Санкт-Петербургский государственный университет
Прикладная математика и информатика
Статистическое моделирование

Бакшинская Екатерина Олеговна

Имитационный подход к решению стратегических игр с неполной информацией

Отчет об учебной практике (научно-исследовательской работе)

Научный руководитель:
к.ф.-м.н., доцент П. В. Шпилев

Санкт-Петербург
2018

Оглавление

Введение

Классическая теория игр — дисциплина, которая изучает оптимальные стратегии в играх. Игра — это процесс, в котором за реализацию своих интересов соревнуются две и более сторон(игроков, коалиций игроков). У каждого из игроков есть своя цель и свой набор стратегий(т.е. множества доступных игрокам действий), который может привести сторону к выигрышу или проигрышу, что зависит от набора стратегий оппонентов. Целью теории игр является выработка методов нахождения оптимальных стратегий (таких, которые с наибольшей вероятностью приведут к выигрышу).

Существуют различные типы игр (см. например [3], [15]). В данной работе мы будем рассматривать стратегические игры с неполной информацией ([10], [5]). Один из самых серьезных недостатков классической теории игр заключается в ее неспособности рассматривать игры, включающие неполную информацию [4]. Такие игры чаще всего возникают в реальном мире, мы не всегда знаем об истинных состояниях процесса и можем оценивать результаты действий соперников только по некоторым ограниченным наборам данных. В этих условиях принятие решений может зависеть от нескольких факторов, в том числе — случайных. Один из возможных подходов к нахождению оптимальной стратегии в таких задачах это использование для описания исследуемой ситуации вероятностных моделей ([9]).

Игры в жизни — это не только спорт, шашки или карты, в жизни это стратегическое мышление, это важная для каждого способность находить способы сотрудничества и проектировать дальновидные стратегии, осмысливать сигналы и предугадывать ходы противника, который в свою очередь старается угадать ваши мысли. Рабочие отношения, политика, конкуренция в различных областях, продажи и т.д., теория игр применима практически везде, поэтому действительно важно уметь работать с ней.

В предыдущей моей работе ([9]) я использовала скрытые марковские модели для решения подобного рода игр. В данной работе для решения таких задач будут применяться алгоритмы нейронных сетей.

Нейронная сеть — это большой распределенный параллельный процессор, который состоит из элементарных единиц, обрабатывающих информацию, накапливающих экспериментальные знания и предоставляющих их для последующей обработки. Нейронную сеть сравнивают с мозгом, так как [7]:

- Знания в нее поступают из окружающей среды и используются в процессе обучения.
- Для накопления знаний применяются связи между нейронами, называемые синаптическими весами.

Преимущества нейронных сетей [7]:

- *Нелинейность*. Отличие от линейных статистических методов (линейная регрессии и

др.) состоит в том, что с помощью нейронных сетей можно строить нелинейные зависимости, которые более точно описывают наборы данных.

- *Отображение входной информации в выходную.* В случае обучения с учителем нейронная сеть учится на известных зависимостях и строит таблицу соответствий между входными и выходными данными для конкретной задачи.

- *Адаптивность.* При изменении окружающего мира нейронная сеть способна изменить свои веса для того, чтобы поддерживать качество предсказаний на необходимом уровне.

- *Отказоустойчивость.* При неблагоприятных условиях (например, поврежден какой-то нейрон) качество нейронных сетей ухудшается незначительно, благодаря распределенному характеру хранения информации.

- *Единообразие анализа и проектирования.* Архитектура нейронных сетей позволяет использовать одно и то же проектное решение в различных предметных областях и задачах.

Нейронные сети применяются в различных областях, таких как например, экономика ([8], [1]), бизнес, медицина, связь, интернет, немалый вклад они внесли и в теорию игр ([6],[2]). В данной работе мы рассмотрим различные алгоритмы нейронных сетей, применимых для нашей задачи, описанной в [9].

Связанные работы

В конце 1960–х годов Джон Харшаньи ввел понятие игр с неполной информацией и разработал концепцию байесовских равновесий. Д. Харшаньи рассматривал ситуации, когда у одного игрока нет информации о возможных выигрышах другого игрока, и выигрыши приходится оценивать вероятностно [5].

На практике часто встречаются ситуации, которые в теории игр принято называть повторной игрой (repeated game) т.е. игрой, которая состоит в некотором числе повторений некоторых игровых стадий (stage game). Впервые повторные игры с неполной информацией были введены Ауманом и Машлером в 1960г.[10]. Рассматривались игры, в которых начальное состояние выбиралось с вероятностью p, и только игрок 1 знает это состояние, в то время как игрок 2 знает все возможные состояния, но не знает фактического состояния соперника. После каждого хода, оба игрока знают исход предыдущего хода и продолжают играть.

Математические игры в наше время встречаются практически во всех сферах деятельность, поэтому им уделяется немалое внимание. В [13] написано о проблемах, которые возникают в теории игр, применимой в реальном мире, а именно: эффективное принятие решений практически в любой задаче почти всегда требует нелинейных отображений входных данных в ответы (выходные данные). В статье показано, что нейронные сети и эволюционные алгоритмы обеспечивают полезные средства для решения этих проблем. С помощью нейронных сетей описываются соответствующие стратегии в играх с нулевой и не нулевой суммой (*игры с нулевой суммой* — игры, в которых выигрыш одной стороны равен проигрышу другой).

