Дорогая Мэри, как говорится в песне,
«…посвящаю себя тому, кого я люблю».
В какое чудесное приключение ты превратила мою жизнь!
С глубочайшей любовью и восхищением.

– Дэнни

Сияющим звездам моей жизни: Аруне, Ришме и Арье Нине –
и моим постоянным четвероногим спутникам Чикки и Хиру.
Без вас вокруг была бы только тьма.

– Анураг

Оглавление

Мы случайно наткнулись друг на друга в пятницу, 13 апреля 2018 года, на многолюдном официальном открытии Школы информатики, вычислительной техники и инженерии Luddy, и решили написать книгу по машинному обучению после очень глубокого и обстоятельного разговора, последовавшего сразу за окончанием мероприятия:

Анураг: Я хотел бы написать с тобой книгу из серии *The Little Book*.

Дэн: Так давай напишем!

⋮

Несколько секунд спустя

⋮

Дэн: На какую тему?

Анураг: Машинное обучение.

Дэн: Ну ладно, интересная тема!

А потом мы долго болтали о всякой всячине…

Предисловие Гая Стила-младшего

Эта книга – именно то, что нужно.

Дэн Фридман уже более четырех десятилетий пишет книги в своем уникальном стиле *The Little Book*[1] в компании талантливых и опытных соавторов. Каждая из них – истинная жемчужина, объясняющая глубокие и важные идеи в области компьютерных наук небольшими порциями. Дэн и его соавторы возвели формат вопрос–ответ в ранг искусства, до уровня беседы, кажущейся почти беззаботной. Кроме того, в этой книге представлены два новшества – каркасный код функций и логические столбцы, – которые еще больше упрощают представление фрагментов программного кода и их поведения.

Что же касается фундаментальных идей машинного обучения, то в этой книге они представлены именно в том виде и именно в том порядке, который вам необходим.

Поэтому читайте книгу последовательно (не забегая вперед!). Сами авторы отмечают:

> *Наши «книжки-малышки» предназначены для аккуратной упаковки небольших идей в маленькие коробочки.*

Какие идеи изложены в этой книге? Конечно же, математика и вычислительные методы машинного обучения: вы узнаете о последовательной аппроксимации, стохастическом градиентном спуске, нейронных сетях и автоматическом методе обратного распространения ошибки. Однако как специалисту по языкам программирования мне также интересно, как авторы используют язык программирования для изложения математики. На мой взгляд, ключевая и всеобъемлющая идея книги заключается в том, как авторы используют функции высшего порядка, то есть функции, которые принимают другие функции в качестве аргументов и/или возвращают другие

[1] С разговорного английского *The Little Book* переводится приблизительно как «книжка-малышка», а еще в Америке так называют адаптированное карманное издание Евангелия. И хотя книги этой серии вполне обычные по количеству страниц, информация в них разбита на маленькие пронумерованные «фреймы», а изложение следует шаг за шагом и сопровождается забавными рисунками и отступлениями, не давая читателю заскучать. – *Прим. перев.*

функции в качестве результатов, для объяснения структур данных и вычислений в машинном обучении.

Фундаментальной структурой данных является тензор, который на первый взгляд выглядит как обычный массив или матрица; но авторы поясняют, что тензорные операции обладают дополнительным свойством автоматического использования функции отображения более высокого порядка, когда это необходимо, и это – почти волшебным образом, как мне кажется, – позволяет использовать функцию, явно написанную для скаляров (одиночных чисел) или тензора определенного ранга, которую без дополнительных усилий можно применять ко всем типам и размерам массивов, векторов и матриц.

Другим применением функций высшего порядка является каррирование[1], которое позволяет вам передать функции некоторые из ее аргументов сейчас, а другие позже – когда вы передаете ей некоторые из ее аргументов, она возвращает другую функцию, которую можно применить к другим аргументам позже, чтобы получить окончательный ответ. Авторы в этой книге используют каррирование в качестве понятного – и, как по мне, приятно неожиданного – способа объяснить разницу между аргументами и параметрами в машинном обучении и почему они должны быть представлены в строго определенном порядке, некоторые – сейчас, некоторые – позже. (Третья разновидность аргумента, гиперпараметры, также объясняется с помощью еще одного механизма языка программирования. Если вы знакомы с модным понятием «динамическая область видимости», то вам будет интересно; если нет, не беспокойтесь – гиперпараметры и их поведение доходчиво объясняются на примере.)

