Александр Чичулин

НЕЙРОСЕТИ

Раскройте всю мощь нейронных сетей: полное руководство по пониманию, внедрению ИИ

Александр Чичулин

Нейросети. Раскройте всю мощь нейронных сетей: полное руководство по пониманию, внедрению ИИ

http://www.litres.ru/pages/biblio_book/?art=69288727
ISBN 9785006012592

Аннотация

Исследуйте увлекательный мир нейронных сетей в этом всеобъемлющем и удобном для начинающих руководстве. Узнайте, как эти мощные алгоритмы имитируют человеческий мозг, поймите их компоненты и реализуйте их с помощью Python. Откройте для себя приложения, этические соображения и будьте в курсе последних тенденций. Раскройте потенциал нейронных сетей и окажите положительное влияние в различных областях.

Содержание

Нейросети
Раскройте всю мощь нейронных сетей: полное руководство по пониманию, внедрению ИИ

Александр Чичулин

ISBN 978-5-0060-1259-2
Создано в интеллектуальной издательской системе Ridero

Общие сведения о нейронных сетях: обзор

Что такое нейронные сети?

Нейронные сети – это вычислительные модели, вдохновленные структурой и функционированием человеческого мозга. Они предназначены для обработки и анализа сложных закономерностей в данных, обучения на примерах и составления прогнозов или решений.

Проще говоря, нейронные сети состоят из взаимосвязанных узлов, называемых нейронами, которые организованы в слои. Каждый нейрон получает входные данные, выполняет вычисления и производит выход, который передается другим нейронам. Эта взаимосвязанность позволяет нейронным сетям захватывать и представлять сложные отношения в данных.

Нейронные сети широко используются в различных областях, таких как распознавание изображений и речи, обработка естественного языка, финансовые прогнозы и многое другое. Они преуспевают в задачах, связанных с распознаванием образов, классификацией, регрессией и обработкой последовательностей.

Используя свою способность учиться на данных, нейронные сети могут автономно улучшать свою производительность с течением времени с помощью процесса, называемого обучением. Во время обучения сеть настраивает свои внутренние параметры, чтобы свести к минимуму разницу между прогнозируемыми выходами и желаемыми выходами. Этот итеративный процесс обучения позволяет нейронным сетям становиться все более точными и эффективными при решении сложных задач.

Нейронные сети привлекли значительное внимание и популярность благодаря своим замечательным возможностям и применимости в широком спектре областей. Понимание основ нейронных сетей открывает захватывающие возможности для использования их возможностей в решении реальных проблем.

Почему нейронные сети важны?

Нейронные сети стали важнейшим инструментом в области искусственного интеллекта и произвели революцию в различных отраслях. Вот несколько причин, по которым нейронные сети важны:

1. Мощное распознавание образов: нейронные сети превосходно распознают закономерности и извлекают ценную информацию из сложных данных. Они могут идентифицировать сложные отношения, особенности и структуры, которые могут быть нелегко различимы традиционными алгоритмами или человеческим анализом. Эта способность делает нейронные сети ценными в таких задачах, как распознавание изображений, понимание речи, обработка естественного языка и анализ данных.

2. Универсальность и адаптивность: нейронные сети очень гибки и могут применяться к широкому кругу проблем в различных областях. Они могут учиться на примерах и адаптировать свои внутренние параметры для выполнения конкретных задач. Эта универсальность позволяет нейронным сетям решать различные задачи, включая классификацию изображений, языковой перевод, обнаружение мошенничества, медицинскую диагностику и многое другое.

3. Принятие решений и прогностические способности: Нейронные сети могут принимать обоснованные решения

и прогнозы на основе изученных закономерностей из исторических данных. Они могут обрабатывать огромные объемы информации, выявлять соответствующие функции и предоставлять аналитические сведения для поддержки процессов принятия решений. Нейронные сети ценны для таких задач, как финансовое прогнозирование, анализ поведения клиентов, оценка рисков и персонализированные рекомендации.

4. Автоматизация и эффективность: Нейронные сети могут автоматизировать сложные задачи и сократить человеческие усилия. После обучения они могут анализировать и обрабатывать данные на высоких скоростях, что делает их идеальными для работы с большими наборами данных и приложениями реального времени. Автоматизируя повторяющиеся и трудоемкие процессы, нейронные сети высвобождают человеческие ресурсы для более стратегических и творческих начинаний.

5. Непрерывное совершенствование: нейронные сети обладают способностью учиться и совершенствоваться с течением времени. С помощью итеративного обучения они уточняют свои внутренние представления и корректируют параметры для оптимизации производительности. Эта адаптивность позволяет нейронным сетям адаптироваться к изменяющимся условиям, обнаруживать возникающие закономерности и постоянно повышать их точность и эффективность.

