МИНОБРНАУКИ РОССИИ

Санкт-Петербургский государственный
электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

А. Ю. ДОРОГОВ

САМОПОДОБНЫЕ НЕЙРОННЫЕ СЕТИ БЫСТРОГО ОБУЧЕНИЯ

Санкт-Петербург
Издательство СПбГЭТУ «ЛЭТИ»
2024

УДК 004.032.26
ББК З 818
Д69

Дорогов А. Ю.

Д69 Самоподобные нейронные сети быстрого обучения. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2024. 188 с.

ISBN 978-5-7629-3344-5

Издание содержит результаты исследования специального класса многослойных нейронных сетей, обладающих свойством структурного самоподобия. Благодаря этому свойству у сетей данного типа есть возможность быстрого обучения без использования итеративной процедуры обратного распространения ошибки. Алгоритмы обучения гарантированно сходятся за конечное число шагов, при этом самоподобные сети могут дообучаться без потери ранее накопленных знаний.

Основой построения самоподобных сетей являются хорошо известные алгоритмы быстрых спектральных преобразований, Фурье (БПФ) и ему подобные, но обладающие возможностью параметрической перестройки. Быстрые перестраиваемые преобразования рассматриваются как нейронные сети с быстрыми алгоритмами обработки данных и обучения. Отсюда возникло название «быстрые нейронные сети» (БНС). В издании развит математический аппарат БНС, предложены способы их обучения к заданной системе функций. Показано, что свойства БНС хорошо согласуются с системными нейробиологическими исследованиями коры головного мозга человека.

Модификация структуры самоподобной сети до пирамидальной дает возможность полностью использовать потенциал ее обучения, реализуя нейронные сети с глубокой степенью обучения. Показано, что пирамидальные сети можно использовать для создания многомерных многоканальных корреляторов, элементов комбинационной логики и памяти многомерных образов. Предложенные алгоритмы эволюционного и непрерывного обучения самоподобных сетей отвечают исследованиям нейрофизиологов по биологическим нейронным сетям и позволяют достигнуть высокой точности в распознавании образов.

Ориентировано на студентов старших курсов и аспирантов, специализирующихся в области создания интеллектуальных систем реального времени.

УДК 004.032.26
ББК З 818

Рецензенты: д-р техн. наук, проф., зам. генерального конструктора, научный руководитель работ А. И. Яшин (ПАО «Интелтех»); д-р техн. наук, проф. И. А. Бессмертный (Университет ИТМО); д-р техн. наук, д-р физ.-мат. наук, гл. научный сотрудник Н. Г. Макаренко (ГАО РАН).

ISBN 978-5-7629-3344-5

1. ВМЕСТО ВВЕДЕНИЯ

В этой главе представлены обзоры трех направлений исследований нейронных сетей, которые явились стимулом написания данной книги. Первое из них относится к быстрым преобразованиям (типа быстрого преобразования Фурье (БПФ)), два других связаны с исследованиями биологических нейронных сетей. Несмотря на очевидную разнородность вопросов, в них прослеживаются общие идеи и дополняющие взгляды на принципы организации и обучения нейронных сетей. Автор полагает, что эти вопросы требуют разработки в технических приложениях, что позволит преодолеть существующие трудности в развитии нейротехнологии.

1.1. Быстрые спектральные преобразования как прототипы быстрых нейронных сетей

Теория быстрых перестраиваемых спектральных преобразований появилась примерно в то же время, что и теория многослойных нейронных сетей. Обе теории развивались параллельно. Различная терминология, различные теоретические основы, отличающиеся области приложения развели два направления довольно далеко друг от друга, хотя общие черты – многослойность и перестраиваемость – были вполне очевидны.

Слово «спектр» (*spectrum*) было использовано Ньютоном в связи с его исследованиями разложения белого света в набор чистых цветов при пропускании света через стеклянную призму. Это слово, по-видимому, является вариантом латинского *specter*, что означает «призрачное явление». Любой непрерывный сигнал может быть однозначно разложен в базисе линейно-независимых функций, однако математически доказано, что наилучшей сходимостью обладает разложение в базисе ортогональных функций. Результат такого разложения представляется набором коэффициентов в линейной композиции ортогональных функций, который и называется спектром сигнала. Коэффициенты непрерывного спектра вычисляются через интегральные преобразования исследуемого сигнала, а дискретного спектра – через матричные преобразования.

Спектральный анализ находит широкое применение во многих областях деятельности человека, например:

– при мониторинге вибрации спектральный состав измеренных сигналов дает информацию об износе и других характеристиках исследуемых механических деталей;

– в экономике, метеорологии, астрономии и ряде других областей спектральный анализ может выявить «скрытые периодичности» в изучаемых данных, которые связаны с циклическим поведением или повторяющимися процессами в объектах исследования;

– при анализе речи спектральные модели речевых сигналов полезны для лучшего понимания процесса ее создания и, кроме того, могут быть использованы как для синтеза (или сжатия), так и для распознавания речи;

– в радиолокационных и гидроакустических системах спектральное содержание принимаемых сигналов предоставляет информацию о местоположении источников (или целей), расположенных в поле зрения;

– в медицине спектральный анализ различных сигналов, измеренных от пациента, таких как сигналы электрокардиограммы (ЭКГ) или электроэнцефалограммы (ЭЭГ), может предоставить полезный материал для диагностики;

– в сейсмологии спектральный анализ сигналов, зарегистрированных до и во время сейсмического события (такого, как извержение вулкана или землетрясение), дает полезную информацию о движении Земли, связанную с этими событиями, и может помочь в их прогнозировании. Сейсмическая спектральная оценка также применяется для прогнозирования геологического строения недр при разведке газа и нефти;

– в системах управления методы спектрального анализа используются как средства оценки характеристик динамического поведения объектов и синтеза системы управления.

Спектральные матричные преобразования представляют собой дискретный аналог спектрального анализа непрерывных сигналов. В дискретном представлении вычисление спектра сводится к умножению матрицы преобразования на вектор сигнала, что существенно проще вычисления интегральных преобразований. Чем больше размерность векторного представления сигнала, тем с большей точностью спектр дискретного сигнала приближается к спектру непрерывного сигнала. Число вычислительных операций дискретного спектра определяется числом ненулевых элементов в матрице преобразования, и в общем случае это значение пропорционально квадрату размерности матрицы. Это обстоятельство (получившее особое название «проклятие размерности») долгое время служило главным препятствием широкого использования дискретных преобразований для спектрального анализа сигналов, особенно при низком уровне развития вычислительной техники в прошлом веке. Однако оказалось, что для некоторых видов базисных функ-

ций число вычислительных операций можно существенно сократить за счет замены прямого умножения вектора сигнала на матрицу преобразования, поэтапным умножением вектора на последовательность слабозаполенных матриц. Так возникли алгоритмы быстрых спектральных преобразований.

Создание алгоритма быстрого преобразования Фурье, безусловно, можно считать одним из выдающихся достижений второй половины XX в. Время его рождения знаменательно совпало с начальным этапом развития вычислительной техники, когда быстродействие вычислительных машин было еще крайне низким. Использование БПФ позволило кардинальным образом уменьшить количество вычислительных операций при выполнении спектральных преобразований, что значительно расширило сферу применения вычислительной техники для спектрального анализа данных. Реализация БПФ в технологии больших интегральных схем привела к существенному уменьшению площади кристаллов для спектральных анализаторов, и соответственно уменьшилось их энергопотребление. Массовое использование алгоритма БПФ стимулировало интерес и к другим видам спектральных преобразований. В задачах фильтрации, сжатия и выделения информативных признаков широкое применение нашли такие преобразования как Адамара–Уолша, Хаара, Виленкина–Кристенсона, Хартли, наклонное, вейвлет и другие, также обладающие быстрыми алгоритмами. Несмотря на отличия по видам функций, оказалось, что большинство алгоритмов быстрых преобразований имеют подобную структуру и отличаются друг от друга не больше чем значениями коэффициентов базовых операций. Осознание этого факта привело к идее построения обобщенных спектральных преобразований, наделенных быстрым алгоритмом. Поскольку в данном классе преобразований появилась возможность варьировать значения коэффициентов при сохранении условий ортогональности и быстрого алгоритма, то такие преобразования стали называть перестраиваемыми быстрыми спектральными преобразованиями [1]. С появлением перестраиваемых спектральных преобразований появились широкие возможности построения преобразований оптимальных для исследуемого класса сигналов.

Следует отметить, что первый шаг в этом направлении был сделан давно. Еще до появления алгоритма БПФ Кули–Тьюки [2] (1965 г.) в исторической работе Гуда [3] (1958 г.) было приведено аналитическое описание быстрых алгоритмов для многомерных преобразований Фурье. В последующие годы тема обобщенных спектральных преобразований разви-

валась в работах Г. Эндрюса, А. И. Солодовникова, А. М. Спиваковского и других авторов [1], [4]–[6].

Возможность перестройки значений весовых коэффициентов и многослойная структура алгоритма роднит быстрые перестраиваемые преобразования с многослойными нейронными сетями прямого распространения. Иногда используют термин «ортогональные нейронные сети». Первые подходы к обучению подобных преобразований были развиты в работах А. И. Солодовникова и его научной группы [1]. В то время подобный класс преобразований называли приспособленными быстрыми преобразованиями. Условие ортогональности влечет за собой совпадение размерностей входного и выходного вектора для каждого слоя алгоритма. Если отказаться от условия ортогональности, то снимается ограничение и на равенство размерностей, причем, как оказалось, структуру быстрого алгоритма при этом можно сохранить.

В рамках данной парадигмы быстрые линейные перестраиваемые преобразования можно рассматривать как многослойные нейронные сети, которые отличаются от последних линейными функциями активации и нулевыми смещениями в нейронах. Предложенная автором парадигма быстрых нейронных сетей [7] (БНС) наследует структурную регулярность алгоритма БПФ, но расширяет ее возможности за счет отказа от ортогональности и жесткой схемы топологической реализации. По структурной организации БНС следует отнести к модульным нейронным сетям с инъективными связями. Роль модуля выполняют базовые операции малой размерности, например, для БПФ с основанием 2 это базовая операция типа «бабочка».

Теоретическая основа быстрых алгоритмов долгое время базировалась на всевозможных теоремах факторизации, которые доказывали возможность разложения матрицы спектрального преобразования в произведение слабозаполненных матриц, где каждая матрица соответствует одному слою быстрого алгоритма [8], [9]. Это породило всплеск работ по теоремам факторизации. На этом пути исследователи столкнулись с тем обстоятельством, что существует большое множество различных матричных разложений для одного и того же спектрального преобразования. Когда число всевозможных теорем факторизации превысило десятки, стало понятно, что этот путь является тупиковым. Тем не менее поток теорем не закончился до сих пор. Проблема была чисто методическая и заключалась в смешивании понятий структуры и топологии быстрого преобразования. Структура является устойчивым системным инвариантом, свойственным всему классу быстрых алгоритмов, а

топология – это не более чем допустимая реализация системного инварианта в связях между базовыми операциями. Каждая теорема факторизации соответствует одной из допустимых форм топологической реализации, а их число быстро растет с ростом размерности преобразования. Автором было показано [10], что в основу теории быстрых преобразований следует положить именно системные инварианты – это устраняет необходимость изобретения новых теорем факторизации и позволяет предложить общий метод построения различных топологических реализаций быстрого преобразования при неизменной структуре. Более того, выделение структурного и топологического уровней дает возможность решать новые задачи, связанные с оценкой параметрической и топологической пластичности перестраиваемых преобразований. Эти вопросы исследовались Эндрюсом [11] в самом начале пути развития обобщенных преобразований, но тогда удовлетворительного решения получено не было, вернее, оно было ошибочным. Показатели пластичности позволяют оценить разделяющую мощность перестраиваемых преобразований при использовании в задачах классификации образов.

БНС представляют собой вариант многослойных сетей, поэтому для их обучения могут быть использованы градиентные методы типа Error Backpropogation, однако структурные особенности БНС позволяют расширить постановку задачи обучения. Типичной задачей для перестраиваемого преобразования является настройка на заданную систему базисных функций, например базис Фурье, Адамара–Уолша и др. Классические системы базисных функций, как правило, имеют аналитическую форму, поэтому настройка преобразования также выполняется в аналитическом виде.

Для построения приспособленных преобразований важной задачей является реализация базисов, приспособленных к одной или нескольким функциям. Разработанные алгоритмы настройки имеют заведомо известное число шагов (определяемое количеством слоев нейронной сети) и принципиально не имеют проблем сходимости и устойчивости для процесса обучения. В основе алгоритмов настройки лежит доказанное свойство структурного самоподобия БНС. Предложены методы фрактальной фильтрации и сжатия данных. Самоподобные свойства БНС позволяют реализовать регулярные фрактальные последовательности как многослойные БНС.

Подобие структуры быстрых преобразований со структурой тензорных произведений векторных пространств позволяет использовать ортогональные перестраиваемые преобразования для построения алгоритмов квантовых нейронных сетей, где роль модулей квантовой сети выполняют унитарные гейты.

1.2. Морфологические модели коры головного мозга

Рафаэль Лоренте де Но (Lorente de No) (1902–1990) – испанский нейробиолог, который своими основополагающими исследованиями [12] продвинул научное понимание структуры и функции коры головного мозга. В публикациях его анатомических исследований 1933, 1934 гг. [13], [14] впервые были представлены доказательства того, что области коры головного мозга млекопитающих организованы как модульные структуры, где каждый модуль представляется вертикальной колонкой, содержащей группу нейронов. Прежде полагали, что эти области представляют собой однородные горизонтальные слои нейронов. Таким образом, де Но впервые сформулировал основные особенности колончатой организации коры головного мозга млекопитающих.

Исследования были продолжены американским неврологом Верноном Бенджамином Маунткаслом из университета Джонса Хопкинса (Vernon Benjamin Mountcastle, 1918–2015). В 1950-х гг. он подтвердил и охарактеризовал модульную организацию неокортекса[1] головного мозга. Это открытие стало поворотной точкой в исследовании новой коры настолько, что почти все теории о сенсорных функциях, возникшие после публикации Маунткасла о соматосенсорной коре, используют модульную организацию как основу [15], [16]. В 1955–1959 гг. колончатая, или модульная организация неокортекса была задокументирована в исследованиях сенсорных, моторных и гомотипических областей во многих экспериментальных условиях и у многих видов животных. Общепризнанное состояние знаний в этой области можно резюмировать следующими положениями [17].

– Основной единицей кортикальной работы является мини-колонка, «элементарная единица» Лоренте де Но. Она содержит порядка 80…100 нейронов, за исключением полосатой коры головного мозга приматов, где их число более чем удвоено.

– Мини-колонка имеет поперечный диаметр порядка 28…40 мкм, отделена от соседних мини-колонок вертикальными зонами с разреженными клетками, которые различаются по размеру в разных областях коры.

---

[1] Неокортекс (новая кора) располагается в верхнем слое полушарий мозга, имеет толщину 2…4 мм и отвечает за высшие нервные функции: сенсорное восприятие, выполнение моторных команд, осознанное мышление, речь у людей. Человеческий неокортекс содержит 6 различных слоев, каждый из которых можно идентифицировать по типу нейронов. Поверхность неокортекса у человека занимает 95,6 % всей поверхности мозга и содержит 10…14 млрд нейронов.

– Каждая мини-колонка содержит все фенотипы коры головного мозга, и каждая из них имеет несколько выходных каналов.

– Пауэлл (Rockel, Hiorns, Powell, 1974) с сотрудниками показал, что для млекопитающих число нейронов по вертикали, идущей через толщу коры, т. е. в мини-колонке диаметром 30 мкм, поразительно постоянно и составляет около 110.

– Все клетки в пределах одной мини-колонки связаны с одним и тем же рецептивным полем; смежные мини-колонки могут быть связаны с разными рецептивными полями [18]. Выходящий из таламуса[2] аксон подходит к 100…300 мини-колонкам.

Таким образом, было установлено, что основная единица функции в новой коре представляет собой вертикально ориентированную группу клеток, соединенных множеством связей по вертикальной оси, проходящей через все слои коры, с малым числом связей по горизонтали. Важным свойством внутрикорковых механизмов обработки является то, что Маунткасл назвал торможением вокруг колонки – это мощный механизм функциональной изоляции активных колонок от их соседей.

Изучение первичной соматосенсорной и моторной коры и двух областей гомотипической[3] коры теменной доли показало, что в новой коре можно идентифицировать обрабатывающие информацию модули, которые гораздо крупнее мини-колонок. Есть данные, что диаметры или ширина этих более крупных модулей составляют для разных областей от 500 до 1000 мкм. Маунткасл назвал их макроколонками. Новая кора человека содержит около 600 тыс. макроколонок, причем каждая из них содержит от нескольких десятков до нескольких сотен мини-колонок.

Про новую кору известно [15], что вся она гораздо однороднее по своему строению, чем думали раньше; стремительное увеличение ее в процессе филогенеза[4] произошло путем умножения колонок, одинаковых в своей ос-

---

[2] Таламус выполняет несколько важных физиологических функций: отвечает за передачу сенсорной и двигательной информации от органов чувств (кроме информации от органов обоняния) к соответствующим областям коры больших полушарий млекопитающих или плаща мозга низших хордовых. Таламус играет важную роль в регуляции уровня сознания, процессов сна и бодрствования, концентрации внимания

[3] Гомотипия (гомо+тип) – сходство в строении органов, симметрично расположенных относительно вертикальной оси тела, например правая и левая руки.

[4] Филогене́з, или филогени́я (др.-греч. φῦλον, phylon – племя, раса и др.-греч. γενετικός, genetikos – имеющий отношение к рождению) – историческое развитие организмов. Филогенез рассматривает эволюцию в качестве процесса, в котором генетическая линия – организмы от предка к потомкам – разветвляется во времени, и ее отдельные ветви могут приобретать те или иные изменения или исчезать в результате вымирания.

нове, а не путем развития новых типов нейронов или разных способов внутренней организации. Внутренняя структура коры везде одинакова и состоит из повторяющихся многоклеточных единиц. Новая кора с афферентными входами внутри корковых колонок имеет выходы почти ко всем главным образованиям центральной нервной системы (ЦНС), причем значительную их часть составляют массивные возвратные системы с многими входными и выходными точками.

Лавинообразное увеличение неокортекса является важной чертой эволюции млекопитающих; степень этого увеличения отличает приматов от остальных млекопитающих, а человека – от остальных приматов. Если период появления ветви человека оценивают в 20 млн лет, то развитие неокортекса всего 2 млн лет. Нейрофизиологи объясняют такое быстрое развитие многократным повторением структуры коры через копирование многоклеточных нейронных групп.

Важным фактором явилось открытие, что каждый модуль большой структуры включен не во все ее связи. Таким образом, вся группа модулей в совокупности разбита на подгруппы, из которых каждая соединена своей системой связей с такими же обособленными подгруппами в других структурах. Связанные между собой группы модулей нескольких структур названы распределенными системами. Таким образом, распределенные системы состоят из множества модульных элементов, связанных между собой в «эшелонированные» параллельные и последовательные объединения. Наконец, распределенные системы являются по определению и по наблюдению одновременно системами повторного входа и звеньями, связующими входные и выходные каналы нервной системы. Это означает, что множество перерабатывающих информацию модулей в новой коре доступно нервной активности, как генерируемой внутри, так и вызываемой извне.

1.3. Нейродарвинизм как модель обучения мозга

Американский нейрофизиолог Джеральд Морис Эдельман (Gerald Maurice Edelman 1929–2014) в 1972 г. вместе с англичанином Р. Портером (Rodney Porter) получил Нобелевскую премию за исследования структуры и механизмов действия антител иммунной системы человека. Основой открытия послужила селективная теория взаимодействия антител с антигенами. Элементы этой теории Эдельман перенес и на исследования нейронных механизмов мозга, назвав это направление нейродарвинизмом [19]. Нейронный

дарвинизм – это биологический, а точнее, дарвинистский и селекционистский подход к пониманию глобальной функции мозга. Фундаментальный принцип модели Эдельмана заключается в том, что мозг приобретает свои способности к умственной деятельности в результате отбора из обширной популяции нейронных структур, которые являются результатом развития организма.

Принцип селективности предусматривает, что в результате онтогенеза[5] и первых этапов развития головной мозг формирует клеточные конфигурации, которые способны каждая по-своему реагировать на внешние сигналы в силу своих генетически обусловленных структур или благодаря изменениям во время роста и деления клеток, происшедшим независимо от характера внешних сигналов. Внешние сигналы служат лишь для отбора (селекции) из числа предсуществующих конфигураций клеток или клеточных групп с тем, чтобы создать надлежащую реакцию. Принцип селективности предполагает, что нейронные группы не пересекаются по нейронным клеткам, это накладывает определенные ограничения на типы межклеточных связей.

Предпосылкой распознавания является категоризация явлений и объектов, т. е. формирование классов. Эта деятельность лежит в основе первоначальной фундаментальной проблемы: как разум накладывает структуру на Мир, который не имеет присущих ему ярлыков? Роль категоризации приводит Эдельмана к радикальному взгляду на память как на способность нейронных групп к ассоциативному обобщению с перцептивными и двигательными актами организма.

Эдельман полагал, что мозг работает как параллельная распределенная система, это приводит к тому, что один и тот же объект может быть классифицирован по-разному в разное время, и мозг может использовать разные параллельные пути для классификации этого объекта в зависимости от контекста. Под контекстом понимается совокупность условий восприятия объекта, которые через сенсорные карты ассоциативно воздействуют на пространственно-распределенные нейронные группы.

Гипотеза о том, что реентерабельная (двунаправленная) передача сигналов служит общим механизмом для сопряжения функционирования множества областей коры головного мозга и таламуса, была впервые предложена

---

[5] Онтогене́з (от др.-греч. ὤν, лат. on > род. ὄντος, ontos «сущий» + γένεσις, genesis – зарождение) – индивидуальное развитие организма, совокупность последовательных морфологических и биохимических преобразований, претерпеваемых организмом от оплодотворения до конца жизни.

Эдельманом в 1977 и развита в 1978 г. [15], [20]. Повторный вход, по Эдельману, – это динамический процесс постоянной пространственно-временной корреляции, происходящий между функционально разделенными нейронными областями, который опосредуется передачей сигналов через массивно параллельные взаимные нервные волокна. Эдельман отмечает, что следует проводить различие между терминами «повторный вход» и «обратная связь». Обратная связь, как она первоначально была определена в теории систем управления, относится к процессу, в котором сигнал ошибки между желаемым и фактическим выходом передается по заранее заданному пути для воздействия на управляемый объект. Напротив, нейронный повторный вход не использует фиксированные функции или пути исправления ошибок, он не определяется заранее и не определяется априори, а скорее приобретается через опыт. Взаимный обмен сигналами между нейронными сетями в распределенных кортикальных и кортикоталамических областях в сочетании с соответствующими механизмами синаптической пластичности приводит к пространственно-временной интеграции паттернов активности нейронной сети. Это позволяет мозгу классифицировать сенсорную информацию, запоминать и манипулировать ментальными конструкциями, а также генерировать моторные команды. С точки зрения теории динамических систем повторный вход подобен итерационному процессу, сходящемуся к некоторому устойчивому аттрактору, обобщающему накопленные знания.

Следует подчеркнуть, что все процессы категоризации, обучения, ассоциации, повторного входа в нейронных структурах мозга развиваются динамично во времени и взаимно влияют друг на друга.

1.4. Подведение итогов

Конструируя человека, Природа остановилась на многослойном модульном варианте нейронных сетей головного мозга. Каждый модуль содержит от нескольких сотен до нескольких тысяч нейронных клеток. Связи между модулями определяют структуру нейронной сети. Есть большие основания считать, что связи строго организованы, а модульная структура самоподобна. Нейронные модули являются адаптивными элементами, способными настраиваться по входным сигналам. Как установлено исследованиями [18], модули имеют непересекающиеся рецепторные поля. При гипотезе инъективных связей нейронное поле распадается на относительно независимые нейронные группы, которые функционально взаимозаменяемы и генетически предна-

строены на возможные сигналы окружающей среды. Нейронные группы имеют различную структуру, и даже если они настроены на выполнение одной и той же функции, они выполняют ее с разным уровнем качества. Структурная избыточность нейронных групп и межмодульных связей позволяет Природе на начальном этапе развития мозга организовать естественную селекцию по уровню качества или пространственной позиции наиболее подходящих нейронных групп с последующей функциональной донастройкой групп за счет адаптации нейронных модулей к поступающим сигналам. При последующем развитии мозга в нейронных группах возникают дополнительные ассоциации с перцептивными и двигательными актами организма, что ведет к категоризации нейронных групп за счет взаимного усиления или торможения межмодульных связей.

В данной работе будет показано, что эта концепция в значительной степени может быть технически реализована в классе модульных самоподобных нейронных сетей, прототипом которых являются быстрые перестраиваемые преобразования. Последовательное развитие концепции приводит к разработке методов создания нейронных сетей быстрого глубокого обучения с возможностью дообучения к новым данным без потери ранее приобретенных знаний.

Теория быстрых преобразований имеет самостоятельную ценность в технических приложениях, и поэтому ей также будет уделено значительное внимание в этой книге.

Используемая в книги терминология соответствует двум научным направлениям: алгоритмам быстрых перестраиваемых преобразований и нейронным сетям. Автор объединяет эти два направления в технологию нейронных сетей и часто для одного и того же объекта использует разные термины из двух научных областей в зависимости от контекста изложения или исторической обусловленности. Читателю всюду следует отождествлять термины: быстрая нейронная сеть и быстрое перестраиваемое преобразование; нейронное ядро и базовая операция; входы ядра и рецепторы; выходы ядра и аксоны; обучение и настройка. Термин «быстрое» означает наличие либо быстрого алгоритма обработки, либо быстрого алгоритма обучения для сети, либо то и другое вместе. Термин «приспособленное преобразование» сохранен как исторический, он тождествен термину «перестраиваемое преобразование».

2. МОРФОЛОГИЧЕСКИЕ МОДЕЛИ МОДУЛЬНЫХ НЕЙРОННЫХ СЕТЕЙ

На морфологическом уровне не определено внутреннее строение нейронных модулей, значение имеет только их присутствие и наличие связей между ними. Характеристики связей также не определены, важен только факт их наличия. Ключевыми моментами концепции морфологического строения нейронных сетей является многослойность, модульность и самоподобие. В этой главе будут построены математические модели, связывающие эти понятия воедино.

2.1. Самоподобие и морфогенез

По определению, компактное топологическое пространство[6] X самоподобно, если существует конечное множество S, индексирующее набор несюръективных[7] отображений $\{f_s\}_{s\in S}$, для которых

$$X = \bigcup_{s\in S} f_s(X).$$

Пример. Компактом X является замкнутый отрезок $[0,1]$. Позиция точки на отрезке (кроме крайне правой точки) в десятичной системе выражается в виде $z = 0.z_{-1}z_{-2}\ldots z_{-n}\ldots$ в общем случае с бесконечном числом разрядов, где $z_i = 0,1,2,\ldots,9$.

Ограничим в позиционном представлении число разрядов значением n, и будем считать, что число $0.z_{-1}z_{-2}\ldots z_{-n}$ соответствует отрезку длиной 10^{-n} на компакте X, а значение числа указывает начальную позицию этого отрезка на интервале $[0,1)$. В частности для $n=1$ число $0.z_{-1}$ будет соответствовать отрезку длиной 0.1, а значение разряда z_{-1} указывает его позицию на X. Введем для этого уровня множество функций $f_{z_{-1}}(X)$, отображающих компакт X на отрезок длиной 0.1 в позиции $z = 0.z_{-1}$, очевидно, что

$$X = \bigcup_{z_{-1}} f_{z_{-1}}(X).$$

---

[6] Топологическое пространство компактно тогда и только тогда, когда каждая направленность в нем имеет предельную точку, принадлежащую данному множеству, и предкомпактно, если существуют предельные точки, не принадлежащие этому множеству.

[7] Сюръективное отображение – это отображение множества прообразов на выходное множество, при котором каждый элемент выходного множества имеет прообраз. При несюръективном отображении не все элементы выходного множества имеют прообразы.

В этом примере символ объединения можно заменить прямой суммой, поскольку при разбиении отрезки не пересекаются. Таким образом, можно записать:

$$X = \bigoplus_{z_{-1}} f_{z_{-1}}(X).$$

Процесс можно итеративно продолжить, выбирая в качестве предкомпактов отрезки построенного разбиения. Тогда для уровня n получим:

$$X = \bigoplus_{z_{-1}} \bigoplus_{z_{-2}} \dots \bigoplus_{z_{-n}} f_{\langle z_{-1}z_{-2}\dots z_{-n}\rangle}(X).$$

В данном выражении кортеж $\langle z_{-1}z_{-2}\dots z_{-n}\rangle$ определяет многомерный индекс множества несюръективных отображений, разбивающих компакт X на части. Для позиционной системы счисления соответствие значения числа и его позиционного представления $z \leftrightarrow \langle z_{-1}z_{-2}\dots z_{-n}\rangle$ является взаимнооднозначными.

Рассмотренное выше классическое понятие самоподобия ориентировано на обслуживание математических моделей фракталов и не вполне подходит для представления самоподобных объектов нефрактальной природы, поэтому существует необходимость введения обобщающего определения, включающего в себя фрактал как частный случай.

Если обратиться к биологии, то подходящим понятием может служить «морфогенез живых систем». Под морфогенезом при этом понимается процесс возникновения новых структур и изменения их формы в ходе индивидуального развития организмов.

В прикладных областях биологии используется более узкое определение: морфогенез – это возникновение и направленное развитие популяции[8] таксономической группы организмов, где таксóн (τάξις от греч. «порядок», «устройство», «организация») – страта в иерархической классификации, состоящая из дискретных упорядоченных объектов, объединяемых на основании общих свойств и признаков.

Не ставя перед собой задачи построения строгой математической модели морфогенеза, введем рабочее определение, достаточное для задач морфологического синтеза самоподобных сетей.

---

[8] Популяция – это структурная единица вида организмов или иных объектов и элементарная единица эволюции.

Определение 1. Будем говорить, что на популяции индексированных объектов $X = \{A_z\}$ определен *морфогенез*, если развитие популяции определяется наличием для каждого объекта точного соответствия «родитель–потомок», однозначно индексирующего множество порождаемых дочерних объектов в составе популяции. Условие точности устанавливает, что каждый дочерний объект имеет только одного родителя.

При морфогенезе для индексации объектов популяции удобно использовать иерархическую систему с многопозиционным представлением индекса $z = \langle z_0 z_1 \ldots z_n \ldots \rangle$. В рамках этой системы нулевая позиция содержит номер объектов начальной популяции, а каждая последующая – локальный номер дочернего объекта следующего поколения. Локальный номер устанавливается по индексу ветви точного соответствия. При такой многопозиционной индексации в популяции можно выделить объекты, принадлежащие разным поколениям.

Например, пусть $X = X_0 = \bigoplus\limits_{z_0} A_{z_0}$ – начальная индексированная популяция, тогда направленное развитие популяции по поколениям определяется последовательностью индексируемых популяций вида:

$$X_0 = \bigoplus_{z_0} A_{z_0}, \quad X_1 = \bigoplus_{z_1}\bigoplus_{z_0} A_{\langle z_1 z_0 \rangle}, \ldots, X_{n-1} = \bigoplus_{z_{n-1}} \ldots \bigoplus_{z_1}\bigoplus_{z_0} A_{\langle z_{n-1}\ldots z_1 z_0 \rangle}.$$

Если X – это компактное топологическое пространство, то популяцией объектов может служить любое конечное семейство компактных множеств, в совокупности покрывающих пространство X. Такое семейство существует, поскольку для компактного множества всегда существует конечное покрытие открытыми множествами, замыкание которых и порождает это семейство.

Пусть точное соответствие, присущее каждому родительскому объекту, порождает конечное подмножество дочерних объектов, в совокупности покрывающих компакт родительского объекта. В этом случае на любой стадии развития популяция покрывает все пространство X. Отображение «родитель–потомок» является несюръективным для родительского объекта, поскольку при числе дочерних объектов больше единицы оно покрывает только часть компакта родительского объекта. Существование индексирующего множества для объектов популяции следует из конечности образов точных соответствий «родитель–потомок». Таким образом, понятие морфогенеза на компакте трансформируется к свойству самоподобия множества X и обобщает его с точки зрения развития популяции.

Определение 2. Морфогенез назовем *регулярным*, если в каждом поколении соответствие «родитель–потомок» совпадает для всех объектов одного поколения.

2.2. Морфогенез многослойной сети

Пусть многослойный сетевой граф Γ имеет n слоев. Обозначим через $\left\{A_i^m\right\}$ множество вершин в слое m, где $m = 0, 1, \ldots, n-1$. Множество вершин входного (нулевого) слоя назовем *афферентом* сети и обозначим $Aff\,(\Gamma)$, а множество вершин конечного слоя назовем *эфферентом*[9] сети и обозначим $Eff\,(\Gamma)$. Пусть A_i^m – некоторая вершина сети в слое m. Назовем афферентом вершины (далее обозначается $Aff\left(A_i^m\right)$) подмножество вершин афферента сети, связанного дугами с вершиной A_i^m, таким образом, имеем $Aff\left(A_i^m\right) \subset Aff\,(\Gamma)$. Аналогично введем понятия эфферента вершины $Eff\left(A_i^m\right)$ как подмножество вершин эфферента сети, связанного дугами с вершиной A_i^m, таким образом – $Eff\left(A_i^m\right) \subset Eff\,(\Gamma)$. Афференты и эфференты вершины будем также называть ее *терминальными проекциями*.

Обозначим через $\Gamma^{-1}\left(A_i^m\right)$ рецепторную окрестность вершины A_i^m, а через $\Gamma\left(A_i^m\right)$ – ее аксоновую окрестность. Правило построения графа зададим следующими выражениями:

$$\begin{aligned} Aff\left(A_i^m\right) &= \bigoplus_{A_k^{m-1} \in \Gamma^{-1}\left(A_i^m\right)} Aff\left(A_k^{m-1}\right), \\ Eff\left(A_i^m\right) &= \bigoplus_{A_k^{m+1} \in \Gamma\left(A_i^m\right)} Eff\left(A_k^{m+1}\right). \end{aligned} \tag{2.1}$$

Эти правила были предложены автором данной книги в 2004 г. [21] и названы условиями слабой связанности, поскольку их реализация порождает слабосвязанные сети. Символ прямой суммы подчеркивает, что для любой вершины сети терминальные проекции вершин ее окрестности не пересекаются.

---

[9] В биологии под афферентами понимают чувствительные нервные окончания (чувствительные рецепторы), расположенные в органах и тканях и способные воспринимать разного рода раздражения. Эфферентами называют выходы нейронных подсистем, воздействующих на двигательные нервные волокна.

В данном случае объектами популяции являются проекции вершин слоев на терминальные множества. Можно показать, что выражения (2.1) двойственны друг другу, и если выполняется одно из них, то будет выполнено и второе [22]. Введенные условия в общем случае не ограничены регулярностью. Мы сохраним за ними термин *условия слабой связанности*, но для общности будем иногда называть их условиями морфогенеза самоподобной сети.

По отношению к проекциям на терминальное поле сети множество вершин слоя разбивается на афферентные и эфферентные классы. Афферентному классу принадлежат вершины нейронного слоя, имеющие общее афферентное множество, назовем эти классы афферентными доменами и обозначим $Dom_p\left(A^m_{z^m}\right) = X^m_{i^m}$, здесь $A^m_{z^m}$ – один из представителей домена, i^m – порядковый номер домена в слое m. Аналогично отношение проекции вершин слоя m на терминальное эфферентное поле сети разбивает множество вершин нейронного слоя на классы – эфферентные домены, которые обозначим $Dom_g\left(A^m_{z^m}\right) = Y^m_{j^m}$, где j^m – порядковый номер эфферентного домена в слое m.

Для многослойных слабосвязанных сетей число слоев зависит от числа вершин в терминальных слоях. Поясним характер этой зависимости на примере регулярных сетей. Будем полагать, что n-слойный граф сети состоит из одной компоненты связанности и удовлетворяет условиям (2.1). В этом случае на основании первого условия каждая вершина выходного слоя будет связана со всеми вершинами входного слоя, это означает, что в выходном слое существует всего один афферентный домен. Пусть этот домен имеет номер 0. При регулярном морфогенезе размеры всех рецепторных окрестностей вершин выходного слоя одинаковы, положим их равными p_{n-1}. Аффе-ренты вершин окрестностей вершины конечного слоя (т. е. слоя $n-1$) не пересекаются, и поэтому множество вершин слоя $n-2$ распадается на домены, которые пронумеруем индексом i_{n-1}. Индекс будет изменяться в диапазоне от 0 до $p_{n-1}-1$. Таким образом, единственный домен конечного слоя в результате первого шага морфогенеза трансформировался в p_{n-1} доменов слоя $n-2$, которые обозначим $Dom_p^{n-2}\langle i_{n-1}\rangle$. Будем полагать, что каждая вершина слоя $n-2$ имеет рецепторную окрестность, состоящую из p_{n-2} вершин. В слое $n-3$ домены в результате морфогенеза получат номера, определяемые двойным индексом $Dom_p^{n-3}\langle i_{n-2}i_{n-1}\rangle$. Таким образом, достигнув

начального слоя, получим, что нумерация доменов этого слоя будет определяться сложным индексом $Dom_p^0\langle i_1 i_2 \ldots i_{n-2} i_{n-1}\rangle$. Но в начальном слое каждый афферентный домен состоит только из одной вершины, поэтому построенный сложный индекс будет определять и номер вершины начального слоя. Афферентные домены промежуточных слоев индексируются выражением $Dom_p^m\langle i_{m+1} i_{m+2} \ldots i_{n-2} i_{n-1}\rangle$.

Аналогично каждая вершина входного слоя связана со всеми вершинами выходного слоя, это означает, что во входном слое существует всего один эфферентный домен. Для регулярного морфогенеза зададим размеры аксоновых окрестностей по слоям числами $g_0, g_1, \ldots, g_{n-2}$ и будем использовать индексы $j_0, j_1, \ldots, j_{n-3}, j_{n-2}$ для нумераций доменов. Повторяя последовательность регулярных разбиений, получим, что для конечного слоя номер домена будет определяться сложным индексом $Dom_g^{n-1}\langle j_0 j_1 \ldots j_{n-3} j_{n-2}\rangle$. Но в конечном слое каждый эфферентный домен состоит только из одной вершины, поэтому построенный сложный индекс будет определять и номер вершины конечного слоя. Эфферентные домены промежуточных слоев индексируются выражением $Dom_g^m\langle j_0 j_1 \ldots j_{m-2} j_{m-1}\rangle$. Следующая теорема является обоснованием использования термина «слабосвязанная сеть» [21].

Теорема о параллельных путях. В слабосвязанных сетях отсутствуют параллельные пути.

Доказательство. Предположим противное. Пусть два параллельных пути сходятся в вершине B, проходя через ее окрестные вершины A_1 и A_2 (рис. 2.1).

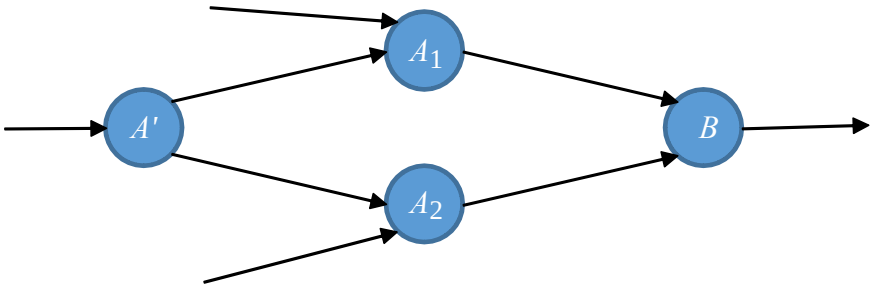


Рис. 2.1. Параллельные пути

По условию параллельности пути имеют точку пересечения в некоторой вершине A', предшествующей вершинам A_1 и A_2. Следовательно, афференты вершин A_1 и A_2 пересекаются так, что $Aff(A_1) \cap Aff(A_2) = Aff(A')$, но это противоречит условию слабой связанности. Поскольку утверждение

справедливо для любой вершины, включая терминальные, то отсюда следует невозможность параллельных путей для всей сети. ■

Теорема о морфологии слабосвязанной сети. В каждом слое слабосвязанной сети с одной компонентой связанности афферентные и эфферентные домены попарно пересекаются во всех возможных сочетаниях точно по одной вершине. То есть для каждой парной комбинации афферентного и эфферентного доменов существует единственная вершина $A_{z^m}^m$, такая что

$$Dom_p\left(A_{z^m}^m\right) \cap Dom_g\left(A_{z^m}^m\right) = A_{z^m}^m .$$

Доказательство. Нужно доказать, что все парные сочетания разнотипных доменов в слое имеют не пустое пересечение и что эти пересечения состоят только из одной вершины слоя.

Докажем первое. Предположим, что в слое m существует пара разнотипных доменов, которые не пересекаются по вершинам слоя. В этом случае вершины афферентного домена не связаны с частью вершин эфферента сети, и поэтому в сети выделяются два правых конуса вершин, не связанных между собой во всех последующих слоях. С другой стороны, вершины эфферентного домена не связаны с частью вершин афферента сети, и в сети выделяются два левых конуса не связанных между собой в предшествующих слоях. Поскольку домены не пресекаются, то в этом случае в сети существуют две не связанные между собой компоненты, что противоречит условию теоремы об однокомпонентности сети.

Докажем второе. Предположим, что домены пересекаются по двум вершинам A_1^m и A_2^m. Тогда эта пара вершин имеет общую афферентную окрестность. То есть в слое $m-1$ существует, по крайней мере, одна вершина A_k^{m-1}, связанная с вершинами A_1^m и A_2^m, и ее эфферент по условию морфогенеза (2.1) должен быть равен прямой сумме эфферентов вершин ее аксоновой окрестности. В эту прямую сумму входят и вершины A_1^m и A_2^m, но поскольку по предположению они принадлежат одному эфферентному домену слоя, то для них выполняется $Eff\left(A_1^m\right) \cap Eff\left(A_2^m\right) \neq \varnothing$ (см. рис. 2.2). Таким образом, мы пришли к нарушению условия морфогенеза (2.1) по эфферентам вершин. Аналогично это положение доказывается с использованием эфферентной окрестности пары вершин A_1^m и A_2^m. ■

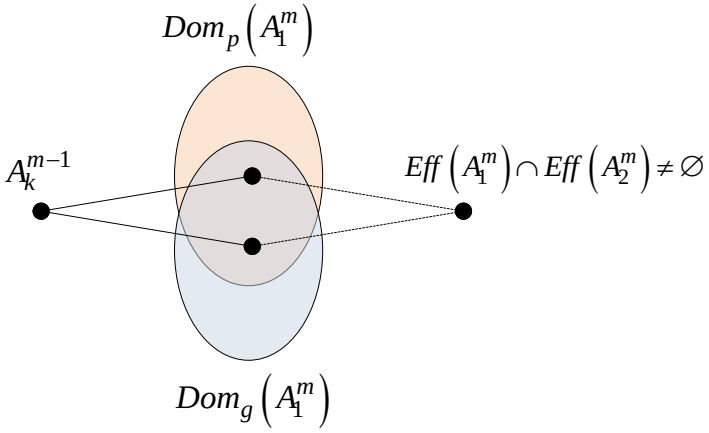


Рис. 2.2. Теорема о морфологии слабосвязанной сети

Из этой теоремы следует, что для самоподобной сети существует взаимно-однозначное соответствие между номером вершины в пределах слоя и индексами пары пересекающихся разнотипных доменов. Соответствие может быть задано кортежем, представляющим собой поразрядное представление числа в многоосновной системе счисления:

$$z^m = \langle j_0 j_1 \dots j_{m-1} i_{m+1} i_{m+2} \dots i_{n-1} \rangle. \qquad (2.2)$$

В этом кортеже допустима любая перестановка индексов.

2.3. Граф самоподобной сети

Из выражения (2.2) для слоя $m-1$ получим:

$$z^{m-1} = \langle j_0 j_1 \dots j_{m-2} i_m i_{m+1} \dots i_{n-1} \rangle.$$

Вершины смежных слоев m и $m-1$ связаны дугой, если их афференты пересекаются. Афферент вершины z^{m-1} определяется разрядными числами кортежа $\langle i_m i_{m+1} \dots i_{n-1} \rangle$, а вершины z^m – разрядными числами кортежа $\langle i_{m+1} i_{m+2} \dots i_{n-1} \rangle$, пересечение афферентов возможно только в том случае, когда значения одноименных разрядов в приведенных кортежах совпадают. Такой же вывод можно сделать, используя условия пресечения эфферентов смежных слоев m и $m+1$: вершины будут связаны дугой, если в кортежах $\langle j_0 j_1 \dots j_{m-1} \rangle$ и $\langle j_0 j_1 \dots j_{m-1} j_m \rangle$ одноименные разряды будут совпадать по значениям. Полученные правила достаточно просто выполнить при построение графического образа самоподобной модульной сети. На рис. 2.3 показан пример построения четырехслойной самоподобной сети для варианта, когда для всех m индексы принимают значения $i_m = \{0,1\}$ и $j_m = \{0,1\}$.

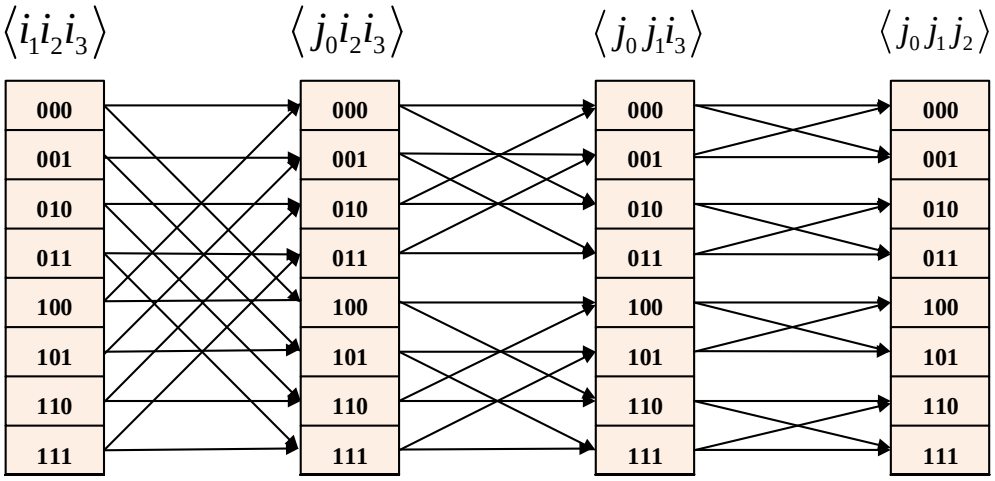


Рис. 2.3. Четырехслойная самоподобная модульная сеть

Построенная модель полностью описывается лингвистическим предложением:

$$\left[\langle i_1 i_2 i_3 \rangle \langle j_0 i_2 i_3 \rangle \langle j_0 j_1 i_3 \rangle \langle j_0 j_1 j_2 \rangle\right].$$

Здесь каждое слово предложения представляет кортеж z^m номера вершины сети. В словах предложения допустима любая перестановка индексов.

Обозначим через I_m и J_m упорядоченные подмножества индексов каждого типа в кортеже z^m, тогда правило построения графического образа n-слойной самоподобной сети в общем виде можно сформулировать следующим образом:

$$m = 0,1,\ldots n-1, \qquad I_m \supset I_{m+1}, \qquad J_m \subset J_{m+1},$$
$$I_{n-1} = \varnothing, \quad J_0 = \varnothing.$$

Полученное правило не связано ни с размерностью сети, ни со структурными характеристиками ее модулей, ни с топологией связей – это правило является инвариантом морфологического уровня многослойной самоподобной модульной сети.

Следует отметить, что класс слабосвязанных сетей несколько шире класса самоподобных сетей, в частности к этому классу относятся полносвязанные многослойные сети, однослойные персептроны, двухслойные ядерные сети с произвольной структурой и инъективными связями.

3. СТРАТИФИЦИРОВАННЫЕ МОДЕЛИ САМОПОДОБНЫХ НЕЙРОННЫХ СЕТЕЙ

Математические модели служат средством изучения и проектирования систем. От модели требуется, чтобы она была простой, но функционально достаточной. Основной проблемой математического моделирования является нахождение приемлемого компромисса между детализацией и простотой описания. Один из путей решения этой проблемы заключается в формировании иерархически вложенных семейств моделей, где каждый уровень иерархии соответствует уровню разумного абстрагирования свойств системы, что ведет к упрощению каждой частной модели. Такое многоуровневое представление модели принято называть *стратификацией* [23], а каждый уровень модельного представления – стратой. Стратификация позволяет исследовать систему на разных стадиях познания и описывать каждый уровень адекватными средствами. Стратифицирование можно рассматривать и как средство последовательного углубления представления о системе: при спуске по иерархии страт вниз система раскрывается в деталях; при подъеме на более абстрактные уровни яснее становится смысл и значение всей системы. В математической формулировке стратификация связана с выделением на каждом уровне иерархии эквивалентных отношений и переходом к факторным моделям, описывающим следующий уровень.

Для самоподобных модульных сетей в предыдущей главе были построены модели морфологического уровня. На морфологическом уровне описываются условия порождающего морфогенеза и системные инварианты самоподобных сетей. В настоящей главе будут рассмотрены модели структурного, топологического и параметрического уровней стратификации.

В *структурной модели* представлены размерности модулей и ранги межмодульных связей, но отсутствует информация об индексах координат вектора данных. Эта модель предназначена для оценки качественных показателей быстрого перестраиваемого преобразования, таких как быстродействие и пластичность.

Привязка векторов к структурной модели может иметь множество вариантов и порождает множество топологических реализаций быстрых преобразований. *Топологическая модель* позволяет выбрать форму алгоритма, удобную для практической реализации.

Топологическая модель, дополненная значениями коэффициентов базовых операций, образует модель *параметрического уровня*. На параметриче-

ском уровне реализуются методы обучения и настройки нейронных сетей по заданным показателям качества. Страты модельных представлений упорядочены по степени абстракции (рис. 3.1).

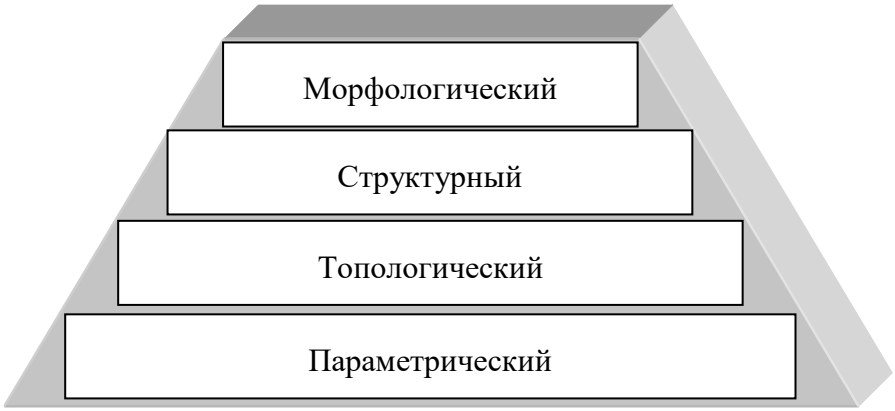


Рис. 3.1. Иерархия уровней стратифицированной модели модульной нейронной сети

Высший уровень абстракции соответствует морфологическому представлению, а низший – параметрическому. Стратификация моделей позволяет разделить проектирование нейронной сети на относительно независимые этапы и использовать для каждого этапа специфичный математический аппарат.

3.1. Структурная модель модульной нейронной сети

Введем структурные характеристики для модулей и межмодульных связей многослойной нейронной сети. Нейронный модуль слоя m с номером z^m обозначим $A^m_{z^m}$. Для модуля $A^m_{z^m}$ обозначим через p_{z^m} размерность рецепторного поля, а через g_{z^m} – размерность аксонового поля. Модуль выполняет обработку данных и в общем случае описывается нелинейным оператором. Математический оператор характеризуется операторным рангом. Когда возможности модуля по обработке данных используются полностью, модуль не имеет «висящих» (т. е. не связанных с другими вершинами) рецепторов или аксонов и его максимальный ранг определяется выражением $rank\left(A^m_{z^m}\right) = \min\left(p_{z^m}, g_{z^m}\right)$. Такой модуль назовем *полным*.

Будем полагать, что для нейронной сети связь между нейронными модулями A^m_l и A^{m+1}_t смежных слоев представляет собой линейный оператор, связывающий аксоновое и рецепторное поле двух соседних модулей. Оператор связи характеризуется рангом и выражается прямоугольной матрицей вида:

$$P_{lt} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1k} \\ b_{21} & b_{22} & \cdots & b_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ b_{s1} & b_{s2} & \cdots & b_{sk} \end{pmatrix}.$$

Ранг оператора связи удовлетворяет условию $r_{lt} = rank\left(P_{lt}\right) \le \min(s,k)$. Целесообразно упростить модель нейронной сети, полагая, что вся обработка сосредоточена в нейронных модулях, а связи осуществляют только передачу данных без внутренней обработки и искажений. Это приводит к необходимости отказаться от ветвящихся связей в нейронной сети и полагать, что все межмодульные связи являются инъективными с тождественной передачей. Данные упрощения не снижают возможностей сети по обработке данных, поскольку ветвления и масштабирование данных могут быть организованы внутри нейронных модулей. При сделанных допущениях все матрицы операторов связей являются единичными:

$$P_{lt} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

и ранг оператора в этом случае равен размерности единичной матрицы. Инъективные неискажающие связи будем далее называть *транслирующими*. Для модульной нейронной сети с инъективными связями для каждого полного модуля A_t^m, очевидно, выполняются следующие ранговые соотношения:

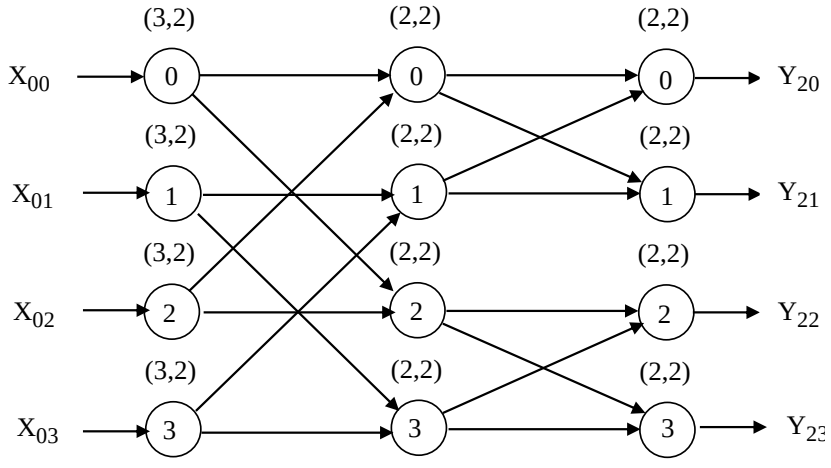


Рис. 3.2. Структурная модель самоподобной модульной нейронной сети с одноранговыми связями

$$p_t = \sum_a r_{at}, \quad g_t = \sum_e r_{te},$$

где r_{at} – ранги входных связей и r_{te} – ранги выходных связей модуля с соседями.

Пример структурной модели модульной самоподобной нейронной сети с одноранговыми инъективными связями показан на рис. 3.2. Модули входного слоя имеют размерность $(3, 2)$, а остальных слоев – $(2, 2)$.

Структурная модель отличается от морфологической наличием весов вершин и дуг на графе модели.

3.1.1. Морфогенез структурного уровня

Условия морфогенеза самоподобной многослойной сети на морфологическом уровне были определены через проекции вершин на терминальные поля сети. По доказанной теореме «О морфологии слабосвязанной сети» граф самоподобной сети может быть выражен лингвистическим предложением, в котором каждое слово представляет поразрядное представление номера вершины z^m слоя в алфавите с двумя типами индексов i и j:

$$z^m = \langle j_0 j_1 \cdots j_{m-1} i_{m+1} i_{m+2} \cdots i_{n-1} \rangle.$$

Покажем, как эти условия морфогенеза трансформируются на структурный уровень. Свяжем с каждым модулем $A^m_{z^m}$ пространства $P(z^m)$ и $G(z^m)$, которые назовем ассоциативными к входу и выходу модуля. Размерности ассоциативных пространств определяются размерностями терминальных полей нейронного модуля так, что $p_{z^m} = \dim(P(z^m))$, $g_{z^m} = \dim(G(z^m))$. Поскольку межмодульные связи для самоподобной сети являются инъективными, то одноименные ассоциативные пространства слоя не пересекаются между собой. Поэтому с афферентом сети будет связано векторное пространство, равное прямой сумме входных ассоциативных векторных пространств нулевого слоя:

$$P(Aff) = \bigoplus_{z^0} P(z^0),$$

а с эфферентом – векторное пространство, равное прямой сумме выходных ассоциативных векторных пространств конечного слоя сети:

$$G(Eff) = \bigoplus_{z^{n-1}} G(z^{n-1}).$$

Напомним, что афферентом сети называется множество вершин входного (нулевого) слоя сети, а эфферентом – множество вершин последнего слоя.

Входной и конечный слои сети называются терминальными слоями. Полагая, что размерность сети по входу равна N, а по выходу – M, можно записать:

$$N = \sum_{z^0} p_{z^0}, \qquad M = \sum_{z^{n-1}} p_{z^{n-1}}.$$

Ассоциативные пространства модуля слоя m через цепочки транслирующих связей проектируются на терминальные пространства сети. Модули слоя, имеющие совпадающие проекции, образуют домены. Собственным пространством домена является прямая сумма ассоциативных векторных пространств его представителей. Проекции собственных пространств доменов в терминальных слоях не пересекаются и равны прямой сумме ассоциативных входных или выходных пространств модулей терминальных слоев.

Если на векторном пространстве фиксировано его разложение в прямую сумму подпространств, то говорят, что векторное пространство градуировано [24]. Таким образом, морфогенез уровня морфологии индуцирует морфогенез на популяции градуированных подпространств терминальных слоев.

Для структурной модели необходимо задать ранги межмодульных связей. Поскольку связи инъективны, то для каждого модуля выполнены условия:

$$p_{z^m} = \sum_{z^{m-1}} r\left(z^{m-1}, z^m\right), \qquad g_{z^m} = \sum_{z^{m+1}} r\left(z^m, z^{m+1}\right),$$

где $r\left(z^{m-1}, z^m\right)$ и $r\left(z^m, z^{m+1}\right)$ – ранги связей между смежными модулями. Условия морфогенеза не регламентируют размерности модулей и ранги связей. Теорема морфологии слабосвязанной многослойной сети требует только взаимно-однозначного соответствия между номером вершины в пределах слоя и индексами пары пересекающихся разнотипных доменов:

$$z^m \leftrightarrow \left(i^m, j^m\right).$$

Сеть назовем *структурно-регулярной*, если в пределах слоя все вершины имеют одинаковые размерности и совпадающие ранги связей по входу и по выходу. При выполнении условий структурной регулярности размерности модулей для n-слойной сети можно задать в виде таблиц, в верхней строчке которых указан номер слоя, а в нижних – размерности модулей слоя по входу и выходу:

$$\begin{pmatrix} 0 & 1 & \dots & n-1 \\ p_0 & p_1 & \cdots & p_{n-1} \end{pmatrix}, \qquad \begin{pmatrix} 0 & 1 & \dots & n-1 \\ g_0 & g_1 & \cdots & g_{n-1} \end{pmatrix}.$$

Левая таблица определяет размерности рецепторных полей нейронных модулей по слоям, а правая – размерности аксоновых полей. Для структурно-

регулярных сетей кортеж индексов можно рассматривать как позиционное представление номера в многоосновной системе счисления с основаниями $\{p_m, g_m\}$. Например, полагая, что левый разряд кортежа номера вершины является старшим, можно записать:

$$z^m = \langle j_0 j_1 \dots j_{m-1} i_{m+1} i_{m+2} \dots i_{n-1} \rangle = j_0 g_1 g_2 \dots g_{m-1} p_{m+1} p_{m+2} \dots p_{n-1} + \dots$$
$$\dots + j_1 g_2 g_3 \dots g_{m-1} p_{m+1} p_{m+2} \dots p_{n-1} + \dots + i_{n-2} p_{n-1} + i_{n-1}.$$

На рис. 3.2 показана самоподобная нейронная сеть, в которой все связи являются одноранговыми, а все модули имеют равные размерности в пределах слоя. Большим преимуществом регулярных самоподобных сетей является возможность получить аналитические выражения для алгоритмов структурного синтеза и обучения нейронных сетей. Структурная регулярность коррелируется с понятием регулярности графа. По определению [25] граф называется регулярным, если степени всех вершины графа равны. Для ориентированных графов раздельно учитываются степени вершин по входящим и исходящим дугам. Для одноранговых самоподобных сетей оба понятия совпадают. Структурный синтез нерегулярных самоподобных сетей рассмотрен автором в работе [26].

