人工神经元网络结合TD(λ)算法在中国象棋机器博弈中的应用

王玨[1]，程然[1]，王骄[1]

1. 东北大学信息科学与工程学院, 沈阳 110004
E-mail: soldierwj@gmail.com

摘要： 本文着重介绍人工神经元网络结合TD(λ)算法在中国象棋机器博弈中的应用。使用人工神经元网络（ANN）表示博弈系统中的评价函数，并采用TD(λ)增强算法达到修改网络的权值参数的目的，大量的专家棋谱用来作为学习的训练数据。设计及开发了基于上述方法的增强学习系统，实验结果表明这些技术可以有效提高中国象棋的计算机博弈水平。

关键词: 中国象棋机器博弈，评价函数，人工神经元网络，TD(λ)算法

Applying Artificial Neural Network combined with TD(λ) to Computer Chinese Chess

Jue Wang[1], Ran Cheng[1], Jiao Wang[1]
1. College of Information Science and Engineering, Northeastern University, Shenyang 110004
Email: soldierwj@gmail.com

Abstract: This paper discusses whether Artificial Neural Network combined with TD(λ) method can be successfully applied to computer Chinese chess . Artificial Neural Network (ANN) is used to represent the evaluation function. Learning occurs by using TD(λ) algorithm on the results of high-level database games. Experiments show that the proposed technique can improve the performance of computer Chinese chess.

Key Words: Computer Chinese chess, Evaluation function, Artificial neural network, TD(λ)algorithm

1 引言

机器博弈一直被称为是人工智能领域的“果蝇”。人工智能的许多方法与技术都可以在机器博弈中得到应用。

在众多棋类中，国际象棋、奥赛罗、西洋双陆棋、西洋跳棋等棋类的研究已经取得了丰硕的成果，而具有两千多年历史的中国象棋的机器博弈研究近些年发展迅速，“棋天大圣”等最高水平的中国象棋机器博弈软件已经达到了象棋特级大师的水平。

中国象棋和国际象棋具有许多类似之处，但其空间复杂度和搜索树的复杂度比后者略高，Allis在1994年在他的博士论文中[1]对于二者的复杂度做出了估计，国际象棋的搜索树复杂度为10^{123}，而中国象棋为10^{150}。

近些年，在西洋双陆棋、国际跳棋、国际象棋等棋类的研究中，许多学者引入了人工智能领域的自学习算法进行优化和学习。和这些棋类相比，中国象棋具有更复杂的玩法，更大的搜索空间。究竟现有的一些自学习算法是否可以在中国象棋机器博弈中取得成功，研究者甚少并且没有定论。

本文介绍的是一种自学习算法与中国象棋机器博弈相结合的新方法，该方法是将人工神经元网络结合TD(λ)算法引入中国象棋机器博弈的评价函数，通过自学习的方法，表现棋子之间的内部联系和潜在规则。

本文第2节中，讨论了基于人工神经元网络的评价函数；第3节介绍了TD(λ)的原理，及机器博弈应用的发展历程；第4节分析了训练数据获取的几种方法；第5节介绍了我们设计的实验，并给出实验数据了；第六节给出我们的结论。

2 基于人工神经元网络的评价函数

2.1 评价函数

像其它棋类一样，中国象棋机器博弈涉及的技术主要包括两个方面，即搜索和评价。搜索技术主要是发挥计算机的计算速度优势，对当前局面以后的各种可能走法进行搜索。由于搜索空间过于庞大以及计算机速度的局限，搜索的层数是有限的。这就要求对没有分出胜负的搜索终点局面使用评价函数进行评价。所谓评价就是对某一个静态局面给出一个区分双方优劣程度的分值，这个分值的准确与否，在很大程度上影响着博弈程序的棋力。

传统意义上的评价函数是利用人的经验知识，从局面中提取一些明显的特征，比如棋子的固定子力值、位置值、灵活度值、威胁与保护值等，利用这些特征计算出对弈双方的分数，二者之差就是该局面的评价值。这种评价函数表现为各种特征的线性多项式。

