Компьютерные инструменты в образовании, 2022
№ 1: 5–15
УДК: 004.891 : 004.032.26
http://cte.eltech.ru
doi:10.32603/2071-2340-2022-1-5-15

алгоритмическая математика и математическое моделирование

УПРАВЛЕНИЕ ДВИЖЕНИЕМ ОБЪЕКТА ПО ПОЛЮ С ПРЕПЯТСТВИЯМИ ПРИ ПОМОЩИ РЕКУРРЕНТНОЙ НЕЙРОННОЙ СЕТИ

Ляхов А. Ф.[1], кандидат физико-математических наук, доцент, ✉ Alf19545@rambler.ru
Королев Д. А.[2], инженер-программист, ✉ corolyov1998@gmail.com

[1]Национальный исследовательский Нижегородский государственный университет им. Н. И. Лобачевского, пр. Гагарина, 23, 603022, Нижний Новгород, Россия
[2]ООО «Научно-производственное предприятие «ПРИМА», Сормовское шоссе, д. 1Ж, 603950, Нижний Новгород, Россия

Аннотация

Рассматриваются модели управления рекуррентной нейронной сетью движением объекта по полю с препятствиями. С помощью генетического алгоритма созданы две нейронные сети разной сложности. Для каждой сети описаны алгоритмы обучения с подкреплением. Проводятся сравнения эффективности их работы.

Ключевые слова: *нейронная сеть, обучение с подкреплением, генетический алгоритм, движение объекта, искусственный интеллект.*

Цитирование: Ляхов А. Ф., Королев Д. А. Управление движением объекта по полю с препятствиями при помощи рекуррентной нейронной сети // Компьютерные инструменты в образовании. 2022. № 1. С. 5–15. doi: 10.32603/2071-2340-2022-1-5-15

1. ВВЕДЕНИЕ

Современное развитие вычислительной техники, IT-технологий позволяют создавать автоматизированные системы управления движением объектов. Диапазон задач управления движением очень широк: это космические планетоходы, роботизированные летательные аппараты, подводные аппараты, но самая важная задача связанна с созданием автоматизированного автомобильного транспорта.

При создании систем управления движением используются компьютерные программы, имитирующие управление человеком, компьютерные нейронные сети различной сложности. Нейронные сети обладают способностью в процессе обучения на тестовых примерах адаптироваться к изменению свойств объекта управления и внешней среды [1].

В данной работе описаны две нейронные сети — однослойная нейронная сеть прямого распространения и двухслойная рекуррентная нейронная сеть, — управляющие движением объекта по смоделированному полю. Объект самостоятельно совершает манёвры, обходит препятствия, уклоняется от границ поля, целенаправленно выбирает призовые клетки.

Было показано, что при управлении нейронной сетью прямого распространения возможно зацикливание движения. Рекуррентная нейронная сеть позволяет устранить этот недостаток. При обучении нейронных сетей использовался генетический алгоритм.

2. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ДВИЖЕНИЯ ОБЪЕКТА

Объект перемещается по полю, которое представлено в виде дискретной сетки на плоскости размером $N \times M$. Движение объекта ограниченно границами поля и случайно сгенерированными препятствиями в отдельных клетках. При столкновении с препятствием объект останавливается и прекращает движение. Объект может совершать ограниченное число шагов, равное $2(N \times M)$. На поле сгенерированы «призовые» клетки, то есть клетки, которые имеют приоритет при выборе направления движения объекта. Когда объект заходит на клетку с «призом», «приз» исчезает, а его запас хода становится равен начальному $2(N \times M)$. Призовая клетка создаётся вновь на свободной клетке поля.

Объект получает информацию об окружающих его восьми клетках: о клетках по вертикали, горизонтали и диагоналях. Объект может перемещаться на одну клетку по вертикали или горизонтали (рис. 1).

Цель управления движением объекта состоит в том, чтобы объект прошёл наибольший путь.

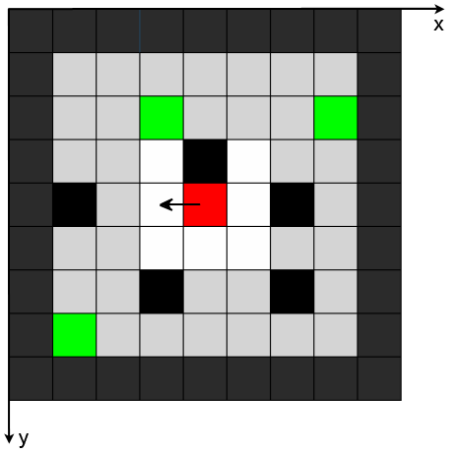

Рис. 1. Объект на игровом поле. Красная клетка — объект, черные клетки — препятствия, зелёные клетки — призовые

При движении объект совершает дискретные переходы из одного квадрата в соседний. Поэтому при исследовании естественно использовать манхэттенскую метрику пространства.