3.2. Топологическая модель модульной нейронной сети

Входы и выходы структурной модели на рис. 3.2 показаны как трех- и двухкоординатные векторы без привязки к номеру отсчета. Структурная модель непосредственно не перекладывается на алгоритм обработки в сети, ее назначение – исследовать общие свойства модульной сети, связанные с вычислительной эффективностью и пластичностью. Для описания алгоритма необходимо построить топологическую модель. В топологической модели элементами рассмотрения являются физические контакты нейронных модулей; это либо входные рецепторы, либо выходные аксоны модуля.

Рассмотрим модульную сеть, где все модули являются полными, а все связи транслирующими. Обозначим для модулей z^m слоя m через $u_{z^m} = \left[0,1,\dots,\left(p_{z^m} - 1\right)\right]$ локальный номер рецептора, а через $v_{z^m} = \left[0,1,\dots,(g_{z^m} - 1)\right]$ – локальный номер аксона модуля. Позиционный номер рецептора в пределах нейронного слоя обозначим через U^m, а позиционный номер аксона – через V^m. Совокупность отображений вида

$$\left\{u_{z^m}\right\} \to U^m, \qquad \left\{v_{z^m}\right\} \to V^m$$

назовем *топологической моделью* нейронного слоя. Эти отображения являются инъективными в виду инъективности межмодульных связей, и проекционными, поскольку в рассматриваемой модели все модули являются полными. Следовательно, для сети данного вида модельные отображения взаимно-однозначны.

Рассмотрим теперь структурно-регулярную самоподобную сеть (в которой все модули в пределах каждого слоя имеют одинаковые структурные характеристики $[p_m, g_m]$). В этом случае можно упростить обозначения для номера рецептора и номера аксона нейронного модуля, записав:

$$u_m = \left[0,1,\ldots,(p_m - 1)\right], \qquad v_m = \left[0,1,\ldots,(g_m - 1)\right].$$

Будем также предполагать, что все связи в сети являются одноранговыми. Тогда топологическая модель слоя может быть выражена кортежами:

$$U^m = \left\langle \left\langle z^m \right\rangle \oplus u_m \right\rangle, \qquad V^m = \left\langle \left\langle z^m \right\rangle \oplus v_m \right\rangle.$$

Символ $\oplus$ в данном случае подчеркивает, что место размещения дополнительных разрядов u_m и v_m в кортеже $z^m = \langle j_0 j_1 \cdots j_{m-1} i_{m+1} i_{m+2} \cdots i_{n-1} \rangle$ может быть произвольным.

При одноранговых связях каждой дуге графа структурной модели однозначно ставится в соответствие дуга топологической модели, например, можно выбрать следующие взаимно-однозначные соответствия: $i_m \leftrightarrow u_m, \quad j_m \leftrightarrow v_m$. Тогда возможен следующий вариант отображений топологической модели:

$$\begin{aligned} U^m &= \langle v_0 v_1 \cdots v_{m-1} u_m u_{m+1} \cdots u_{n-1} \rangle, \\ V^m &= \langle v_0 v_1 \cdots v_{m-1} v_m u_{m+1} \cdots u_{n-1} \rangle, \\ z^m &= \langle v_0 v_1 \cdots v_{m-1} u_{m+1} \cdots u_{n-1} \rangle. \end{aligned} \tag{3.1}$$

В особой ситуации находятся терминальные поля сети, поскольку для разрядных переменных u_0 и v_{n-1} нет соответствующих разрядов в кортеже z^m, но это не препятствует построению кортежей топологических моделей слоя. Например, для структурной модели $\left[\langle i_1 i_2 i_3 \rangle \langle j_0 i_2 i_3 \rangle \langle j_0 j_1 i_3 \rangle \langle j_0 j_1 j_2 \rangle\right]$ можно построить следующие лингвистические предложения топологической модели.

Для рецепторных полей:

$$Rp = \left[\langle u_0 u_1 u_2 u_3 \rangle \langle v_0 u_1 u_2 u_3 \rangle \langle v_0 v_1 u_2 u_3 \rangle \langle v_0 v_1 v_2 u_3 \rangle\right].$$

Для аксоновых полей:

$$Ax = \left[\langle v_0 u_1 u_2 u_3 \rangle \langle v_0 v_1 u_2 u_3 \rangle \langle v_0 v_1 v_2 u_3 \rangle \langle v_0 v_1 v_2 v_3 \rangle \right].$$

Удобно эту модель представить одним лингвистическим предложением:

$$\left[\langle u_0 u_1 u_2 u_3 \rangle \langle v_0 u_1 u_2 u_3 \rangle \langle v_0 v_1 u_2 u_3 \rangle \langle v_0 v_1 v_2 u_3 \rangle \langle v_0 v_1 v_2 v_3 \rangle \right].$$

Первое слово в этом предложении соответствует рецепторному полю сети, а последнее – аксоновому. Граф топологической модели строится по тем же правилам, что и граф морфологической модели: дугами соединяются вершины, имеющие одинаковые значения разрядных переменных в смежных слоях. Граф топологической модели для данного примера показан на рис. 3.3.

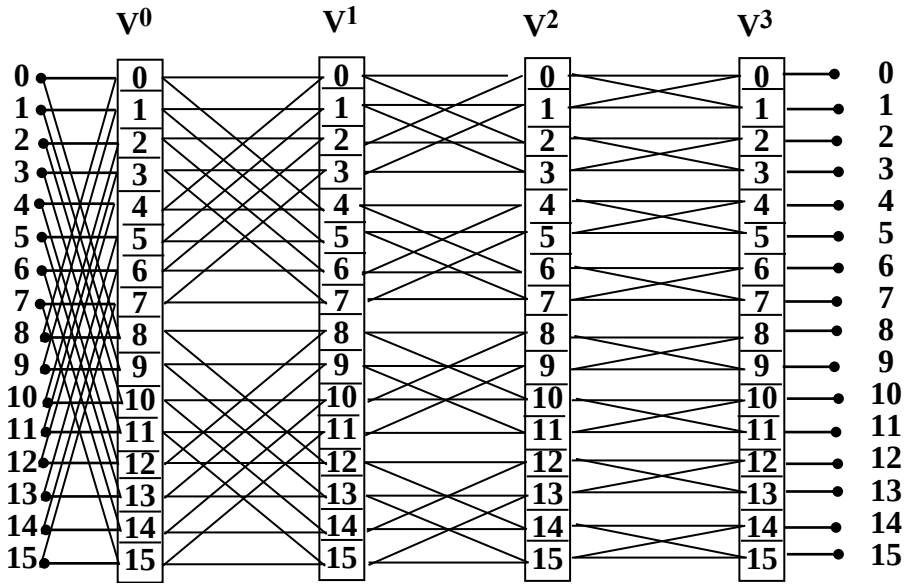


Рис. 3.3. Топологическая модель алгоритма БПФ «с прореживанием по частоте»

Нетрудно убедиться, что данная модель соответствует графу быстрого преобразования Фурье в топологии Кули–Тьюки «с прореживанием по частоте» [8]. Другой вариант топологической модели может быть задан в виде:

$$\begin{aligned} U^m &= \langle u_{n-1} u_{n-2} \cdots u_{m+1} u_m v_{m-1} v_{m-2} \cdots v_1 v_0 \rangle, \\ V^m &= \langle u_{n-1} u_{n-2} \cdots u_{m+1} v_m v_{m-1} v_{m-2} \cdots v_1 v_0 \rangle, \\ z^m &= \langle u_{n-1} u_{n-2} \cdots u_{m+1} v_{m-1} v_{m-2} \cdots v_1 v_0 \rangle. \end{aligned} \qquad (3.2)$$

Эта модель соответствует графу БПФ в топологии Кули–Тьюки «с прореживанием по времени». Граф данной топологической модели показан на рис. 3.4.

Из полученных результатов можно сделать вывод, что алгоритм быстрого преобразования Фурье является топологической реализацией самоподобной модульной сети, где модулями являются базовые операции типа «бабочка».

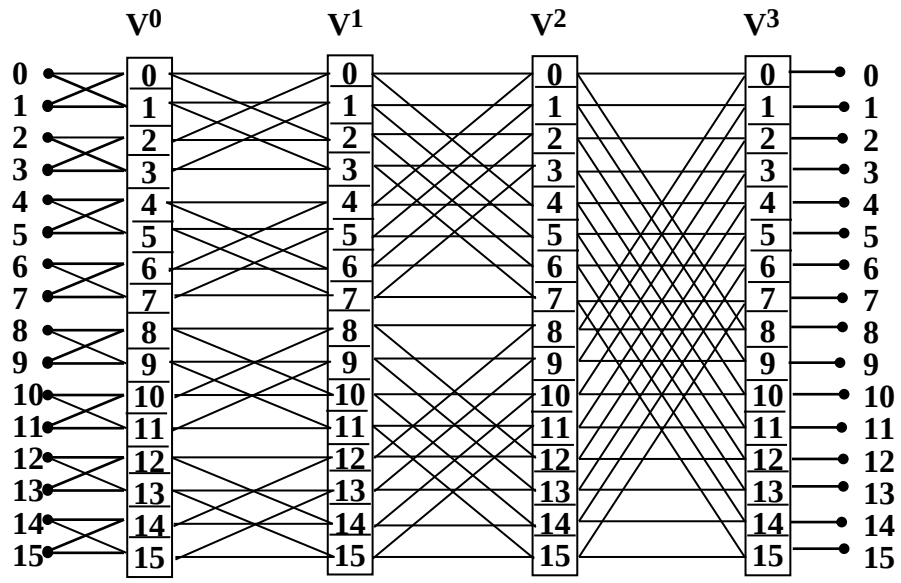


Рис. 3.4. Граф топологической модели алгоритма БПФ
«с прореживанием по времени»

Для быстрых нейронных сетей вместо базовой операции используется термин *нейронное ядро*. Нетрудно проверить, что для обеих выше приведенных топологий выполняется $V^{m-1} = U^m$, такие топологии будем называть *компактными.* Для некомпактных топологий для межслойного перехода существует нетривиальная подстановка q^m такая, что $V^{m-1} = q^m\left(U^m\right)$. Некомпактные топологии могут быть полезны при аппаратной реализации быстрых преобразований. Для программной реализации в подавляющем числе случаев достаточно использовать компактные топологии.

Из топологической модели (3.1) для терминальных слоев получим:

$$U^0 = \langle u_{n-1}u_{n-2}\cdots u_1u_0\rangle, \qquad V^{n-1} = \langle v_{n-1}v_{n-2}\cdots v_1v_0\rangle.$$

Если N – размерность рецепторного поля, а M – размерность аксонового поля сети, то из последних выражений непосредственно следует:

$$N = p_{n-1}\ldots p_1p_0, \quad M = g_{n-1}\ldots g_1g_0.$$

Таким образом, размерности терминальных полей сети определяются произведением размерностей нейронных модулей. Рассмотренные выше примеры самоподобных сетей были построены для размерностей $p_i = g_i = 2$. На рис. 3.5 для сравнения показан граф топологии «с прореживанием по частоте» для размерностей:

$$\begin{pmatrix} p_0 & p_1 & p_2 \\ 3 & 2 & 2 \end{pmatrix}, \qquad \begin{pmatrix} g_0 & g_1 & g_2 \\ 2 & 2 & 2 \end{pmatrix}.$$

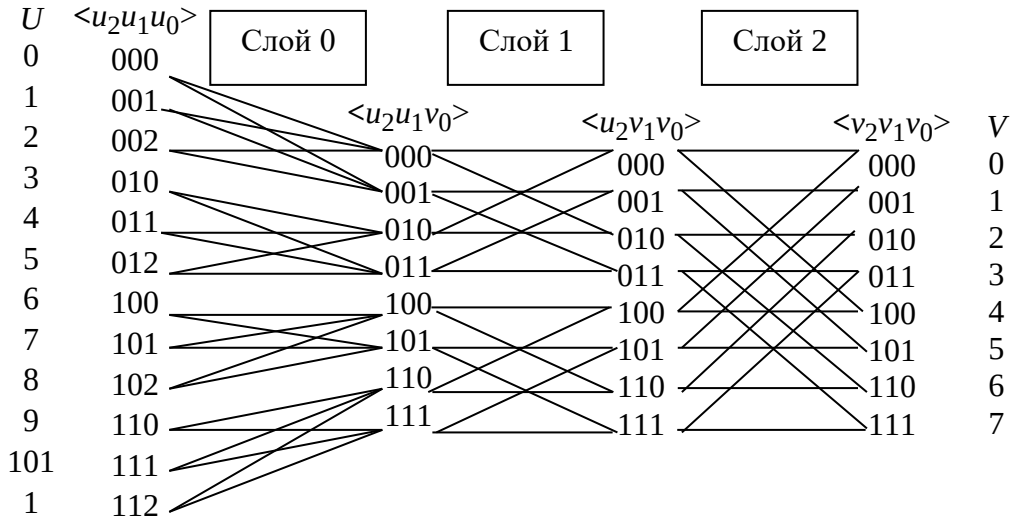


Рис. 3.5. Топологический граф быстрого преобразования размерности 12×8

Топологические предложения для данного примера имеют вид:

$$\begin{aligned} Rp &= \{U^m\} = \left[\langle u_2u_1u_0\rangle \langle u_2u_1v_0\rangle \langle u_2v_1v_0\rangle\right], \\ Ax &= \{V^m\} = \left[\langle u_2u_1v_0\rangle \langle u_2v_1v_0\rangle \langle v_2v_1v_0\rangle\right], \\ \{z^m\} &= \left[\langle u_2u_1\rangle \langle u_2v_0\rangle \langle v_1v_0\rangle\right]. \end{aligned} \qquad (3.3)$$

Данный граф является примером топологической модели быстрой нейронной сети. Структурная модель для данной сети показана на рис. 3.2. Для этой структурной модели существует множество других вариантов построения топологических моделей, любые взаимно-однозначные отображения $\{u_{z^m}\} \to U^m$, $\{v_{z^m}\} \to V^m$ допустимы, но далеко не все из них представимы в виде поразрядных кортежей.

Топологические модели, которые можно описать поразрядными кортежами, естественно считать *топологически-регулярными.* Рассмотрим еще несколько типов регулярных топологий. На рис. 3.6 показан граф преобразования с топологией Гуда [3]. На графе явным образом выделены базовые операции. Работа Гуда по быстрому преобразованию Фурье появилась на семь лет раньше хорошо известной работ Кули и Тьюки [2], но прошла незамеченной.

Топологическая схема описывается моделью

$$\begin{aligned} U^m &= \langle u_m u_{m+1} \ldots u_{n-1} v_0 v_1 \ldots v_{m-1}\rangle, \\ V^m &= \langle u_{m+1} u_{m+2} \ldots u_{n-1} v_0 v_1 \ldots v_m\rangle, \\ i^m &= \langle u_{m+1} u_{m+2} \ldots u_{n-1} v_0 v_1 \ldots v_{m-1}\rangle. \end{aligned}$$

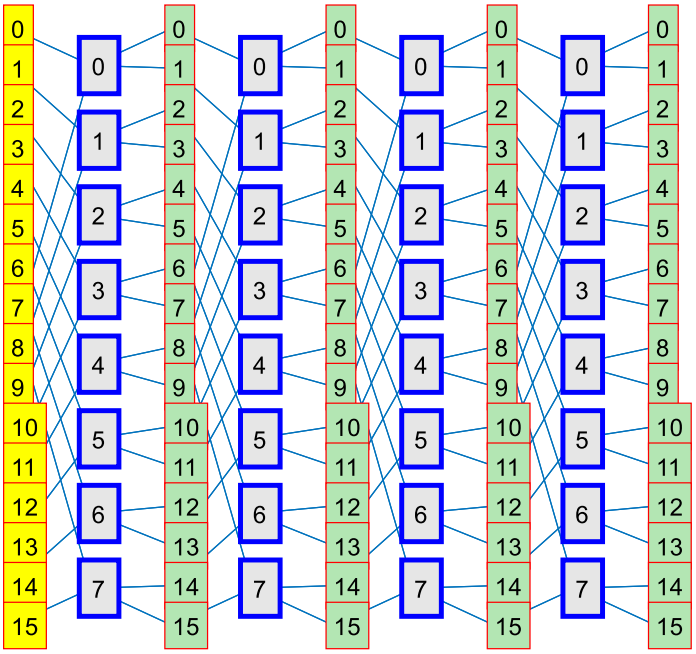


Рис. 3.6. Граф быстрого преобразования с топологией Гуда

Особенность схемы Гуда состоит в том, что все слои графа имеют одинаковый топологический образ. Однако выходные операнды на каждой базовой операции не могут замещать входные отсчеты, поэтому требуется дополнительная память для хранения выходного вектора слоя. Рассмотренные выше топологические схемы Кули–Тьюки с «прореживанием по времени» и по «частоте» являются двойственными друг другу и могут быть получены зеркальным отражением относительно вертикальной оси, что соответствует транспонированию факторизованного представления топологических матриц (см. 3.2.1). Алгоритмически переход к двойственной схеме выполняется заменой переменных в топологической модели по правилам: $U^m \leftrightarrow V^{n-1-m}$, $u_m \leftrightarrow v_{n-1-m}$. Двойственная топология Гуда описывается моделью:

$$U^m = \langle v_{m-1} v_{m-2} \ldots v_0 u_{n-1} u_{n-2} \ldots u_m \rangle,$$

$$V^m = \langle v_m v_{m-1} \ldots v_0 u_{n-1} u_{n-2} \ldots u_{m+1} \rangle,$$

$$i^m = \langle v_{m-1} \ldots v_0 u_{n-1} u_{n-2} \ldots u_{m+1} \rangle.$$

На рис. 3.7 показан граф быстрого преобразования с двойственной топологией Гуда.

Алгоритм БПФ для схем Кули–Тьюки по основанию 2 на выходе имеет спектр, гармоники которого имеют двоично-инверсную упорядоченность по частоте. Следующая топологическая схема позволяет получить на выходе спектр с естественным упорядочением по частоте при произвольных основаниях:

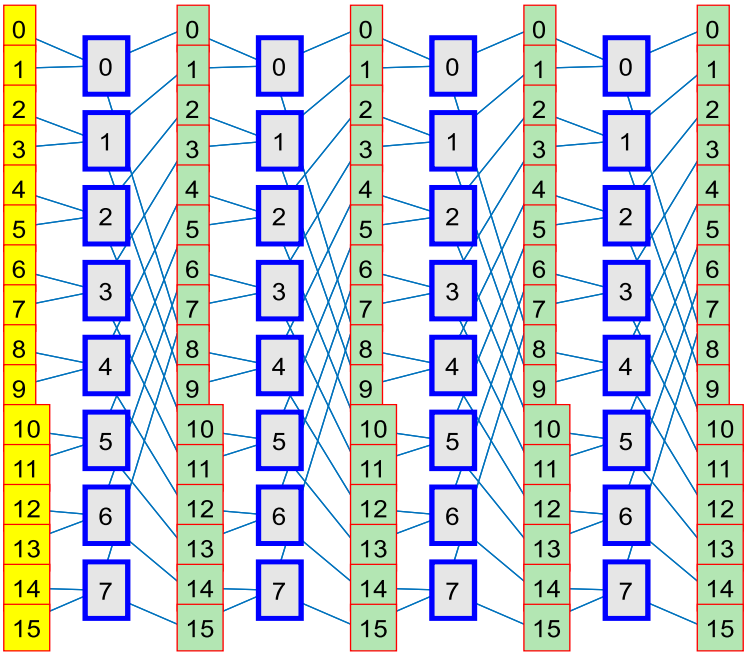


Рис. 3.7. Граф быстрого преобразования
с двойственной топологией Гуда

$$U^m = \langle u_m u_{m+1} \cdots u_{n-2} u_{n-1} v_{m-1} v_{m-2} \cdots v_1 v_0 \rangle,$$

$$V^m = \langle u_{m+1} u_{m+2} \cdots u_{n-1} v_m v_{m-1} v_{m-2} \cdots v_1 v_0 \rangle,$$

$$z^m = \langle u_{m+1} \cdots u_{n-2} u_{n-1} v_{m-1} v_{m-2} \cdots v_1 v_0 \rangle.$$

На рис. 3.8 показан граф быстрого преобразования, построенный по данной топологии.

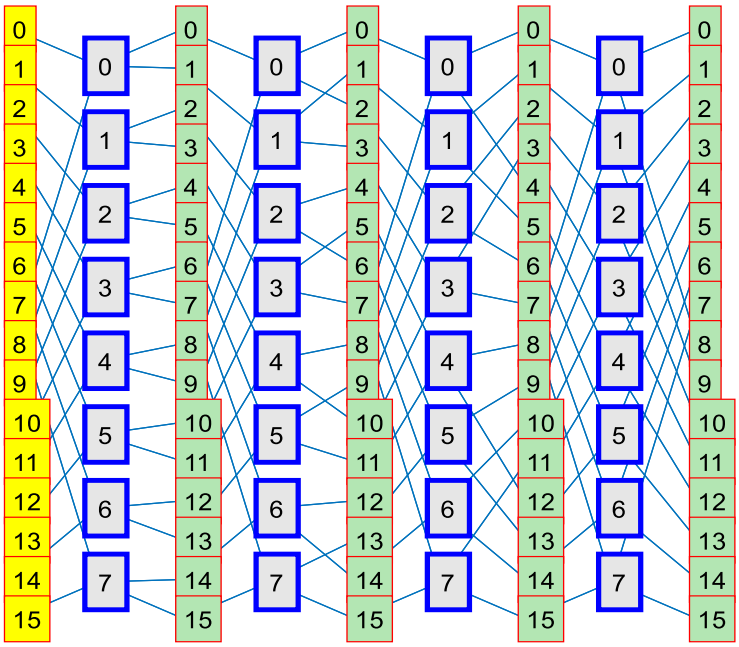


Рис. 3.8. Граф алгоритма БПФ с естественным
упорядочением спектра по частотам следования

Двойственная топологическая модель описывается выражениями:

$$U^m = \langle v_{m-1} v_{m-2} \cdots v_1 v_0 u_m u_{m+1} \cdots u_{n-2} u_{n-1} \rangle,$$

$$V^m = \langle v_m v_{m-1} v_{m-2} \cdots v_1 v_0 u_{m+1} \cdots u_{n-2} u_{n-1} \rangle,$$

$$z^m = \langle v_{m-1} v_{m-2} \cdots v_1 v_0 u_{m+1} \cdots u_{n-2} u_{n-1} \rangle.$$

Граф алгоритма БПФ с естественным упорядочением спектра по частотам и двойственной топологией показан на рис. 3.9.

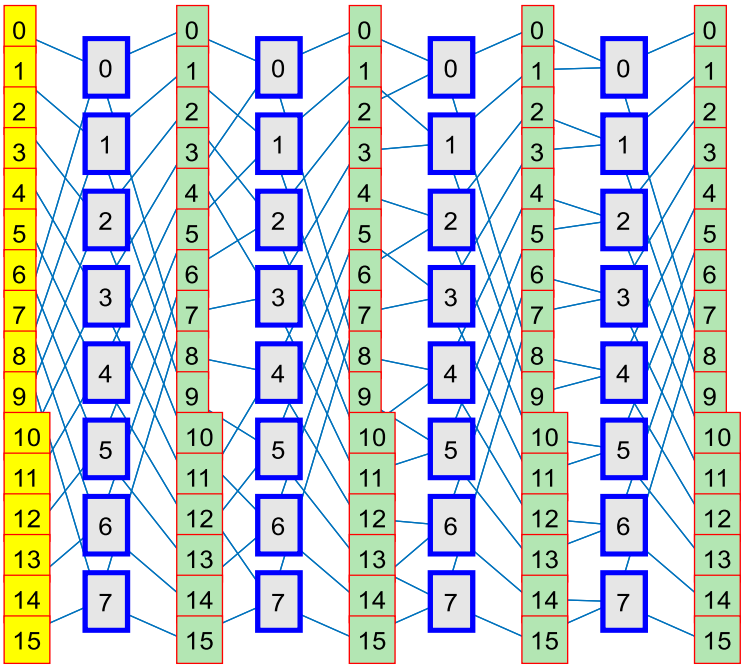


Рис. 3.9. Граф алгоритма БПФ с естественным упорядочением спектра по частотам следования и двойственной топологией

Для быстрого преобразования Хаара используется следующая топологическая схема [22]:

$$U^m = \langle u_{n-1} u_{n-2} \cdots u_{m+1} u_m v_0 v_1 \cdots v_{m-2} v_{m-1} \rangle,$$

$$V^m = \langle u_{n-1} u_{n-2} \cdots u_{m+1} v_0 v_1 \cdots v_{m-2} v_{m-1} v_m \rangle,$$

$$z^m = \langle u_{n-1} u_{n-2} \cdots u_{m+1} v_0 v_1 \cdots v_{m-2} v_{m-1} \rangle.$$

Граф топологической схемы показан на рис. 3.10.

Двойственна топологическая модель описывается выражениями:

$$U^m = \langle v_0 v_1 \cdots v_{m-2} v_{m-1} u_{n-1} u_{n-2} \cdots u_{m+1} u_m \rangle,$$

$$V^m = \langle v_0 v_1 \cdots v_{m-2} v_{m-1} v_m u_{n-1} u_{n-2} \cdots u_{m+1} \rangle,$$

$$z^m = \langle v_0 v_1 \cdots v_{m-2} v_{m-1} u_{n-1} u_{n-2} \cdots u_{m+1} \rangle.$$

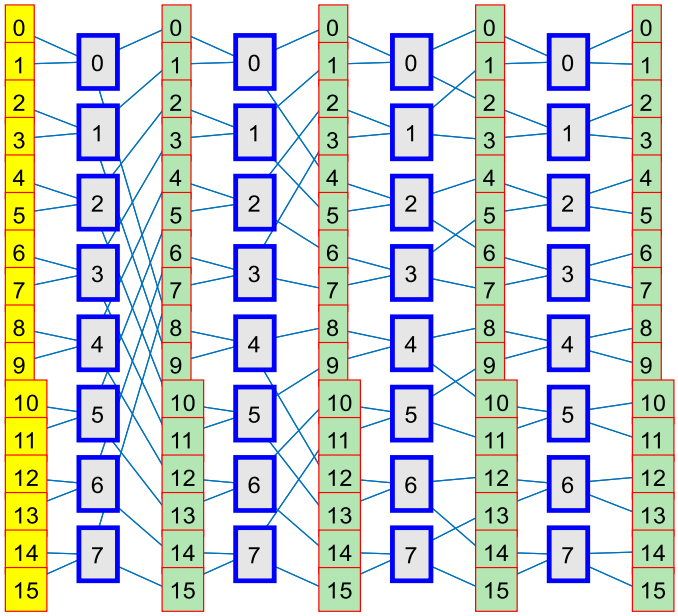


Рис. 3.10. Топологический граф преобразования Хаара

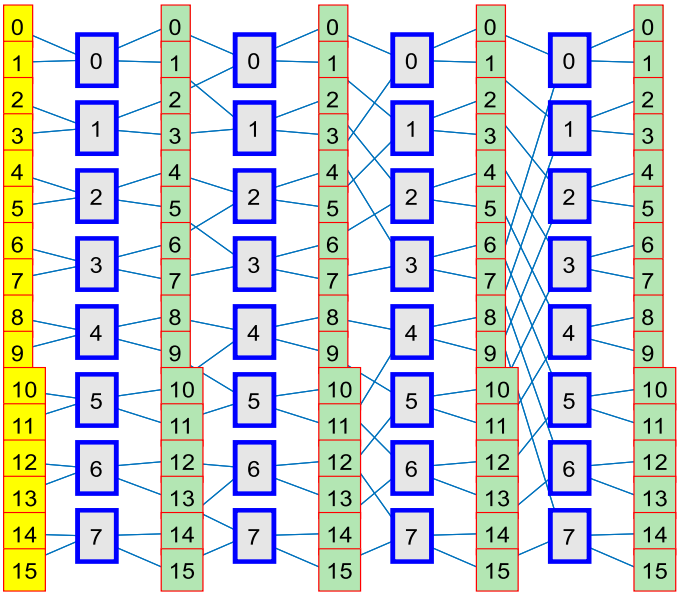


Рис. 3.11. Граф топологической схемы быстрого
преобразования Хаара с двойственной топологией

Топологический граф двойственной модели показан на рис. 3.11.

3.2.1. Топологические матрицы быстрых преобразований

К классу регулярных топологических моделей относятся графы БПФ и других быстрых преобразований. В описаниях алгоритмов быстрых спектральных преобразований вместо графа алгоритма часто используется матричная форма, когда алгоритм представляется в виде произведения слабозаполненных матриц, где каждая матрица соответствует одному слою преобразования:

$$H = H_0 H_1 \ldots H_{n-1}.$$

Такую форму называют *факторизованным представлением* быстрого алгоритма. Структура и параметры слабозаполненных матриц при этом определяются некоторой теоремой факторизации спектрального преобразования. Таких теорем придумано великое множество, БПФ с топологиями Кули–Тьюки являются типичным примером. Для каждого нового алгоритма быстрого преобразования до недавнего времени доказывали собственную теорему факторизации. Покажем, что структура слабозаполненных матриц непосредственно следует из топологической модели одноранговой самоподобной сети, и поэтому описанное выше многообразие топологических моделей покрывает все возможные реализации быстрых алгоритмов, что исключает необходимость какого-либо доказательства новых теорем факторизации.

Для матричного представления алгоритма быстрого преобразования введем топологические матрицы T_m, состоящие из нулей и единиц, обозначив их элементы $T_m\left(U^m, V^m\right)$, где U^m – номер строки матрицы, а V^m – номер столбца. Топологическая матрица по структуре подобна слабозаполненной матрице H_m и отличается от нее только тем, что в позициях ненулевых элементов размещаются единицы. Для каждой матрицы можно задать свои правила нумерации строк и столбцов, для регулярных моделей удобно использовать поразрядные представления вида:

$$U^m = \left\langle U^m_{n-1} U^m_{n-2} \ldots U^m_0 \right\rangle, \quad V^m = \left\langle V^m_{n-1} V^m_{n-2} \ldots V^m_0 \right\rangle,$$

которые назовем *внешними* в отличие от *внутренних*, заданных отображениями вида (3.1). Соответствие между внешними разрядными переменными и внутренними можно задать таблицей, например, для топологии (3.2) таблица будет иметь следующий вид (табл. 3.1).

Таблица 3.1

Соответствия между внешними и внутренними разрядными переменными

$U^m =$	u_{n-1}	u_{n-2}	...	u_{m+1}	u_m	v_{m-1}	v_{m-2}	...	v_1	v_0
$U^m =$	U^m_{n-1}	U^m_{n-2}	...	U^m_{m+1}	U^m_m	U^m_{m-1}	U^m_{m-2}	...	U^m_1	U^m_0
$V^m =$	u_{n-1}	u_{n-2}	...	u_{m+1}	v_m	v_{m-1}	v_{m-2}	...	v_1	v_0
$V^m =$	V^m_{n-1}	V^m_{n-2}	...	V^m_{m+1}	V^m_m	V^m_{m-1}	V^m_{m-2}	...	V^m_1	V^m_0

Единицам топологической матрицы T_m слоя m в слабозаполненной матрице соответствуют элементы ядер (базовых операций) с номерами:

$$z^m = \left\langle u_{n-1} u_{n-2} \cdots u_{m+1} v_{m-1} v_{m-2} \cdots v_1 v_0 \right\rangle.$$

Это означает, что для каждого единичного элемента топологической матрицы $T_m\left(U^m, V^m\right)$ пара разрядных переменных U_l^m, V_l^m из внешних представлений должны соответствовать одной из разрядных переменных внутреннего представления ядра z^m. Следуя данному правилу и используя таблицу соответствий, можно записать аналитическое представление структуры топологической матрицы в виде:

$$T_m\left(U^m, V^m\right) = \delta\left(U_{n-1}^m, V_{n-1}^m\right)\delta\left(U_{n-2}^m, V_{n-2}^m\right)\ldots$$
$$\ldots\delta\left(U_{m+1}^m, V_{m+1}^m\right)\delta\left(U_{m-1}^m, V_{m-1}^m\right)\ldots\delta\left(U_0^m, V_0^m\right),$$

где $\delta(\bullet)$ – дельта-функция Кронекера. Номер ядра также может быть выражен в разрядных переменных внешнего представления:

$$z^m = \left\langle U_{n-1}^m U_{n-2}^m \ldots U_{m+1}^m V_{m-1}^m V_{m-2}^m \ldots V_1^m V_0^m \right\rangle.$$

Для графа, показанного на рис. 3.5, топологические матрицы и номера ядер будут определяться следующим выражениями:

$$T_0\left(U^0, V^0\right) = \delta\left(U_2^0, V_2^0\right)\delta\left(U_1^0, V_1^0\right), \qquad z^0 = \left\langle U_2^0 U_1^0 \right\rangle,$$
$$T_1\left(U^1, V^1\right) = \delta\left(U_2^1, V_2^1\right)\delta\left(U_0^1, V_0^1\right), \qquad z^1 = \left\langle U_2^1 V_0^1 \right\rangle,$$
$$T_2\left(U^2, V^2\right) = \delta\left(U_1^2, V_1^2\right)\delta\left(U_0^2, V_0^2\right) \qquad z^2 = \left\langle V_1^2 V_0^2 \right\rangle.$$

V_2^0	0	0	0	0	1	1	1	1
V_1^0	0	0	1	1	0	0	1	1
V_0^0	0	1	0	1	0	1	0	1
$U_2^0U_1^0U_0^0$								
000	1	1						
001	1	1						
002	1	1						
010			1	1				
011			1	1				
012			1	1				
100					1	1		
101					1	1		
102					1	1		
110							1	1
111							1	1
112							1	1

•

V_2^1	0	0	0	0	1	1	1	1
V_1^1	0	0	1	1	0	0	1	1
V_0^1	0	1	0	1	0	1	0	1
$U_2^1U_1^1U_0^1$								
000	1		1					
001		1		1				
010	1		1					
011		1		1				
100					1		1	
101						1		1
110					1		1	
111						1		1

•

V_2^2	0	0	0	0	1	1	1	1
V_1^2	0	0	1	1	0	0	1	1
V_0^2	0	1	0	1	0	1	0	1
$U_2^2U_1^2U_0^2$								
000	$\hat{0}$				$\hat{0}$			
001		$\hat{1}$				$\hat{1}$		
010			$\hat{2}$				$\hat{2}$	
011				$\hat{3}$				$\hat{3}$
100	$\hat{0}$				$\hat{0}$			
101		$\hat{1}$				$\hat{1}$		
110			$\hat{2}$				$\hat{2}$	
111				$\hat{3}$				$\hat{3}$

Рис. 3.12. Матричное представление топологии самоподобной сети

На рис. 3.12 показано матричное представление топологии для данной сети. В последней матрице вместо единичных элементов для наглядности показаны номера ядер.

3.3. Параметрические модели быстрых преобразований

В графе быстрого алгоритма модулями являются базовые операции (нейронные ядра), представимые в виде матриц небольшой размерности. Для алгоритма быстрого преобразования в базовой операции z^m слоя m выполняется линейная обработка компонентов входного вектора слоя:

$$y^m_{z^m}(v_m) = \sum_{u_m} x^m_{z^m}(u_m)\, w^m_{z^m}(u_m, v_m),$$

где $x^m_{z^m}$ и $y^m_{z^m}$ – координаты входного и выходного векторов базовой операции; $w^m_{z^m}$ – матрица весов базовой операции.

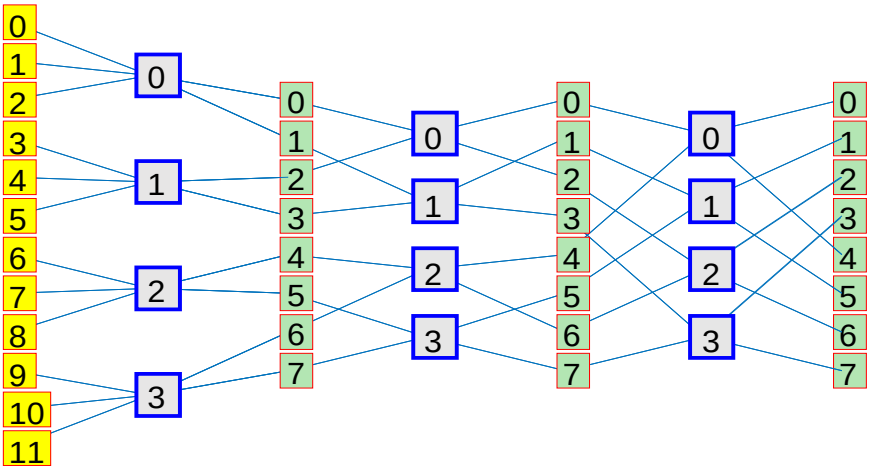


Рис. 3.13. Граф топологической модели с выделенными базовыми операциями

Для построения алгоритма преобразования необходимо перейти от локальных переменных ядра к внешним переменным слоя. Этот переход реализуется на основе топологической модели. Параметрическое описание базовых операций вместе с топологической моделью образует параметрическую модель быстрого преобразования. На рис. 3.13 показан граф топологической модели (3.3) с выделенными базовыми операциями.

3.4. Топологические модели двумерных быстрых преобразований

Обозначим через $F(U_y, U_x)$ матрицу изображения размерностью $N_y \times N_x$. При воздействии на изображение линейного преобразования $H(U_y, U_x; V_y, V_x)$ получается массив из $M_y \times M_x$ коэффициентов. Двумерное преобразование выполняется по правилу:

$$S\left(V_y,V_x\right)=\sum_{Uy=0}^{Ny-1}\sum_{Ux=0}^{Nx-1}F\left(U_y,U_x\right)H\left(U_y,U_x;V_y,V_x\right).$$

Необходимым условием существования быстрого алгоритма является возможность мультипликативной декомпозиции значений входных и выходных размерностей преобразования в равное число сомножителей:

$$\begin{aligned} N_y &= p_0^y p_1^y \dots p_{n-1}^y, \qquad M_y = g_0^y g_1^y \dots g_{n-1}^y, \\ N_x &= p_0^x p_1^x \dots p_{n-1}^x, \qquad M_x = g_0^x g_1^x \dots g_{n-1}^x. \end{aligned} \tag{3.4}$$

Индексы x, y здесь означают принадлежность к осям координат исходного изображения, а значение n определяет число слоев в графе быстрого алгоритма. Для быстрых регулярных преобразований общего вида данное условие не является жестким ограничением, поскольку некоторые сомножители могут быть единичными. Тем не менее, чем больше число не единичных сомножителей, тем выше вычислительная эффективность алгоритма преобразования. Используя сомножители декомпозиций (3.4), координаты точек изображения представим в позиционной системе счисление со смешанными основаниями:

$$\begin{aligned} U_y &= \left\langle u_{n-1}^y u_{n-2}^y \dots u_1^y u_0^y \right\rangle, \\ U_x &= \left\langle u_{n-1}^x u_{n-2}^x \dots u_1^x u_0^x \right\rangle, \end{aligned} \tag{3.5}$$

где вес m-го разряда определяется выражением $p_{m-1}^* p_{m-2}^* \dots p_1^* p_0^*$, а u_m^* является разрядной переменной, принимающей значения $\left[0, p_m^* - 1\right]$ (звездочка здесь заменяет индексы x, y). Аналогично можно представить координаты коэффициентов в плоскости $\left[V_y, V_x\right]$:

$$\begin{aligned} V_y &= \left\langle v_{n-1}^y v_{n-2}^y \cdots v_1^y v_0^y \right\rangle, \\ V_x &= \left\langle v_{n-1}^x v_{n-2}^x \cdots v_1^x v_0^x \right\rangle, \end{aligned} \tag{3.6}$$

где вес m-го разряда определяется выражением $g_{m-1}^* g_{m-2}^* \dots g_1^* g_0^*$, а v_m^* является разрядной переменной, принимающей значения $\left[0, g_m^* - 1\right]$.

Алгоритм быстрого преобразования обычно представляется в виде графа с топологией различного вида. Поразрядную форму чисел удобно использовать для аналитического описания графа топологии быстрого алгоритма.

Например, для топологии Кули–Тьюки «с прореживанием по времени» граф может быть описан в виде лингвистического предложения [6] (топологической модели):

$$\left[\left\langle u^*_{n-1}u^*_{n-2}\dots u^*_1u^*_0\right\rangle\left\langle u^*_{n-1}u^*_{n-2}\dots u^*_1v^*_0\right\rangle\dots\left\langle u^*_{n-1}u^*_{n-2}\dots u^*_{m+1}u^*_mv^*_{m-1}v^*_{m-2}\dots v^*_0\right\rangle\dots\right.$$
$$\left.\dots\left\langle v^*_{n-1}v^*_{n-2}\dots v^*_1v^*_0\right\rangle\right],$$

где словами являются поразрядные представления координатных чисел, а буквами – имена разрядных переменных. Число слов в предложении равно $n+1$. Первое и последнее слова в предложении соответствуют координатам точек терминальных плоскостей (см. выражения (3.5) и (3.6)). Промежуточные слова определяют координаты U^m_y, U^m_x и V^m_y, V^m_x в плоскостях внутренних слоев быстрого алгоритма. Для алгоритма с замещением значений выполняется условие:

$$U^{m+1}_y = V^m_y, \qquad U^{m+1}_x = V^m_x.$$

Граф топологии в слое m содержит базовые операции (нейронные ядра) $W^m_{z^m_x, z^m_y}\left(u^y_m u^x_m; v^y_m v^x_m\right)$, представляющие собой четырехмерные матрицы размерности $\left[p^y_m, p^x_m; g^y_m, g^x_m\right]$, где поразрядные выражения индексов ядер слоя m для выбранной топологии имеют вид:

$$\begin{aligned} z^m_x &= \left\langle u^x_{n-1}u^x_{n-2}\dots u^x_{m+1}v^x_{m-1}v^x_{m-2}\dots v^x_0\right\rangle, \\ z^m_y &= \left\langle u^y_{n-1}u^y_{n-2}\dots u^y_{m+1}v^y_{m-1}v^y_{m-2}\dots v^y_0\right\rangle. \end{aligned} \tag{3.7}$$

Выражение (3.7) является аналитическим представлением системного инварианта быстрых алгоритмов [10]. В общем случае топологии для направлений x и y могут быть различными. Взаимосвязь между базовыми операциями определяется фактор-графом (структурной моделью быстрого преобразования), где каждой вершине соответствует нейронное ядро. Для выбранной топологии граф структурной модели описывается лингвистическим предложением:

$$\left[\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_1\right\rangle\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_2v^*_0\right\rangle\ldots\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_{m+1}v^*_{m-1}v^*_{m-2}\ldots v^*_0\right\rangle\ldots\right.$$

$$\left.\ldots\left\langle v^*_{n-1}v^*_{n-2}\ldots v^*_1v^*_0\right\rangle\right].$$

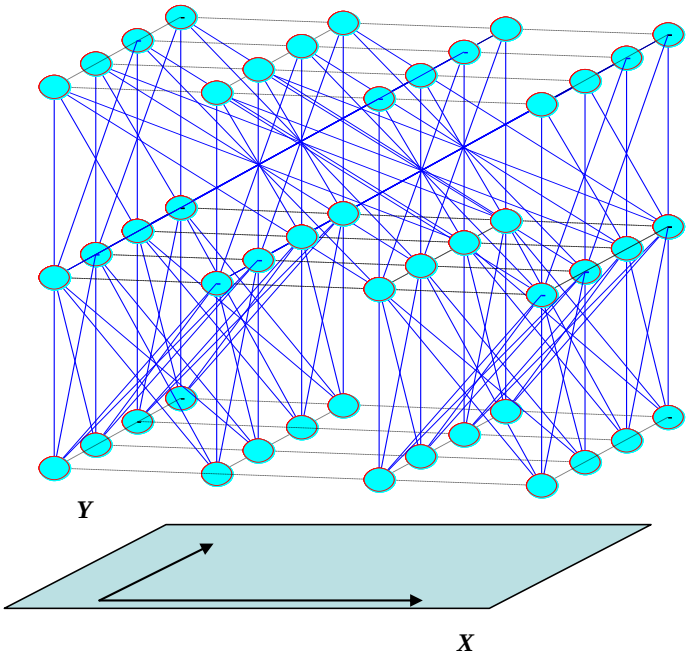


Рис. 3.14. Структурная модель двумерного быстрого преобразования

Каждое слово в этом предложении определяет номер базовой операции z^m_* в слое m. Число слов в предложении равно n. На рис. 3.14 показана структурная модель быстрого двумерного преобразования для размерности изображения 8×8. Входное изображение подается на нижний слой, а спектральные коэффициенты получаются в верхнем слое. Вершинам модели соответствуют базовые операции (нейронные ядра) размерности [2,2; 2,2]. Ядро в слое m выполняет двумерное преобразование над пространственным блоком размером $p^y_m\times p^x_m$:

$$S^m\left(V^m_y,V^m_x\right)=\sum_{u^y_m}\sum_{u^x_m}F^m\left(U^m_y,U^m_x\right)W^m_{i^m_x,i^m_y}\left(u^y_mu^x_m;v^y_mv^x_m\right).$$

Соответствия $U^m_*\leftrightarrow\left(z^m_*,u^m_*\right)$ взаимно однозначно определяются по лингвистическим предложениям топологической модели.

4. НАСТРОЙКА БЫСТРЫХ ПЕРЕСТРАИВАЕМЫХ ПРЕОБРАЗОВАНИЙ

В предыдущих разделах была рассмотрена стратифицированная модель модульных самоподобных нейронных сетей, где модулями являются нейронные ядра. В простейшем варианте на параметрическом уровне нейронные ядра можно задать линейными матрицами небольшой размерности. Переход от одного преобразования к другому происходит за счет изменения параметров нейронных ядер. В этом случае модульная самоподобная нейронная сеть является представлением линейного перестраиваемого преобразования, обладающего быстрым алгоритмом. Поэтому можно поставить задачу разработать метод обучения этого класса нейронных сетей к известным типам быстрых преобразований, подобных алгоритму БПФ. Исторически процедура обучения перестраиваемого преобразования называется настройкой на заданную систему функций. В данной главе будет показано, что предложенный метод настройки покрывает известные виды алгоритмов быстрых спектральных преобразований и может быть также использован для порождения фрактальных последовательностей. Метод основан на теореме о мультипликативной факторизации элементов матрицы быстрых преобразований, с которой мы и начнем.

4.1. Параметрическая факторизация элементов матриц быстрых преобразований

Обозначим через $H = H_0 H_1 \ldots H_{n-1}$ матрицу быстрого преобразования, полученную как результат произведения слабозаполненных матриц слоев (см. 3.2.1). Обозначим через $h(U,V)$ элементы этой матрицы, где $U = U^0$ и $V = V^{n-1}$ – номера рецепторов и аксонов терминальных слоев. Линейное преобразование H переводит входной вектор $x = x^0$ в выходной вектор $y = y^{n-1}$. Для линейного преобразования элементы матрицы H можно выразить как частные производные:

$$h(U,V) = \frac{\partial y^{n-1}(V^{n-1})}{\partial x^0(U^0)}. \qquad (4.1)$$

Дифференцируя (4.1) по правилу дифференцирования сложной функции, получим:

$$h(U,V) = \frac{\partial y^{n-1}(V^{n-1})}{\partial x^0(U^0)} = \frac{\partial y^{n-1}(V^{n-1})}{\partial x^{n-1}(U^{n-1})} \frac{\partial x^{n-1}(U^{n-1})}{\partial y^{n-2}(V^{n-2})} \cdots \frac{\partial x^1(U^1)}{\partial y^0(V^0)} \frac{\partial y^0(V^0)}{\partial x^0(U^0)}. \qquad (4.2)$$

Поскольку для межслойного перехода выполнено $x^{m+1}\left(U^{m+1}\right)=y^{m}\left(V^{m}\right)$, то для всех m

$$\frac{\partial x^{m+1}\left(U^{m+1}\right)}{\partial y^{m}\left(V^{m}\right)}=1,$$

а из выражения (4.1) следует:

$$\frac{\partial y^{m}\left(V^{m}\right)}{\partial x^{m}\left(U^{m}\right)}=w_{z^{m}}^{m}\left(u_{m},v_{m}\right).$$

Подставляя значения частных производных в (4.2) получим:

$$h\left(U,V\right)=w_{z^{n-1}}^{n-1}\left(u_{n-1},v_{n-1}\right)w_{z^{n-2}}^{n-2}\left(u_{n-2},v_{n-2}\right)\ldots w_{z^{0}}^{0}\left(u_{0},v_{0}\right). \qquad (4.3)$$

Таким образом, элементы матрицы быстрого преобразования представимы в виде произведения элементов ядер. Следует отметить, что еще в 1958 г. в работе Гуда [3] впервые было показано, что элементы матрицы быстрого преобразования Фурье можно представить в виде произведения элементов матриц, базовых операций, которые также являются преобразованиями Фурье, но малых размерностей.

Следствие 1. Обозначим через $H^{m}=H_{m}H_{m+1}\ldots H_{n-1}$ неполную матрицу быстрого преобразования, полученную как результат произведения слабозаполненных матриц, начиная со слоя m и заканчивая последним слоем $n-1$. Элементы матрицы H^{m} можно выразить как частные производные:

$$h^{m}\left(U^{m},V\right)=\frac{\partial y^{n-1}\left(V^{n-1}\right)}{\partial x^{m}\left(U^{0}\right)}.$$

Повторяя вывод по аналогии с полным преобразованием, придем к результату:

$$h^{m}\left(U^{m},V\right)=w_{z^{n-1}}^{n-1}\left(u_{n-1},v_{n-1}\right)w_{z^{n-2}}^{n-2}\left(u_{n-2},v_{n-2}\right)\ldots$$
$$\ldots w_{z^{m+1}}^{m+1}\left(u_{m+1},v_{m+1}\right)w_{z^{m}}^{m}\left(u_{m},v_{m}\right).$$

4.2. Настройка на базис Адамара

Функции базиса Адамара задаются на интервале длиной $N=2^{n}$ следующим выражением:

$$\mathrm{had}\left(U,V\right)=\prod_{m=0}^{n-1}\left(-1\right)^{u_{m}v_{m}},$$

где $U=\langle u_{n-1}u_{n-2}\ldots u_0\rangle$ и $V=\langle v_{n-1}v_{n-2}\ldots v_0\rangle$. Все разрядные числа принимают значения $\{0,1\}$. Сравнивая (4.3) с определением функций Адамара, непосредственно получим:

$$w_{z^m}^{m}(u_m,v_m)=(-1)^{u_m v_m}.$$

Этому выражению соответствует матрица ядра $W=\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$, очевидно, что все ядра преобразования одинаковы. Выберем для определенности топологию Кули–Тьюки «с прореживанием по времени»:

$$U^m=\langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m v_{m-1}v_{m-2}\ldots v_1v_0\rangle,$$
$$V^m=\langle u_{n-1}u_{n-2}\ldots u_{m+1}v_m v_{m-1}v_{m-2}\ldots v_1v_0\rangle,$$
$$z^m=\langle u_{n-1}u_{n-2}\ldots u_{m+1}v_{m-1}v_{m-2}\ldots v_1v_0\rangle.$$

На рис. 4.1 показано матричное представление быстрого алгоритма преобразования Адамара для размерности $N=2^3$.

		V_2^m	0	0	0	0	1	1	1	1
		V_1^m	0	0	1	1	0	0	1	1
		V_0^m	0	1	0	1	0	1	0	1
U_2^m	U_1^m	U_0^m								
0	0	0	1	1						
0	0	1	1	-1						
0	1	0			1	1				
0	1	1			1	-1				
1	0	0					1	1		
1	0	1					1	-1		
1	1	0							1	1
1	1	1							1	-1

×

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
1		1					
	1		1				
1		-1					
	1		-1				
				1		1	
					1		1
				1		-1	
					1		-1

×

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
1				1			
	1				1		
		1				1	
			1				1
1				-1			
	1				-1		
		1				-1	
			1				-1

Рис. 4.1. Матричное представление быстрого преобразования Адамара

Перемножив матрицы факторизованного представления, получим следующую матрицу базисных функций:

$$H=\begin{vmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1\\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1\\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1\\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1\\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1\\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{vmatrix}$$

4.3. Аппроксимация фракталов

Рассмотрим быстрое преобразование размерности $N = 3^2$ и выберем все ядра преобразования в виде

$$W = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

Факторизованное представление алгоритма в той же топологии будет иметь вид (рис. 4.2).

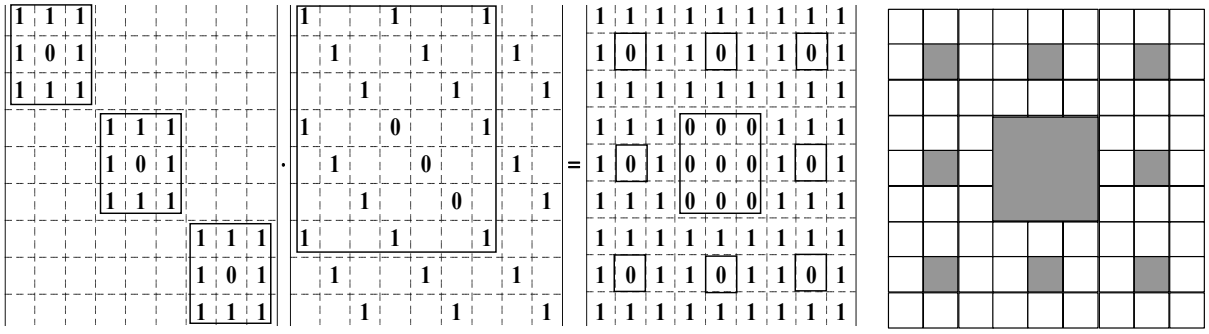


Рис. 4.2. Аппроксимация фрактала Кантора

Нетрудно видеть, что результирующая матрица представляет собой итерацию фрактала Кантора.

Рассмотрим быстрое преобразование размерности $N = 2^3$ и выберем все ядра преобразования в виде матрицы

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Факторизованное представление быстрого алгоритма показано на рис. 4.3. В этом варианте параметрической настройки результирующая матрица представляет собой итерацию фрактала Серпинского.

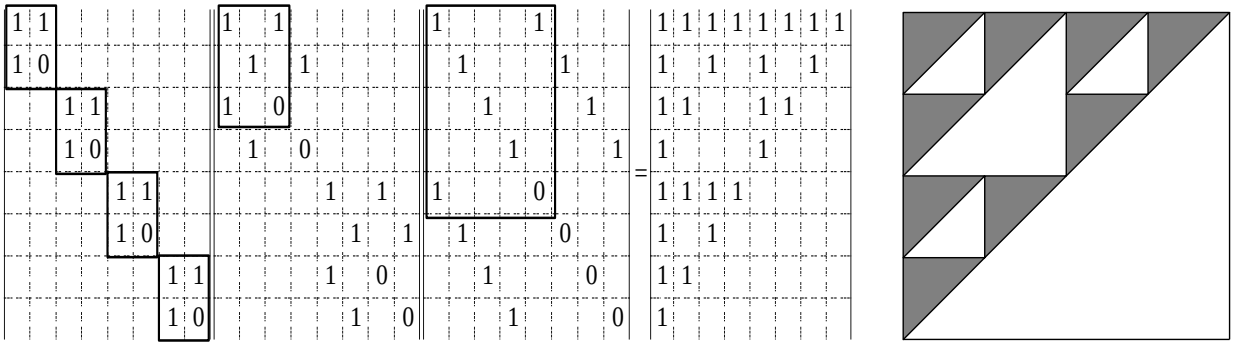


Рис. 4.3. Аппроксимация фрактала Серпинского

4.4. Настройка на базис Виленкина–Крестенсона

Обобщенная система ортогональных ненормированных функций Виленкина–Крестенсона в упорядочении Пэли [27] определяется выражениями:

$$vcf\left(U,V\right)=\prod_{m=0}^{n-1}\left(\omega_m\right)^{U_mV_m},\qquad \omega_m=e^{j\frac{2\pi}{p_m}},$$

где $U=\langle U_{n-1}U_{n-2}\ldots U_0\rangle$, $V=\langle V_0V_1\ldots V_{n-1}\rangle$, $U_m,V_m\in 0,1,\ldots p_m-1$, $\mathrm{j}=\sqrt{-1}$. Система функций задана на интервале длиной $N=p_0p_1\ldots p_{n-1}$. Поворачивающий множитель ω_m является комплексным числом, выраженным в экспоненциальной форме. Переменная U соответствует дискретному времени, а переменная V – порядковому номеру функции. Построим реализацию быстрого спектрального преобразования так, чтобы функции базиса располагались вдоль столбцов результирующей матрицы. Будем полагать, что топологическая модель быстрого преобразования задана выражением

$$U^m=V^{m-1}=\langle v_0v_1\ldots v_{m-2}v_{m-1}u_{n-1}u_{n-2}\ldots u_m\rangle.$$

Откуда для терминальных слоев будем иметь:

$$U=\langle U_{n-1}U_{n-2}\ldots U_0\rangle=\langle u_{n-1}u_{n-2}\ldots u_0\rangle,$$
$$V=\langle V_0V_1\ldots V_{n-1}\rangle=\langle v_0v_1\ldots v_{n-1}\rangle.$$

Сравнивая (4.3) с определением функций базиса, непосредственно получим

$$w_{i^m}^m\left(u_m,v_m\right)=\left(\omega_m\right)^{u_mv_m},\quad u_m=U_m,\quad v_m=V_m.$$

Очевидно, что в пределах слоя все ядра одинаковы.

Пример. Пусть $N=3^2$. Матрица преобразования факторизуется в произведение двух матриц: $H=H_0H_1$. Поразрядное представление строк и столбцов матрицы H можно записать в виде $U=\langle U_1U_0\rangle$, $V=\langle V_0V_1\rangle$. На основании 3.2.1 имеем, что слабо заполненные матрицы по слоям определяются выражениями:

Слой 0. $h_0\left(U^0,V^0\right)=w_{i^0}^0\left(U_0^0,V_0^0\right)\delta\left(U_1^0,V_1^0\right),$

Слой 1. $h_1\left(U^1,V^1\right)=w_{i^1}^1\left(U_0^1,V_1^1\right)\delta\left(U_1^1,V_0^1\right).$

Для рассматриваемого примера поворачивающие множители совпадают для обоих слоев, поэтому все ядра одинаковы и имеют вид:

$$W = \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{pmatrix},$$

где $\omega = e^{j\frac{2\pi}{3}}$.

На рис. 4.4 показано факторизованное представление построенного спектрального преобразования.

