ТБИЛИССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. И. Н. ДЖАВАХИШВИЛИ

На правах рукописи

ОГАНЕЗОВ АЛЕКСЕЙ ЛЕВАНОВИЧ

ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ В ЗАДАЧАХ РАСПОЗНАВАНИЯ ОБРАЗОВ

05.13.11 – МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ МАШИН, СИСТЕМ, КОМПЛЕКСОВ И СЕТЕЙ

ДИССЕРТАЦИЯ НА СОИСКАНИЕ УЧЕНОЙ СТЕПЕНИ КАНДИДАТА ФИЗИКО-МАТЕМАТИЧЕСКИХ НАУК

НАУЧНЫЙ РУКОВОДИТЕЛЬ:
КАНДИДАТ ТЕХНИЧЕСКИХ НАУК, ДОЦЕНТ
З. Ю. КОЧЛАДЗЕ

ТБИЛИСИ
2006

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ.

Актуальность темы.

Одним из наиболее актуальных направлений в области кибернетики Грузии является построение програмных приложений для распознавания печатных шрифтов грузинского алфавита. В настоящее время существют некоторые реализации для распознавания грузинских шрифтов, тем не менее повышение эффективности распознавания с применением новых модифицированных алгоритмов классификации образов на основе искусственных нейронных сетей является важным и актуальным вопросом. На кафедре математической кибернетики и информатики Тбилисского Государственного Университета им. И. Джавахишвили (ТГУ) подготавливалась научная база и проводились новаторские эксперименты в области алгоритмизации и математического моделирования информационных систем для распознававания образов (РО).

Релевантные научные исследования применяемые для РО в Грузии проводились в области дискретной математики, исследования операций, математического анализа, искусственного интеллекта и нечеткой логики, теории игр и теории автоматов, систем автоматического проектирования (САПР), автоматических систем управления (АСУ), когерентных, экспертных и нейродинамических систем, аналитической эвристики и вероятностно концептуальным принципам организации памяти, искусственных нейронных сетей, опознавания и классификации объектов, вопросов математического принятия решений, формирования аппликативной грамматики, синтаксиса и семантики языка, парадигмы символических систем и др.

Кибернетика возникла на стыке многих областей знания: математики, логики, семиотики, синергетики и биологии. Одним из самых восстребованных направлений кибернетической науки является искусственный интеллект и проблема распознавания образов (ПРО). Термин интеллект (intelligence) происходит от латинского intellectus – что означает ум, мыслительные способности человека. Искусственный интеллект (ИИ) как раздел информатики, изучает методы, способы и приемы моделирования и воспроизведения с помощью ЭВМ разумной деятельности человека, связанной с решением задач. В рамках искусственного интеллекта различают два основных направления:

- символьное (семиотическое, нисходящее) основано на моделировании высокоуровневых процессов мышления человека и использовании знаний;
- нейрокибернетическое (нейросетевое, восходящее) основано на моделировании отдельных низкоуровневых структур мозга (нейронов).

Главной задачей искусственного интеллекта является построение интеллектуальных информационных систем, которые обладали бы уровнем эффективности решений неформализованных задач, сравнимых с человеческими возможностями или превосходящим его. При построении системы искусственного интеллекта должны обладать следующими характерными особенностями:

1) наличие в них собственной внутренней модели внешнего мира; эта модель обеспечивает индивидуальность, относительную самостоятельность системы в оценке ситуации, возможность семантической и прагматической интерпретации запросов к системе;
2) способность пополнения имеющихся знаний;
3) способность к дедуктивному выводу, т.е. к генерации информации, которая в явном виде не содержится в системе; это качество позволяет системе конструировать информационную структуру с новой семантикой и практической направленностью;
4) умение оперировать в ситуациях, связанных с различными аспектами нечеткости, включая «понимание» естественного языка;
5) способность к диалоговому взаимодействию с человеком;
6) способность к адаптации.

Создание нейрокомпьютеров и моделирование адаптивных нейронных сетей рассматривается как наиболее востребованное направление в решении многих проблем искусственного интеллекта [31]. Актуальность исследований в этом направлении подтверждается массой различных применений, например: автоматизация процессов распознавания речи и образов, доказательство математических теорем, искусственный синтез речи по тексту, создание экспертных систем, аппроксимация функционалов, машинный перевод текстов, прогноз финансовых показателей на биржевом рынке, управление бизнес процессами, построении нейронных самообучающихся систем и многое другое [6].

Искусственная нейронная сеть (ИНС) применяемая для РО, представляет собой математическую модель параллельных вычислений, содержащую взаимодействующие между собой простые процессорные элементы – искусственные нейроны. Преимуществом нейронных сетей перед традиционными алгоритмами является возможность их обучения. Д. Хебб, высказал постулат, что обучение заключается в первую очередь в изменениях силы синоптических связей. Для решения задач на основе нейронной сетей разработчикам требуется:

- выбрать соответствующую модель сети;
- определить топологию сети (число элементов и их связи);

- указать параметры обучения.

Обучение ИНС может вестись с учителем или без него. В первом случае сети предъявляются значения как входных, так и желательных выходных сигналов, и она по некоторому внутреннему алгоритму подстраивает веса своих синаптических связей. Во втором случае выходы ИНС формируются самостоятельно, а веса изменяются по алгоритму, учитывающему только входные и производные от них сигналы. Существует большое множество алгоритмов обучения, которые, однако, делятся на два класса: детерминистские и стохастические. В первом подстройка весов представляет собой жесткую последовательность действий, во втором - она производится на основе действий, подчиняющихся некоторому случайному процессу.

В настоящее время одним из актуальных вопросов информационных технологий в области искусственного интеллекта является проблема распознавания и классификации образов с применением принципиально новых методов и алгоритмов, на основе применения искусственных нейронных сетей. Налаженные нейронные сети можно применять для решения самых различных задач, от восстановления пропусков в данных до анализа и поиска закономерностей. Для осуществления преобразования рукописной и печатной информации с бумажного листа в электронный вид наиболее актуальной проблемой в Грузии является распознавание шрифтов грузинского алфавита. В настоящей работе рассматривается моделирование и проводится апробация алгоритмов искусственной нейронной сети, позволяющие распознавать символы – цифр и шрифтов латинского и грузинского алфавитов с высокой эффективностью.

Цель работы.

Основной целью данной работы является: разработка, тестирование и проектирование принципиально новой по своему действию искусственной нейро-матричной вероятностно-статистической сети для облегчения распознавания плоских символьных образов (в частности цифр, символов латинского и грузинского алфавитов), использующей мульти-параллельные процессы с новой разделенной сетевой структурой искусственных нейронов.

Для достижения этой цели в работе поставлены и решены следующие задачи:

- сбор и номинальной ввод базы данных исследуемых растровых объектов трех алфавитов;
- исследование и анализ графических шрифтов и изображений для выявления математических закономерностей и свойств визуальных объектов, применимых в ИНС;
- построение графиков и диаграмм, определяющих степень сравнительной удаленности между различными объектами;
- формирование пошагового алгоритма и графической блок-схемы ИНС;

- достижение высокого результата классификации при минимально возможном разрешении матрицы рецепторов и количестве представителей образов из каждого набора символьных элементов базы данных;
- моделирование, тестирование и отладка программной реализации принципиально новой комплексной структурной модели ИНС, использующей вероятно-статистический анализ нейро-матричных массивов графической информации с мульти разделенной сетью нейронов, для параллельных процессов классификации образов на языке Object Pascal в среде проектной разработки RAD Delphi.

Методы исследования.

В работе использованы теория искусственных самоорганизующихся нейронных сетей для графических массивов базы данных с применением процесса распознавания без учителя, алгоритмы обратного распространения, метод зондов, теория вероятностей и статистический анализ, теория кодирования/декодирования и дискретных систем, концептуального анализа [13], теория множеств и кластеризации/категоризации объектов, методы алгоритмизации, методы компьютерного моделирования в VBA для Access и на основе MS Excel статистических таблиц, а также объектно-ориентированного программирования (ООП) на языке Object Pascal в среде Rapid Application Development (RAD) оболочки Borland Delphi.

Научная новизна.

Имеющиеся в арсенале алгоритмы классификации в основном используют неадаптивные алгоритмы распознавания образов, которые работают для определенных печатных шрифтов. Незначительное отклонение от стандартов начертания шрифтов приводит к низким результатам распознавания. На основе проведенных исследований в настоящей работе предложена новая и эффективная методика распознавания образов с помощью разделенной схемы искусственной нейронной сети без учителя на основе вероятностно-статистического анализа не только для печатных, но и рукописных символов, в частности цифр и букв грузинского и латинского алфавитов, позволяющие существенно повысить коэффициент распознавания образов. Научная новизна заключается в формулировании методики и создании нового, адаптивного алгоритма, с использованием разделенной искусственной нейронной сети без учителя и параллельного вероятностно-статистического анализа графической информации. Были введены понятия диверсивных, конгруэнтных, пассивных и нейтральных масс, представляющие собой различные категории весов нейронной сети, для детерминирования принадлежности изображения к тому или иному классу визуальных объектов.

Практическая и теоретическая ценность работы.

На основе построения принципиально новой модели мульти-параллельной искусственной нейронной схемы и применения вероятностно-статистических методов были получены высокие результаты распознавания не только печатных, но и прописных образов. Обычно для качественного исследования набора символов используют эталонные символы в номинальном количестве 1 000 экземпляров и разрешении матрицы 120x160 точек. При учете если мы имеем дело, например, как в нашем случае, с 69 разновидностями символов и 25%-ой активности воспринимающей матрицы точечной закраски, то получаем 331 200 000 активных закрашенных точек для аналитической обработки и классификации. Основной ценностью нового метода является то, что вопреки уменьшению базы набора для исследования графической информации, получается больший эффект, т.е. при минимальном количестве введенных эталонных образов, а именно в количестве всего 40 прототипов для каждого символа, вводимых в базу данных и при минимально возможной матрице рецепторов с разрешением 12x16 точек получается высокий результат распознавания. Таким образом, вместо 331 200 000 вносимых для обработки и классификации точек, имеем дело с 132 480 точками, что на 331 067 520 точек меньше для ввода и информационной обработки изображений. Существенное уменьшение базы данных набора и повышения эффективности распознавания объектов также является практической ценностью нового метода данной работы. Новым подходом данной работы также является специфичность разработанного алгоритма и программного приложения, реализующих модель искусственной нейронной сети, в которой каждому образу распознавания соответствует сформированная база данных наборов и искусственный матричный нейрон, осуществляющий распознавание образа. Структура построенной модели распознавания образов осуществляется в мульти-параллельном режиме, что является уникальным методом, существенно повышающим скорость, точность и эффективность классификации.

Реализация результатов исследования.

Результаты исследований предложены для имплементации на кафедре Кибернетики Тбилисского Государственного Университета им. И. Джавахишвили.

Апробация работы.

Основные положения и результаты диссертационной работы докладывались на семинарах кафедры искусственного интеллекта Тбилисского Государственного Университета им. И. Джавахишвили, на конференциях (см. перечень публикаций) Московского Государственного Университета Экономики Статистики и Информатики (МЭСИ).

Публикации.

Основные результаты диссертации содержатся в списке публикаций 8 научных работ. В работах, написанных в соавторстве, научные результаты были получены с активным участием диссертанта.

Диссертационная работа состоит из введения, трех глав, заключения, списка использованной литературы, содержащего 81 наименований, 102 ссылки ресурсов Интернета и Приложения, включающего 3 страницы фрагментов программы, 9 страниц листинга и 22 страницы иллюстраций. Текст диссертации изложен на 149 страницах, в том числе содержит 73 рисунка, 2 таблицы, 1 блок-схему и 2 листинга.

В первой главе диссертации рассматриваются вопросы нейробиологии, в частности дается общее понятие о структуре и строении мозга, биологическом нейроне, составляющих его элементах (аксоны, синапсы, дендриты), типах его клеток, синаптичесокой силе, нейронных сетях и принципах их работы, проводится аналогия между компьютером и человеческим мозгом. Излагается понятие искусственного нейрона, весов, активационных функций (сигмоидальной логистической и гиперболического тангенса), пороговой величины, коэффициентов усилений; также были рассмотрены однослойные и многослойные искусственные нейронные сети; архитектура и свойства сетей с обратным распространением ошибки (Eerror Backpropagation), нейронные сети Хебба прямопоточных (Feedforward) сетей, алгоритм Кохонена обучения без учителя и основанные на этом методе нейронные сети особого типа – так называемые самоорганизующиеся структуры с подстройкой синапсов – Self-organizing Feature Maps (SOM), однослойные сети Хопфилда и двухслойные сети Хэмминга с ассоциативной памятью, а также формальные алгоритмы обучения сетей.

Во второй главе диссертации рассмотрены рецепторная структура восприятия информации, понятие образа, проблема обучения распознаванию образов (ПРО), геометрический и структурный подходы проблемы распознавания образов, гипотеза компактности и понятие адаптации. Приведен метод зондов и преобразование изображений в цифровой код. Рассмотрены алгоритмические построения и процесс обучения искусственных нейронных сетей с учителем и без учителя. Даны алгоритмы распознавания образов, в частности алгоритм секущих плоскостей, алгоритм, основанный на методе потенциалов. Рассмотрен метод обучения Уидроу-Хоффа и исследования Кохонена на самоорганизующихся структурах. Описаны составные элементы (R / S / A) и интегральное устройство персептрона Розенблатта, анализируется работа персептрона, основывающаяся на принципах структуры мозга и зрительного аппарата и его применение для распознавания образов. Приведена также схемотехника и функциональность узнающей машины Гамба.

Затронут вопрос персептронной представляемости, потенциал многослойных персептронов и проблема функции исключающего ИЛИ.

В третьей главе диссертации рассмотрены вопросы моделирования искусственных нейронных сетей для распознавания образов, проблемы, возникающие при этом, такие как достаточность набора базы данных образов, качество эталонных составляющих, а также этапы разрешения проблем, в частности операции группировки, секвестирования, ассоциирования. Описывается матрица рецепторов, проводятся подсчеты статистических показателей, сумм весов и вероятностей матрицы, усредненных значений и медиан весов матрицы, а также анализ количества точек, имеющих вероятности появления активных (закрашенных) точек из диапазонов [0.00], (0.00-0.25], (0.25-0.50], (0.50-0.75], (0.75-1.00], [1.00], для выявления характеристик каждого символьного образа. Также построены графики зависимостей вертикальных и горизонтальных составляющих зондов изображения, и линейный график точечных весов, построены 3D-колонные диаграммы точечных вероятностей для каждого образа, описан новый метод точечной интерполяции весов первого и второго уровня на основе метода потенциалов. Были созданы усредненные контуры каждого символа и построена нейронная самообучающаяся сеть обработки изображения для улучшения показателей сходимости и распознавания. Разработана блок-схема и функциональный граф искусственной матричной нейронной сети для распознавания образов.

В результате проведенной большой исследовательской работы была построена гибридная модель разделенной искусственной нейронной сети, использующая вероятностно-статистический анализ изображений. Данная комплексная модель программно была реализована и апробирована на примере распознавания арабских цифр, латинского и грузинского алфавитов. В результате апробации было выявлено, что преимуществом данной программы является то, что она может распознавать непечатные символы, может быть применена для классификации любых алфавитов и контурных изображений, дает высокие результаты распознавания символов при экспериментально низком разрешении матрицы рецепторов и малой базе набора эталонных изображений. В данной главе представлены также фрагменты программы и листинг алгоритмов распознавания образов, на основе разработанной модели многослойной вероятностно-статистической искусственной нейросети, использующей обработку двумерных массивов, трансформацию изображений с машинного в цифровой код, матричные калькулированния, вероятно-статистический анализ табличных данных и графических диаграмм, а также различные оценочные операции над дискретными множествами.

ГЛАВА 1. СТРУКТУРА И СВОЙСТВА ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ.

1.1 Мозг и биологический нейрон.

1.1.1 Общее понятие о мозге и его строении.

Мозг (рис 1.1) и Центральный Процессор (ЦП) ЭВМ (рис. 1.2), являются самыми сложными системами переработки информации. ЦП интегрирует миллионы полупроводниковых транзисторов и подобен мозгу, в котором содержится около 100 миллиардов нейронов, каждый из которых имеет в среднем 10 000 связей. По аналогии с схемотехникой больших интегральных схем (БИС) ЭВМ, где происходит прием, передача и переработка цифровой информации, биологическая сеть нейронов отвечает за все явления, которые мы называем мыслями, эмоциями, памятью, сенсомоторными и автономными функциями [23]. Мозг, в котором, в отличие от ЦП, ежедневно погибает большое количество нейронов, чрезвычайно надежен и продолжает функционировать. Обработка огромных объемов информации мозгом осуществляется очень быстро, за доли секунды, несмотря на то, что сам нейрон является медленнодействующим элементом со временем реакции нескольких миллисекунд. В отличие от биологического нейрона, электрические полупроводники ЦП являются более быстродействующими и безошибочными, однако количество связей между

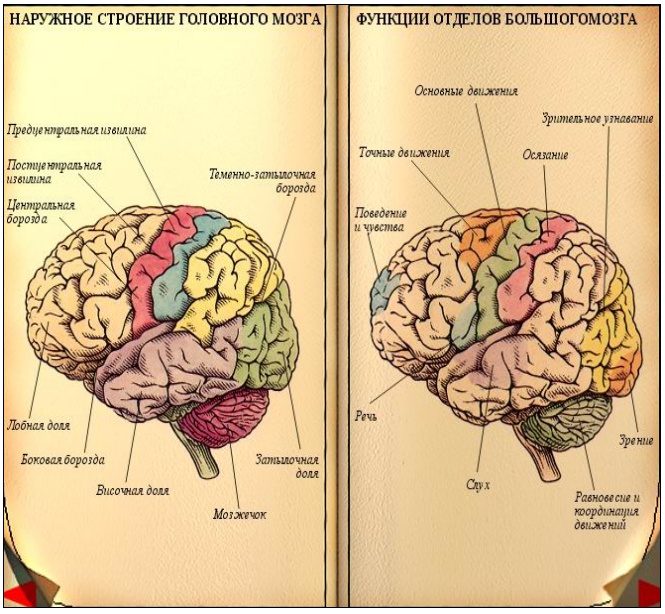


Рис. 1.1 Строение и функции головного мозга.

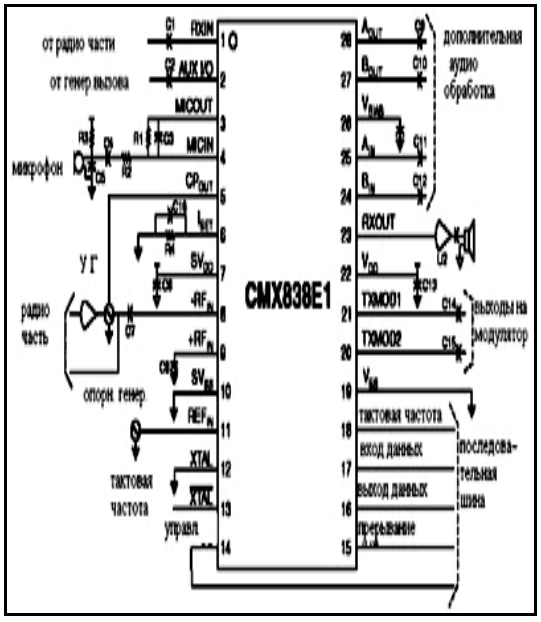


Рис. 1.2 Пример схемы ЦП.

нейронами мозга во много раз превышает связи между полупроводниками ЦП, что обеспечивает параллелизм и многопоточность обработки информации [30]. Головной мозг человека состоит из серого и белого вещества. Серое вещество представляет собой сеть дендритов, аксонов и тел нервных клеток. Миэлинизированные волокна, соединяющие различные области мозга друг с другом, с органами чувств и мускулами, образуют белое вещество. Мозг состоит из двух полушарий, каждое полушарие в свою очередь из четырех частей - лобной, теменной, височной и затылочной. В лобной части находится отдел эмоций и центры управления движениями - правое полушарие отвечает за движение левой руки и ноги, а левое за движение правой руки и ноги. В теменной части – зона телесных ощущений и осязаний. К ней примыкает височная зона, в которой расположены центр речи, слуха, вкуса. В затылочной части расположен зрительный отдел, также отдел распознавания окружающих предметов посредством зрения. В мозге существуют структурно обособленные отделы такие, как кора, гиппокамп, таламус, мозжечок, миндалина, полосатое тело и т. д. Каждый из отделов, в свою очередь, имеет сложное модульное строение. Особое место в мозге зажимает церебральная кора. Считается, что именно здесь происходят важнейшие процессы ассоциативной переработки информации. Мозг является основным потребителем энергии тела. Включая в себя лишь 2% массы тела, в состоянии покоя, мозг использует приблизительно 20% кислорода тела. Мозг также содержит густую сеть кровеносных сосудов, которые обеспечивают кислородом и питательными веществами нейроны и другие ткани. Эта система кровоснабжения связана с главной системой кровообращения посредством высокоэффективной фильтрующей системы, называемой гематоэнцефалическим барьером, являющимся механизмом защиты, предохраняющим мозг от возможных токсичных веществ, находящихся в крови. Защита обеспечивается низкой проницаемостью кровеносных сосудов мозга, а также плотным перекрытием глиальных клеток, окружающих нейроны. Фактически весь объем мозга, не занятый нейронами и кровеносными сосудами, заполнен глиальными клетками, которые обеспечивают структурную основу мозга.

1.1.2 Биологический нейрон и нейронные сети.

Каждый нейрон обладает многими качествами, общими с другими элементами тела, но его уникальной способностью является прием, обработка и передача электрохимических сигналов по нервным путям [19], которые образуют коммуникационную систему мозга. На рис. 1.3 и рис. 1.4 показана структура и строение типичных биологических нейронов. Нейрон состоит из трех частей: тела клетки, дендритов и аксона, каждая часть со своими, но

взаимосвязанными функциями. Дендриты идут от тела нервной клетки (сома) к другим нейронам, где они принимают сигналы в точках соединения, называемых синапсами [60]. Принятые синапсом входные сигналы подводятся к телу нейрона. Здесь они суммируются, причем одни входы стремятся возбудить нейрон, другие – воспрепятствовать его возбуждению. Когда суммарное возбуждение в теле нейрона превышает некоторый порог, нейрон возбуждается, посылая по аксону сигнал другим нейронам. У этой основной функциональной схемы много усложнений и исключений, тем не менее большинство искусственных нейронных сетей моделируют лишь простые свойства.

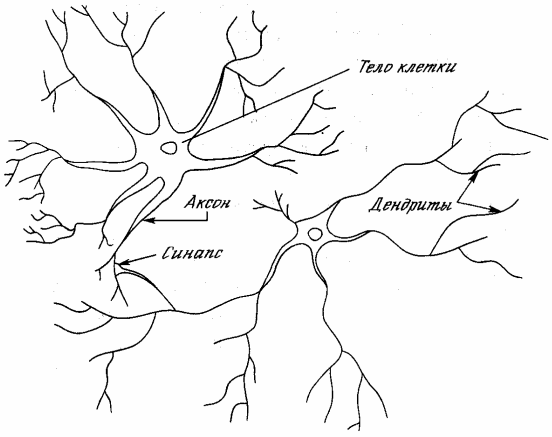


Рис. 1.3 Биологический нейрон.

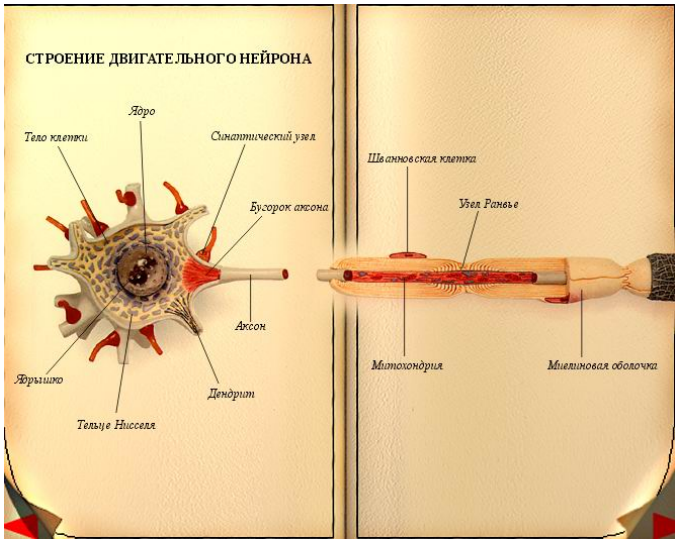


Рис. 1.4 Строение двигательного нейрона.

Нервные клетки, или нейроны, представляют собой особый вид клеток в живых организмах, обладающих электрической активностью, основное назначение которых заключается в оперативном управлении организмом. Сома, как правило, имеет поперечный размер несколько десятков микрон. Длина дендритов может достигать 1 мм**,** дендриты сильно ветвятся, пронизывая сравнительно большое пространство в окрестности нейрона. Длина аксона может достигать сотен миллиметров. На соме и на дендритах располагаются окончания (коллатерали) аксонов, идущих от других нервных клеток. Каждое такое окончание имеет вид утолщения, называемого синаптической бляшкой, или синапсом. Поперечные размеры синапса, как правило, не превышают нескольких микрон, чаще всего эти размеры составляют около 1 мкм.

Входные сигналы дендритного дерева (постсинаптические потенциалы) взвешиваются и суммируются на пути к аксонному холмику, где генерируется выходной импульс (спайк) или пачка импульсов. Его наличие (или интенсивность), следовательно, является функцией взвешенной суммы входных сигналов. Выходной сигнал проходит по ветвям аксона и достигает синапсов, которые соединяют аксоны с дендритными деревьями

других нейронов. Через синапсы сигнал трансформируется в новый входной сигнал для смежных нейронов. Этот входной сигнал может быть положительным и отрицательным (возбуждающим или тормозящим) в зависимости от вида синапсов. Величина входного сигнала, генерируемого синапсом, может быть различной даже при одинаковой величине сигнала, приходящего в синапс. Эти различия определяются эффективностью или весом синапса. Синаптический вес может изменяться в процессе функционирования синапса. Нейроны можно разбить на три большие группы: рецепторные, промежуточные и эффекторные. Рецепторные нейроны обеспечивают ввод в мозг сенсорной информации. Они трансформируют сигналы, поступающие на органы чувств (оптические сигналы в сетчатке глаза [80], акустические в ушной улитке или обонятельные в хеморецепторах носа), в электрическую импульсацию своих аксонов. Эффекторные нейроны передают приходящие на них сигналы исполнительным органам. На конце их аксонов имеются специальные синаптические соединения с исполнительными органами, например мышцами, где возбуждение нейронов трансформируется в сокращения мышц. Промежуточные нейроны осуществляют обработку информации, получаемой от рецепторов, и формируют управляющие сигналы для эффекторов. Они образуют центральную нервную систему.

1.1.3 Аналогия между компьютером и человеческим мозгом.

Основные положения теории деятельности головного мозга и математическая модель нейрона были разработаны У. Мак-Каллоком и Ч. Питтсом в 1943 году. Согласно предложенной модели мозг представляет собой множество нейронов, имеющих одинаковую структуру. Каждый нейрон реализует некоторую функцию, называемую пороговой, над входными значениями [38]. Если значение функции превышает определенную величину - порог (что характеризует суммарную значимость полученной нейроном информации), нейрон возбуждается и формирует выходной сигнал для передачи его другим нейронам. Пройдя путь от рецепторов (слуховых, зрительных и других) через нейронные структуры мозга до исполнительных органов, входная информация преобразуется в набор управляющих воздействий, адекватных ситуации.

Отдельные нейроны, соединяясь между собой, образуют новое качество, которое, в зависимости от характера межнейронных соединений, имеет различные уровни биологического моделирования:

- группа нейронов;
- нейронная сеть;
- нервная система;

- мыслительная деятельность;
- мозг.

Существует подобие между мозгом и цифровым компьютером: оба оперируют электронными сигналами, оба состоят из большого количества простых элементов, оба выполняют функции, являющиеся, грубо говоря, вычислительными, тем не менее существуют и фундаментальные отличия [28]. По сравнению с микросекундными и даже наносекундными интервалами вычислений современных компьютеров нервные импульсы являются слишком медленными. Хотя каждый нейрон требует наличия миллисекундного интервала между передаваемыми сигналами, высокая скорость вычислений мозга обеспечивается огромным числом параллельных вычислительных блоков, причем количество их намного превышает доступное современным ЭВМ [29]. Диапазон ошибок представляет другое фундаментальное отличие: ЭВМ присуща свобода от ошибок, если входные сигналы безупречно точны и ее аппаратное и программное обеспечение не повреждены. Мозг же часто производит лучшее угадывание и приближение при частично незавершенных и неточных входных сигналах. Часто он ошибается, но величина ошибки должна гарантировать наше выживание в течение миллионов лет [11]. Эти две системы явно различаются в каждой своей части. Они оптимизированы для решения различных типов проблем, имеют существенные различия в структуре и их работа оценивается различными критериями.

Развитие искусственных нейронных сетей вдохновляется биологией. Структура искусственных нейронных сетей была смоделирована как результат изучения человеческого мозга. Искусственные нейронные сети чрезвычайно разнообразны по своим конфигурациям, функциональности и целевому назначению [42]. Рассматривая сетевые конфигурации и алгоритмы, исследователи мыслят их в терминах организации мозговой деятельности, но на этом аналогия может и закончиться. Сходство между ними очень незначительно, однако, даже эта скромная эмуляция мозга дает ощутимые результаты. Например, искусственные нейронные сети имеют такие аналогичные мозгу свойства, как способность обучаться на опыте, основанном на знаниях, делать абстрактные умозаключения и совершать ошибки, что является более характерным для человеческой мысли, чем для созданных человеком компьютеров. Разработчикам сетей приходится выходить за пределы современных биологических знаний в поисках структур, способных выполнять полезные функции [4]. Во многих случаях это приводит к необходимости отказа от биологического правдоподобия, мозг становится просто метафорой, и создаются сети, невозможные в живой материи или требующие неправдоподобно больших допущений об анатомии и функционировании мозга.

В настоящее время возникли и остаются две взаимно обогащающие друг друга цели нейронного моделирования: первая – понять функционирование нервной системы человека на уровне физиологии и психологии и вторая – создать вычислительные системы (искусственные нейронные сети), выполняющие функции, сходные с функциями мозга [51]. Нейроны можно моделировать довольно простыми автоматами, а вся сложность мозга, гибкость его функционирования и другие важнейшие качества определяются связями между нейронами. Каждая связь представляется как простой элемент, служащий для передачи сигнала и его линейного усиления или ослабления.

Существует большой класс задач: нейронные системы ассоциативной памяти, статистической обработки, фильтрации и другие, для которых связи формируются по определенным формулам, при этом обучение нейронных сетей оказалось осуществимым при моделировании задач на обычных персональных компьютерах.

1.2 Искусственный нейрон.

1.2.1 Модель искусственного нейрона.

Искусственный нейрон имитирует в первом приближении свойства биологического нейрона. На вход искусственного нейрона поступает некоторое множество сигналов, каждый из которых является выходом другого нейрона. Каждый вход умножается на соответствующий вес, аналогичный синаптической силе, и все произведения суммируются, определяя уровень активации нейрона. На рис. 1.5 представлена модель, реализующая эту идею. Хотя сетевые парадигмы весьма разнообразны, в основе почти всех их лежит эта конфигурация. Здесь множество входных сигналов, обозначенных x_1, x_2,…, x_n, поступает на искусственный нейрон. Эти входные сигналы, в совокупности, обозначаемые вектором **X**, соответствуют сигналам, приходящим в синапсы биологического нейрона [39]. Каждый сигнал умножается на соответствующий вес w_1, w_2,…, w_n, и поступает на суммирующий блок, обозначенный Σ. Каждый вес соответствует «силе» одной биологической синаптической связи. (Множество весов в совокупности обозначается вектором **W**). Суммирующий блок, соответствует телу биологического элемента, складывает взвешенные входы алгебраически, создавая выход – NET. В векторных обозначениях это может быть компактно записано следующим образом:

$$\mathrm{NET} = \mathbf{XW}. \qquad (1.1)$$

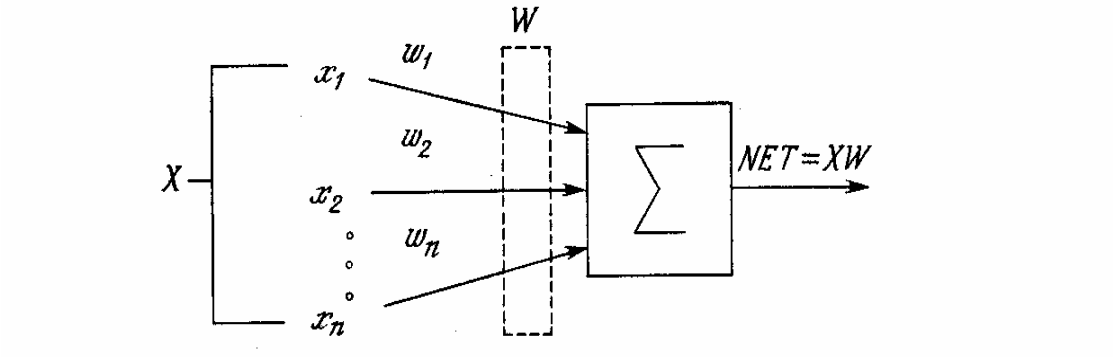


Рис. 1.5 Искусственный нейрон.

1.2.2 Активационные функции.

Сигнал NET преобразуется активационной функцией *F* и дает выходной нейронный сигнал OUT [37]. Активационная функция может быть обычной линейной функцией

$$OUT = K(NET), \qquad (1.2)$$

где *K* – постоянная, пороговой функции

OUT = 1, если NET > T,

OUT = 0 в остальных случаях,

где *T* – некоторая постоянная пороговая величина, или же функцией, более точно моделирующей нелинейную передаточную характеристику биологического нейрона и представляющей нейронной сети большие возможности.

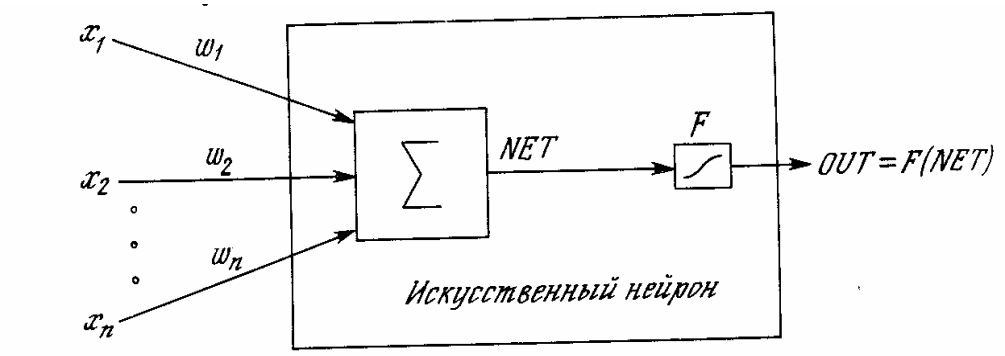


Рис. 1.6 Искусственный нейрон с активационной функцией.

На рис. 1.6 блок, обозначенный *F,* принимает сигнал NET и выдает сигнал OUT. Если блок *F* сужает диапазон изменения величины NET так, что при любых значениях NET значения OUT принадлежат некоторому конечному интервалу, то *F* называется «сжимающей» функцией. В качестве «сжимающей» функции часто используется

логистическая или «сигмоидальная» (S-образная) функция, показанная на рис. 1.7. Эта функция математически выражается как $F(x) = 1/(1 + e^{-x})$. Таким образом,

$$\mathrm{OUT} = \frac{1}{1 + \mathrm{e}^{-\mathrm{NET}}}. \qquad (1.3)$$

По аналогии с электронными системами активационную функцию можно считать нелинейной усилительной характеристикой искусственного нейрона. Коэффициент усиления вычисляется как отношение приращения величины OUT к вызвавшему его небольшому приращению величины NET. Он выражается наклоном кривой при определенном уровне возбуждения и изменяется от малых значений при больших отрицательных возбуждениях (кривая почти горизонтальна) до максимального значения при нулевом возбуждении и снова уменьшается, когда возбуждение становится большим положительным. Гроссберг обнаружил, что подобная нелинейная характеристика решает поставленную им дилемму шумового насыщения [41]. Каким образом одна и та же сеть может обрабатывать как слабые, так и сильные сигналы? Слабые сигналы нуждаются в большом сетевом усилении, чтобы дать пригодный к использованию выходной сигнал. Однако усилительные каскады с большими коэффициентами усиления могут привести к насыщению выхода шумами усилителей (случайными флуктуациями), которые присутствуют в любой физически реализованной сети. Сильные входные сигналы в свою очередь также будут приводить к насыщению усилительных каскадов, исключая возможность полезного использования выхода. Центральная область логистической функции, имеющая большой коэффициент усиления, решает проблему обработки слабых сигналов, в то время как области с падающим усилением на положительном и отрицательном концах подходят для больших возбуждений. Таким образом, нейрон функционирует с большим усилением в широком диапазоне уровня входного сигнала.

$$\mathrm{OUT} = \frac{1}{1 + \mathrm{e}^{-\mathrm{NET}}} = F(\mathrm{NET}).$$

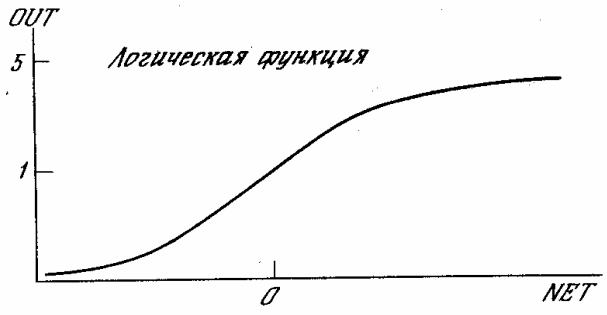


Рис. 1.7 Сигмоидальная логистическая функция.

Другой широко используемой активационной функцией является гиперболический тангенс. По форме она сходна с логистической функцией и часто используется биологами в качестве математической модели активации нервной клетки. В качестве активационной функции искусственной нейронной сети она записывается следующим образом:

$$OUT = th(x). \qquad (1.4)$$

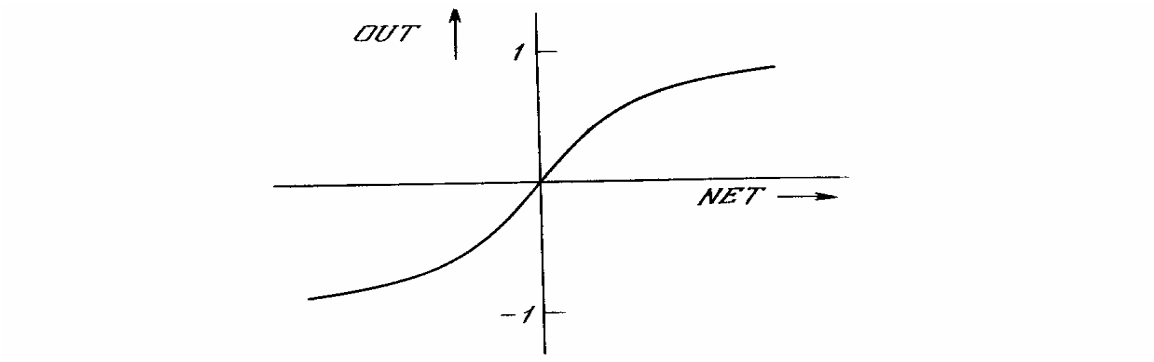


Рис. 1.8 Функция гиперболического тангенса.

Подобно логистической функции, гиперболический тангенс является S-образной функцией, но он симметричен относительно начала координат, и в точке NET = 0 значение выходного сигнала OUT равно нулю (см. рис. 1.8). В отличие от логистической функции, гиперболический тангенс принимает значения различных знаков, что оказывается выгодным для ряда сетей.

Рассмотренная простая модель искусственного нейрона игнорирует многие свойства своего биологического двойника. Например, она не принимает во внимание задержки во времени, которые воздействуют на динамику системы. Входные сигналы сразу же порождают выходной сигнал. И, что более важно, она не учитывает воздействий функции частотной модуляции или синхронизирующей функции биологического нейрона, которые ряд исследователей считают решающими. Несмотря на эти ограничения, сети, построенные из этих нейронов, обнаруживают свойства, сильно напоминающие биологическую систему.

1.3 Искусственные нейронные сети (ИНС).

1.3.1 Однослойные искусственные нейронные сети.

Хотя один нейрон и способен выполнять простейшие процедуры распознавания, сила нейронных вычислений проистекает от соединений нейронов в сетях. Простейшая сеть [50]

состоит из группы нейронов, образующих слой, как показано в правой части рис.1.9. Вершины-круги слева служат лишь для распределения входных сигналов. Они не выполняют каких-либо вычислений, и поэтому не будут считаться слоем. По этой причине они обозначены кругами, чтобы отличать их от вычисляющих нейронов, обозначенных квадратами. Каждый элемент из множества входов **X** отдельным весом соединен с каждым искусственным нейроном, а каждый нейрон выдает взвешенную сумму входов в сеть. В искусственных и биологических сетях многие соединения могут отсутствовать, все соединения показаны в целях общности. Могут иметь место также соединения между выходами и входами элементов в слое [72].

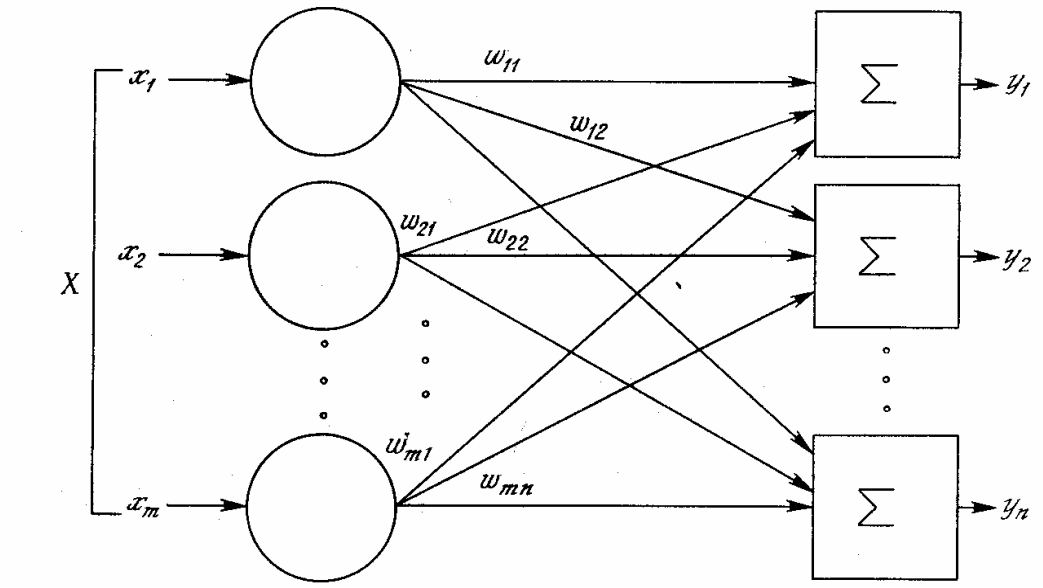


Рис. 1.9 Однослойная нейронная сеть.

Веса всех элементов матрицы можно обозначить через **W. М**атрица имеет m строк и n столбцов, где m – число входов, а n – число нейронов. Например, $w_{2,3}$ – это вес, связывающий третий вход со вторым нейроном. Таким образом, вычисление выходного вектора **N**, компонентами которого являются выходы OUT нейронов, сводится к матричному умножению **N** = **XW,** где **N** и **X** – векторы-строки.

1.3.2 Многослойные искусственные нейронные сети.

Более крупные и сложные нейронные сети обладают, как правило, и большими вычислительными возможностями [43]. Хотя созданы сети всех конфигураций, какие только можно себе представить, послойная организация нейронов копирует слоистые структуры определенных отделов мозга. Такие многослойные сети обладают большими возможностями, чем однослойные.

Многослойные сети могут образовываться каскадами слоев. Выход одного слоя является входом для последующего слоя [50]. Подобная сеть показана на рис. 1.10 и снова

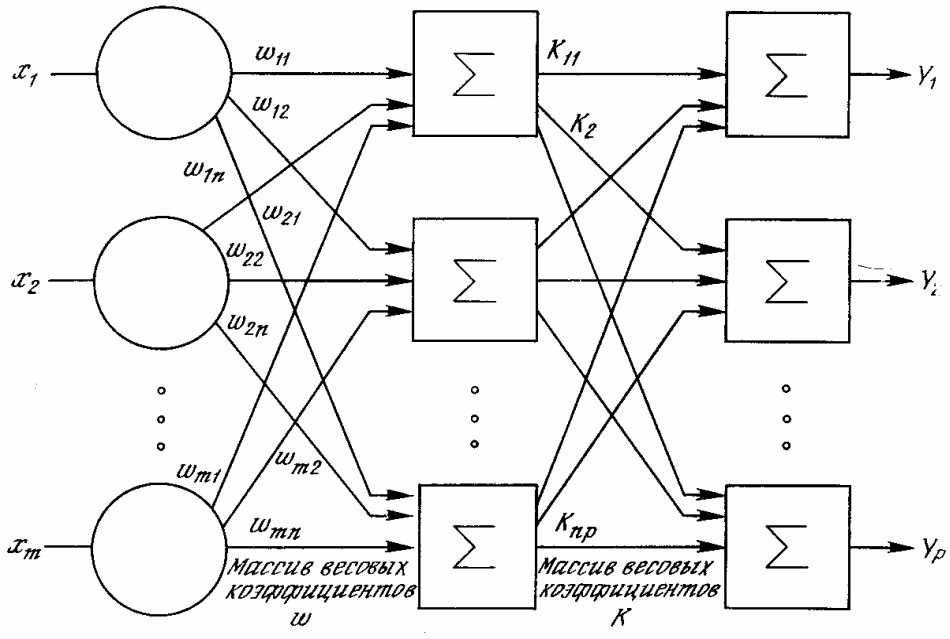


Рис. 1.10 Двухслойная нейронная сеть.

