(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 111330255 B
(45)授权公告日 2021.06.08

(21)申请号 202010226567.6
(22)申请日 2020.03.27
(65)同一申请的已公布的文献号
申请公布号 CN 111330255 A
(43)申请公布日 2020.06.26
(66)本国优先权数据
202010046222.2 2020.01.16 CN
(73)专利权人 北京理工大学
地址 100081 北京市海淀区中关村南大街5号
(72)发明人 施重阳 廖兆和 柴增豪
(74)专利代理机构 北京正阳理工知识产权代理事务所(普通合伙) 11639
代理人 王民盛
(51)Int.Cl.
A63F 3/02(2006.01)
G06N 3/04(2006.01)
G06N 3/08(2006.01)

(56)对比文件
CN 109032935 A,2018.12.18
CN 110119804 A,2019.08.13
CN 110555517 A,2019.12.10
CN 109923500 A,2019.06.21
CN 109726721 A,2019.05.07
CN 109146067 A,2019.01.04
WO 2017214968 A1,2017.12.21
王力."价值神经网络在计算机围棋的优化研究".《中国优秀硕士学位论文全文数据库》.2018,(第11期),全文.

审查员 倪晨辉

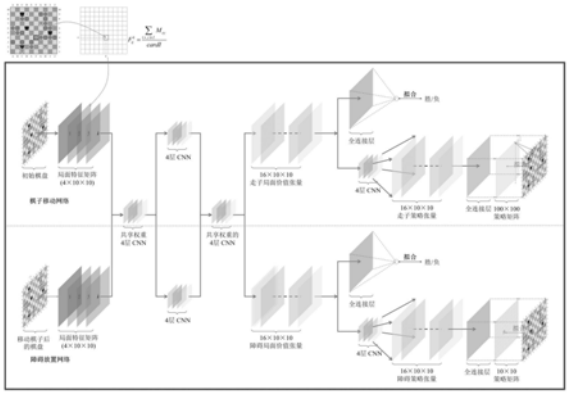
权利要求书2页 说明书13页 附图7页

(54)发明名称

一种基于深度卷积神经网络的亚马逊棋招法生成方法

(57)摘要

本发明涉及一种基于深度卷积神经网络的亚马逊棋招法生成方法,属于人工智能技术领域。本发明包含深度网络模型,网络模型训练,以及最优招法生成三部分。深度网络模型包括棋子移动网络和障碍放置网络。网络模型训练使用RMSProp算法优化网络权重,通过监督学习的方式训练网络。最优招法生成通过组合各网络的输出指导亚马逊棋智能系统生成可靠的最优招法及预测的胜率。对比现有技术,本发明方法的招法生成不完全依赖于人类先验知识,具有效率稳定,不受极端局面影响的优势;采用分步决策的方式,提高了招法生成效率和准确率;通过共享部分网络层的方式,体现了各决策步骤的相似性和连续性。



CN 111330255 B

1.一种基于深度卷积神经网络的亚马逊棋招法生成方法,其特征在于,包括以下内容:

针对当前棋局,使用经训练的网络模型生成最优招法;

所述网络模型包括棋子移动网络和障碍放置网络,其中棋子移动网络用于生成棋子移动前后的坐标,障碍放置网络用于生成放置障碍的坐标;

(1)棋子移动网络包括走子局面价值子网和走子价值-策略转换子网,走子局面价值子网对当前局面特征矩阵F经过12层卷积层和3层Dropout层处理后得到走子局面价值张量,3层Dropout层分别处于第4、8、12层卷积层之后;走子价值-策略转换子网包含两个部分,第一部分将走子局面价值张量经过1层全连接层得到表示预估胜率的标量,第二部分包含4层卷积层和1层全连接层,走子局面价值张量经过4层卷积层后得到走子策略张量,走子策略张量经过1层全连接层后输出策略矩阵P,其中每一个元素都代表一种走法;在将P中所有非法走法所对应的概率置零后进行softmax操作,以得到所有非零元素均代表合法走法的策略矩阵P_{valid},选取P_{valid}中概率最大的一点(x^*,y^*)通过下式映射为棋子走法:

$$\begin{cases}(x^*\%10+1,x^*/10+1)=(\hat{x}_1,\hat{y}_1)\\(y^*\%10+1,y^*/10+1)=(\hat{x}_2,\hat{y}_2)\end{cases};$$

其中,"/"表示整除运算,"%"表示取余运算,且棋盘坐标A,…,J与1,…,10一一对应,$(\hat{x}_1,\hat{y}_1)$为预测的棋子坐标,$(\hat{x}_2,\hat{y}_2)$为预测的落子坐标;

F由F^k组成,k∈{1,2,3,4},F^k中元素通过如下公式计算:

$$F_{ij}^k=\frac{\sum_{(x,y)\in I}M_{xy}}{cardI};$$

其中,局面特征矩阵F^k的第ij号元素的值F_{ij}^k表示为:在棋盘矩阵中,以第k号棋子所在位置(k_i,k_j)与目标位置(i,j)为对角线所围成的矩形区域I各元素值之和的均值;M_{xy}表示棋盘矩阵M中位置(x,y)处的元素值;cardI表示集合I中元素的个数;

根据F^k的计算公式分别计算各个棋子所对应的局面特征矩阵得到矩阵组F,作为神经网络的输入;

棋子移动网络的损失函数通过如下公式计算:

$$Loss=L_1^2+L_2^2;$$

L_1为走子局面价值张量经过全连接层处理得到的预测结果误差,具体通过如下公式计算:

$$L_1=\sqrt{(y-\hat{y})^2}=|y-\hat{y}|;$$

其中$\hat{y}\in[0,1]$为棋子移动网络给出的预测值,y∈{0,1}为实际标签值;

L_2为棋子移动的预测误差,具体通过如下公式计算:

$$L_2=\sqrt{(x_1-\hat{x}_1)^2+(y_1-\hat{y}_1)^2}+\sqrt{(x_2-\hat{x}_2)^2+(y_2-\hat{y}_2)^2};$$

其中(x_1,y_1)为数据样本选择的棋子坐标,(x_2,y_2)为数据样本选择的落子坐标;$(\hat{x}_1,\hat{y}_1)$

为预测的棋子坐标，$(\hat{x}_2,\hat{y}_2)$为预测的落子坐标；

(2)障碍放置网络包括障碍局面价值子网和障碍价值-策略转换子网，障碍局面价值子网对移动棋子后的局面特征矩阵F′经过12层卷积层和3层Dropout层处理后得到障碍局面价值张量，3层Dropout层分别处于第4、8、12层卷积层之后；障碍价值-策略转换子网包含两个部分，第一部分将障碍局面价值张量经过1层全连接层得到表示预估胜率的标量，第二部分包含4层卷积层和1层全连接层，障碍局面价值张量经过4层卷积层后得到障碍策略张量，障碍策略张量经过1层全连接层后输出策略矩阵P′，其中每一个元素都代表一种放置障碍的坐标；在将P′中所有非法障碍坐标所对应的概率置零后进行softmax操作，以得到所有非零元素均代表合法坐标的策略矩阵P'_{valid}，选取P'_{valid}中概率最大的一点(x'^*,y'^*)作为放置障碍物的坐标$(\hat{x}_3,\hat{y}_3)$；其中，F′的计算方式同F的计算方法；

障碍放置网络的损失函数通过如下公式计算：

$$Loss' = L_1'^2 + L_2'^2;$$

L'_1为障碍局面价值张量经过全连接层处理得到的预测结果误差，具体通过如下公式计算：

$$L_1' = \sqrt{(y' - \hat{y}')^2} = |y' - \hat{y}'|;$$

其中$\hat{y}' \in [0,1]$为障碍放置网络给出的预测值，$y' \in \{0,1\}$为实际标签值；

L'_2为障碍放置坐标的预测误差，具体通过如下公式计算：

$$L_2' = \sqrt{(x_3 - \hat{x}_3)^2 + (y_3 - \hat{y}_3)^2};$$

其中$(\hat{x}_3,\hat{y}_3)$为预测的障碍坐标，(x_3,y_3)为数据样本实际障碍坐标。

2.根据权利要求1所述的方法，其特征在于，所述Dropout层的丢弃概率为0.3。

3.根据权利要求1所述的方法，其特征在于，所述棋子移动网络的走子局面价值子网的前、后4层卷积层与障碍放置网络的障碍局面价值子网的前、后4层卷积层分别共享权重。

4.根据权利要求1-3任一所述的方法，其特征在于，所述网络模型训练时采用使用Nesterov动量的RMSprop算法对参数进行调整。

一种基于深度卷积神经网络的亚马逊棋招法生成方法

技术领域

[0001] 本发明涉及一种基于深度卷积神经网络的亚马逊棋招法生成方法，属于人工智能技术领域，按照国际专利分类表(IPC)属于人类生活必须部，保健；救生；娱乐分部。主要采用深度卷积神经网络缓解亚马逊棋智能系统的招法生成过分依赖人类先验知识的局限性，指导其更加高效地生成可靠的最优招法。

背景技术

[0002] 计算机问世使得人类的计算能力得到了飞跃性的提升。计算机虽然能够提供强大的计算能力，却无法像人类一样在复杂环境中做出决策。为了解决这类问题，人工智能领域应运而生。

[0003] 深度学习与大数据的兴起带来了人工智能的爆发，作为人工智能领域的重要研究方向之一，计算机博弈能够帮助人们更加深入地研究和理解人工智能。亚马逊棋作为一种计算机博弈的重要形式，结合了围棋与国际象棋的行棋规则，具有招法多样，棋局复杂的特点。

