Искусственные нейронные сети. Машинное обучение на основе нейронных сетей.

Особенности человеческого восприятия информации и современных систем управления. Вычислительный интеллект (ВИ). Методы ВИ. Искусственные нейронные сети (ИНС). Историческая справка. Основные проблемы, решаемые ИНС.

Особенности человеческого восприятия информации и современных систем управления

Изучение особенностей человеческого восприятия показывает, что одним из его недостатков являются ограниченные возможности по переработке поступающей информации, тем более что эта информация может быть неполной, чрезвычайно разнородной и искаженной. Очевидно, что эти обстоятельства значительно усложняют процесс распознавания оперативных единиц восприятия информации и оперативных образов управляемого объекта (паттернов), необходимых для формулирования решения.

Функциональное предназначение мнемических процессов (запоминание, воспроизведение, сохранение и забывание усвоенной ранее информации) при принятии управленческих решений заключается в обеспечении:

быстрой актуализации необходимой ЛПР в конкретной ситуации информации;

ситуативной избирательности памяти (ранжировании данных по их значимости);

надлежащего объема памяти и поддержания ее в рабочем состоянии при постоянно возрастающем утомлении;

структурирования единиц (паттернов) в памяти в соответствии с их логическими, функциональными и семантическими взаимосвязями;

удерживания в памяти противоречивой информации.

Помимо этого, в когнитивной психологии установлено, что кратковременная память ЛПР способна оперировать не более чем с 7±2 факторами. Следить за динамикой изменения большого числа взаимосвязанных факторов мозг не в состоянии. Кроме этого, памяти ЛПР свойственны такие когнитивные деформации, как: эвристика доступности

(более вероятно то событие, которое легче вспомнить); эвристика конкретности (более значима простая и понятная информация); эффект Ирвина (наиболее вероятно наступление желаемого события); эвристика репрезентативности (более вероятно наступление того события, которое соответствует накопленному опыту человека).

Показано, что интеллектуальные системы позволяют не только избежать приведенных когнитивных деформаций памяти ЛПР, но и обладают возможностями позволяющими структурировать, ранжировать и актуализировать чрезвычайно сложную и большую по объемам информацию. Более того, интеллектуальные системы способны хранить информацию в форме сценариев (последовательности действий), соотнесенных с конкретной ситуацией, что существенным образом упрощает процедуру принятия управленческих решений.

Подводя итог всему вышесказанному, а также учитывая современные тенденции в теории принятия решений можно сделать следующие выводы:

нельзя переоценивать значение математических методов и считать, что формальные методы математики являются универсальным средством решения задач в сфере управления военной, производственной, экономической и других видов деятельности;

методы, основанные на результатах опыта и интуиции, будут актуальны еще продолжительное время;

рассуждения правдоподобного характера (с позиции «здравого смысла») помогают сформировать математические модели, в основе которых лежит накопленный опыт разработки различных моделей, причем с формальной точки зрения эти математические модели можно рассматривать как некоторую систему аксиом. Другими словами, эти **модели обладают совокупностью знаний, которые определяют взаимосвязь между различными наблюдениями явлений в соответствии с фундаментальной теорией, но не следующих непосредственно из этой теории**. **В последнее время такой подход становится общепринятым при проведении различных системных исследований в военной, экономической, социальной и других областях;**

военные, экономические и социальные системы являются управляемыми, поэтому для них должна формироваться цель управления, что

с очевидностью приводит к понятию программы, находящейся вне модели, процедуру формирования которой невозможно полностью формализовать. Кроме того, элементы эвристики присутствует и в понятии критерия качества (эффективности), которое позволяет выбрать обоснованное решение из числа допустимых. **Следовательно, эвристические процедуры и методы в системных исследованиях и в конкретных задачах будут иметь большое значение;**

эвристические процедуры и методы, без которых нельзя представить себе функционирование сложных систем управления – это способы принятия решения, использующие накопленный и обобщенный опыт;

нельзя противопоставлять неформальные и строгие математические методы анализа, т.к. решение должно приниматься на основе сочетания обоих способов мышления.

Кроме того, необходимо учитывать и то, что быстрое увеличение уровня структурной сложности современных систем различного управления (СУ) и большое разнообразие выполняемых ими операций предопределяет трудности принятия решений ЛПР. При этом свою деятельность они должны основывать на понимании того, что каждая СУ обладает следующими особенностями:

нестационарностью (изменчивостью) отдельных параметров системы и стохастичностью своего поведения в целом;

уникальностью и непредсказуемостью поведения системы в конкретных условиях, т. к. в системе присутствуют активные элементы - люди, приводящие систему к «свободе воли»;

способностью изменения своей структуры при сохранении целостности;

формированием различных вариантов поведения, что обусловлено наличием в системе активных элементов;

способностью противостоять энтропийным тенденциям, т.е. обладают свойством **гомеостатичности**;

способностью адаптироваться к изменяющимся условиям;

способностью и стремлением к целеобразованию (в системах с активными элементами цели формируются внутри системы).

Одно время приведенные особенности многие исследователи пытались преодолеть с помощью **классических систем искусственного интеллекта**,

т.е. систем, основанных на символьных вычислениях и других формальных методах. Однако довольно скоро выяснилось, что с помощью символьной обработки информации, в большинстве случаев, не удается решить прикладные задачи для сложных эргодических систем поддержки принятия решений (систем экономического планирования, социальных систем большой размерности), если для них невозможно получить полную информацию или если их определение недостаточно полно.

Как показали исследования, проведенные 1980 - 90-х годах, выходом в сложившейся ситуации явилось использование **систем на основе вычислительного интеллекта** (в зарубежной литературе чаще употребляется название - мягкие вычисления **- Soft Computing**).

Вычислительный интеллект

Под **вычислительным интеллектом** (ВИ) понимают научное направление, где решаются задачи искусственного интеллекта на основе новых нетрадиционных методов вычислений, а под технологией ВИ понимают совокупность нетрадиционных методов вычислений и средств обработки знаний, документооборота, методов выработки и выбора альтернативных вариантов решений, объединенных в целостную технологическую систему для принятия и доведения решений до исполнителей.

В настоящее время считают, что ВИ включает в себя следующие основные методы:

нейросетевые – методы, использующие обучение, адаптацию, классификацию, системное моделирование и идентификацию систем на основе исходных данных;

нечеткой логики – методы, основанные на теории нечетких множеств и обеспечивающей эффективные средства математического отражения неопределенности и нечеткости исходной информации, позволяющие построить модель, адекватную исследуемой предметной области;

генетические – методы, использующие синтез, настройку и оптимизацию исследуемых систем с помощью специальным образом организованного случайного поиска и эволюционного моделирования.

Перечисленные методы считаются основными в ВИ, однако, необходимо заметить, что число новых методов примкнувших к ним в последнее время постоянно расширяется, не являясь строго определенным. Например:

когнитивная компьютерная графика – методы визуализации данных, позволяющие активировать наглядно-образные механизмы мышления ЛПР, облегчающие принятие решения в сложной обстановке или нахождение решения сложной проблемы;

нелинейная динамика - это наука, изучающая структуру и свойства эволюционных процессов в нелинейных динамических системах.

На сегодняшний день существует достаточно большое количество разнообразных классификаций современных информационных технологий учитывающих парадигмы рассматриваемых предметных областей. Приведем одну из них

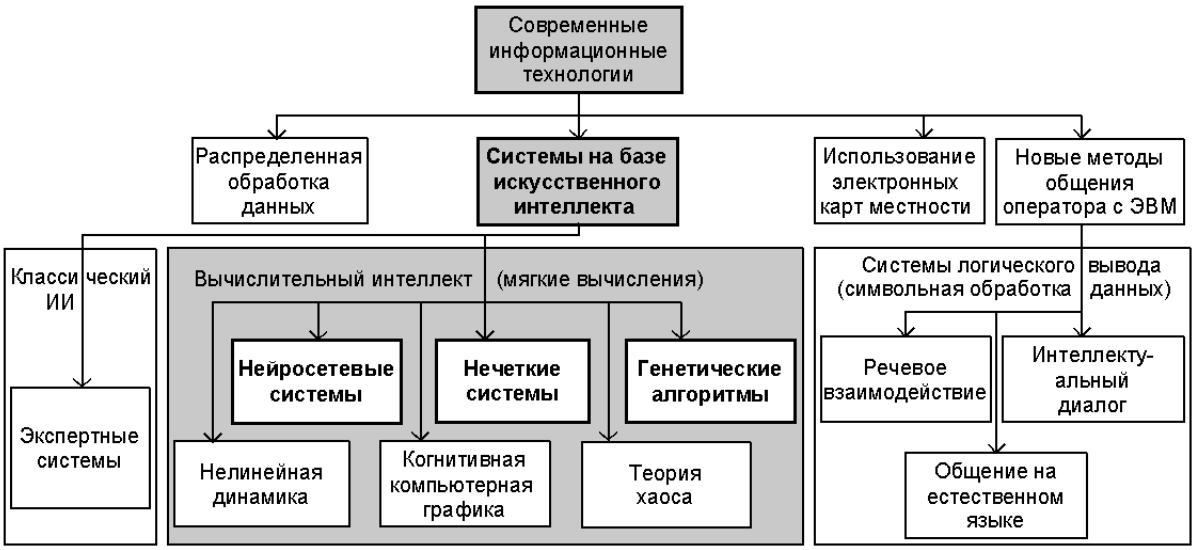

Рис. 1. Классификация современных информационных технологий

Основными характеристиками интеллектуальных систем (ИС) на базе ВИ, определяющими их применение в области управления, являются следующие:

способность к обучению и самообучению – способность ИС на базе ВИ после предъявления входной информации самонастраиваться, обеспечивая получение выходной информации с требуемой точностью, в отличие от систем,

основанных на символьных вычислениях в принципе не способных к самообучению;

способность к адаптации – свойство ИС на базе ВИ быстро изменять свои параметры в соответствии с изменяющейся обстановкой;

«прозрачность» объяснения – способность ИС на базе ВИ представлять извлеченные из данных знания в понятном эксперту или ЛПР виде;

способность открывать новое – способность ИС на базе ВИ выявлять ранее неизвестные, скрытые связи и отношения в больших массивах числовой, текстовой и визуальной информации, прогнозировать появление новых процессов и тенденций;

нелинейность – способность ИС на базе ВИ аппроксимировать сколь угодно сложные нелинейные функции с любой, заранее заданной точностью;

универсальность – способность ИС на базе ВИ решать широкий круг задач и быть свободной от каких-либо предположений относительно источника данных;

параллелизм – параллельная процедура обработки информации;

устойчивость – способность ИС на базе ВИ продолжать выполнение задачи, сохраняя заданное качество решений, в условиях, когда повреждена часть ее структуры;

креативность – способность ИС на базе ВИ порождать новые (не встречающиеся при обучении) варианты решения задачи.

Методы ВИ

Результаты многочисленных исследований показали, что одним из способов, обеспечивающих существенное повышение оперативности, многовариантности, гибкости и простоты реализации в процессе выбора решения, является применение нейронных сетей (НС).

Нейронные сети позволяют с любой степенью точности аппроксимировать произвольную непрерывную функцию многих переменных, функционируя на основе принципа самообучения. В результате предъявления НС множества обучающих примеров (возможных вариантов решений с учетом изменяющейся обстановки) и соответствующей корректировки выходных параметров в ней устанавливаются такие связи, которые обеспечивают

получение решения при анализе реальных ситуаций, которые НС не предъявлялись. Особенно хорошо НС зарекомендовали себя при решении задач распознавания и классификации. Практическое применение НС показало их преимущество перед другими системами - **быстро выдавать альтернативные варианты решения (предложения)**, что является одним из важнейших условий принятия решения.

Нечеткие системы основаны на теории нечетких множеств и основное их преимущество по сравнению с другими методами ВИ состоит в **легкой проверке и понимании правил, на основе которых строится их база данных**.

Генетические алгоритмы (ГА) представляют собой разновидность эволюционных методов и помогают находить оптимальные решения многофакторных задач. Они основаны на механизме биологической эволюции и могут быть использованы как инструмент поиска новых (ранее неизвестных) закономерностей. ГА обеспечивают прозрачность толкования решений и легко адаптируются к изменяющимся исходным данным.

На рис. 2 представлены экспертные оценки основных методов ИИ в соответствии с требуемыми характеристиками [Фролов, 2000].

Сравнение основных методов ИИ показывает что, нейросетевые методы лидируют, опережая другие методы по таким характеристикам, как способность к обучению, обобщение и адаптация.

Такой вывод подтверждается результатами проведенных в России и за рубежом многочисленных исследований в области нейроматематики. Кроме этого, в последнее десятилетие, были получены новые результаты, позволяющие значительно расширить представления о потенциальных возможностях НС, как инструмента решения достаточно широкого круга задач, характерных для автоматизированных систем управления.

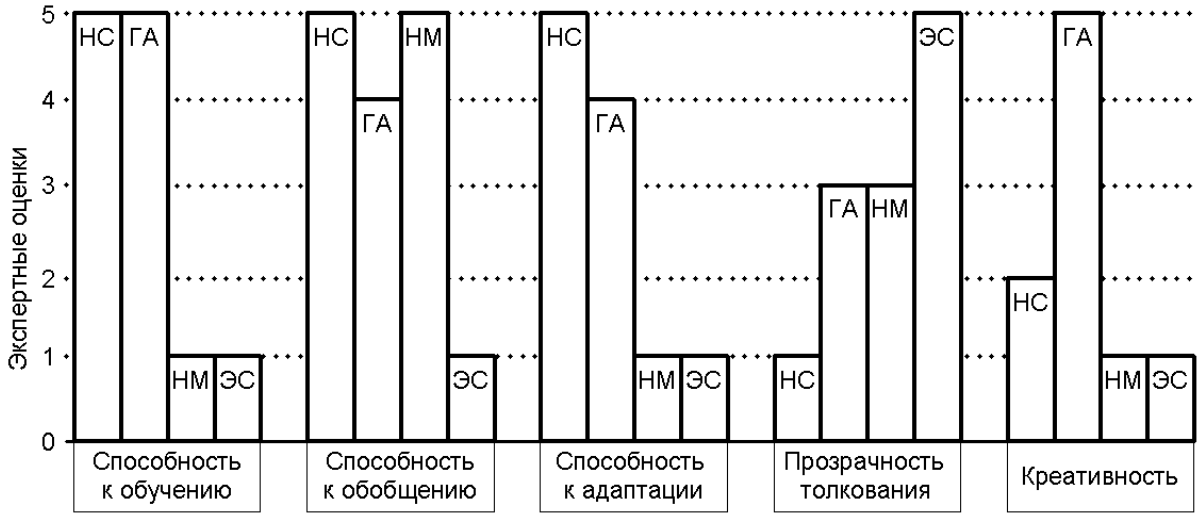

Рис. 2. Сравнение экспертных оценок основных методов ИИ: НС – нейронные сети;
ГА – генетические алгоритмы; НМ – нечеткие множества; ЭС – экспертные системы

Когнитивная компьютерная графика (ККГ) – компьютерные системы визуализации данных, позволяющие активировать наглядно-образные механизмы мышления ЛПР, облегчающие принятие решения в сложной обстановке или нахождение решения сложной проблемы. Суть концепции ККГ заключается в том, что если на экране дисплея удается визуализировать существенные свойства и отношения между объектами некоторой предметной области (любой степени абстрактности), то такой ККГ – образ, как правило, содержит в себе информацию (на уровне графических деталей компьютерного изображения) о возможных и не всегда заранее известных следствиях этих свойств и отношений, помогающую проанализировать новые закономерности исследуемой предметной области.

Нелинейная динамика - это наука, изучающая структуру и свойства эволюционных процессов в нелинейных динамических системах. Особенностью, присущей исключительно нелинейным системам, является возможность реализации в них множества различных режимов функционирования, которые зависят от начального состояния, параметров системы и внешних воздействий. В частности, в нелинейных системах возможны режимы детерминированного хаоса в виде незатухающих апериодических колебаний, напоминающих случайный процесс. Изучение

свойств нелинейных систем обусловлено тем, что социотехнические системы в своем существовании и развитии подчиняются нелинейным законам. Линейные закономерности также имеют место, однако они представляют собой лишь частный случай общих нелинейных законов.

Теория хаоса представляет собой математический аппарат, описывающий поведение некоторых нелинейных динамических систем, подверженных, при определённых условиях, явлению, известному как хаос, которое характеризуется сильной чувствительностью поведения системы к начальным условиям. Результатом такой чувствительности является то, что поведение такой системы кажется случайным, даже если модель, описывающая систему, является детерминированной. Примерами подобных систем являются атмосфера, турбулентные потоки, биологические популяции, общество как система коммуникаций и его военные, экономические, политические, социальные и другие подсистемы.

Необходимо особо отметить, что ККГ в последнее время приобретает настолько большое значение в процессе эффективного способа организации значимой для ЛПР информации, что необходимо провести подробный анализ ее роли в этом процессе, чтои будет сделано в последующих лекциях.

Искусственные нейронные сети

Искусственные нейронные сети (ИНС) строятся по принципам организации и функционирования их биологических аналогов. Такие сети предназначены для решения широкого круга задач: распознавания образов, идентификации, прогнозирования, оптимизации, управления сложными объектами. Дальнейшее повышение производительности компьютеров все в большей мере связывают с ИНС, в частности, с нейрокомпьютерами (НК), основу которых составляет искусственная нейронная сеть.

До недавнего времени технологии искусственного интеллекта считали излишней аппеляцию к архитектуре мозга, его нейронным структурам, и декларировали необходимость моделирования работы человека со знаниями. Сейчас более популярна точка зрения, что искусственные нейронные сети могут заменить собой современный искусственный интеллект. Однако многое свидетельствует о том, что оба подхода будут существовать вместе,

объединяясь в системах, где каждый из них используется для решения тех задач, с которыми он лучше справляется.

Историческая справка

Часто теорию нейронных сетей считают молодой наукой. Однако ее истоки относятся к периоду ранних работ в области компьютерных наук, психологии и философии. Например, еще Джон фон Нейман восхищался теорией клеточных автоматов и нейроподобным подходом к вычислениям.

Окончательно термин «нейронные сети» сформировался к середине 50 годов XX века. Основные результаты в этой области связаны с именами У. Маккалоха, Д. Хебба, Ф. Розенблатта, М. Минского, Дж. Хопфилда.

В 1943 году У. Маккалох (W. McCulloch) и У. Питтс (W. Pitts) предложили модель нейрона и сформулировали основные положения теории функционирования головного мозга.

В 1949 году Д. Хебб (D. Hebb) высказал идеи о характере соединений нейронов мозга и их взаимодействии и описал правила обучения нейронной сети.

В 1958 году Ф. Розенблатт (F. Rosenblatt) разработал принципы организации и функционирования персептронов, предложил вариант технической реализации первого в мире нейрокомпьютера Mark.

В 1969 году была опубликована книга М. Минского (M. Minsky) и С. Пейперта (S. Papert) «Персептроны», в которой была доказана принципиальная ограниченность возможностей персептронов, что послужило причиной угасания интереса к искусственным нейронным сетям.

В начале 1980 годов происходит возобновление интереса к искусственным нейронным сетям как следствие накопления новых знаний о деятельности мозга, а также значительного прогресса в области микроэлектроники и компьютерной техники.

В 1982—1985 годах Дж. Хопфилд (J. Hopfield) предложил семейство оптимизирующих нейронных сетей, моделирующих ассоциативную память.

1987 год стал началом широкомасштабного финансирования разработок в области ИНС и НК в США, Японии и Западной Европе.

В 1989 году разработки и исследования в области ИНС и НК ведутся практически всеми крупными электротехническими фирмами.

Нейрокомпьютеры становятся одним из самых динамичных секторов рынка (за два года объем продаж вырос в пять раз). В 1997 году объем продаж на рынке ИНС и НК превысил 2 млрд. долларов, а ежегодный прирост составил 50%.

В 2000 году благодаря переходу на субмикронные и нанотехнологии, а также успехам молекулярной и биомолекулярной технологий происходит переход к принципиально новым архитектурным и технологическим решениям по созданию нейрокомпьютеров.

Основные проблемы, решаемые искусственными нейронными сетями

Классификация образов. Задача состоит в указании принадлежности входного образа, представленного вектором признаков, одному или нескольким предварительно определенным классам. К известным приложениям относятся распознавание букв, распознавание речи, классификация сигнала электрокардиограммы, классификация клеток крови, задачи рейтингования.

Кластеризация и категоризация. При решении задачи кластеризации, которая известна также как классификация образов без учителя, отсутствует обучающая выборка с образцами классов. Алгоритм кластеризации основан на подобии образов и размещает близкие образы в один кластер. Известны случаи применения кластеризации для извлечения знаний, сжатия данных и исследования их свойств.

Аппроксимация функций. Предположим, что имеется обучающая выборка $(X_1, Y_1), (X_2, Y_2), \ldots, (X_N, Y_N),$ которая генерируется неизвестной функцией, искаженной шумом. Задача аппроксимации состоит в нахождении оценки этой функции.

Предсказание / прогноз. Пусть заданы *N* дискретных отсчетов $\{y(t_1), y(t_2), \ldots, y(t_n)\}$ в последовательные моменты времени $t_1, t_2, \ldots, t_n$. Задача состоит в предсказании значения $y(t_{n+1})$ в момент t_{n+1} . Прогнозы имеют значительное влияние на принятие решений в бизнесе, науке и технике.

Оптимизация. Многие проблемы в математике, статистике, технике, науке, медицине и экономике могут рассматриваться как проблемы оптимизации. Задачей оптимизации является нахождение решения, которое

удовлетворяет системе ограничений и максимизирует или минимизирует целевую функцию.

Каким образом нейронная сеть решает все эти иногда неформализуемые или трудно формализуемые задачи? Как известно, для решения таких задач традиционно применяются два основных подхода.

Первый, основанный на правилах (***rule-based***), характерен для экспертных систем. Он базируется на описании предметной области в виде набора правил (аксиом) **«ЕСЛИ ..., ТО ...»** и **правил вывода**. Искомое знание представляется в этом случае теоремой, истинность которой доказывается посредством построения цепочки вывода. При этом подходе, однако, необходимо **заранее знать весь набор закономерностей, описывающих предметную область**.

Второй подход основан на примерах (***case-based***), надо лишь иметь достаточное количество примеров для настройки адаптивной системы с заданной степенью достоверности. Нейронные сети представляют собой классический пример такого подхода.

Искусственные нейронные сети. Персептроны

Биологический нейрон. Искусственный нейрон. Активационная функция. Классификация и свойства искусственных нейронных сетей (ИНС). Обучение ИНС. Теорема Колмогорова. Алгоритм обучения персептрона. Линейная разделимость и персептронная представляемость.

Биологический нейрон.

Нервная система и мозг человека состоят из нейронов, соединенных между собой нервными волокнами. Нервные волокна способны передавать электрические импульсы между нейронами. Все процессы перехода раздражений от кожи, ушей и глаз к мозгу, процессы мышления и управления действиями реализуются в живом организме как передача электрических импульсов между нейронами. Нейрон (нервная клетка) является особой биологической клеткой, которая обрабатывает информацию (рис.1).

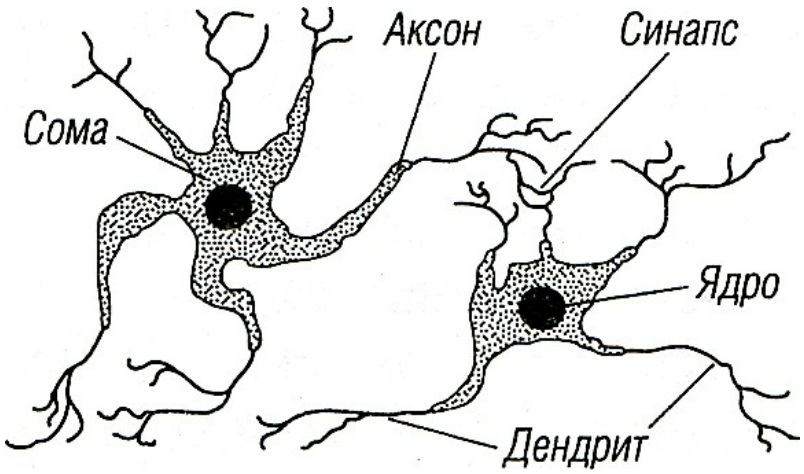

Рис. 1. Взаимосвязь биологических нейронов

Он состоит из тела, или сомы, и отростков нервных волокон двух типов: дендритов, по которым принимаются импульсы, и единственного аксона, по которому нейрон может передавать импульс. Тело нейрона включает ядро, которое содержит информацию о наследственных свойствах, и плазму, обладающую молекулярными средствами для производства необходимых нейрону материалов. Нейрон получает сигналы (импульсы) от аксонов других нейронов через дендриты (приемники) и передает сигналы, сгенерированные телом клетки, вдоль своего аксона (передатчика), который в конце разветвляется на волокна. На окончаниях этих волокон находятся специальные образования - синапсы, которые влияют на величину импульсов.

Кора головного мозга человека содержит около 10^{11} нейронов. Каждый нейрон связан с 10^{3}-10^{4} другими нейронами. В целом мозг человека содержит приблизительно от 10^{14} до 10^{15} взаимосвязей.

85 000 000 000 нейронов в организме человека.

Искусственный нейрон.

Каждый искусственный нейрон характеризуется своим текущим состоянием по аналогии с нервными клетками головного мозга, которые могут быть возбуждены или заторможены. Искусственный нейрон обладает группой синапсов — однонаправленных входных связей, соединенных с выходами других нейронов, а также имеет аксон — выходную связь данного нейрона, с

которой сигнал (возбуждения или торможения) поступает на синапсы следующих нейронов. Общий вид искусственного нейрона приведен на рис. 2.

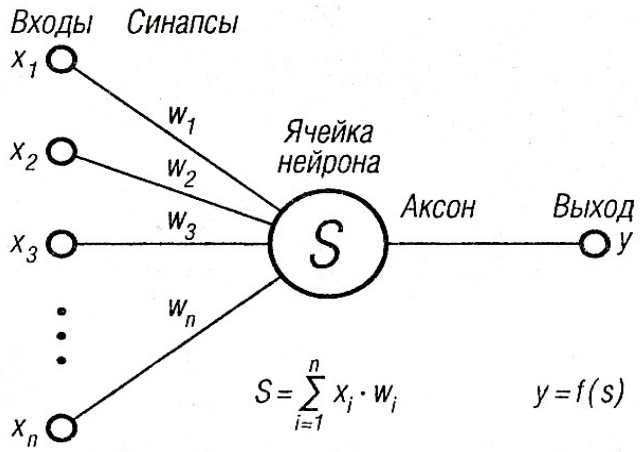

Рис. 2. Модель искусственного нейрона

Искусственный нейрон в первом приближении имитирует свойства биологического нейрона. Здесь множество входных сигналов, обозначенных $x_1, x_2, \dots x_n$, поступает на искусственный нейрон. Эти входные сигналы, в совокупности обозначаемые вектором X, соответствуют сигналам, приходящим в синапсы биологического нейрона. Каждый синапс характеризуется величиной **синапсической связи** или ее **весом** w_i. Каждый сигнал умножается на соответствующий вес $w_1, w_2, \dots, w_n$, и поступает на суммирующий блок. Каждый вес соответствует «силе» одной биологической синапсической связи. (Множество весов в совокупности обозначаются вектором W) Суммирующий блок, соответствующий телу биологического элемента, складывает взвешенные входы алгебраически, создавая величину S.

Таким образом, текущее состояние нейрона определяется как как взвешенная сумма его входов:

$$S=\sum_{i-1}^{n} x_i \cdot w_i.$$

Выход нейрона есть функция его состояния: $y=f\{S\}$, где f - активационная функция, более точно моделирующая нелинейную передаточную характеристику биологического нейрона и предоставляющая нейронной сети большие возможности. Примеры некоторых активационных

функций представлены в табл. 1 и на рис. 3. Наиболее распространенными являются пороговая и сигмоидальная активационные функции.

Таблица 1.

Функции активации нейронов

Название	Формула	Область значений
Пороговая (функция единичного скачка)	$f(s)=\begin{cases}0, & s<\Theta;\\ 1, & s\geqslant\Theta\end{cases}$	$\{0, 1\}$
Линейная	$f(s)=ks$	$(-\infty, +\infty)$
Логистическая (сигмоидальная)	$f(s)=\frac{1}{1+e^{-as}}$	$(0, 1)$
Гиперболический тангенс	$f(s)=\frac{e^{as}-e^{-as}}{e^{as}+e^{-as}}$	$(-1, 1)$
Линейная с насыщением (линейный порог)	$f(s)=\begin{cases}0, & s<0;\\ s/\Theta, & 0\leqslant s<\Theta\\ 1, & s\geqslant\Theta\end{cases}$	$[0, 1]$

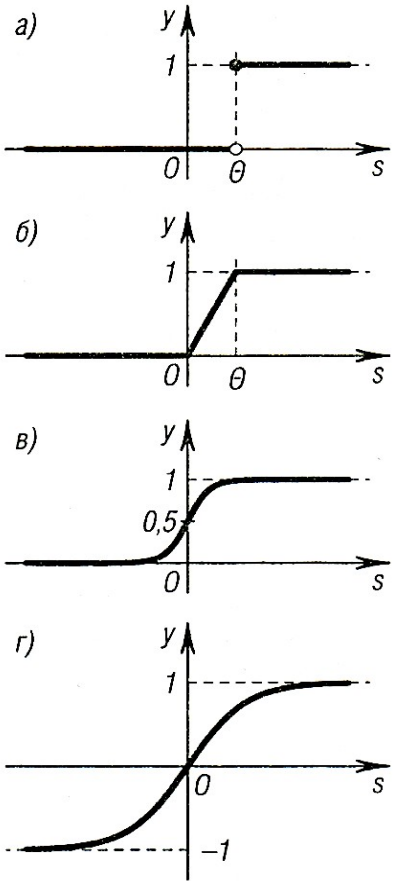

Рис. 3. Примеры активационных функций: а) функция единичного скачка; б) линейный порог; в) логистическая функция; г) гиперболический тангенс

Пороговая функция ограничивает активность нейрона значениями 0 или 1 в зависимости от величины комбинированного входа s . Как правило, входные значения в этом случае также используются бинарные: $x_i \in \{0,1\}$. Чаще всего удобнее вычесть пороговое значение θ , называемое смещением, из величины комбинированного входа и рассмотреть пороговую функцию в математически эквивалентной форме:

$$s = w_0 + \sum_{i=1}^{n} x_i \cdot w_i, f(s) = \begin{cases} 0, s < 0 \\ 1, s \geq 0 \end{cases}.$$

Здесь $w_0 = -\theta$ - **величина смещения**, взятая с противоположным знаком. Смещение обычно интерпретируется как связь, исходящая от элемента, значение которого всегда равно 1. Комбинированный вход тогда можно представить в виде $s = \sum_{i=0}^{n} x_i \cdot w_i$, где x_0 всегда считается равным 1.