Границы поля имеют координаты

$$\begin{cases} x = 0 \cup x = N, \ \ \forall y, \\ y = 0 \cup y = M, \ \ \forall x. \end{cases} \tag{1}$$

Перемещение объекта можно записать в следующем виде

$$\begin{cases} X(t_i) = X(t_{i-1}) + \dot{x}(t_i) \cdot \Delta t, \\ Y(t_i) = Y(t_{i-1}) + \dot{y}(t_i) \cdot \Delta t, \end{cases} \tag{2}$$

где t_i — номер хода, $X(t_i), Y(t_i)$ — координаты объекта после перемещения, $X(t_{i-1})$, $Y(t_{i-1})$ — координаты объекта до перемещения. Время в рассматриваемой модели движения изменяется дискретно $t_{i+1} = t_i + \Delta t$, $\Delta t = 1$, $\dot{x}(t_i)$ — горизонтальная проекция, а $\dot{y}(t_i)$ — вертикальная проекция скорости. Проекции скорости объекта принимают значения либо 0, либо 1 в зависимости от направления движения.

Скорость движения объекта

$$\vec{V}(t_i) = \dot{x}(t_i)\vec{e}_1 + \dot{y}(t_i)\vec{e}_2, \quad \left|\vec{V}(t_i)\right| = 1, \quad \forall i. \tag{3}$$

Условие того, что объект не покинет поле:

$$\begin{cases} 0 < X(t_i) < N, \\ 0 < Y(t_i) < M. \end{cases} \tag{4}$$

Путь, пройдённый за i шагов, будет равен

$$\begin{gathered} S(t_i) = |V(t_{i-1})|\Delta t + S(t_{i-1}), \\ S(t_i) = 1. \end{gathered} \tag{5}$$

Задача управления движением объекта состоит в том, чтобы объект прошёл максимально большой путь $S(t_i)$, не сталкиваясь с препятствиями, то есть совершил максимальное число шагов.

Объект имеет координаты $\big(X(t_i), Y(t_i)\big)$ и получает информацию из восьми клеток окружающих его

$$\begin{gathered} \big(X(t_i)-1, Y(t_i)-1\big), \ \big(X(t_i), Y(t_i)-1\big), \ \big(X(t_i)+1, Y(t_i)-1\big), \ \big(X(t_i)-1, Y(t_i)\big), \\ \big(X(t_i)+1, Y(t_i)\big), \ \big(X(t_i)-1, Y(t_i)+1\big), \ \big(X(t_i), Y(t_i)+1\big), \ \big(X(t_i)+1, Y(t_i)+1\big). \end{gathered}$$

Если в клетках

$$\big(X(t_i), Y(t_i)-1\big), \ \big(X(t_i), Y(t_i)+1\big), \ \big(X(t_i)-1, Y(t_i)\big), \ \big(X(t_i)+1, Y(t_i)\big)$$

обнаружено препятствие, то объекту запрещено перемещаться в их сторону.

На первом этапе исследования для управления объектом была создана однослойная нейронная сеть на основе персептрона Розеблатта. Созданная нейронная сеть имеет восемь входных нейронов по числу клеток, которые может «видеть» объект, а выходной слой — четыре нейрона, соответствующие направлениям движения (рис. 2). Данные об окружении объекта отправляются на вход в нейронную сеть. После вычислений, выполненных нейронной сетью, объект совершает некоторое действие и вновь считывает данные об окружающих клетках. При реальном использовании нейронной сети данные об окружении собираются датчиками, расположенными на объекте.

В качестве функции активации нейронов был выбран гиперболический тангенс

$$act(x) = \frac{1 - e^{-x}}{1 + e^{-x}}. \tag{6}$$

Преимущество данной функции заключается в нелинейности функции, непрерывной области значений и непрерывной области определения

$$E(x) = (-\infty, +\infty), \quad D(act) = (-1, 1).$$

Область значений функции включает в себя две ключевые подобласти: область меньше нуля и область больше нуля. Эти особенности позволяют создавать более гибкие связи между нейронами, так как значение нейрона может быть не только положительным, но и отрицательным. Такая нейронная сеть при добавлении скрытых слоёв даёт более точные ответы, но при этом усложняется обучение.