[0004] 传统的亚马逊棋招法生成方法建立在人类现有的棋类知识的基础上，根据对亚马逊棋的理解手动编写估值函数，对现有局面优劣进行估值判断。这种方式难以高效的生成准确的最优招法，并且仅仅依靠估值函数一定程度上限制了亚马逊棋智能系统的棋力，因而，有必要寻找一种更为高效的招法生成方法，生成亚马逊棋博弈过程中的最优招法。

[0005] 本发明针对现有亚马逊棋招法生成方法的局限性，创新性地在亚马逊棋领域引入深度卷积神经网络，使得亚马逊棋招法生成不完全依赖于人类先验知识。

[0006] 本发明中涉及的相关技术如下：

[0007] 1.计算机博弈

[0008] 计算机博弈是人工智能领域的重要研究方向，作为机器智能、兵棋推演、智能决策系统等人工智能领域的重要科研基础，计算机博弈被认为是人工智能领域最具挑战性的研究方向之一。

[0009] 亚马逊棋(Game of the Amazons)是在1988年推出的两人棋类博弈，是奥林匹亚电脑游戏程式竞赛的比赛指定棋类，由于局面过于复杂，每一步可行的招法可高达数千种，故该棋类多不用于人类之间比赛，而是用于计算机博弈相关方面的比赛与研究。

[0010] 亚马逊棋的研究及实现涉及编程语言，算法思想，博弈思想等，常用的算法有蒙特卡洛算法，退火算法，遗传算法等。随着机器学习与深度学习的发展，亚马逊棋作为计算机博弈的一个项目正逐渐的被更广泛的熟知。

[0011] 2.深度学习

[0012] 深度学习(Deep Learning)是机器学习(Machine Learning)的分支，是一种以人工神经网络为架构，对数据进行表征学习的算法。深度学习的优势之一是其采用了非监督式或半监督式的特征学习和分层特征的方式，替代了手工获取样本特征的方式。

[0013] 深度学习被广泛应用于学习样本数据的内在规律和表示层次，这些学习过程中获

得的信息对文本、图像、音频等数据的解释有很大的帮助。

[0014] 深度学习在数据挖掘,计算机视觉,自然语言处理等诸多领域都取得丰富的成果。深度学习使机器模仿视听和思考等人类的活动,解决了很多复杂的模式识别难题,使得人工智能相关技术取得了巨大进步。其最终目标是让机器能够像人一样具有分析学习能力,能够识别文字、图像、音频等数据。作为一个复杂的机器学习算法,深度学习在语音和图像识别方面取得的效果,远远超过先前相关技术。

[0015] 3.卷积神经网络

[0016] 卷积神经网络(Convolutional Neural Network)是一类包含卷积计算且具有深度结构的前馈神经网络,是深度学习的代表算法之一。卷积神经网络具有表征学习能力,能够按其阶层结构对输入信息进行平移不变分类,因此也被称为“平移不变人工神经网络”。

[0017] 卷积神经网络模仿生物的视知觉机制设计构建,可以进行监督学习和非监督学习,其隐藏层内的卷积核参数共享和层间连接的稀疏性使得卷积神经网络能够以较小的计算量对格点化特征(例如像素、音频)进行学习,并且有稳定的效果且对数据没有额外的特征工程要求。

[0018] 卷积神经网络的相关研究始于二十世纪80至90年代,时间延迟网络和LeNet-5是最早出现的卷积神经网络;在二十一世纪后,随着深度学习理论的提出和数值计算设备的改进,卷积神经网络得到了快速发展,并被广泛应用于各种计算机视觉、自然语言处理等任务,并取得了巨大的成果。

[0019] 卷积神经网络由一个或多个卷积层(Convolution Layer)和顶端的全连通层(Dense Layer)组成,同时也包括共享权重和池化层(Pooling Layer)。这一结构使得卷积神经网络能够充分利用输入数据的二维结构进行特征的提取,与其他深度学习结构相比,卷积神经网络在图像和语音识别方面能够给出更好的结果。卷积神经网络可以使用反向传播算法(Back Propagation)进行训练。相比较其他深度、前馈神经网络,卷积神经网络需要训练的参数更少。

发明内容

[0020] 本发明的目的是基于深度卷积神经网络,针对亚马逊棋的规则和先验知识,设计适用于亚马逊棋招法生成的网络结构,并基于这一网络结构实现基于深度卷积神经网络的亚马逊棋招法生成方法,从而使得亚马逊棋智能系统能在博弈对局过程中生成己方的最优招法。

[0021] 本发明的目的是通过以下技术方案实现的:

[0022] 一种基于深度卷积神经网络的亚马逊棋招法生成方法,包括以下内容:

[0023] 针对当前棋局,使用经训练的网络模型生成最优招法(棋子移动前后的坐标以及放置障碍的坐标);

[0024] 所述网络模型包括棋子移动网络和障碍放置网络,其中棋子移动网络用于生成棋子移动前后的坐标,障碍放置网络用于生成放置障碍的坐标;

[0025] (1)棋子移动网络包括走子局面价值子网和走子价值-策略转换子网,走子局面价值子网对当前局面特征矩阵F经过12层卷积层和3层Dropout层处理后得到走子局面价值张量,3层Dropout层分别处于第4、8、12层卷积层之后;走子价值-策略转换子网包含两个部

分，第一部分将走子局面价值张量经过1层全连接层得到表示预估胜率的标量，第二部分包含4层卷积层和1层全连接层，走子局面价值张量经过4层卷积层后得到走子策略张量，走子策略张量经过1层全连接层后输出策略矩阵P，其中每一个元素都代表一种走法；在将P中所有非法走法所对应的概率置零后进行softmax操作，以得到所有非零元素均代表合法走法的策略矩阵P_{valid}，选取P_{valid}中概率最大的一点(x^*,y^*)通过下式映射为棋子走法：

[0026] $$\begin{cases}(x^*\%10+1,x^*/10+1)=(\hat{x}_1,\hat{y}_1)\\(y^*\%10+1,y^*/10+1)=(\hat{x}_2,\hat{y}_2)\end{cases};$$

[0027] 其中，“/”表示整除运算，“%”表示取余运算，且棋盘坐标A,…,J与1,…,10一一对应，$(\hat{x}_1,\hat{y}_1)$为预测的棋子坐标，$(\hat{x}_2,\hat{y}_2)$为预测的落子坐标；

[0028] F由F^k组成，$k\in\{1,2,3,4\}$，F^k中元素通过如下公式计算：

[0029] $$F_{ij}^k=\frac{\sum_{(x,y)\in I}M_{xy}}{cardI};$$

[0030] 其中，局面特征矩阵F^k的第ij号元素的值F_{ij}^k表示为：在棋盘矩阵中，以第k号棋子所在位置(k_i,k_j)与目标位置(i,j)为对角线所围成的矩形区域I各元素值之和的均值；M_{xy}表示棋盘矩阵M中位置(x,y)处的元素值；cardI表示集合I中元素的个数；

[0031] 棋子移动网络的损失函数通过如下公式计算：

[0032] $$Loss=L_1^2+L_2^2;$$

[0033] L_1为走子局面价值张量经过全连接层处理得到的预测结果误差，具体通过如下公式计算：

[0034] $$L_1=\sqrt{(y-\hat{y})^2}=|y-\hat{y}|;$$

[0035] 其中$\hat{y}\in[0,1]$为棋子移动网络给出的预测值，$y\in\{0,1\}$为实际标签值；

[0036] L_2为棋子移动的预测误差，具体通过如下公式计算：

[0037] $$L_2=\sqrt{(x_1-\hat{x}_1)^2+(y_1-\hat{y}_1)^2}+\sqrt{(x_2-\hat{x}_2)^2+(y_2-\hat{y}_2)^2};$$

[0038] 其中(x_1,y_1)为数据样本选择的棋子坐标，(x_2,y_2)为数据样本选择的落子坐标；

[0039] (2)障碍放置网络包括障碍局面价值子网和障碍价值-策略转换子网，障碍局面价值子网对移动棋子后的局面特征矩阵F′经过12层卷积层和3层Dropout层处理后得到障碍局面价值张量，3层Dropout层分别处于第4、8、12层卷积层之后；障碍价值-策略转换子网包含两个部分，第一部分将障碍局面价值张量经过1层全连接层得到表示预估胜率的标量，第二部分包含4层卷积层和1层全连接层，障碍局面价值张量经过4层卷积层后得到障碍策略张量，障碍策略张量经过1层全连接层后输出策略矩阵P′，其中每一个元素都代表一种放置障碍的坐标；在将P′中所有非法障碍坐标所对应的概率置零后进行soft max操作，以得到所有非零元素均代表合法坐标的策略矩阵P'_{valid}，选取P'_{valid}中概率最大的一点(x'^*,y'^*)作为放置障碍物的坐标$(\hat{x}_3,\hat{y}_3)$；其中，F′的计算方式同F的计算方法；