Логистическая функция, или сигмоид, $f(s) = \frac{1}{1 + e^{-a \cdot s}}$ непрерывно заполняет своими значениями диапазон от 0 до 1; параметр a всегда положителен. При уменьшении параметра a график сигмоида становится более пологим, в пределе при a =0 вырождаясь в горизонтальную линию на уровне 0,5, при увеличении параметра a график сигмоида приближается к виду функции единичного скачка с порогом 0. **Следует отметить**, что сигмоидальная функция дифференцируема на всей оси абсцисс, что используется в некоторых алгоритмах обучения. Кроме того, она обладает **свойством усиливать слабые сигналы и предотвращает насыщение от больших сигналов, так как они соответствуют тем областям аргументов, где сигмоид имеет пологий наклон.**

Классификация и свойства искусственных нейронных сетей (ИНС). Однослойные ИНС.

Хотя один нейрон и способен выполнять простейшие процедуры распознавания, сила нейронных вычислений в соединениях нейронов. Простейшая сеть состоит из группы нейронов, образующих слой рис. 4.

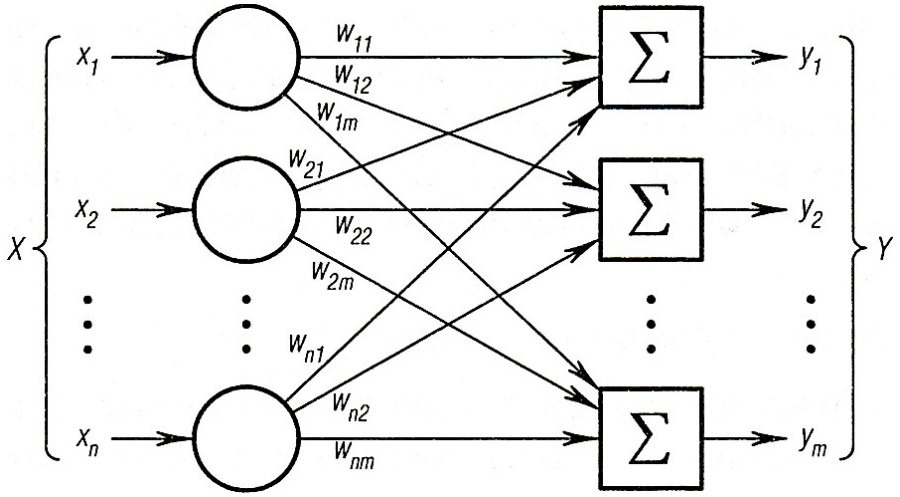

Рис. 4. Простейшая однослойная нейронная сеть

Отметим, что вершины-круги - распределение входных сигналов. Они не выполняют каких-либо вычислений и не считаются слоем. Вычисляющие нейроны обозначены квадратами. Каждый элемент из множества входов X соединен с каждым искусственным нейроном отдельной связью, которой приписан вес. А каждый нейрон выдает взвешенную сумму входов в сеть. В **искусственных и биологических сетях многие соединения могут отсутствовать**, на рисунке все соединения показаны в целях общности. Могут иметь место также соединения между выходами и входами элементов в слое. Удобно считать веса элементами матрицы W. Матрица имеет n строк и m столбцов, где $n-¿$ число входов, а $m-¿$ число нейронов. Таким образом, вычисление выходного вектора Y, компонентами которого являются выходы y_i нейронов, сводится к матричному умножению $Y = X \cdot W$.

Многослойные ИНС.

Более крупные и сложные нейронные сети обладают, как правило, и большими вычислительными возможностями. Хотя в литературе иногда рассматриваются полносвязные нейронные сети (где каждый нейрон передает свой выходной сигнал остальным нейронам, включая самого себя), именно послойная организация нейронов (где каждый нейрон передает свой выходной сигнал только нейронам из соседнего слоя) копирует слоистые структуры определенных отделов мозга. Оказалось, что такие многослойные сети обладают большими возможностями, чем однослойные, и в последние годы были разработаны многообразные алгоритмы для их обучения.

Многослойные сети могут образовываться каскадами слоев. Выход одного слоя является входом для последующего слоя. Подобная сеть показана на рис. 5 и изображена со всеми соединениями.

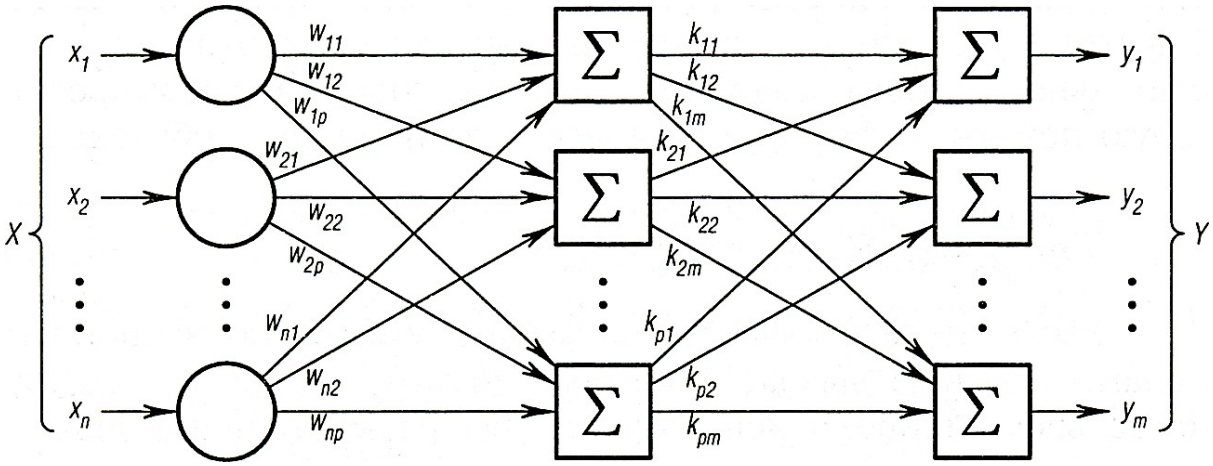

Рис. 5. Многослойная нейронная сеть

Многослойные сети могут привести к увеличению вычислительной мощности по сравнению с однослойной сетью лишь в том случае, если активационная функция между слоями **нелинейна**. Вычисление выхода слоя заключается в умножении входного вектора на первую весовую матрицу с последующим умножением (**если отсутствует нелинейная активационная функция**) результирующего вектора на вторую весовую матрицу. Так как умножение матриц **ассоциативно**, то двухслойная линейная сеть эквивалентна одному слою с весовой матрицей, равной произведению двух весовых матриц. **Следовательно, любая многослойная линейная сеть может быть заменена эквивалентной однослойной сетью**.

Обучение ИНС.

Сеть обучается, чтобы для некоторого множества входов давать требуемое (или, по крайней мере, сообразное с ним) множество выходов. Каждое такое входное (или выходное) множество рассматривается как вектор. Обучение осуществляется путем последовательного предъявления входных векторов с одновременной подстройкой весов в **соответствии с определенной процедурой**. В процессе обучения веса сети постепенно становятся такими, чтобы каждый входной вектор вырабатывал выходной вектор. **Различают алгоритмы обучения с учителем и без учителя.**

Обучение с учителем предполагает, что для каждого входного вектора существует целевой вектор, представляющий собой требуемый выход. Вместе они называются обучающей парой. Обычно сеть обучается на некотором числе таких обучающих пар. Предъявляется выходной вектор, вычисляется выход сети и сравнивается с соответствующим целевым вектором, разность (ошибка) с помощью обратной связи подается в сеть, и веса изменяются в соответствии с алгоритмом, стремящимся минимизировать ошибку. Векторы обучающего множества предъявляются последовательно, для каждого вектора вычисляются ошибки и подстраиваются веса до тех пор, пока ошибка по всему обучающему массиву не достигнет приемлемо низкого уровня.

Обучение без учителя не нуждается в целевом векторе для выходов и, следовательно, не требует сравнения с предопределенными идеальными ответами. **Обучающее множество состоит лишь из входных векторов**. Обучающий алгоритм подстраивает веса сети так, чтобы получались согласованные выходные векторы, т. е. чтобы предъявление достаточно близких входных векторов давало одинаковые выходы. **Процесс обучения выделяет статистические свойства обучающего множества и группирует сходные векторы в классы**. Предъявление на вход вектора из данного класса даст определенный выходной вектор, но до обучения невозможно предсказать, какой выход будет активироваться данным классом входных векторов. Следовательно, выходы подобной сети должны трансформироваться в некоторую понятную форму, обусловленную процессом обучения.

Теорема Колмогорова.

Рассмотрим двухслойную нейронную сеть с п входами и одним выходом, которая проста по структуре и широко используется для решения прикладных задач рис. 6.

Каждый -й нейрон первого слоя ($i=1,2,\dots,m$) имеет n входов, которым приписаны веса $w_{1i}, w_{2i}, \dots, w_{ni}$.

Получив входные сигналы, нейрон суммирует их с соответствующими весами, затем применяет к этой сумме передаточную функцию и пересылает результат на вход нейрона второго (выходного) слоя. В свою очередь, нейрон выходного слоя суммирует полученные от второго слоя сигналы с некоторыми

весами v_i. Для определенности будем предполагать, что передаточные функции в скрытом слое являются **сигмоидальными**, а в выходном слое используется **тождественная** функция, т. е. взвешенная сумма выходов второго слоя и будет ответом сети.

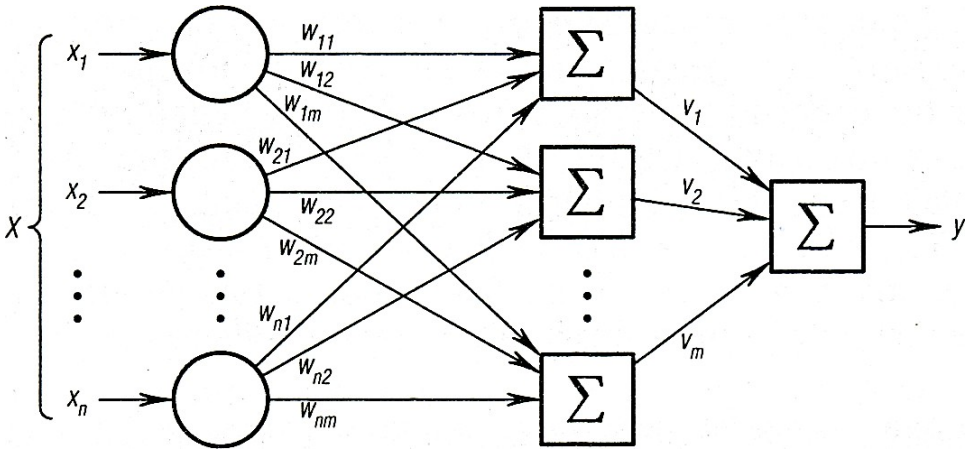

Рис. 6. Пример ИНС

Подавая на входы любые числа $x_1, x_2, \ldots, x_n$, мы получим на выходе значение некоторой функции $y = F(x_1, x_2, \ldots, x_n)$, которое является ответом (реакцией) сети. Ответ сети зависит как от **входного сигнала**, так и от **значений ее внутренних параметров** - весов нейронов. Точный вид этой функции:

$$F(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{m} ¿ \; v_i \sigma ¿ \sum_{j=0}^{n} ¿ \; x_j w_{ji} ¿,$$

где $\sigma(s) = \dfrac{1}{1+e^{-a \cdot s}}$

В 1957 году математик А. Н. Колмогоров доказал следующую теорему.

Теорема Колмогорова. Любая непрерывная функция F , определенная на n -мерном единичном кубе, может быть представлена в виде суммы $2n+1$ суперпозиций непрерывных и монотонных отображениий единичных отрезков:

$$h_{ij}(x_j)$$
$$\sum_{j=1}^{n} ¿$$
$$¿$$
$$g_i ¿$$
$$F(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{2n+1} ¿$$
$$x$$
$$x = (¿¿1, x_2, \ldots, x_n), 0 \le x_i \le 1,$$
$$¿$$

где g_i и h_{ij} - непрерывные функции, причем h_{ij} не зависят от функции F .

Эта теорема означает, что **для реализации функций многих переменных достаточно операций суммирования и композиции функций одной переменной**. К сожалению, при всей своей математической красоте, теорема Колмогорова малоприменима на практике. Это связано с тем, что функции $h_{ij} - ¿$ в общем случае негладкие и трудновычислимые; также неясно, каким образом можно подбирать функции g_i для данной функции F . **Роль этой теоремы состоит в том, что она показала принципиальную возможность реализации сколь угодно сложных зависимостей с помощью относительно простых автоматов типа нейронных сетей.** Более значимые для практики результаты в этом направлении были получены только в 1989 году, зато одновременно несколькими исследователями.

Пусть $F(x_1, x_2, \ldots, x_n)$ - любая непрерывная функция, определенная на ограниченном множестве, и $\varepsilon > 0$ - любое сколь угодно малое число, означающее точность аппроксимации. Через σ обозначена сигмоидальная функция.

Теорема. Существуют число m , набор чисел w_{ji} и набор чисел v_i такие, что функция

$$x_j w_{ji}$$
$$\sum_{j=0}^{n} ¿,$$
$$¿$$
$$v_i \sigma ¿$$
$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{m} ¿$$

приближает данную функцию $F(x_1, x_2, \ldots, x_n)$ с погрешностью не более ε на всей области определения.

Легко заметить, что эта формула полностью совпадает с выражением, полученным для функции, реализуемой нейросетью. **В терминах теории ИНС эта теорема формулируется так: любую непрерывную функцию нескольких переменных можно с любой точностью реализовать с помощью двухслойной нейросети с достаточным количеством нейронов в скрытом слое.**

Алгоритм обучения персептрона

Одной из первых искусственных сетей, способных к перцепции (восприятию) и формированию реакции на воспринятый стимул, явился PERCEPTRON Розенблатта (F. Rosenblatt, 1958). **Персептроном**, как правило, называют однослойную нейронную сеть, при этом каждый персептронный нейрон в качестве активационной функции использует функцию единичного скачка (пороговую).

Для простоты рассмотрим вначале процедуру обучения персептрона, состоящего только из одного нейрона. Подробная схема такого персептрона изображена на рис. 7.

Как отмечалось ранее, будем считать, что персептрон имеет дополнительный вход x_0 , который всегда равен 1. В таком случае, пороговое смещение $\theta=0$ и $y=f(s)=\{0, s<0 1, s\geq 0.$

Обучение персептрона состоит в подстройке весовых коэффициентов w_i , где $i=1,2,...,n$. Обученный персептрон способен разделять требуемое множество образов на два

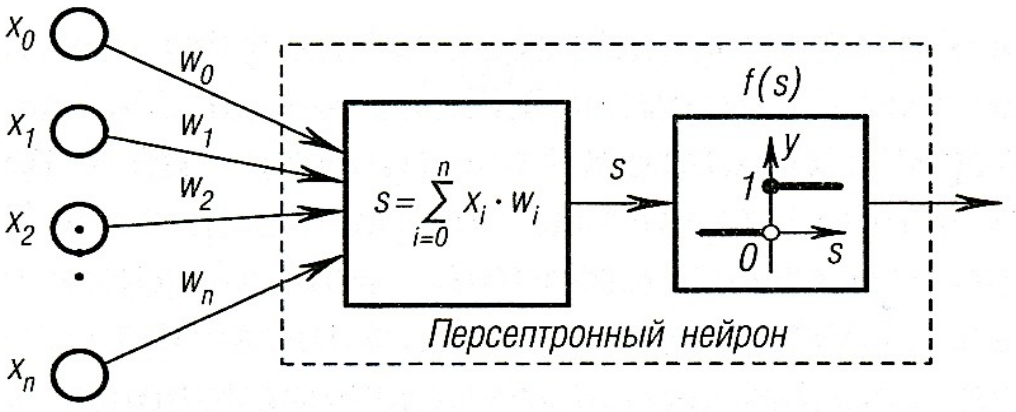

Рис. 7. Однонейронный персептрон с n входами

класса. (К первому классу относятся входные образы, для которых на выходе персептрона получено нулевое значение, ко второму классу - образы, для которых получено единичное значение.)

Обучение персептрона - это обучение с учителем, т. е. должен существовать набор векторов $(X^k, y_k), k=1,2,\dots,p$, называемый обучающей выборкой. Здесь $X^k=(x_0^k, x_1^k, x_2^k, \dots, x_n^k), k=1,2,\dots,p$ - примеры входных образов, для которых заранее известна их принадлежность к одному из двух данных классов.

Будем называть персептрон обученным на данной обучающей выборке, если при подаче на вход каждого вектора X^k на выходе всякий раз получается соответствующее значение $y_k \in \{0,1\}$. Предложенный Ф. Розенблаттом метод обучения состоит в итерационной подстройке весовых коэффициентов w_i последовательно уменьшающей выходные ошибки.

Алгоритм включает несколько шагов:

Шаг 0. Проинициализировать весовые коэффициенты $w_i, i=1,2,\dots,n$ небольшими случайными значениями, например, из диапазона [-0,3; 0,3].

Шаг 1. Подать на вход персептрона один из обучающих векторов X^k и вычислить его выход y .

Шаг 2. Если выход правильный ($y=y_k$), перейти на шаг 4. Иначе вычислить ошибку — разницу между верным и полученным значениями выхода: $\delta=y_k-y$.

Шаг 3. Весовые коэффициенты модифицируются по следующей формуле:

$$w_{ij}^{t+1}=w_{ij}^t+v\cdot\delta\cdot x_i.$$

Здесь t и $t+1$ - номера соответственно текущей и следующей итераций; v - коэффициент скорости обучения ($0<v\le 1$); x_i. - i -я компонента входного вектора X^k .

Шаг 4. Шаги 1—3 повторяются для всех обучающих векторов. **Один цикл последовательного предъявления всей выборки называется эпохой**. Обучение завершается по истечении нескольких эпох, когда сеть перестанет ошибаться.

Замечание 1. Коэффициент скорости обучения v является параметром данного алгоритма. Как правило, его выбирают из диапазона [0,5; 0,7]. В некоторых случаях (при большом объеме обучающей выборки) целесообразно постепенно уменьшать значение v начиная, например, с 1.

Замечание 2. Используемая на шаге 3 формула модифицирует только весовые коэффициенты, отвечающие ненулевым значениям входов x_i^k, поскольку только они влияли на величину $s=\sum_{i=1}^{n} x_i \cdot w_i$, а следовательно, и на значение y.

Очевидно, что если $y_k > y$ (получен неправильный нулевой выход вместо правильного единичного), то, поскольку $\delta > 0$, весовые коэффициенты (а вместе с ними и величина s) будут увеличены и тем самым уменьшат ошибку. В противном случае весовые коэффициенты будут уменьшены и сумма s тоже уменьшится, приближая тем самым y к значению y_k.

Обобщим теперь этот алгоритм обучения на случай однослойной сети, включающей n персептронных нейронов (рис. 4). Такая сеть (при достаточно большом числе нейронов) может осуществлять разбиение образов на произвольное требуемое число классов.

Пусть имеется обучающая выборка, состоящая из пар векторов (X^k, Y^k), $k=1,2,\ldots,p$. Назовем нейронную сеть обученной на данной обучающей выборке, если при подаче на входы сети каждого вектора X^k на выходах всякий раз получается соответствующий вектор Y^k. Обучение заключается в итерационной подстройке матрицы весов W (w_{ij} — вес синаптической связи между i-м входом и j-м нейроном), последовательно уменьшающей ошибку в выходных векторах. Алгоритм включает следующие шаги:

Шаг 0. Проинициализировать элементы весовой матрицы W небольшими случайными значениями.

Шаг 1. Подать на входы один из входных векторов X^k и вычислить его выход Y.

Шаг 2. Если выход правильный $Y=Y$ $(¿¿k)$ $¿$, перейти на шаг 4. Иначе, вычислить вектор ошибки - разницу между идеальным и полученным значениями выхода: $\delta=Y^k-Y$.

Шаг 3. Матрица весов модифицируется по следующей формуле: $w_{ij}^{t+1}=w_{ij}^t+v\cdot\delta\cdot x_i$. Здесь t и $t+1$ - номера соответственно текущей и следующей итераций; v - коэффициент скорости обучения ($0<v\le 1$); x_i. - i -я компонента входного вектора X^k ; j - номер нейрона в слое.

Шаг 4. Шаги 1—3 повторяются для всех обучающих векторов. Обучение завершается, когда сеть перестанет ошибаться.

Представленный метод обучения носит название δ - правило. Доказанная Розенблаттом теорема о сходимости обучения по δ - правилу говорит о том, что персептрон способен обучиться любому обучающему набору, который он способен представить.

Линейная разделимость и персептронная представляемость.

Каждый нейрон персептрона является формальным пороговым элементом, принимающим единичные значения в случае, если суммарный взвешенный вход больше некоторого порогового значения:

$$Y_j=\{0,\sum_{i=1}^{n}x_i\cdot w_i<\Theta\ 1,\sum_{i=1}^{n}x_i\cdot w_i\ge\Theta.$$

Таким образом, при заданных значениях весов и порогов, нейрон имеет определенное выходное значение для каждого возможного вектора входов. Множество входных векторов, при которых нейрон активен ($Y_j=1$), отделено от множества векторов, на которых нейрон пассивен ($Y_j=0$), гиперплоскостью, уравнение которой $\sum_{i=1}^{n}x_i\cdot w_i=\Theta$.

Следовательно, нейрон способен отделить только такие два множества векторов входов, для которых имеется гиперплоскость, отсекающая одно множество от другого. Такие множества называют линейно разделимыми.

Рассмотрим однослойный персептрон, состоящий из одного нейрона с двумя входами. Входной вектор содержит только две булевы компоненты x_1

и x_2 , определяющие плоскость. На данной плоскости возможные значения входных векторов отвечают вершинам единичного квадрата. В каждой вершине зададим требуемое значение выхода нейрона: 1 (на рис. 8 — белая точка) или 0 (черная точка). Требуется определить, существует ли такой набор весов и порогов нейрона, при котором нейрон сможет получить эти значения выходов? На рис. 8 представлена одна из ситуаций, когда этого сделать нельзя вследствие линейной неразделимости множеств белых и черных точек.

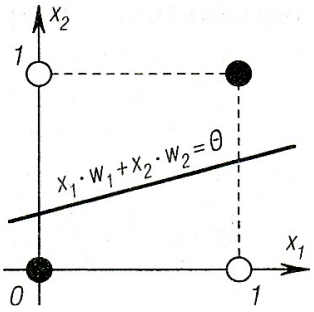

Рис. 8. Белые точки не могут быть отделены одной прямой от черных

Требуемые выходы нейрона для этого рисунка определяются табл. 2, в которой легко узнать задание логической функции «исключающее ИЛИ» (XOR).

Невозможность реализации однослойным персептроном этой функции получила название проблемы исключающего ИЛИ. Видно, что однослойный персептрон крайне ограничен в своих возможностях точно представить наперед заданную логическую функцию.

Таблица 6.2

Логическая функция XOR

X_1	X_2	Y
0	0	0
1	0	1
0	1	1
1	1	0

Хотя данный пример нагляден, он не является серьезным ограничением для нейросетей. Функция XOR легко реализуется простейшей двухслойной сетью, причем многими способами. Один из примеров такой сети показан на рис. 9. Весовые коэффициенты первого слоя все равны единице, весовые

коэффициенты второго слоя v_1, v_2 соответственно равны 1 и -1, пороговые значения каждого нейрона указаны на рисунке. Значения на входах и выходах нейронов сети приведены в табл. 3: $s_1 = x_1 \cdot w_{11} + x_2 \cdot w_{21}$ - значение, поступающее на вход первого нейрона первого слоя, s_2 — вход второго нейрона первого слоя; y_1, y_2 - выходы соответствующих нейронов первого слоя; S — входное значение нейрона второго слоя; Y - выход сети.

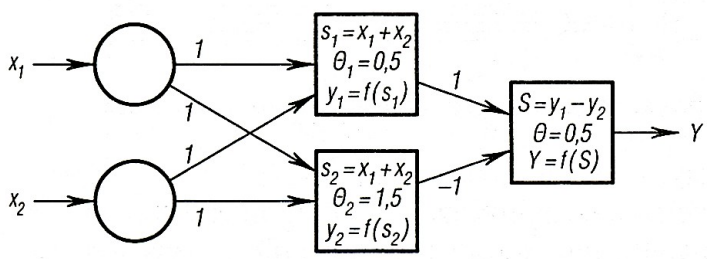

Рис. 9. Двухслойная сеть, реализующая функцию XOR

Таблица 3

Входы и выходы нейронов сети, реализующей функцию XOR

Искусственные нейронные сети. Персептроны

Биологический нейрон. Искусственный нейрон. Активационная функция. Классификация и свойства искусственных нейронных сетей (ИНС). Обучение ИНС. Теорема Колмогорова. Алгоритм обучения персептрона. Линейная разделимость и персептронная представляемость.

Биологический нейрон.

Нервная система и мозг человека состоят из нейронов, соединенных между собой нервными волокнами. Нервные волокна способны передавать электрические импульсы между нейронами. Все процессы перехода раздражений от кожи, ушей и глаз к мозгу, процессы мышления и управления действиями реализуются в живом организме как передача электрических импульсов между нейронами. Нейрон (нервная клетка) является особой биологической клеткой, которая обрабатывает информацию (рис.1).

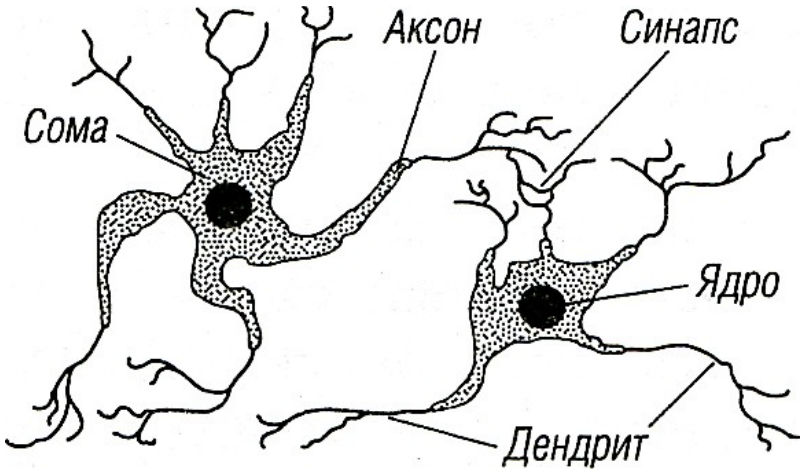

Рис. 1. Взаимосвязь биологических нейронов

Он состоит из тела, или сомы, и отростков нервных волокон двух типов: дендритов, по которым принимаются импульсы, и единственного аксона, по которому нейрон может передавать импульс. Тело нейрона включает ядро, которое содержит информацию о наследственных свойствах, и плазму, обладающую молекулярными средствами для производства необходимых нейрону материалов. Нейрон получает сигналы (импульсы) от аксонов других нейронов через дендриты (приемники) и передает сигналы, сгенерированные телом клетки, вдоль своего аксона (передатчика), который в конце разветвляется на волокна. На окончаниях этих волокон находятся специальные образования - синапсы, которые влияют на величину импульсов.

Кора головного мозга человека содержит около 10^{11} нейронов. Каждый нейрон связан с 10^{3}-10^{4} другими нейронами. В целом мозг человека содержит приблизительно от 10^{14} до 10^{15} взаимосвязей.

85 000 000 000 нейронов в организме человека.

Искусственный нейрон.

Каждый искусственный нейрон характеризуется своим текущим состоянием по аналогии с нервными клетками головного мозга, которые могут быть возбуждены или заторможены. Искусственный нейрон обладает группой синапсов — однонаправленных входных связей, соединенных с выходами других нейронов, а также имеет аксон — выходную связь данного нейрона, с которой сигнал (возбуждения или торможения) поступает на синапсы следующих нейронов. Общий вид искусственного нейрона приведен на рис. 2.