В [6] рассматривается случай многошаговых игр нескольких игроков с неполной информацией, в статье обсуждались алгоритмы обучения нейронной сети для нахождения оптимальных стратегий. Наибольшую популярность в применимости к теории игр имеют рекуррентные нейронные сети [12], позволяющие наиболее эффективно решать поставленные задачи. О рекуррентных нейронных сетях будет подробно написано в Главе 1.

Нейронные сети уже были применены к различным реальным играм, например в карточной игре “дурака” [2], игре в шашки [14]. Всемирно известная нейронная сеть AlphaGo от Google одержала победу над одним из сильнейших игроков мира в игре го, а 19 октября 2017 года была опубликована статья [17], в которой показано, что новая модель AlphaGo Zero не только обыгрывает прошлые версии нейронной сети, но ещё и не требует никакого человеческого участия в процессе тренировки. Эта нейронная сеть так же была применена к игре в шахматы и другим играм.

Глава 1

Теоретическое описание

1.1. Постановка задачи

Цель работы: применение моделей *искусственного интеллекта (ИИ)* для решения задач теории игр с неполной информацией на основе математического аппарата нейронных сетей.

Задачи.

1. Изучение существующих моделей и алгоритмов обучения искусственного интеллекта.
2. Изучение типов нейронных сетей, необходимых для построения ИИ.
3. Построение различных моделей игр с неполной информацией.
4. Реализация конкретных моделей ИИ.
5. Сравнение и анализ разработанных моделей.

1.2. Нейронные сети

Нейронные сети — это набор алгоритмов, смоделированных по принципу работы человеческого мозга, которые предназначены для распознавания паттернов. Образцы, которые они распознают, являются числовыми, содержащимися в векторах, в которые должны быть переведены все реальные данные, будь то изображения, звук, текст или временные ряды.

Глубокое обучение — «сложенные нейронные сети»; то есть сети, состоящие из нескольких слоев. Слои состоят из узлов (нейронов). Нейрон (узел) объединяет входные данные с набором коэффициентов или весов, которые либо усиливают, либо ослабляют этот входной сигнал, тем самым назначая значимость входным данным для задачи, которую алгоритм пытается решить. (Например, какой вход является наиболее полезным, классифицирует данные без ошибок?) Эти произведения весов с входными данными суммируются, и сумма передается через так называемую функцию активации нейрона, чтобы определить, будет ли и в какой степени этот сигнал продвигаться дальше через сеть, чтобы повлиять на конечный результат.

Схема того, как может выглядеть один нейрон представлена на рис.1.1.

Функционирование нейрона можно описать следующим образом:

$$u = \sum_{i=0}^{n} w_i x_i,$$

$$y = \varphi(u + b),$$

где $x_1, ..., x_n$ — входные данные, $w_1, ..., w_n$ — веса, с которыми эти данные далее суммируются, v — сумматор (линейная комбинация входов с весами), φ — функция активации, y — выход нейрона, b — *порог (смещение)*, значение которого позволяет нам "двигать" функцию активации влево или вправо (аналогично параметру b в уравнении $y = ax + b$).

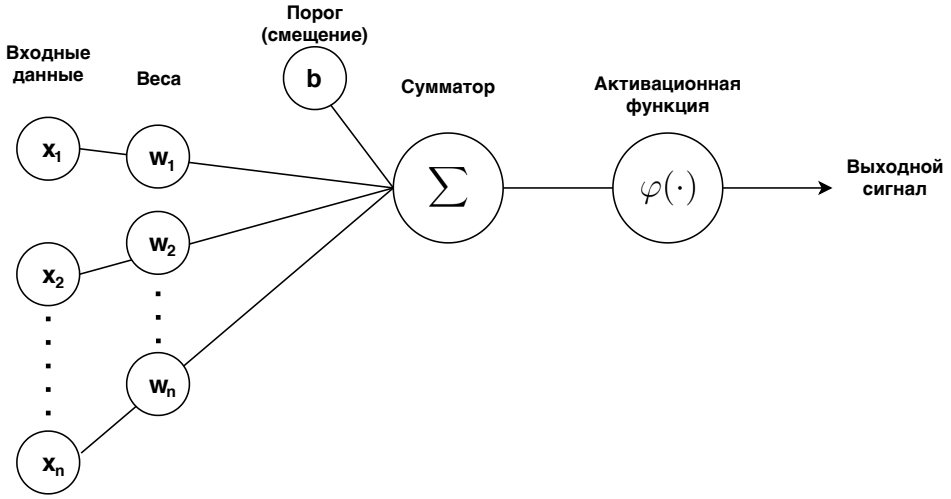

Рис. 1.1. Нелинейная модель нейрона с порогом

Для удобства смещение часто представлют в виде еще одного входа $x_0 = 1$ и веса для него $w_0 = b$, в таком случае функционирование нейрона записывается следующим образом:

$$v = u + b,$$

$$y = \varphi(v).$$

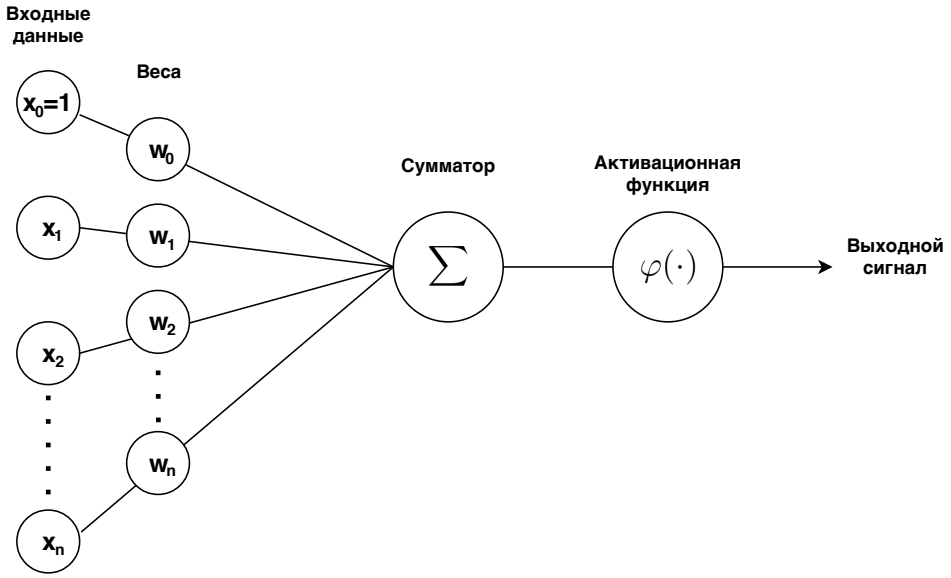

Рис. 1.2. Нелинейная модель нейрона

Некоторые типы нейронов (графики которых представлены на рис.1.3):

1. *Перцептрон.* Функция активации (она претерпевает разрыв в 0, не дифференцируема):

$$\varphi(v) = \begin{cases} 1 & \text{если } v \geq 0 \\ 0 & \text{если } v < 0 \end{cases}.$$

2. *Сигмоидальный нейрон.* Функция активации (непрерывна, дифференцируема и нелинейна):

$$\varphi(v) = \sigma(v) = \frac{1}{1 + e^{-v}}.$$

3. *Гиперболический тангенс.* Функция активации (непрерывна, дифференцируема и нелинейна):

$$\varphi(v) = \tanh(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}.$$

4. *ReLU (Улучшенный линейный нейрон).* Функция активации:

$$\varphi(v) = \max(v, 0) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}.$$

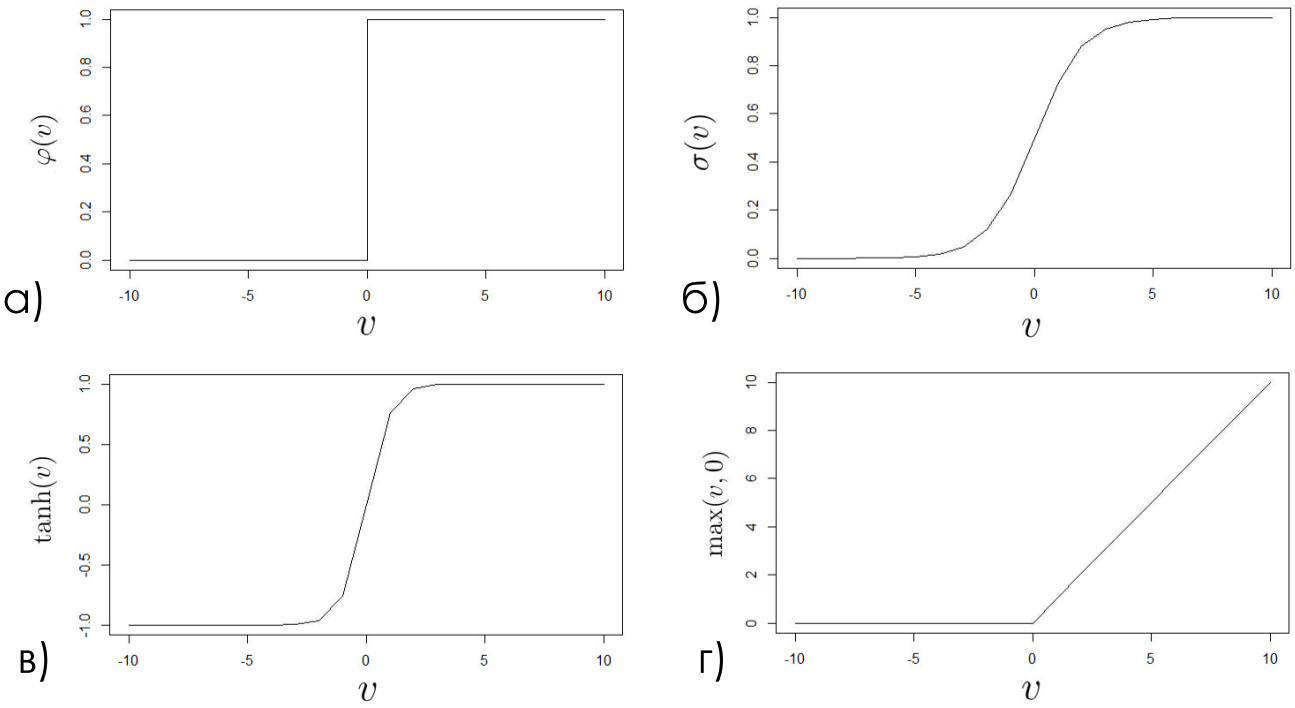

Рис. 1.3. Некоторые типы нейронов: Перцептрон (а); Сигмоидальный нейрон (б); Гиперболический тангенс (в); ReLU (Улучшенный линейный нейрон) (г).