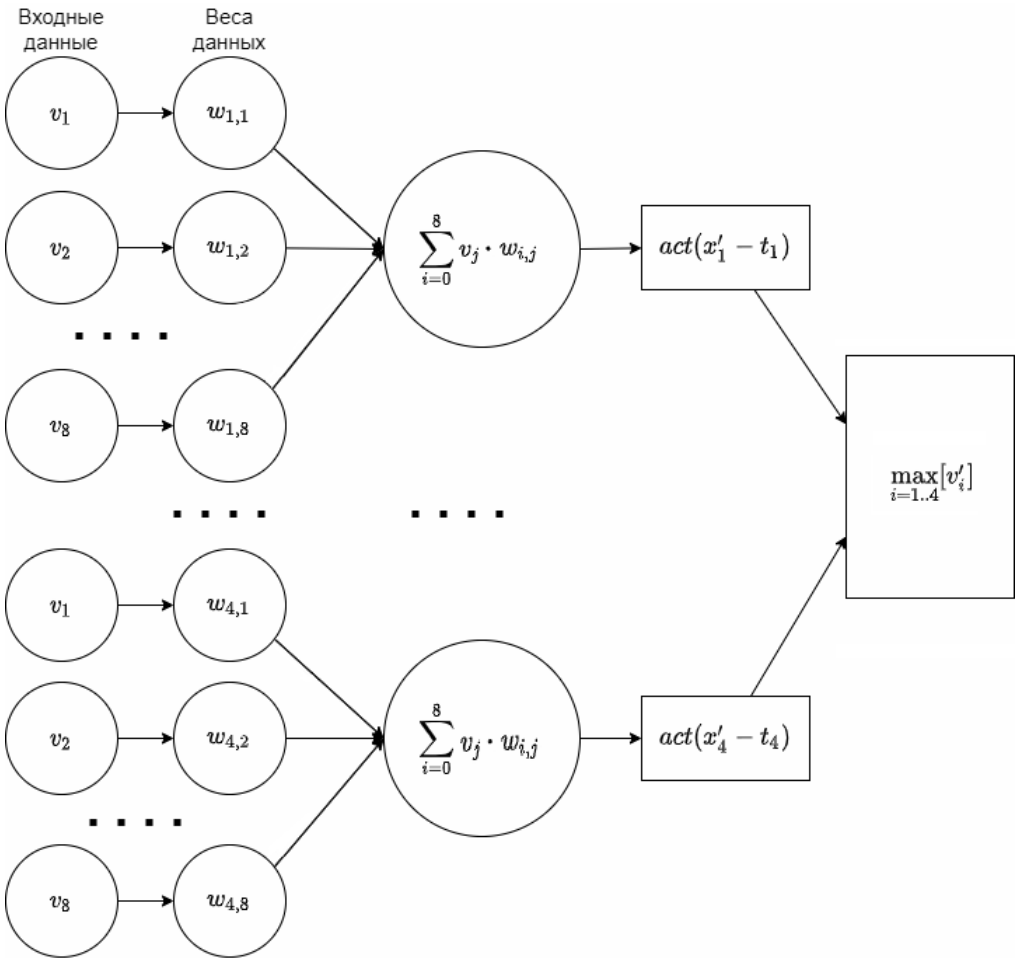

Рис. 2. Развёрнутый граф нейронной сети

Неизвестной величиной является вектор-столбец величин нейронов на выходе $\begin{pmatrix} v'_1 \\ \vdots \\ v'_4 \end{pmatrix}_o$.

$$act = \left(\begin{pmatrix} w_{1,1} & \dots & w_{1,8} \\ \vdots & \ddots & \vdots \\ w_{4,1} & \dots & w_{4,8} \end{pmatrix}_i \begin{pmatrix} v_1 \\ \vdots \\ v_4 \end{pmatrix}_i + \begin{pmatrix} t_{0_1} \\ \vdots \\ t_{0_4} \end{pmatrix}_o \right) = \begin{pmatrix} v'_1 \\ \vdots \\ v'_4 \end{pmatrix}_o, \quad (7)$$

где $w_{i,j}$ — веса нейронной сети, v_1 — величина нейрона (сигнала), t_{0_i} — пороговое значение нейронов следующего слоя, пороговые значения имеют все нейроны, кроме нейронов входного слоя, индекс i означает, что данные значения принадлежат входному слою, индекс o означает, что значения принадлежат выходному слою.

Обучение нейронной сети заключается в том, что необходимо найти значение весов и пороговых значений, при которых объект совершит максимальное количество шагов. При обучении значения v_i входных и выходных слоёв заданы, в качестве неизвестных выступают переменные $w_{i,j}$ и t_{0_k}. Веса и пороговые значения в дальнейшем будут называться просто веса.