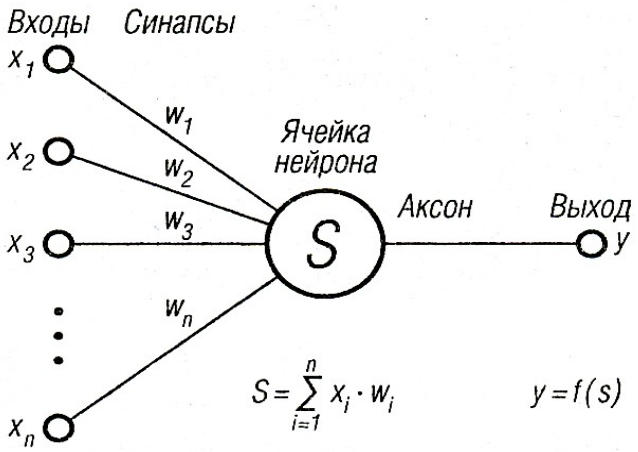

Рис. 2. Модель искусственного нейрона

Искусственный нейрон в первом приближении имитирует свойства биологического нейрона. Здесь множество входных сигналов, обозначенных $x_1, x_2, \dots x_n$, поступает на искусственный нейрон. Эти входные сигналы, в совокупности обозначаемые вектором X, соответствуют сигналам, приходящим в синапсы биологического нейрона. Каждый синапс характеризуется величиной **синапсической связи** или ее **весом** w_i. Каждый сигнал умножается на соответствующий вес $w_1, w_2, \dots, w_n$, и поступает на суммирующий блок. Каждый вес соответствует «силе» одной биологической синапсической связи. (Множество весов в совокупности обозначаются вектором W) Суммирующий блок, соответствующий телу биологического элемента, складывает взвешенные входы алгебраически, создавая величину S.

Таким образом, текущее состояние нейрона определяется как как взвешенная сумма его входов:

$$S=\sum_{i-1}^{n} x_i \cdot w_i.$$

Выход нейрона есть функция его состояния: $y=f\{S\}$, где f - активационная функция, более точно моделирующая нелинейную передаточную характеристику биологического нейрона и предоставляющая нейронной сети большие возможности. Примеры некоторых активационных функций представлены в табл. 1 и на рис. 3. Наиболее распространенными являются пороговая и сигмоидальная активационные функции.

Таблица 1.

Функции активации нейронов

Название	Формула	Область значений
Пороговая (функция единичного скачка)	$f(s)=\begin{cases}0, & s<\Theta;\\ 1, & s\geqslant\Theta\end{cases}$	$\{0, 1\}$
Линейная	$f(s)=ks$	$(-\infty, +\infty)$
Логистическая (сигмоидальная)	$f(s)=\frac{1}{1+e^{-as}}$	$(0, 1)$
Гиперболический тангенс	$f(s)=\frac{e^{as}-e^{-as}}{e^{as}+e^{-as}}$	$(-1, 1)$
Линейная с насыщением (линейный порог)	$f(s)=\begin{cases}0, & s<0;\\ s/\Theta, & 0\leqslant s<\Theta\\ 1, & s\geqslant\Theta\end{cases}$	$[0, 1]$

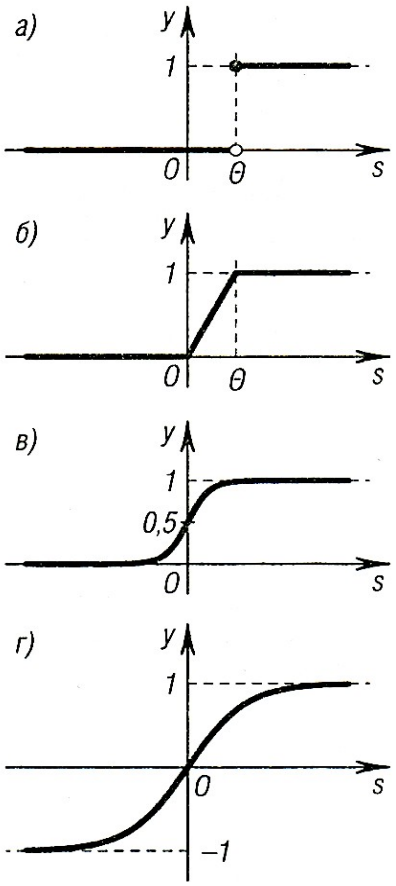

Рис. 3. Примеры активационных функций: а) функция единичного скачка; б) линейный порог; в) логистическая функция; г) гиперболический тангенс

Пороговая функция ограничивает активность нейрона значениями 0 или 1 в зависимости от величины комбинированного входа s . Как правило, входные значения в этом случае также используются бинарные: $x_i \in \{0,1\}$.

Чаще всего удобнее вычесть пороговое значение θ, называемое смещением, из величины комбинированного входа и рассмотреть пороговую функцию в математически эквивалентной форме:

$$s = w_0 + \sum_{i=1}^{n} x_i \cdot w_i, f(s) = \begin{cases} 0, s < 0 \\ 1, s \geq 0 \end{cases}.$$

Здесь $w_0 = -\theta$ - **величина смещения**, взятая с противоположным знаком. Смещение обычно интерпретируется как связь, исходящая от элемента, значение которого всегда равно 1. Комбинированный вход тогда можно представить в виде $s = \sum_{i=0}^{n} x_i \cdot w_i$, где x_0 всегда считается равным 1.

Логистическая функция, или сигмоид, $f(s) = \frac{1}{1 + e^{-a \cdot s}}$ непрерывно заполняет своими значениями диапазон от 0 до 1; параметр a всегда положителен. При уменьшении параметра a график сигмоида становится более пологим, в пределе при a =0 вырождаясь в горизонтальную линию на уровне 0,5, при увеличении параметра a график сигмоида приближается к виду функции единичного скачка с порогом 0. **Следует отметить**, что сигмоидальная функция дифференцируема на всей оси абсцисс, что используется в некоторых алгоритмах обучения. Кроме того, она обладает **свойством усиливать слабые сигналы и предотвращает насыщение от больших сигналов, так как они соответствуют тем областям аргументов, где сигмоид имеет пологий наклон.**

Классификация и свойства искусственных нейронных сетей (ИНС). Однослойные ИНС.

Хотя один нейрон и способен выполнять простейшие процедуры распознавания, сила нейронных вычислений в соединениях нейронов. Простейшая сеть состоит из группы нейронов, образующих слой рис. 4.

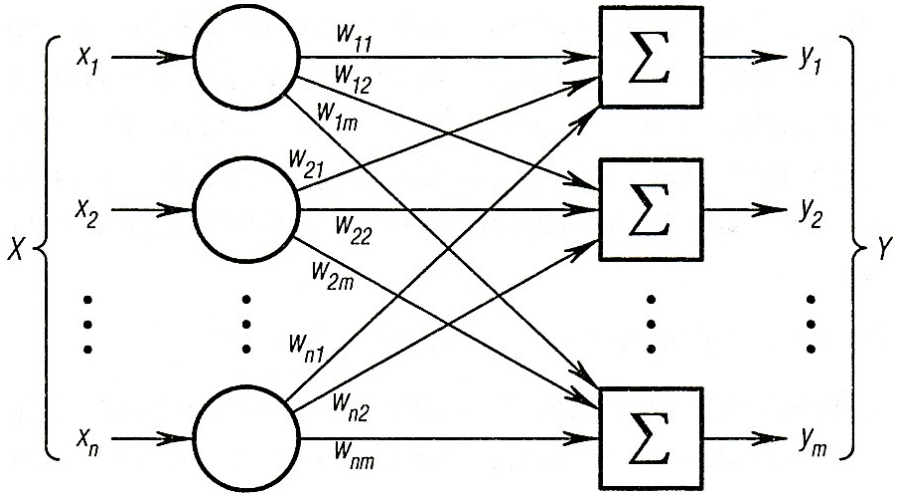

Рис. 4. Простейшая однослойная нейронная сеть

Отметим, что вершины-круги - распределение входных сигналов. Они не выполняют каких-либо вычислений и не считаются слоем. Вычисляющие нейроны обозначены квадратами. Каждый элемент из множества входов X соединен с каждым искусственным нейроном отдельной связью, которой приписан вес. А каждый нейрон выдает взвешенную сумму входов в сеть. В **искусственных и биологических сетях многие соединения могут отсутствовать**, на рисунке все соединения показаны в целях общности. Могут иметь место также соединения между выходами и входами элементов в слое. Удобно считать веса элементами матрицы W. Матрица имеет n строк и m столбцов, где $n-¿$ число входов, а $m-¿$ число нейронов. Таким образом, вычисление выходного вектора Y, компонентами которого являются выходы y_i нейронов, сводится к матричному умножению $Y = X \cdot W$.

Многослойные ИНС.

Более крупные и сложные нейронные сети обладают, как правило, и большими вычислительными возможностями. Хотя в литературе иногда рассматриваются полносвязные нейронные сети (где каждый нейрон передает свой выходной сигнал остальным нейронам, включая самого себя), именно послойная организация нейронов (где каждый нейрон передает свой выходной сигнал только нейронам из соседнего слоя) копирует слоистые структуры определенных отделов мозга. Оказалось, что такие многослойные сети обладают большими возможностями, чем однослойные, и в последние годы были разработаны многообразные алгоритмы для их обучения.

Многослойные сети могут образовываться каскадами слоев. Выход одного слоя является входом для последующего слоя. Подобная сеть показана на рис. 5 и изображена со всеми соединениями.

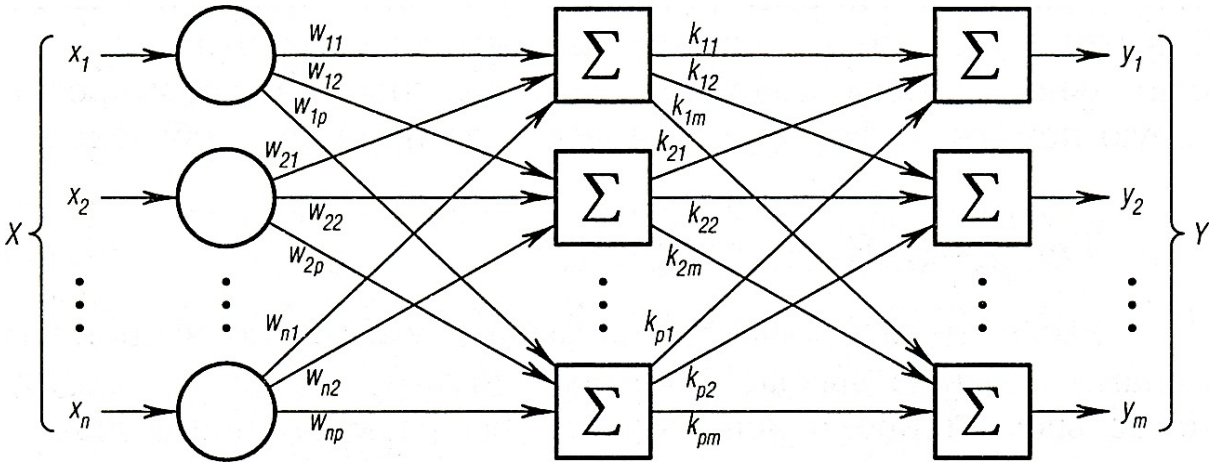

Рис. 5. Многослойная нейронная сеть

Многослойные сети могут привести к увеличению вычислительной мощности по сравнению с однослойной сетью лишь в том случае, если активационная функция между слоями **нелинейна**. Вычисление выхода слоя заключается в умножении входного вектора на первую весовую матрицу с последующим умножением (**если отсутствует нелинейная активационная функция**) результирующего вектора на вторую весовую матрицу. Так как умножение матриц **ассоциативно**, то двухслойная линейная сеть эквивалентна одному слою с весовой матрицей, равной произведению двух весовых матриц. **Следовательно, любая многослойная линейная сеть может быть заменена эквивалентной однослойной сетью**.

Обучение ИНС.

Сеть обучается, чтобы для некоторого множества входов давать требуемое (или, по крайней мере, сообразное с ним) множество выходов. Каждое такое входное (или выходное) множество рассматривается как вектор. Обучение осуществляется путем последовательного предъявления входных векторов с одновременной подстройкой весов в **соответствии с определенной процедурой**. В процессе обучения веса сети постепенно становятся такими, чтобы каждый входной вектор вырабатывал выходной вектор. **Различают алгоритмы обучения с учителем и без учителя.**

Обучение с учителем предполагает, что для каждого входного вектора существует целевой вектор, представляющий собой требуемый выход. Вместе они называются обучающей парой. Обычно сеть обучается на некотором числе таких обучающих пар. Предъявляется выходной вектор, вычисляется выход сети и сравнивается с соответствующим целевым вектором, разность (ошибка) с помощью обратной связи подается в сеть, и веса изменяются в соответствии с алгоритмом, стремящимся минимизировать ошибку. Векторы обучающего множества предъявляются последовательно, для каждого вектора вычисляются ошибки и подстраиваются веса до тех пор, пока ошибка по всему обучающему массиву не достигнет приемлемо низкого уровня.

Обучение без учителя не нуждается в целевом векторе для выходов и, следовательно, не требует сравнения с предопределенными идеальными ответами. **Обучающее множество состоит лишь из входных векторов**. Обучающий алгоритм подстраивает веса сети так, чтобы получались согласованные выходные векторы, т. е. чтобы предъявление достаточно близких входных векторов давало одинаковые выходы. **Процесс обучения выделяет статистические свойства обучающего множества и группирует сходные векторы в классы**. Предъявление на вход вектора из данного класса даст определенный выходной вектор, но до обучения невозможно предсказать, какой выход будет активироваться данным классом входных векторов. Следовательно, выходы подобной сети должны трансформироваться в некоторую понятную форму, обусловленную процессом обучения.

Теорема Колмогорова.

Рассмотрим двухслойную нейронную сеть с п входами и одним выходом, которая проста по структуре и широко используется для решения прикладных задач рис. 6.

Каждый -й нейрон первого слоя ($i=1,2,\dots,m$) имеет n входов, которым приписаны веса $w_{1i}, w_{2i}, \dots, w_{ni}$.

Получив входные сигналы, нейрон суммирует их с соответствующими весами, затем применяет к этой сумме передаточную функцию и пересылает результат на вход нейрона второго (выходного) слоя. В свою очередь, нейрон выходного слоя суммирует полученные от второго слоя сигналы с некоторыми

весами v_i. Для определенности будем предполагать, что передаточные функции в скрытом слое являются **сигмоидальными**, а в выходном слое используется **тождественная** функция, т. е. взвешенная сумма выходов второго слоя и будет ответом сети.

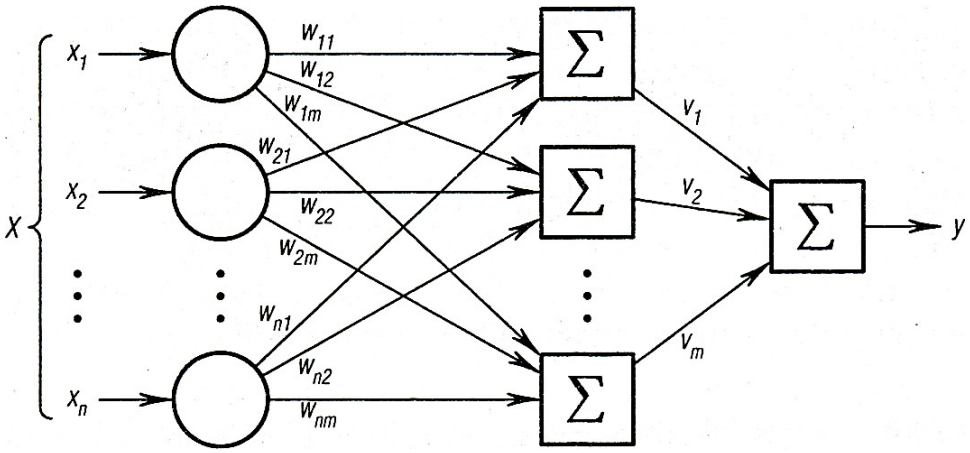

Рис. 6. Пример ИНС

Подавая на входы любые числа $x_1, x_2, \ldots, x_n$, мы получим на выходе значение некоторой функции $y = F(x_1, x_2, \ldots, x_n)$, которое является ответом (реакцией) сети. Ответ сети зависит как от **входного сигнала**, так и от **значений ее внутренних параметров** - весов нейронов. Точный вид этой функции:

$$F(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{m} ¿ \; v_i \sigma ¿ \sum_{j=0}^{n} ¿ \; x_j w_{ji} ¿,$$

где $\sigma(s) = \dfrac{1}{1+e^{-a \cdot s}}$

В 1957 году математик А. Н. Колмогоров доказал следующую теорему.

Теорема Колмогорова. Любая непрерывная функция F , определенная на n -мерном единичном кубе, может быть представлена в виде суммы $2n+1$ суперпозиций непрерывных и монотонных отображениий единичных отрезков:

$$h_{ij}(x_j)$$
$$\sum_{j=1}^{n} ¿$$
$$¿$$
$$g_i ¿$$
$$F(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{2n+1} ¿$$
$$x$$
$$x = (¿¿1, x_2, \ldots, x_n), 0 \le x_i \le 1,$$
$$¿$$

где g_i и h_{ij} - непрерывные функции, причем h_{ij} не зависят от функции F .

Эта теорема означает, что **для реализации функций многих переменных достаточно операций суммирования и композиции функций одной переменной**. К сожалению, при всей своей математической красоте, теорема Колмогорова малоприменима на практике. Это связано с тем, что функции $h_{ij} - ¿$ в общем случае негладкие и трудновычислимые; также неясно, каким образом можно подбирать функции g_i для данной функции F . **Роль этой теоремы состоит в том, что она показала принципиальную возможность реализации сколь угодно сложных зависимостей с помощью относительно простых автоматов типа нейронных сетей.** Более значимые для практики результаты в этом направлении были получены только в 1989 году, зато одновременно несколькими исследователями.

Пусть $F(x_1, x_2, \ldots, x_n)$ - любая непрерывная функция, определенная на ограниченном множестве, и $\varepsilon > 0$ - любое сколь угодно малое число, означающее точность аппроксимации. Через σ обозначена сигмоидальная функция.

Теорема. Существуют число m , набор чисел w_{ji} и набор чисел v_i такие, что функция

$$x_j w_{ji}$$
$$\sum_{j=0}^{n} ¿,$$
$$¿$$
$$v_i \sigma ¿$$
$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{m} ¿$$

приближает данную функцию $F(x_1, x_2, \ldots, x_n)$ с погрешностью не более ε на всей области определения.

Легко заметить, что эта формула полностью совпадает с выражением, полученным для функции, реализуемой нейросетью. **В терминах теории ИНС эта теорема формулируется так: любую непрерывную функцию нескольких переменных можно с любой точностью реализовать с помощью двухслойной нейросети с достаточным количеством нейронов в скрытом слое.**

Алгоритм обучения персептрона

Одной из первых искусственных сетей, способных к перцепции (восприятию) и формированию реакции на воспринятый стимул, явился PERCEPTRON Розенблатта (F. Rosenblatt, 1958). **Персептроном**, как правило, называют однослойную нейронную сеть, при этом каждый персептронный нейрон в качестве активационной функции использует функцию единичного скачка (пороговую).

Для простоты рассмотрим вначале процедуру обучения персептрона, состоящего только из одного нейрона. Подробная схема такого персептрона изображена на рис. 7.

Как отмечалось ранее, будем считать, что персептрон имеет дополнительный вход x_0 , который всегда равен 1. В таком случае, пороговое смещение $\theta=0$ и $y=f(s)=\{0, s<0 1, s\geq 0.$

Обучение персептрона состоит в подстройке весовых коэффициентов w_i , где $i=1,2,...,n$. Обученный персептрон способен разделять требуемое множество образов на два

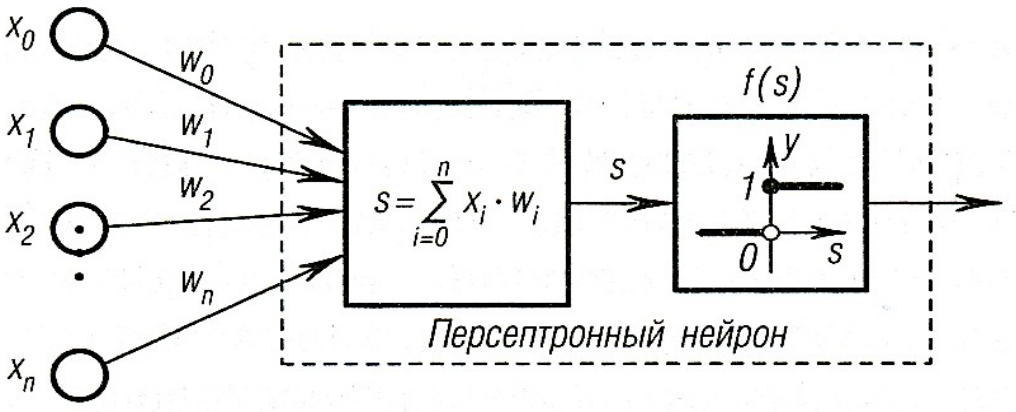

Рис. 7. Однонейронный персептрон с n входами

класса. (К первому классу относятся входные образы, для которых на выходе персептрона получено нулевое значение, ко второму классу - образы, для которых получено единичное значение.)

Обучение персептрона - это обучение с учителем, т. е. должен существовать набор векторов $(X^k, y_k), k=1,2,\dots,p$, называемый обучающей выборкой. Здесь $X^k=(x_0^k, x_1^k, x_2^k, \dots, x_n^k), k=1,2,\dots,p$ - примеры входных образов, для которых заранее известна их принадлежность к одному из двух данных классов.

Будем называть персептрон обученным на данной обучающей выборке, если при подаче на вход каждого вектора X^k на выходе всякий раз получается соответствующее значение $y_k \in \{0,1\}$. Предложенный Ф. Розенблаттом метод обучения состоит в итерационной подстройке весовых коэффициентов w_i последовательно уменьшающей выходные ошибки.

Алгоритм включает несколько шагов:

Шаг 0. Проинициализировать весовые коэффициенты $w_i, i=1,2,\dots,n$ небольшими случайными значениями, например, из диапазона [-0,3; 0,3].

Шаг 1. Подать на вход персептрона один из обучающих векторов X^k и вычислить его выход y .

Шаг 2. Если выход правильный ($y=y_k$), перейти на шаг 4. Иначе вычислить ошибку — разницу между верным и полученным значениями выхода: $\delta=y_k-y$.

Шаг 3. Весовые коэффициенты модифицируются по следующей формуле:

$$w_{ij}^{t+1}=w_{ij}^t+v\cdot\delta\cdot x_i.$$

Здесь t и $t+1$ - номера соответственно текущей и следующей итераций; v - коэффициент скорости обучения ($0<v\le 1$); x_i. - i -я компонента входного вектора X^k .

Шаг 4. Шаги 1—3 повторяются для всех обучающих векторов. **Один цикл последовательного предъявления всей выборки называется эпохой**. Обучение завершается по истечении нескольких эпох, когда сеть перестанет ошибаться.

Замечание 1. Коэффициент скорости обучения v является параметром данного алгоритма. Как правило, его выбирают из диапазона [0,5; 0,7]. В некоторых случаях (при большом объеме обучающей выборки) целесообразно постепенно уменьшать значение v начиная, например, с 1.

Замечание 2. Используемая на шаге 3 формула модифицирует только весовые коэффициенты, отвечающие ненулевым значениям входов x_i^k, поскольку только они влияли на величину $s=\sum_{i=1}^{n} x_i \cdot w_i$, а следовательно, и на значение y.

Очевидно, что если $y_k > y$ (получен неправильный нулевой выход вместо правильного единичного), то, поскольку $\delta > 0$, весовые коэффициенты (а вместе с ними и величина s) будут увеличены и тем самым уменьшат ошибку. В противном случае весовые коэффициенты будут уменьшены и сумма s тоже уменьшится, приближая тем самым y к значению y_k.

Обобщим теперь этот алгоритм обучения на случай однослойной сети, включающей n персептронных нейронов (рис. 4). Такая сеть (при достаточно большом числе нейронов) может осуществлять разбиение образов на произвольное требуемое число классов.

Пусть имеется обучающая выборка, состоящая из пар векторов (X^k, Y^k), $k=1,2,\ldots,p$. Назовем нейронную сеть обученной на данной обучающей выборке, если при подаче на входы сети каждого вектора X^k на выходах всякий раз получается соответствующий вектор Y^k. Обучение заключается в итерационной подстройке матрицы весов W (w_{ij} — вес синаптической связи между i-м входом и j-м нейроном), последовательно уменьшающей ошибку в выходных векторах. Алгоритм включает следующие шаги:

Шаг 0. Проинициализировать элементы весовой матрицы W небольшими случайными значениями.

Шаг 1. Подать на входы один из входных векторов X^k и вычислить его выход Y.

Шаг 2. Если выход правильный $Y=Y$ $(¿¿k)$ $¿$, перейти на шаг 4. Иначе, вычислить вектор ошибки - разницу между идеальным и полученным значениями выхода: $\delta=Y^k-Y$.

Шаг 3. Матрица весов модифицируется по следующей формуле: $w_{ij}^{t+1}=w_{ij}^t+v\cdot\delta\cdot x_i$. Здесь t и $t+1$ - номера соответственно текущей и следующей итераций; v - коэффициент скорости обучения ($0<v\le 1$); x_i. - i -я компонента входного вектора X^k ; j - номер нейрона в слое.

Шаг 4. Шаги 1—3 повторяются для всех обучающих векторов. Обучение завершается, когда сеть перестанет ошибаться.

Представленный метод обучения носит название δ - правило. Доказанная Розенблаттом теорема о сходимости обучения по δ - правилу говорит о том, что персептрон способен обучиться любому обучающему набору, который он способен представить.

Линейная разделимость и персептронная представляемость.

Каждый нейрон персептрона является формальным пороговым элементом, принимающим единичные значения в случае, если суммарный взвешенный вход больше некоторого порогового значения:

$$Y_j=\{0,\sum_{i=1}^{n}x_i\cdot w_i<\Theta\ 1,\sum_{i=1}^{n}x_i\cdot w_i\ge\Theta.$$

Таким образом, при заданных значениях весов и порогов, нейрон имеет определенное выходное значение для каждого возможного вектора входов. Множество входных векторов, при которых нейрон активен ($Y_j=1$), отделено от множества векторов, на которых нейрон пассивен ($Y_j=0$), гиперплоскостью, уравнение которой $\sum_{i=1}^{n}x_i\cdot w_i=\Theta$.

Следовательно, нейрон способен отделить только такие два множества векторов входов, для которых имеется гиперплоскость, отсекающая одно множество от другого. Такие множества называют линейно разделимыми.

Рассмотрим однослойный персептрон, состоящий из одного нейрона с двумя входами. Входной вектор содержит только две булевы компоненты x_1

и x_2 , определяющие плоскость. На данной плоскости возможные значения входных векторов отвечают вершинам единичного квадрата. В каждой вершине зададим требуемое значение выхода нейрона: 1 (на рис. 8 — белая точка) или 0 (черная точка). Требуется определить, существует ли такой набор весов и порогов нейрона, при котором нейрон сможет получить эти значения выходов? На рис. 8 представлена одна из ситуаций, когда этого сделать нельзя вследствие линейной неразделимости множеств белых и черных точек.

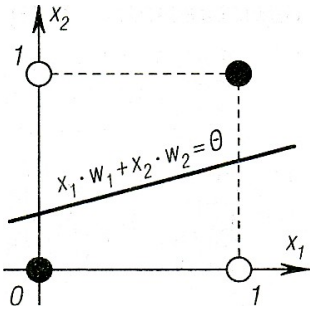

Рис. 8. Белые точки не могут быть отделены одной прямой от черных

Требуемые выходы нейрона для этого рисунка определяются табл. 2, в которой легко узнать задание логической функции «исключающее ИЛИ» (XOR).

Невозможность реализации однослойным персептроном этой функции получила название проблемы исключающего ИЛИ. Видно, что однослойный персептрон крайне ограничен в своих возможностях точно представить наперед заданную логическую функцию.

Таблица 6.2

Логическая функция XOR

X_1	X_2	Y
0	0	0
1	0	1
0	1	1
1	1	0

Хотя данный пример нагляден, он не является серьезным ограничением для нейросетей. Функция XOR легко реализуется простейшей двухслойной сетью, причем многими способами. Один из примеров такой сети показан на рис. 9. Весовые коэффициенты первого слоя все равны единице, весовые

коэффициенты второго слоя v_1, v_2 соответственно равны 1 и -1, пороговые значения каждого нейрона указаны на рисунке. Значения на входах и выходах нейронов сети приведены в табл. 3: $s_1 = x_1 \cdot w_{11} + x_2 \cdot w_{21}$ - значение, поступающее на вход первого нейрона первого слоя, s_2 — вход второго нейрона первого слоя; y_1, y_2 - выходы соответствующих нейронов первого слоя; S — входное значение нейрона второго слоя; Y - выход сети.