人类的知识毕竟有限，将所有可能情况都考虑到是不现实的，而且评价函数过于复杂也会带来搜索深度降

博士启动基金（合同编号：76105115）

低等负面影响。这自然会让研究者想起用人工智能领域的自学习算法学习棋子之间潜在的联系和规则。

将自学习技术应用于各种棋类的评价函数是近年来的研究热门，其中应用最广的是使用人工神经元网络（ANN）结合TD(λ)算法，已经取得巨大成功的是西洋双陆棋和西洋跳棋，本文将就这种方法在中国象棋评价函数中的应用进行深入的探讨。

2.2 人工神经元网络

人工神经元网络（ANN）是受生物学的启发，通过模拟人类大脑的结构和思维方式建立起来的一种学习网络。它反映了人脑的某些基本特征，是人脑的某种抽象、简化或者模仿。

构成人工神经元网络的基本单元是人工神经元。人工神经元相当于一个多输入单输出的非线性阈值器件。其模型如图1所示：

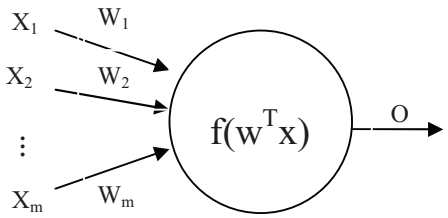

图 1: 人工神经元模型

图中的f为激活函数，可归结为三种形式：阈值型、线性型和S型。人工神经元网络的学习方式有：有监督式学习，无监督式学习和增强式学习。

人工神经元网络有30多种模型被开发和应用，如反向传播（BP）网络、Hopfield网络、自组织映射网络（SOM）等等。其中应用最为广泛的网络模型之一是反向传播（BP）网络。

2.3 用BP网络结构表示评价函数

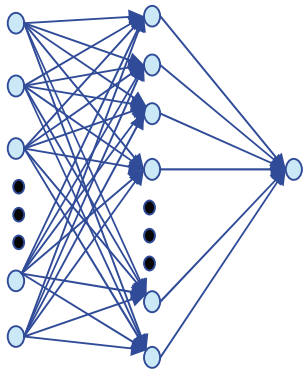

图 2: 反向传播(BP)网络结构

如图2所示，反向传播（BP）网络是一种多层前馈网络。图中所示的BP网具有三层的网络结构，分别是输入层、隐藏层和输出层。每层都具有一定数量的神经元。

本文使用BP网络结构来表示中国象棋机器博弈中的评价函数。我们建立了一个91×N×1的BP网络（N为隐藏层的个数），输入层代表的是局面信息，其中90个输入表示中国象棋棋盘上每个交叉点上棋子的固定子力值，剩下的一个是当前走棋方的表示值（如轮到红方走棋为1，轮到黑方走棋为-1）。隐藏层个数是决定学习效果的一个因素，虽然有一些指导性的原则用于确定隐藏层个数，但一般需要根据具体的学习应用进行实验和调整。输出层是对当前局面的评价值，范围为[-1，1]。输出值越趋向于1，表示红方胜率越大，越趋向于-1，表示黑方胜率越大，越趋向于0，则表示双方和棋的概率较大。隐藏层与输出层的神经元的激活函数均使用双极型的双曲正切函数。

BP网络的反向传播训练算法是一种迭代梯度算法，用于求解前馈网络的实际输出与期望输出间的最小均方差值。其缺点是训练时间较长，且容易陷入局部极小点。

传统的BP网络学习算法是一种有监督的学习，即对于每个输入值要有个期望输出值，输入值与期望输出值构成了一对训练样本。在我们的应用中，一盘棋结束后能得到的训练信息仅是对弈最后的胜负，而我们需要的训练样例是为每个棋盘状态都赋予一个分数。然而一盘棋的最终输赢未必能说明这盘棋每一个中间局面的好坏，所以中间局面训练值具有内在的模糊性。传统的BP网络的反向传播训练算法是不能适应这种学习的。