Использование для обучения сети метода обратного распространения ошибки показало его, не эффективность в данной задаче, то есть при тестовых испытаниях путь, проходимый объектом, был мал. Поэтому было принято решение создать нейронную сеть с использованием генетического алгоритма.

Для поиска оптимальных весов нейронной сети используется генетический алгоритм (ГА) с турнирным отбором [2]. Генетический алгоритм представляет собой комбинированный подход. Механизмы скрещивания и мутации реализуют переборную часть метода, а отбор лучших решений — градиентный спуск [3].

3. МАТЕМАТИЧЕСКОЕ ОПИСАНИЕ ГЕНЕТИЧЕСКОГО АЛГОРИТМА ОБУЧЕНИЯ НЕЙРОННОЙ СЕТИ

Рассмотрим пространство весов нейронной сети R^{36}. В качестве целевой функции задачи оптимизации весов $S(W)$ выберем количество шагов, сделанных объектом, где W — точка в этом пространстве.

$$f : S(W) \to +\infty, \tag{8}$$

$$W = \{w_1, ..., w_{36}\}. \tag{9}$$

Функция $S(W)$ является дискретным счётчиком шагов.

Под особью, которая используется в генетическом алгоритме, будем понимать динамический объект, находящийся под управлением конкретной нейронной сети. Множество таких объектов образуют популяцию. В качестве хромосом этих объектов выступают веса нейронной сети. Каждая хромосома — это отдельный элемент из матрицы весов.

Алгоритм представляет собой итерационный численный метод. ГА имеет преимущество над другими итерационными методами. В обычных итерационных методах осуществляется сходимость к единственному наилучшему решению. При применении генетического алгоритма ищется набор потенциальных решений, одно из которых является лучшим. При этом вероятность попасть в локальный оптимум целевой функции ниже, чем при применении обычных методов [4].

Начальная популяция размером k является набором точек в 36-мерном пространстве, где координатами точки являются веса нейронной сети:

$$v = \{w_{1,1}, ..., w_{4,8}, w_{0_1}, ..., w_{0_4}\}. \tag{10}$$

Для тестирования особей создаётся набор обучающих полей. В набор входят всевозможные комбинации полей размером 3×3, с пустой клеткой в центре и со сгенерированными на этих полях препятствиями и призами. Рассматривается все множество таких полей за исключением полей с тупиковыми ситуациями, то есть полей в которых препятствия одновременно расположены по вертикальной и горизонтальной осях относительно центра поля. Испытуемая особь помещается в центр поля и совершает один шаг. Если шаг успешный, то он засчитывается, и прибавляется к счётчику шагов. Если шаг

сделан в сторону препятствия, то счётчик шагов не изменяется. Шаг, сделанный в сторону призовой клетки, засчитывается и при этом ещё засчитывается количество призовых очков.

Выбираем особи, которые показали наилучший результат на обучающих полях и применяем к ним метод кроссинговера, где p — случайная величина, а $T_{\text{К}}$ — пороговое значение для данного метода [5]:

$$cross(v^1, v^2) = \begin{cases} v_i^1 = v_i^2, & \text{если } p \leqslant T_{\text{К}}, \\ v_i^1 = v_i^2, & \text{если } p > T_{\text{К}}, \end{cases} \tag{11}$$

$$p, T \in [0,1], \quad p, T_{\text{К}} \in R.$$

Затем применяем к ним метод мутации, где p и q — случайные величины, $T_{\text{М}}$ — пороговое значение для данного метода, а c — коэффициент мутации:

$$mutation(v) = \begin{cases} v_i = v_i, & \text{если } p > T_{\text{М}}, \\ v_i = v_i + c, & \text{если } p \leqslant T_{\text{М}} \text{ и } q = 1, \\ v_i = v_i, -c, & \text{если } p \leqslant T_{\text{М}} \text{ и } q = 0, \end{cases} \tag{12}$$

$$p, T \in [0,1], \quad p, T_{\text{М}} \in R,$$
$$q \in [0,1], \quad q \in Z.$$

Алгоритм выполняется до тех пор, пока не будет, достигнут один из критериев останова. Первый критерий остановка происходит при достижении заданной точности, которая вычисляется как отношение числа шагов сделанной наилучшей особью к количеству обучающих полей. Второй критерий — это ограничение на число поколений, то есть при достижении максимального поколения ответом являются гены наилучшей особи.