	V_0	0	0	0	1	1	1	2	2	2	0	0	0	1	1	1	2	2	2
	V_1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2
U_1	U_0																		
0	0	1			1			1			1	1	1						
0	1	1			ω			ω^2			1	ω	ω^2						
0	2	1			ω^2			ω			1	ω^2	ω						
1	0		1			1			1					1	1	1			
1	1		1			ω			ω^2					1	ω	ω^2			
1	2		1			ω^2			ω					1	ω^2	ω			
2	0			1			1			1							1	1	1
2	1			1			ω			ω^2							1	ω	ω^2
2	2			1			ω^2			ω							1	ω^2	ω

Рис. 4.4. Факторизованная форма преобразования Виленкина–Крестенсона

Перемножив матрицы факторизованного представления, получим следующую матрицу базисных функций:

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & \omega & \omega & \omega & \omega^2 & \omega^2 & \omega^2 \\ 1 & 1 & 1 & \omega^2 & \omega^2 & \omega^2 & \omega & \omega & \omega \\ 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 \\ 1 & \omega & \omega^2 & \omega & \omega^2 & 1 & \omega^2 & 1 & \omega \\ 1 & \omega & \omega^2 & \omega^2 & 1 & \omega & \omega^2 & 1 & \omega \\ 1 & \omega^2 & \omega & 1 & \omega^2 & \omega & 1 & \omega^2 & \omega \\ 1 & \omega^2 & \omega & \omega & 1 & \omega^2 & \omega^2 & \omega & 1 \\ 1 & \omega^2 & \omega & \omega^2 & \omega & 1 & \omega & 1 & \omega^2 \end{pmatrix}$$

Рис. 4.5. Функции базиса Виленкина–Кристенсона для $N = 9$

4.5. Настройка на базис Фурье

По определению, функции базиса Фурье задаются выражением

$$F_N(U,V)=\frac{1}{\sqrt{N}}\exp\left(-j\frac{2\pi}{N}UV\right), \tag{4.4}$$

где U – временной отсчет; V – частота (или номер) базисной функции; $j=\sqrt{-1}$; $N=p_0p_1\dots p_{\kappa-1}$ – размерность преобразования (для быстрых преобразований размерность всегда является составным числом). Выполним настройку быстрого алгоритма для базиса Фурье с естественным упорядочением функций по частотам следования. Для построения алгоритма воспользуемся топологической моделью:

$$U^m=\langle u_m u_{m+1}\dots u_{n-2}u_{n-1}v_{m-1}v_{m-2}\dots v_1v_0\rangle,$$
$$V^m=\langle u_{m+1}u_{m+2}\dots u_{n-1}v_m v_{m-1}v_{m-2}\dots v_1v_0\rangle,$$
$$z^m=\langle u_{m+1}\dots u_{n-2}u_{n-1}v_{m-1}v_{m-2}\dots v_1v_0\rangle.$$

Откуда для входного слоя следует:

$$U=U^0=\langle u_0u_1\dots u_{n-2}u_{n-1}\rangle=\sum_{m=0}^{n-1}u_m p_{m+1}p_{m+2}\cdots p_{n-1}\,.$$

Подставив последнее выражение в (4.4), получим:

$$F(U,V)=\prod_{m=0}^{n-1}\frac{1}{\sqrt{p_m}}\exp\left(-j\frac{2\pi V}{p_{n-1}p_{n-2}\cdots p_m}u_m\right). \tag{4.5}$$

Для выходного слоя из топологической схемы получим:

$$V=V^{n-1}=\langle v_{n-1}v_{n-2}v_{n-3}\dots v_1v_0\rangle.$$

Для любого m последнее выражение можно записать в виде:

$$V=\langle v_{n-1}v_{n-2}v_{n-3}\dots v_1v_0\rangle=\langle v_{n-1}v_{n-2}\dots v_{m+1}\rangle\, p_m p_{m-1}\cdots p_0+$$
$$+v_m p_{m-1}p_{m-2}\cdots p_0+\langle v_{m-1}v_{m-2}\dots v_0\rangle.$$

Подставляя в (4.5) и учитывая периодичность комплексной экспоненты по периоду 2π, после преобразований получим:

$$F(U,V)=\prod_{m=0}^{n-1}\frac{1}{\sqrt{p_m}}\exp\left(-j2\pi\left[\frac{u_mv_m}{p_m}+u_m\frac{\langle v_{m-1}v_{m-2}\dots v_0\rangle}{p_0p_1\cdots p_m}\right]\right).$$

Каждый сомножитель в этом произведении соответствует элементу базовой операции. Таким образом, базовые операции алгоритма БПФ определяются выражением:

$$w^{m}_{z^m}(u_m, v_m) = \frac{1}{\sqrt{p_m}} \exp\left(-j\frac{2\pi}{p_m} u_m v_m\right) \exp\left(-j2\pi u_m \frac{\langle v_{m-1} v_{m-2} \ldots v_0 \rangle}{p_0 p_1 \ldots p_m}\right).$$

Кроме разрядных переменных $v_{m-1}v_{m-2}\ldots v_0$ выражение для номера ядра z^m содержит переменные $u_{m+1}\ldots u_{n-2}u_{n-1}$, это означает, что базовые операции в пределах слоя будут повторяться. На рис. 4.6 показано факторизованное представление алгоритма БПФ для размерности $N = 8$.

			v_0	0	0	0	0	1	1	1	1
			v_1	0	0	1	1	0	0	1	1
			v_2	0	1	0	1	0	1	0	1
u_0	u_1	u_2									
0	0	0		1	1						
0	0	1				1	1				
0	1	0						1	1		
0	1	1								1	1
1	0	0		1	−1						
1	0	1				1	−1				
1	1	0						1	−1		
1	1	1								1	−1

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
1		1					
	1		1				
				1		1	
					1		1
1		−1					
	ω^2		$-\omega^2$				
				1		−1	
					ω^2		$-\omega^2$

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
1				1			
	1				1		
		1				1	
			1				1
1				−1			
	ω				$-\omega$		
		ω^2				$-\omega^2$	
			ω^3				$-\omega^3$

Рис. 4.6. Факторизованное представление алгоритма БПФ

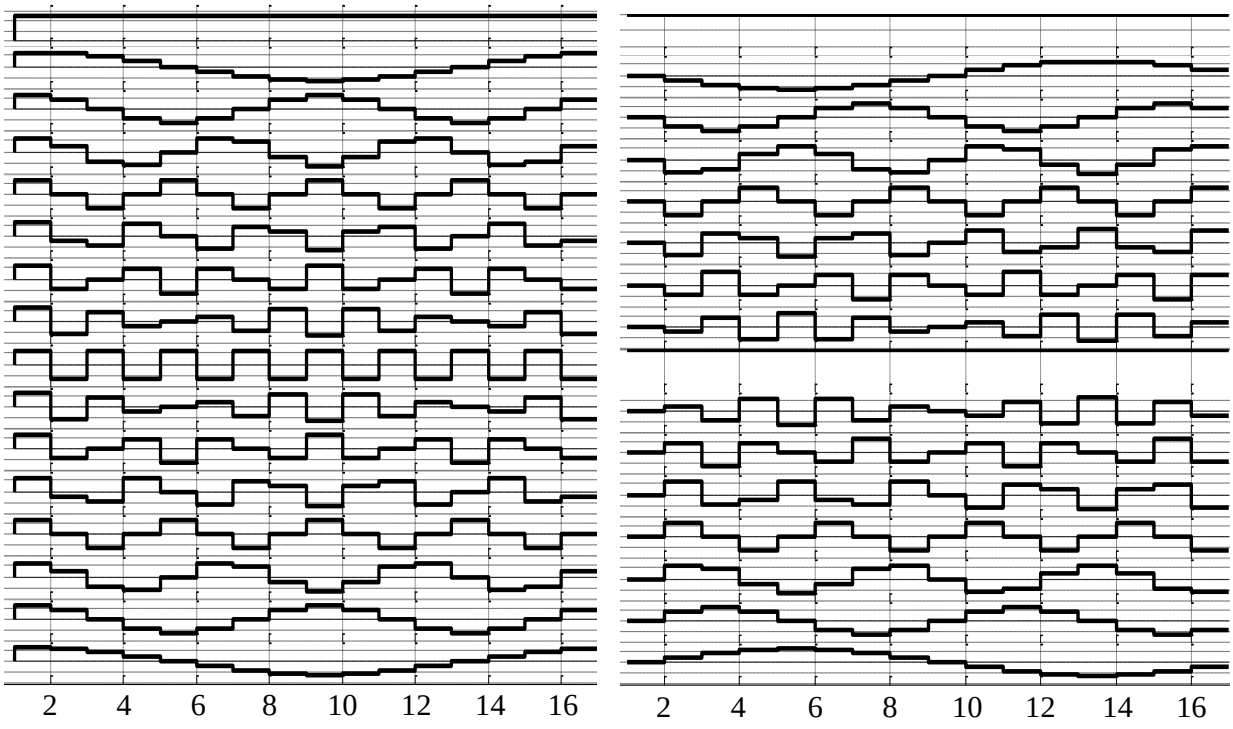


Рис. 4.7. Функции базиса Фурье с естественным упорядочением по частотам следования

На рис. 4.6 множители $1/\sqrt{2}$ с целью упрощения не показаны, использовано также обозначение $\omega = \exp\left(-j\frac{2\pi}{N}\right)$. На рис. 4.7 приведены вещественные и мнимые компоненты функций базиса Фурье, полученные перемножением матриц факторизованного представления.

4.6. Вычислительная эффективность быстрых преобразований

Быстродействие вычислительных алгоритмов, как правило, оценивается числом операций умножения, необходимых для их реализации. Базовая операция (нейронное ядро) с размерностью рецепторного поля p_m и числом аксонов g_m представляет собой матрицу размерности $p_m \times g_m$. Обработка входного вектора в нейронном ядре соответствует умножению вектора на матрицу. Для выполнения матричного умножения требуется $p_m \times g_m$ скалярных операций умножения.

Рассмотрим n-слойное быстрое преобразование с характеристиками ядер: $(p_0, g_0), (p_1, g_1), \ldots, (p_{n-1}, g_{n-1})$. Преобразование имеет размерность по входу $N = p_0 p_1 \ldots p_{n-1}$, а по выходу $M = g_0 g_1 \ldots g_{n-1}$. Номер ядра определяется топологической моделью, например, для модели (3.1) имеем

$$z^m = \langle v_0 v_1 \ldots v_{m-1} u_{m+1} \ldots u_{n-1} \rangle, \qquad (4.8)$$

где основания разрядных чисел заданы таблицами:

$$\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} u_0 & u_1 & \cdots & u_{n-1} \\ p_0 & p_1 & \cdots & p_{n-1} \end{pmatrix}, \qquad \begin{pmatrix} v \\ g \end{pmatrix} = \begin{pmatrix} v_0 & v_1 & \cdots & v_{n-1} \\ g_0 & g_1 & \cdots & g_{n-1} \end{pmatrix}.$$

Из (4.8) следует, что число ядер в слое m будет равно произведению:

$$k_m = g_0 g_1 \ldots g_{m-1} p_{m+1} p_{m+2} \ldots p_{n-1}.$$

При этом в каждом ядре слоя m выполняется $p_m \times g_m$ операций умножения. Суммируя по всем слоям, получим число операций умножения для алгоритма быстрого преобразования:

$$Z = \sum_{m=0}^{n-1} p_m p_{m+1} \cdots p_{n-1} g_0 g_1 \cdots g_{m-1} g_m .$$

На практике для оценки вычислительной эффективности преобразования удобно пользоваться относительными оценками. В качестве базы для сравнения будем использовать прямое преобразование той же размерности. Когда быстрый алгоритм не используется, число вычислительных операций умножения равно $Z_0 = N \cdot M$. Вычислительную эффективность определим относительной характеристикой: $\theta = Z_0 / Z$.

Для оценки порядка величин рассмотрим вариант быстрого преобразования, для которого $p_m = g_m = p$. В этом случае $Z = np^{n+1}$ и $Z_0 = N^2$. Поскольку $N = p^n$, то вычислительная эффективность быстрого перестраиваемого преобразования будет равна:

$$\theta = \frac{Z_0}{Z} = \frac{p^{2n}}{np^{n+1}} = \frac{p^{n-1}}{n} = \frac{N}{p \log_p N}.$$

С ростом размерности обрабатываемого вектора вычислительная эффективность быстро растет, например, при $p = 2$ и $n = 8$ $(N = 256)$ она равна $\theta = 16$, а при $p = 2, n = 10$ $(N = 1024)$ – $\theta = 51{,}2$. Зависимость вычислительной эффективности от размерности преобразования при $p = 2$ показана на рис. 4.8. При расчете предполагалось, что все коэффициенты базовых операций отличны от нуля и единицы.

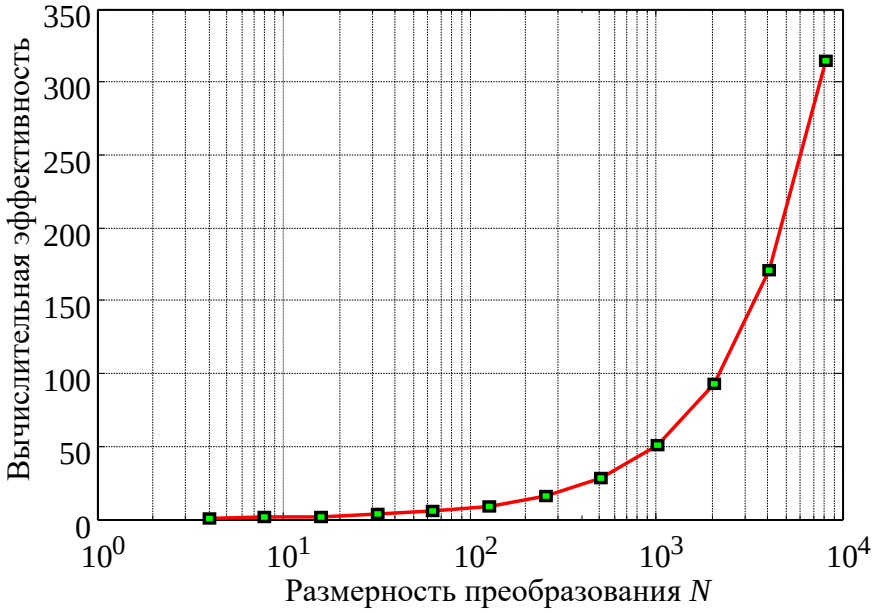


Рис. 4.8. Зависимость вычислительной эффективности быстрого перестраиваемого преобразования от его размерности

В конкретной реализации преобразования может быть получен дополнительный существенный выигрыш за счет того, что не требуется выполнять операции умножения для нулевых и единичных значений коэффициентов базовых операций.

5. ФРАКТАЛЬНЫЕ СВОЙСТВА БЫСТРЫХ ПРЕОБРАЗОВАНИЙ

Термин *фрактал* был введен Менуа Мандельбротом в 1975 г. для обозначения особого класса многомерных функций, обладающих свойством самоподобия и дробной размерности. В гл. 2 было показано, что морфология быстрых преобразований является самоподобной. Это обстоятельство позволяет реализовать генераторы различных фрактальных объектов в классе быстрых перестраиваемых преобразований.

Самоподобие – это ключевое свойство фракталов, отражающее независимость основных геометрических особенностей фрактального объекта от изменения масштаба. Большинство объектов утрачивают детали, когда их приближают, т. е. уменьшают масштаб для более детального рассмотрения. Фрактал же можно приближать до бесконечности без потери детализации. Для фракталов допустимые изменения масштаба всегда кратны некоторому целому числу. Это целое определяет число разбиений масштабной единицы носителя фрактала. На каждой составной части фрактал повторяет себя по геометрическим свойствам. Таким образом, свойство самоподобия фракталов характеризуется двумя моментами: во-первых, последовательным итерационным разбиением масштабной единицы и, во-вторых, повторением геометрии фрактала на каждой составной части. При точном повторении геометрии самоподобие влечет за собой условие тождественности образуемых составных частей. Геометрия объекта может повторяться не полностью, а только по отдельным свойствам или в статистическом смысле. В этом случае идеальная модель фрактала нарушается, однако такие объекты удобно рассматривать как расширения концепции фрактала. Примером могут служить квазифракталы и мультифракталы. Точную грань между фрактальными и не фрактальными объектами провести трудно. Иногда удобно не фрактальный объект рассматривать как фрактал с ограничениями по свойствам самоподобия. Минимально необходимым инвариантом самоподобного процесса является только разбиение масштабной единицы на более мелкие составные компоненты. Если процесс допускает подобную интерпретацию, то его можно рассматривать как фрактальный объект.

Когда процесс генерации фрактала ограничивается каким-либо шагом k, то построенный объект называется предфракталом порядка k. Предфрактал представляет собой аппроксимацию фрактала, которая обладает свойством самоподобия только на ограниченном числе масштабных уровней. В данной

главе приведены примеры аппроксимации классических фракталов быстрыми перестраиваемыми преобразованиями [28], что является конструктивным подтверждением фрактальности быстрых преобразований.

5.1. Аналитическая форма регулярного фрактала

На рис. 5.1 показаны 3 последовательных итерации фрактала Кантора. На каждой итерации из непрерывной области удаляется средняя треть, что эквивалентно масштабной деформации исходной функции в отношении 1:3. Построим аналитическую форму вначале для фрактала Кантора, а затем обобщим ее на более широкий класс регулярных фракталов. Будем полагать, что фрактал Кантора задан на непрерывном интервале $U=[0,1)$. Любую точку данного интервала $u\in U$ можно записать в троичной системе счисления в виде дроби:

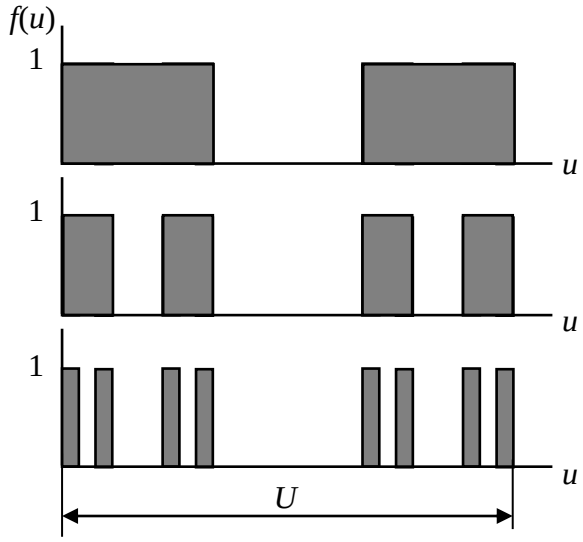


Рис. 5.1. Одномерный фрактал Кантора

$$u=0,u_1u_2\ldots u_n\ldots, \qquad (5.1)$$

где разрядные числа $u_i\in\{0,1,2\}$ следует рассматривать как целые переменные, определяющие положение точки на интервале. Введем также непрерывные переменные $\tilde{u}_i\in[0,3)$. Очевидно, что последовательность разрядных чисел (5.1) можно в любом месте оборвать, завершив ее непрерывной переменной, в результате точка интервала будет представлена в виде

$$u=0,u_1u_2\ldots u_{n-1}\tilde{u}_n.$$

Дискретные числа u_i будем рассматривать как результат действия оператора, выделяющего целую часть из непрерывной переменной $\tilde{u}_i$. Операцию масштабирования можно рассматривать как изменение правила представления точек интервала. Например, для базовой функции Кантора аргумент определяется правилом $u=\langle 0,\tilde{u}_1\rangle=3^{-1}\tilde{u}_1$. Для функции второй итерации: $u=\langle 0,u_1\tilde{u}_2\rangle=3^{-1}u_1+3^{-2}\tilde{u}_2$. Для функции третьей итерации: $u=\langle 0,u_1u_2\tilde{u}_3\rangle=$ $=3^{-1}u_1+3^{-2}u_2+3^{-3}\tilde{u}_3$ и т. д. На непрерывном интервале $[0,3)$ определим функцию:

$$\varphi(\tilde{u}) = \begin{cases} 1 & \text{для} \quad \tilde{u} \in [0,1), \\ 0 & \text{для} \quad \tilde{u} \in [1,2), \\ 1 & \text{для} \quad \tilde{u} \in [2,3). \end{cases}$$

Нетрудно проверить, что фрактал Кантора аналитически можно записать в виде бесконечного произведения:

$$f(u) = \varphi(\tilde{u}_1)\varphi(\tilde{u}_2)\varphi(\tilde{u}_3)\ldots\varphi(\tilde{u}_n)\ldots,$$

где правило вычисления точки интервала u задается выражением (5.1). Квазифрактал отличается от фрактала нестрогим повторение структуры на масштабных уровнях. Обобщенная аналитическая форма квазифрактала имеет вид

$$f(u) = \varphi_{i^1}(\tilde{u}_1)\varphi_{i^2}(\tilde{u}_2)\varphi_{i^3}(\tilde{u}_3)\ldots\varphi_{i^n}(\tilde{u}_n)\ldots, \qquad (5.2)$$

где индексы $i^1, i^2, i^3 \ldots i^n \ldots$ определяют множества допустимых функций для каждого шага фрактальной итерации. Правило выбора функций из множества может быть детерминированным или случайным. Например, возможно следующее детерминированное правило:

$$i^1 = 0, \quad i^2 = \langle u_1 \rangle, \quad i^3 = \langle u_1 u_2 \rangle, \ldots, i^n = \langle u_1 u_2 \ldots u_{n-1} \rangle, \ldots .$$

Угловые скобки, как и прежде, определяют поразрядное представление индекса в соответствующей системе счисления.

5.2. Фрактал Кантора

Дискретная аппроксимация фрактала строится на последовательности расширяющихся носителей, заданных дискретными интервалами:

$$U_i = \{0, 1, 2, \ldots, N_i - 1\}, \quad i = 0, 1, 2, \ldots, n-1.$$

Точки интервалов определяются правилом $u = \langle u_{n-1} u_{n-2} \ldots u_0 \rangle$. В общем случае разрядные переменные имеют различные локальные области определения: $u_i \in \{0,1,\ldots, p_i - 1\}$. На локальных интервалах определены кусочно-постоянные дискретные функции $\varphi_i(u_i)$. Аппроксимация фрактала порядка n называется предфракталом и записывается в виде конечного произведения:

$$f(u) \approx \varphi_{i^0}(u_0)\varphi_{i^1}(u_1)\ldots\varphi_{i^{n-1}}(u_{n-1}).$$

Например, для фрактала Кантора все дискретные образующие функции одинаковы и имеют вид

$$\varphi(u) = \begin{cases} 1 & \text{для} \quad u = 0, \\ 0 & \text{для} \quad u = 1, \\ 1 & \text{для} \quad u = 2. \end{cases}$$

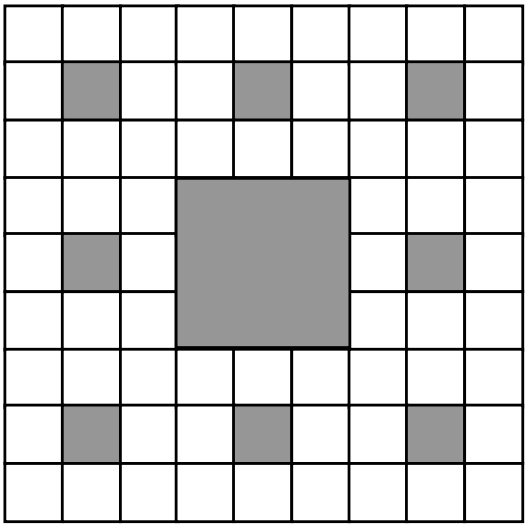


Рис. 5.2. Двумерный фрактал Кантора

Двумерным вариантом фрактала Кантора является плоская фигура, заданная на непрерывных интервалах $U=[0,1)$, $V=[0,1)$. На рис. 5.2 показана одна из итераций фрактала. Координаты точки в плоскости фигуры определяются бесконечными дробями:

$$u=0,u_1u_2\ldots u_n\ldots,\quad u_i\in\{0,1,2\},$$

$$v=0,v_1v_2\ldots v_n\ldots,\quad v_i\in\{0,1,2\}.$$

Обозначим через $\tilde{u}_i\in[0,3)$ и $\tilde{v}_i=[0,3)$ соответствующие непрерывные переменные, тогда двумерный фрактал Кантора в аналитической форме можно записать в виде бесконечного произведения:

$$f(u,v)=\varphi(\tilde{u}_1,\tilde{v}_1)\varphi(\tilde{u}_2,\tilde{v}_2)\varphi(\tilde{u}_3,\tilde{v}_3)\ldots\varphi(\tilde{u}_n,\tilde{v}_n)\ldots$$

Дискретная аппроксимация порядка n для двумерного фрактала Кантора представляет собой конечное произведение:

$$f(u,v)\approx\varphi(u_0,v_0)\varphi(u_1,v_1)\varphi(u_2,v_2)\ldots\varphi(u_{n-1},v_{n-1}),\qquad(5.3)$$

где все дискретные образующие функции имеют вид:

$$\varphi(u_i,v_i)=\begin{pmatrix}1&1&1\\1&0&1\\1&1&1\end{pmatrix}.$$

Нетрудно заметить, что полученное выражение по форме совпадает с представлением (4.3) для элементов матрицы H быстрого преобразования. Сравнивая (4.3) и (5.3), получим следующие правила определения параметров нейронных ядер:

$$w_{i^m}^{m}(u_m,v_m)=\varphi(u_m,v_m).$$

На рис. 5.3 показана аппроксимация для двумерного фрактала Кантора в классе быстрых преобразований на интервалах длиной $N=3^2$. Аппроксимирующее преобразование имеет 2 слоя, каждой матрице-сомножителю отвечает один нейронный слой. Все непоказанные элементы матриц равны нулю.

Из рисунка видно, что результирующая матрица подобна одной из итераций двумерного фрактала Кантора.

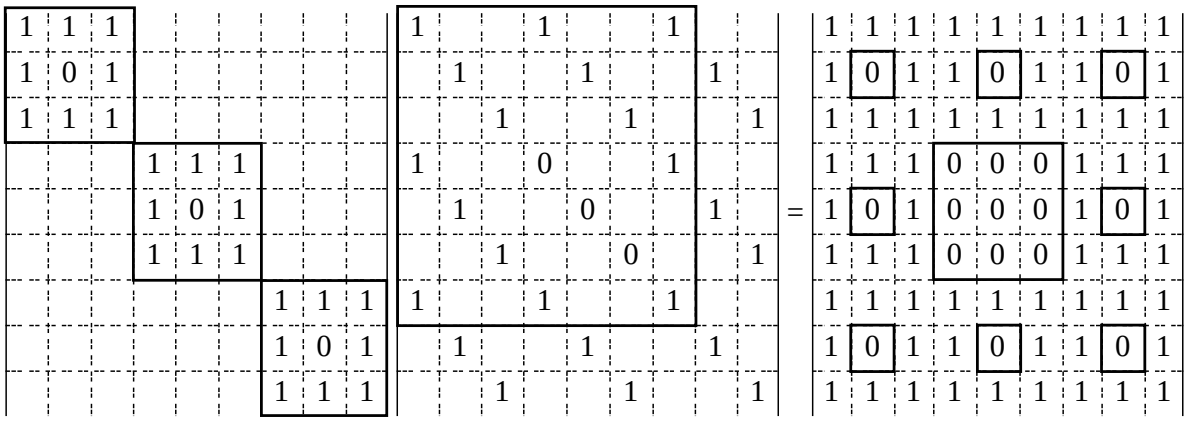


Рис. 5.3. Аппроксимация 2-го порядка двумерного фрактала Кантора

Предфрактал 4-го порядка для ковра Серпинского, построенный с помощью программного генератора, показан на рис. 5.4. Матрица размером 81 × 81 содержит 4096 ненулевых элементов. Зависимость числа ненулевых элементов от размеров матрицы представлена в табл. 5.1.

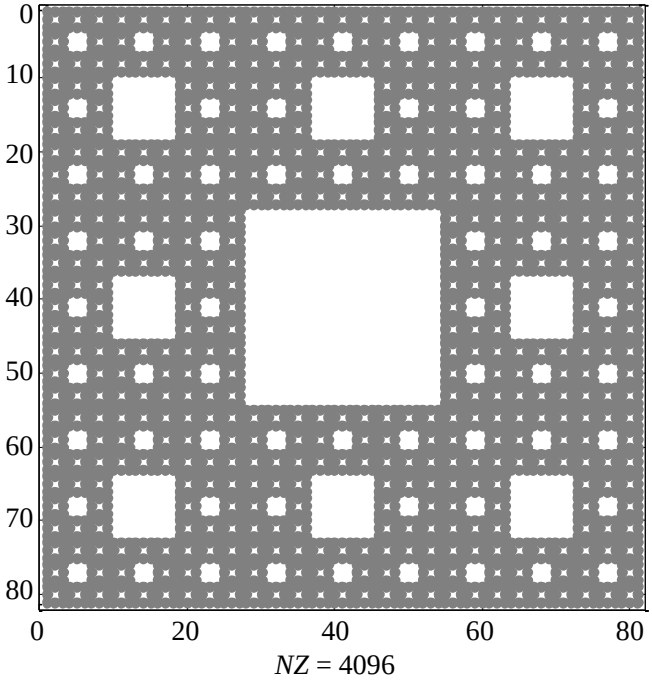


Рис. 5.4. Аппроксимация 4-го порядка двумерного фрактала Кантора

Таблица 5.1

Зависимость числа ненулевых элементов от размеров матрицы предфрактала Кантора

Порядок предфрактала	2	3	4	5
Размер матрицы	9	27	81	243
Число ненулевых элементов	64	512	4096	32 768

Фрактальная размерность. На рис. 5.5 в логарифмическом масштабе представлена зависимость числа ненулевых элементов от размера матрицы предфрактала. Тангенс угла наклона линии характеризует значение фрак-

тальной размерности. Расчетная формула для вычисления фрактальной размерности имеет вид

$$D = \frac{\log\left(\frac{NZ}{NZ'}\right)}{\log\left(\frac{N}{N'}\right)}, \tag{5.4}$$

где NZ и NZ' – число ненулевых элементов в матрицах предфракталов порядка n и n'; N и N' – размеры матриц предфракталов.

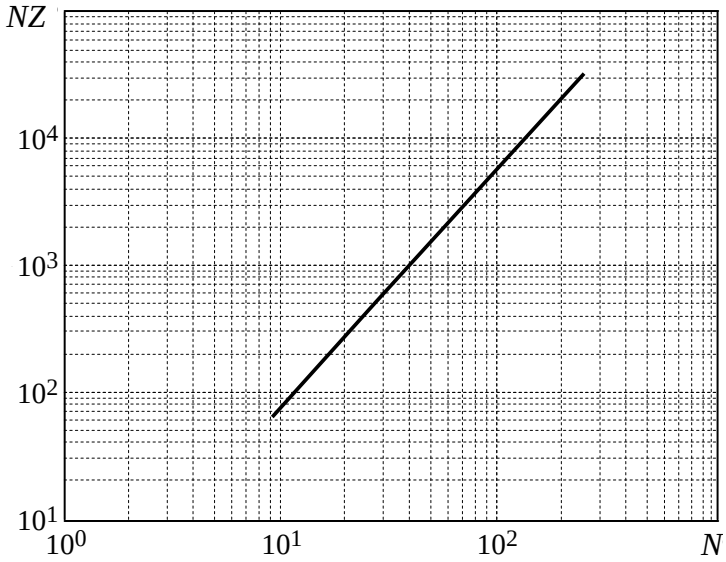


Рис. 5.5. Зависимость числа ненулевых элементов от размера матрицы предфракталов Кантора

Например, используя значения табл. 5.1, для предфракталов Кантора порядка 4 и 5 получим:

$$D = \frac{\log\left(\frac{32768}{4096}\right)}{\log\left(\frac{243}{81}\right)} = 1.8928,$$

что с точностью до четвертого знака после запятой совпадает с теоретическим значением.

5.3. Фрактал «салфетка Серпинского»

На рис. 5.6 показана одна из итераций двумерного фрактала, называемого «салфеткой Серпинского». Фрактал задан на непрерывных интервалах $U = [0,1)$, $V = [0,1)$. Дискретная аппроксимация фрактала на интервале $N = 2^n$ определяется выражением (5.3), где образующая функция имеет вид

$$\varphi(u_i, v_i) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

На рис. 5.7 показана аппроксимация для «салфетки Серпинского» в классе быстрых преобразований на интервале длиной $N = 2^3$. Аппроксимирующее преобразование факторизуется в произведение трех матриц. Из рисунка видно, что результирующая матрица подобна непрерывной форме данного фрактала. Предфрактал 7-го порядка для «салфетки Серпинского» показан на рис. 5.8. Матрица размером 128 × 128 содержит 2187 ненулевых элементов.

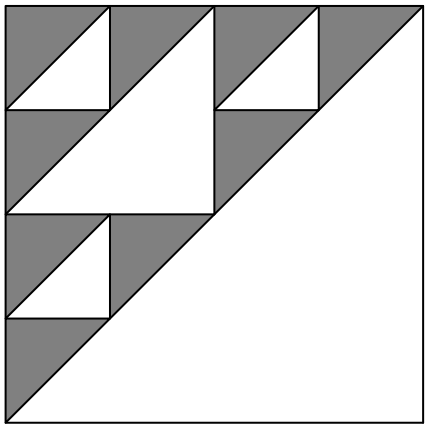


Рис. 5.6. Двумерный фрактал «салфетка Серпинского»

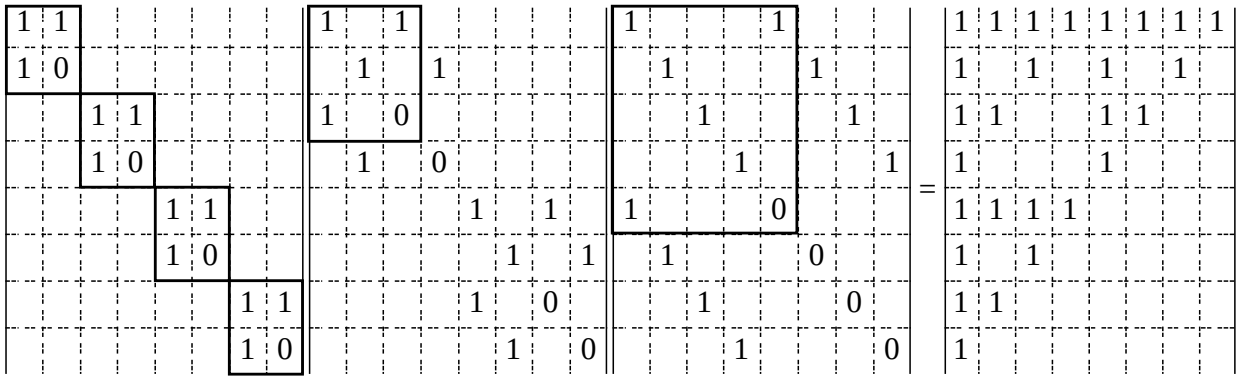


Рис. 5.7. Аппроксимация фрактала Серпинского

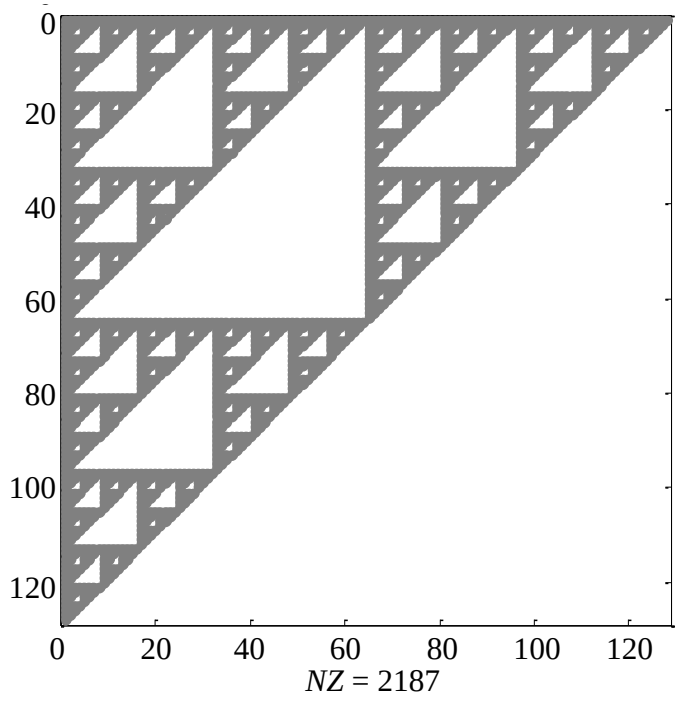


Рис. 5.8. Двумерный предфрактал Серпинского 7-го порядка

Размерность фрактала. В табл. 5.2 приведена зависимость числа ненулевых элементов предфрактала от размера несущей матрицы.

Таблица 5.2

Зависимость числа ненулевых элементов в матрице двумерного предфрактала Серпинского от размера матрицы

Порядок предфрактала	2	3	4	5	6	7
Размер матрицы	4	8	16	32	64	128
Число ненулевых элементов	9	27	81	243	729	2187

Подставляя в формулу (5.4) значения для предфракталов 6-го и 7-го порядков, получим фрактальную размерность:

$$D = \frac{\log\left(\frac{2187}{729}\right)}{\log\left(\frac{128}{64}\right)} = 1.5849\,.$$

6. БЫСТРЫЕ НЕЙРОННЫЕ СЕТИ

В предыдущей главе была рассмотрена процедура настройки самоподобных перестраиваемых преобразований к системам базисных функций. Это оказалось возможным благодаря тому, что рассмотренные базисы также обладают самоподобной структурой. Теперь обратимся к методам, позволяющим выполнять настройку перестраиваемых преобразований к произвольным сигнальным функциям. Произвольность выбора объектов обучения дает возможность рассматривать перестраиваемые преобразования как особый класс нейронных сетей, обладающих быстрым алгоритмом выполнения. Отсюда возникло название – быстрые нейронные сети (БНС).

В общем случае произвольная система функций не обладает самоподобной структурой, поэтому такую настройку (или обучение – в терминологии нейронных сетей) можно выполнить только частично, для нескольких функций, но и этого оказывается достаточным для решения многих задач избирательной селекции сигналов. Настроенная БНС представляет собой быстрое линейное преобразование, приспособленное для распознавания сигнальных функций. Первоначально термин «приспособленное быстрое преобразование» [1] относился только к спектральным преобразованиям, поскольку алгоритмы настройки к неортогональным функциям отсутствовали, а для спектральных преобразований в то время уже существовали первые решения на основе параметрического представления ортогональных ядер. В настоящей главе будут рассмотрены общие методы настройки неортогональных и ортогональных приспособленных преобразований, основанные на возможности мультипликативного представления произвольной сигнальной функции, заданной на дискретном интервале.

6.1. Мультипликативное представление сигнальных функций

Будем полагать, что эталонный сигнал задан функцией $f(u)$ на дискретном интервале длиной $N = p_0 p_1 \dots p_{n-1}$, где p_m – произвольные целые числа. Представим аргумент функции в позиционной многоосновной системе счисления с основаниями $p_0, p_1, \dots, p_{n-1}$. Формула перехода, как известно, имеет вид:

$$u = \langle u_{n-1} u_{n-2} \dots u_0 \rangle =$$
$$= u_{n-1} p_{n-2} p_{n-3} \dots p_0 + u_{n-2} p_{n-3} p_{n-4} \dots p_0 + \dots + u_1 p_0 + u_0,$$

где $u_i \in [0,1,\ldots p_i - 1]$ – разрядные переменные. В результате данного преобразования сигнал представляется как многомерная функция $f\langle u_{n-1}u_{n-2}\ldots u_0\rangle$. Каждый аргумент функции определяет некоторый масштабный срез сигнала. Зафиксируем все аргументы функции, кроме u_m. Варьируя свободный аргумент u_m, получим выборку S_m (с числом элементов p_m). Фрактальным фильтром [22] частотной локализации m назовем произвольный функционал $F(S_m)$, определенный на выборке S_m. Операцию фрактальной фильтрации можно записать в виде:

$$f_{out}\langle u_{n-1}u_{n-2}\ldots u_{m+1}u_{m-1}\ldots u_0\rangle = \underset{u_m}{F}\left(f_{inp}\langle u_{n-1}u_{n-2}\ldots u_0\rangle\right).$$

В простейшем варианте фрактальный фильтр выполняет суммирование значений функции по аргументу u_m. Если $m=0$, то такой фильтр генерализует сигнал, сглаживая мелкие детали. Фрактальная фильтрация по аргументу u_m приводит к сокращению интервала определения сигнала в p_m раз.

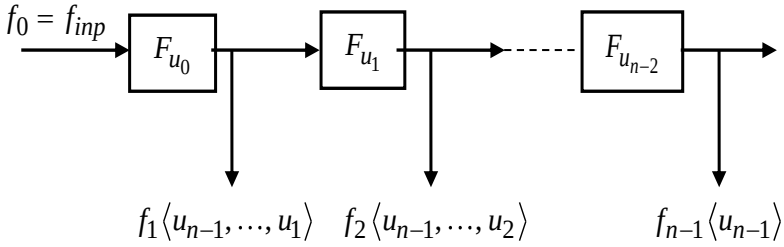


Рис. 6.1. Цепочка фрактальных фильтров

Рассмотрим цепочку фрактальных фильтров, показанную на рис. 6.1. Выходные сигналы для фильтров цепочки определяются рекуррентным соотношением:

$$f_m\langle u_{n-1}\ldots u_m\rangle = \underset{u_{m-1}}{F}\left(f_{m-1}\langle u_{n-1}\ldots u_m u_{m-1}\rangle\right).$$

Введем функции:

$$\varphi_{i^m}(u_m) = \frac{f_m\langle u_{n-1}u_{n-2}\ldots u_m\rangle}{f_{m+1}\langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle}, \qquad m = 0,1,\ldots n-2, \tag{6.1}$$

где $i^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle$. Используя определения функций, можно записать:

$$f_m\langle u_{n-1}u_{n-2}\ldots u_m\rangle = \varphi_{i^m}(u_m)\, f_{m+1}\langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle, m = 0,1,\ldots n-2.$$

Из рекуррентных соотношений (6.1) непосредственно следует:

$$f(u) = f_0\langle u_{n-1}u_{n-2}\ldots u_0\rangle = \varphi_{i^0}(u_0)\varphi_{i^1}(u_1)\ldots\varphi_{i^{n-2}}(u_{n-2})\varphi_{i^{n-1}}(u_{n-1}). \tag{6.2}$$

Полученное выражение позволяет утверждать, что любая функция, заданная на дискретном интервале, может быть представлена в мультипликативной форме в виде конечного произведения функций-сомножителей. Число сомножителей в произведении равно числу сомножителей в разложении целочисленной длины интервала в произведение целых чисел. Сравнивая (6.2) с (5.2), сделаем вывод, что полученное выражение можно рассматривать как дискретную аппроксимацию квазифрактала. Отсюда следует используемая далее терминология: фрактальное разложение, фрактальная фильтрация, фрактальный множитель.

Замечание. Декомпозицию функции можно начать со старшего разряда u_{n-1}, в этом случае получим другое разложение:

$$f(u) = f_0\langle u_{n-1}u_{n-2}\ldots u_0\rangle = \varphi_{i^{n-1}}(u_{n-1})\varphi_{i^{n-2}}(u_{n-2})\ldots\varphi_{i^1}(u_1)\varphi_{i^0}(u_0),$$

где $i^m = \langle u_{m-1}u_{m-2}\ldots u_1u_0\rangle$.

6.1.1. Цензурирование нулей

При алгоритмической реализации фрактального разложения необходимо проверять значение знаменателя в выражении (6.1) на равенство нулю. Благодаря наличию множества возможных решений при мультипликативном разложении, нулевое значение знаменателя всегда можно заменить на любое удобное ненулевое (например, единичное), откорректировав при этом очередной сомножитель фрактального разложения. Это ведет к локальному изменению одного фрактального фильтра в цепочке, но не влияет на результат мультипликативной аппроксимации. Менее очевидно, что необходимо также проверять на нуль значения числителя. Если для всех значений разрядной переменной u_m числитель в выражении (6.1) равен нулю, то равен нулю и фрактальный множитель $\varphi_{i^m}(u_m)$, что препятствует построению быстрых приспособленных преобразований с невырожденными ядрами. Если значение знаменателя и числителя близки к нулю, то фрактальный множитель выбирается не нулевым, например, в виде единичного импульса $\varphi(u) = (1\,0\,0\ldots0)$ на интервале определения множителя. Знаменатель при этом не корректируется. Если числитель отличен от нуля, а знаменатель близок к нулю, то значение знаменателя принимается равным единице. Фрактальный множитель в этом случае совпадает с функцией числителя.

6.2. Алгоритм обучения быстрых нейронных сетей

Выберем для определенности нейронную сеть, заданную топологией Кули–Тьюки «с прореживанием по времени». Топологическая схема данного вида описывается следующей моделью (см. гл. 3):

$$\begin{aligned} U^m &= \langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m v_{m-1}v_{m-2}\ldots v_1 v_0\rangle, \\ V^m &= \langle u_{n-1}u_{n-2}\ldots u_{m+1}v_m v_{m-1}v_{m-2}\ldots v_1 v_0\rangle, \\ z^m &= \langle u_{n-1}u_{n-2}\ldots u_{m+1}v_{m-1}v_{m-2}\ldots v_1 v_0\rangle. \end{aligned} \qquad (6.3)$$

Обучающим множеством являются одна или несколько функций, представленных в мультипликативной форме (6.2), которые назовем опорными функциями. Цель обучения состоит в следующем: надо настроить нейронные ядра таким образом, чтобы результирующая матрица преобразования содержала опорные функции в виде столбцов. Номер столбца, содержащего опорную функцию, назовем точкой приспособления. Точки приспособления относятся к координатному пространству V^{n-1}. В гл. 4 было доказано, что элементы матрицы быстрой нейронной сети связаны с элементами нейронных ядер выражением

$$h(U,V) = w^{n-1}_{z^{n-1}}(u_{n-1},v_{n-1})\, w^{n-2}_{z^{n-2}}(u_{n-2},v_{n-2})\ldots w^{0}_{z^{0}}(u_0,v_0).$$

Сравнивая это выражение с (6.2), непосредственно получим правило обучения нейронных ядер:

$$w^{m}_{z^m}(u_m,v_m) = \varphi^{k}_{i^m}(u_m). \qquad (6.4)$$

Здесь k – номер опорной функции. Зададим точку приспособления числом, представленным в поразрядной форме:

$$x = \langle x_{n-1}x_{n-2}\ldots x_0\rangle,$$

тогда номер настраиваемых ядер по слоям будет определяться выражением

$$z^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}x_{m-1}x_{m-2}\ldots x_1 x_0\rangle.$$

Для $m=0$ имеем $z^0 = \langle u_{n-1}u_{n-2}\ldots u_1\rangle$, это означает, что, независимо от выбора точки приспособления, все ядра слоя будут настраиваться, причем номер ядра определяется из условия: $z^0 = i^0$. Настройка элементов ядер этого слоя выполняется по правилу:

$$w^{0}_{z^0}(u_0,v_0) = \varphi^{k}_{i^0}(u_0).$$

Очевидно, должно быть задано взаимно-однозначное соответствие $k \leftrightarrow v_0$ между номером опорной функции и разрядной переменной v_0. Эта разрядная переменная принимает значения $0,1,\ldots, g_0 - 1$. Отсюда следует вывод, что число точек приспособления не может быть больше, чем g_0, и для данной топологии точки приспособления размещаются в соседних столбцах, т. е. принимают значения:

$$x = \langle x_{n-1}x_{n-2}\ldots x_1 0\rangle, \langle x_{n-1}x_{n-2}\ldots x_1 1\rangle, \ldots, \langle x_{n-1}x_{n-2}\ldots x_1 g_0 - 1\rangle.$$

Далее, для $m = n-1$ имеем $z^{n-1} = \langle x_{n-2}x_{m-3}\ldots x_1 v_0\rangle$, $i^{n-1} = \langle\ \rangle$, т. е. в последнем слое настраиваться будут g_0 ядер, и только по одному столбцу. Для внутренних слоев настраиваемые ядра также будут заполнены частично. На рис. 6.2 показано факторизованное представление быстрого преобразования в топологии Кули–Тьюки, приспособленного к двум функциям, размещенным в двух первых столбцах матрицы преобразования.

		v_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
		v_1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
		v_0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
u_2	u_1	u_0																								
0	0	0	1	1							1		*						1				*			
0	0	1	1	1								1		*						1				*		
0	1	0			1	1					1		*								*				*	
0	1	1			1	1						1		*								*				*
1	0	0					1	1							1		*		1				*			
1	0	1					1	1								1		*		1				*		
1	1	0							1	1					1		*				*				*	
1	1	1							1	1						1		*				*				*

Рис. 6.2. Свободные параметры быстрого преобразования, приспособленного к двум функциям

Символом «*» выделены элементы ядер, которые выбираются произвольно (отсчет слоев ведется слева направо, начиная с 0). Незадействованные степени свободы можно использовать, следуя различным стратегиям:

- Если требуется обеспечить максимальную скорость вычисления преобразования, то целесообразно заполнить свободные элементы комбинациями нулей и единиц, поскольку в этом случае операции умножения вырождаются в операции присвоения.
- Если требуется упростить алгоритм настройки преобразования, то можно использовать копирование ядер в пределах слоя.

• Если требуется более глубокое приспособление к набору функций, то можно расширить опорную базу приспособления за счет использования фрактально-фильтрованных образов дополнительных функций (см. [22]).

Можно изменить структуру преобразования, так чтобы удалить неиспользуемые степени свободы, в этом случае мы приходим к пирамидальным самоподобным сетям глубокого обучения (см. гл. 10).

Можно декомпозировать структуру преобразования, так чтобы при добавлении дополнительных входов полностью использовать степени свободы, в этом случае мы приходим к фовеальным сетям глубокого обучения (см. гл. 9).

6.2.1. Вычислительная эффективность алгоритма обучения

Вычислительные затраты алгоритма обучения в основном связаны с мультипликативной декомпозицией опорных функций. При этом наиболее затратной является операция деления (6.1) связанная с формированием фрактального множителя. Учитывая, что номер фрактального множителя для уровня m определяется кортежем $i^m = \langle u_{n-1}u_{n-2}\dots u_{m+1}\rangle$ и для получения каждого множителя требуется p_m операций деления, число операций деления для декомпозиции одной функции будет равно $\sum_{m=0}^{n-1} p_{n-1}p_{n-2}\cdots p_m$.

Приспособленное преобразование в максимальном варианте может быть обучено к g_0 опорным функциям, поэтому окончательно число вычислительных операций, требуемых для алгоритма обучения, будет равно:

$$Z_t = g_0 \sum_{m=0}^{n-1} p_{n-1}p_{n-2}\cdots p_m .$$

Для оценки порядка величин рассмотрим вариант быстрого преобразования, для которого $p_m = g_m = p$. В этом случае

$$Z_t = p\sum_{m=0}^{n-1} p^{n-m} = \frac{p^2\left(p^n-1\right)}{p-1}.$$

Напомним, что для этого варианта число операций умножения, требуемых для выполнения быстрого преобразования, равно $Z = np^{n+1}$ (см. 4.6). Полагая, что операции деления и умножения примерно одинаковы по вычислительным затратам, рассмотрим отношение

$$\frac{Z_t}{Z} = \frac{p^2\left(p^n-1\right)}{(p-1)np^{n+1}} < \frac{p}{n(p-1)} \ll 1.$$

Таким образом, мы пришли к неожиданному результату: вычислительные затраты на обучение приспособленного преобразования меньше, чем затраты на выполнение его быстрого алгоритма, причем с ростом размерности преобразования этот эффект только увеличивается.

6.2.2. Согласование опорных функций с терминальным полем быстрого преобразования

При мультипликативной декомпозиции функции по алгоритму, представленному в 6.1, преобразование аргумента опорной функции к многоразрядному представлению реализуется по правилу:

$$U = U^0 = \langle u_{n-1} u_{n-2} \ldots u_0 \rangle .$$

Это правило согласовано с топологией Кули–Тьюки «с прореживанием по времени», следует из выражения (6.3) при $m = 0$ и в дальнейшем считается фиксированным. При выполнении настройки опорные функции размещаются вдоль столбцов матрицы перестраиваемого преобразования, и аргументом для них является номер строки. Для быстрого перестраиваемого преобразования представление номера строки в поразрядной форме зависит от выбранной топологии, а точнее, от топологии входного терминального поля. Например, для топологии Гуда из (3.3) имеем:

$$U = U^0 = \langle u_0 u_1 \ldots u_{n-2} u_{n-1} \rangle .$$

Это представление отличается от принятого правила мультипликативной декомпозиции. Несогласованность топологических слов устраняется предварительной перестановкой отсчетов функции перед выполнением алгоритма обучения.

6.3. Спектральные приспособленные преобразования

Для задач классификации и распознавания сигналов существенное значение имеют процедуры предварительной обработки, ориентированные на устранение избыточности и выделение информативных признаков. Использование ортогональных преобразований для этих целей позволяет представить информацию, содержащуюся в исходном сигнале, в виде взаимно независимых спектральных составляющих. Поскольку энергия сигнала определяется суммой квадратов спектральных коэффициентов, то по модулю спектрального коэффициента можно непосредственно судить о значимости информативного признака. Известно, что максимальное сокращение избыточности данных обеспечивается ортогональным преобразованием Карунена–Лоэва, которое

образуется собственными векторами ковариационной матрицы сигнала. Однако использование данного преобразования сопряжено со значительными вычислительными затратами. По этой причине метод Карунена–Лоэва не применяется для обработки данных высокой размерности. Если при построении преобразования Карунена–Лоэва ограничиться только одним собственным вектором, имеющим максимальную значимость (главной компонентой), то объем вычислений резко сокращается. Достаточно часто главную компоненту можно выделить из сигнала непосредственно, как опорный образ, относительно которого наблюдаются все сигнальные вариации. Ортогональное преобразование, настроенное на одну главную компоненту, относится к классу приспособленных, иногда их называют ортогональными нейронными сетями.

В силу ортогональности результатом воздействия приспособленного преобразования на опорный образ является выходной вектор, у которого одна из координат равна единице, а остальные координаты – нулю. Таким образом, приспособленное ортогональное преобразование идеально подходит для селективного распознавания сигналов.

6.3.1. Достаточные условия приспособленности быстрого ортогонального преобразования

Для спектральных преобразований удобно использовать конструктивное определение условия приспособленности, которое можно сформулировать следующим образом. Ортогональное преобразование, заданное матрицей $h(u,v)$, приспособлено к функции $f(u)$ по столбцу x, если выполнены условия:

$$\begin{cases} \sum\limits_{u} h(u,v)\, f(u) = 1 \quad \text{при} \quad v = x, \\ \sum\limits_{u} h(u,v)\, f(u) = 0 \quad \text{при} \quad v \neq x. \end{cases} \tag{6.5}$$

(Предполагается, что опорная функция нормирована условием: $\sum\limits_{u} f^2(u) = 1.$) Подставив в левую часть (6.5) выражения (4.3) и (6.2), после преобразований получим:

$$\sum_{u} h(u,v)\, f(u) = \sum_{u_0} w_{z^0}(u_0, v_0)\varphi_{i^0}(u_0) \sum_{u_1} w_{z^1}(u_1, v_1)\varphi_{i^1}(u_1) \times \ldots \times$$
$$\times \sum_{u_{n-1}} w_{z^{n-1}}(u_{n-1}, v_{n-1})\varphi_{i^{n-1}}(u_{n-1}).$$

Из данной формулы очевидно, что приспособленность спектрального преобразования обеспечивается, когда настраиваемые ядра любого слоя m приспособлены к соответствующим компонентам фрактального разложения опорной функции:

$$\begin{cases} \sum\limits_{u_m} w_{z^m}(u_m, v_m)\varphi_{i^m}(u_m) = 1 \quad \text{при} \quad v_m = x_m, \\ \sum\limits_{u_m} w_{z^m}(u_m, v_m)\varphi_{i^m}(u_m) = 0 \quad \text{при} \quad v_m \neq x_m. \end{cases}$$

Соответствие между индексами z^m и i^m определяется выбором точки приспособления. Таким образом, задача настройки быстрого приспособленного спектрального преобразования сводится к построению приспособленных ортогональных ядер.

6.3.2. Генерация приспособленных ортогональных ядер

Рассмотрим возможный алгоритм построения ортогональных ядер. Ортонормированный базис с векторами $\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{p-1}$ будем называть приспособленным к вектору $\mathbf{f} = (f_0, f_1, \ldots, f_{p-1})$, если один из векторов базиса с точностью до скаляра совпадает с $\mathbf{f}$. Для определенности будем полагать, что вектор $\mathbf{f}$ имеет единичную длину и $\mathbf{w}_0 = \mathbf{f}$. Для построения приспособленного базиса воспользуемся процедурой ортогонализации векторов Грамма–Шмидта [28]. Ортонормированный базис строится из опорной системы линейно независимых векторов $\{\mathbf{x}_i\}$, в которой $\mathbf{x}_0 = \mathbf{w}_0$.

Реализация процедуры. На первом шаге находим вектор $\mathbf{z}_1$, ортогональный вектору $\mathbf{w}_0$. Вектор $\mathbf{z}_1$ ищется в виде линейной комбинации: $\mathbf{z}_1 = \mathbf{x}_1 + a_{10}\mathbf{w}_0$. Коэффициент a_{10} определяется из условия ортогональности векторов $\mathbf{z}_1$ и $\mathbf{w}_0$. Скалярное произведение ортогональных векторов равно нулю, а так как $(\mathbf{w}_0, \mathbf{w}_0) = 1$, получим

$$a_{10} = -(\mathbf{x}_1, \mathbf{w}_0).$$

Вектор $\mathbf{w}_1$ получается из вектора $\mathbf{z}_1$ нормированием к его длине. Все последующие векторы ортонормированного базиса определяются как линейные комбинации уже построенных векторов $\mathbf{w}_i$ и очередного вектора $\mathbf{x}_m$:

$$\mathbf{z}_m = \mathbf{x}_m + a_{m0}\mathbf{w}_0 + a_{m1}\mathbf{w}_1 + \ldots + a_{m,m-1}\mathbf{w}_{m-1} \ .$$

Из условий ортогональности вектора $\mathbf{z}_m$ всем уже построенным векторам $\mathbf{w}_i$ получим следующее правило определения коэффициентов:

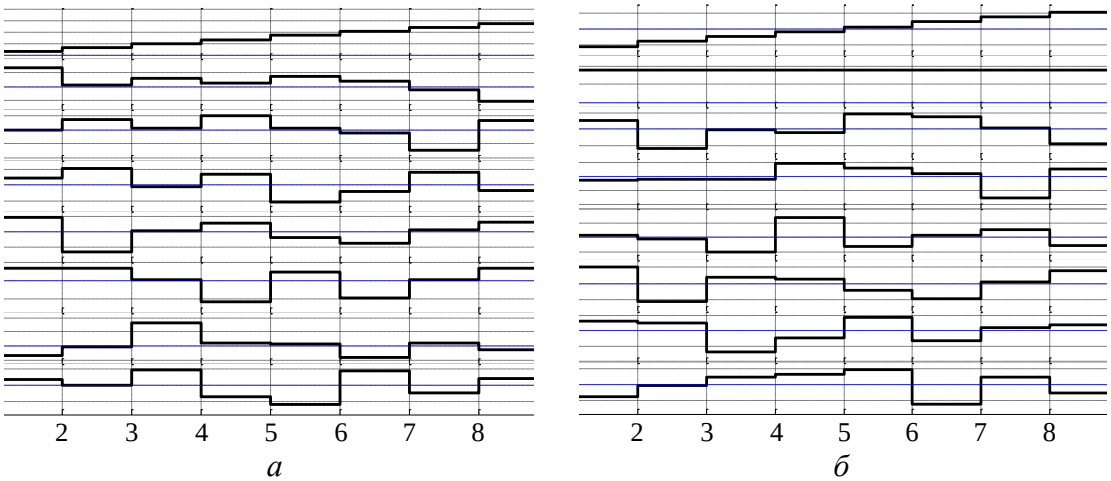
$$a_{mi} = -(\mathbf{x}_m, \mathbf{w}_i).$$

Вектор $\mathbf{w}_m$ получается из вектора $\mathbf{z}_m$ нормированием к его длине.

Остался открытым вопрос выбора системы векторов $\{\mathbf{x}_i\}$. Любая линейно независимая система векторов с начальным вектором $\mathbf{x}_0 = \mathbf{w}_0$ является подходящей, поэтому существует несчетное множество ортонормированных базисов, приспособленных к одной и той же функции. Наиболее простой вариант построения опорной системы заключается в случайной генерации очередного вектора $\mathbf{x}_m$ на каждом шаге процедуры ортогонализации. При этом, конечно, существует вероятность, что вновь сгенерированный вектор будет линейно зависим от уже построенных векторов $\mathbf{w}_i$. Тогда при применении процедуры ортогонализации получим, что очередной вектор $\mathbf{z}_m$ окажется нулевым, и на этом процедура генерации закончится. Для исключения подобной ситуации при $\mathbf{z}_m = 0$ в алгоритме реализуется повторная случайная генерация вектора $\mathbf{x}_m$, а затем процедура вычисления коэффициентов повторяется.

Для построения приспособленного ортогонального ядра перестраиваемого преобразования координатные представления ортонормированных векторов $\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{p-1}$ в арифметическом пространстве объединяются в матрицу по строкам или по столбцам. В арифметическом пространстве каждый вектор можно рассматривать как функцию от номера координаты.

На рис. 6.3, *а* показан пример генерации ортогонального базиса, приспособленного к линейно изменяющейся функции в пространстве размерности $p = 7$. Представленные функции ядра имеют ненулевое среднее.



а *б*

Рис. 6.3. Функции ортогонального приспособленного ядра: *а* – с ненулевым средним; *б* – с выделенной постоянной составляющей и нулевым средним

Ортогональные ядра с нулевым средним. При обработке сигналов часто требуется выделять постоянную составляющую, с этой целью используют базисы, в которых одна из функций постоянна по значению, а остальные имеют нулевое среднее. Такой базис приспосабливается к переменной составляющей функции. Пример генерации базиса с выделенной постоянной составляющей показан на рис. 6.3, *б*. Для размерности $p = 2$ ядра, приспособленные к функции $\mathbf{f}$, строятся наиболее просто: одна строка будет содержать нормированную постоянную составляющую, а вторая – нормированную переменную составляющую.