综上，本文使用了BP网络结构表示评价函数并存储象棋知识，但在网络具体调整权值参数时和传统的BP训练算法不同，使用的是TD(λ)算法的增强式学习方法。

3 TD(λ)算法

3.1 TD(λ)算法在棋类游戏中的应用历史

TD（时间差分学习方法）最初是由Arthur Samuel[2]发明并应用于他的西洋跳棋（checkers）程序中。Richard Sutton[3]在Samuel的方法的基础上又提出了TD(λ)算法。

TD(λ)算法在棋类中的最成功的运用应用当属Gerald Tesauro[4]在1992年开发的西洋双陆棋程序TD-Gammon。TD-Gammon不仅战胜了人类的西洋双陆棋冠军，而且能走出人类过去未曾想到的妙招，它的出现震惊了人工智能领域。虽然TD(λ)算法在西洋双陆棋中评价函数的应用取得了重大成功，但随后在国际象棋[5]和围棋中[6] [7]的研究结果却不理想。这使得研究者们认为TD(λ)算法在西洋双陆棋中之所以取得成功是由于西洋双陆棋的内在特性造成的，而TD(λ)算法在其他棋类中并不适用。TD(λ)算法也被研究者们忽视了数年之久。

近些年来，越来越多的研究者又重新开始关注这种增强学习方法，一些学者在Nine Men's Morris[8]和国际象棋中又取得了许多重要的成果。TD(λ)算法结合人工神经元网络的方法在机器博弈中的应用研究正逐渐升温。

虽然TD(λ)算法在棋类游戏中的应用已有很长的历史，但在中国象棋中的应用研究却仍未有耳闻。

3.2 TD(λ)算法简介

本文采用BP网络表示中国象棋局面的评价函数，采用的Richard Sutton[1]提出的TD(λ)的方法通过学习调整该网络的权值。现以中国象棋为例介绍这种增强学习的方法。

假设X1、X2、X3 …Xm 是一个中间局面序列，使用上一章提到的人工神经元网络评价函数对这些局面进行评价，这时网络的权值取随机值，所以得到的评价值是完全不准确的，假设对应于局面序列的评价值是P1、P2、P_3 … P_m，最终的胜负结果用数值Z（如-1表示黑方获胜，0表示和棋，1表示红方获胜）来表示。这时就可以对人工神经元网络的权值进行调整，调整公式为：

$$w = w + \sum_{t=1}^{m} \Delta w_t \qquad (1)$$

△Wt的计算公式为：

$$\Delta w_t = \alpha(P_{t+1} - P_t)\sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k \qquad (2)$$

其中，α为学习速率，取值范围为(0,1]；$\nabla_w P_k$ 为第K个局面的评价值对该权值的偏导数； λ是为解决“时间信用度”而引入的值，取值范围为[0，1]。λ的引入可以调整第t+1个局面出现的评价差异对1到t个局面评价的修正，当λ为0时，只对第t个局面的评价进行修正，当λ为1时，对1到t个局面的评价做同等程度的修正，当λ的范围为(0,1)时，从第t个局面到第1个局面做λ的指数的衰减修正。

注意，当t= m时，即得出最终胜负结果时，要用 Z-Pt 代替公式中的Pt+1-Pt, 即公式(2)变为：

$$\Delta w_t = \alpha(Z - P_t)\sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k \qquad (3)$$

另外，△Wt还有另一种计算公式：

$$\Delta w_t = \alpha \nabla_w P_t \sum_{k=t}^{m} \lambda^{k-t}(P_{k+1} - P_k) \qquad (4)$$

