2018 年臺灣國際科學展覽會
優勝作品專輯

作品編號　**190010**

參展科別　電腦科學與資訊工程

作品名稱　學習目標行為的機器學習系統——以獎勵回饋去蕪存菁修剪人工神經元

得獎獎項　大會獎：四等獎

就讀學校　國立新竹高級中學

指導教師　李濬屹

作者姓名　郭東穎

關鍵詞　**目標行為(Goal Behavior)、修剪神經元(Dropout Neurons)、強化學習(Reinforcement Learning)**

作者簡介

大家好，我是就讀於新竹高中二年級的郭東穎。科學展覽的機會對我影響深遠，從小學三年級開始，便陸續把自己平常的發現藉由科展機會與各地的同學老師交流，受益良多。一開始的嘗試，後來成為我的興趣。在近幾年中，我以結合生物與資訊為基礎，使用電腦模擬生物界的各種現象，包括魚的群聚行為、合作與背叛者在演化中的族群發展。國中時，獲得出國交流的機會，拓展了我學習的視野，也鼓勵我繼續發現與研究。上了高中，開始接觸機器學習領域，發現機器學習仍有很多不自然之處，與真實的生物學習模式大為不同。這次科展，我便是以生物行為作為啟發，改良資訊上的學習系統。

這個探索，讓我更深入的瞭解到底生物的智慧是什麼？卻也同時打開更多疑問的門。過程中或肯定自己的發現，或質疑自己的解釋，是一場與自己的拉扯，但到頭來還是值得的。未來希望能繼續探索這個驚奇的領域。

Abstract

In this research, rewards directed to dropping out neurons provides a new direction to tackle current obstacles of learning goal behaviors in Reinforcement Learning (RL).

The accuracy and efficiency of five different learning systems are compared, in learning three different goal behaviors. The system using Reinforcement Learning to dropout neurons in a Convolutional Neural Network (CNN) is found to be the most efficient system to achieve the highest score. The concept in such a system is inspired by biological evidence, where animals learn a set of behaviors to achieve their goal, instead of only copying and memorizing individual actions. Therefore, using Reinforcement Learning to directly affect neurons that are responsible for making decisions instead of feeding back to each action, can provide a faster and more direct method to achieve goal learning.

Simulations indicate that current Reinforcement Learning techniques (Deep Q-learning & Prioritized Experience Replay) can only achieve a 19.14% accuracy in goal learning. However, when rewards are fed back to selecting and dropping out irrelevant neurons, a 200% improvement in accuracy, can be achieved, relative to current Reinforcement Learning techniques. Moreover, even with just randomly dropping out neurons, a 100% increase in accuracy can be reached. Apparently, feeding back rewards to dropping out neurons can improve accuracy and lower the time required to identify a goal behavior.

Using rewards that are directed to dropping neurons, this approach provides a new direction for further research on Reinforcement Learning. Applying this technique can achieve generalization in Learning from Demonstration (Imitation Learning).

壹、 摘要

本研究利用回饋機制修剪人工神經元的方式，可在較短時間內達到較高的目標行為學習正確次數。這樣的系統設計源於生物行為的啟發，因為回饋機制能直接影響決策中樞的神經元，而非只是影響動作本身，在達成目標學習上會更直接、快速。以傳統強化學習（Reinforcement Learning）系統為例，本研究模擬結果顯示，若回饋機制影響動作本身，目標達成正確率只有 19.14%。而若獎勵回饋到神經元的修剪上，則相較強化學習提高超過 2 倍的正確率。甚至，隨機修剪神經元也可相較強化學習提高 1 倍的正確率。顯然地，本系統能確實提高正確次數並縮短目標達成時程。利用回饋機制修剪人工神經元，可為強化學習在目標學習上遇到的困境，提供一個新的思考方向，實務應用上，可彌補強化學習在學習行為上無法一般化的缺點。

貳、 研究動機

一、 生物觀察

本研究起始於對生物學習行為的一些有趣觀察，以下列舉幾個觀察實例：

（一） 楔斑豬齒魚

美國演化生物學家 Giacomo Bernardi 發現楔斑豬齒魚（*Choerodon anchorago*）為了達成牠的目標（在不同條件下獲取食物——蛤蠣），成功使用工具，達成敲開蛤蠣殼的目的。甚至為了達成目標，楔斑豬齒魚具有學習如何使用不同工具的能力。牠們不只是重複動作而已，而是有目的性的選擇動作。（Balcombe，2016）

（二） 射水魚

射水魚（*Toxotes*）的觀點取替（perspective taking）行為。新手射水魚觀摩同伴射擊移動目標（包括成功例子及失敗例子）1000 次後，便能成功學習困難技巧，達成目標。更有趣的是，牠們的覓食方法明顯的會根據不同情境具有靈活性，也就是說，牠們可藉由使用不同的行為達到同一目標（Balcombe，2016）。

（三） 蜜蜂

蜜蜂（Bumblebees）學習推球的實驗。這個實驗結果發表於 Science（Loukola, Perry, Coscos, & Chittka，2017）。在這個實驗中，實驗者先示範如何把球推進洞中，以得到一滴糖水獎賞。蜜蜂先重複實驗者示範的行為，最後成功的認知目標，並能使用其他方式，達到目的。例如：選擇推比較近的球入洞，甚至會懂得推與訓練時不同顏色的球，是目標取向，而不只是行為學習，具有靈活的認知能力（cognitive flexibility）。

蜜蜂在人類訓練完推球行為後，並不只是單純複製行為而已，而能有靈活的認知能力，即能選擇最近的球推入洞裡，以最有效率的方式獲得糖水獎賞。

二、 生物決策能量與效率觀點

從生物目標學習的觀點出發，我們發現其歷程並不需要花很久的時間，因為自身的能量限制必須被考慮。

動物在從事每日簡單或複雜的決策過程時，能避免耗時費力的過程（Xie & Padoa-Schioppa，2016）。根據這一份 Nature 的研究，猴子每日面對簡單而不斷重複的決策，如：選擇吃蘋果或橘子時，能在不同的決策情境下，使用同一群神經元指定量值（assign value）給各個選項，在有限的能量下，快速的根據現狀及歷史資料而下決定。此外，一般認為生物的大腦會透過增強及縮減神經連結的平衡來增加學習效率，避免過度強壯的神經連結及記憶干擾。（Vivo, Bellesi, Marshall, Bushong, Ellisman, Tononi & Cirelli，2017）。**本研究的立基便是：只留相關神經元做行為決策，並透過環境的獎勵，成為下次決策的依據。**

三、 啟發

這些生物觀點帶給本研究以下啟發及後續的機器學習系統研究：

（一） 生物的學習是目標式學習（goal-based learning），也就是為了達成一個目的而學到一套的行為。

（二） 能夠達成目的的行為不只一種，生物在達成目的之前必須根據行為結果是否有效，而調整自己的行為。

（三） 生物可以藉由一套簡單的原則（為達目的）而「去蕪存菁」、「舉一反三」，發展出一套有效省時節能的目標行為。

本研究即是要透過擷取生物目標學習的行為特徵，發展出一個學習系統，使機器人在學習某一行為後，能有效根據獎勵應變、調整行為以達成目標行為。

四、 文獻探討

於此，我們先介紹目前控制機器人、學習行為的方法。

（一） 手動控制（Biggs & MacDonald，2003）

這是最常使用控制機器人的方法，即以固定不變的條件邏輯（linear conditional logic）來控制機器人。雖然此方法在工業上已長期被使用（如德國 KUKA 與瑞士 ABB），但這種方法存在兩大缺點：

1. 缺乏靈活性

機器人只能做某一特定的行為，如果要改變其行為的任何部分，則必須重新撰寫程式。此外，當環境改變時，機器人也無從適應。

2. 只適用於簡單行為

當行為變得複雜，或者機器人的輸入（如：感應器）輸出（如：馬達）眾多時，並不容易撰寫程式使之學習一個行為，尤其光要使機器人協調各個馬達就必須牽涉到各種複雜的機器人控制系統。

（二） Fido（Gruenstein & Truell，2016）

Fido 使用強化學習（Reinforcement Learning）的機器學習模式，大大簡化了控制機器人的方法。Fido 藉由人給予機器人獎勵（numerical rewards）的方式，告訴機器人目前所做的行為與目標行為的相似程度，使得機器人能夠越來越近似目標行為，達成控制機器人的效果。利用強化學習經由獎勵而控制機器人行為，是一大突破。強化學習系統也讓機器人有探索新行為（exploring）的可能。

但是在實際的操作上 Fido 也有許多限制。例如：Fido 的控制方法只能使得機器人「記憶」某一行為，即使有更有效率的行為能夠達成目標，機器人也只會把所記憶的行為一個動作一個動作的照樣做出來，與許多生物行為不一樣。另外，機器人必須探索到符合的目標行為，控制者才得給予獎勵，但不一定在所有情況之下機器人皆能自己產生目標行為，此時機器人無法在短時間內學習到目標行為。

（三） 強化學習（Reinforcement Learning）領域

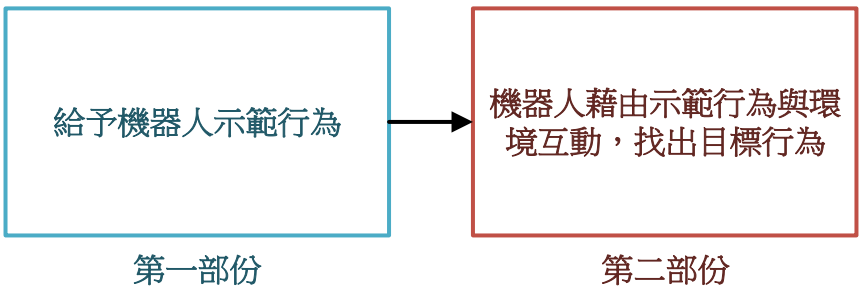
近日在強化學習領域當中，有許多學者投入在 Learning from Demonstration（又稱 Imitation Learning）當中，都是將強化學習應用在目標行為學習上。雖然理論上，強化學習最終一定能找出目標行為，但是實際上非常耗時，如果完全依賴探索，不能保證能在合理時間範圍內找到目標行為。這是之所以為何必須仰賴示範行為（Hester, Vecerik, Pietquin, Lanctot, Schaul, Piot, ... & Leibo，2017）（Večerík, Hester, Scholz, Wang, Pietquin, Piot, ...

& Riedmiller，2017）以及 Prioritized Experience Replay（Schaul, Quan, Antonoglou & Silver，2015）等技術，這些方法皆是為了縮短訓練時程或提高學習效率而發展出來的。如同本研究也是在解決強化學習的效率問題所發展出來的學習系統。

本研究的目的是如何有效設計一個仿生機器學習系統，使機器人在學習某一行為後，能夠彈性的改變行為，以有效率的方式達成行為背後的目標。不過，考慮「有效率」牽涉許多其他系統，例如：空間認知（葉暘、何政勳，2017）、時間認知、對自己動作的認知等等，在本研究中要凸顯的是增加成功達到目標的次數（或總獎勵），學到所謂的「目標行為」。

參、研究目的

本研究希望機器人系統能解決的問題如下：



圖（一）：研究問題示意圖

因此，本研究想具體回答下列問題：

一、 怎樣的資料結構適合儲存示範行為？是否上述資料結構能使機器人學習任何示範者給予的其他示範行為？

二、 是否可以透過修剪人工神經元（drop neurons）使得神經網路可以去蕪存菁，更快找到與目標行為相關的特徵（features），例如：顏色、形狀？

三、 如何有效率的修剪人工神經元使得學習系統能在比較短的時間內學到修剪人工神經元的最佳方法，進而找到與目標行為相關的特徵？

四、 在強化學習的系統裡，是否也可以透過修剪人工神經元去蕪存菁，找到與目標行為相關的特徵，增加達成目標行為的正確次數？

五、 比較現行常用的強化學習系統是否能更有效率的學習目標行為？

肆、 研究設備及器材

研究設備及器材如下：

一、 電腦

二、 V-REP 機器人模擬軟體

三、 Python＋Keras 機器學習函式庫（後台使用 TensorFlow）

四、 OpenCV 電腦視覺函式庫

五、 NXT＋Raspberry Pi 2

伍、 研究過程和方法

本章節分四部分說明：

一、名詞說明、定義

二、研究方法一：運用人工神經網路

三、研究方法二：運用強化學習

四、研究流程

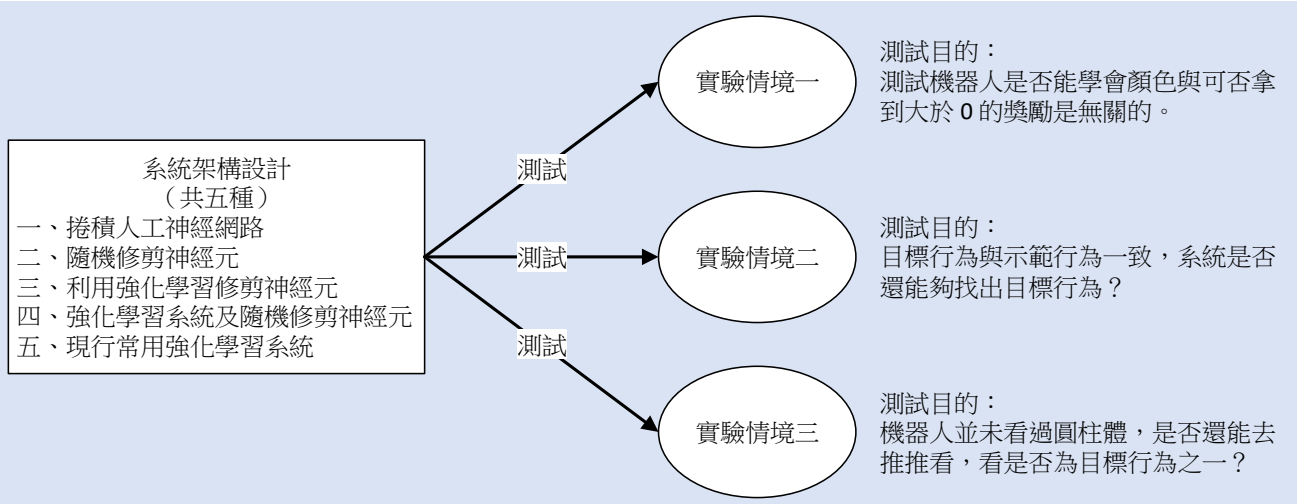
測試情境設計：

1. 測試情境一：測試機器人是否能學會顏色與可否拿到大於 0 的獎勵是無關的。
2. 測試情境二：目標行為與示範行為一致，系統是否還能找出目標行為？
3. 測試情境三：機器人未看過圓柱體，是否能去推推看，看是否為目標行為之一？

本研究測試了以下五種學習系統設計，各系統會在研究結果中做說明：

1. 捲積人工神經網路
2. 隨機修剪神經元
3. 利用強化學習修剪神經元
4. 強化學習系統及隨機修剪神經元
5. 現行常用強化學習系統

本研究測試系統的流程如圖（二）所示。



圖（二）：研究流程示意圖

一、 名詞定義

以下說明本研究中何謂「環境」、「行為」、「目標」、「示範行為」、「目標行為」。

（一） 環境

本研究將環境建構為一馬可夫決策過程（Markov Decision Process，MDP）以便發展儲存行為的方法。

環境是相對於機器人的控制系統而言，機器人所無法控制的元素，包括機器人身上的感應器也可視為環境的一部份。而馬可夫決策過程常用以正式建構機器人所在的環境模型。一個可建構成 MDP 的環境，每一個狀態具有馬可夫特性（Markov property），即下一個時間點的狀態只依據當前時間點的狀態而定。這個性質可以大大簡化建構環境模型的難易度。具體而言，建構一個 MDP 需包含五個元素(S, A, P, R, γ)：