6.3.3. Алгоритм обучения быстрого приспособленного спектрального преобразования

Фрактальное разложение опорной функции для спектрального преобразования отличается тем, что фрактальные множители должны быть нормированы к единице. Ортогональное приспособленное преобразование не полностью использует степени свободы, которыми оно располагает. Если функция приспособления совпадает с нулевым столбцом результирующей матрицы, то будут настроены только ядра с номерами $z^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}0_{m-1}0_{m-2}\ldots 0_0\rangle$ (для топологии Кули–Тьюки «с прореживанием по времени»). На рис. 6.4 приведено факторизованное представление ортогонального преобразования, приспособленного к одной функции, размещенной в нулевом столбце. Символом «*» отмечены элементы незаполненных ядер. Если точка приспособления задана кортежем

$$x = \langle x_{n-1}x_{n-2}\ldots x_0\rangle,$$

тогда номер настраиваемых ядер по слоям будет определяться выражением

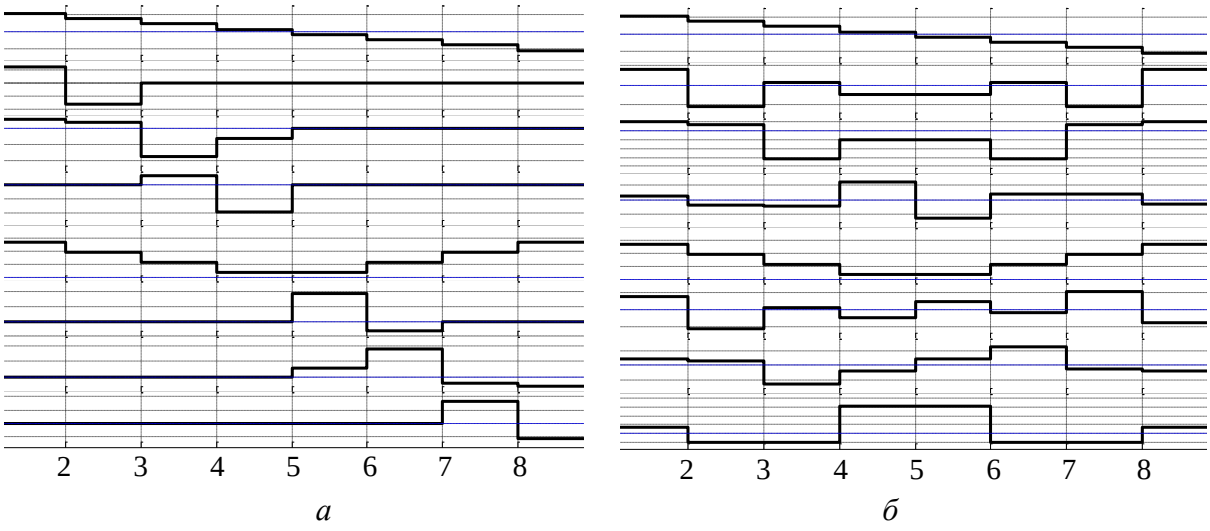
$$z^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}x_{m-1}x_{m-2}\ldots x_1x_0\rangle.$$

В каждом ортогональном ядре только один столбец совпадает с фрактальным множителем опорной функции, остальные столбцы выбираются ортогональными к нему, следуя процедуре Грамма–Шмидта, поэтому они не могут быть произвольными. Это означает, что ортогональное приспособленное преобразование может быть настроено только на одну опорную функцию (не считая постоянной составляющей).

Для того чтобы преобразование в целом было ортогональным, достаточно заполнить свободные ядра элементами любых ортогональных матриц соответствующих размеров. В практических алгоритмах можно либо заполнить свободные ядра единичными матрицами вида $W = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, либо повторить ядра, уже найденные для текущего слоя. В первом случае будут получены вейвлет-подобные ортогональные преобразования с явно выраженными свойствами пространственной локализации, во втором – все столбцы преобразования будут содержать примерно равное число ненулевых элементов.

		v_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
		v_1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
		v_0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
u_2	u_1	u_0																								
0	0	0	1	1							1		1						1				1			
0	0	1	1	1								*		*						*				*		
0	1	0			1	1					1		1								*				*	
0	1	1			1	1						*		*								*				*
1	0	0					1	1							1		1		1				1			
1	0	1					1	1								*		*		*				*		
1	1	0							1	1					1		1				*				*	
1	1	1							1	1						*		*				*				*

Рис. 6.4. Свободные параметры быстрого ортогонального преобразования, приспособленного к одной функции



а *б*

Рис. 6.5. Базисные функции приспособленных ортогональных преобразований: *а* – с доопределением ядер единичными матрицами; *б* – с приспособлением к фильтрованным образам опорных функций

Также возможно использовать более глубокое приспособление, например к фильтрованным образам опорных функций. На рис. 6.5 приведены базисные функции быстрых ортогональных преобразований, приспособленных к линейно изменяющейся опорной функции.

6.4. Квантовая нотация быстрых преобразований

Квантовые вычисления (вычисления, основанные на методах квантовой механики) были инициированы в начале 80-х гг. XX в. работами Ричарда Фейнмана. В 1994 г. Питер Шор показал, что NP-сложная задача – разложение целых чисел на множители – может быть решена на квантовом компьютере за полиномиальное время [29]. Этот результат дал толчок к бурному развитию квантовых вычислений. В августе 2000 г. было объявлено о первом практическом успехе: [Reuters] «IBM Says It Develops Most Advanced Quantum Computer («IBM – разработала первый квантовый компьютер»). В данной разработке впервые был реализован квантовый регистр на пяти атомах. Практически было доказано, что квантовый компьютер способен решать некоторые задачи гораздо быстрее, чем обычный.

Классический компьютер работает с состояниями из конечного числа бит. Регистр длиной n имеет конечное множество состояний 2^n, поскольку каждый бит имеет только два состояния. В квантовом компьютере функциональным аналогом бита является *q*-бит, который ассоциируется с двумерным комплексным пространством. В пространстве *q*-бита выделены два базисных вектора $|0\rangle$ и $|1\rangle$. Кроме базисных состояний, для *q*-бита возможны также состояния, представляющие собой любые линейные комбинации базисных векторов: $\varphi = \alpha|0\rangle + \beta|1\rangle$, где $\alpha^2 + \beta^2 = 1$. Квантовый регистр представляет собой конечный набор *q*-бит. Базисными состояниями квантового регистра являются тензорные произведения базисных состояний входящих в него *q*-бит. Для обозначения тензорных произведений в квантовой механике используются сокращения:

$$\left|x_{n-1}\right\rangle \otimes \left|x_{n-2}\right\rangle \otimes \ldots \otimes \left|x_0\right\rangle = \left|x_{n-1}x_{n-2}\ldots x_0\right\rangle.$$

Согласно общим принципам квантовой механики, возможным состоянием квантового регистра длиной n может быть любой вектор единичной длины в комплексном пространстве размерностью 2^n. Квантовое состояние регистра записывается в виде

$$\left|\varphi\right\rangle = \sum_{|x\rangle \in \left|x_{n-1}x_{n-2}\ldots x_0\right\rangle} a_x |x\rangle,$$

где суммирование производится по всем возможным базисным состояниям. Для вектора единичной длины сумма квадратов коэффициентов a_x всегда равна единице. Квантовые состояния в комплексном пространстве задаются с точностью до фазового множителя $e^{j\varphi}$. Квадрат значения каждого коэффициента a_x интерпретируется как вероятность нахождения квантового регистра в данном базовом состоянии. Состояние квантового регистра называется запутанным, если его невозможно представить в виде произведения квантовых состояний входящих в него q-бит.

Эволюция состояний квантового регистра во времени описывается уравнением Шредингера, решением которого являются унитарные преобразования размерностью 2^n. При квантовых вычислениях в q-регистре приготавливается начальное состояние в виде базовых состояний q-бит. Затем используется некоторое унитарное преобразование над отдельными q-битами, зависящее от решаемой задачи, в итоге возникает новое состояние квантового регистра. Результатом вычислительной операции считается состояние квантового регистра, которое имеет вероятность, достаточно близкую к единице. Эффективность квантовых вычислений достигается за счет того, что одновременно с унитарным преобразованием над состояниями q-бита «бесплатно» выполняется тензорное произведение, экспоненциально расширяющее область действия унитарного преобразования. Наибольший эффект достигается для факторизуемых квантовых состояний, однако и для широкого класса запутанных состояний возможна реализация полиномиально эффективных алгоритмов.

Элементарная квантовая операция на регистре определяется двумерным унитарным преобразованием, действующим в пространстве одного q-бита. Данное преобразование (унарный вентиль или гейт) задается комплекснозначной матрицей размером 2×2:

$$W = \begin{array}{c|cc} & |0\rangle & |1\rangle \\ \hline |0\rangle & w_{00} & w_{01} \\ |1\rangle & w_{10} & w_{11} \end{array}.$$

В канонической форме унитарная матрица 2-го порядка может быть записана в виде:

$$W = \begin{pmatrix} e^{j\gamma_{00}}\cos\theta & e^{j\gamma_{01}}\sin\theta \\ e^{j\gamma_{10}}\sin\theta & -e^{j\gamma_{11}}\cos\theta \end{pmatrix}.$$

Условие унитарности в этом случае сводится к соотношению между углами комплексных экспонент:

$$\gamma_{00} - \gamma_{01} = \gamma_{10} - \gamma_{11}. \qquad (6.6)$$

Бинарные вентили задаются унитарными матрицами размером 4×4. Примером может служить элемент «управляемый фазовый сдвиг» [30], задаваемый матрицей

$$W = \begin{array}{c|cccc} & |00\rangle & |01\rangle & |10\rangle & |11\rangle \\ \hline |00\rangle & 1 & 0 & 0 & 0 \\ |01\rangle & 0 & 1 & 0 & 0 \\ |10\rangle & 0 & 0 & 1 & 0 \\ |11\rangle & 0 & 0 & 0 & e^{j\varphi} \end{array}.$$

Можно показать, что результирующее состояние регистра после бинарного вентиля является запутанным. Последовательная комбинация элементарных преобразований в принципе позволяет выполнить любое квантовое вычисление. Существенным вопросом является эффективность вычислений. В работе [29] для построения эффективного квантового вычисления Шор использовал модификацию быстрого алгоритма Фурье с основанием 2. Как было показано ранее, быстрое преобразование Фурье может быть реализовано с помощью быстрых перестраиваемых преобразований. В общем случае быстрые перестраиваемые преобразования могут быть использованы для формирования широкого набора унитарных преобразований. Применительно к квантовым вычислениям размерность преобразования должна быть равна степени двойки. Это ведет к тому, что все ядра преобразования структурно-подобны и представляют собой унитарные матрицы размером 2×2. Когда все ядра в пределах слоя параметрически одинаковы, то каждый слой преобразования однозначно сопоставляется с одним q-битом квантового регистра. В этом случае эквивалентный унитарный оператор, который описывает эволюцию состояния q-битного регистра размерностью n, выражается кронекеровским произведением матриц ядер: $H = W_0 \times W_1 \times \ldots \times W_{n-1}$. На рис. 6.6 показана квантовая схема подобного вычисления. В ней число квантовых операций равно размерности квантового регистра, поэтому данное преоб-

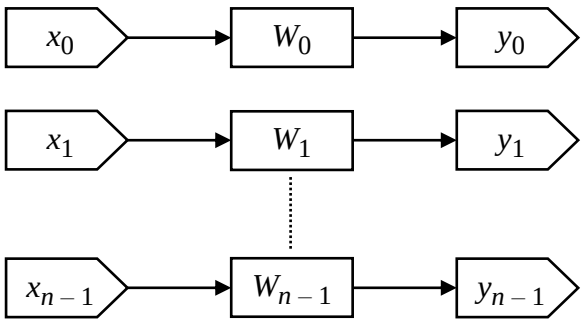


Рис. 6.6. Квантовая схема БНС с однородными ядрами в слое

разование является максимально эффективным. Результирующее состояние для данной квантовой схемы всегда факторизуемо.

В общем случае унитарное преобразование выполняет трансформацию входного вектора x в выходной вектор y:

$$y(V) = \sum_{U} x(U) h(U,V).$$

В квантовой нотации векторы входа и выхода представляются в форме:

$$|x\rangle = |x_{n-1}x_{n-2}\ldots x_0\rangle = \sum_{|u\rangle} x(u)|u_{n-1}u_{n-2}\ldots u_0\rangle,$$

$$|y\rangle = |y_{n-1}y_{n-2}\cdots y_0\rangle = \sum_{|v\rangle} y(v)|v_{n-1}v_{n-2}\cdots v_0\rangle,$$

где n – размерность q-битного регистра, $N = 2^n$ определяет размерность преобразования. Если преобразование является быстрым, то оно может быть выражено через элементы ядер:

$$h(U,V) = w^{n-1}_{z^{n-1}}(u_{n-1}, v_{n-1}) w^{n-2}_{z^{n-2}}(u_{n-2}, v_{n-2}) \cdots w^{0}_{z^{0}}(u_0, v_0),$$

где $z^m = \langle u_{n-1}u_{n-2}\cdots u_{m+1}x_{m-1}x_{m-2}\cdots x_1x_0\rangle$ – номер нейронного ядра в слое m или номер гейта в квантовой нотации. Для кубитного регистра все числа p_m равны 2. Обобщением понятия «кубит» является кудит (Q-энк, куэнк; qudit), который ассоциируется с комплексным пространством размерности p_m, где p_m – произвольные натуральные числа. В этом случае $N = p_0 p_1 \ldots p_{n-1}$

Технология классификации обычно имеет дело с образами. В квантовом варианте образом является квантовое состояние регистра. Задача распознавания образа сводится к построению такого унитарного преобразования, которое формирует на выходе одно из базисных состояний регистра с вероятностью, близкой или равной единице. По существу, подобная задача была рассмотрена при построении приспособленных спектральных преобразований. Было показано, что настройка сводится к фрактальной декомпозиции функции и выбору параметров ядер преобразования. Методы декомпозиции основаны на фрактальной фильтрации и отличаются только комплексным характером фрактальных фильтров.

Для размерности 2 можно предложить более общий способ выбора параметров ядер, чем было изложено ранее. Обозначим через $\dot{w}_i(u,v)$ коэффициенты ядра в слое i, а через $\dot{\varphi}_i(u)$ (здесь и далее точка над символом означает комплексную величину, а «черточка» – комплексно-сопряженную) – соответ-

ствующую компоненту фрактального разложения функции-образа $\dot{f}(u)$. Значениями функции $\dot{f}(u)$, заданной на интервале длиной 2^n, являются коэффициенты базисных векторов состояния квантового регистра. Как было показано ранее, спектральный оператор будет приспособлен к функции, если ядра будут приспособлены к соответствующим компонентам фрактального разложения функции. Для унитарных ядер условие приспособленности имеет вид:

$$\begin{cases} \sum\limits_{u_m} \dot{w}_{i^m}(u_m, v_m)\overline{\varphi}_{i^m}(u_m) = 1 \quad \text{при} \quad v_m = x_m, \\ \sum\limits_{u_m} \dot{w}_{i^m}(u_m, v_m)\overline{\varphi}_{i^m}(u_m) = 0 \quad \text{при} \quad v_m \neq x_m. \end{cases}$$

Например, для компоненты $\dot{\varphi}_{i^m}(u_m) = \varphi_{i^m}(u_m)e^{j\alpha_{u_m}}$ и ядра вида (6.6), приспособленного по первому столбцу, данные условия приводятся к виду:

$$\varphi_{i^m}(0)e^{j(\gamma_{00}-\alpha_0)}\cos\theta + \varphi_{i^m}(1)e^{j(\gamma_{10}-\alpha_1)}\sin\theta = 1,$$

$$\varphi_{i^m}(0)e^{j(\gamma_{01}-\alpha_0)}\sin\theta - \varphi_{i^m}(1)e^{j(\gamma_{11}-\alpha_1)}\cos\theta = 0.$$

Отсюда следуют расчетные формулы для определения параметров ядер:

$$\operatorname{tg}\theta = \varphi_{i^m}(1)\big/\varphi_{i^m}(0), \qquad \gamma_{00} = \alpha_0, \quad \gamma_{10} = \alpha_1, \quad \gamma_{01} - \alpha_0 = \gamma_{11} - \alpha_1.$$

Углы γ_{01}, γ_{11} данными формулами определяются неоднозначно, для определенности можно положить $\gamma_{01} = \alpha_0$, $\gamma_{11} = \alpha_1$. Если исходное состояние квантового регистра совпадает с функцией образа, то квантовое состояние после воздействия оператора быстрого преобразования будет точно совпадать с одним из базовых состояний.

7. НАСТРОЙКА ДВУМЕРНЫХ БЫСТРЫХ ПЕРЕСТРАИВАЕМЫХ ПРЕОБРАЗОВАНИЙ

Традиционно для обработки изображений используются быстрые ортогональные преобразования. Цель обработки обычно заключается в фильтрации или сжатии изображения. В обоих случаях необходимы некоторые априорные знания, которые касаются либо вида изображения, либо помехи. В зависимости от этой информации выбирается тип используемого преобразования. Например, если известны пространственные частоты помехи, то применяется преобразование Фурье и фильтрация помехи производится в частотной области. Для решения задачи сжатия необходима статистическая информация о классе изображений, для того чтобы без особых потерь для восприятия удалить малозначимые компоненты сжимаемых отображений. Для сжатия обычно применяются ортогональные косинусные и вейвлет-преобразования, базисные функции которых близки к собственным векторам ковариационных матриц широкого класса изображений. Сами собственные векторы, упорядоченные в матрицу, образуют ортогональное преобразование Карунена–Лоэва. К сожалению, это преобразование не имеет быстрого алгоритма и поэтому не используется при больших размерностях данных. Однако его важной отличительной особенностью является возможность обучения по статистически накопленным данным.

Подобное качество можно распространить и на быстрые преобразования, поставив перед собой цель настроить значения коэффициентов базовых операций таким образом, чтобы учесть априорную информацию и сохранить при этом условие ортогональности. Быстрые преобразования имеют меньшее число степеней свободы, чем преобразование Карунена–Лоэва, поэтому возможности обучения будут ограничены. Тем не менее их достаточно для настройки, по крайней мере, на одну главную компоненту преобразования Карунена–Лоэва, и уже это позволяет решать целый класс задач, связанных с эффективным сжатием, адаптивной фильтрацией и распознаванием изображений.

7.1. Двумерные быстрые преобразования

Обозначим через $F\left(U_y,\ U_x\right)$ матрицу изображения размером $N_y \times N_x$. При воздействии на изображение линейного преобразования $H\left(U_y, U_x; V_y, V_x\right)$ получается массив из $M_y \times M_x$ коэффициентов. Двумерное преобразование выполняется по правилу

$$S(V_y, V_x) = \sum_{U_y=0}^{N_y-1} \sum_{U_x=0}^{N_x-1} F(U_y, U_x) H(U_y, U_x; V_y, V_x). \qquad (7.1)$$

Двумерное преобразование называется ортогональным (точнее, унитарным), если выполняется условие

$$\sum_{Uy=0}^{Ny-1} \sum_{Ux=0}^{Nx-1} H(U_y, U_x; V_y, V_x) \bar{H}(U_y, U_x; V_y', V_x') = \begin{cases} 1 \text{ если } V_y = V_y' \text{ и } V_x = V_x'; \\ 0 \text{ если } V_y \neq V_y' \text{ или } V_x \neq V_x', \end{cases} \qquad (7.2)$$

где символом надчеркивания «–» помечено комплексно-сопряженное преобразование. Для ортогонального преобразования выполняется: $N_y = M_y$, $N_x = M_x$. Рассмотренная ранее парадигма быстрых перестраиваемых преобразований достаточно просто распространяется на многомерное пространство.

Необходимым условием существования быстрого алгоритма является возможность мультипликативной декомпозиции значений обоих размерностей изображения в равное число сомножителей:

$$N_y = p_0^y p_1^y \dots p_{n-1}^y,$$

$$N_x = p_0^x p_1^x \dots p_{n-1}^x.$$

Индексы x, y здесь означают принадлежность к осям координат исходного изображения. Данное условие не является жестким ограничением, поскольку некоторые сомножители могут быть единичными. Тем не менее, чем большее число неединичных сомножителей содержат декомпозиции, тем выше вычислительная эффективность быстрого алгоритма. Используя сомножители декомпозиций как основания системы счисления, представим значения координат точек изображения в позиционной форме:

$$U_y = \left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y u_0^y \right\rangle, \quad U_x = \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x u_0^x \right\rangle, \qquad (7.3)$$

где вес m-го разряда определяется выражением $p_{m-1}^* p_{m-2}^* \dots p_1^* p_0^*$, а u_m^* – разрядная переменная, принимающая значения $\left[0, p_m^* - 1\right]$ (звездочка здесь заменяет индексы x, y). Аналогично можно выразить координаты спектральных коэффициентов в плоскости $\left[V_y, V_x\right]$:

$$V_y = \left\langle v_{n-1}^y v_{n-2}^y \cdots v_1^y v_0^y \right\rangle, \quad V_x = \left\langle v_{n-1}^x v_{n-2}^x \cdots v_1^x v_0^x \right\rangle.$$

Алгоритм быстрого преобразования обычно представляется в виде графа с различной топологией. Поразрядную форму удобно использовать для описаний топологий быстрого алгоритма. Например, для топологии Кули–Тьюки «с прореживанием по времени» граф может быть описан в виде лингвистического предложения

$$\left[\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_1u^*_0\right\rangle\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_1v^*_0\right\rangle\ldots\right.$$
$$\left.\ldots\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_{m+1}u^*_mv^*_{m-1}v^*_{m-2}\ldots v^*_0\right\rangle\ldots\left\langle v^*_{n-1}v^*_{n-2}\ldots v^*_1v^*_0\right\rangle\right].$$

Первое и последнее слова предложения соответствуют координатам значений в пространственной и спектральной областях. Число слов в предложении равно $n+1$. Промежуточные слова определяют координаты U_y^m, U_x^m и V_y^m, V_x^m для точек преобразуемого изображения во внутренних слоях быстрого алгоритма. Для алгоритма с замещением значений выполняется условие

$$U_y^{m+1}=V_y^m,\quad U_x^{m+1}=V_x^m. \tag{7.4}$$

По топологическому предложению несложно построить граф быстрого преобразования. В общем случае топологии для направлений x и y могут быть разными, но правило их построения всегда должно удовлетворять системному инварианту быстрых алгоритмов. На графе топологии можно выделить базовые операции (ядра) $W^m_{z_x^m,\,z_y^m}\left(u_m^y u_m^x; v_m^y v_m^x\right)$, представляющие собой четырехмерные матрицы размером $\left[p_m^y,\, p_m^x;\, p_m^y,\, p_m^x\right]$ ($m=0,1,2,\ldots,n-1$). Взаимосвязь между базовыми операциями определяется фактор-графом – структурной моделью быстрого преобразования. Для выбранной топологии граф структурной модели описывается лингвистическим предложением

$$\left[\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_1\right\rangle\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_2v^*_0\right\rangle\ldots\right.$$
$$\left.\ldots\left\langle u^*_{n-1}u^*_{n-2}\ldots u^*_{m+1}v^*_{m-1}v^*_{m-2}\ldots v^*_0\right\rangle\ldots\left\langle v^*_{n-2}v^*_{n-3}\ldots v^*_1v^*_0\right\rangle\right].$$

Каждое слово в этом предложении определяет номер базовой операции i^m_* в слое m. Число слов в предложении равно n. Хотя порядок символов в словах подобен выбранной топологической модели, но это подобие весьма условно, по любой структурной модели можно построить множество различных топологий. На рис. 7.1 показана структурная модель быстрого двумерно-

го преобразования для размерности изображения 8×8. Входное изображение поступает на нижний слой, а спектральные коэффициенты получаются в верхнем слое. Вершинам модели соответствуют ядра преобразования. Ядро в слое m выполняет двумерное преобразование над пространственным блоком размером $p_m^y \times p_m^x$:

$$S^m\left(V_y^m, V_x^m\right) = \sum_{u_m^y}\sum_{u_m^x} F^m\left(U_y^m, U_x^m\right) W^m_{z_x^m, z_y^m}\left(u_m^y u_m^x; v_m^y v_m^x\right). \qquad (7.5)$$

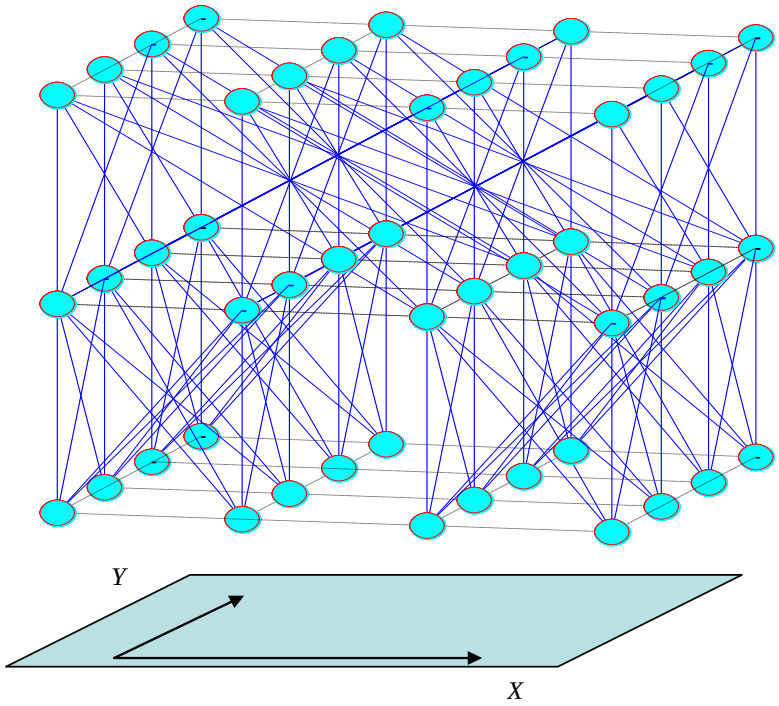


Рис. 7.1. Структурная модель двумерного быстрого преобразования

Соответствия $U_*^m \leftrightarrow \left(z_*^m, u_*^m\right)$ взаимно-однозначно определяются по лингвистическим предложениям топологической и структурной модели. При компактной топологии в межслойном переходе устанавливаются соответствия $U_x^{m+1} = V_x^m$, $U_y^{m+1} = V_y^m$. Алгоритм быстрого двумерного преобразования заключается в последовательном вычислении выражений (7.5) по всем возможным слоям.

7.2. Вычислительная эффективность быстрого двумерного преобразования

Оценим вычислительную эффективность быстрого алгоритма для ортогонального двумерного преобразования. В качестве базы для сравнения будем использовать вычислительные затраты при прямом выполнении преобра-

зования, которые, как и прежде, станем оценивать числом операций умножения. Полное число операций умножения равно числу элементов четырехмерной матрицы преобразования. При размерностях изображения $N_x \times N_y$ это значение будет равно

$$Z_0 = \left(N_x N_y\right)^2 .$$

Для быстрого алгоритма (см. (6.5)) количество ядер в слое m:

$$k_m = \left(N_x \big/ p_m^x\right)\left(N_y \big/ p_m^y\right).$$

Для каждого ядра в слое требуется выполнить $z_m = \left(p_m^x p_m^y\right)^2$ операций умножения. При числе слоев, равном n, общее число операций умножения для быстрого алгоритма

$$Z = \sum_{m=0}^{n-1} k_m z_m .$$

Выигрыш по быстродействию будем оценивать величиной $\theta = Z_0 / Z$. Подставив в эту оценку значения затрат, получим:

$$\theta = \frac{N_x N_y}{\sum\limits_{m=0}^{n-1} p_m^x p_m^y}.$$

Для численной оценки выполним расчет для изображений, размеры которых кратны степени двойки. Размеры ядер в этом случае целесообразно выбрать равными двум. Полагая $N_x = N_y = N$, получим $n = \log_2 N$, тогда

$$\theta = \frac{N^2}{4\log_2 N}.$$

Например, для $N = 128 = 2^7$ получим $\theta \approx 585$, а для $N = 256 = 2^8$ будем иметь $\theta = 2048$. Вычислительная эффективность быстрого алгоритма быстро растет с увеличением размерности изображения.

7.3. Мультипликативная декомпозиция элементов матрицы быстрого преобразования

В гл. 4 было построено мультипликативное разложение элементов матрицы быстрого преобразования для одномерного случая. Покажем, что подобное разложение существует и в двумерном варианте.

Задание конкретных значений для всех разрядных переменных u_m^*, v_m^* (где m пробегает значения $0,1,\ldots,n-1$) определяет некоторый путь в топологическом графе между парой элементов начального и конечного слоев. Из однозначности поразрядного представления чисел следует, что такой путь единствен для каждого парного сочетания пространственных точек входного и выходного слоев. Из выражения (7.1) следует, что

$$H\left(U_y,U_x;V_y,V_x\right)=\frac{\partial S\left(V_y,V_x\right)}{\partial F\left(U_y,U_x\right)}. \tag{7.6}$$

Дифференцируя (7.6) по правилу дифференцирования сложной функции получим:

$$H\left(U_y,U_x;V_y,V_x\right)=\frac{\partial S^{n-1}}{\partial F^{n-1}}\frac{\partial F^{n-1}}{\partial S^{n-2}}\frac{\partial S^{n-2}}{\partial F^{n-2}}\cdots\frac{\partial F^{1}}{\partial S^{0}}\frac{\partial S^{0}}{\partial F^{0}}.$$

Из условия (7.4) следует, что для всех m справедливо $\frac{\partial F^m}{\partial S^{m-1}}=1$, а из (7.4) – что $\frac{\partial S^m}{\partial F^m}=W^m_{z_x^m,\,z_y^m}\left(u_m^y u_m^x; v_m^y v_m^x\right)$. Таким образом получим, что каждый элемент четырехмерной матрицы преобразования H выражается через элементы ядер в виде произведения:

$$H\left(U_y,U_x;V_y,V_x\right)=W^{n-1}_{z_x^{n-1},\,z_y^{n-1}}\left(u_{n-1}^y u_{n-1}^x; v_{n-1}^y v_{n-1}^x\right)\ldots W^0_{z_x^0,z_y^0}\left(u_0^y u_0^x; v_0^y v_0^x\right), \tag{7.7}$$

где поразрядные выражения индексов ядер слоя m для выбранной топологии имеют вид:

$$\begin{aligned} z_x^m &= \left\langle u_{n-1}^x u_{n-2}^x \ldots u_{m+1}^x v_{m-1}^x v_{m-2}^x \ldots v_0^x \right\rangle, \\ z_y^m &= \left\langle u_{n-1}^y u_{n-2}^y \ldots u_{m+1}^y v_{m-1}^y v_{m-2}^y \ldots v_0^y \right\rangle. \end{aligned} \tag{7.8}$$

Подставив выражение (7.7) в условие ортогональности (7.2), после преобразований получим, что оно выполняется, когда ортогональными являются все ядра, т. е. для любых m, z_x^m, z_y^m справедливо:

$$\sum_{u_m^y}\sum_{u_m^x} W^m_{z_x^m,\,z_y^m}\left(u_m^y u_m^x; v_m^y v_m^x\right)\bar{W}^m_{z_x^m,\,z_y^m}\left(u_m^y u_m^x; \grave{v}_m^y \grave{v}_m^x\right)=\begin{cases}1, \text{ если } v_m^y=\grave{v}_m^y \text{ и } v_m^x=\grave{v}_m^x,\\ 0, \text{ если } v_m^y\neq\grave{v}_m^y \text{ или } v_m^x\neq\grave{v}_m^x.\end{cases}$$

7.4. Мультипликативное представление изображений

В гл. 4 было показано, что последовательная фрактальная фильтрация служит удобным инструментом для настройки одномерных быстрых приспособленных преобразований. На каждом шаге фрактальный фильтр кратно сокращает размер временной базы одномерного сигнала. Подобным образом для двумерного случая фрактальная фильтрация представляет собой кратномасштабную обработку изображений, последовательно сжимающую его размеры вплоть до единственной точки. Схему последовательной фрактальной фильтрации изображений можно представить в виде пирамиды (рис. 7.2). Основанием пирамиды является исходное изображение $F\left(U_y, U_x\right)$, для которого аргументы U_y и U_x представлены в позиционной системе счисления (см. (7.3)).

Для выполнения первого шага зафиксируем в этом позиционном представлении все разряды, кроме двух младших u_0^y и u_0^x. Если варьировать младшие разряды по всем возможным значениям, то получим выборку размером $p_0^y \times p_0^x$. Под фрактальным фильтром будем понимать произвольный функционал Φ, заданный на этой выборке. Формально это можно записать в виде выражения

$$F_1\left(\left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y \right\rangle, \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x \right\rangle\right) =$$
$$= \underset{\left(u_0^y, u_0^x\right)}{\Phi}\left[F\left(\left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y u_0^y \right\rangle, \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x u_0^x \right\rangle\right)\right].$$

Очевидно, что изображение F_1 будет кратно уменьшенным по размерам по отношению к исходному изображению. Функционалом, например, может быть правило вычисления среднего значения выборки или ее медианы. Результат фильтрации можно восстановить до исходного размера. Если выполнить скалярное деление исходного изображения на восстановленный фильтрованный образ, то получим первый множитель искомой декомпозиции. После первого шага фрактальной фильтрации исходное изображение можно формально представить в виде произведения

$$F\left(\left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y u_0^y \right\rangle, \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x \right\rangle\right) =$$
$$= F_1\left(\left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y \right\rangle, \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x \right\rangle\right) f_{i_0^y, i_0^x}\left(u_0^y, u_0^x\right),$$

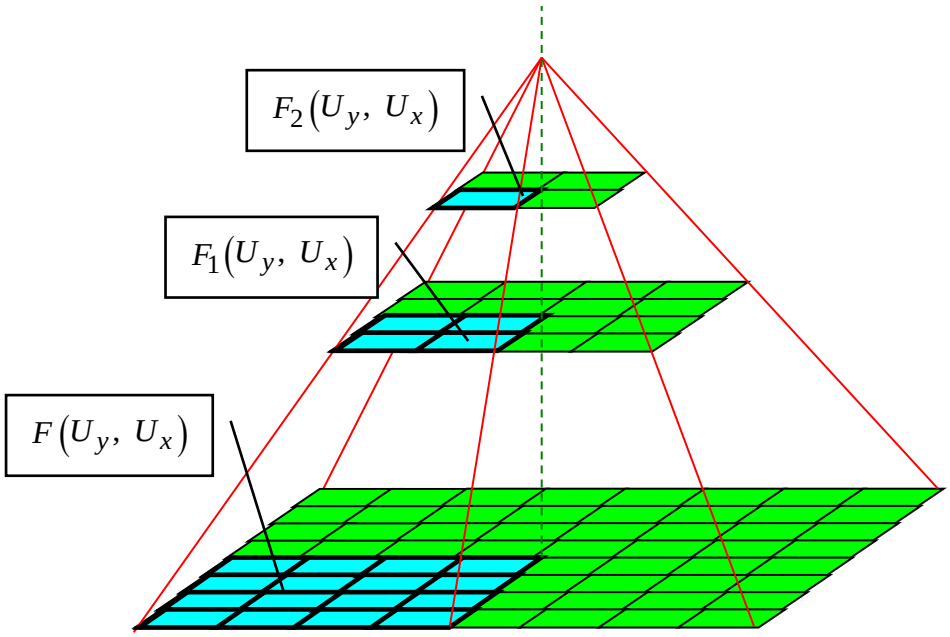


Рис. 7.2. Схема фрактальной фильтрации изображения

где $f_{i_0^y i_0^x}\left(u_0^y, u_0^x\right)$ – набор двумерных функций-множителей, зависящих от разрядных переменных u_0^y и u_0^x, а индексы i_0^y, i_0^x выделяют функцию из набора. Значение этих индексов устанавливается равным значениям аргументов изображения F_1, т. е. $i_0^y = \left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y \right\rangle$ и $i_0^x = \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x \right\rangle$. Изображение F_1 в свою очередь также может быть представлено как произведение изображения F_2 на множители из набора $f_{i_1^y i_1^x}\left(u_1^y, u_1^x\right)$. Повторяя операцию фрактальной фильтрации и разложения многократно, достигнем вершины пирамиды и в результате получим мультипликативную декомпозицию изображения в виде:

$$F\left(\left\langle u_{n-1}^y u_{n-2}^y \cdots u_1^y u_0^y \right\rangle, \left\langle u_{n-1}^x u_{n-2}^x \cdots u_1^x u_0^x \right\rangle\right) =$$

$$= f_{i_{n-1}^y i_{n-1}^x}\left(u_{n-1}^y, u_{n-1}^x\right) f_{i_{n-2}^y i_{n-2}^x}\left(u_{n-2}^y, u_{n-2}^x\right) \ldots$$

$$\ldots f_{i_1^y i_1^x}\left(u_1^y, u_1^x\right) f_{i_0^y i_0^x}\left(u_0^y, u_0^x\right), \qquad (7.9)$$

где $i_m^y = \left\langle u_{n-1}^y u_{n-2}^y \cdots u_{m+1}^y \right\rangle$ и $i_0^x = \left\langle u_{n-1}^x u_{n-2}^x \cdots u_{m+1}^x \right\rangle$. При использовании мультипликативной декомпозиции для построения ортогональных преобразований необходимо при разложении двумерной функции нормировать формируемые множители к уровню единичной энергии.

7.5. Алгоритм обучения приспособленных быстрых двумерных преобразований

Предположим, что точка приспособления задана кортежем

$$s^* = \left\langle s^*_{n-1} s^*_{n-2} \cdots s^*_0 \right\rangle,$$

где $(*)$ обозначает пространственную координату x или y. Тогда номера настраиваемых ядер по слоям будут определяться выражениями:

$$z^m_x = \left\langle u^x_{n-1} u^x_{n-2} \ldots u^x_{m+1} s^x_{m-1} s^x_{m-2} \ldots s^x_1 s^x_0 \right\rangle,$$

$$z^m_y = \left\langle u^y_{n-1} u^y_{n-2} \ldots u^y_{m+1} s^y_{m-1} s^y_{m-2} \ldots s^y_1 s^y_0 \right\rangle.$$

Сравнивая выражения (7.7) и (7.9), получим общее правило настройки нейронных ядер на заданный образ:

$$W^m_{z^m_x,\, z^m_y}\left(u^y_m u^x_m; s^y_m s^x_m\right) = f^k_{i^m_x, i^m_y}\left(u^y_m, u^x_m\right), \tag{7.10}$$

где k – номер эталонного образа. Преобразование можно настроить к $g^y_0 \times g^x_0$ числу образов. Взаимно-однозначное соответствие $\left(s^x_0, s^y_0\right) \leftrightarrow k$ задает упорядочение образов по точкам приспособления на выходной плоскости.

Множество индексов ядер (7.8) в каждом слое превышает множество индексов функций-множителей разложения эталонного образа. Ядра двумерного преобразования, не использованные для настройки, заполняются по тем же принципам, что и для одномерного преобразования (см. 6.2).

7.6. Настройка быстрых двумерных спектральных преобразований

Так же, как и для одномерного преобразования, настройка сводится к построению приспособленных ортогональных ядер. Фактически для каждой двумерной функции $f_{i^m_x i^m_y}\left(u^y_m, u^x_m\right)$ мультипликативной декомпозиции изображения необходимо достроить ортогональные функции до полного набора двумерного ортогонального ядра. Это можно сделать, используя описанную в 6.3.2 процедуру Грамма–Шмидта. Ядра преобразования, не использованные при настройке, заполняются по тем же принципам, что и для одномерного спектрального преобразования (см. 6.3.3).

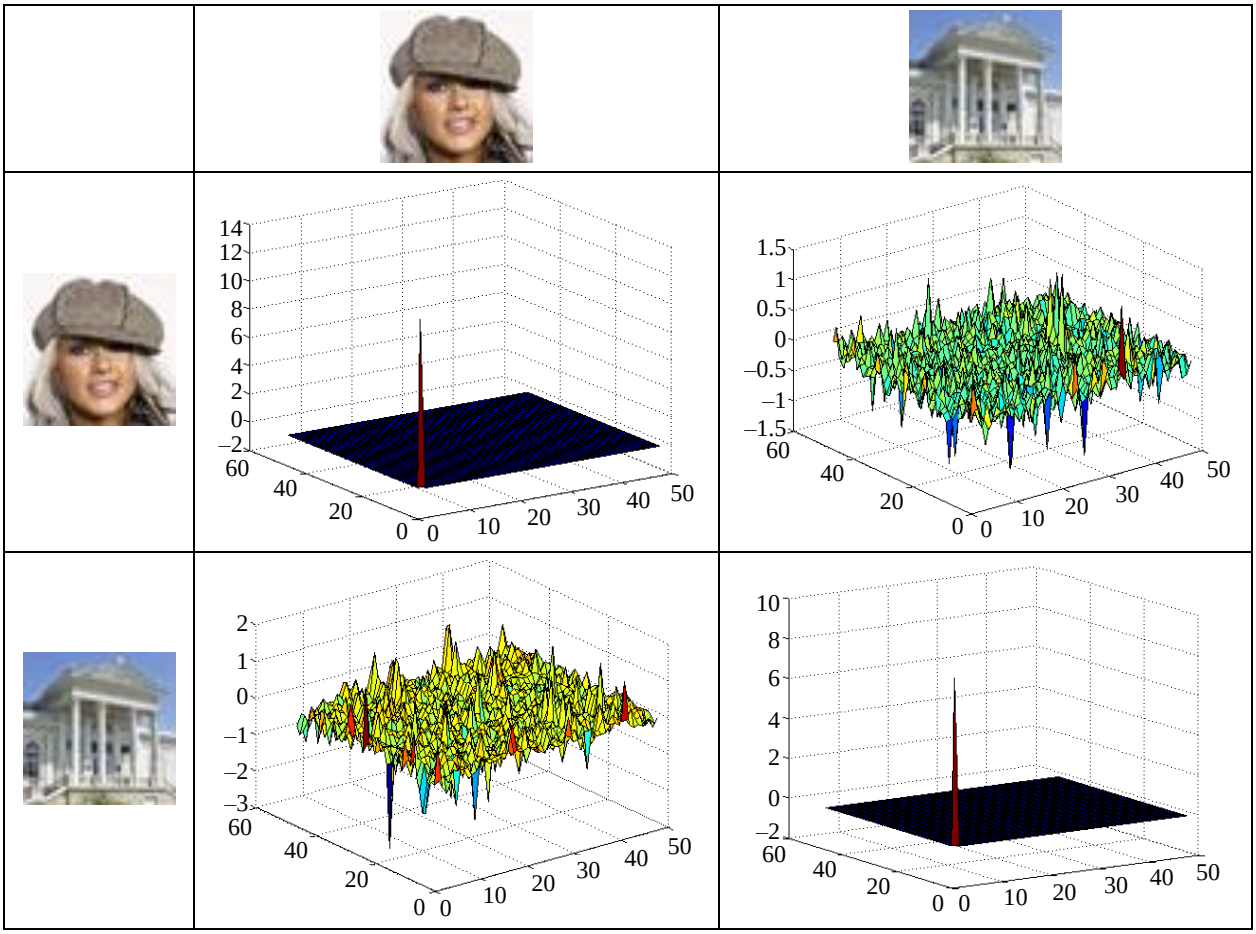


Рис. 7.3. Обработка изображений приспособленными двумерными преобразованиями

На рис. 7.3 представлены результаты экспериментов с приспособленными спектральными преобразованиями для двух изображений. Цветные изображения преобразовывались к черно-белому варианту и из них вычиталась постоянная составляющая. Для каждого изображения было построено быстрое приспособленное преобразование. Это преобразование использовалось для обработки собственного и чужого изображений. На рисунке представлены двумерные спектры для обоих случаев.

7.7. Построение объемных фракталов

Двумерные быстрые преобразования наследуют фрактальные свойства одномерных преобразований, поэтому их можно использовать для построения регулярных фракталов. Задача эта существенно проще, чем настройка приспособленного преобразования, поскольку для построения регулярного фрактала все ядра преобразования должны быть выбраны одинаковыми.

Губка Кантора. Фрактал является обобщением фрактала Кантора на трехмерный случай. В качестве ядра следует выбрать трехмерную матрицу размером $3\times3\times3$. Рис. 7.4 отображает графический образ матрицы. Элементы матрицы принимают значения 0 и 1. Для построения фрактальной итерации

необходимо по элементам ядер быстрого преобразования восстановить значения матрицы преобразования. Для этой цели можно воспользоваться выражением (7.7), определяющим декомпозицию элементов матрицы в произведение элементов ядер.

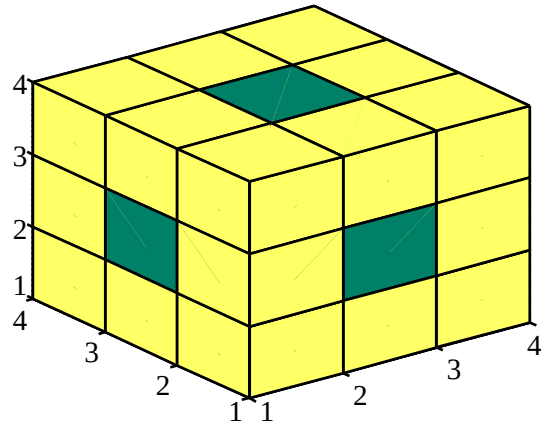


Рис. 7.4. Трехмерное ядро для 3D-фрактала Кантора

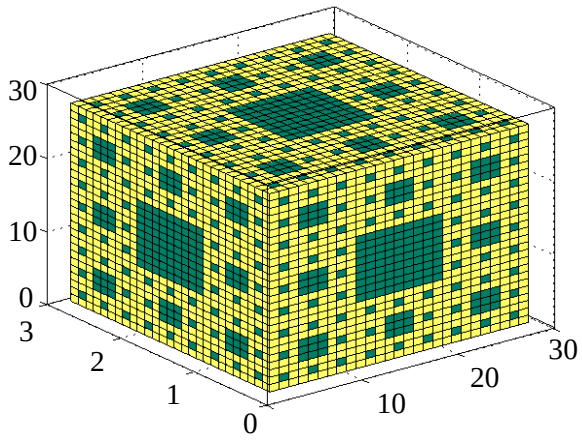
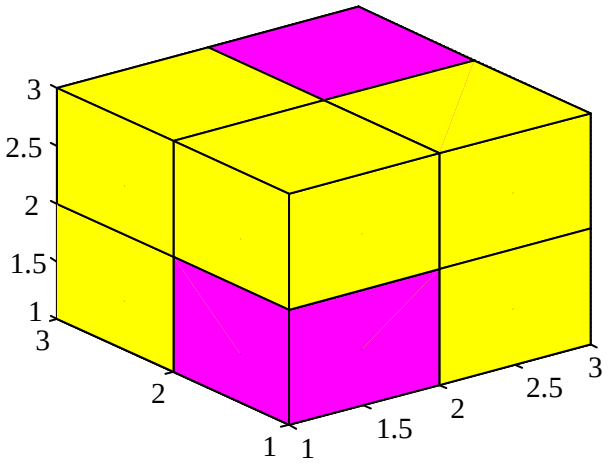


Рис. 7.5. 3D-фрактал Кантора

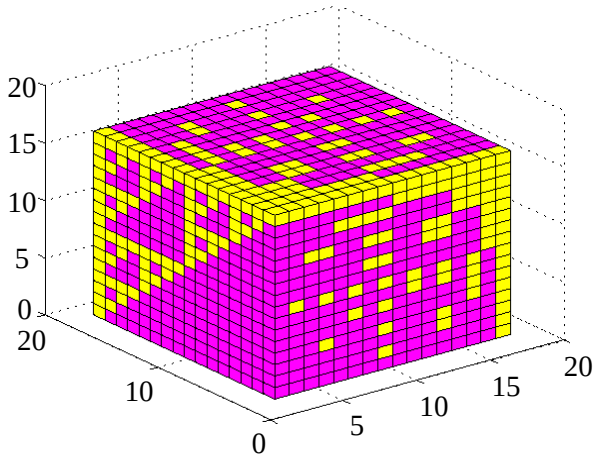
Рис. 7.5 демонстрирует графический образ матрицы преобразования, построенной по данному алгоритму для предфрактала Кантора 3-го порядка. Размер сформированной матрицы $3^3 \times 3^3 \times 3^3$. В отличие от спектрального преобразования, основания разрядных переменных по одной из координат спектральной области выбираются единичными, в частности, для рассмотренного случая было выбрано

$$P_x = P_y = G_y = [3\ 3\ 3]; \qquad G_x = [1\ 1\ 1];$$

Это первый случай, когда приходится специально выбирать основания системы счисления для получения преобразований с заданными свойствами.



а



б

Рис. 7.6. 3D-ядро и фрактал Серпинского

Губка Серпинского. Трехмерный фрактал Серпинского строится на пространственной базе, кратной степени двойки. Рис. 7.6, *а* представляет графический образ ядра, используемого для построения фрактала. Элементы матрицы ядра принимают значения 0 и 1. Ядро имеет размеры $2 \times 2 \times 2$. Рис. 7.6, *б* представляет четвертую 3D-итерацию фрактала Серпинского.

8. ПИРАМИДАЛЬНЫЕ НЕЙРОННЫЕ СЕТИ БЫСТРОГО ОБУЧЕНИЯ

В работе [31] была предложена архитектура пирамидальной нейронной сети PyraNet, предназначенной для визуального распознавания образов. Новый вид нейронной сети был мотивирован пирамидами изображений, которые успешно использовались в задачах обработки изображений [32]. Пирамиды изображений имеют давнюю связь с машинным обучением и компьютерным зрением. Было предложено несколько моделей, основанных на концепции пирамид, например, рекурсивные модели, неокогнитрон, ранний LENET, пирамидальная нейронная сеть, пространственные пирамиды и ряд других [33]–[36].

Сеть PyraNet отличается от пирамид изображений тем, что нелинейная обработка на пирамидальных этапах может быть настроена путем обучения для конкретных задач распознавания. Сеть PyraNet имеет иерархическую многослойную структуру, обрабатывающую 2D-образы. Выходами сети является 1D-слой, представляющий категории входных образов. Структура сети задается на основе предложенных правил формирования пирамиды. Эти правила эвристические и не имеют какого-либо обоснования. Пирамидальная нейронная сеть PyraNet для своего обучения использует тот же арсенал методов, основанный на алгоритмах обратного распространения ошибок, что и классические многослойные нейронные сети. Общий недостаток этих методов − не гарантируемая сходимость по ошибкам обучения к глобальному минимуму и трудоемкость в вычислениях, особенно для глубоких нейронных сетей [37].

В настоящей главе будет показано, что пирамидальные нейронные сети можно построить на основе регулярных быстрых нейронных сетей. От пирамидальных сетей PyraNet они отличаются самоподобной структурой, наследуют методы обучения от быстрых нейронных сетей и являются глубокими по потенциалу обучения.

8.1. Построение пирамидальной нейронной сети

Рассмотрим принцип построения пирамидальной сети. Для определенности будем использовать топологию Гуда, заданную моделью:

$$\begin{aligned} U^m &= \langle v_{m-1} v_{m-2} \dots v_0 u_{n-1} u_{n-2} \dots u_m \rangle, \\ V^m &= \langle v_m v_{m-1} \dots v_0 u_{n-1} u_{n-2} \dots u_{m+1} \rangle, \\ z^m &= \langle v_{m-1} \dots v_0 u_{n-1} u_{n-2} \dots u_{m+1} \rangle. \end{aligned} \tag{8.1}$$

Из топологической модели следует, что для нулевого слоя номер нейронного ядра определяется выражением

$$z^0 = \langle u_{n-1} u_{n-2} \ldots u_1 \rangle,$$

а номер рецептора выражением

$$U^0 = \langle u_{n-1} u_{n-2} \ldots u_0 \rangle. \qquad (8.2)$$

Локальные переменные u_0, v_0 устанавливают позиционный номер рецептора и позиционный номер аксона в пределах каждого ядра нулевого слоя (нумерация рецепторов и аксонов начинается с нуля). Будем полагать, что число p_0 определяет число рецепторов ядра нулевого слоя, а число g_0 – число аксонов. Аналогичные обозначения p_m, g_m, где $m = 0,1,\ldots,n-1$, будем использовать для размерностей рецепторных и аксоновых полей остальных слоев сети. Выберем числа $g_m = 1$ для всех $m = 1,2,\ldots,n-1$. В этом случае из (8.1) следует, что аксоны последнего слоя получат глобальные позиционные номера

$$V^{n-1} = \langle 0_{n-1} 0_{n-2} \ldots 0_1 v_0 \rangle.$$

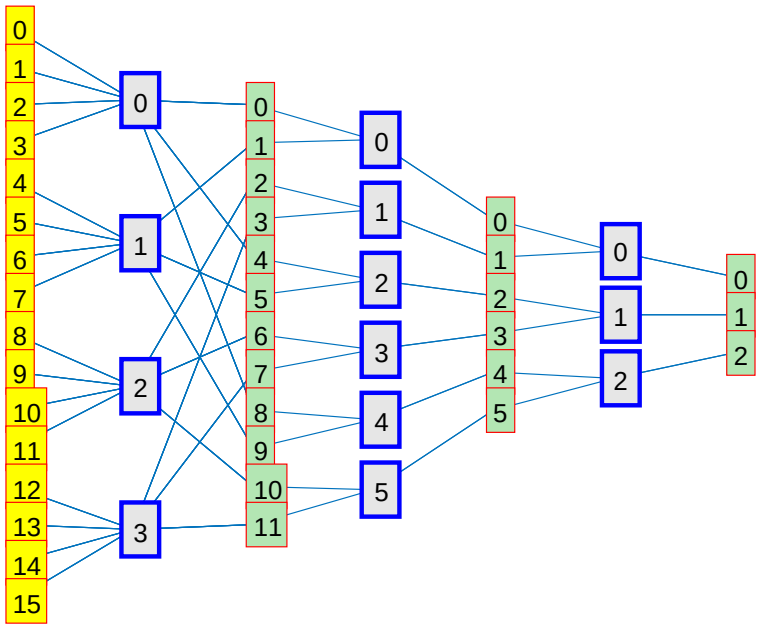


Рис. 8.1. Трехслойная быстрая пирамидальная нейронная сеть с выделенными нейронными ядрами

Таким образом, нейронная сеть будет иметь ровно g_0 выходов, а из (8.2) следует, что число входов будет равно $N = p_0 \cdot p_1 \cdot \ldots \cdot p_{n-1}$. Построенная сеть является пирамидальной, с размерностью N по входу и g_0 по выходу. На рис. 8.1 приведен пример трехслойной пирамидальной сети для структуры $(p_0, p_1, p_2) = (4,2,2)$, $(g_0, g_1, g_2) = (3,1,1)$.

8.2. Обучение пирамидальной нейронной сети для задач многоканальной корреляции сигналов

Один из основных способов распознавания сигналов состоит в сопоставлении сигнала с эталоном. Если сходство между неизвестным сигналом и эталоном велико, то сигнал помечается как соответствующий эталонному. Простейшей мерой сходства является линейный дискриминант Фишера [38]. В частном случае этот дискриминант представляет собой меру взаимной энергии сигналов и выражается через нормированное скалярное произведение

$$d = (x,e) \Big/ \sqrt{(x,x)(e,e)} > \alpha,$$

где x – неизвестный сигнал, представленный вектором; e – вектор эталонного сигнала; скобки $(,)$ обозначают скалярное произведение векторов; скаляр α определяет пороговый уровень различения сигналов. В многоканальном дискриминаторе используются несколько эталонных сигналов. Будем полагать, что входной и эталонные сигналы нормированы к энергии, так что $(x,x)=1$ и $(e,e)=1$. В этом случае дискриминант определяется скалярным произведением $d=(x,e)$.

Скалярное произведение можно рассматривать как произведение вектора-строки x на одностолбцовую матрицу $H = e'$. Или в покоординатном выражении:

$$d = \sum_U x(U)h(U,0).$$

А для многоканального дискриминатора матрица H будет состоять из g_0 столбцов и выполнять операцию:

$$d(k) = \sum_U x(U)h(U,V).$$

Будем полагать, что матрица H является матрицей быстрого пирамидального преобразования с топологией Гуда. Поскольку сеть имеет g_0 выходов, то

$$V^{n-1} = \langle 0_{n-1}0_{n-2}\ldots 0_1 v_0 \rangle.$$

На каждом выходе будет формироваться значение скалярного произведения, отвечающего эталонному сигналу. Пирамидальная сеть, показанная на рис. 8.1, может быть обучена к трем сигнальным функциям, заданным на интервале длиной 16. Выполним мультипликативную декомпозицию эталонных функций (см. 6.1), в результате получим

$$f^k(u) = f_0^k \langle u_{n-1}u_{n-2}\ldots u_0 \rangle = \varphi_{i^0}^k(u_0)\varphi_{i^1}^k(u_1)\ldots\varphi_{i^{n-2}}^k(u_{n-2})\varphi_{i^{n-1}}^k(u_{n-1}),$$

где k – порядковый номер функции, $z^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle$. Учитывая, что для построенной пирамидальной сети

$$z^m = \langle 0_{m-1}\ldots 0_1 v_0 u_{n-1}u_{n-2}\ldots u_{m+1}\rangle,$$

следуя (6.5), можно записать следующее правило обучения нейронных ядер:

$$\begin{aligned} &w^0_{z^0}(u_0, v_0) = \varphi^k_{i^0}(u_0), \quad m = 0, \\ &w^m_{z^m}(u_m, 0) = \varphi^k_{i^m}(u_m), \quad m = 1, 2, \ldots, n-1, \\ &z^m = \langle v_0 u_{n-1}u_{n-2}\ldots u_{m+1}\rangle. \end{aligned} \tag{8.3}$$

Индексы z^m и i^m отличаются только разрядной переменной v_0.

Взаимно-однозначное соответствие $k \leftrightarrow v_0$ определяет упорядочение сигнальных функций в выходном слое.

Замечание 1. Топология пирамидальной нейронной сети получена из общей топологии БНС за счет ограничения размерностей аксоновых полей условием: $g_0 > 1$ и $g_m = 1$ для $m = 1,2,\ldots,n-1$. Потенциально возможно увеличить количество выходов, положив, например, $g_0 > 1$ и $g_1 > 1$, оставив остальные значения $g_m = 1$. Мы получим пирамидальную сеть с другой архитектурой и расширенным количество выходов. Но в этом случае не удастся обучить сеть к большему числу эталонов, поскольку значение g_0 определяет число выходов нейронных ядер нулевого нейронного слоя и значит число независимых эталонов не может быть больше этой величины. Можно утверждать, что предложенная архитектура пирамидальной нейронной сети полностью адекватна задаче многоканальной корреляции сигналов.

Замечание 2. Поскольку для предложенной архитектуры пирамидальной сети выполняются условия: $g_m = 1$ для всех $m = 1,2,\ldots,n-1$, то из (8.1) следует

$$\begin{aligned} U^m &= \langle 0_{m-1}0_{m-2}\ldots 0_1 v_0 u_{n-1}u_{n-2}\ldots u_m\rangle, \\ V^m &= \langle 0_m 0_{m-1}\ldots 0_1 v_0 u_{n-1}u_{n-2}\ldots u_{m+1}\rangle, \\ z^m &= \langle 0_{m-1}\ldots 0_1 v_0 u_{n-1}u_{n-2}\ldots u_{m+1}\rangle. \end{aligned} \tag{8.4}$$

В частности, для нулевого слоя имеем:

$$\begin{aligned} U^0 &= \langle u_{n-1}u_{n-2}\ldots u_0\rangle, \\ V^0 &= \langle v_0 u_{n-1}u_{n-2}\ldots u_1\rangle, \\ z^0 &= \langle u_{n-1}u_{n-2}\ldots u_1\rangle. \end{aligned}$$

Кортеж V^m в (8.4) определяет множество номеров аксонов в пределах слоя. Нетрудно заметить, что это множество распадается на непересекающиеся подмножества, отличающиеся значением разряда v_0. Напомним, что в БНС ветвления аксонов не допускается, и каждый нейрон слоя имеет точно один аксон [22].

Кортеж U^m определяют множества номеров рецепторов в каждом слое. С каждым аксоном нейронного ядра связано подмножество рецепторного поля, для которого разряды номеров рецепторов и аксона совпадают по значениям с одноименными разрядами номера нейронного ядра. Рецепторные поля нейронных ядер слоя не пересекаются.