[0040] 障碍放置网络的损失函数通过如下公式计算：

[0041] $Loss' = L_1'^2 + L_2'^2$；

[0042] L'_1为障碍局面价值张量经过全连接层处理得到的预测结果误差，具体通过如下公式计算：

[0043] $L_1' = \sqrt{(y' - \hat{y}')^2} = |y' - \hat{y}'|$；

[0044] 其中$\hat{y}' \in [0,1]$为障碍放置网络给出的预测值，$y' \in \{0,1\}$为实际标签值；

[0045] L_2'为障碍放置坐标的预测误差，具体通过如下公式计算：

[0046] $L_2' = \sqrt{(x_3 - \hat{x}_3)^2 + (y_3 - \hat{y}_3)^2}$；

[0047] 其中$(\hat{x}_3, \hat{y}_3)$为预测的障碍坐标，(x_3, y_3)为数据样本实际障碍坐标。

[0048] 作为优选，所述Dropout层的丢弃概率为0.3。

[0049] 作为优选，所述棋子移动网络的走子局面价值子网与障碍放置网络的障碍局面价值子网的前、后4层卷积层共享权重。

[0050] 作为优选，所述网络模型训练时采用使用Nesterov动量的RMSprop算法对参数进行调整。

[0051] 有益效果

[0052] 对比现有的亚马逊棋招法生成方法，本发明方法具有以下优势：

[0053] (1)本发明方法相对现有的基于博弈树的招法生成方法而言，其招法生成不完全依赖于人类先验知识，具有效率稳定，不受极端局面影响的优势。

[0054] (2)本发明方法相对现有的单步决策网络而言，采用分步决策的方式，缓和了策略矩阵元素个数过多、矩阵稀疏等问题，提高了招法生成效率和准确率。

[0055] (3)本发明方法通过两个不同的决策过程共享部分网络层的方式，体现各决策步骤的相似性和连续性，模型的表达能力和泛化能力得到提高。

附图说明

[0056] 图1为本发明网络结构示意图；

[0057] 图2为局面特征矩阵提取模块示意图；

[0058] 图3为棋子移动网络(Move-Net)结构示意图；

[0059] 图4为走子局面价值子网结构示意图；

[0060] 图5为走子价值-策略转换子网结构示意图；

[0061] 图6为障碍放置网络(Arrow-Net)结构示意图；

[0062] 图7为障碍局面价值子网结构示意图；

[0063] 图8为障碍价值-策略转换子网结构示意图；

[0064] 图9为初始棋盘局面示例；

[0065] 图10为移动棋子并放置障碍后的棋盘局面示例。

具体实施方式

[0066] 为了便于理解本发明的技术方案,现对本项发明网络模型的训练过程做进一步的说明。

[0067] 具体而言,一种基于深度卷积神经网络的亚马逊棋招法生成方法由深度网络模型,网络模型训练,以及最优招法生成三部分组成。

[0068] 1.深度网络模型

[0069] 如图1所示,一种基于卷积神经网络的亚马逊棋网络结构(Amazons-Net)包含两个相互关联的网络,分别为棋子移动网络(Move-Net)和障碍放置网络(Arrow-Net),分别如图3和图6所示。

[0070] 棋子移动网络:

[0071] 如图2所示,网络中局面特征提取单元用于提取每个己方棋子的局面特征,在棋盘矩阵M中,针对第k号棋子,其对应的局面特征矩阵F^k计算方式如公式(1)所示:

[0072] $$F_{ij}^{k}=\frac{\sum_{(x,y)\in I} M_{xy}}{cardI} \qquad (1)$$

[0073] 局面特征矩阵F^k的第ij号元素的值F_{ij}^{k}表示为:在棋盘矩阵中,以第k号棋子所在位置(k_i,k_j)与目标位置(i,j)为对角线所围成的矩形区域I各元素值之和的均值,k∈{1,2,3,4};M_{xy}表示棋盘矩阵M中位置(x,y)处的元素值,cardI表示集合I基数。根据公式(1),分别计算各个棋子所对应的局面特征矩阵,得到4×10×10的矩阵组F,作为神经网络的输入。

[0074] 棋子移动网络结构图如图3所示,棋子移动网络包括走子局面价值子网和走子价值-策略转换子网,分别如图4和图5所示。在走子局面价值子网中,对输入的局面特征矩阵F,它将经过12个卷积层和3个丢弃概率为0.3的Dropout层。各个Dropout层分别处于第4,8,12个卷积层之后。

[0075] 走子局面价值子网各卷积层的参数见表1。

[0076] 表1走子局面价值子网各卷积层的参数

[0077]

层数	卷积核大小	滤波器数目	激活函数	输出张量形状
1	6×6	128	ReLu	128×10×10
2	5×5	64	ReLu	64×10×10
3	4×4	32	tanh	32×10×10
4	2×2	16	tanh	16×10×10
5	4×4	128	ReLu	128×10×10
6	3×3	128	ReLu	128×10×10
7	3×3	64	tanh	64×10×10
8	2×2	64	tanh	64×10×10
9	3×3	64	ReLu	64×10×10
10	4×4	64	ReLu	64×10×10
11	5×5	32	tanh	32×10×10
12	6×6	16	tanh	16×10×10

[0078] 其中，第1,2,3,4,9,10,11,12个卷积层与障碍放置网络中的第1,2,3,4,9,10,11,12个卷积层共享全部权重。

[0079] 之后将走子局面价值子网输出的走子局面价值张量输入走子价值-策略转换子网，进行两部分并行操作：

[0080] 第一部分，将走子局面价值张量传入全连接层，输出一个标量，并对数据样本中当前局面对应的胜负情况进行拟合，这一部分的误差记为：

[0081] $$L_1=\sqrt{(y-\hat{y})^2}=|y-\hat{y}| \tag{2}$$

[0082] 其中$\hat{y}\in[0,1]$为网络给出的预测值，$y\in\{0,1\}$为实际标签值。

[0083] 第二部分，将走子局面价值张量输入4层卷积层和1层全连接层，输出100×100的策略矩阵P，策略矩阵中的每一个元素都代表一种走法。

[0084] 走子价值-策略转换子网各卷积层的参数见表2：

[0085] 表2走子价值-策略转换子网各卷积层的参数

[0086]

层数	卷积核大小	滤波器数目	激活函数	输出张量形状
1	2×2	128	ReLu	128×10×10
2	2×2	64	ReLu	64×10×10
3	2×2	32	tanh	32×10×10
4	2×2	16	tanh	16×10×10

[0087] 需要注意的是：网络所给出策略矩阵P中，概率最大值所对应的棋子走法并不一定是合法走法。所以，在将策略矩阵最大值的坐标转化为棋子走法时，需要筛选合法走法，将所有非法走法所对应的概率置零，之后再进行softmax操作，以得到所有非零单元均代表合法走法的策略矩阵P_{valid}。选取P_{valid}中概率最大的一点(x^*,y^*)：

[0088] $$(x^*,y^*)=\arg\max_{(x,y)\in P}P_{valid}(x,y) \tag{3}$$

[0089] 公式(3)中，(x^*,y^*)隐式地表示神经网络所预测的棋子走法，(x^*,y^*)和棋子走子的映射关系为：

[0090] $$\begin{cases}(x^*\%10+1,x^*/10+1)=(\hat{x}_1,\hat{y}_1)\\(y^*\%10+1,y^*/10+1)=(\hat{x}_2,\hat{y}_2)\end{cases} \tag{4}$$

[0091] 其中，“/”表示整除运算，“%”表示取余运算，且棋盘坐标A,…,J与1,…,10一一对应，$(\hat{x}_1,\hat{y}_1)$为预测的棋子坐标，$(\hat{x}_2,\hat{y}_2)$为预测的落子坐标。

[0092] 记(x_1,y_1)为数据样本选择的棋子坐标，(x_2,y_2)为数据样本选择的落子坐标。(x^*,y^*)用于拟合训练数据给出的棋子选择和落子坐标。这部分误差定义为：

[0093] $$L_2=\sqrt{(x_1-\hat{x}_1)^2+(y_1-\hat{y}_1)^2}+\sqrt{(x_2-\hat{x}_2)^2+(y_2-\hat{y}_2)^2} \tag{5}$$

[0094] 最终的训练误差定义为：

[0095] $$Loss=L_1^2+L_2^2 \tag{6}$$

[0096] 障碍放置网络：

[0097] 障碍放置网络结构图如图6所示，下分为障碍局面价值子网和障碍价值-策略转换子网，分别如图7和图8所示。障碍放置网络在卷积层与棋子移动网络共享部分权重，以体现亚马逊棋移动棋子和放置障碍过程的连续性，并通过共享权重减少训练过程中网络参数的数量。

[0098] 在障碍放置网络中，输入的棋盘矩阵不再是初始棋盘，而是正确移动亚马逊棋子后的局面，与棋子选择网络的局面特征提取操作相同，根据公式(1)计算棋盘当前局面特征矩阵F′，将其作为障碍放置网络的输入。

[0099] 在障碍局面价值子网中，对局面特征矩阵F′，它将经过12个卷积层和3个丢弃概率为0.3的Dropout层。各个Dropout层分别处于第4,8,12个卷积层之后。

[0100] 障碍局面价值子网各卷积层的参数见表3。

[0101] 表3障碍局面价值子网各卷积层的参数

[0102]

层数	卷积核大小	滤波器数目	激活函数	输出张量形状

[0103]

1	6×6	128	ReLu	128×10×10
2	5×5	64	ReLu	64×10×10
3	4×4	32	tanh	32×10×10
4	2×2	16	tanh	16×10×10
5	4×4	128	ReLu	128×10×10
6	3×3	128	ReLu	128×10×10
7	3×3	64	tanh	64×10×10
8	2×2	64	tanh	64×10×10
9	3×3	64	ReLu	64×10×10
10	4×4	64	ReLu	64×10×10
11	5×5	32	tanh	32×10×10
12	6×6	16	tanh	16×10×10