5. *Стохастическая модель нейрона.* Функция активации здесь имеет вероятностную интерпретацию.

$$x = \begin{cases} 1 & \text{с вероятностью } P(v) \\ -1 & \text{с вероятностью } 1 - P(v) \end{cases},$$

где x — состояние нейрона, $P(v) = \frac{1}{1+\exp(\frac{-v}{T})}$ — вероятность активации нейрона, в которой T — используется для управления уровнем шума, и, таким образом, степенью неопределенности переключения (если параметр $T \to 0$, то эта модель превращается в перцептрон).

1.3. Структура нейронных сетей

Более ранние версии нейронных сетей, такие как первые перцептроны, не были глубокими, состояли из одного входного и одного выходного слоя с не более чем одним скрытым слоем между ними. Более трех слоев (включая ввод и вывод) в нейронной сети квалифицируются как *глубокое обучение*.

В сетях с глубоким обучением каждый слой узлов обучается определенному набору функций на основе выходных данных предыдущего уровня. Чем дальше вы продвигаетесь по нейронной сети, тем сложнее объекты, которые могут распознаваться вашими узлами, поскольку они объединяют и комбинируют объекты из предыдущего уровня.

1.3.1. Однослойные сети прямого распространения

Однослойные сети прямого распространения — простейшие нейронные сети, в которых имеются входные данные (входной слой), и от них передается информация через сумматоры и функции активации на выходной слой.

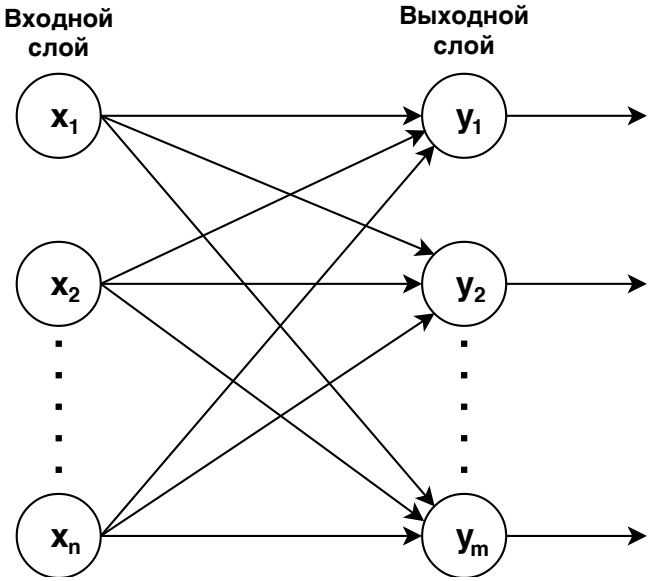

Рис. 1.4. Пример однослойной сети прямого распространения

1.3.2. Многослойные сети прямого распространения

Многослойные сети прямого распространения имеют один или несколько скрытых слоев, узлы которых называются скрытыми нейронами. Многослойная нейронная сеть способна моделировать функцию практически любой степени сложности, причем число слоев и число элементов в каждом слое определяют сложность функции. Такая сеть позволяет выделять глобальные свойства данных с помощью локальных соединений за счет наличия дополнительных синаптических связей и повышения уровня взаимодействия нейронов [7].