Из описанного следует вывод, что пирамидальная нейронная сеть предложенной архитектуры распадается на независимые группы нейронов (слоты), индексированных разрядом v_0. Каждый слот нейронов способен обучаться независимо от других слотов. Таким образом, пирамидальная нейронная сеть обладает уникальной возможностью последовательного дообучения к эталонам без потери ранее накопленных знаний (здесь уместно вспомнить высказывания Эдельмана о независимых нейронных группах, см. 1.3).

8.3. Вычислительная эффективность пирамидальной нейронной сети

Оценим вычислительную эффективность пирамидальной нейронной сети по числу операций при выполнении преобразования и при обучении. Для пирамидальной сети основания разрядных чисел заданы таблицами:

$$\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} u_0 & u_1 & \dots & u_{n-1} \\ p_0 & p_1 & \dots & p_{n-1} \end{pmatrix}, \qquad \begin{pmatrix} v \\ g \end{pmatrix} = \begin{pmatrix} v_0 & v_1 & \dots & v_{n-1} \\ g_0 & 1 & \dots & 1 \end{pmatrix}.$$

Из (8.4) следует, что число ядер в слое m определяется выражением:

$$k_m = \begin{cases} p_{n-1}p_{n-2}\cdots p_1, & m = 0, \\ g_0 p_{n-1}p_{n-2}\cdots p_{m+1}, & m > 0. \end{cases}.$$

Ядра нулевого слоя имеют размерность $p_0 \times g_0$, а ядра всех остальных слоев (для $m > 0$) имеют размерность $p_m \times 1$. Таким образом, полное число операций умножения будет равно:

$$Z = k_0 p_0 g_0 + \sum_{m=1}^{n-1} k_m p_m = g_0 p_{n-1}p_{n-2}\cdots p_0 + g_0 \sum_{m=1}^{n-1} p_{n-1}p_{n-2}\cdots p_m.$$

Для оценки порядка величин рассмотрим вариант быстрого преобразования, для которого $p_m = g_0 = p$. В этом случае

$$Z = p^{n+1} + \frac{p^2\left(p^{n-1}-1\right)}{p-1} = \frac{p^2\left(p^n-1\right)}{p-1}. \qquad (8.5)$$

Для пирамидальной сети число операций, требуемых для обучения, такое же, как и для быстрой нейронной сети. Напомним, что для оценочного варианта число операций деления, необходимых для выполнения обучения быстрого преобразования, равно

$$Z_t = \frac{p^2\left(p^n-1\right)}{p-1}.$$

Как видим, значения Z и Z_t совпадают, т. е. для обучения пирамидальной сети требуется ровно столько же операций, сколько и для выполнения преобразования.

Сравним также полученный результат с прямым матричным выполнением преобразования. Матрица прямого преобразования по входу имеет размерность $N = p_0 p_1 \ldots p_{n-1}$ а по выходу g_0. Для оценки также рассмотрим вариант $p_m = g_0 = p$. В этом случае матрица будет иметь $Ng_0 = p^{n+1}$ элементов, и столько же операций умножения требуется для выполнения прямого преобразования на обычном процессоре. Поделив в выражении (8.5) полином $\left(p^n - 1\right)$ на знаменатель, получим

$$Z = p^2\left(p^{n-1} + p^{n-2} + \cdots + p^2 + p + 1\right).$$

Отношение $\frac{Z}{p^{n+1}} = 1 + \frac{1}{p} + \frac{1}{p^2} + \cdots + \frac{1}{p^{n-1}}$ превышает 1, это означает, что использование сети на обычном процессоре с последовательным выполнением операций менее эффективно, чем прямое матричное умножение. Но для процессора с параллельным выполнением операций и конвейерной обработкой время выполнения будет определяться длиной конвейера, т. е. равняться значению n. И здесь выигрыш пирамидальной сети по вычислительной эффективности будет существенен.

8.4. Пластичность пирамидальной нейронной сети

В биологии термин пластичность используется как качественная характеристика способности нейронной сети обучаться при воздействии внешних факторов. Для искусственных нейронных сетей адекватной количественной

оценкой может служить число независимых настроек, называемое также числом степеней свободы. Это значение, как правило, меньше полного количества настраиваемых синаптических весов нейронной сети. Для БНС получены точные формулы расчета числа степеней свободы [22] (см. также гл. 12). Исходными данными для расчета является структурная модель нейронной сети. Структурная модель – это взвешенный граф, вершинами которого являются нейронные ядра. Вес каждого ядра определяется парой чисел, представляющих размерность по входу и выходу, а вес дуг определяется рангами операторов межъядерной связи. На рис. 8.2 представлена структурная модель для сети, показанной на рис. 8.1.

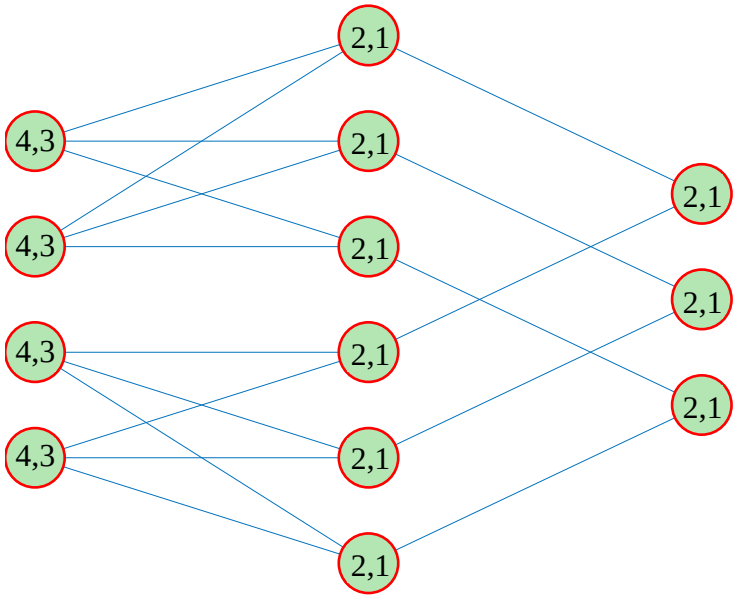


Рис. 8.2. Структурная модель пирамидальной нейронной сети

Для БНС ранги операторов межъядерных связей равны 1 и на графе структурной модели не показаны. Формула расчета числа степеней свободы для БНС имеет вид (см. также гл. 12):

$$S(H)=\sum_{m=0}^{n-1} p_m g_m q_m - \sum_{m=0}^{n-2} D_m ,$$

где p_m, g_m – размерности рецепторного и аксонового поля ядра в слое m; q_m – число ядер в слое; D_m – количество одноранговых связей в межслойном переходе с номером m. Для структурной модели, представленной на рис. 8.2, непосредственно получим $S(H)=4\cdot3\cdot4+2\cdot1\cdot6+2\cdot1\cdot3-12-6=48$. Напомним, что данная сеть может обучиться к трем эталонным функциям, заданным на интервале длиной 16. При произвольном выборе трех функций необходимо задать 48 значений, что совпадает с числом степеней свободы пирамидальной сети. Таким образом, рассматриваемая сеть является глубокой в

том смысле, что ее потенциал обучения используется полностью и покрывает все допустимое многообразие эталонных сигналов.

8.5. Двумерная пирамидальная сеть

Для двумерной пирамидальной сети на вход подается изображение. Будем полагать, что нейронная сеть задана топологией Гуда по обеим координатам. Топологическая модель задается выражениями:

$$\begin{aligned} U_*^m &= \left\langle v_{m-1}^* v_{m-2}^* \dots v_0^* u_{n-1}^* u_{n-2}^* \dots u_m^* \right\rangle, \\ V_*^m &= \left\langle v_m^* v_{m-1}^* \dots v_0^* u_{n-1}^* u_{n-2}^* \dots u_{m+1}^* \right\rangle, \\ z_*^m &= \left\langle v_{m-1}^* v_{m-2^*}^* \dots v_0^* u_{n-1}^* u_{n-2}^* \dots u_{m+1}^* \right\rangle. \end{aligned} \tag{8.6}$$

Символ (*) здесь заменяет индексы $x, y,$ означающие принадлежность к осям координат исходного изображения. Для пирамидальной сети число выходов равно $g_0^x \cdot g_0^y$, а их нумерация определяется выражениями:

$$V_y = \left\langle 0_{n-1}^y 0_{n-2}^y \cdots 0_1^y v_0^y \right\rangle, \qquad V_x = \left\langle 0_{n-1}^x 0_{n-2}^x \cdots 0_1^x v_0^x \right\rangle.$$

Таким образом, все разрядные переменные $v_m^* = 0$ для $m = 1, \cdots, n-1$. То есть каждая базовая операция также будет иметь только один выход с локальным номером (0, 0). Выражения (8.6) для топологии сети в этом случае примут вид:

$$\begin{aligned} U_*^m &= \left\langle 0_{m-1}^* 0_{m-2}^* \dots 0_1^* v_0^* u_{n-1}^* u_{n-2}^* \dots u_m^* \right\rangle, \\ V_*^m &= \left\langle 0_m^* 0_{m-1}^* \dots 0_1^* v_0^* u_{n-1}^* u_{n-2}^* \dots u_{m+1}^* \right\rangle, \\ z_*^m &= \left\langle 0_{m-1}^* 0_{m-2^*}^* \dots 0_1^* v_0^* u_{n-1}^* u_{n-2}^* \dots u_{m+1}^* \right\rangle. \end{aligned} \tag{8.7}$$

На рис. 8.3 представлен топологический граф коррелирующей пирамидальной сети для изображения $N \times N = 2^3 \times 2^3$ со структурными параметрами $g_0^x = 4, \quad g_0^y = 2$.

С учетом (8.7) из выражения (7.10) получим следующее правило обучения нейронных ядер:

$$\begin{aligned} w_{z_x^m, z_y^m}^m \left(u_m^y u_m^x ; v_m^y v_m^x \right) &= f_{i_x^m, i_y^m}^k \left(u_m^y, u_m^x \right), \quad m = 0, \\ w_{z_x^m, z_y^m}^m \left(u_m^y u_m^x ; 0_m^y 0_m^x \right) &= f_{i_x^m, i_y^m}^k \left(u_m^y, u_m^x \right), \quad m > 0, \\ z_*^m &= \left\langle 0_{m-1}^* 0_{m-2^*}^* \dots 0_1^* v_0^* u_{n-1}^* u_{n-2}^* \dots u_{m+1}^* \right\rangle. \end{aligned} \tag{8.8}$$

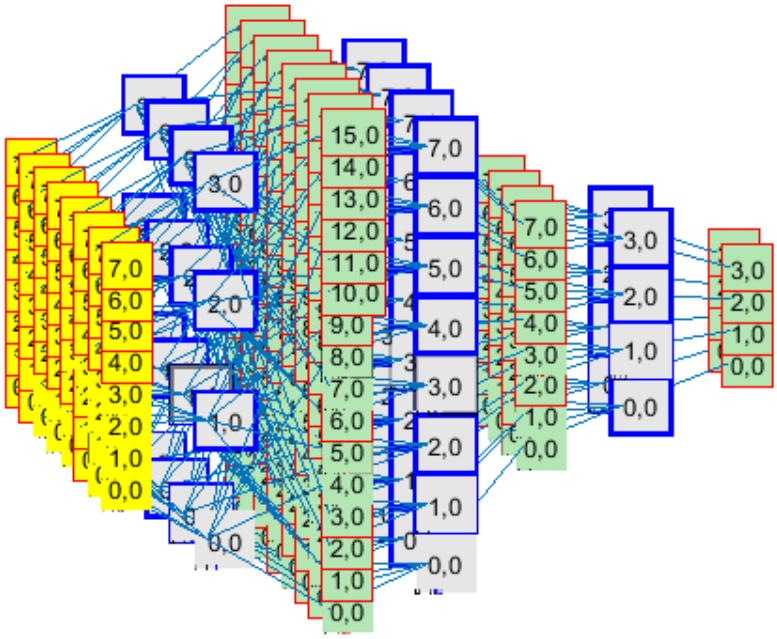


Рис. 8.3. Топология пирамидальной 2D-сети

Взаимно-однозначное соответствие $k \leftrightarrow \left(v_m^y, v_m^x\right)$ определяет упорядочение эталонных функций в выходной плоскости.

8.6. Корреляционные нейронные сети

Корреляционные нейронные сети подобны пирамидальным нейронным сетям, но имеют один выход. Сети данного типа могут быть эффективно использованы для корреляционных измерений в одномерном сигнальном пространстве.

8.6.1. Одномерная корреляционная сеть

Входом для сети является одномерный вектор. Как и прежде, будем полагать, что сеть описывается топологической моделью Гуда. Поскольку сеть имеет только один выход, то из (8.1) следует

$$V = V^{n-1} = \langle 0_{n-1} 0_{n-2} \ldots 0_1 0_0 \rangle .$$

Таким образом, все разрядные переменные $v_m = 0$ для $m = 0,1,\cdots,n-1$. То есть каждая базовая операция также будет иметь только один выход с локальным номером 0. Выражения (8.1) для топологии сети в этом случае примут вид:

$$U^m = \langle 0_{m-1} 0_{m-2} \ldots 0_0 u_{n-1} u_{n-2} \ldots u_m \rangle,$$

$$V^m = \langle 0_m 0_{m-1} \ldots 0_0 u_{n-1} u_{n-2} \ldots u_{m+1} \rangle,$$

$$z^m = \langle 0_{m-1} \ldots 0_0 u_{n-1} u_{n-2} \ldots u_{m+1} \rangle .$$

В частности, для входного слоя сети $m=0$, в этом случае имеем $i^0=\langle u_{n-1}u_{n-2}\ldots u_1\rangle$. Полагая, что разрядные переменные принимают значения $u_i\in[0,1,\ldots p_i-1]$, можно сделать вывод, что слой будет содержать $Z^0=$ $=p_{n-1}p_{n-2}\ldots p_1$ базовых операций. Следующий слой, $m=1$, будет содержать $Z^1=p_{n-1}p_{n-2}\ldots p_2$ базовых операций. Для последнего слоя имеем $z^{n-1}=$ $=\langle 0_{n-1}0_{n-2}\ldots 0_0\rangle=0$, и поэтому этот слой будет содержать только одну базовую операцию. Топология межслойных переходов определяется правилом $V^{m-1}=U^m$.

На рис. 8.4 представлен топологический граф корреляционной сети для $N=2^3$. Каждая базовая операция данной нейронной сети представляет собой один нейрон, имеющий два входа. Граф сети строится непосредственно по топологическим выражениям (см. гл. 2).

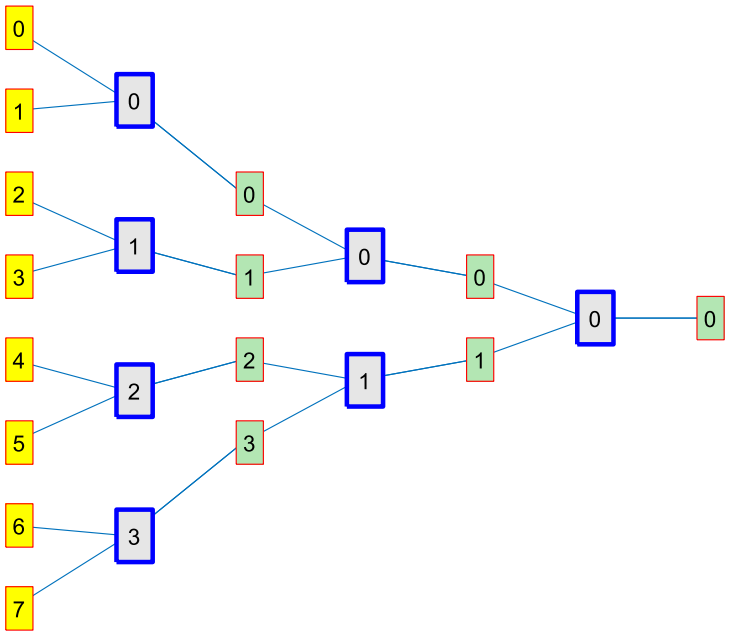


Рис. 8.4. Топология корреляционной нейронной сети

Правило обучения нейронной сети непосредственно следует из выражения (8.3) и имеет вид:

$$w^m_{z^m}(u_m,0)=\varphi_{z^m}(u_m).$$

8.6.2. Двумерная корреляционная сеть

Входом для сети является изображение. Будем полагать, что входное и эталонное изображения нормированы к энергии, так что $(x,x)=1$ и $(e,e)=1$.

В этом случае дискриминант определяется скалярным произведением $d=(x,e)$. Скалярное произведение можно рассматривать как произведение изображения x на одноплоскостную матрицу $H=e'$. Или в покоординатном выражении

$$d=\sum_{U_x,U_y} x(U_x,U_y)h\left(U_x,U_y;V_x,V_y\right).$$

Будем полагать, что матрица H является матрицей быстрого преобразования с топологией Гуда. Поскольку сеть имеет только один выход, то

$$V_y=\left\langle 0_{n-1}^y 0_{n-2}^y \cdots 0_1^y 0_0^y \right\rangle, \qquad V_x=\left\langle 0_{n-1}^x 0_{n-2}^x \cdots 0_1^x 0_0^x \right\rangle.$$

Таким образом, все разрядные переменные $v_m^*=0$ для $m=0,1,\ldots,n-1$. То есть каждая базовая операция также будет иметь только один выход с локальным номером (0,0). Выражения (8.7) для топологии сети в этом случае примут вид:

$$U_*^m=\left\langle 0_{m-1}^* 0_{m-2}^* \ldots 0_0^* u_{n-1}^* u_{n-2}^* \ldots u_m^* \right\rangle,$$
$$V_*^m=\left\langle 0_m^* 0_{m-1}^* \ldots 0_0^* u_{n-1}^* u_{n-2}^* \ldots u_{m+1}^* \right\rangle,$$
$$z_*^m=\left\langle 0_{m-1}^* 0_{m-2*}^* \ldots 0_0^* u_{n-1}^* u_{n-2}^* \ldots u_{m+1}^* \right\rangle.$$

На рис. 8.5 представлен топологический граф корреляционной сети для изображения $N\times N=2^3\times 2^3$. Каждая базовая операция данной нейронной сети представляет собой один нейрон, имеющий четыре входа.

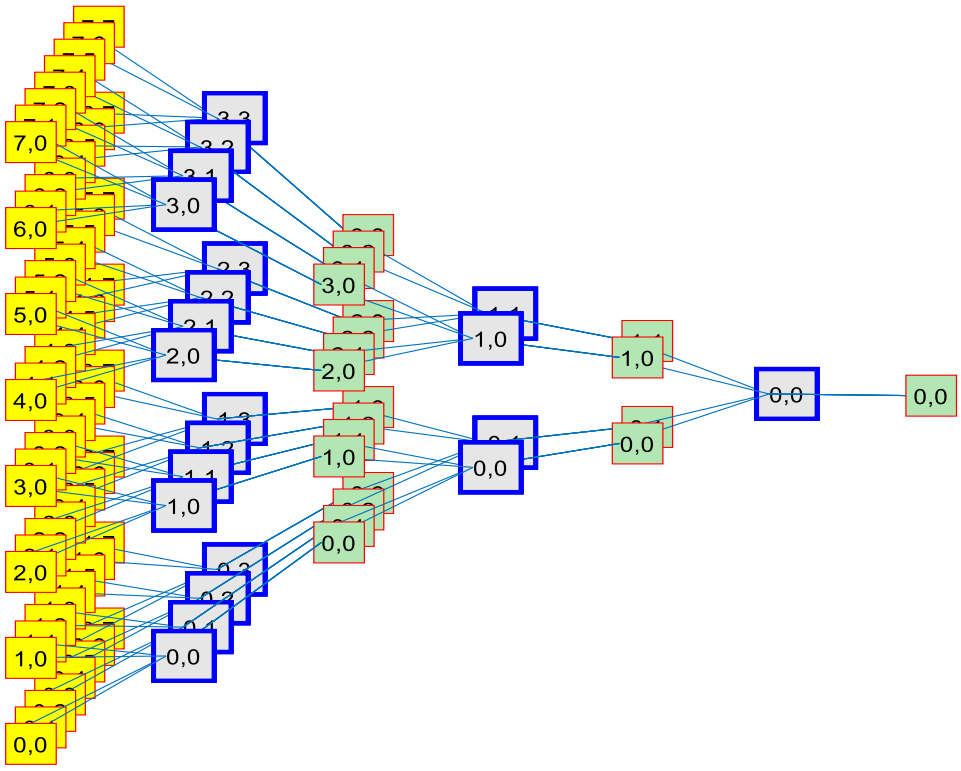


Рис. 8.5. Топология корреляционной 2D нейронной сети

Правило обучения нейронной сети следует из выражения (8.8) и имеет вид:

$$w^{m}_{z^{m}_{x},z^{m}_{y}}\left(u^{y}_{m}u^{x}_{m};0^{y}_{m}0^{x}_{m}\right)=f_{i^{m}_{x},i^{m}_{y}}\left(u^{y}_{m},u^{x}_{m}\right);$$

$$z^{m}_{*}=\left\langle 0^{*}_{m-1}0^{*}_{m-2*}\ldots 0^{*}_{1}0^{*}_{0}u^{*}_{n-1}u^{*}_{n-2}\ldots u^{*}_{m+1}\right\rangle;$$

$$i^{m}_{*}=\left\langle u^{*}_{n-1}u^{*}_{n-2}\ldots u^{*}_{m-1}\right\rangle.$$

9. РЕГУЛЯРНЫЕ НЕЙРОННЫЕ СЕТИ С ФОВЕАЛЬНОЙ АРХИТЕКТУРОЙ

Для большинства млекопитающих, включая человека, характерна фовеальная организация структуры глаза. При такой структуре сетчатка глаза имеет сегментированные поля с разным разрешением: центральное поле (фовеа) – с высоким разрешением и периферические поля – с низкой разрешающей способностью. Исследователи полагают, что процесс восприятия зрительной информации у человека основан на двухэтапной схеме: первоначально восприятие ведется в периферическом поле зрения и далее, при наличии интереса, зрительная система переводит точку внимания в фовеальную (центральную) часть поля зрения для детального анализа информации. На этом основана концепция активного зрения, используемая в робототехнике.

В работе [39] отмечалось, что фовеальную структуру целенаправленного зрительного восприятия роботов следует реализовать в нейросетевом базисе. При этом, по мнению автора работы, искусственная нейронная сеть должна удовлетворять ряду требований:

– должно отсутствовать разобучение «старого» при обучении «новому»;

– сеть должна быть управляемой в пространстве параметров и функционально универсальной;

– универсальная структура должна обеспечивать динамическую предварительную настройку сети на распознавание заданного класса объектов;

– обучение и перенастройка нейронной сети должна осуществляться в реальном времени;

– архитектура сети должна поддерживать фовеальное восприятие информации.

В данной главе на основе концепции быстрых нейронных сетей будет рассмотрена архитектура регулярной фовеальной нейронной сети, в той или иной мере покрывающая указанные требования.

9.1. Базовые сведения

Число эталонов, к которому может обучиться БНС, определяется выходной размерностью нейронных ядер входного слоя. При этом часть степеней свободы БНС остается неиспользованной (рис. 9.1). В 6.2 отмечалось, что реализовать все степени свободы можно в нейронной сети с фовеальной архитектурой.

	u_2	0	0	0	0	1	1	1	1
	u_1	0	0	1	1	0	0	1	1
	v_0	0	1	0	1	0	1	0	1
u_2	u_1	u_0							
0	0	0	1	1					
0	0	1	1	1					
0	1	0			1	1			
0	1	1			1	1			
1	0	0					1	1	
1	0	1					1	1	
1	1	0							1	1
1	1	1							1	1

	u_2	0	0	0	0	1	1	1	1
	v_1	0	0	1	1	0	0	1	1
	v_0	0	1	0	1	0	1	0	1
u_2	u_1	v_0							
0	0	0	1		*				
0	0	1		1		*			
0	1	0	1		*				
0	1	1		1		*			
1	0	0					1		*
1	0	1						1	*
1	1	0					1		*
1	1	1						1	*

	v_2	0	0	0	0	1	1	1	1
	v_1	0	0	1	1	0	0	1	1
	v_0	0	1	0	1	0	1	0	1
u_2	v_1	v_0							
0	0	0	1				*		
0	0	1		1				*	
0	1	0			*				*
0	1	1				*				*
1	0	0	1				*		
1	0	1		1				*	
1	1	0			*				*
1	1	1				*				*

Рис. 9.1. Топологические матрицы БНС,
звездочками отмечены неиспользуемые степени свободы

На рис. 9.2 показан граф сетевой модели, соответствующий матричному представлению, показанному на рис. 9.1. Дополнительно на графе выделены нейронные ядра. В данной модели каждый слой содержит четыре нейронных ядра размерности 2×2. Структурные характеристики сети задаются размерностями $P=[2\,2\,2]$, $G=[2\,2\,2]$.

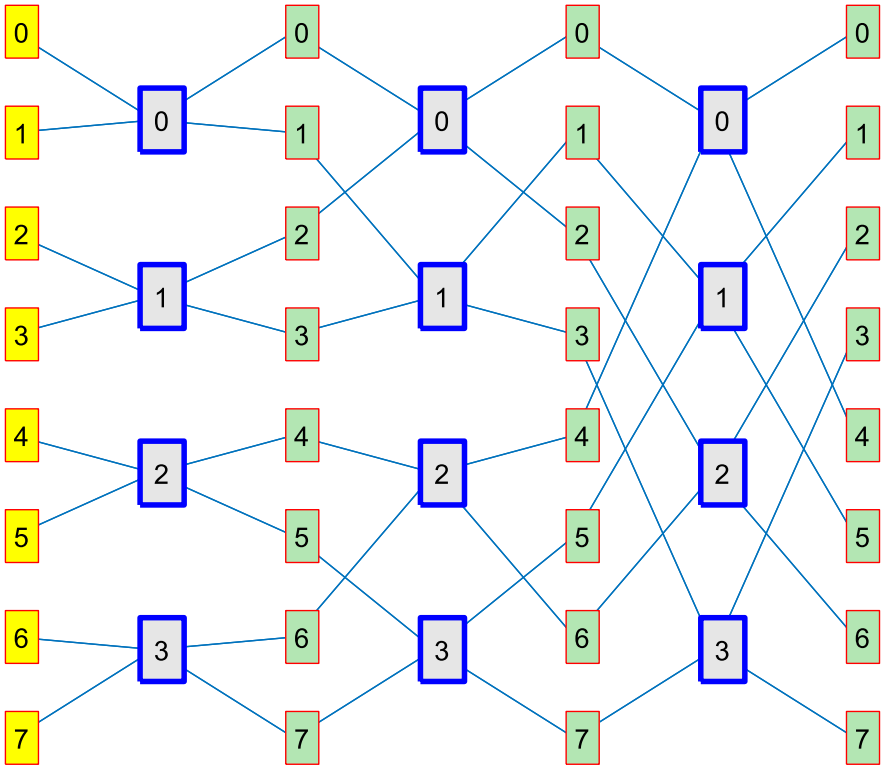


Рис. 9.2. Топологический граф БНС с выделенными нейронными ядрами

Для сравнения на рис. 9.3 показан топологический граф той же БНС без явного выделения нейронных ядер. Цвет вершин графа определяется их степенью. Такое представление графа более компактно и удобно при высокой размерности сети.

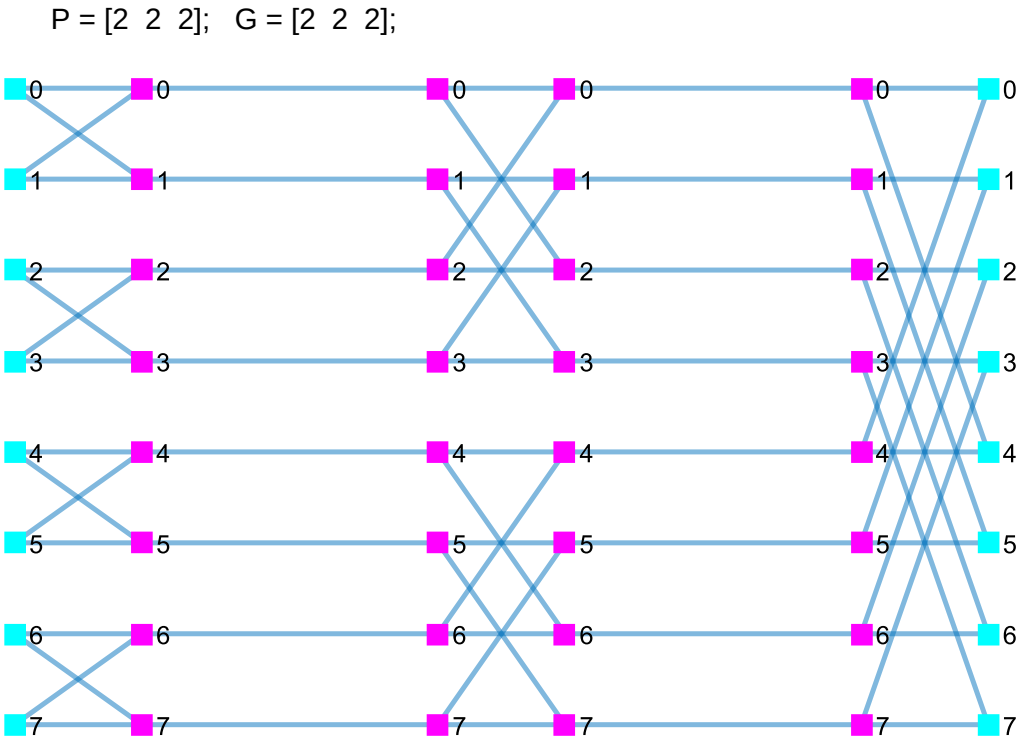


Рис. 9.3. Топологический граф БНС без выделения нейронных ядер

Сетевая модель данной топологии соответствует быстрому преобразованию с топологией Кули–Тьюки «с прореживанием по времени» и описывается набором кортежей:

$$\begin{aligned} U^m &= \langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m v_{m-1}v_{m-2}\ldots v_1 v_0\rangle, \\ V^m &= \langle u_{n-1}u_{n-2}\ldots u_{m+1}v_m v_{m-1}v_{m-2}\ldots v_1 v_0\rangle, \\ z^m &= \langle u_{n-1}u_{n-2}\ldots u_{m+1}v_{m-1}v_{m-2}\ldots v_1 v_0\rangle, \end{aligned} \qquad (9.1)$$

где z^m – порядковый номер ядра; U^m, V^m – порядковые номера рецепторов и аксонов в слое m; u_m, v_m – локальные номера рецепторов и аксонов в пределах ядра слоя m. Каждый кортеж представляет собой поразрядную форму представления порядкового номера через разрядные переменные u_m, v_m. Основания разрядных переменных u_m, v_m определяются целыми положительными числами, зададим их соответствиями:

$$\begin{pmatrix} u_0 & u_1 & \cdots & u_{n-2} & u_{n-1} \\ p_0 & p_1 & \cdots & p_{n-2} & p_{n-1} \end{pmatrix}, \quad \begin{pmatrix} v_0 & v_1 & \cdots & v_{n-2} & v_{n-1} \\ g_0 & g_1 & \cdots & g_{n-2} & g_{n-1} \end{pmatrix}.$$

Для ядер размерности 2×2 все разрядные переменные принимают значения $\{0,1\}$. Координатные направления U^m и V^m в дальнейшем будем называть входной и выходной плоскостью нейронного слоя.

9.2. Построение сети с фовеальной архитектурой

Число не использованных при настройке БНС степеней свободы недостаточно для обучения еще к одному эталону высокого разрешения, но их с запасом хватит для обучения к эталону с кратно-уменьшенной размерностью. Снижение размерности эталона эквивалентно уменьшению разрешения в фовеальной структуре. Эти обстоятельства являются мотивацией создания фовеальной архитектуры нейронной сети на основе БНС.

Рассмотрим этапы построения фовеальной архитектуры на основе модели БНС. Для определенности будем использовать преобразование, заданное топологической моделью (9.1).

В выходной плоскости V^{n-1} отметим точку C с координатами $C=\langle c_{n-1}c_{n-2}\ldots c_0\rangle$, которую назовем фовеальным центром. Область в выходной плоскости определяемые множеством точек:

$$A^m=\langle c_{n-1}c_{n-2}\ldots c_{m+1}v_m v_{m-1}\ldots v_0\rangle,$$

где разряды $v_m v_{m-1}\ldots v_0$ варьируются по всем возможным значениям, назовем областью ответственности слоя m относительно выбранного фовеального центра. Для $m=0$ область ответственности равна $A^0=\langle c_{n-1}c_{n-2}\ldots c_2c_1v_0\rangle$, для $m=1$ – $A^1=\langle c_{n-1}c_{n-2}\ldots c_3c_2v_1v_0\rangle$, для $m=2$ область ответственности – $A^2=\langle c_{n-1}c_{n-2}\ldots c_3v_2v_1v_0\rangle$ и т. д. В общем случае – $A^m=\langle c_{n-1}c_{n-2}\ldots$ $\ldots c_{m+1}v_m v_{m-1}\ldots v_0\rangle$.

Для области A^m при выборе $v_m=c_m$ получим $\langle c_{n-1}c_{n-2}\ldots$ $\ldots c_{m+1}c_m v_{m-1}\ldots v_0\rangle=A^{m-1}$, т. е. область A^m содержит в себе область A^{m-1}, отвечающую слою $m-1$. Эту часть области ответственности A^m назовем центральной зоной. Зону $\bar{A}^m=\langle c_{n-1}c_{n-2}\ldots c_{m+1}\bar{v}_m v_{m-1}\ldots v_0\rangle$, где $\bar{v}^m$ принимает все значения, отличные от c_m, будем называть периферийной зоной области A^m. Таким образом, область ответственности разделяется на центральную и периферийную, и все центральные зоны областей ответственности являются вложенными по слоям.

Для преобразования топологии БНС в фовеальную сеть выполним в топологической модели (9.1) расщепление всех слоев сети вдоль плоскости V^m по

сечениям $\langle v_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle$. Образованные компоненты назовем картами слоя, а число $\langle v_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle$ – номером карты. Число карт, образованных в слое m, равно произведению $g_m g_{m-1} g_{m-2} \ldots g_1 g_0$. Карты слоя разделяются на два подмножества: карты с номерами $\langle c_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle$ соответствуют центральной зоне, а с номерами $\langle \bar{v}_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle$ – периферийной зоне.

В каждом слое выполним клонирование входной плоскости U^m, создав входные плоскости для каждой карты слоя. Общее число входных плоскостей в слое будет равно числу карт. Новые плоскости сохраняют топологическую схему слоя (в данном случае $U^m = \langle u_{n-1} u_{n-2} \ldots u_{m+1} u_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle$) и наследуют связи карт слоя с входной плоскостью.

Плоскости центральных зон сохраняют межслойные связи для обеспечения вложенности центральных зон по выбранному центру. В плоскостях периферийных зон можно организовать дополнительные векторные входы размерности $p_{n-1} p_{n-2} \ldots p_m$, число таких входов будет равно $g_{m-1} g_{m-2} \ldots g_0$, а их позиции в плоскости определяются числами $\langle v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle$. На рис. 9.4 представлена сеть с фовеальной архитектурой, построенная на основе БНС с характеристиками $P = [2\ 2\ 3]$, $G = [2\ 2\ 3]$ и с топологией (9.1).

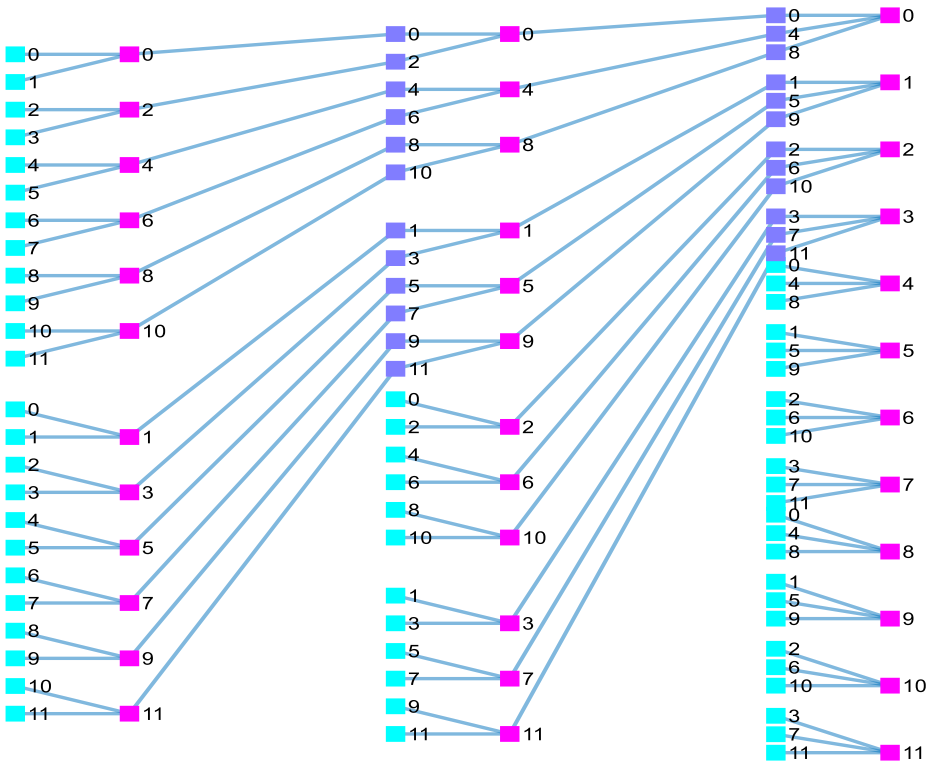


Рис. 9.4. Топология нейронной сети с фовеальной архитектурой

В построенной сети каждая карта слоя имеет свой вход. Входы карт центральных зон связаны с выходами предыдущего слоя. Входы карт периферийных зон свободны и могут использоваться для передачи в сеть образов меньшего разрешения. Как видно из рисунка, сеть распадается на несколько корреляционных сетей с различной размерностью по входу. Так же как и корреляционные сети, фовеальная нейронная сеть относится к категории сетей глубокого обучения. Особенностью построенной архитектуры является то, что все образы одного разрешения обрабатываются одновременно, это может быть полезно для более общих задач, чем фовеальное восприятие. Сеть будет решать задачу фовеальной обработки, если на входы меньшего разрешения будет поступать тот же образ, что и на фовеальный центр, но с меньшим разрешением. По этому условию должно быть выполнено и обучение сети. С уменьшением разрешения увеличивается количество эталонов, к которым можно обучить сеть; если для слоя 0 это число равно g_0, для слоя 1 – $(g_1-1)g_0$ и в общем случае $(g_m-1)g_{m-1}g_{m-2}\dots g_0$ для слоя m.

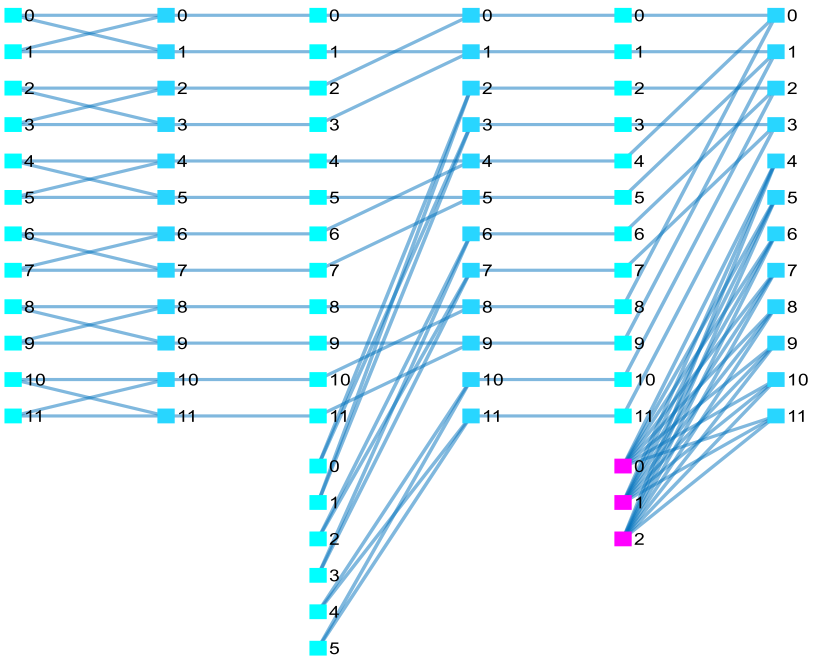


Рис. 9.5. Топология фовеальной нейронной сети с объединенными входами

Если в фовеальном классификаторе входной образ меньшего разрешения формируется в единственном варианте из образа большего разрешения, то все свободные входы сети в каждом слое можно объединить в один, что ведет к упрощению топологии сети. На рис. 9.5 показана топология фовеальной сети с объединенными входами, построенная на основе БНС с характеристиками $P=[2\,2\,3], G=[2\,2\,3]$.

9.2.1. Обучение фовеальной нейронной сети

Фовеальная сеть обучается к эталонам различного разрешения. Образы наивысшего разрешения будут поступать на вход нулевого слоя и относиться к центральной зоне нулевого слоя. Эта фовеальная зона покрывает в выходной плоскости фовеальный центр и ближайшие к нему позиции. Количество эталонов в фовеальной зоне очевидно равно g_0. Множество эталонов зоны описывается выражением:

$$f_0^{k_0}(U) = f_0^{k_0}\langle u_{n-1}u_{n-2}\ldots u_1u_0\rangle = \prod_{m=0}^{n-1}\varphi_{i^m}^{k_0}(u_m), \qquad (9.2)$$

где $i^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle$ – номер фрактального множителя; k_0 – порядковый номер эталона в нулевой фовеальной зоне. Согласно 4.1, элементы матрицы БНС выражаются через элементы нейронных ядер:

$$h(U,V) = w_{z^{n-1}}^{n-1}(u_{n-1},v_{n-1})\,w_{z^{n-2}}^{n-2}(u_{n-2},v_{n-2})\ldots w_{z^0}^{0}(u_0,v_0).$$

Сравнивая последнее выражение с представлением эталонов (9.2), получим правило обучения ядер для фовеальной зоны нулевого порядка:

$$w_{z^m}^{m}(u_m,c_m) = \varphi_{i^m}^{k_0}(u_m),$$

$$w_{z^0}^{0}(u_0,v_0) = \varphi_{i^0}^{k_0}(u_0),$$

где $z^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}c_{m-1}c_{m-2}\ldots c_1v_0\rangle$, $i^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle$, $m = 0,1,\ldots,n-1$, $v_0 = 0,1,\ldots,g_0$. Взаимно-однозначное соответствие $v_0 \leftrightarrow k_0$ задает упорядочение эталонов в нулевой фовеальной зоне $A^0 = \langle c_{n-1}c_{n-2}\ldots c_2c_1v_0\rangle$. Эталонные образы, поступающие на свободные входы фовеальной сети внутренних слоев, описываются выражением:

$$f_m^{k_m}(U) = f_m^{k_m}\langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m\rangle = \prod_{s=m}^{n-1}\varphi_{i^s}^{k_m}(u_s),$$

$$i^s = \langle u_{n-1}u_{n-2}\ldots u_{s+1}\rangle,\ s = m, m+1,\ldots,n-1.$$

Согласно следствию теоремы факторизации (см. 4.1), элементы матрицы неполного преобразования могут быть представлены произведением:

$$h^m(U^m,V) = w_{z^{n-1}}^{n-1}(u_{n-1},v_{n-1})\,w_{z^{n-2}}^{n-2}(u_{n-2},v_{n-2})\ldots$$

$$\ldots w_{z^{m+1}}^{m+1}(u_{m+1},v_{m+1})\,w_{z^m}^{m}(u_m,v_m).$$

Сравнивая последнее выражение с представлением эталонов, получим правило обучения ядер для периферийной фовеальной зоны s-го порядка:

$$w^{s}_{z^{s}}\left(u_s, v_s\right) = \varphi^{k_m}_{i^{s}}\left(u_s\right),$$

$$w^{m}_{z^{m}}\left(u_m, \bar{v}_m\right) = \varphi^{k_m}_{i^{m}}\left(u_m\right),$$

где $z^{s} = \left\langle u_{n-1}u_{n-2}\ldots u_{s+1}v_{s-1}v_{s-2}\ldots v_1v_0\right\rangle$, $i^{s} = \left\langle u_{n-1}u_{n-2}\ldots u_{s+1}\right\rangle$, $s = m, m+1, \ldots, n-1$, $\bar{v}_m \neq c_m$, k_m – порядковый номер эталона в фовеальной зоне m-го порядка. Число эталонов равно $\left(g_m - 1\right)g_{m-1}g_{m-2}\ldots g_0$ и соответствует размеру периферийной зоны области ответственности слоя m. Взаимно-однозначное соответствие $\left\langle \bar{v}_m v_{m-1}v_{m-2}\ldots v_1v_0\right\rangle \leftrightarrow k_m$ задает упорядочение эталонов в периферийной фовеальной зоне m-го порядка $\bar{A}^{m} = \left\langle c_{n-1}c_{n-2}\ldots c_{m+1}\bar{v}_m v_{m-1}\ldots v_0\right\rangle$.

9.3. Усеченная БНС

Следствие из теоремы факторизации (см. 4.1) определяет представление элементов матрицы БНС через элементы ядер при неполном числе слоев. В усеченной сети удалены слои с номерами $0, 1, \ldots, m-1$, но сохранены с номерами $m, m+1, \ldots, n-1$. Правило построения топологической модели при этом осталось неизменным. Как и прежде, для определенности будем полагать, что в преобразовании используется топология Кули–Тьюки «с прореживанием по времени». Тогда рецепторное поле входного слоя усеченной сети будет соответствовать топологической схеме:

$$U^{m} = \left\langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m v_{m-1}v_{m-2}\ldots v_1v_0\right\rangle.$$

Выполним декомпозицию входного рецепторного поля на поля размерности $p_{n-1}p_{n-2}\ldots p_m$, число образованных полей будет равно $g_{m-1}g_{m-2}\ldots g_0$, а позиция каждого поля будет определяться числом $\left\langle v_{m-1}v_{m-2}\ldots v_1v_0\right\rangle$, заданным в поразрядной форме. Новые поля в дальнейшем используются как частные входы в нейронную сеть. На каждый из образованных входов можно независимо подать входные векторы размерности $p_{n-1}p_{n-2}\ldots p_m$. Каждый частный вход имеет в выходной плоскости сети зону ответственности размером $g_{n-1}g_{n-2}\ldots g_m$. Эти зоны между собой не пересекаются, а их позиции в выходной плоскости определяются топологической схемой

$$V^{n-1} = \left\langle v_{n-1}v_{n-2}\ldots v_m v_{m-1}v_{m-2}\ldots v_0\right\rangle.$$

Каждая область ответственности по существу является выходом частной сети, вход которой определен номером $\langle v_{m-1}v_{m-2}\ldots v_1v_0\rangle$, и этот же номер определяет размещение выхода частной сети в выходной плоскости. Иными словами, сеть оказывается сегментированной на частные сети, которые между собой не пресекаются и выполняют обработку входных векторов, независимо друг от друга. На рис. 9.6 показана топология усеченной сети, построенная на основе БНС с топологией Кули–Тьюки (9.1) с характеристиками $P=[2\,2\,3], G=[2\,2\,3]$ при отсечении по слою $m=1$.

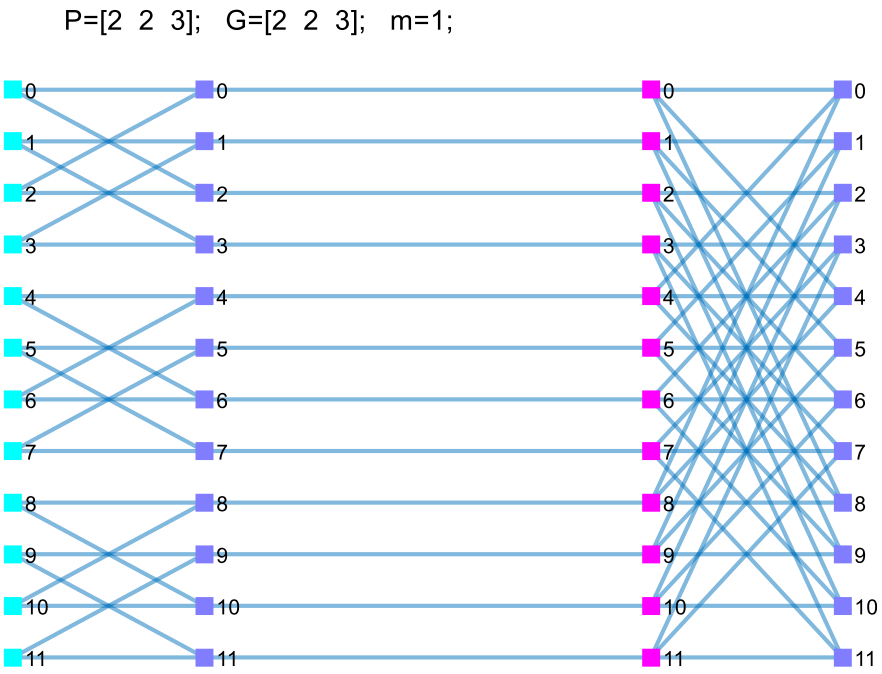


Рис. 9.6. Топология усеченной сети

Топологическая схема для входного слоя определяется кортежем $V^1=\langle u_2u_1v_0\rangle$. Поскольку $g_0=2$, то данная сеть имеет два векторных входа размерностью $p_2p_1=3\cdot 2=6$. Все входные рецепторы с четными номерами относятся к первому входу, а с нечетными – ко второму. Зоны ответственности входов определяются топологической схемой $V^2=\langle v_2v_1v_0\rangle$, все четные аксоны выходного слоя относятся к первому входу, а нечетные – ко второму. Сегменты сети независимы, поэтому при построении одиночного классификатора входы можно параллельно объединить.

9.3.1. Обучение усеченной БНС сети

Частные сети обучаются к эталонам подобно полной БНС. Так же как и для фовеальной сети, обучающие эталоны представим в мультипликативной форме:

$$f_m^{k_m}(U) = f_m^{k_m}\langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m\rangle = \prod_{s=m}^{n-1}\varphi_{i^s}^{k_m}(u_s),$$

$$i^s = \langle u_{n-1}u_{n-2}\ldots u_{s+1}\rangle,\ s = m, m+1, \ldots, n-1.$$

Сравнивая последнее с представлением элементов матрицы усеченной БНС, получим следующее правило обучения сети:

$$w_{z^s}^{s}(u_s, v_s) = \varphi_{i^s}^{k_m}(u_s),$$

где $z^s = \langle u_{n-1}u_{n-2}\ldots u_{s+1}v_{s-1}v_{s-2}\ldots v_1v_0\rangle$, $i^s = \langle u_{n-1}u_{n-2}\ldots u_{s+1}\rangle$, $s = m, m+1, \ldots, n-1$, k_m – порядковый номер эталона в зоне ответственности входа. Задаваемое взаимно-однозначное соответствие $k_m \leftrightarrow \langle v_{n-1}v_{n-2}\ldots v_m\rangle$ определяет порядок размещения точек приспособления в зоне ответственности входа.

9.4. Сравнение сетей

Фовеальная и усеченная сеть строятся на основе БНС. В обоих типах порождаются новые входы, размерности которых кратно меньше размерности входов материнской БНС. Обе сети представимы в виде независимых частных нейронных сетей. В фовеальной сети частные сети – это корреляционные сети различной размерности, поэтому эта сеть является сетью глубокого обучения, использующей все степени свободы БНС. В усеченной сети частные сети – это опять БНС, но меньшей размерности, поэтому часть степеней свободы оказывается неиспользованной.

Если строить лес усеченных БНС различной размерности, то он может в той или иной мере заменить фовеальную структуру, возможно с меньшей вычислительной эффективностью, но и с меньшими ограничениями на состав фовеальной структуры. Частные сети одинаковой размерности в сетях обоих типов можно объединять по входам, наращивая мощность нейросетевого классификатора. Независимость частных сетей позволяет строить классификаторы, которые обучаются к новым данным без потери накопленных знаний. Сети обоих типов наследуют высокие скорости обучения БНС, сопоставимые со скоростью обработки входных данных в нейросетевой структуре или даже выше этого уровня.

Усеченную БНС можно трансформировать в сеть глубокого обучения, если частные БНС заменить пирамидальными сетями. Для этого достаточно в сети, усеченной по слою m, выбрать единичными основания разрядных

переменных $v_{m+1}, v_{m+2}, \ldots, v_{n-1}$, т. е. положить $g_{m+1} = g_{m+2} = \ldots = g_{n-1} = 1$. На рис. 9.7 приведен пример топологии усеченной сети с характеристиками материнской БНС: $P = [2\ 2\ 2\ 2]$, $G = [2\ 2\ 1\ 1]$, усеченной по слою $m = 1$. Сеть использует топологическую модель (9.1).

```
P = [2  2  2  2];   G = [2  2  2  2];   m = 1;
```

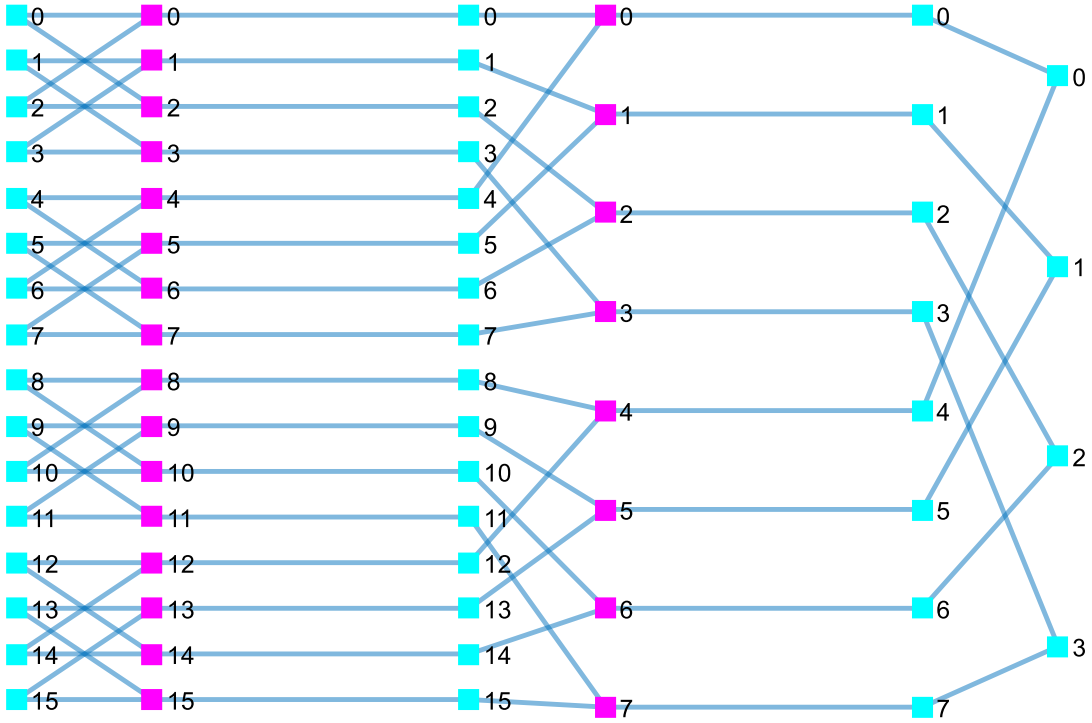


Рис. 9.7. Усеченная нейронная сеть глубокого обучения

Построенная сеть имеет два векторных входа размерностью $p_3 p_2 p_1 = 2 \cdot 2 \cdot 2 = 8$. К первому входу относятся все четные рецепторы входного слоя, а ко второму – все начетные. Частными сетями являются две пирамидальные сети, каждая из которых имеет два аксона на выходе. Все четные аксоны выходного слоя относятся к первому входу, а нечетные – ко второму. Сегменты сети независимы, поэтому при построении одиночного классификатора входы можно параллельно объединить.

Рассмотренные в этой главе структуры сетей построены на основе модели БНС, что позволяет сохранить аналитическое описание для топологических моделей, это свойство является особенно полезным при построении нейронных сетей для обработки данных высокой размерности. Тем не менее, следует заметить, что фовеальные и усеченные нейронные сети можно строить, не используя в качестве базовой модель БНС, а произвольно комбинируя между собой корреляционные сети.

10. БЫСТРЫЕ НЕЙРОННЫЕ СЕТИ ГЛУБОКОГО ОБУЧЕНИЯ С КОММУТАЦИЕЙ ПЛОСКОСТЕЙ

Технология глубокого обучения предполагает наличие процесса конфигурационного усложнения информативных признаков в последовательности нейронных слоев. Начиная с неокогнитрона К. Фукусима [40], предложено множество вариантов реализации этой идеи. Одним из удачных решений является архитектура сверточных нейронных сетей [41], которая показала высокую эффективность при решении различных задач. Отличительной особенностью этой архитектуры является наличие в сверточных слоях нескольких каналов обработки данных (называемых картами или плоскостями). В каждой плоскости выполняется свертка выходного образа предшествующего слоя с фиксированным ядром небольшой размерности. Сверточные слои чередуются со слоями пулинга, кратно снижающими размерность пространства признаков. Слои пулинга не являются обязательными и существуют варианты их полного устранения из архитектуры сети [42]. Вторая отличительная особенность архитектуры сверточных сетей – использование полулинейных функций активации (RELU), выполняющих роль коммутационных ключей, управляемых значениями переменных скрытых слоев.

Недостатком архитектуры сверточных сетей является отсутствие теоретически обоснованных методов выбора структуры сети и параметров ядер свертки. Существуют несколько прекрасно работающих конфигураций сетей для решения конкретных задач, но не понятно, как строить сеть для новой задачи. До сих пор выбор структуры сверточной сети является предметом искусства. Второй существенный недостаток связан со временем обучения сверточных сетей, обусловленных значительным количеством настраиваемых параметров и необходимостью использования обучающих выборок большого объема. На типовом процессоре время обучения может варьироваться от нескольких часов до нескольких суток, поэтому для обучения сетей часто используют высокопроизводительные графические процессоры.

В настоящей главе будет показано, что небольшая модификация топологической модели быстрых алгоритмов приводит к многоплоскостным структурам, подобным структурам сверточных нейронных сетей. При этом удается сохранить алгоритм быстрого обучения и расширить информационную емкость сети по распознаваемым образам вплоть до максимально возможной, определяемой числом выходов сети. Предлагаемый вариант архитектуры

нельзя назвать сверточными сетями, поскольку в плоскостях нейронных слоев используются более общие преобразования, чем свертка, слои пулинга в преобразовании отсутствуют, но принцип пулинга применяется при обучении сети. Все нейроны имеют линейные функции активации, нелинейная обработка существует, но реализуется не за счет функций активации, а за счет коммутации плоскостей нейронных сетей. В какой-то мере это подобно коммутирующим полулинейным функциям активации RELU сверточных нейронных сетей. Управление коммутацией плоскостей осуществляется по координатам нейронов в выходной плоскости сети. Назовем данный класс сетей нейронными сетями с управляемой коммутацией плоскостей (Control Switching of Planes Neural Networks – CSPNN).

Известная архитектура сверточных сетей ориентирована на обработку изображений, входом сети являются несколько параллельных двумерных плоскостей, а выходом – одномерная плоскость классов. В этой главе будет рассмотрено построение как двумерных, так и одномерных сетей с управляемыми плоскостями. Предлагаемая топология идеологически близка к топологиям сверточных нейронных сетей, но является регулярной с числом слоев, устанавливаемым факторным представлением размерностей изображения и выходной плоскости классов. Алгоритм обучения имеет аналитическое представление, является абсолютно устойчивым и сходится за конечное число шагов. Дополнительные плоскости расширяют информационную емкость быстрой нейронной сети до максимально возможной, превращая ее в сеть глубокого обучения. Управление плоскостями в режиме обучения и обработки реализуется числовыми кодами координат выходной плоскости.