[0104] 其中，第1,2,3,4,9,10,11,12个卷积层与棋子移动网络中的第1,2,3,4,9,10,11,12个卷积层共享全部权重。

[0105] 之后将障碍局面价值子网输出的障碍局面价值张量输入障碍价值-策略转换子网，进行两部分并行操作：

[0106] 第一部分，将障碍局面价值张量传入全连接层，输出一个标量，并对数据样本中当前局面对应的胜负情况进行拟合。这一部分的误差记为：

[0107] $$L_1' = \sqrt{(y' - \hat{y}')^2} = |y' - \hat{y}'| \qquad (7)$$

[0108] 其中$\hat{y}' \in [0,1]$为网络给出的预测值，$y' \in \{0,1\}$为实际标签值。

[0109] 第二部分，将障碍局面价值张量输入4层卷积层和1层全连接层，输出10×10的策略矩阵P′，策略矩阵中的每一个元素均代表一种放置障碍的坐标。

[0110] 障碍价值-策略转换子网各卷积层的参数见表4：

[0111] 表4障碍价值-策略转换子网各卷积层的参数

[0112]

层数	卷积核大小	滤波器数目	激活函数	输出张量形状

1	2×2	128	ReLu	128×10×10
2	2×2	64	ReLu	64×10×10
3	2×2	32	tanh	32×10×10
4	2×2	16	tanh	16×10×10

[0113] 与棋子移动网络相同，给出策略矩阵P′中，概率最大者所代表的障碍坐标并不一定是合法走法。所以，在将策略矩阵最大值的坐标转化为障碍坐标时，也需要筛选合法坐标，将所有非法坐标所对应的概率置零。之后再进行softmax操作，以得到所有非零单元均代表合法坐标的策略矩阵P'_{valid}。选取P'_{valid}中概率最大的一点：

[0114] $$(x'^*,y'^*)=\arg\max_{(x',y')\in P} P'_{valid}(x',y') \tag{8}$$

[0115] 公式(8)中，(x'^*,y'^*)表示神经网络所预测的放置障碍物的坐标$(\hat{x}_3,\hat{y}_3)$，定义该部分误差为：

[0116] $$L'_2=\sqrt{(x_3-\hat{x}_3)^2+(y_3-\hat{y}_3)^2} \tag{9}$$

[0117] 其中，$(\hat{x}_3,\hat{y}_3)$为预测的障碍坐标，(x_3,y_3)为数据样本实际障碍坐标。

[0118] 最终的训练误差定义为：

[0119] $$Loss'=L_1'^2+L_2'^2 \tag{10}$$

[0120] 2.网络模型训练

[0121] 网络模型训练过程可使用现有的各种梯度下降方法对网络模型参数进行调整，如SGD，Adadelta，Adagrad等，本例采用基于Nesterov动量的RMSprop算法对网络模型参数进行调整，使得模型最终的输出结果(包括招法信息和胜率信息)尽可能的与训练集数据拟合。

[0122] 通过RMSProp算法的充分训练，可以得到模型预测结果与训练集尽可能一致的网络模型，将得到的模型进行保存。通过使用tf.estimator.Estimator对象的Estimator.train()函数，在训练的过程中自动保存模型文件。保存的模型包含权重和数据流图的结构。保存的网络模型将用于亚马逊棋智能系统搜索模块最优招法的评估与生成。

[0123] 3.最优招法生成

[0124] 最优招法生成包括模型导入，棋盘局面输入，最优招法输出三个部分。

[0125] 在构建tf.estimator.Estimator对象时传入模型文件路径，将经过RMSProp算法充分训练后保存的模型导入亚马逊棋智能系统搜索算法模块中，构建相应的网络模型，并设置对应权重。

[0126] 将棋盘矩阵作为网络模型的输入，按照网络模型的结构，将初始的棋盘矩阵输入棋子移动网络，计算得到棋子移动前后的坐标。使用移动棋子后的局面作为障碍放置网络的输入，计算得到放置障碍的坐标。将棋子移动网络和障碍放置网络的计算结果合并，作为招法输出，并给出该招法的胜率。

[0127] 经过网络模型的计算，得到了模型预测的最优招法及其胜率。通过在亚马逊棋智能系统上运行走子命令，执行网络模型预测的最优招法。

[0128] 实验结果：

[0129] 1.训练数据集的获取

[0130] 训练数据来自http://littlegolem.net/网页中亚马逊棋的对局记录，Little Golem是一个在线回合制棋类游戏对局系统，棋类对局通过对弈双方在浏览器中进行，通过网络爬虫获取并记录亚马逊棋的历史对局信息，总共获得25000局的完整的对局信息，包括对局双方的走子记录，对局双方的胜负情况。

[0131] 2.数据预处理

[0132] 获取的25000局对局中存在部分无效的对局信息，首先需要对无效的对局信息进行筛选(包括未分出胜负的对局、走子数过少的对局等)。由于训练数据只包含全局走子信息以及最终胜负信息，需要先对数据集进行预处理，得到单步招法及其对应局面以及胜负标签。为了便于加快网络训练、数据读写，将数据修改并存储为TFRecord格式。数据的存储格式如下：

[0133] 表5训练数据存储格式

[0134]

名称	数据类型	数据大小	描述
Board	Int8	10×10	当前待走子的棋盘矩阵
Type	String	Scalar	当前行动模式：移动棋子/放置障碍，取值为“Amazon”或“Arrow”
From	Int8	10×10	Type 为“Amazon”时表示移动的棋子坐标，为“Arrow”时置-1
To	Int8	10×10	Type 为“Amazon”时表示落子点的坐标，为“Arrow”时置-1
Arrow	Int8	10×10	Type 为“Amazon”时置-1，为“Arrow”时表示放置障碍的坐标

[0135] 由此，一共得到1879931条训练数据用于模型的训练，并将数据集以7:2:1划分为训练集、验证集、测试集三部分。

[0136] 3.网络训练

[0137] 为了便于解释网络的训练过程，我们以其中一条训练记录为例说明数据在网络中的变化过程以及参数的训练过程。

[0138] 初始棋盘和走子后的棋盘分别如图9和图10所示，所走招法记为D9-E10/E5，表示移动棋盘中坐标为D9处的棋子，并将其移动到E10处，并在E5处放置一个障碍，在训练数据中，该步招法记录为移动棋子和放置障碍两部分。下面以拟合该步招法为例进行说明，训练数据的表示如下所示：

[0139] (1)移动棋子的训练数据

[0140] {'Board':

[0141] [[100,100,100,100,100,100,100,100,100,100],

[0142] [100,100,200,300,100,100,0,0,100,100],

[0143] [100,100,0,0,100,100,100,0,100,100],

[0144] [100,100,100,100,100,100,300,0,100,100],

[0145] [0,0,100,200,100,100,100,0,0,100],

[0146] [100,0,0,100,100,100,100,100,200,100],

[0147] [100,100,100,100,100,100,0,0,0,0],

[0148] [0,100,0,100,0,100,100,100,100,100],

[0149] [100,100,200,100,0,100,100,300,100,100],

[0150] [100,100,300,100,100,100,100,100,100,100]],

[0151] 'Type':'Amazon',

[0152] 'From':[3,1],

[0153] 'To':[4,0],

[0154] 'Arrow':-1,

[0155] 'Result':'Lose'

[0156] }

[0157] (2)放置障碍的训练数据

[0158] {'Board':

[0159] [[100,100,100,100,300,100,100,100,100,100],

[0160] [100,100,200,100,100,100,0,0,100,100],

[0161] [100,100,0,0,100,100,100,0,100,100],

[0162] [100,100,100,100,100,100,300,0,100,100],

[0163] [0,0,100,200,100,100,100,0,0,100],

[0164] [100,0,0,100,100,100,100,100,200,100],

[0165] [100,100,100,100,100,100,0,0,0,0],

[0166] [0,100,0,100,0,100,100,100,100,100],

[0167] [100,100,200,100,0,100,100,300,100,100],

[0168] [100,100,300,100,100,100,100,100,100,100]],

[0169] 'Type':'Arrow',

[0170] 'From':-1,

[0171] 'To':-1,

[0172] 'Arrow':[4,5],

[0173] 'Result':'Lose'

[0174] }

[0175] 其中,0代表障碍,100代表空地,200代表黑棋,300代表白棋,在训练过程中,会将“Board”,“Type”作为特征,其他项作为标签传给神经网络。

[0176] 由此在棋子移动网络中,我们得到一个初始棋盘矩阵M,作为矩阵的输入。

[0177] $$M=\begin{bmatrix}100&100&100&100&100&100&100&100&100&100\\100&100&200&300&100&100&0&0&100&100\\100&100&0&0&100&100&100&0&100&100\\100&100&100&100&100&100&300&0&100&100\\0&0&100&200&100&100&100&0&0&100\\100&0&0&100&100&100&100&100&200&100\\100&100&100&100&100&100&0&0&0&0\\0&100&0&100&0&100&100&100&100&100\\100&100&200&100&0&100&100&300&100&100\\100&100&300&100&100&100&100&100&100&100\end{bmatrix}$$

[0178] 第一步，提取棋盘局面特征，第k个图像局面特征矩阵F^k的第(i,j)坐标(记作F_{ij}^k)通过计算我方(例中为白方)编号为k的棋子坐标(i_k,j_k)与棋盘(i,j)所围成的矩形区域的元素值之和得到，由此可以得到局面特征矩阵F，F的规模为4×10×10。

[0179] 以(C,1)处的白棋为例，设其编号为1，则根据公式(1)给出的计算方法，计算可以得到编号为1的棋子对应的局面特征矩阵F^1。

[0180] $$F^1=\begin{bmatrix}90.0&95.0&110.0&115.0&103.3&102.5&102.0&96.7&95.7&95.0\\88.9&94.4&111.1&116.7&103.7&102.8&102.2&96.3&95.2&94.4\\83.3&87.5&100.0&100.0&91.7&93.8&97.5&93.8&92.9&92.2\\85.7&92.9&114.3&114.3&100.0&100.0&102.9&100.0&98.0&96.4\\83.3&91.7&116.7&116.7&100.0&100.0&96.7&97.2&95.2&93.8\\93.3&100.0&120.0&110.0&93.3&95.0&92.0&96.7&97.1&95.0\\108.3&125.0&150.0&125.0&100.0&100.0&95.0&100.0&96.4&93.8\\111.1&133.3&166.7&133.3&100.0&100.0&100.0&111.1&109.5&108.3\\150.0&175.0&250.0&175.0&133.3&125.0&120.0&133.3&128.6&125.0\\166.7&200.0&300.0&200.0&166.7&150.0&140.0&133.3&128.6&125.0\end{bmatrix}$$