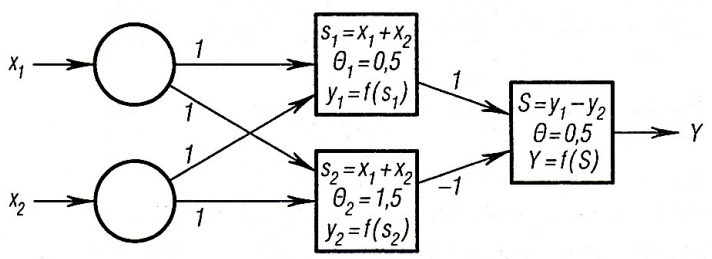

Рис. 9. Двухслойная сеть, реализующая функцию XOR

Таблица 3

Входы и выходы нейронов сети, реализующей функцию XOR

x_1	x_2	s_1	s_2	y_1	y_2	S	Y
0	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
0	1	1	1	1	0	1	1
1	1	2	2	1	1	0	0

Лекция 14. Искусственные нейронные сети. Нейронные сети обратного распространения

Нейронная сеть обратного распространения. Алгоритм обучения сети обратного распространения. Сеть встречного распространения (сеть Кохонена). Алгоритм обучения сети Кохонена. Входные и выходные звезды и их обучение. Входные и выходные звезды Гроссберга и их обучение. Двухслойная сеть встречного распространения. Алгоритм обучения сети встречного распространения.

Рассмотренный в предыдущем параграфе алгоритм обучения однослойного персептрона очень прост. Однако долгие годы не удавалось обобщить этот алгоритм на случай многослойных сетей, что спровоцировало в научных кругах значительный спад интереса к нейронным сетям. Только в 1986

году Румельхарт (D. E. Rumelhart) разработал эффективный алгоритм корректировки весов, названный **алгоритмом обратного распространения ошибок** (back propagation).

Нейронные сети обратного распространения - это современный инструмент поиска закономерностей, прогнозирования, качественного анализа. Такое название - **сети обратного распространения** - они получили из-за используемого алгоритма обучения, в котором ошибка распространяется от выходного слоя к входному, т.е. в направлении, противоположном направлению распространения сигнала при нормальном функционировании сети.

Нейронная сеть обратного распространения состоит из нескольких слоев нейронов, причем каждый нейрон предыдущего слоя связан с каждым нейроном последующего слоя. В большинстве практических приложений оказывается достаточно рассмотрения двухслойной нейронной сети, имеющей входной (скрытый) слой нейронов и выходной слой (рис. 1).

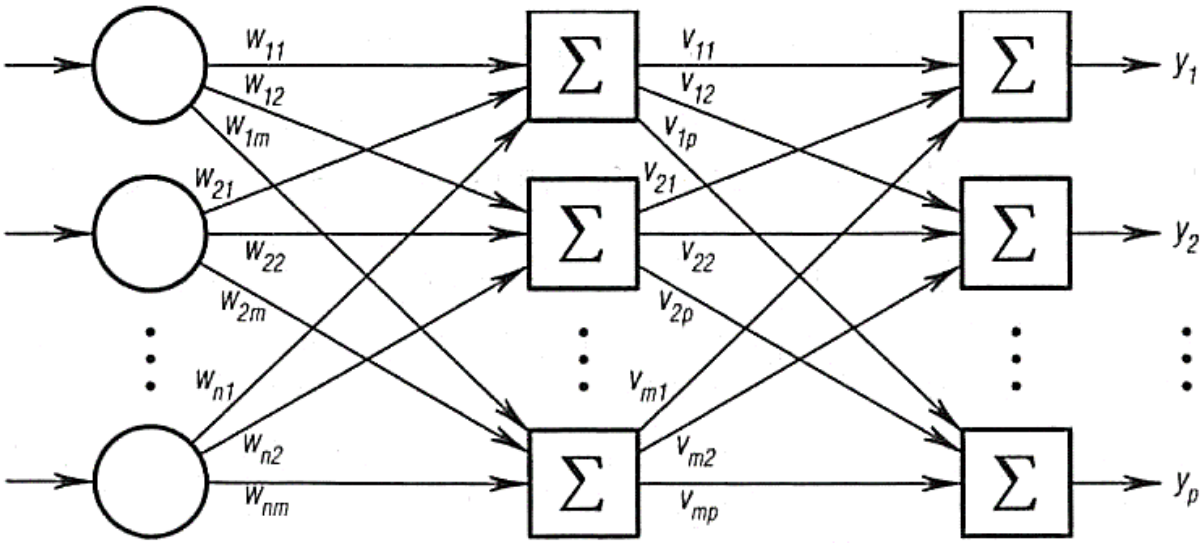

Рис. 1. Нейронная сеть обратного распространения

Матрица весовых коэффициентов от входов к скрытому слою - W , а матрица весов, соединяющих скрытый и выходной слой - V . Обозначения для индексов: входы индекс i , элементы скрытого слоя - y , а выходы - k . Число входов сети - n , число нейронов в скрытом слое - n , число

нейронов в выходном слое - p . Пусть сеть обучается на выборке $(X^t, Y^t), t = 1,2,\ldots,T$.

При **обучении нейронной сети ставится задача минимизации целевой функции ошибки**, которая находится по методу наименьших квадратов:

$$E(W,V) = \frac{1}{2}\sum_{k=1}^{p}(y_k - d_k)^2,$$

где y_k — полученное реальное значение k *-го* выхода нейросети при подаче на нее одного из входных образов обучающей выборки; d_k – требуемое значение k -го выхода для этого образца.

Обучение нейросети производится известным **оптимизационным методом градиентного спуска**, т. е. на каждой итерации производится следующее изменение веса

$$w_{ij}^{N+1} = w_{ij}^{N} - \alpha\frac{\partial E}{\partial w_{ij}}, v_{jk}^{N+1} = v_{jk}^{N} - \alpha\frac{\partial E}{\partial v_{jk}}$$

где α - параметр, определяющий скорость обучения.

В качестве активационной функции в сети обратного распространения обычно используется сигмоидальная функция

$$f(s) = \frac{1}{1+e^{-x}} \quad , ,$$

где s - взвешенная сумма входов нейрона. Эта функция удобна для вычислений в градиентном методе, так как имеет простую производную:

$$f'(s) = \frac{e^{-x}}{(1+e^{-x})^2} = f(s)(1-f(s)).$$

Функция ошибки в явном виде не содержит зависимости от весовых коэффициентов w_{ij} и v_{jk} .

После некоторых упрощений, вычислений и введения обозначений

$$\delta_k = \frac{\partial E}{\partial y_k}\cdot\frac{\partial y_k}{\partial s_k} = (y_k - d_k)\,y_k(1-y_k),$$

Получим следующие выражения для производных:

$$\delta_k v_{jk}$$

$$\sum_{k=1}^{p} ¿\, y_j^c(1-y_j^c)x_i.$$

$$\frac{\partial E}{\partial v_{jk}} = \delta_k y_j^c, \frac{\partial E}{\partial w_{ij}} = ¿$$

Алгоритм обучения сети обратного распространения

Рассмотрим полный алгоритм обучения нейросети:

Шаг 1. Инициализация сети.

Весовым коэффициентам присваиваются малые случайные значения, например, из диапазона (-0,3; 0,3); задаются ε - параметр точности обучения, α - параметр скорости обучения (как правило, $\alpha \approx 0{,}1$ и может еще уменьшаться в процессе обучения), N_{max} - максимально допустимое число итераций.

Шаг 2. Вычисление текущего выходного сигнала.

На вход сети подается один из образов обучающей выборки и определяются значения выходов всех нейронов нейросети.

Шаг 3. Настройка синаптических весов.

Рассчитать изменение весов для выходного слоя нейронной сети по формулам

$$v_{jk}^{N+1} = v_{jk}^{N} - \alpha \frac{\partial E}{\partial v_{jk}}, \text{где} \frac{\partial E}{\partial v_{jk}} = \delta_k y_j^c, \delta_k = (y_k - d_k) y_k (1 - y_k),$$

Рассчитать изменение весов для скрытого слоя по формулам

$$\delta_k v_{jk}$$

$$\sum_{k=1}^{p} ¿ y_j^c (1 - y_j^c) x_i .$$

$$w_{ij}^{N+1} = w_{ij}^{N} - \alpha \frac{\partial E}{\partial w_{ij}}, \text{где} \frac{\partial E}{\partial w_{ij}} = ¿$$

Шаг 4. Шаги 2-3 повторяются для всех обучающих векторов. Обучение завершается по достижении для каждого из обучающих образов значения функции ошибки, не превосходящего ε или после максимально допустимого числа итераций - N_{max} .

Замечание 1. На шаге 2 векторы из обучающей последовательности лучше предъявлять на вход в случайном порядке.

Замечание 2. Желательно наделять каждый нейрон обучаемым смещением. Это позволяет сдвигать начало отсчета логистической функции, с эффектом, аналогичном подстройке порога персептронного нейрона, и приводит к **ускорению процесса обучения**. Эта возможность может быть легко введена в обучающий алгоритм с помощью добавления к каждому нейрону дополнительного входа. Его вес обучается так же, как и все остальные

веса, за исключением того, что подаваемый на него сигнал всегда равен +1, а не выходу нейрона предыдущего слоя.

Замечание 3. Количество входов и выходов сети, как правило, диктуется условиями задачи, а размер скрытого слоя находят экспериментально. Обычно число нейронов в нем составляет 30—50% от числа входов. Слишком большое количество нейронов скрытого слоя приводит к тому, что сеть теряет способность к обобщению (она просто досконально запоминает элементы обучающей выборки и не реагирует на схожие образцы, что неприемлемо для задач распознавания). Если же число нейронов в скрытом слое слишком мало, сеть оказывается просто не в состоянии обучиться.

Замечание 4. Выходы каждого нейрона сети лежат в диапазоне (0, 1) - области значений логистической функции - это надо учитывать при формировании обучающей выборки. Если необходимо получить от сети бинарный выход, то, как правило, вместо 0 используют 0,1, а вместо 1 - 0,9, так как границы интервала недостижимы.

Модификации алгоритма обратного распространения связаны с использованием различных функций ошибки, других активационных функций, различных процедур определения направления и величины шага.

Несмотря на многочисленные успешные применения обратного распространения, оно не является панацеей. Больше всего неприятностей приносит неопределенно долгий процесс обучения. В сложных задачах для обучения сети могут потребоваться часы или даже дни, она может и вообще не обучиться. Неудачи в обучении часто возникают по причине попадания сети в локальный минимум, что является характерной особенностью методов градиентного спуска. Исправить ситуацию в таком случае иногда помогают небольшие случайные изменения весовых значений сети.

Сеть встречного распространения (сеть Кохонена)

Рассмотрим алгоритм обучения «победитель забирает все» для задачи классификации Кохонена (T. Kohonen). При предъявлении входного вектора возбуждается единственный нейрон, наиболее точно соответствующий этому образу.

Сеть Кохонена. Классификация образов

Задача классификации заключается в разбиении объектов на классы, причем основой разбиения служит вектор параметров объекта. Чаще бывает так, что сами классы заранее неизвестны, и их приходится формировать динамически. Назовем **прототипом** класса объект, наиболее типичный для своего класса. Один из самых простых подходов к классификации состоит в том, чтобы предположить существование определенного числа классов и произвольным образом выбрать координаты прототипов. Затем каждый вектор из набора данных связывается с ближайшим к нему прототипом, и новыми прототипами становятся центроиды всех векторов, связанных с исходным прототипом. В качестве меры близости двух векторов обычно выбирается евклидово расстояние: $d(x,y)=\sqrt{\square}$

На этих принципах основано функционирование сети Кохонена, обычно используемой для решения задач классификации. Данная сеть обучается **без учителя** на основе самоорганизации. По мере обучения векторы весов нейронов становятся прототипами классов - групп векторов обучающей выборки. На этапе решения информационных задач сеть относит новый предъявленный образ к одному из сформированных классов.

Сеть Кохонена состоит из одного слоя нейронов. Число входов каждого нейрона n равно размерности вектора параметров объекта. Количество нейронов *t* совпадает с требуемым числом классов, на которые нужно разбить объекты (меняя число нейронов, можно динамически менять число классов).

Обучение начинается с задания небольших случайных значений элементам весовой матрицы W . В дальнейшем происходит процесс самоорганизации, состоящий в модификации весов при предъявлении на вход векторов обучающей выборки. Каждый столбец весовой матрицы представляет собой параметры соответствующего нейрона-классификатора. Для каждого -го нейрона $(j=1,2,\ldots,m)$ определяется расстояние от него до входного вектора X :

$$d_j=\sum_{i=1}^{n}(x_i-w_{ij})^2.$$

Далее выбирается нейрон с номером $k,1\le k\le m$, для которого это расстояние минимально (т. е. сеть отнесла входной вектор к классу с номером k). На текущем шаге обучения N будут модифицироваться только веса нейронов из окрестности нейрона k:

$$w_{ij}^{N+1}=w_{ij}^{N}+\alpha_N(x_i-w_{ij}^{N}).$$

Первоначально в окрестности любого из нейронов находятся все нейроны сети, но с каждым шагом эта окрестность сужается. В конце этапа обучения подстраиваются только веса нейрона с номером k . Темп обучения α_N с течением времени также уменьшается (часто полагают $\alpha_0=0{,}9, \alpha_{N+1}=\alpha_N-0{,}001.$ Образы обучающей выборки предъявляются последовательно, и каждый раз происходит подстройка весов.

Алгоритм обучения сети Кохонена

Шаг 1. Инициализация сети. Весовым коэффициентам сети $w_{ij}, i=1,2,\ldots,n, j=1,2,\ldots,m$ присваиваются малые случайные значения. Задаются значения α_0 - начальный темп обучения и D_0 — максимальное расстояние между весовыми векторами (столбцами матрицы W).

Шаг 2. Предъявление сети нового входного сигнала X .

Шаг 3. Вычисление расстояния от входа X до всех нейронов сети:

$$d_j=\sum_{i=1}^{n}(x_i-w_{ij})^2, j=1,2,\ldots,m.$$

Шаг 4. Выбор нейрона $k, 1\le k\le m$ с наименьшим расстоянием d_k .

Шаг 5. Настройка весов нейрона k и всех нейронов, находящихся от него на расстоянии, не превосходящем D_N:

$$w_{ij}^{N+1}=w_{ij}^{N}+\alpha_N(x_i-w_{ij}^{N}).$$

Шаг 6. Уменьшение значений $\alpha_N, D_N.$

Шаг 7. Шаги 2—6 повторяются до тех пор, пока веса не перестанут меняться (или пока суммарное изменение всех весов станет очень мало).

После обучения классификация выполняется посредством подачи на вход сети испытуемого вектора, вычисления расстояния от него до каждого нейрона с последующим выбором нейрона с наименьшим расстоянием как индикатора правильной классификации.

Замечание. Если предварительно провести единичную нормировку всех входных векторов, т. е. подавать на вход сети образы X' , компоненты которого связаны с компонентами вектора X по формулам

$$x_i'=\frac{x_i}{\sqrt{\square}}$$

а также если после каждой итерации процесса обучения осуществлять нормировку весов каждого нейрона (столбцов матрицы W), то в качестве меры близости входных векторов и весовых векторов нейронов сети можно рассматривать скалярное произведение между ними. Действительно, в этом случае

$$d_j = 2 - 2\sum_{i=1}^{n} x_i w_{ij}^N .$$

Таким образом, наименьшим будет расстояние до того нейрона, скалярное произведение с весами которого у входного вектора максимально. В этом случае можно считать, что каждый нейрон Кохонена реализует тождественную активационную функцию $f(s) = s$, где

$$s = \sum_{i=1}^{n} w_{ij} x_i .$$

Нейрон с максимальным значением активационной функции объявляется «победителем» и его веса (а также веса нейронов из его окружения) пересчитываются.

Сеть Кохонена нашла широкое применение в задачах финансового анализа. С ее помощью успешно решаются задачи предсказания рисков, нахождения рейтинга и др.

Входные и выходные звезды Гроссберга

Входная звезда Гроссберга (S. Grossberg), как показано на рис. 2, состоит из нейрона, на который подается группа входов, умноженных на синапсические веса.

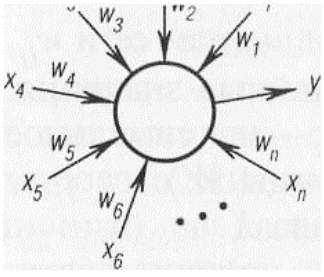

Рис. 2. Входная звезда Гроссберга

Выходная звезда, показанная на рис. 3, является нейроном, управляющим группой весов. Входные и выходные звезды могут быть взаимно соединены в сети любой сложности.

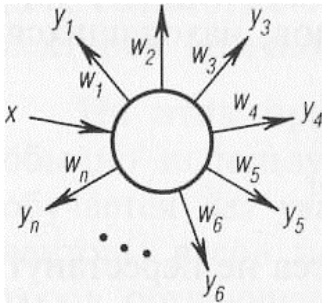

Рис. 3. Выходная звезда Гроссберга

Обучение входной звезды

Входная звезда выполняет распознавание образов, т. е. она обучается реагировать на определенный входной вектор X и ни на какой другой. Это обучение реализуется путем настройки весов таким образом, чтобы они соответствовали входному вектору. Входная звезда имеет тождественную активационную функцию $f(s)=s$, т. е. выход входной звезды определяется как взвешенная сумма ее входов:

$$Y=\sum_{i=1}^{n} w_i x_i .$$

С другой точки зрения, выход можно рассматривать как скалярное произведение входного вектора с весовым вектором. Если эти векторы имеют единичную норму, то скалярное произведение будет максимальным для того входного образа, которому нейрон был обучен.

В процессе обучения веса корректируются следующим образом:

$$w_i^{N+1}=w_i^N+\propto_N\left(x_i-w_i^N\right),$$

где w_i . — весовой коэффициент входа x_i ; $\propto_N$ - нормирующий коэффициент обучения, который имеет начальное значение 0,1 и постепенно уменьшается в процессе обучения.

После завершения обучения предъявление входного вектора X будет активизировать обученный входной нейрон. Хорошо обученная входная звезда будет реагировать не только на определенный запомненный вектор, но также и на незначительные изменения этого вектора. Это достигается постепенной настройкой нейронных весов при предъявлении в процессе обучения векторов,

представляющих нормированные вариации входного вектора. Веса настраиваются таким образом, чтобы усреднить величины обучающих векторов, и нейроны получают способность реагировать на любой вектор этого класса.

Обучение выходной звезды

Если входная звезда учится реагировать на определенный вход, то выходная звезда обучается выдавать требуемый целевой выход.

Чтобы обучить нейрон выходной звезды, его веса настраиваются в соответствии с требуемым целевым вектором Y .

Формула коррекции весов имеет следующий вид:

$$w_i^{N+1} = w_i^N + \beta_N (y_i - w_i^N),$$

где β_N представляет собой нормирующий коэффициент обучения, который вначале приблизительно равен единице и постепенно уменьшается до нуля в процессе обучения.

Как и в случае входной звезды, веса выходной звезды постепенно настраиваются на множество векторов, представляющих собой возможные вариации запоминаемого выходного вектора.

Двухслойная сеть встречного распространения

Сеть встречного распространения состоит из двух слоев: слоя нейронов Кохонена и слоя нейронов Гроссберга. Автор сети Р. Хехт-Нильсен (R. Hecht-Nielsen) объединил эти две архитектуры, в результате чего сеть приобрела свойства, которых не было у каждой из них в отдельности.

Слой Кохонена классифицирует входные векторы в группы схожих. Это достигается с помощью такой подстройки весов слоя Кохонена, что близкие входные векторы активируют один и тот же нейрон данного слоя. Затем слой Гроссберга дает требуемые выходы.

На рис. 4 показана сеть встречного распространения.

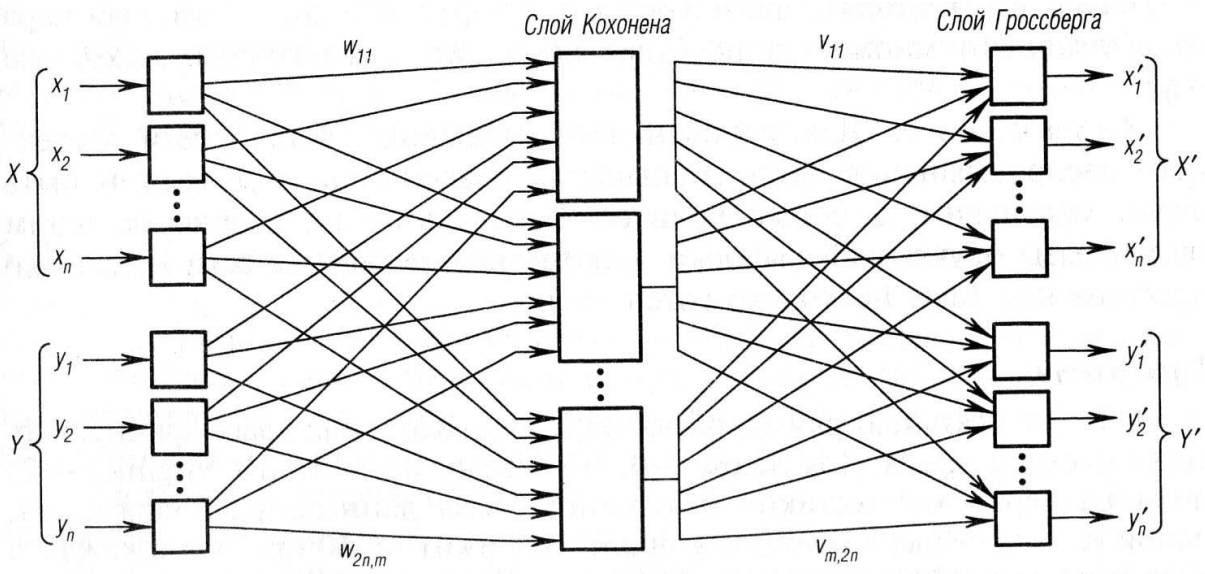

Рис.4. Сеть встречного распространения

В режиме нормального функционирования предъявляются входные векторы X и Y , и обученная сеть дает на выходе векторы X' и Y' , являющиеся аппроксимациями соответственно для X и Y . Векторы X и Y предполагаются здесь нормированными векторами единичной длины, следовательно, порождаемые на выходе векторы также должны быть нормированными.

В процессе обучения векторы X и Y подаются одновременно и как входные векторы сети, и как желаемые выходные сигналы. В результате получается отображение, при котором предъявление пары входных векторов порождает их копии на выходе.

Это не было бы интересно, если не учитывать способность этой сети к обобщению. Благодаря обобщению предъявление только вектора X (с вектором $Y=0$) порождает как выходы X' так и выходы Y' . Если F - функция, отображающая X в Y' , то сеть аппроксимирует ее. Кроме того, если функция F обратима (если функция *y = f(x)* такова, что для любого её значения y_0 уравнение $f(x) = y_0$ имеет относительно *x* единственный корень, то говорят, что функция *f* обратима), то предъявление только вектора Y (при $X=0$) порождает выходы X' .

Уникальная способность порождать функцию и обратную к ней делает сеть встречного распространения полезной в ряде приложений. Например, в

задаче аппроксимации многомерной векторной функции сеть обучается на известных значениях этой функции.

Алгоритм обучения сети встречного распространения

Шаг 1. Произвести единичную нормировку всех векторов X, Y обучающего множества.

Шаг 2. Весовым коэффициентам сети $w_{ij}, v_{ji}, i=1,2,\ldots,2n, j=1,2,\ldots,m$ присвоить малые случайные значения и произвести единичную нормировку матриц W, V по столбцам. Положить $\propto_0=0{,}7, \beta_0=0{,}1.$

Шаг 3. Подать на вход сети обучающий набор *(* X, Y *)* и определить единственный нейрон - «победитель» в слое Кохонена (весовой вектор которого дает максимальное скалярное произведение с входным вектором). Выход этого нейрона установить равным 1, выходы всех остальных нейронов слоя Кохонена положить равными 0. Скорректировать веса выигравшего нейрона:

$$w_{ij}^{N+1}=w_{ij}^{N}+\alpha_N\left(z_i-w_{ij}^{N}\right), \quad \text{где} \quad Z=(X,Y).$$

Шаг 4. Подать выходной вектор слоя Кохонена на вход слоя Гроссберга. Скорректировать веса слоя Гроссберга, связанные с выигравшим нейроном слоя Кохонена:

$v_{ki}^{N+1}=v_{ki}^{N}+\beta_N\left(z_i-v_{ki}^{N}\right)$ (здесь k - номер выигравшего нейрона).

Шаг 5. Уменьшить значения $\propto_0$, β_0.

Шаг 6. Повторять шаги 3—5 до тех пор, пока каждая входная пара из обучающего множества не будет порождать аналогичную выходную пару.

Замечание. Для улучшения обобщающих свойств сети встречного распространения темп уменьшения значений $\propto$ и β должен быть очень маленьким, а общее количество итераций достаточно большим (все образы обучающей выборки желательно предъявить сети несколько десятков или даже несколько сотен раз).

Лекция 15. Искусственные нейронные сети. Стохастические сети и сети с обратными связями

Стохастические методы обучения. Обучение Больцмана. Алгоритм обучения Больцмана. Обучение Коши. Сети с обратными связями. Сеть Хопфилда. Правило обучения Хебба. Процедура ортогонализации образов. Сеть ДАП (двунаправленная ассоциативная память). Сеть АРТ (адаптивная резонансная теория).

Стохастические методы обучения

Искусственная нейронная сеть обучается посредством некоторого процесса, модифицирующего ее веса. Если обучение успешно, то предъявление сети множества входных сигналов приводит к появлению желаемого множества выходных сигналов. **Стохастические методы обучения** выполняют псевдослучайные изменения величин весов, сохраняя те изменения, которые ведут к улучшениям.

Локальные минимумы мешают всем алгоритмам обучения, основанным на поиске минимума функции ошибки, включая сети обратного распространения, и представляют серьезную и широко распространенную проблему.

Стохастические методы позволяют решить эту проблему. Стратегия коррекции весов, вынуждающая веса принимать значение глобального оптимума, возможна.

В качестве объясняющей аналогии предположим, что на рис. 1 изображен шарик на поверхности в коробке. Если коробку сильно потрясти в горизонтальном направлении, то шарик будет быстро перекатываться от одного края к другому.

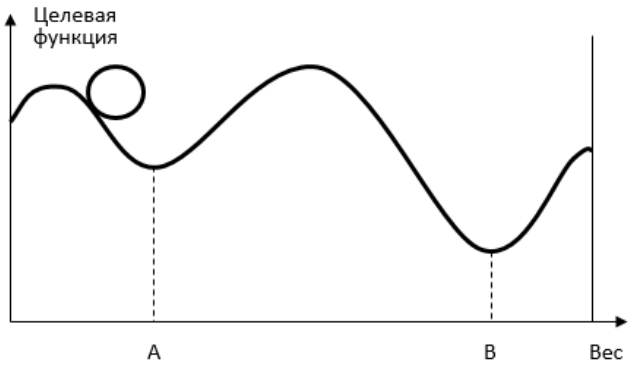

Рис. 1. Проблема локальных минимумов

Нигде не задерживаясь, в каждый момент шарик будет с равной вероятностью находиться в любой точке поверхности. Если постепенно уменьшать силу встряхивания, то будет достигнуто условие, при котором шарик будет на короткое время «застревать» в точке В. При еще более слабом встряхивании шарик будет на короткое время останавливаться как в точке А, так и в точке В. При непрерывном уменьшении силы встряхивания будет достигнута критическая точка, когда сила встряхивания достаточна для перемещения шарика из точки *A* в точку *B*, но недостаточна для того, чтобы шарик мог выбраться из *B* в *A.* Таким образом, окончательно шарик остановится в точке глобального минимума, когда амплитуда встряхивания уменьшится до нуля.

Обучение Больцмана

Искусственные нейронные сети могут обучаться, по существу, тем же самым образом посредством случайной коррекции весов. Вначале делаются большие случайные коррекции с сохранением только тех изменений весов, которые уменьшают целевую функцию. Затем средний размер шага постепенно уменьшается, и глобальный минимум в конце концов достигается.

Такая процедура напоминает отжиг металла, поэтому для ее описания часто используют термин **«имитация отжига».** В металле, нагретом до температуры, превышающей его точку плавления, атомы находятся в сильном беспорядочном движении. Как и во всех физических системах, атомы стремятся к состоянию минимума энергии, но при высоких температурах энергия атомных движений препятствует этому. В процессе постепенного охлаждения металла возникают все более низкоэнергетические состояния, пока в конце концов не будет достигнуто наиболее низкое из возможных состояний, глобальный минимум. В процессе отжига распределение энергетических уровней описывается следующим соотношением:

$$P(E)=e^{\frac{-E}{kT}}$$

где $P(E)$ - вероятность того, что система находится в состоянии с энергией E, k - постоянная Больцмана; T - температура по шкале Кельвина.

При высоких температурах вероятность $P(E)$ приближается к единице для всех энергетических состояний. Таким образом, высокоэнергетическое

состояние почти столь же вероятно, как и низкоэнергетическое. По мере уменьшения температуры вероятность высокоэнергетических состояний уменьшается по сравнению с низкоэнергетическими. При приближении температуры к нулю становится весьма маловероятным, чтобы система находилась в высокоэнергетическом состоянии.

Этот стохастический метод непосредственно применим к обучению искусственных нейронных сетей и относится к классу алгоритмов обучения с учителем.

Алгоритм обучения Больцмана

Шаг 1. Определить переменную T, представляющую искусственную температуру. Придать T большое начальное значение.

Шаг 2. Подать на вход сети один из входных образов обучающей выборки и вычислить реальный выход и значение функции ошибки сети (как в алгоритме обратного распространения).

Шаг 3. Придать случайное изменение Δw_{ij} выбранному весу w_{ij} и пересчитать выход сети и изменение функции ошибки в соответствии со сделанным изменением веса.