10.1. Одномерные нейронные сети с коммутацией плоскостей

Возьмем для определенности в качестве базовой топологию Кули–Тьюки «с прореживанием по времени» (см. (9.1)) и дополним модель плоскостями нейронных ядер, номера которых определим кортежем π_m:

$$U^m = \langle u_{n-1}u_{n-2}\dots u_{m+1}u_m v_{m-1}v_{m-2}\dots v_1 v_0 \rangle,$$
$$V^m = \langle u_{n-1}u_{n-2}\dots u_{m+1}v_m v_{m-1}v_{m-2}\dots v_1 v_0 \rangle,$$
$$z^m = \langle u_{n-1}u_{n-2}\dots u_{m+1}v_{m-1}v_{m-2}\dots v_1 v_0 \rangle,$$
$$\pi_m = \langle v_{n-1}v_{n-2}\dots v_{m+2}v_{m+1} \rangle.$$

Максимальное количество дополнительных плоскостей появится в нулевом слое. Номер плоскости в слое будет определяться кортежем $\pi_0 = \langle v_{n-1} v_{n-2} \ldots v_2 v_1 \rangle$. Плоскость с номером $\pi_0 = 0$ будем считать плоскостью базовой структуры. Число плоскостей в нулевом слое будет равно произведению оснований: $g_{n-1} g_{n-2} \ldots g_2 g_1$. По мере движения к выходному слою число дополнительных плоскостей будет уменьшаться, и для последнего слоя $\pi_{n-1} = \langle\ \rangle$, т. е. их не будет совсем, останется только одна плоскость базовой топологии. Таким образом, в новой топологии плоскость последнего слоя останется прежней, а в младших слоях появятся дополнительные плоскости. Номер ядра теперь следует уточнять его размещением в дополнительной плоскости. На рис. 10.1 показана новая топология, построенная на базе трехслойной сети с основаниями 2.

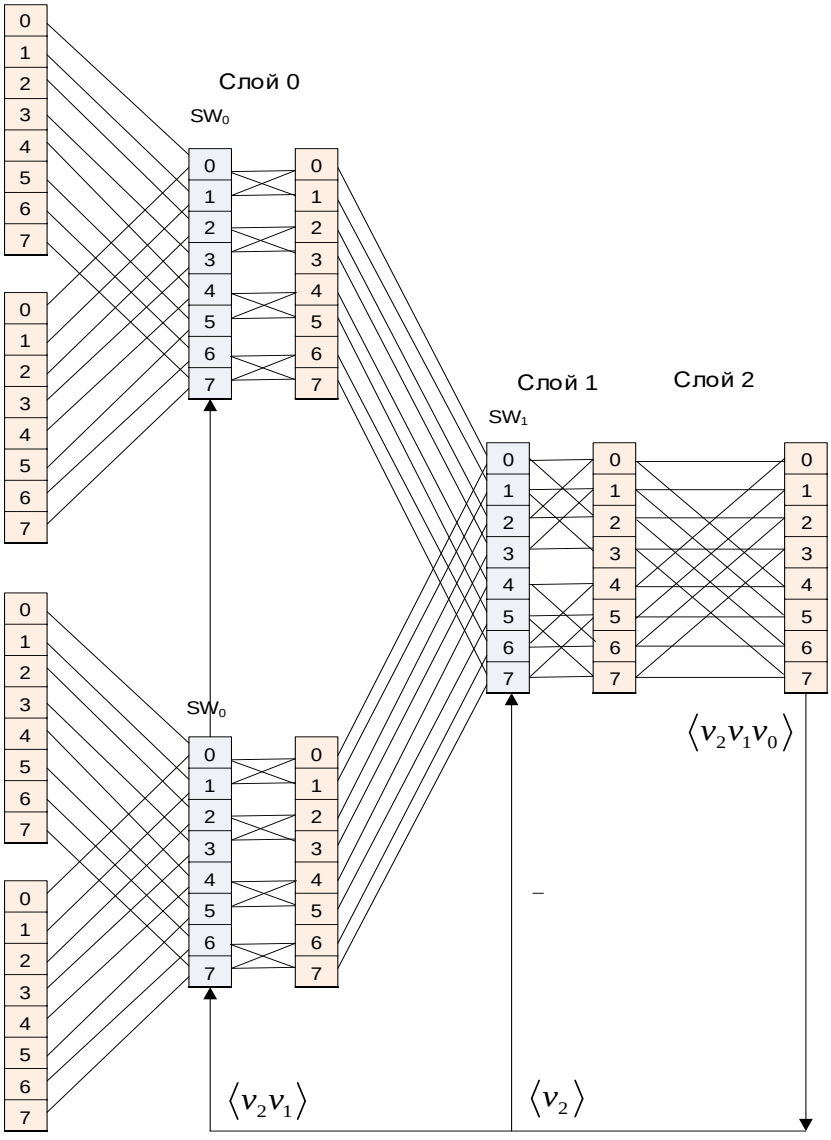


Рис. 10.1. Топология одномерной нейронной сети с дополнительными плоскостями

В сети условно показаны коммутаторы, но при программной реализации их фактически нет, они реализуются в составе алгоритма. Коммутаторы управляются разрядными переменными точек выходной плоскости. В выходной плоскости точки ассоциируются с классами, поэтому привязка класса к координатам выходной плоскости предопределяет выбор дополнительных плоскостей, которые используются для обработки входных образов. При построении одиночного классификатора все векторные входы параллельно объединяются. Сеть может быть использована и для параллельной классификации нескольких образов, тогда объединение векторных входов не требуется.

Поскольку правило порождения новых плоскостей не противоречит базовой топологической модели, то для настройки ядер преобразования к эталону можно использовать прежнее правило (6.4), расширив его аргументом для дополнительных плоскостей:

$$w^{m}_{z^m}\left\langle \pi_m \right\rangle \left(u_m, x_m\right) = \varphi^{k}_{i^m}\left(u_m\right),$$

здесь k – номер эталонной функции; $x = \left\langle x_{n-1}x_{n-2}\ldots x_0 \right\rangle$ – точка приспособления; $z^m = \left\langle u_{n-1}u_{n-2}\ldots u_{m+1}x_{m-1}x_{m-2}\ldots x_1x_0 \right\rangle$ – номер настраиваемых ядер по слоям; $\pi_m = \left\langle x_{n-1}x_{n-2}\ldots x_{m-1} \right\rangle$ – номер плоскости размещения ядер. Индекс k в правой части нумерует точку приспособления. Для $m = 0$ имеем $z^0 = i^0 = \left\langle u_{n-1}u_{n-2}\ldots u_1 \right\rangle$, а варьируемыми переменными в левой части являются номер плоскости $\pi_0 = \left\langle x_{n-1}x_{n-2}\ldots x_1 \right\rangle$ и разряд x_0. Вместе они покрывают весь диапазон координат выходной плоскости. Этому диапазону отвечают возможные значения индекса k в правой части, отсюда следует, что каждая точка выходной плоскости может быть связана с собственным эталоном. То есть построенная сеть обладает максимально возможным числом точек приспособления, покрывающих всю выходную плоскость и, таким образом, является сетью с глубокой степенью обучения.

К этому результату можно прийти с другой стороны. С каждой входной плоскостью $\pi_0 = \left\langle x_{n-1}x_{n-2}\ldots x_1 \right\rangle$ связаны область ответственности в выходной плоскости, определяемая схемой $V^{n-1} = \left\langle x_{n-1}x_{n-2}x_{n-3}\ldots x_1v_0 \right\rangle$. Размер области определяется основанием разрядной переменной v_0. Нетрудно видеть, что выделенный сегмент вход-выход представляет собой пирамидальную сеть. Таким образом, вся сеть с коммутацией плоскостей представляет собой лес непересекающихся пирамидальных сетей, регулярно упакованных

в матричные структуры. Как было показано ранее, пирамидальная сеть является сетью глубокого обучения, а значит, и вся сеть с коммутацией плоскостей является сетью глубокого обучения.

10.2. Вычислительная эффективность

В 8.3 было показано, что время обучения пирамидальной сети оценивается той же величиной, что и время обработки входного вектора нейронной сетью. Поскольку сеть с коммутацией плоскостей представляет собой лес однотипных пирамидальных сетей, то полученный результат справедлив и для данной сети. Число вычислительных операций можно определить, зная количество пирамидальных сетей в лесу и количество вычислительных операций в каждой из них. Количество сетей определяется числом входных плоскостей и, следовательно, равно $g_{n-1}g_{n-2}\dots g_2 g_1$. Число операций умножения в пирамидальной сети было определено в 8.3. Для оценки числа вычислительных операций, как и прежде, будем полагать, что для любого m имеет место $p_m = g_m = p$. Используя выражение (8.5) и учитывая количество пирамидальных сетей, получим

$$Z = \frac{p^2\left(p^n - 1\right)}{p-1} p^{n-1} = p^{n+1}\left(p^{n-1} + p^{n-2} + \dots + p^2 + p + 1\right) =$$

$$= p^{2n} + p^{n+1}\left(p^{n-2} + \dots + p^2 + p + 1\right).$$

При прямом матричном преобразовании оценочное количество операций равно p^{2n}. То есть сеть требует несколько больше операций, чем прямое преобразование. Выигрыш по вычислительной эффективности для сети достигается только при использовании процессоров с распараллеливанием операций и конвейерной обработки данных.

10.3. Двумерные сети с коммутацией плоскостей

Будем использовать прежнюю топологическую модель, но в двумерном варианте. Выразим координаты точки приспособления в позиционной системе счисления, обозначив разрядные переменные через y и x:

$$V_y = \left\langle y_{n-1}, y_{n-2} \dots y_0 \right\rangle, \quad V_x = \left\langle x_{n-1}, x_{n-2} \dots x_0 \right\rangle.$$

Фиксированные значения разрядных переменных y_m, x_m соответствуют переменным v_m^y, v_m^x, поэтому (как это следует из топологической модели) при

выборе данной точки приспособления должны настраиваться только ядра с номерами:

$$z_x^m = \left\langle u_{n-1}^x u_{n-2}^x \dots u_{m+1}^x x_{m-1} x_{m-2} \dots x_0 \right\rangle,$$

$$z_y^m = \left\langle u_{n-1}^y u_{n-2}^y \dots u_{m+1}^y y_{m-1} y_{m-2} \dots y_0 \right\rangle.$$

Номера плоскостей в пределах слоя определим правилом:

$$\pi_m = \left\langle x_{n-1} x_{n-2} \dots x_{m+1}, y_{n-1} y_{n-2} \dots y_{m+1} \right\rangle.$$

Очевидно, что максимальное количество дополнительных плоскостей будет в нулевом слое, а по мере увеличения номера слоя количество дополнительных плоскостей будет уменьшаться, и в последнем слое получим $\pi_{n-1} = \langle\ \rangle$, т. е. дополнительных плоскостей не будет совсем. Таким образом, в новой топологии плоскость последнего слоя останется прежней, а в младших слоях появятся дополнительные плоскости.

Архитектура сети с дополнительными плоскостями показана на рис. 10.2. При построении одиночного классификатора входное изображение подается одновременно на все плоскости входного слоя, слои разделены коммутаторами, которые управляются разрядными переменными координатных чисел выходной плоскости классов.

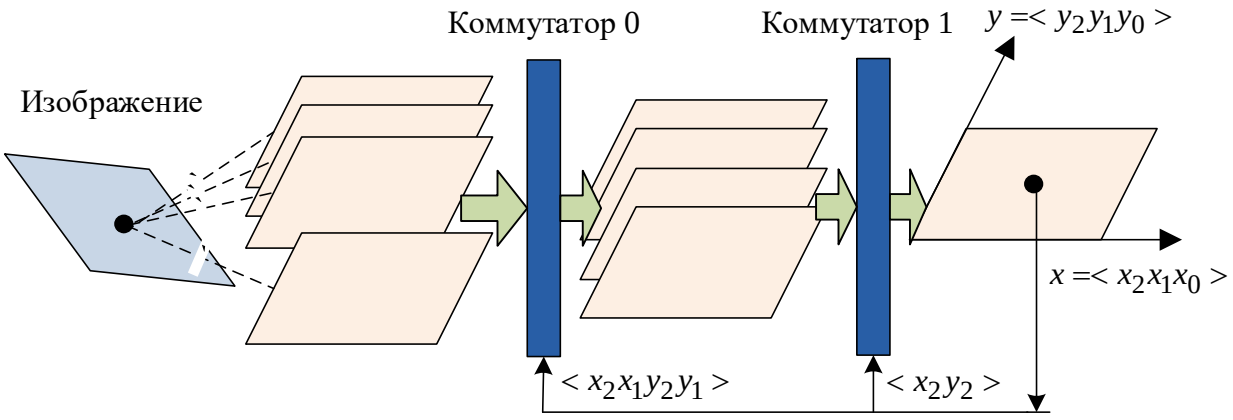


Рис. 10.2. Архитектура двумерной регулярной нейронной сети с дополнительными плоскостями

Для настройки ядер преобразования можно использовать прежнее правило (7.10), расширив его для дополнительных плоскостей дополнительным аргументом:

$$W^m_{z_x^m, z_y^m} \langle \pi_m \rangle \left(u_m^y u_m^x ; v_m^y v_m^x \right) = f^k_{i_x^m\, i_y^m} \left(u_m^y, u_m^x \right).$$

Индекс k в правой части нумерует точку приспособления. Для $m=0$ имеем $z_x^0 = i_x^0$ и $z_y^0 = i_y^0$, а варьируемыми переменными в левой части являются номер плоскости $\pi_0 = \langle x_{n-1}x_{n-2}\ldots x_1, y_{n-1}y_{n-2}\ldots y_1\rangle$ и разрядные переменные $v_0^y v_0^x$. Вместе они покрывают весь диапазон координат выходной плоскости. Этому диапазону отвечают возможные значения индекса k в правой части последнего выражения. Таким образом, преобразование с дополнительными плоскостями может быть приспособлено к $D = M_y M_x$ изображениям, т. е. каждой точке выходной плоскости будет соответствовать один опорный образ. Это означает, что построенная сеть является сетью с глубокой степенью обучения.

10.4. Выводы

В главе показано, что топология БНС легко расширяется дополнительными плоскостями, при этом число распознаваемых образов кардинально возрастает и покрывает все элементы выходной плоскости. Причем расширение топологии не нарушает принципа построения обучающего алгоритма. Предложенные архитектуры реализуют сети с глубокой степенью обучения. Показано, что они сегментируются в лес независимых пирамидальных сетей. Это порождает уникальное качество – возможность дообучения сети к новым образам без изменения или потери ранее накопленных знаний. Сети могут работать в системах реального времени, поскольку время обучения не превышает времени обработки данных в сети. Алгоритмы обучения являются абсолютно устойчивыми и завершаются за конечное число шагов. Построенная топология двумерных сетей идеологически близка к топологиям сверточных сетей глубокого обучения, но является регулярной. Выбор структуры сети не вызывает проблем и определяется системными инвариантами самоподобных сетей. Возможны различные структурные решения, обладающие одинаковыми качествами. Представленное решение позволяет дать конструктивные ответы на принципиальные вопросы нейронных сетей глубокого обучения: как выбрать топологию глубокой нейронной сети? Как сократить время обучения сети?

11. ЭЛЕМЕНТЫ ПАМЯТИ И ЛОГИКИ НА НЕЙРОННЫХ СЕТЯХ БЫСТРОГО ОБУЧЕНИЯ

Реализация памяти в нейронных сетях поддерживается большим классом нейронных сетей с обратными связями, к ним относятся сети Элмана, Хопфилда, Хэмминга, АРТ и другие. Эти сети решают задачи восстановления искаженных образов, ассоциативной памяти, кратковременной динамической памяти и пр. Память в этих сетях используется в контексте распознавания образов, задача хранения и точного восстановления образа не ставится. Напротив, в настоящей главе будет рассмотрено использование быстрых нейронных сетей для точного хранения и восстановления образов, так как это принято для хранения образов в памяти ЭВМ. В этом же ключе будет показано использование БНС для реализации типового логического элемента – дешифратора кодов для различных оснований.

11.1. Реализация элемента памяти на обратно-ориентированной корреляционной БНС

Реализация элементов памяти предполагает сохранение образа в синапсах нейронной сети и точное восстановление его на выходе сети при подаче входного сигнала. Возьмем для определенности БНС в топологии Кули–Тьюки «с прореживанием по времени»:

$$\begin{aligned} U^m &= \langle u_{n-1}u_{n-2}\dots u_{m+1}u_m v_{m-1}v_{m-2}\dots v_1 v_0\rangle, \\ V^m &= \langle u_{n-1}u_{n-2}\dots u_{m+1}v_m v_{m-1}v_{m-2}\dots v_1 v_0\rangle, \\ z^m &= \langle u_{n-1}u_{n-2}\dots u_{m+1}v_{m-1}v_{m-2}\dots v_1 v_0\rangle. \end{aligned} \qquad (11.1)$$

Элементы матрицы преобразования БНС выражаются через элементы нейронных ядер в виде:

$$h(U,V) = w^{n-1}_{z^{n-1}}(u_{n-1},v_{n-1})\, w^{n-2}_{z^{n-2}}(u_{n-2},v_{n-2})\dots w^{0}_{z^{0}}(u_0,v_0).$$

Рассмотрим сеть с размерностью по входу равной единице, в этом случае $U=0$, и топологическая модель сети примет вид:

$$\begin{aligned} U^m &= \langle 0_{n-1}0_{n-2}\dots 0_{m+1}0_m v_{m-1}v_{m-2}\dots v_1 v_0\rangle, \\ V^m &= \langle 0_{n-1}0_{n-2}\dots 0_{m+1}v_m v_{m-1}v_{m-2}\dots v_1 v_0\rangle, \\ z^m &= \langle 0_{n-1}0_{n-2}\dots 0_{m+1}v_{m-1}v_{m-2}\dots v_1 v_0\rangle. \end{aligned}$$

В свою очередь элементы матрицы преобразования будут представлены в виде:

$$h(U,V)=w^{n-1}_{z^{n-1}}(0_{n-1},v_{n-1})\,w^{n-2}_{z^{n-2}}(0_{n-2},v_{n-2})\ldots w^{0}_{z_0}(0_0,v_0). \qquad (11.2)$$

Размерность сети по выходу $M=g_0g_1\ldots g_{n-1}$ определяется произведением оснований разрядных переменных v_i. Топология сети для структурных параметров $[p_0p_1p_2p_3]=[1111]$, $[g_0g_1g_2g_3]=[2222]$ показана на рис. 11.1. Все нейронные ядра по входу имеют размерность единица, а по выходу – два.

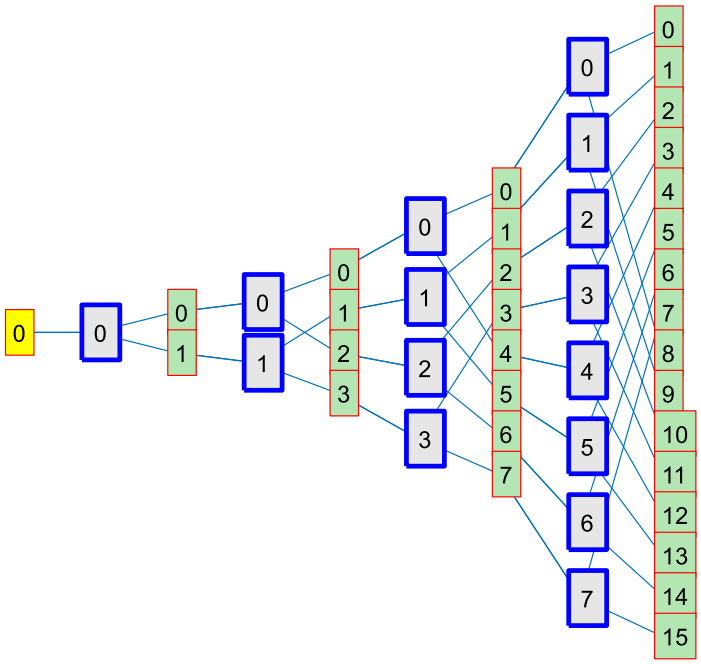


Рис. 11.1. Топология БНС для реализации элемента памяти

Пусть запоминаемый образ $y(V)$ представляет собой дискретную функцию, заданную на интервале длиной M. Выполним мультипликативную декомпозицию этой функции по переменным v_i, начиная со старшего разряда (см. замечание к 6.1), в результате получим:

$$y(V)=\varphi_{i^{n-1}}(v_{n-1})\,\varphi_{i^{n-2}}(v_{n-2})\ldots\varphi_{i^{0}}(v_0), \qquad (11.3)$$

где $i^m=\langle v_{m-1}v_{m-2}\ldots v_1v_0\rangle$. Сравнивая выражения (11.2) и (11.3), получим правило обучения нейронной сети памяти:

$$w^{m}_{z^m}(0_m,v_m)=\varphi_{i^m}(v_m),\quad z^m=i^m.$$

Считывание памяти происходит при подаче на вход нейронной сети значения $x(0)=1$. При значении $x(0)=-1$ выходной сигнал изменяет знак.

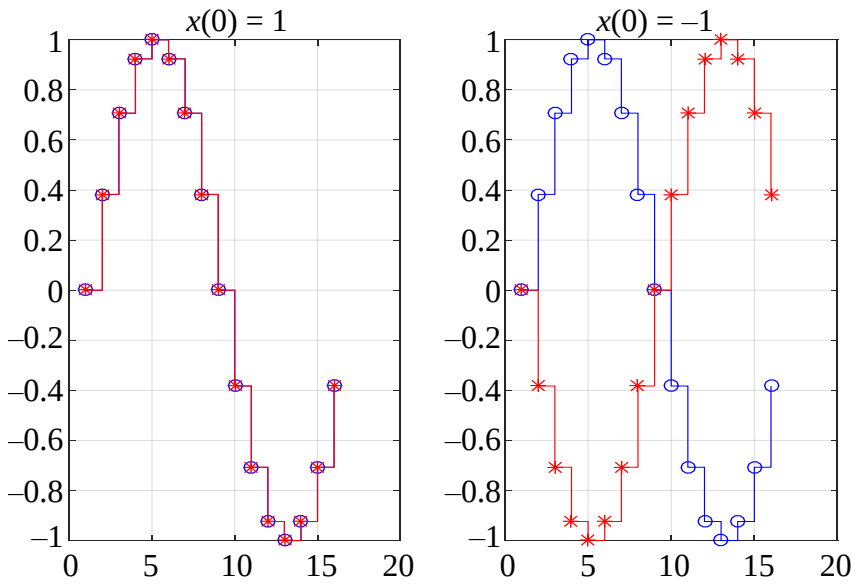


Рис. 11.2. Сохранение и считывание элемента памяти на БНС

На рис. 11.2 показаны графики хранимого сигнала в БНС (синий цвет, маркер «o») и считываемого сигнала (красный цвет, маркер «*») при полярных входных значениях сети.

11.2. Элементы памяти на обратно-ориентированных пирамидальных нейронных сетях

Возьмем за основу топологическую модель Кули–Тьюки «с прореживанием по времени». Модель описывается выражением (11.1). Выберем структурные характеристики следующим образом:

• размерности рецепторных полей сети положим равными $p_0 = p_1 = \ldots = p_{n-2} = 1$, $p_{n-1} \neq 1$. В этом случае размерность сети по входу будет равна $N = p_{n-1}$;

• размерности аксоновых полей сети зададим произвольными натуральными числами $g_0, g_1, \ldots, g_{n-1}$. Размерность сети по выходу в этом случае будет равна $M = g_0 g_1 \ldots g_{n-1}$.

Топологическая модель при данных структурных характеристиках будет иметь вид:

$$U^m = \langle u_{n-1} 0_{n-2} \ldots 0_{m+1} 0_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle,$$
$$V^m = \langle u_{n-1} 0_{n-2} \ldots 0_{m+1} v_m v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle,$$
$$z^m = \langle u_{n-1} 0_{n-2} \ldots 0_{m+1} v_{m-1} v_{m-2} \ldots v_1 v_0 \rangle.$$

Топология сети для структурных параметров $[p_0 p_1 p_2 p_3] = [1112]$ $[g_0 g_1 g_2 g_3] = [2222]$ показана на рис. 11.3.

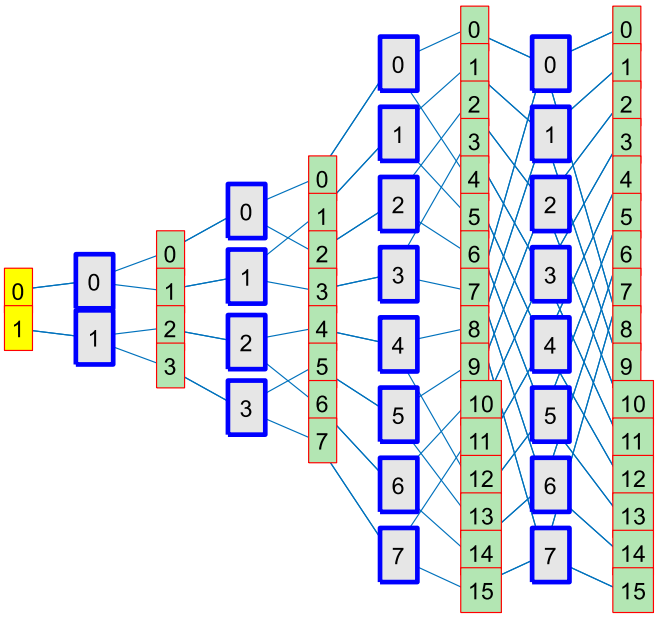


Рис. 11.3. Обратно-ориентированная пирамидальная сеть для реализации памяти

Пусть запоминаемые образы $y^k(V)$ представляют собой набор, состоящий из p_{n-1} дискретных функций, заданных на интервале длиной M. Выполним мультипликативную декомпозицию каждой функции по переменным v_i, начиная со старшего разряда, в результате получим:

$$y^k(V) = \varphi^k_{i^{n-1}}(v_{n-1})\varphi^k_{i^{n-2}}(v_{n-2})\ldots\varphi^k_{i^0}(v_0), \qquad (11.4)$$

где $i^m = \langle v_{m-1}v_{m-2}\ldots v_1v_0\rangle$, k – порядковый номер функции. Сравнивая выражения (11.2) и (11.4), получим правило обучения нейронной сети:

$$w^{n-1}_{z^{n-1}}(u_{n-1}, v_{n-1}) = \varphi^k_{i^{n-1}}(v_{n-1}) \quad \text{для } m = n-1,$$

$$w^m_{z^m}(0_m, v_m) = \varphi^k_{i^m}(v_m) \quad \text{для } m < n-1,$$

$$z^m = \langle u_{n-1}0_{n-2}\ldots 0_{m+1}v_{m-1}v_{m-2}\ldots v_1v_0\rangle,$$

$$i^m = \langle v_{m-1}v_{m-2}\ldots v_1v_0\rangle.$$

Предварительно для упорядочивания хранимых функций должно быть установлено взаимно-однозначное соответствие $k \Leftrightarrow u_{n-1}$ между порядковым номером функции и разрядной переменной u_{n-1}. Считывание памяти реализуется установкой на входе нейронной сети унитарного кода, в котором только один из разрядов равен 1, а остальные нули. Для сети, показанной на рис. 11.3, это будут кодовые комбинации $[01]$ или $[10]$. При использовании

кодов $[11]$ хранимые образы складываются, а при кодах $[1\ -1]$ из первого образа вычитается второй. На рис. 11.4 показаны результаты считывания образов, представляющих собой дискретные функции синуса, отличающиеся по частоте в два раза.

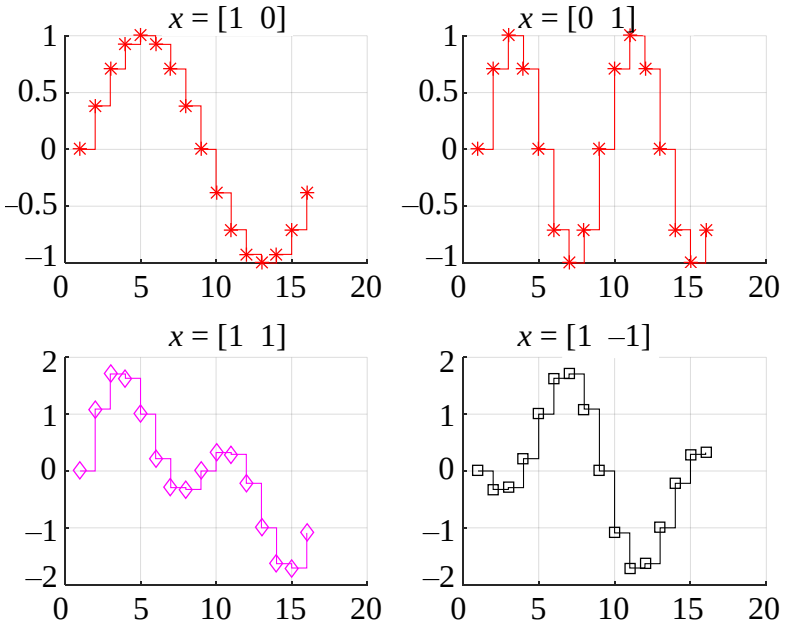


Рис. 11.4. Результаты считывания памяти, реализованной на пирамидальной БНС

На рис. 11.5 показана структурная модель нейронной сети, представленной на рис. 11.3. Каждая вершина структурной модели является нейронным ядром, пара цифр, записанных в вершине, определяют размерность рецепторного и аксонового поля нейронного ядра, все связи между вершинами имеют ранги, равные единице.

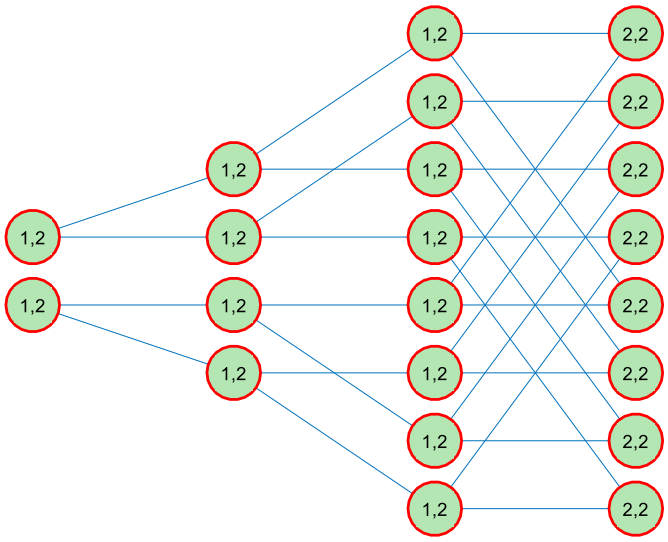


Рис. 11.5. Структурная модель обратно-ориентированной пирамидальной нейронной сети

В гл. 12 получена формула расчета числа степеней свободы БНС, которая имеет вид:

$$S(H)=\sum_{m=0}^{n-1} p_m g_m q_m - \sum_{m=0}^{n-2} D_m,$$

где p_m, g_m – размерности рецепторного и аксонового поля ядра в слое m; q_m – число ядер в слое; D_m – количество одноранговых связей в межслойном переходе с номером m. Непосредственно из структурной модели следует $S(H)=(1\cdot 2)\cdot 14+(2\cdot 2)\cdot 8-28=32$. Данная нейронная сеть обеспечивает хранение двух произвольных дискретных функций, заданных на интервале длиной $M=16$. Для задания двух функций требуется определить 32 значения. Таким образом, сеть полностью использует свой потенциал для хранения функций и поэтому относится к категории сетей с глубокой степенью обучения.

11.3. Банк памяти на обратно-ориентированной сети с коммутацией плоскостей

Дополним топологическую модель Кули–Тьюки «с прореживанием по времени» дополнительными плоскостями нейронных ядер, номера которых определим кортежем π_m:

$$U^m=\langle u_{n-1}u_{n-2}\ldots u_{m+1}u_m v_{m-1}v_{m-2}\ldots v_1 v_0\rangle,$$
$$V^m=\langle u_{n-1}u_{n-2}\ldots u_{m+1}v_m v_{m-1}v_{m-2}\ldots v_1 v_0\rangle,$$
$$z^m=\langle u_{n-1}u_{n-2}\ldots u_{m+1}v_{m-1}v_{m-2}\ldots v_1 v_0\rangle,$$
$$\pi_m=\langle u_{m-1}u_{m-2}\ldots u_1 u_0\rangle.$$

Максимальное количество дополнительных плоскостей появится в последнем слое. Номер плоскости в этом слое будет определяться кортежем $\pi_{n-1}=\langle u_{n-2}u_{n-3}\ldots u_1 u_0\rangle$, а число плоскостей будет равно произведению оснований: $p_{n-2}p_{n-3}\ldots p_1 p_0$. По мере движения к входному слою число дополнительных плоскостей будет уменьшаться, и для нулевого слоя имеем $\pi_0=\langle\ \rangle$, т. е. их не будет совсем, останется только одна плоскость базовой топологии. Таким образом, в новой топологии плоскость нулевого слоя останется прежней, а в старших слоях появятся дополнительные плоскости. На рис. 11.6 показана топологическая модель сети памяти со структурными характеристиками: $[p_0p_1p_2p_3]=[2222]$, $[g_0g_1g_2g_3]=[2222]$.

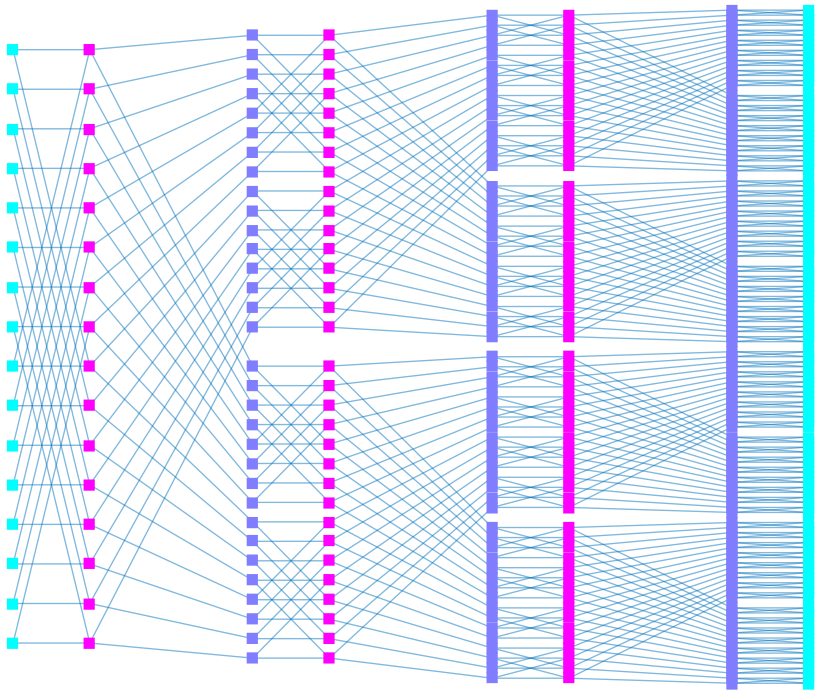


Рис. 11.6. Топологическая модель нейронной сети банка памяти

Будем полагать, что запоминаемые образы представлены в мультипликативной форме (11.4). В качестве образов могут использоваться произвольные функции, заданные на дискретном интервале длиной $M = g_0 g_1 \ldots g_{n-1}$. Запись образа в память соответствует обучению нейронной сети. Для настройки нейронных ядер можно использовать базовое правило, вытекающее из обобщенной теоремы факторизации БНС, дополнив его номером плоскости:

$$w^{m}_{z^m}\left\langle \pi_m \right\rangle (u_m, v_m) = \varphi^{k}_{i^m}(v_m),$$

здесь k – порядковый номер функции; $V = \left\langle v_{n-1} v_{n-2} \ldots v_0 \right\rangle$ – точка выходной плоскости; $z^m = \left\langle u_{n-1} u_{n-2} \ldots u_{m+1} v_{m-1} v_{m-2} \ldots v_1 v_0 \right\rangle$ – номер настраиваемых ядер по слоям; $\pi_m = \left\langle u_{m-1} u_{m-2} \ldots u_1 u_0 \right\rangle$ – номер плоскости размещения ядер. Для $m = n-1$ имеем $z^{n-1} = i^{n-1} = \left\langle v_{n-2} v_{n-3} \ldots v_1 v_0 \right\rangle$, а варьируемыми переменными в левой части являются номер плоскости $\pi_{n-1} = \left\langle u_{n-2} u_{n-3} \ldots u_1 u_0 \right\rangle$ и разряд u_{n-1}. Вместе они покрывают весь диапазон координат входной плоскости. Этому диапазону отвечают возможные значения индекса k в правой части, отсюда следует, что каждая точка входной плоскости является активирующим адресом для хранимого образа. То есть построенная сеть обладает максимально возможной памятью с адресным пространством, покрывающим всю входную плоскость, и, следовательно, является сетью памяти с глубокой сте-

пенью обучения. К этому результату можно прийти с другой стороны. С каждой выходной плоскостью $\pi_{n-1} = \langle u_{n-2} u_{n-3} \ldots u_1 u_0 \rangle$ связана область ответственности во входной плоскости, определяемая схемой $U^0 = \langle u_{n-1} u_{n-2} \ldots u_1 u_0 \rangle$. Размер области определяется основанием разрядной переменной u_{n-1}. Нетрудно видеть, что выделенный сегмент вход-выход представляет собой пирамидальную обратно-ориентированную сеть памяти. Таким образом, вся сеть с коммутацией плоскостей представляет собой лес непересекающихся пирамидальных сетей памяти, регулярно упакованных в матричные структуры. Как было показано ранее, пирамидальная сеть является сетью глубокого обучения, а значит, и вся сеть с коммутацией плоскостей является сетью глубокого обучения. Считывание памяти реализуется установкой на входе нейронной сети унитарного кода в области ответственности каждой выходной плоскости. Хранимые образы воспроизводятся в выходных плоскостях сети. В любой выходной плоскости может воспроизводиться p_{n-1} образов, независимо от воспроизведения хранимых образов в других областях. Сеть памяти, показанная на рис. 11.6, имеет 8 выходных плоскостей, с каждой плоскости можно считывать две функции. Таким образом, нейронная сеть обеспечивает хранение 16 произвольных функций, заданных на интервале длинной $M = 16$. Данная сеть имеет 256 степеней свободы.

При программной реализации нейронной сети и ограничении числа одновременно отображаемых образов до одного можно реализовать считывание всех образов с одного выхода. Это достигается за счет неявного переключения плоскостей по значению адреса считывания U. Адрес считывания в поразрядном представлении имеет вид $U = \langle u_{n-1} u_{n-2} \ldots u_1 u_0 \rangle$, поэтому при считывании образа по данному адресу должны быть выбраны плоскости $\pi_m = \langle u_{m-1} u_{m-2} \ldots u_1 u_0 \rangle$, где m – номер слоя. Такой прием выделяет из сети только те нейронные цепи, которые ответственны за хранение данного образа, сокращая до минимума число вычислительных операций при его восстановлении.

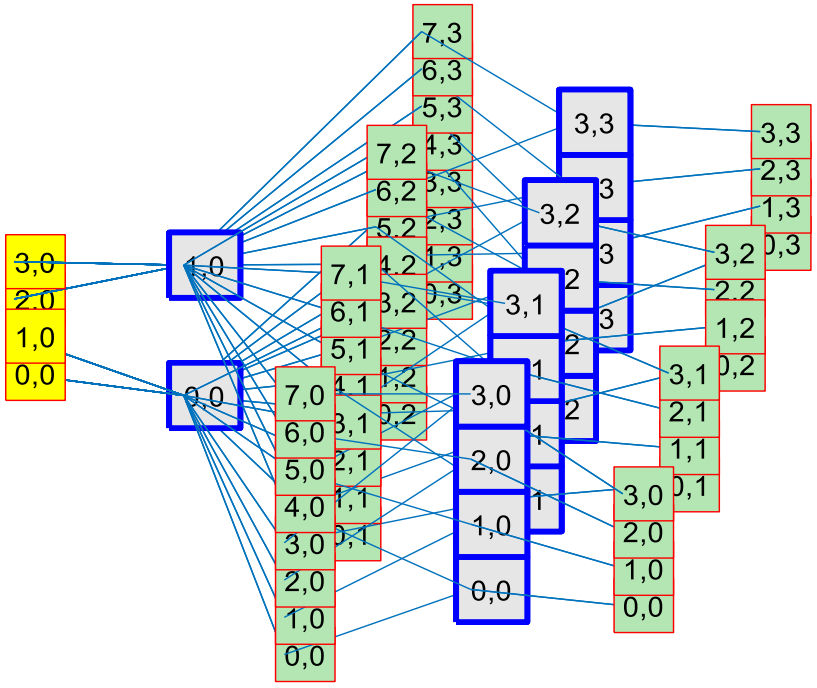
В памяти на основе БНС могут храниться двумерные и многомерные образы. В отличие от кристаллов цифровой памяти с последовательным хранением данных, где восстановление образа происходит за счет последовательного опроса ячеек, в памяти на БНС все пикселы образа восстанавливаются одновременно, что потенциально обеспечивает сверхвысокое быстродействие при считывании памяти.

11.4. Логические элементы на пирамидальных нейронных сетях

Логический дешифратор. Пирамидальные сети быстрого обучения могут быть точно настроены на произвольные эталонные функции. Количество эталонов определяется выбранной топологией сети. Будем полагать, что настройка сети выполнена к эталонным функциям, нормированным по энергии к единице и представляющим собой логические коды по основанию 2. В рабочем режиме сеть выполняет вычисление скалярных произведений входной вектор-функции с эталонными функциями. Если входная функция нормирована по энергии к единице и совпадает с эталоном, то на соответствующем выходе нейронной сети получим единичное значение, в то время как остальные выходы будут иметь значения меньше единицы. Это обстоятельство можно использовать для построения дешифраторов логических кодов.

Рассмотрим, например, построение дешифратора для четырехбуквенных кодовых слов. Для кода с основанием 2 дешифратор должен иметь 4 входа и 16 выходов. Нейронная сеть должна иметь такие же размерности по входу и выходу. Выберем двухслойную сеть, одномерную по входу и двумерную по выходу, топология сети показана на рис. 11.7. Сеть обучается к кодам, представленным в таблице на этом же рисунке. Коды рассматриваются как вектор-функции. Обучающие эталонные функции формируются из кодов нормированием по энергии к единице. Сеть имеет структурные характеристики:

NN: PX=[2 2], PY=[1 1], GX=[4 1], GY=[4 1]



Входной код	Номер выхода
0000	0,0
0001	1,0
0010	2,0
0011	3,0
0100	0,1
0101	1,1
0110	2,1
0111	3,1
1000	0,2
1001	1,2
1010	2,2
1011	3,2
1100	0,3
1101	1,3
1110	2,3
1111	3,3

Рис. 11.7. Дешифратор логических кодов на пирамидальной нейронной сети

по рецепторным полям $P_x = [2\,2]$, $P_y = [11]$, по аксоновым полям $G_x = [41]$, $G_y = [41]$. Результаты классификации входных кодов показаны на рис. 11.8. Максимальный уровень выхода при классификации ближайшего кода не превышает значения 0.886. На этот уровень или выше должны быть настроены компараторы логического нуля на выходах сети.

Проблемным является входной код [0000]. Скалярное произведение этого кода с эталоном будет равно нулю, в то время как в классическом дешифраторе оно должно равняться единице.

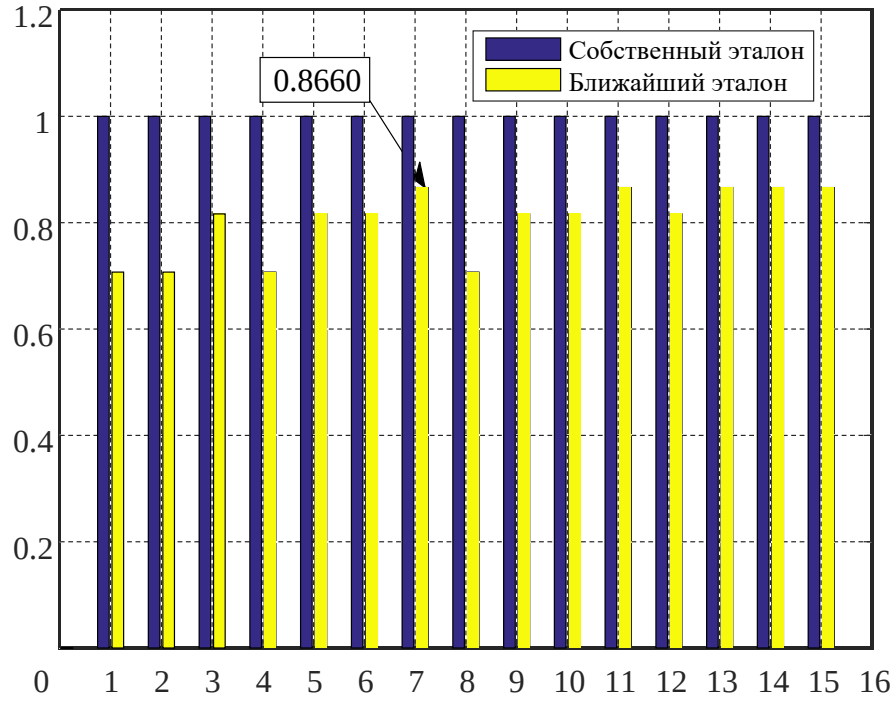


Рис. 11.8. Классификация логических кодов пирамидальной нейронной сетью

Решением является видоизменение нулевого кода, например, замена одного из нулей кодового слова значением (–1). Более радикальный вариант – замена во всех кодовых словах значения (0) на значение (–1), т. е. использование двухполярного бинарного кода со значениями (–1,+1). Результаты классификации входных кодов для этого случая показаны на рис. 11.9. Выходные компараторы логического нуля должны быть настроены на уровень 0.5+.

Подобным образом могут быть построены дешифраторы кодов с основанием больше двух. Дешифратор можно использовать в комбинации с элементом памяти на БНС с коммутацией плоскостей, в этом случае адресация памяти будет подобна адресации цифровой памяти с произвольным доступом, но адресуемой единицей здесь является не ячейка памяти для одного числа, а образ в виде числового вектора, изображения или многомерного куба.

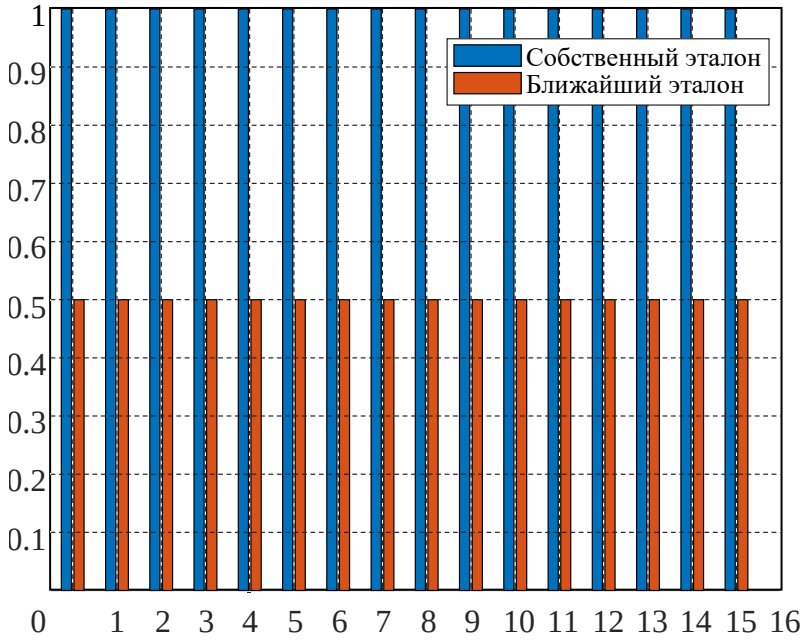


Рис. 11.9. Классификация двух полярных бинарных кодов

Логический шифратор. Шифратор выполняет операцию, обратную дешифрированию, т. е. преобразует унарный код в кодовые слова. Рассмотрим построение шифратора для двоичного четырехразрядного кода. Нейронная сеть по входу должна иметь размерность 16, а по выходу 4. Операцию, выполняемую нейронной сетью, можно представить как умножение унарного вектор-строки длиной 16 на матрицу кодовых слов размерностью 16×4.

$$OUT = [V_3V_2V_1V_0] = [U_0U_1U_2U_3U_4U_5U_6U_7U_8U_9U_{10}U_{11}U_{12}U_{13}U_{14}U_{15}] \begin{bmatrix} 0000 \\ 0001 \\ 0010 \\ 0011 \\ 0100 \\ 0101 \\ 0110 \\ 0111 \\ 1000 \\ 1001 \\ 1010 \\ 1011 \\ 1100 \\ 1101 \\ 1111 \end{bmatrix}$$

Унарный вектор-строка имеет единицу только по одной координате, все остальные равны нулю. Для выполнения матричного умножения можно использовать обратно-ориентированную пирамидальную сеть памяти, один из возможных вариантов топологической модели сети показан на рис. 11.10.

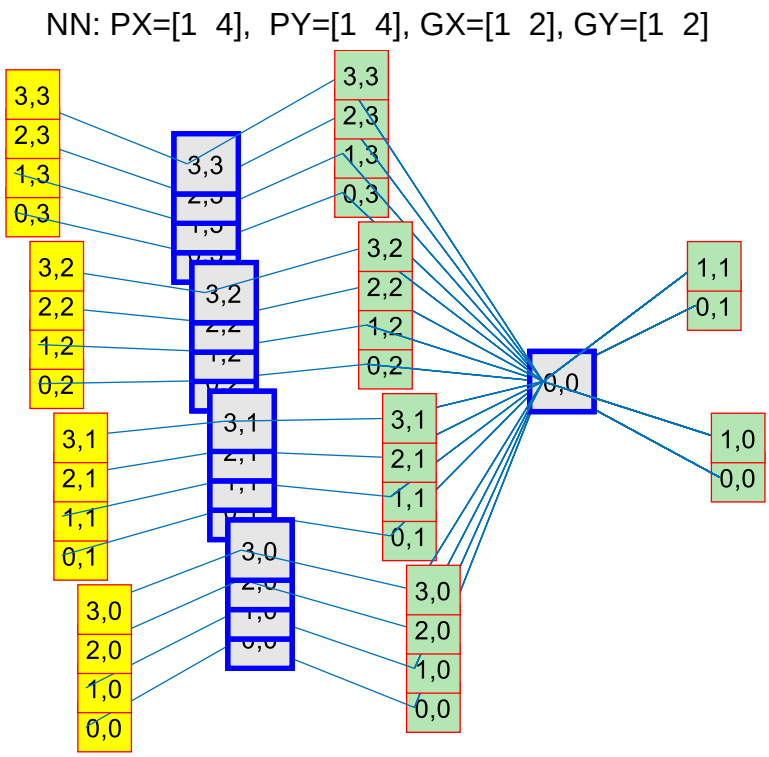


Рис. 11.10. Топологическая модель двумерной сети, реализующей матричное умножение

Двуслойная нейронная сеть двумерна по входу и по выходу, структурные характеристики сети приведены на рисунке.

12. ПЛАСТИЧНОСТЬ БЫСТРЫХ НЕЙРОННЫХ СЕТЕЙ

В исследованиях нейробиологов термин «пластичность» используется, как качественная характеристика способности нейронной сети обучаться при воздействии внешних факторов. Для искусственных нейронных сетей адекватной количественной оценкой может служить число независимых настроек, существующих в сети. Это значение, как правило, меньше полного количества синаптических весов (исключением является однослойный персептрон, для которого соблюдается равенство).

В механике для оценки числа независимых координат используется понятие «число степеней свободы». Близкую аналогию можно провести и для нейронных сетей. Действительно, нейронную сеть прямого распространения можно представить как нелинейный оператор, преобразующий входной вектор в выходной. Полное множество операторов образует многообразие[1] в многомерном пространстве, в котором каждый оператор сети можно рассматривать как некоторую материальную точку. Изменение синаптических весов нейронной сети приводит к перемещению точки-оператора в пространстве операторов. Следуя далее механической аналогии, будем называть число независимых координат, необходимое и достаточное для однозначного определения местоположения точки-оператора в пространстве операторов, числом степеней свободы нейронной сети и рассматривать эту величину как количественную оценку пластичности сети.

В предложенном способе оценки пластичности неявно предполагается, что операторное многообразие нейронной сети локально дифференцируемо по всем координатам, т. е. является непрерывным и гладким. Поскольку оценивается локальная размерность линейного пространства, то в контексте оценки пластичности можно ограничиться линейными операторами. Более того, для оценки достижимых свойств предлагается использовать максимальную локальную размерность пространства операторного образа. Понятно, что данная оценка не изменится, если вместо полного операторного многообразия нейронной сети рассматривать только его часть (трансверсальную область), в которой размерность локального пространства достигает максимума. Эту выделенную часть обычно называют моделью «общего положения».

---

[1] Многообразие – геометрический объект, локально устроенный как векторное линейное пространство.

12.1. Системные модели «общего положения»

Концепция «общего положения» в моделировании является выражением фундаментального понятия «типичности» при совместном размещении геометрических образов в векторном пространстве. В задаче проектирования использование моделей общего положения сужает область поиска оптимальных решений, в задаче анализа позволяет оценить потенциально достижимые возможности исследуемой системы. В обоих случаях это приводит к упрощению методов исследования.

Операторное многообразие нейронной сети формируется взаимным объединением операторных многообразий нейронных модулей. Локально каждое операторное многообразие устроено как некоторое линейное подпространство в пространстве операторов, поэтому многообразия локально пересекаются по векторным подпространствам. Для двух подпространств L_1, L_2 векторного пространства L под общим положением понимается [24] такое их размещение, при котором их пересечение имеет минимальную, а сумма – максимальную размерность (под суммой или объединением подпространств понимается их линейная оболочка). Другой термин для этого же понятия – L_1, L_2 пересекаются трансверсально. Подпространства в «общем положении» не пересекаются, а если пересекаются, то размерность их объединения $\dim L_1 \cup L_2 = \dim L_1 + \dim L_2 - \dim L_1 \cap L_2$. Название «общее положение» обусловлено тем, что в некотором смысле большинство пар подпространств L_1, L_2 находятся в общем положении, а другие расположения являются вырожденными. Примеры общего положения: 1) две прямые двумерного пространства, пересекающиеся в одной точке; 2) непересекающаяся пара прямых в трехмерном пространстве; 3) пара плоскостей трехмерного пространства, пересекающихся вдоль линии, и т. д. Теоретическим обоснованием моделей «общего положения» являются теоремы Сарда [43], раскрывающие смысл типичности и большинства. По теореме Сарда, трансверсальных отображений настолько много, что их можно обнаружить в сколь угодно малой окрестности любого гладкого отображения.

Для модульной нейронной сети задача расчета степени пластичности разделяется на две подзадачи: в первой требуется определить пластичность отдельных модулей, а во второй – используя полученные данные и информацию о структуре сети, определить пластичность всей сети, следуя при этом принципу трансверсальности при объединении операторных многообразий. Рассмотрим обе эти задачи последовательно.

12.2. Операторные многообразия нейронных модулей

Как уже отмечалось, в контексте оценки пластичности нейронной сети можно полагать, что нейронный модуль описывается линейным оператором. Пусть E и D – ассоциированные пространства входа и выхода для модуля A с размерностями P и g. Ограничиваясь моделью общего положения, будем считать, что при варьировании параметров операторное многообразие модуля совпадает с многообразием линейных отображений $\mathrm{A}_r: E \to D$ ранга r. В [44] было показано, что множество матриц ранга r, имеющих размеры $p \times g$, образуют многообразие размерностью $\dim A_r = gr + pr - r^2$. Покажем, что этот результат применим к операторному множеству нейронного модуля.

Как известно [45], полное множество линейных операторов, действующих из пространства E в D, изоморфно тензорному произведению $E \otimes D$, которое является линейным пространством размерности $p \cdot g$. По теореме о структуре линейного отображения [43] для каждого оператора ранга r существуют такие прямые разложения пространств

$$E = E_0 \oplus E_1 \Leftrightarrow D_1 \oplus D_0 = D, \tag{12.1}$$

что D_1 изоморфно E_1 и имеет размерность r, а E_0 составляет ядро отображения. Подпространство D_0 называют коядром, оно характеризует степень неопределенности (неоднозначности) оператора.

Из разложения (12.1) нетрудно видеть, что подмножество операторов A_r представляет собой объединение подмножества операторов ранга r, осуществляющих отображение из пространства E в D_1, и подмножества операторов ранга r, осуществляющих отображение из E_1 в D. На языке тензорных произведений это можно записать так:

$$A_r \cong \left(E \otimes D_1\right) \cup \left(E_1 \otimes D\right).$$

Отсюда, следуя правилу вычисления размерности объединения подпространств общего положения [43], получим:

$$\dim A_r = \dim\left(E \otimes D_1\right) + \dim\left(E_1 \otimes D\right) - \dim\left(E \otimes D_1\right) \cap \left(E_1 \otimes D\right). \tag{12.2}$$

Пересечением множеств является подмножество операторов, осуществляющих отображение из подпространства E_1 в D_1, поэтому

$$\dim\left(E \otimes D_1\right) \cap \left(E_1 \otimes D\right) = r^2.$$

Поскольку $\dim E = g$, $\dim D = p$, $\dim E_1 = \dim D_1 = r$, то из (12.2) следует:

$$\dim A_r = gr + pr - r^2.$$

Модуль двойственного функционирования в составе сети. Обозначим $\mathrm{A}_r^{\#}: D \to E$ множество всех операторов ранга r, действующих из пространства D в пространство E. Разложение (12.1) симметрично для класса прямых и обратных отображений, поэтому $\mathrm{A}_r^{\#}$ также представляет собой многообразие, и существует естественный изоморфизм $\mathrm{A}_r \cong \mathrm{A}_r^{\#}$, который задается совпадением пар (E_1, D_1). Данный изоморфизм является выражением двойственности в представлении модуля (рис. 12.1).

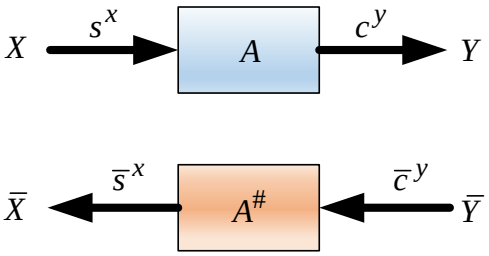


Рис. 12.1. Нейронные модули в прямой и двойственной сети

В режимах прямого и обратного функционирования размерность выходного пространства операторного модуля не может быть больше r. Следовательно, модуль может реализовать не более чем gr степеней свободы при прямом функционировании и не более чем pr – при обратном. При этом r^2 степеней свободы являются общими. Если модуль находится в составе сети, то размерности входных сигнальных пространств $X \subseteq E$ и $\bar{Y} \subseteq D$ в прямой и двойственной сетях в общем случае меньше максимально возможных. Это ограничивает возможности по реализации степеней свободы модуля. Действующее число степеней свободы модуля в составе сети

$$\dim \hat{A}_r = s^x \hat{r} + \bar{c}^y \hat{r} - \hat{r}^2, \tag{12.3}$$

где $s^x, \bar{c}^y$ – размерности входных подпространств в прямой и двойственной сети (которые далее будем называть модальными состояниями сети); $\hat{r} = \min\left(r, s^x, c^y\right)$ – действующий ранг нейронного модуля.

При системном анализе наибольший интерес представляют потенциально достижимые возможности нейронной сети. Понятно, что с увеличением операторных рангов модулей такие возможности увеличиваются. Поэтому при анализе можно полагать, что всегда выполняется равенство $r = \min(g, p)$. В дифференциальной геометрии такой оператор называется *регулярным* или *правильным*, этот же термин будем использовать для нейронного модуля. Для регулярного модуля выполняется условие $r \geq \min\left(s^x, c^y\right)$ и, следовательно, $\hat{r} = \min\left(s^x, \bar{c}^y\right)$, тогда из (12.3) нетрудно получить, что

$$\dim \hat{A}_r = s^x \bar{c}^y. \tag{12.4}$$

Изоморфизм между множеством операторов и тензорных произведений пространств позволяет в расчетах дуально перейти от операторного многообразия к одному оператору фиксированного ранга, отображающего входное пространство векторов в выходное векторное пространство. Условия общего положения предъявляются теперь к выбору оператора нейронного модуля.

Дуальный переход от многообразия операторов к векторным пространствам, ассоциированным с каждым модулем сети, естественным образом вводит сигнальные состояния на структурной модели нейронной сети. Размерности сигнальных подпространств в различных сечениях двойственной сети теперь будем рассматривать как модальные состояния структурного уровня (от латинского modus – размер). Операторные модули общего положения осуществляют преобразование входного модального состояния в выходное.

Для операции нахождения минимума в дальнейшем будем использовать обозначение «$\circ$», иначе называемое логическим умножением. Например, в этом обозначении условие *регулярности* модуля будет записываться в виде $r = g \circ p$.

12.3. Пластичность модульных нейронных сетей с биективными связями

Будем полагать, что многообразие операторов в модульной нейронной сети образуется только за счет вариации параметров модулей. Межмодульные связи при этом считаются фиксированными.