6. Передовые технологические достижения: Нейронные

сети находятся на переднем крае технологических достижений. Исследователи и практики постоянно изучают новые архитектуры, алгоритмы и методы для повышения производительности нейронных сетей и решения сложных задач. Информирование о разработках нейронных сетей позволяет отдельным лицам и организациям использовать последние достижения в области искусственного интеллекта и сохранять конкурентное преимущество.

Таким образом, нейронные сети играют жизненно важную роль в решении сложных проблем, извлечении информации из данных и продвижении достижений в различных отраслях. Их способность распознавать закономерности, делать прогнозы и адаптироваться к новой информации делает их незаменимыми инструментами в эпоху искусственного интеллекта.

Как нейронные сети имитируют человеческий мозг

Нейронные сети вдохновлены структурой и функционированием человеческого мозга. Хотя они не являются точными копиями мозга, нейронные сети пытаются имитировать определенные аспекты его архитектуры и вычислительных процессов. Вот как нейронные сети эмулируют работу человеческого мозга:

1. Нейроны: Нейронные сети состоят из взаимосвязанных узлов, называемых нейронами, которые аналогичны нейронам в человеческом мозге. Эти искусственные нейроны получают входные данные от других нейронов, выполняют вычисления и генерируют выходные сигналы.

2. Слои: Нейронные сети организованы в слои нейронов, похожие на слои нейронов, обнаруженные в мозге. Как правило, существуют входные, скрытые и выходные слои. Информация передается от входного слоя через скрытые слои к выходному, при этом каждый слой обрабатывает и преобразует данные.

3. Активация: Искусственные нейроны в нейронных сетях используют функции активации, которые определяют их выход на основе полученного входа. Активационные функции имитируют возбуждение или торможение нейронов в мозге, переводя входные сигналы в значимые выходы.

4. Обучение на основе данных: Нейронные сети учатся на примерах с помощью процесса, называемого обучением. Во время обучения сеть корректирует свои внутренние параметры, известные как веса и смещения, чтобы свести к минимуму разницу между прогнозируемыми выходами и желаемыми выходами. Это имитирует способность мозга учиться на опыте и адаптировать свои синаптические связи.

5. Параллельная обработка: Нейронные сети способны к параллельной обработке, что означает, что несколько нейронов могут выполнять вычисления одновременно. Этот параллелизм позволяет нейронным сетям обрабатывать большие объемы данных и эффективно выполнять сложные вычисления, напоминающие возможности распределенной обработки мозга.

6. Извлечение признаков: Нейронные сети умеют автоматически извлекать соответствующие функции из входных данных. Благодаря иерархической организации слоев нейронные сети могут научиться распознавать и представлять сложные паттерны и особенности, подобно тому, как мозг обрабатывает сенсорную информацию и извлекает значимые представления.

7. Обобщение: Нейронные сети стремятся обобщать примеры, на которых они обучаются, что позволяет им делать точные прогнозы или принимать решения на основе невидимых данных. Это отражает способность мозга выводить и распознавать закономерности, выходящие за рамки кон-

кретных случаев, с которыми он столкнулся.

В то время как нейронные сети стремятся имитировать определенные аспекты структуры и функционирования мозга, важно отметить, что они являются упрощенными представлениями и не охватывают всю сложность человеческого мозга. Тем не менее, черпая вдохновение из мозга, нейронные сети обеспечивают мощную вычислительную основу для решения широкого круга проблем и развития области искусственного интеллекта.

Часть I: Начало работы с нейронными сетями

Основы искусственных нейронных сетей

Компоненты нейронной сети

Нейронные сети состоят из нескольких компонентов, которые работают вместе для обработки данных и составления прогнозов. Давайте рассмотрим ключевые компоненты нейронной сети:

1. Нейроны: Нейроны являются фундаментальными единицами нейронной сети. Они принимают входные сигналы, выполняют вычисления и производят выходные сигналы. Каждый нейрон связан с другими нейронами через взвешенные связи.

2. Веса и смещения: Связи между нейронами в нейронной сети связаны с весами. Эти веса представляют силу или важность соединения. Во время обучения сеть корректирует эти веса, чтобы учиться на данных. Смещения – это дополнительные параметры, которые помогают регулировать вы-

ход нейронов, обеспечивая гибкость сети.

3. Функции активации: Функции активации привносят нелинейность в нейронную сеть. Они преобразуют взвешенную сумму входов в нейроне в выходной сигнал. Общие функции активации включают сигмоидную функцию, которая отображает входы в диапазоне от 0 до 1, и выпрямленную линейную единицу (ReLU), которая выводит вход, если он положительный, и 0 в противном случае.

4. Слои: Нейронные сети организованы в слои, которые представляют собой группы нейронов. Три основных типа слоев:

– Входной слой: входной слой получает исходные данные и передает их следующему слою.

– Скрытые слои: Скрытые слои обрабатывают промежуточные представления данных. Они извлекают особенности и изучают сложные шаблоны.

– Выходной слой: Выходной слой производит окончательный вывод или прогноз нейронной сети. Количество нейронов в этом слое зависит от конкретной проблемы, для решения которой предназначена сеть.