изображена со всеми соединениями. Многослойные сети не могут привести к увеличению вычислительной мощности по сравнению с однослойной сетью лишь в том случае, если активационная функция между слоями будет нелинейной [20]. 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.

$$(\mathbf{X}\mathbf{W}_1)\mathbf{W}_2$$

Так как умножение матриц ассоциативно, то X(W1W2). Это показывает, что двухслойная линейная сеть эквивалентна одному слою с весовой матрицей, равной произведению двух весовых матриц [14]. Следовательно, любая многослойная линейная сеть может быть заменена эквивалентной однослойной сетью, однако, однослойные сети весьма ограничены по своим вычислительным возможностям. Таким образом, для расширения возможностей сетей по сравнению с однослойной сетью необходима нелинейная активационная функция .

1.3.3 Нейронной сети с обратным распространением ошибки.

Когда в сети только один слой, алгоритм ее обучения с учителем очевиден, так как правильные выходные состояния нейронов единственного слоя заведомо известны, и

подстройка синаптических связей идет в направлении, минимизирующем ошибку на выходе сети. По этому принципу строится, например, алгоритм обучения однослойного персептрона. В многослойных же сетях оптимальные выходные значения нейронов всех слоев, кроме последнего, как правило, не известны, и двух или более слойный персептрон уже невозможно обучить, руководствуясь только величинами ошибок на выходах ИНС.

У сетей, рассмотренных выше, не было обратных связей, т. е. соединений, идущих от выходов некоторого слоя к входам этого же слоя или предшествующих слоев. Этот специальный класс сетей, называемых сетями без обратных связей или сетями прямого распространения. У сетей без обратных связей нет памяти, их выход полностью определяется текущими входами и значениями весов.

Один из вариантов решения этой проблемы – разработка наборов выходных сигналов, соответствующих входным, для каждого слоя ИНС, что, конечно, является очень трудоемкой операцией и не всегда осуществимо. Второй вариант – динамическая подстройка весовых коэффициентов синапсов, в ходе которой выбираются, как правило, наиболее слабые связи и изменяются на малую величину в ту или иную сторону, а сохраняются только те изменения, которые повлекли уменьшение ошибки на выходе всей сети. Третий вариант – распространение сигналов ошибки от выходов ИНС к ее входам, в направлении, обратном прямому распространению сигналов в обычном режиме работы. Этот алгоритм обучения ИНС получил название процедуры обратного распространения.

Согласно методу наименьших квадратов [36], минимизируемой целевой функцией ошибки ИНС является величина:

$$E(w) = \frac{1}{2}\sum_{j,p}(y_{j,p}^{(N)} - d_{j,p})^2 \qquad (1.5)$$

где $y_{j,p}^{(N)}$ – реальное выходное состояние нейрона j выходного слоя N нейронной сети при подаче на ее входы p-го образа; d_{jp} – идеальное (желаемое) выходное состояние этого нейрона.

Суммирование ведется по всем нейронам выходного слоя и по всем обрабатываемым сетью образам. Минимизация ведется методом градиентного спуска, что означает подстройку весовых коэффициентов следующим образом:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \frac{\partial E}{\partial w_{ij}} \qquad (1.6)$$

Здесь w_{ij} – весовой коэффициент синаптической связи, соединяющей i-ый нейрон слоя n-1 с j-ым нейроном слоя n, η – коэффициент скорости обучения, $0<\eta<1$.

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j} \cdot \frac{\partial s_j}{\partial w_{ij}} \tag{1.7}$$

Здесь под y_j, подразумевается выход нейрона j, а под s_j – взвешенная сумма его входных сигналов, то есть аргумент активационной функции. Так как множитель dy_j/ds_j является производной этой функции по ее аргументу, из этого следует, что производная активационной функция должна быть определена на всей оси абсцисс. В связи с этим функция единичного скачка и прочие активационные функции с неоднородностями не подходят для рассматриваемых ИНС. В них применяются такие гладкие функции, как гиперболический тангенс или классический сигмоид с экспонентой [68]. В случае гиперболического тангенса

$$\frac{dy}{ds} = 1 - s^2 \tag{1.8}$$

Третий множитель $\partial s_j/\partial w_{ij}$, очевидно, равен выходу нейрона предыдущего слоя $y_i^{(n-1)}$.

Первый множитель в (3), он легко раскладывается следующим образом:

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} \cdot \frac{\partial s_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} \cdot w_{jk}^{(n+1)} \tag{1.9}$$

Здесь суммирование по k выполняется среди нейронов слоя n+1.

Введя новую переменную

$$\delta_j^{(n)} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j} \tag{1.10}$$

мы получим рекурсивную формулу для расчетов величин $\delta_j^{(n)}$ слоя n из величин $\delta_k^{(n+1)}$ более старшего слоя n+1.

$$\delta_j^{(n)} = \left[\sum_k \delta_k^{(n+1)} \cdot w_{jk}^{(n+1)} \right] \cdot \frac{dy_j}{ds_j} \tag{1.11}$$

Для выходного же слоя

$$\delta_l^{(N)} = (y_l^{(N)} - d_l) \cdot \frac{dy_l}{ds_l} \tag{1.12}$$

Теперь мы можем записать (1.6) в раскрытом виде:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \delta_j^{(n)} \cdot y_i^{(n-1)} \tag{1.13}$$

Иногда для придания процессу коррекции весов некоторой инерционности, сглаживающей резкие скачки при перемещении по поверхности целевой функции, (1.13) дополняется значением изменения веса на предыдущей итерации [46]

$$\Delta w_{ij}^{(n)}(t) = -\eta \cdot (\mu \cdot \Delta w_{ij}^{(n)}(t-1) + (1-\mu) \cdot \delta_j^{(n)} \cdot y_i^{(n-1)}) \tag{1.14}$$

где μ – коэффициент инерционности, t – номер текущей итерации.

Таким образом, полный алгоритм обучения ИНС с помощью процедуры обратного распространения строится так [69]:

1. Подать на входы сети один из возможных образов и в режиме обычного функционирования ИНС, когда сигналы распространяются от входов к выходам, рассчитать значения последних. Известно, что

$$s_j^{(n)} = \sum_{i=0}^{M} y_i^{(n-1)} \cdot w_{ij}^{(n)} \qquad (1.15)$$

где M – число нейронов в слое n-1 с учетом нейрона с постоянным выходным состоянием +1, задающего смещение; $y_i^{(n-1)}=x_{ij}^{(n)}$ – i-ый вход нейрона j слоя n.

$$y_j^{(n)} = f(s_j^{(n)}), \text{ где } f() \text{ – сигмоид} \qquad (1.16)$$

$$y_q^{(0)}=I_q, \qquad (1.17)$$

где I_q – q-ая компонента вектора входного образа.

2. Рассчитать $\delta^{(N)}$ для выходного слоя по формуле (1.12).

Рассчитать по формуле (1.13) или (1.14) изменения весов $\Delta w^{(N)}$ слоя N.

3. Рассчитать по формулам (1.11) и (1.13) (или (1.11) и (1.14)) соответственно $\delta^{(n)}$ и $\Delta w^{(n)}$ для всех остальных слоев, n=N-1,...1.

4. Скорректировать все веса в ИНС

$$w_{ij}^{(n)}(t) = w_{ij}^{(n)}(t-1) + \Delta w_{ij}^{(n)}(t) \qquad (1.18)$$

5. Если ошибка сети существенна, перейти на шаг 1. В противном случае – конец.

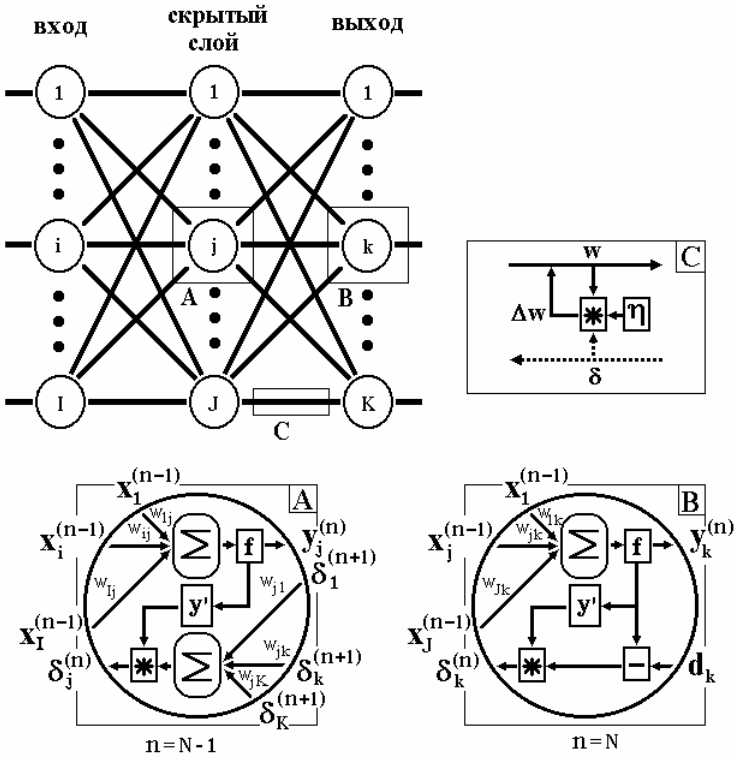


Рис. 1.11 Диаграмма сигналов в сети при обучении по алгоритму обратного распространения.

Сети на шаге 1 попеременно в случайном порядке предъявляются все тренировочные образы [36], чтобы сеть, не забывала одни образы по мере запоминания других. Алгоритм схематически представлен на рис. 1.11.

Из выражения (1.13) следует, что когда выходное значение $y_i^{(n-1)}$ стремится к нулю, эффективность обучения заметно снижается. При двоичных входных векторах в среднем половина весовых коэффициентов не будет корректироваться, поэтому область возможных значений выходов нейронов [0,1] желательно сдвинуть в пределы [-0.5,+0.5], что достигается простыми модификациями логистических функций. Например, сигмоид с экспонентой преобразуется к виду

$$f(x) = -0.5 + \frac{1}{1 + e^{-\alpha \cdot x}} \tag{1.19}$$

В процессе обучения может возникнуть ситуация, когда большие положительные или отрицательные значения весовых коэффициентов сместят рабочую точку на сигмоидах многих нейронов в область насыщения. Малые величины производной от логистической функции приведут в соответствие с (1.11) и (1.12) к остановке обучения, что парализует ИНС. Во-вторых, применение метода градиентного спуска не гарантирует, что будет найден глобальный, а не локальный минимум целевой функции. Эта проблема связана еще с одной, а именно – с выбором величины скорости обучения. Доказательство сходимости обучения в процессе обратного распространения основано на производных, то есть приращения весов и, следовательно, скорость обучения должна быть бесконечно малой, однако в этом случае обучение будет происходить неприемлемо медленно. С другой стороны, слишком большие коррекции весов могут привести к постоянной неустойчивости процесса обучения. Поэтому в качестве η обычно выбирается число меньше 1, но не очень маленькое, например, 0.1, и оно может постепенно уменьшаться в процессе обучения. Кроме того, для исключения случайных попаданий в локальные минимумы иногда, после того как значения весовых коэффициентов стабилизируются, η кратковременно сильно увеличивают, чтобы начать градиентный спуск из новой точки. Если повторение этой процедуры несколько раз приведет алгоритм в одно и то же состояние ИНС, можно более или менее уверенно сказать, что найден глобальный максимум, а не какой-то другой [67].

Существует и иной метод исключения локальных минимумов, а заодно и паралича ИНС, заключающийся в применении стохастических ИНС [74].

Алгоритм обучения нейронной сети с помощью процедуры обратного распространения подразумевает наличие некоего внешнего звена, предоставляющего сети кроме входных также и целевые выходные образы. Алгоритмы, пользующиеся подобной концепцией, называются алгоритмами обучения с учителем. Для их успешного функционирования

необходимо наличие экспертов [55], создающих на предварительном этапе для каждого входного образа эталонный выходной. Например, обучение человеческого мозга, на первый взгляд, происходит без учителя: на зрительные, слуховые, тактильные и прочие рецепторы поступает информация извне, и внутри нервной системы происходит некая самоорганизация. Однако, нельзя отрицать и того, что в жизни человека не мало учителей – и в буквальном, и в переносном смысле, – которые координируют внешние воздействия.

Главная черта, делающая обучение без учителя привлекательным, – это его "самостоятельность". Процесс обучения, как и в случае обучения с учителем, заключается в подстраивании весов синапсов. Некоторые алгоритмы изменяют структуру сети, то есть взаимосвязи нейронов или даже их количество, такие преобразования называются – самоорганизацией. Очевидно, что подстройка синапсов может проводиться только на основании информации, доступной в нейроне, то есть его состояния и уже имеющихся весовых коэффициентов. Исходя из этого соображения и по аналогии с известными принципами самоорганизации нервных клеток, построены алгоритмы обучения Хебба.

1.3.4 Нейронные сети Хебба.

Минский и Пейперт отмечали, что недостатки простых персептронов можно преодолеть как с помощью многослойных сетей, так и введением в сеть обратных связей, допускающих циркуляцию сигналов по замкнутым контурам. Использовать свойства такого рода сетей для моделирования функций мозга еще в 1949 г. предложил Хебб [61].

Согласно взглядам Хебба нервные клетки мозга соединены друг с другом большим количеством прямых и обратных возбуждающих связей и образуют нейронную сеть. Каждый нейрон осуществляет пространственно-временную суммацию приходящих к нему сигналов от возбужденных нейронов, определяя потенциал на своей мембране. Когда потенциал на мембране превышает пороговое значение, нейрон возбуждается. Нейрон обладает рефрактерностью и усталостью. Эффективность связей может изменяться в процессе функционирования сети, повышаясь между одновременно возбужденными нейронами. Это приводит к объединению нейронов в клеточные ансамбли – группы клеток, которые чаще всего возбуждались вместе, и к обособлению ансамблей друг от друга. При возбуждении достаточной части ансамбля он возбуждается целиком. Различные ансамбли могут пересекаться: один и тот же нейрон может входить в разные ансамбли. Электрическая активность мозга обусловлена последовательным возбуждением отдельных ансамблей.

Идеи Хебба оказали большое воздействие на представления о работе мозга и послужили основой для создания нейронных моделей долговременной памяти. Ансамблевую

нейронную сеть можно рассматривать как структуру, реализующую функции распределенной ассоциативной памяти. Формирование ансамблей в такой сети соответствует запоминанию образов (признаков, объектов, событий, понятий), закодированных паттерном активности нейронов, а сформированные ансамбли являются их внутренним представлением. Процесс возбуждения всего ансамбля при активации части его нейронов можно интерпретировать как извлечение запомненной информации по ее части – ключу памяти. Модель памяти на основе ансамблевой нейронной сети обладает некоторыми свойствами, присущими биологической памяти, такими, как ассоциативность, распределенность, параллельность, устойчивость к шумам или сбоям и надежность. Проводятся также структурные аналогии между ансамблевыми моделями нейронных сетей и строением коры головного мозга. Имеются экспериментальные данные о синаптической пластичности, постулированной Хеббом [61].

Сигнальный метод обучения Хебба заключается в изменении весов по следующему правилу:

$$w_{ij}(t) = w_{ij}(t-1) + \alpha \cdot y_i^{(n-1)} \cdot y_j^{(n)} \qquad (1.20)$$

где $y_i^{(n-1)}$ – выходное значение нейрона i слоя (n-1), $y_j^{(n)}$ – выходное значение нейрона j слоя n; $w_{ij}(t)$ и $w_{ij}(t-1)$ – весовой коэффициент синапса, соединяющего эти нейроны, на итерациях t и t-1 соответственно; α – коэффициент скорости обучения; n – произвольный слой сети. При обучении по данному методу [36] усиливаются связи между возбужденными нейронами.

Существует также и дифференциальный метод обучения Хебба.

$$w_{ij}(t) = w_{ij}(t-1) + \alpha \cdot \left[y_i^{(n-1)}(t) - y_i^{(n-1)}(t-1)\right] \cdot \left[y_j^{(n)}(t) - y_j^{(n)}(t-1)\right] \qquad (1.21)$$

Здесь $y_i^{(n-1)}(t)$ и $y_i^{(n-1)}(t-1)$ – выходное значение нейрона i слоя n-1 соответственно на итерациях t и t-1; $y_j^{(n)}(t)$ и $y_j^{(n)}(t-1)$ – то же самое для нейрона j слоя n. Как видно из формулы (1.21), сильнее всего обучаются синапсы, соединяющие те нейроны, выходы которых наиболее динамично изменились в сторону увеличения.

Полный алгоритм обучения с применением вышеприведенных формул будет выглядеть так:

1. На стадии инициализации всем весовым коэффициентам присваиваются небольшие случайные значения.

2. На входы сети подается входной образ, и сигналы возбуждения распространяются по всем слоям согласно принципам классических прямопоточных (feedforward) сетей [40], то есть для каждого нейрона рассчитывается взвешенная сумма его входов, к которой затем применяется активационная (передаточная) функция нейрона, в результате чего получается

его выходное значение $y_i^{(n)}$, i=0...M_i-1, где M_i – число нейронов в слое i; n=0...N-1, а N – число слоев в сети.

3. На основании полученных выходных значений нейронов по формуле (1.20) или (1.21) производится изменение весовых коэффициентов.

4. Цикл – переход на шаг 2, до тех пор, пока выходные значения сети не застабилизируются с заданной точностью.

Применение этого нового способа определения завершения обучения, отличного от использовавшегося для сети обратного распространения, обусловлено тем, что подстраиваемые значения синапсов фактически не ограничены.

На втором шаге цикла попеременно предъявляются все образы из входного набора.

Следует отметить, что вид откликов на каждый класс входных образов не известен заранее и будет представлять собой произвольное сочетание состояний нейронов выходного слоя, обусловленное случайным распределением весов на стадии инициализации. Вместе с тем, сеть способна обобщать схожие образы, относя их к одному классу. Тестирование обученной сети позволяет определить топологию классов в выходном слое. Для приведения откликов обученной сети к удобному представлению можно дополнить сеть одним слоем, который, например, по алгоритму обучения однослойного персептрона необходимо заставить отображать выходные реакции сети в требуемые образы [65].

Другой алгоритм обучения без учителя – алгоритм Кохонена – предусматривает подстройку синапсов на основании их значений от предыдущей итерации.

$$w_{ij}(t) = w_{ij}(t-1) + \alpha \cdot \left[y_i^{(n-1)} - w_{ij}(t-1) \right] \qquad (1.22)$$

Из вышеприведенной формулы видно, что обучение сводится к минимизации разницы между входными сигналами нейрона, поступающими с выходов нейронов предыдущего слоя $y_i^{(n-1)}$, и весовыми коэффициентами его синапсов.

Полный алгоритм обучения имеет примерно такую же структуру, как в методах Хебба, но на шаге 3 из всего слоя выбирается нейрон, значения синапсов которого максимально походят на входной образ, и подстройка весов по формуле (1.22) проводится только для него. Эта, так называемая, аккредитация может сопровождаться затормаживанием всех остальных нейронов слоя и введением выбранного нейрона в насыщение. Выбор такого нейрона может осуществляться, например, расчетом скалярного произведения вектора весовых коэффициентов с вектором входных значений. Максимальное произведение дает выигравший нейрон.

Другой вариант – расчет расстояния между этими векторами в p-мерном пространстве [36], где p – размер векторов.

$$D_j = \sqrt{\sum_{i=0}^{p-1} (y_i^{(n-1)} - w_{ij})^2}\ , \qquad (1.23)$$

где j – индекс нейрона в слое n, i – индекс суммирования по нейронам слоя (n-1), w_{ij} – вес синапса, соединяющего нейроны; выходы нейронов слоя (n-1) являются входными значениями для слоя n. Корень в формуле (1.23) брать не обязательно, так как важна лишь относительная оценка различных D_j.

В данном случае, "побеждает" нейрон с наименьшим расстоянием. Иногда слишком часто получающие аккредитацию нейроны принудительно исключаются из рассмотрения, чтобы "уравнять права" всех нейронов слоя. Простейший вариант такого алгоритма заключается в торможении только что выигравшего нейрона [81].

При использовании обучения по алгоритму Кохонена существует практика нормализации входных образов, а также – на стадии инициализации – и нормализации начальных значений весовых коэффициентов.

$$x_i = x_i \,/\, \sqrt{\sum_{j=0}^{n-1} x_j^2}\ , \qquad (1.24)$$

где x_i – i-ая компонента вектора входного образа или вектора весовых коэффициентов, а n – его размерность. Это позволяет сократить длительность процесса обучения.

Инициализация весовых коэффициентов случайными значениями может привести к тому, что различные классы, которым соответствуют плотно распределенные входные образы, сольются или, наоборот, раздробятся на дополнительные подклассы в случае близких образов одного и того же класса. Для избежания такой ситуации используется метод выпуклой комбинации. Суть его сводится к тому, что входные нормализованные образы подвергаются преобразованию:

$$x_i = \alpha(t) \cdot x_i + (1 - \alpha(t)) \cdot \frac{1}{\sqrt{n}}, \qquad (1.25)$$

где x_i – i-ая компонента входного образа, n – общее число его компонент, $\alpha(t)$ – коэффициент, изменяющийся в процессе обучения от нуля до единицы, в результате чего вначале на входы сети подаются практически одинаковые образы, а с течением времени они все больше сходятся к исходным. Весовые коэффициенты устанавливаются на шаге инициализации равными величине

$$w_o = \frac{1}{\sqrt{n}}, \qquad (1.26)$$

где n – размерность вектора весов для нейронов инициализируемого слоя.

На основе рассмотренного выше метода строятся нейронные сети особого типа – так называемые самоорганизующиеся структуры с подстройкой синапсов – self-organizing feature

maps. Для них после выбора из слоя n нейрона j с минимальным расстоянием D_j (1.23) обучается по формуле (1.22) не только этот нейрон, но и его соседи, расположенные в окрестности R. Величина R на первых итерациях очень большая, так что обучаются все нейроны, но с течением времени она уменьшается до нуля. Таким образом, чем ближе конец обучения, тем точнее определяется группа нейронов, отвечающих каждому классу образов.

1.3.5 Нейронные сети Хопфилда.

Среди различных конфигураций искусственных нейронных сетей встречаются такие, при классификации которых по принципу обучения, строго говоря, не подходят ни обучение с учителем, ни обучение без учителя. В таких сетях весовые коэффициенты синапсов рассчитываются только однажды перед началом функционирования сети на основе информации об обрабатываемых данных, и все обучение сети сводится именно к этому расчету. С одной стороны, предъявление априорной информации можно расценивать, как помощь учителя, но с другой – сеть фактически просто запоминает образцы до того, как на ее вход поступают реальные данные, и не может изменять свое поведение, поэтому говорить о звене обратной связи с "миром" (учителем) не приходится. Из сетей с подобной логикой работы наиболее известны сеть Хопфилда и сеть Хэмминга, которые обычно используются для организации ассоциативной памяти.

Структурная схема сети Хопфилда приведена на рис. 1.12. Она состоит из единственного слоя нейронов, число которых является одновременно числом входов и выходов сети [36]. Каждый нейрон связан синапсами со всеми остальными нейронами, а также имеет один входной синапс, через который осуществляется ввод сигнала. Выходные сигналы, как обычно, образуются на аксонах.

Задача, решаемая данной сетью в качестве ассоциативной памяти, как правило, формулируется следующим образом. Известен некоторый набор двоичных сигналов (изображений, звуковых оцифровок, прочих данных, описывающих некие объекты или характеристики процессов), которые считаются образцовыми. Сеть должна уметь из произвольного неидеального сигнала, поданного на ее вход, выделить ("вспомнить" по частичной информации) соответствующий образец (если такой есть) или "дать заключение" о том, что входные данные не соответствуют ни одному из образцов. В общем случае, любой сигнал может быть описан вектором $\mathbf{X} = \{x_i: i=0, ..., n-1\}$, n – число нейронов в сети и размерность входных и выходных векторов. Каждый элемент x_i равен либо +1, либо -1. Обозначим вектор, описывающий k-ый образец, через $\mathbf{X^k}$, а его компоненты, соответственно,

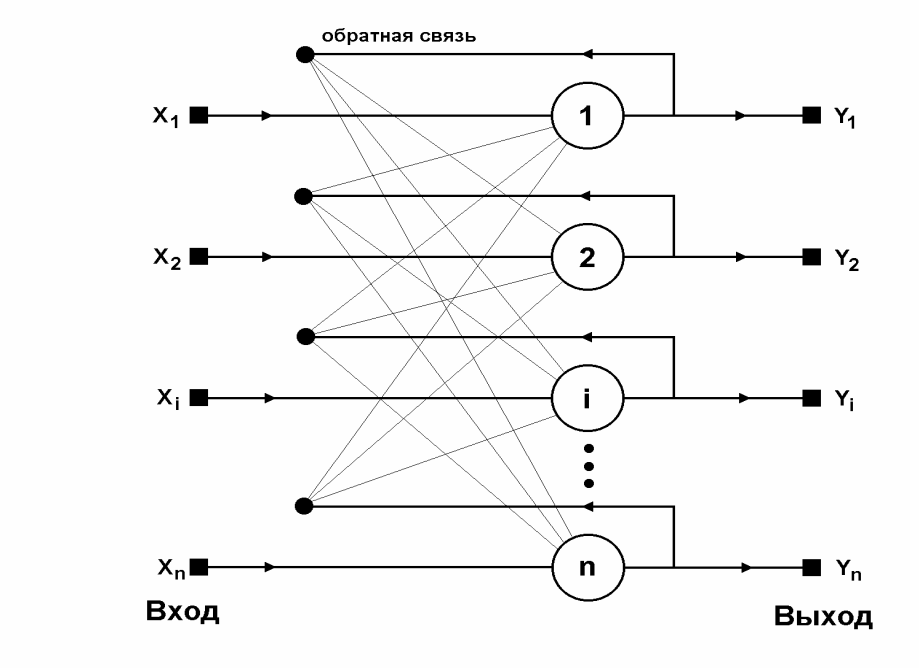


Рис. 1.12 Структурная схема сети Хопфилда.

– x_i^k, k=0...m-1, m – число образцов. Когда сеть распознает какой-либо образец на основе предъявленных ей данных, ее выходы будут содержать именно его, то есть $\mathbf{Y} = \mathbf{X^k}$, где **Y** – вектор выходных значений сети: $\mathbf{Y} = \{y_i: i=0, ..., n-1\}$. В противном случае, выходной вектор не совпадет ни с одним образцовым.

Если, например, сигналы представляют собой некие изображения, то, отобразив в графическом виде данные с выхода сети, можно будет увидеть картинку, полностью совпадающую с одной из образцовых (в случае успеха) или же "вольную импровизацию" сети (в случае неудачи).

На стадии инициализации сети весовые коэффициенты синапсов устанавливаются следующим образом:

$$w_{ij} = \begin{cases} \sum_{k=0}^{m-1} x_i^k x_j^k, i \neq j \\ 0, \qquad i = j \end{cases} \tag{1.27}$$

Здесь i и j – индексы, соответственно, предсинаптического и постсинаптического нейронов; x_i^k, x_j^k – i-ый и j-ый элементы вектора k-ого образца [73].

Алгоритм функционирования сети следующий (p – номер итерации):

1. На входы сети подается неизвестный сигнал. Фактически его ввод осуществляется непосредственной установкой значений аксонов:

$$y_i(0) = x_i, \; i = 0...n-1, \tag{1.28}$$

поэтому обозначение на схеме сети входных синапсов в явном виде носит чисто условный характер. Ноль в скобке справа от y_i означает нулевую итерацию в цикле работы сети.

2. Рассчитывается новое состояние нейронов

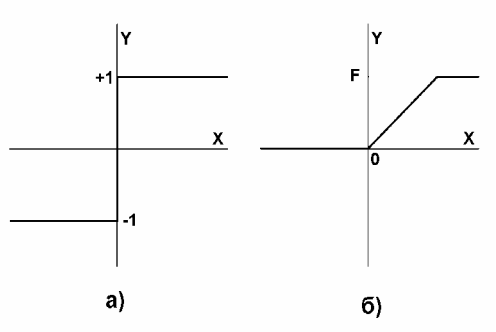
$$s_j(p+1) = \sum_{i=0}^{n-1} w_{ij} y_i(p), \; j=0...n-1 \qquad (1.29)$$

и новые значения аксонов

$$y_j(p+1) = f\left[s_j(p+1)\right] \qquad (1.30)$$

где f – активационная функция в виде скачка, приведенная на рис 1.13а.

Рис. 1.13 Активационные функции.



3. Проверка, изменились ли выходные значения аксонов за последнюю итерацию. Если да – переход к пункту 2, иначе (если выходы застабилизировались) – конец. При этом выходной вектор представляет собой образец, наилучшим образом сочетающийся с входными данными.

Иногда сеть не может провести распознавание и выдает на выходе несуществующий образ. Это связано с проблемой ограниченности возможностей сети. Для сети Хопфилда число запоминаемых образов m не должно превышать величины, примерно равной 0.15*n. Кроме того, если два образа А и Б сильно похожи, они, возможно, будут вызывать у сети перекрестные ассоциации, то есть предъявление на входы сети вектора А приведет к появлению на ее выходах вектора Б и наоборот.

1.3.6 Нейронные сети Хэмминга.

Когда нет необходимости, чтобы сеть в явном виде выдавала образец, то есть достаточно получать номер образца, ассоциативную память успешно реализует сеть Хэмминга. Данная сеть характеризуется, по сравнению с сетью Хопфилда, меньшими затратами на память и объемом вычислений, что становится очевидным из ее структуры.

Сеть Хэмминга (рис. 1.14) состоит из двух слоев. Первый и второй слои имеют по m нейронов, где m – число образцов. Нейроны первого слоя имеют по n синапсов, соединенных с входами сети (образующими фиктивный нулевой слой). Нейроны второго слоя связаны между собой ингибиторными (отрицательными обратными) синаптическими связями [63]. Единственный синапс с положительной обратной связью для каждого нейрона соединен с его же аксоном.

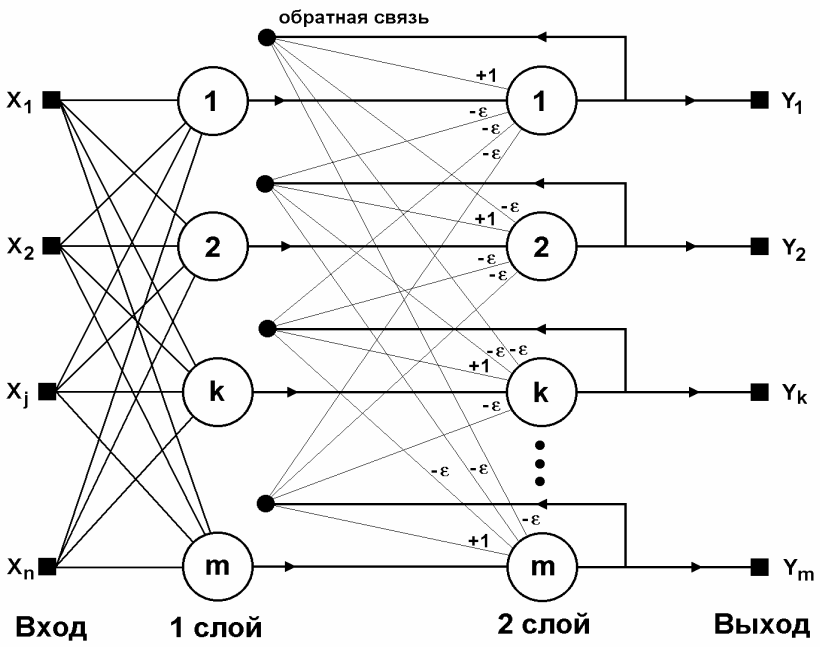


Рис. 1.14 Структурная схема сети Хэмминга.

Идея работы сети состоит в нахождении расстояния Хэмминга от тестируемого образа до всех образцов [36]. Расстоянием Хэмминга называется число отличающихся битов в двух бинарных векторах. Сеть должна выбрать образец с минимальным расстоянием Хэмминга до неизвестного входного сигнала, в результате чего будет активизирован только один выход сети, соответствующий этому образцу.

На стадии инициализации весовым коэффициентам первого слоя и порогу активационной функции присваиваются следующие значения:

$$w_{ik} = \frac{x_i^k}{2},\ i=0...n-1,\ k=0...m-1 \qquad (1.31)$$

$$T_k = n / 2,\ k = 0...m-1 \qquad (1.32)$$

Здесь x_i^k – i-ый элемент k-ого образца.

Весовые коэффициенты тормозящих синапсов во втором слое берут равными некоторой величине $0 < \varepsilon < 1/m$. Синапс нейрона, связанный с его же аксоном имеет вес +1.

Алгоритм функционирования сети Хэмминга следующий:

1. На входы сети подается неизвестный вектор $\mathbf{X} = \{x_i: i=0...n-1\}$, исходя из которого рассчитываются состояния нейронов первого слоя (верхний индекс в скобках указывает номер слоя):

$$y_j^{(1)} = s_j^{(1)} = \sum_{i=0}^{n-1} w_{ij} x_i + T_j,\ j=0...m-1 \qquad (1.33)$$

После этого полученными значениями инициализируются значения аксонов второго слоя:

$$y_j^{(2)} = y_j^{(1)}, j = 0...m\text{-}1 \quad (1.34)$$

2. Вычислить новые состояния нейронов второго слоя:

$$s_j^{(2)}(p+1) = y_j(p) - \varepsilon \sum_{k=0}^{m-1} y_k^{(2)}(p), k \neq j, j = 0...m-1 \quad (1.35)$$

и значения их аксонов:

$$y_j^{(2)}(p+1) = f\left[s_j^{(2)}(p+1)\right], \; j = 0...m-1 \quad (1.36)$$

Активационная функция f имеет вид порога (рис. 1.13б), причем величина F должна быть достаточно большой, чтобы любые возможные значения аргумента не приводили к насыщению [64].

3. Проверить, изменились ли выходы нейронов второго слоя за последнюю итерацию. Если да – перейди к шагу 2. Иначе – конец.

Из оценки алгоритма видно, что роль первого слоя весьма условна: воспользовавшись один раз на шаге 1 значениями его весовых коэффициентов, сеть больше не обращается к нему, поэтому первый слой может быть вообще исключен (заменен на матрицу весовых коэффициентов) из сети.

В настоящее время основными направлениями реализации Нейроподобной сети (НПС) являются:

- программная реализация на цифровых ЭВМ традиционной архитектуры;
- программно-аппаратная реализация в виде сопроцессоров к ЭВМ общего назначения;
- аппаратная реализация путем создания нейрокомпьютеров на базе нейроплат в виде параллельных нейроподобных структур [27].

ГЛАВА 2. ПРОБЛЕМА КЛАССИФИКАЦИИ ОБРАЗОВ.

2.1 Рецепторная структура восприятия информации.

Для того, чтобы человек сознательно воспринял информацию, она должна пройти довольно длительный цикл предварительной обработки. Вначале свет попадает в глаз. Пройдя через всю оптическую систему фотоны, в конце концов, попадают на сетчатку – слой светочувствительных клеток – палочек и колбочек.

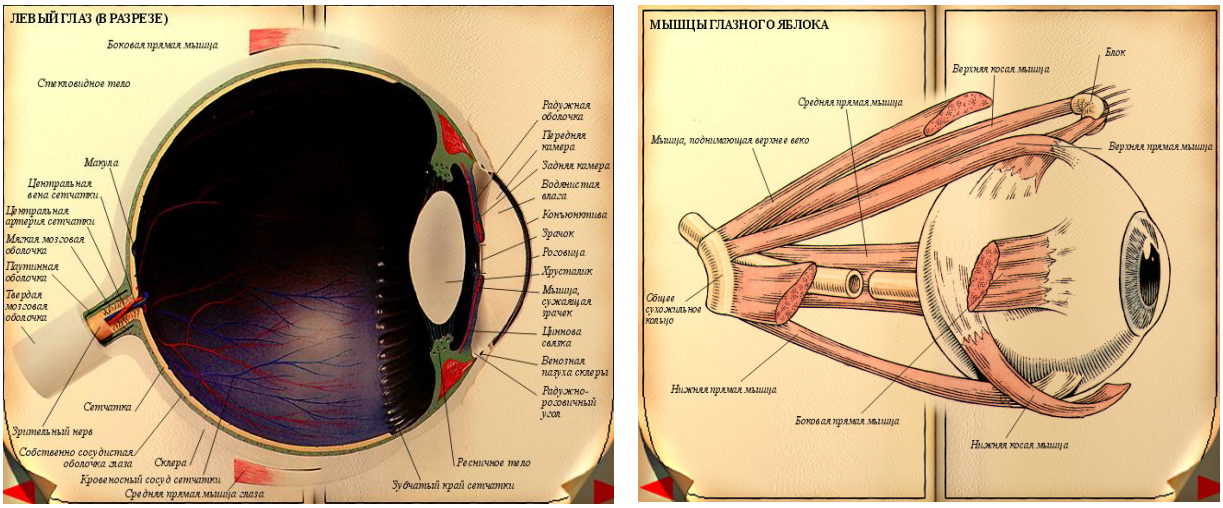


Рис. 2.1 Структура зрительного аппарата.

Уже здесь – еще очень далеко от головного мозга, происходит первый этап обработки информации, поскольку, например, у млекопитающих, сразу за светочувствительными клетками находится обычно два слоя нервных клеток, которые выполняют сравнительно несложную обработку.

Теперь информация поступает по зрительному нерву в головной мозг человека, в так называемые "зрительные бугры", на которые проецируется зрительная информация то, что мы видим. Далее зрительная информация поступает в отделы мозга, которые уже выделяют из нее отдельные составляющие – горизонтальные, вертикальные, диагональные линии, контуры, области светлого, темного, цветного. Постепенно образы становятся все более сложными и размытыми, но графический образ картины пройдет еще долгий путь, прежде чем достигнет уровня сознания.

К проблеме распознавания образов можно подходить, отталкиваясь от аналогии с биологическими процессами. В некоторых условиях способности животных к распознаванию образов превышают способности любой машины, которую только можно построить [9].

При классификации, основанной на непосредственном сенсорном опыте, т.е. при распознавании лиц или произнесенных слов, люди легко превосходят технические устройства. В “несенсорных ситуациях” действия людей не столь эффективны. Например, люди не могут соперничать с программами классификации образов, если правильный образ классификации включает логические комбинации абстрактных свойств, таких, как цвет, размер и форма. Поскольку распознавание образов должно быть функцией нейронов животного, можно искать ключ к биологическому распознаванию образов в свойствах самого нейрона. Для многих целей нейрон можно рассматривать как пороговый элемент. Это значит, что он либо дает на выходе некоторую постоянную величину, если сумма его входов достигает определенного значения, либо же остается пассивным. Мак-Каллок и Питтс доказали, что любую вычислимую функцию можно реализовать с помощью должным образом организованной сети идеальных нейронов – пороговых элементов, логические свойства которых с достаточным основанием можно приписать реальному нейрону. Проблема состоит в том, можно ли найти разумный принцип реорганизации сети, позволяющий случайно объединенной вначале группе идеальных нейронов самоорганизоваться в “вычислительное устройство”, способное решать произвольную задачу распознавания образов. Такой принцип реорганизации явился бы теорией обучения, применимой на уровне отдельного нейрона [16].

Нейрологическая теория обучения, выдвинутая канадским психологом Хеббом, была рассчитана на использование в качестве модели, предназначенной для психологии, оказала большое влияние на искусственный интеллект. Ее модификация применялась при определении принципов системы распознавания образов, получивших название персептрон. Персептроны, описанные Розенблаттом, могут существовать и в форме программ, и как специально сконструированные вычислительные машины.

2.1.1 Понятие образа.

Образ, класс – классификационная группировка в системе классификации, объединяющая определенную группу объектов по некоторому признаку. Образное восприятие мира – одно из свойств живого мозга, позволяющее разобраться в бесконечном потоке воспринимаемой информации и сохранять ориентацию в разрозненных данных о внешнем мире. Воспринимая внешний мир, мы всегда производим классификацию информации, т. е. разбиваем их на группы похожих, но не тождественных явлений. Например, несмотря на существенное различие, к одной группе относятся все буквы “А”, написанные различными почерками, или все звуки, соответствующие одной и той же ноте,

взятой в любой октаве и на любом инструменте. Для составления понятия о группе восприятий достаточно ознакомиться с незначительным количеством ее представителей. Это свойство мозга позволяет сформулировать такое понятие, как образ.

Образы обладают характерным свойством, проявляющимся в том, что ознакомление с конечным числом явлений из одного и того же множества дает возможность узнавать сколь угодно большое число его представителей [12]. Образы обладают характерными объективными свойствами в том смысле, что разные люди, обучающиеся на различном материале наблюдений, большей частью одинаково и независимо друг от друга классифицируют одни и те же объекты. Именно эта объективность образов позволяет людям всего мира понимать друг друга.

Способность восприятия внешнего мира в форме образов позволяет с определенной достоверностью узнавать бесконечное число объектов на основании ознакомления с конечным их числом, а объективный характер основного свойства образов позволяет моделировать процесс их распознавания.

2.1.2 Проблема распознавания образов.

Распознавание образов – это задача идентификации объекта или определения каких-либо его свойств по его изображению (оптическое распознавание) или аудиозаписи (акустическое распознавание) [25]. В процессе биологической эволюции многие животные с помощью зрительного и слухового аппарата решили эту задачу достаточно хорошо. Создание искусственных систем с функциями распознавания образов остаётся сложной технической проблемой.

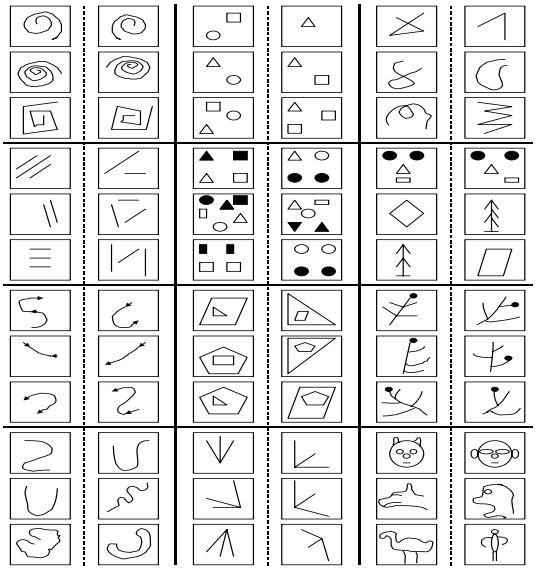


Рис. 2.2 Пример объектов обучения.

В целом проблема распознавания образов (ПРО) состоит из двух частей: обучения и распознавания. Обучение осуществляется путем показа отдельных объектов с указанием их принадлежности тому или другому образу. В результате обучения распознающая система должна приобрести способность реагировать одинаковыми реакциями на все объекты одного образа и другими реакциями - на все объекты отличимых образов. Очень важно, что процесс обучения должен завершиться только путем показов конечного числа объектов. В качестве объектов обучения могут быть либо картинки (рис. 2.2), либо другие визуальные изображения (буквы, цифры) [36]. Важно, что в процессе обучения указываются только сами объекты и их принадлежность образу. За обучением следует процесс распознавания новых объектов, который характеризует действия уже обученной системы. Автоматизация этих процедур и составляет проблему обучения распознаванию образов [10]. В том случае, когда человек сам разгадывает или придумывает, а затем навязывает машине правило классификации, проблема распознавания решается частично, так как основную и главную часть проблемы (обучение) человек берет на себя.

Круг задач, которые могут решаться с помощью распознающих систем, чрезвычайно широк. Сюда относятся не только задачи распознавания зрительных и слуховых образов, но и задачи классификации сложных процессов и явлений, возникающих, например, при выборе целесообразных действий руководителем предприятия или выборе оптимального управления технологическими, экономическими, транспортными или военными задачами. Прежде чем начать анализ какого-либо объекта, нужно получить о нем определенную, упорядоченную информацию.

Выбор исходного описания объектов является одной из центральных задач проблемы распознавания образов. При удачном выборе исходного описания (пространства признаков) задача распознавания может оказаться тривиальной и, наоборот, неудачно выбранное исходное описание может привести либо к очень сложной дальнейшей переработку информации, либо вообще к отсутствию решения.

2.1.3 Геометрический и структурный подходы.

Любое изображение, которое возникает в результате наблюдения какого-либо объекта в процессе обучения или экзамена, можно представить в виде вектора, а значит и в виде точки некоторого пространства признаков. Если утверждается, что при показе изображений возможно однозначно отнести их к одному из двух (или нескольких) образов, то тем самым утверждается, что в некотором пространстве существует две (или несколько) области, не

имеющие общих точек, и что изображения – точки из этих областей. Каждой такой области можно приписать наименование, т. е. дать название, соответствующее образу.

Проинтерпретируем теперь в терминах геометрической картины процесс обучения распознаванию образов, ограничившись пока случаем распознавания только двух образов. Заранее считается известным лишь только то, что требуется разделить две области в некотором пространстве, и что показываются точки только из этих областей. Сами эти области заранее не определены, т. е. нет каких-либо сведений о расположении их границ или правил определения принадлежности точки к той или иной области.

В ходе обучения предъявляются точки, случайно выбранные из этих областей, и сообщается информация о том, к какой области принадлежат предъявляемые точки. Никакой дополнительной информации об этих областях, т. е. о расположении их границ, в ходе обучения не сообщается. Цель обучения состоит либо в построении поверхности, которая разделяла бы не только показанные в процессе обучения точки, но и все остальные точки, принадлежащие этим областям, либо в построении поверхностей, ограничивающих эти области так, чтобы в каждой из них находились только точки одного образа. Цель обучения состоит в построении таких функций от векторов-изображений, которые были бы, например, положительны на всех точках одного и отрицательны на всех точках другого образа. В связи с тем, что области не имеют общих точек, всегда существует целое множество таких разделяющих функций, а в результате обучения должна быть построена одна из них.

Если предъявляемые изображения принадлежат не двум, а большему числу образов, то задача состоит в построении по показанным в ходе обучения точкам поверхности, разделяющей все области, соответствующие этим образам, друг от друга. Задача эта может быть решена, например, путем построения функции, принимающей над точками каждой из областей одинаковое значение, а над точками из разных областей значение этой функции должно быть различно [15].