Третье назначение функций высшего порядка в этой книге состоит в том, чтобы разбить большие нейронные сети на мелкие составляющие блоки и объяснить поведение и обучение этих сетей.

Четвертое назначение функций высшего порядка – обеспечение абстракции данных. Поначалу параметры всегда представляют собой простые числа, но код их обработки настолько искусно определен с помощью всего нескольких интерфейсных функций – именно тех, которые нужны, – что код более высокого порядка не нужно менять при расширении представления параметров. (Ключевой идеей является проекция: у нас есть две функции – одна проецирует данные в альтернативное представление, на котором легче вычислять, а

[1] Преобразование функции от нескольких аргументов в набор вложенных функций, каждая из которых является функцией от одного аргумента. – *Прим. перев.*

другая извлекает вычисленный результат обратно в исходное представление. Тогда функция, которая принимает две такие функции в качестве аргументов, может использоваться в очень общем виде.) Аналогично, скаляры – это простые числа на протяжении большей части книги, но когда возникает необходимость обобщить их на двойственные числа в приложении А, функции более высокого порядка упрощают задачу.

> Поэтому если вас интересуют общие идеи языков программирования, помните об этих применениях функций высшего порядка.
>
> Возможно, вам понравится замечать их по ходу чтения.
>
> Но если вам не важны функции высшего порядка, то пожалуйста – игнорируйте все, что я только что сказал.
>
> Погрузите себя в мир машинного обучения.
>
> Вообще-то, эта книга не нуждается в представлении; она идеально подходит для своей цели.

Я прочитал всю серию *The Little Book* и с каждой новой книгой говорил себе: «Эта лучшая из всех!» Но Дэн и Анураг вновь превзошли себя – эта книга лучше прежних. На книжной полке она должна стоять особняком; вам не нужно читать предыдущие «книжки-малышки», чтобы понять эту, и вам не нужно заранее разбираться в Scheme или любом другом языке программирования. Около дюжины нужных вам идей по языку программирования объясняются по пути, каждая именно тогда, когда она вам нужна, и с использованием многочисленных примеров. Не спешите и наслаждайтесь повествованием.

Гай Стил-мл.
Лексингтон, Массачусетс,
август 2022 г.

Предисловие Питера Норвига

Привет, я Питер Норвиг, опытный исследователь и практик в области машинного обучения. Мне повезло стать одним из первых читателей этой книги, и меня попросили оставить несколько комментариев к ней. Я собираюсь сделать это в форме диалога с моим уважаемым коллегой, типичным читателем. Добро пожаловать в это предисловие, уважаемый читатель, как дела?	1	Спасибо, я счастлив быть здесь, даже если я всего лишь плод воображения. И кстати, зовите меня просто Типи.
Хорошо, Типи. Я могу сказать, что мне очень понравилась книга и то, как авторы постепенно и тщательно раскрывают ключевые понятия. А вы что думаете о книге?	2	Честно говоря, я еще не вчитывался. Пока я только привыкаю к необычному стилю. Книга выглядит увлекательно, но я еще не решил, стоит ли она моего времени и усилий.
Вы верно подметили. Эта книга более интерактивна, чем большинство других книг, и требует постоянного участия читателя, задавая ему вопросы. Я не могу сказать наперед, подходит ли вам книга, но я могу сказать, что лучший способ получить опыт в любой области – это осмысленная практика, а не просто пассивное слушание или чтение.	3	Несомненно, она требует больше усилий при чтении, чем обычная книга, потому что вам придется старательно проработать все примеры.

Работа с примерами сильно помогает закрепить материал!

Припоминаю, как однажды я обсуждал технику машинного обучения с коллегой, и он сказал, что не понимает, как работает одна особенно сложная часть. Я беззаботно ответил, что это кажется сложным, но на самом деле все просто.