Шаг 4. Если функция ошибки уменьшилась, то сохранить изменение веса. Если изменение веса приводит к увеличению функции ошибки, то вероятность сохранения этого изменения вычисляется с помощью распределения Больцмана: $P(\Delta w_{ij}) = e^{\frac{-\Delta w_{ij}}{kT}}$. Выбирается случайное число r из равномерного распределения от нуля до единицы. Если вероятность $P(\Delta w_{ij})$ больше, чем r, то изменение сохраняется, в противном случае величина веса возвращается к предыдущему значению.

Шаг 5. Повторять шаги 3 и 4 для каждого из весов сети, постепенно уменьшая температуру T, пока не будет достигнуто допустимо низкое значение целевой функции.

Шаг 6. Повторять шаги 2—5 для всех векторов обучающей выборки (возможно неоднократно), пока функция ошибки не станет допустимой для каждого из них.

Замечание 1. На шаге 4 система может делать случайный шаг в направлении, портящем функцию ошибки, позволяя ей тем самым вырываться из локальных минимумов, где любой малый шаг увеличивает целевую функцию.

Замечание 2. В работах, посвященных больцмановскому обучению, показано, что для достижения сходимости к глобальному минимуму скорость уменьшения искусственной температуры должна подчиняться закону

$$T_N = \frac{T_0}{\ln(1+N)}$$

где N - номер итерации обучения. Этот результат предсказывает очень медленную сходимость процесса обучения, что является существенным недостатком метода.

Обучение Коши

В этом методе распределение Больцмана заменяется на распределение Коши. Распределение Коши имеет, как показано на рис. 2, более высокую вероятность больших шагов. Дисперсия распределения Коши бесконечна.

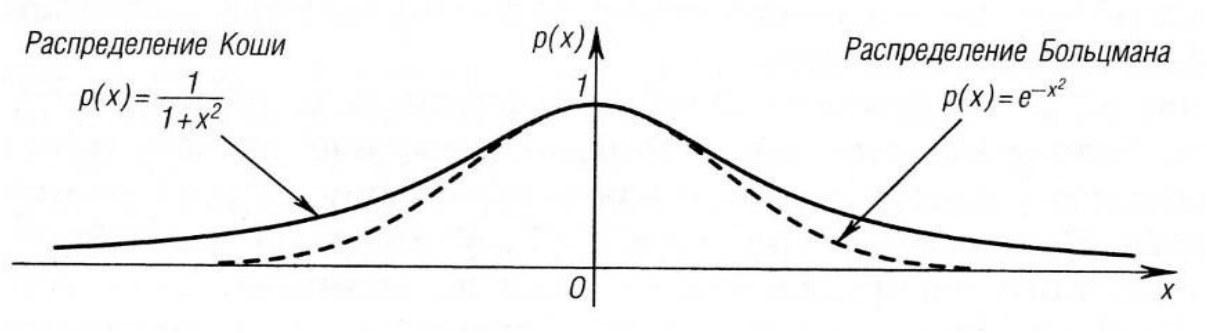

Рис. 2. Распределение Коши и распределение Больцмана

С помощью такого простого изменения максимальная скорость уменьшения температуры становится обратно пропорциональной линейной величине, а не логарифму, как для алгоритма обучения Больцмана. Эта связь может быть выражена следующим образом: $T_N = \frac{T_0}{1+N}$.

Таким образом, время обучения резко уменьшается. Распределение Коши имеет вид

$$P(\Delta w_{ij}) = \frac{T_N}{T_N^2 + \Delta w_{ij}^2}$$

где $P(\Delta w_{ij})$ - вероятность принять изменение веса Δw_{ij} .

Несмотря на улучшение скорости обучения, даваемое распределением Коши по сравнению с распределением Больцмана, время сходимости все еще может в 100 раз превышать время для алгоритма обратного распространения.

Комбинирование обратного распространения с обучением Коши.

Коррекция весов в комбинированном алгоритме, использующем обратное распространение и обучение Коши, состоит из двух компонент: компоненты, вычисляемой с использованием алгоритма обратного распространения, и случайной компоненты, определяемой распределением Коши.

Эти компоненты вычисляются для каждого веса, и их сумма является величиной, на которую изменяется вес. Как и в алгоритме Коши, после вычисления изменения веса вычисляется целевая функция. Если имеет место улучшение, изменение сохраняется. В противном случае оно сохраняется с вероятностью, определяемой распределением Коши.

Коррекция веса вычисляется с использованием представленных ранее уравнений для каждого из алгоритмов:

$$w_{ij}^{N+1} = w_{ij}^N - \eta\alpha \frac{\partial E}{\partial w_{ij}} + (1-\eta)\Delta w_{ij}^N ,$$

где η - коэффициент, управляющий относительными величинами обучения Коши и обратного распространения в компонентах весового шага.

Если η приравнивается нулю, метод становится обучением Коши. Если η приравнивается единице, метод становится алгоритмом обратного распространения.

Комбинированная сеть, использующая обратное распространение и обучение Коши, обучается быстрее, чем каждый из алгоритмов в отдельности. Сходимость к глобальному минимуму гарантируется алгоритмом Коши, и во многих экспериментах по обучению сеть практически не попадала в локальные минимумы.

Сети с обратными связями

Рассмотренные ранее нейросетевые архитектуры относятся к классу сетей с направленным потоком распространения информации и не содержат обратных связей. После обучения на этапе функционирования сети каждый нейрон выполняет свою функцию - передачу выходного сигнала - ровно один раз. В общем случае может быть рассмотрена нейронная сеть, содержащая произвольные **обратные связи**, т. е. пути, передающие сигналы от выходов к входам. Отклик таких сетей является динамическим, т. е. после подачи нового входа вычисляется выход и, передаваясь по обратной связи, модифицирует вход. Затем выход повторно вычисляется, и процесс повторяется снова и снова. Для устойчивой сети последовательные итерации приводят к все меньшим изменениям выхода, и в результате выход становится постоянным. Для многих сетей процесс никогда не заканчивается, такие сети называются неустойчивыми. Неустойчивые сети обладают интересными свойствами и могут рассматриваться в качестве примера хаотических систем, но для большинства практических приложений используются сети, которые дают постоянный выход.

Сеть Хопфилда

Нейросетевые архитектуры получили всеобщее признание во многом благодаря исследованиям Джона Хопфилда, физика из Калифорнийского технологического института. Он изучал свойства сходимости сетей на основе принципа минимизации энергии, а также разработал на основе этого принципа семейство нейросетевых архитектур.

Рассмотрим однослойную сеть с обратными связями, состоящую из n входов и n нейронов (рис. 3). Каждый вход связан со всеми нейронами. Так как выходы сети заново подаются на входы, то y_i - это значение i -го выхода, который на следующем этапе функционирования сети становится i -м входом.

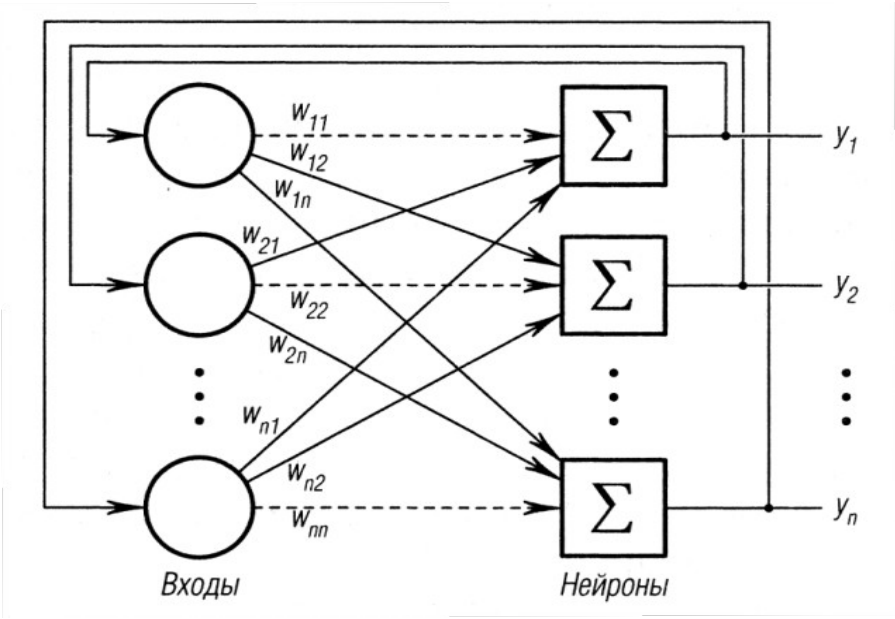

Рис. 3. Модель сети Хопфилда

Совокупность выходных значений всех нейронов y_i на некотором этапе N образует **вектор состояния сети** Y^N. Нейродинамика приводит к изменению вектора состояния на Y^{N+1}.

Обозначим силу синаптической связи от i -го входа к j -му нейрону как w_{ij} . Каждый j -й нейрон сети реализует пороговую активационную функцию следующего вида:

$$y_j^{N+1}=f(y_i)=\{-1, s_j<\Theta_j; 1, s_j>\Theta_j; y_j^N=\Theta_j.$$

Здесь $s_j=\sum_{i=1}^{n} ¿\, y_j^N \cdot w_{ij}; y_{\square\square}$ - значение выхода j -го нейрона на предыдущем этапе функционирования сети; Θ_j - пороговое значение j -го нейрона.

В модели Хопфилда предполагается условие симметричности связей $w_{ij}=¿\quad w_{ji}$ с нулевыми диагональными элементами $w_{ii}=0$. Устойчивость такой сети может быть доказана следующим образом. Введем в рассмотрение функцию, зависящую от состояния сети Y и называемую функцией энергии сети Хопфилда:

$$E(Y)=\frac{-1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} y_i y_j+\sum_{j=1}^{n}\Theta_j y_j$$

Вычислим изменение функции энергии ΔE , вызванное изменением состояния j -го нейрона Δy_j .

$$w_{ij} y_i + \Theta_j$$

$$-\sum_{i \neq j}^{\square} ¿ \Delta y_j = -(s_j - \Theta_j) \Delta y_j$$

$$\Delta E = ¿$$

(здесь мы воспользовались симметричностью связей и тем, что $w_{ii} = 0$). Допустим, что величина s_j больше порога Θ_j . Тогда выражение в правой части положительно, а из вида активационной функции следует, что новый выход нейрона j должен стать равным 1, т. е. измениться в положительную сторону (или остаться без изменения). Это значит, что $\Delta y_j \geq 0$, и тогда $\Delta E \leq 0$. Следовательно, энергия сети либо уменьшится, либо останется без изменения. Далее допустим, что величина s_j меньше порога. Тогда получается новое значение $y_j = -1,$ и величина Δy_j . может быть только отрицательной или нулем. Следовательно, опять энергия должна уменьшиться или остаться без изменения. Если величина s_j равна порогу Θ_j изменение Δy_j равно нулю и энергия остается без изменения.

Эти рассуждения показывают, что любое изменение состояния нейрона либо уменьшит функцию энергии, либо оставит ее без изменения. Так как функция энергии задана на конечном множестве $(y_i \in \{-1,1\}),$ то она ограничена снизу, и вследствие непрерывного стремления к уменьшению, в конце концов, должна достигнуть минимума и прекратить изменение. По определению такая сеть является устойчивой.

Поверхность функции энергии E в пространстве состояний имеет весьма сложную форму с большим количеством локальных минимумов. Стационарные состояния, отвечающие минимумам, могут интерпретироваться как **образы памяти нейронной сети**. Сходимость к такому образу соответствует процессу извлечения из памяти. При произвольной матрице связей W образы также произвольны. Для записи в память сети какой-либо конкретной информации требуется определенное значение весов W , которое может получаться в процессе обучения.

Правило обучения Хебба

Метод обучения для сети Хопфилда опирается на исследования Дональда Хебба, реализовавшего простой механизм обучения, названный **правилом Хебба**.

Пусть задана обучающая выборка образов $X^k, k=1,2,\ldots,K$. Требуется построить матрицу связей W такую, что соответствующая нейронная сеть будет иметь в качестве стационарных состояний образы обучающей выборки (значения порогов нейронов Θ_j положим равными нулю). В случае одного обучающего образа $X=(x_1,x_2,\ldots,x_n), x_i\in\{-1,1\}$, правило Хебба приводит к матрице $w_{ij}=x_i x_j, i\neq j, w_{ii}=0.$ Покажем, что состояние $Y=X$ является стационарным для сети Хопфилда с данной матрицей W.

Действительно, значение функции энергии в состоянии X является для нее глобальным минимумом:

$$E(X)=\frac{-1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}x_i x_j=\frac{-1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j x_i x_j=\frac{-1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^2 x_j^2=\frac{-1}{2}n^2,$$

т. е. сеть прекратит изменения, достигнув состояния X.

Для запоминания K образов применяется итерационный процесс: $w_{ij}^k=w_{ij}^{k-1}+x_i^k x_j^k, k=1,2,\ldots,K$ (считаем, что $w_{ij}^0=0$). Этот процесс приводит к полной матрице связей:

$$w_{ij}=\sum_{k=1}^{K} x_i^k x_j^k$$

Сеть Хопфилда нашла широкое применение в системах **ассоциативной памяти**, позволяющих восстанавливать идеальный образ по имеющейся неполной или зашумленной его версии.

Процедура ортогонализации образов

Два различных запоминаемых векторных образа сети $X^k, X^l (k\neq l)$ называются **ортогональными**, если их скалярное произведение равно нулю: $\sum_{j=1}^{n} x_j^k x_j^l=0$. Если все запоминаемые образы сети $X^k, k=1,2,\ldots,K$ попарно ортогональны, емкость памяти сети Хопфилда увеличивается до n, т. е. сеть может запомнить количество образов, не превосходящее число нейронов в ней. На этом свойстве основано улучшение правила Хебба: перед запоминанием в нейронной сети исходные образы следует ортогонализовать. Процедура расчета весовых коэффициентов в этом случае имеет следующий вид:

Шаг 1. Вычисляются элементы матрицы $B=(b_{kl}), k=1,2,\ldots,K$:

$$b_{kl} = \sum_{j=1}^{n} x_j^k x_j^l.$$

Шаг 2. Определяется матрица C, обратная к матрице $B: C = B^{-1}$.

Шаг 3. Задаются весовые коэффициенты сети Хопфилда:

$$w_{ij} = \sum_{j=1}^{n} x_j^k x_j^l c_{kl}.$$

Существенным недостатком метода является его нелокальность: прежде чем начать обучение, необходимо наперед знать все обучающие образы. Добавление нового образа требует полного переобучения сети.

Сеть ДАП (двунаправленная ассоциативная память)

Сеть Хопфилда реализует так называемую автоассоциативную память. Это означает, что образ может быть завершен или исправлен, но не может быть ассоциирован с другим образом. Двунаправленная ассоциативная память (ДАП), разработанная в 1988 году Бертом Коско (B. Kosko), является гетероассоциативной: она сохраняет пары образов и выдает второй образец пары, когда ассоциированный с ним первый образец подается на вход сети. Как и сеть Хопфилда, сеть ДАП способна к обобщению, вырабатывая правильные реакции, несмотря на искаженные входы. Сеть ДАП (рис. 4) содержит два слоя нейронов. Элементы весовой матрицы w_{ij} отражают связь между j -м нейроном первого слоя и j -м нейроном второго слоя, $i = 1, 2, \ldots, n, j = 1, 2, \ldots, m$.

В процессе функционирования сети входной вектор X умножается на транспонированную матрицу весов сети W^T и подается на вход первого слоя, в результате чего вырабатывается вектор выходных сигналов нейронов первого слоя Y. Вектор Y затем умножается на матрицу W и подается на вход второго слоя, который вырабатывает выходные сигналы, представляющие собой новый входной вектор X. Этот процесс повторяется до тех пор, пока сеть не достигнет стабильного состояния, в котором ни вектор X, ни вектор Y не изменяются. Нейроны в обоих слоях сети ДАП функционируют аналогично нейронам сети Хопфилда.

Этот процесс может быть выражен следующим образом:

$$x_i^N w_{ji}$$

$$\sum_{i=1}^{n} ¿, j=1,2,\ldots,m,$$

$$y_j^{N+1}=f ¿$$

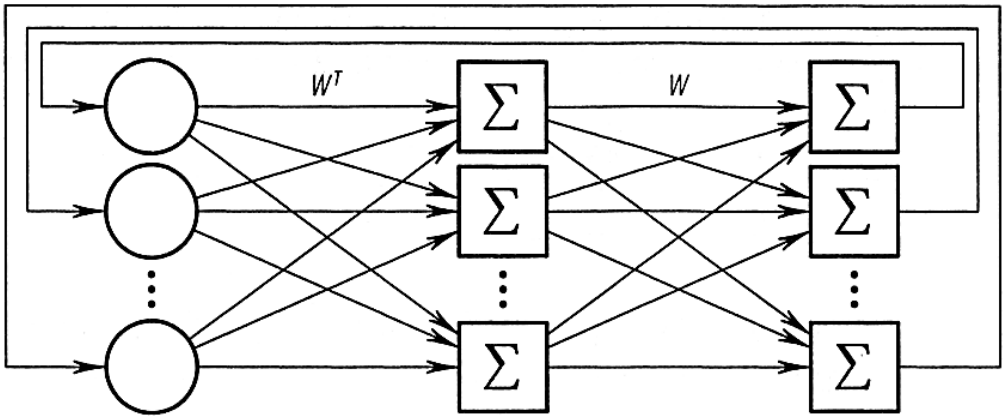

Рис. 4. Структура сети ДАП

$$y_j^{N+1} w_{ji}$$

$$\sum_{j=1}^{m} ¿, i=1,2,\ldots,n,$$

$$x_i^{N+1}=f ¿$$

где

$$f(s)=\{-1, s<\Theta_j; 1, s>\Theta_j; f^{prev}(s), s=\Theta_j.$$

$f^{prev}(s)$ значение функции активации данного нейрона на предыдущем шаге.

Пусть задана обучающая выборка ассоциированных образов $(X^k, Y^k), k=1,2,\ldots,K$. Весовая матрица сети ДАП вычисляется как сумма произведений всех векторных пар обучающего набора:

$$W_{ij}=\sum_{k=1}^{K} x_i^k y_j^k, i=1,2,\ldots,n, j=1,2,\ldots,m.$$

В отличие от сети Хопфилда, весовая матрица в сети ДАП не квадратная, что во многих случаях позволяет оптимизировать вычислительные затраты, необходимые для функционирования сети. Если рассмотреть пример с запоминанием букв A, B, C, то сеть Хопфилда в этом случае имела бы 10x7=70 входов и требовала для своей работы хранения весовой матрицы размером 70x70, содержащей 4900 элементов. Ассоциируем с каждым из входных образов сети двухбитовый вектор: символ A будет связан с вектором (1,-1,-1), символ B с вектором (-1, 1,-1), символ C с вектором (-1, -1, 1). Таким образом,

например, при подаче на вход искаженной версии буквы А, сеть после стабилизации должна выдавать образ (1, -1, -1). Так как ассоциированные пары заранее известны, это приведет к правильному распознаванию зашумленного входа. Но для работы такой сети требуется хранение всего 70x3 = 210 элементов весовой матрицы.

Основным недостатком сети ДАП, как и сети Хопфилда, является небольшая емкость памяти. Так, число запоминаемых ассоциаций не может превышать числа нейронов в меньшем слое. Если все пороговые значения Θ_j нулевые, то оценка еще ухудшается: размер запоминаемой выборки не должен превосходить $\frac{1}{2\log_2 l}$, где l — число нейронов в меньшем слое. Если этот лимит превышен, сеть начинает вырабатывать неверные выходные сигналы, воспроизводя ассоциации, которым не обучена.

Сеть АРТ (адаптивная резонансная теория)

Адаптивная резонансная теория включает две парадигмы, каждая из которых определяется формой входных данных и способом их обработки. АРТ-1 разработана для обработки двоичных входных векторов, в то время как АРТ-2 может классифицировать как двоичные, так и непрерывные векторы. Рассмотрим подробно сеть АРТ-1, так как несмотря на более простую архитектуру, именно она используется в большинстве практических приложений.

Сеть АРТ-1 обучается без учителя и реализует простой алгоритм кластеризации. В соответствии с этим алгоритмом первый входной сигнал считается образцом первого кластера. Следующий входной сигнал сравнивается с образцом первого кластера. Говорят, что входной сигнал принадлежит первому кластеру, если расстояние до образца первого кластера меньше порога. В противном случае второй входной сигнал - образец второго кластера. Этот процесс повторяется для всех следующих входных сигналов. Таким образом, число кластеров растет с течением времени и зависит как от значения порога, так и от метрики расстояния, использующейся для сравнения входных сигналов и образцов классов.

Сеть АРТ-1 содержит два слоя нейронов. Число нейронов первого слоя совпадает с размерностью входных образов. Число нейронов второго слоя изменяется в процессе настройки сети и совпадает с числом сформированных кластеров.

Алгоритм функционирования сети АРТ-1

Шаг 1. Инициализация сети:

$N=1, m=1; t_{ij}^N = b_{ij}^N = 1, i=1,2,\ldots,n, j=1,2,\ldots,m,$

где b_{ij}^N - синаптический вес связи от i -го нейрона первого слоя к j – му нейрону второго слоя на итерации с номером N , $t_{ij}^N - ¿$ синаптический вес связи от j -го нейрона второго слоя к i -му нейрону первого слоя на итерации с номером N . Веса b_{ij}^N и $t_{ij}^N, i=1,2,\ldots,n$ определяют образец, соответствующий нейрону j .

Задать $0<r<1$ - значение порога.

Шаг 2. Предъявление сети нового бинарного входного сигнала $X=(x_1, x_2, \ldots, x_n)$.

Шаг 3. Вычисление значений соответствия:

$$y_i = \sum_{i=1}^{n} b_{ij}^N x_i, j=1,2,\ldots,m.$$

Шаг 4. Выбор образца с наибольшим соответствием:

$$y_k = max_{1 \le j \le m} y_i.$$

Если $y_k = 0$, создать новый кластер, соответствующий входному образцу с весами $t_{ij}^N = b_{ij}^N = x_i,$ положить $m = m+1$ и перейти на шаг 8.

Шаг 5. Сравнение с порогом:

$$\| X \| = \sum_{i=0}^{n} x_i, \| TX \| = \sum_{i=0}^{n} t_{ik} x_i.$$

Если $\frac{\| TX \|}{\| X \|} > r$, перейти к шагу 7.

Шаг 6. Исключение примера с наибольшим значением соответствия. Значение соответствия образца y_k временно устанавливается равным нулю. Переход к шагу 4 (поиск нового значения y_k).

Ш а г 7. Адаптация примера с наибольшим значением соответствия:

$$t_{ik}^{N+1}=t_{ik}^{N}x_i, b_{ik}^{N+1}=\frac{t_{ik}^{N}x_i}{0{,}5+\sum_{i=1}^{n}t_{ik}^{N}x_i, i=1,2,\ldots,n.}$$

Шаг 8. Включение всех исключенных на шаге 6 образцов. Положить $N=N+1$. Возврат к шагу 2.

Замечание. Порог r показывает, насколько должен входной сигнал совпадать с одним их запомненных образцов, чтобы они считались похожими. Близкое к единице значение порога требует почти полного совпадения. При малых значениях порога даже сильно различающиеся входной сигнал и образец считаются принадлежащими одному кластеру.

На шаге 5 вычисляется отношение скалярного произведения входного сигнала и образца с наибольшим значением соответствия к числу единичных бит входного сигнала. Значение отношения сравнивается с порогом, введенном на первом шаге.

Если значение отношения больше порога, то входной сигнал считается похожим на образец с наибольшим значением соответствия. В этом случае образец кластера модифицируется путем выполнения операции AND (логическое «И») с входным вектором.

Если значение отношения меньше порога, то считается, что входной сигнал отличается от данного образца и осуществляется поиск другого похожего вектора. Если входной вектор отличается от всех образцов, то он рассматривается как новый образец. В сеть вводится нейрон, соответствующий новому образцу и рассчитываются значения синаптических весов.

Генетические алгоритмы

Методы оптимизации комбинаторных задач различной степени сложности. Генетические алгоритмы. Базовый генетический алгоритм. Последовательные модификации базового генетического алгоритма. Параллельные модификации базового генетического алгоритма. Классификация генетических алгоритмов.

Методы оптимизации комбинаторных задач различной степени сложности

В общем случае оптимизация или поиск наилучшего значения (набора параметров) некоторой заданной целевой функции является достаточно сложной задачей. Сложность оптимизации обусловливается, прежде всего,

видом целевой функции, которая может иметь как глобальный, так и локальный оптимумы.

В настоящее время не существует метода оптимизации, который позволил бы решить любую задачу (был универсальным) и при этом однозначно определен как лучший среди других методов по точности решения.

По степени приближения к точному решению, а также по характеру пространства поиска задачи могут быть разделены на следующие категории.

Комбинаторные задачи - характеризуются конечным и дискретным пространством поиска. Сущность любой комбинаторной задачи можно сформулировать следующим образом: найти на множестве X элемент x, удовлетворяющий совокупности условий $K(x)$, в предположении, что пространство поиска X содержит некоторое конечное число различных точек.

Общие задачи без ограничений - имеют нелинейное и неограниченное пространство поиска. Методы оптимизации для таких задач обычно полагаются на правильность аналитической формулировки целевой функции. Оптимизация функции без ограничений заключается в максимизации или минимизации некоторой функции $U(x_1,\ldots,x_p)$.

Общие задачи с ограничениями - могут быть сформулированы как задачи минимизации функции $U(x_1,\ldots,x_p)$. при следующих ограничениях:

$$g_i(x_1,\ldots,x_p)\geq 0 \quad \text{для} \quad 1\leq i\leq m, h_j(x_1,\ldots,x_p)=0 \quad \text{для} \quad 1\leq j\leq n.$$

Обычно задачи с ограничениями могут быть сведены к задачам без ограничений с помощью метода штрафов.

Если пространство поиска содержит конечное число точек, то наиболее точное решение может быть уверенно получено методом полного перебора. Этот метод имеет один очевидный недостаток - сложность вычислений, а следовательно, время, затрачиваемое на нахождение оптимального решения, существенно зависит от размерности пространства поиска. Метод перебора может быть достаточно эффективным только в небольшом пространстве поиска.

Градиентные методы, являющиеся основой линейного и нелинейного, динамического программирования, а также численного анализа, более универсальны, но менее точны. При этом усложнение ландшафта

пространства поиска приводит к снижению эффективности таких методов. Методы градиента не гарантируют получение единственного оптимального решения, за исключением случая, когда пространство отображения является выпуклым и не допускает появления второстепенных вершин, плато и т. д.

С другой стороны, **эвристические методы**, к которым относятся **генетические алгоритмы (ГА)**, являются наиболее универсальными, поэтому не гарантируют нахождения глобального оптимума, являющегося единственным решением задачи.

Характеристикой задачи и, соответственно, основой для классификации методов оптимизации является также сложность задачи. По степени сложности однозначно выделяются следующие задачи.

Линейные задачи - сложность которых определяется как $O(n)$, где n — размерность входных данных задачи.

Полиномиальные задачи (P) - для них известен алгоритм, сложность которого составляет полином заданной, постоянной и не зависящей от размерности входной величины n степени.

Экспоненциальные задачи - сложность которых не менее порядка f^n , где f - константа или полином от n .

Однако существует большое число задач, которые не попадают ни в один из перечисленных классов. Сложность решения таких задач не может быть определена априорно. К ним относятся: оптимизация пути коммивояжера, оптимальная загрузка емкости, оптимизация маршрутов, инвестиций и т. д.

В общем случае задача оптимизации в настоящее время не может быть отнесена к какому-либо классу.

ГА являются стохастическим эвристическим методом, в котором вероятность выбора состояния $S(t+1)$ зависит от состояния $S(t)$ и косвенно от предыдущих состояний. Стохастические методы позволяют решать широкий класс таких задач, поскольку не требуют жесткой формализации. Следует отметить, что стохастические методы оптимизации используются для решения NP-сложных комбинаторных задач, т. е. таких задач, к которым сводима любая задача из класса NP. При этом NP-сложные задачи не обязательно относятся к классу NP.

Каждый из стохастических и эвристических методов имеет свои достоинства и недостатки, обусловленные формулировкой и размерностью решаемой задачи. **При этом математически доказано, что для комбинаторных задач оптимизации средняя эффективность всех алгоритмов для всех возможных задач одинакова**. На рис. 1 приведена классификация эвристических и стохастических алгоритмов.

Полученные в результате решения большого количества задач результаты, усредненные по 10 запускам, доказывают справедливость утверждения о сравнимости эффективности всех перечисленных алгоритмов поиска глобального оптимума. Вместе с тем результаты, полученные при одном запуске, говорят о наибольшей эффективности двух методов поиска - ГА и поиска с учетом запретов.

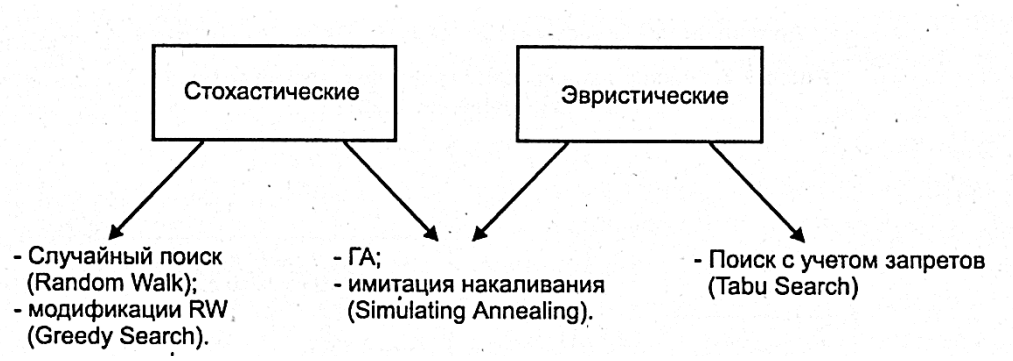

Рис. 1. Классификация эвристических и стохастических алгоритмов

Стохастические методы позволяют решать широкий класс таких задач, поскольку не требуют жесткой формализации.