1. S 是狀態的有限集合，包含所有機器人可能走訪的狀態。狀態可包括各種資料輸入，也可以是處理過的資料。以本研究為例，機器人的狀態為機器人所看到的影像。

2. A 是動作的有限集合，包含所有機器人可以行使的動作。

3. P 是狀態轉移機率矩陣（state transition probability matrix）：

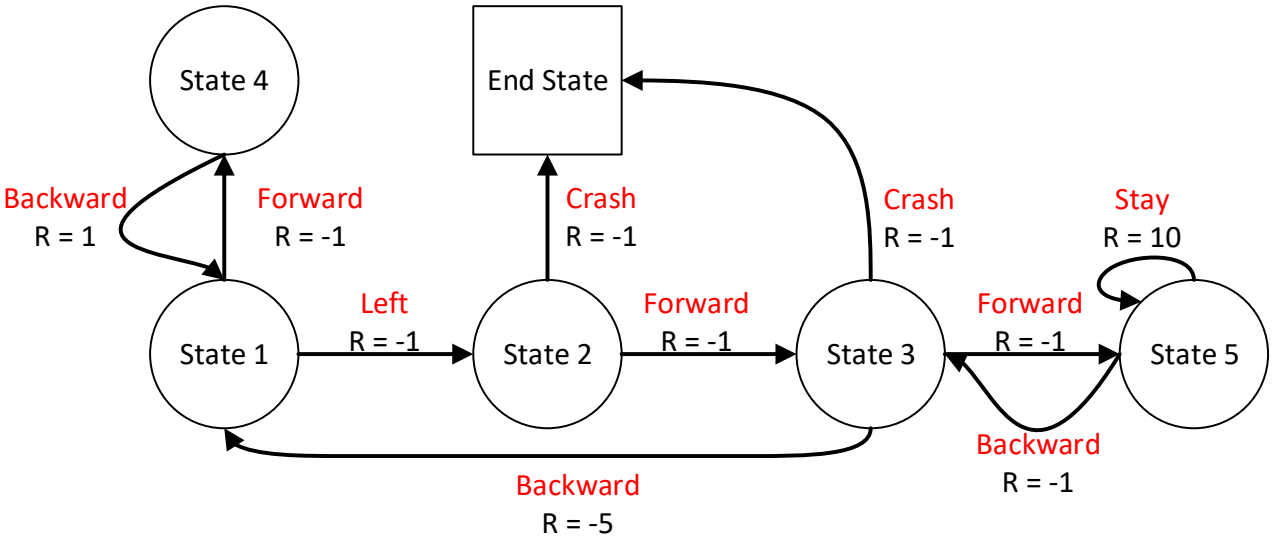
$$P_{ss'}^{a} = \mathrm{P}(S_{t+1} = s' \mid S_t = s, A_t = a)$$

亦即完成動作 a 的前提下，由狀態 s 轉移到 s' 的機率。

4. R 是獎勵函數：

$$R_s^a = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$$

5. γ 是折扣係數（discount factor），代表未來獎勵及當下獲得獎勵的重要性差異。



圖（三）：範例 MDP。

（二） 行為

本研究借用強化學習對於政策（policy）的定義。

當環境可以被視為一個 MDP 時，則因為由當下的狀態轉移到下一個狀態，僅由當下的狀態決定，所以機器人的行為可以定義為特定狀態下，行使各個動作的機率分布：

$$\pi(a|s) = P(A_t = a|S_t = s)$$

（三） 目標（goal）

本研究亦使用強化學習對於目標的定義。

根據獎勵假說（reward hypothesis），所有目標、意圖皆可視為最大化（或最小化）一個數值訊號（通常稱為獎勵，reward）的總和（Sutton，1998）。以研究動機中的蜜蜂實驗為例，「糖水」即為獎勵，而目標就是「蒐集越多糖水越好」。

（四） 示範行為（demonstration behavior）

所謂示範行為是一開始示範者教給機器人目標行為的一個子集（subset）。示範行為中，機器人必會得到（大於 0）的獎勵。

（五） 目標行為（goal behavior）

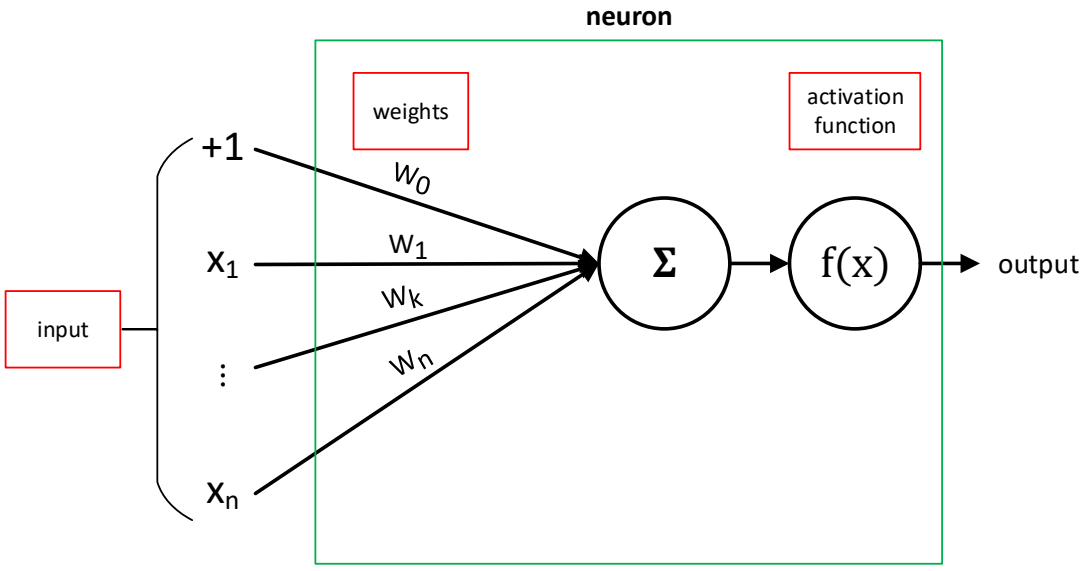
在本研究定義為所有能達到目標的行為，也就是說依循著「目標行為」，則能夠得到最多獎勵。

本研究目的即藉由某一示範機器人的行為中，嘗試找出目標行為。這是由於若依循示範的行為，不一定能最大化獎勵。

二、 研究方法一：運用人工神經網路（Artificial Neural Network，ANN）

人腦以及多數動物大腦由數億個神經元所構成，而各個神經元之間透過電與化學訊號互相溝通，因此神經元可視為生物智慧（intelligence）的基本單位。眾多神經元聯合成網路，形成生物智慧來源的構造。人工神經網路模擬了生物神經網路的數學性質，藉以近似各種數學函數。

（一） 人工神經元



圖（四）：單一神經元模型示意圖

上圖表示單一神經元的模型。人工神經元為人工神經網路中的一個節點，與生物的神經元性質相似。每一個人工神經元擁有眾多個輸入、對每個輸入有一個相對應的權重值（weight）、一個偏差值（bias）、一個激活函數（activation function）以及一個輸出（output）。以數學形式來說明，對於 n 個輸入 $x_1 \dots x_n$ 與權重 $w_1 \dots w_n$、一個偏差值 w_0、激活函數 $f(x)$，則此神經元的輸出為：

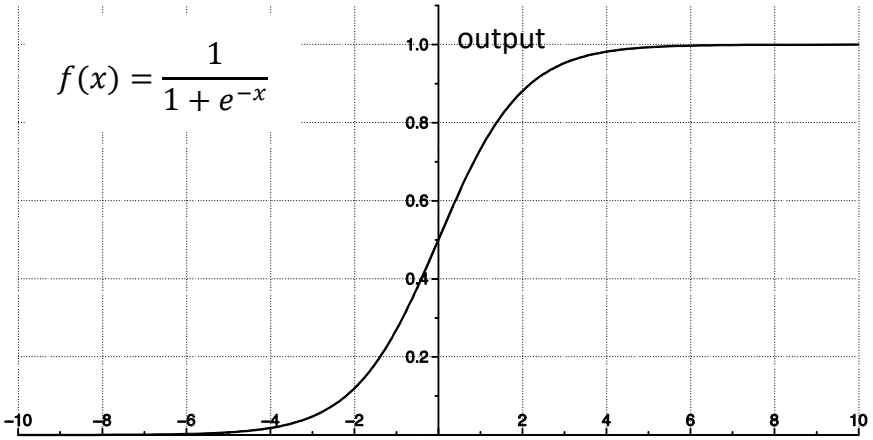
$$\text{output} = f\left(\sum_{i=1}^{n} x_i w_i + w_0\right)$$

注意到習慣上我們常令 $x_0 = 1$，則上式可簡化為：

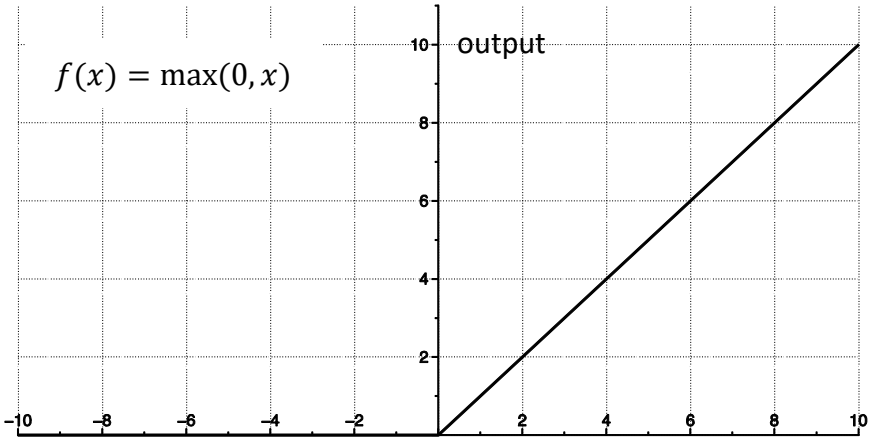
$$\text{output} = f(\sum_{i=0}^{n} x_i w_i)$$

常見的激活函數包括 Sigmoid 函數以及 Rectifier 函數。

在本研究中使用 Rectifier 函數是取其運算速度快，常用於機器學習影像辨識的應用上，另外在前置實驗中（P.43）其正確率較高。



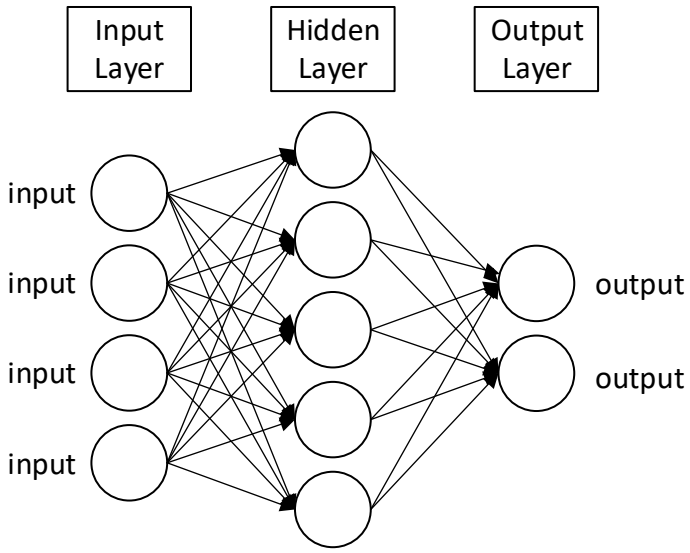
圖（五）：Sigmoid 激活函數。



圖（六）：Rectifier 激活函數——本研究所使用的激活函數。

（二） 網路模型

一層神經網路由多個神經元構成，而神經網路則是由多層構成，每一層的神經元輸出為下一層神經元的輸入。最後一層神經元的輸出為整個神經網路的輸出。



圖（七）：神經網路模型

（三） 學習權重值

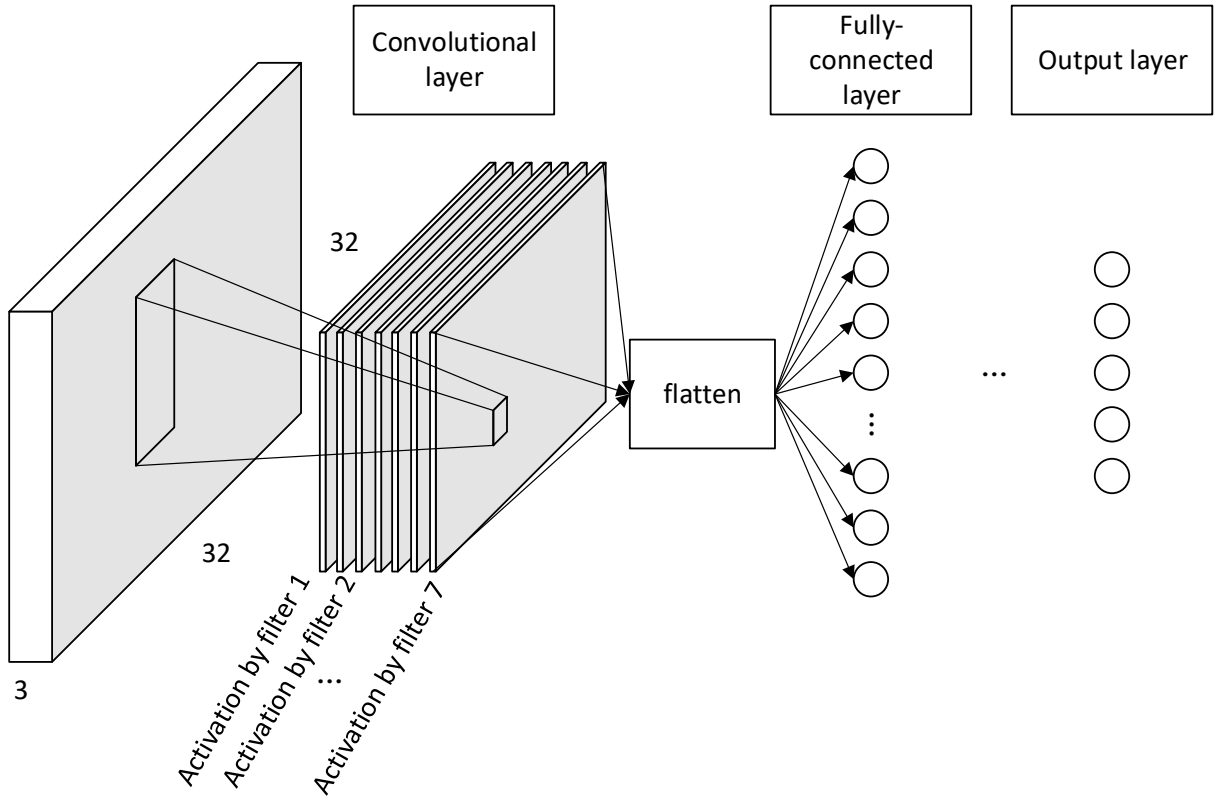
有了神經網路之後，所謂訓練（train）一個神經網路是調整神經網路當中的各個權重值，使得神經網路的輸入與輸出能夠近似訓練資料。有很多訓練的演算法，其基本原理的共通點是將輸出層與訓練資料的誤差值回饋於前一層的權重值當中（稱為 backpropagation）。