[0181] 第二步，将局面特征矩阵F作为网络的输入，卷积层中，对输入的矩阵F，它将经过12个卷积层，和3个丢弃概率为0.3的Dropout层。3个Dropout层分别处于第4,8,12个卷积层之后。表1给出了各卷积层的参数。

[0182] 第三步，对第二步输出的Tensor分别进行两部分并行操作：

[0183] 第一部分，将Tensor传入全连接层，输出一个标量，并对数据样本中当前局面对应的胜负情况进行拟合，本例中标签值为0(表示白方负)，误差如公式(2)所示。

[0184] 第二部分，将Tensor输入4层卷积层和1层全连接层，输出100×100的策略矩阵。策略矩阵中的每一个元素都代表一种走法，这一部分的误差如公式(5)所示。表2给出了这一部分的卷积层参数。

[0185] 如本例中D9-E10这一步，我们希望训练得到的策略矩阵在(83,94)处的概率取值

最大，根据公式(4)所示的坐标转换关系，D9-E10与(83,94)对应。

[0186] 在Arrow-Net中，棋盘为初始棋盘进行走子操作后对应的局面，以图9所示的初始棋盘为例，Arrow-Net输入的矩阵应为移动D9-E10这一步操作后的矩阵M′。

[0187] $$M'=\begin{bmatrix}100&100&100&100&300&100&100&100&100&100\\100&100&200&100&100&100&0&0&100&100\\100&100&0&0&100&100&100&0&100&100\\100&100&100&100&100&100&300&0&100&100\\0&0&100&200&100&100&100&0&0&100\\100&0&0&100&100&100&100&100&200&100\\100&100&100&100&100&100&0&0&0&0\\0&100&0&100&0&100&100&100&100&100\\100&100&200&100&0&100&100&300&100&100\\100&100&300&100&100&100&100&100&100&100\end{bmatrix}$$

[0188] 第一步，同棋子移动网络的第一步操作，但此时输入的矩阵不再是原始棋盘矩阵，而是移动棋子后的棋盘矩阵M′，由此得到Arrow-Net中的输入局部特征矩阵F′，F′的规模也为4×10×10。

[0189] 第二步，将局面特征矩阵F′作为网络的输入，卷积层中，对输入的矩阵F′，它将经过12个卷积层，和3个丢弃概率为0.3的Dropout层。3个Dropout层分别处于第4,8,12个卷积层之后。表3给出了各卷积层的参数。

[0190] 第三步，对输出的Tensor分别的进行两步部分并行操作：

[0191] 第一部分，将Tensor传入全连接层，输出一个标量，并对数据样本中当前局面对应的胜负情况进行拟合，误差如公式(7)所示。

[0192] 第二部分，将Tensor输入4层卷积层和1层全连接层，输出10×10的策略矩阵。策略矩阵中的每一个值都代表一种障碍坐标在该处坐标放置的概率，这一部分的误差如公式(9)所示。表4给出了各卷积层的参数。

[0193] 如本例中E5处的障碍，我们希望训练得到的策略矩阵在(5,5)处的概率取值最大，策略矩阵P′与棋盘矩阵坐标一一对应，因此棋盘中的E5坐标与策略矩阵(5,5)对应。

[0194] 公式(6)、公式(10)分别定义了棋子移动网络、障碍放置网络的网络误差，通过RMSProp算法，将网络的误差最小化，对网络参数进行调整。

[0195] RMSProp算法优化了损失函数在更新中存在摆动幅度过大的问题，消除了摆动幅度大的方向，使得各个维度的摆动幅度都较小，并且使得网络函数收敛更快。

[0196] 通过对训练集的拟合，实现网络误差的最小化，进而得到具有招法生成功能的网络Amazons-Net，然后应用该网络生成当前棋局下的最优招法。

[0197] 本发明方法相对现有的基于博弈树的招法生成方法而言，具有效率稳定，不受极端局面影响的优势。基于博弈树的剪枝搜索算法很大程度上依赖于招法的估值函数和搜索节点排列顺序，一方面，搜索节点排列顺序的不确定性导致搜索过程中不能保证搜索效率的稳定性，另一方面，搜索结果的好坏强烈依赖于估值函数的准确性，而估值函数受限于人

类先验知识，无法准确估计亚马逊棋的复杂局面。在最优招法生成的过程中，基于博弈树的招法生成方法针对一部分局面能够较快的搜索出最优招法，而针对一些极端局面往往无法在一定时间内给出理想的搜索结果，而本发明提出的招法生成方法基于深度卷积神经网络，其搜索效率不依赖于局面的好坏和估值函数。对于任意给定的局面，通过计算两次前向传播即可得到最优招法，其效率与局面信息无关，因此具有对亚马逊棋局面的鲁棒性。

[0198] 此外，本发明创造性地将亚马逊棋的行棋过程分离为棋子走子和放置障碍两个过程，避免神经网络单步生成最优招法时，全连接层参数矩阵过于庞大，生成效率低下以及准确率过低的缺陷。若不采用分步决策，最终策略矩阵的元素个数将会达到百万之多，且策略矩阵将过于稀疏。本发明将走子和放置障碍的过程分离，采用分步决策的方式，使得最终的决策计算分别只涉及100×100和10×10的策略矩阵，大大减少了全连接层的神经元数量，缓和了策略矩阵元素个数过多、矩阵稀疏等问题。

[0199] 在采用分步决策的招法生成方式的同时，本发明又考虑了棋子走子和放置障碍这两个过程的连续性，通过共享权重的方式建立棋子移动网络和障碍放置网络之间的联系。通过两个不同的决策过程共享部分网络层的方式，各决策步骤间的相似性得以表达。同时，共享的网络层可以有效减少神经网络中的参数数目，一定程度上避免模型过拟合，提高模型的泛化能力。

[0200] 为了说明本发明的内容及实施方法，本说明书给出了上述具体实施例。但是，本领域技术人员应理解，本发明不局限于上述最佳实施方式，任何人在本发明的启示下都可得出其他各种形式的产品，但不论在其形状或结构上作任何变化，凡是具有与本申请相同或相近似的技术方案，均落在本发明的保护范围之内。

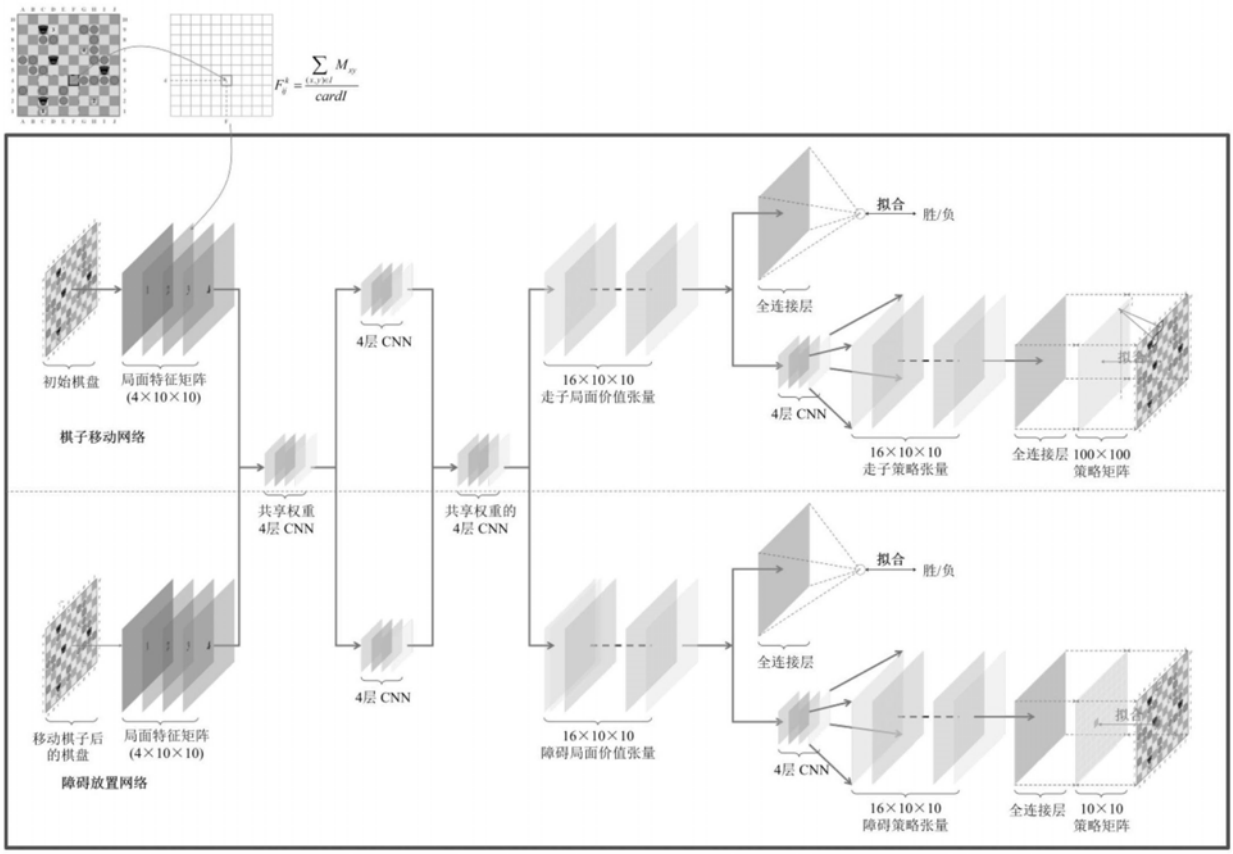
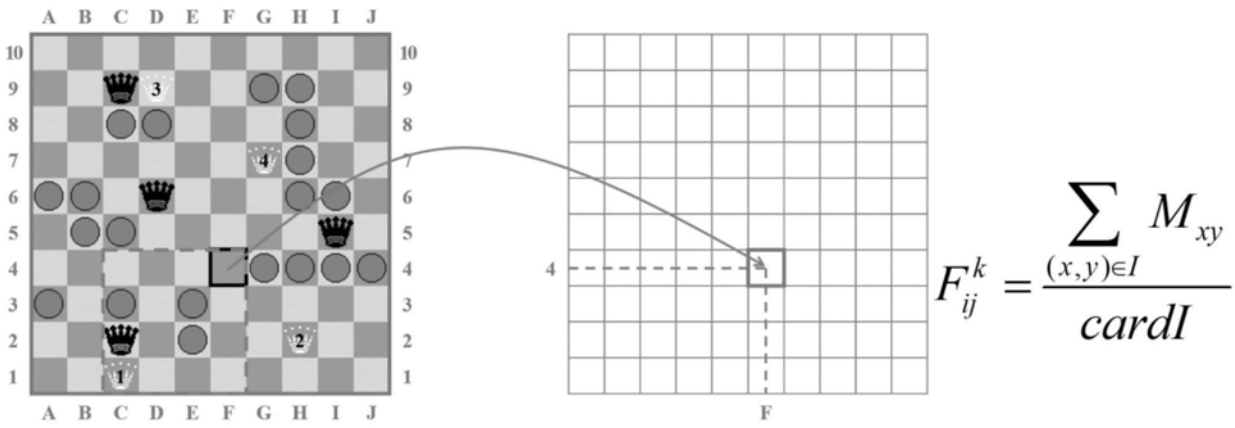