Организация слоев и связи между нейронами позволяют информации проходить через сеть, при этом каждый слой вносит свой вклад в общие вычисления и преобразование данных.

Понимание компонентов нейронной сети имеет важное значение для настройки сетевой архитектуры, установки на-

чальных весов и смещений, а также реализации соответствующих функций активации. Эти компоненты в совокупности позволяют сети учиться на данных, делать прогнозы и решать сложные проблемы.

Функции активации

Активационные функции играют решающую роль в нейронных сетях, привнося нелинейность в вычисления, выполняемые нейронами. Они преобразуют взвешенную сумму входов в выходной сигнал, позволяя нейронным сетям моделировать сложные отношения и делать точные прогнозы. Давайте рассмотрим некоторые распространенные функции активации, используемые в нейронных сетях:

1. Сигмовидная функция: сигмовидная функция отображает входные данные в диапазоне от 0 до 1. Он имеет S-образную кривую и часто используется в задачах бинарной классификации. Сигмовидная функция определяется как:

f (x) = 1 / (1 + e^ (-x))

Выходные данные сигмовидной функции представляют собой уровень вероятности или достоверности, связанный с конкретным классом или событием.

2. Выпрямленная линейная единица (ReLU): Функция ReLU – популярная функция активации, используемая в скрытых слоях нейронных сетей. Он выводит входное значение, если оно положительное, и 0 в противном случае. Ма-

тематически функция ReLU определяется как:

f (x) = max (0, x)

ReLU привносит в сеть разреженность и нелинейность, помогая ей изучать и представлять сложные функции в данных.

3. Функция Softmax: Функция softmax обычно используется в задачах классификации нескольких классов. Он принимает набор входных данных и преобразует их в вероятности, гарантируя, что вероятности в сумме равны 1. Функция softmax определяется как:

f (x_i) = e^ (x_i) / sum (e^ (x_j)), для каждой x_i в наборе входных данных

Выходные данные функции softmax представляют собой распределение вероятностей по нескольким классам, что позволяет сети делать прогнозы для каждого класса.

Это всего лишь несколько примеров функций активации, используемых в нейронных сетях. Другие функции активации, такие как tanh (гиперболический тангенс), Leaky ReLU и экспоненциальная линейная единица (ELU), также существуют и используются в зависимости от характера проблемы и архитектуры сети.

Выбор подходящей функции активации имеет решающее значение, поскольку она влияет на динамику обучения, конвергенцию и общую производительность сети. Часто требуется экспериментирование и знание предметной области, чтобы определить наиболее подходящую функцию актива-

ции для данной задачи.

Архитектуры нейронных сетей

Архитектуры нейронных сетей относятся к конкретным расположениям и конфигурациям нейронов и слоев внутри нейронной сети. Различные архитектуры предназначены для обработки различных типов данных и решения конкретных задач. Давайте рассмотрим некоторые распространенные архитектуры нейронных сетей:

1. Нейронные сети с прямой связью (FNN):

– Нейронные сети с прямой связью – самый простой и распространенный тип нейронных сетей.

– Информация течет в одном направлении, от входного слоя через скрытые слои к выходному слою, без циклов и циклов.

– FNN широко используются для таких задач, как классификация, регрессия и распознавание образов.

– Они могут иметь различное количество скрытых слоев и нейронов внутри каждого слоя.

2. Сверточные нейронные сети (CNN):

– Сверточные нейронные сети в основном используются для обработки сетчатых данных, таких как изображения, видеокадры или данные временных рядов.

– Они используют специализированные слои, такие как сверточные и объединяющие слои, для извлечения про-

странственных или временных объектов из данных.

– CNN отлично справляются с такими задачами, как классификация изображений, обнаружение объектов и сегментация изображений.

– Они предназначены для захвата локальных шаблонов и иерархий в данных.

3. Рекуррентные нейронные сети (RNN):

– Рекуррентные нейронные сети предназначены для последовательной обработки данных, где выход зависит не только от текущего входа, но и от прошлых входов.

– Они имеют повторяющиеся соединения внутри сети, что позволяет хранить информацию и передавать ее между временными шагами.

– RNN используются в таких задачах, как обработка естественного языка, распознавание речи и прогнозирование временных рядов.

– Long Short-Term Memory (LSTM) и Gated Recurrent Unit (GRU) – популярные варианты RNN, которые помогают решить проблему исчезающего градиента и зафиксировать долгосрочные зависимости.

4. Генеративно-состязательные сети (GAN):

– Генеративно-состязательные сети состоят из двух сетей: генератора и дискриминатора.

– Сеть-генератор учится генерировать синтетические данные, напоминающие реальные данные, в то время как сеть дискриминаторов учится различать реальные и поддельные

данные.

– GAN используются для таких задач, как генерация изображений, генерация текста и синтез данных.

– Они показали замечательный успех в создании реалистичных и высококачественных образцов.