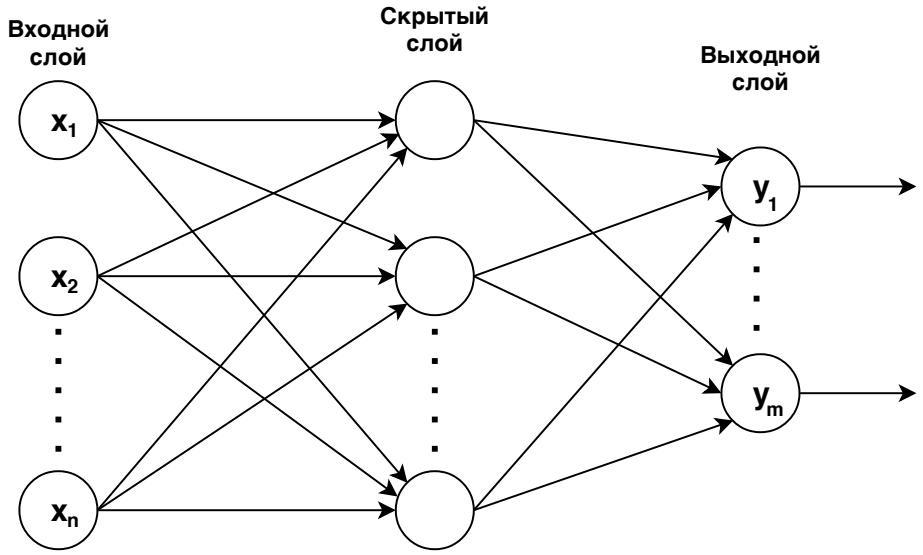

Рис. 1.5. Пример многослойной сети прямого распространения

1.3.3. Рекуррентные нейронные сети

Часть сигналов в *рекуррентных нейронных сетях (РНС)* (сетях обратного распространения) со скрытых слоев нейронов или выходных слоев возвращается на входы нейронов входного слоя. Идея РНС заключается в последовательном использовании информации. В сетях прямого распространения считается, что все входы и выходы независимы. Однако это накладывает серьезные ограничения на многие задачи. Допустим, если мы хотим предсказать следующий ход соперника, то зачастую нам стоит учитывать его предыдущие ходы. Такие нейронные сети — рекуррентные, так как они производят одно и то же действие над каждым элементом последовательности и выход нейронной сети зависит от предыдущих вычислений. Еще один взгляд на РНС заключается в следующем: это сети, у которых есть «память», способная учитывать предыдущую информацию.

Как уже было сказано, нейронные сети прямого распространения состоят из входного, скрытых и выходного слоев. У РНС схожая структура, за исключением того, что добавляется еще слой *временной задержки*. Например, скрытый слой связан с временной задержкой. Мы посылаем сигналы от входного слоя на скрытый, скрытый слой посылает обработанную информацию на слой временной задержки и на выходной слой. В следующий раз, когда мы посылаем опять сигналы, информация идёт от входного слоя к скрытому, да и ещё от слоя

задержки идут сигналы через такие же синапсы(веса). После этого скрытый слой обрабатывает информацию, так же посылает новые сигналы на слой временной задержки и на выходной слой [7].

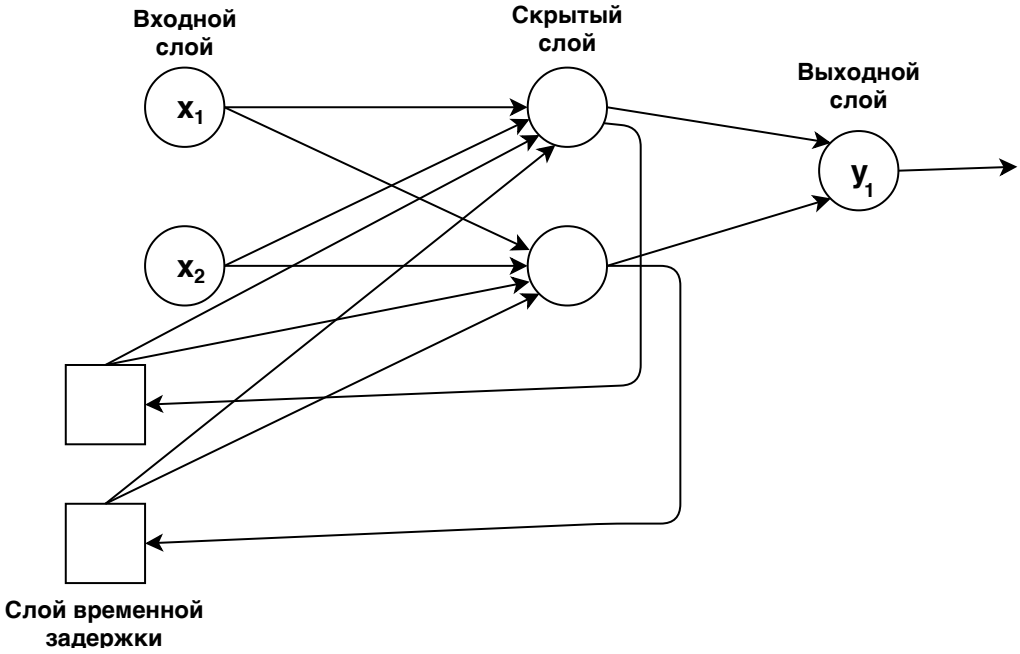

Рис. 1.6. Пример рекуррентной сети

Такие сети наиболее подходят для решения многошаговых игр с неполной информацией (в таких играх игра состоит более, чем из одного хода и для них нам чаще всего важна последовательность ходов), поэтому их мы и будем рассматривать далее.

1.4. LSTM

Одна из привлекательных сторон *рекуррентных нейронных сетей (РНС)* заключается в том, что они могут соединить предыдущую информацию с текущей задачей, например, предыдущие ходы противника могут лучше помочь понять его стратегию.

Иногда нам нужно только взглянуть на недавнюю информацию, чтобы выполнить текущую задачу. Например, можно рассмотреть модель, в которой состояние противника в игре зависит только от его трех последних состояний. В таких случаях, когда разрыв между соответствующей информацией и местом, в котором она необходима, невелик, РНС могут научиться использовать прошлую информацию.

Но есть также случаи, когда нам нужно больше прошлой информации, бывают задачи, в которых разрыв между соответствующей информацией и местом, в котором она необходима, становится очень большим. К сожалению, по мере того, как этот разрыв увеличивается, РНС становятся неспособными научиться соединять информацию. Эта проблема была подробно исследована в [11], в которой было найдено несколько довольно фундаментальных причин, почему это может быть затруднительно.

К счастью, у LSTM нет этой проблемы!