图1



$$F_{ij}^{k}=\frac{\sum_{(x,y)\in I} M_{xy}}{cardI}$$

图2

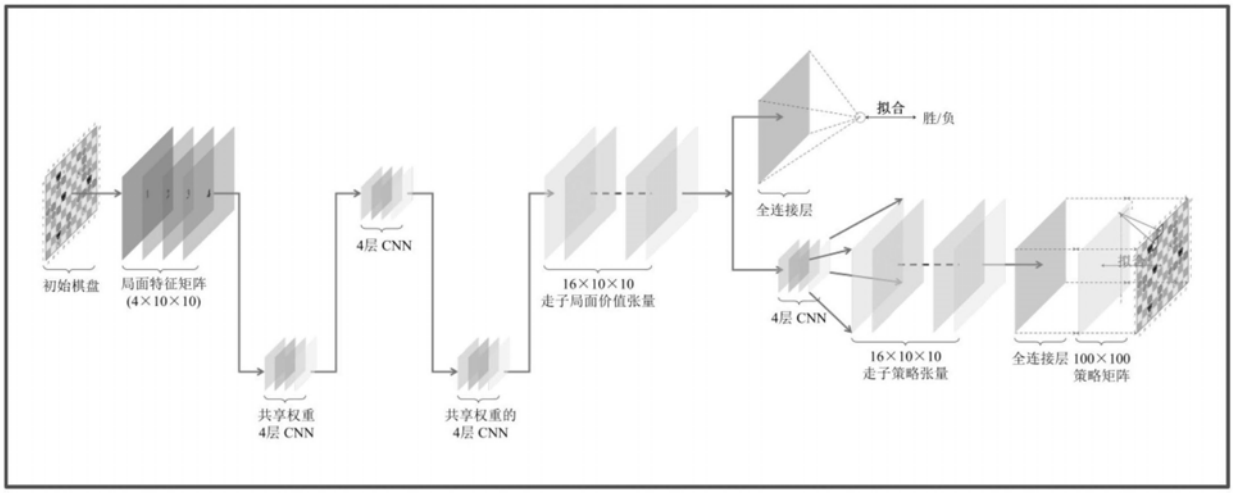

初始棋盘
局面特征矩阵
(4×10×10)
共享权重
4层 CNN
4层 CNN
共享权重的
4层 CNN
16×10×10
走子局面价值张量
拟合
胜/负
全连接层
4层 CNN
16×10×10
走子策略张量
全连接层
100×100
策略矩阵