在本研究中我們使用了 Adam（Kingma & Ba，2014）演算法，取其訓練快速，在前置實驗當中配合 Rectifier 函數其正確率較高（P.43）。

本研究將權重值視為對於輸入資料的某種數據轉換，而激活函數蒐集這些轉換後的資料做判斷，如果大於門檻值（threshold value），則表示原始資料擁有某一特徵，人工神經元將此訊息傳遞給下一層的神經元。

而學習權重值即在學習原始資料的轉換方式，使得足夠多（或足夠少）的人工神經元激發。因此，當我們將某人工神經元的輸出壓抑（suppress），指的是將激活函數設為 $f(x) = 0$ 或將神經元所有連出的權重值設為 0（因為負值有告訴下一層神經元「缺乏某特徵」的意味），則相當於此神經網路看不見（blinded）某特徵，或對於某特徵不靈敏。

（四） 捲積神經網路（Convolutional Neural Networks，CNN）



圖（八）：捲積人工神經網路示意圖

本研究使用了捲積神經網路使機器人辨別形狀差異。

如果人工神經網路的輸入為一張圖片時，由於圖片的畫素與畫素之間擁有空間上相鄰相關的性質，我們可以藉此簡化計算並得出更妥當的特徵。捲積層（convolutional layer）含有多種可學習的濾鏡（filter），分別對於前一層影像的一小區域做如同普通人工神經網路的線性轉換，再把結果輸入激活函數成為下一層的輸入影像，相當於在凸顯輸入影像的某種特徵（feature），例如：顏色塊（color blobs）、曲線。

捲積人工神經網路所學到的濾鏡權重，為本研究使機器人辨別圓球與立方體（形狀）差異的主要方式，並無使用其他系統告訴機器人眼前為何物。

在本研究中，對於捲積層的「修剪神經元」，我們採取將濾鏡矩陣上的某些數值設為 0，而不是將整個濾鏡陣列設為 0。這樣做是為了讓學習系統有自由選擇開啟或關閉對於某一影像深度（depth，如 R、G、B channels）的特徵辨識，而不是直接關閉整個濾鏡。

（五） Dropout（Srivastava, Hinton, Krizhevsky, Sutskever & Salakhutdinov，2014）

另一個與本研究相關的方法為 Dropout。Dropout 透過修剪人工神經元，使得人工神經網路較不會對訓練資料過度擬合（overfit），增加人工神經網路對於未知測試資料的正確率。

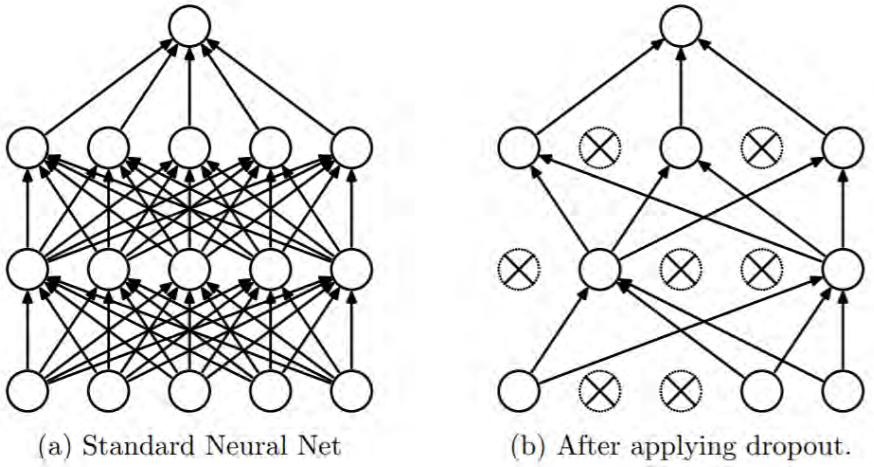
Dropout 系統作用於訓練人工神經網路之時，在訓練每筆訓練資料（training case）時，使每個神經元有 p 的機率被修剪。而在測試時，Dropout 將神經元的所有權重值乘以 p。這樣的效果相當於系統同時訓練 2^N 個簡化版（thinned）的人工神經網路，並在測試時取其輸出的平均。

在本研究中修剪神經元的作法與 Dropout 不同如下：

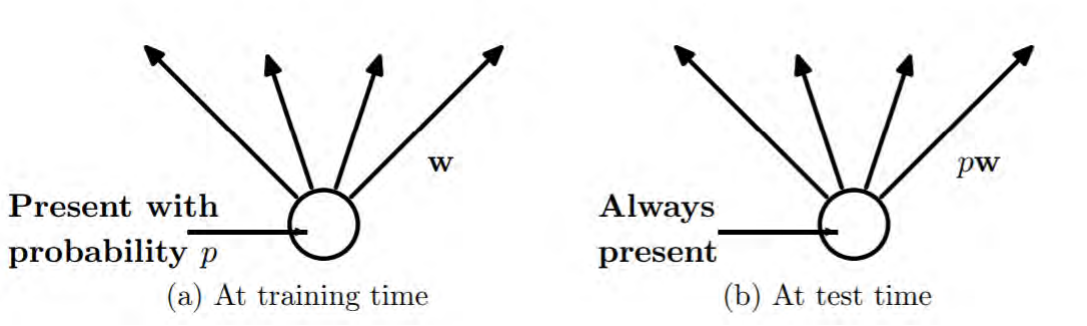
1. Dropout 將修剪神經元看成是訓練多個簡化版的人工神經網路，而**本研究中修剪神經元可看成是使得人工神經網路對於某特徵的不靈敏，讓機器人能去嘗試不同的行為。**
2. Dropout 修剪神經元的目的是避免神經網路訓練時過度擬合，而**本研究中是在人工神經網路已經訓練好的狀態之下，於測試階段（testing）修剪神經元。**
3. Dropout 只做隨機修剪神經元，而**本研究中嘗試「有系統地」修剪神經元。**

DropConnect（Wan, Zeiler, Zhang, Cun, & Fergus，2013）則延續了 Dropout 的精神，藉由隨機修剪權重值來避免訓練資料的過度擬合。

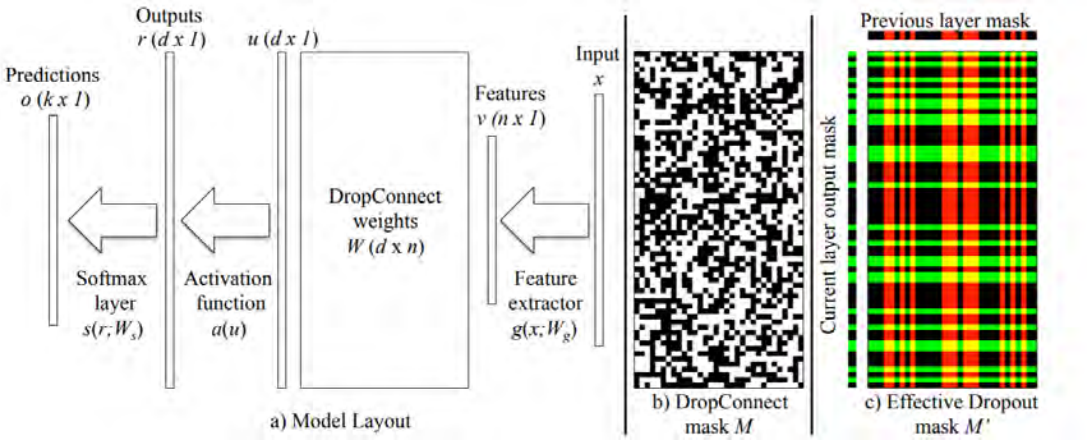
在本研究中嘗試修剪權重值，發現對於小的人工神經網路，修剪權重值尚可適用，但對於大的人工神經網路，由於權重值過多，狀態數、動作數多而使得強化學習的學習速度緩慢。雖然這個作法較符合生物上的意義，但是在實際操作上修剪神經元較為有效（見討論）。



圖（九）：Dropout 系統示意圖。左為標準的人工神經網路，右為 dropout 過後可能產生的人工神經網路。（Srivastava et al.，2014，Figure 1）



圖（十）：Dropout 修剪神經元示意圖。當訓練時，單獨神經元有 p 的機率被修剪，而測試時將神經元的所有權重值乘以 p。（Srivastava et al.，2014，Figure 2）



圖（十一）：DropConnect 系統示意圖。注意 b) 與 c) 比較了 Dropout 與 DropConnect 的差異，DropConnect 所修剪的權重較無結構（lack of structure）。（Wan et al.，2013，Figure 1）

三、 研究方法二：運用強化學習（Reinforcement Learning，RL）

本研究結合另外一個重要的學習系統為強化學習。

強化學習是機器學習的一個領域，強調如何藉由嘗試（trial and error），以找尋能獲得最多獎勵（reward）的最佳行為（或政策，policy）。它與其他機器學習領域（例如：監督式學習（Supervised Learning）、非監督式學習（Unsupervised Learning））最大的不同是訓練資料（training data）並沒有標籤（label），取而代之的是獎勵訊號（reward signal）。此外，機器人在某一時刻的動作，亦會影響未來收到的資料。

（一） Q-Learning、Deep Q-network（DQN）

本研究在強化學習領域的利用是 Q-Learning 及 Deep Q-network。

$q^{\pi}(s,a)$ 函數的值為機器人在狀態 s、行為 π 之下，行使動作 a 的獎勵期望值，而機器人可以依此函數找出更接近目標行為的 π。$Q(s,a)$ 函數則在近似 $q^{\pi}(s,a)$，是機器人自己認為 $q^{\pi}(s,a)$ 應具有何值。Q-Learning 演算法，則藉由以下數學式不斷更新 $Q(s,a)$ 函數，同時讓 $Q(s,a)$ 函數更接近 $q^{\pi}(s,a)$，也同時更新 π，使之越來越接近目標行為。

$$\mathrm{Q}(s_t,a_t) \leftarrow \mathrm{Q}(s_t,a_t) + \alpha\left(r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)\right)$$

其中 s_t 為機器人在時刻 t 的狀態；a_t 為機器人在時刻 t 行使的動作；r_{t+1} 為機器人做了 a_t 動作之後得到的獎勵；α 為學習係數，可增減此更新式改變 $Q(s,a)$ 的程度；γ 為折扣係數，說明了對未來的不確定性。

傳統上使用表格來儲存 $Q(s,a)$ 函數值，但隨著問題複雜度增加，狀態數與動作數增加至無法使用表格儲存，近年來常使用近似函數（function approximators）來代表 $Q(s,a)$ 的值，尤以人工神經網路最為普遍，如 Deep Q-network（Mnih, Kavukcuoglu, Silver, Rusu, Veness, Bellemare, ... & Petersen，2015）。另外一個使用近似函數的好處是可以推估還未走訪過的 $Q(s,a)$ 值，使訓練速度增加。

在本研究中，由於狀態數偏多，在所有強化學習的應用皆使用了 Q-Learning＋Deep Q-network。

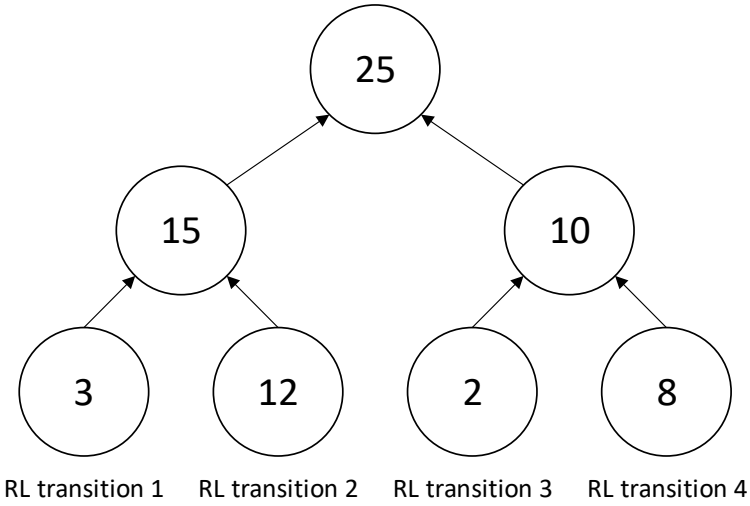
（二） Prioritized Experience Replay（PER）

在 DQN 中，為了穩定用以近似 $Q(s,a)$ 的人工神經網路，必須不定時地拿舊經驗再次訓練人工神經網路，稱為 experience replay（Mnih et al.，2015）。

但是在本研究的測試情境，獎勵是稀少（sparse）的，如果同等對待所有的經驗會使得訓練時間指數增加。因此我們採用 proportional prioritized experience replay（Schaul et al.，2015），優先訓練人工神經網路較特別的經驗，而具體來說使 TD-error 較大者有較大機率被訓練到。本研究中利用 Sum Tree 結構加速算機率的時間，且取 $\alpha = 0.6$。

$$TD\ \ error(\delta) = r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)$$

$$probability\ \ of\ \ RL\ \ transition\ \ i\ \ chosen\ \ to\ \ experience\ \ replay(P_i) = \frac{(TD\ \ error\ \ of\ \ transition\ \ i)^{\alpha}}{\sum_j (TD\ \ error\ \ of\ \ transition\ \ j)^{\alpha}}$$



圖（十二）：Sum Tree 示意圖。子葉為各個 RL transition 的 $(TD\ \ error)^{\alpha}$ 值，每個節點為其左右子樹之和，而樹根為所有 $(TD\ \ error)^{\alpha}$ 之總和。取 RL transition 時，隨機由 0 到樹根之值選一數，對應其為哪一個 RL transition。以此樹為例，如果隨機取的數為介於 $[0, 3]$ 則取第一個 RL transition；介於 $(3, 12]$，則取第二個 RL transition，依此類推。

四、 研究流程：測試情境設計

（一） 測試情境一

測試目的	示範行為	目標行為	測試環境	獎勵方式
1. 模仿蜜蜂的實驗 2. 測試機器人是否能學會：顏色與是否可以拿到大於 0 的獎勵是無關的。	推紅球 避紅立方體	推任意顏色球 避任意顏色立方體	一個圓球 一個立方體	推下圓球：+1 推下立方體或自己掉下平台：-1 無動作：-0.5

1. 測試目的

這個環境設計主要是模仿蜜蜂的實驗（Loukola et al.，2017）。在蜜蜂的實驗當中，研究人員在一隻蜜蜂眼前控制假蜜蜂，讓假蜜蜂把一顆黃球推進洞裡，再給予蜜蜂糖水吃。蜜蜂藉由觀察假蜜蜂的行為，進一步學到推任何顏色球進洞的行為。

2. 環境

基於模擬環境不易處理有負體積的物體（挖洞），以及為了簡化實驗配置，我們定義一回合（episode）為讓機器人在一平台的中心位置旋轉，直到機器人看到它認為推下平台可以獲得獎勵的物品，接著往前走，把物品推到平台下，啟動平台下的感應器。