Начальная популяция создаётся из объектов, гены которых сгенерированы случайным образом. Каждая особь последовательно запускается на все обучающие поля. При этом ведётся подсчёт сделанных шагов, количество шагов, сделанных в сторону еды, определяет число очков. Сортируем особи по шагам, где первая особь будет особь, сделавшая наибольшее число шагов. Если особей с наибольшим числом шагов больше одной, то их сортируем по количеству очков. Фиксируем первую особь, она без изменений в генах переходит в следующее поколение. Далее выбираем первые десть процентов особей и попарно применяем к ним метод кроссинговера.

Данный метод заключается в том, что берутся две разные особи, и из их генов формируется новая особь. Каждый ген новой особи выбирается случайным образом, если случайное число меньше или равно пороговому значению, то значение гена новой особи равно значению гена первой особи, иначе значение гена будет равно значению гена второй особи.

Процесс кроссинговера применяется до тех пор, пока число особей новой популяции не будет равно числу особей предыдущей.

Ко всем особям новой популяции применяется операция мутации. Каждый ген особи, подверженной мутации, изменяется случайным образом. Если случайное число меньше или равно пороговому значению, то к гену прибавляется или вычитается коэффициент мутации, операция выбирается случайным образом, обе операции имеют одинаковую вероятность. Коэффициент мутации равен 1, и с каждым поколением уменьшается на шаг, равный 0,01, до значения 0.01. Алгоритм выполняется до тех пор, пока не будет выполнен критерий останова [2].

Для исследования движения объекта и эффективности его управления была написана программа **MOVE_FNN** на языке программирования C++. Тест программы проводился на компьютере с процессором Intel Core i5 7200u, ОЗУ 8Гб.

Было проведено десять тестов при следующих параметрах:

1. Объем популяция — 100 особей.
2. Пороговое значение для кроссинговера 0,5.
3. Пороговое значение для мутации 0,5.
4. Количество поколений — неограниченно.
5. Требуемая точность 0,999.

В ходе теста алгоритма были обучены нейронные сети прямого распространения. Каждый объект успешно обходил препятствия и собирал призы, когда последние были в радиусе обзора на тестовом поле.

Исследование движения объекта под управлением обученной нейронной сети показало, что движение объекта можно разделить на два этапа. Первый этап, «до цикла», когда при выборе n последних ходов будет выполняться следующее условие:

$$\exists A_i \neq A_{i+n/2}, \quad i = 1, \ldots, n, \quad 2 \leqslant n \leqslant h \cdot w. \tag{13}$$

где A_i — точка с координатами (X_i, Y_i), i — номер хода, h, w — натуральные числа, обозначающие высоту и ширину поля. Второй этап — когда нейронная сеть принимает одни и те же решения, вследствие чего объект ходит по одному и тому же маршруту:

$$\forall i = 1, \ldots, n/2, \quad A_i = A_{i+n/2}, \quad 2 \leqslant n \leqslant h \cdot w. \tag{14}$$

Эта проблема возникает из-за того, что нейронная сеть обрабатывает каждый ход независимо от предыдущих ходов.

Для устранения возможного зацикливания движения объекта создана рекуррентная нейронная сеть (РНС), которая использует данные двух последовательных ходов (рис. 3).