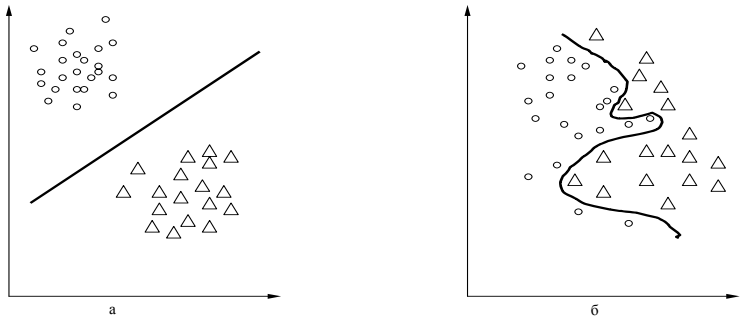


Рис. 2.3 Разделение двух образов в пространстве.

На первый взгляд кажется, что знание всего лишь некоторого количества точек из области недостаточно, чтобы отделить всю область. Действительно, можно указать

бесчисленное количество различных областей, которые содержат эти точки, и как бы ни была построена по ним поверхность, выделяющая область, всегда можно указать другую область, которая пересекает поверхность и вместе с тем содержит показанные точки. Однако известно, что задача о приближении функции по информации о ней в ограниченном множестве точек, существенно более узкой, чем все множество, на котором функция задана, является обычной математической задачей об аппроксимации функций. Решение таких задач требует введения определенных ограничений на классе рассматриваемых функций, а выбор этих ограничений зависит от характера информации, которую может добавить учитель в процессе обучения. Одной из таких подсказок является гипотеза о компактности образов. Аппроксимация разделяющей функции будет задачей тем более легкой, чем более компактны и чем более разнесены в пространстве области, подлежащие разделению. Так, например, в случае, показанном на рис. 2.3а, разделение заведомо более просто, чем в случае, показанном на рис. 2.3б. Действительно, в случае, изображенном на рис 2.3а, области могут быть разделены плоскостью, и даже при больших погрешностях в определении разделяющей функции она все же будет продолжать разделять области [36]. В случае же на рис. 2б, разделение осуществляется замысловатой поверхностью, и даже незначительные отклонения в ее форме приводят к ошибкам разделения. Именно это интуитивное представление о сравнительно легко разделимых областях привело к гипотезе компактности.

Наряду с геометрической интерпретацией проблемы обучения распознаванию образов существует и иной подход, который назван структурным, или лингвистическим [53]. Поясним лингвистический подход на примере распознавания зрительных изображений. Сначала выделяется набор исходных понятий – типичных фрагментов, встречающихся на изображениях, и характеристик взаимного расположения фрагментов – "слева", "снизу", "внутри" и т. д. Эти исходные понятия образуют словарь, позволяющий строить различные логические высказывания. Задача состоит в том, чтобы из большого количества высказываний, которые могли бы быть построены с использованием этих понятий, отобрать наиболее существенные для данного конкретного случая.

Далее, просматривая конечное и по возможности небольшое число объектов из каждого образа, нужно построить описание этих образов. Построенные описания должны быть столь полными, чтобы решить вопрос о том, к какому образу принадлежит данный объект. При реализации лингвистического подхода возникают две задачи: задача построения исходного словаря, т. е. набор типичных фрагментов, и задача построения правил описания из элементов заданного словаря.

В рамках лингвистической интерпретации проводится аналогия между структурой изображений и синтаксисом языка. Стремление к этой аналогии было вызвано

возможностью, использовать аппарат математической лингвистики, т. е. методы по своей природе являются синтаксическими. Использование аппарата математической лингвистики для описания структуры изображений можно применять только после того, как произведена сегментация изображений на составные части, т. е. выработаны слова для описания типичных фрагментов и методы их поиска. После предварительной работы, обеспечивающей выделение слов, возникают собственно лингвистические задачи, состоящие из задач автоматического грамматического разбора описаний для распознавания изображений. При этом проявляется самостоятельная область исследований, которая требует не только знания основ математической лингвистики, но и овладения приемами, которые разработаны специально для лингвистической обработки изображений [45].

2.1.4 Гипотеза компактности.

Если предположить, что в процессе обучения пространство признаков формируется исходя из задуманной классификации, то задание в пространстве признаков само по себе задает свойство, под действием которого образы в этом пространстве легко разделяются. Гипотеза компактности гласит: компактным образам соответствуют компактные множества в пространстве признаков. Под компактным множеством понимаются некие "сгустки" точек в пространстве изображений, предполагая, что между этими сгустками существуют разделяющие их разряжения. Эту гипотезу не всегда удавалось подтвердить экспериментально, однако те задачи, в рамках которых гипотеза компактности хорошо выполнялась (рис. 2.3.а), находилось простое решение. И наоборот, те задачи, для которых гипотеза не подтверждалась (рис. 2.3б), либо совсем не решались, либо решались с большим трудом [36]. Этот факт заставил, по меньшей мере, усомниться в справедливости гипотезы компактности, так как для опровержения любой гипотезы достаточно одного отрицающего ее примера. Вместе с этим, выполнение гипотезы всюду там, где удавалось хорошо решить задачу обучения распознаванию образов, сохраняло к этой гипотезе интерес. Сама гипотеза компактности превратилась в признак возможности удовлетворительного решения задач распознавания.

Формулировка гипотезы компактности подводит вплотную к понятию абстрактного образа. Если координаты пространства выбирать случайно, то и изображения в нем будут распределены случайно. Они будут в некоторых частях пространства располагаться более плотно, чем в других. В случайно выбранном абстрактном пространстве почти наверняка будут существовать компактные множества точек. Поэтому в соответствии с гипотезой

компактности множества объектов, которым в абстрактном пространстве соответствуют компактные множества точек, будут абстрактными образами данного пространства.

2.1.5 Обучение, самообучение и адаптация.

Обучение – это процесс, в результате которого система постепенно приобретает способность отвечать нужными реакциями на определенные совокупности внешних воздействий, а адаптация – это подстройка параметров и структуры системы с целью достижения требуемого качества управления в условиях непрерывных изменений внешних условий. Все картинки, представленные на рис. 2.2, характеризуют задачу обучения. В каждой из этих задач задается несколько примеров (обучающая последовательность) правильно решенных задач. Если бы удалось подметить некое всеобщее свойство, не зависящее ни от природы образов, ни от их изображений, а определяющее лишь их способность к разделимости, то наряду с обычной задачей обучения распознаванию с использованием информации о принадлежности каждого объекта из обучающей последовательности тому или иному образу, можно было бы поставить иную классификационную задачу – так называемую задачу обучения без учителя. Задачу такого рода на описательном уровне можно сформулировать следующим образом: системе одновременно или последовательно предъявляются объекты без каких-либо указаний об их принадлежности к образам. Входное устройство системы отображает множество объектов на множество изображений и, используя некоторое заложенное в нее заранее свойство разделимости образов, производит самостоятельную классификацию этих объектов. После такого процесса самообучения система должна приобрести способность к распознаванию не только уже знакомых объектов (объектов из обучающей последовательности), но и тех, которые ранее не предъявлялись. Процессом самообучения некоторой системы называется такой процесс, в результате которого эта система без подсказки учителя приобретает способность к выработке одинаковых реакций на изображения объектов одного и того же образа и различных реакций на изображения различных образов [36]. Роль учителя при этом состоит лишь в подсказке системе некоторого объективного свойства, одинакового для всех образов и определяющего способность к разделению множества объектов на образы. Таким объективным свойством является свойство компактности образов. Взаимное расположение точек в выбранном пространстве уже содержит информацию о том, как следует разделить множество точек. Эта информация и определяет то свойство разделимости образов, которое оказывается достаточным для самообучения системы распознаванию образов.

Обучением обычно называют процесс выработки в некоторой системе той или иной реакции на группы внешних идентичных сигналов путем многократного воздействия на систему внешней корректировки. Такую внешнюю корректировку в обучении принято называть "поощрениями" и "наказаниями". Механизм генерации этой корректировки практически полностью определяет алгоритм обучения [18]. Самообучение отличается от обучения тем, что здесь дополнительная информация о верности реакции системе не сообщается.

Большинство известных алгоритмов самообучения способны выделять только абстрактные образы, т. е. компактные множества в заданных пространствах. Различие между ними состоит, в формализации понятия компактности. Результат самообучения характеризует пригодность выбранного пространства для конкретной задачи обучения распознаванию. Если абстрактные образы, выделяемые в процессе самообучения, совпадают с реальными, то пространство выбрано удачно. Чем сильнее абстрактные образы отличаются от реальных, тем "неудобнее" выбранное пространство для конкретной задачи.

Адаптация – это процесс изменения параметров и структуры системы, а возможно, и управляющих воздействий на основе текущей информации с целью достижения определенного состояния системы при начальной неопределенности и изменяющихся условиях работы.

Возможен способ построения распознающих машин, основанный на различении каких-либо признаков подлежащих распознаванию фигур. В качестве признаков могут быть выбраны различные особенности фигур, например, их геометрические свойства (характеристики составляющих фигуры кривых), топологические свойства (взаимное расположение элементов фигуры) и т.п. Известны распознающие машины, в которых различение букв или цифр производится, по так называемому “методу зондов” (рис. 2.4), т.е. по числу пересечений контура фигуры с несколькими особым образом расположенными прямыми [21].

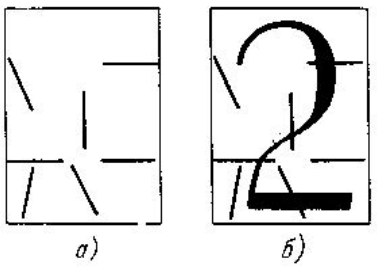


Рис. 2.4 Схема расположения зондов для распознавания цифр.

Если проектировать цифры на поле с зондами, то окажется, что каждая из цифр пересекает вполне определенные зонды, причем комбинации пересекаемых зондов различны

для всех десяти цифр. Эти комбинации и используются в качестве признаков, по которым производится различение цифр. Такие машины успешно справляются, например, с чтением машинописного текста, но их возможности ограничены тем шрифтом (или группой сходных шрифтов), для которого была разработана система признаков [32]. Работа по созданию набора эталонных фигур или системы признаков должна производиться человеком. Качество работы машины, т. е. надежность “узнавания” предъявляемых фигур определяется качеством этой предварительной подготовки и без участия человека не может быть повышено. Описанная машина не являются обучающейся машиной.

Моделирование процесса обучения подразумевает обучение, которому не предшествует сообщение машине каких-либо сведений о тех образах, распознаванию которых она должна научиться; само обучение заключается в предъявлении машине некоторого конечного числа объектов каждого образа [75]. В результате обучения машина должна оказаться способной узнавать сколь угодно большое число новых объектов, относящихся к тем же образам. Таким образом, имеется в виду следующая схема экспериментов:

а) никакие сведения о подлежащих классификации образах в машину заранее не вводятся;

б) в ходе обучения машине предъявляется некоторое количество объектов каждого из подлежащих классификации образов и (при моделировании процесса обучения “с учителем”) сообщается, к какому образу относится каждый объект;

в) машина автоматически обрабатывает полученную информацию, после чего

г) с достаточной надежностью различает сколь угодно большое число новых, ранее ей не предъявлявшихся объектов из образов.

Машины, работающие по такой схеме, называются узнающими машинами.

2.1.6 Преобразование изображений в цифровой код.

Для того чтобы ввести изображение в машину, нужно перевести его на машинный язык, т.е. закодировать, представить в виде некоторой комбинации символов, которыми может оперировать машина. Кодирование плоских фигур можно осуществить самым различным образом. Лучше стремиться к наиболее “естественному” кодированию изображений. Будем рисовать фигуры на некотором поле, разбитом вертикальными и горизонтальными прямыми на одинаковые элементы – квадратики. Элементы, на которые упало изображение, будем сплошь зачернять, остальные – оставлять белыми. Условимся обозначать черные элементы единицей, белые – нулем. Введем последовательную

нумерацию всех элементов поля, например, в каждой строке слева направо и по строкам сверху вниз. Тогда каждая фигура, нарисованная на таком поле, будет однозначно отображаться кодом, состоящим из стольких цифр (единиц и нулей), сколько элементов содержит поле.

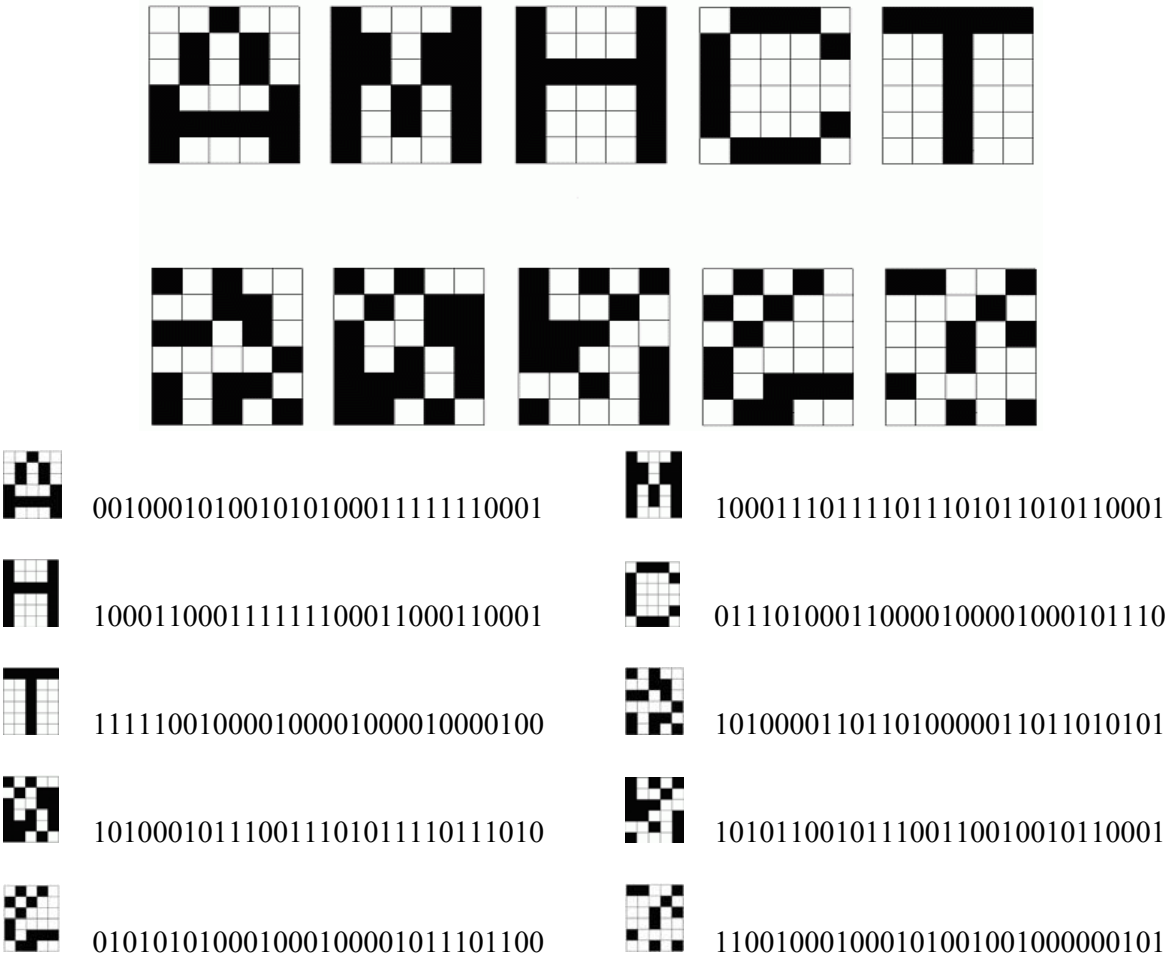


Рис 2.5 Примеры проецирования и кодирования изображений.

Такое кодирование (рис. 2.5) считается “естественным” потому, что разбиение изображения на элементы лежит в основе работы нашего зрительного аппарата. Действительно, сетчатка глаза (рис. 2.1) состоит из большого числа отдельных чувствительных элементов (так называемых палочек и колбочек), связанных нервными волокнами со зрительными отделами головного мозга. Чувствительные элементы сетчатки передают по своим нервным волокнам в головной мозг сигналы, интенсивность которых зависит от освещенности данного элемента. Таким образом, изображение, спроектированное оптической системой глаза на сетчатку, разбивается палочками и колбочками на отдельные участки, и по элементам в некотором коде передается в мозг [36]. Отдельные элементы поля называются рецепторами, а само поле – полем рецепторов.

Совокупность всех плоских фигур, которые можно изобразить на поле рецепторов, составляет некое множество. Каждая конкретная фигура из этой совокупности есть объект этого множества. Любому их таких объектов соответствует определенный код. Точно также любому коду соответствует определенное изображение на поле рецепторов. Взаимно однозначное соответствие между кодами и изображениями позволит оперировать только кодами, помня о том, что изображение всегда может быть воспроизведено по его коду.

Емкость ИНС – число образов, предъявляемых на входы ИНС для распознавания. Для разделения множества входных образов, например, по двум классам достаточно всего одного выхода. При этом каждый логический уровень – "1" и "0" – будет обозначать отдельный класс [47]. На двух выходах можно закодировать уже 4 класса и так далее. Для повышения достоверности классификации желательно ввести избыточность путем выделения каждому классу одного нейрона в выходном слое или, что еще лучше, нескольких, каждый из которых обучается определять принадлежность образа к классу со своей степенью достоверности, например: высокой, средней и низкой. Такие ИНС позволяют проводить классификацию входных образов, объединенных в нечеткие (размытые или пересекающиеся) множества. Это свойство приближает подобные ИНС к условиям реальной жизни.

2.2 Алгоритмические построения.

Алгоритмическая универсальность ЭВМ означает, что на них можно программно реализовывать (т. е. представить в виде машинной программы) любые алгоритмы преобразования информации, будь то вычислительные алгоритмы, алгоритмы управления, поиска доказательства теорем, трехмерные графические или аудио композиции.

Однако не следует думать, что вычислительные машины и роботы могут в принципе решать любые задачи [34]. Было строго доказано существование таких типов задач, для которых невозможен единый и эффективный алгоритм, решающий все задачи данного типа; в этом смысле невозможно решение таких задач и с помощью вычислительных машин. Этот факт способствует лучшему пониманию того, что могут делать машины и чего они не могут сделать.

2.2.1 Обучение искусственных нейронных сетей.

Среди свойств искусственных нейронных сетей основным является их способность к обучению, они обучаются самыми разнообразными методами. Большинство методов обучения исходят из общих предпосылок, и имеет много идентичных характеристик. Их

обучение напоминает процесс интеллектуального развития человеческой личности. Возможности обучения искусственных нейронных сетей ограниченны. Тем не менее, уже получены убедительные достижения, такие как “говорящая сеть” Сейновского, и возникает много других практических применений [58]. Сеть обучается, чтобы для некоторого множества входов давать необходимое 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.

Обучающие алгоритмы могут быть классифицированы как алгоритмы обучения с учителем и без учителя.

2.2.2 Обучение с учителем.

При обучении с учителем существует учитель, который предъявляет входные образы сети, сравнивает результирующие выходы с требуемыми значениями, а затем настраивает веса сети таким образом, чтобы уменьшить различия. Обучение с учителем предполагает, что для каждого входного вектора существует целевой вектор, представляющий собой требуемый выход [35]. 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, вычисляются ошибки и веса подстраиваются для каждого вектора до тех пор, пока ошибка по всему обучающему массиву не достигнет приемлемо низкого уровня.

2.2.3 Обучение без учителя.

Обучение с учителем критиковалось за свою биологическую неправдоподобность. Трудно вообразить обучающий механизм в мозге, который бы сравнивал желаемые и действительные значения выходов, выполняя коррекцию с помощью обратной связи. Обучение без учителя является намного более правдоподобной моделью обучения в биологической системе. Развитая Кохоненом и другими, она 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, следовательно, 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 производиться данным классом входных векторов. Следовательно, выходы подобной сети должны трансформироваться в некоторую понятную форму, обусловленную процессом обучения.

2.2.4 Процесс обучения нейронных сетей.

Процесс обучения Искусственной Нейронной Сети (ИНС) новому классу задач включает следующие стадии [35]:

1. Формулируется постановка задачи и выделяется набор ключевых параметров, характеризующих предметную область.
2. Выбирается парадигма нейронной сети (модель, включающая в себя вид входных данных, пороговой функции, структуры сети и алгоритмов обучения), наиболее подходящая для решения данного класса задач. Как правило, современные нейропакеты, нейроплаты и нейрокомпьютеры [26] позволяют реализовать не одну, а несколько базовых парадигм.
3. Подготавливается, возможно, более широкий набор обучающих примеров, организованных в виде наборов входных данных, ассоциированных с известными выходными значениями. Входные значения для обучения могут быть неполны и частично противоречивы.
4. Входные данные по очереди предъявляются ИНС, а полученное выходное значение сравнивается с эталоном. Затем производится подстройка весовых коэффициентов межнейронных соединений для минимизации ошибки между реальным и желаемым выходом сети.
5. Обучение повторяется до тех пор, пока суммарная ошибка во всем множестве входных значений не достигнет приемлемого уровня, либо ИНС не придет в стационарное состояние. Рассмотренный метод обучения нейроподобной сети носит название «обратное распространение ошибки» (error backpropagation) и относится к числу классических алгоритмов нейроматематики.

Настроенная и обученная ИНС может использоваться на реальных входных данных, не только подсказывая пользователю корректное решение, но и оценивая степень его достоверности.

2.2.5 Алгоритм секущих плоскостей.

Существуют различные алгоритмы, позволяющие распознавать образы. Алгоритм обучения машины “узнаванию” образов, основанный на методе секущих гиперплоскостей (рис. 2.6), заключается в аппроксимации разделяющей гиперповерхности “кусками” гиперплоскостей и состоит из следующих основных этапов:

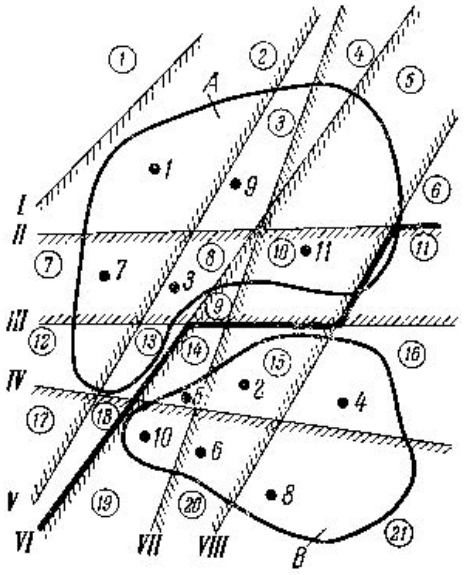


Рис. 2.6 Метод секущих плоскостей.

А. Обучение

(формирование разделяющей поверхности):

- Проведение секущих плоскостей;
- Исключение лишних плоскостей;
- Исключение лишних кусков плоскостей.

Б. Распознавание новых объектов.

При использовании метода параллельных вариантов одновременно и независимо друг от друга на одном и том же материале обучаются несколько машин. При опознавании новых объектов каждая машина будет относить эти объекты к какому-то образу, может быть, не к одному и тому же. Окончательное решение принимается “голосованием” машин – объект относиться к тому образу, к которому его отнесло большее число машин.

Способ повышения надежности распознавания состоит в некотором улучшении метода проведения секущих плоскостей. Можно предположить, что если проводить секущие плоскости близко к плоскости, проходящей через середину прямой, соединяющей объект и оппонент, перпендикулярный этой прямой, то результирующая поверхность будет ближе к истинной границе между образами. Эксперименты подтверждают это предположение.

2.2.6 Алгоритмы, основанные на методе потенциалов.

В алгоритме, основанном на методе потенциалов, с каждым возбужденным элементом поля рецепторов можно связать некоторую функцию, равную единице на этом элементе и убывающую по всем направлениям от него, т.е. функцию φ, аналогичную электрическому потенциалу с той лишь разницей, что в данном случае R есть расстояние между двумя соседними элементами поля рецепторов [56].

$$\varphi(R) = 1 \Big/ (1 + \alpha R^2) \qquad (2.1)$$

Для подсчета пользуются следующим простым правилом: каждый возбужденный элемент поля рецепторов имеет “собственный” потенциал, равный единице, и который в свою очередь увеличивает на ½ потенциалы всех (в том числе и возбужденных) соседних с ним элементов по горизонтали, вертикали и диагоналям. Однако этот метод кодирования может быть улучшен. Если связать с каждым возбужденным элементом поля рецепторов некоторую функцию, равную единице на этом элементе и убывающую по всем направлениям от него, т.е. функцию, аналогичную потенциалу φ, с той лишь разницей, что в данном случае R есть расстояние между двумя соседними элементами поля рецепторов (рис. 2.7). Эта функция, может быть, аппроксимирована ступенчатой функцией, постоянной в пределах одного рецептора и скачкообразно изменяющейся на границах рецепторов.

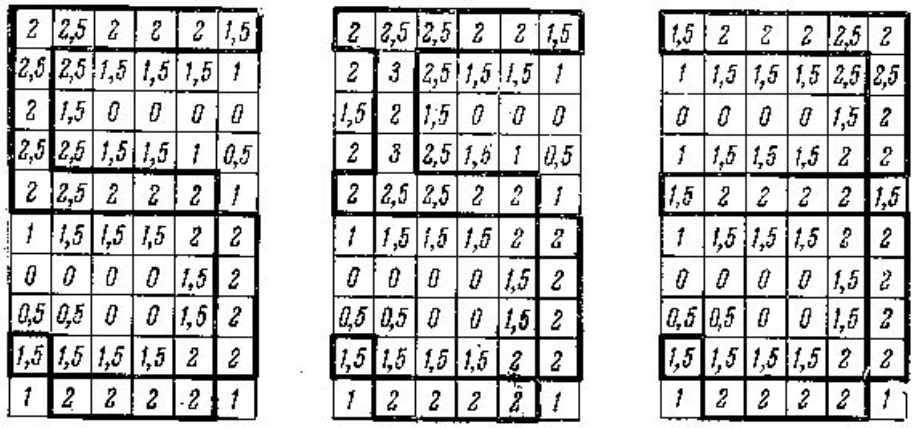


Рис. 2.7 Метод потенциалов.

Простейший алгоритм узнавания, построенный на методе потенциалов, можно осуществить в два этапа:

1. Обучение (в процессе обучения запоминаются коды всех появившихся точек и указания, к какому из образов относится каждая точки).
2. Узнавание (в процессе узнавания производится идентификация и выдается информация, к какому образу принадлежит закодированная матрица).

2.2.7 Пример алгоритма обучения Персептрона с нерекуррентной сетью.

Несмотря на некоторые ограничения исходной формы модели Персептрона Розенблатта, она стала основой для многих современных наиболее сложных алгоритмов обучения с учителем [57]. Примером Персептрона с нерекуррентной сетью может служить вид, показанный на рис. 2.8. Она использует алгоритм обучения с учителем; другими словами, обучающая выборка состоит из множества входных векторов, для каждого из

которых указан свой требуемый вектор цели. Компоненты входного вектора представлены непрерывным диапазоном значений; компоненты вектора цели являются двоичными величинами (0 или 1). После обучения сеть получает на входе набор непрерывных входов и вырабатывает требуемый выход в виде вектора с бинарными компонентами [69].

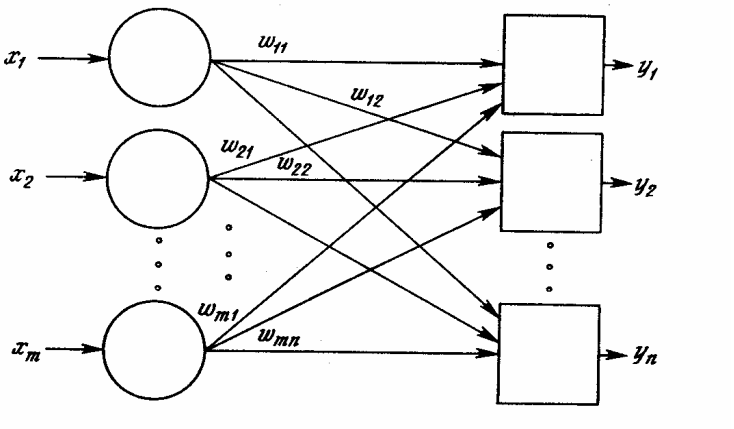


Рис. 2.8 Однослойная нейронная сеть.

Обучение осуществляется следующим образом:

1. Рандомизируются все веса сети в малые величины.
2. На вход сети подается входной обучающий вектор **X** и вычисляется сигнал NET от каждого нейрона, используя стандартное выражение

$$\mathrm{NET}_j = \sum_i x_i w_{ij} \ . \qquad (2.2)$$

3. Вычисляется значение пороговой функции активации для сигнала NET от каждого нейрона следующим образом:

$\mathrm{OUT}_j = 1$, если NET_j больше чем порог θ_j,

$\mathrm{OUT}_j = 0$ в противном случае.

Здесь θ_j представляет собой порог, соответствующий нейрону j (в простейшем случае, все нейроны имеют один и тот же порог).

4. Вычисляется ошибка для каждого нейрона посредством вычитания полученного выхода из требуемого выхода:

$$\mathrm{error}_j = \mathrm{target}_j - \mathrm{OUT}_j. \qquad (2.3)$$

5. Каждый вес модифицируется следующим образом:

$$W_{ij}(t+1) = w_{ij}(t) + \alpha\, x_i\, \mathrm{error}_j. \qquad (2.4)$$

6. Повторяются шаги со второго по пятый до тех пор, пока ошибка не станет достаточно малой.

2.2.8 Метод обучения Уидроу-Хоффа.

Персептрон Розенблатта ограничивается бинарными выходами. Уидроу вместе с Хоффом расширили алгоритм обучения персептрона на случай непрерывных выходов, используя сигмоидальную функцию. Они разработали математическое доказательство того, что сеть при определенных условиях будет сходиться к любой функции, которую она может представить. Их первая модель – Адалин имеет один выходной нейрон, более поздняя модель – Мадалин расширяет ее на случай со многими выходными нейронами.

Выражения, описывающие процесс обучения Адалина, очень схожи с персептронными. Существенные отличия имеются в четвертом шаге, где используются непрерывные сигналы NET вместо бинарных OUT. Модифицированный шаг 4 в этом случае реализуется следующим образом [51]:

4. Вычисляется ошибка для каждого нейрона посредством вычитания полученного выхода из требуемого выхода:

$$error_j = target_j - NET_j. \qquad (2.5)$$

2.2.9 Нейро-сетевая самоорганизация.

Результаты исследований Кохонена на самоорганизующихся структурах, для задач распознавания образов классифицируют образы, представленные векторными величинами, в которых каждый компонент вектора соответствует элементу образа. Алгоритмы Кохонена основываются на технике обучения без учителя. После обучения подача входного вектора из данного класса будет приводить к выработке возбуждающего уровня в каждом выходном нейроне; нейрон с максимальным возбуждением представляет классификацию. Так как обучение проводится без указания целевого вектора, то нет возможности определять заранее, какой нейрон будет соответствовать данному классу входных векторов. Тем не менее, это планирование легко проводится путем тестирования сети после обучения.

Алгоритм трактует набор из n входных весов нейрона как вектор в n-мерном пространстве. Перед обучением каждый компонент этого вектора весов инициализируется в случайную величину. Затем каждый вектор нормализуется в вектор с единичной длиной в пространстве весов. Это делается делением каждого случайного веса на квадратный корень из суммы квадратов компонент этого весового вектора.

Все входные вектора обучающего набора также нормализуются, и сеть обучается согласно следующему алгоритму [62]:

1. Вектор **X** подается на вход сети.
2. Определяются расстояния D_j (в n-мерном пространстве) между **X** и весовыми векторами $\mathbf{W}_j$ каждого нейрона. В евклидовом пространстве это расстояние вычисляется по следующей формуле

$$\mathbf{D}_j = \sqrt{\sum_i (x_i - w_i)_j^2}\,, \qquad (2.6)$$

где x_i – компонента i входного вектора **X**, w_{ij} – вес входа i нейрона j.

3. Нейрон, который имеет весовой вектор, самый близкий к **X**, объявляется победителем. Этот весовой вектор, называемый $\mathbf{W}_c$, становится основным в группе весовых векторов, которые лежат в пределах расстояния D от $\mathbf{W}_c$.
4. Группа весовых векторов настраивается в соответствии со следующим выражением:

$$\mathbf{W}_j(t+1) = \mathbf{W}_j(t) + \alpha[\mathbf{X} - \mathbf{W}_j(t)] \qquad (2.7)$$

для всех весовых векторов в пределах расстояния D от $\mathbf{W}_c$

5. Повторяются шаги с 1 по 4 для каждого входного вектора.

В процессе обучения нейронной сети значения D и α постепенно уменьшаются. Коэффициент α в начале обучения луче устанавливать приблизительно равным 1 и уменьшался в процессе обучения до 0, в то время как D может в начале обучения равняться максимальному расстоянию между весовыми векторами и в конце обучения стать настолько маленьким, что будет обучаться только один нейрон.

В соответствии с существующей точкой зрения, точность классификации будет улучшаться при дополнительном обучении. Согласно рекомендации Кохонена, для получения хорошей статистической точности количество обучающих циклов должно быть, по крайней мере, в 500 раз больше количества выходных нейронов.

Обучающий алгоритм настраивает весовые векторы в окрестности возбужденного нейрона таким образом, чтобы они были более похожими на входной вектор. Так как все векторы нормализуются в векторы с единичной длиной, они могут рассматриваться как точки на поверхности единичной гиперсферы. В процессе обучения группа соседних весовых точек перемещается ближе к точке входного вектора. Предполагается, что входные векторы фактически группируются в классы в соответствии с их положением в векторном пространстве [59]. Определенный класс будет ассоциироваться с определенным нейроном, перемещая его весовой вектор в направлении центра класса и способствуя его возбуждению при появлении на входе любого вектора данного класса. После обучения классификация выполняется посредством подачи на вход сети испытуемого вектора, вычисления

возбуждения для каждого нейрона с последующим выбором нейрона с наивысшим возбуждением как индикатора правильной классификации.

2.3 Персептрон как модель распознавания.

2.3.1 Устройство персептрона.

Крупный толчок развитию нейрокибернетики дал нейрофизиолог Френк Розенблатт, предложивший модель узнающей машины, названную им “Персептрон” (от латинского percepto - понимаю, познаю). При ее разработке он исходил из некоторых принятых представлений о структуре мозга и зрительного аппарата. Стремясь воспроизвести функции человеческого мозга, он использовал простую модель биологического нейрона (рис. 2.9) и систему связей между ними.

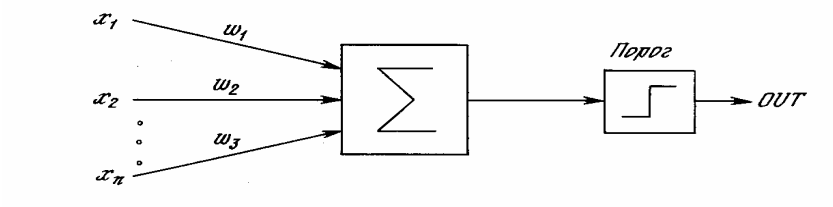


Рис. 2.9 Персептронный нейрон.

Воспринимающим устройством персептрона служит фотоэлектрическая модель сетчатки – поле рецепторов, состоящее из нескольких сотен фотосопротивлений (S-элементов) (см. рис. 2.10). Каждый элемент поля рецепторов может находиться в двух состояниях – возбужденном или невозбужденном состоянии, в зависимости от того, падает или нет на соответствующее фотосопротивление контур проектируемой на поле фигуры. На выходе каждого элемента появляется сигнал x_i (i = 1, 2, …, n, где n- число элементов), равный единице, если элемент возбужден, и нулю – в противном случае. Следующей ступенью персептрона служат, так называемые, ассоциативные элементы или А-элементы. Каждый А-элемент имеет несколько входов и один выход. При подготовке персептрона к эксперименту к входам А-элемента подключаются выходы рецепторов, причем подключение любого из них может быть произведено со знаком плюс или со знаком минус.

Выбор рецепторов, подключаемых к данному А-элементу, также как и выбор знака подключения, производится случайно. В ходе эксперимента связь рецепторов с А-элементами остается неизменной. А-элементы производят алгебраическое суммирование сигналов [33], поступивших на их входы, и полученную сумму сравнивают с одинаковой для всех А-элементов величиной ϑ.

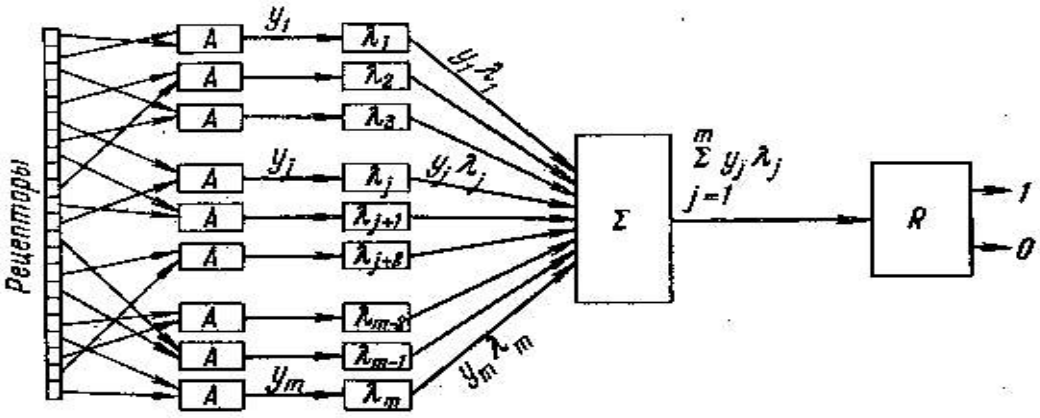


Рис 2.10 Персептронное устройство с одним выходом.

Если сумма больше ϑ, А-элемент возбуждается и выдает на выходе сигнал, равный единице. Если сумма меньше ϑ, А-элемент остается невозбужденным и выходной его сигнал равен нулю [50]. Таким образом, выходной сигнал j-го А-элемента:

$$y_i = \begin{cases} 1, & \text{если } \left(\sum_{i=1}^{n} r_{ij} * x_i - \vartheta\right) \geq 0, \\ 0, & \text{если } \left(\sum_{i=1}^{n} r_{ij} * x_i - \vartheta\right) < 0, \end{cases} \qquad (2.8)$$

где величина r_{ij} принимает значение +1, если i -й рецептор подключен ко входу j-го А-элемента со знаком плюс; и значение -1, если рецептор подключен со знаком минус, и значение 0, если i-ый рецептор к j-му А-элементу не подключается (j = 1, 2, …, m, где m – число А-элементов). Выходные сигналы А-элементов с помощью специальных устройств (усилителей) умножаются на переменные коэффициенты λ_j. Каждый из этих коэффициентов может быть положительным, отрицательным или равным нулю и меняться независимо от других коэффициентов. Выходные сигналы усилителей суммируются, и суммарный сигнал поступает на вход,

$$\sigma = \sum_{j=1}^{m} \lambda_j * y_j \qquad (2.9)$$

так называемого, реагирующего элемента или R-элемента. Если σ положительна или равна нулю, R-элемент выдает на выходе единицу, а если σ отрицательна – нуль.

Таким образом, выходной сигнал R-элемента (являющийся также выходным сигналом персептрона) будет иметь следующий вид:

$$R = \begin{cases} 1, & \text{если } \sum_{j=1}^{m} \lambda_j * y_j \geq 0, \\ 0, & \text{если } \sum_{j=1}^{m} \lambda_j * y_j < 0, \end{cases} \qquad (2.10)$$

Предположим, что на поле рецепторов проектируются фигуры, принадлежащие к двум различным образам. Если окажется возможным привести персептрон в такое состояние, чтобы он с достаточной надежностью выдавал на выходе 1, при появлении на его входе фигур одного образа, то это будет означать, что персептрон обладает способностью обучаться различению двух образов [48].

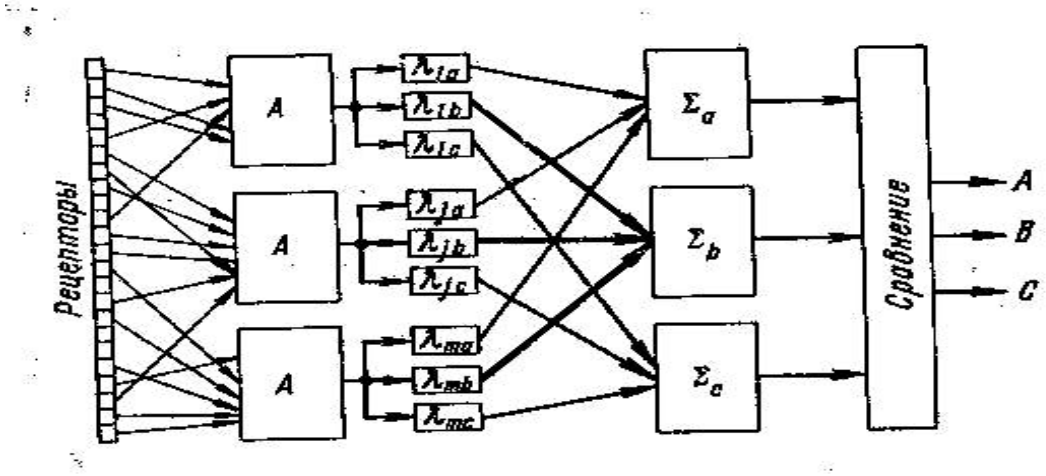


Рис 2.11 Персептронное устройство с несколькими выходами.

Описанная структура персептрона позволяет разделять предъявляемые объекты только на два множества. Для распознавания большего числа образов, например, трех A, B и C может быть применен персептрон, построенный по схеме (рис. 2.11). Выходной сигнал каждого A-элемента поступает не на один, а на несколько (по числу различаемых образов) усилителей. После умножения на λ выходные сигналы поступают на сумматоры Σ, количество которых также равно числу различаемых образов. Вместо R-элемента установлено устройство, сравнивающее между собой выходные сигналы сумматоров. Предъявленный объект относится к тому образу, сумматор которого имеет наибольший сигнал.

Для распознавания нескольких образов может быть использован персептрон несколько иной структуры. В таком персептроне А-элементы разбиты на несколько групп, каждая из которых связана со своим сумматором и R-элементом. Совокупность выходных сигналов R-элементов можно рассматривать как выраженный в двоичном коде номер образа, что и дает такому персептрону возможность разбивать объекты на несколько классов. Например, для классификации на восемь классов достаточно трех групп. В этом случае возможны следующие восемь комбинаций выходных сигналов трех R-элементов: 000, 001, 010, 011, 100, 101, 110, 111. Появление каждой из этих комбинаций можно рассматривать как отнесение персептроном предъявленной ему фигуры к одному из восьми образов.

Каждая из групп А-элементов, соединенных со своим R-элементом, по структуре и действию вполне аналогична персептрону, способному разбивать объекты на два класса. Обучение персептрона состоит из ряда последовательных тактов. В каждом такте персептрону предъявляется объект одного из образов. В зависимости от реакции персептрона на предъявленную ему фигуру производится по определенным правилам изменение коэффициентов λ_j. Оказывается возможным за некоторое конечное количество тактов привести персептрон в такое состояние, что он с достаточной уверенностью распознает предъявляемые ему фигуры.

Возможны два типа алгоритмов обучения персептрона. Первый из них не учитывает правильности ответов персептрона в процессе обучения, и изменение λ_j в каждом такте производится независимо от того, «узнал» или не «узнал» персептрон предъявленную в этом такте фигуру. В алгоритмах второго типа коэффициенты λ_j изменяются с учетом правильности ответов персептрона.

Алгоритм первого типа осуществляется следующим образом. Заранее уславливаются, что после обучения персептрон должен выдавать на выходе 1 при предъявлении ему, например, объектов образа А, и нуль – при предъявлении образа – В. Затем предъявляют персептрону объекты каждого из образов. В каждом такте персептрон отвечает на предъявленный ему объект возбуждением некоторых А-элементов. Обучение состоит в том, что коэффициенты λ_j возбужденных в данном такте А-элементов увеличиваются на некоторую величину (например на единицу), если в этом такте был предъявлен объект образа А, и уменьшается на эту же величину, если был предъявлен объект образа В. Естественно, что такое изменение коэффициентов λ_j должно приводить к повышению правильности ответов персептрона, так как увеличение λ_j возбужденных А-элементов приводит к увеличению сигнала на входе R-элемента, а их уменьшение – к уменьшению сигнала. В соответствии с принятым условием персептрон будет давать правильные ответы,

если образу А будут соответствовать положительные, а образу В – отрицательные сигналы на входе R-элемента.

2.3.2 Функции А-элементов.

Выходной сигнал каждого А-элемента зависит от знака выражения

$$\sum_{i=1}^{n} r_{ij} * x_i - \vartheta, \qquad (2.11)$$

где x_i – выходной сигнал i–го рецептора. Это выражение для некоторых комбинаций x_i может принимать положительные (или равные нулю), а для других – отрицательные значения. Каждая комбинация x_i соответствует определенной фигуре, спроектированной на поле рецепторов.

Таким образом, каждый А-элемент разбивает все фигуры, которые могут быть спроектированы на поле рецепторов, на два класса. Для фигур одного класса выход А-элемента положителен или равен нулю, для фигур второго класса – отрицателен. Этому факту можно дать геометрическую интерпретацию. Действительно, каждую комбинацию x_i можно рассматривать как код предъявленной персептрону фигуры или как координаты некоторой точки в пространстве рецепторов. Выражение (2.11) можно рассматривать как левую часть уравнения некоторой плоскости :

$$\sum_{i=1}^{n} r_{ij} * x_i - \vartheta = 0. \qquad (2.12)$$

Знак сигнала на выходе А-элемента говорит о том, по какую сторону от этой плоскости лежит точка, соответствующая предъявленной персептрону фигуре.

Итак, каждый элемент, после того как для него установлены коэффициенты r_{ij} (т. е. произведено подключение рецепторов на его входы), определяет некую плоскость в пространстве рецепторов. Все А-элементы персептрона осуществляют разбиение пространства рецепторов с помощью m плоскостей на некоторое количество многогранников (число их весьма велико, так как количество плоскостей измеряется сотнями) [79]. Разбиение осуществляется вполне случайно, так как коэффициенты r_{ij} выбираются по жребию.