3. 獎勵方式

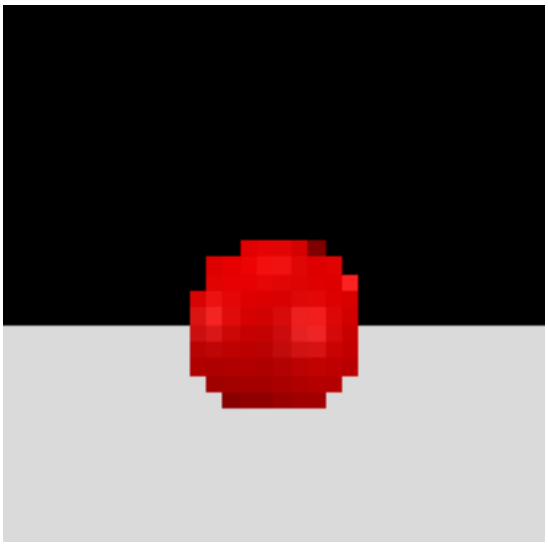
示範者對機器人示範推紅球、看到紅立方體繼續左轉的行為。我們希望機器人最終能學到的目標行為是能去推任意顏色的球、看到任意顏色立方體不去推。在一回合當中，如果機器人最終成功把球推到平台下，則給予獎勵 1 單位。如果機器人將立方體推到平台下，或自己掉到平台下，則給予獎勵-1 單位。如果機器人在時間限制內並未把任何物品推到平台下，則給予獎勵-0.5 單位。給予獎勵的方式只看最終結果，在回合執行其間並未給予任何獎勵（或說給予獎勵皆為 0），因此這是個獎勵稀少的環境（sparse-reward environment）。所謂獎勵稀疏，就是機器人在做完一個動作之後並沒有獎勵回饋，要等待一段時間（通常在完成最終任務時）才有獎勵。例子如：插桿子入洞，只有在桿子入洞時才有獎勵，其餘時刻的獎勵為 0（Večerík et al.，2017）。

4. 示範行為

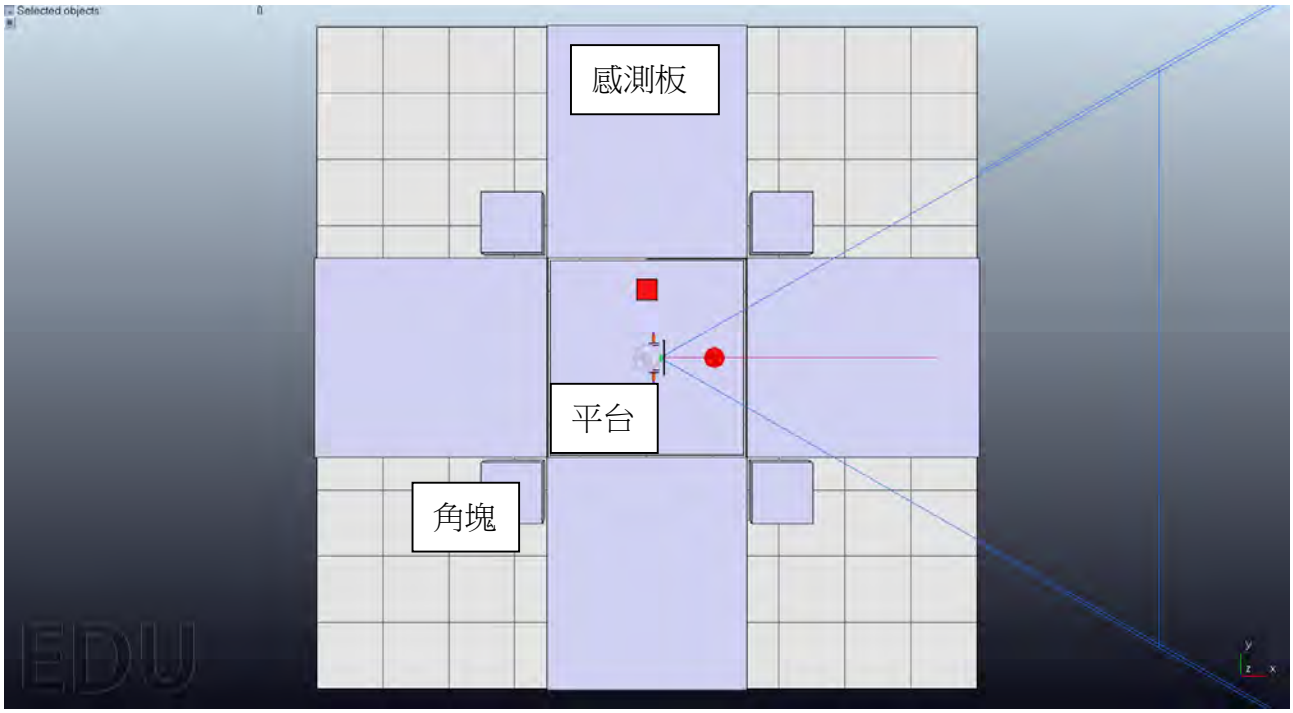
示範行為中，我們在機器人活動範圍的四個邊，挑選兩不同邊分別放一顆紅球與一個紅色立方體。我們每一種放法皆示範一次，即總共示範了 12 個回合。我們給予機器人的示範資料中包括機器人的攝影機在某時刻所看到的影像，以及在這個時刻應當行使（前進、後退、左轉、右轉、停止）的動作。

5. 目標行為

在這個實驗當中，我們想測試機器人是否能由示範行為中「去蕪存菁」，找到真正與目標行為有關的特徵。以此為例，機器人必須學會顏色與是否可以拿到大於 0 的獎勵是無關的。

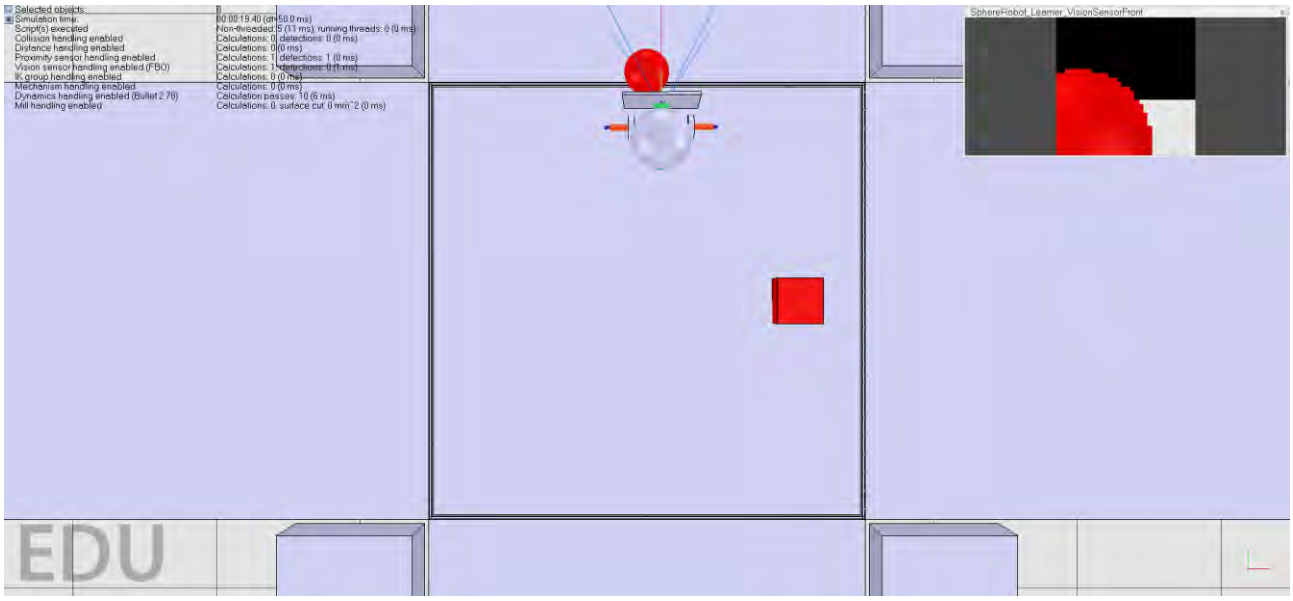


圖（十三）：機器人所看到的 32x32 畫素影像，其中輸入至神經網路的影像為一 32x32x4 陣列，數介於 0 與 1 之間，分別為各個畫素之 R、G、B、灰階亮度值。

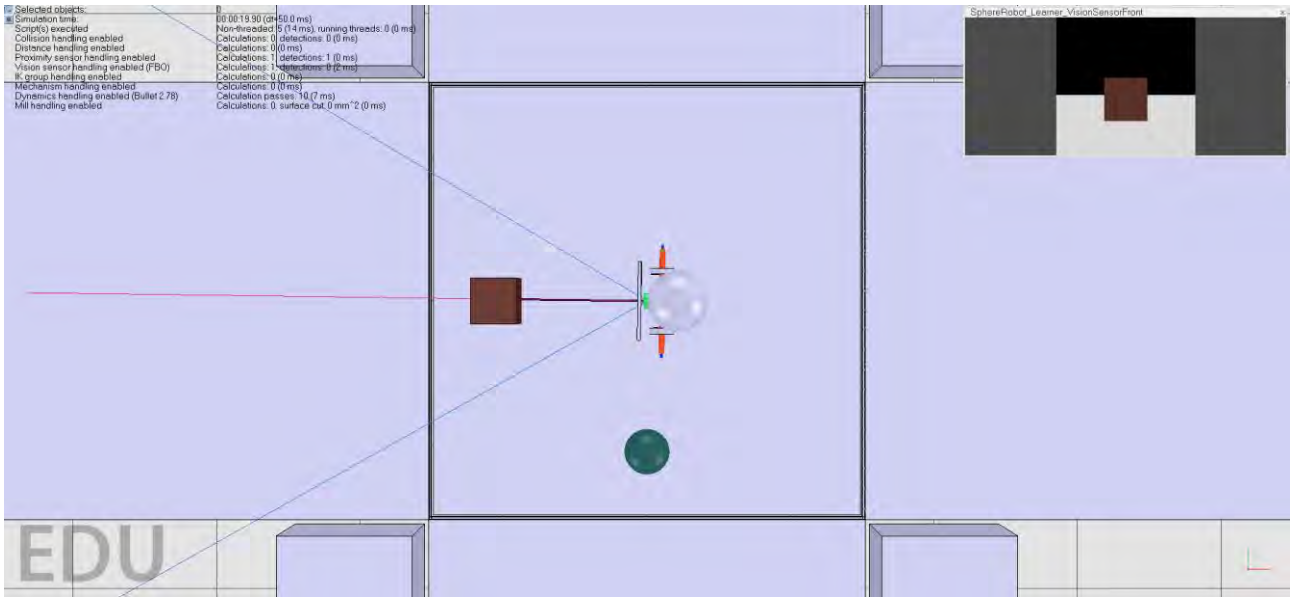


圖（十四）測試情境一、二、三之設置。

感測板：如果曾有任何物品掉至上面會啟動感測器。
角塊：為了避免機器人將物品推至感測版之外，造成感測版無法正常辨識。
平台：略高於感測板，為機器人活動範圍。其中感測版、角塊是機器人看不到的。機器人前的紅色線為行使示範行為時判斷前方物品的感應器，而藍色線條是機器人上攝影機的可觀察區域。



圖（十五）：示範行為中，告訴機器人在看到紅球時要往前走。



圖（十六）：一回合中，立方體及圓球的顏色是隨機選取的。右上為機器人眼中的立方體。

（二） 測試情境二

測試目的	示範行為	目標行為	測試環境	獎勵方式
目標行為與示範行為一致，系統是否還能夠找出目標行為？	推紅球 避紅立方體	推紅球 避立方體	一個圓球 一個立方體	推下紅球：+1 沒有紅球，而無動作：+0.5 推下立方體、他色圓球或自己掉下平台：-1 有紅球，而無動作：-0.5

1. 測試目的

如果目標行為與示範行為一致，系統是否還能夠找出目標行為？這個環境設計主要是實驗一的延伸，目的是為了瞭解系統是否能靈活地找出正確的目標行為。我們示範給機器人的行為與實驗一相同，不同之處只在於環境物體的配置以及給予獎勵的情形。

2. 測試環境

在一回合當中，有 50%的機率環境會放置紅球，有 50%的機率球是其他顏色。物體放置的位置也是隨機的。

3. 獎勵方式

如果機器人最終成功把球推到平台下，則給予獎勵 1 單位。如果環境中沒有紅球，而機器人沒有動作，則給予獎勵 0.5 單位。如果機器人將立方體推到平台下，或自己掉到平台下，則給予獎勵-1 單位。如果機器人在時間限制內並未把任何物品推到平台下，則給予獎勵-0.5 單位。

（三） 測試情境三

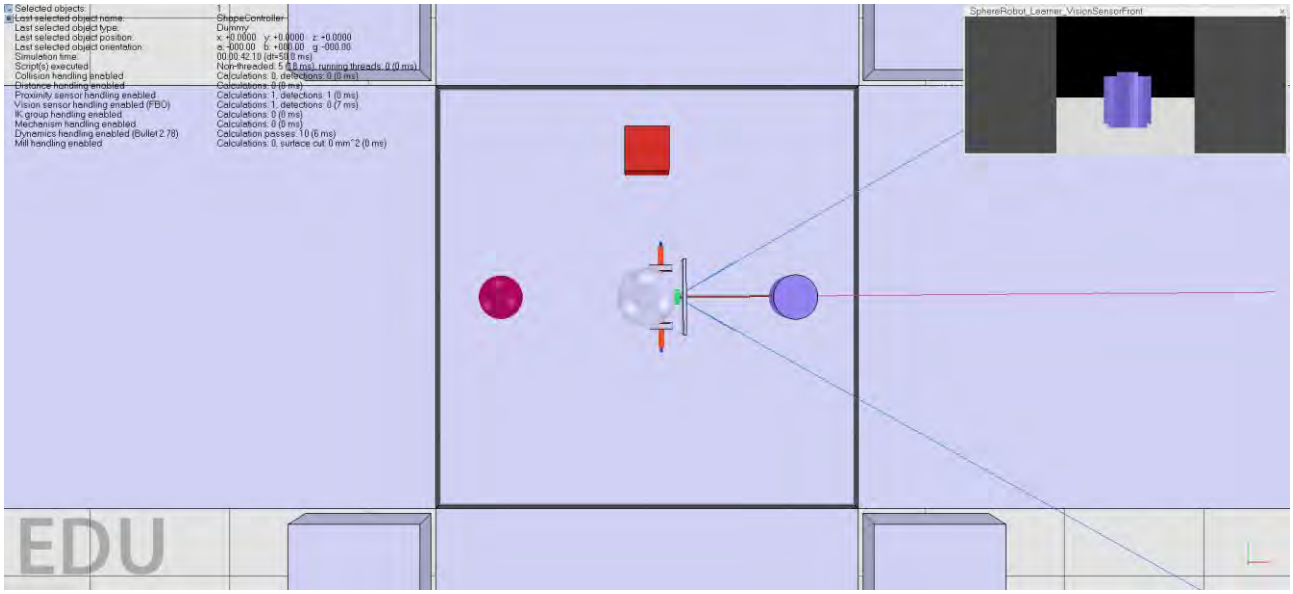
測試目的	示範行為	目標行為	測試環境	獎勵方式
機器人並未看過圓柱體，是否還能去推推看，看是否為目標行為之一？	推紅球 避紅立方體	推任意顏色球 避任意顏色立方體及任意顏色圓柱體	一個圓球 一個立方體 一個圓柱體	推下圓球：+1 推下其他物體或自己掉下平台：-1 無動作：-0.5