5. Сети обучения с подкреплением (RLN):

– Сети обучения с подкреплением объединяют нейронные сети с алгоритмами обучения с подкреплением.

– Они учатся принимать оптимальные решения в окружающей среде, взаимодействуя с ней и получая награды или штрафы.

– RLN используются в автономной робототехнике, играх и задачах последовательного принятия решений.

– Deep Q-Networks (DQN) и Proximal Policy Optimization (PPO) – популярные алгоритмы RLN.

Это всего лишь несколько примеров архитектур нейронных сетей, и существует множество вариаций и комбинаций, основанных на конкретных потребностях и достижениях исследований. Понимание характеристик и приложений различных архитектур позволяет практикам выбрать наиболее подходящий дизайн для своей конкретной проблемной области.

Обучение нейронных сетей

Обучение нейронных сетей включает в себя процесс оп-

тимизации параметров сети, чтобы учиться на данных и делать точные прогнозы. Обучение позволяет сети корректировать свои веса и предубеждения на основе предоставленных примеров. Давайте углубимся в ключевые аспекты обучения нейронных сетей:

1. Функции потерь:

– Функции потерь измеряют разницу между прогнозируемыми выходами сети и желаемыми выходами.

– Общие функции потерь включают среднюю квадратичную ошибку (MSE) для задач регрессии и категориальную перекрестную энтропию для задач классификации.

– Выбор функции потерь зависит от характера проблемы и желаемой цели оптимизации.

2. Обратное распространение:

– Обратное распространение – фундаментальный алгоритм обучения нейронных сетей.

– Вычисляет градиенты функции потерь по отношению к параметрам сети (весам и смещениям).

– Градиенты представляют собой направление и величину самого крутого спуска, указывая, как должны быть обновлены параметры, чтобы минимизировать потери.

– Обратное распространение распространяет градиенты назад по сети, слой за слоем, используя правило цепи исчисления.

3. Градиентный спуск:

– Градиентный спуск – алгоритм оптимизации, использу-

емый для обновления параметров сети на основе рассчитанных градиентов.

– Он итеративно регулирует веса и смещения в направлении, противоположном градиентам, постепенно минимизируя потери.

– Скорость обучения определяет размер шага, выполняемого в каждой итерации. Он уравновешивает компромисс между скоростью конвергенции и превышением.

– Популярные варианты градиентного спуска включают стохастический градиентный спуск (SGD), мини-пакетный градиентный спуск и оптимизацию Адама.

4. Обучающие данные и пакеты:

– Нейронные сети обучаются с использованием большого набора данных, который содержит входные примеры и соответствующие им желаемые выходы.

– Обучающие данные разделены на пакеты, которые являются меньшими подмножествами всего набора данных.

– Пакеты используются для итеративного обновления параметров сети, что снижает вычислительные требования и позволяет лучше обобщать.

5. Переобучение и регуляризация:

– Переобучение происходит, когда нейронная сеть учится хорошо работать на обучающих данных, но не может обобщить невидимые данные.

– Методы регуляризации, такие как регуляризация L1 или L2, отсев или досрочное прекращение, помогают предотвра-

тить переобучение.

– Регуляризация накладывает ограничения на параметры сети, способствуя простоте и снижению чрезмерной сложности.

6. Настройка гиперпараметров:

– Гиперпараметры – настройки, которые управляют поведением и производительностью нейронной сети во время обучения.

– Примеры гиперпараметров включают скорость обучения, количество скрытых слоев, количество нейронов в слое, функции активации и силу регуляризации.

– Настройка гиперпараметров включает в себя выбор оптимальной комбинации гиперпараметров с помощью экспериментов или автоматизированных методов, таких как поиск по сетке или случайный поиск.

Обучение нейронных сетей требует тщательного учета различных факторов, включая выбор функции потерь, правильную реализацию обратного распространения, оптимизацию с помощью градиентного спуска и обработку переобучения. Эксперименты и тонкая настройка гиперпараметров играют решающую роль в достижении наилучшей производительности и обеспечении того, чтобы сеть хорошо обобщала невидимые данные.

Подготовка данных для нейронных сетей

Представление данных и масштабирование объектов

В этой главе мы рассмотрим важность представления данных и масштабирования признаков в нейронных сетях. То, как данные представляются и масштабируются, может существенно повлиять на производительность и эффективность сети. Давайте углубимся в эти ключевые понятия:

1. Представление данных:

– Способ представления и кодирования данных влияет на то, насколько хорошо нейронная сеть может извлекать значимые закономерности и делать точные прогнозы.

– Категориальные данные, такие как текст или номинальные переменные, часто необходимо преобразовать в числовые представления. Этот процесс называется одногорячим кодированием, где каждая категория представлена в виде двоичного вектора.

– Числовые данные должны быть масштабированы до аналогичного диапазона, чтобы одни функции не доминировали над другими. Масштабирование гарантирует, что каждая

функция вносит пропорциональный вклад в общий прогноз.