Я вспомнил, что у Эндрю Ына было отличное видео, которое совершенно элегантным образом объясняло, как это работает. Я попытался повторить то, что узнал из видео, и… не смог этого сделать!

Объяснение Эндрю было настолько понятным, когда я смотрел его, что я не стал утруждать себя его изучением. Мой коллега быстро нашел это видео, и сказал: «Это, должно быть, то самое видео, давайте посмотрим его еще раз», но я отказался. На этот раз я намеревался получить ответ самостоятельно, а затем запомнить его.

И я помню его по сей день.

4 Хорошо, вы убедили меня в том, что практика приносит пользу.

Но я привык видеть инструменты машинного обучения в основном на языке Python, или, может быть, Java, или R, а Scheme встречается не очень часто.

Эта книга не о языках программирования, и речь идет не о наборе инструментов машинного обучения.

Авторы объясняют фундаментальные концепции машинного обучения и их реализации простейшим образом.

5 Вы полагаете, язык Scheme – удачный выбор для этой цели?

Я уверен в этом.

Это один из самых простых языков программирования. Вам достаточно придерживаться основных математических нотаций, которые превращаются в исполняемый код, поэтому нет никакой двусмысленности в отношении того, как все это работает.

6 Но научит ли меня эта книга делать правильные вещи?

Должны ли вы взять код Scheme из этой книги и применить его в своем следующем проекте машинного обучения? Не обязательно. Но даже если вы используете готовый инструментарий для машинного обучения, такой как Tensorflow или Pytorch, из этой книги вы вынесете понимание того, как работают основы. Это понимание позволит вам намного проще решить, что делать дальше, когда Tensorflow или Pytorch ведет себя не так, как вы задумали.	7	Полностью согласен. Но я опасаюсь, что в этой книге сплошная теория и математика... Я стремлюсь к глубокому пониманию основ, но хочу, чтобы это было полезно на практике.
Эта книга проведет вас через ключевые идеи машинного обучения от теории к практике. В ней минимальное количество математических формул и большинство тем излагается с помощью кода и разговорной речи, а не уравнений. Я из тех людей, которым всегда удобнее читать код, чем формулы, поэтому книга наполнила меня уютным чувством комфорта, а не страхом.	8	Звучит привлекательно. О каких ключевых темах я узнаю? И сколько времени уходит на то, чтобы добраться до них?
Думаю, темы представлены в правильном темпе. Книга начинается с основ языка Scheme и линейных уравнений прямой. На стр. **61** дано определение функции потерь L2, скорости обучения и градиентного спуска. Часто темы освещаются сначала упрощенно, а потом детально; например, мы возвращаемся к градиентному спуску на стр. **140** и переходим к алгоритму оптимизации Adam (современная версия градиентного спуска) на стр. **171**.	9	Вы меня убедили, что недостаточно лишь пролистать книгу, и я намерен ее хорошенько проработать. Кстати, в книге было что-то, с чем вы не согласны?

Когда мы перейдем к стр. 201, вы увидите, что определение «Искусственный нейрон – это параметризованная линейная функция, совмещенная с нелинейной функцией принятия решения» идеально объединяет знания, полученные на предыдущих страницах.

Вы узнаете о наиболее часто используемых функциях решателей, таких как *relu*, и на стр. 209 найдете простое, но исчерпывающее описание того, как любая функция может быть аппроксимирована инструментами, которые у нас уже есть.

Книга постоянно опирается на ранее заложенный фундамент, чтобы исследовать проблемы, которые возникают на практике, например при обсуждении исчезающих и взрывающихся градиентов на стр. 259.

Я всецело согласен с педагогическим стилем изложения, когда мы вместе с авторами прорабатываем проблему, достигаем осязаемого результата, а потом наслаждаемся перерывом.

Я использовал технику Помодоро, чтобы сосредоточиться на рабочих интервалах; эта книга прекрасно подходит для такого метода организации работы.

Но я боюсь, что если вы будете строго следовать советам книги по употреблению десертов в перерывах, это будет явное излишество.

Побольше функций, поменьше сладкого!

10 Как типичный читатель я потребляю 77 г сахара в день.

Но я прислушаюсь к вашему предупреждению и постараюсь приблизиться к суточному потреблению в 24 г в день.