1. 測試目的

這個環境設計主要在考驗機器人對於新的物品是否仍能找出目標行為。即使機器人並未看過圓柱體，是否還能去推推看，看是否為目標行為之一？

2. 獎勵方式

在一回合當中，物體放置的位置也是隨機的。如果機器人最終成功把球推到平台下，則給予獎勵 1 單位。如果機器人將立方體、圓柱體推到平台下或自己掉到平台下，則給予獎勵-1 單位。如果機器人在時間限制內並未把任何物品推到平台下，則給予獎勵-0.5 單位。



圖（十七）：測試情境三。圖的右上角可看到機器人眼中的圓柱體。

陸、 研究結果

本研究結果的呈現方式是依照研究目的所提出的五點來回答，並呈現五個學習系統架構在不同測試情境下的表現。

一、 尋找適合儲存行為的資料結構

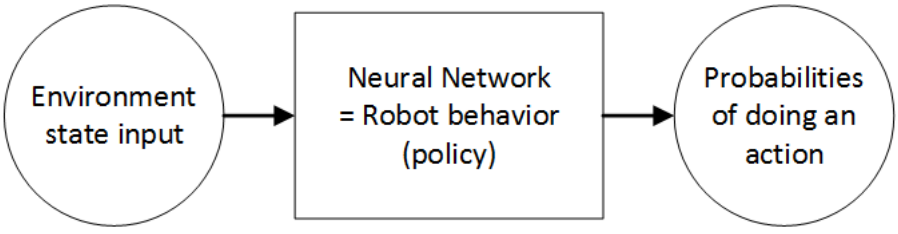
機器人行為的儲存方式必須能讓機器人自行更改行為，以方便讓電腦學習目標行為。最簡單、傳統的方式是直接撰寫程式，但是這樣的方法會使得改變行為不容易，因為如果要對於示範行為做任何改變，就需要人工重新撰寫程式。尤其在示範行為非常複雜或機器人的結構複雜時，直接撰寫程式並不容易。我們希望示範行為的資料只包括目前機器人的狀態，以及在此狀態之下，機器人應該行使的動作，因為這些資料比較容易取得。

因此本研究使用以下兩方式解決這個問題：

（一）神經網路

（二）強化學習。

（一） **神經網路**（參見系統架構一，P.25 的說明）



圖（十八）：行為是狀態（environment state）的函數，輸出是接下來行使每個動作的機率。

這種儲存行為的方法，好處是容易訓練，只要實際對機器人操作，機器人就能根據操作時感應到的輸入（環境），對應一個輸出。從研究方法的討論當中，機器人的行為是給定某一狀態下，行使動作的機率函數，而神經網路可充當此機率函數。在系統架構一、二、三中，我們使用此種方式儲存示範行為。

（二） 強化學習（Reinforcement Learning）

如果以強化學習的模式儲存，機器人的行為則由 $q^{\pi}(s,a)$ 決定。這個函數告訴機器人在狀態 s 之下，行使動作 a 的獎勵期望值。如果機器人在任何時間點 t，狀態 s_t 下都執行 $q^{\pi}(s_t,a)$ 值最大的動作 a，則預期機器人可以得到最多的獎勵。

而此函數可以利用 Deep Q-Learning 演算法建構。特別的是，Deep Q-Learning 需要將示範行為分解成許多的強化學習轉移（RL Transitions）(s, a, r, s')。茲列舉如下：

1. s：當前機器人的狀態。
2. a：機器人在此狀態下做的動作。
3. r：機器人做完這個動作後所獲得的獎勵。
4. s'：機器人做了動作 a 之後，轉移到的新狀態。

二、 透過隨機修剪人工神經元（drop neurons）找到與目標行為相關的特徵（features）

系統架構一：對照組，捲積人工神經網路，沒有修剪神經元
系統架構二：實驗組，捲積人工神經網路，隨機修剪神經元

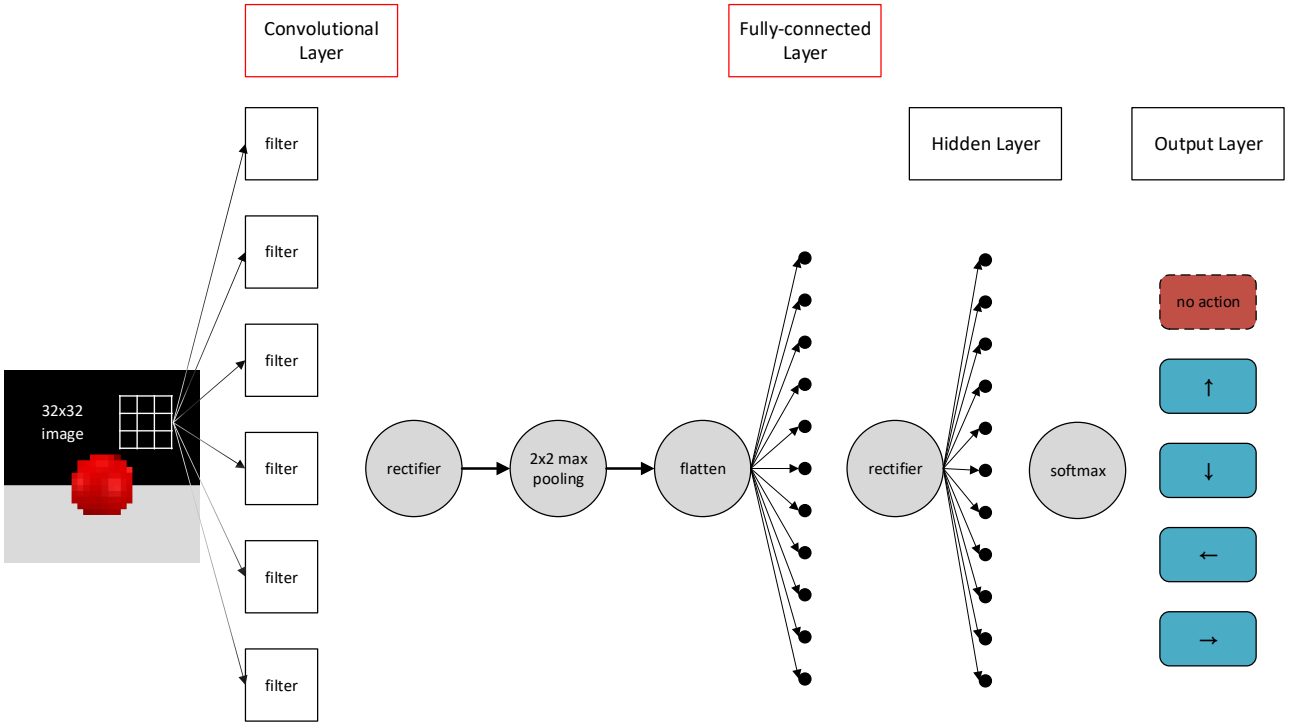
如果只依靠強化學習，理論上機器人最終皆可學到目標行為。但是在本研究的實驗環境中獎勵是稀少的，如果機器人只透過不斷地嘗試以學習目標行為，一回合中能得到獎勵的機率微乎其微，在沒有獲得獎勵的情況下，機器人無從學習在某個狀態下做什麼動作能獲得最大的獎勵，而使得訓練時間隨任務的難易度成指數增加（Schaul et al.，2015）。

因此本研究提出利用修剪人工神經元的方式，來使得機器人學會在示範行為中真正與目標行為相關的特徵，以縮短機器人學習目標行為的時間。這樣處理有其生物背景，而此依據啟發本研究修剪人工神經元的作法（見研究動機：生物決策能量與效率觀點）。

（一） 系統架構一：對照組，捲積人工神經網路，沒有修剪人工神經元

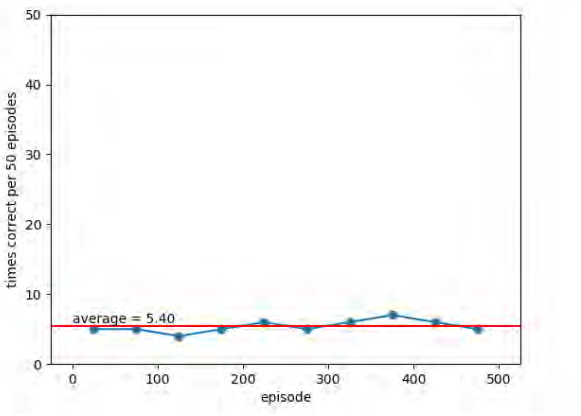
1. 系統說明

系統架構一使用一個捲積人工神經網路代表一個行為。捲積層使用了 32 個 3x3 濾鏡，而完全連結層使用 128 個神經元，最後輸出層藉由 softmax 激活函數輸出五個動作的值。機器人每次行使輸出層中數值最大的動作。



圖（十九）：系統架構一示意圖。
圖中輸入圖片上有一灰色正方形，代表濾鏡的感官區域，為 3x3 畫素。

2. 測試情境一：推任意球



圖（二十）：測試情境一，捲積人工神經網路在目標學習上的正確率。
模擬 500 個回合，紀錄每 50 個回合達到目標行為的次數。

模擬了 500 個回合，紀錄每 50 個回合機器人達到目標行為的次數。由於系統並沒有動態改變，繼續模擬的結果會大致一致，受於時間限制我們只進行了 500 個回合。

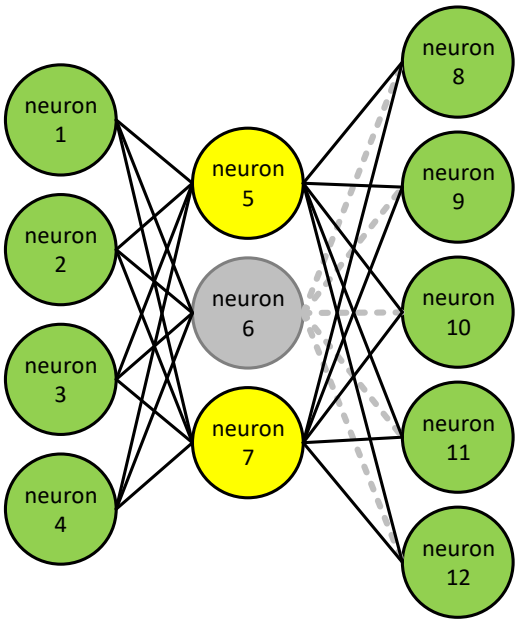
此外，本研究如此繪圖是因為環境的獎勵是稀疏的，每一回合的獎勵總和只有-1、-0.5 與 1 而已，如果按照一般的「回合總獎勵對回合」作圖，則不易看出趨勢及系統效能。

為何這個測試情境對於電腦頗有難度？在訓練人工神經網路示範行為時，神經網路只看過「紅色的球」以及「紅色的立方體」。當神經網路看到不同顏色的球或立方體時，會因為顏色完全不同而認為是兩完全相異的物品，而一直旋轉。即使我們已把去掉顏色的形狀資訊分離出來（在原攝影影像增添了一層灰階照片），捲積人工神經網路仍無法有效的生成目標行為。

（二） 系統架構二：捲積人工神經網路＋隨機修剪神經元

1. 系統介紹

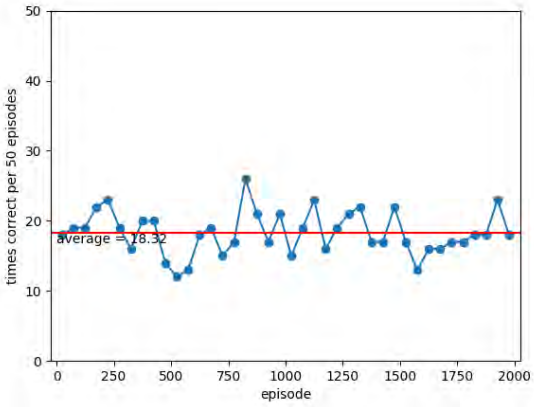
在這個系統當中，我們取一存有示範行為的捲積人工神經網路（同系統架構一的捲積人工神經網路），在每回合前隨機修剪大約一半的神經元，然後紀錄在這個回合中所得的獎勵。所謂「修剪神經元」，即隨機將捲積層（convolution layer）濾鏡矩陣（filter matrix）中的某些權重值設為 0、完全連接層（fully connected layers）中某些神經元連出的全部權重值設為 0。



圖（二十一）：修剪神經元示意圖。圖中神經元 6 號為被修剪的狀態。

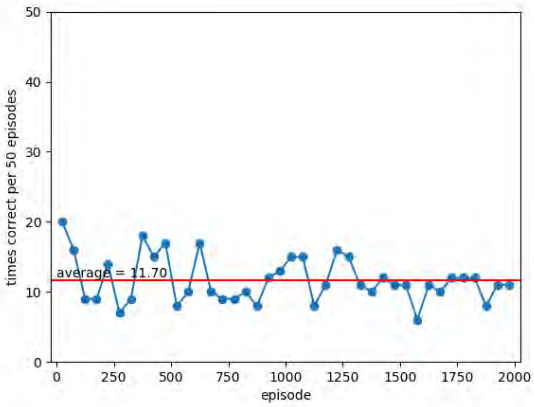
回合結束後，我們恢復被修剪的神經元，到了下一回合又會隨機修剪不同組的神經元。值得強調的是修剪神經元的依據是隨機的，可說成在靠運氣學習「目標行為」。

2. 測試情境一：推任意球



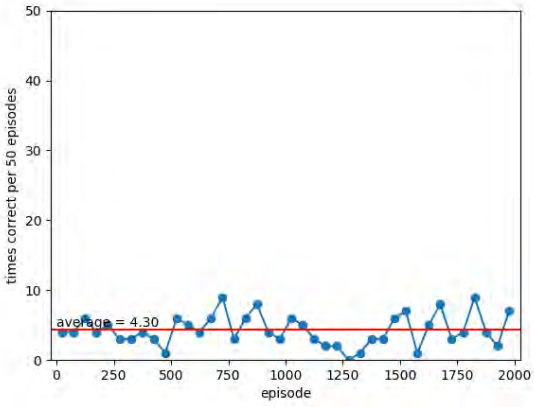
圖（二十二）：測試情境一，捲積人工神經網路＋隨機修剪神經元在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合達到目標行為的次數。

3. 測試情境二：只能推紅球



圖（二十三）：測試情境二，捲積人工神經網路＋隨機修剪神經元在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合達到目標行為的次數。

4. 測試情境三：環境中加圓柱體



圖（二十四）：測試情境三，捲積人工神經網路＋隨機修剪神經元在目標學習上的正確率。

修剪神經元是否真的能更有效率地產生目標行為？在前置實驗當中（P.42），少回合數內，即可得出一含有目標行為的神經網路。這表示雖然總共修剪神經元的方法有高達 2^N 種（N為神經元總數），事實上有很多修剪神經元的方式皆是目標行為。而測試情境一的結果亦能驗證前置實驗的結果，能有效的提高正確率。

不過，隨機修剪神經元正確率相較系統架構三偏低，原因是此系統並不確定目前修剪過後的神經網路為一目標行為，而繼續在其他回合中隨機修剪另外一組神經元，因此可比喻成沒有效率的修剪或學習。一個比較符合生物有效的學習方式，修剪神經元應與獎勵系統做連結。