Несмотря на некоторые различия в классификациях, ГА и поиск с учетом запретов имеют общую основу. Их объединяет использование эвристики для перехода из текущего состояния в последующее. **Особенностью ГА является работа с пространством поиска с помощью комбинирования решений, а поиска с учетом запретов - использование памяти состояний**.

Эффективностью обоих методов обусловлено появление метода HGT (гибридной стратегии), использующей поиск с учетом запретов для повышения эффективности генетических операторов рекомбинации и мутации. Вместе с тем, если по эффективности отмеченные методы сравнимы, то по надежности поиск с учетом запретов уступает ГА, поскольку для данного метода качество решения существенно зависит от начального состояния (или решения). Поэтому начальное решение с высоким значением оценочной функции (в случае решения задачи минимизации) может быстро привести к желаемому

решению, а начальное решение с низким значением оценочной функции - существенно снизить скорость поиска.

Анализ результатов использования ГА позволяет выделить следующие условия, при выполнении которых задача решается эффективно:

- Q большое пространство поиска, ландшафт которого является негладким (содержит несколько экстремумов);

- Q сложность формализации оценки качества решения функцией степени пригодности;

- Q многокритериальность поиска;

- Q поиск приемлемого решения по заданным критериям в отличие от поиска единственного оптимального.

Генетические алгоритмы

Генетические алгоритмы (ГА) относятся к числу универсальных методов оптимизации, позволяющих решать задачи различных типов (комбинаторные, общие задачи с ограничениями и без ограничений) и различной степени сложности. При этом ГА характеризуются возможностью как однокритериального, так и многокритериального поиска в большом пространстве, ландшафт которого является негладким.

В последние годы резко возросло число работ, прежде всего зарубежных ученых, посвященных развитию теории ГА и вопросам их практического использования. Результаты данных исследований показывают, в частности, что ГА могут получить более широкое распространение при интеграции с другими методами и технологиями. Появились работы, в которых доказывается эффективность интеграции ГА и методов теории нечеткости, а также нейронных вычислений и систем.

Эффективность такой интеграции нашла практическое подтверждение в разработке соответствующих инструментальных средств (ИС). Так, фирма Attar Software включила ГА-компонент, ориентированный на решение задач оптимизации, в свои ИС, предназначенные для разработки экспертной системы. Фирма California Scientific Software связала ИС для нейронных сетей с FA-компонентами, обеспечивающими автоматическую генерацию и настройку нейронной сети. Фирма NIBS Inc. включила в свои ИС для нейронных етей, ориентированные на прогнозирование рынка ценных бумаг, ГА-компоненты. которые, по мнению финансовых экспертов, позволяют уточнять прогнозирование.

Несмотря на известные общие подходы к такой интеграции ГА и нечеткой логики, по-прежнему актуальна задача определения наиболее значимых параметров операционного базиса ГА с целью их адаптации в процессе работы ГА за счет использования нечеткого продукционного алгоритма (НПА).

Перечисленные далее причины коммерческого успеха инструментальных средств в области искусственного интеллекта могут рассматриваться как общие требования к разработке систем анализа данных, используемых ГА:

интегрированность - разработка ИС, легко интегрирующихся с другими информационными технологиями и средствами;

открытость и переносимость - разработка ИС в соответствии со стандартами, обеспечивающими возможность исполнения в разнородном программно-аппаратном окружении, и переносимость на другие платформы без перепрограммирования;

использование языков традиционного программирования - переход к ИС, реализованным на языках традиционного программирования (С, С++ и т. д.), что упрощает обеспечение интегрированности, снижает требования приложений к быстродействию ЭВМ и к объемам оперативной памяти;

архитектура "клиент-сервер" - разработка ИС, поддерживающих распределенные вычисления в архитектуре "клиент-сервер", что позволяет снизить стоимость оборудования, используемого в приложениях, децентрализовать приложения и повысить их производительность.

Перечисленные требования обусловлены необходимостью создания интегрированных приложений, т. е. приложений, объединяющих в рамках единого комплекса традиционные программные системы с системами искусственного интеллекта и ГА в частности.

Интеграция ГА и нейронных сетей позволяет решать проблемы поиска оптимальных значений весов входов нейронов, а **интеграция ГА и нечеткой лог**ики позволяет оптимизировать систему продукционных правил, которые могут быть использованы для управления операторами ГА (двунаправленная интеграция).

Одним из наиболее востребованных приложений ГА в области Data Mining является поиск наиболее оптимальной модели (поиск алгоритма, соответствующего специфике конкретной области).

Базовый генетический алгоритм

Эволюционные алгоритмы, моделирующие процессы естественной эволюции, были предложены уже в 60-х годах прошлого века. Их особенностью является то, что они опираются на естественную эволюцию в природе, используя основные ее механизмы (отбор или селекцию, скрещивание и мутацию). Известны утверждения: "алгоритм является хорошим оптимизационным методом, потому что его принцип используется в природе", и наоборот: "алгоритм не может быть хорошим оптимизационным методом, потому что вы не находите его в природе".

Моделирование процесса естественной эволюции для эффективной оптимизации является первостепенной задачей теоретических и практических исследований в области эволюционных алгоритмов.

В 70-х годах прошлого века независимо друг от друга появились **два различных направления в области эволюционных алгоритмов: генетический алгоритм Холланда и эволюционные стратегии (ЭС) Речёнберга и Швефела**. Эволюционные стратегии используют операторы селекции и мутации, а если использовать биологические термины, то эволюционная стратегия моделирует естественную эволюцию с помощью непарной репродукции (рис. 2).

Эволюционные стратегии ($\mu + \lambda$).
Шаг 1. Создание первоначальной популяции размера λ.
Шаг 2. Вычисление пригодности $F(x_i)$ $i = 1, \ldots, \lambda$.
Шаг 3. Селекция (отбор) $\mu < \lambda$ лучших индивидов.
Шаг 4. Создание λ / μ потомков каждого из μ индивидов с небольшими вариациями.
Шаг 5. Возврат к шагу 2.

Рис. 2. Разновидность эволюционных алгоритмов - эволюционные стратегии

Алгоритмы поиска, которые моделируют **парную репродукцию**, называются генетическими алгоритмами. Парная репродукция характеризуется рекомбинацией двух родительских строк для создания потомков. Эта рекомбинация называется **скрещиванием**.

Предпочтение разных генетических операторов в ЭС и ГА определило отношение к используемому размеру популяции. Так, Холланд

подчеркивал важность рекомбинации в больших популяциях, в то время как Реченберг и Швефел, главным образом, рассматривали мутацию в очень маленьких популяциях.

При работе с ГА решения задачи должны быть представлены в виде строки с бинарной, целочисленной или вещественной кодировкой. Способ кодирования предполагает работу со строками фиксированной или переменной длины, возможна также и контекстно-зависимая кодировка. Основным отличием генетических программ (ГП) от ГА является работа с деревьями решений. При этом в ГП отсутствует необходимость в генетическом представлении задачи. Такая схема представления вносит гибкость в описание структур данных, однако решения могут стать очень объемными без улучшения производительности. Это справедливо и для эволюционных программ (ЭП).

На рис. 3 приведен базовый или стандартный ГА (СГА), предложенный Холландом, который явился основой для различных модификаций.

СГА.
Шаг 0. Определение генетического представления задачи.
Шаг 1. Создание первоначальной популяции индивидов $P(0) = x_1^0, \ldots, x^0_N$, $t = 0$.
Шаг 2. Вычисление средней пригодности $f_{ср}(t) = \sum_i^N f(x_i)/N$. Вычисление нормализованного значения степени пригодности $f(x_i)/f_{ср}(t)$ для каждого индивида.
Шаг 3. Назначение каждому индивиду x_i вероятности $p(x_i,t)$ пропорционально нормализованной пригодности. Выбор N векторов из $P(t)$, используя полученное распределение. Это дает набор отобранных родителей.
Шаг 4. Формирование случайным образом из данного набора $N/2$ пар. Применение к каждой паре скрещивания, а также других генетических операторов, таких как мутация, для формирования новой популяции $P(t + 1)$.
Шаг 5. $t = t + 1$, возврат к шагу 2.

Рис. 3. Стандартный генетический алгоритм

Последовательные модификации базового генетического алгоритма

Как показывает анализ, модификации ГА отличаются, прежде всего, способом селекции индивидов. В основных модификациях ГА несколько способов селекции используется для достижения различных целей - упрощения формирования промежуточной популяции, распараллеливания работы алгоритма, возможности анализа и предсказания поведения ГА. Было произведено сравнение четырех различных схем селекции (для СГА и SSGA, рассматриваемых далее, показавшее, что эффективность всех методов

примерно одинакова. Таким образом, в настоящее время абсолютно лучший метод селекции не определен.

Модификация стандартного варианта ГА (Steady State GA) [Whitley и Kauth, 1988] затронула способ формирования промежуточной популяции (Mating Pool), являющейся результатом отбора (селекции) для формирования наследников с помощью генетических операторов. SSGA не формируют промежуточную популяцию как стандартный ГА, а осуществляют последовательно выбор пары наилучших индивидов, применяя к ним генетические операторы с целью формирования наследников, которые заменяют худшие индивиды популяции. Данная модификация ГА представлена на рис. 4.

SSGA.
Шаг 0. Определение генетического представления задачи.
Шаг 1. Создание первоначальной популяции $P(0) = x_1^0, \ldots, x_N^0$, $t = 0$.
Шаг 2. Вычисление относительной (нормализованной) степени пригодности $f_n(x_i) = f(x_i)/\sum_i^N f(x_i)/N$.
Шаг 3. Выбор пары из лучших индивидов. Выбор худшего индивида. Применение скрещивания и мутации к выбранной паре лучших индивидов. Результат замещает худший индивид.
Шаг 4. $t = t + 1$, возврат к шагу 2.

Рис. 4. Steady State GA

При проектировании ГА могут быть выгодно использованы знания, полученные селекционерами в области искусственной селекции. Генетические алгоритмы селекционеров (ГАС) моделируют именно искусственную селекцию. ГАС представлен на рис. 5, где под виртуальным селекционером понимается некоторый механизм селекции, который и является основным отличием ГАС от стандартного ГА.

ГАС.
Шаг 0. Определение генетического представления задачи.
Шаг 1. Создание первоначальной популяции $P(0)$ размером N, $t = 0$.
Шаг 2. Виртуальный селекционер отбирает T % популяции для создания потомков. Это дает набор отобранных родителей.
Шаг 3. Формирование случайным образом из данного набора $N/2$ пар. Применение к каждой паре скрещивания и мутации, формируя новую популяцию $P(t + 1)$.
Шаг 5. $t = t + 1$, возврат к шагу 2.
Шаг 6. Возврат к шагу 3.

Рис. 5. Генетический алгоритм селекционеров

Селекция основывается преимущественно на статистических методах, которые позволяют выполнить теоретический анализ и прогнозировать эффективность механизмов селекции, мутации и рекомбинации с помощью введенных уравнений селекции, реакции на селекцию и понятия наследственности.

Еще одна модификация ГА затрагивает решение многокритериальных задач. Многокритериальный ГА (МГА) также является модификацией стандартного ГА и отличается способом селекции, поскольку при отборе пар родителей 1 этом случае используется не один, а несколько критериев. При этом предлагается большое число вариантов схем селекции и соответственно вариантов МГА. На рис. 6 приведен вариант МГА, предложенный Schaffer в 1984 г.,- векторный ГА (VEGA). Сравнительные оценки показывают, что по эффективности VEGA имеет средние показатели, однако не оценивалась вычислительная сложность для различных вариантов МГА, по которой VEGA может существенно улучшить свои показатели.

Многокритериальный ГА.
Шаг 0. Определение генетического представления задачи.
Шаг 1. Создание первоначальной популяции $P(0) = x^0_1, \ldots, x^0_N$, $t = 0$.
Шаг 2. Последовательное выполнение шагов 2.1–2.3.
Шаг 2.1. Вычисление значения степени пригодности каждого индивида по критерию $i = 1, \ldots, k$.
Шаг 2.2. Для j от 1 до N/k осуществление селекции индивида из популяции в промежуточную популяцию.
Шаг 2.3. Возврат к шагу 2.1, если $I < k$.
Шаг 3. Формирование случайным образом из данного набора $N/2$ пар. Применение к каждой паре скрещивания, а также других генетических операторов, таких как мутация, формируя новую популяцию $P(t + 1)$.
Шаг 4. $t = t + 1$, возврат к шагу 2.

Рис. 6. Многокритериальный генетический алгоритм

Параллельные модификации базового генетического алгоритма

Стандартный ГА представляет собой строго синхронизованный последовательный алгоритм, который в условиях большого пространства поиска или сложного ландшафта пространства поиска может быть неэффективен по критерию времени. Эту проблему позволяет решить другой

вид ГА - параллельный генетический алгоритм (ПГА). Следует отметить, что любая последовательная модификация стандартного ГА может быть преобразована в параллельную.

По степени распараллеливания можно выделить следующие типы параллельных ГА:

ПГА на базе популяции;

ПГА на базе подпопуляций;

ПГА на базе индивидов.

ПГА на базе популяции сохраняет стандартную структуру ГА, работающего с целой популяцией, распараллеливание реализуется на этапе скрещивания и мутации (см. шаг 4, рис. 3). По степени распараллеливания процессов можно выделить следующие модели:

синхронная модель "ведущий-ведомый", где главный процесс хранит целую популяцию в собственной памяти, выполняет селекцию, скрещивание и мутацию, но оставляет вычисление степени пригодности новых индивидов к подчиненным процессам;

полусинхронная модель "ведущий-ведомый", где новый индивид обрабатывается по мере освобождения одного из процессов;

асинхронная параллельная модель, где индивиды популяции хранятся в общей памяти, к которой можно обращаться к параллельным процессам. Каждый процесс выполняет оценку степени пригодности, а также генетические операции.

Каждый процесс работает независимо от других. Единственное отличие между этой моделью и стандартным ГА заключается в механизме селекции. Очевидным в этом случае является вариант использования N/2 параллельных процессоров при популяции в N индивидов. Тогда каждый процессор дважды случайным образом выбирает два индивида из общей памяти и оставляет лучшего. Два выбранных индивида затем подвергаются скрещиванию, мутации и оценке степени пригодности. Возникающие в результате наследники размещаются в общей памяти.

Распределенный ПГА.
Шаг 0. Определение генетического представления задачи.
Шаг 1. Создание первоначальной популяции индивидов и разделение на подпопуляции $SP_1, \ldots, SP_N$.
Шаг 2. Формирование структуры подпопуляций.
Шаг 3. Для SPi , $i = 1, \ldots, N$ — выполнение параллельно шагов 3.1–3.3.
Шаг 3.1. Применение в течение m поколений селекции и генетических операторов.
Шаг 3.2. Перемещение k хромосом в соседние подпопуляции.
Шаг 3.3. Получение хромосом из соседних подпопуляций.
Шаг 4. Возврат к шагу 3.

Рис. 7. Распределенный параллельный генетический алгоритм

Особенность **ПГА на базе подпопуляции** заключается в использовании независимых конкурирующих подпопуляций, которые обмениваются индивидами с заданной частотой (распределенный ПГА, рис. 7). При этом каждый процессорный блок выполняет последовательный ГА с собственной подпопуляцией, при условии максимизации одной общей для всех функции степени пригодности. В этом случае для обмена индивидами должна быть определена структура связей подпопуляций. С точки зрения оценки и сравнения эффективности может быть рассмотрен вариант распределенной модели, в которой обмен индивидами не осуществляется. Результаты проведенных экспериментов свидетельствуют о большей эффективности распределенного ПГА по сравнению с этим частным случаем, а также со стандартным ГА.

Существенным недостатком модели может стать снижение степени разнообразия при интенсивном обмене индивидами. С другой стороны, недостаточно частое перемещение может привести к преждевременной сходимости подпопуляций. При построении такой модели важно определить следующее:

связи между процессорами для обмена индивидами;

частоту обмена индивидами (оптимальной является частота обмена через 20 поколений);

степень перемещения или число обмениваемых индивидов (оптимальным является 20 % подпопуляции);

способ селекции индивида для обмена;

критерий, по которому полученный индивид сможет заменить члена подпопуляции,

С точки зрения времени и даже числа поколений, затрачиваемых на решение задачи, ПГА эффективнее стандартного ГА, но при этом некоторые задачи могут быть слишком простыми для ПГА. Параллельный поиск имеет смысл в том случае, если пространство поиска большое и сложное. Увеличение числа процессоров в данной модели улучшает скорость сходимости, но не качество решения.

ПГА на базе индивидов имеют одну строку индивида, постоянно находящуюся в каждом процессорном элементе (ячейке). Индивиды выбирают пары и рекомбинируют с другими индивидами в их непосредственном ближайшем окружении (по вертикали и горизонтали). Выбранный индивид затем совмещается с индивидом, постоянно находящимся в ячейке. В результате формируется один наследник, который может или не может заменить индивида в ячейке в зависимости от выбранной схемы замещения. Таким образом, модель является полностью распределенной и не нуждается в централизованном управлении (рис. 8).

ПГА на базе индивидов.
Шаг 0. Определение генетического представления задачи.
Шаг 1. Создание первоначальной популяции индивидов и формирование структуры популяции.
Шаг 2. Локальное повышение каждым индивидом своей производительности (hill-climbing).
Шаг 3. Выполнение каждым индивидом селекции с целью поиска пары.
Шаг 4. Применение к паре скрещивания, а также других генетических операторов, таких как мутация.
Шаг 5. Локальное повышение наследником своей производительности (hill-climbing). Замещение наследником родителя в соответствии с заданным критерием качества.
Шаг 6. Возврат к шагу 3.

Рис. 8. Параллельный генетический алгоритм на базе индивидов

При работе с моделью на базе индивидов необходимо задать:

структуру связей ячеек;

схему селекции;

схему замещения.

Исследования этой модели показали, что для сложных задач она способна обеспечить лучшие решения, чем стандартный ГА.

Классификация генетических алгоритмов

В ходе исследований в области генетических алгоритмов и эволюционных алгоритмов в целом появилось большое количество направлений, и их число непрерывно растет.

Классификация ЭА и основные модификации стандартного ГА, приведенного на рис. 2, отражены на рис. 9.

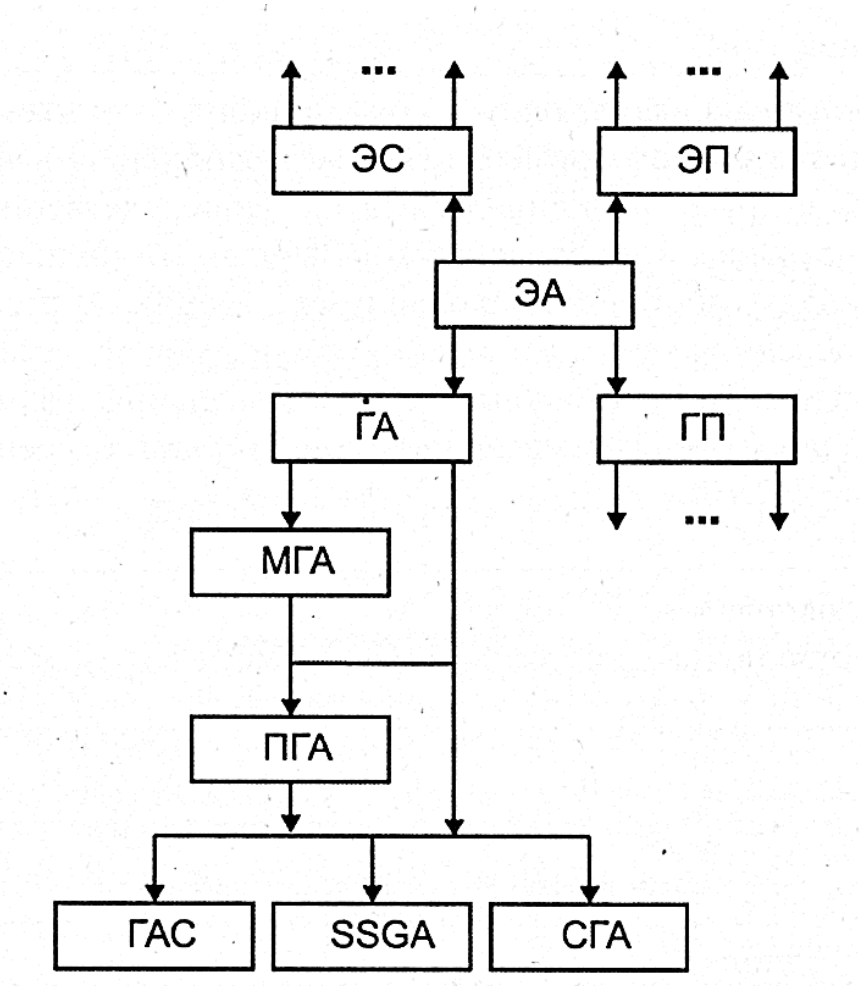

Рис. 9. Классификация эволюционных алгоритмов

Когнитивная компьютерная графика

Определение. Степень организованности информации. Задача сжатия информации. Анаморфозы. Метод анаморфирования. Проблемы реализации анаморфоз. Численные методы построения анаморфоз.

Определение.

Под вычислительным интеллектом (ВИ) понимают научное направление, где решаются задачи искусственного интеллекта на основе новых нетрадиционных методов вычислений, а под технологией ВИ понимают совокупность нетрадиционных методов вычислений и средств обработки знаний, документооборота, методов выработки и выбора альтернативных вариантов решений, объединенных в целостную технологическую систему для принятия и доведения решений до исполнителей.

В настоящее время считают, что ВИ включает в себя следующие основные методы:

нечеткой логики – методы, основанные на теории нечетких множеств и обеспечивающей эффективные средства математического отражения неопределенности и нечеткости исходной информации, позволяющие построить модель, адекватную исследуемой предметной области;

нейросетевые – методы, использующие обучение, адаптацию, классификацию, системное моделирование и идентификацию систем на основе исходных данных;

генетические – методы, использующие синтез, настройку и оптимизацию исследуемых систем с помощью специальным образом организованного случайного поиска и эволюционного моделирования.

Перечисленные методы считаются основными в ВИ, однако, необходимо заметить, что число новых методов примкнувших к ним в последнее время постоянно расширяется, не являясь строго определенным. Ниже перечислены наиболее значимые из них:

когнитивная компьютерная графика;

нелинейная динамика.

Когнитивная компьютерная графика (ККГ) – компьютерные системы визуализации данных, позволяющие активировать наглядно-образные механизмы мышления ЛПР, облегчающие принятие решения в сложной обстановке или нахождение решения сложной проблемы. Суть концепции ККГ заключается в том, что если на экране дисплея удается визуализировать существенные свойства и отношения между объектами некоторой предметной области (любой степени абстрактности), то такой ККГ – образ, как правило, содержит в себе информацию (на уровне графических деталей компьютерного изображения) о возможных и не всегда заранее известных следствиях этих

свойств и отношений, помогающую проанализировать новые закономерности исследуемой предметной области [Агеев , 2002, Зенкин, 1991].

Необходимо особо отметить, что ККГ в последнее время приобретает настолько большое значение в процессе эффективного способа организации значимой для ЛПР информации, что необходимо провести подробный анализ ее роли в этом процессе.

Накопленный опыт управления показывает, что для принятия обоснованных решений необходима не просто информация (как совокупность сведений, которые могут быть полезными), а высокоорганизованная информация, упорядоченная в определенную систему. Именно высокоорганизованное системное упорядочение информации, объективно и достоверно отражающее сложившуюся обстановку, обеспечивает интенсификацию информационных процессов, облегчающих принятие решения ЛПР.

В качестве эффективных способов организации информации, облегчающих процесс ее использования для выбора варианта решения на этапе выработки замысла, могут выступать различные формы ее представления, степень организации которых соответствует полноте знаний, несущих функционально необходимую и достаточно обоснованную системную информацию об объекте принимаемого решения.

Степень организованности информации

В общем случае, применительно к искусственным системам, **степень организованности информации** включает в себя **структурную, параметрическую и динамическую организованности**. Так, например, при сопоставлении возможных вариантов организованности информационных образов: одномерных, и двухмерных, целесообразно использовать пространственно-структурно-параметрические показатели, которые будут более высокими у двухмерных информационных образов. Для того чтобы достигнуть высокой степени организации информационных образов в информационных системах отображения, передачи, воспроизведения и представления информации необходимо располагать запасами ее пространственных, структурных, параметрических и динамических ресурсов.

Одним из путей получения высокоорганизованной информации и подбора интеллектуальных средств, реализующих эту информацию, является визуализация информационных образов. В свою очередь, выбор рациональных вариантов реализации интеллектуальных средств требует решения главной задачи: как адаптировать выбираемые средства к конкретному ЛПР с его субъективными психофизиологическими данными. Если эта задача не будет хотя бы частично решена, то и главная цель повышения оперативности принятия решения, при резком увеличении информационной трудоемкости его выработки, также не будет достигнута.

Для достижения этой цели, необходимо выяснить, по каким каналам органов чувств ЛПР легче всего воспринимает поступающую к нему информацию. Как известно, наиболее развиты зрительные и слуховые каналы или, как принято, в психологии, каналы зрительной и слуховой модальности по которым человек получает порядка 95-97% информации об окружающем мире. Одно из самых первых оснований для выбора той или иной модальности было предложено Н. Винером (N. Wiener), который рассматривал критерии эффективности этих каналов. По Винеру, критерием является отношение между числом зрительных и слуховых образов на уровне коры головного мозга. Организация различных полей коры головного мозга, связанных со зрением, и площадей коры головного мозга, обслуживающих слух, может быть использована при сравнении эффективности зрения и слуха. Проведенные в работе [Глезер, Цукерман, 1961] опыты показали, что это отношение равно 100:1.

Анализ трудов по теории информации и инженерной психологии [Беляев, Капустян, 1999; Белый и др., 1999] выявил количество информации, которое может быть передано по зрительному и слуховому каналам. При прохождении по нервным волокнам, идущим от глаза и уха к мозгу ЛПР (в среднем 300 имп/с), пропускная способность зрительного канала равна $6 \cdot 10^8$ дв.ед./с, а слухового - $18 \cdot 10^6$ дв.ед./с, что значительно меньше ($6 \cdot 10^8 / 18 \cdot 10^6 = 33{,}33$) количества информации проходящей через зрительный канал. Но сам по себе способ определения количества информации, с учетом лишь проводимости нервного волокна, не принимая во внимание

мыслительную деятельность, доказывает лишь то, что поток информации к ЛПР поступает, в основном по зрительному каналу.

Задачу выбора ЛПР варианта решения на основе восприятия им информации и его мыслительной деятельности, можно рассмотреть с точки зрения гипотезы выдвинутой Ньюэллом [Newell], Саймоном [Simon] и Шоу [Show] [Глезер, Цукерман, 1961] Эта гипотеза, в рамках информационного анализа, объясняет, как представляются объекты мышления в психике ЛПР. В результате проведенного анализа было выяснено, что внутренний образ может быть отнесен к зрительной модальности в том случае, если он по природе своей таков, что может служить точкой приложения, входом для информационных процессов, характерных для обработки визуальных образов. Эти процессы для переработки визуальной информации, применяемые при обработке внутренних образов, могут, по мнению этих исследователей, расцениваться как «мыслительные образы».

Отсюда можно сделать вывод, что при реализации средств интеллектуальной поддержки принятия решения следует ориентироваться на то, чтобы информация, представляемая ЛПР носила бы визуальный характер в виде визуальных информационных образов. Тогда ее восприятие и обработка не встретят противоречий со стороны внутренних образов мыслительных процессов, и приведет к резкому сокращению времени принятия решения.

Задача сжатия информации

Способы визуализации информационных образов для представления информации в средствах интеллектуальной поддержки принятия решений **выполняют еще одну важную задачу – задачу сжатия информации**.

Пусть визуальный информационный образ – это вход нашей системы переработки информации, а показания, снятые человеком и пропущенные через его мыслительный аппарат – выход. Количество информации, переданное от входа к выходу, при прочих равных условиях будет изменяться с изменением характера входа. И это связано не только с изменением его информационного содержания, но и с изменением способа кодирования информационного образа и формы его представления. Допустим, что ЛПР получает информационное сообщение с помощью двух видов визуальных

информационных образов: одномерного, двухмерного. Первый – одномерный визуальный образ, в котором сообщение передается текстом, а слова следуют одно за другим («бегущая строка»). Второй вид визуального образа передает то же самое информационное сообщение с помощью двухмерного представления информации на плоскости: графика, чертежа, пиктограммы, графического образа. Для того чтобы оценить эффективность передачи и восприятия информационного образа необходимо вероятность правильного восприятия P, разделить на время восприятия t (ч) – $Q=P/t$ ($ч^{-1}$). Если вероятность правильного восприятия в обоих случаях равна единице, то время оказывается разным, так как длительность восприятия сообщения в первом и во втором случае разная. Во втором случае, ЛПР достаточно одного взгляда на графический образ, чтобы полностью воспринять информационное сообщение. Так, например, ЛПР, работающему с электронной картой местности (ЭКМ), достаточно одного взгляда на условный знак, чтобы получить информацию, которая состоит из нескольких предложений, описывающих содержательное значение этого знака.