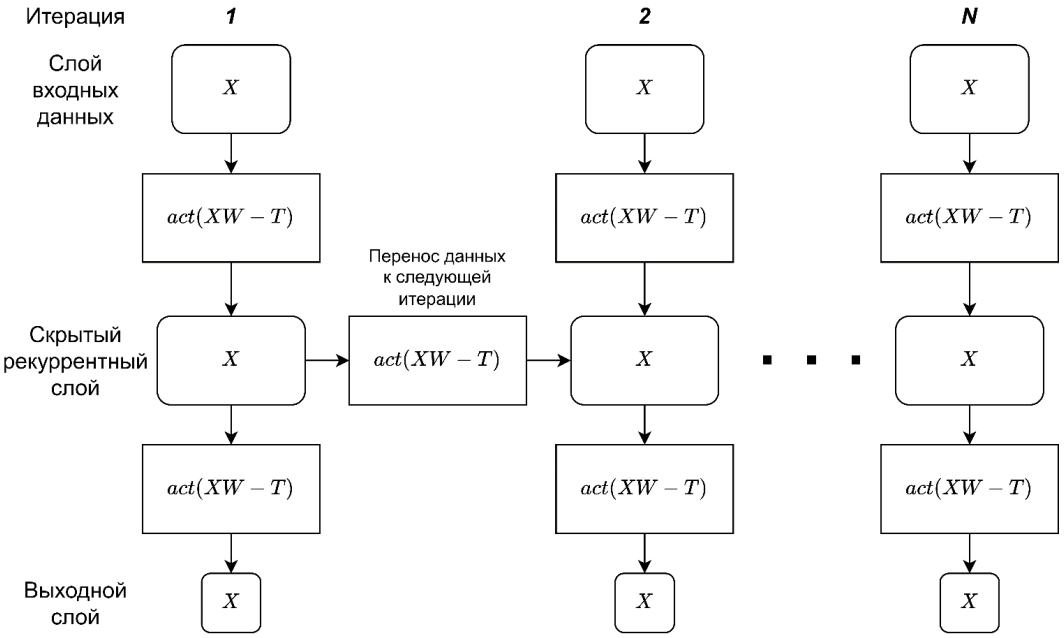

Рис. 3. Взаимодействие данных в РНС между итерациями

За основу РНС взята однослойная нейронная сеть, созданная на предыдущем этапе работы. Входной и выходной слои остаются без изменений. Создаётся дополнительный скрытый слой и блок памяти. Размер скрытого слоя равен 24 нейронам, это следует из условия, что каждая клетка вокруг объекта может принимать одно из трёх значений. Блок памяти равен скрытому слою. Такая архитектура обеспечит синхронизированные последовательности входов и выходов во время движения объекта на поле [6].

Входной слой нейронной сети — это вектор-столбец, размер которого равен восьми. Он полносвязно соединён со скрытым слоем. Скрытый слой — это вектор-столбец, который равен двадцати четырём нейронам. Он соединён с блоком памяти и полносвязно с выходным слоем. Блок памяти — это вектор-столбец такого же размера, что и скрытый слой. Ему передаются значения из текущего скрытого слоя без изменений, при следующем подсчёте РНС с помощью полной связи блок памяти передаёт значения в текущий скрытый слой. Выходной слой — это вектор-столбец размером, равным четырём. Полная связь реализована при помощи весов и пороговых значений [7]:

$$act\left(W_{n,m} \times X_{m,1} - T_{n,1}\right) = X'_{n,1}, \tag{15}$$

где $W_{n,m}$ — матрица весов текущего слоя, $X_{m,1}$ — вектор-столбец нейронов текущего слоя, $T_{n,1}$ — вектор-столбец пороговых значений для слоя назначения, $X'_{n,1}$ — вектор-столбец нейронов слоя назначения, функция активации используется для нормализации значений:

$$act\left(V_{m,m} \times X_{m,1} - R_{m,1}\right) = X'_{m,1}. \tag{16}$$

Для поиска оптимальных весов решается задача математического программирования, целевая функция которой — это количество шагов, сделанных объектом. Для РНС пространство весовых коэффициентов будет иметь размерность R^{916}.

Для нахождения оптимальных весов вновь использовался генетический алгоритм. Обучение было разделено на два этапа.

Первый этап, идентичный алгоритму обучения нейронной сети прямого распространения. На этом этапе после каждого подсчёта выходных значений нейронной сети необходимо очищать значения блока памяти. При использовании методов кроссинговера и мутации веса, участвующие в рекуррентной связи, не используются. После завершения первого этапа на выходе получаем набор нейронных сетей, способных решать задачу выбора пути на один шаг.

На втором этапе обучения используется блок памяти. Значения нейронов в блоке памяти приравниваются к нулевому вектору соответствующего размера. В качестве начальной, то есть нулевой популяции выбирается последняя популяция первого этапа. Каждая особь помещается в поле модели, и ведётся подсчёт количества шагов. Отбираются особи, совершившие наибольшее число шагов. Из лучших особей при помощи скрещивания создаётся новая популяция. К каждой особи новой популяции применяется мутация. Скрещивание и мутация применяются ко всем весам. Цикл продолжается до тех пор, пока не будет достигнут критерий останова. На рис. 4 показана структура применения ГА для обучения РНС.

Для реализации ГА были написаны программа MOVE_RNN на языке C++. Тест программы проводился на компьютере с процессором Intel Core i5 7200u, ОЗУ 8Гб.
Было проведено десять тестов, для каждого теста были выбраны параметры:

1. Популяция равна 50000.

Для первого этапа обучения

2. Пороговое значение для кроссинговера 0,5.
3. Пороговое значение для мутации 0,5.
4. Количество поколений — 50000.
5. Точность 0,999.

Для второго этапа

6. Количество поколений 100000.

