deep Feedforward Neural Network的結合過程當中(詳見Figure 1)，我們是先讓玩家column先走(先讓玩家column的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(1, 0)的輸入值)，再讓玩家row再走(再讓玩家row的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(0, 1)的輸入值)，最後重複以上動作，最終我們可以讓機器自動找到Nash Equilibrium。  相同地，在序列賽局Sequential Game與遞迴網路神經底的結合過程當中，我們也是將遞迴網路神經的每一層輸入端，當作是序列賽局底下，每一個玩家每一次所選取的策略，詳如下圖:  “sequential game”的图片搜索结果  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  首先，我們也是先透過Back Propagation，先讓第四層的玩家P2先走(先讓玩家P2在第四層的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(0, 1)的輸入值)，詳如下圖:  Target Neurons = (0, 1)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  再來，我們保留第四層的玩家P2的剛剛所選取的策略，再讓第三層的玩家P1再走(再讓玩家P1在第三層的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(1, 0)的輸入值)，詳如下圖:  Target Neurons = (1, 0)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  再來，我們保留剛剛所有玩家所選取的策略，再讓第二層的玩家P2再走(再讓玩家P2在第二層的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(0, 1)的輸入值)，詳如下圖:  Target Neurons = (0, 1)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  最後，我們保留剛剛所有玩家所選取的策略，再讓第一層的玩家P1再走(再讓玩家P1在第一層的輸入端透過Back Propagation以及Gradient Descent先找出切進目標數值(1, 0)的輸入值)，詳如下圖:  Target Neurons = (1, 0)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  最後的最後，我們讓機器重複以上動作。  最終第一層的玩家P1，很神奇地，可以找出對自己最有利的策略。  實際的應用上，以最簡單的井字遊戲，也就是Tic Tac Toe為例，我們先讓機器理解到當下情形以後所有可能的棋步與其附隨的結果(這可以透過預先設定，或者是讓機器從錯誤當中不斷學習、累積資訊而得)，訓練完畢機器之後，我們再透過以上方式，使當下要舉棋下棋的玩家可以自動找到對自己往後最有利的落棋點。  此部分詳細的內容可以詳細參酌作者的Github:  <https://github.com/Brownwang0426/Deep_Neural_Game_Theory>  很有趣的是，賽局理論有反向推論法Back Deduction，而人工智慧有反向傳播法Back Propagation，賽局理論有同時賽局Simultaneous Game以及序列賽局Sequential Game，而人工智慧有傳統深度網路神經DFNN以及遞迴網路神經RNN，而賽局理論與人工智慧兩邊的概念可以巧妙結合如上，發揮極致破壞力。  技術展望方面，這種技術第一個可以用到遊戲(即時戰略、動作遊戲、棋類遊戲)上面，加強機器的推理能力，抗衡人類玩家。  第二個可以用到股市等涉及人類思維因素的時間序列數值，使機器依照現有狀況，隨時、即時做出最佳反應。  第三個，也就是最後一個，就是使機器擔任類似人類政策顧問的角色，輔助人類對於政策做出最佳判斷，此一政策包含交通管制、立法政策、犯罪預防等，也就是機器立法者、司法者的雛型。 | |

**Bo-Long Wang**

0926-137427‧Brownwang0426@gmail.com

|  |  |  |
| --- | --- | --- |
| **EDUCATION** | **NATIONAL TAIWAN UNIVERSITY**, Master of Law May, 2015  <https://goo.gl/mLreNN> | |
| **NATIONAL TAIWAN UNIVERSITY**, Bachelor of Law July, 2011  <https://goo.gl/9XEQha> | |
| Honors: | Winner of “Best Study Group”, National Taiwan University, 2011 |
| **KAOHSIUNG MUNICIPLE KAOHSIUNG SENIOR HIGH SCHOOL**  July, 2006 | |
| **EXPERIENCE** | **EUROPEAN UNION LAW RESEARCH CENTER (EULRC)**, NTU 2012–2013  *Administrative Assistant*  Engaged in collating and publishing periodicals. Coordinated several international seminar events, including: “Criminal Law International Symposium, Preemptive Deterrence of Corruption Series”, sponsored by Legislative Yuan (ROC), “The International Academic Symposium on The Values of Peace, Freedom and Equality in Europe and Asia”, and “Human Right and Holocaust”. | |
|  | **NATIONAL SCIENCE COUNCIL PROJECT,** Prof. Tzu-Chiang Chen  2012–2014  *Research Assistant*  In charged of translating UN Laws and organizing legislative data concerning DCFR, European Union. Designed research reports. Observed the legislative progress of Japanese contract law. | |
|  | **THE DIALOGUE BETWEEN EU LAW AND LIFE COURSE**, NTU 2012–2014  *Teaching Assistant*  Designed courses activities. Helped students with questions. Connected students with teachers. | |
|  | **MERIDIAN ATTORNEYS-AT-LAW**  2016–2018  *Employed Attorney*  Practice in legal suit. | |
| **LICENSE** | BAR EXAM (ROC), rank 62 2012 | |
|  | TOEIC, score 975 2014  <https://www.dropbox.com/s/xhj2e8vmke7fww3/IMG_3613.JPG?dl=0> | |
| **PUBLICATIONS** | On Specific Performance: From Common Law, the Modernization of Japanese Contract Law and Game Theory Perspectives, Master Thesis, National Taiwan University  Site:  <http://handle.ncl.edu.tw/11296/hng27s> | |
| **SKILLS** | Computer: Python, C, C++ | |
|  | Language: Chinese, English, Japanese (reading) | |
| **PERSONAL** | Interests: Artificial intelligence, Game Theory, Programming, Mathematics, Machine Learning, and Natural Language Process  Linkedln:  <https://www.linkedin.com/in/brown-%E7%8E%8B%E6%9F%8F%E9%9A%86-68bb4938/>  Github:  <https://github.com/Brownwang0426/Deep_Neural_Game_Theory> | |
| **SOURCE CODE**  **EXPLANATION** | Representative source code: Deep Neural Game Theory  Under traditional deep learning, we exert the notion of Back Propagation to adjust the synapses of neural network to make the output neurons to match its expected target neurons.  This is at least what Jeoffrey Hinton originally intended around 1980s when he first invented the notion of back propagation and deep learning.  In this notion, we let the machine to infer the target neurons by already-known input neurons.  However, when some information is lost in the input neurons, how can the machine make deduction about the supposed but lost input neurons?  For example, as tiny as a mosquito, whenever faced with a waving hand, it can make deduction about what movement best fits its interest – that is just flying away less that it gets smashed by human hands.  