При переходе от одномерного визуального информационного образа к двухмерному, ЛПР группирует первичные элементы (слова и смысловые понятия первичного информационного сообщения), увеличивая число этих элементов в информационной единице, «спрессовывая» их в новый визуальный образ, т.е. производя перекодировку входа с изменением формы представления визуального образа и его информационной емкости. Такая перекодировка требует существования правила перевода одной последовательности в другую и дополнительных затрат времени.

Таким образом, проанализировав психофизиологические факторы ЛПР при принятии решения можно сделать вывод о том, что при рассмотрении альтернативных вариантов решений ЛПР целесообразно использовать такой метод ККГ, который позволял бы визуализировать особенности рассматриваемой задачи с одновременным уменьшением ее размерности. Кроме того, необходимо помнить, что визуальные образы вариантов решения наилучшим образом выполняют функции коммуникативного средства, между различными ЛПР участвующими в процессе выработки решения.

Принятие решения во многих практических задачах связанных с местностью сталкивается с необходимостью получения и анализа весьма

обширной и специфической информации. В основном это сведения об отдельных характеристиках местности (оценки проходимости и свойств местности, анализа дорожной сети и т.п.) и объектах, расположенных на ней.

В настоящее время при выборе варианта решения в задачах связанных с местностью в качестве одного из источника исходных данных в большинстве случаев используют топографические карты, несущие ограниченную информацию. Более полную информацию дают возможность получить различные геоинформационные системы (ГИС), а дополнительную информацию об отдельных участках местности получают в ходе космической, воздушной или наземной разведки.

Известно, что топографические карты и ЭКМ, обладая хорошей наглядностью, не всегда несут необходимую информацию. Именно поэтому ЛПР, пользуясь их основой, наносят на них дополнительную справочную информацию или результаты решения задач, для наиболее вероятных сценариев действий.

Практика подтверждает необходимость дополнения и обобщения части информации, отображаемой топографическими картами и ЭКМ, а также частичную обработку и предварительное получение необходимых для последующего анализа данных, в интересах принятия решения в задачах связанных с местностью. То есть проявляется тенденция сочетания специализированной ГИС с пакетом методик, позволяющих решать конкретные задачи, а также использовать результаты уже решенных задач, для наиболее вероятных сценариев действий.

Анаморфозы

Развитие многих прикладных задач, связанных с пространственно-временным анализом, предполагает не только совершенствование способов отображения географических явлений, но и показ отношений и связей с другими явлениями, особенно в тех случаях, когда их анализируют как системы. Возникает необходимость исследовать изменяющиеся в пространстве характеристики сразу нескольких явлений. Выполнить такой анализ проще, если хотя бы одну из меняющихся в пространстве характеристик полагать равномерно распределенной и на ее фоне анализировать все остальные с ней взаимосвязанные. Для этой цели

прибегают к преобразованию визуального образа, (площадной фигуры) взятого за основу, из евклидовой метрики в условное «усредненное пространство». Под таким преобразованием понимается переход от визуального образа, в основу которого положена обычная метрика, к другому визуальному образу, в основу которого положена метрика рассматриваемого явления. Подобные преобразованные визуальные образы принято называть анаморфозами.

АНАМОРФОЗА — жен., греч. безобразная, но правильно искаженная картина, принимающая в граненом или гнутом зеркале свой вид; безобраза. Толковый словарь Даля. В.И. Даль. 1863 1866.

АНАМОРФОЗА — (греч. ana на, сверх и morfe форма), эффект наложения одного изобразительного мотива на другой, их зрительного слияния.

Среди анаморфоз можно выделить линейные, площадные и объемные (в виде трансформированных диаграмм, обычных или рельефных карт).

Первые линейные анаморфозы появились в середине 20-го века и строились главным образом на основе масштаба времени, так как сложность ориентации во все усложняющемся техногенном пространстве обуславливала выделение **времени** в качестве одного из основных факторов принятия решений. Простейшей линейной анаморфозой, отражающей порядок следования станций и строящейся вручную, является схема Московского метрополитена. В других примерах, физические расстояния от одной начальной точки до всех трансформируемых пунктов исходной карты заменяются условными расстояниями, измеряемыми в единицах времени, стоимости и т. д. К настоящему времени разработан целый спектр методик трансформации масштаба длин [Muller, Honsaker, 1980; Тикунов, Юдин, 1987], успешно применяющихся при построении линейных анаморфоз.

Среди анаморфоз наибольшее распространение получили их площадные разновидности, которые позволяют выравнивать в пространстве какие-либо показатели (например проходимость местности). В этом случае площади изображаемых площадных фигур становятся пропорциональными величинам показателя, закладываемого в основу анаморфозы. При этом от анаморфированных изображений требуется **максимально возможное сохранение их формы**.

Большинство известных анаморфоз, начиная с «картограмм людности» Г. Вихеля [Haack, Wiechel, 1903] и вплоть до 80-х годов 20-го столетия, в

основном строилось вручную. Различными авторами было предложено большое количество способов ручного построения анаморфоз, таких как: механической аналогии [Hunter, Young, 1968; Skoda, Robertson, 1972], электрического [Raspolozenski и др., 1974] и фотомоделирования [Брюханов, Тикунов, 1983].

Приведенные способы построения анаморфоз имеют ряд общих существенных недостатков: зависимость решения от субъективизма авторов, большая трудоемкость подготовительного этапа построения модели, малая точность решения, и использование специального оборудования, которое, в отличие от компьютера, не имеет универсального характера. Поэтому все современные методы построения анаморфоз являются численными и предназначены для реализации на компьютере.

Предложенный способ трансформации ЭКМ из эвклидовой метрики в метрику выбранного показателя, для анализа на его фоне взаимосвязанных с ним характеристик, можно применить следующим образом:

1. На предварительном этапе выработки решения - для формирования библиотеки визуальных образов (альтернативных решений) ЛПР или экспертами.

2. В процессе подготовки варианта решения, в соответствии с замыслом ЛПР, и возможными вариантами развития обстановки.

Метод анаморфирования

Метод анаморфирования для формирования варианта решения рассмотрим на примере связанном с анализом проходимости местности и ее изменениями в ходе различных мероприятий.

Под проходимостью понимают возможность самостоятельного, без проведения инженерных мероприятий, движения машины (машин) данного типа и класса по неподготовленной для этого местности, с учетом расположенных на ней объектов. Количественно проходимость оценивается главным параметром движения – скоростью движения.

С учетом этого, в каждой ячейке предварительно нанесенной на ЭКМ регулярной сетки шага k, рассчитываются значения скоростей движения колесной (гусеничной) техники в заданном направлении, зависящие от целого ряда условий (рельефа местности, грунтов, времени года, залесенности,

водных преград и т. д.) и принять их за показатели анаморфирования каждой ячейки. После применения алгоритма анаморфирования регулярная сетка деформируется пропорционально значениям скоростей движения колесной (гусеничной) техники в каждой ячейке, на ней образуются группы ячеек увеличенной площади (относительно ячеек, показатель анаморфирования которых равен среднему, а площадь S=1 усл. ед.) показывающие направления и участки местности наиболее вероятного передвижения колесной техники, а также группы ячеек уменьшенной площади показывающие места, где передвижение колесной (гусеничной) техники маловероятно или невозможно.

Пусть D – область на плоскости $\boldsymbol{R^2}$ (площадная фигура, построенная на основе выбранного показателя), которая должна быть анаморфирована. Распределение показателя описывается функцией плотности $\rho(z)$, определенной априори на части D *(z = (x, y)* – точка на плоскости $\boldsymbol{R^2}$). Без потери общности можно полагать, что $\rho(z)$ определена на всей плоскости $\boldsymbol{R^2}$. Тогда она является *const* вне области D (например, как среднее значение $\overline{\rho}$ функции $\rho(z)$).

Анаморфоза задается преобразованием h: $\boldsymbol{R^2} \to \boldsymbol{R^2}$ *(h: (x, y)* □ *(u, v))* или двумя функциями двух переменных: *U(x, y) и V(x, y)*,

где $u = U(x, y)$, $v = V(x, y)$. (1)

Эти функции должны быть определены и непрерывны на D.

Коэффициент изменения площади в окрестности точки *(x, y)* преобразованием h равен значению якобиана преобразования h в этой точке

$$J(U,V) = \frac{\partial U}{\partial x} \cdot \frac{\partial V}{\partial y} - \frac{\partial U}{\partial y} \cdot \frac{\partial V}{\partial x}.$$

Поэтому условие того, что преобразование (1) делает величину $\rho(x,y)$ постоянной и равной $\overline{\rho}$, может быть записано как $J(U,V) = \rho(x,y) / \overline{\rho}$.

Таким образом, задача нахождения анаморфозы сводится к задаче решения уравнения

$$\frac{\partial U}{\partial x}\cdot\frac{\partial V}{\partial y}-\frac{\partial U}{\partial y}\cdot\frac{\partial V}{\partial x}=\frac{\rho(x,y)}{\overline{\rho}},$$

(2)

для которого $[U(x,\ y),\ V(x,\ y)]$ определяют взаимно-однозначное преобразование.

Проблемы реализации анаморфоз

Построение анаморфоз на компьютере встречает трудности двух типов. Первые из них связаны с компьютерной реализацией алгоритма. Это задачи преобразования исходных площадных фигур в форму, пригодную для компьютерной обработки (с этой задачей достаточно успешно справляются такие ГИС, как: ArcInfo, MapInfo, «Панорама» и «Интеграция»), численной реализации алгоритма с контролем сохранения взаимной однозначности преобразованных площадных фигур и представления результатов расчетов в виде, удобным для их дальнейшей обработки.

Второй класс трудностей связан с тем, что условие выравнивания заданной плотности не определяет анаморфозу однозначно. Существует бесконечно много преобразований, удовлетворяющих этому условию. Это следует из того, что для двух неизвестных функций $U(x, y)$ и $V(x, y)$, задающих анаморфозу, имеется только одно уравнение (2). Построенное анаморфированное изображение без нарушения постоянства плотности может быть изменено применением любого преобразования, сохраняющего площадь, например:

1) $(u,v)\to(k\cdot u,k^{-1}\cdot v)$ (растяжение вдоль одной из осей и сжатие вдоль другой с тем же коэффициентом);

2) $(u,v)\to(u+f(v),v)$, $(u,v)\to(u,v+g(u))$ (сдвиг горизонтальных и вертикальных прямых вдоль самих себя на различные расстояния).

В качестве требования при выборе анаморфозы можно использовать условие конформности преобразования (1). Конформное преобразование изменяет все расстояния умножением на один и тот же коэффициент, не зависящий от направления (углы между прямыми линиями сохраняются).

Преобразование, которое локально изменяет все расстояния с помощью умножения на $\sqrt{\rho(x,y)}$ единственно, и не зависит от выбора системы координат.

Конформное преобразование с заданным коэффициентом линейного растяжения, равным $\sqrt{\rho(x,y)}$ (или с коэффициентом изменения площадей, равным $\rho(x,y)$) существует не всегда. Условие конформности преобразования (2) может быть записано в виде

$$\begin{cases} \left(\frac{\partial U}{\partial x}\right)^2 + \left(\frac{\partial V}{\partial x}\right)^2 = \rho(x,y) \\ \left(\frac{\partial U}{\partial y}\right)^2 + \left(\frac{\partial V}{\partial y}\right)^2 = \rho(x,y) \\ \frac{\partial U}{\partial x}\cdot\frac{\partial V}{\partial y} + \frac{\partial V}{\partial x}\cdot\frac{\partial U}{\partial y} = 0 \end{cases}.$$

Это условие состоит из трех уравнений относительно двух неизвестных функций *U(x, y) и V(x, y)*. Обычно такая система не имеет решений. Для существования такого преобразования необходимо, чтобы функция плотности $\sqrt{\rho(x,y)}$ удовлетворяла уравнению

$$\Delta \ln \rho \equiv \frac{\partial}{\partial x}\left(\frac{\partial \rho / \partial x}{\rho}\right) + \frac{\partial}{\partial y}\left(\frac{\partial \rho / \partial y}{\rho}\right) = 0,$$

где $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ - лапласиан.

Это уравнение означает что кривизна, определенная метрикой $\rho(x,y)(dx^2 + dy^2)$, должна быть равна нулю.

Таким образом, возникает задача поиска среди различных преобразований того, которое инвариантно к любой замене декартовой системы координат и конформно.

Численные методы построения анаморфоз

На сегодняшний день существует достаточно большое количество численных методов построения анаморфоз, каждый из которых обладает рядом достоинств и недостатков.

Один из первых численных методов построения анаморфоз был предложен У. Тоблером [Tobler, 1979]. Достоинства этого алгоритма состоят в его простоте и отсутствии преимуществ одних ячеек по отношению к другим. Очевидные недостатки: результат существенно зависит от выбора направлений координатных осей; итерационный процесс обладает медленной сходимостью.

Метод треугольников разработан в Московском государственном университете им. Ломоносова в 1983 году [Петров и др., 1983]. Очевидные достоинства метода: простота, независимость от выбора какой-либо системы координат, сохранение топологического подобия с оригиналом. Основной недостаток метода состоит в существенной зависимости результата от случайных выборов, при его реализации: в порядке обработки вершин, в случайном поиске нового положения выбранной вершины.

Алгоритм построения анаморфированных изображений лаборатории Лоуренс Беркли [Selvin и др., 1984]. Этот метод позволяет получить анаморфозы хорошего качества и его реализация относительно проста. Основные недостатки метода состоят в том, что:

- окончательный результат существенно зависит от порядка, в котором берутся ячейки;

- ошибки, допущенные на каждом шаге, не исправляются позже и накапливаются.

Анализируя недостатки приведенных алгоритмов, можно сделать выводы о путях их улучшения:

- необходимо, чтобы на каждом шаге на сдвиг точек (в том числе – вершин) в той или иной степени влияли все ячейки разбиения. Т.е. на каждом шаге сдвиг точки должен быть равен векторной сумме сдвигов, от влияния отдельных ячеек.

Во-вторых, влияние ячейки на точку должно состоять в перемещении этой точки вдоль прямой, соединяющей ее с некоторой точкой ячейки (например, с ее центром масс). Это условие связано с требованием инвариантности алгоритма по отношению к выбору системы координат.

В-третьих, перемещение точки под влиянием ячейки должно убывать с увеличением расстояния от этой ячейки.

Первый алгоритм такого рода был предложен в [Dougenik и др.,1985]. В нем рассмотрена ситуация, когда на бесконечной территории имеется ограниченная часть (без потери общности можно предположить, что эта часть имеет форму круга), на которой плотность распределения показателя отличается от средней по всей плоскости (на остальной части территории плотность равна средней). При естественном анаморфировании сдвиги точек меньше для тех из них, которые расположены дальше от указанной части территории. В [Гусейн-Заде, Тикунов, 1999] показано, что при таком определении влияния ячейки на вершины, которое уменьшается с увеличением расстояния, рассматриваемый алгоритм будет сходиться быстрее. В качестве ячеек используются любые связные области произвольной формы.

Этот алгоритм можно описать следующим образом. На каждом шаге вычисляется векторная сумма влияний центров ячеек на вершины и центры многоугольных ячеек, составляющих визуальный образ, которые перемещаются в соответствии с полученными векторами сдвигов. Для полученной конфигурации вычисляются новые площади ячеек. Итерационный процесс прекращается, когда все относительные отклонения площадей ячеек становятся меньше заданной величины ε.

В общем случае очевидными достоинствами анаморфозы электронной карты местности на основе выбранного показателя являются:

наглядность – выявление скрытых закономерностей поведения параметров, зависящих от выбранного показателя анаморфирования;

быстрый визуальный анализ – анаморфирование исходной матрицы выбранного показателя, по любому его допустимому значению позволяет принимать решения в направлениях и областях с данным значением показателя;

возможность построения сценариев развития ситуации на основе анализа динамической анаморфозы, учитывающей выполнение задач связанных с быстрой эволюцией выбранного показателя;

возможность принятия решения с учетом комплексного показателя анаморфирования (например, скорости движения колесной техники и степени заражения местности и т.д.) рис.1-3;

уменьшение пространства принятия решения на количество взятых показателей анаморфирования.

Гибридные интеллектуальные системы

Основные законы гибридного интеллекта и основные методы гибридизации. Общий подход к построению гибридной интеллектуальной системы. Принципы построения гибридных интеллектуальных систем.

Основные законы гибридного интеллекта и основные методы гибридизации.

В теории управления гибридная парадигма (гибрид - организм, возникающий в результате гибридизации, а гибридизация - соединение в одном организме разнородных наследственностей) зародилась в середине 60-х годов и проявилась в разработке интегрированных моделей и методов. Переход к генетической парадигме обозначил отказ от взгляда на объект исследования как простую, однородную сущность и привел к принятию мировоззрения сложного, составного, неоднородного объекта, что позволило подняться на качественно новый уровень анализа известных и синтеза новых объектов, обладающих не существовавшими ранее полезными свойствами. Одной из возможных форм такой интеграции объектов являются гибридные интеллектуальные системы (ГиИС).

Семантика термина – «гибридная система» раскрывается через понятия «гибрид» и «система». Понятие «гибрид» содержит в себе признаки системы – состав, структуру, порядок, эмерджентность и др. Поэтому сочетание понятий «гибрид» и «система» говорит о том, что родовой смысл этого сочетания выражает понятие «система», а видовое отличие от других систем выражено в понятии «гибрида». Это видовое отличие состоит, прежде всего, в методе его получения – через гибридизацию и подчеркивает неоднородность, многоаспектность отображения в одной сущности – модели различных признаков одного или нескольких оригиналов. Это могут быть такие признаки, как «дискретность и непрерывность», «абстрактность и конкретность», «интеллект и рутина», «эвристика и закон», «алгоритм и формула», «слабость и сила» и др.

Если, например, в качестве признаков-компонент использовать различные виды моделей, методов, знаний, естественные интеллекты, то можно ввести понятия гибридных моделей, гибридных методов, гибридных знаний.

Обобщающий эти понятия термин гибридного интеллекта ввел В.Ф. Венда в 1990, чтобы выдвинуть контрпонятие искусственному интеллекту; подчеркнуть эволюционно-историческое значение взаимодействия естественных интеллектов в природе, обществе и технике; показать диалектическую взаимосвязь искусственного и гибридного интеллекта, которые переходят один в другой.

В.Ф.Венда сформулировал **три закона гибридного интеллекта**.

Закон взаимной адаптации. Синтез и динамика развития любого гибрида-метода - процесс взаимной адаптации компонент гибрида.

Закон утверждает, что необходимое и достаточное условие возникновения и развития гибрида - наличие процессов внутренней (между компонентами гибрида) и внешней (гибрида с внешней средой) взаимной адаптации. Структура гибрида - отображение определенной закономерности процесса взаимной адаптации его внутренних компонентов. Эта структура устойчива, если взаимоадаптация гибрида и внешней среды характеризируется состоянием, при котором эффективность решения задач гибридом лежит в заданном интервале.

Закон дискретных рядов структур. Любой гибрид может быть реализован посредством одной из дискретного ряда его возможных структур.

Закон утверждает, что существует некоторый метод получения одной структуры, входящей в дискретный ряд, из другой структуры этого ряда. Кроме этого, в таком ряду должны существовать целевые структуры, позволяющие сделать гибридизацию целенаправленной.

Закон трансформации. Трансформация одной структуры гибрида в другую может происходить только через общие для обеих структур знания.

Закон описывает образование новых знаний их интерференцию между собой, построение умозаключений, а также возникновение и роль ассоциаций в трансформации и связывании образов и мыслей. Любые новые знания могут быть получены только путем перехода от одного знания к новому, взаимосвязанному, ассоциированному с предыдущим.

В соответствии с законом трансформации новая структура не может быть порождена как таковая, и возникает только на базе предыдущей структуры. При этом сохраняется достигнутая при старой структуре взаимная адаптация части компонентов, которая соответствует новой структуре. Закон трансформации показывает роль фундаментальных, формализованных знаний, которые постепенно развиваясь играют роль того общего, что связывает вновь появляющееся эвристическое знание.

В теории систем приведенные законы описывают гибридизацию как процесс создания гибридных методов, гибридных моделей, гибридных алгоритмов и гибридных программ.

Рассмотрим следующие методы гибридизации.

Метод H1 (рис. 1). Метод основан на том, что одна и также задача может быть решена несколькими известными автономными методами, а вычисления, начиная с некоторого пункта алгоритма, могут быть продолжены тем или иным методом.

В структуре алгоритма выделены четыре фрагмента. Фрагмент 1 выполняет предобработку исходных данных общую для всех используемых методов и готовит информацию для принятия решений о том, в соответствии с каким методом будут продолжаться вычисления во фрагменте 3. Принятие решения выполняет фрагмент 2 – гибридизатор, который должен располагать знаниями достаточными для принятия решений о том, как на очередном шаге вычислений сделать выбор метода для продолжения процесса решения задачи.

Рис. 1. Структура метода гибридизации H1

Фрагмент 4 выполняет функции оценки эффективности принятого гибридизатором решения, корректировки, в случае необходимости, знаний гибридизатора, продолжения решения и определения ситуации завершения работы алгоритма для выдачи результатов решения задачи.

Принципиальной особенностью метода H1 является то, что решение принимает блок, встроенный в алгоритм решения задачи. Решение задачи продолжает один метод, а гибрид после завершения работы алгоритма перестает существовать. Это дает основание говорить о гибридном алгоритме решения задачи, а формируемая в ходе работы алгоритма последовательность применяемых методов может рассматриваться как гибридный метод решения задачи.

Метод H2 (рис. 2).

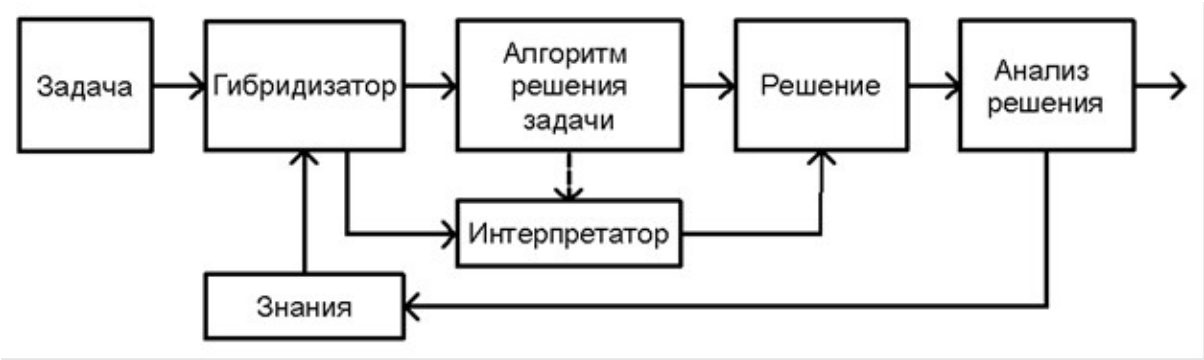

Рис. 2. Структура метода гибридизации H2

В отличие от метода H1 данный подход предполагает, что метод решения исходной неоднородной задачи неизвестен, однако она предварительно была редуцирована на некоторое количество однородных задач, для каждой из которых известно некоторое множество методов ее решения. Есть также знания об областях релевантности автономных методов. В этом случае гибридизатор, используя декомпозицию неоднородной задачи, множество автономных моделей, знания об областях релевантности, а также знания ЛПР, строит алгоритм решения задачи как структуру над алгоритмами решения однородных задач, обрабатываемую интерпретатором. После получения интерпретатором решения задачи, оно анализируется ЛПР и принимается. В противном случае знания гибридизатора корректируются, может быть построен новый алгоритм, с выполнением повторного решения неоднородной задачи и

так далее, до тех пор, пока итерационный процесс, по мнению ЛПР, может быть завершен и найдено решение исходной задачи.

При использовании рассматриваемого метода символьная структура, построенная гибридизатором, может рассматриваться как гибридная модель и как гибридный метод решения неоднородной задачи.

Особенность метода Н2 - ориентация на неоднородные задачи, конструирование символьного эквивалента гибрида, который может быть сохранен, отредактирован и интерпретирован в любое время в зависимости от ситуации решения задачи. Важным моментом метода Н2 является использование для решения задачи знаний ЛПР и возможность коррекции знаний гибридизатора.

Метод Н2 может применяться в двух вариантах в зависимости от того, какие знания будут использоваться в гибридизаторе.

В первом варианте используются модели, построенные в полном соответствии с методом без каких-либо его изменений.

Во втором случае метод может быть изменен. Например, некоторая часть процедуры метода может быть заменена на процедуру, взятую из другого метода.

Гибридизацию первого вида в литературе называют крупнозернистой, а второго – мелкозернистой.

Обобщая вышесказанное, можно сделать вывод о том, что гибридизация это сложный и трудоемкий процесс, требующий, во-первых - привлечения широкого спектра знаний ЛПР и экспертов о предметной области, задачах, методах их решения и во-вторых выполнения сложной переработки информации и организации большого количества экспериментов.

В связи с резким расширением в последние годы круга решаемых с помощью ГиИС задач управления актуальным становится их разработка и применение, как инструмента решения задач такого типа.

Общий подход к построению гибридной интеллектуальной системы

Гибридные интеллектуальные системы были анонсированы в 1994-1995 г.г. в работах профессора L. Medsker 1994, 1995 из Вашингтонского университета и, по существу, совпадают с «интеллектуальными гибридными системами» [Intelligent Hybrid Systems, 1995, 1997], «гибридными

интегрированными системами» [HYBRID and Integrated System-Special Interest Group Home Page, 2001], «гибридными информационными системами» [Hybrid Intelligent Systems Group (HIS), 2000], «гибридными интеллектуальными адаптивными системами» [Kasabov, Kozma, 1998]. Появление термина «интеллектуальные» в сочетании с гибридными системами обозначает выход за рамки аналитических и статистических знаний, с которыми работают ГС и применение которых в теории управления отработано многолетней практикой, их интеграцию с другими видами знаний, используемых в ИИ - экспертными системами, генетическими алгоритмами, искусственными нейронными сетями, нечеткими системами, кейс - системами и др. Тем самым обеспечивается как многоаспектность исследования, выработки, оценки и принятия решений, так и расширяется класс решаемых с помощью гибридов задач.

Бурному развитию ГиИС в середине девяностых годов способствовали успехи нечетких систем и ИНС, появление программных продуктов, поддерживающих разработку и создание автономных моделей, развитие средств и технологий информатики (использование больших объемов памяти, увеличение быстродействия процессоров, применение компьютерных сетей, стандартизация межпрограммного обмена и операционных систем). Эти успехи привели к многообразию терминологии, возникновению множества школ, исследующих узкие области применения ГиИС.

За последние 15 лет в России и за рубежом опубликовано много работ, посвященных ГиИС, в которых введены различные формализмы. В качестве примера можно привести работу, где, развивая идеи Венда [Венда, 1990] С.В.Астанин, и В.Г.Захаревич [Астанин, Захаревич, 1997], сформулировали модель ГиИС следующим образом:

$$m =< R, C, P, S, \varphi, f, \psi, \chi, \lambda >$$

где: S- состояние внутренней среды; $\varphi : R \times C \to V(R)$ - функция, ставящая в соответствие набору оценок ресурсов по каждой цели оценку системного ресурса; $f : V(R(t)) \times P_{i,l} \to V(R(t+1))$ - функция, ставящая в соответствие текущей оценки системного ресурса и управляющему решению оценку исхода применения управляющего решения;

$\psi : V(R(t+1)) \times P_{i,l} \to P_{i,g}$ - функция выбора управляющего решения в зависимости от оценки исходов; $\chi : P_{i,g} \times S \to P_{i,g}^k$ - функция выбора процедуры реализации управляющего решения $P_{i,g}$ в зависимости от состояния внешней среды; $\lambda : P_{i,l} \times S \to [0,1]$ - функция оценки управляющего решения в зависимости от состояния внутренней среды.

Анализируя приведенную модель можно сделать вывод о том, что в ней можно выделить две информационные составляющие - информацию, поступающую от пользователя и различных уровней иерархии (I_n) и информацию, передаваемую ЛПР ($I_Э$).

Информация I_n состоит из совокупности целей, реализуемых пользователем и уровнями иерархии - $C = \{c_i\} | i = 1,...,n$ и оценок ресурсов $R = \{r_i\}$, предназначенных для достижения цели c_i. Информация $I_Э$ состоит из программ P_i достижения c_i. В общем случае P_i представляет собой набор управляющих воздействий $P_{i,l} | l = 1,...,d$, применение каждого из которых определяется текущей ситуацией на множестве R и оценкой исхода. Каждое управляющее решение $P_{i,l}$ может быть реализовано одной из процедур $P_{i,l}^k | k = 1,...,w$. Изменение оценок ресурсов зависит как от выбора и реализации управляющего решения $P_{i,l}$, так и от воздействий внешней среды. Выбор процедуры реализации $P_{i,l}^k$ определяется состоянием внутренней среды (быстродействием, памятью и т.п.).

Прежде, чем приступить к рассмотрению общего подхода к построению ГиИС, применяемых при решении задач ИО АО, рассмотрим классификацию неоднородных задач, предложенную в [Kolesnikov, Yashin, 1999, 2000, 2001] и

рассматривающую задачи с позиций системного анализа с использованием понятий «гомогенной» - однородной и «гетерогенной» - неоднородной систем.