三、 尋找更有效率的修剪人工神經元方式以找到目標行為

（一） 系統架構三：捲積人工神經網路＋Reinforcement Learning 修剪神經元

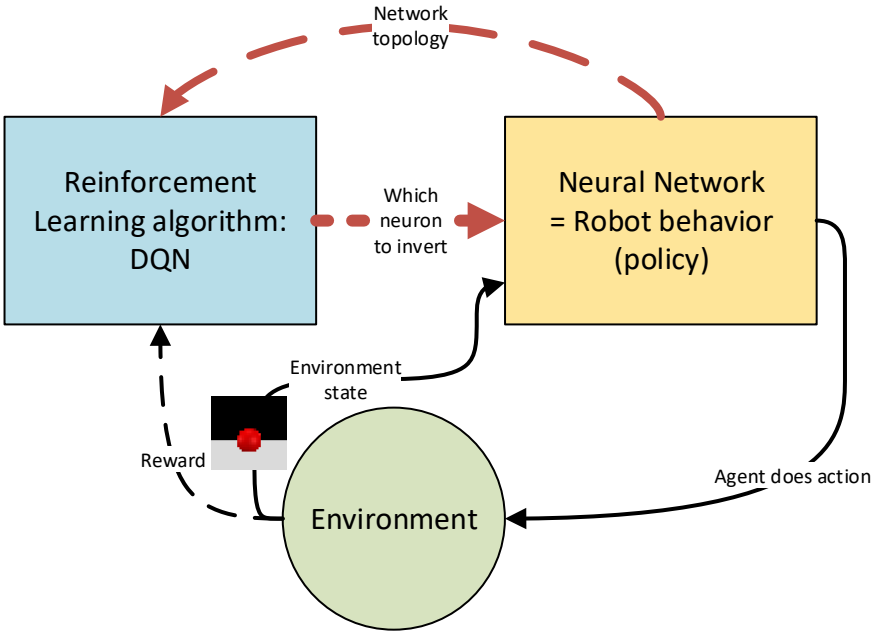
1. 系統架構說明

在這個系統架構中，不同於系統架構二的是，每回合選擇修剪神經元時，是依照強化學習中的 $q^{\pi}(s_t, a)$ 函數而定。

我們把這裡的 s_t 定為第 t 回合中，代表機器人行為神經網路的「剪法」，以一個大於等於 0，小於等於 $2^N - 1$（其中 N 為總神經元數）的數字代表。如果第 0 個神經元已被修剪，則將 s_t 二進位表法中的 2^0 項設為 1，若神經元存在則設為 0；如果第 1 個神經元被修剪則將 s_t 二進位表法中的 2^1項設為 1，若神經元還存在則設為 0，依此類推，總共 2^N 種狀態（剪法）。

可以行使的動作 a 則為一界於 $[0, N]$ 的數字，0 代表不做任何動作，其餘數值代表要翻轉（invert）神經元 $a - 1$ 的狀態，即把神經元由被修剪的狀態變成連結的狀態，或把神經元由連結的狀態變成被修剪的狀態。

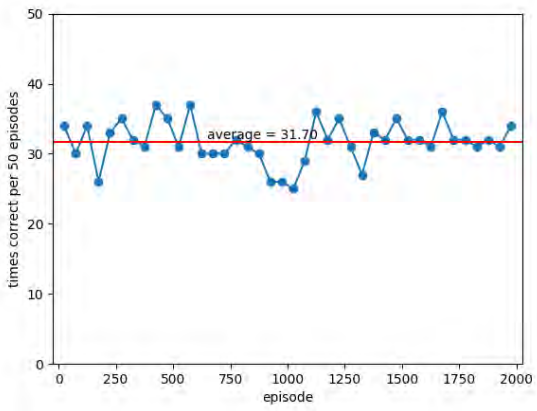
當 N 小時，可以使用表格儲存，但是通常神經網路的神經元數量非常多，因此我們使用 Deep Q-network 來近似此 $Q(s, a)$ 函數。



圖（二十五）：系統架構三示意圖

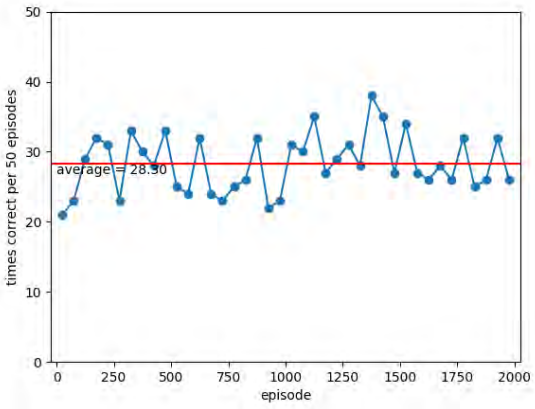
2. 測試情境一：推任意球

模擬 2000 個回合，紀錄每 50 個回合機器人達到目標行為的次數。系統架構二是本研究中在測試情境一中達到正確率最高的系統。



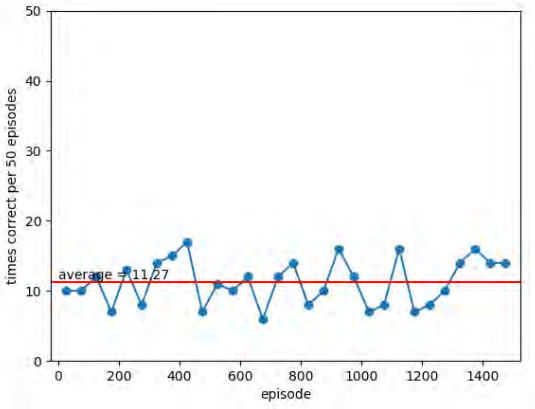
圖（二十六）：測試情境一，CNN＋RL 修剪神經元系統在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合達到目標行為的次數。

3. 測試情境二：只能推紅球



圖（二十七）測試情境二，CNN＋RL 修剪神經元系統在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合達到目標行為的次數。

4. 測試情境三：環境中加圓柱體



圖（二十八）：測試情境三中，CNN＋RL 修剪神經元系統在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合機器人達到目標行為的次數。

從結果圖中可發現，在系統架構三下，學習快速，在測試情境中，於 50 個回合內已可使正確率增加至每回合正確 31 次。

圖中亦可發現測試情境二（只許推紅球）的「50 回平均正確率」比測試情境一低，但無論測試情境一（推任意球）或測試情境二的「50 回平均正確率」，皆比隨機修剪神經元的系統架構二高。我們認為有以下三點原因：

（1） 由隨機修剪神經元系統架構二的前置實驗（P.42）可知，隨機修剪神經元在極少的回合數內即可找到目標行為。這表示系統架構三中，一開始隨機修剪神經元的神經網路狀態已定了「大勢」，其後依賴 RL 增減神經元方式是一個一個慢慢加入或修剪，系統較能維持在目標行為的神經網路拓普，提高正確率。

（2） 如果對系統架構三做和系統架構二相同的前置實驗，可發現如果把系統架構三的探索 ε 機率設為較低，則可以出現答對率100%的結果。我們認為由於此目標行為簡單，不必嘗試太多修剪方法即可達到目標行為，因而每次當系統架構三的 RL 系統嘗試修剪其他神經元時，總會使得正確率降低。正確率降低，又會回饋給 RL 系統，使得 RL 系統將神經網路拓普恢復為原本的樣子，而使得實驗結果中的正確率有所波動。

（3） 而實驗環境二的正確率偏低，若以生物的角度解釋，也是合理的。雖然目標行為與示範行為相同看似比實驗環境一容易，但是在此實驗環境之下，機器人必須嘗試推過所有或大部分的顏色球，才能夠確認「只有推紅色球是會有獎勵的」。這就好像學習時達成目標的行為越單一，其實是越困難的。這是當學習是「學什麼方法可以達成目標」和「學哪些方法不能達成目標」，後者較難。而事實上目標行為與示範行為相同的應用也並不多。

測試情境三的結果，會進一步於討論（P.37）中說明。

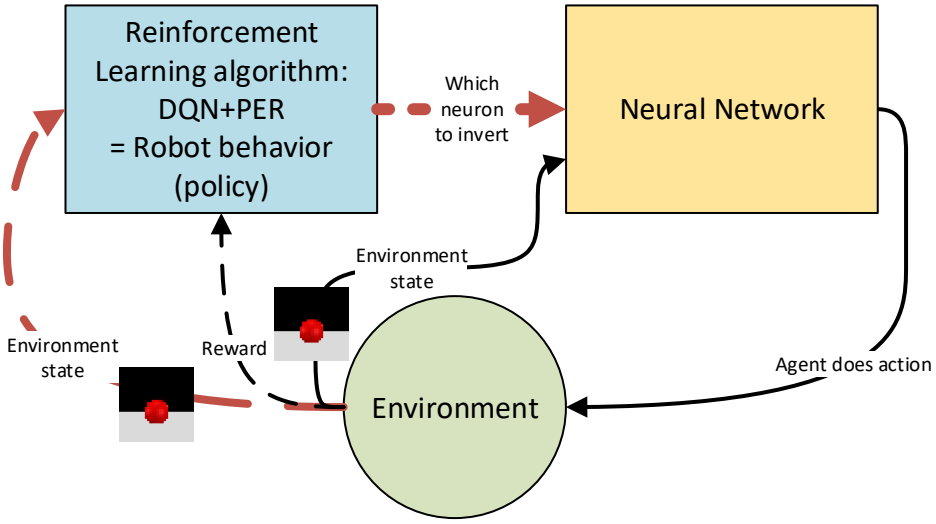
四、 在強化學習系統裡，透過修剪人工神經元，使得強化學習能直接探索一新的行為，增加探索效率及達成目標行為的正確率

（一） 系統架構四：藉由修剪神經元改良現行常用之強化學習系統

1. 系統架構介紹

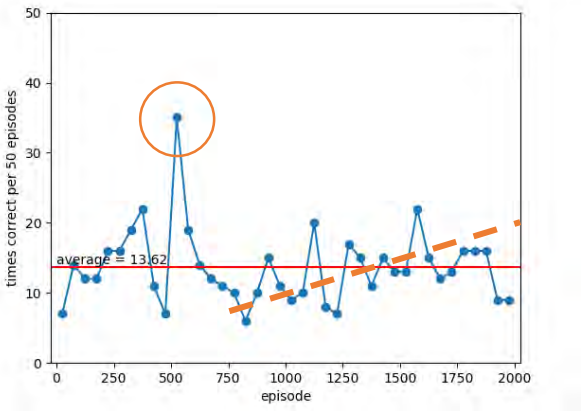
系統架構四中，我們以翻轉（invert）示範神經網路各個神經元，作為強化學習系統的動作輸出。在每個時間點（time step），強化學習會選擇一個神經元翻轉。修剪後的示範神經網路依照環境輸入計算機器人應該執行何種動作。

在此系統架構中，強化學習系統輸入為環境影像，與系統架構三不同。



圖（二十九）：系統架構四示意圖

2. 測試情境一：推任意球



圖（三十）：測試情境一中，改良式強化學習系統在目標學習上的正確率。
模擬 2000 個回合，紀錄每 50 個回合機器人達到目標行為的次數。

系統架構四中，我們發現當機器人在探索新行為時，並非如現行常用的強化學習系統一般，一個動作一個動作探索，而是直接生成一連續新的行為。藉由修剪神經元的探索機制，能增加系統訓練的效率、機器人行使行為的穩定度（探索時不會執行隨機無理的動作）以及達成目標行為的平均正確率。

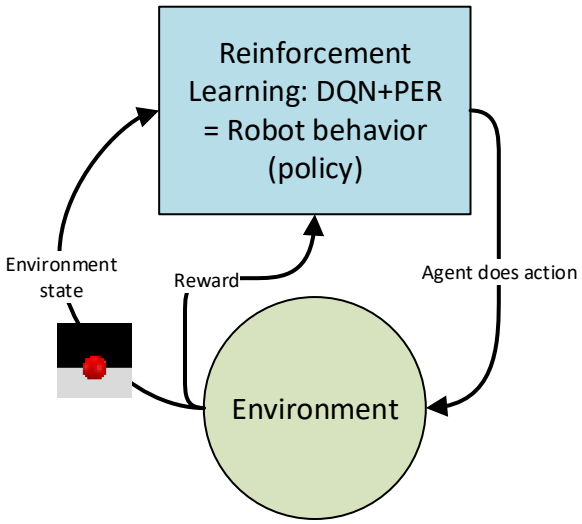
圖中可發現在第 500～550 回合時，正確次數高達 35 的顛峰，推測這 50 回合當中強化學習系統恰好探索到能達成目標的神經網路修剪方法，且維持了一陣子，使得正確率變高。之後又因執行探索，神經網路不再是目標行為，正確率急遽下降，但之後又緩慢上升，強化學習確實有學到如何修剪神經元比較好。

五、 現行常用的強化學習（Reinforcement Learning）系統學習目標行為的效率

（一） 系統架構五：Deep Q-Network（DQN）＋Prioritized Experience Replay

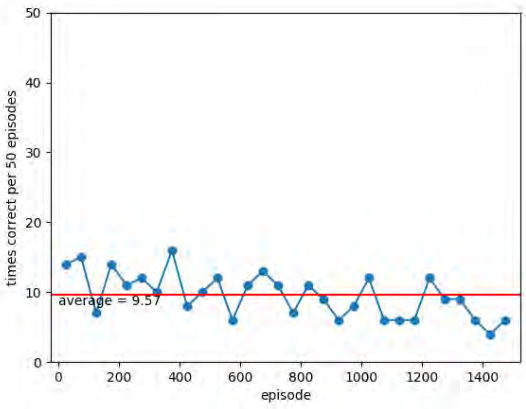
1. 系統架構介紹

在這個系統當中，我們以強化學習為主體控制機器人。在示範行為階段，我們先利用 Prioritized Experience Replay 訓練 DQN 16 個回合，每回合取 32 個示範行為的 RL transition 來訓練。在機器人自由與環境互動的階段，不同於一般的強化學習系統，我們讓示範行為的所有 RL transition 永久保留於記憶體（experience replay buffer）中，當記憶體已滿，新的經驗不會清除示範行為的 RL transition，這是為了讓系統可以保留「對的經驗」(在與示範行為情境相同的情境下，依循示範行為機器人總是能得到獎勵)。



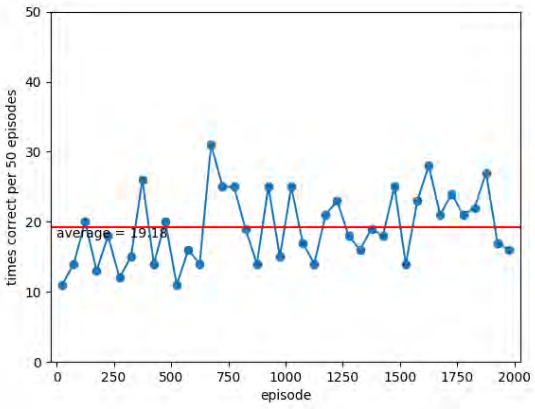
圖（三十一）：系統架構五（Deep Q-network＋Prioritized Experience Replay）示意圖。

2. 測試情境一：推任意球



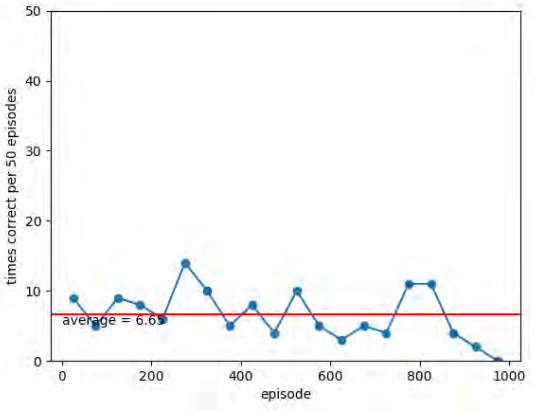
圖（三十二）：測試情境一中，現行強化學習系統（DQN+PER）在目標學習上的正確率。