Влияние модулей. Выделим в сети некоторый модуль A ранга r и предположим, что его параметры варьируются, в то время как параметры всех остальных модулей фиксированы в общем положении. Поскольку модуль находится в составе сети, то размерности пространств состояний модуля в прямой и двойственной сетях в общем случае меньше размерностей модуля по входу и выходу. Действующее число степеней свободы модуля обозначим $S(A)$. Подобным образом выделяя и поодиночно варьируя параметры остальных модулей, получим, что вклад всех модулей в общее число степеней свободы сети определяется суммой

$$\sum_A S(A).$$

Влияние связей. Биективные межмодульные связи устанавливают точное и однозначное соответствие между модальными состояниями смежных модулей, т. е. каждая связь, рассматриваемая как двойственное отображение, является *регулярной,* и ее влияние можно оценивать выражением (12.4). Од-

нако в отличие от модулей связи фиксированы, и поэтому их присутствие в сети приводит к «связыванию» степеней свободы нейронных модулей. Из этого следует, что каждая фиксированная межмодульная связь (допустим, между модулями $A \Rightarrow B$) ранга r^{AB} уменьшает общее количество степеней свободы сети на величину

$$s^B \overline{c}^A = \left(c^A \circ r^{AB}\right)\left(\overline{s}^B \circ r^{AB}\right),$$

где s^B, c^A и $\overline{c}^A, \overline{s}^B$ – размерности пространств модальных состояний в прямой и двойственной сетях (вторая форма записи более удобна при анализе конуса связей). Рис. 12.2 иллюстрирует действие фиксированной межмодульной связи в сети двойственного распространения.

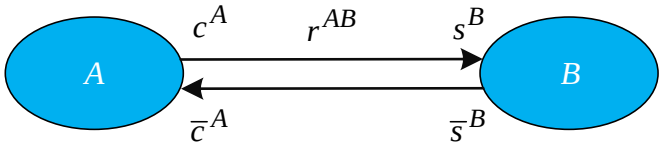


Рис. 12.2. Межмодульная связь в нейронной сети двойственного распространения

Таким образом, формула расчета числа степеней свободы для всей модульной сети будет иметь вид:

$$S(H) = \sum_A S(A) - \sum_{A \Rightarrow B} \sum \left(c^A \circ r^{AB}\right)\left(\overline{s}^B \circ r^{AB}\right). \qquad (12.5)$$

Если все модули *регулярны*:

$$S(H) = \sum_A s^A \overline{c}^A - \sum_{A \Rightarrow B} \sum \left(c^A \circ r^{AB}\right)\left(\overline{s}^B \circ r^{AB}\right). \qquad (12.6)$$

В последних выражениях двойная сумма распространяется на все существующие связи.

12.4. Расчет модальных состояний в слабосвязанных нейронных сетях

Выражения (12.5), (12.6) можно использовать для расчета степени пластичности только в том случае, если известны модальные состояния нейронной сети. Модальные состояния в прямой и двойственной сети порождаются наличием на их входах сигнальных пространств. Для оценки потенциальных свойств нейронной сети будем полагать, что эти пространства имеют максимально возможную размерность, определяемую размерностью терминальных полей сети:

$$s^{inp} = N, \quad \overline{c}^{out} = M .$$

Здесь N – размерность нейронной сети по входу, M – размерность нейронной сети по выходу. Напомним, что выход прямой сети является одновременно входом двойственной сети, а состояния s^{inp}, $\bar{c}^{out}$ – источниками, воздействующими на прямую и двойственную сеть. Схема формирования модальных состояний двойственной сети показана на рис. 12.3.

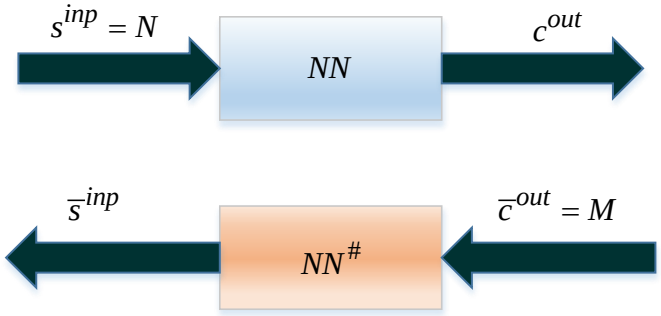


Рис. 12.3. Схема формирования модальных состояний в прямой и двойственной нейронной сети

Рассмотрим конус связей (рис. 12.4) в модульной сети прямого распространения. Модальные состояния для конуса связей в общем виде определяются выражениями:

$$c^B = s^B \circ r^B, \quad s^B = \bigcup_{A_i \in \Gamma^{-1}(B)} c^{A_i} \circ r^{A_i B},$$

где символ $\bigcup$ условно показывает, что размерность s^B следует определять для объединения подпространств, порождаемых модулями окрестности. Размерности этих подпространств известны и равны $c^{A_i} \circ r^{A_i B}$, однако размерность их объединения в общем случае вычислить не удается, поскольку пространства могут иметь пересечения, обусловленные наличием предшествующих связей между модулями.

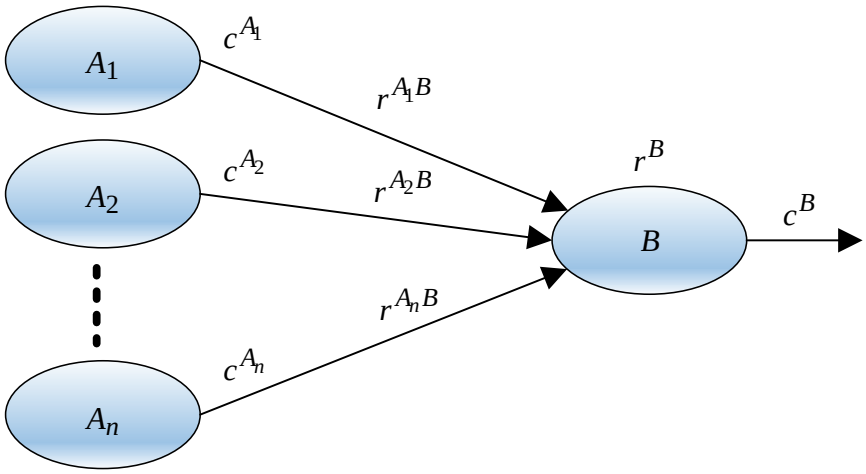


Рис. 12.4. Конус связей в сети прямого распространения

В случае слабосвязанных сетей задача существенно упрощается, поскольку по теореме «О параллельных путях» (см. 2.2) для любого модуля слабосвязанной сети существует единственный путь, связывающий его с терминальной вершиной. Этот путь выделяет в терминальном пространстве независимое подпространство. Размерность этого подпространства легко вычислить: оно будет равно логическому произведению рангов всех дуг пути от терминальной области до данного модуля. Поскольку все выделенные таким образом подпространства независимы, то объединение подпространств перед модулем B на самом деле будет прямой суммой, а это означает, что формула для вычисления размерности трансформируется к виду:

$$c^B = s^B \circ r^B, \qquad s^B = \sum_{A_i \in \Gamma^{-1}(B)} c^{A_i} \circ r^{A_i B} = p^B. \tag{12.7}$$

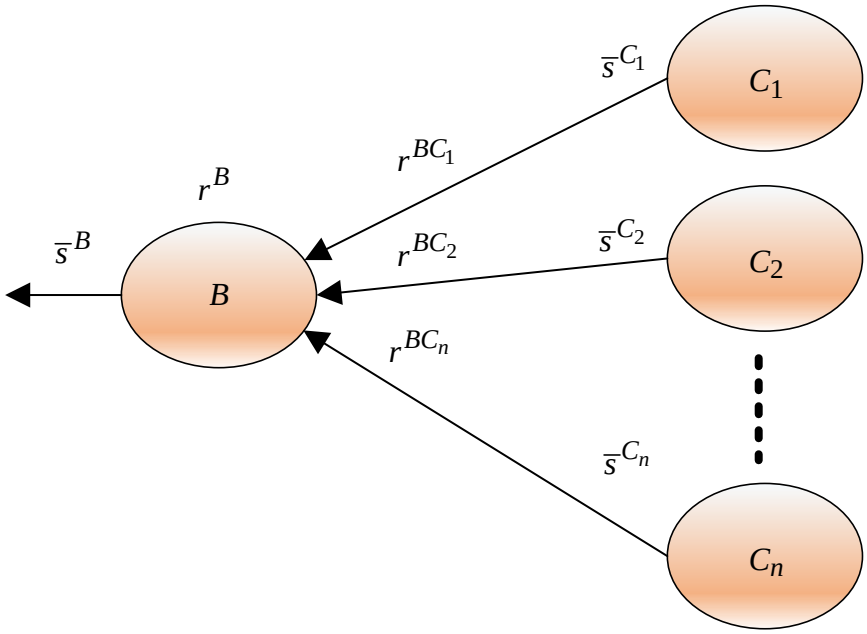


Рис. 12.5. Конус связей в сети обратного распространения

Поскольку модули *регулярные*, то $c^{A_i} \circ r^{A_i B} = r^{A_i B}$. Более того, в слабосвязанных сетях все связи инъективны (см. 3.1), поэтому сумма будет равняться входной размерности модуля, т. е. $s^B = p^B$. Условия слабой связанности двойственны, следуя принципу двойственности, для сети обратного распространения (рис. 12.5) можно записать аналогичные выражения для расчета модальностей:

$$\overline{s}^B = \overline{c}^B \circ r^B, \qquad \overline{c}^B = \sum_{C_i \in \Gamma(B)} \overline{s}^{C_i} \circ r^{C_i B} = g^B, \tag{12.8}$$

где g^B – размерность модуля по выходу.

12.5. Пластичность быстрых нейронных сетей

Быстрое перестраиваемое преобразование будем рассматривать как модульную нейронную сеть, где роль модулей выполняют базовые операции (нейронные ядра). Эта сеть является слабосвязанной. По построению быстрого преобразования ранги всех межмодульных связей равны единице, а все ядра в пределах слоя имеют совпадающие структурные характеристики. Обозначим через (p_m, g_m) размерности ядер слоя m, а через k_m – их число. Нетрудно заметить, что при единичных рангах каждая фиксированная связь удаляет одну степень свободы.

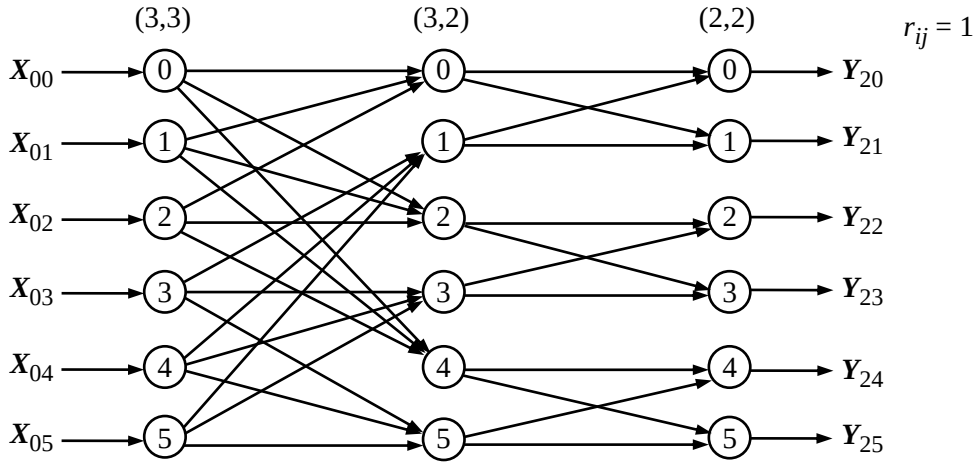


Рис. 12.6. Структурная модель трехслойной БНС

Учитывая это обстоятельство и используя (12.7), (12.8), из формулы (12.6) получим:

$$S(H) = \sum_{m=0}^{n-1} p_m g_m k_m - \sum_{m=0}^{n-2} D_m, \qquad (12.9)$$

где D_m – количество связей в межслойном переходе с номером m. Например, для сети, показанной на рис. 12.6, число степеней свободы $S(H) = (3 \cdot 3 \cdot 6 + 3 \cdot 2 \cdot 6 + 2 \cdot 2 \cdot 6) - (18 + 12) = 84.$

13. МЕТОДЫ ФОРМИРОВАНИЯ ЭТАЛОННЫХ ОБРАЗОВ

Быстрые нейронные сети обучаются к эталонам классов. В простейшем варианте эталоном может служить средний вектор класса (центроид класса). Число эталонов в этом случае равно числу классов. Конфигурация классов в признаковом пространстве заведомо неизвестна и может быть сколь угодно сложной, поэтому использование среднего вектора для обучения БНС может оказаться неэффективным для последующего решения задачи классификации образов. Типична ситуация, когда сеть будет идеально распознавать эталоны, но ошибаться на образах, удаленных от эталонов. В этом случае говорят, что обученная нейронная сеть обладает плохой обобщающей способностью. Для улучшения качества нейросетевой классификации целесообразно в классе выделить несколько эталонов и обучить к ним нейронную сеть. Результат классификации в этом случае будет формироваться по оценке близости образа ко всем эталонам класса, что позволяет существенно улучшить обобщающую способность классификатора. Сеть при этом увеличится по числу нейронов и связей. Такое расширение сети вполне оправдано высоким быстродействием БНС как в режиме обучения, так и в режиме классификации данных.

Для выбора эталонов целесообразно использовать существующие методы кластеризации данных, применяя их к классам обучающей выборки. Каждый класс разбивается на подклассы, а в подклассах находятся центроиды, которые в дальнейшем используются как эталоны классов. На одной и той же выборке различные методы кластеризации формируют отличающиеся наборы эталонов. Выбор того или иного метода определяется опытом исследователя, а в конечном счете результатом классификации. Для БНС выделение эталонов является необходимым этапом в задаче обучения сети, поэтому в данной главе будут рассмотрены несколько методов кластеризации данных, применимых для ее решения.

Обученная нейронная сеть предназначена для решения задачи распознавания образов. Распознавание заключается в отнесении входного образа нейронной сети к тому или иному классу по значению выхода нейронной сети. Если сеть обучается к эталонам классов, то алгоритм распознавания заключается в оценке близости образа к эталонам классов. Эту оценку выполняет нейронная сеть, формируя результат оценки на своих выходах. За пределами нейронной сети часто используют наращивание решающего правила для получения наиболее контрастного результата распознавания.

В векторном пространстве образов степень близости векторов оценивается функцией расстояния. На этой основе в данном разделе будут рассмотрены метрические методы кластеризации и классификации образов.

13.1. Расстояния, дискриминанты, метрики

Для параметрической кластеризации необходимо определить функцию меры сходства объектов класса, это может быть функция расстояния или какая-либо иная дискриминантная функция. При использовании меры расстояния объекты класса рассматриваются как точки многомерного метрического пространства, в котором для каждой пары точек x, y однозначно определена неотрицательная функция расстояния $l(x, y)$ или метрика, удовлетворяющая трем аксиомам:

1) $l(x, y) = 0$ тогда и только тогда, когда $x = y$;

2) $l(x, y) = l(y, x)$ – аксиома симметрии;

3) для любой точки z $l(x, y) \le l(x, z) + l(z, y)$ – аксиома треугольника.

Точкой метрического пространства размерности N является вектор $x = (x_1, x_2, \ldots, x_N)$ с набором признаков объекта. Ниже представлены несколько вариантов функции расстояния.

Наиболее распространенной функцией расстояния является *метрика Евклида,* представляющая сумму квадратов покоординатных разностей векторов:

$$l = \sqrt{\sum_{j=1}^{N} (x_j - y_j)^2}.$$

Обобщением метрики Евклида является *метрика Минковского*:

$$l = \sqrt[p]{\sum_{j=1}^{N} (x_j - y_j)^p},$$

где p – целое положительное число. При $p = 1$ имеем специальный случай – *метрика «городских кварталов»*:

$$l = \sum_{j=1}^{N} |x_j - y_j|.$$

Метрика Чебышёва определяется как максимум расхождения по координатам векторов:

$$l = \max_j |x_j - y_j|.$$

Косинусная метрика вычисляется как угловое расстояние между векторами:

$$l = 1 - \cos(\alpha),$$

где $\cos(\alpha) = \dfrac{(x, y)}{\sqrt{(x, x)(y, y)}}$, $(x, y) = \sum_j x_j y_j$ – скалярное произведение векторов, которое иначе можно записать в векторной нотации $(x, y) = xy'$ как произведение вектор-строки на вектор-столбец. В случае, если векторы образов нормированы по длине к единице, т. е. $(x, x) = 1$ и $(y, y) = 1$, то $\cos(\alpha) = (x, y)$. Функция косинуса является мерой близости нормированных векторов, она не удовлетворяет аксиомам расстояния, но ее также можно использовать в задаче кластеризации как дискриминантную функцию оценки сходства.

Корреляционное расстояние, для вычислений используется функция корреляции векторов:

$$l = 1 - corr(x, y),$$

где $corr(x, y) = \dfrac{(x - m_x)(y - m_y)'}{\sqrt{(x - m_x)(x - m_x)'}\sqrt{(y - m_y)(y - m_y)'}}$ – функция корреляции векторов, $m_x = \dfrac{1}{N}\sum_{j=1}^{N} x_j$ и $m_y = \dfrac{1}{N}\sum_{j=1}^{N} y_j$ – средние значения векторов. Если векторы имеют нулевое среднее, тогда корреляционное расстояние совпадает с косинусной метрикой. В задаче кластеризации функцию корреляции можно использовать самостоятельно как дискриминантную функцию.

Расстояние Хэмминга вычисляется как количество различий по координатам векторов:

$$l = \left(\#\left(x_j \neq y_j\right)\right) / N .$$

Эта функция обычно используется при сравнении кодовых слов, записанных в виде векторов. На основе мер расстояния разработаны метрические методы классификации образов.

13.2. Метрическая классификация образов

Методы метрической классификации являются алгоритмической альтернативой нейросетевой классификации и часто используются для сравнительной оценки эффективности нейронной сети в задаче распознавания образов.

В общем случае алгоритм метрической классификации можно описать следующим образом. Пусть $D_1, D_2, \ldots, D_n$ – различные классы, каждый из которых включает в себя n_i состояний (точек), подлежащих классификации, и L_B есть мера межклассового расстояния. Состояние x относят к классу D_α, если мера расстояния между вектором x и классом минимальна:

$$x \in D_\alpha, \text{ если } L_B(x, D_\alpha) = \min_i L_B(x, D_i).$$

Степень надежности классификации оценивается по соотношению между межклассовым и внутриклассовым расстоянием L_W. Все методы метрической классификации различаются в основном видом меры межклассового расстояния L_B. Наиболее распространенными методами являются следующие:

1. Метод классификации по минимальному расстоянию до множества (принцип «ближнего соседа»). В качестве меры межклассового расстояния принимают минимальное расстояние среди всех расстояний от точки x до всех точек a_i^s, входящих в класс D_i:

$$L_B(x, D_i) = \min_s L_B\left(x, a_i^s\right).$$

2. Метод классификации по максимальному расстоянию между множествами (принцип «дальнего соседа»). В качестве межклассовой меры принимают максимальное расстояние от точки x до всех точек a_i^s, входящих в класс D_i:

$$L_B(x, D_i) = \max_s L_B\left(x, a_i^s\right).$$

3. Метод среднего расстояния до множества (принцип «средней связи»). Мерой межклассового расстояния является среднее расстояние от точки x до всех точек a_i^s, входящих в класс D_i:

$$L_B(x, D_i) = \frac{1}{n_i} \sum_{s=1}^{n_i} L_B\left(x, a_i^s\right).$$

4. Метод классификации по расстоянию до эталона. Для каждого класса D_i определяется эталонный вектор:

$$a_i^* = \frac{1}{n_i} \sum_{s=1}^{n_i} a_i^s .$$

В качестве меры межклассового расстояния принимают величину

$$L_B\left(x, D_i\right) = L\left(x, a_i^*\right).$$

Выбор той или иной межклассовой меры определяется пространственной конфигурацией класса. Априорно конфигурации классов, как правило, неизвестны, и чаще всего подходящую меру подбирают экспериментально. Среди мер внутриклассового расстояния наиболее часто применяется мера среднего расстояния:

$$L_W\left(D_i\right) = \frac{1}{n_i}\sum_{s=1}^{n_i} L\left(a_i^*, a_i^s\right)$$

и мера максимального радиуса:

$$L_W\left(D_i\right) = \max_i L\left(a_i^*, a_i^s\right).$$

При использовании евклидовой метрики в качестве мер L_B и L_W обычно используют квадраты расстояний.

13.3. Методы кластеризации

Для набора данных с *n* точками в *N*-мерном пространстве и количеством желаемых кластеров *k* цель кластеризации состоит в том, чтобы разделить набор данных на *k* групп, называемых кластерами: $\{C_1, C_2, \cdots C_k\}$.

Метод k-средних (англ. k-means) – наиболее популярный метод кластеризации. Был изобретен в 1950-х гг. математиком Гуго Штейнгаузом [46] и почти одновременно Стюартом Ллойдом [47]. Алгоритм итерационным способом решает оптимизационную задачу – минимизации критерия:

$$V = \sum_{i=1}^{k}\sum_{x\in C_i}\left\|x - \mu_i\right\|^2,$$

где $x = \left(x_1, x_2, \ldots, x_N\right)$ – вектор координат точек, принадлежащих кластеру; $\mu_i = \left(\mu_{i1}, \mu_{i2}, \cdots, \mu_{iN}\right)$ – центроид кластера; скобки $\|\cdot\|$ обозначают меру расстояния между точками. Алгоритм стремится минимизировать суммарное квадратичное отклонение точек кластеров от центров этих кластеров. Алгоритм завершается, когда на какой-то итерации не происходит изменения внутри кластерного расстояния. Это происходит за конечное число итераций, так как количество возможных разбиений конечного множества конечно, а на каждом шаге суммарное квадратичное отклонение V уменьшается, поэтому зацикливание невозможно. Существует несколько модификаций этого алгоритма, перечислим некоторые из них.

Алгоритм Г. Бола и *Д. Холла* (1965). Случайно выбираются k объектов (эталонов); каждый объект присоединяется к ближайшему эталону (тем самым образуется k классов). В качестве новых эталонов принимаются центры масс классов (центроиды). После пересчета эталонов объекты снова распределяются по ближайшим эталонам и т. д.

*Алгоритм Дж. Мак-Кин*а (1967) отличается от метода Г. Бола и Д. Холла тем, что при просмотре списка объектов пересчет центра масс класса происходит после присоединения к нему каждого очередного элемента.

Алгоритм «ФОРЕЛЬ» (первоначальное название ФОРЭЛ – ФОРмальный ЭЛемент, предложен В. Н. Елкиной и Н. Г. Загоруйко в 1966 г.). Случайный объект объявляется центром класса, все объекты, находящиеся от него на расстоянии не большем R, входят в первый класс. В нем определяется центр масс, который объявляется новым центром класса, и так далее до стабилизации центра. Затем все объекты, попавшие в первый класс, изымаются, и процедура повторяется с новым случайным центром.

Метод k-средних гарантирует достижение не глобального минимума суммарного квадратичного отклонения V, а только одного из локальных минимумов. Результат зависит от выбора исходных центров кластеров, их оптимальный выбор неизвестен. Число кластеров надо знать заранее.

Использование метода для поиска обучающих эталонов нейронной сети удобно тем, что для каждого класса задается одно и то же число эталонов, что упрощает топологию классифицирующей нейронной сети.

На рис. 13.1 в верхнем ряду показаны примеры тестовых классов трех типов в двумерной плоскости, а ниже представлены результаты кластеризации различными методами. Результаты работы метода k-средних при использовании евклидова расстояния показаны во втором ряду. Метод хорошо работает для конфигурации классов типа «сгущения» и при пересекающихся классах, но ошибается на классах типа «ленты». Метод не может самостоятельно определить количество классов, и требуется задание этого значения. Для результатов, представленных на рисунке, количество классов было выбрано равным 2.

Иерархическая кластеризация группирует данные в различных масштабах путем создания кластерного дерева или дендрограммы [48]. Дерево – это многоуровневая иерархия, где кластеры на одном уровне объединяются в кластеры на следующем уровне. Это позволяет выбрать уровень или масштаб кластеризации, наиболее подходящий для конкретной конфигурации классов.

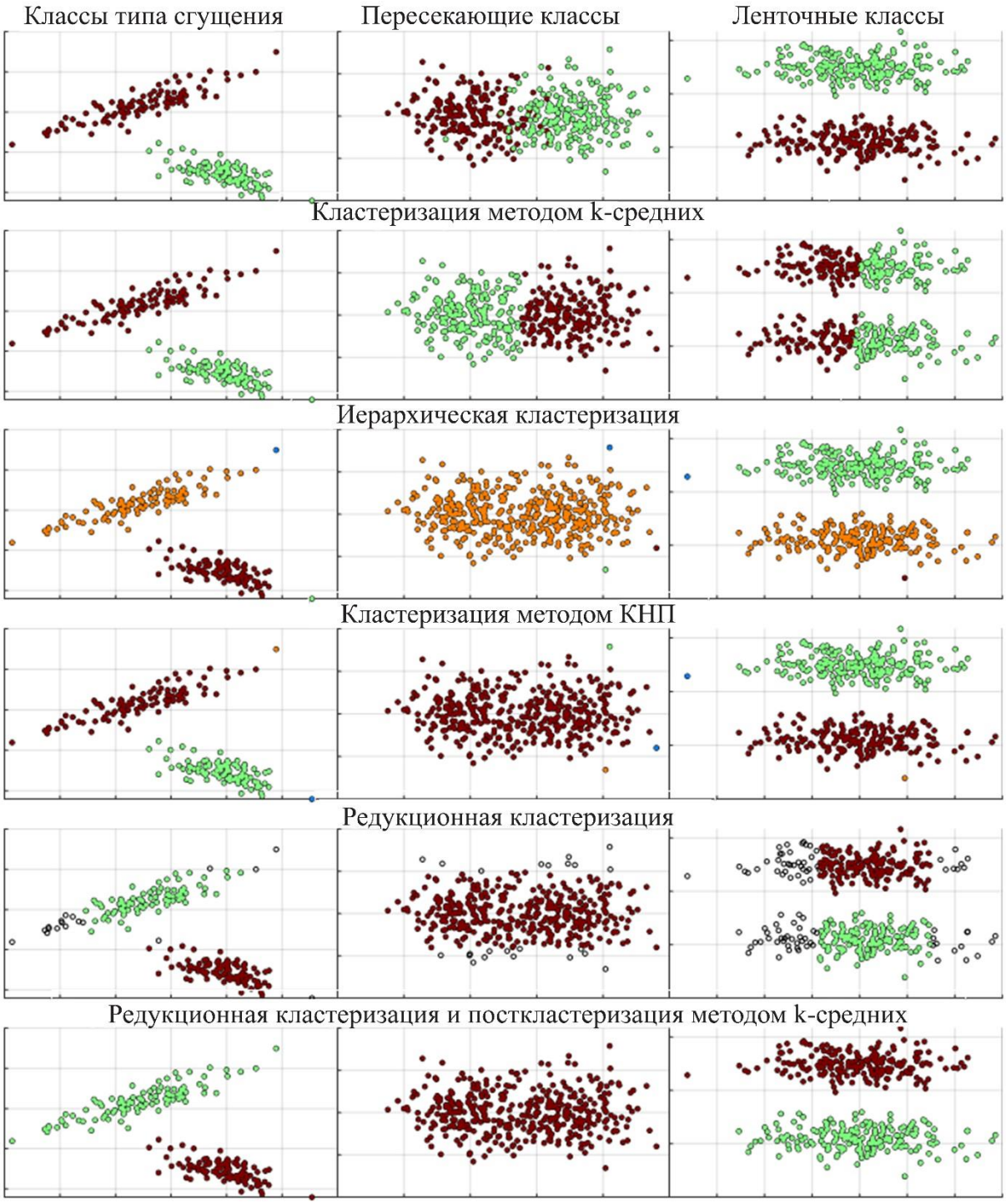


Рис. 13.1. Методы кластеризации

Описание алгоритма:

1. Определяется сходство или различие между каждой парой объектов в наборе данных. На этом шаге вычисляется расстояние между объектами одним из возможных способов вычисления этого расстояния.

2. Объекты группируются в двоичное иерархическое кластерное дерево. На этом шаге близкие объекты связываются в пары на основе информации о парных расстояниях, полученной на шаге 1.

3. Поскольку объекты объединяются в двоичные кластеры, вновь сформированные кластеры группируются в более крупные кластеры до тех пор, пока не будет сформировано иерархическое дерево.

4. Иерархическое дерево разрезается на некотором пороговом уровне. Все объекты под каждым срезом одной ветви назначаются одному кластеру. Место разреза дерева определяется конфигурацией классов.

Дендрограмма – это образ кластерного дерева в координатных осях, где по горизонтали откладываются номера объектов, а по вертикали – значения мер отдаленности, при которых происходили объединения кластеров. На рис. 13.2 приведен пример построения дендрограммы. При оценке расстояния между кластерами используются меры межмножественного расстояния, изложенные в 13.2.

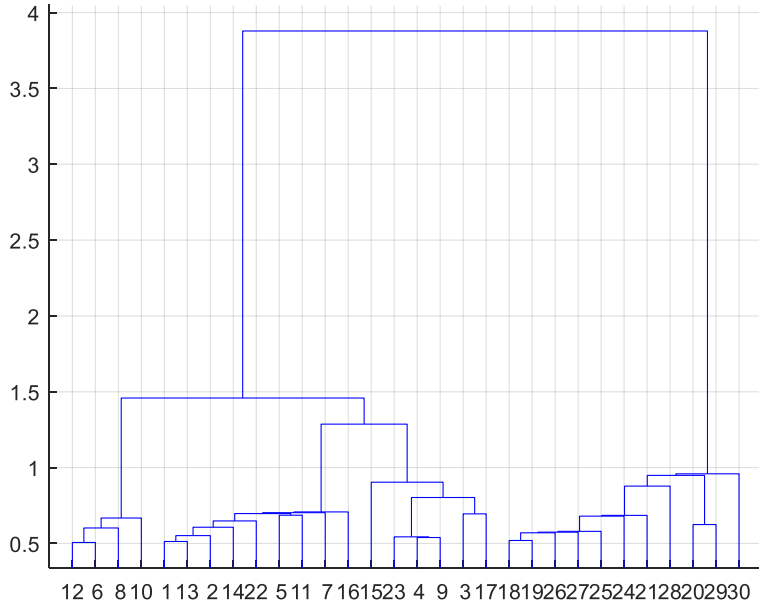


Рис. 13.2. Дендрограмма для конфигурации классов типа «сгущение»

Результаты иерархической кластеризации для тестовых конфигураций классов приведены в третьем ряду рис. 13.1. В эксперименте для измерения расстояния между точками использовалась метрика Евклида, а для измерения межмножественного расстояния − метрика «ближнего соседа». Метод требует задания порогового уровня разреза дерева или числа классов. При малом объеме данных уровень можно выбрать непосредственно по дендрограмме. При больших объемах используются эмпирические методы, которые не всегда приводят к хорошему результату. Для результатов, представленных на рисунке, количество классов было выбрано 3. Задание значения 2 приводит к неудовлетворительным результатам для всех типов классов.

Метод кратчайшего незамкнутого пути (КНП), его также называют минимальным покрывающим деревом или каркасом [49]. Алгоритм реализуется в следующей последовательности: соединяются ребром две ближайшие точки, затем среди оставшихся отыскивается точка, ближайшая к любой из уже соединенных точек, и присоединяется к ним, и так далее до исчерпания всех точек. В построенном дереве затем отбрасываются k–1 самых длинных дуг и получают k классов (рис. 13.3). Утверждается, что метод позволяет выделять классы произвольной формы.

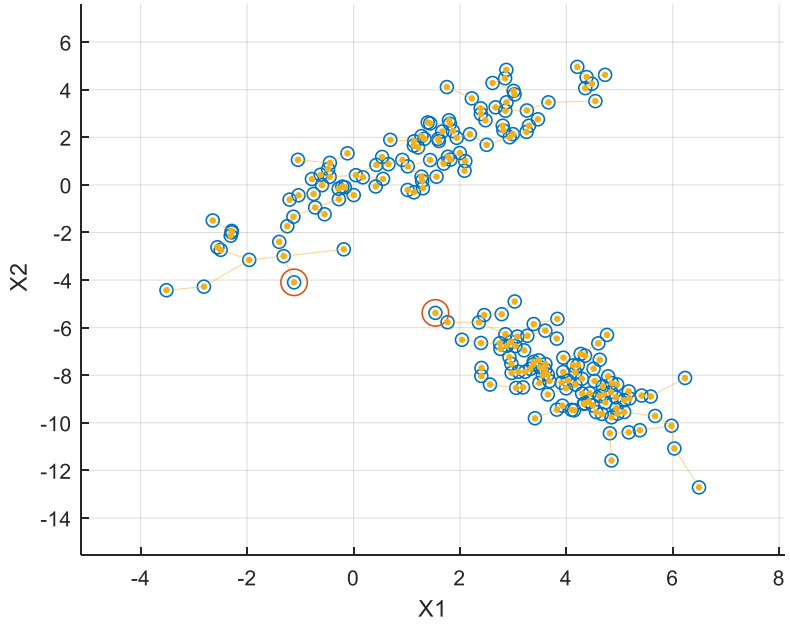


Рис. 13.3. Кластеризация методом КНП на два класса.
Крупными маркерами выделены вершины рассекаемой дуги

Результаты кластеризации методом КНП для различных конфигураций классов приведены в четвертом ряду рис. 13.1. Для измерения расстояния между точками использовалась метрика Евклида. Метод требует задания числа образующихся классов. Для результатов, представленных на рисунке, количество классов было выбрано 3. Задание значения 2 приводит к неудовлетворительным результатам для всех типов классов, это связано с чувствительностью метода к удаленным точкам классов, которые трактуются как отдельные кластеры.

Редукционная кластеризация. Основной задачей кластеризации является локализация центров кластеров в многомерном пространстве. При выполнении гипотезы компактности кластеров можно ожидать, что форма функции плотности точек-образов вдоль каждой координаты будет в той или иной мере отражать кластерную структуру данных. Координатные функции плотности (назовем их *координатными профилями*) можно оценить, построив гистограммы по каждой координате многомерной выборки.

Число мод на координатном профиле (рис. 13.4), очевидно, не может превысить числа кластеров. Точки с координатами, принадлежащими малым окрестностям какой-либо моды, потенциально являются кандидатами точек, принадлежащих кластерам класса. При построении профильных гистограмм определим также принадлежность точек класса бинам каждого координатного профиля и назовем эти данные *профильными векторами принадлежности точек*.

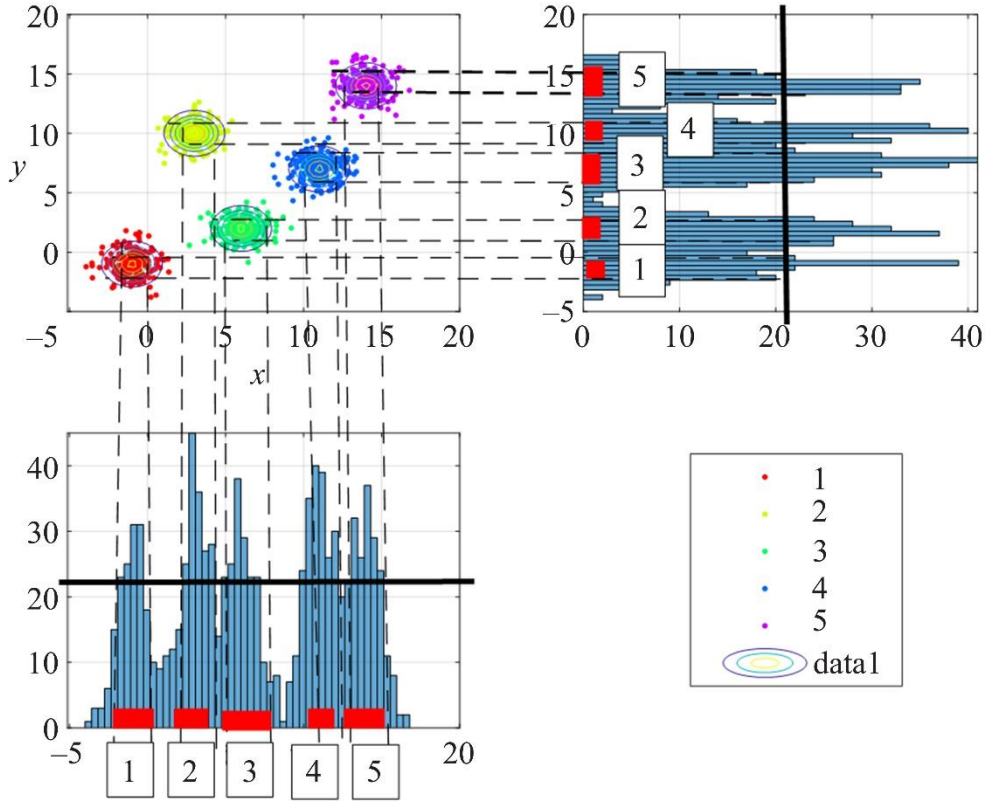


Рис. 13.4. Метод редукционной кластеризации

На координатном профиле нанесем линию порогового уровня и выделим области, в которых плотность точек превышает пороговый уровень. На рис. 13.4 эти области выделены красным цветом. Число построенных зон является оценкой числа кластеров класса. Эта величина, очевидно, зависит от позиции линии порогового уровня. Для каждого профиля определяется свое значение порогового уровня исходя из числа образованных зон и объема покрытия зонами исходных данных. Присвоим кластерным зонам индивидуальные локальные метки (на рисунке это цифры от 1 до 5).

Используя построенные ранее профильные векторы принадлежности, для каждого координатного профиля пометим все точки признакового пространства класса метками кластерных зон. В отдельных профилях некоторые точки могут остаться не помеченными, если они не принадлежат ни одной кластерной зоне. Напротив, если точка-образ проявляется в кластерных зонах всех координатных

профилей, то это означает, что она принадлежит некоторому кластеру в признаковом подпространстве. Группируя все точки с выбранным набором меток, получим состав данного кластера. Точки, видимые во всех профилях с другим набором меток, формируют наполнение следующего кластера и т. д.

Объем точек-образов, принадлежащих кластерным зонам и видимых во всех профилях, меньше, чем исходный набор точек, но при этой редукции объема данных кластерная структура исходного множества сохраняется. Поэтому при больших объемах данных процедура выделения кластерных зон может быть многократно повторена. Этим свойством метода обусловлен выбор его названия.

Следует заметить, что алгоритм редукционной кластеризации не использует какую-либо меру близости в признаковом пространстве, поэтому он применим к выборкам данных, представленных в любой шкале – от интервальной до номинальной. Метод позволяет определить фактическое количество кластеров в наборе данных и построить их центроиды. Основные трудности использования метода связаны с определением числа бинов в представлении координатных профилей и пороговых уровней отсечки кластерных зон. Некоторые рекомендации по выбору системных метапараметров даны в работе [50]. Результаты работы метода редукционной кластеризации для различных конфигураций классов – в пятом ряду рис. 13.1. Количество бинов гистограммы определялось эвристическим правилом bins=sqrt(N), где N – число наблюдений. Это правило часто используется в математических пакетах. Минимальный пороговый уровень отсечки кластерных зон был равен 0.75 от числа наблюдений.

Метод достаточно хорошо находит центры кластеров, однако часть точек оказываются неидентифицированными, а удаленные точки могут интерпретироваться как отдельные кластеры. Для полной кластеризации точек алгоритм следует дополнить методом постkластеризации.

Постkластеризация. Назначение метода постkластеризации состоит в том, чтобы наполнить кластеры точками класса, которые не попали в построенную кластерную структуру. Наиболее просто это сделать, если ввести функцию расстояния между точкой класса и кластером:

$$L\left(a, C_k\right) = \min_i \left(d\left(a, c_{ki}\right)\right),$$

где $C_k = \left\{c_{ki}\right\}$ – множество точек, определяющих кластер; a – внешняя точка по отношению к кластерной структуре; $d\left(a, c_{ki}\right)$ – некоторая метрика расстояния между точками. В алгоритме постkластеризации просматриваются все внешние точки, и относят их к ближайшему кластеру.

Следует заметить, что многократное повторение процедуры постластеризации подобно методу k-средних с заданными стартовыми центрами кластеров. Применение комбинации методов редукционной кластеризации и k-средних представлены в последнем ряду рис. 13.1. В методе k-средних для измерения расстояния между точками использовалась евклидова метрика.

13.4. Валидация методов кластеризации для обучения БНС

Применение методов кластеризации к данным обучающей выборки позволяет построить эталоны классов, которые БНС использует для своего обучения. Применимость метода кластеризации в конечном счете оценивается способностью обученной нейронной сети решать задачу классификации данных. Валидацию методов можно провести на полной обучающей выборке по числу ошибок классификации, допущенных обученной нейронной сетью с выбранной топологией.

Рассмотрим оценку применимости методов на примере набора данных «Ирисы Фишера» [51]. Этот набор данных стал классическим и часто используется в литературе для иллюстрации работы различных статистических алгоритмов. В наборе данных представлены три вида растений: setosa, versicolor, virginica (табл. 13.1), всего 150 примеров, по 50 примеров для каждого вида.

Для каждого экземпляра измерялись четыре характеристики (в сантиметрах):

– X1 длина чашелистика (англ. sepal length);

– X2 ширина чашелистика (англ. sepal width);

– X3 длина лепестка (англ. petal length);

– X4 ширина лепестка (англ. petal width).

На основании этого набора данных требуется построить классифицирующую нейронную сеть, определяющую вид растения по данным измерений.

Таблица 13.1

Ирисы Фишера

setosa	versicolor	virginica
		

На рис. 13.5 представлены двумерные проекции набора данных в пространстве признаков. Нумерация признаков соответствует приведенному списку характеристик.

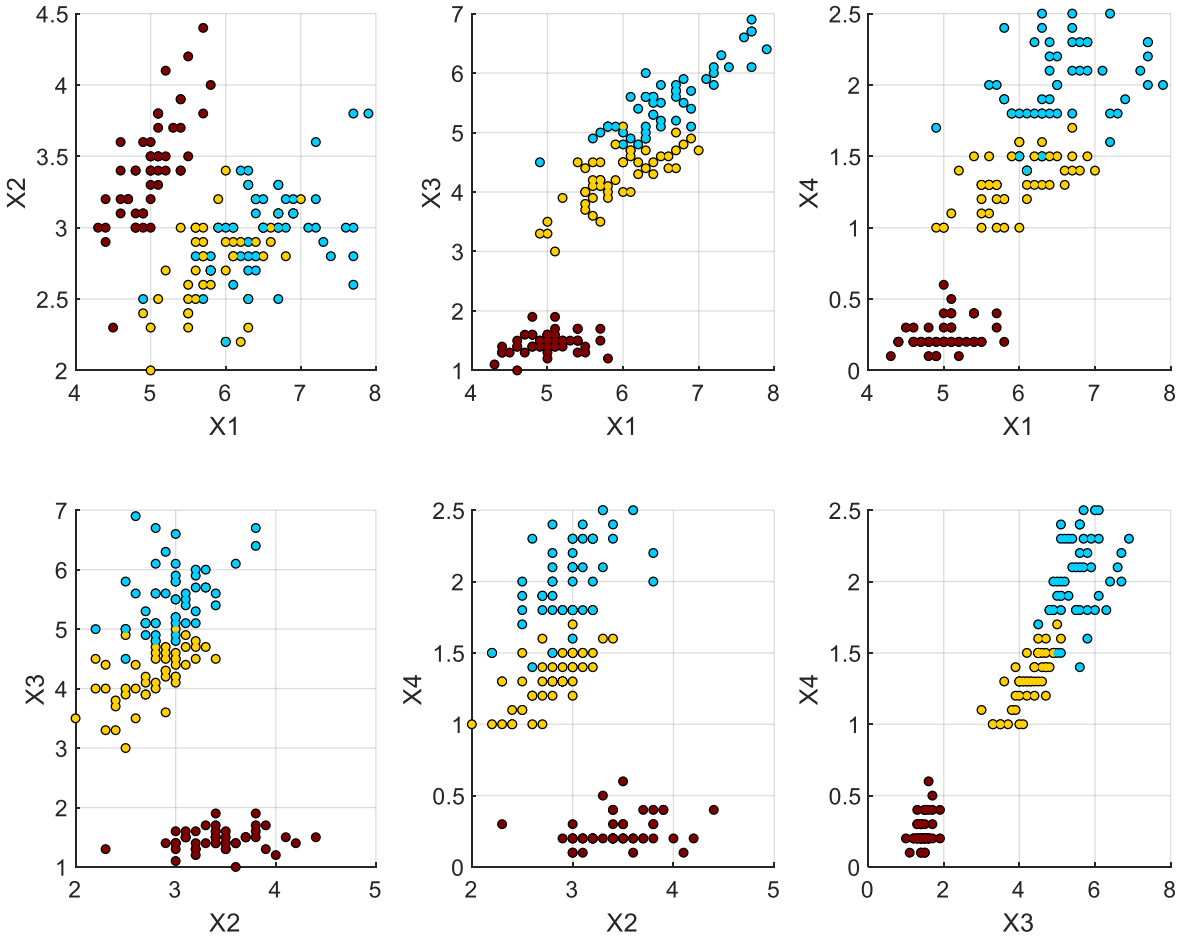


Рис. 13.5. Двумерные проекции набора данных «Ирисы Фишера»

Для построения классификатора выбрана пирамидальная нейронная сеть, одномерная по входу и двумерная по выходу. Размерность входа по числу признаков равна четырем. Число классов в наборе данных равно трем, каждый класс представлен двумя эталонами, поэтому размерность выхода сети равна 3×2. На рис. 13.6 показана топология использованной сети. Сеть состоит из двух слоев. Нейронные ядра (на рисунке выделены синим цветом) первого слоя имеют одномерный вход размерностью 2 и двумерный выход размерностью 3×2. Нейронные ядра второго слоя имеют одномерный вход размерностью 2 и выход размерностью 1. Для нумерации нейронов используются две цифры, разделенные запятой; первая цифра определяет нумерацию вдоль пространственной координаты X, а вторая – вдоль пространственной координаты Y. Этот же принцип используется для нумерации рецепторных и аксоновых полей. Нумерация всегда начинается с нулевого индекса.

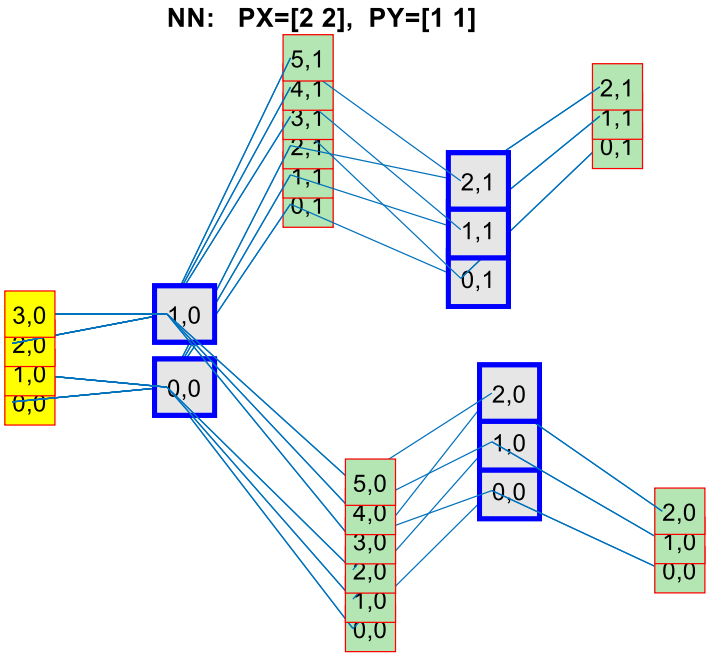


Рис. 13.6. Топология валидационной БНС

В табл. 13.2 представлены результаты валидации кластеризующих методов для обучения БНС с данной топологией. Цифры в ячейках таблицы обозначают число ошибок, допущенных нейронной сетью при классификации полного набора данных.

Таблица 13.2

Результаты валидации методов кластеризации на полном наборе данных «Ирисы Фишера»

Метод кластеризации	Классы		
	setosa	versicolor	virginica
k-средних	0	4	1
Иерархический	0	3	0
Кратчайший незамкнутый путь (КНП)		4	1
Редукционный	0	3	0

В методах кластеризации в качестве расстояния между точками использовалась метрика Евклида. Для оценки межкластерного расстояния в иерархическом методе применялся принцип «дальнего соседа», для которого были получены наилучшие результаты. Нейронная сеть обучалась к эталонам, нормированным по энергии к единичному уровню. Такое же нормирование выполнялось для входных образов нейронной сети, поэтому выходные значения сети принадлежат диапазону [–1,+1]. С каждым выходом сети однозначно связан эталон класса. Результат классификации устанавливался по выходу, имеющему максимальное значение.

По результатам валидации можно сделать вывод, что все методы кластеризации могут быть использованы для выделения обучающих эталонов БНС данной топологии.

14. МЕТОДЫ УСИЛЕНИЯ ОБОБЩАЮЩЕЙ СПОСОБНОСТИ БНС

Быстрая нейронная сеть гарантированно обучается к эталонам классов, однако образы, принадлежащие классам, но удаленные от эталонов могут классифицироваться неверно. В этом случае говорят, что нейронная сеть имеет недостаточную обобщающую способность. В идеале сеть должна распознавать образы, с которыми она раньше не сталкивалась, на основе тех знаний, которые были получены ею в процессе обучения. Для моделирования данной ситуации обучающую выборку разделяют на две части: собственно обучающую и тестовую. На обучающей части формируются эталоны классов, по которым БНС настраивается, а результаты настройки проверяются на тестовой части. Соотношение объемов обучающей и тестовой частей является предметом договоренности, типичное процентное соотношение составляет 50 на 50. При малых объемах данных объем обучающего множества обычно больше тестового. В табл. 14.1 приведены результаты валидации методов кластеризации при объеме обучающего множества для каждого класса набора «Ирисы Фишера» равном 30 и объеме тестового множества равном 20 наблюдениям.

Таблица 14.1

Результаты валидации методов кластеризации при выделении тестового множества

Метод кластеризации	Классы		
	setosa	versicolor	virginica
k-средних	0; 0	2; 2	1; 0
Иерархический	0; 0	2; 1	4; 3
Кратчайший незамкнутый путь (КНП)	0; 0	2; 2	8; 7
Редукционный	0; 0	3; 2	0; 0

В каждой клетке таблицы размещены две цифры, разделенные точкой с запятой, первая цифра обозначает число ошибок классификации, допущенных нейронной сетью на обучающем множестве, вторая − число ошибок сети на тестовом множестве. Условия проведения эксперимента такие же, как в предыдущей главе, но объем обучающей выборки уменьшился за счет выделения тестовой выборки. Поэтому, как и ожидалось, сумма числа ошибок на обучающей и тестовой выборках возросло по сравнению с результатами, представленными в гл. 13.

В классических нейронных сетях для усиления обобщающей способности широко используются нелинейные функции активации нейронов. Для БНС этот путь непригоден, поскольку метод быстрого обучения принципи-

ально ориентирован на линейный вариант сети. Очевидно, что обобщающую способность БНС можно улучшить увеличением числа обучающих эталонов в каждом классе, при этом соответственно будет изменяться и топология сети. В данной главе будет рассмотрен алгоритм автоматического наращивания числа эталонов по ошибкам на обучающей выборке. Алгоритм реализуется вне нейронной сети на линейном классификаторе с выбором результата классификации по максимальному значению на выходах. Алгоритм позволяет для каждого класса определить число эталонов, требуемое для минимизации ошибок обучения и тестирования. По полученному набору эталонов строится топология нейронной сети и выполняется ее настройка.

Перспективным методом усиления обобщающей способности является метод проекций на подпространства классов. В этом случае наборы векторов классов аппроксимируются оптимальными подпространствами с усеченным набором базовых векторов. На этой основе далее развивается новый метод обучения, реализующий принцип повторного входа Эдельмана. Предложены эволюционные классификаторы в виде наращиваемых комитетов классификаторов с непересекающимися областями компетенции. Представлен принцип непрерывного обучения классификаторов в процессе эволюции. Эффективность предложенных решений будет продемонстрирована на примерах.

14.1. Алгоритм автоматического наращивания числа эталонов

1. Алгоритм стартует с ситуации, когда в каждом классе определен только один эталон. Эти эталоны получаются усреднением примеров по каждому классу и нормированию результата к единичной энергии. Эталоны образуют начальный классификатор.

2. Классификатору предъявляется обучающая выборка. Примеры выборки предварительно также подвергаются нормированию по энергии. По результатам классификации фиксируются ошибки по каждому классу и сохраняются номера примеров обучающей выборки, на которых произошли ошибки.

3. На подмножествах ошибочных примеров каждого класса строятся новые одиночные эталоны, дополняющие существующие в каждом классе.

4. На обновленном составе эталонов каждого класса выполняется процедура посткластеризации методом k-means, который стартует с текущего набора эталонов класса. По результатам работы метода k-means на каждом классе эталоны классификатора уточняются. По обновленным эталонам формируется новый классификатор.

5. Процедура классификации обучающей выборки повторяется на обновленном классификаторе, фиксируются новые ошибки и примеры, на которых они происходят.

6. Алгоритм наращивания эталонов многократно повторяется и может продолжаться вплоть до нулевой ошибки на обучающей выборке.

Вместе с классификацией обучающей выборки выполняется классификация тестовой выборки, для которой фиксируется только число ошибок. Процедура наращивания эталонов может завершиться досрочно, если на заданном числе шагов не происходит уменьшения числа ошибок на тестовой выборке. В этом случае полагают, что классификатор построен и эталоны классов полностью определены.

Нормирование эталонов и примеров выборок по энергии не является обязательным, но упрощает оценку результатов классификации, сводя ее к оценке скалярного произведения нормированных векторов. Для минимизации числа эталонов важное значение имеет соотношение между обучающей и тестовой выборкой:

– при малом объеме обучающей выборки по сравнению с тестовой алгоритм быстро сходится к нулевой ошибке на обучающей выборке, при этом ошибка на тестовой выборке может остаться ненулевой (поскольку не хватает разнообразия эталонов для правильного распознавания примеров теста). Число порожденных эталонов при этом минимально;

– при увеличении объема обучающей выборки количество порождаемых эталонов возрастает. Это может приводить к тому, что на тестовой выборке может появиться нулевое количество ошибок раньше, чем это произойдет на обучающей выборке;

– соотношение между обучающей и тестовой выборками будет оптимальным, если нулевое количество ошибок на тестовой выборке появляется примерно на том же шаге, что и нулевое значение ошибок на обучающей выборке.

14.2. Нейросетевой классификатор для набора данных «Ирисы Фишера»

В табл. 14.2 приведены характеристики работы алгоритма наращивания числа эталонов в зависимости от доли обучающей выборки для набора данных «Ирисы Фишера».

Оптимальное значение доли обучающей выборки принадлежит диапазону 0.7…0.75. Суммарное число эталонов при этом равно 16 и 15.

Таблица 14.2

Зависимость числа эталонов классификатора «Ирисы Фишера»

Доля обучающей выборки	Стартовое число ошибок на обучающей и тестовой выборках	Число ошибок на обучающей и тестовой выборках после наращивания эталонов	Суммарное число эталонов	Распределение эталонов по классам
0.50	2;2	0;3	9	1;4;4
0.55	2;2	0;2	8	1;4;3
0.60	2;2	0;3	8	1;5;4
0.65	2;2	0;1	9	1;4;4
0.70	4;0	0;0	16	1;7;8
0.75	4;0	0;0	15	1;6;8
0.80	4;0	0;0	15	1;6;8

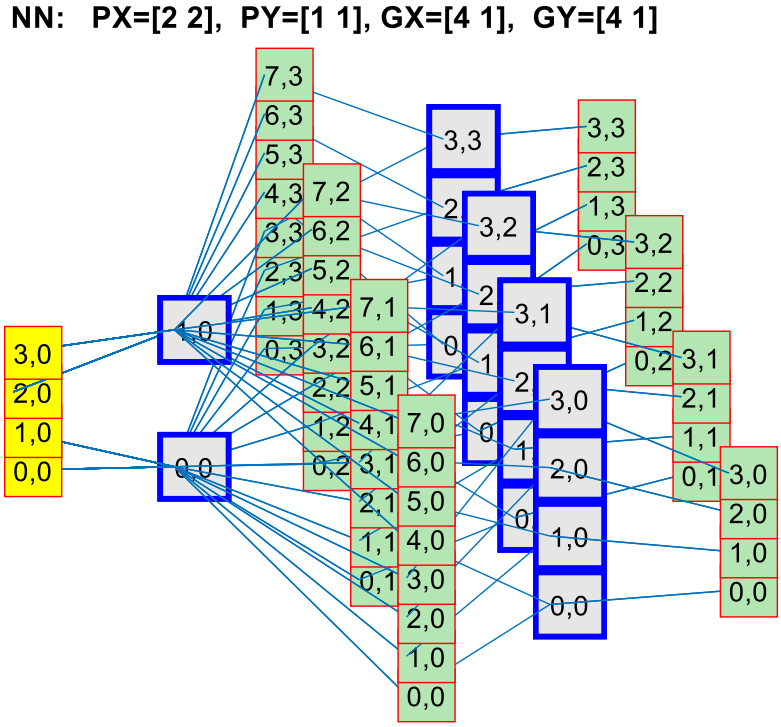


Рис. 14.1. Топологическая модель пирамидальной нейронной сети для классификатора «Ирисы Фишера»

На рис. 14.1 показана топология пирамидальной сети с выходами на 16 эталонов. Распределение выходов сети задается начальным упорядочиванием эталонов.

14.2. Нейросетевой классификатор для набора данных MNIST

Набор данных MNIST [52] содержит двумерные образы рукописных цифр от 0 до 9 в виде пиксельных изображений размером 28×28. Объем обучающей выборки равен 60 000 образов и 10 000 изображений содержит тестовая выборка. На рис. 14.2 показана выборка с десятью представителями для каждого класса.

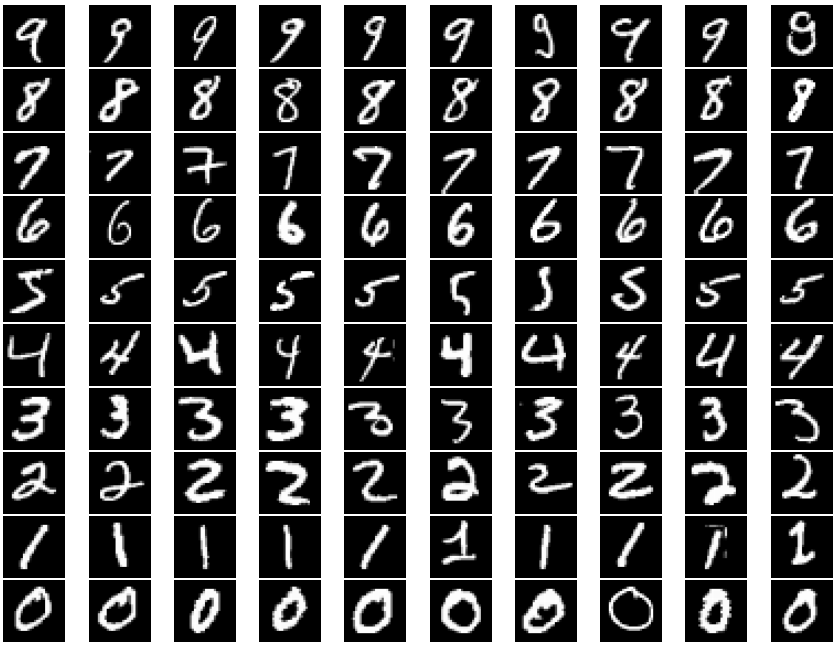


Рис. 14.2. Выборка из набора рукописных цифр

На примере набора данных MNIST рассмотрим различные варианты построения классификатора.

14.3.1. Классификация по минимуму углового расстояния

Обучение сети выполняется к нормированным эталонам классов, входные образы также нормируются к единичной энергии. Нейронная сеть вычисляет скалярные произведения между образом и всеми эталонами. Образ относится к классу, для которого скалярное произведение образа с эталоном класса максимально. Ошибка классификации зависит от числа эталонов. Эталоны классов находятся иерархическим методом кластеризации. В эксперименте использовались обучающая и тестовая выборки объемом 5000 примеров. В табл. 14.3 приведено значение точности классификатора в зависимости от числа эталонов. Точность выражается в процентах и связана со значением ошибки соотношением

$$\text{Accuracy} = 100 - \text{error}\%.$$

Таблица 14.3

Зависимость точности классификатора по угловому расстоянию от числа эталонов

Число эталонов в классе	1	10	20	30	40	50
Точность	71.8	80.1	84.0	84.8	84.5	85.8

Дальнейшее увеличение числа эталонов не дает ощутимого эффекта по точности распознавания. На рис. 14.3 показана матрица ошибок для значения точности 85.8.