Сети с долговременной кратковременной памятью, обычно называемые просто "*LSTM*", представляют собой особый тип РНС, способный изучать долгосрочные зависимости. Они были представлены в [16], и с тех пор были усовершенствованы и популяризированы многими людьми. Они отлично справляются с множеством задач и в настоящее время широко используются. LSTM явно разработаны, чтобы избежать проблемы долгосрочной зависимости. Запоминание информации в течение длительных периодов времени является практически их поведением по умолчанию, а не тем, что они пытаются усвоить!

Все рекуррентные нейронные сети имеют форму цепочки повторяющихся модулей нейронной сети. Единицы (или блоки) долговременной памяти (LSTM) являются строительными единицами для слоев рекуррентной нейронной сети (РНС). РНС, состоящие из блоков LSTM, часто называют *сетью LSTM*. Обычный модуль LSTM состоит из ячейки, входного вентиля, выходного вентиля и логического вентиля забывания. Ячейка отвечает за "запоминание"значений за произвольные промежутки времени; отсюда и слово "память"в LSTM. Каждый из этих трех элементов можно рассматривать как обычный искусственный нейрон, как в многослойной (или однослойной) нейронной сети: то есть они вычисляют активацию (используя функцию активации) взвешенной суммы. Выражение "долговременная кратковременная" в названии LSTM относится к тому факту, что LSTM является моделью для краткосрочной памяти, но при этом может сохранять информацию в течение длительного периода времени.

Таким образом, ячейка LSTM содержит следующие компоненты:

1. Вентиль забывания f (нейронная сеть с сигмоидой).
2. Слой новых значений $\hat{C}$: кандидатов, которые могут быть добавлены в ячейку (нейронная сеть с tanh).
3. Входной вентиль i (нейронная сеть с сигмоидой).
4. Выходной вентиль O (нейронная сеть с сигмоидой).
5. Скрытое состояние H (вектор).
6. Состояние памяти C (вектор).

Входами в ячейку LSTM на этапе времени t являются X_t (текущий вход), H_{t-1} (предыдущее скрытое состояние) и C_{t-1} (предыдущее состояние памяти). Выходами из ячейки LSTM являются H_t (текущее скрытое состояние) и C_t (текущее состояние памяти).

Ниже на рис.1.7 представлена диаграмма ячейки LSTM на этапе времени t. Работа ячейки описывается следующим образом:

"$*$"— обозначает поэлементное умножение, "$+$"— поэлементное сложение.

$$f_t = \sigma(X_t * U_f + H_{t-1} * W_f),$$

$$\hat{C}_t = \tanh(X_t * U_c + H_{t-1} * W_c),$$

$$i_t = \sigma(X_t * U_i + H_{t-1} * W_i),$$

$$O_t = \sigma(X_t * U_o + H_{t-1} * W_o),$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t,$$

$$H_t = O_t * \tanh(C_t).$$

W, U — векторы весов для вентиля забытия (f), новых значений $(\hat{C})$, входного вентиля (i) и выходного вентиля (O). Стоит отметить, что это все разные веса для разных вентилей.

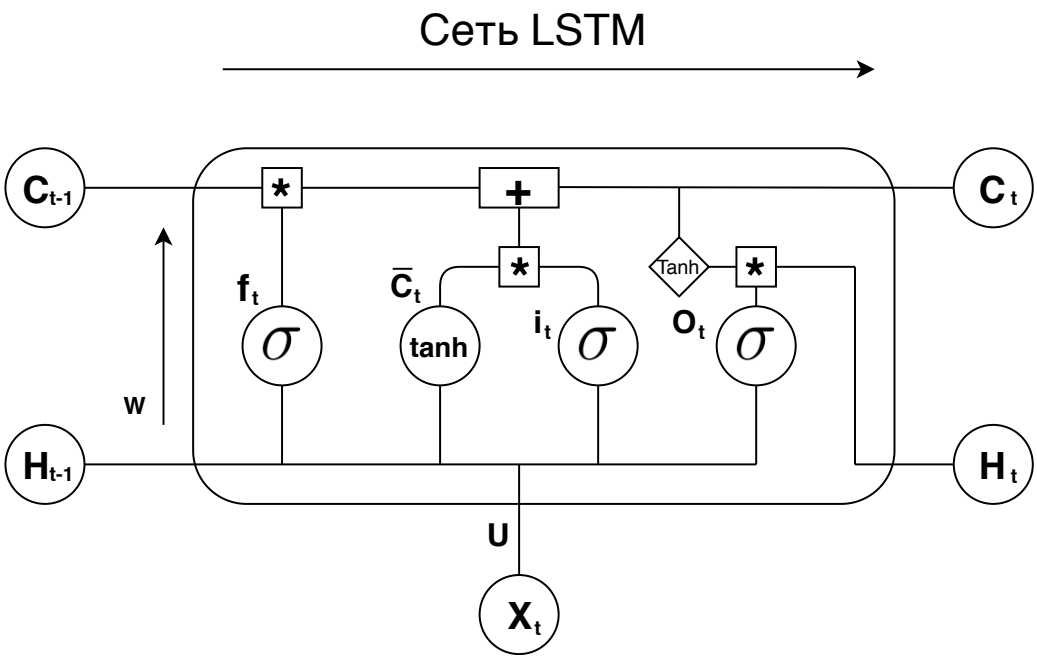

Рис. 1.7. Ячейка LSTM на этапе времени t

Глава 2

Реализация и вычислительные эксперименты

2.1. LSTM на примере игры в “теннис”

В моей прошлой работе ([9]) был построен алгоритм решения игр с неполной информацией, описанных скрытыми марковскими моделями. Сейчас же нам интересно применить LSTM для рассматриваемой ранее задачи.