通过推导可以证明，公式(4)与公式(2)的计算结果虽然不同，但用二者的结果计算出的累加的权值修改量 $\sum_{t=1}^{m} \Delta w_t$ 却是相同的。而公式(4)是必须保存整盘棋的棋盘状态序列并知道最终的胜负结果Z以后，才能计算权值的修改量的。在一盘棋还不知道最终胜负结果Z的时候，公式(2)就可以根据当前下棋的局面，计算权值的修改量，这样就不必保存整盘棋的棋盘状态序列。公式(2)比公式(4)更节省内存。所以在本文的实验中使用的是公式(2)。

3.3 TD(λ)算法的几种改进

传统上，神经元网络在机器博弈评价函数领域都是通过自学习进行训练的。所谓自学习，指的是计算机与自己对弈一盘，产生的棋谱序列用作训练数据，使用TD(λ)算法修正人工神经元网络权值，然后继续对弈下一盘，通过这种对弈，学习，再对弈，再学习的循环过程，不断提高计算机的博弈水平。

TD(λ)算法用于自学习时，标准的TD(λ)算法只搜索一层，或者说没有用到搜索树。如果将TD(λ)算法结合搜索树，便有两种改进，即TD-Directed 和 TD-Leaf。

(1) TD-Directed

TD-Directed算法与标准的TD(λ)算法的区别就在于结合了搜索树，至少搜索两层。

(2) TD-Leaf

TD-Leaf算法也是结合了搜索树，但它的时间差异不是指对弈双方实际走棋得到的棋谱序列中相邻局面间的差异，而指的是搜索引擎在搜索树上用以判断的“关键”局面之间的差异。TD-Leaf算法由Baxter[9][10]等人发明，用于国际象棋程序KnightCap，并取得了不错的效果。

4 获得训练数据

4.1 获得训练数据的几种方法

在确定使用BP网络表示评价函数，采取TD(λ)算法作为学习算法之后，仍然需要大量的训练样本对网络进行训练，机器博弈领域中用于学习的训练数据是大量的棋谱序列，这些序列可由以下几种方法获得。

- **随机对弈**

随机对弈包括两种方法：一种是让被训练的博弈软件与一个能产生随机走法的博弈软件进行对弈；另外一种方法是让能产生随机走法的博弈软件和自己对弈，将其产生的对弈序列作为训练数据。Imran Ghory[11]的研究证明，随机对弈的方法效果不是太好。

- **固定对手对弈**

固定对手对弈是指让计算机与一个固定的对手（可以是人类专家或者另外一个博弈软件）进行对局，在不断的胜利与失败中提高自己。但这种方法的缺点是容易产生出局限于该固定对手的评价函数，当与其它对手对弈时，计算机会出现不适应的情况[12]。

- **自学习**

在3.3中已经对这种方法进行了解释，这种方法目前应用很多，实际上，TD(λ)算法最成功的应用 是Gerald Tesauro的西洋双陆棋程序[4]，它就是通过自学习而达到了很高水平的。但这种方法的缺点是需要进行大量对局，从而需要更多的训练时间。

- **在线学习**

Baxter等人在训练他们的国际象棋程序KnightCap时，使用了这种训练方式。他们把KnightCap放在一个网络服务器上运行，众多棋手通过许多客户端与服务器联网对局。这种在线学习的方法效果不错，通过3天内308盘的对局，KnightCap的等级分由1650（相当于人类的B级水平）增长到2150（相当于人类的大师级）。

- **专家棋谱数据库**

在专家棋谱中蕴藏着大量的象棋知识，直接从专家棋谱中学习的好处是学习的速度快。这表现在两个方面，一方面，因为棋谱已经存在，直接向棋谱学习可以省掉对弈的时间。另一方面，由于对弈双方都是人类专

家，不会轻易出现蠢招，所走过的招法也是代表了中国象棋对弈的精华，可以很快提高机器博弈的水平。

4.2 选择适当的方法

本文重点是研究神经元网络结合TD(λ)算法在中国象棋评价中的应用效果，所以，在选择训练数据的获得方法时，采取了学习速度最快的一种方式，即专家棋谱学习。选择这种方法的好处是，可以尽快学习到基本的博弈知识，从而尽快验证神经元网络结合TD(λ)算法在中国象棋机器博弈中的应用情况。