2. Масштабирование функций:

– Масштабирование объектов – это процесс нормализации или стандартизации числовых признаков в наборе данных.

– Нормализация масштабирует данные до диапазона от 0 до 1 путем вычитания минимального значения и деления на диапазон (максимум минус минимум).

– Стандартизация преобразует данные в среднее значение 0 и стандартное отклонение 1 путем вычитания среднего значения и деления на стандартное отклонение.

– Масштабирование функций помогает предотвратить доминирование одних объектов над другими из-за различий в их величинах, обеспечивая справедливое и сбалансированное обучение.

3. Обработка недостающих данных:

– Отсутствующие данные могут создавать проблемы при обучении нейронных сетей.

– Для обработки отсутствующих данных можно использовать различные подходы, такие как методы условного исчисления, которые заполняют недостающие значения на основе статистических показателей, или использование выделенных архитектур нейронных сетей, которые могут обрабатывать отсутствующие значения напрямую.

– Выбор способа обработки отсутствующих данных зависит от характера и количества отсутствующих значений в на-

боре данных.

4. Работа с несбалансированными данными:

– Несбалансированность данных возникает, когда один класс или категория значительно более распространены, чем другие в наборе данных.

– Несбалансированные данные могут привести к предвзятым прогнозам, когда сеть склоняется в пользу класса большинства.

– Методы устранения несбалансированных данных включают передискретизацию класса меньшинства, недовыборку класса большинства или использование алгоритмов, специально разработанных для несбалансированных данных, таких как SMOTE (метод синтетической избыточной выборки меньшинств).

5. Инженерия функций:

– Проектирование признаков включает в себя преобразование или создание новых объектов из существующего набора данных для повышения предсказательной силы сети.

– Такие методы, как полиномиальные признаки, термины взаимодействия или преобразования, специфичные для предметной области, могут применяться для получения более информативных признаков.

– Проектирование функций требует знания предметной области и понимания проблемы.

Правильное представление данных, масштабирование признаков, обработка отсутствующих данных, работа

с несбалансированными данными и продуманное проектирование признаков являются важными шагами в подготовке данных для обучения нейронной сети. Эти процессы гарантируют, что данные находятся в подходящей форме, чтобы сеть могла эффективно учиться и делать точные прогнозы.

Методы предварительной обработки данных

Предварительная обработка данных играет жизненно важную роль в подготовке данных к обучению нейронной сети. Он включает в себя ряд методов и шагов по очистке, преобразованию и нормализации данных. В этой главе мы рассмотрим некоторые распространенные методы предварительной обработки данных, используемые в нейронных сетях:

1. Очистка данных:

– Очистка данных включает в себя обработку отсутствующих значений, выбросов и несоответствий в наборе данных.

– Отсутствующие значения могут быть вменены с использованием таких методов, как среднее условное исчисление, медианное условное исчисление или условное исчисление на основе статистических моделей.

– Выбросы, которые представляют собой экстремальные значения, отклоняющиеся от большинства данных, могут быть обнаружены и либо удалены, либо обработаны с помо-

щью таких методов, как Winsorization или замена статистически правдоподобными значениями.

– Несогласованные данные, такие как конфликтующие записи или проблемы с форматированием, могут быть устранены путем проверки и стандартизации данных.

2. Нормализация и стандартизация данных:

– Нормализация и стандартизация данных – это методы, используемые для масштабирования числовых признаков до аналогичного диапазона.

– Нормализация масштабирует данные до диапазона от 0 до 1, в то время как стандартизация преобразует данные в среднее значение 0 и стандартное отклонение 1.

– Нормализация часто подходит для алгоритмов, которые предполагают ограниченный входной диапазон, в то время как стандартизация полезна, когда объекты имеют различные масштабы и распределения.

3. Одноразовое горячее кодирование:

– Одноразовое кодирование используется для представления категориальных переменных в виде двоичных векторов.

– Каждая категория преобразуется в двоичный вектор, где только один элемент равен 1 (что указывает на наличие этой категории), а остальные равны 0.

– Одноразовое кодирование позволяет использовать категориальные данные в качестве входных данных в нейронных сетях, позволяя им обрабатывать нечисловую информацию.

4. Масштабирование функций:

– Масштабирование признаков гарантирует, что числовые объекты находятся в аналогичном масштабе, не позволяя одним объектам доминировать над другими из-за различий в величинах.

– Общие методы включают минимальное и максимальное масштабирование, когда функции масштабируются до определенного диапазона, и стандартизацию, как упоминалось ранее.

5. Уменьшение размерности:

– Методы уменьшения размерности уменьшают количество входных элементов, сохраняя при этом важную информацию.

– Анализ главных компонент (PCA) и t-SNE (t-распределенное стохастическое встраивание соседей) являются популярными методами уменьшения размерности.

– Уменьшение размерности может помочь смягчить проклятие размерности и повысить эффективность обучения.