Питер Норвиг
Пало Альто, Калифорния,
август 2022 г.

Благодарности

Гай Стил и Питер Норвиг стали идеальными авторами предисловий. Они точно поняли, что мы пытались сказать, и каждый из них прекрасно уловил дух повествования. Мы сожалеем лишь о том, что наши слова не могут в полной мере передать, насколько мы были взволнованы, когда включали в книгу их изящно сформулированные соображения.

Рисункам Цинцин Сю присущ совершенно особый стиль, вдохновленный иллюстрациями из предыдущих изданий *The Little Book*, но ее творения обладают собственной уникальной индивидуальностью. Мы благодарим ее за мастерство, с которым она создала эти рисунки.

Мы благодарны Сюзанне Мензел и Митчу Ванду, которые долгое время работали над серией *The Little Book*. Они старательно и мягко поясняли нам, что было неправильным, сбивающим с толку или что можно было бы улучшить, и благодаря им мы внесли много существенных улучшений. Мы признательны и благодарны Джули Сассман, которая вмешалась буквально в предпоследнюю минуту, чтобы избавить эту книгу от большего количества недостатков в написании, чем мы ожидали.

Наши первые читатели, Стив Бецольд, Рон Гарсиа, Ник Фаро и Ричард Оттен, вселили в нас уверенность в результатах собственного труда. Мы хотим особенно поблагодарить Джейсона Хеманна, чей отзыв было приятно читать. Спасибо также Паркеру Моресу и Заку Вилкерсону, которые поддержали нас в использовании необычного стиля нумерации страниц.

Мы хотели бы поблагодарить Ника Дрозда, Ширама Кришнамурти, Вейцзи Ма, Зака Сейлигера, Адитхая Селвапритхвирая и Майкла Ваньера за их терпеливое прочтение и отзывы. Цинъи Цзи помог нам проявить сочувствие к нашим читателям, особенно в первых главах. Джон Росси был ярым энтузиастом обучения, который показал нам, как помочь некоторым читателям, которых часто упускают из виду.

Мы особенно благодарны Даршалу Шетти и Чаникье Вманагари не только за отзывы, но и за их работу по улучшению примеров кода. Мы также благодарим Юфея Янга, чей острый глаз уловил несколько очень тонких ошибок уже на этапе завершения книги.

Мы благодарим Маттиаса Феллейзена и всю команду Racket, которые привнесли дух языка Scheme в Racket. Мы восхищаемся

этой чрезвычайно успешной, продолжающейся много лет работой. И особая благодарность Дораи Ситараму за его создание SLATEX.

Наша редакционная команда в MIT Press, состоящая из Элизабет Суэйзи, Мэтью Валадеса и Джея Марси, заслужила нашу благодарность и признательность за то, что сделала процесс подготовки издания максимально беспроблемным. Спасибо также Алекс Хупс за ее первоначальное участие в процессе.

Мы благодарим Южень Е, Линн Миколон, Бениту Браун и Чарльза Поупа, которые сделали нашу жизнь в Университете Индианы намного проще, чем кто-либо мог ожидать. Мы благодарим Кристину Лао, Рунака Виджана и Джонатана Ву, а также Спенсера Балло и Амога Батвала, которые были первыми читателями, за вдохновение, полученное в результате обсуждений и технических диалогов.

И наконец, мы благодарим наши семьи за их поддержку, терпение и стойкость на протяжении бесконечных видеозвонков между Дэном и Анурагом, а также за все выходные дни и будние ночи, поглощенные нашей писательской страстью.

Дэн благодарит свою замечательную жену Мэри; Роберту (читательницу рукописи) и Шеннона и их детей, Саманту и Чейза; Рэйчел и Джозефа и их дочь Уиллоу; Сару и Трэвиса и их детей, Бруклин, Арью и Вьятт.

Анураг благодарит свою замечательную жену Аруну и двух дочерей, Ришму и Арью Нину (также одну из первых читательниц).