5 实验

5.1 实验过程

在实验中，选择了2.3中所述的神经元网络结构表示评价函数，即建立了一个91×N×1的BP网络（N为隐藏层的个数，分别取30、50、80、100进行实验），隐藏层与输出层神经元的激活函数均使用双极型的双曲正切函数。网络权值的修改公式采用公式(2)，即标准的TD(λ)算法公式。

训练的棋谱选用的是1990年至2004年全国象棋比赛团体赛、个人赛以及五羊杯赛的对弈棋谱，共7621盘。增强学习系统从棋谱数据库中读入一个对弈序列，然后按照棋谱进行对弈，在对弈中采用TD(λ)算法计算神经元网络所有权值的修改量，在对弈结束后进行权值修改，然后重复以上的对读入棋谱、对弈学习和修改权值的过程，直到所有棋谱学习完毕。对棋谱数据库可以进行多次学习。

5.2 实验结果

为了检验实验的效果，将采用增强学习后的神经元网络作为评价的博弈软件与采用原有方法评价（计算了固定子力值、位置值、灵活度值和威胁与保护值）的博弈软件进行了对弈，为了保证对弈的公平性，均采用相同的搜索引擎，并使用同样的搜索深度。对弈的开局除了使用完整的中国象棋开局外，还随机从棋谱数据库中选择了499盘开局，所以共500个开局，对每个开局双方交替先后手进行对弈，所以总共需要进行1000盘较量。

实验中有两个变量，即神经元网络的隐藏层神经元个数和棋谱数据库的学习次数。另外，神经元网络的初始权值不同，对学习的效果影响也很大，所以为了在同样的初始条件下进行学习和比较，以下设计的几个测试中均采用相同的初始权值。

分别采用30、50、80、100个隐藏层神经元，对棋谱数据库进行一次学习后，与原有博弈软件的对弈结果如表1所示。

表 1 一次学习结果

	胜	平	负	总计
30	9	20	971	1000
50	13	32	955	1000
80	12	52	936	1000
100	19	64	917	1000

分别采用30、50、80、100个隐藏层神经元，对棋谱数据库进行二次学习后，与原有博弈软件的对弈结果如表2所示。

表 2 二次学习结果

	胜	平	负	总计
30	11	29	960	1000
50	26	30	944	1000
80	32	51	917	1000
100	30	65	905	1000

分别采用30、50、80、100个隐藏层神经元，对棋谱数据库进行三次学习后，与原有博弈软件的对弈结果如表3所示。

表 3 三次学习结果

	胜	平	负	总计
30	10	28	962	1000
50	13	32	955	1000
80	37	45	918	1000
100	28	55	917	1000

从以上三个表中可以看出：神经元网络的隐藏层神经元个数对学习的效果有一定影响，随着隐藏层神经元个数的增加，棋力有增强的趋势，这体现在三个表中输棋的盘数呈递减趋势，例如在表1中，对应不同隐藏层个数的输棋盘数依次为971、955、936、917。其它两表中也有类似的趋势。但这种棋力的增强并没有完全转化为胜利的盘数，从三个表中可以看出，主要表现在和棋的盘数呈增长趋势。