6. Сплит и перекрестная проверка обучения-тестирования:

– Чтобы оценить производительность нейронной сети, важно разделить данные на обучающий и тестовый наборы.

– Обучающий набор используется для обучения сети, а тестовый – для оценки ее производительности на невидимых данных.

– Перекрестная проверка – это еще один метод, при котором набор данных разделяется на несколько подмножеств

(складок) для итеративного обучения и тестирования сети, получения более надежной оценки ее производительности.

Эти методы предварительной обработки данных применяются для обеспечения того, чтобы данные находились в подходящей форме для обучения нейронных сетей. Очищая данные, обрабатывая отсутствующие значения, масштабируя функции и уменьшая размерность, мы можем улучшить производительность сети, повысить ее эффективность и добиться лучшего обобщения невидимых данных.

Обработка отсутствующих данных

Отсутствующие данные являются распространенной проблемой в наборах данных и могут существенно повлиять на производительность и надежность нейронных сетей. В этой главе мы рассмотрим различные методы эффективной обработки отсутствующих данных:

1. Удаление отсутствующих данных:

– Одним из простых подходов является удаление экземпляров или объектов, содержащих отсутствующие значения.

– Если только небольшая часть данных имеет отсутствующие значения, удаление этих экземпляров или функций может не оказать существенного влияния на общий набор данных.

– Однако этот подход следует использовать с осторожностью, так как он может привести к потере ценной информа-

ции, особенно если отсутствующие данные не являются случайными.

2. Среднее/медианное условное исчисление:

– Среднее или медианное условное исчисление предполагает замену отсутствующих значений средним или медианным значением соответствующего признака.

– Этот метод предполагает, что отсутствующие значения отсутствуют случайным образом (MAR), а непропущенные значения обладают теми же статистическими свойствами.

– Условное исчисление помогает сохранить размер выборки и поддерживать распределение признака, но может привести к смещению, если пропуск не является случайным.

3. Регрессионное вменение:

– Регрессионное условное исчисление предполагает прогнозирование пропущенных значений с использованием регрессионных моделей.

– Регрессионная модель обучается на непропущенных значениях, а затем модель используется для прогнозирования отсутствующих значений.

– Этот метод фиксирует взаимосвязи между отсутствующим признаком и другими признаками, что позволяет более точно вменить.

– Тем не менее, он предполагает, что отсутствие функции может быть разумно предсказано другими переменными.

4. Множественное вменение:

– Множественное условное исчисление – это метод, при

котором отсутствующие значения вменяются несколько раз для создания нескольких полных наборов данных.

– Каждому набору данных присваиваются различные правдоподобные значения, основанные на наблюдаемых данных и их неопределенности.

– Затем нейронная сеть обучается на каждом вмененном наборе данных, и результаты объединяются для получения более надежных прогнозов.

– Множественное условное исчисление объясняет неопределенность в условном исчислении недостающих значений и может привести к более надежным результатам.

5. Выделенные архитектуры нейронных сетей:

– Существуют специальные архитектуры нейронных сетей, предназначенные для непосредственной обработки отсутствующих данных.

– Например, замаскированный автоэнкодер для оценки распределения (MADE) и автоэнкодер шумоподавления (DAE) могут обрабатывать пропущенные значения во время обучения и вывода.

– Эти архитектуры учатся восстанавливать отсутствующие значения на основе имеющейся информации и могут обеспечить повышенную производительность наборов данных с отсутствующими данными.

Выбор метода обработки отсутствующих данных зависит от характера и степени отсутствия, предположений о механизме отсутствующих данных и характеристик набора дан-

ных. Важно тщательно рассмотреть последствия каждого метода и выбрать тот, который наилучшим образом соответствует конкретным требованиям и ограничениям имеющегося набора данных.

Работа с категориальными переменными

Категориальные переменные создают уникальные проблемы в нейронных сетях, поскольку для их эффективного использования требуется соответствующее представление и кодирование. В этой главе мы рассмотрим методы работы с категориальными переменными в нейронных сетях:

1. Кодирование этикетки:

– Кодировка меток присваивает уникальную числовую метку каждой категории в категориальной переменной.

– Каждая категория сопоставляется с целочисленным значением, что позволяет нейронным сетям обрабатывать данные.

– Однако кодирование меток может привести к появлению порядковых отношений между категориями, которых не существует, что может привести к неправильным интерпретациям.

2. Одногорячее кодирование:

– Одноразовое кодирование – популярный метод представления категориальных переменных в нейронной сети.

– Каждая категория преобразуется в двоичный вектор, где каждый элемент представляет наличие или отсутствие определенной категории.

– Однотонная кодировка гарантирует, что каждая категория представлена одинаково, и удаляет любые подразумеваемые порядковые отношения.

– Это позволяет нейронной сети рассматривать каждую категорию как отдельную функцию.