Любому изображению, проектируемому на поле рецепторов, в данном эксперименте соответствует определенное состояние каждого из А-элементов персептрона. Из выходных сигналов А-элементов, т. е. из последовательности величин y_i, может быть образован m–разрядный двоичный код, причем в j-м разряде кода будет стоять единица, если j-й А-

элемент возбужден, и нуль в противном случае. Этот же самый код, описывающий состояние А-элементов, характеризует также положение многогранника, в который попала точка, соответствующая данному изображению. Единица в j-м разряде кода означает, что многогранник лежит по одну сторону j-й плоскости; нуль соответственно указывает, что многогранник лежит по другую сторону этой плоскости.

На вход R-элемента попадает сумма произведений разрядов этого кода на коэффициенты λ_j:

$$\sum_{j=1}^{m} \lambda_j * y_j. \qquad (2.13)$$

В данном такте работы персептрона, т.е. при данной комбинации коэффициентов λ_j, некоторым кодам будет соответствовать единица, а остальным кодам – нуль на выходе R-элемента. А это означает, что часть многогранников персептрон относит к первому классу, а другую часть – ко второму. Иными словами, персептрон формирует из кусков случайно проведенных секущих плоскостей границу между двумя частями пространства рецепторов. Результирующее положение граничной поверхности определяется комбинацией λ_j в данном такте. Цель обучения, очевидно, должна заключаться в наилучшем приближении этой поверхности в действительной границе между двумя образами (т.е. между соответствующими этим образам множествами точек в пространстве рецепторов).

2.3.3 Персептрон как модель мозга.

При разработке персептрона Ф. Розенблатт пытался моделировать некоторые свойства живого мозга [50]. Персептрон или любая программа, имитирующая процесс распознавания, работают в двух режимах: в режиме обучения и в режиме распознавания. Прежде всего, в отличие от рассмотренных ранее алгоритмов, алгоритм персептрона в ходе обучения не требует запоминания предъявленных объектов, а при распознавании – перебора всех “известных” ему фигур. В этом смысле работа персептрона имеет определенное сходство с работой мозга, который формирует представление об образе, не запоминая отдельных его объектов, и узнает новые объекты без сравнения их с каждым из ранее встречающихся. Далее, структура персептрона имеет некоторые общие черты со структурой высшей нервной системы. В частности, рецепторы персептрона являются достаточно близкой аналогией рецепторов зрительного аппарата, а А-элементы имеют определенное сходство с нейронами. Известно, что нейроны обладают свойством возбуждаться, если интенсивность сигнала, получаемого нейроном от связанных с ним рецепторов (или других нейронов), превосходит некоторую пороговую величину. Свойство персептрона допускать

случайный характер связей “рецептор – А-элемент”, аналогично некоторым свойствам структуры головного мозга. Возможно, что связи между нейронами мозга в большинстве случаев также имеют случайный характер, т.е. случайно варьируются у разных животных одного биологического вида. Если предположить обратное, т.е. допустить, что все связи между нейронами мозга точно фиксированы и одинаковы у всех животных данного вида, и что изменение этих связей может привести к резкому нарушению работы мозга, то придется допустить также, что сведения обо всех этих связях должны передаваться по наследству. А так как количество нейронов мозга исчисляется миллиардами, то такое предположение приводит к фантастически большому объему генетической информации. Вместе с тем на примере персептрона видно, сколь биологически естественным является понятие компактного множества, ибо обучение распознаванию таких множеств оказывается возможным при случайных связях между рецепторами и нейронами.

Известно, что мозг способен сохранять или восстанавливать многие свои функции при серьезных повреждениях, вызванных травмами или заболеваниями. Устойчивость персептрона к нарушениям его структуры имеет определенное сходство с этим свойством мозга. Из всех этих соображений нельзя сделать вывод, что алгоритмы мозга и персептрона совпадают. Однако в настоящее время персептрон является, по-видимому, наиболее правдоподобной моделью мозга.

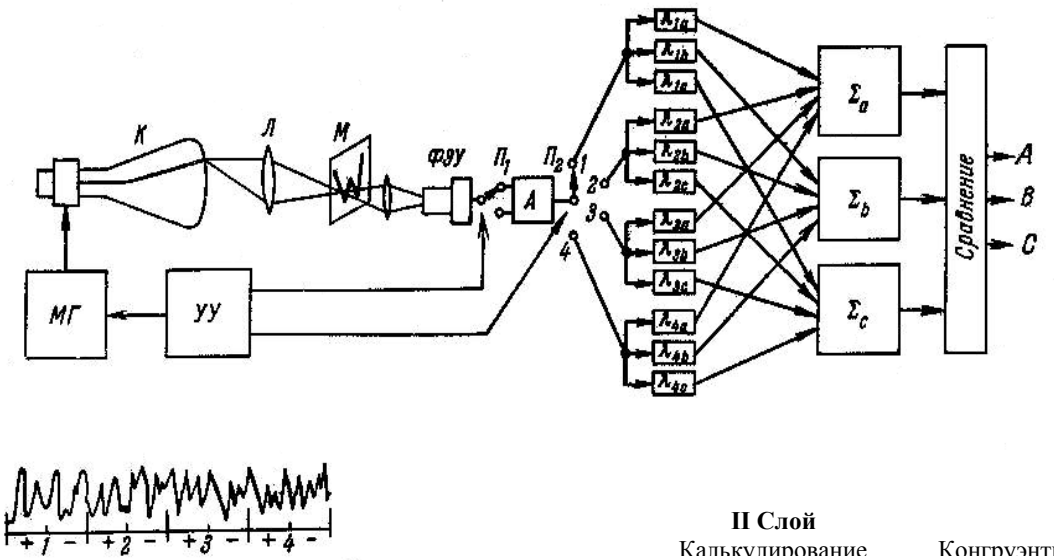
2.3.4 Узнающая машина Гамба.

Рассмотрим работу персептрона еще с одной стороны. Состояние j-ого А-элемента зависит от знака выражения (2.11). В этом выражении величина x_i соответствует состоянию i-го рецептора и равна единице, если он возбужден. Величина r_{ij} для части рецепторов (не подключенных к j-му А-элементу) равна нулю, а для остальных принимает значение +1 или -1 в зависимости от того, с каким знаком подключен данный рецептор к входу А-элемента. Все рецепторы, соединенные с данным А-элементом, можно, таким образом, разбить на две группы в зависимости от знака их подключения. Рецепторы каждой группы образуют на поле рецепторов некую мозаику. При предъявлении персептрону, какой – либо фигуры возбуждаются те рецепторы каждой из мозаик, на которые падает контур фигуры. Соответствующие этим рецепторам x_i становиться равным единице (остальные x_i соответствуют невозбужденным рецепторам и равны нулю). Обозначим число возбужденных рецепторов первой “положительной” мозаики через K_j, а число возбужденных рецепторов второй “отрицательной” мозаики – через Q_j. Тогда выражение, описывающее работу j-го А-элемента, примет вид:

$$K_j - Q_j - \vartheta. \qquad (2.14)$$

А-элемент возбуждается, если $(K_j - Q_j) \geq \vartheta$. Состояние А-элемента определяется числом рецепторов, возбуждаемых в каждой из этих пятен предъявленной персептрону фигурой.

Машина Гамба (рис. 2.12), по принципу действия вполне аналогична персептрону. Разница состоит в способе реализации А-элементов. В отличие от персептрона, все А-элементы которого действуют одновременно, в машине Гамба А-элементы воспроизводятся последовательно, один за другим. Каждый такт работы машины, соответствующий распознаванию одной фигуры, распадается, таким образом, на ряд “малых” тактов, в каждом из которых воспроизводится один А-элемент [66]. Действует машина таким образом: для реализации “мозаики” используется магнитофон, на ленте которого записана случайная функция (“шум”). Записанная на ленте функция повторяется при распознавании каждой фигуры и в виде электрического напряжения подается на управляющие пластины кинескопа К. Эта функция разделена во времени на некоторое количество интервалов, каждый из которых соответствует “малому” такту, т.е. одному А-элементу (на рис. 2.12 машины Гамба сигнал, условно изображенный в виде кривой, разбит на четыре такта). Под воздействием этого сигнала светящееся пятно на экране кинескопа выписывает случайные кривые.



Изображенные пятна проектируется с помощью оптической системы Л на предъявленный машине объект [80], который представляет собой прозрачную фигуру на не прозрачной маске М. Количество света, прошедшее через маску за некоторое время, оказывается пропорциональным числу пересечений контура фигуры с тем силуэтом, который прочертил за это время на маске изображение светящегося пятна. Если измерить

количество света k_j, прошедшее сквозь маску за первую половину j-го малого такта, а также количество света q_j, прошедшее сквозь маску за вторую половину этого такта, и вычесть одно количество из другого, то полученная разность $k_j - q_j$ будет эквивалентна разности $К_j$ - Q_j. Для выполнения этих операций служат фотоэлектрический умножитель ФЭУ [52] и накопитель А, связанный с выходом ФЭУ через переключатель П1. Переключатель управляется переключающим устройством УУ таким образом, что в первые половины всех малых тактов выходной сигнал ФЭУ поступает на "положительный" вход накопителя, а во вторые половины – на "отрицательный." Накопитель в каждом малом такте образует разность $к_j - q_j$ и сравнивает ее с пороговой величиной ϑ. Если $к_j - q_j \geq \vartheta$, выходной сигнал накопителя равен единице, в противном случае – нулю. Таким образом, в каждом малом такте система "магнитофон – кинескоп – фото-умножитель – накопитель " реализует А-элемент. Остальная часть схемы вполне аналогична соответствующей части персептрона. Разница состоит лишь в том, что умножение выходных сигналов накопителя на коэффициенты λ и суммирование полученных произведений сумматорами $\sum$ производиться последовательно во времени. С этой целью выход накопителя А подключается к элементам через переключатель П2, положение которого всегда совпадает с номером текущего "малого" такта. Сравнение выходных сигналов сумматора и отнесение предъявленной фигуры к одному из образов происходит после завершения всей последовательности малых тактов, т. е. изменение коэффициента λ_j, может производиться по любому из алгоритмов, пригодных для персептрона.

К недостаткам машины Гамба можно отнести последовательное воспроизведение А-элементов, которое является относительно длительным для "рассматривания" каждой фигуры. Однако быстродействие системы "кинескоп – фото-умножитель – накопитель" может быть реализовано настолько высоким, что время экспозиции каждого объекта будет достаточно мало.

2.3.5 Многослойные персептроны и их представляемость.

О персептронах было сформулировано и доказано несколько основополагающих теорем, две из которых, определяющие основные свойства персептрона [36].

Теорема 1. Класс элементарных α-персептронов, для которых существует решение для любой задуманной классификации, не является пустым.

Эта теорема утверждает, что для любой классификации, обучающей последовательности, можно подобрать такой набор (из бесконечного набора) А-элементов, в

котором будет осуществлено задуманное разделение обучающей последовательности при помощи линейного решающего правила.

Теорема 2. Если для некоторой классификации C(W) решение существует, то в процессе обучения α-персептрона с коррекцией ошибок, начинающегося с произвольного исходного состояния, это решение будет достигнуто в течение конечного промежутка времени.

Смысл этой теоремы состоит в том, что если относительно задуманной классификации можно найти набор A-элементов, в котором существует решение, то в рамках этого набора оно будет достигнуто в конечный промежуток времени.

Предложены различные правила обучения персептрона. Один из алгоритмов называется процедурой сходимости персептрона Розенблатта и является вариантом хеббовского правила изменения весов связей с учителем. Таким образом, если два класса образов могут быть разделены гиперплоскостью, то при достаточно долгом обучении персептрон будет различать их правильно. Однако линейная разделяющая поверхность, упрощающая анализ персептрона, ограничивает решаемый им круг задач [25]. Этот вопрос тщательно исследовали Минский и Пейперт, показав, какие задачи в принципе не может решить персептрон с одним слоем обучаемых связей. Одним из таких примеров является выполнение логической операции «исключающее ИЛИ». Персептрон с одним слоем обучаемых связей формирует границы областей решений в виде гиперплоскостей. Двухслойный персептрон может выполнять операцию логического «И» над полупространствами, образованными гиперплоскостями первого слоя весов. Это позволяет формировать любые, возможно неограниченные, выпуклые области в пространстве входных сигналов [78]. С помощью трехслойного персептрона, комбинируя логическими «ИЛИ» нужные выпуклые области, можно получить уже области решений произвольной формы и сложности, в том числе невыпуклые и несвязные. То, что многослойные персептроны с достаточным множеством внутренних нейроподобных элементов и соответствующей матрицей связей в принципе способны осуществлять любое отображение вход–выход, отмечали Минский и Пейперт, однако они сомневались в том, что можно открыть для них мощный аналог процедуры обучения простого персептрона. Алгоритм обратного распространения ошибки “Error Back Propagation” является обобщением одной из процедур обучения простого персептрона, известной, как правило, Уидроу-Хоффа (или дельта-правило), и требует представления обучающей выборки. Выборка состоит из набора пар образов, между которыми надо установить соответствие, и может рассматриваться как обширное задание векторной функции, область определения которой – набор входных образов, а множество значений – набор выходов.

Обобщенной моделью сети может служить многослойная нейроподобная сеть с прямыми связями (рис. 2.13). В настоящее время многослойные персептроны являются наиболее популярной моделью нейронных сетей. Это в значительной степени объясняется тем, что с их помощью удалось продемонстрировать решение ряда задач, в том числе классической для персептронов задачи «исключающего ИЛИ», задачи синтеза речи по тексту [17], а также задач, требующих принятия экспертных решений. Методы обучения многослойных сетей дали толчок для дальнейших исследований в этой области.

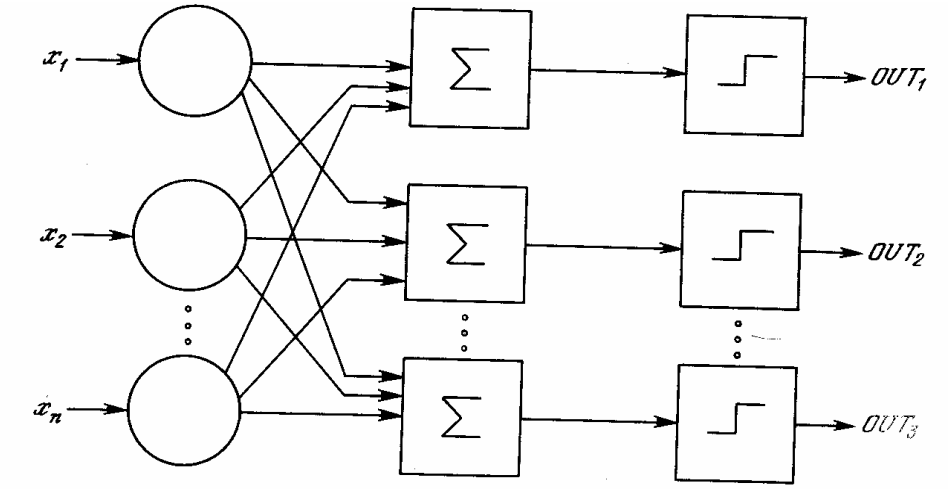


Рис. 2.13 Персептрон со многими выходами.

Теория персептронов является основой для многих других типов искусственных нейронных сетей, и персептроны иллюстрируют важные принципы. В силу этих причин они являются логической исходной точкой для изучения искусственных нейронных сетей.

Доказательство теоремы обучения персептрона показало, что персептрон способен научиться всему, что он способен представлять. Важно при этом уметь различать представляемость и обучаемость. Понятие представляемости относится к способности персептрона или сети моделировать определенную функцию. Обучаемость же требует наличия систематической процедуры настройки весов сети для реализации этой функции.

2.3.6 Проблема функции исключающее ИЛИ для персептронов.

Один из самых пессимистических результатов Минского показывает, что однослойный персептрон не может воспроизвести такую простую функцию, как исключающее ИЛИ. Это функция от двух аргументов, каждый из которых может быть нулем или единицей. Она принимает значение единицы, когда один из аргументов равен единице (но не оба).

Серьезное ограничение представляемости однослойными сетями можно преодолеть, добавив дополнительные слои. Например, двухслойные сети можно получить каскадным соединением двух однослойных сетей [77]. Они способны выполнять более общие классификации, отделяя те точки, которые содержатся в выпуклых ограниченных или неограниченных областях. Область называется выпуклой, если для любых двух ее точек соединяющий их отрезок целиком лежит в области. Область называется ограниченной, если ее можно заключить в некоторый круг. Неограниченную область невозможно заключить внутрь круга (например, область между двумя параллельными линиями). Примеры выпуклых ограниченных и неограниченных областей представлены на рис. 2.14.

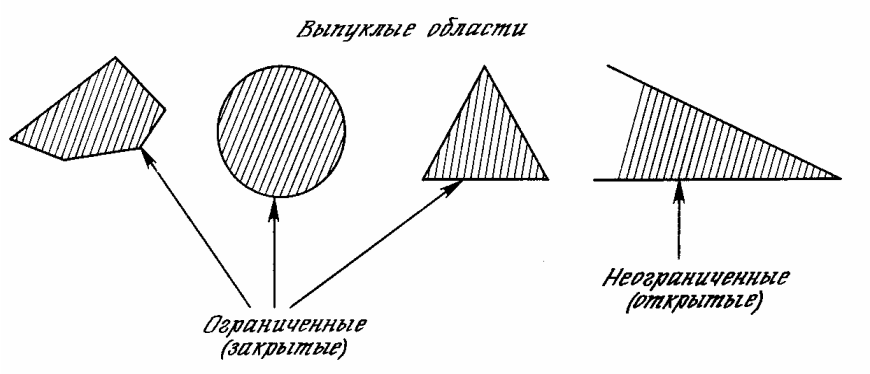


Рис. 2.14 Выпуклые, ограниченные и неограниченные области.

Для уточнения требований выпуклости, рассмотрим простую двухслойную сеть с двумя входами, подведенными к двум нейронам первого слоя, соединенными с единственным нейроном в слое 2 (см. рис. 2.15). Пусть порог выходного нейрона равен 0,75, а оба его веса равны 0,5. В этом случае для того, чтобы порог был превышен, и на выходе появилась единица, требуется, чтобы оба нейрона первого уровня на выходе имели единицу. Таким образом, выходной нейрон реализует логическую функцию И [47]. На рис. 2.15 каждый нейрон слоя 1 разбивает плоскость x-y на две полуплоскости, один обеспечивает единичный выход для входов ниже верхней линии, другой – для входов выше нижней линии. На рис. 2.15 показан результат такого двойного разбиения, где выходной сигнал нейрона второго слоя равен единице только внутри V-образной области. Аналогично во втором слое может быть использовано три нейрона с дальнейшим разбиением плоскости и созданием области треугольной формы. Включением достаточного числа нейронов во входной слой может быть образован выпуклый многоугольник любой желаемой формы. Так как они образованы с помощью операции «И» над областями, задаваемыми линиями, то все такие многогранники выпуклы, следовательно, только выпуклые области и возникают. Точки, не

составляющие выпуклой области, не могут быть отделены от других точек плоскости двухслойной сетью.

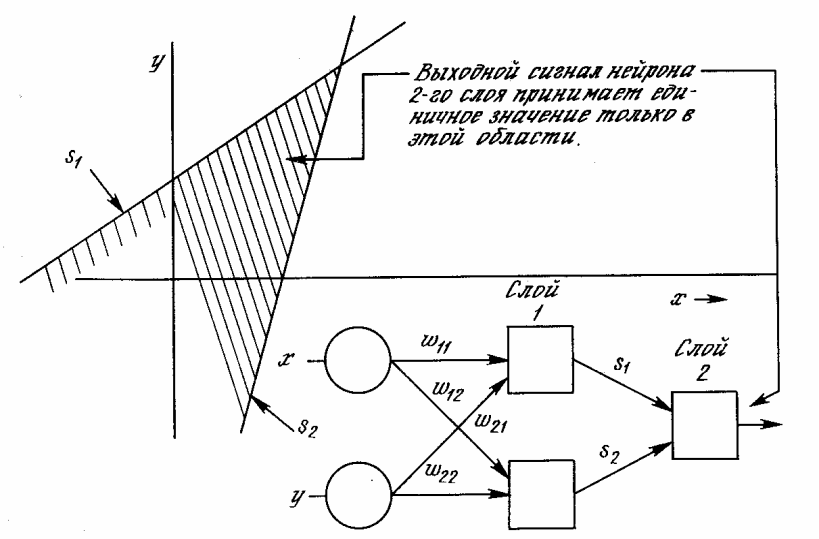


Рис. 2.15 Выпуклая область решений, задаваемая двухслойной сетью.

Нейрон второго слоя не ограничен функцией «И». Он может реализовывать многие другие функции при подходящем выборе весов и порога. Например, можно сделать так, чтобы единичный выход любого из нейронов первого слоя приводил к появлению единицы на выходе нейрона второго слоя, реализовав тем самым логическое «ИЛИ». Имеется 16 двоичных функций от двух переменных. Если выбирать подходящим образом веса и порог, то можно воспроизвести 14 из них (все, кроме «ИСКЛЮЧАЮЩЕЕ ИЛИ» и «ИСКЛЮЧАЮЩЕЕ НЕТ») [37].

Входы не обязательно должны быть двоичными. Вектор непрерывных входов может представлять собой произвольную точку на плоскости x-y. В этом случае мы имеем дело, со способностью сети разбивать плоскость на непрерывные области, а не с разделением дискретных множеств точек. Для всех этих функций, однако, линейная разделимость показывает, что выход нейрона второго слоя равен единице только в части плоскости x-y, ограниченной многоугольной областью. Поэтому для разделения плоскостей P и Q необходимо, чтобы все P лежали внутри выпуклой многоугольной области, не содержащей точек Q (или наоборот).

Трехслойная сеть, однако, является более общей. Ее классифицирующие возможности ограничены лишь числом искусственных нейронов и весов. Ограничения на выпуклость отсутствуют. Теперь нейрон третьего слоя принимает в качестве входа набор выпуклых многоугольников, и их логическая комбинация может быть невыпуклой. На рис. 2.16 иллюстрируется случай, когда два треугольника A и B, скомбинированные с помощью

функций «А и не В», задают невыпуклую область. При добавлении нейронов и весов число сторон многоугольников может неограниченно возрастать. Это позволяет аппроксимировать область любой формы с любой точностью. Вдобавок не все выходные области второго слоя должны пересекаться.

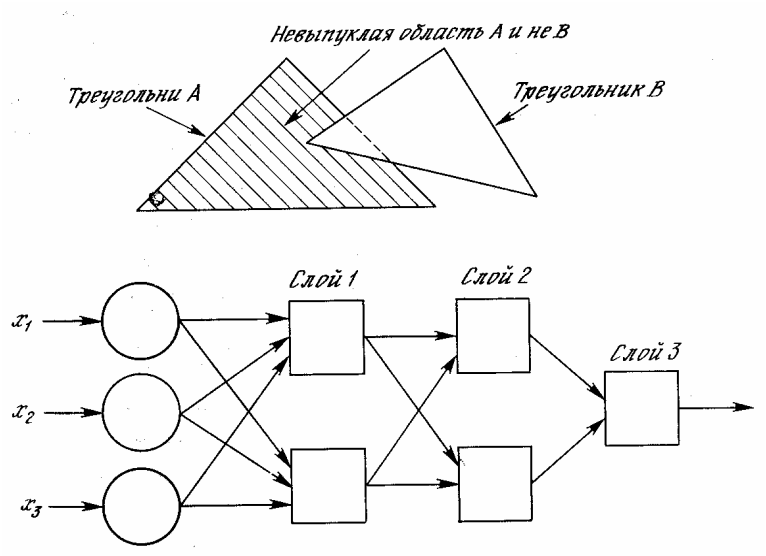


Рис. 2.16 “Вогнутая” область решений, задаваемая трехслойной сетью.

Возможно, следовательно, объединять различные области, выпуклые и невыпуклые, выдавая на выходе единицу, всякий раз, когда входной вектор принадлежит одной из них. Несмотря на то, что возможности многослойных сетей были известны давно, в течение многих лет не было теоретически обоснованного алгоритма для настройки их весов.

2.3.7 Эффективность запоминания.

Серьезные вопросы имеются относительно эффективности запоминания информации в персептроне (или любых других нейронных сетях) по сравнению с обычной компьютерной памятью и методами поиска информации в ней [49]. Например, в компьютерной памяти можно хранить все входные образы вместе с классифицирующими битами. Компьютер должен найти требуемый образ и дать его классификацию. Различные хорошо известные методы могли бы быть использованы для ускорения поиска. Если точное соответствие не найдено, то для ответа может быть использовано правило ближайшего соседа.

Число битов, необходимое для хранения этой же информации в весах персептрона, может быть значительно меньшим по сравнению с методом обычной компьютерной памяти, если образы допускают экономичную запись. Однако, Минский [48] построил примеры, в которых число битов, требуемых для представления весов, растет с размерностью задачи

быстрее, чем экспоненциально. В этих случаях требования к памяти с ростом размерности задачи быстро становятся невыполнимыми. Если, как он предположил, эта ситуация не является исключением, то персептроны часто могут быть ограничены только малыми задачами.

2.3.8 Обучение персептрона.

Способность искусственных нейронных сетей обучаться является их наиболее важным свойством. Подобно биологическим системам, которые они моделируют, эти нейронные сети сами моделируют себя в результате попыток достичь лучшей модели поведения [70].

Используя критерий линейной разделимости, можно решить, способна ли однослойная нейронная сеть реализовывать требуемую функцию. Даже в том случае, когда ответ положительный, это принесет мало пользы, если у нас нет способа найти нужные значения для весов и порогов. Чтобы сеть представляла практическую ценность, нужен систематический метод (алгоритм) для вычисления этих значений. Розенблатт сделал это в своем алгоритме обучения персептрона вместе с доказательством того, что персептрон может быть обучен всему, что он может реализовывать.

Обучение может быть с учителем или без него. Для обучения с учителем нужен «внешний» учитель, который оценивал бы поведение системы и управлял ее последующими модификациями. При обучении без учителя, сеть путем самоорганизации делает требуемые изменения [24].

Алгоритм обучения персептрона может быть реализован на цифровом компьютере или другом электронном устройстве, и сеть становится в определенном смысле самоподстраивающейся. По этой причине процедуру подстройки весов обычно называют «обучением» и говорят, что сеть «обучается». Доказательство Розенблатта стало основной вехой и дало мощный импульс исследованиям в этой области.

ГЛАВА 3. ЗАДАЧА МОДЕЛИРОВАНИЯ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ ДЛЯ РАСПОЗНАВАНИЯ ОБРАЗОВ.

3.1 Постановка задачи и сложности, связанные с ее реализацией.

В настоящее время одним из актуальных вопросов информационных технологий является проблема распознавания образов с применением эффективных методов нейрообработки информации. Стандартные схемы анализа изображений и оценки данных, не всегда приносят желаемых результатов, т.к. они негибкие и привязаны к определенному неадаптируемому алгоритму.

Существует различные модели, которые способны распознавать печатные образы, однако их основным недостатком является тот факт, что, они подстроены к конкретным образам и отсутствует возможность самоадаптации [71]. Более универсальным было бы применение альтернатив, позволяющих модели самонастроиться под новые недетерминированные объекты.

Данной работой ставиться задача возможности распознавания рукописных символов с помощью комплексной модели, искусственной нейронной сети, использующей разделенную модель нейронов и вероятностно-статистический анализ растровых массивов. Для выполнения этой цели была построена гибридная программная модель, апробированная на примере распознавания арабских цифр, латинского и грузинского алфавитов, но не привязанная ни к какому конкретному набору символов и которая может быть применена для любых алфавитов и контурных изображений [5].

Разрешимость проблемы зависит от многих факторов. Прежде всего, необходимо набрать достаточную информацию об исследуемых образах – Базу Данных (БД) представителей каждого класса. Точность и эффективность распознавания символов непосредственно зависит от хорошо подобранных и откалиброванных эталонных представителей. Внесение в эталонную БД плохо детерминированных и сильно искаженных объектов может повлечь за собой трудности при классификации образов, что существенно снизит процент распознавания [76].

Условием для качественного распознавания является достаточный набор введенных эталонов, используя которые можно проводить аналитические сверки и далее применять алгоритмы нейросетовой идентификации. Для самодостаточности эталонов представители каждого символа БД должны обладать следующими характеристиками: быть классифицируемыми, не быть сильно схожими между собой и не содержать сильно деформированные элементы внутри класса.

В процессе ассоциирования новых изображений с объектами из БД, могут возникать проблемы, связанные с изобразительными неопределенностями, когда трудно или даже невозможно человеку или машине дать однозначный ответ, к какому образу принадлежит изображение. Очевидным примером трудно классифицируемых образов могут быть символы, не попавшие в замкнутые области, приведенные на рис 3.1.

Между “хорошей ”тройкой и “хорошей” пятеркой лежит целый ряд фигур. Часть этих фигур мы уверенно примем за тройку, часть за пятерку, некоторые вызовут сомнения. Чтобы исключить подобную неоднозначность, можно прибегнуть к голосованию, условившись называть тройкой (пятеркой, шестеркой, восьмеркой и т. д.) объекты, которые будут признаны таковыми определенным большинством людей из каждой тысячи. Из рисунка видно, как последовательными и небольшими изменениями превратить цифру три в пять и далее либо в шесть, либо в восемь. Этот пример хорошо иллюстрирует, как небольшие видоизменения приводят к неопределенности и трудности идентификации. Учитывая то, что в нашей задаче мы имеем дело с алфавитно-цифровым набором, состоящим из 69 символов (10 цифр, 26 букв латинского и 33 буквы грузинского алфавитов), можно предвидеть, что здесь может иметь место большое количество перекрестных искажений. Например, только из нижнего рисунка видно, что переходной символ между тройкой и пятеркой может быть похож не только на “3” или “5”, но и на грузинскую букву “ვ”, “ვ" или даже “Დ”, а усредненный символ между “5” и “8” может быть схожим с латинской буквой “S” или грузинской буквой “Ს”. Поэтому построена модель программы, использующая статистическую обработку изображений. В процессе обработки эта программа выявляет характеристики символов подаваемых на вход и группирует наиболее близкие друг к другу символы.

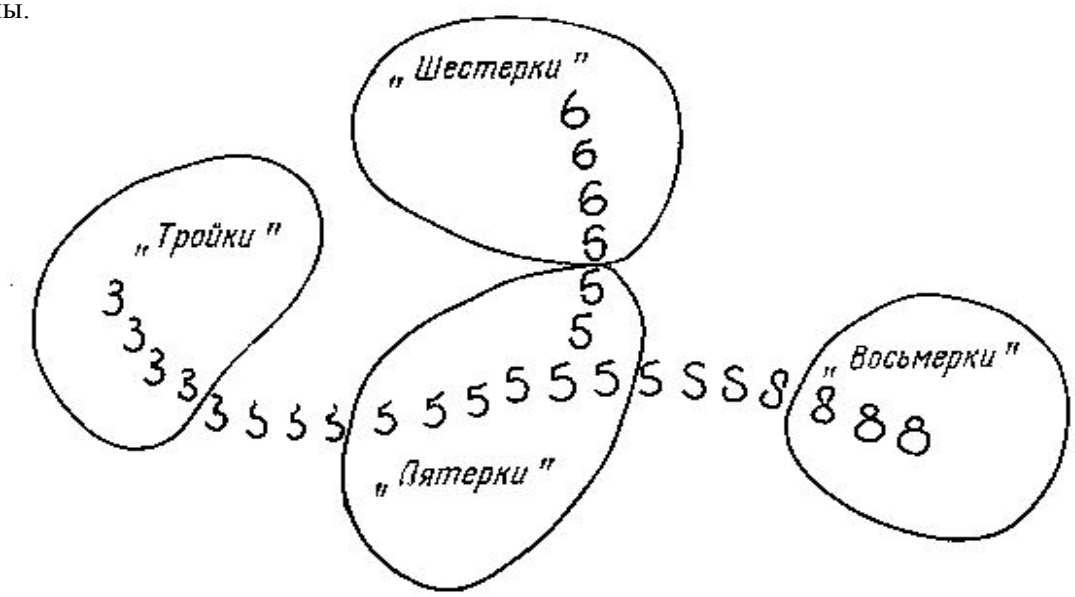


Рис. 3.1 Искажения образов.

Необходимо отметить, что изображения символов одного и того же класса, могут существенно отличаться друг от друга, хотя по сути мы имеем дело с одним и тем же образом так, например, цифра четыре имеет двоякое символьное изображение: “треугольное” и “четырехугольное” (рис. 3.2). Если собирать “треугольные” и “прямоугольные” представители цифры четыре в один и тот же класс, то это может привести к неразрешимости распознавания данного образа. Для разрешения этой проблемы необходимо разбить базу данных эталонов четверок на два составных класса: в один класс будут входить “треугольные” четверки, а другой – “прямоугольные”. При распознавании символа, необходимо применять сверки с каждым классом и если введенное значение близко к виду “прямоугольного” или “треугольного” символьного написания четверки, то в программе оба символа нужно ассоциировать с одним и тем же образом – цифрой четыре. Аналогично на два составных класса необходимо разбить базу данных эталонов семерок, единиц и других символов, которые имеют одно и тоже смысловое значение (рис 3.2), но имеют разное изображение.

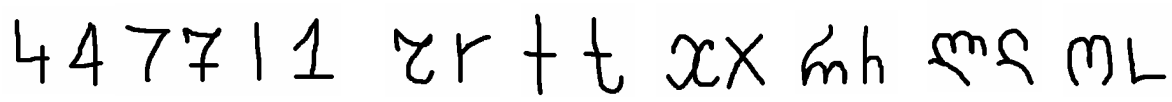


Рис. 3.2 Различное представление одних и тех же букв и цифр.

Написанная от руки одна и та же буква может иметь различное изображение в виде прописных и печатных начертаний, как видно из рис. 3.3. В таком случае необходимо также разбивать базу этих образов на столько классов, сколько принципиально различных символьных представителей имеет соответствующий образ.

Рис. 3.3 Примеры прописных и печатных изображение.

Аналогичным образом необходимо разделять заглавные и строчные буквы в случае латинского алфавита на два независимых класса, так как, например, изображение буквы “A” и “a” по сути дела различны (рис. 3.4). При распознавании, изображение должно быть ассоциировано конкретно с заглавной или строчной буквой, так как это имеет существенное значение для текста, и в особенности для математических формул. Разбивка на такие классы способствует правильности определения, точности и сходимости распознавания.

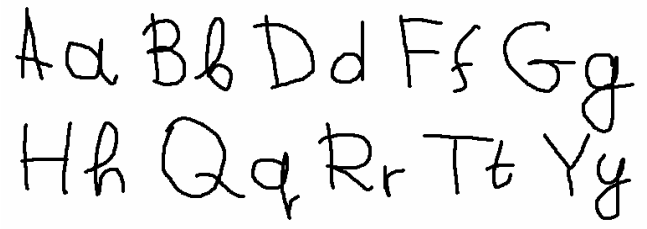


Рис. 3.4 Примеры изображений заглавных и строчных букв.

Таким образом, в случае необходимости, БД для каждого образа должна быть сепарирована на классы синонимных символов, которые должны ассоциироваться с одним и тем же образом. Если подсчитать все разновидности символьных изображений, то их количество будет уже не 69 (цифр, букв латинского и грузинского алфавитов), а гораздо больше с учетом разнообразности представителей.

Если БД, с учетом всего выше сказанного, подобранна удачно, то разработанная модель будет меньше нуждаться в корректировке и правильнее распознавать образы.

После разделения гомоморфных изображения на ассоциированные классы, следующим этапом задачи является построение нейросетевой модели распознавания образов. При удачном выборе архитектуры и механики функционирования сети можно использовать модель, схожую по свойствам с нейронной моделью Хопфилда, когда, предварительно исследуя и выявляя свойства различных образов, находят определенные закономерности и тщательно подбирают весовые коэффициенты синоптических узлов ИНС на основе экспериментальных калибровок.

При плохо диверсифицированной БД образов на различные группы символов по определенным нейросетевым характеристикам, задача может быть трудноразрешимой, когда некоторые символы будут иметь схожие свойства и взаимно накладываться друг на друга в пространстве признаков. В этом случае использование метода Хопфилда будет более эффективным в сочетании с корректирующими циклическими обработками. Для лучшего разрешения проблемы при необходимости можно использовать комбинированную нейросетевую схему настройки весов с использованием алгоритма обратного распространения ошибки. Это позволит ИНС в рабочих циклах самонастроиться на этапе обучения и постепенно устранить погрешности [44]. Самоподстройка или обучение может осуществляться как с учителем, т.е. когда пользователь может делать предпочтение, к какому образу более близок поступивший на вход программы символ, так и без учителя, путем автоматической самонастройки соответствующих весов нейронной матрицы на разных уровнях без участия человека. Реализация обучения без учителя является наиболее эффективным и автоматизированным способом, но в тоже время более сложным для имплементации, тем не менее, именно этот способ был заложен в основу алгоритма для РО.

Трансформирование введенного изображения из графического в матричный вид.

При распознавании образов, изображение подают на матрицу рецепторов по аналогии с тем, как изображение попадает на сетчатку человеческого глаза. Воспринимаемое изображение далее поступает в мозг для последующей переработки и распознавания, т.е. причисления образа к тому или иному классу символов. По аналогии строится модель с полем рецепторов, представляющим собой прямоугольный массив, на котором можно изображать всевозможные конфигурации символов. Ввод информации в компьютер, представляет собой процесс сканирования изображения символов с помощью периферийного устройства – сканера, и сохранения изображения в формате графического файла. Другим способом может служить создание изображения в любой графической программе. Сохраненное изображение представляет собой графический файл, т.е. последовательность кодовых знаков, которые несут в себе информацию о структуре визуального изображения. Однако для использования этой информации в математических целях, необходимо преобразовать этот последовательный зашифрованный код в более доступное матричное представление нулей и единиц (0 – не закрашенные, 1 – закрашенные точки).

Существующие на рынке программы распознавания текстов применяют лексикон слов для сверки текста с целью устранения возможных ошибок, что позволяет искусственно поднять процент распознавания образов. Кроме того, имеющиеся ПО для распознавания образов, в основном, работают под определенные шрифты, что является другим недостатком и сужает круг задач распознавания. Современные программы распознавания и исследования образов также используют матрицы с высоким разрешением. Чем выше разрешение (например, номинальное 120 по горизонтали на 160 по вертикали = 19 200 точек), тем более точно можно выявить закономерности и правильно подстроить рекогнационную модель. Для получения более точных характеристик должна быть введена достаточно большая БД эталонных элементов, состоящая не менее 1 000 разновидностей одного элемента. Набор эталонной БД для последующего скрупулезного исследования графической структуры изображения занимает очень трудоемкое время и ресурсы. Если, например, номинальное количество эталонных экземпляров 1 000 для одного символа и разрешение 19 200 точек (120x160), то объем информации при 25%-ой закраске будет занимать 4 800 000 точек. При учете, если мы в общем случае имеем дело, например со 100 разновидностями символов (заглавные, строчные, синонимные и др.), то объем информации будет составлять 480 000 000 (480 миллионов точек) эталонной БД. Очевидно, что такая задача не посильна для одного исследователя и требуется большой коллектив операторов.

Настоящей работой ставилась задача достижения распознавания при редуцировании количества эталонных экземпляров и разрешения матрицы рецепторов без потери качества. Поэтому была выбрана оптимальная матрица с разрешением 12x16 (т.е. 192 точки, что в 100 раз меньше, чем номинальное – 19 200 точек) и количеством представителей каждого символа – 40 (вместо 1 000 номинально вводимых в БД), которая не смотря на ухудшение условий идентификации, дала высокие результаты распознавания объектов благодаря хорошо подобранной ИСНС. Таким образом, достоинством данного метода является, то, что вместо 480 000 000 вносимых точек, над которой должна была трудиться большая группа операторов БД для ввода информации, мы имеем дело со 192 000 точками, что на 479 808 000 меньше для ввода изображений, тем не менее позволяет достичь высокие результаты. Уменьшение трудоемкости работы при тех же результатах распознавания является одним из позитивных моментов созданной модели. Таким образом, для каждого символа из набора цифр, букв латинского и грузинского алфавитов вручную было введено 40 экземпляров рукописных начертаний.

Процесс ввода заключался в закрашивании белого поля рецепторов, черными точками. Поле рецепторов состоит из 192-ух квадратных клеток, что получается в результате произведения 12 колонок по вертикали и 16 клеток по горизонтали. На рис. 3.5 приведен пример заполнения поля рецептора 34-мя точками, в результате которого получается графически-цифровое изображение цифры семь.

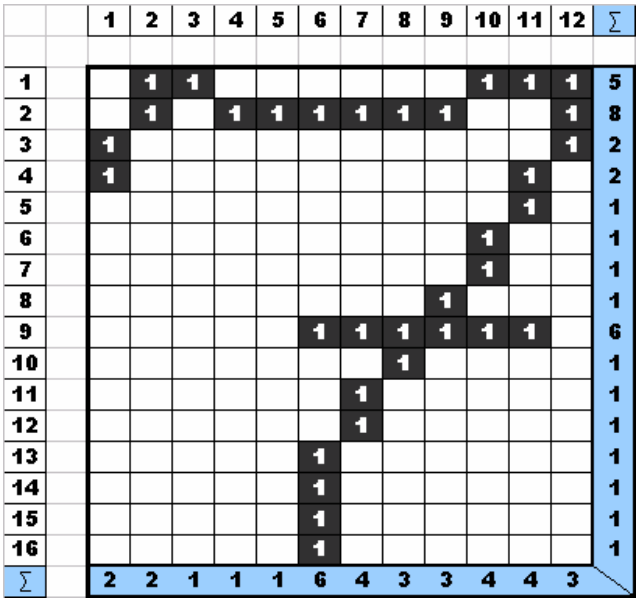


Рис. 3.5 Матрица рецепторов.

Очевидно, что такое разрешение является невысоким, но достаточным для изображения отличимых друг от друга символов. Чем больше количество точек матрицы рецепторов, иначе разрешения тем точнее можно анализировать и идентифицировать признаки символов. Чем меньше матрица рецепторов, тем сложнее выявить характеристики изображений и больше погрешностей будет появляться связанных с метаморфозным сходством. Тривиальный пример показывает, что если у нас, например, матрица рецепторов 2x2 точек, то в этом случае не будет возможности представить каждый символ уникально. Всевозможное количество комбинаций графических изображений на поле рецепторов будет составлять 2^4, что составит 16 изображений (коды 16-ти всех возможных комбинаций

изображений: 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111, где 0 – не закрашенная точка, а 1 – закрашенная точка на поле) и естественно при наборе 69 символов будут метаморфозные сходства; в этом случае невозможно будет ввести уникальные и отличимые друг от друга символы. Как уже было сказано, в результате исследований были рассмотрены различные разрешения матриц, из которых была выбрана оптимальная матрица для ввода всех символьных изображений с разрешением 12x16 точек. Также в исследуемой задаче было экспериментально подобрано необходимое и достаточное количество вводимых изображений – 40 экземпляров эталонов для каждого из 69 символов (цифр, букв латинского и грузинского алфавитов). При вводе намного меньшего, чем 40 количества экземпляров, характеристики объектов получались ”размытыми” и выявление по признакам усложнялось. При вводе же существенно большего, чем 40 эталонов, объем ввода геометрически увеличивался (в пересчете на введенные черные и белые точки в случае 69 символов, разрешении 120x160 точек и количестве эталонных представителей 1 000, получаем 1.32 миллиарда закрашенных и незакрашенных точек), без существенного эффекта для распознавания.

На рис. 3.6 приводятся фрагменты некоторых начертаний цифр и букв в количестве 40 представителей для каждого образа из БД [2]. Как видно из рисунка, все начертания каждого образа отличаются друг от друга произвольностью изображения.

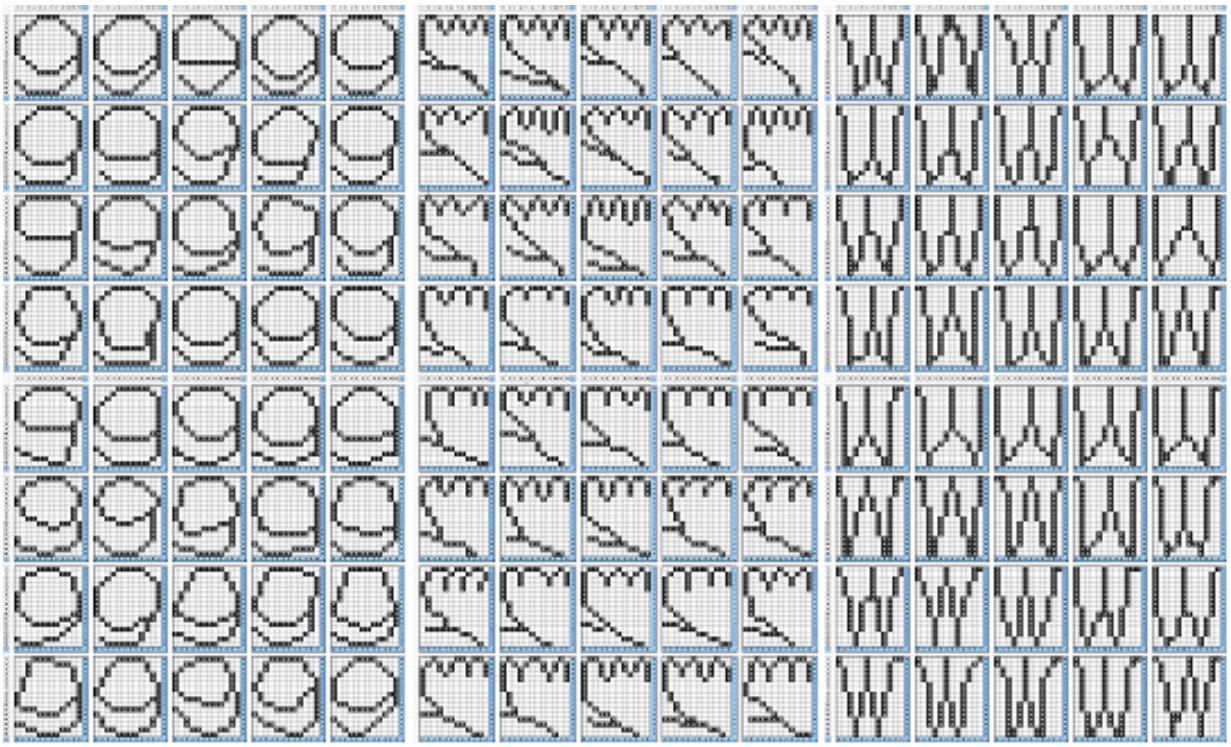


Рис. 3.6 Примеры введенных символов.