2.1.1. Описание игры

В рассматриваемом примере игры есть 2 игрока (подающий и принимающий). Набор стратегий для каждого из них:

1. подающий может подать мяч в центр (Центр) или в открытую зону (Открытый);
2. принимающий должен быть готов к одному из этих действий.

У подающего есть 3 состояния: Агрессивный, Спокойный или Защитный (переход от состояния к состоянию происходит после каждого хода с заранее заданными вероятностями). Для каждого состояния имеются свои вероятности подать мяч в центр или в открытую зону. Наша задача состоит в том, чтобы предсказать направление подачи (центр или открытая зона).

Модель подающего игрока, для которой применялись LSTM:

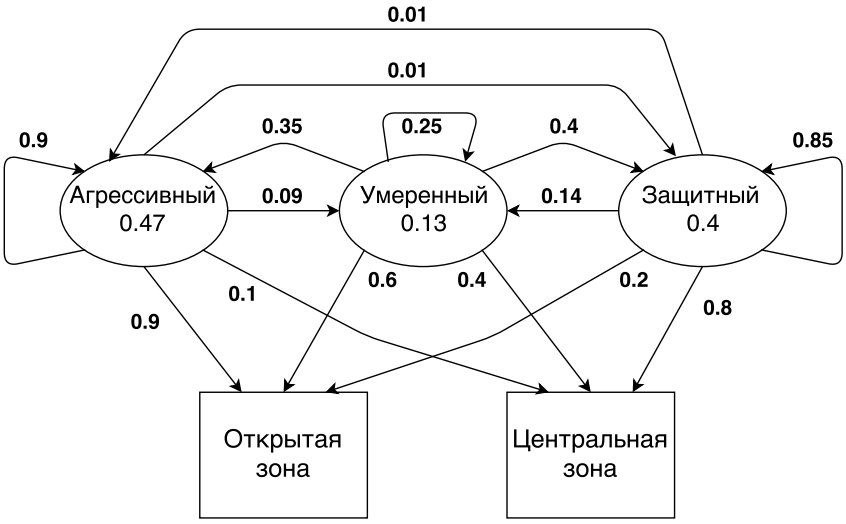

Рис. 2.1. Модель игры подающего игрока в игре “теннис”

Игра будет состоять из 10000 ходов, в каждый из которых игроки одновременно делают свой ход, для оценки качества обучения LSTM нас будет интересовать количество ударов, отраженных принимающим игроком, т.е. сколько ходов из 10000 оказались выигрышными для него.

2.1.2. Применение LSTM

Для начала была построена сеть LSTM с одной LSTM ячейкой, в которой 2 входа и два выхода. Активационная функция: *softmax* — функция принимает ненормированный вектор, и нормирует его в распределение вероятностей. То есть до применения softmax некоторые векторные элементы могут быть отрицательными или больше единицы; но после применения softmax каждый элемент x_i находится в интервале $[0, 1]$, а также $\sum_i x_i = 1$. Softmax часто используется в нейронных сетях, чтобы отобразить ненормированный выходной сигнал в распределение вероятностей по прогнозируемым выходным классам.

После обучения LSTM каждые 100 ходов игры вычислялось количество отраженных ударов, после чего был построен следующий график на котором можно видеть как изменялась доля отраженных ударов в процессе обучения LSTM.