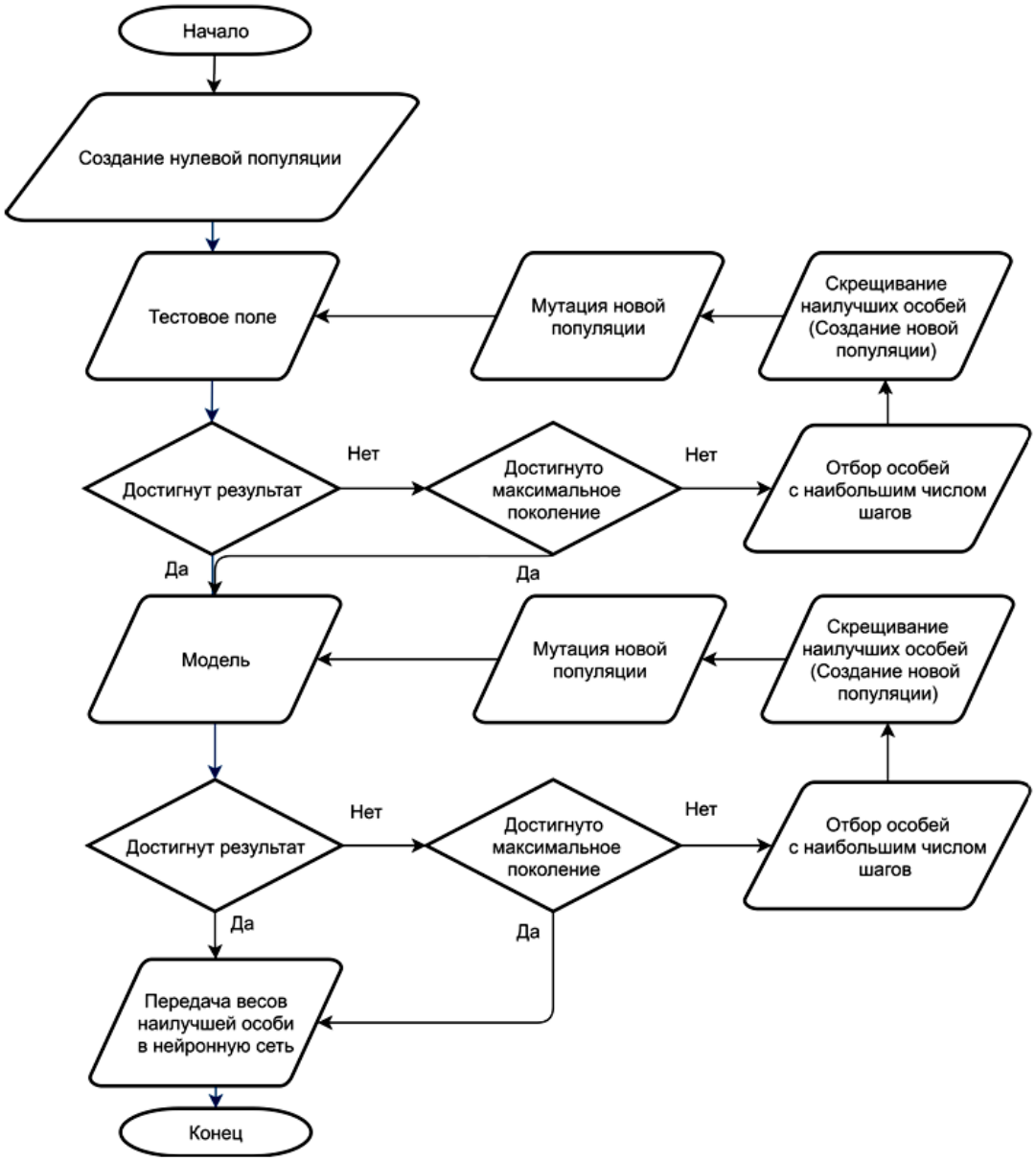

Рис. 4. Структурная схема применения генетического алгоритма для создания рекуррентной нейронной оси

Данная архитектура РНС позволила решить проблему с зацикливанием ходов, однако при этом снизилась точность правильных решений. Из 10 тестов успешно были пройдены 7. Тест считается успешно пройдённым, если объект под управлением нейронной

сетью сделал шагов больше, чем четырёхкратное значение площади. Не пройдённые тесты заканчивались заходом объекта на препятствие.

Можно сделать вывод, что это происходило из-за того, что в блоке памяти РНС в ходе последовательных вычислений накапливаются значения, которые становятся шумом и мешают принять верное решение.

4. ЗАКЛЮЧЕНИЕ

Применение нейронной сети созданной с помощью генетического алгоритма, показало, что при ограниченных входных данных изучаемый объект способен выполнять свой функционал, то есть проходить путь по полю с препятствиями. Изменение архитектуры нейронной сети с полносвязной сети прямого распространения на рекуррентную нейронную сеть уменьшило вероятность прохождения объекта по клеткам, на которых он был ранее.