Для диверсификации эталонных элементов БД изображения символов были введены разнообразным начертаниями вручную. Далее в процессе анализа статистических и вероятностных закономерностей были подсчитаны характеристики каждого представителя класса. Эффективность алгоритма и метода должна основываться на базе набора, но ключевую роль играет сам алгоритм интеллектуальной обработки и группировки данных. Первоначально для анализа изображения был использован модифицированный метод, основанный на методе зондов (рис. 3.7). Как видно из рисунка, в традиционном методе используются 7 зондов для идентификации цифр, причем исключительно печатных. По количеству пересечений определяется принадлежность к той или иной цифре (например, цифра 2 из рисунка имеет 4 пересечения с зондами). Для идентификации дополнительных символов, например, латинского и грузинских алфавитов, такая схема уже не срабатывает в связи большим набором представителей и имеющихся сходных свойств между различными символами. Кроме того, этот метод очень чувствителен к малейшим искажениям, например, если удлинить кончик головки цифры два как на рис. 3.8, то получим вместо 4-ех пересечений с зондами – 5 пересечений. Этот простой пример показывает, как малейшее изменение изображения символа негативно влияет на процесс распознавания. Для экспериментального определения характеристик различных символов, в изображения вводилось более насыщенное количество условных зондов, а именно горизонтальные и вертикальные, совпадающее с количеством пикселей по вертикали и горизонтали для подсчета количества их пересечений с изображением. Очевидно, что такая структура лучше будет идентифицировать изображение по сравнению с “разжиженными” зондами. Однако, вопрос остается открытым в случае искажения символа, когда мы имеем дело с непечатными знаками, так как это будет влиять на количество пересечений с горизонтальными/вертикальными зондами, и будет трудно, или даже невозможно, однозначно определить точную принадлежность поступившего изображения на сетчатку. Можно попытаться объединить в группы гомоморфные изображения одного и того же образа, путем подбора и ассоциирования мульти характеристик пересечений зондов для дальнейшего детектирования конкретного образа, однако это не разрешает полностью вопрос распознавания не строго формализованных изображений.

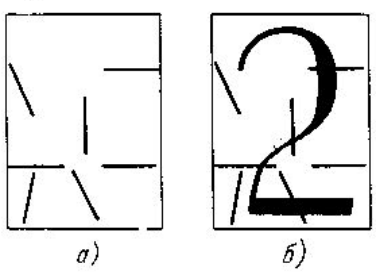


Рис. 3.7 Схема зондов.


Рис. 3.8 Пример искажения.

Для поиска возможных решений каждый символ был тщательно изучен, построены графики зависимостей вертикальных и горизонтальных составляющих изображения – зонды, подсчитаны вероятности, медианы, контрольные суммы и построена 3D колонная диаграмма точечных вероятностей. Также были созданы усредненные контуры каждого символа и построена нейронная самообучающаяся сеть обработки изображения для улучшения показателей сходимости и распознавания. Изображение, представляющее символ, предназначенное для распознавания может быть отсканировано как с разрешением 12x16 точек (пикселей), так и с другим разрешением и после чего должно быть приведенным к этому разрешению путем операции сжатия/растягивания, соответствующему матрице рецепторов для обработки и оценки изображения.

Очистка погрешностей и придание контрастности.

Рассмотрим, какие проблемы могут возникнуть после сканирования изображения, и какие механизмы могут быть задействованы. Очистка погрешностей должна предшествовать структурному анализу изображения [3]. Отсканированное изображение может содержать элементы погрешностей, такие как пыль, отпечатки пальцев и царапины на стекле сканера, размазанность изображения карандашом, изгибы, шероховатость поверхности на листе бумаге и др. Необходимо проанализировать и выявить погрешности с изображения с последующим удалением нежелательных дефектов, негативно влияющих на процесс распознавания образов. Оттенок бумаги также может создавать проблему при сканировании. Для этого нужно преобразовать оттеночный фон в монотонно белый цвет для получения контрастного электронного файла изображения. Также, необходимо учитывать, что введенное изображение должно касаться всех четырех границ матрицы рецепторов, т.е.

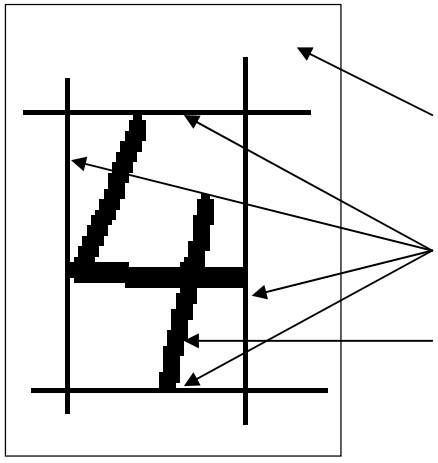
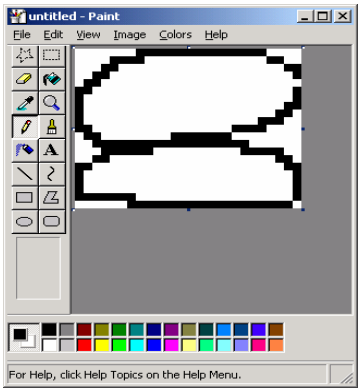


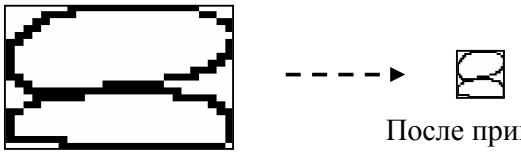
Рис 3.9 Границы отсечения контуров символа.

верхней, нижней, левой и правой. Если к слову мы нарисовали на листе бумаги символ и отсканировали его, то естественно необходимо будет найти все четыре границы нарисованного символа и применить операцию отсечения (рис. 3.9). Процедура поиска

изображения на листе бумаги будет заключаться в нахождении закрашенного объекта и определении максимумов и минимумов в области изображения по горизонтальным и вертикальным составляющим.

Далее если размеры объекта изображения с отсеченными окаймлениями не совпадает с размером матрицы 12x16, необходимо привести его к этому разрешению путем операции сжатия/растяжения в любом графическом ПО.





После приведения к пропорции 12x16 сжатия.

Рис 3.10 Приведение изображения к пропорции 12x16.

Пример приведения изображения к разрешению 12x16, соответствующего матрице рецепторов программы, показан на рис 3.10. Таким образом, всегда можно преобразовать изображение к необходимому размеру матрицы. Это может повлечь за собой некоторое искажение изображения, однако программа распознавания успешно с ними справляется.

Можно нарисовать изображение какого-либо символа вручную (рис. 3.11) в любой графической программе, как, например, в MS Paint и сохранить его в графическом формате Bit Map (BMP).

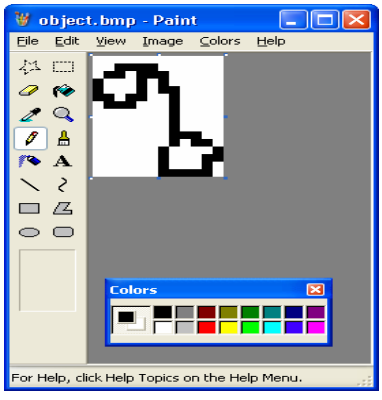


Рис 3.11 Пример создания буквы в программе MS Paint.

Графический формат BMP имеет различные разновидности, включая монохромный, 16-цветный, 256-ти цветовой палитрой и 24-ех битный (16.4 миллионов цветов). Для распознавания шрифтов достаточно использовать и монохромный формат. В настоящей работе, использован 16-цветный формат. Кроме цветовой палитры, которая определяет

количество различных цветов для каждой точки изображения, все эти форматы BMP существенно отличаются друг от друга структурой и сложностью хранения графической информации.

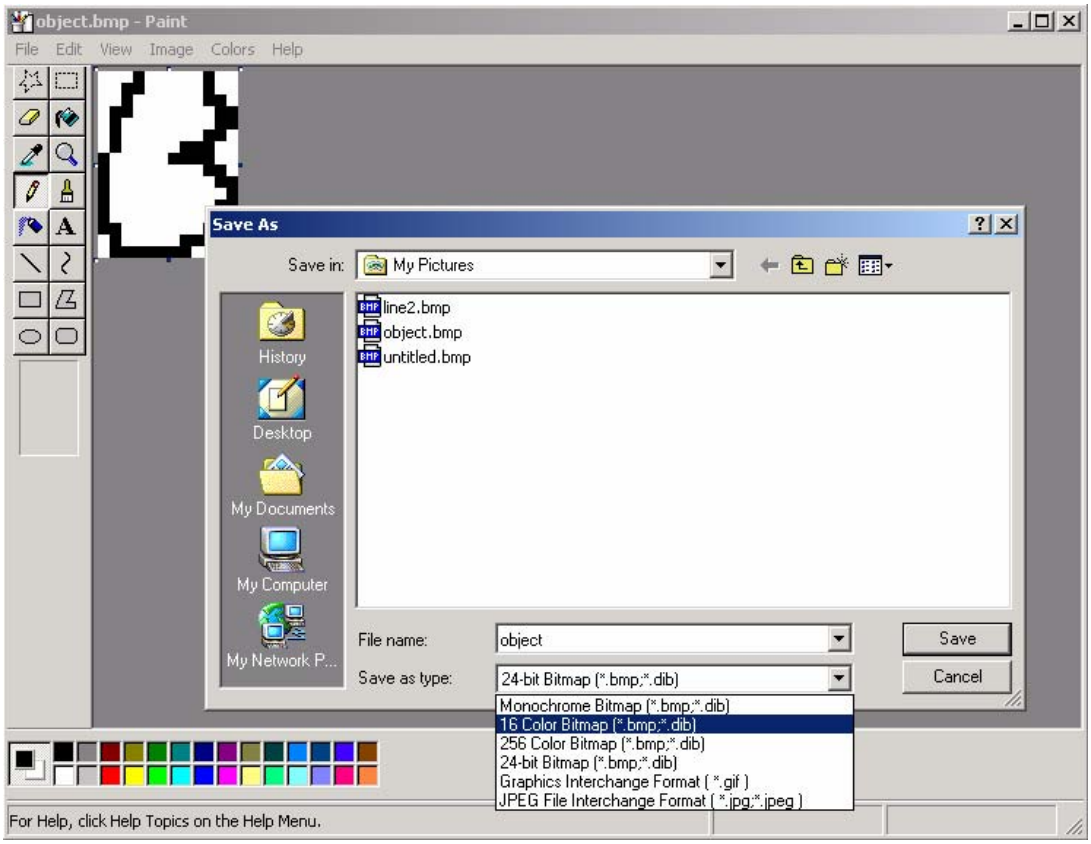


Рис. 3.12 Сохранение изображения в формате 16-ти цветного BMP.

Приведем таблицу 3.1, описывающую структуру построения графического изображения в 16-ти цветовом формате файла BMP.

Таблица 3.1 Структура блоков графического BMP файла.

Имя	**Длина**	**Смещение**	**Описание**
Заголовок файла (BitMapFileHeader)			
Type	2	0	Сигнатура "BM"
Size	4	2	Размер файла
Reserved 1	2	6	Зарезервировано
Reserved 2	2	8	Зарезервировано
OffsetBits	4	10	Смещение изображения от начала файла

Информационный заголовок (BitMapInfoHeader)			
Size	4	14	Длина заголовка
Width	4	18	Ширина изображения, точки
Height	4	22	Высота изображения, точки
Planes	2	26	Число плоскостей
BitCount	2	28	Глубина цвета, бит на точку
Compression	4	30	Тип компрессии (0 - несжатое изображение)
SizeImage	4	34	Размер изображения, байт
XpelsPerMeter	4	38	Горизонтальное разрешение, точки на метр
YpelsPerMeter	4	42	Вертикальное разрешение, точки на метр
ColorsUsed	4	46	Число используемых цветов (0 - максимально возможное для данной глубины цвета)
ColorsImportant	4	50	Число основных цветов
Таблица цветов (палитра) (ColorTable)			
ColorTable	1024	54	256 элементов по 4 байта
Данные изображения (BitMap Array)			
Image	Size	1078	Изображение, записанное по строкам слева

Просмотр кодированного BMP файла хранящего изображение.

На примере созданного или отсканированного изображения буквы “Q” (рис. 3.13) рассмотрим содержание и структуру графического файла BMP.

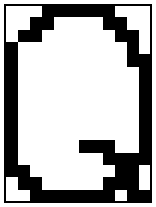


Рис. 3.13 Отсканированная буква “Q”.

Если заглянуть внутрь графического файла, хранящего изображенную букву ”Q”, с помощью стандартного текстового редактора, то можно увидеть следующий кадр зашифрованного изображения (рис. 3.14), однако он не предоставляет полную информацию и скрывает, невидимые символы, например, переноса каретки и строк [1].

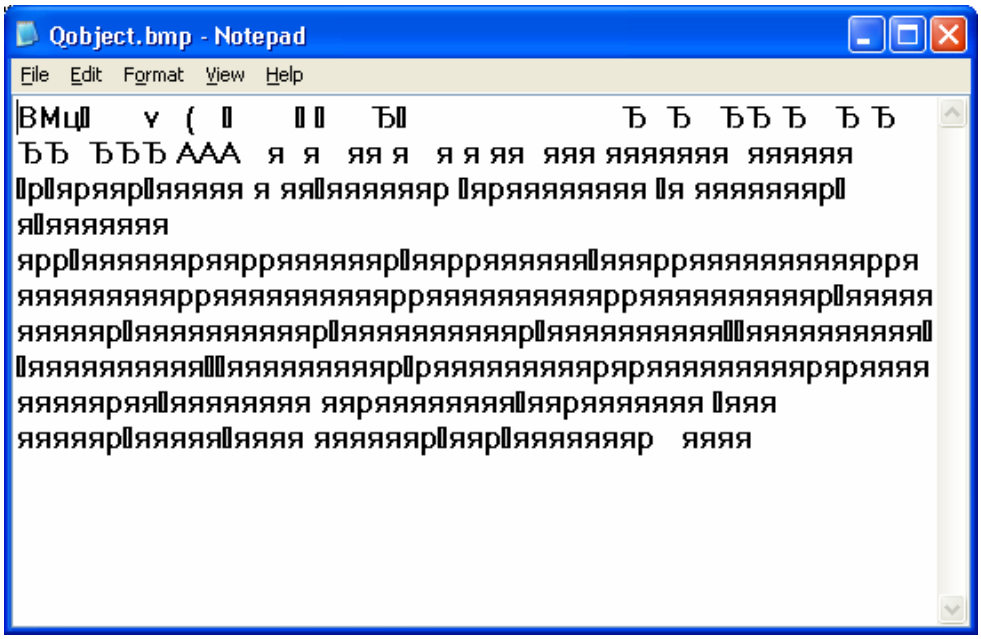


Рис. 3.14 Содержание BMP файла, представляющего букву “Q” в MS Notepad.

Просмотр в 16-ом HEX / ASCII формате с помощью, например, программы VC дает более детализированное и точное представление о структуре графического изображения.

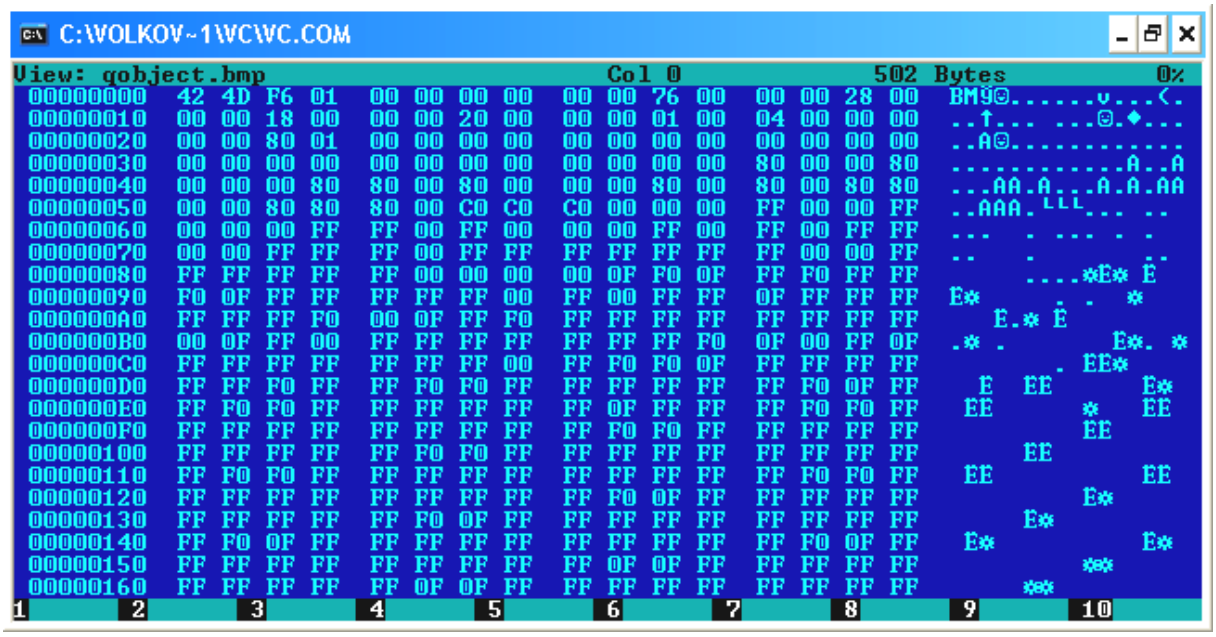


Рис. 3.15 Содержание изображения в 16-ном машинном представлении.

3.1.2 Декодирование изображения в BMP.

Для анализа изображения необходимо декодировать его в более доступный для восприятия матричный вид [3], используя вышеприведенную таблицу 3.1 структуры BMP файла. Конечно, можно использовать алгоритм преобразования внутреннего анализа, но от

этого пострадает эффективность и трансмиссия задачи. Задача заключается в том, чтобы программа после предъявления нового изображения для его распознавания на начальном этапе, смогла считать с графического файла (записанного на диске), представляющего собой последовательность зашифрованной информации (рис. 3.15) и преобразовать ее в доступный для анализа табличный вид, последовательности “0” и “1” (белых и черных точек, однозначно определяющих изображение) табличного массива, т.е. декодировать это изображение для дальнейшего математического анализа. Будем представлять изображение в виде матрицы 12x16. Путем декодирования и преобразования из черно-белого Black&White BMP формата, буквы “Q”, можно получить ее табличное представление в двухмерном массиве программы в виде записи “0” и “1”.

Используя программу WinHex, можно просмотреть в более наглядном виде структуру закодированного графического изображения символа в 16-ричной системе исчисления (рис. 3.16). Можно заметить, что код “FF” внутри шифра, определяет белую, не закрашенную точку, а “00” – черную. Однако коды “00” находятся и в начале файла, хотя они и не связаны с реальным изображением буквы “Q”, а несут только служебную информацию заголовка файла BMP. Заголовок BMP хранит служебную информацию о структуре, формате, размере, глубине цвета, типе компрессии [54], числе плоскостей, ширине, высоте, зарезервированных полях и др. Эта информация не используется для декодирования символов в программе.

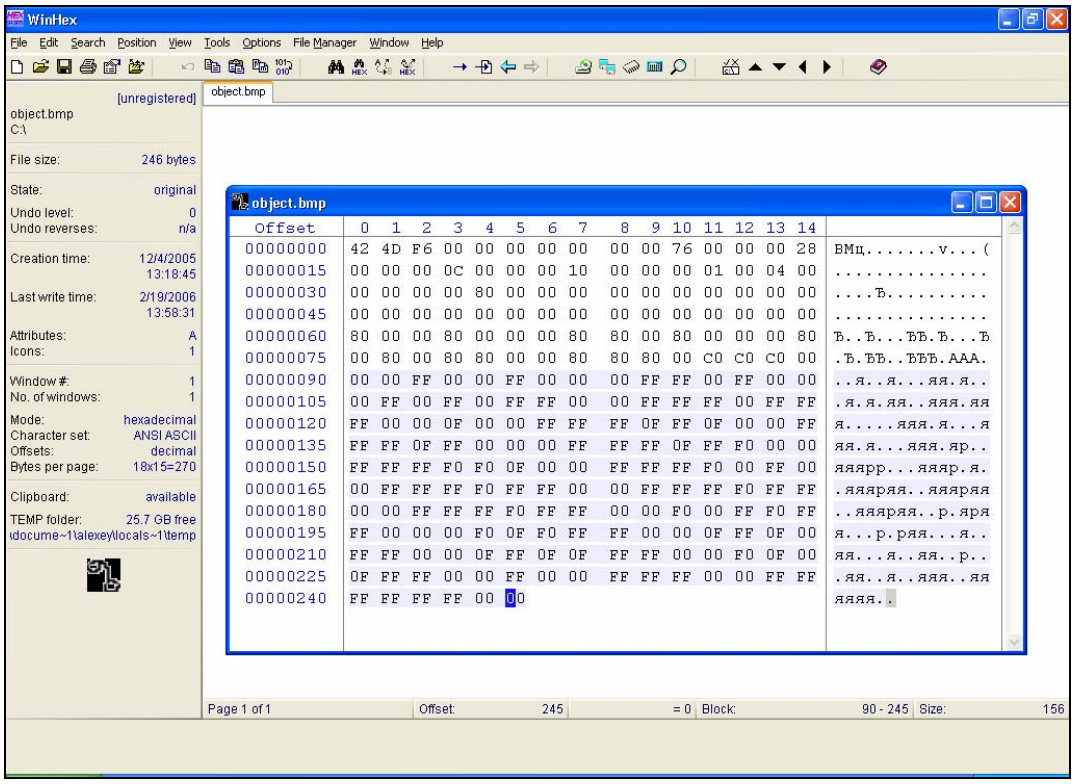


Рис. 3.16 Содержание графического файла буквы “Q” в HEX формате в программе WinHex.

Само изображение начинается с кодового блока с индексом 118 в HEX формате. Однако, при преобразовании с закодированного изображения в доступный матричный вид, необходимо учитывать множество нюансов, связанных с кодированием изображения в файле. Из данного примера видно, что изображение содержит разделительные блоки между строками, хранится в перевернутом и транспонированном виде (изображение, записанное по строкам слева направо и снизу вверх), поэтому в процессе дешифрования необходимо провести операции преобразования и транспонирования для приведения матрицы в нормализованный вид. Процедура всех этих преобразований, декодирования, удаления служебных блоков, транспонирования матрицы и отображения приведены в листинге 3.1 программы.

Листинг 3.1 – Процедура трансформации графического изображения в 2-ух мерный массив.

```
procedure TForm1.Button57Click(Sender: TObject);
var
  FromF: file;
  FileFrom : string;
  NumRead: Word;
  Buffer: array[1..1024] of Byte;
  i, j, k, l, m, count: Integer;
  pw, ph : Integer;
begin
  FileFrom := 'c:\object.bmp';
  AssignFile(FromF, FileFrom);
  Reset(FromF, 1);
//  NumRead := 5;
  BlockRead(FromF, Buffer, SizeOf(Buffer), count);
  pw := Buffer[19];
  ph := Buffer[23];

  label9.caption := InttoStr(Buffer[19]);
  label10.caption := InttoStr(Buffer[23]);

  CloseFile(FromF) ;
  k:=0; i:=1; j:=0;

  for i:=1 to ph do
   begin
    while j<pw do
//    for j:=1 to 12 do
     begin
     j:= j+2;
     //    Form1.StringGrid1.Cells[j,i] := Inttostr(Buffer[118+k+j]);
      m := trunc(j/2);
      if Buffer[118+k+m]=0 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '1';
        Form1.StringGrid1.Cells[j,ph+1-i] := '1';
       end;
      if Buffer[118+k+m]=15 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '1';
        Form1.StringGrid1.Cells[j,ph+1-i] := '0';
       end;
```

```
      if Buffer[118+k+m]=240 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '0';
        Form1.StringGrid1.Cells[j,ph+1-i] := '1';
       end;
      if Buffer[118+k+m]=255 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '0';
        Form1.StringGrid1.Cells[j,ph+1-i] := '0';
       end;
     end;
//    Form1.StringGrid1.Cells[j,i] := Inttostr(Buffer[118+k+j]);
   k:=k+trunc(pw/2)+2;
   j:=0;
 end;
end;
```

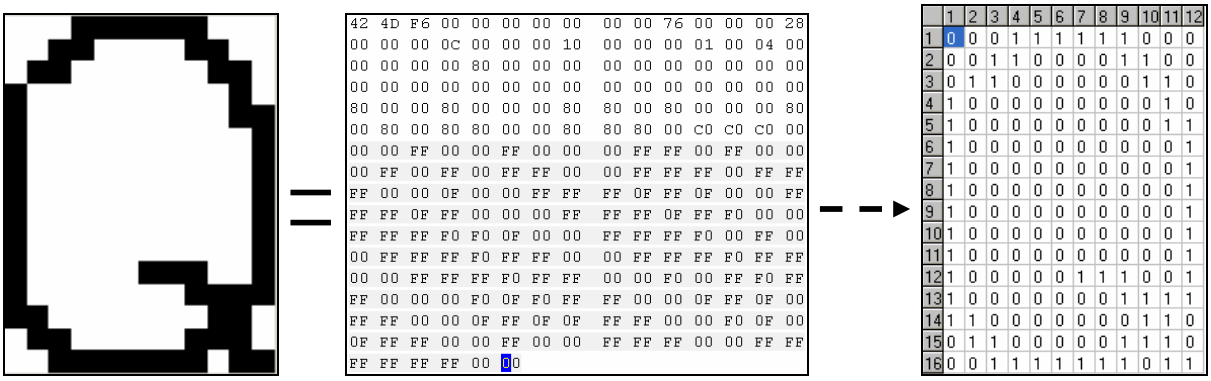


Рис. 3.17 Преобразование изображения из графического вида в цифровой табличный массив.

После получения матричной записи в виде массива программы (рис 3.17), следующим этапом является комбинированная обработка поступившего на вход изображения с БД, для последующего распознавания. Основной целью поставленной задачи было, не просто добиться распознавания образов, а построить самообучающуюся модель без учителя, применяя искусственные нейро-матричные схемы (с обратным распространением) и вероятностно-статистический анализ с использованием минимальной БД представителей каждого символа и низкой плотности матрицы рецепторов (т.е. идентификация образов на основе введенного компактного набора эталонных элементов).

3.2 Алгоритм и блок-схема распознавания образов.

Для каждой из 69 исследуемых символов (цифр и букв) были построены две таблицы: названные сумматорными матрицами точечных весов и точечных вероятностей. Эти матрицы представляли собой таблицы, содержащие соответственно суммарные значения точечных весов и точечных вероятностей для введенных эталонов. Были введены новые термины, определяющие расположения зондов по вертикали и горизонтали, названые v-zond

(вертикальные) и h-zond (горизонтальные) соответственно и подсчитаны суммы весов по вертикальным и горизонтальным составляющим матриц в количестве соответственно 12 и 16 значений. На примере буквы “ა” грузинского алфавита, приведена сумматорная матрица точечных весов (рис. 3.18).

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	27	11	2	3	1	0	0	0	0	0	0	0	44
2	13	14	9	7	1	2	2	0	0	0	0	0	48
3	1	15	12	6	10	5	0	2	0	0	0	0	51
4	0	2	13	16	9	8	7	3	2	0	0	0	60
5	0	0	3	7	13	13	10	8	7	3	0	0	64
6	0	0	0	3	6	9	14	13	8	8	4	0	65
7	0	0	0	0	0	3	7	10	14	13	9	4	60
8	0	0	0	0	0	0	0	4	9	13	17	13	56
9	0	0	0	0	0	0	0	0	0	4	9	30	43
10	2	5	2	0	0	0	0	0	0	0	2	38	49
11	10	2	1	0	0	0	0	0	0	0	0	40	53
12	17	5	1	0	0	0	0	0	0	0	2	38	63
13	18	12	0	0	0	0	0	0	0	0	6	34	70
14	18	16	3	0	0	0	0	0	1	5	25	12	80
15	13	11	18	3	1	0	2	11	19	23	14	2	117
16	0	11	20	37	39	40	38	29	20	12	2	0	248
∑	119	104	84	82	80	80	80	80	80	81	90	211	

Рис. 3.18 Сумматорная матрица точечных весов.

Сумматорная таблица точечных вероятностей (рис. 3.19) имеет такое же структурное представление, но в отличие от первой, содержит вероятностно-статистические характеристики закрашенных точек для эталонных образов на поле рецепторов [2], а также содержит суммы по зондам v-zond и h-zond. В отличие от стандартного метода зондов здесь применяется насыщенное количество абстрактных зондов, в количестве – 28, полностью захватывающих поле рецепторов с целью улучшения детектирования характерных признаков различий между символами. Также, в отличие от стандартного метода зондов (в котором микро изменения формы ухудшает идентификацию), новый подход зондов, использует большее количество детекторов пересечений v/h-zond с активными точками, и в результате может лучше детектировать символы. Однако, эксперименты, проведанные в результате работы показали, что и эта модификация зондов при некоторых искажениях, также не разрешает вопрос идентификации образов, т.к. в этом методе количество пересечений точек матрицы с зондами не однозначно определяет позицию пересечения и в результате не однозначно будет идентифицировать патерн опознаваемого изображения. Вследствии этого для выявления характеристик изображений были исследованы сумматорные точечные вероятности, они были условно разделены на четыре диапазона (0,0.25], (0.25,0.5], (0.5,0.75],

(0.75,1.00] и ассоциированы с цветовой палитрой для визуального выделения доминирующих и рецессивных вероятностей:

Диапазоны вероятностей и соответствие цветов точек:

(0,0.25] – белые точки

(0.25,0.5] – светло-серые точки

рецессивные

(0.5,0.75] – темно-серые точки

(0.75,1.00] – черные точки

доминирующие

Это дает суммарную визуальную картину вероятностного распределения точек каждого символа на поле рецепторов. Общее количество статистически проанализированных точек данной задачи составило 529 920 (69 символов x 40 эталонов БД x 192 точек матрицы рецепторов).

1	2	3	4	5	6	7	8	9	10	11	12	∑
0.68	0.28	0.05	0.08	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.10
0.33	0.35	0.23	0.18	0.03	0.05	0.05	0.00	0.00	0.00	0.00	0.00	1.20
0.03	0.38	0.30	0.15	0.25	0.13	0.00	0.05	0.00	0.00	0.00	0.00	1.28
0.00	0.05	0.33	0.40	0.23	0.20	0.18	0.08	0.05	0.00	0.00	0.00	1.50
0.00	0.00	0.08	0.18	0.33	0.33	0.25	0.20	0.18	0.08	0.00	0.00	1.60
0.00	0.00	0.00	0.08	0.15	0.23	0.35	0.33	0.20	0.20	0.10	0.00	1.63
0.00	0.00	0.00	0.00	0.00	0.08	0.18	0.25	0.35	0.33	0.23	0.10	1.50
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.23	0.33	0.43	0.33	1.40
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.23	0.75	1.08
0.05	0.13	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.95	1.23
0.25	0.05	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.33
0.43	0.13	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.95	1.58
0.45	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15	0.85	1.75
0.45	0.40	0.08	0.00	0.00	0.00	0.00	0.00	0.03	0.13	0.63	0.30	2.00
0.33	0.28	0.45	0.08	0.03	0.00	0.05	0.28	0.48	0.58	0.35	0.05	2.93
0.00	0.28	0.50	0.93	0.98	1.00	0.95	0.73	0.50	0.30	0.05	0.00	6.20
2.98	2.60	2.10	2.05	2.00	2.00	2.00	2.00	2.00	2.03	2.25	5.28	

Рис. 3.19 Сумматорная матрица точечных вероятностей.

На основе матриц точечных вероятностей для каждого из 69 представителей по 40 экземпляров, были подсчитаны общие статистические данные (табл. 3.2): суммы весов матрицы, суммы вероятностей матрицы, среднее значение матрицы, медианна весов матрицы, а также количество точек, имеющие различные диапазоны вероятностей.

Автоматическая процедура (листинг 3.2) извлечения данных из MS Excel, хранящая базу данных характеристик каждого класса и импортирования в приложение следующая:

Листинг 3.2 Алгоритм преобразовния эталонной БД из MS Excel и загрузка в программу.

```
function Xls_To_StringGrid(AGrid: TStringGrid; AXLSFile: string): Boolean;
const
  xlCellTypeLastCell = $0000000B;
```

```
var
  XLApp, Sheet: OLEVariant;
  RangeMatrix: Variant;
  o, x7, y7, k7, r7: Integer;
begin
   Result := False;
  // Create Excel-OLE Object
  XLApp := CreateOleObject('Excel.Application');
  try
    // Hide Excel
    XLApp.Visible := False;
//    AXLSFile := 'C:\odb.xls';
    // Open the Workbook
    XLApp.Workbooks.Open(AXLSFile);

      // Sheet := XLApp.Workbooks[1].WorkSheets[1];
      Sheet := XLApp.Workbooks[ExtractFileName(AXLSFile)].WorkSheets[1];

    // In order to know the dimension of the WorkSheet, i.e the number of rows
    // and the number of columns, we activate the last non-empty cell of it

//    Sheet.Cells.SpecialCells(xlCellTypeLastCell, EmptyParam).Activate;
    // Get the value of the last row
    x7 := (16+4)*70+1; //XLApp.ActiveCell.Row;
    // Get the value of the last column
    y7 := 12+1; //XLApp.ActiveCell.Column;

    // Set Stringgrid's row &col dimensions.

    AGrid.RowCount := x7;
    AGrid.ColCount := y7;

    // Assign the Variant associated with the WorkSheet to the Delphi Variant
    RangeMatrix := XLApp.Range['A1', XLApp.Cells.Item[x7, y7]].Value;

    //  Define the loop for filling in the TStringGrid
    k7 := 1; o := 1;
    repeat
      for r7 := 1 to y7 do
        begin
        AGrid.Cells[r7, k7] := RangeMatrix[k7, r7];
//        odb[o,r7, k7] := AGrid.Cells[r7, k7];
        end;
      Inc(k7, 1); Inc(o,1);
      AGrid.RowCount := k7 + 1;
    until k7 > x7;

    // Unassign the Delphi Variant Matrix
    RangeMatrix := Unassigned;

  finally
    // Quit Excel
    if not VarIsEmpty(XLApp) then
     begin
      // XLApp.DisplayAlerts := False;
      XLApp.Quit;
      XLAPP := Unassigned;
      Sheet := Unassigned;
      Result := True;
     end;
  end;

end;
```

(черных) точек из следующих диапазонов: [0.00], (0.00-0.25], (0.25-0.50], (0.50-0.75], (0.75-1.00], [1.00] для выявления их графических характеристик. Фактически сумматорная матрица точечных весов (рис. 3.20) представляет собой матрицу наложений всех символьных прототипов на одну генерируемую матрицу, в результате чего на ней вырисовывается изображение символа при запрограммированном выявлении градиентного закрашивания на табличном поле значений, если значение точечных весов перевешивает определенные пороговые значения. Для выявления характеристик образов и балансирования распределения точечных вероятностей, была дополнительно реализована модификация метода потенциалов для сумматорной матрицы, названная интерполяционным сглаживанием точечных весов первого уровня. Этот метод позволяет заполнять пустоты между эталонными начертаниями и выявлять доминирующие и устранять рецессивные группы точек путем естественного отбора масс. Это позволяет на основе мини набора эталонной БД автоматически получить картину сумматорной матрицы, соответствующую количеству макси набора БД, что резко повышает эффективность и уменьшает надобность ввода огромных массивов эталонных экземпляров в БД.

Таблица 3.2 Статистические данные одного из образов.

Колличество точек матрицы	**192**
Сумма весов матрицы	**1,171**
Сумма вероятностей матрицы	**29.28**
Среднее значение матрицы	**6.10**
Медиана весов матрицы	**2.00**
Вероятности:	**Кол. точек**
[0.00]	**88**
диапазон (0.00-0.25]	**59**
диапазон (0.25-0.50]	**32**
диапазон (0.50-0.75]	**5**
диапазон (0.75-1.00)	**6**
[1.00]	**2**

0.99	0.60	0.27	0.18	0.09	0.03	0.02	0.01	0.00	0.00	0.00	0.00	2.19
0.67	0.63	0.44	0.31	0.13	0.11	0.08	0.01	0.01	0.00	0.00	0.00	2.39
0.25	0.58	0.56	0.39	0.42	0.25	0.09	0.09	0.02	0.01	0.00	0.00	2.65
0.09	0.19	0.52	0.63	0.47	0.41	0.33	0.19	0.12	0.04	0.01	0.00	2.98
0.01	0.06	0.20	0.37	0.55	0.56	0.48	0.40	0.32	0.17	0.05	0.02	3.18
0.00	0.01	0.04	0.17	0.30	0.43	0.58	0.57	0.44	0.38	0.22	0.09	3.21
0.00	0.00	0.01	0.03	0.07	0.19	0.34	0.47	0.59	0.58	0.45	0.32	3.04
0.00	0.00	0.00	0.00	0.01	0.03	0.08	0.23	0.41	0.56	0.72	0.67	2.70
0.04	0.03	0.02	0.01	0.00	0.00	0.01	0.04	0.09	0.26	0.59	1.15	2.23
0.14	0.18	0.08	0.01	0.00	0.00	0.00	0.00	0.01	0.05	0.43	1.36	2.24
0.41	0.18	0.07	0.01	0.00	0.00	0.00	0.00	0.00	0.01	0.38	1.40	2.46
0.66	0.32	0.09	0.01	0.00	0.00	0.00	0.00	0.00	0.03	0.42	1.36	2.87
0.79	0.54	0.12	0.01	0.00	0.00	0.00	0.00	0.02	0.12	0.51	1.27	3.38
0.80	0.69	0.26	0.08	0.01	0.01	0.04	0.10	0.21	0.40	0.93	0.71	4.23
0.61	0.58	0.77	0.44	0.40	0.38	0.42	0.62	0.79	0.88	0.60	0.32	6.80
0.29	0.28	0.50	0.93	0.98	1.00	0.95	0.73	0.50	0.30	0.05	0.15	6.64
5.72	4.86	3.94	3.57	3.42	3.39	3.42	3.45	3.53	3.78	5.35	8.79	

Рис. 3.20 Сумматорная матрица точечных вероятностей, сглаженная методом потенциалов.

На основе полученных матриц были построены 3-D колонная диаграмма точечных вероятностей (рис. 3.21) и графики, названные “дактилоскопическими паттернами”,

представляющие собой диаграммы пересечения v-zond и h-zond с точками узора на сумматорной матрице (рис. 3.22). Из диаграмм видна информационно-графическая специфика картин, несущая информационную уникальность паттерна определенного

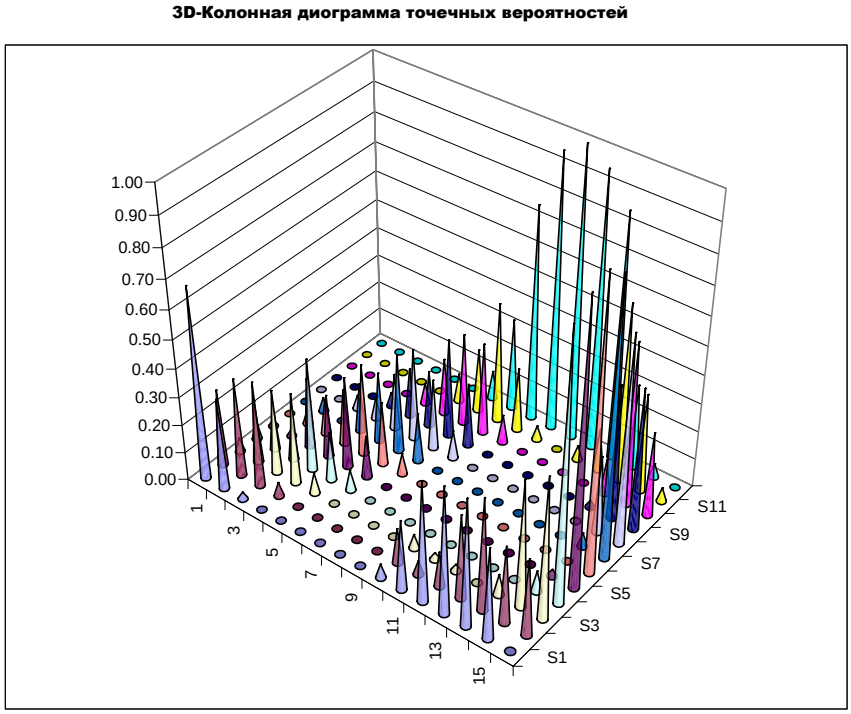


Рис. 3.21 3-D колонная диограмма точечных вероятностей грузинской буквы ”ა”.

символа, подобно дактилоскопическому рисунку и в отличие от метода v/z-zond более точно сможет идентифицировать объекты [2].

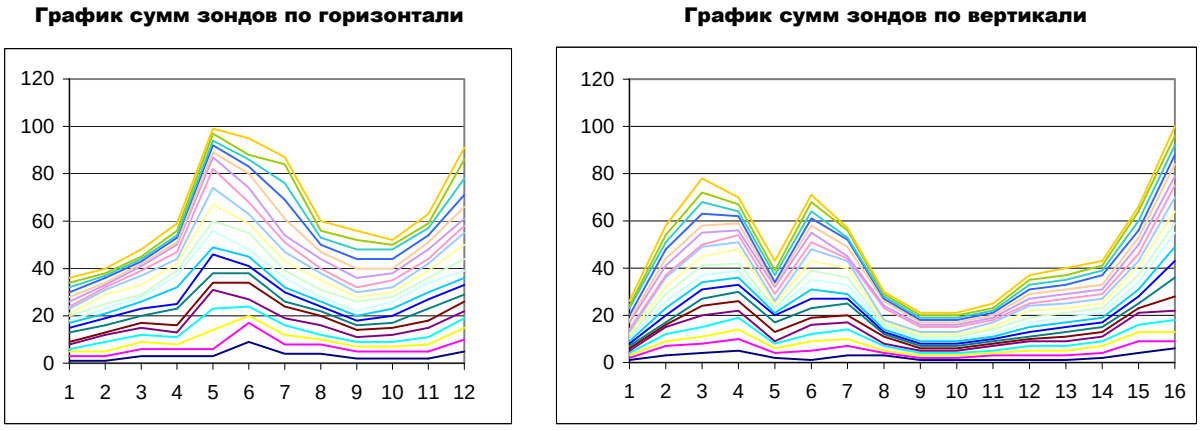


Рис. 3.22 Графики сумм зондов по вертикали и горизонтали.

На примере цифры “5”, буквы латинского “G” и буквы “ბ” грузинского алфавитов (Рис. 3.23 – 3.28), приведем матрицы эталонных элементов БД, таблицы статистических подсчетов, графики зондов по горизонтали и вертикали и диаграммы точечных вероятностей.

Рис. 3.23 Эталонные элементы цифры “5” введенные вручную и перенесенные в БД.

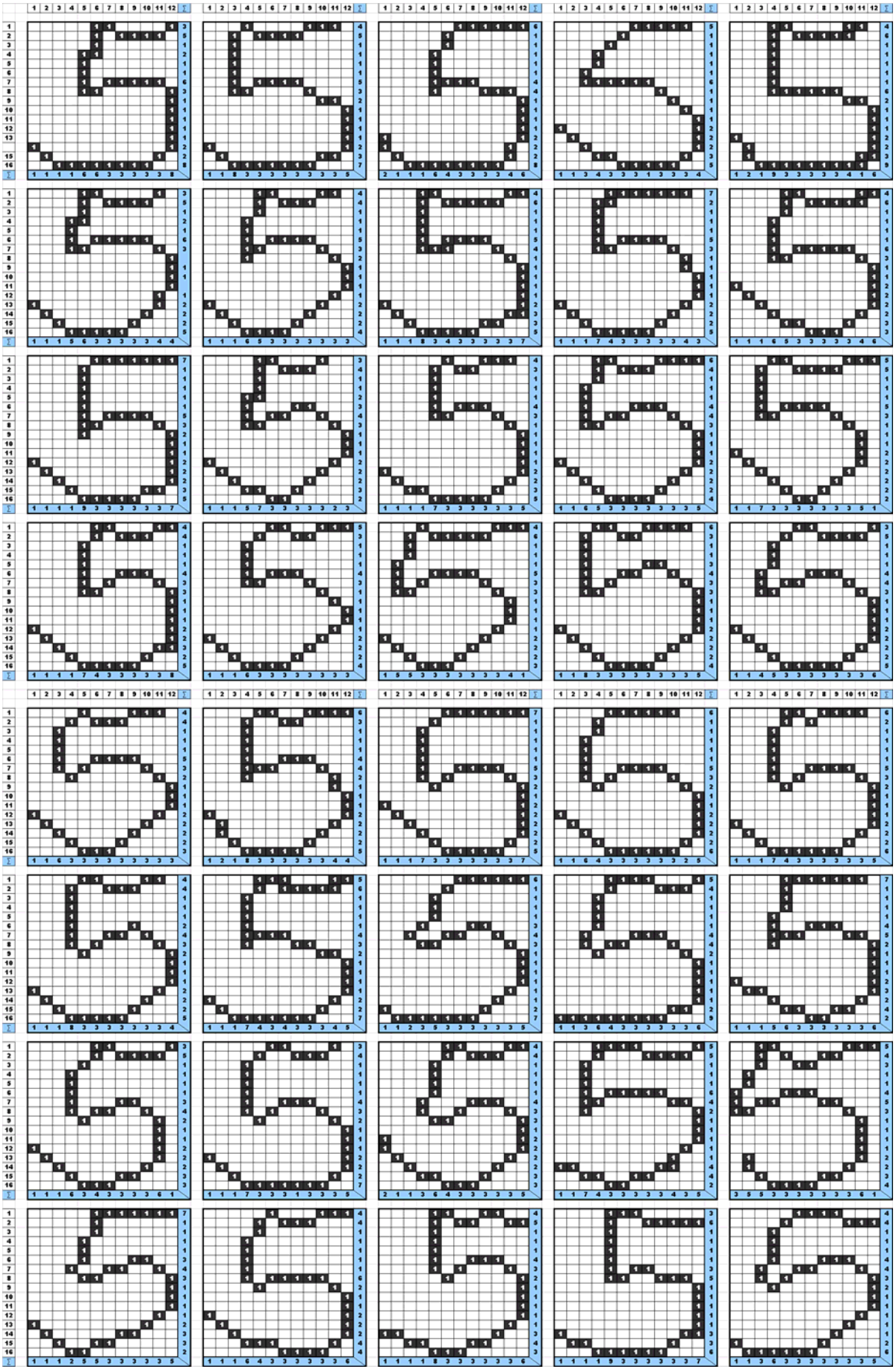


Рис. 3.24 Вероятностно-статистический анализ и дактилоскопический разбор цифры “5”.