Введение

> Нельзя пропускать главы, в жизни так не бывает. Надо читать каждую строчку, знакомиться с каждым персонажем. Никто не обещал вам сплошное удовольствие. Некоторые главы повергнут вас в ад и заставят рыдать над ними целыми неделями. Вам придется читать вещи, которые мучительно неприятны; иногда вы, напротив, будете мечтать, чтобы страницы не заканчивались. Но надо идти вперед. Истории движут мир. Живите полной жизнью, не пропускайте главы.
>
> – Кортни Пеппернелл, цитата из книги *Pillow Thoughts II*, издание Andrews McMeel Publishing, © 2018

Глубокое обучение не только стало новой областью искусственного интеллекта, оно произвело революцию в способах решения задач, будь то победа в го, распознавание кошек на картинках или просьба к умной колонке заказать пиццу. Самое прекрасное в глубоком обучении – это то, как изящно простые части объединяются для решения больших и сложных задач. Как лучше всего понять, что заставляет эти инструменты глубокого обучения работать? Наш подход заключается в том, чтобы создавать базовые функции – составные части глубокого обучения – и наблюдать за их работой сначала по отдельности, а потом вместе.

Способность справляться с зашумленными данными – вот что делает глубокое обучение, являющееся разновидностью машинного обучения, таким заманчивым. Теперь стало ясно, что получать на 100 % однозначные ответы на зашумленных данных не только невозможно, но и не нужно. Наше ощущение обязательной точности решений, характерное для многих задач, для которых мы пишем программы, постепенно исчезает. Тем не менее мы можем и должны поддерживать чувство уверенности в функциях, которые мы определяем, чтобы не противоречить собственной интуиции, чтобы мы могли быть уверены, что эти функции соответствуют нашим ожиданиям.

Для освоения инструментов глубокого обучения вам потребуются только базовые знания математики старшей средней школы, а также некоторый опыт программирования. Функции, которые мы определяем, предназначены для запуска и экспериментов с кодом. Конечно, эту книгу можно читать, не запуская их, но каждое определение необходимо хорошо понимать.

Как читать эту книгу

Если тема этой книги для вас абсолютно нова или вам кажется непривычным способ изложения, вы должны читать каждую главу, пока полностью не поймете ее. Возможно, некоторые главы придется прочитать несколько раз. Не продвигайтесь вперед, пока у вас не появится чувство полного понимания. Обменивайтесь знаниями с другими и обсуждайте с ними непонятное, пока каждая мелочь не встанет на свои места.

В этой книге бесполезно забегать вперед, потому что изложение строго упорядочено в виде последовательности пронумерованных фреймов, разделенных горизонтальными линиями. Каждая глава довольно короткая и содержит много пробелов, поэтому вам не составит труда внимательно перечитать всю главу, чтобы поймать потерянную нить.

В конце книги есть два приложения – тоже строго упорядоченных. Они объясняют, как создавать базовые инструменты, помогающие в глубоком обучении. Эти приложения также предполагают наличие лишь минимальных базовых знаний, но предъявляют к читателю чуть более высокие требования, чем остальная часть книги.

Как написаны наши программы

Мы оформляем свои идеи в виде небольших программ Scheme, которые позволяют выражать мысли наиболее ясно и прямо. Это очень лаконичный и интуитивно понятный язык, так что код говорит сам за себя.

Мы используем очень небольшое подмножество Scheme: `define` (или `let`) позволяет присвоить значению глобальное (или локальное) имя, `lambda` создает функцию как значение, а `cond` выполняет диспетчеризацию по последовательности пар (тестовых значений). Существуют также примитивные функции, такие как +, –, и *, которые работают с числами. О них будет рассказано в главе 0, которая служит своего рода развернутым введением.

Мы используем маленькие рамки для выделения кода в тексте и объясняем, как работают программы в этих рамках. Если вы не пропускаете главы, понять объяснения не составит труда. Эта книга представляет собой набор понятий, почти все из которых являются функциями. Если мы просим дать определение функции (в левой части фрейма), потратьте некоторое время и создайте правдоподобное определение, прежде чем смотреть на наш ответ (в правой части фрейма). Вот простой пример фрейма с определениями функций в рамках:

Это рамка с кодом

```
(define функция
  ... которая хорошо сюда помещается
...)
```

[1] Это красная рамка.
Она встречается не очень часто.
Впервые мы используем ее во фрейме 26:25,
и
эта запись означает фрейм 25 на стр. 26

```
(define другая_функция
  ... запомните эту функцию! ...)
```

Мы также выделяем ключевые свойства как правила или законы. Правила касаются структуры сущностей, например размеров. Законы описывают поведение сущностей, такое как их равенства и инварианты.