3. 測試情境二：只能推紅球



圖（三十三）：測試情境二中，現行強化學習系統（DQN+PER）在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合機器人達到目標行為的次數。

4. 測試情境三：環境中加圓柱體



圖（三十四）：測試情境三中，現行強化學習系統（DQN+PER）在目標學習上的正確率。模擬 2000 個回合，紀錄每 50 個回合機器人達到目標行為的次數。

實驗結果顯示此系統在目標行為的學習正確率並不高，平均 50 回合中低於 10 次成功達成目標。其中測試情境二之正確率較高是因為在沒有特別修剪神經元的情形下，DQN 直接輸出示範行為，而示範行為又與目標行為一致，不太需要進行探索。不過正確率仍比單純 CNN（近 100%正確）或者系統架構三（CNN＋RL 修剪神經元）低，因為這個系統在學習示範行為時，必須將示範行為的資料（RL Transitions）轉成 Q-function（即 Q-learning 演算法），比人工神經網路學習（監督式學習）耗時。

綜合五個系統在目標行為學習上的的學習表現，整理如下：

系統架構	測試情境一：推隨意球	測試情境二：推紅色球	測試情境三：加圓柱體
CNN	10.80%	-	-
CNN + random dropout	36.64%	23.40%	8.60%
CNN + RL dropout	63.40%	56.60%	22.54%
Enhanced RL + PER	27.24% (peak 70%)	-	-
RL + PER	19.14%	38.36%	13.30%

表（一）：綜合五個系統架構學習目標行為的平均正確率

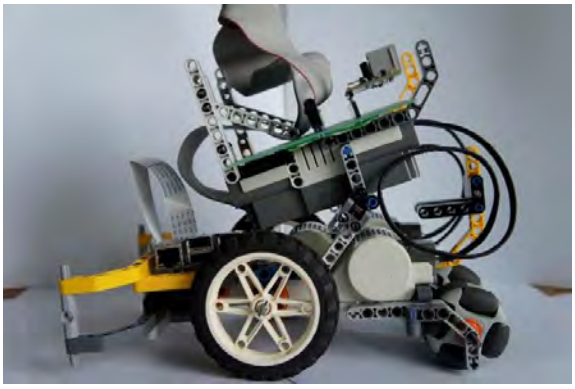
從表中可以清楚看出，本研究所提出的獎勵回饋修剪神經元方式確實大大提升了目標行為學習的效率。雖然隨機修剪神經元有時可達到比較好的學習成果，但不一定在每個環境中都能適當的修剪神經元，所以在測試情境二與測試情映三中學習成果較差。另一方面，傳統強化學習只有在測試情境二下表現比較好，因為目標行為與示範行為一致，但仍比 RL 修剪神經元略遜一籌。

六、 實作於 NXT 機器人

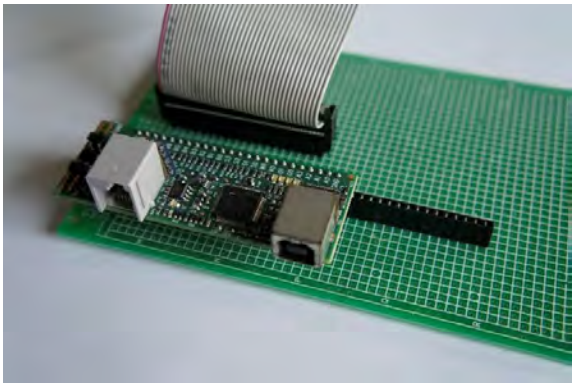
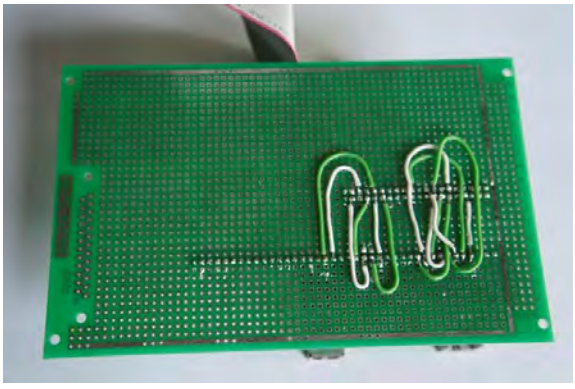
（一） NXT 機器人實體

本機器人主要由下列零件所組成：

1. 一個 NXT 與 HiTechnic NXT SuperPro Sensor Board（SPR2010）
2. 一個樹莓派（Raspberry Pi 2 B+）和樹莓派攝影機模組
3. 其他樂高零件、排線等

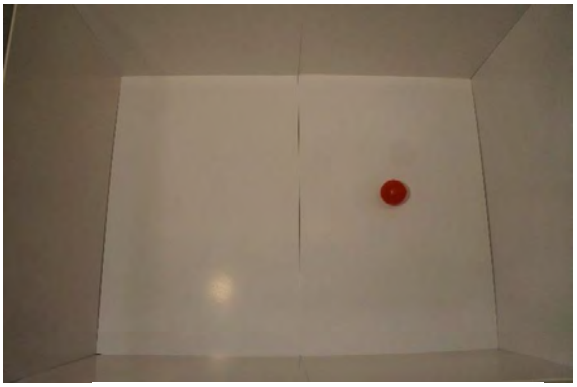



圖（三十五）：機器人側面及正面照

圖（三十六）：HiTechnic NXT SuperPro Sensor Board 與自製樹莓派連接板

圖（三十七）：樹莓派及其相機

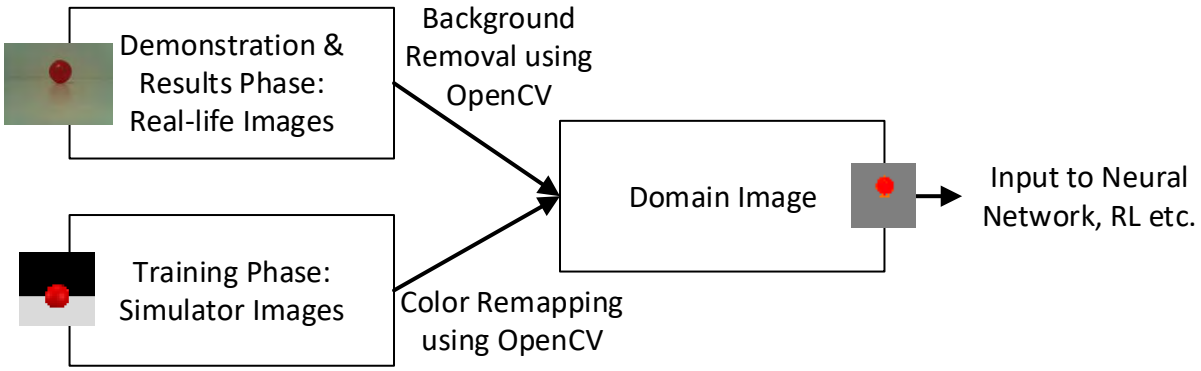


圖（三十八）：真實測試環境

（二） 實作上所遇困難及解決方法

由於訓練捲積人工神經網路及強化學習等運算需求較大，我們使用電腦訓練。首先，我們示範示範行為給實體機器人，使之蒐集影像及動作資料。接下來，我們使用蒐集的資料於電腦模擬上訓練系統。最後，再將訓練完畢的系統裝載於樹莓派上執行。

對於真實世界與模擬世界的連結，我們使用 OpenCV 將真實世界及模擬世界轉換為同一世界（domain），在此世界上訓練系統。這樣不僅能解決樹莓派運算量小的問題，亦能增進訓練效率，這是因為神經網路不需要處理真實世界的各種雜訊，輸入為較簡單的影像。



圖（三十九）：真實及模擬世界轉換示意圖

柒、討論

一、 為何本研究採修剪神經元（dropout neurons）方式而非修剪權重值（weights）？

雖然修剪權重值較符合生物現象，但本研究採取修剪神經元的原因如下：

（一） 本研究曾嘗試修剪權重值，在小神經網路中，加強學習確實可以達到目標行為的修剪權重值方法。但發現在人工神經網路中，每新增一層，權重隨神經元數呈平方級數增加，會導致狀態數過多，加強學習系統學習緩慢。

（二） 雖然 DropConnect（Wan et al.，2013）在論文中表示修剪權重可以提高人工神經網路的正確率，但是仔細觀察可以發現，利用 DropConnect 訓練一個人工神經網路的正確率反而較 Dropout 低。推測 DropConnect 是因為同時訓練五個人工神經網路，在輸出時平均結果而使得正確率提高。此外，Dropout 的正確率與 DropConnect 的正確率差異甚小（1%以內，大多小於 0.5%），因此不妨使用修剪神經元的方式。

二、 當目標行為有示範行為中沒有出現的新特徵（例如實驗環境三中的圓柱體）時：

（一） 人工神經網路的限制：

我們以系統架構三（CNN + RL Dropout）於測試情境三（圓柱體）中產生的行為為例說明。在模擬當中，由於機器人（人工捲積神經網路）未曾看過圓柱體，加上顏色是隨機，機器人也未曾看過這些顏色，通常機器人會選擇避開圓柱體（繼續左轉）。特別的是，觀察機器人選擇推的圓柱體，可發現大多呈紅色的相近色。

如果把圓柱體的顏色維持紅色不變，可以進一步發現每當機器人面前置有此紅色圓柱體時，機器人都會去推，即使機器人一直收到負獎勵，還是行使一樣的行為。這說明了人工神經網路的一個限制，也就是在示範行為當中，必須告訴人工神經網路所有的特徵及該行使的行為，才能夠藉由修剪神經元嘗試各種特徵的組合，例如：「圓球要前進」、「立方體要左轉」;「圓球上無斑點要前進」、「圓球上有斑點要左轉」。

我們假想，若生物遇到同樣的情境又會如何呢？假設在測試情境三裡，目標行為是「要推任何顏色的圓球，也要推任何顏色的圓柱體」，那麼對於生物來說，由於看過圓球，而且推了紅色的圓球是有獎勵的，當面對不同顏色的圓球時，也會去推，這是因為這個新的物體與原來物體有某種程度的相似性。但當生物看到既不是看過的形狀，又不是看過的顏色的物體時，即使實際上推了這個物體會有獎勵，在不知道會否有獎勵的情況之下，生物應該會依照其願意嘗試（勇敢）的程度選擇要不要去推物體。

在生物的一般任務當中，目標行為總有某些特徵與示範行為相同或相似，而藉由加強學習修剪神經元的方式，可以快速地找出目標行為與示範行為共同的特徵。

另外，要讓機器人從無（或少數範例）中生有，產生新的特徵，是非常不容易的事情。這個問題在機器學習的領域其實並不陌生，稱為 One-shot Learning，是近年來 Google 與 DeepMind 正熱烈研發的領域。本研究定位在「機器人對於環境有基本的認識，可以分辨不同的特徵」的前提下，如何由示範行為中找出與目標行為相關的特徵，因此對於環境的基本認識，必須包含於示範行為之中，也就是「示範行為必須示範相對的特徵與其相對應行使的行為」機器人才能有依據「去蕪存菁」，達成「舉一反三」的效果。

（二） 強化學習系統可能出現的問題：

強化學習系統在面對未知的物體，仍必須仰賴人工神經網路（DQN）做決策是否要推這個物體，否則，強化學習必須完全仰賴探索機制（exploration）。所謂探索機制是指：每一次行使動作時，在某個機率之下，嘗試行使**不同於**人工神經網路認為能最大化獎勵的動作。

對於獎勵豐富的任務，此探索機制能讓強化學習系統，快速地找出目標行為；但對於獎勵稀少（如本研究的三個測試情境）的任務，此探索機制雖然保證總有一天能夠得到獎勵、找到目標行為，但機率甚小。因為推球過程中每一個動作皆要探索一次，機率成指數速度減小，在小環境中也許能在有限時間內找到目標行為，但在大環境中就幾乎永遠無法找出目標行為。

在以上兩種方式皆無法面對新特徵的前提之下，在特定的情境下，利用修剪神經元的方式探索目標行為的優勢便浮現。**修剪好的人工神經網路可以一次探索一個新的、連續的行為，而強化學習僅能以一個動作一個動作的方式慢慢探索，效能上可見一斑。**

三、 強化學習＋Learning from Demonstration 與本研究的差異為何？

強化學習的 Learning from Demonstration 大多為了使訓練時間縮短。例如，Večerík et al.（2017）使用示範行為應用於機器手臂插桿子入盒、機器手臂裝硬碟等任務上。強化學習能使得插洞行為更精準，或適用於各種盒子的擺法，但當遇到與示範行為不同顏色的盒子時，由於要行使什麼動作仍需仰賴人工神經網路的抉擇，在沒有修剪神經元的情況之下，機器人可能就會認為此盒與示範行為中的盒子不一樣，而無法正常插入桿子。在這個情形之下，強化學習唯一能仰賴的是其探索機制，而又由於插桿子為一種獎勵稀疏的任務，強化學習系統會無法有效地學習「在其他顏色的盒中插桿子」的行為。此即本研究想解決的問題。

四、 研究貢獻

綜合本研究結果，本研究中利用強化學習中 Deep Q-Learning 修剪神經元的系統可運用在清楚簡單的示範行為，例如：撿拾散落地上的特定形狀的螺絲、夾起娃娃機裡特定形狀或顏色的娃娃。這個系統可以簡化控制機器人的時間，因為只要給予機器人目標行為的子集，不需要全部都給，機器人就可以自己藉由與環境互動，快速地學到目標行為。

五、 未來研究可進一步回答之問題

（一） 如何進一步縮小狀態與動作量級使得系統架構三、四中的 DQN 能夠在較短時間內找到最佳的修剪神經元方式？

（二） 如果將修剪神經元應用在獎勵充沛的情境下，會有什麼樣的結果？

捌、 結論

綜合模擬結果，本研究回答了實驗目的中提出的問題如下：

一、 神經網路與強化學習兩種模型皆可儲存行為，且利於機器人自行修正。
二、 修剪神經元可以去蕪存菁神經網路，找到與目標行為相關的特徵。
三、 利用獎勵回饋於修剪人工神經元，可提高正確率，並在較短時間內學到目標行為。
四、 在強化學習系統中加入修剪人工神經元的元素，可使之在目標學習上更有效率，因為探索時可以直接產生完整連續的行為，改良了強化學習直接探索動作的缺點。
五、 和現行常用的學習系統（Deep Q-Learning＋Prioritized Experience Replay）相比，在目標學習上，無論隨機修剪或獎勵回饋地修剪人工神經元表現都比較好。

本研究比較了五種系統在學習目標行為的效率，結果顯示，利用回饋機制修剪人工神經元可以在較短時間內達到較高的正確次數。傳統的強化學習（Reinforcement Learning）系統，在本研究的模擬中，如果回饋機制影響的是動作本身，目標達成正確率只有 19.14％。而若獎勵回饋到神經元的修剪上，系統修剪掉與目標行為無關的神經元，則可以相對提高超過 2 倍的正確率。甚至，隨機修剪神經元也可提高了 1 倍的目標達成正確率。顯然地，本系統能確實提高目標學習正確次數，並縮短目標達成時程。

利用回饋機制修剪人工神經元，可為強化學習系統在目標學習上遇到的困境，提供一個新的思考方向。在實務的應用上，可彌補加強學習在學習行為上無法一般化的缺點。

未來研究方向，可著重在解決當目標行為中有示範行為沒有的新特徵出現時，如何舉一反三。因此，可加強發展結合獎勵回饋修剪神經元與加強學習系統上的優點，前者可以提升目標行為的學習效率，而後者可以對環境中的新變數提供可能的應變。

玖、 參考資料及附錄

葉暘、何政勳（2017）。用於機器人空間建模的仿生認知系統。2017 臺灣國際科學展覽會。

Balcombe, J.（2016）。大智若魚（姚若潔譯）。科學人雜誌，2016 年 08 月，86-89。

Gopnik, A.（2017）。機器學習舉一反三（鍾樹人譯）。科學人，2017 年 09 月，34-39。

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. Robotics and autonomous systems, 57(5), 469-483.