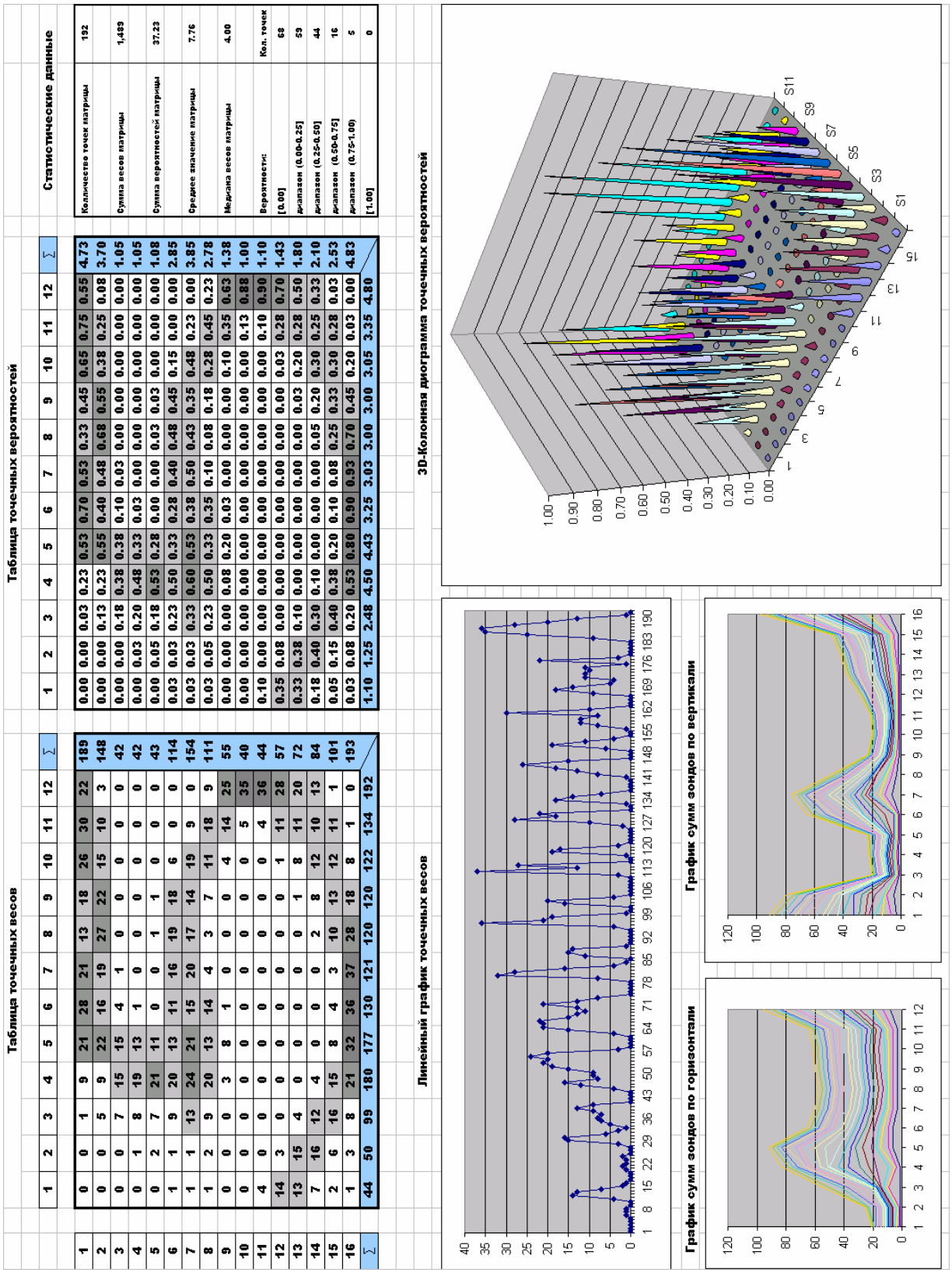


Рис. 3.25 Эталонные элементы буквы “G” введенные вручную и перенесенные в БД.

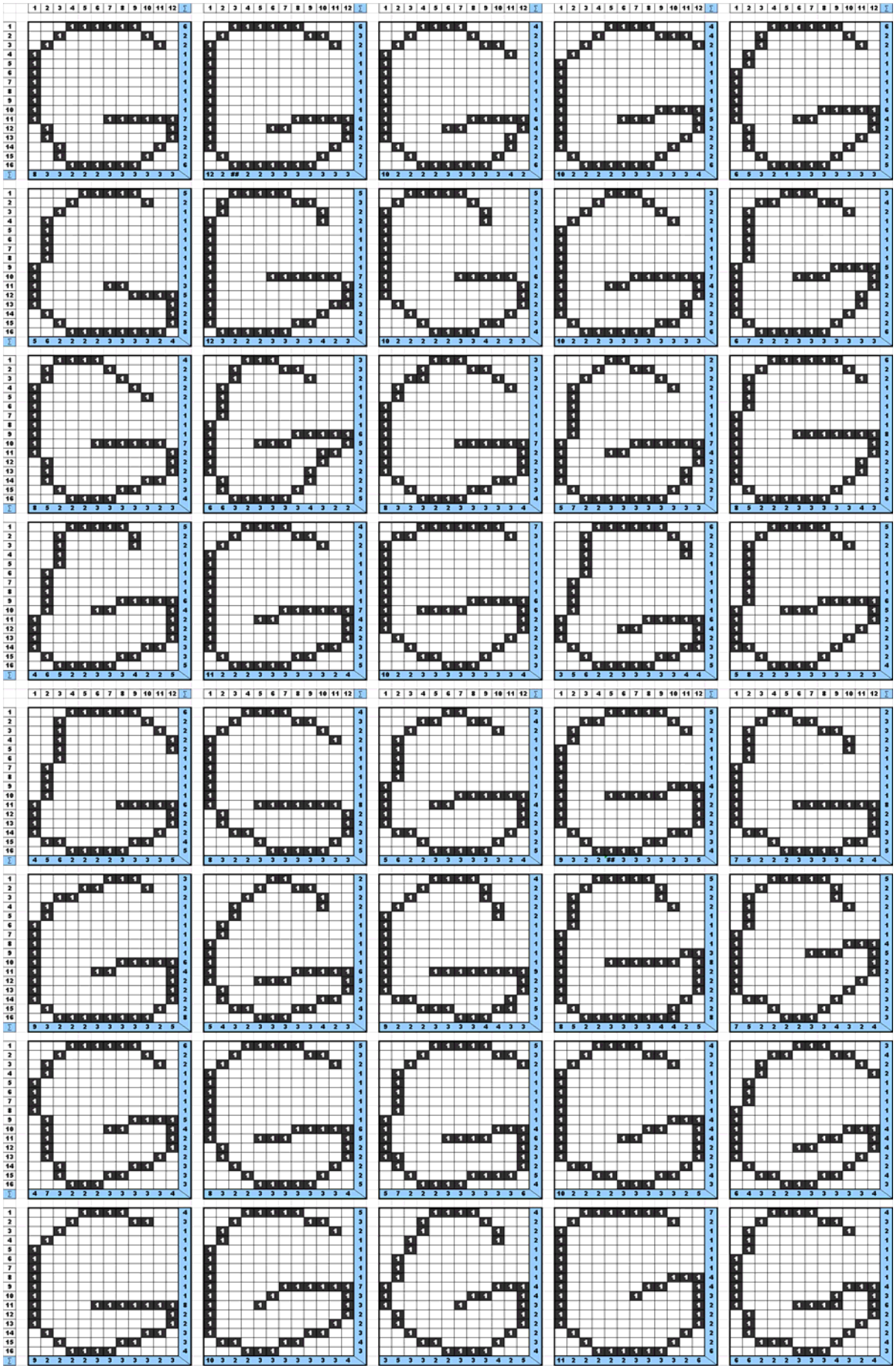


Рис. 3.26 Вероятностно-статистический анализ и дактилоскопический разбор буквы “G”.

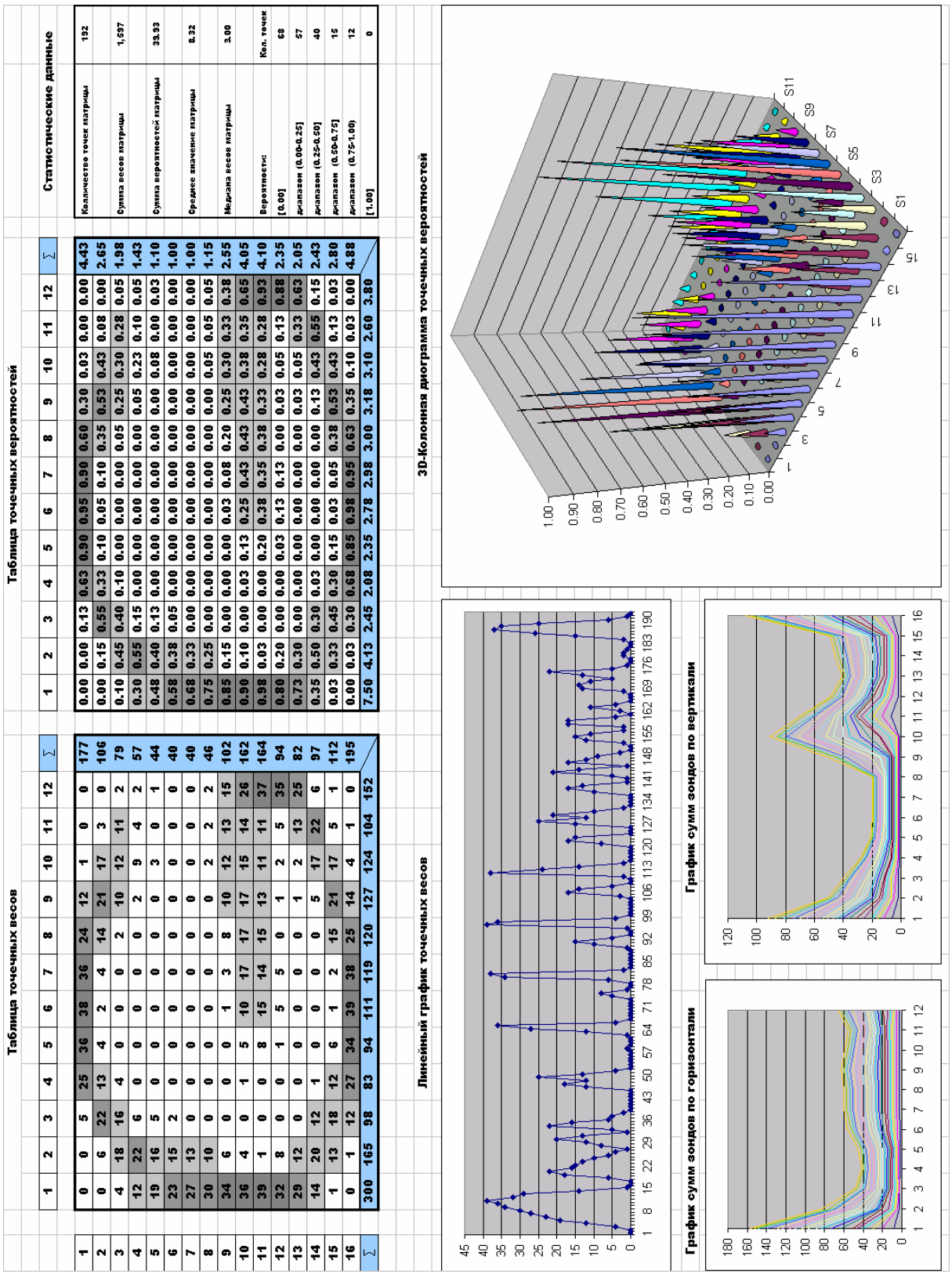


Рис. 3.27 Эталонные элементы буквы “ъ” введенные вручную и перенесенные в БД.

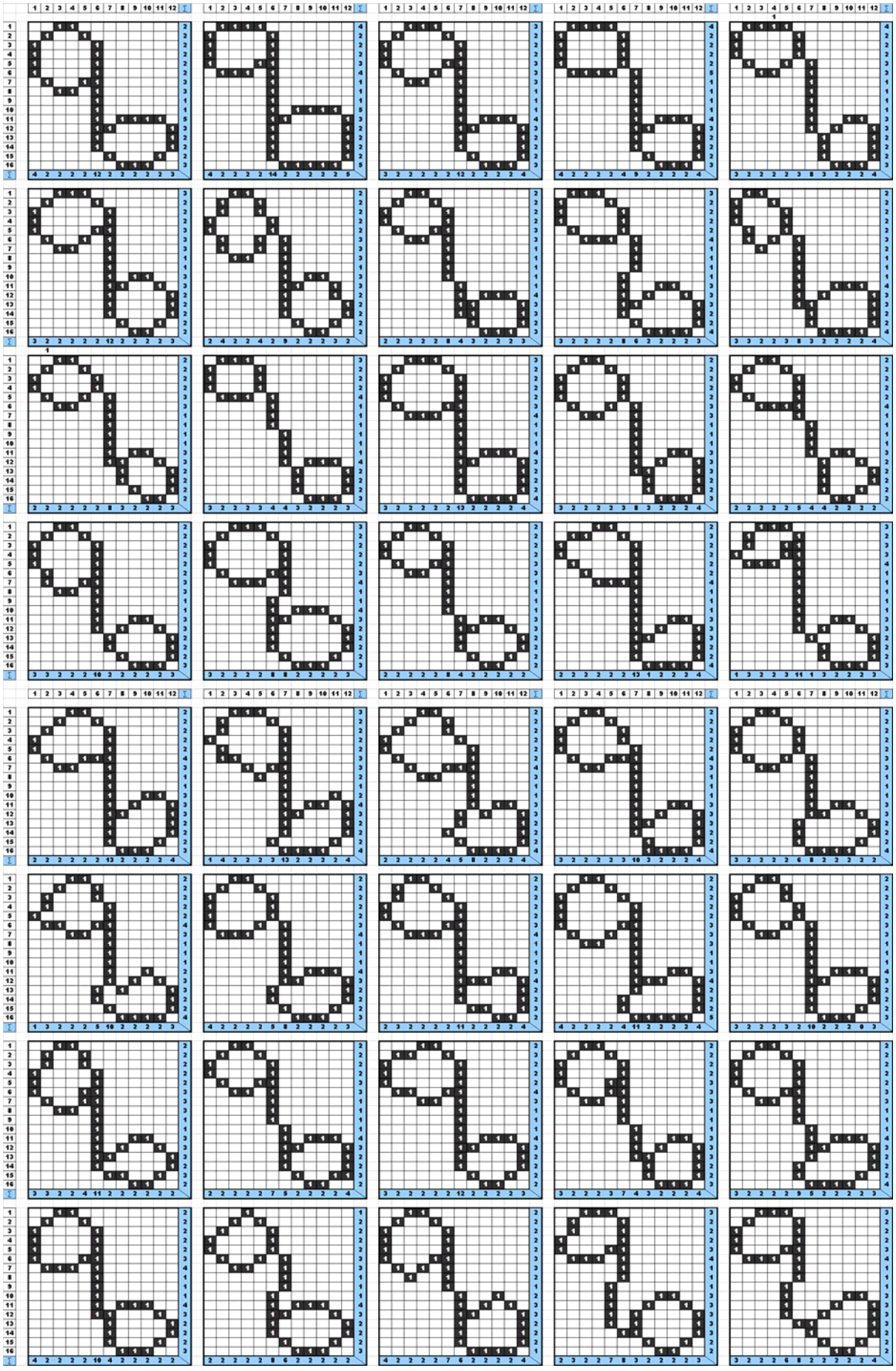
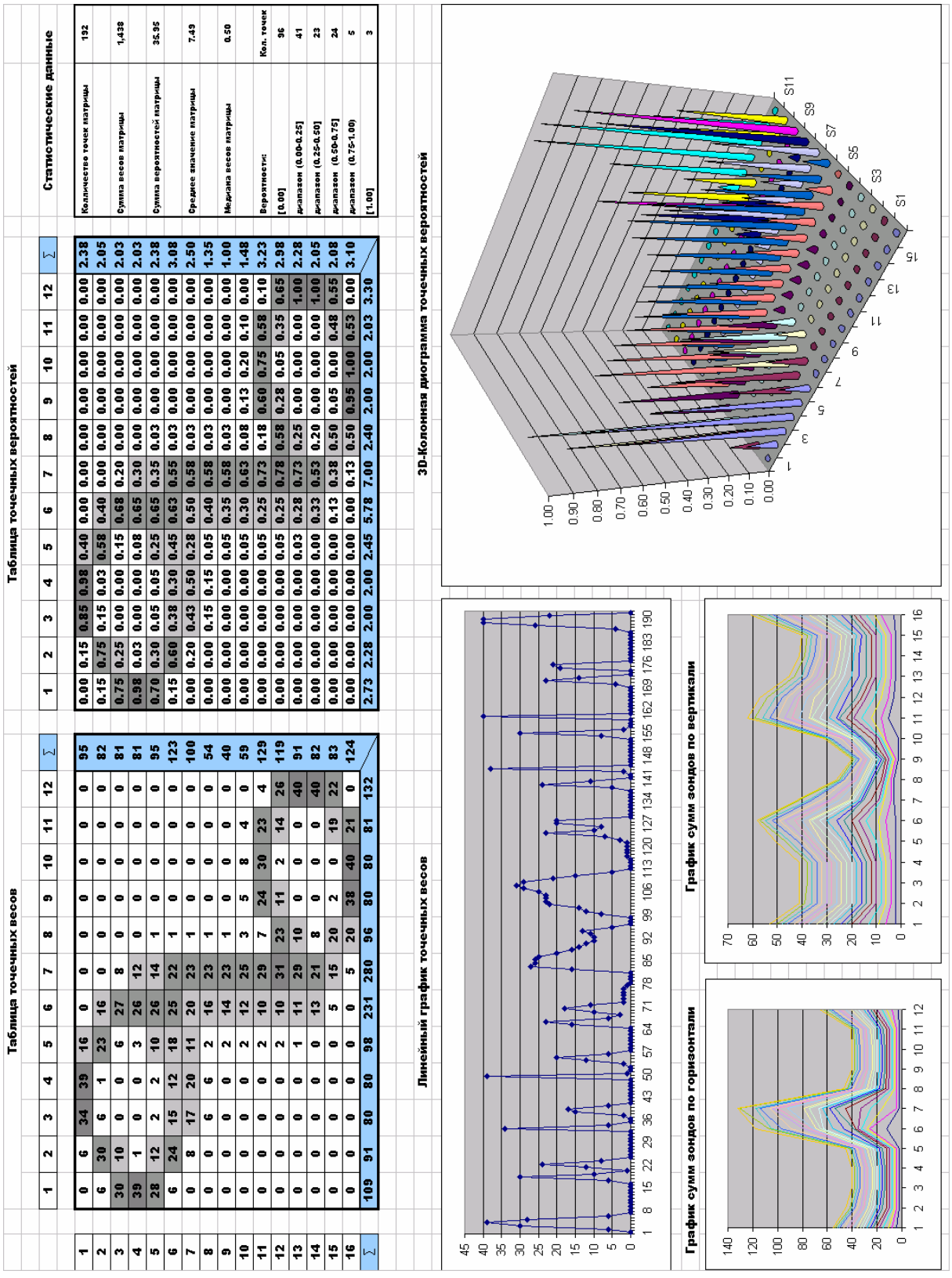


Рис. 3.28 Вероятностно-статистический анализ и дактилоскопический разбор буквы “ъ”.



1.62	1.20	0.70	0.42	0.24	0.12	0.07	0.03	0.01	0.00	0.00	0.00	4.42
1.27	1.17	0.88	0.62	0.35	0.25	0.15	0.05	0.02	0.00	0.00	0.00	4.79
0.67	0.99	1.02	0.83	0.76	0.50	0.27	0.20	0.08	0.03	0.01	0.00	5.36
0.30	0.47	0.89	1.06	0.92	0.80	0.64	0.42	0.28	0.12	0.04	0.02	5.96
0.08	0.19	0.45	0.73	0.96	1.01	0.92	0.78	0.60	0.36	0.16	0.09	6.33
0.02	0.05	0.15	0.36	0.59	0.81	1.01	1.02	0.87	0.73	0.47	0.29	6.38
0.00	0.01	0.04	0.10	0.21	0.42	0.66	0.87	1.05	1.05	0.89	0.74	6.04
0.01	0.01	0.01	0.02	0.05	0.12	0.24	0.48	0.76	1.02	1.29	1.31	5.32
0.10	0.08	0.06	0.02	0.01	0.02	0.06	0.14	0.29	0.61	1.24	1.90	4.53
0.30	0.30	0.14	0.03	0.00	0.00	0.01	0.02	0.07	0.27	1.08	2.14	4.36
0.70	0.43	0.18	0.04	0.00	0.00	0.00	0.00	0.01	0.18	1.01	2.19	4.74
1.11	0.67	0.25	0.04	0.00	0.00	0.00	0.00	0.02	0.21	1.05	2.15	5.51
1.39	1.01	0.37	0.08	0.01	0.01	0.02	0.05	0.13	0.44	1.17	2.05	6.72
1.44	1.25	0.67	0.33	0.18	0.16	0.23	0.36	0.57	0.91	1.53	1.43	9.06
1.13	1.11	1.24	0.93	0.87	0.85	0.90	1.08	1.26	1.35	1.07	0.80	12.59
0.78	0.28	0.50	0.93	0.98	1.00	0.95	0.73	0.50	0.30	0.05	0.47	7.45
10.94	9.21	7.53	6.56	6.14	6.06	6.12	6.24	6.51	7.59	11.05	15.60	

Рис. 3.29 Интерполирование второгоу уровня для сумматорной матрицы точечных вероятностей.

Интерполяционным сглаживанием точечных весов второго уровня (рис. 3.29) дает более сбалансированное распределение весов матрицы в результате чего можно идентифицировать опознаваемые объекты с более высокой вероятностью или меньшей погрешностью ошибки.

Может быть также использована рекуррентная интерполяция второго уровня на основе использования сглаживания матрицы первого уровня. Если сравнить вертикальные и горизонтальные графики зондов различных символов, например, латинскую букву “M” с грузинской буквой ““ჟ”, то можно заметить их графическую диверсификацию (рис. 3.30). Путем наложений и сравнения графиков и вероятностей можно определить удаленность поступившего на вход изображения с каждой из сумматорной матрицей различных образов. Дистанция разрыва будет определять приближенность или удаленность между введенным изображением и символами из БД.

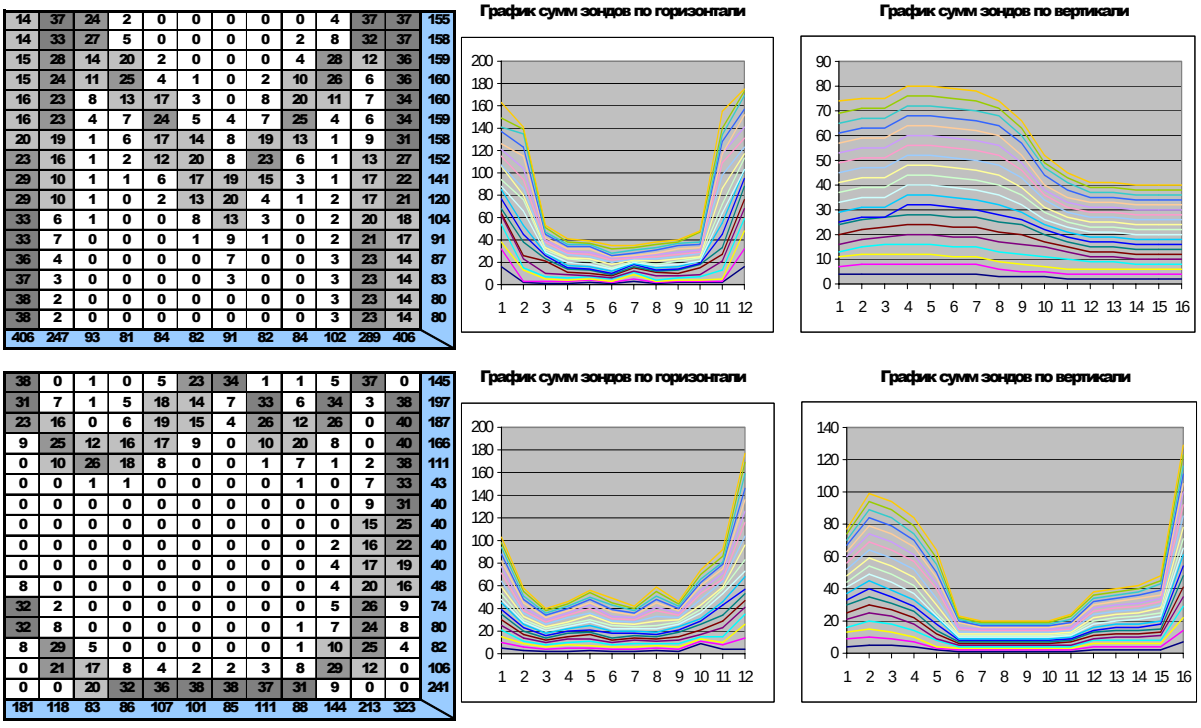


Рис. 3.30 “Дактилоскопическое” сравнения паттернов графиков.

Можно использовать другую диаграмму - линейный график точечных весов, который представляет собой комбинированный график (рис. 3.31) распределения точечных весов и несет более собирательную информацию распределения точек на поле рецепторов, по сравнению с горизонтальным и вертикальными графиками зондов [8].

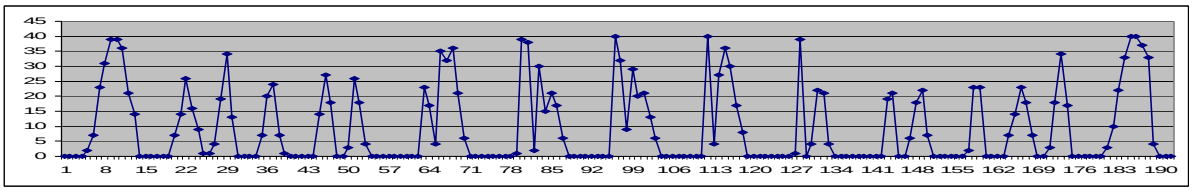
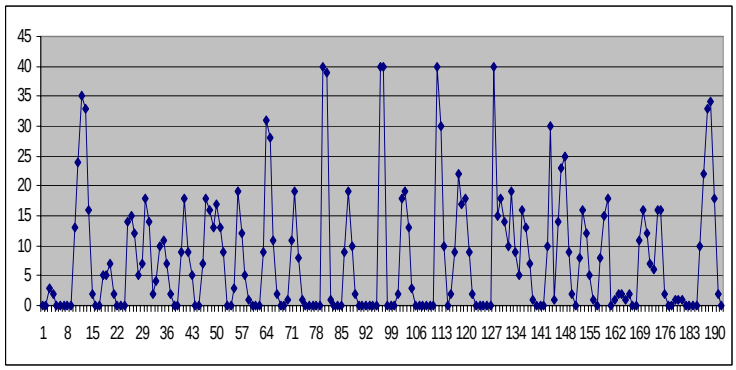


Рис. 3.31 Линейный график точечных весов.

Как видно, можно заметить отличия в паттерне начертаний различных символов. Действительно, узор имеет “дактилоскопические” отличительные признаки даже при визуальном сравнении линейных графиков точечных весов грузинской буквы “ბ” и латинской буквы “V” (рис. 3.32).

0.00	0.00	0.10	0.33	0.70	0.98	1.00	0.75	0.38	0.03	0.00	0.00	4.25
0.00	0.13	0.25	0.43	0.28	0.03	0.00	0.25	0.45	0.35	0.03	0.00	2.18
0.08	0.13	0.28	0.33	0.05	0.00	0.00	0.00	0.35	0.58	0.05	0.03	1.85
0.05	0.18	0.18	0.23	0.00	0.00	0.00	0.05	0.25	0.63	0.05	0.03	1.63
0.00	0.05	0.05	0.00	0.00	0.00	0.05	0.23	0.48	0.23	0.03	0.03	1.13
0.00	0.00	0.00	0.00	0.03	0.23	0.45	0.55	0.23	0.05	0.05	0.00	1.58
0.00	0.00	0.00	0.08	0.28	0.48	0.48	0.43	0.13	0.00	0.00	0.00	1.85
0.00	0.00	0.23	0.48	0.48	0.25	0.33	0.45	0.40	0.20	0.00	0.00	2.80
0.00	0.35	0.45	0.30	0.20	0.05	0.08	0.23	0.33	0.40	0.28	0.00	2.65
0.33	0.38	0.23	0.13	0.03	0.00	0.00	0.05	0.18	0.30	0.40	0.25	2.25
0.60	0.30	0.13	0.03	0.00	0.00	0.00	0.00	0.03	0.13	0.30	0.55	2.05
0.88	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.18	0.83	2.03
0.83	0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15	0.85	2.00
0.40	0.45	0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.40	0.45	2.08
0.05	0.35	0.45	0.23	0.00	0.00	0.00	0.00	0.25	0.38	0.40	0.05	2.15
0.00	0.05	0.40	0.78	1.00	1.00	1.00	1.00	0.75	0.45	0.05	0.00	6.48
3.20	2.65	2.90	3.30	3.03	3.00	3.38	3.98	4.18	3.93	2.35	3.05	



0.73	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.95	1.98
0.55	0.28	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.50	0.40	1.98
0.38	0.35	0.25	0.00	0.00	0.00	0.00	0.00	0.08	0.20	0.45	0.28	1.98
0.33	0.33	0.30	0.05	0.00	0.00	0.00	0.00	0.08	0.33	0.43	0.18	2.00
0.10	0.43	0.38	0.10	0.00	0.00	0.00	0.05	0.05	0.50	0.35	0.05	2.00
0.03	0.43	0.30	0.25	0.00	0.00	0.00	0.05	0.13	0.55	0.28	0.00	2.00
0.03	0.33	0.25	0.38	0.03	0.00	0.00	0.10	0.23	0.53	0.15	0.00	2.00
0.00	0.20	0.33	0.38	0.10	0.00	0.00	0.10	0.38	0.40	0.13	0.00	2.00
0.00	0.08	0.33	0.35	0.23	0.03	0.03	0.18	0.50	0.25	0.05	0.00	2.00
0.00	0.03	0.33	0.35	0.25	0.05	0.05	0.25	0.43	0.28	0.00	0.00	2.00
0.00	0.00	0.28	0.28	0.38	0.08	0.10	0.28	0.48	0.15	0.00	0.00	2.00
0.00	0.00	0.18	0.33	0.33	0.15	0.15	0.38	0.48	0.03	0.00	0.00	2.00
0.00	0.00	0.05	0.33	0.40	0.20	0.25	0.45	0.30	0.03	0.00	0.00	2.00
0.00	0.00	0.00	0.18	0.43	0.43	0.38	0.48	0.13	0.00	0.00	0.00	2.00
0.00	0.00	0.00	0.08	0.40	0.50	0.53	0.40	0.05	0.00	0.00	0.00	1.95
0.00	0.00	0.00	0.00	0.08	0.43	0.48	0.05	0.00	0.00	0.00	0.00	1.03
2.13	2.53	3.10	3.03	2.60	1.85	1.95	2.75	3.28	3.33	2.53	1.85	

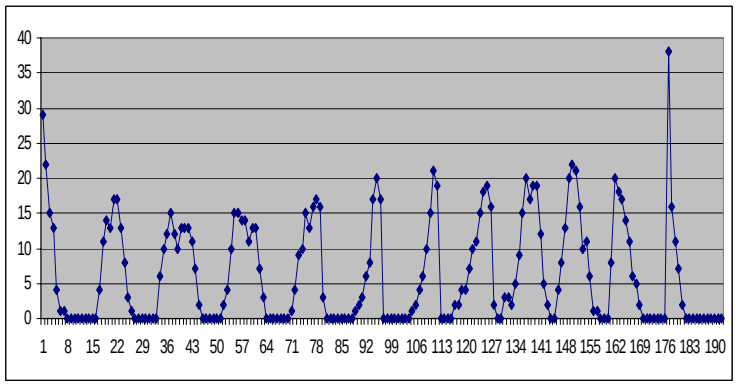


Рис. 3.32 Сравнение линейных гарфиков точечных весов.

Для определения удаленности между двумя БД представляющими определенные образы, например латинской буквой “V” и грузинской буквой “ბ”, можно использовать промежуточную матрицу разницы по точечным весам, на основе которой строится

комбинированный линейный график сравнений (рис. 3.33). Чем меньше разница или суммарное значение, тем ближе друг к другу изображения из БД и наоборот.

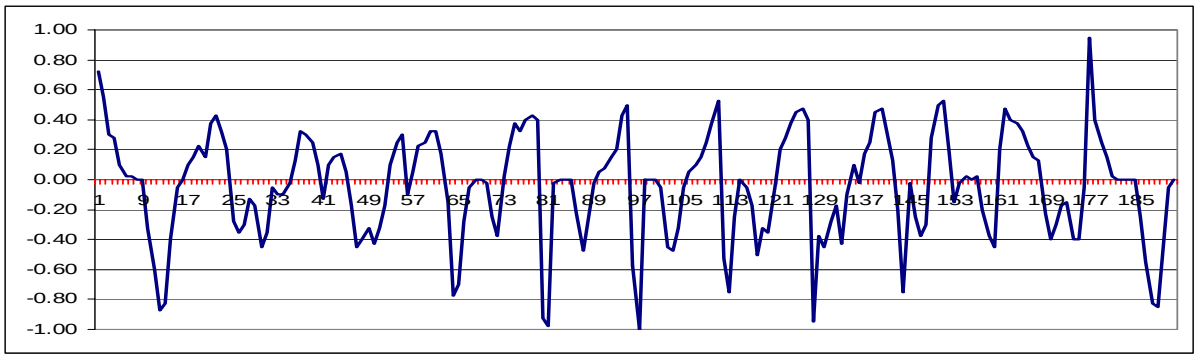


Рис. 3.33 График сравнения несимилярных образов “V” с “ბ”.

Этот механизм можно использовать для определения дисперсии между поступившим на вход изображением и сумматорной матрицей для сравнения с эталонной БД. Если сравнить симилярные образы из БД, например цифру “3” и грузинскую букву “ჱ”, то получим следующую диаграмму.

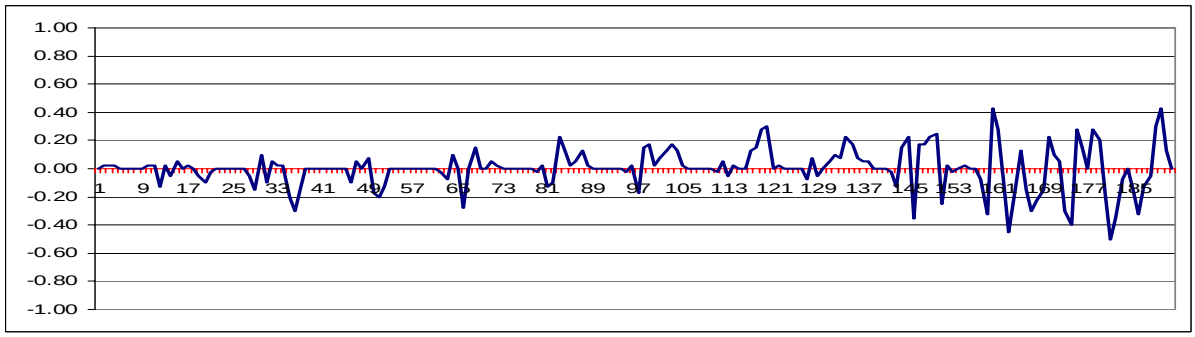


Рис. 3.34 График сравнения симилярных образов цифры “3” и грузинской буквы “ჱ”.

Из сравнения графиков на рис. 3.33 и 3.34 видно, что чем выше амплитуды графиков, тем менее симилярные образы из БД и наоборот, чем ниже амплитуда на графике, тем более визуально приближенны образы. Приведем эти же графики в модульном представлении, где подсчитывается сумма по модулю значений этих разниц (рис. 3.35 и 3.36).

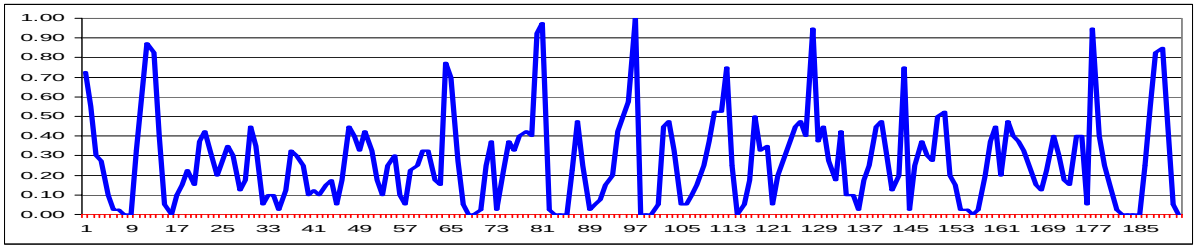


Рис. 3.35 Модульный график сравнения несимилярных образов “V” с “ბ”.

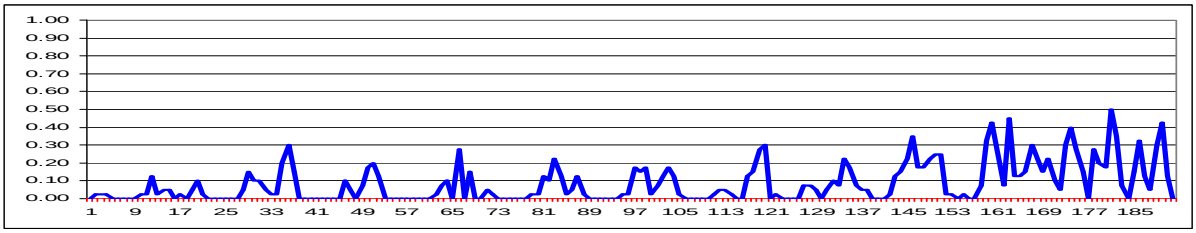


Рис. 3.36 Модульный график сравнения симилярных образов цифры “3” и буквы “З”.

Приведенные новые схемы и графики, хоть и приближают нас к разрешению задачи распознавания, но не разрешают ее, в виду того, что не учитывают некоторые факторы, связанные с категоризацией точек на гибридных матрицах. Поэтому, для улучшения распознавания, была введена новая методика сравнения точек рецепторной матрицы с сумматорной эталонной БД (рис. 3.37), заключающаяся в точечном анализе и подсчете различных кластеров точечных масс [7].

	1	1	1	1	1			1	1	1		0.03	0.03	0.53	0.93	0.60	0.08	0.03	0.30	0.88	0.78	0.13	0.00
1						1	1				1	0.03	0.50	0.45	0.10	0.43	0.50	0.18	0.70	0.15	0.23	0.68	0.13
1						1					1	0.38	0.48	0.15	0.00	0.28	0.53	0.38	0.53	0.05	0.05	0.38	0.60
1						1					1	0.65	0.33	0.03	0.00	0.18	0.43	0.58	0.30	0.03	0.00	0.23	0.78
1						1					1	0.85	0.15	0.00	0.00	0.05	0.48	0.63	0.20	0.00	0.00	0.13	0.90
	1				1						1	0.95	0.05	0.00	0.00	0.00	0.35	0.65	0.10	0.00	0.00	0.00	1.00
		1			1					1		0.98	0.05	0.00	0.00	0.00	0.18	0.43	0.03	0.00	0.00	0.03	0.98
			1	1				1	1			0.95	0.05	0.00	0.00	0.00	0.13	0.25	0.03	0.00	0.00	0.05	0.95
							1					0.93	0.08	0.00	0.00	0.00	0.08	0.18	0.03	0.00	0.00	0.08	0.93
								1	1	1		0.85	0.15	0.00	0.00	0.00	0.05	0.10	0.00	0.00	0.00	0.15	0.85
											1	0.78	0.23	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.23	0.78
1											1	0.58	0.40	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.40	0.58
1											1	0.40	0.53	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.53	0.68
	1										1	0.30	0.48	0.23	0.00	0.00	0.00	0.00	0.00	0.00	0.23	0.48	0.30
		1								1		0.10	0.43	0.43	0.05	0.00	0.00	0.00	0.00	0.05	0.43	0.43	0.10
			1	1	1	1	1	1	1			0.00	0.18	0.68	0.20	0.00	0.00	0.00	0.00	0.20	0.70	0.15	0.00

Рис. 3.37 Пример точек для буквы “З” матрицы рецепторов и сумматорной БД буквы “О”.

Введем новые термины, представляющие разные категории матриц (рис. 3.38):

- Конгруэнтные матрицы
- Диверсивные матрицы
- Пассивные матрицы

Конгруэнтные матрицы получаются в результате операции умножения значений точек матрицы, поступившего на вход изображения, с соответствующими по позиции значениями точек из сумматорных матриц вероятностей.

Диверсивные матрицы получаются следующим образом: если разница между значением точки из матрицы входного изображения и соответствующим значением точки из сумматорной матрицы равно 1, то в диверсивной матрице соответствующая по позиции

точка принимает значение равное 1, в противном случае – записывается значение равное 0.

Пассивные матрицы получаются в результате операции вычитания значений точек сумматорных матриц вероятностей с соответствующими по позиции значениями точек из конгруэнтных матриц.

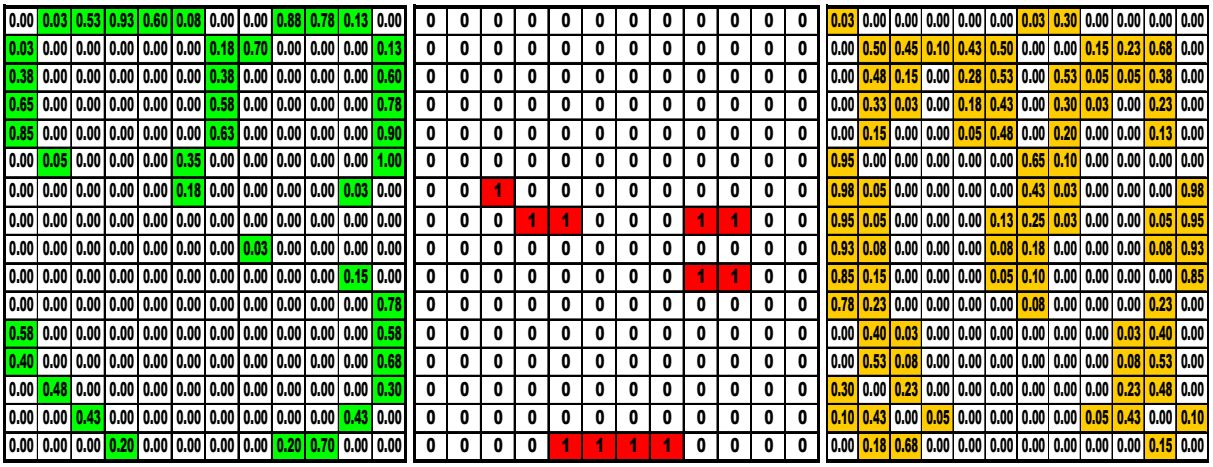


Рис. 3.38 Пример выделения конгруэнтных, диверсивных и пассивных точек для буквы “Ȝ” при сравнивании с эталонной БД буквы “ᵞ”.

Симбиозные матрицы получаются в результате перенесения значений точек (не равных нулю) из конгруэнтных, диверсивных и пассивных матриц в одну общую матрицу. При этом в симбиозной матрице, эти точки не будут накладываться по координатам друг на друга в силу построения.

0.03	2.03	2.53	2.93	2.60	2.08	0.03	0.30	2.88	2.78	2.13	0.00
2.03	0.50	0.45	0.10	0.43	0.50	2.18	2.70	0.15	0.23	0.68	2.13
2.38	0.48	0.15	0.00	0.28	0.53	2.38	0.53	0.05	0.05	0.38	2.60
2.65	0.33	0.03	0.00	0.18	0.43	2.58	0.30	0.03	0.00	0.23	2.78
2.85	0.15	0.00	0.00	0.05	0.48	2.63	0.20	0.00	0.00	0.13	2.90
0.95	2.05	0.00	0.00	0.00	2.35	0.65	0.10	0.00	0.00	0.00	3.00
0.98	0.05	2.00	0.00	0.00	2.18	0.43	0.03	0.00	0.00	2.03	0.98
0.95	0.05	0.00	2.00	2.00	0.13	0.25	0.03	2.00	2.00	0.05	0.95
0.93	0.08	0.00	0.00	0.00	0.08	0.18	2.03	0.00	0.00	0.08	0.93
0.85	0.15	0.00	0.00	0.00	0.05	0.10	0.00	2.00	2.00	2.15	0.85
0.78	0.23	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.23	2.78
2.58	0.40	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.40	2.58
2.40	0.53	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.53	2.68
0.30	2.48	0.23	0.00	0.00	0.00	0.00	0.00	0.00	0.23	0.48	2.30
0.10	0.43	2.43	0.05	0.00	0.00	0.00	0.00	0.05	0.43	2.43	0.10
0.00	0.18	0.68	2.20	2.00	2.00	2.00	2.00	2.20	2.70	0.15	0.00

Рис. 3.39 Пример получения симбиозной матрицы для буквы “Ȝ” из БД буквы “ᵞ”.