图3

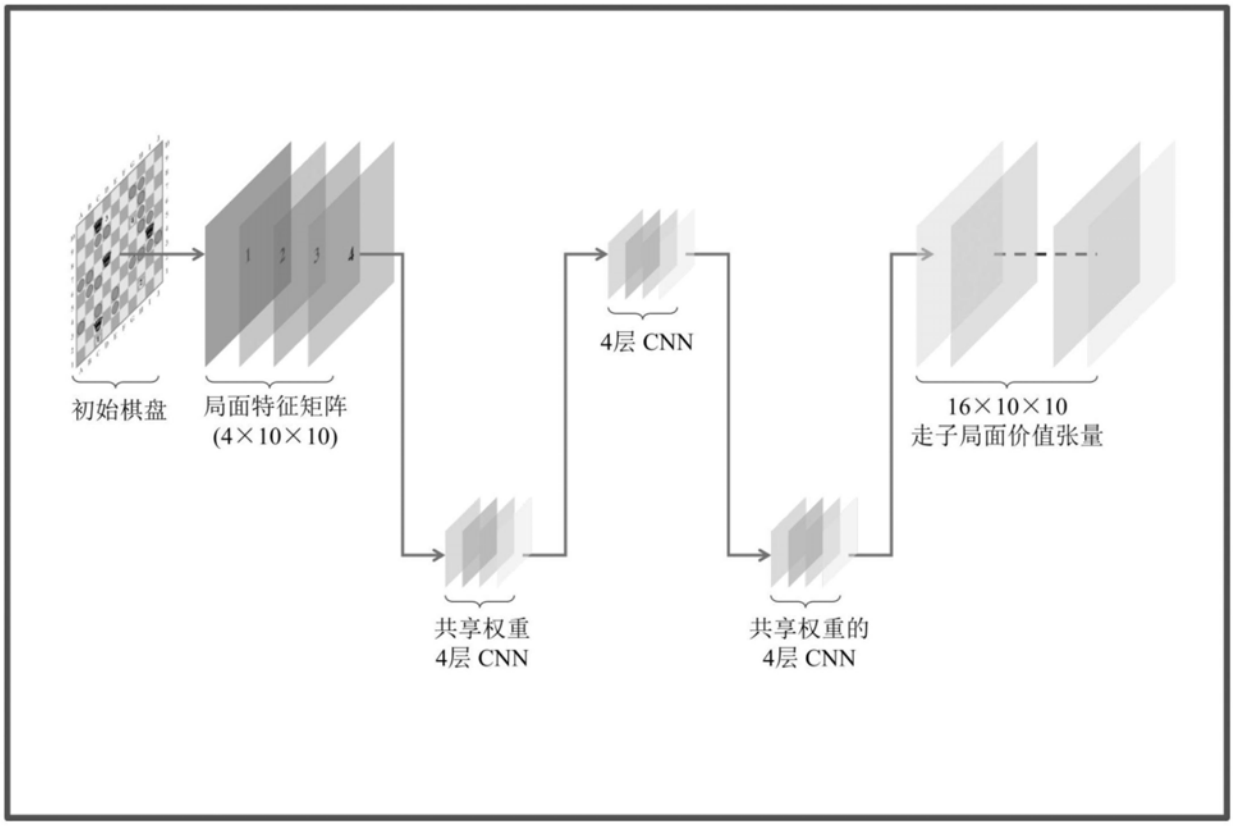

初始棋盘
局面特征矩阵
(4×10×10)
共享权重
4层 CNN
4层 CNN
共享权重的
4层 CNN
16×10×10
走子局面价值张量

图4

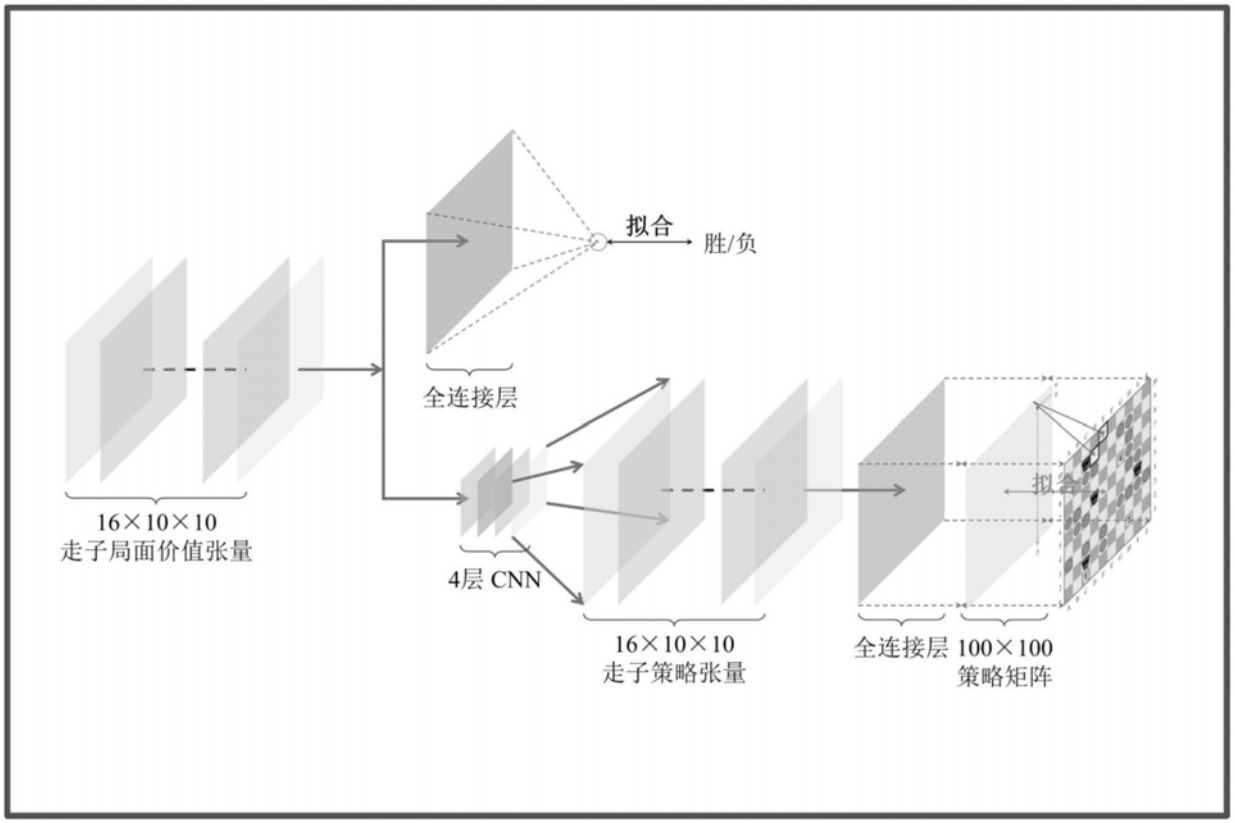

拟合
胜/负
全连接层
16×10×10
走子局面价值张量
4层 CNN
16×10×10
走子策略张量
全连接层
100×100
策略矩阵

图5

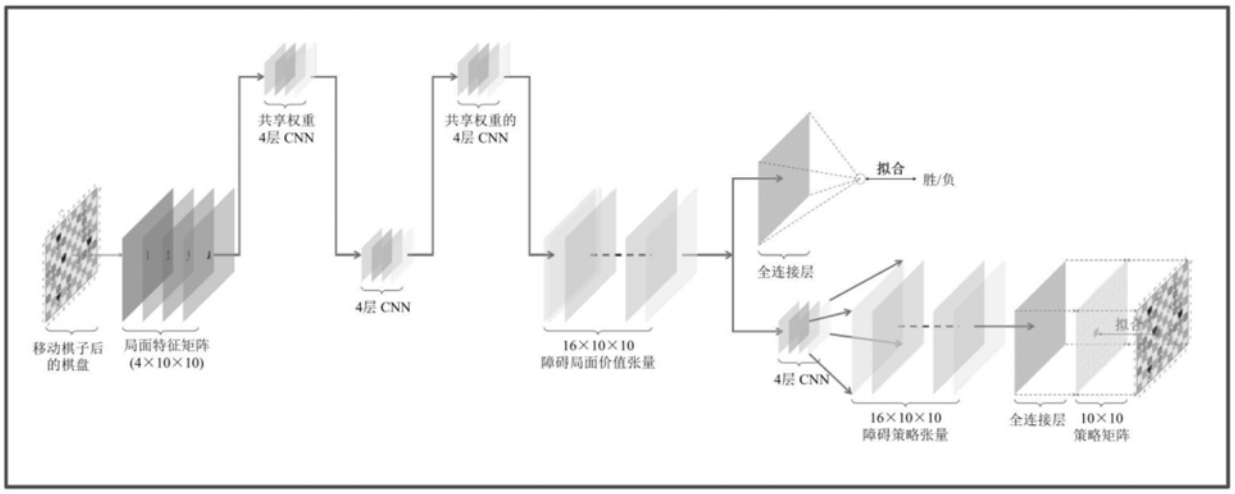

共享权重
4层 CNN
共享权重的
4层 CNN
拟合
胜/负
全连接层
移动棋子后
的棋盘
局面特征矩阵
(4×10×10)
4层 CNN
16×10×10
障碍局面价值张量
4层 CNN
16×10×10
障碍策略张量
全连接层
10×10
策略矩阵