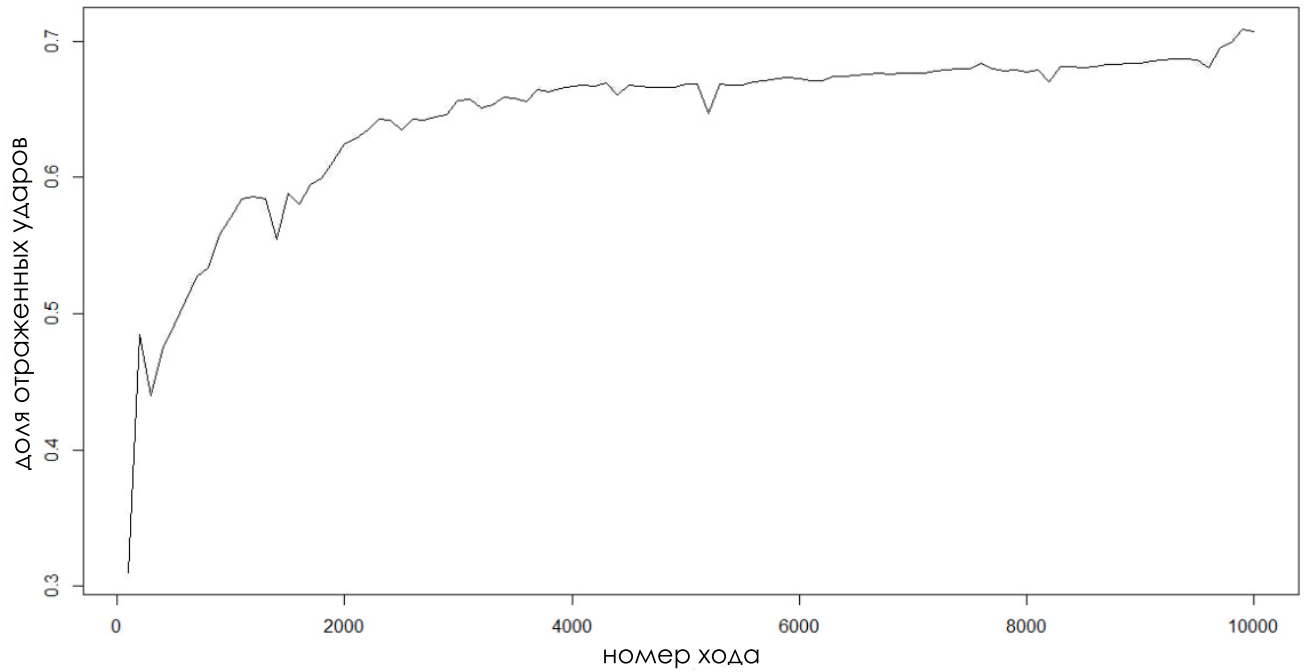

Рис. 2.2. Доля отраженных ударов во время игры

В итоге количество отраженных ударов составило 70.6%. Этот результат хуже, представленного в работе [9]. Таким образом, ставится задача: оптимизировать модель LSTM для решения поставленной задачи.

Заключение

Таким образом, в данной работе:

1. проведен обзор литературы на близкие темы;
2. изучены нейронные сети, архитектура нейронных сетей;
3. изучены сети LSTM как наиболее подходящий алгоритм решения поставленной задачи;
4. в практической части: были применены сети LSTM для решения гипотетического примера игры в “теннис”.

В планах:

- улучшение предложенной модели LSTM для решения гипотетического примера игры в “теннис”;
- применение сетей LSTM для решения других игр с неполной информацией;
- применение других подходов к решению поставленных задач (например, сети GRU);
- выявление лучшего подхода и его оптимизация.

Список литературы

1. Кравченко Мария Леонидовна, Грекова Татьяна Ивановна. Моделирование экономических систем с применением нейронных сетей // Вестник Томского государственного университета. — 2006. — № 290.
2. Ляхов Александр Федорович, Тришин Илья Михайлович. Компьютерное моделирование поведения игрока в интеллектуальной карточной игре с помощью нейронной сети // Компьютерные инструменты в образовании. — 2013. — № 5.
3. Фон-Нейман Дж, Моргенштерн О. Теория игр и экономическое поведение. Пер. с англ. под ред. и с доб. Н. Н. Воробьева. — М. : Наука, 1970.
4. Зельтен Р, Харшаньи Дж. Общая теория выбора равновесия в играх // книга. — 2001.
5. Харшаньи Дж. Игры с неполной информацией // Мировая экономическая мысль. Всемирное признание (лекции нобелевских лауреатов). — 2005. — Т. 5. — С. 44–63.
6. Терехов СА. Адаптивные нейросетевые методы в многошаговых играх с неполной информацией // В сб.:«Лекции по нейроинформатике».–М.: Изд-во МИФИ. — 2005. — С. 111–139.
7. Хайкин Саймон. Нейронные сети: полный курс, 2-е издание. — Издательский дом Вильямс, 2008.
8. Николаева ИВ. Применение искусственных нейронных сетей для прогнозирования динамики экономических показателей // Сфера услуг: инновации и качество. — 2012. — № 8. — С. 22–22.
9. Е. Бакшинская. Использование CMM в играх с неполной информацией // Third Conference on Software Engineering and Information Management (SEIM-2018) (short papers), Saint Petersburg, April 14, 2018 / Под ред. Yurii Litvinov, Marat Akhin, Boris Novikov, Vladimir Itsykson. — ООО Цифровая фабрика “Быстрый Цвет ”, 2018. — С. 15–23.
10. Aumann Robert J, Maschler Michael, Stearns Richard E. Repeated games with incomplete information. — Cambridge, MA. : MIT press, 1995.
11. Bengio Yoshua, Simard Patrice, Frasconi Paolo. Learning long-term dependencies with gradient descent is difficult // IEEE transactions on neural networks. — 1994. — Vol. 5, no. 2. — P. 157–166.
12. Bhatia Sudeep, Golman Russell. A recurrent neural network for game theoretic decision making // Proceedings of the Annual Meeting of the Cognitive Science Society. — Vol. 36. — 2014.
13. Chellapilla Kumar, Fogel David B. Evolution, neural networks, games, and intelligence // Proceedings of the IEEE. — 1999. — Vol. 87, no. 9. — P. 1471–1496.
14. Chellapilla Kumar, Fogel David B. Evolving neural networks to play checkers without relying on expert knowledge // IEEE transactions on neural networks. — 1999. — Vol. 10, no. 6. — P. 1382–1391.

15. Gibbons Robert. A primer in game theory. — Harvester Wheatsheaf, 1992.
16. Hochreiter Sepp, Schmidhuber Jürgen. Long short-term memory // Neural computation. — 1997. — Vol. 9, no. 8. — P. 1735–1780.
17. Mastering the game of Go without human knowledge / David Silver, Julian Schrittwieser, Karen Simonyan et al. // Nature. — 2017. — Vol. 550, no. 7676. — P. 354.