Нулевые точки – это точки симбиозной матрицы, значения которых равны нулю. Симбиозная матрица, является итоговой матрицей, которая содержит в себе информацию, связанную с различными категориями масс – конгруэнтных, диверсивных, пассивных и нулевых (остаточных масс с нулевыми значениями). На основании глубокого анализа

проделанной работы были выявлены закономерности для распознавания образов.

Гипотеза симбиоза масс

Чем больше конгруэнтных и нулевых точек в симбиозной матрице и меньше диверсивных и пассивных точек, тем больше приближение изображения матрицы рецепторов к искомой сумматорной матрице образа т.е. реального изображения.

Используя градацию деления точек сумматорной матрицы на различные категории, и получая симбиозную матрицу, можно выявлять по определенным признакам принадлежность того или иного образа к определенному классу символов, применяя нейроматричную самоорганизацию сети.

Алгоритм нейросетевой организации матриц для распознавания образов.

Фактически задача заключается в том, что на основе 40 введенных конфигураций в БД для каждого символа выявить их характерные признаки и распознать изображение представленного на опознавание символа. Задача распадается на следующие шаги:

1. Загрузка БД содержащую 2760 введенных эталонных символов из MS Excel файла в программу;
2. Введение изображения произвольного символа для последующего опознавания на матрице рецепторов (12x16) и сохранение в формате .BMP (16 Color);
3. Считывание введенного изображения для распознавания из файла в программу;
4. Преобразование изображения из двоичного представления в табличный массив 0 и 1, программы где 0 – определяет незакрашенные, а 1 – закрашенные ячейки массива;
5. Подсчет конгруэнтной промежуточной нейронной матрицы 2-го слоя;
6. Подсчет диверсивной промежуточной нейронной матрицы 2-го слоя;
7. Подсчет пассивной промежуточной нейронной матрицы 2-го слоя;
8. Подсчет итоговой симбиозной нейронной матрицы 3-го слоя, основанной на характеристиках матриц первого и второго слоев;
9. Выявление характеристик симбиозных матриц и вычисление итоговых значений на выходах;
10. Выявление наибольшего значения из выходов симбиозных матриц, определение его кода и ассоциирование с поступившим на вход изображением;
11. Корректировка сумматорной матрицы 1-го уровня (с учителем или без учителя) в случае высокой погрешности;
12. Переход к шагу 2, пока величина погрешности не будет удовлетворяющей.

Блок схема соответствующего алгоритма будет иметь следующий вид:

Искусственная самоорганизующаяся нейро-матричная сеть (ИСНМС), построена по следующей схеме (рис. 3.40): на вход нейронной сети подается растровое изображение символа, вход можно представить, как матрицу рецепторов, а изображение может быть представлено в виде последовательности 0 и 1, где 0 - незакрашенные ячейки, а 1 - закрашенные ячейки. Информация из матрицы рецепторов подается, и трансформируются на второй уровень - класс нейронных матриц представителей и критериев оценок для каждого образа базы данных, после этого подсчитываются промежуточные веса и применяются критерии отбора элементов, которые также прописаны в матричном виде - третий уровень.

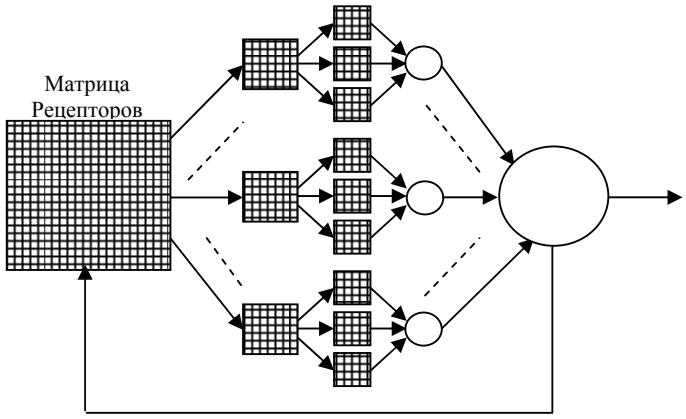


Рис. 3.40 Условный граф матричной нейронной самоорганизующейся сети.

Следующим этапом является взвешивание полученных результатов с помощью вычисления целевой функции каждой матричной схемы, учитывая различные критерии. После вычисления всех весов на последнем этапе просматриваются полученные результаты, и наивысшему значению будет соответствовать наиболее приближенный образ. В созданной схеме используется самоорганизация сети, т.е. после определения образа происходит перенастройка весов различных матриц критериев в цикле, таким образом, схема не является статической, а является динамически подстраиваемой и эффективно сходящейся.

На основе построенной принципиально новой искусственной нейронной схемы с независимыми нейронами и применения вероятностно-статистических анализов были получены высокие результаты распознавания. Данный граф показывает структурную схему построенной искусственной самоорганизующейся нейронной сети (ИСНС) для распознавания графических образов.

Ниже приводится более детализированная блок-схема нейронной сети для распознавания образов, на которой описаны основные этапы, включающие загрузку БД, ввод изображения для опознавания, получение сумматорных, конгруэнтных, диверсивных, пассивных и симбиозных матриц для каждого образа и подсчет целевой функции для выявления наивысшего приближенного значения определяющего код распознавания образа.

3.2.2 Блок схема нейронной сети для распознавания образов.

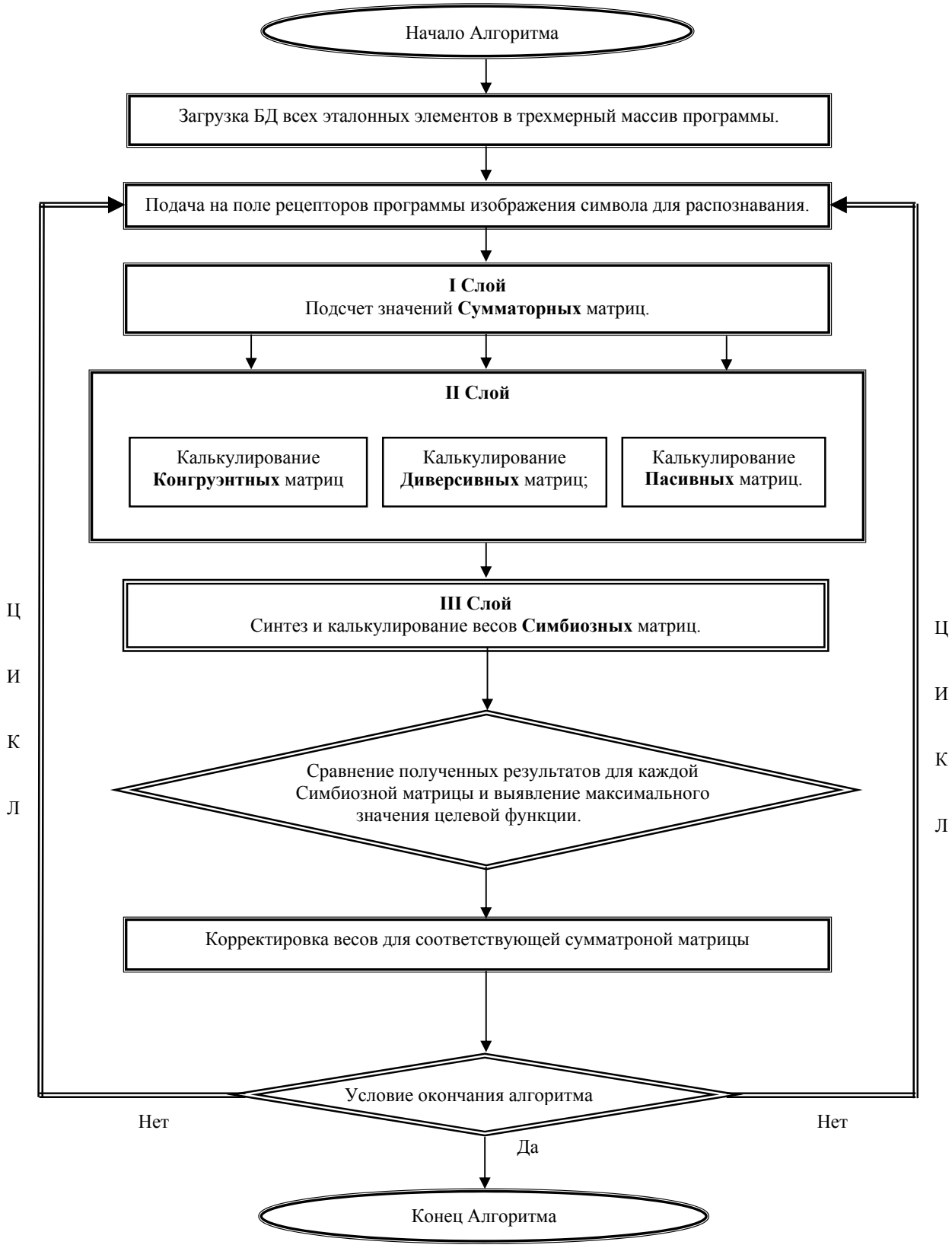


Рис. 3.41 Блок схема нейронной сети для РО.

3.3 Полученные результаты.

В результате проделанной работы была смоделирована программная модель (рис. 3.42), реализующая распознавание образов (арабских цифр, букв грузинского и латинского алфавитов) с помощью мульти нейронной, параллельной искусственной сети, применяя вероятностно-статистического растровый анализа визуальных образов. Программа реализует перевод графической информации изображения символа в матричный вид, загружает внешнюю эталонную БД сумматорных таблиц в программу и вычисляет промежуточные: конгруэнтные, диверсивные и пассивные таблицы для каждого символа. Далее производится подсчет симбиозных матриц каждого символа, анализ взвешивание весов и получение результирующих значений из которых выявляется наибольшее значение, определяющее код принадлежности к тому или иному символу (рис. 3.43).

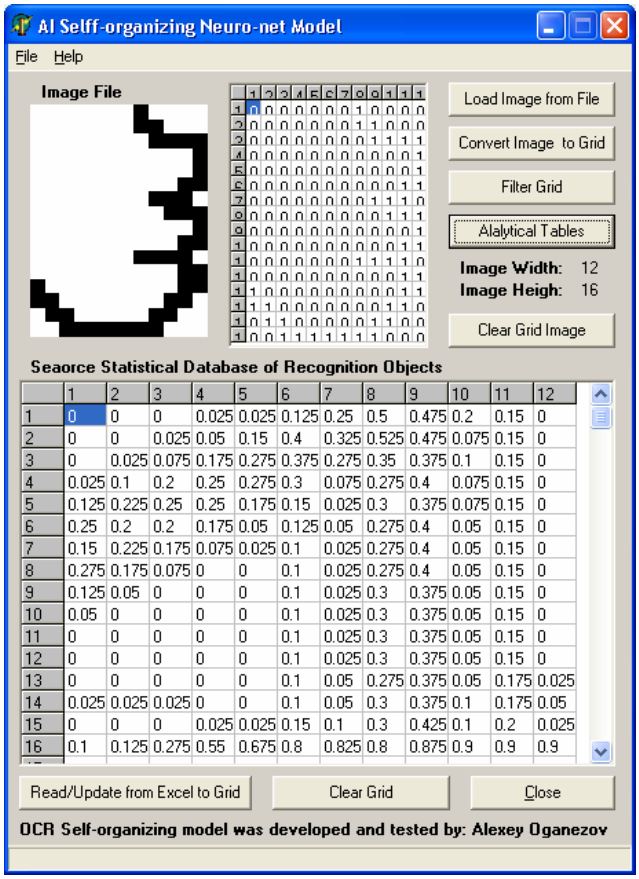


Рис. 3.42 Фрагмент программы

Программа содержит новый подход в методологии с применением вероятностно-статистической ИСНС. В результате апробации программы были получены высокие результаты распознавания образов. Программа смоделирована для рукописного почерка и легко адаптируется под конкретный шрифт в целях улучшения распознавания. Конечно

также как и в любых современных программах в данной модели существуют определенные ограничения, но в целом она осуществляет высокоточное распознавание для довольно таки большого класса символов. При выборе конкретного шрифта для распознавания и нейро подстройки модели под этот шрифт с помощью откалиброванной начальной БД, программа существенно повышает процент распознавания и достигает высокие результаты распознавания 92% и выше в зависимости от качества подстройки, что позволяет применить это программное модульное ядро для современного использования в других программ.

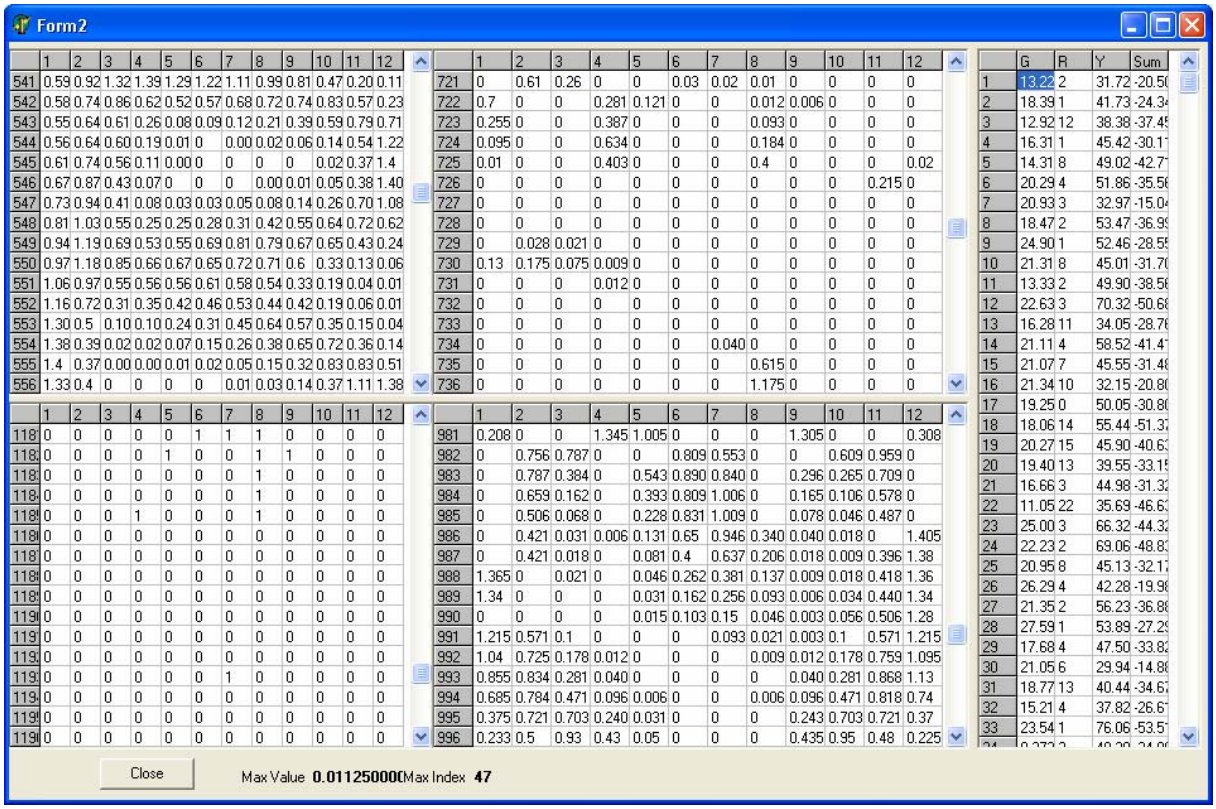


Рис. 3.43 Пример анализа и подсчета сумматорной, конгруэнтной, диверсивной и пасивной матриц подсчетов.

При лабораторной апробации программы, использовался ПК Intel Celeron 2.4 GHz, 256 MB RAM, OS: Microsoft Windows XP SP2. Программа сравнивает, анализирует и изменяет весовые характеристики весовых матриц, повышая эффективность и точность модели. Кроме этого распознавание символов осуществляется довольно оперативно за миллисекунды, не используя при этом возможность распараллеливания, несмотря на структурное блочное разделение процессов. В целом созданная программа является уникальной и эффективной для распознавания любых символов, которая основывается на рукописной эталонной БД с внедренной вероятностно-статистической ИСНС и механизмами аналитических подсчетов и подстроек мульти матричных слоев.

ЗАКЛЮЧЕНИЕ.

На основе проведенных исследований предложен принципиально новый метод распознавания образов с помощью неординарного алгоритма и программной реализации, осуществляющей функционирование новой разделенной структуры искусственной нейронной сети (ИНС), использующей нейро-матричную реализацию (конгруентных, диверсивных, пассивных и нейтральных массивов растровых точек), с применением вероятностно-статистического анализа для классификации обрабатываемых изображений.

Основные результаты диссертационной работы следующие:

1. На основе анализа и обработки специальной литературы и ресурсов Интернет были проработаны, выявлены и применены принципы и знания из теории нейробиологии, нейроинформатики, кибернетики, искусственного интеллекта, искусственных нейронных сетей и нейросетвой самоорганизации, персептронов, кластеризации и категоризации объектов, анализа сцен, машинного распознавания образов, теории множеств, дискретной математики, кодирования/декодирования, структуры графических форматов файлов, теории вероятности и математической статистики, алгоритмических построений блок-схем и модульного объектно-ориентированного программирования (ООП) на языке Object Pascal в оперативной среде разработки RAD под Borland Delphi.
2. Для исследования проблемы была введена большая база данных (БД) из эталонных образов, состоящая из разновидностей 10 цифр, 26 букв латинского и 33 букв грузинского алфавитов, что в сумме составило 69 различных символьных знаков, а с учетом количества 40 экземпляров на каждый символ, составило 69x40 = 2 760 введенных эталонных образов. В силу дискретизации растрового изображения, умножая эту сумму на разрешение 12x16 матрицы рецепторов, имеем 529 920 введенных точек (закрашенных и не закрашенных). В работе описывается матрица рецепторов, проводятся подсчеты статистических показателей, сумм весов и вероятностей матрицы, усредненных значений и медиан весов матрицы, а также анализ количества точек, имеющих вероятности появления активных (закрашенных) точек из диапазонов [0.00], (0.00-0.25], (0.25-0.50], (0.50-0.75], (0.75-1.00], [1.00], для выявления характеристик каждого символьного образа. Также построены графики зависимостей вертикальных и горизонтальных составляющих зондов изображения, и линейный график точечных весов, построены 3D-колонные диаграммы точечных вероятностей для каждого образа. Были созданы усредненные контуры каждого символа на основе сумматорных матриц первого слоя ИНС, применена интерполяции

первого и второго уровня для сглаживания и балансирования весовых коэффициентов сумматорной матрицы, построены промежуточные нейро-массивы второго слоя: конгруэнтные, диверсивные и нейтральные, а также симбиозные матрицы третьего слоя. После формирования трехслойной структуры, была построена искусственная самообучающаяся нейро-матричная сеть ИСНМС обработки изображения для распознавания графических образов.

3. В работе создана принципиально новая искусственная нейро-матричная сеть (ИНМС) без учителя, которая обрабатывает и модифицирует эталонную базу образов – сумматорную матрицу при сравнении с поданными на вход тестовыми изображениями. Подстройка ведется посимвольно, меняя весовые коэффициентные характеристики двухмерных массивов из БД, и применяет новые методы адаптации для повышения эффективности распознавания образов. Введены новые понятия диверсивных, конгруэнтных и нейтральных масс. Основой нового метода является вероятностно-статистический фундамент оценок в качестве весовых коэффициентов для многослойной сети и применение новой идеи смешивания и оценок пиксельных масс для выявления дискретной принадлежности к классифицируемым группам.
4. Разработаны новые алгоритмы обучения ИНС, блок-схема, проектные модули и программное приложение соответствующее требованной методике. При запуске программы изображение на входе для распознавания декодируется из двоичного машинного представления формата BMP в матричный вид для последующих преобразований и калькуляций. Программа извлекает необходимые эталонные характеристики из табличного архива в формате MS Excel и преобразовывает в трехмерный массив БД для последующего анализа и обработки с поступившим изображением в формате BMP. Также программа производит глубокий анализ незнакомого образа и синтетические наложения, пропуская его через ИСНС и сравнивая с каждым эталонным матричным слепком из БД, выявляя закономерности, и в результате идентификации определяет его принадлежность к конкретному классу символов.
5. В результате проведенной работы предложена новая методика распознавания символьных образов, в частности для цифр, букв латинского и грузинского алфавитов, которая может быть применена к любым символьным знакам или алфавитам на основе нейросетевого самообучения и новых методов сравнения диверсивных, конгруэнтных и нейтральных растровых масс ИНС. Была успешно достигнута задача распознавания образов с помощью подстройки ИНС, невзирая на низкое количество эталонных наборов и разрешение воспринимающей матрицы

рецепторов, по сравнению с условно стандартными и минимальными требованиями. Основным достижением программы является получение беспрецедентного распознавания символов для сравнительно компактных и разделенных множеств на первом же цикле, что является удачным подбором созданной структуры ИНС и принципиально новой возможностью для обработки визуальных объектов. Для взаимно диффузионных множеств, программа ведет самоанализ и реорганизацию весовых коэффициентов, за несколько этапов, производит самоподстройку и хорошее разделение гомоморфных групп для улучшения детектирования объектов. Основной ценностью нового метода является также то, что вопреки минимальному количеству введенных эталонов базы набора для одного символа графической информации, получается качественный результат. Таким образом, вместо к примеру номинального сочетания 120 (точек по горизонтали) х 160 (точек по вертикали) х 1 000 (эталонов для каждого символа) х 69 (цифр + букв латинского и грузинского алфавитов) = 1 324 800 000 (1.32 миллиарда закрашенных и незакрашенных точек) в данном методе используется 12 х 16 х 40 х 69 = 529 920 (черных и белых точек), что на 1 324 270 080 точек меньше или в 25 000 раз уменьшает трудоемкость задачи для ввода информации и представляет высокую эффективность при получении необходимых результатов не только с точки зрения введенных затрат, но также уменьшает машинное время для обработки вычислительных процессов. Существенное уменьшение базы данных набора и повышения эффективности распознавания объектов также является практической ценностью нового метода данной работы.

6. Результаты экспериментальных исследований достижения высоких результатов классификации при минимально возможном разрешении матрицы рецепторов и количестве представителей образов из каждого набора символьных элементов базы данных, подтверждают эффективность принципиально нового разработанного метода весовых настроек синаптических массивов для обучения ИНС, имеют практическую ценность и может быть внедрено в различных областях науки и практики.
7. Смоделирована, протестирована и отлажена компьютерная программа с принципиально новой комплексной структурной модели ИНС, использующей вероятно-статистический анализ нейро-матричных массивов графической информации с мульти разделенной сетью нейронов, для параллельных процессов классификации образов.

ЛИТЕРАТУРА.

1. Кочладзе З.Ю., Оганезов А.Л. Использование двухслойной нейронной модели в процессах распознавания образов. Научно-технический журнал «Энергия» №4 (36) I. Тбилиси, 2005. стр. 78.
2. ოგანეზოვი ა. ინფორმაციული დამუშავების ალგათობითი და ნეირონული სქემების გამოყენება ქართული შრიფტების ამოცნობის პროცესისათვის, “მეცნიერება და ტექნოლოგიები”, №1-3, თბილისი, 2006, გვ. 30
3. Оганезов А.Л., Кочладзе З.Ю. Задача распознавания инженерных 2D чертежей и преобразования в электронные 3D стереометрические объекты. Научно-технический журнал «Энергия» №4 (36) I. Тбилиси, 2005. стр. 73.
4. ოგანეზოვი ვ., ოგანეზოვი ა. არატრადიციულ ენერგეტიკაში გამოთვლითი ტექნიკის გამოყენება. “ბუნება და ენერგეტიკა”, თბილისი, 1998, გვ. 137
5. Кочладзе З.Ю., Оганезов А.Л. Об одном возможном подходе к проблеме распознавания плоских фигур, Университетский журнал. Тбилиси, 2006.
6. A. Oganezov. Creation of Energy Efficiency Fund for Georgia. Proceedings of the Second International Energy Conference in Armenia, Part 2, Yerevan, 2001
7. Oganezov A. “Neuro-matrix object recognition model”, “Bulletin of the Georgian National Academy of Sciences”, Tbilisi, V. 173 №3, 2006.
8. Oganezov A. “Flat figures recognition based on fingerprinting signs in neural nets”, “Bulletin of the Georgian National Academy of Sciences”, Tbilisi, V. 174 №1, 2006.
9. ვერულავა ო., ხუროძე რ. ამომცნობი სისტემების თეორიის საფუძვლები, “ტექნიკური უნივერსიტეტი”, თბილისი, 2001.
10. ტყემალაძე ნ., სწავლების სახეთა გამოცნობის ავტომატიზირებული სისტემა, თბილისი, 2000.
11. რამიშვილი გ., ძეპირსიტყვიერი დიალოგი ადამიანსა და მანქანას შორის, თბილისი, “მეცნიერება”, 1985
12. Чавчанидзе В.В., Аналитическое решение задачи формирования понятий и распознавания образов, Сборник докладов, т. 61, №1, Тбилиси, 1971.
13. Кочладзе З.Ю., Кларджеишвили И.Д., Маисурадзе А.И. О некоторых проблемах создания автоматизированной системы концептуального управления сложными производственными дискретно-непрерывными процессами. Материалы международной объединенной конференции по ИИ, Тбилиси, 1976.
14. Чигвиндзе О.Д., Чоговадзе Р.А. Выполнение групповой операции умножения в ассоциативном параллельном процессе с расширенными внутримодульными функциями, Труды Грузинского политехнического института, №8 (181), Тбилиси, 1975.Э. Хант. Искусственный интеллект, Москва, «Мир», 1978.
15. Верулава О.Г., Хуродзе Р.А., Чоговадзе Р.А. Формирование меры сходства между множествами. Академия наук Грузии, Институт систем управления, Сборник докладов международной научной конференции, Проблемы управления и энергетики PCPE-2004”, №8, Тбилиси, 2004, стр. 189-193.
16. Чоговадзе Р.А. Синтез искусственных нейронных сетей для процессов распознавания образов. Труды Грузинского технического университета, №4 (454), Тбилиси, 2004.
17. Квиташвили А.А., Тхинвалели Р.Г., Канделаки М.К., Гелдиашвили Н.И. Однородная система распознавания трехмерных объектов и речевых сигналов, Материалы международной объединенной конференции по ИИ, Тбилиси, 1976.

18. Дундуа А.Г., Кецба М.И., Бахтадзе Н.М., Кадагишвили Л.Г., Иремашвили Н.И.. Эксперименты по концептуальным механизмам установочного поведения, Материалы VI всесоюзного симпозиума по кибернетике, Тбилиси, 1972.
19. Купарадзе М.Р. Структура и функции нервной системы, Москва, Сборник докладов, 1962
20. Чумбуридзе И.Ш., Нуцубидзе М.А., Микашавидзе Г.Н., Береникашвили Н.Н. К вопросу об организации нейронных сетей, способных к обучению классификации событий. Тезисы докладов V Всесоюзной конференции по нейрокибернетике, РГУ Ростов на-Дону, 1973.
21. Аркадьев А. Г., Браверманн Э. М. Обучение машины классификации объектов, Москва, “Наука”, 1971.
22. Бахтадзе Н.М., Кадагишвили Л.Г., Кецба М.Н. Методы определения параметров феномена бессознательной памяти, Материалы международной объединенной конференции по ИИ, Тбилиси, 1976.
23. Мегрелидзе К.Р. Основные проблемы социологии мышления, “Мецниэреба”, Тбилиси, 1973.
24. Chavchanidze V., Self-organization of Discrete System, 6th International Congress of Cybernetics, p. 187, “Namur”, 1972.
25. Дуда Р., Харт П. Распознавание образов и анализ сцен, Москва, «Мир», 1976.
26. Цыганков В.Д. Нейрокомпьютер и его применение. Москва, СолСистем, 1993.
27. Соколов Е.Н., Вайткявичус Г.Г. Нейроинтеллект от нейрона к нейрокомпьютеру. Москва, Наука, 1989.
28. Амосова Н.М. Нейрокомпьютеры и интеллектуальные роботы. Киев, 1991.
29. Чернухин Ю.В. Микропроцессорное и нейрокомпьютерное управление адаптивными мобильными роботами. Таганрог, 1993.
30. Чернухин Ю.В. Нейропроцессоры. Таганрог, 1994.
31. Труды третьего международного симпозиума «Интеллектуальные системы», Псков, 1998.
32. Венда В.Ф. Системы гибридного интеллекта. Москва, Машиностроение, 1990
33. Волгин Л.И. Комплиментарная алгебра нейросетей. Таллин, АО KLTK, 1993.
34. Кунфе Ф. Взаимодействие робота с внешней средой. Москва, «Мир», 1985.
35. Горбань А.Н. Обучение нейронных сетей. Москва, “ПараГраф”, 1990.
36. Сотник С. Л, Конспект лекций по курсу "основы проектирования систем искусственного интеллекта", Москва, 1998.
37. Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика, Москва, 1992.
38. Горбань А.Н., Методы Нейроинформатики, сборник научных трудов, Красноярск КГТУ, 1998
39. Kimura T., Shima T. Synapse weight accuracy of analog neuro chip // Proceedings of International Joint Conference on Neural Networks. – Japan, Nagoya, October 25-29, 1993. – Vol.1. – P. 891-894.
40. Anguita D., Ridella S., Rovetta S. Limiting the effects of weight errors in feed forward networks using interval arithmetic // Proceedings of International Conference on Neural Networks (ICNN’96). – USA, Washington, June 3-6, 1996. – Vol.1. – P. 414-417.
41. Edwards P., Murray A. Modeling weight- and input-noise in MLP learning // Proceedings Of International Conference on Neural Networks (ICNN’96). – USA, Washington, June 3-6, 1996. – Vol.1. – P. 78-83.
42. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука, 1996. 276 с.

43. Senashova Masha Yu., Gorban Alexander N., and Wunsch Donald, “Back-Propagation of Accuracy”// Proc. IEEE/INNS International Conference of Neural Networks, Houston, IEEE, 1997, pp.1998-2001
44. Горбань А.Н., Сенашова М.Ю. Погрешности в нейронных сетях// Вычислительный центр СО РАН в г.Красноярске. Красноярск, 1997. 38 с., библиогр. 8 назв. (Рукопись деп. в ВИНИТИ, 25.07.97, №2509-В97)
45. Отчет о НИР «Принципы представления знаний в системах управления роботов». Под руководством Н.М. Амосова, Киев, 1985.
46. Сенашова М.Ю. Глава 6. Погрешности в нейронных сетях // Нейроинформатика / А.Н. Горбань, В.Л. Дунин-Барковский, А.Н. Кирдин, Е.М. Миркес и др. Новосибирск: Наука, 1998.
47. McCulloch W. W., Pitts W. 1943. A logical calculus of the ideas imminent in nervous activity. Bulletin of Mathematical Biophysics 5:115-33. (Русский перевод: Мак-Каллок У. С., Питтс У. Логическое исчисление идей, относящихся к нервной деятельности. Автоматы. – М.: ИЛ. – 1956.
48. Minsky M. L, Papert S. 1969. Perseptrons. Cambridge, MA: MIT Press. (Русский перевод: Минский М. Л., Пейперт С. Персептроны. – М: Мир. – 1971.)
49. Qarey M. R., Johnson D. S. 1979. Computers and intrac-tality. New York: W.H. Freeman.
50. Rosenblatt F. 1962. Principles of Neurodinamics. New York: Spartan Books. (Русский перевод: Розенблатт Ф. Принципы нейродинамики. – М: Мир. – 1965.)
51. Widrow B. 1961. The speed of adaptation in adaptive control system, paper *1933-61. American Rocket Society Guidance Control and Navigation Conference.
52. Jannson T., Karagaleff C., Stoll K. M. 1986. Photo-refractive LiNbO3 as a storage medium for high-density optical neural networks. 1986 Optical Society of America Annual Meeting.
53. Burr, D. J. 1987. Experiments with a connectionist text reader. In Proceedings of the First International on Neural Networks, eds. M. Caudill and C. Butler, vol. 4, pp. 717–24. San Diego, CA: SOS Printing.
54. Cottrell, G. W., Munro P., and Zipser D., 1987. Image compressions by backpropagation: An example of extensional programming. Advances in cognitive science (vol.3). Norwood, NJ: Ablex.
55. Gallant S. I., 1988. Connectionist expert system. Communications of the ACM 31:152–69.
56. Parker, D. B. 1982. Learning-logic. Invention Report, s. 81–64, File 1. Office of Technology Licensing, Stanford University.
57. Rumelhart D. E., Hinton G. E., and Williams R. J. 1986. Learning internal representations by error propagation. In Parallel distributed processing, vol. 1, pp. 318–62. Cambridg, MA: MIT Press.
58. Sejnowski T. J., and Rosenberg C. R. 1987. Parallel Networks that learn to pronounce English text. Complex Systems 3:145–68.
59. Werbos P. J. 1974. Beyond regression: New tools for prediction and analysis in the behavioral sciences. Masters thesis. Harvard University.
60. Grossberg S. 1974. Classical and instrumental learning by neural networks. Progress in theoretical biology, vol. 3, pp. 51–141. New York: Academic Press.
61. Hebb D. 0. 1961. Organization of behavior. New York: Science Edition.
62. Kohonen T. 1984. Self-organization and associative memory. Series in Information Sciences, vol. 8. Berlin: Springer Verlag.
63. Tank D. W., Horfield J. J. 1986. Simple «neural» optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit. Circuits and Systems IEEE Transactions on CAS-33(5):533-41.

64. Van den Bout D. E. and Miller T. K. 1988. A traveling salesman objective function that works. Proceedings of the IEEE International Conference on Neural Networks, vol. 2, pp. 299-304. San Diego, CA: SOS Printing.
65. Almeida L. B. 1987. Neural computers. Proceedings of NATO ARW on Neural Computers, Dusseldorf. Heidelberg: Springer-Verlag.
66. Burr D. J. 1987. Experiments with a connectionist text reader. In Proceedings of the IEEE First International Conferense on Neural Networks, eds. M. Caudill and C.Butler, vol. 4, pp. 717-24. San Diego, CA: SOS Printing.
67. Pineda F. J. 1988. Generalization of backpropagation to recurrent and higher order networks. In Neural information processing systems, ed. Dana Z. Anderson, pp. 602-11. New York: American Institute of Physics.
68. Stornetta W. S., Huberman B. A. 1987. An improved three-layer, backpropagation algorithm. In Proceedings of the IEEE First International Conference on Neural Networks, eds. M. Caudill and C. Butler. San Diego, CA: SOS Printing.
69. Wasserman P. D. 1988a. Combined backpropagation/Cauchy machine. Proceedings of the International Neural Network Society. New York: Pergamon Press.
70. Wasserman P. D. 1988b. Experiments in translating Chinese characters using backpropagation. Proceedings of the Thirty-Third IEEE Computer Society International Conference. Washington, D. C.: Computer Society Press of the IEEE.
71. DeSieno D. 1988. Adding a conscience to competitive learning Proceedings of the IEEE International Conference on Neural Networks, pp. 117-24. San Diego, CA: SOS Printing.
72. Qrossberg S. 1969. Some networks that can learn, remember and reproduce any number of complicated space-time patterns. Journal of Mathematics and Mechanics, 19:53-91.
73. Hecht-Nielsen R. 1987a. Counterpropagation networks. In Proceedings of the IEEE First International Conference on Neural Networks, eds. M. Caudill and C. Butler, vol. 2, pp. 19-32. San Diego, CA: SOS Printing.
74. Geman S., Geman D. 1984. Stohastic relaxation, Gibbs distribution and Bayesian restoration of images. IEEE Transactions on Pattern Analysis and Machine Intelligence 6:721-41.
75. Hinton G. E., Sejnowski T. J. 1986. Learning and relearning in Boltzmann machines. In Parallel distributed processing, vol. 1, p. 282-317. Cambridge, MA: MIT Press.
76. Metropolis N., Rosenbluth A. W-.Rosenbluth M. N., Teller A. N., Teller E. 1953. Equations of state calculations by fast computing machines. Journal of Chemistry and Physics. 21:1087-91.
77. Szu H., Hartley R. 1987. Fast Simulated annealing. Physics Letters. 1222(3,4): 157-62.
78. Kosko B. (1987a). Bi-directional associative memories. IEEE Transactions on Systems, Man and Cybernetics 18(1):49-60.
79. Lippman R. P. 1987. An introduction to computing with neural nets. IEEE Transactions on Acosufics, Speech and Signal Processing, April, pp. 4-22.
80. Psaltis D., Wagner K., Brady D. 1987 Learning in optical neural computers. In Proceedings of IEEE First International Conference on Neural Networks, M. Caudill and C. Butler. San Diego, CA:SOS Printing.
81. Perlovsky L. Neural Networks and Intellect. Using Mode-based concepts. 2001. Oxford University Press

ИНТЕРНЕТ РЕСУРСЫ.

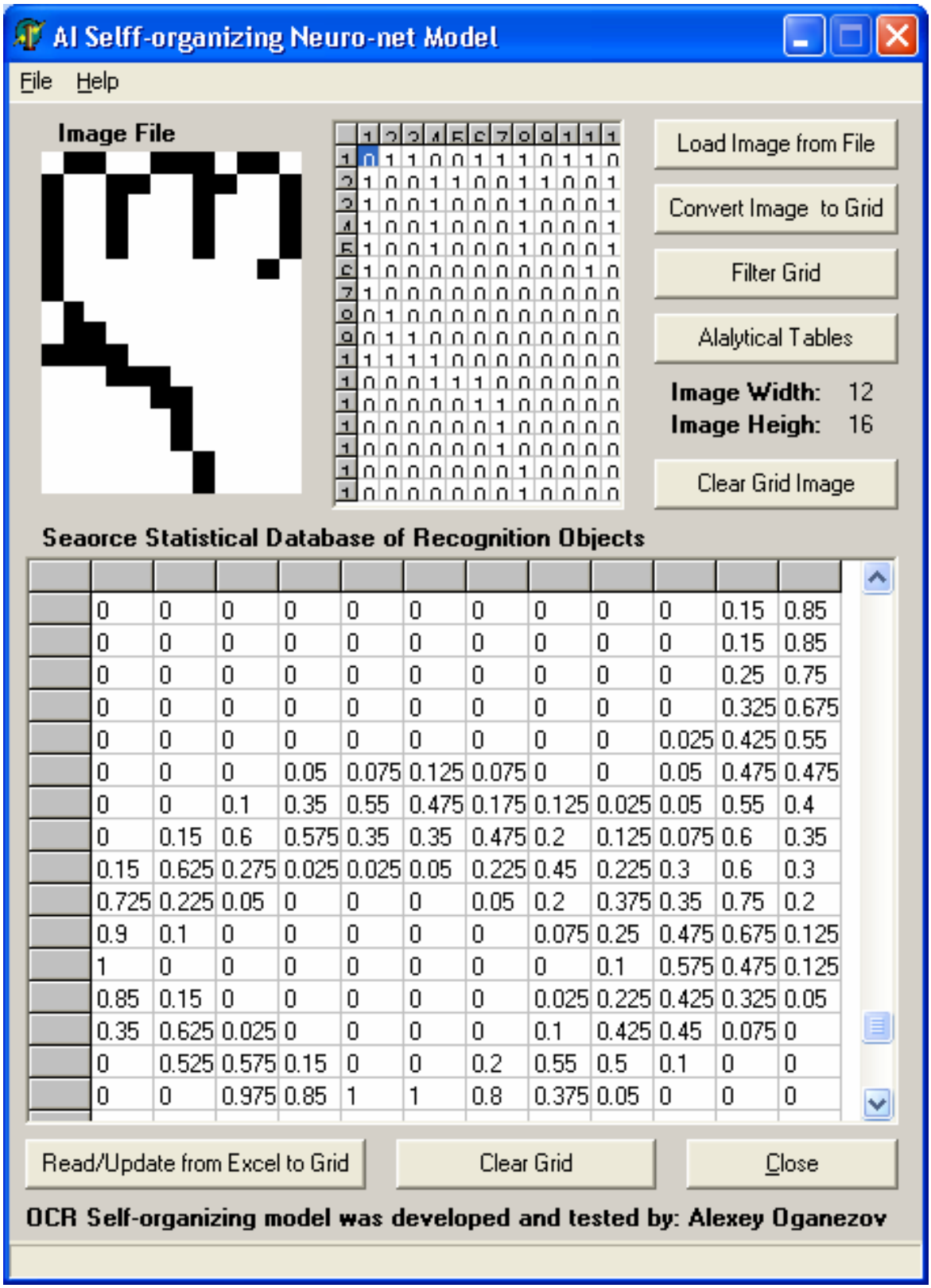
1. http://neuroproject.ru
2. http://www.brain.com
3. http://www2.cyber.rdg.ac.uk/cyber/
4. http://artsandscience3.concordia.ca/edtech/ETEC606/
5. http://www.ai.univie.ac.at/emcsr/
6. http://www.ieee.org/organizations/society/
7. http://www.lnt.e-technik.tu-muenchen.de
8. http://www.mpik-tueb.mpg.de/
9. http://www.medical-cybernetics.de/
10. http://www.frederic-vester.de/
11. http://link.springer.de/link/service/journals/00422/
12. http://www.cyber.brad.ac.uk//
13. http://pespmc1.vub.ac.be/ASC/
14. http://www.stormloader.com/gmdh/
15. http://context.umcs.maine.edu/
16. http://library.advanced.org/18242/index.shtml
17. http://www.robotwisdom.com
18. http://www.cybsoc.org/
19. http://www.uni-bielefeld.de/biologie/Kybernetik/
20. http://www.control-automatico.net/
21. http://www.neuron.co.uk/
22. http://www.andcorporation.com/
23. http://www.abspro.com/
24. http://www.bio-comp.com/
25. http://www.neurodynamx.com/
26. http://www.abo.fi/~abulsari/EANN95.html
27. http://www.kcl.ac.uk/neuronet/
28. http://www.neuralmachines.com/
29. http://www.csse.monash.edu.au/~app/CSC437nn/
30. ftp://ftp.sas.com/pub/neural/FAQ.html
31. http://www.mathworks.com/products/neuralnet/
32. http://www.elsevier.nl/locate/inca/841
33. http://www.cis.hut.fi/research/
34. http://www.nd.com/
35. http://www.ewh.ieee.org/tc/nnc/research/othernnsoc.html
36. http://www.ewh.ieee.org/tc/nnc/research/othernnsoc.html
37. http://www.cs.umbc.edu/~zwa/Gator/Description.html
38. http://liinwww.ira.uka.de/bibliography/Neural/index.html
39. http://www-ra.informatik.uni-tuebingen.de/SNNS/
40. http://www.ai.univie.ac.at/oefai/nn/nngroup.html
41. http://www.hav.com/
42. http://www.nua-tech.com/paddy/ai_etc.shtml
43. http://pespmc1.vub.ac.be/
44. http://www.devinci.fr/iim/vw2000/
45. http://tcw2.ppsw.rug.nl/~jorrit/ai/ai.html
46. http://www.mlnet.org/
47. http://www.aic.nrl.navy.mil/~aha/research/machine-learning.html
48. http://www.learningtheory.org/
49. http://www.recursive-partitioning.com/cgi-bin/mlfaq.cgi
50. http://forum.swarthmore.edu/~jay/learn-game/

51. http://attrasoft.com/decision/
52. http://www.csse.monash.edu.au/~dld/mixture.modelling.page.html
53. http://www.novuweb.com/index2.htm
54. http://www.acm.org/crossroads/xrds3-1/racra.html
55. http://svm.first.gmd.de/
56. http://www.mines.u-nancy.fr/~gueniffe/CoursEMN/I31/fcourse/fuzzy_appl.10.html
57. http://www.abo.fi/~rfuller/fuzs.html
58. http://www.ortech-engr.com/fuzzy/reservoir.html
59. http://www.cms.dmu.ac.uk/~rij/fuzzy.html
60. http://www.dbai.tuwien.ac.at/marchives/fuzzy-mail/index.html
61. http://www.emsl.pnl.gov:2080/proj/neuron/fuzzy/gateway/index.html
62. http://www.ics.uci.edu/~mlearn/MLRepository.html
63. http://www.fjeld.ch/ce/
64. http://www.bestweb.net/~sowa/ontology/
65. http://www.acm.org/crossroads/xrds5-2/kdd.html
66. http://www.kscape.com/
67. http://www.aridolan.com/
68. http://www.imagination-engines.com/xsum2.htm
69. http://www.cellmatrix.com/
70. http://www.geatbx.com/docu/algindex.html
71. http://www.natural-selection.com/eps/
72. http://www.pcai.com/pcai/
73. http://www.intelligentfirm.com/
74. http://sigart.acm.org/
75. http://ai.iit.nrc.ca/ai_point.html
76. http://www.brainmakers.org/
77. http://www.andcorporation.com/
78. http://www.abspro.com/
79. http://attrasoft.com
80. http://www.webbrain.com/
81. http://www.generation5.org/
82. http://www.artificialbrains.com/
83. http://www.dontveter.com/bpr/bpr.html
84. http://www.bio-comp.com/
85. http://www.mathworks.com/products/neuralnet/
86. http://www.aiai.ed.ac.uk/
87. http://www3.cm.deakin.edu.au/~chengqi/aies/
88. http://www.bcp.psych.ualberta.ca/
89. http://demo.cs.brandeis.edu/
90. http://www.cs.ubc.ca/nest/lci/
91. http://ai.obrazec.ru/
92. http://www.raai.org/about/persons/osipov/pages/ai/ai.html
93. http://www.conf.aintell.ru/2006/
94. http://pmg.org.ru/ai/index.html
95. http://www.neuroscience.ru/
96. http://riac.volsu.ru/neiro.html
97. http://library.mephi.ru/
98. http://www.synergetic.ru/science/print.php?print=redko1r
99. http://nisms.krinc.ru/papers/icnc99_3.pdf
100. http://rostov.rsu.ru/rsu-new1/about/sch6_.html
101. http://hmelnov.icc.ru/stud/lit/NN/neurinf.txt
102. http://www.neuroinf.org/

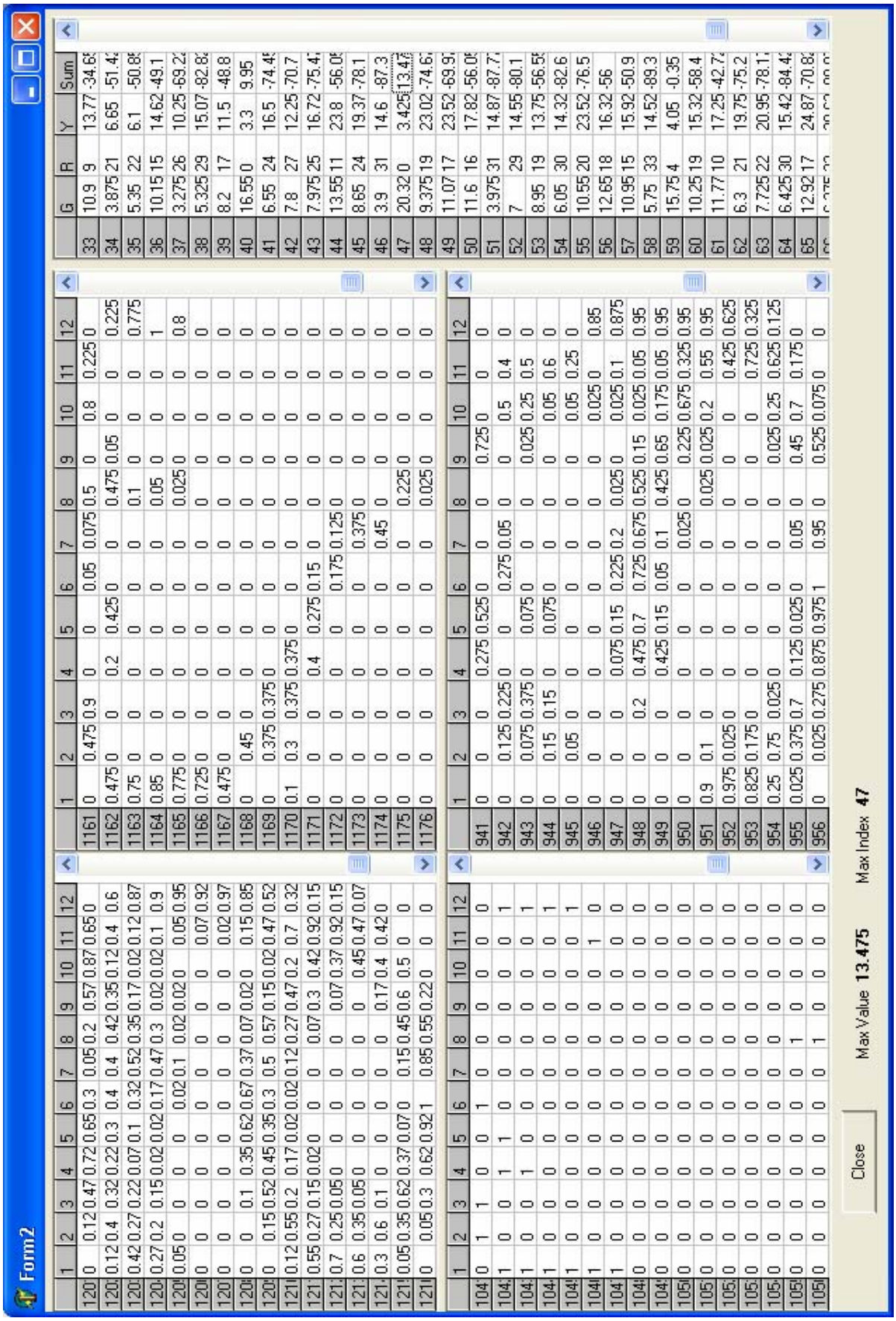
ПРИЛОЖЕНИЕ.