Список литературы

1. *Чернодуб А. Н., Дзюба Д. А.* Обзор методов нейроуправления // Проблемы программирования. 2011. № 2. С. 79–94.
2. Классические задачи Computer Science на языке Python. СПб.: Питер, 2020. 256 с.
3. *Килин Г. А., Ждановский Е. О.* Преимущества использования обучения с подкреплением для обучения нейронных сетей // Материалы всероссийской научно-технической конференции «Автоматизированные системы управления и информационные технологии». Пермь, 17 мая 2018 года. Пермь: Пермский национальный исследовательский политехнический университет, 2018. Т. 1. С. 152–158.
4. *Пратик Д.* Искусственный интеллект с примерами на Python. СПб.: ООО «Диалектика», 2019. 448 с.
5. *Николенко С. И. Тулупьев А. Л.* Самообучающиеся системы. М.: МЦНМО, 2009. 288 с.
6. *Николенко С., Кадурин А., Архангельская Е.* Глубокое обучение. СПб.: Питер, 2018. 480 с.
7. *Гудфеллоу Я., Бенджио И., Курвилль А.* Глубокое обучение / пер. с анг. А. А. Слинкина. 2-е изд., испр. М.: ДМК Пресс, 2018. 652 с.

Поступила в редакцию 22.02.2022, окончательный вариант — 24.03.2022.

Ляхов Александр Фёдорович, кандидат физико-математических наук, доцент кафедры теоретической, компьютерной, экспериментальной механики Институт Информационных технологий, математики и механики ННГУ, ✉ **Alf19545@rambler.ru**

Королев Денис Алексеевич, инженер-программист ООО НПП «ПРИМА»,
✉ **corolyov1998@gmail.com**

Computer tools in education, 2022
№ 1: 5–15
http://cte.eltech.ru
doi:10.32603/2071-2340-2022-1-5-15

Controlling the Movement of an Object on a Field with Barrier Using a Recurrent Neural Network

Lyakhov A. F.[1], PhD, Associate Professor, ✉ Alf19545@rambler.ru
Korolev D. A.[2], Software Engineer, ✉ corolyov1998@gmail.com

[1]Lobachevsky State University of Nizhny Novgorod, 23 Prospekt Gagarina, 603022, Nizhny Novgorod, Russia
[2]PRIMA Research & Production Enterprise, LLC, 1Zh, Sormovskoye Shosse, 603950, Nizhny Novgorod, Russia

Abstract

Consider control model recurrent neural network moving object on a field with barrier using a recurrent neural network. Via genetic algorithm create two neural network different complexity. For each neural network describe algorithm reinforcement learning. Comparison of the effectiveness of their work.

Keywords: *neural network, reinforcement learning, genetic algorithm, movement object, artificial intelligence.*

Citation: A. F. Lyakhov and D. A. Korolev, “Controlling the Movement of an Object on a Field with Barrier Using a Recurrent Neural Network,” *Computer tools in education*, no. 1, pp. 5–15, 2022 (in Russian); doi: 10.32603/2071-2340-2022-1-5-15

References

1. A. N. Chernodub and D. A. Dzyuba, “Review of neurocontrol methods,” *Problemy programmirovaniya*, no. 2, pp. 79–94, 2011 (in Russian).
2. D. Kopec, *Classic Computer Science Problems in Python*, St. Petersburg, Russia: Piter, 2020 (in Russian).
3. G. A. Kilin and E. O. Zhdanovsky, “Benefits of using reinforcement learning for training neural networks,” in Proc. *Review of Automated control systems and information technologies neurocontrol methods, Perm, Russia, 17 May 2018*, Perm, Russia: Perm National Research Polytechnic University, vol. 1, 2018, pp. 152–158 (in Russian).
4. J. Prateek, *Artificial intelligence with Python*, St. Petersburg, Russia: OOO "Dialektika 2019 (in Russian).
5. S. I. Nikolenko and A. L. Tulup’ev, *Self-learning systems*, Moscow: MCCME, 2009 (in Russian).
6. S. Nikolenko, A. Kadurin, and E. Arkhangelskaya, *Deep learning*, St. Petersburg, Russia: Piter, 2018 (in Russian).
7. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, Moscow: DMK Press, 2018 (in Russian).

Received 22-02-2022, the final version — 24-03-2022.

Alexander Lyakhov, PhD, Associate Professor, Institute of Information Technologies, Mathematics and Mechanics Department of Theoretical, Computer, Experimental Mechanics, Lobachevsky State University of Nizhny Novgorod, ✉ Alf19545@rambler.ru

Denis Korolev, Software Engineer PRIMA RPE LLC, ✉ corolyov1998@gmail.com