3. Встраивание:

– Встраивание – это метод, который изучает низкоразмерное представление категориальных переменных в нейронной сети.

– Он сопоставляет каждую категорию с плотным вектором непрерывных значений, при этом аналогичные категории имеют векторы, расположенные ближе в пространстве внедрения.

– Встраивание особенно полезно при работе с многомерными категориальными переменными или когда отношения между категориями важны для задачи.

– Нейронные сети могут изучать вложения в процессе обучения, фиксируя значимые представления категориальных данных.

4. Встраивание сущностей:

– Встраивание сущностей – это специализированная форма внедрения, использующая преимущества связей между категориями.

– Например, в рекомендательных системах встраивание сущностей может представлять категории пользователей и элементов в совместном пространстве внедрения.

– Встраивание сущностей позволяет нейронной сети изучать отношения и взаимодействия между различными категориями, повышая ее предсказательную силу.

5. Хеширование функций:

– Хеширование признаков, или трюк с хешированием, – это метод, который преобразует категориальные переменные в векторное представление фиксированной длины.

– Он применяет хеш-функцию к категориям, сопоставляя их с предопределенным количеством измерений.

– Хеширование функций может быть полезно, когда количество категорий велико и их кодирование по отдельности становится непрактичным.

Выбор метода работы с категориальными переменными зависит от характера данных, количества категорий и отношений между категориями. Обычно используются одноразовое кодирование и внедрение, причем встраивание особенно эффективно при захвате сложных взаимодействий категорий. Тщательное рассмотрение соответствующего метода кодирования гарантирует, что категориальные переменные правильно представлены и могут внести значимый вклад в предсказания нейронной сети.

Часть II: Построение и обучение нейронных сетей

Нейронные сети с прямой связью

Структура и принципы работы

Понимание структуры и принципов работы нейронных сетей имеет решающее значение для их эффективного использования. В этой главе мы рассмотрим ключевые компоненты и принципы работы нейронных сетей:

1. Нейроны:

– Нейроны являются основными строительными блоками нейронных сетей.

– Они принимают входные сигналы, выполняют вычисления и выдают выходные сигналы.

– Каждый нейрон применяет линейное преобразование ко входу, за которым следует нелинейная функция активации для введения нелинейности.

2. Слои:

– Нейронные сети состоят из нескольких слоев взаимосвязанных нейронов.

– Входной слой получает входные данные, выходной слой создает окончательные прогнозы, и между ними может быть один или несколько скрытых слоев.

– Скрытые слои позволяют сети изучать сложные представления данных, извлекая соответствующие функции.

3. Веса и смещения:

– Каждая связь между нейронами в нейронной сети связана с весом.

– Веса определяют силу связи и контролируют влияние выхода одного нейрона на вход другого.

– Смещения – это дополнительные параметры, связанные с каждым нейроном, позволяющие им вносить сдвиг или смещение в вычисления.

4. Функции активации:

– Активационные функции привносят нелинейность в вычисления нейронов.

– Они определяют, следует ли активировать нейрон или нет, основываясь на его входе.

– Общие функции активации включают сигмоид, tanh, ReLU (выпрямленный линейный блок) и softmax.

5. Распространение с прямой связью:

– Распространение с прямой связью – это процесс передачи входных данных через слои сети для создания прогнозов.

– Каждый слой выполняет вычисления на основе входных данных, полученных от предыдущего слоя, применяя веса, смещения и функции активации.

– Выходы одного слоя служат входными данными для следующего слоя, продвигаясь по сети до тех пор, пока не будут получены окончательные прогнозы.

6. Обратное распространение:

– Обратное распространение – алгоритм, используемый для обучения нейронных сетей.

– Он вычисляет градиенты функции потерь по отношению к весам и смещениям сети.

– Градиенты указывают направление и величину самого крутого спуска, направляя обновления параметров сети для минимизации потерь.

– Обратное распространение распространяет градиенты назад по сети, слой за слоем, используя правило цепи исчисления.

7. Обучение и оптимизация:

– Обучение нейронной сети включает в себя итеративную настройку ее весов и смещений, чтобы свести к минимуму разницу между прогнозируемыми и фактическими результатами.

– Алгоритмы оптимизации, такие как градиентный спуск, используются для обновления параметров на основе рассчитанных градиентов.

– Обучение обычно включает в себя подачу в сеть помеченных обучающих данных, сравнение прогнозов с истинными метками и соответствующее обновление параметров.

Понимание структуры и принципов работы нейронных

сетей помогает в разработке и обучении эффективных моделей. Регулируя архитектуру, функции активации и процесс обучения, нейронные сети могут изучать сложные взаимосвязи и делать точные прогнозы по различным задачам.

Реализация нейронной сети с прямой связью

Реализация нейронной сети с прямой связью включает в себя перевод концепций и принципов в практическую реализацию кода. В этой главе мы рассмотрим шаги по реализации базовой нейронной сети с прямой связью:

1. Определите сетевую архитектуру:

– Определите количество слоев и количество нейронов в каждом слое.