Balcombe, Jonathan P. What a fish knows: the inner lives of our underwater cousins. Oneworld, 2016.

Biggs, G., & MacDonald, B. (2003, December). A survey of robot programming systems. In *Proceedings of the Australasian conference on robotics and automation* (pp. 1-3).

Gruenstein, J., & Truell, M. (2016). Fido: A Universal Robot Control System using Reinforcement Learning with Limited Feedback. Intel International Science and Engineering Fair.

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., ... & Leibo, J. Z. (2017). Learning from Demonstrations for Real World Reinforcement Learning. *arXiv preprint arXiv:1704.03732*.

Kaehler, A., & Bradski, G. R. (2017). Learning OpenCV 3: computer vision in C++ with the OpenCV library. Sebastopol: OReilly.

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

Loukola, O. J., Perry, C. J., Coscos, L., & Chittka, L. (2017). Bumblebees show cognitive flexibility by improving on an observed complex behavior. *Science,355*(6327), 833-836. doi:10.1126/science.aag2360

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529-533.

Montana, J., & Gonzalez, A. (2011). Towards a unified framework for learning from observation. In *IJCAI Workshop on Agents Learning Interactively from Human Teachers*.

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, *15*(1), 1929-1958.

Sutton, R. S. (1998). *Introduction to reinforcement learning*. Cambridge, Mass: MIT Press.

Večerík, M., Hester, T., Scholz, J., Wang, F., Pietquin, O., Piot, B., ... & Riedmiller, M. (2017). Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*.

Vivo, L. D., Bellesi, M., Marshall, W., Bushong, E. A., Ellisman, M. H., Tononi, G., & Cirelli, C. (2017). Ultrastructural evidence for synaptic scaling across the wake/sleep cycle. *Science,355*(6324), 507-510. doi:10.1126/science.aah5982

Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., & Fergus, R. (2013). Regularization of neural networks using dropconnect. In *Proceedings of the 30th international conference on machine learning (ICML-13)* (pp. 1058-1066).

Xie, J., & Padoa-Schioppa, C. (2016). Neuronal remapping and circuit persistence in economic decisions. *Nature neuroscience*, *19*(6), 855-861.

附錄

一、 前置實驗（pilot study）設計

前置實驗的目有二：

（一） 找出使用何種激活函數與何種學習權重值的演算法效能比較好？

（二） 了解在較簡單的設置下，藉由修剪神經元是否可由小神經網路中擷取與輸出結果有關的特徵。

為了簡化問題，我們設計了以下「示範行為」及「測試環境」數據，並使用一個 3x5x1（不包括偏差值）的人工神經網路代表行為。

我們設計的目標行為是「輸出數只與第一行的數據有關，只要第一行數據為 0，則輸出 1，反之，則輸出 0」。示範輸入為第一階段示範行為、訓練人工神經網路的數據，而測試環境的輸入及輸出，分別為系統與環境互動時的「題目」(與主實驗設置中的「不同顏色球」相對應）以及應輸出的答案。

示範輸入	示範輸出
[0, 1/3, 0.5]	[1]
[2/3, 1/3, 0.5]	[0]

表（二）：前置實驗中的示範資料

測試輸入	輸出	測試輸入	輸出	測試輸入	輸出
[0, 0, 0]	[1]	[1/3, 0, 0]	[0]	[2/3, 0, 0]	[0]
[0, 0, 1/3]	[1]	[1/3, 0, 1/3]	[0]	[2/3, 0, 1/3]	[0]
[0, 0, 0.5]	[1]	[1/3, 0, 0.5]	[0]	[2/3, 0, 0.5]	[0]
[0, 0, 0.8]	[1]	[1/3, 0, 0.8]	[0]	[2/3, 0, 0.8]	[0]
[0, 0, 1]	[1]	[1/3, 0, 1]	[0]	[2/3, 0, 1]	[0]
[0, 1/3, 0]	[1]	[1/3, 1/3, 0]	[0]	[2/3, 1/3, 0]	[0]
[0, 1/3, 1/3]	[1]	[1/3, 1/3, 1/3]	[0]	[2/3, 1/3, 1/3]	[0]
[0, 1/3, 0.5]	[1]	[1/3, 1/3, 0.5]	[0]	[2/3, 1/3, 0.5]	[0]
[0, 1/3, 0.8]	[1]	[1/3, 1/3, 0.8]	[0]	[2/3, 1/3, 0.8]	[0]
[0, 1/3, 1]	[1]	[1/3, 1/3, 1]	[0]	[2/3, 1/3, 1]	[0]
[0, 0.5, 0]	[1]	[1/3, 0.5, 0]	[0]	[2/3, 0.5, 0]	[0]
[0, 0.5, 1/3]	[1]	[1/3, 0.5, 1/3]	[0]	[2/3, 0.5, 1/3]	[0]
[0, 0.5, 0.5]	[1]	[1/3, 0.5, 0.5]	[0]	[2/3, 0.5, 0.5]	[0]
[0, 0.5, 0.8]	[1]	[1/3, 0.5, 0.8]	[0]	[2/3, 0.5, 0.8]	[0]
[0, 0.5, 1]	[1]	[1/3, 0.5, 1]	[0]	[2/3, 0.5, 1]	[0]

[0, 0.8, 0]	[1]	[1/3, 0.8, 0]	[0]	[2/3, 0.8, 0]	[0]
[0, 0.8, 1/3]	[1]	[1/3, 0.8, 1/3]	[0]	[2/3, 0.8, 1/3]	[0]
[0, 0.8, 0.5]	[1]	[1/3, 0.8, 0.5]	[0]	[2/3, 0.8, 0.5]	[0]
[0, 0.8, 0.8]	[1]	[1/3, 0.8, 0.8]	[0]	[2/3, 0.8, 0.8]	[0]
[0, 0.8, 1]	[1]	[1/3, 0.8, 1]	[0]	[2/3, 0.8, 1]	[0]
[0, 1, 0]	[1]	[1/3, 1, 0]	[0]	[2/3, 1, 0]	[0]
[0, 1, 1/3]	[1]	[1/3, 1, 1/3]	[0]	[2/3, 1, 1/3]	[0]
[0, 1, 0.5]	[1]	[1/3, 1, 0.5]	[0]	[2/3, 1, 0.5]	[0]
[0, 1, 0.8]	[1]	[1/3, 1, 0.8]	[0]	[2/3, 1, 0.8]	[0]
[0, 1, 1]	[1]	[1/3, 1, 1]	[0]	[2/3, 1, 1]	[0]

表（三）：前置實驗的測試環境

二、 前置實驗結果

（一） 找出使用何種激活函數與何種學習權重值的演算法效能比較好？

隱藏層 激活函數 / 演算法	Sigmoid	tanh（$\frac{1-e^{-2x}}{1+e^{-2x}}$）	Rectifier
Stochastic gradient descent（SGD）	33%	84%	79%
Adam	77%	77%	79%

表（四）：在未經修剪的情形之下，環境測試資料的正確率

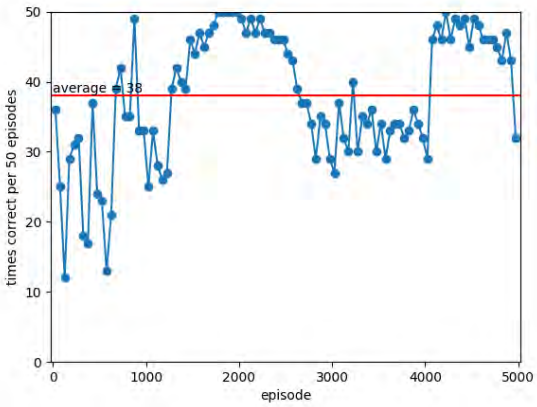
由表中可知，綜合穩定性及正確率的考量下，本研究使用 Adam 及 Rectifier 為人工神經網路的權重學習方法及激活函數。

（二） 了解藉由修剪神經元是否可由小神經網路中擷取與輸出結果有關的特徵。

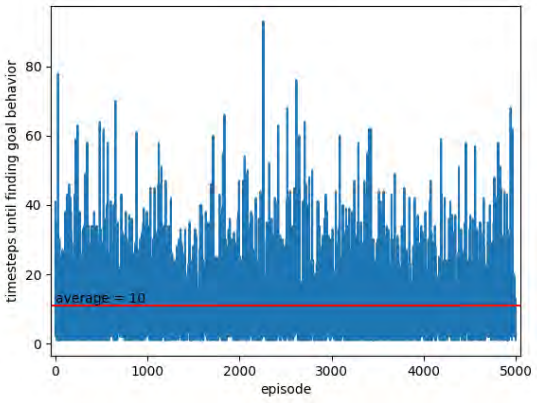
在這個前置實驗當中使用了一個 *state space* x 50 x 50 x *action space* 的 DQN 來修剪**權重**。可以看到如果在大約 2000 回合時就不再降低探索 ε 值，可以使得正確率維持在 100%，且避免正確率起伏（fluctuation）。但是由於 ε 代表探索新修剪方法的程度，我們仍然希望最終可以維持一固定 ε 值，讓學習系統在環境改變時，可以充分應變。

此外，在此前置實驗中也測量了隨機修剪神經元及 DQN 修剪神經元需時間以修剪出目標行為。所謂修剪出目標行為是當環境測試資料的 75 筆資料皆答對時。

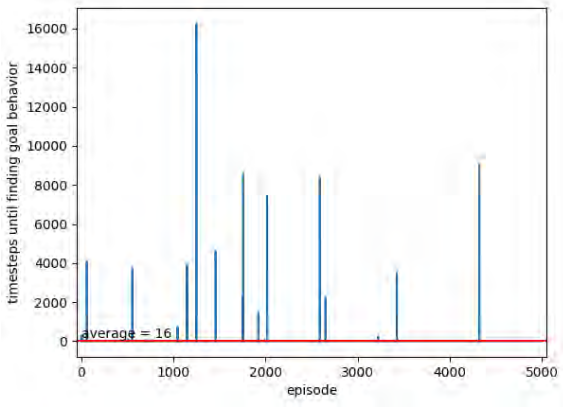
由隨機修剪神經元的圖可發現，在平均十次之內就可修剪出一目標行為，表示在這 2^{26} 種修剪權重方法當中，其實有很多種修剪方式是目標行為。



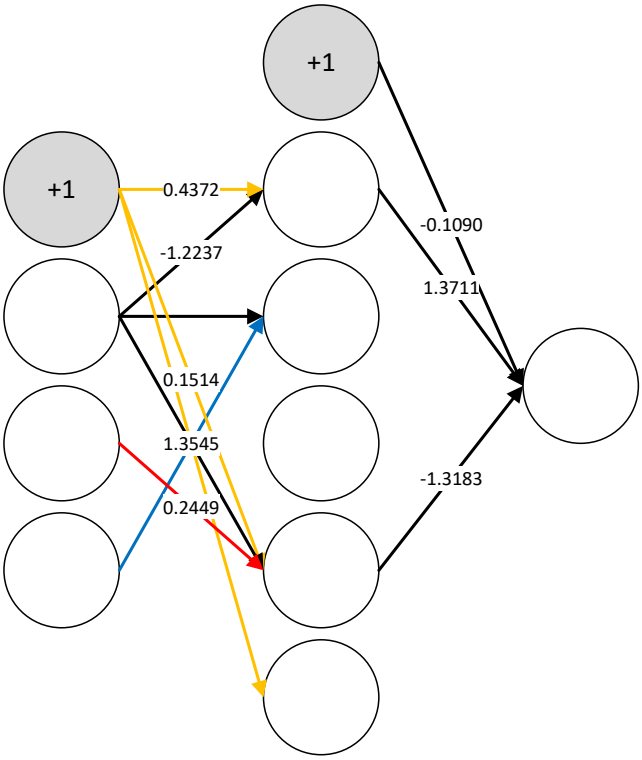
圖（四十）：利用 DQN 修剪簡易人工神經網路。
ε 由 0.3 以每回合 0.998 倍的方式遞減至小於 0.01 後停止下降。



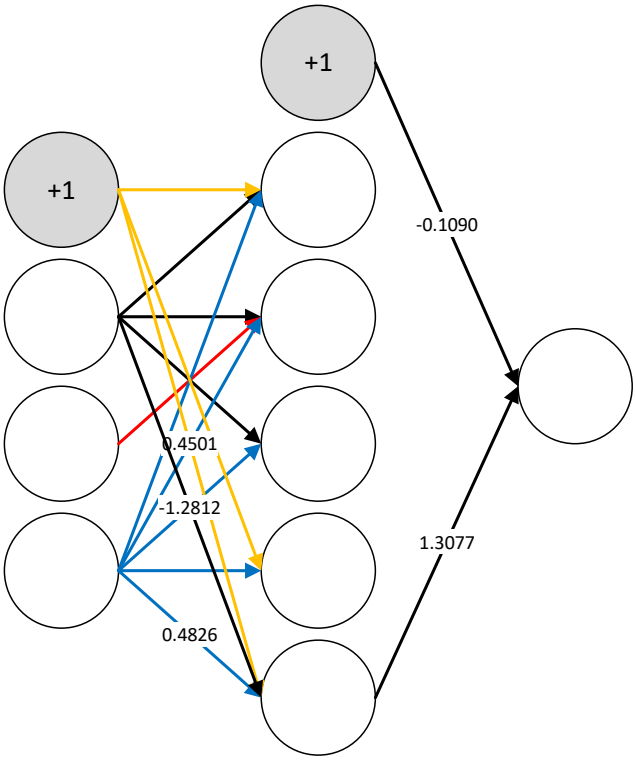
圖（四十一）：隨機修剪神經元在每回合修剪至目標行為的動作次數。
縱軸為在第 n 次修剪恰好修剪到目標行為。



圖（四十二）：DQN 剪神經元在每回合修剪至目標行為的動作次數。
縱軸為在第 n 次修剪恰好修剪到目標行為。



圖（四十三）：利用 DQN 修剪簡易人工神經網路於 2000 回合時的網路模型。
未連結之權重代表已經被修剪，其中未連結至輸出層的權重值省略不寫。
此人工神經網路在環境測試的 75 筆資料中可答對 93%。



圖（四十四）：利用 DQN 修剪簡易人工神經網路於 5000 回合時的網路模型。
未連結之權重代表已經被修剪，其中未連結至輸出層的權重值省略不寫。
此人工神經網路在環境測試的 75 筆資料中可答對 73%。

【評語】190010

此作品提出一個機器人的學習系統，相較於一般機器人使用的強化學習方式（RL），此作品提出用 CNN 的方式，再佐以 RL 來去掉與目標相關性弱的人工神經元。

此作品有創意、有發展潛能，建議對此創意的動機，能有更扎實的說明。還有，需要再嘗試更複雜的目標，以確認此方式的實用性。