A traditional method is that we manually set the input neurons as “a waving hand” and the target neurons as “flying away”. In this notion, we can expect that this mechanic mosquito can fly away whenever it sees a waving hand.  However, did you notice that, in this process, we implicitly install a human-bestowed knowledge that is “flying away save your life”? The mechanic mosquito never needs to learn about whether flying way saves its life or not. When next time this mosquito meets a mosquito beat, it won’t fly away and it dies. Even though this mosquito dies with this information in its soul (or in the google cloud), it still does not know how to use it. It dies next time as well. Human engineer must further state the input neurons as “a mosquito beat” and the target neurons as “flying away” this time.  If this mechanic mosquito were to human and human engineer were to God, God will be tired to death.  In this case, can we really proudly say that we invented a machine that can make deduction and infer the best strategy for itself? Probably not.  To solve this problem, this paper further exploits and explore the notion of Back Propagation to bestow the machine with the ability to make deduction. For a well-trained neural network, this paper purposely vacuumizes some of its input neurons. Through the notion of Back Propagation, we force the neural network to make deduction about the best input neurons (which was vacuumized) to fit its expected pre-set target neurons. However, the well-trained synapse is intact and left unchanged in this process.  For example, we can force the machine to learn some basic algebra such as 1+1=2, 1+2=3, A(Input Neurons)+B(Input Neurons)=C(Target Neurons). When the machine is trained well, we tell that B=2 and C=10 and force the machine to make deduction that A=8 through back propagation.  Maybe it does not seem much. But if the input neurons are  “flying away”(Input Neurons) + “a waving hand”(Input Neurons) = “survive”(Target Neurons)  “stay”(Input Neurons) + “a waving hand”(Input Neurons) = “death”(Target Neurons)  We can force the machine to make inference or deduction about what strategy best fits its interest (to survive) when faced with a waving hand.  In a sense, the purpose of the Hinton machine was to force the output neurons to approximate the target neurons by adjusting the synapse. However, this machine takes a reversed direction – that is this machine forces the output neurons to approximate the target neurons by adjusting part of the input neurons while left the whole synapse unchanged and intact. In this process, the machine gains “self-awareness” to maximize its profit in different circumstances either (1) through trial-and-error learning or (2) through pre-set information by human engineer.  But what does it have anything to do with game theory?  Under game theory, every player adjust their own strategy on the basis of the action of other players to maximize its profit. It is upon the same rationale that was described in the mosquito case. Since the machine can imitate the thinking of a mosquito to maximize its profit, why not let it imitate the thinkings of two players?  For example, in the most traditional Simultaneous Game in game theory as illustrated as below:  “simultaneous game”的图片搜索结果Figure 1  If we view the strategies of the two players (column and row) as two different input neurons and the payoffs in Figure 1 as target neurons. When the machine is trained well (either by telling the machine the strategies along with the consequences or by letting the machine to learn by trial-and-error) , we can duplicate the notion of  A(Input Neurons) + B(Input Neurons) = C(Target Neurons)  as stated above, except that this time the Input Neurons A was designed to output (1, 0) and the Input Neurons B was designed to output (0, 1). The exact way of exploiting the well-trained machine is that we let player column to move first (let the Output Neurons of Input Neurons column to approximate Target Neurons (1, 0) through back propagation and gradient descent), and then we let player row to move afterward (let the Output Neurons of Input Neurons row to approximate Target Neurons (0, 1) through back propagation and gradient descent). At least, we let machine repeat the movement above.  In the end, this machine can automatically find Nash Equilibrium in this simultaneous game—that is (T, L) and (M, C).  Wanna see something more interesting? Sure.  The technique as illustrated above is just one combination of Deep Feedforward Neural Network in Artificial Intelligence and Simultaneous Game in Game Theory. However, this technique is not limited to this and can also be exerted in a broader sense such as combining Recurrent Neural Network and Sequential Game in Game Theory. But how does it really work? We will elaborate as below.  