把以上的三个表换个角度看，即以对棋谱数据库学习的次数的角度看，可以得到以下四个表：

采用30个隐藏层神经元，对棋谱数据库进行分别进行1、2、3次学习后，与原有博弈软件的对弈结果如表4所示。

表 4 30 个隐藏层的学习结果

	胜	平	负	总计
1	9	20	971	1000
2	11	29	960	1000
3	10	28	962	1000

采用50个隐藏层神经元，对棋谱数据库进行分别进行1、2、3次学习后，与原有博弈软件的对弈结果如表5所示。

表 5 50 个隐藏层的学习结果

	胜	平	负	总计
1	13	32	955	1000
2	26	30	944	1000
3	13	32	955	1000

采用80个隐藏层神经元，对棋谱数据库进行分别进行1、2、3次学习后，与原有统博弈软件的对弈结果如表6所示。

表 6 80 个隐藏层的学习结果

	胜	平	负	总计
1	12	52	936	1000
2	32	51	917	1000
3	37	45	918	1000

采用100个隐藏层神经元，对棋谱数据库进行分别进行1、2、3次学习后，与原有博弈软件的对弈结果如表7所示。

表 7 100 个隐藏层的学习结果

	胜	平	负	总计
1	19	64	917	1000
2	30	65	905	1000
3	28	55	917	1000

从以上四个表中可以看出：随着对棋谱数据库学习次数的增加，棋力不是呈正比的增强的，从第一次学习到第二次学习棋力有增强的表现，但进行第三次学习后，战绩有下滑趋势。

从以上的实验也可以看到，自学习可以有效的提高博弈水平，但和采用传统方法的评价函数相比还有不小差距。这并不是出乎意料的结果，原因是影响增强学习效果的因素有很多，本文只是考虑了最基本的几个因素以进行初步的实验，从而验证这种方法的可行性。

6 结论

本文探讨了如何应用神经元网络中的BP网络结合TD(λ)算法，进行中国象棋机器博弈中评价函数的自学习增强。并设计了几个实验对自学习的效果进行了验证，实验结果证明了采用这种人工智能的学习评价的方法，对提高中国象棋机器博弈的水平具有可行性。

下一步要做的工作仍有许多，需要深入研究各种影响学习效果的因素，以后的改进将集中在本文提到的三个方面：神经元网络的优化、TD(λ)算法的改进以及选择合适的获得训练数据的方式等。

参考文献

[1] L.V. Allis, Searching for solutions in games and artificial intelligence. Ph.D. Thesis, University of Limburg, The Netherlands, ISBN 90-9007488-0 (1994)

[2] Arthur L. Samuel (1959) Some studies in machine learning using the game of checkers. IBM Journal of Research and Development 3, 210-229.

[3] Richard S. Sutton (1988) Learning to predict by the methods of temporal difference. Machine Learning 3, 9-44.

[4] Gerald Tesauro (1992) Practical issues in temporal difference learning. Machine Learning 4, 257-277.

[5] Henk Mannen and Marco Wiering (2004) Learning to play chess using TD(λ)-learning with database games. Benelearn'04: Proceedings of the Thirteenth Belgian-Dutch Conference on Machine Learning.

[6] Peter Dayan, Nicol N. Schraudolph, and Terrence J. Sejnowski (2001) Learning to evaluate Go positions via temporal difference methods. Computational Intelligence in Games, Springer Verlag, 74-96.

[7] R. Ekker, E.C.D. van der Werf, and L.R.B. Schomaker (2004) Dedicated TD-Learning for Stronger Gameplay: applications to Go. Benelearn'04: Proceedings of the Thirteenth Belgian-Dutch Conference on Machine Learning.

[8] Thomas Ragg, Heinrich Braunn and Johannes Feulner (1994) Improving Temporal Difference Learning for Deterministic sequential Decision Problems. Proceedings of the International Conference on Artificial Neural Networks- ICANN '95, 117-122.

[9] Jonathan Baxter, Andrew Tridgell, and Lex Weaver (1997) KnightCap: A chess program that learns by combining TD(λ) with minimax search. Technical Report, Learning Systems Group, Australian National University.

[10] Jonathan Baxter, Andrew Tridgell, and Lex Weaver (1998) TDLeaf(λ): Combining Temporal Difference Learning with Game-Tree Search. Australian Journal of Intelligent Information Processing Systems (Autumn 1998), 39-43.

[11] Imran Ghory, Reinforcement Learning in Board Games. Volume, Department of Computer Science, University of Bristol. May 2004.

[12] D. F. Beal (2002) Learn from your opponent - but what if he/she/it knows less than you？. Step by Step. Proceedings of the 4th colloquium “Board Games in Academia”.