Как запустить код

Мы собрали функции и синтаксические расширения, необходимые для кода из этой книги, в пакет Machine Learning Toolkit под названием Malt. Malt – это пакет в Racket, который является надмножеством нашего небольшого подмножества Scheme. Пакет включает наш код и примеры, а также инструменты, необходимые для экспериментов с ними. Рекомендации по его использованию доступны на сайте www.thelittlelearner.com.

Как есть десерты

Те, кто знаком с предыдущими «книжками-малышками», уже знают, что мы обожаем игру слов и стараемся добавить немного юмора при каждом удобном случае. Очень трудно найти юмор в цифрах, поэтому мы ищем его в других местах. Мы составили список десертов для употребления в ближайшем кафе. Не откажите себе в удовольствии! Но и не слишком налегайте на сладкое и не забывайте сначала съесть отварные овощи.

Мы надеемся, что этот небольшой экскурс в глубокое обучение доставит вам удовольствие и что читать его так же интересно, как и писать.

Приятного аппетита!

Дэниел Фридман, Блумингтон, Индиана
Анураг Мендхекар Лос-Альтос, Калифорния

Перевод сокращенной записи в язык Scheme

Некоторые из наших функций мы пишем с помощью более компактной нотации, чтобы они были легче для чтения и плотно помещались в «маленьких коробочках». Перед запуском программы обязательно переведите нашу запись в код Scheme. В таблице ниже показано, как это сделать.

Первый столбец в таблице ниже указывает на место самого первого применения в книге обозначений, показанных во втором столбце. Третий столбец показывает, как переводить запись программ (например, [5 (+ 10 2) 28] переводится в `(tensor 5 (+ 10 2) 28)`).

Страница:фрейм	Обозначение	Расшифровка
23:17	$[t\ ts\ \ldots]$	`(tensor t ts . . .)`
25:24	l_i	`(ref l i)`
26:27	(list m ...)	`(list m . . .)`
32:17	$\natural t \natural$	`(tlen t)`
34:24	$t\|_i$	`(tref t i)`
39:42	$\|l\|$	`(len l)`
48:22	$(\langle op \rangle^{\langle rank \rangle} t)$	$(\langle op \rangle - \langle rank \rangle\ t)$
74:15	$(\nabla\ f\ \theta)$	`(gradient-of f θ)`
102:25	$(\bullet\ t\ u)$	`(dot-product t u)`
103:26	$(\langle op \rangle^{\langle rank_1 \rangle, \langle rank_2 \rangle} t)$	$(\langle op \rangle - \langle rank_1 \rangle - \langle rank_2 \rangle\ t)$
121:27	$t\|\|_b$	`(trefs t b)`
226:49	$l_{i\downarrow}$	`(refr l i)`

Например, на стр. 48, фрейм 22, мы вводим дефис между *sum* и 1 для перевода записи sum^1 в `sum - 1`, а на стр. 103, фрейм 26, вводим второй дефис для перевода записи $\bullet^{1,1}$ в `dot-product-1-1`.

Греческие буквы и письменные варианты имен переменной, такие как $\hat{a}$, α, $\hat{\alpha}$, *an-*α, β, $\hat{c}$, ϵ, θ, Θ, λ, μ и π, в коде программы пишут-

ся соответственно как `a-hat`, `alpha`, `alpha-hat`, `an-alpha`, `beta`, `c-hat`, `epsilon`, `theta`, `big-theta`, `lambda`, `mu` и `pi`. В именах формальных аргументов, функций и в ключевых словах можно использовать символы Unicode.

Для успешного выполнения кода требуется установка пакета Malt для Racket v8.0 или более поздней версии. Подробную информацию о загрузке и запуске кода можно найти на www.thelittlelearner.com.