In Game Theory, aside from Simultaneous Game, there is one more game called Sequential Game as below:  “sequential game”的图片搜索结果 Figure 2  In this kind of game, each player tries to outsmart other players by a technique call “Back Deduction” which states that each player in a upper layer of a game tree will try to manipulate the players in the lower layer of the game tree by imitating the rationale of the lower players and try to chose the best strategy in the upper layer.  But how can Recurrent Neural Network be combined with Sequential Game?  In Artificial Intelligence, aside from traditional Deep Feedforward Neural Network, there is also Recurrent Neural Network (including LSTM, Neural Turing Machine, etc.). It has its famous long body as below:  Input Neurons  Hidden Neurons  Output Neurons  In the same sense as we elaborated in the combination of Simultaneous Game and DFNN (see Figure 1) which we let player column to move first (let the Output Neurons of Input Neurons column to approximate Target Neurons (1, 0) through back propagation and gradient descent), and then we let player row to move afterward (let the Output Neurons of Input Neurons row to approximate Target Neurons (0, 1) through back propagation and gradient descent). At least, we let machine repeat the movement above and we can find Nash Equilibrium automatically.  Conversely, in the combination of Sequential Game and RNN, we also view each input neurons of RNN as the strategies of each player in each layer in the game tree. Take a view as below:  :  “sequential game”的图片搜索结果  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  First, we randomize the strategies of each player (randomize the input neurons. Don’t worry, they will converge to steady points separately in the end). Then, by Back Propagation, we let the player P2 in the fourth layer to move first (let the Output Neurons of Input Neurons P2 in layer 4 to approximate Target Neurons (0, 1) through back propagation and gradient descent) as illustrated as below:  Target Neurons = (0, 1)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  Then we preserve the strategy (input neurons) of P2 in the fourth layer. And then we let the player P1 in the third layer move consecutively (let the Output Neurons of Input Neurons P1 in layer 3 to approximate Target Neurons (1, 0) through back propagation and gradient descent) as illustrated as below:  Target Neurons = (1, 0)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  The we preserve the strategies (input neurons) of P2 in the fourth layer and P1 in the third layer. And then we let the player P2 in the second layer move consecutively (let the Output Neurons of Input Neurons P2 in layer 2 to approximate Target Neurons (0, 1) through back propagation and gradient descent) as illustrated as below:  Target Neurons = (0, 1)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  In the end, we preserve all the strategies seen so far, and then we let the player P1 in the first layer move consecutively (let the Output Neurons of Input Neurons P1 in layer 1 to approximate Target Neurons (1, 0) through back propagation and gradient descent) as illustrated as below:  Target Neurons = (1, 0)  Input Neurons  Hidden Neurons  Output Neurons  P1 acts  P2 acts  P1 acts  P2 acts  In layer 4  In layer 2  At least, we let the machine repeat all the moves aboe.  In the end, the input neurons of P1 in layer 1 is the best strategy for player 1.  It is a miracle!  In actual application, we can take Tic Tac Toe for example. In the present movement, we can first train the machine with future possible strategies and outcomes for each player. When the machine is trained well, we can further exploit the technique as illustrated above and force the machine to find the optimal strategy for the present player.  For more detail information, please see:  <https://github.com/Brownwang0426/Deep_Neural_Game_Theory>  Interesting enough, there is Back Deduction in Game Theory while there is Back Propagation in Artificial Intelligence. There is Simultaneous Game in Game Theory while there is Deep Feedforward Neural Network in Artificial Intelligence. there is Sequential Game in Game Theory while there is Recurrent Neural Network in Artificial Intelligence.  We can skillfully merge these two giants into one leviathan, exerting the most devastating power.  In the future, this technique can be applied to video games or online games (including real-time strategy games, action games, board games, etc.) to reinforce the intelligence of the monsters or AI opponents in order to conquer human players.  The second usage of this technique is to make prediction about numbers concerning human tactical thinking in a time sequence such as stock market etc..  The third and the most influential usage of this technique to bestow the machine with self-awareness and assume the role as human-policy consultancy to assist human in policy making such as traffic control, criminal deterrence. You can also say it is the prototype of Skynet. | |