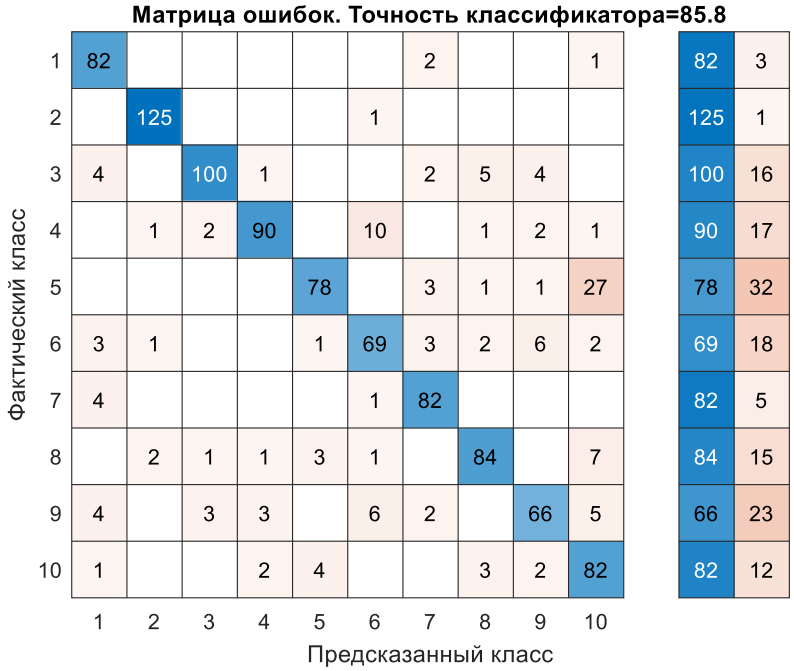


Рис. 14.3. Таблица ошибок при классификации
по угловому расстоянию до эталонов класса

В матрицы ошибок на диагонали размещается число правильно распознанных примеров, а в остальных клетках – число ошибок классификатора. Двухстолбцовая таблица справа в каждой строке содержит число правильно распознанных примеров и суммарное значения ошибок по классу.

14.3.2. Классификация по максимальной проекции образа на подпространства эталонов

Нейронная сеть, как и прежде, обучается к эталонам классов. Векторы эталонов каждого класса образуют подпространства векторов, на которые проектируется вектор образа. Для нахождения проекции из конца вектора образа необходимо опустить перпендикуляр на подпространство класса. Обозначим через x вектор образа, а через p – его проекцию на подпространство класса, тогда вектор перпендикуляра будет равен разности векторов $x-p$. Этот вектор ортогонален к подпространству класса, а значит, и ко всем эталонам этого класса. Пусть e_i – некоторый эталон класса. Из условия ортогональности следует равенство нулю скалярного произведения:

$$(x-p,e_i)=(x,e_i)-(p,e_i)=0. \qquad (14.1)$$

Скалярные произведения $(x,e_i)=b_i$ вычисляются нейронной сетью. Положим, что число эталонов в классе равно m, что они линейно независимы и образуют базис подпространства класса. Тогда вектор проекции p представляется в этом базисе в виде суммы:

$$p = \alpha_1 e_1 + \alpha_2 e_2 + \ldots + \alpha_m e_m, \qquad (14.2)$$

где α_i – скаляры. Из выражения (14.1) непосредственно следует следующая система линейных уравнений:

$$\begin{cases} \alpha_1 (e_1, e_1) + \alpha_2 (e_1, e_2) + \ldots + \alpha_m (e_1, e_m) = b_1; \\ \alpha_1 (e_2, e_1) + \alpha_2 (e_2, e_2) + \ldots + \alpha_m (e_2, e_m) = b_2; \\ \ldots \\ \alpha_1 (e_m, e_1) + \alpha_2 (e_m, e_2) + \ldots + \alpha_m (e_m, e_m) = b_m. \end{cases}$$

Определитель этой системы известен как определитель Гама, он отличен от нуля при линейно независимых векторах e_i. Система имеет единственное решение и позволяет найти коэффициенты $\alpha_1, \alpha_2, \ldots, \alpha_m$ для каждого класса. Из разложения (14.2) находится вектор проекции в каждом классе s и определяется квадрат его длины $l_s^2 = (p_s, p_s)$. Образ относится к классу, для которого длина проекции максимальна. В табл. 14.4 приведено значение точности классификатора в зависимости от числа эталонов в условиях предыдущего эксперимента.

Таблица 14.4

Зависимость точности классификатора по проекциям от числа эталонов

Число эталонов в классе	1	10	20	30	40	50
Точность	71.8	83.9	87.3	89.3	89.6	89.4

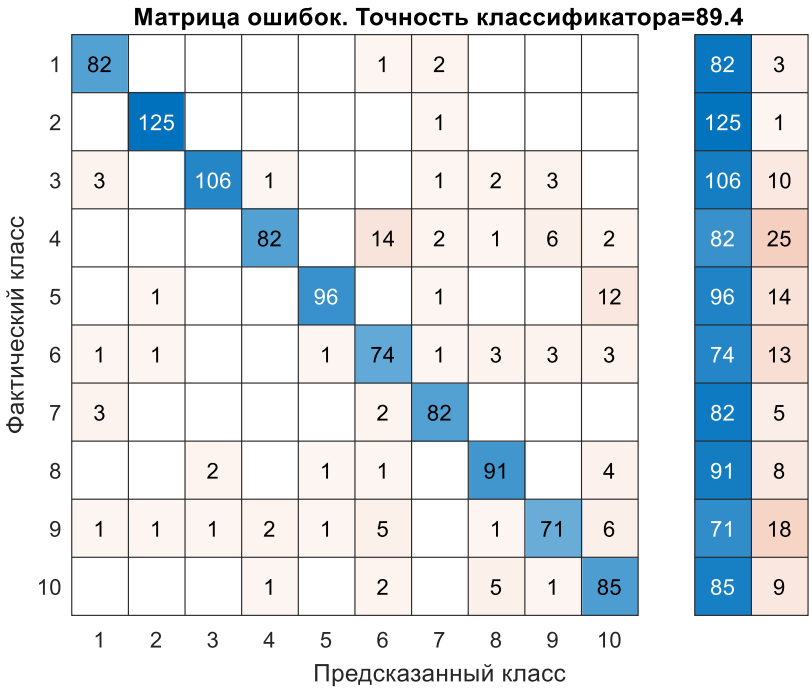


Рис. 14.4. Таблица ошибок при классификации по длине проекции на подпространство класса

Этот метод классификации дает лучшие результаты и более устойчив к помехам, чем предыдущий. На рис. 14.4 показана матрица ошибок для последнего столбца табл. 14.4.

14.3.3. Классификация по максимальной проекции в ортогональных базисах классов

Этот метод отличается от предыдущего предварительной ортогонализацией базисных векторов на подпространствах классов. Ортогонализации подвергаются подмножества вектора обучающей выборки, относящиеся к каждому классу. Ортогонализация выполняется на основе сингулярного разложения матрицы [53], составленной из векторов класса (см. прил. П.1). Векторы ортогонального базиса упорядочиваются по степени значимости в представлении эталонов. Степень значимости оценивается суммой квадратов проекций эталонов класса на вектор ортогонального базиса. Нейронная сеть настраивается на ортогональные векторы классов с максимальной значимостью. Для нейросетевого классификатора задается количество удерживаемых векторов для каждого класса.

Таблица 14.5

Зависимость точности метода классификации в ортогональных базисах от размерности модели

Число удерживаемых ортогональных векторов в модели классификатора	1	10	20	30	40	50
Точность	69.6	89.1	89.9	90.1	90.7	91.1

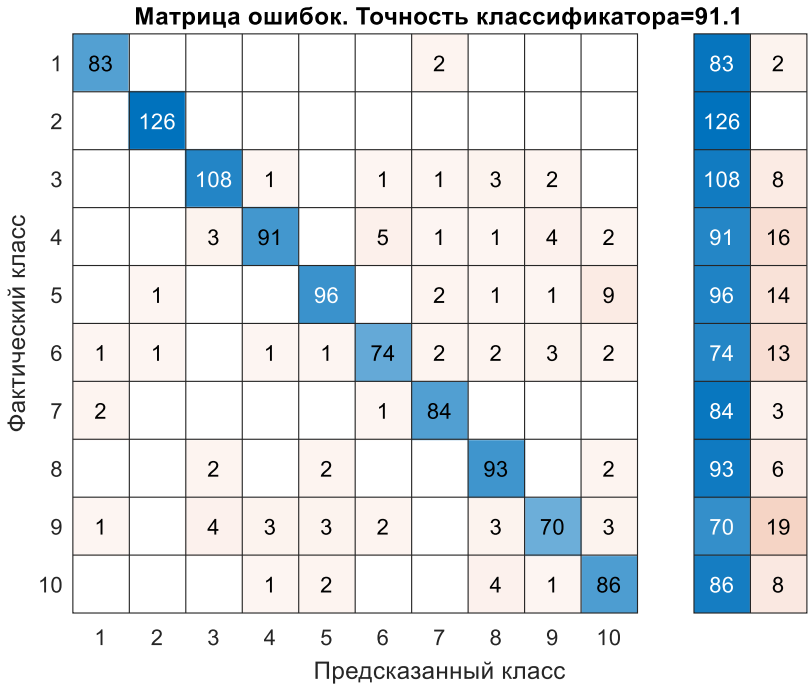


Рис. 14.5. Таблица ошибок при классификации методом проекций в ортогональных базисах

При распознавании образа нейронная сеть вычисляет скалярные произведения между образом и всеми ортогональными векторами подпространств классов. Вследствие ортогональности длина проекции образа на подпространство класса (квадрат длины) вычисляется как сумма квадратов скалярных произведений по ортогональным векторам класса. Этот метод работает быстрее предыдущего, поскольку не требует формирования и решения систем уравнений для определения проекций на подпространства классов. Кроме того, он позволяет уменьшить размерность модели классификатора по числу удерживаемых ортогональных векторов. В табл. 14.5 показана зависимость точности метода от числа сохраняемых ортогональных векторов в каждом классе.

Дальнейшее увеличение числа ортогональных векторов не приводит к улучшению точности классификатора. Этот метод классификации позволяет получить точность лучше, чем предыдущий. На рис. 14.5 показана матрица ошибок для последнего столбца табл. 14.5.

14.3.4. Классификация методом главных компонент

Метод главных компонент (англ. *principal component analysis – PCA*) изобретен Карлом Пирсоном в 1901 г. Иногда этот метод называют преобразованием Карунена–Лоэва. Выделение главных компонент приводит к задаче диагонализации выборочной ковариационной матрицы для каждого класса (см. прил. П.2). Эмпирическая или выборочная ковариационная матрица класса определяется выражением:

$$C = \frac{1}{m-1}\sum_{i=1}^{m}\left(x_i - \overline{x}\right)^T\left(x_i - \overline{x}\right),$$

где m – число примеров в классе; x_i – вектор-строка представителя класса; $\overline{x}$ – вектор-строка – среднее значение примеров в классе. Ковариационная матрица является симметричной и положительно определенной. Для такой матрицы можно найти собственные ортогональные векторы v, такие что

$$Cv = \lambda v.$$

Число собственных векторов равно размерности ковариационной матрицы. Скаляр λ называется собственным числом, соответствующим собственному вектору v. Собственные числа положительные, и чем большее значение они имеют, тем большей значимостью обладает собственный вектор. Для модели классификатора отбирают наиболее значимые векторы.

Распознаваемый вектор образа центрируется по среднему значению примеров в классе и раскладывается по выбранным собственным векторам

класса. Далее по коэффициентам разложения вычисляется длина проекции образа на подпространство, образованное собственными векторами. Образу присваивается метка класса, в котором длина проекции максимальна. В табл. 14.6 представлена зависимость точности PCA-классификатора от числа отобранных векторов при прежних условиях эксперимента.

Таблица 14.6

Зависимость точности PCA-классификатора от размерности модели по числу удерживаемых собственных векторов

Число собственных векторов в классе	1	10	20	30	40	50
Точность	83.88	92.14	93.4	92.96	93.08	92.54

Из таблицы видно, что увеличение числа векторов более 20 не улучшает точность. На рис. 14.6 показана матрица ошибок для 20 собственных векторов в каждом классе. Можно отметить, что точность классификатора, сравнимая с методом ортогональных проекций, достигается при меньшей размерности модели.

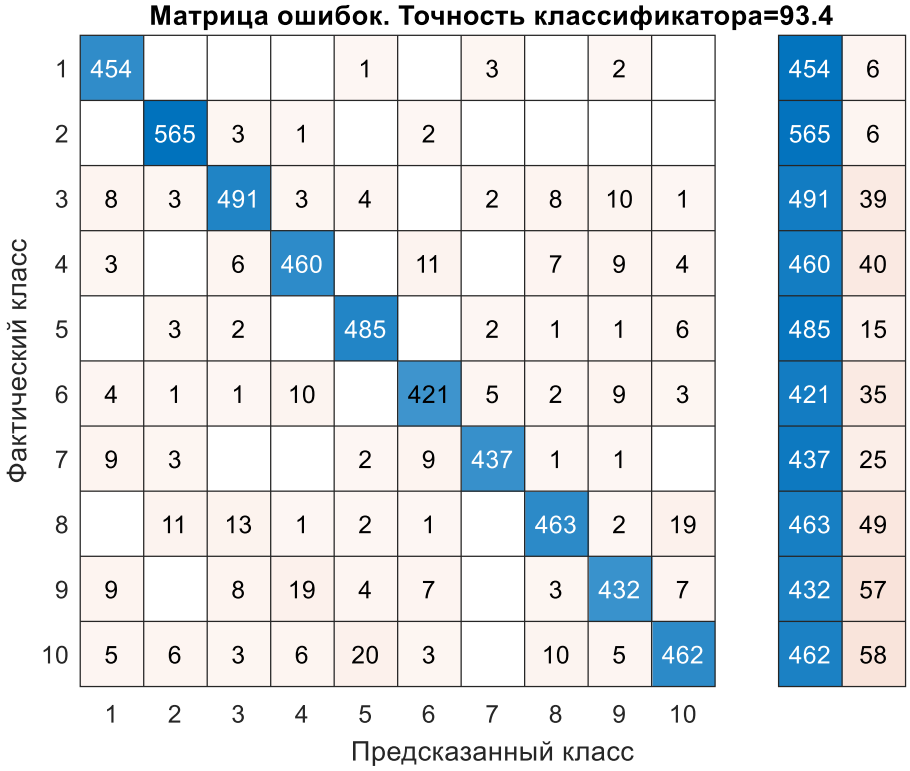


Рис. 14.6. Таблица ошибок при классификации методом PCA

Классификаторы, построенные на основе метода главных компонент и метода сингулярного разложения, близки по своим характеристикам. При использовании сингулярного разложения скорость обучения классификатора несколько выше, а точность немного меньше, чем у PCA-классификатора.

14.3.5. Классификация комитетом PCA-классификаторов

Для увеличения точности распознавания можно образовать комитет нейронных сетей, которые обучаются на разных фрагментах обучающей выборки. Комитет сетей принимает решение по принципу голосования. В табл. 14.7 показана зависимость точности распознавания от размера комитета. Комитет образован PCA-классификаторами с 20 собственными векторами в каждом классе и обученными на выборках, состоящих из 5000 примеров.

Таблица 14.7

Зависимость точности классификации от размера комитета

Число классификаторов в комитете	1	3	5	7	9	11
Точность	93.40	93.60	94.38	94.54	94.66	94.54

Рис. 14.7. Таблица ошибок при классификации голосующим комитетом

Лучший результат получен для комитета из 9 классификаторов. На рис. 14.7 приведена матрица ошибок для этого варианта.

14.3.6. PCA-классификатор с повторным входом

Классификатор реализует идеологию Эдельмана (см. 1.3), согласно которой эволюция биологических нейронных структур реализуется за счет повторного входа обучающих данных. Воспользуемся этим принципом для обучения PCA-классификатора. В обучающей выборке (далее будем называть ее контрольной) произвольно выделим базовое множество примеров, которое будем использовать для построения PCA-модели классификатора. На

каждом шаге эволюции по результатам классификации выполняется контроль ошибок модели на контрольном множестве. Примеры, на которых произошли ошибки, добавляются к базовому множеству с дифференциацией по классам, и модель PCA-классификатора строится заново. Процесс многократно повторяется до полного устранения ошибок на контрольном множестве. В процессе эволюции размер базового множества постепенно возрастает. Темп сходимости к нулевой ошибке зависит от размеров контрольного, начального базового множества и размерности модели (числа удерживаемых собственных векторов). Для набора данных MNIST максимальный объем контрольной выборки, на котором эволюционный процесс еще сходится, составляет примерно 20 000 примеров. Наилучшие результаты достигаются на малых выборках. На рис. 14.8 показана зависимость точности классификатора примеров с контрольной выборкой из 5000 примеров и начальной базовой выборки из 1000 примеров.

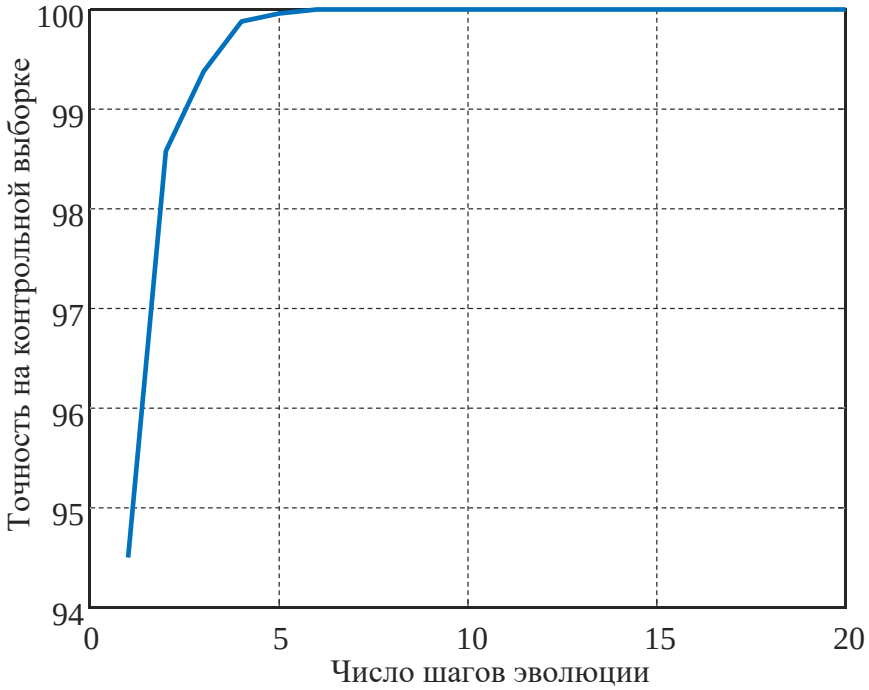


Рис. 14.8. Зависимость точности классификатора на контрольной выборке от числа шагов эволюции

Как видно из рис. 14.8, итерационный процесс сходится примерно за 6 шагов. Размер моделей в этом эксперименте составлял 30 собственных векторов. В процессе эволюции базовое множество возросло с 1000 до 1385 примеров. Обученный классификатор проверялся на тестовой выборке размером 10 000 примеров. На рис. 14.9 показана таблица ошибок классификатора. Достигнутая точность на тестовой выборке составила 94.6 %.

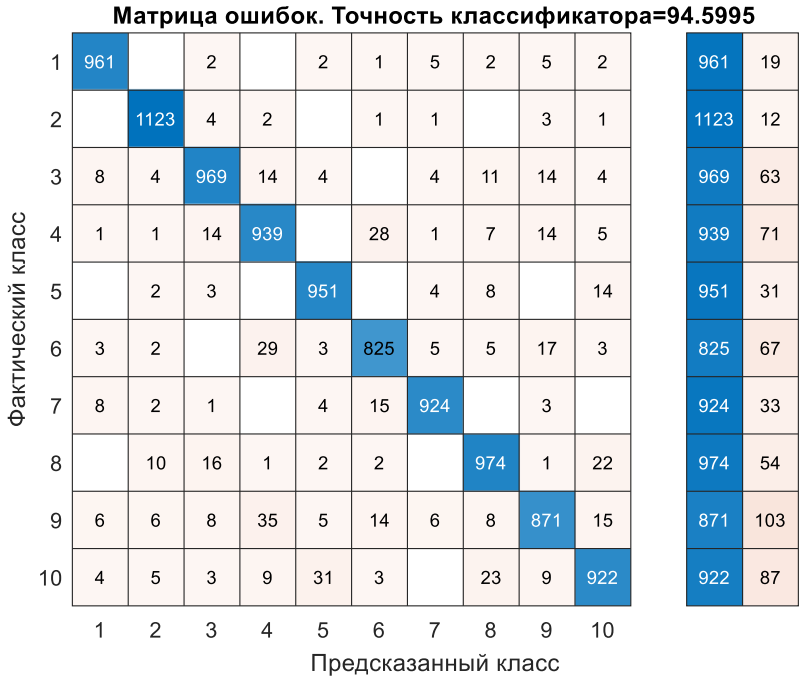


Рис. 14.9. Таблица ошибок эволюционного РСА-классификатора

РСА-классификатор при вычислении векторных проекций предварительно осуществляет разложение вектора-образа по усеченному ортогональному базису. Эти операции являются линейными и соответствуют скалярным произведениям векторов. Для их выполнения можно использовать рассмотренные ранее самоподобные пирамидальные нейронные сети, что обеспечивает распараллеливание алгоритмов классификации. Регулярный лес пирамидальных сетей, реализованный в виде сетей с коммутацией плоскостей, позволяет в параллельном варианте реализовать и комитет РСА-классификаторов.

14.3.7. Комитет эволюционных классификаторов

Эволюционный принцип можно распространить и на комитет классификаторов. Разделим всю обучающую выборку на контрольные множества небольшой размерности и для каждого контрольного множества обучим РСА-классификатор с повторным входом. Размещение контрольного множества классификатора на обучающей выборке назовем областью компетенции классификатора, а классификатор – компетентным. В отличие от ранее рассмотренного мажорирующего комитета, эволюционный комитет классификаторов использует принцип максимума при формировании результата. Фактически за результат классификации будет отвечать классификатор, в область компетенции которого наилучшим образом вписался распознаваемый образ.

На рис. 14.10 показана матрица ошибок для полного обучающего множества, состоящего из 60 000 примеров. Точность классификации составляет

чуть более 99 % при размере комитета, равном 12. На рис. 14.11 показана матрица ошибок этого же классификатора на полной тестовой выборке, состоящей из 10 000 примеров. Точность классификации составляет примерно 97 %.

Рис. 14.10. Таблица ошибок комитета эволюционных классификаторов для обучающей выборки

Рис. 14.11. Таблица ошибок комитета эволюционных классификаторов для тестовой выборки

Для эксперимента использовались классификаторы повторного входа с областью компетенции 5000 примеров и начальным размером базового множества 1000 примеров.

14.4. Набор данных Emnist Letters

Набор содержит 26 классов, каждый класс относится к одной букве латинского алфавита. Буквы написаны от руки участниками сбора данных. В каждом классе есть заглавные и строчные буквы. Образы представлены полутоновыми черно-белыми изображениями размером 28×28 пикселей (рис. 14.12). Обучающая выборка набора содержит 124 800 образов, а тестовая – 20 800 образов.

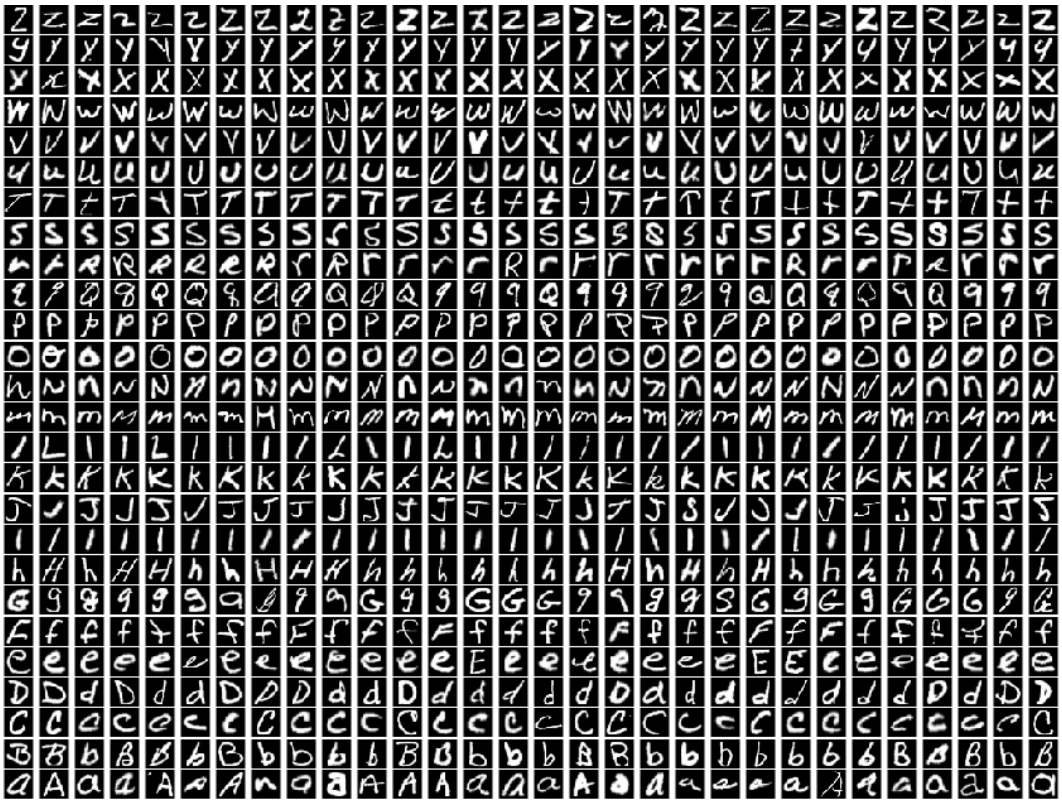


Рис. 14.12. Набор данных Emnist Letters

В эксперименте использовался эволюционный классификатор, который представлял собой комитет из 13 классификаторов повторного входа. Классификатор обучался на полном обучающем множестве. Двенадцать первых классификаторов тренировались на контрольном множестве из 10 000 примеров с начальным базовым множеством 2000 примеров. В последнем – 13 классификаторе размер контрольного множества составлял 4800 примеров. Подпространства классов аппроксимировались линейными подпространствами с удержанием в модели 50 векторов оптимального ортогональ-

ного базиса. Классы разбивались на два подкласса. Время обучения составило 1497 с на процессоре Intel(R) Core(TM) i3-6006U CPU 2.00GHz 1.99 GHz. Точность на обучающей выборке составила 98.1 %. Точность классификатора на тестовой выборке – 86.55 %. На момент первой публикации набора [54] (2017 г.) точность на тестовой выборке составляла 85.15 %. На конец 2023 г. достигнутая точность на тестовом наборе была 95.96 % на нейронной сети WaveMixLite-112/6 с числом настраиваемых параметров 4 млн.

14.5. Набор данных Fashion MNIST

Набор состоит из полутоновых черно-белых изображений размером 28×28 с числом классов 10. К классам относятся элементы одежды, обувь и сумки (рис. 14.13). Обучающий набор состоит из 60 000 образов, тестовый – из 10 000 образов.

Рис. 14.13. Набор данных Fashion MNIST

Для эксперимента использовался эволюционный классификатор, представляющий собой комитет из 6 классификаторов повторного входа. Классификатор тренировался на полном обучающем множестве. Размер контрольного множества для всех классификаторов – 10 000 примеров. Размер базового начального множества – 2000 примеров. Каждый класс разбивался на пять подклассов. Подпространства классов аппроксимировались линейными подпространствами с удержанием 50 векторов оптимального ортогонального базиса. Время обучения составило 528 с. Точность на обучающей выборке – 97.72 % (рис. 14.14). Точность на тестовой выборке – 88.18 % (рис. 14.15).

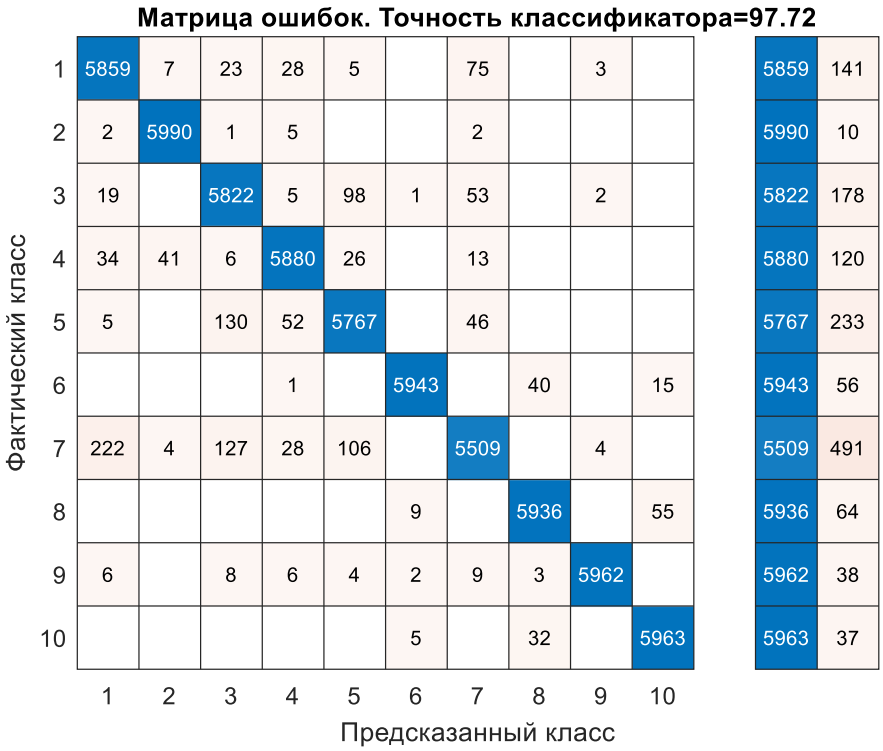


Рис. 14.14. Матрица ошибок классификатора
на обучающем множестве набора Fashion MNIST

На конец 2023 г. достигнутая точность на тестовом наборе составляет 96.91 % на нейронной сети Fine-Tuning DARTSC с числом настраиваемых параметров 3.2 млн.

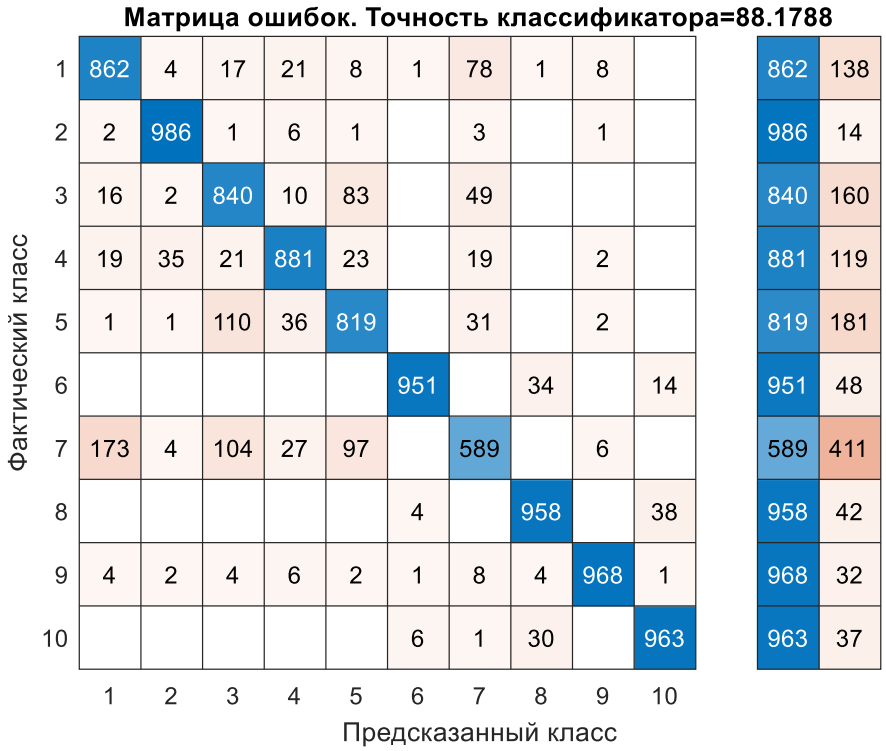


Рис. 14.15. Матрица ошибок классификатора
на тестовом множестве Fashion MNIST

В проведенных экспериментах не ставилась задача оптимальным подбором гиперпараметров классификаторов добиться максимальной точности классификатора. Цель данной главы состояла в том, чтобы последовательно продемонстрировать различные способы усиления обобщающей способности линейных классификаторов и показать применимость предложенных новых методов эволюционного обучения для различных наборов данных.

14.6. Эволюционный классификатор непрерывного обучения

Природа не разделяет данные на обучающую и тестовую выборки, биологический мозг обучается непрерывным потоком поступающих данных. С этой точки зрения имеет смысл отказаться от общепринятого деления выборки на обучающую и тестовую. Принцип разделения классификаторов по областям компетенции позволяет неограниченно наращивать размер комитета, покрывая всю выборку – и обучающую, и тестовую, а также все последующие поступившие данные. На рис. 14.16 приведена матрица ошибок для объединенной выборки комитетного классификатора для объединенной выборки MNIST с числом примеров 70 000. Точность классификатора составила 99.49 %.

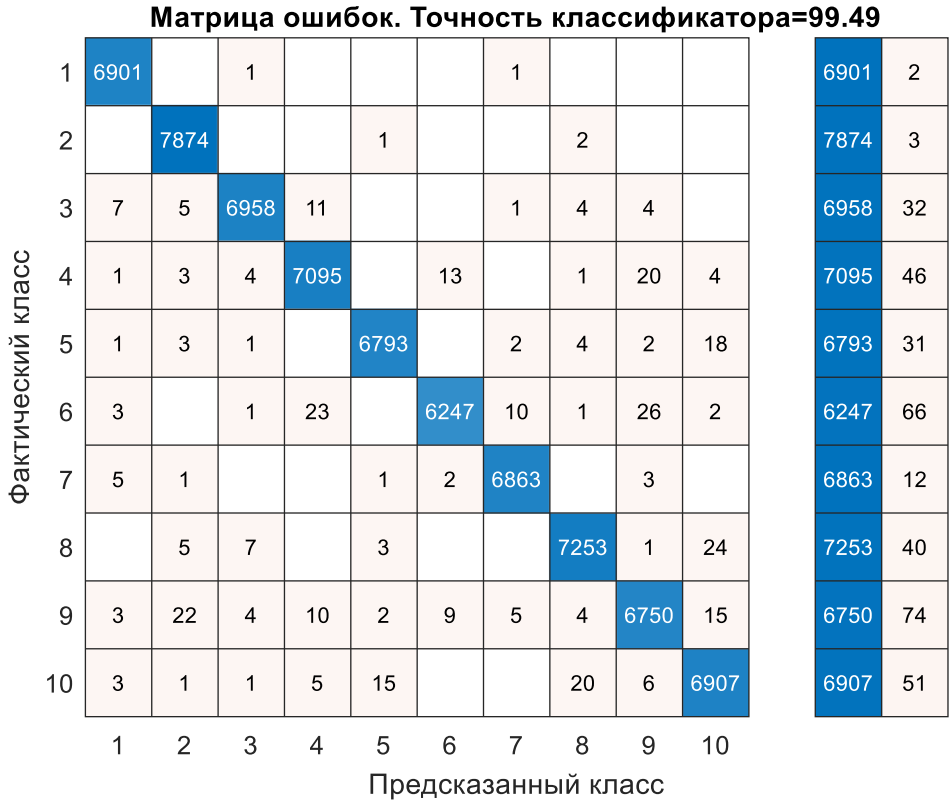


Рис. 14.16. Таблица ошибок непрерывного классификатора на объединенной выборке MNIST

В комитете использовались эволюционные РСА-классификаторы с областью компетенции 5000 примеров, начальным базовым множеством 1000 примеров и размером моделей 30 удерживаемых векторов. Число компетентных классификаторов для объединенной выборки из 70 000 примеров составило 14.

Эволюционный принцип повторного входа можно распространить также на весь комитет классификаторов. С этой целью в модели каждого компетентного классификатора запоминаются номера примеров базового множества на момент завершения обучения (т. е. достижения 100 % точности на области компетенции). Алгоритм эволюции для комитета на каждом шаге фиксирует номера примеров в объединенной выборке, на которых произошли ошибки. Эти примеры разделяются по областям компетенции частных классификаторов и добавляются в списки примеров базовых множеств. Далее заново выполняется обучение частных классификаторов с обновленными базовыми множествами до достижения 100 % точности на областях компетенции и вновь контролируются ошибки комитета классификаторов, затем итерации повторяются. Эволюционный процесс сходится к 100-процентной точности. На рис. 14.17 показана зависимость точности комитета классификаторов от числа шагов эволюции.

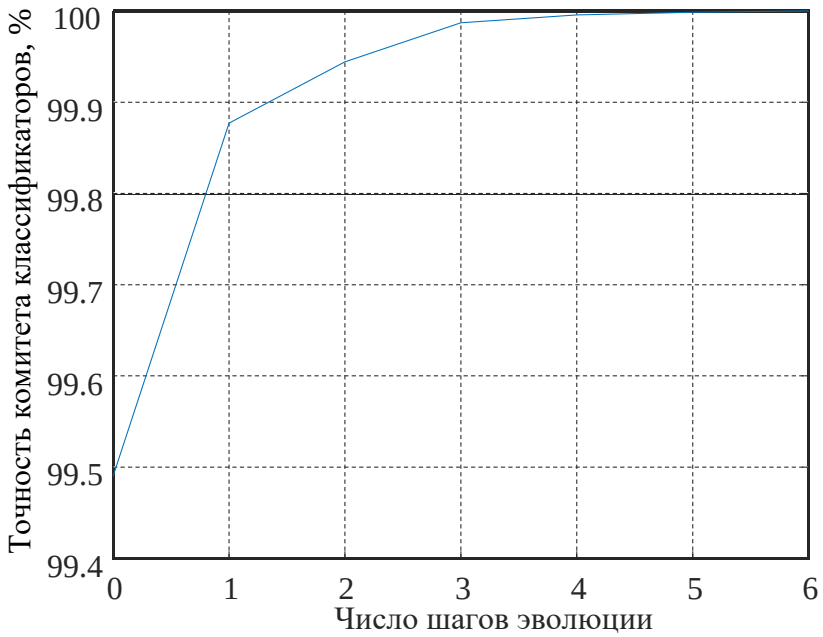


Рис. 14.17. Зависимость точности комитета классификаторов от числа шагов эволюции

Предложенная модель непрерывного эволюционного обучения комитета классификаторов обеспечивает 100-процентную точность распознавания объединенной выборки и может быть использована для построения самообу-

чающихся систем распознавания реального времени. Рассмотренный метод построения комитета классификаторов отличается от методов bagging и adaboost [54]–[56] использованием классификаторов с заданными областями компетенции и способом получения агрегативного решения комитета.

В отличие от метода коллективного распознавания Л. А. Растригина и Р. Х. Эренштейна [58], области компетенции классификаторов не локализуются расчетным путем, а назначаются исходя из условия достаточной представительности классов в базовых и контрольных множествах. Допустимо также частичное пересечение областей компетенции. Благодаря принципу повторного входа, частные классификаторы идеально обучаются на своих контрольных множествах. Это позволяет сделать процедуру классификации одноступенчатой, без предварительного определения области компетенции.

ЗАКЛЮЧЕНИЕ

Быстрые нейронные сети оказались удобным объектом, позволяющим реализовать результаты современных исследований биологического мозга. В работе показано, что модульность, самоподобие и эволюционное обучение биологических сетей имеют свое техническое воплощение в быстрых нейронных сетях. Благодаря структурному самоподобию, алгоритмы обучения БНС всегда сходятся за конечное число шагов, и время обучения сопоставимо со временем обработки данных сетью. Структура нейронной сети позволяет наращивать обучение без потери ранее накопленных знаний. Самоподобная структура дает возможность также аналитически оценить информационную емкость нейронной сети и реализовать сети, полностью ее использующие. Таким образом, термин «глубокое обучение» получает точный критерий оценки. Неожиданным свойством оказалась возможность реализации на БНС памяти образов и комбинационной логики. Это дает возможность построения сложных интеллектуальных систем с использованием только одной нейросетевой технологии.

Повторный вход – ключевое свойство эволюционного обучения биологических нейронных сетей. Согласно этому принципу на вход нейронной сети поступают данные, которые раньше уже использовались для обучения, и вместе с ними поступают новые данные, которые в процессе жизнедеятельности организма оказались верифицированы его параллельными системами, может быть и другой модальности. Такой процесс сходится в живых организмах и приводит к уточнению ранее полученных знаний. Предложенные в монографии PCA-классификаторы повторного входа моделируют этот принцип.

Биологический мозг эволюционирует под воздействием непрерывного потока данных. Модель этого процесса может быть описана эволюционирующим комитетом компетентных классификаторов, что дает надежду на создание эволюционирующих интеллектуальных систем, функционирующих и самостоятельно развивающихся в реальном мире. В данной работе показано, что для БНС не возникает вопрос объяснимости искусственного интеллекта, поскольку их создание, обучение и использование хорошо поддержано аналитикой математических моделей.

Список литературы

1. Солодовников А. И., Спиваковский А. М. Основы теории и методы спектральной обработки информации. Л.: Изд-во ЛГУ, 1986. 272 с.

2. Cooley J., Tukey J. An algorithm for the machine calculation of complex Fourier series// Math. Comput. 1965. Vol. 19. P. 297–301.

3. Good I. J. The Interaction Algorithm and Practical Fourier Analysis // J. of Royal Statistical Soseity. Ser. B. 1958. Vol. 20, no. 2. P. 361–372.

4. Andrews H. C., Caspari K. L. A General Techniques for Spectral Analysis // IEEE. Tr. Computer. 1970. Vol C-19, no. 1. P. 16–25.

5. Эндрюс Г. Применение вычислительных машин для обработки изображений / пер. с англ.; под ред. Б. Ф. Курьянова. М.: Энергия, 1977. 160 с.

6. Лабунец В. Г. Единый подход к алгоритмам быстрых преобразований // Применение ортогональных методов при обработке сигналов и анализа систем: межвуз. сб. / Свердловск: Уральск. политехн. ин-т, 1980. С. 4–14.

7. Дорогов А. Ю. Быстрые нейронные сети. СПб.: Изд-во Санкт-Петерб. ун-та, 2002. 80 с.

8. Рабинер Л., Гоулд Б. Теория и применение цифровой обработки сигналов / пер. с англ. М.: Мир, 1978. 848 с.

9. Дагман Э. Е., Кухарев Г. А. Быстрые дискретные ортогональные преобразования. Новосибирск: Наука, 1983. 228 с.

10. Дорогов А. Ю. Системные инварианты быстрых преобразований // Нейрокомпьютеры: разработка, применение. 2006. № 6. С. 43–50.

11. Andrews H. C., Caspari K. L. Degrees of Freedom and Modular Structure in Matrix Multiplication // IEEE Tp. Compt. 1971. Vol. C-20. P. 113–141.

12. Lorente de Nó, R. Cerebral Cortex: Architecture, intracortical connections, motor projections // Physiology of the Nervous System / J. F. Fulton (ed.). NY: Oxford University Press, 1949. P. 288–312.

13. Lorente de Nó R. Studies of the structure of the cerebral cortex. I. The area entorhinalis // J. Psychol. Neurol. 1933. Vol. 45. P. 381–438.

14. Lorente de Nó R. Studies of the structure of the cerebral cortex. II. Continuation of the study of the ammonic system // J. Psychol. Neurol. 1934. Vol. 46. P. 113–177.

15. Дж. Эделмен, В. Маунткасл. Разумный мозг. М.: Мир, 1981.

16. Mountcastle V. B. Sensory Hand: Neural Mechanisms of Somatic Sensation. Cambridge: Harvard University Press, 2005.

17. Vernon B. Mountcastle Introduction // Cerebral Cortex. 2003. Vol. 13, iss. 1. P. 2–4. URL: https://doi.org/10.1093/cercor/13.1.2 (дата обращения: 16.04.2024).

18. Jones E. G. Microcolumns in the cerebral cortex // Proc. of the National Academy of Sciences of the United States of America: journal. 2000. Vol. 97, no. 10. P. 5019–5021. PMID 10805761.

19. Edelman G. M. Neural Darwinism: selection and reentrant signaling in higher brain function // Neuron. 1993. Vol. 10 (2). P. 115–25. DOI: 10.1016/0896-6273(93)90304-a. PMID 8094962. S2CID 8001773.

20. Edelman G. M. Group selection and phasic reentrant signaling: a theory of higher brain function // The Mindful Brain: Cortical Organization and the Group-Selective Theory of Higher Brain Function. Cambridge Massachusetts: MIT Press, 1978.

21. Дорогов А. Ю. Быстрые нейронные сети: проектирование, настройка, приложения: лекции по нейроинформатике. Ч. 1 // Нейроинформатика-2004: тр. науч.-техн. конф., Москва, 28–30 января 2004 г. / МИФИ. Москва, 2004. С. 69–135.

22. Дорогов А. Ю. Теория и проектирование быстрых перестраиваемых преобразований и слабосвязанных нейронных сетей. СПб.: Политехника, 2014. 328 с.

23. Волкова В. Н., Денисов А. А. Основы теории систем и системного анализа: учеб. для студентов вузов. СПб.: Изд-во СПбГТУ, 1999. 512 с.

24. Кострикин А. И., Манин Ю. И. Линейная алгебра и геометрия: учеб. пособ. для вузов. 2-е изд. перераб. М.: Наука, 1986. 304 с.

25. Регулярный граф. URL: https://ru.wikipedia.org/wiki/Регулярный_граф (дата обращения: 16.04.2024).

26. Дорогов А. Ю. Структурный синтез модульных слабосвязанных нейронных сетей. Ч. 2: Ядерные нейронные сети / Кибернетика и системный анализ. 2001. № 4. С. 13–20.

27. Трахтман А. М., Трахтман В. А. Основы теории дискретных сигналов на конечных интервалах. М.: Сов. радио, 1975. 208 с.

28. Белман Р. Введение в теорию матриц. М.: Наука, 1976. 352 с.

29. Shor P. W. Algorithms for quantum computation: Discrete logarithms and factoring // Proc. 35th Annual Symposium on Foundations of Computer Science (Shafi Goldwasser, ed.) / IEEE Computer Society Press. 1994. P. 124–134.

30. Риффель Э., Полак В. Основы квантовых вычислений // Квантовый компьютер и квантовые вычисления. 2000. Т. 1, № 1. С. 4–57.

31. Phung S. L., Bouzerdoum A. A pyramidal neural network for visual pattern recognition / IEEE Transactions on Neural Networks. 2007. Vol. 18, no. 2. P. 329–343.

32. Gonzalez R. C., Woods R. E. Digital Image Pprocessing. Englewood Cliffs, NJ: Prentice-Hall, 2002.

33. Александров В. В., Горский Н. Д. Представление и обработка изображений: рекурсивный подход. Л.: Наука, 1985. 192 с.

34. Bischof H., Kropatsch W. G. Neural networks versus image pyramids // Artificial Neural Nets and Genetic Algorithms / Proc. of International Conference in Innsbruck, Austria, 1993, pp. 145–153. URL: http://dx.doi.org/10.1007/978-3-7091-7533-0

35. Gradient-based learning applied to document recognition // Lecun Y., Bottou L., Bengio Y., Haffner P. / Proc. of the IEEE 86(11):2278–2324. DOI: 10.1109/5.726791.

36. Lazebnik S., Schmid C., Ponce J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories // USA: CVPR, 2006. P. 2169–2178.

37. Ullahy I., Petrosinoy A. About Pyramid Structure in Convolutional Neural Networks. arXiv:1608.04064v1 [cs.CV] 14 Aug. 2016 URL: https://arxiv.org/pdf/1608.04064 (дата обращения: 16.04.2024).

38. Фишер Р. А. Статистические методы для исследователей. М.: Госиздат, 1958. 267 с.

39. Самарин А. И. Нейросетевые модели в задачах управления поведением робота // Лекции по нейроинформатике: материалы 3-й Всерос. науч.-техн. конф. «Нейроинформатика 2001» / МИФИ. Москва, 2001. С. 60–102.

40. Fukushima K., Miyake S., Takayuki I. Neocognitron: A neural network model for a mechanism of visual pattern recognition // IEEE Transaction on Systems, Man and Cybernetics SMC. 1983. Vol. 13 (5). P. 826–834.

41. Backpropagation Applied to Handwritten Zip Code Recognition / Y. LeCun, B. Boser, J. S. Denker et al. // Neural Computation. 1989. Vol. 1 (4). P. 541–551.

42. Striving for Simplicity: The All Convolutional Net / J. T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller (2014-12-21) arXiv:1412.6806. URL: https://arxiv.org/pdf/1412.6806 (дата обращения: 16.04.2024).

43. Дубровин Б. А., Новиков С. П., Фоменко А. Т. Современная геометрия: методы и приложения. Т. 2: Геометрия и топология многообразий. 5-е изд., испр. М.: Эдиториал УРСС, 2001. 296 с.

44. Введение в топологию / Ю. Г. Борисович, Н. М. Близняков, Я. А. Израилевич, Т. Н. Фоменко. М.: Наука, Физматлит, 1995. 416 с.

45. Ефимов Н. В., Розендорн Э. Р. Линейная алгебра и многомерная геометрия. М.: Наука, 1970. 528 с.

46. Steinhaus H. Sur la division des corps materiels en parties // Bull. Acad. Polon. Sci., 1956. C1. III. Vol. IV. P. 801–804.

47. Lloyd S. P. Least squares quantization in PCM // IEEE Transactions on Information Theory, 1 March 1982, Physics. DOI:10.1109/TIT.1982.1056489 Corpus ID: 10833328.

48. Прикладная статистика: классификация и снижение размерности / С. А. Айвазян, В. М. Бухштабер, И. С. Енюков, Л. Д. Мешалкин. М.: Финансы и статистика, 1989. 607 с.

49. Лагутин М. Б. Наглядная математическая статистика: учеб. пособие. М.: БИНОМ. Лаборатория знаний, 2007. 472 с.

50. Дорогов А. Ю. Кластерный анализ высокоразмерных данных в задаче классификации образов / Нейроинформатика, ее приложения и анализ данных: материалы XXVIII Всерос. семинара, Красноярск, 25–27 сентября 2020 г.; под ред. М. Г. Садовского / Институт вычислительного моделирования СО РАН. Красноярск, 2020. С. 42–49.

51. UCI Machine Learning Repository: Iris Data Set. URL: http://archive.ics.uci.edu/ml/datasets/Iris (дата обращения: 19.02.2023).

52. MNIST (база данных) // Википедия. URL: https://ru.wikipedia.org/wiki/MNIST_(база_данных) (дата обращения: 16.04.2024).

53. Воеводин В. В., Кузнецов Ю. А. Матрицы и вычисления. М.: Наука, 1984. 320 с.

54. Cohen G., Afshar S., Tapson J., van Schaik A. EMNIST: an extension of MNIST to handwritten letters. (2017). URL: http://arxiv.org/abs/1702.05373 (дата обращения: 16.04.2024).

55. Bauer E., Kohavi R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants // Machine Learning. 1999. No. 36. P. 105–142.

56. Breiman L. Bagging Predictors. Department of Statistics University of California Berkeley, California. 1994. Technical Report No. 421.

57. Yoav Freund, Robert E. Schapire. A decision-theoretic generalization of online learning and an application to boosting // Second European Conference on Computational Learning Theory (EuroCOLT '95), held in Barcelona, Spain in March 1995. URL: https://www.ee.columbia.edu/~sfchang/course/spr/papers/freund95decisiontheoretic-adaboost.pdf. (дата обращения: 16.04.2024).

58. Растригин Л. А., Эренштейн Р. Х. Метод коллективного распознавания. М.: Энергоиздат, 1981.

59. Воеводин В. В., Кузнецов Ю. А. Матрицы и вычисления. М.: Наука, 1984. 320 с.

60. Лагутин М. В. Наглядная математическая статистика: учеб. пособие. М.: БИНОМ. Лаборатория знаний, 2007. 472 с.

ПРИЛОЖЕНИЯ

П.1. Оптимальная аппроксимация данных линейными подпространствами

Пусть данные представлены конечным множеством векторов $X = x_1, x_2, \ldots, x_m \subset R^n$ заданных в n-мерном пространстве. Решение задачи аппроксимации для $k = 1,2,\ldots,n-1$ дается набором вложенных линейных подпространств $L_1 \subset L_2 \subset \cdots \subset L_k$, максимизирующих суммы квадратов проекций векторов данных $Q_k = \sum_{i=1}^{m} \Pr^2(x_i, L_k)$.

Будем полагать, что линейные подпространства $\{L_1, L_2, \cdots, L_{n-1}\}$ определяются ортонормированным набором векторов $W = \{w_1, w_2, \ldots, w_{n-1}\}$. В этом случае проекции вычисляются через скалярные произведения $\Pr^2(x_i, L_k) = \sum_{j=1}^{k} (x_i, w_j)^2$, а критерии оптимальности могут быть записаны в виде:

$$Q_k = \sum_{i=1}^{m} \sum_{j=1}^{k} (x_i, w_j)^2 = Q_{k-1} + \sum_{i=1}^{m} (x_i, w_k)^2 \to \max. \qquad (П.1)$$

Предположим, что ортонормированные векторы $\{w_1, w_2, \ldots, w_{k-1}\}$ уже выбраны оптимальным образом и, следовательно, значения критериев $\{Q_1, Q_2, \cdots, Q_{k-1}\}$ достигают максимума. Из (П.1) следует, что для максимизации критерия Q_k необходимо выбрать ортонормированный вектор w_k, обеспечивающий максимум суммы $\sum_{i=1}^{m} (x_i, w_k)^2$. Будем полагать, что векторы x_i являются строками матрицы X, а вектор w_k – столбцом матрицы W, тогда последнею сумму можно записать в матричной форме:

$$\Delta Q = \sum_{i=1}^{m} (x_i, w_k)^2 = w_k^* X^* (w_k^* X^*)^* = w_k^* X^* X w_k. \qquad (П.2)$$

Здесь символ (*) означает транспонирование векторов и матриц (или операцию сопряжения в случае задания данных в комплексном простран-

стве). Сумма ΔQ максимизируется при условии $w_k^* w_k = 1$. Задача поиска максимума является задачей на условный экстремум, функционал Лагранжа будет иметь вид

$$F = w_k^* S w_k - \lambda_k (w_k^* w_k - 1),$$

где использовано обозначение $S = X^* X$. В точке экстремума вариация функционала F равна нулю. Варьируя функционал F по векторам w_k и учитывая симметричность матрицы S, получим

$$\delta F = 2\delta w_k^* S w_k - 2\lambda_k \delta w_k^* w_k = 0.$$

Отсюда вследствие независимости вариаций δw_k следует необходимое условие экстремума $S w_k = \lambda_k w_k$, т. е. вектор w_k должен являться собственным вектором матрицы $S = X^* X$, а λ_k – собственное число, соответствующее данному вектору. Из (П.2) следует, что значение максимума для ΔQ будет равно λ_k. В наборе W векторы упорядочиваются в порядке убывания собственных чисел, это обеспечивает наилучшую аппроксимацию конечным набором вложенных линейных подпространств, начиная с подпространства L_1. Матрица W, составленная из собственных векторов матрицы S, очевидно, диагонализирует матрицу S, т. е. $W^* S W = \Lambda$, где Λ – диагональная матрица, содержащая собственные числа на главной диагонали.

Представленный вывод является частным случаем сингулярного разложения [58] произвольной матрицы в произведение матриц

$$X = U \Sigma W^*,$$

где W – ортогональная матрица, составленная из собственных векторов матрицы $S = X^* X$; U – ортогональная матрица, составленная из собственных векторов матрицы $G = X X^*$; $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r 0, \ldots, 0)$ – диагональная матрица сингулярных чисел с числом ненулевых элементов на диагонали, равным рангу матрицы X. Поскольку $S = X^* X = (U \Sigma W^*)^*, U \Sigma W^* = W \Sigma^2 W^*$. Отсюда следует $W^* S W = \Sigma^2 = \Lambda$, т. е. сингулярные числа связаны с собственными числами матрицы S соотношением $\lambda_i = \sigma_i^2$. В пакете МАТЛАБ для нахождения сингулярного разложения используется функция svd().

П.2. Главные компоненты класса

Пусть данные принадлежат одному классу и представлены конечным множеством векторов $X = x_1, x_2, \ldots, x_m \subset R^n$, заданных в n-мерном пространстве. Для аппроксимации данных будем использовать линейные многообразия $\{L_0, L_1, L_2, \cdots, L_{n-1}\}$ возрастающей размерности. Всякое k-мерное линейное многообразие может быть задано линейной комбинаций векторов $L_k = \{w_0 + \beta_1 w_1 + \beta_2 w_2 + \cdots + \beta_k w_k\}$, где скаляры β_i пробегают вещественную прямую, вектор w_0 определяет смещение k-мерной плоскости Π_k относительно начала координат, а векторы $W = \{w_1, w_2, \ldots, w_k\}$ являются ортонормированной системой векторов в плоскости Π_k.

Вектор w_0 ищется как задача минимизации суммы евклидовых расстояний между вектором w_0 и векторами $x_1, x_2, \ldots, x_m$:

$$D = \sum_{I=1}^{M} (w_0 - x_i, w_0 - x_i) = \sum_{i=1}^{m} (w_0 - x_i)^* (w_0 - x_i).$$

Варьируя функционал по вектору w_0 и учитывая симметрию произведений векторов, получим:

$$\delta D = \sum_{i=1}^{m} \left(\delta w_0^* (w_0 - x_i) + (w_0 - x_i)^* \delta w_0 \right) = 2\delta w_0 \sum_{i=1}^{m} (w_0 - x_i) = 0.$$

Откуда $w_0 = \frac{1}{m} \sum_{i=1}^{m} x_i = \bar{x}$. Центрируем исходные данные, положив $y_i = x_i - \bar{x}$. Составим матрицу Y из строк y_i и найдем ортонормированную систему векторов $W = \{w_1, w_2, \ldots, w_k\}$ из условия максимизации суммы проекций векторов y_i на плоскости Π_k. Как было показано в П.1, векторы-столбцы w_i должны быть собственными векторами матрицы $S = Y^* Y$. В методе главных компонент [59] вместо матрицы S используют выборочную ковариационную матрицу

$$C = \frac{1}{m-1} (X - \overline{X})^* (X - \overline{X}),$$

где $\overline{X}$ – матрица, составленная из математических ожиданий строк матрицы X. В случае, если аппроксимируются данные одного класса, все строки матрицы $\overline{X}$ совпадают с $\bar{x}$. Тогда ковариационная матрица будет равна

$$C = \frac{1}{m-1} Y^* Y .$$

Эта матрица имеет тот же набор собственных векторов, что и матрица $S = Y^* Y$, но собственные числа отличаются от собственных чисел матрицы S множителем $1/(m-1)$. Следует отметить, что ортонормированные векторы $W = \{w_1, w_2, \ldots, w_k\}$, полученные методом главных компонент, не совпадают с подобными векторами, полученными при аппроксимации данных линейными пространствами. Кроме того, модель классификатора на основе главных компонент включает в себя еще вектор смещения w_0. В пакете МАТЛАБ для нахождения собственных векторов используется функция eig().

ОГЛАВЛЕНИЕ

Научное издание

Дорогов Александр Юрьевич

Самоподобные нейронные сети быстрого обучения

Редактор Е. А. Ушакова

Компьютерная верстка Е. Н. Стекачевой

Подписано в печать 30.05.24. Формат 60×84 1/16. Бумага офсетная.
Печать цифровая. Гарнитура «Times New Roman». Печ. л. 11,75.
Тираж 500 экз. (1-й завод 1–50 экз.) Заказ .

Издательство СПбГЭТУ «ЛЭТИ»
197022, С.-Петербург, ул. Проф. Попова, 5Ф