图6

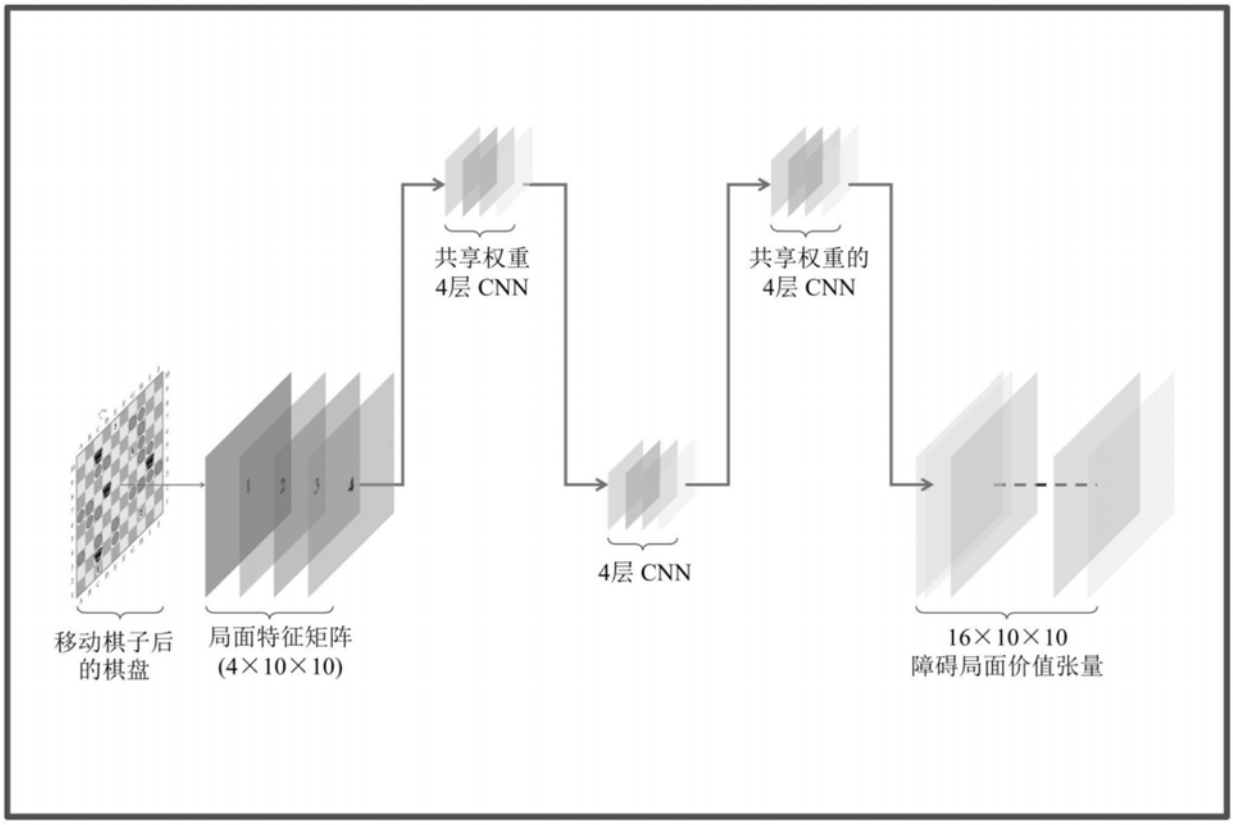

共享权重
4层 CNN
共享权重的
4层 CNN
4层 CNN
移动棋子后
的棋盘
局面特征矩阵
(4×10×10)
16×10×10
障碍局面价值张量

图7

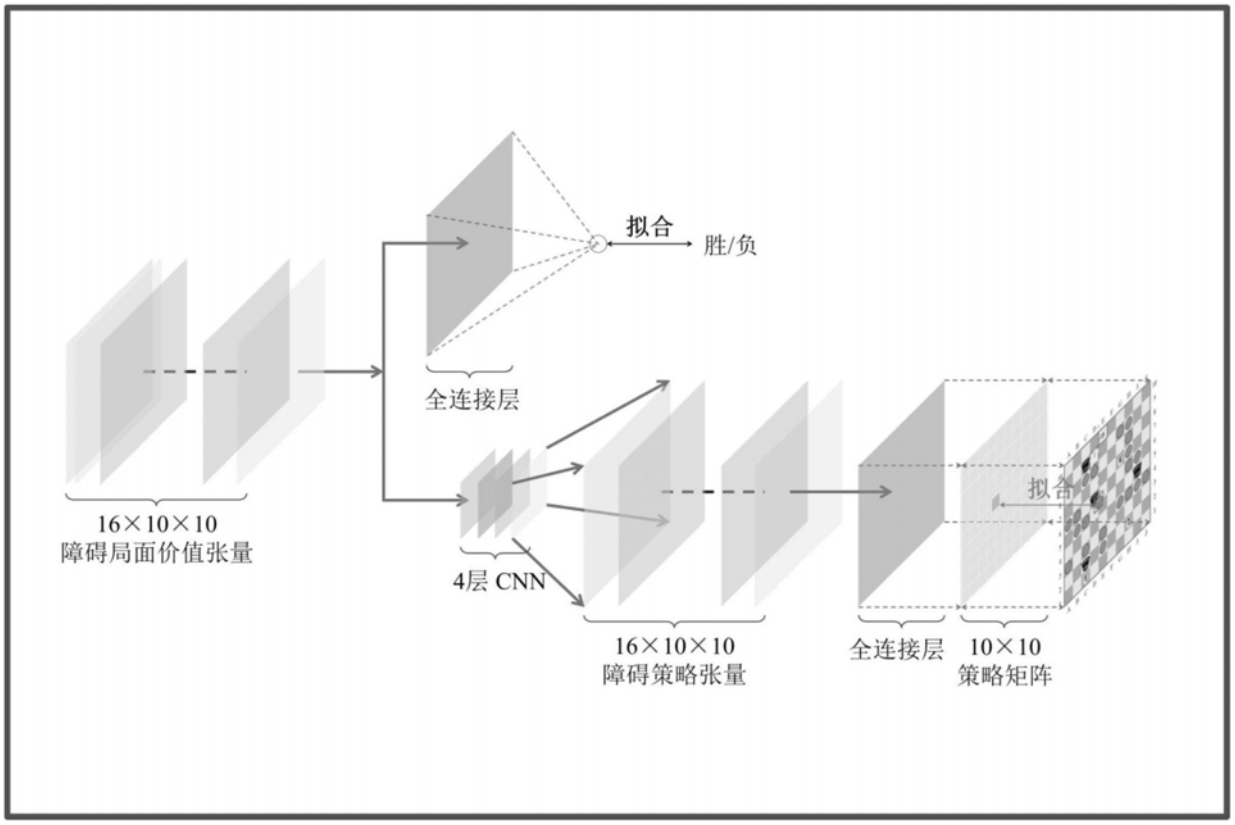

拟合
胜/负
全连接层
16×10×10
障碍局面价值张量
4层 CNN
16×10×10
障碍策略张量
全连接层
10×10
策略矩阵
拟合

图8

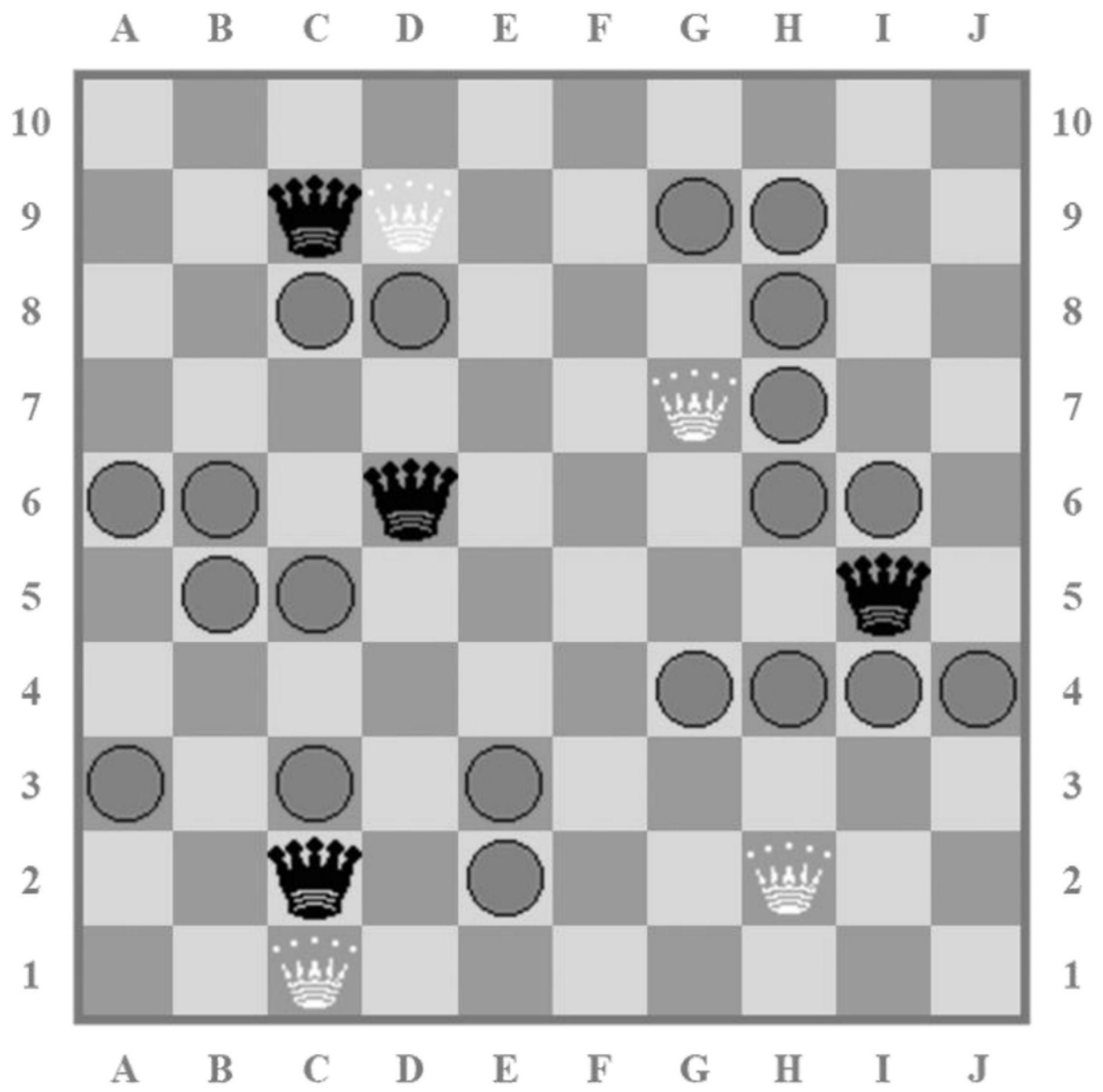


图9

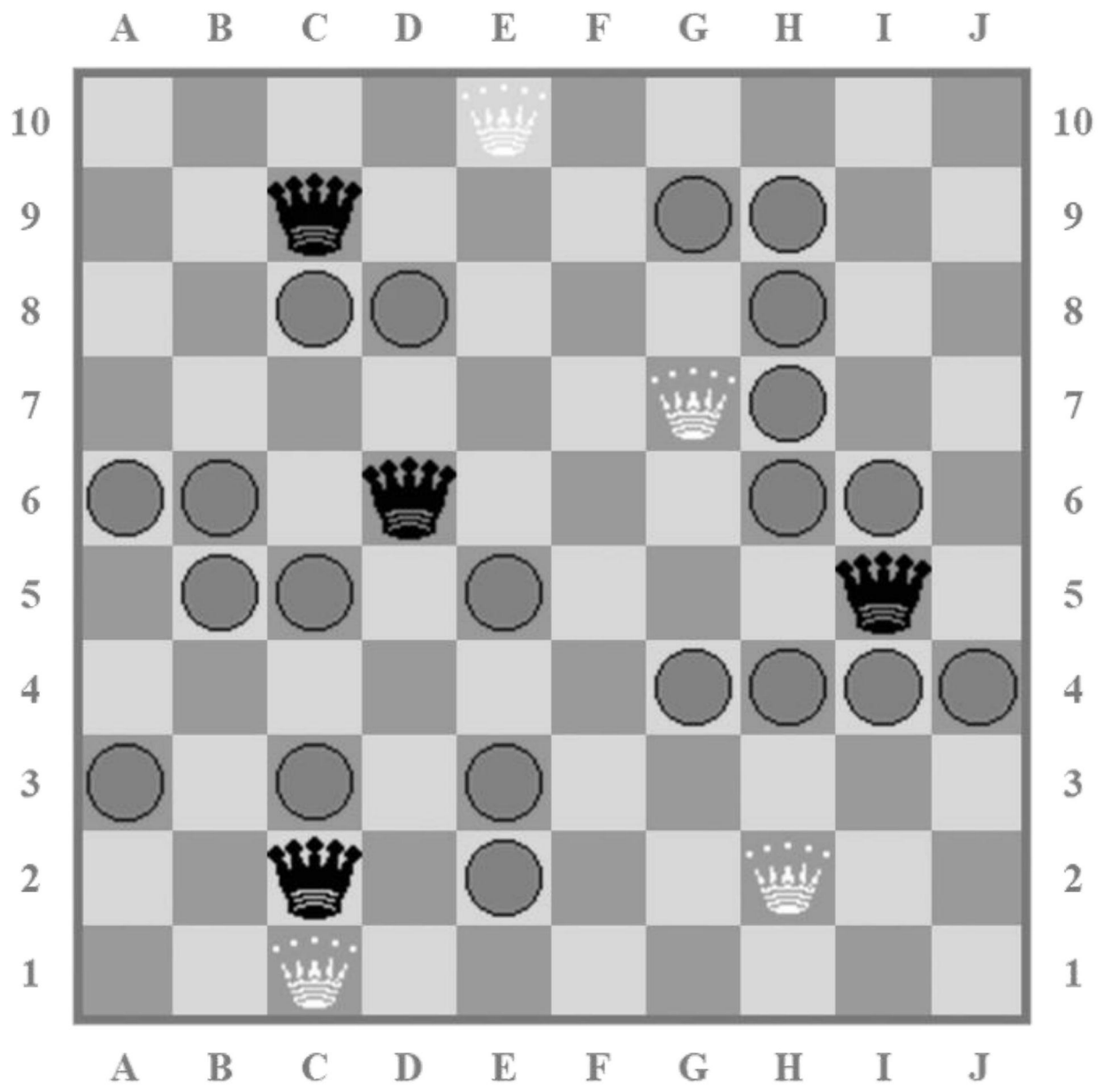


图10