– Определитесь с функциями активации, которые будут использоваться в каждом слое.

– Определите входные и выходные размеры в зависимости от поставленной задачи.

2. Инициализируйте параметры:

– Инициализируйте веса и смещения для каждого нейрона в сети.

– Случайная инициализация обычно используется, чтобы нарушить симметрию и избежать застревания в локальных минимумах.

3. Реализуйте распространение с прямой связью:

– Передавайте входные данные через слои сети, по одному слою за раз.

– Для каждого слоя вычислите взвешенную сумму входных данных и примените функцию активации для получения выходных данных слоя.

– Прямое распространение продолжается до тех пор, пока не будет достигнут выходной уровень, генерируя прогнозы сети.

4. Определите функцию потерь:

– Выберите подходящую функцию потерь, которая измеряет расхождение между прогнозируемыми выходными данными и истинными метками.

– Общие функции потерь включают среднеквадратичную ошибку (MSE) для задач регрессии и потери кросс-энтропии для задач классификации.

5. Реализуйте обратное распространение:

– Вычислить градиенты функции потерь по отношению к весам и смещениям сети.

– Распространяйте градиенты назад по сети, слой за слоем, используя правило цепи исчисления.

– Обновите веса и смещения с помощью алгоритма оптимизации, такого как градиентный спуск, на основе вычисленных градиентов.

6. Обучите сеть:

– Перебирайте обучающие данные, передавая их в сеть, выполняя прямое распространение, вычисляя потери и об-

новляя параметры с помощью обратного распространения.

– Отрегулируйте скорость обучения, которая контролирует размер шага обновления параметров, чтобы сбалансировать скорость сходимости и стабильность.

– Отслеживайте прогресс обучения, оценивая потери на отдельном проверочном наборе.

7. Оцените сеть:

– После того, как сеть будет обучена, оцените ее производительность на невидимых данных.

– Используйте прямое распространение для создания прогнозов для набора оценочных данных.

– Вычисляйте соответствующие показатели, такие как точность, прецизионность, отзыв или среднеквадратичная ошибка, в зависимости от типа проблемы.

8. Итерация и тонкая настройка:

– Экспериментируйте с различными сетевыми архитектурами, функциями активации и параметрами оптимизации для повышения производительности.

– Настройте модель, настроив гиперпараметры, такие как скорость обучения, размер пакета и методы регуляризации, такие как отсев или регуляризация L2.

Реализация нейронной сети с прямой связью включает в себя перевод математических концепций в код с использованием языка программирования и фреймворка глубокого обучения, такого как TensorFlow или PyTorch. Следуя шагам, описанным выше, и экспериментируя с различными

конфигурациями, вы можете обучать и использовать нейронные сети для различных задач.

Тонкая настройка модели

Тонкая настройка нейронной сети предполагает оптимизацию ее производительности путем корректировки различных аспектов модели. В этой главе мы рассмотрим приемы тонкой настройки нейронной сети:

1. Настройка гиперпараметров:

– Гиперпараметры – это настройки, которые определяют поведение нейронной сети, но не изучаются на основе данных.

– Примеры гиперпараметров включают скорость обучения, размер пакета, количество скрытых слоев, количество нейронов в каждом слое, параметры регуляризации и функции активации.

– Тонкая настройка включает в себя систематическое изменение этих гиперпараметров и оценку производительности сети для поиска оптимальной конфигурации.

2. Планирование скорости обучения:

– Скорость обучения определяет размер шага при обновлении параметров во время обучения.

– Выбор подходящей скорости обучения имеет решающее значение для конвергенции и предотвращения превышения или застревания в локальных минимумах.

– Методы планирования скорости обучения, такие как снижение скорости обучения с течением времени или использование адаптивных методов, таких как Adam или RMSprop, могут помочь точно настроить производительность модели.

3. Методы регуляризации:

– Методы регуляризации предотвращают переобучение и улучшают обобщение, добавляя дополнительные ограничения или штрафы к функции потерь.

– Регуляризация L1 и L2 добавляет штрафной термин к функции потерь в зависимости от величины весов, поощряя меньшие веса и уменьшая чрезмерную зависимость от определенных признаков.

– Dropout случайным образом деактивирует часть нейронов во время обучения, заставляя сеть изучать более надежные и разнообразные представления.

4. Увеличение данных:

– Методы дополнения данных модифицируют обучающие данные, чтобы увеличить их размер и разнообразие, помогая сети лучше обобщать.

– Распространенные методы увеличения данных включают случайную обрезку, поворот, переворачивание и добавление шума или искажений к входным данным.

– Увеличение данных может помочь уменьшить переобучение и улучшить способность модели обрабатывать изменения в реальных данных.

5. Перенос обучения:

– Transfer Learning использует предварительно обученные модели на больших наборах данных и адаптирует их к новым задачам или областям.

Конец ознакомительного фрагмента.