Гомогенная система (греч.- однородный) это физическая система, не содержащая частей, отличающихся по составу или свойствам и отделенных друг от друга поверхностями раздела.

Гетерогенная система (греч.- разнородный) – понимается как неоднородная физико-химическая система, состоящая из различных по физическим свойствам или химическому составу частей (различных фаз). Одна фаза отделена от другой поверхностью раздела, на которой скачком изменяется одно или несколько свойств системы: состав, поле, структура и др.

Применение этих двух терминов к понятию «задача» приводит к существованию двух моделей:

1) модель «однородная (гомогенная) задача»;

2) модель «неоднородная (гетерогенная)» задача.

Использование свойств однородности и неоднородности, широко наблюдаемых в деятельности различных ДЛ ОУ, применительно к задачам управления, уже достаточно давно используется многими учеными. В частности, в [Александров, 1975] подчеркнуто, что особо сложные задачи в условиях неполной, недостаточной оперативной информации возникают в военной области, в экономике, исследованиях космоса и других областях - везде, где имеют дело с функционированием систем, зависящих от многих разнородных переменных. В работах Ларичева 1979, 2002 г.г. отмечается, что среди множества проблем выделяются проблемы уникального выбора, для которых характерны: уникальность и неповторимость ситуации выбора; сложный для оценки характер рассматриваемых проблем; недостаточная определенность последствий принимаемых решений; наличие совокупности разнородных факторов, которую следует принимать во внимание; наличие лица или группы, ответственных за принятие решений. В статье [Забежайло, 1998] подчеркиваются характерные особенности ставшего актуальным в последнее время нового класса задач управления и поддержки принятия решений: обработка больших объемов накапливаемой информации; принятие решений, в том числе, в режиме реального времени; манипулирование разнородными по своему характеру данными и др. В [Нильсон, 1985] отмечалось, что «знание» как цель исследования - слишком обширное и неоднородное понятие, и попытки решать основанные на знаниях задачи в общем виде - преждевременны. В статье [Гладун, 2001] подчеркнуто, что знания разных типов объединены в цельную иерархическую, неоднородную структуру - сеть понятий, а в [Емельянов, Зафиров, 2000] разрабатываются гетерогенные подсистемы в составе гибридной системы для решения практических задач. В [Тарасов, 1997, 1998 в] подчеркнуто, что многие задачи в современном обществе неоднородны и рас-пределены: а) в пространстве;

б) в функциональном плане, поскольку ни один человек не может создать сложную систему в одиночку.

На рис. 3 представлены понятия неоднородной задачи и гибридной интеллектуальной системы по [Колесников, 2001; Колесников, Кириков, 2007].

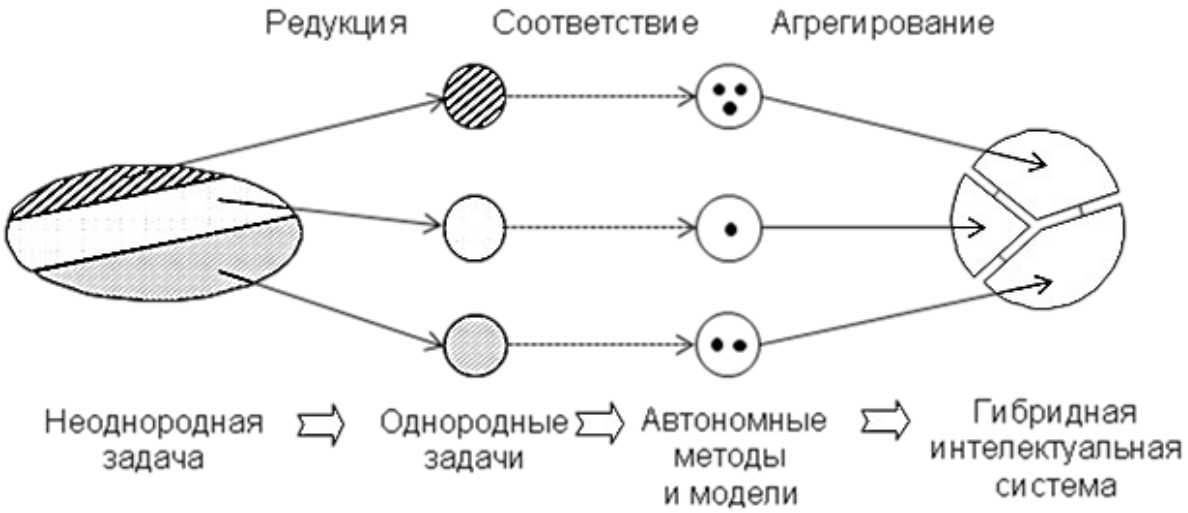

Рис. 3. Соответствие понятий неоднородной задачи и гибридной интеллектуальной системы

В качестве примера на рис. 3 неоднородная задача редуцирована (редукция - упрощение по функциональному признаку). Цель редукции - выяснить, какие области однородных параметров существуют в задаче - системе, и сформулировать внутри этих областей другие более простые, одноаспектные (однородные) задачи, которые уже нельзя соотнести с моделью «неоднородная задача») на три однородные задачи (связи между ними опущены). Это подчеркивает, многократно высказываемое многими учеными мнение, что гибридные интеллектуальные системы используются там, где характеристики проблемной среды отвечают требованиям, которые находятся вне границ или возможностей любого автономного метода или технологии.

Для любой неоднородной задачи трудно сформулировать автономное представление или технологию, которые адекватны всем требованиям. Поэтому разработка гибридной системы предполагает редукцию задач из предметной области и предоставляет возможность интегрировать технологии, чтобы решить конкретные неоднородные задачи. Таким образом, гибридные

интеллектуальны системы могут решать задачи большой сложности. Типичный пример таких задач - задачи принятия решения.

После снижения сложности неоднородной задачи методом редукции должно быть установлено соответствие однородных задач и автономных методов их решения. При этом гибрид возникает только в том случае, если в результате выбора в состав ГиИС будут включены методы, приводящие к разнородным моделям. Желательно и разнообразие моделей для решения одной и той же однородной задачи (многообразие моделей решения одной и той же однородной задачи показано на рис. 3 множествами черных точек).

Таким образом, общий подход к построению ГиИС, применяемых при решении задач может быть представлен следующим образом.

ГиИС, применяемая при решении задач синтезируется агрегированием на множестве автономных моделей с помощью отношений интеграции. Отношения интеграции должны заменить отношения редукции и отношения, связывающие однородные задачи. В результате агрегирования синтезируется, подбирается структура релевантная неоднородностям задачи, которая может рассматриваться как метод ее решения. Метод можно динамично изменять в зависимости от результатов анализа неоднородной задачи, что приводит к неоднократному повторению редукции, установления соответствия и агрегированию в цикле «анализ неоднородной задачи - синтез ГиИС».

Классификация гибридных интеллектуальных систем

Анализ большого количества существующих на сегодняшний день позволяет сделать вывод о том, что все они, имея много общего, идентифицируют следующие пять классов стратегий разработки ГиИС (рис. 4): **автономные и трансформационные модели, слабосвязанные, сильносвязанные и полностью интегрированные модели**.

Остановимся более подробно на рассмотрении пяти классов моделей, приведенных на рис. 4.

Автономные модели приложений ГиИС содержат независимые программные компоненты, реализующие обработку информации на моделях с использованием методов вычислительного интеллекта.

Несмотря на очевидную вырожденность интеграции знаний в этом случае, разработка автономных моделей актуальна и может иметь несколько целей.

Во-первых, такие модели предоставляют способ сравнения возможностей решения задачи двумя или более различными методами. Выполнение моделей параллельно компенсирует (аппроксимирует) отсутствие явной интеграции на уровне информационных технологий за счет когнитивной деятельности ЛПР. Этот эффект можно назвать «когнитивным гибридом». Он возникает в памяти ЛПР как результат визуального восприятия параллельного выполнения программ, его анализа и обобщения. Устойчивость таких «гибридов-образов» требует дополнительного исследования.

Во-вторых, последовательная реализация двух и более автономных моделей может подтвердить или опровергнуть правильность ранее разработанного процесса обработки информации. В итоге, новая автономная модель для решения задачи верифицирует уже созданное приложение и приводит к более адекватным моделям.

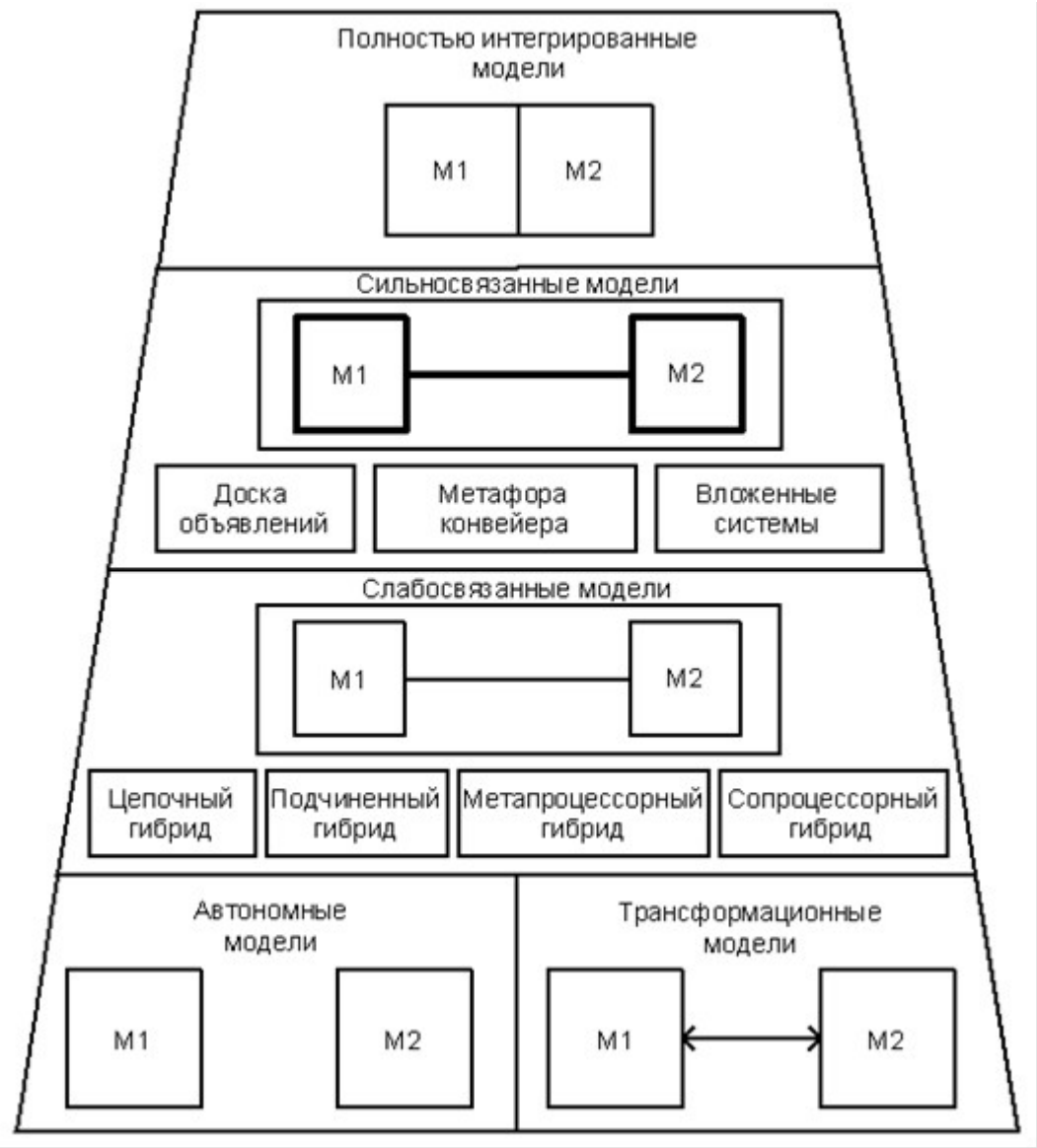

Рис. 4. Классификация гибридных интеллектуальных систем

В-третьих, автономные модели могут использоваться для быстрого создания начального прототипа, после чего разрабатывается собственно ГиИС. Эффективность автономных моделей заключается в простоте разработки и применении в этом процессе коммерчески доступных программ. С другой стороны, такие модели могут усилить автономный метод, который плохо гибридизируется с другими методами.

Автономные модели имеют и существенный недостаток – ни одна из моделей не может помочь другой в ситуации обновления информации – все должны модифицироваться одновременно. Однако, не вызывает сомнения тот

факт, что ГиИС должны обеспечивать и возможность автономного моделирования.

Трансформационные модели похожи на автономные, так как конечный результат их разработки - это независимая, невзаимодействующая с другими частями модель. Основное отличие состоит в том, что такая модель начинает работать как система, использующая один автономный метод, а заканчивает как система, использующая другой автономный метод.

Трансформационные модели дают **несколько преимуществ**. Они быстро создаются и требуют меньших затрат, поскольку эксплуатируется только одна модель, а окончательный метод наилучшим образом адаптирует результаты к окружению. Однако существуют проблемы создания инструмента автоматического преобразования одной модели в другую и существенная модификация модели, сравнимая по объему с разработкой новой модели.

Слабосвязанные модели являются первой реальной формой интеграции, когда приложение разбивается на отдельные компоненты, связываемые через файлы данных. Классификация таких моделей рассмотрена ниже.

Цепочный гибрид - использует как составные части две функционально завершенные компоненты, одна из которых главный процессор, а другая пре- или постпроцессор.

Подчиненный гибрид - также использует как составные части, функционально завершенные компоненты, однако в этом случае одна из них, подчиненная, включена в другую, действующую как главный решатель задачи.

Метапроцессорный гибрид - включает как составные части один метапроцессор и функциональных компонент.

Сопроцессорный гибрид использует при решении задачи компоненты, выступающие как равные партнеры. Каждая может передавать информацию каждой и, взаимодействуя, обрабатывать подзадачи одной и той же задачи.

Если сравнить рассмотренные слабосвязанные модели с другими более интегрированными приложениями, то они проще для разработки и допускают применение коммерчески доступных программ, значительно снижающих объем программирования. Время работы таких моделей сокращается вследствие простоты интерфейсов файлов данных, однако, увеличивается цена коммуникации и уменьшается их производительность. Кроме того, в связи с

появлением нескольких альтернативных вариантов решения одной и той же задачи, необходимо решать проблему выбора одной их них.

Классы слабо- и сильносвязанных гибридов имеют значительное перекрытие. Однако сильносвязанные модели используют обмен информацией через резидентные структуры памяти, в отличие от обмена через внешние файлы данных в слабосвязанных моделях. Это улучшает интерактивные возможности и дает более высокую производительность. Сильносвязанные модели могут функционировать в тех же формах, что и слабосвязанные, однако их пре-, пост - и сопроцессорные варианты быстрее. При их разработке используются следящие методы: «доска объявлений», «метафора конвейера» и «вложенные» системы.

Метод «доска объявлений» имитирует группу специалистов - источников знаний, решающих задачу и использующих доску как рабочее место для выработки решения. Никакой источник знаний не требует других источников знаний для решения задачи, может быть добавлен, удален и изменен, не влияя на другие. Источники знаний активно не наблюдают доску, а запускаются в ответ на события. Доска знает, какой тип события ищет каждый источник. Она исследует события и активирует источник всякий раз, когда этот тип события происходит. Системы доски эффективны для многошагового решения сложной задачи. Они работают, используя метод сравнений, и исследуют наиболее эффективные маршруты в решении задачи.

Метод «метафора конвейера» в некотором смысле противоположный «доске объявлений». Она исходит из того, что источники знаний - решатели статичны и размещены в некоторых позициях. Относительно решателей, как по ленте конвейера, перемещаются динамические объекты: цели, данные, знания, планы, задачи и др., которые символизируют работу группы курьеров, обслуживающих решатели и переносящих информацию для коллективного решения проблемы. Курьеры создаются генераторами, циркулируют по сети решателей и исчезают в терминаторах. В сети есть и «помощники» для вспомогательных действий с курьерами: управления движением, распознавания, проверки и др. Функционированием сети-конвейера, созданной разработчиком, управляет алгоритм, моделирующий обработку текущих и будущих событий. Очевидное преимущество такого подхода - достаточно

общий характер, простота представления и модификации связей решателей, естественность и прозрачность метафоры, параллельность работы сегментов.

«Вложенные» системы» - третий вариант архитектуры сильносвязанных моделей, использующей модули одного метода для помощи в управлении функционированием других моделей.

Сильносвязанные ГиИС, как показывает практика их разработки, имеют низкие коммуникационные затраты и более высокую производительность по сравнению со слабосвязанными моделями. Тем не менее, эти модели имеют три принципиальных ограничения:

- сложность их разработки и поддержки возрастает из-за наличия внешнего интерфейса данных;
- их сильная связанность страдает от излишнего накопления данных;
- проверка их адекватности затруднена, особенно для «вложенных» систем.

Рассмотренные выше слабо- и сильносвязанные модели в силу того, что их состав и структура во многом зависят от решаемой задачи, принято называть функциональными гибридами. Именно функциональные ГиИС релевантны понятию неоднородной задачи и, следовательно, пара «неоднородная задача – функциональная ГиИС» в наибольшей степени соответствует схеме на рис. 3.

Полностью интегрированные модели - совместно используют общие структуры данных и представления знаний, а взаимосвязь между компонентами достигается посредством двойственной природы структур.

Преимущества полной интеграции заключается в увеличении надежности, скорости обработки данных, адаптации, обобщении, снижении шума, аргументации и логической дедукции т.е. всего того, чего в сумме не найти ни в одном автономном.

На основе приведенной классификации строится большое количество разнообразных архитектур ГиИС, однако наибольший интерес, в свете проводимого исследования по решению задач поддержки принятия решения, представляют модульные ГиИС, которые рассмотрим подробнее.

Модульные ГиИС обладают иерархичностью конфигурации, которая характеризует сложность потоков информации между модулями. Так, например, последовательный поток определяет, что один процесс должен быть

завершен, прежде чем данные могут быть переданы в другой модуль, а параллельный - может потребовать одновременной обработки данных или даже обратной связи между модулями. В зависимости от задачи архитектура модульной системы может быть различной и усложняться по мере увеличения сложности задачи.

По мере развития технологии гибридизации, начинают разрабатываться все более сложные архитектуры. В них сложность ГиИС может измеряться в терминах потоков информации между модулями, которые могут быть однонаправленными и двунаправленными, а также степенью связанности модулей: слабой, сильной и слоистой.

В зависимости от требований обработки, ГиИС может иметь различные архитектуры, наиболее часто встречающиеся из которых показаны на рис. 5.

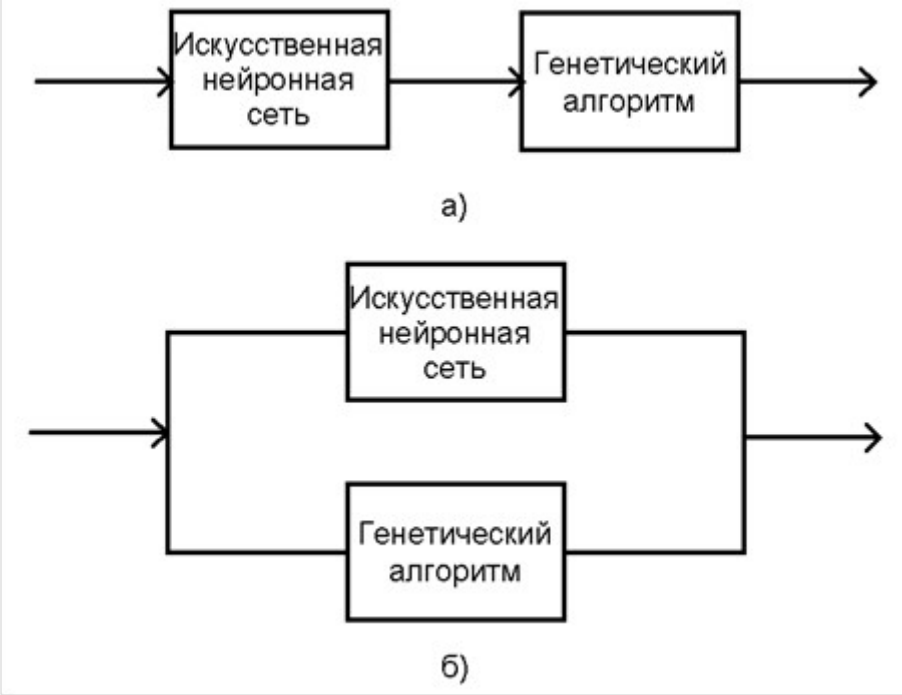

Рис. 5. Модульная конфигурация ГиИС, а) последовательная архитектура,
б) параллельная архитектура

Главной характеристикой последовательной архитектуры (см. рис. 5 а) является последовательная обработка данных, когда один модуль действует как препроцессор извлечения, обработки и представления данных в форме удобной для обработки следующим модулем – постпроцессором.

В случае параллельной архитектуры архитектуры (см. рис. 5 б) данные обоих методов являются общими. Преимуществом такого подхода является повышение достоверности результатов и доверия к ним у ЛПР. Другая возможность параллельной обработки возникает, когда модули работают на разных данных, а результаты их работы комбинируются. Возможны также и другие варианты, когда один из модулей находится в обратной связи и может управлять другим модулем.

Связывание модулей выражает уровень коммуникационной деятельности между модулями, возможности и качественные характеристики обмена информацией между ними. При этом можно говорить о пассивном связывании, при котором методы работают практически автономно, а коммуникация между ними выполняется через файлы. После того как первый модуль отработал, результаты его работы записываются в файл, читаемый другим модулем. Пассивное связывание не требует утонченного, с подтверждением связи управления, для синхронизации модулей. Потоки в этом случае однонаправленные и все данные сохраняются в результате однократных действий, т.е. записи в файл.

Системы с активным связыванием – более сложные, чем пассивные модели, поскольку коммуникация осуществляется с помощью средств разделения память/данные и дополнительных усилий на синхронизацию моделей. Межмодульная коммуникация через разделенные структуры данных создает предпосылки быстрой работы в реальном времени, но требует более сложной передачи данных. Коммуникация может быть двунаправленной, делая возможными обратные связи между модулями. Использование обратных связей позволяет динамически изменять поведение ГиИС, что актуально в задачах управления, где актуальна работа в условиях изменяющейся внешней среды.

Слоистое связывание осуществляется на мелкозернистом уровне, когда модули высоко активны и в процессе обработки информации многократно вызываются в качестве функций. Использование обратных связей играет в таких конфигурациях еще более значимую роль. Слоистые системы предоставляют широкие возможности для взаимодействия модулей, для чего между ними организуется и выполняется специальный коммуникационный протокол. Он управляет разнообразием межмодульных команд и обменом информации в режимах последовательной или параллельной работы модулей.

Принципы построения гибридных интеллектуальных систем

Как было показано в предыдущих разделах гибридизация методов вычислительного интеллекта при решении задач инженерного обеспечения армейской операции является чрезвычайно сложным процессом, который интегрирует в себе многоэтапную работу с информацией. Эти этапы включают в себя работу:

- со сложными субъективными сущностями - задачами, которые выступают в качестве оригинала;
- с широким спектром разнообразных методов моделирования - прототипами, выступающими в качестве носителей первичных родительских признаков, которые затем комбинируются в гибридах;
- с результатами гибридизации, т.е. теми целями, которые ставят и стремятся достичь разработчики ГиИС

Носитель задачи - субъект (ЛПР, эксперт) - зачастую не в состоянии без специальных приемов понять поставленную перед ним задачу, определить границы, отделяющие ее от других задач, осмыслить ситуацию или условия, в которых решается данная задача, выбрать или выработать метод ее решения. Все это приводит к тому, что казавшийся относительно простым и выполнимым процесс постановки задачи, обработки информации и управления, объединяясь с извлечением экспертных знаний, становится непреодолимым обстоятельством на начальной стадии проектов разработки ГиИС.

Помимо этого, сложность гибридизации заключается еще и в том, что:

- несмотря на первые результаты [Аверкин и др., 1990; Гель-фандбейн и др., 1991], информатика до сих пор не выработала взгляда на низкоуровневое представление метода, как делимой и редуцируемой на составные части сущности, не выявила закономерностей, которым подчиняется наследование методами-потомками, конструируемыми из родительского «генетического материала», тех или иных существенных, с точки зрения цели гибридизации, первичных признаков;
- в результате ее проведения создается чрезвычайно сложный программный продукт. Вследствие этого программные проекты часто прерываются, выходят за рамки сроков и бюджета, приводят к некачественным

результатам. Разработка программного обеспечения ГиИС, на сегодняшний день, по-прежнему искусство, а не наука.

Непредсказуемость гибридизации объясняется тем, что часто допускаемые ошибки при разработке моделей с использованием традиционных математических методов, методов теории принятия решений, исследования операций и ВИ переходят и в гибриды. Поэтому потомки-методы могут оказаться потенциально «слабее», чем методы-родители. Более того, на сегодняшний день отсутствуют оценки вычислительной сложности гибридных алгоритмов и устойчивости решений, вырабатываемых в системах управления с гибридными моделями, а производительность ГиИС в ряде случаев потенциально ниже, по сравнению с производительностью интеллектуальных систем, работающих в режиме эксплуатации автономных моделей.

Хотя попытки выработки требований к любой символьно - коннекционисткой гибридной модели небольшой сложности были предприняты еще в 90-х годах, они не привели к выработке принципов, ее построения, т.к. рассматривали гибриды в отрыве от решения практических задач.

Обобщая содержание многих работ можно сформулировать несколько принципов, которые должны лежать в основе методологии и технологии разработки ГиИС. Руководствуясь этими принципами при гибридизации, можно полагать, что конечный результат, т.е. ГиИС, будет построена с меньшими трудозатратами и содержать меньше ошибок, чем в случае игнорирования предлагаемых принципов.

1. **Принцип системного анализа сложной задачи**. Решению сложной задачи должен предшествовать системный анализ ее свойств, состава и структуры, что позволяет сделать более отчетливыми границы однородных областей и подобрать релевантные этим областям автономные методы. Вывод о применении для решения задачи ГиИС делается по результатам ее системного анализа.

2. **Принцип неоднородности**. При переходе к решению практических задач, возникающих не в искусственно созданной, а в реально сложившейся и эволюционирующей среде, разработчик неизбежно сталкивается с многообразием парадигм, методов и переменных в науке, дисциплин в обучении, мнений и моделей внешнего мира на практике, фаз управления, целей решения задач, отношений на знаниях. Следствием такого многообразия

является неоднородность сложных задач, требующая от разработчика отказа от попыток применить для моделирования решения автономные методы.

3. **Принцип конструирования**. Метод решения сложной задачи синтезируется из методов, моделей, модулей, инструментальных средств и технологий всякий раз заново, когда возникает необходимость решения сложной задачи. Приступая к конструированию, необходимо знать плюсы и минусы методов и инструментария, из которых строятся ГиИС, оценки надежности знаний о подзадаче, трудоемкость решения подзадачи тем или иным методом, или инструментарием.

4. **Принцип плюрализма**. Нет ни одного, окончательно разработанного метода для объяснения или решения сложной, неоднородной задачи. Тем не менее, существует некоторое подмножество уже разработанного множества методов и моделей, которое может быть использовано для моделирования решения сложной, неоднородной задачи.

5. **Принцип приоритета знаний**. ГиИС должна быть построена таким образом, чтобы первый приоритет в решении сложной задачи отдавался точным знаниям и жестким вычислениям и только второй - эвристическим знаниям и мягким вычислениям. При этом использование эвристик должно рассматриваться как коррекция решения, полученного на точных знаниях. В случае отсутствия точных знаний возможно применение одних эвристик, однако в этом случае должна быть заранее поставлена цель получения таких точных знаний.

6. **Принцип постепенности**. Прежде чем разрабатывать ГиИС, необходимо накопить опыт и знания построения или использования автономных моделей. Это позволит избежать ошибок в автономных моделях, которые неизбежно перейдут в ГиИС и снизят ее качество.

7. **Первый принцип наследования**. Чтобы функциональная ГиИС унаследовала сумму плюсов методов-прототипов, необходимо, чтобы степень зернистости и интерфейсы, реализующие отношения интеграции, были установлены как на уровне декларативных, так и процедурных представлений.

8. **Второй принцип наследования**. Архитектура ГиИС наследуется исходя как из состава и структуры сложной задачи, так и от плюсов и минусов автономных методов. Отказ от релевантности проблеме ведет к хорошим, но

бесполезным гибридам. Отказ от учета плюсов и минусов автономных методов ведет к снижению качества ГиИС.

9. **Принцип самоорганизации**. ГиИС должна обучаться и извлекать знания из одного элемента для совершенствования других элементов. Это обеспечивает адаптацию ГиИС к новым условиям и снижает трудозатраты на ее эксплуатацию.

10. **Принцип полноты**. ГиИС должна строиться с использованием как можно большего числа классов автономных методов: аналитических, статистических, символьных, коннекционистских и эволюционных.

11. **Принцип снижения производительности**. ГиИС может использоваться там, где не существенно снижение производительности вычислений по сравнению с автономными моделями.

Данные принципы не претендуют на универсальный и строгий характер математических или логических правил. Эти принципы сформулированы в результате исследований опыта разработки ГиИС в мировой практике и позволяют разрабатывать ГиИС адекватные сложности поставленной задачи.