Фрагменты программы.

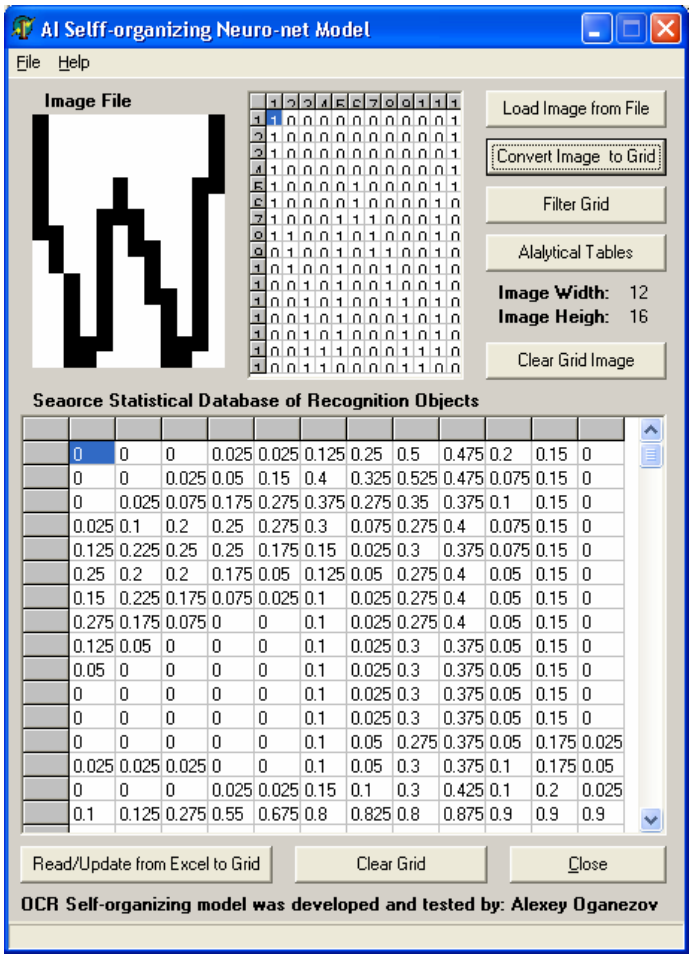
Фронтальная форма программы.

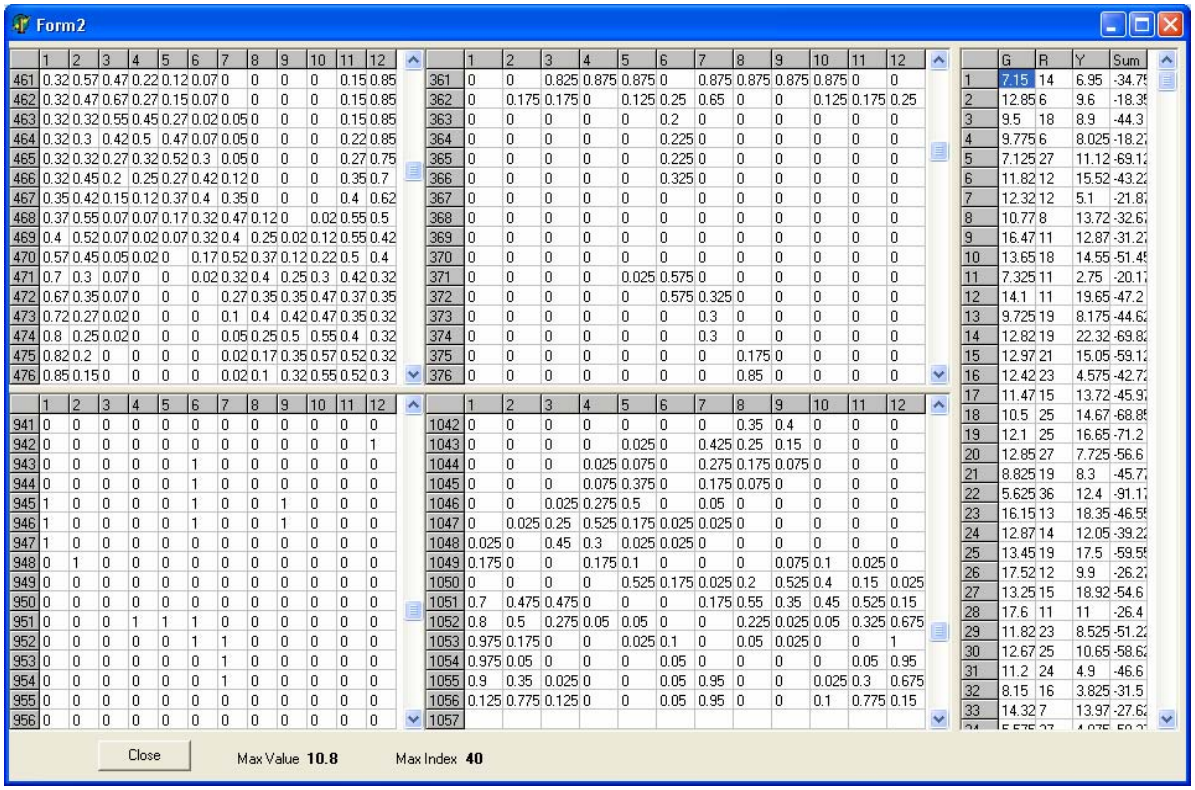


Аналитические таблицы подсчетов весов ИСНС для распознавания образов.



Пример идентификации буквы “W” и детектирования с помошью матричной ИСНС.





Листинг.

Проектный Файл.

```
program Project1;

uses
  Forms,
  Unit1 in 'Unit1.pas' {Form1},
  Unit13 in 'Unit13.pas' {OKBottomDlg},
  Unit14 in 'Unit14.pas' {Help},
  Unit2 in 'Unit2.pas' {Form2};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TOKBottomDlg, OKBottomDlg);
  Application.CreateForm(THelp, Help);
  Application.CreateForm(TForm2, Form2);
  Application.Run;
end.
```

Модуль 1.

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, StdCtrls, ExtCtrls, Math, Buttons, Spin, ImgList, ComCtrls, Menus,
  ExtDlgs, Outline, FileCtrl, ColorGrd, DirOutln, OleServer, Excel97, ComObj;

const
   m = 12;
   n = 16;

type
  TForm1 = class(TForm)
    StringGrid1: TStringGrid;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Help1: TMenuItem;
    Exit1: TMenuItem;
    About1: TMenuItem;
    Button14: TButton;
```

```
    hELP2: TMenuItem;
    Image1: TImage;
    Button58: TButton;
    Button83: TButton;
    Button84: TButton;
    Label9: TLabel;
    Label10: TLabel;
    StringGrid2: TStringGrid;
    Button86: TButton;
    Button57: TButton;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label1: TLabel;
    Button4: TButton;
    Button5: TButton;
    Label2: TLabel;

    procedure FormCreate(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure Button14Click(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure hELP2Click(Sender: TObject);
    procedure Button57Click(Sender: TObject);
    procedure Button58Click(Sender: TObject);
    procedure Button86Click(Sender: TObject);
    procedure Button84Click(Sender: TObject);
    procedure Button83Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
  private

    { Private declarations }
  public

    { Public declarations }
  end;

var
  Form1: TForm1;
  odb, M1, M2, M3 : Array [1..70,1..1400, 1..12] of String;

implementation

uses Unit13, Unit14, Unit2;

{$R *.DFM}

function Xls_To_StringGrid(AGrid: TStringGrid; AXLSFile: string): Boolean;
const
  xlCellTypeLastCell = $0000000B;
var
```

```
  XLApp, Sheet: OLEVariant;
  RangeMatrix: Variant;
  o, x7, y7, k7, r7: Integer;
begin
   Result := False;
  // Create Excel-OLE Object
  XLApp := CreateOleObject('Excel.Application');
  try
    // Hide Excel
    XLApp.Visible := False;
//    AXLSFile := 'C:\odb.xls';
    // Open the Workbook
    XLApp.Workbooks.Open(AXLSFile);

      // Sheet := XLApp.Workbooks[1].WorkSheets[1];
      Sheet := XLApp.Workbooks[ExtractFileName(AXLSFile)].WorkSheets[1];

    // In order to know the dimension of the WorkSheet, i.e the number of rows
    // and the number of columns, we activate the last non-empty cell of it

//    Sheet.Cells.SpecialCells(xlCellTypeLastCell, EmptyParam).Activate;
    // Get the value of the last row
    x7 := (16+4)*70+1; //XLApp.ActiveCell.Row;
    // Get the value of the last column
    y7 := 12+1; //XLApp.ActiveCell.Column;

    // Set Stringgrid's row &col dimensions.

    AGrid.RowCount := x7;
    AGrid.ColCount := y7;

    // Assign the Variant associated with the WorkSheet to the Delphi Variant
    RangeMatrix := XLApp.Range['A1', XLApp.Cells.Item[x7, y7]].Value;

    //  Define the loop for filling in the TStringGrid
    k7 := 1; o := 1;
    repeat
      for r7 := 1 to y7 do
        begin
        AGrid.Cells[r7, k7] := RangeMatrix[k7, r7];
//        odb[o,r7, k7] := AGrid.Cells[r7, k7];
        end;
      Inc(k7, 1); Inc(o,1);
      AGrid.RowCount := k7 + 1;
    until k7 > x7;

    // Unassign the Delphi Variant Matrix
    RangeMatrix := Unassigned;

  finally
    // Quit Excel
    if not VarIsEmpty(XLApp) then
```

```
    begin
     // XLApp.DisplayAlerts := False;
     XLApp.Quit;
     XLAPP := Unassigned;
     Sheet := Unassigned;
     Result := True;
    end;
  end;

end;


procedure TForm1.FormCreate(Sender: TObject);
var
 i, j: Integer;
begin
 // Header titles for table
 for i:=1 to n do
   StringGrid1.Cells[0,i] := IntToStr(i);
 for j:=1 to m do
   StringGrid1.Cells[j,0] := IntToStr(j);

 Image1.Picture.LoadFromFile('c:\object.bmp');
 Image1.Stretch := true;

 Xls_To_StringGrid(Form1.StringGrid2, 'C:\odb.xls')

end;


procedure TForm1.Exit1Click(Sender: TObject);
begin
 Close;
end;

procedure TForm1.Button14Click(Sender: TObject);
var
 i, j: Integer;
begin
 for i:=1 to 12 do
  for j:=1 to 16 do
   Form1.StringGrid1.Cells[i,j] := '';
end;


procedure TForm1.About1Click(Sender: TObject);
begin
 OKBottomDlg.Show;
end;


procedure TForm1.hELP2Click(Sender: TObject);
```

```
begin
 Help.Show;
end;


procedure TForm1.Button57Click(Sender: TObject);
var
  FromF: file;
  FileFrom : string;
  NumRead: Word;
  Buffer: array[1..1024] of Byte;
  i, j, k, l, m, count: Integer;
  pw, ph : Integer;
begin
  FileFrom := 'c:\object.bmp';
  AssignFile(FromF, FileFrom);
  Reset(FromF, 1);
//  NumRead := 5;
  BlockRead(FromF, Buffer, SizeOf(Buffer), count);
  pw := Buffer[19];
  ph := Buffer[23];

  label9.caption := InttoStr(Buffer[19]);
  label10.caption := InttoStr(Buffer[23]);

  CloseFile(FromF) ;
  k:=0; i:=1; j:=0;

  for i:=1 to ph do
   begin
    while j<pw do
//    for j:=1 to 12 do
     begin
     j:= j+2;
     //    Form1.StringGrid1.Cells[j,i] := Inttostr(Buffer[118+k+j]);
      m := trunc(j/2);
      if Buffer[118+k+m]=0 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '1';
        Form1.StringGrid1.Cells[j,ph+1-i] := '1';
       end;
      if Buffer[118+k+m]=15 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '1';
        Form1.StringGrid1.Cells[j,ph+1-i] := '0';
       end;
      if Buffer[118+k+m]=240 then
       begin
        Form1.StringGrid1.Cells[j-1,ph+1-i] := '0';
        Form1.StringGrid1.Cells[j,ph+1-i] := '1';
       end;
      if Buffer[118+k+m]=255 then
```

```
      begin
       Form1.StringGrid1.Cells[j-1,ph+1-i] := '0';
       Form1.StringGrid1.Cells[j,ph+1-i] := '0';
      end;
     end;
//    Form1.StringGrid1.Cells[j,i] := Inttostr(Buffer[118+k+j]);
   k:=k+trunc(pw/2)+2;
   j:=0;
 end;
end;


procedure TForm1.Button58Click(Sender: TObject);
begin
 Image1.Picture.LoadFromFile('c:\object.bmp');
 Image1.Stretch := true;
end;

procedure TForm1.Button86Click(Sender: TObject);
 var
   i,j: Integer;
begin
 for j:=1 to 12 do
   StringGrid2.Cells[j,0] := InttoStr(j);

 for i:=1 to 1400 do
   StringGrid2.Cells[0,i] := InttoStr(i);

if Xls_To_StringGrid(StringGrid2, 'C:\odb.xls') then
    ShowMessage('Table has been exported!');

{for i:=1 to 1400 do
  begin
   if ((i+2)/20 = trunc((i+2)/20)) then
     for j:=1 to 12 do
      begin
       StringGrid2.Cells[j,i] := '||||||||||||||||||';
       StringGrid2.Cells[j,i+1] := '||||||||||||||||||';
      end;
  end;}

end;


procedure TForm1.Button84Click(Sender: TObject);
 var
  i,j: integer;
begin
 for i:=1 to 12 do
 for j:=1 to (16+4)*70 do
    StringGrid2.Cells[i,j] := '';
end;
```

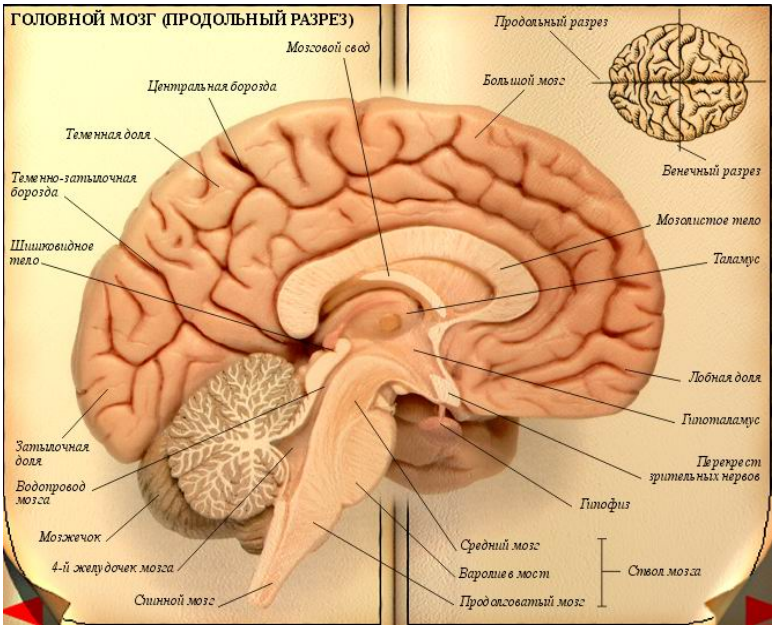
```
procedure TForm1.Button83Click(Sender: TObject);
var
 i, j, k, r, maxi: Integer;
 s2, s3, s4, maxv : real;
begin
 Form2.Show;

 for j:=1 to 12 do
  begin
   Form2.StringGrid1.Cells[j,0] := InttoStr(j);
   Form2.StringGrid2.Cells[j,0] := InttoStr(j);
   Form2.StringGrid3.Cells[j,0] := InttoStr(j);
   Form2.StringGrid4.Cells[j,0] := InttoStr(j);
  end;

 for i:=1 to 1400 do
  begin
   Form2.StringGrid1.Cells[0,i] := InttoStr(i);
   Form2.StringGrid2.Cells[0,i] := InttoStr(i);
   Form2.StringGrid3.Cells[0,i] := InttoStr(i);
   Form2.StringGrid4.Cells[0,i] := InttoStr(i);
  end;

   Form2.StringGrid5.Cells[1,0] := 'G';
   Form2.StringGrid5.Cells[2,0] := 'R';
   Form2.StringGrid5.Cells[3,0] := 'Y';
   Form2.StringGrid5.Cells[4,0] := 'Sum';

 for i:=1 to 70 do
   Form2.StringGrid5.Cells[0,i] := InttoStr(i);

 for j:=1 to 1400 do
  for i:=1 to 12 do
   Form2.StringGrid1.Cells[i,j] := Form1.StringGrid2.Cells[i,j];

 k:=0; r:=1;
 repeat
 for j:=1 to 16 do
  for i:=1 to 12 do
   begin
    Form2.StringGrid2.Cells[i,j+k]:=
FloattoStr(StrtoFloat(Form1.StringGrid2.Cells[i,j+k])*StrtoFloat(Form1.StringGrid1.Cells[i,j]));
    if StrtoFloat(Form1.StringGrid1.Cells[i,j])-StrtoFloat(Form2.StringGrid2.Cells[i,j+k]) = 1
     then Form2.StringGrid3.Cells[i,j+k] := '1'
      else Form2.StringGrid3.Cells[i,j+k] := '0' ;
    Form2.StringGrid4.Cells[i,j+k]  :=  FloattoStr(StrtoFloat(Form2.StringGrid1.Cells[i,j+k])-
StrtoFloat(Form2.StringGrid2.Cells[i,j+k]));
   end;

 s2 := 0; s3 := 0; s4 := 0;
```

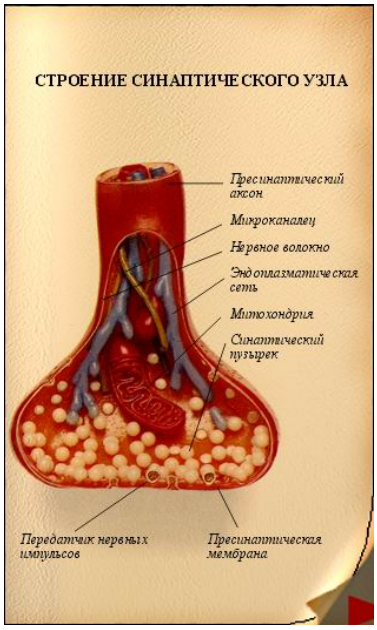
```
 for j:=1 to 16 do
  for i:=1 to 12 do
   begin
     s2 := s2 + StrtoFloat(Form2.StringGrid2.Cells[i,j+k]);
     s3 := s3 + StrtoFloat(Form2.StringGrid3.Cells[i,j+k]);
     if StrtoFloat(Form2.StringGrid4.Cells[i,j+k]) > 0.5
      then s4 := s4 + StrtoFloat(Form2.StringGrid4.Cells[i,j+k]);
   end;
 Form2.StringGrid5.Cells[1,r] := FloattoStr(s2);
 Form2.StringGrid5.Cells[2,r] := FloattoStr(s3);
 Form2.StringGrid5.Cells[3,r] := FloattoStr(s4);
 Form2.StringGrid5.Cells[4,r] := FloattoStr(s2-s3*2-s4*2);
 r := r+1;
 k := k + 20;
 until k>1376;

 maxv :=  StrtoFloat(Form2.StringGrid5.Cells[4,1]);
 maxi := 1;

 for i:=2 to 69 do
  if  StrtoFloat(Form2.StringGrid5.Cells[4,i]) > maxv
   then
    begin
     maxv := StrtoFloat(Form2.StringGrid5.Cells[4,i]);
     maxi := i;
    end;

  Form2.Label2.Caption := FloattoStr(maxv);
  Form2.Label4.Caption := InttoStr(maxi);

end;

procedure TForm1.Button4Click(Sender: TObject);
begin
 Form1.Close;
end;

procedure TForm1.Button5Click(Sender: TObject);
var i,j: Integer;
begin

 for i:=1 to 12 do
  for j:=1 to 16 do
   begin
    if Form1.StringGrid1.Cells[i,j] = '0'
     then Form1.StringGrid1.Cells[i,j] := ''
   end;

end;

end.
```

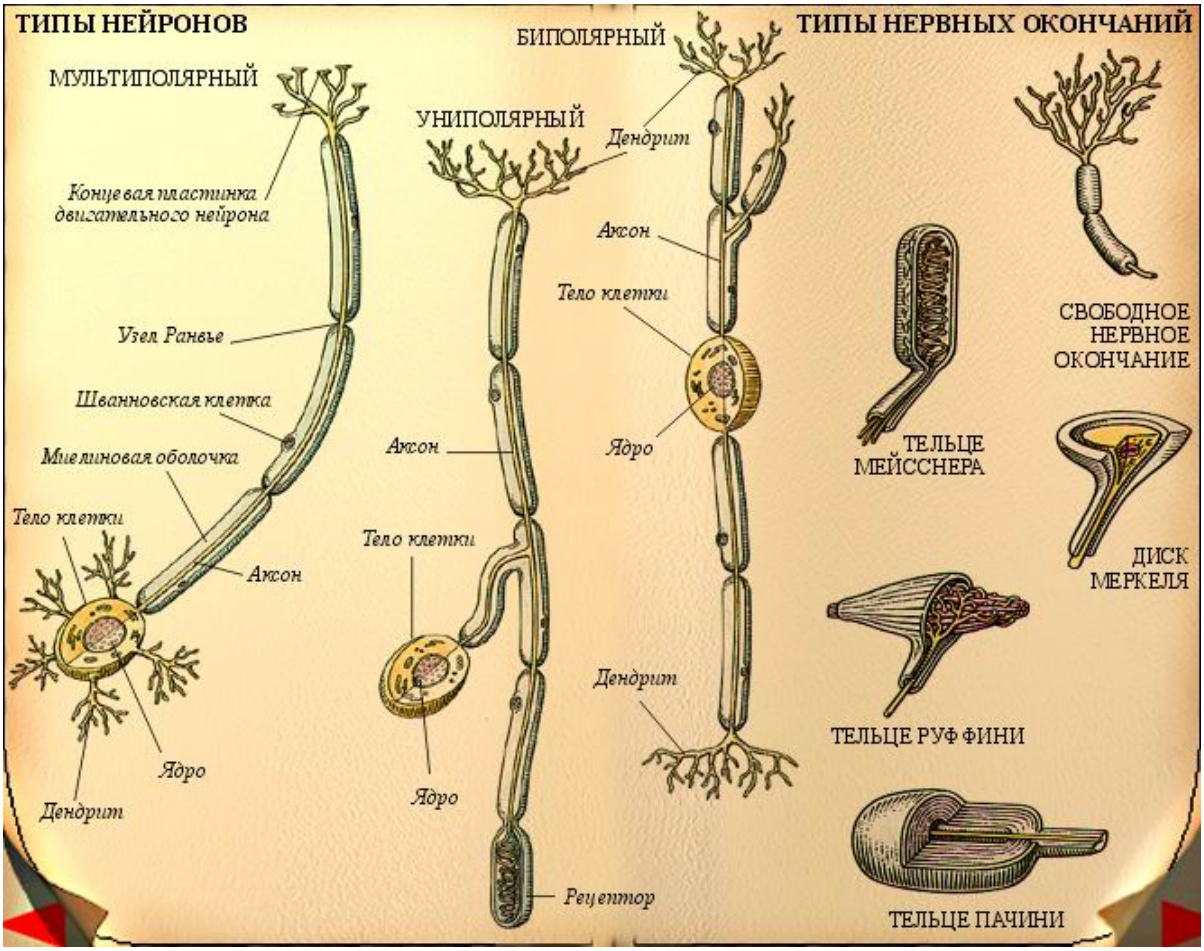
Модуль 2.

```
unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Grids;

type
  TForm2 = class(TForm)
    StringGrid3: TStringGrid;
    Button1: TButton;
    StringGrid1: TStringGrid;
    StringGrid2: TStringGrid;
    StringGrid4: TStringGrid;
    StringGrid5: TStringGrid;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

procedure TForm2.Button1Click(Sender: TObject);
begin

 Form2.Close;

end;

end.
```

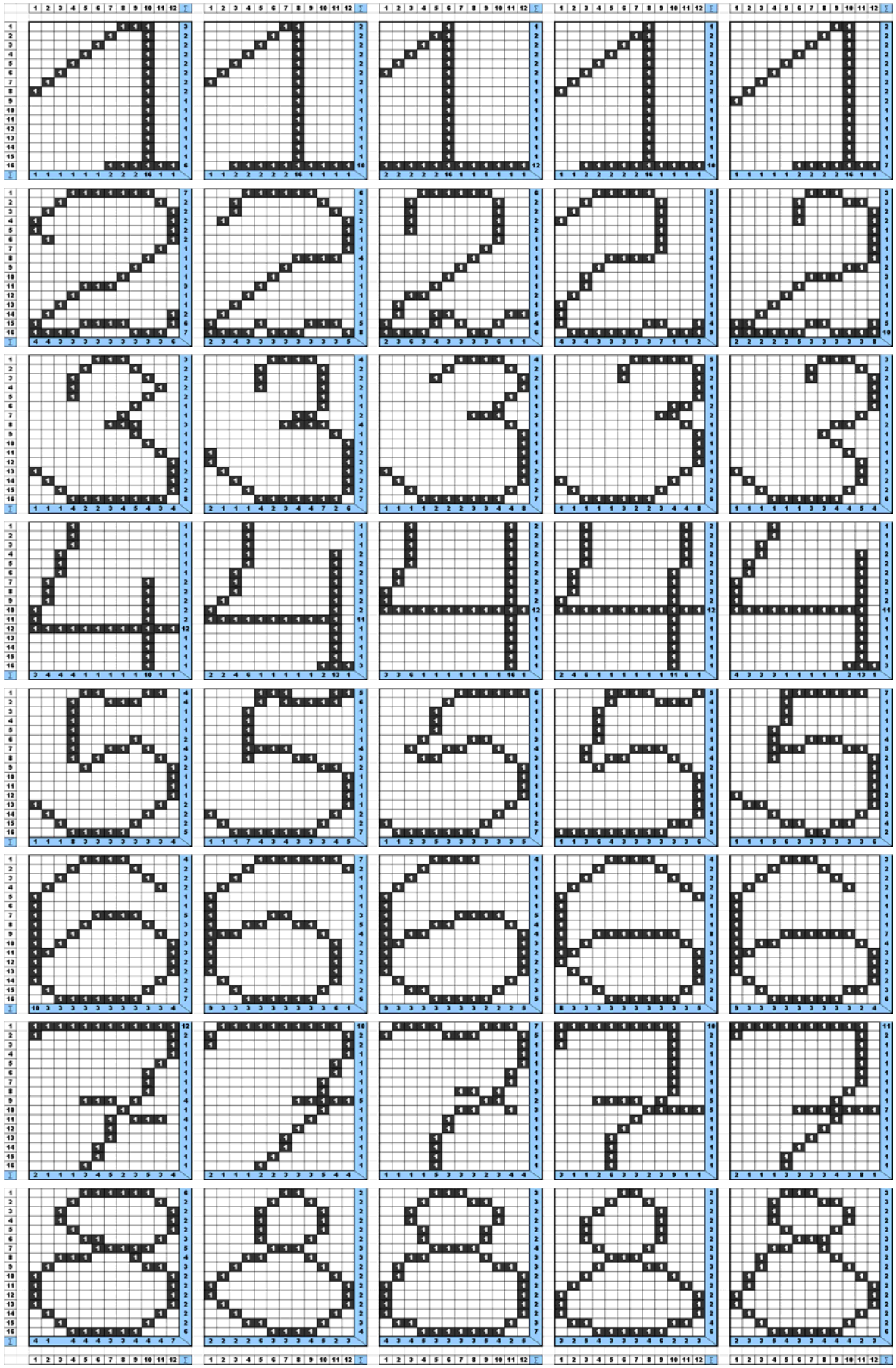
Иллюстрации.



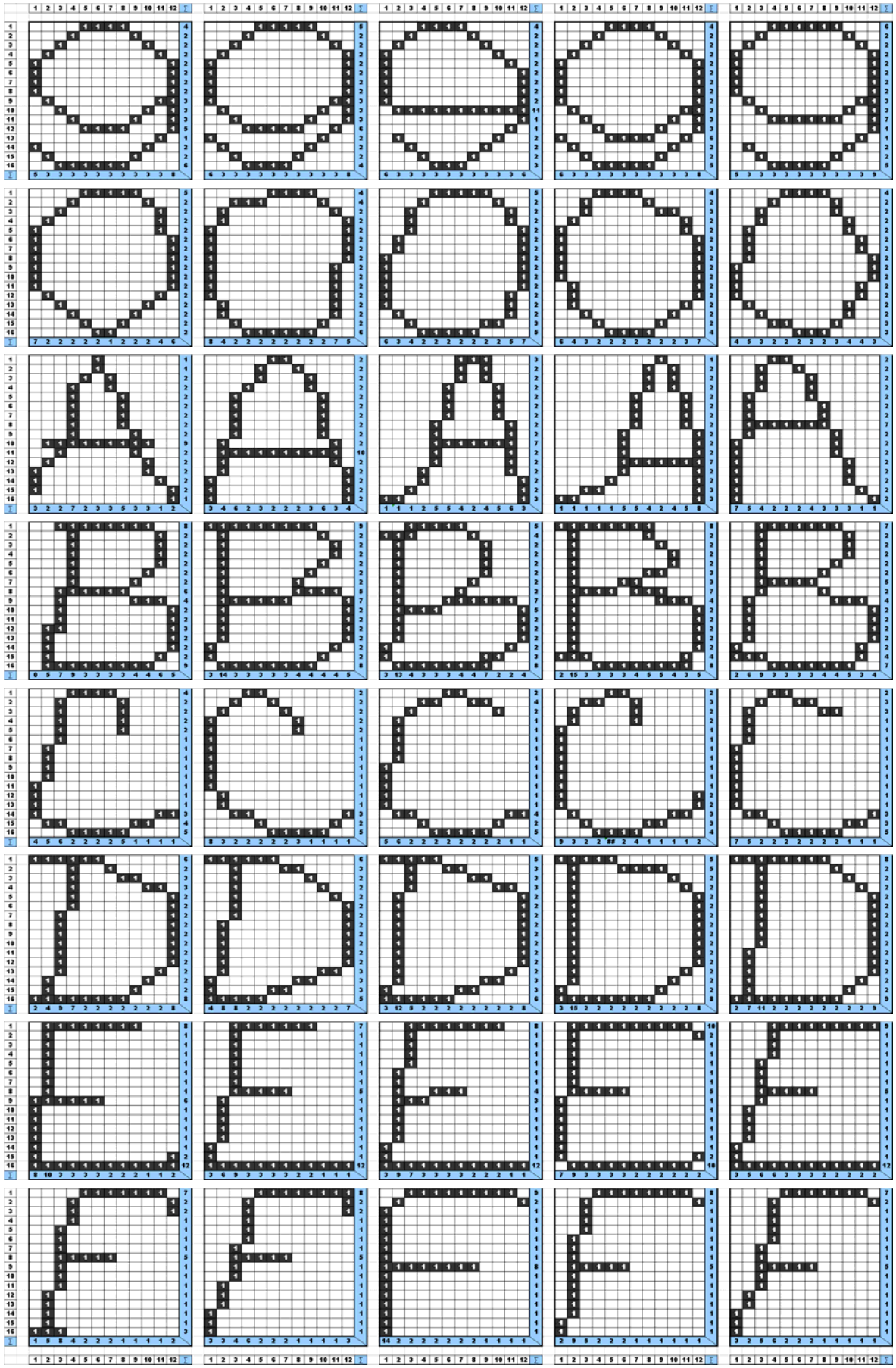




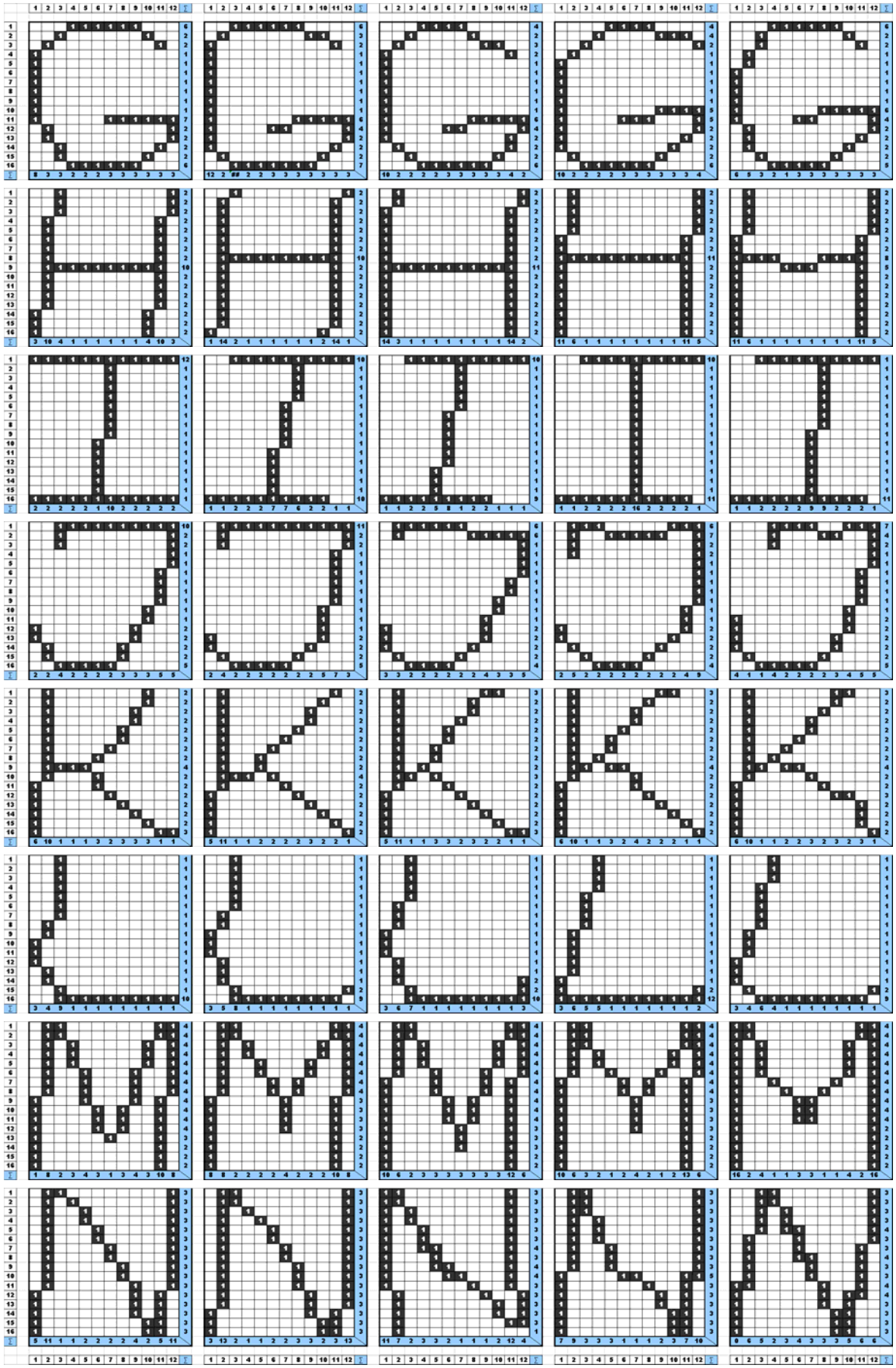
Примеры введенных эталонных символов “1” – “8” базы данных.



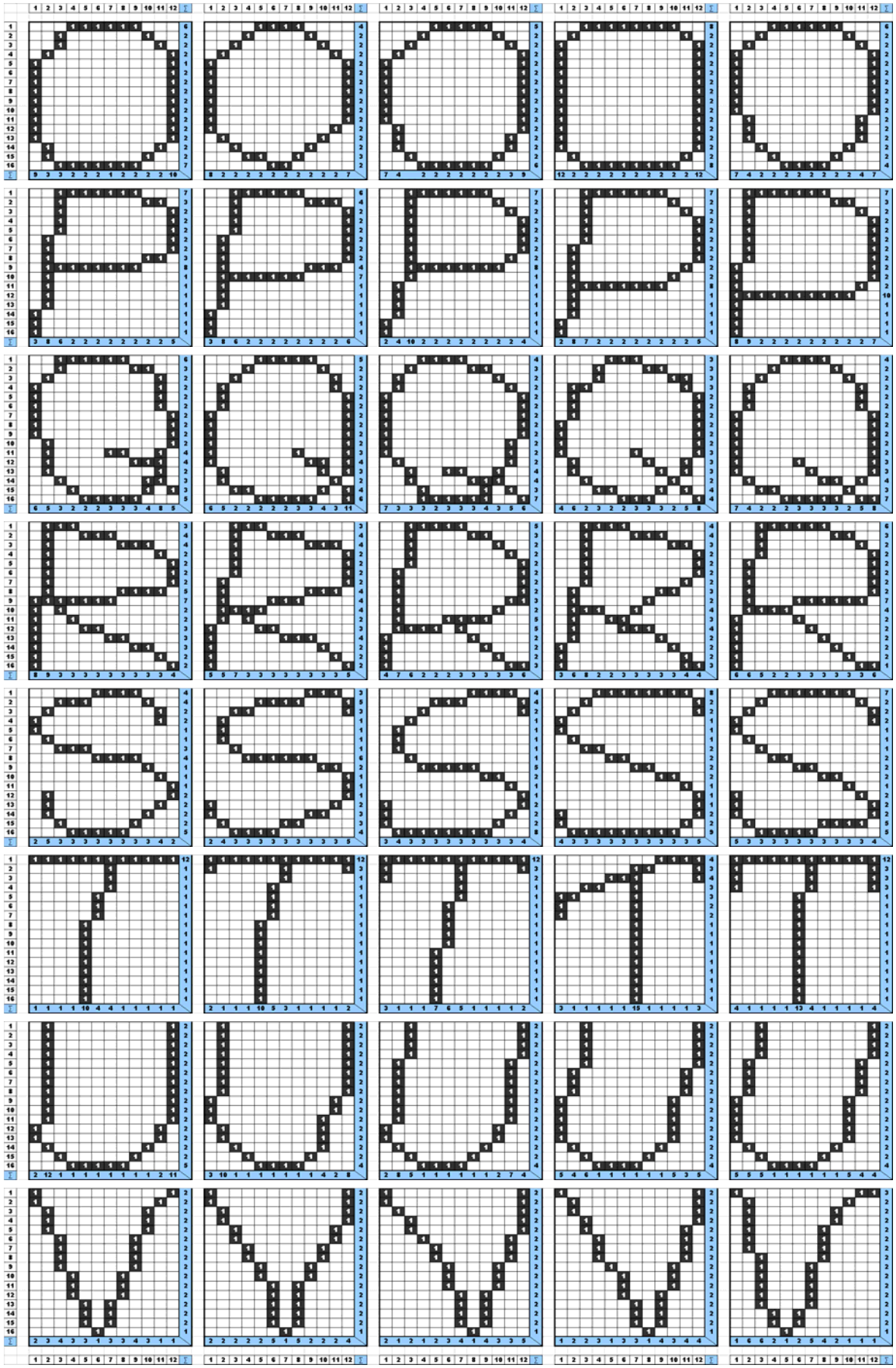
Примеры введенных эталонных символов “9” – “F” базы данных.



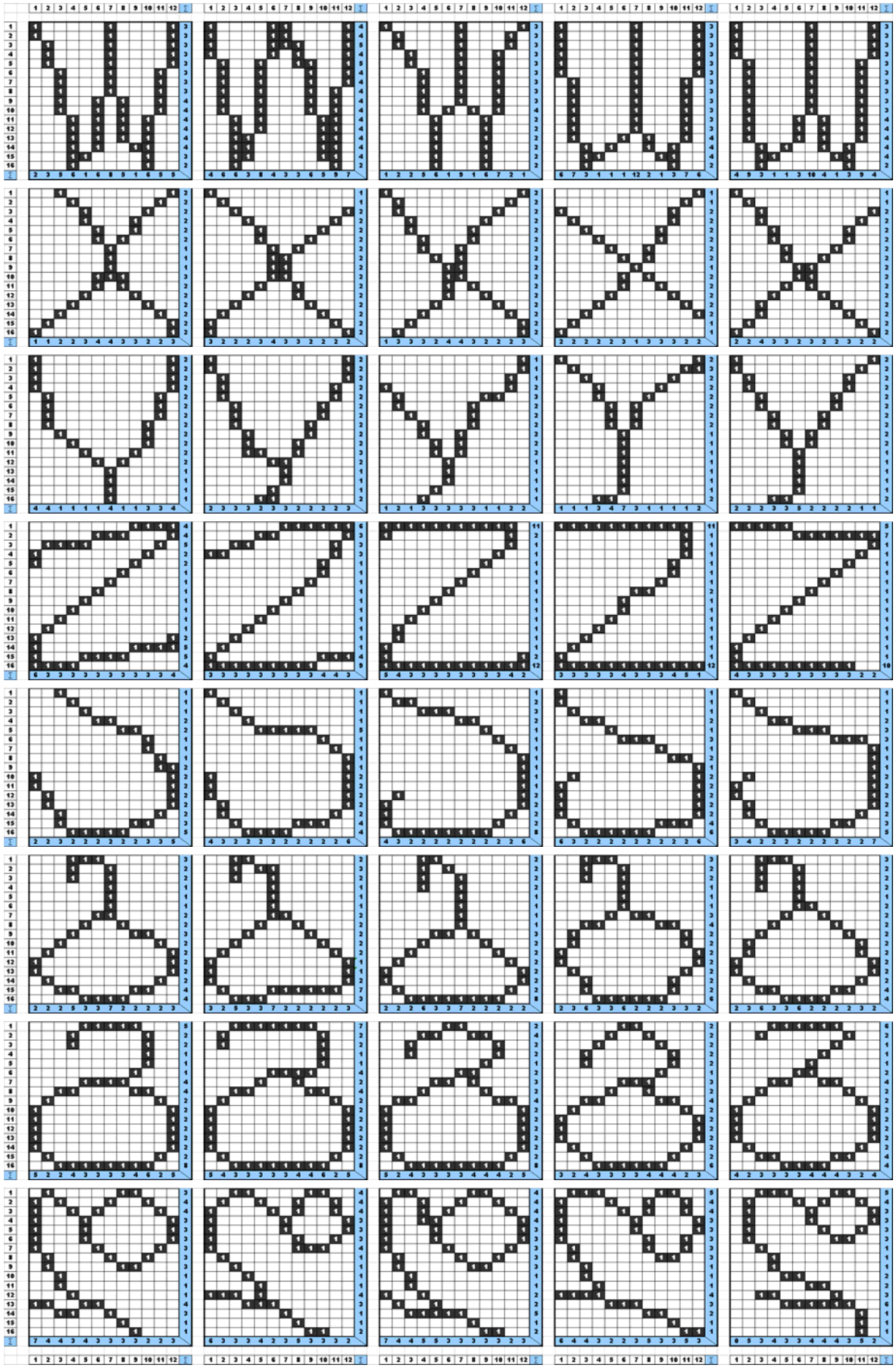
Примеры введенных эталонных символов “G” – “N” базы данных.



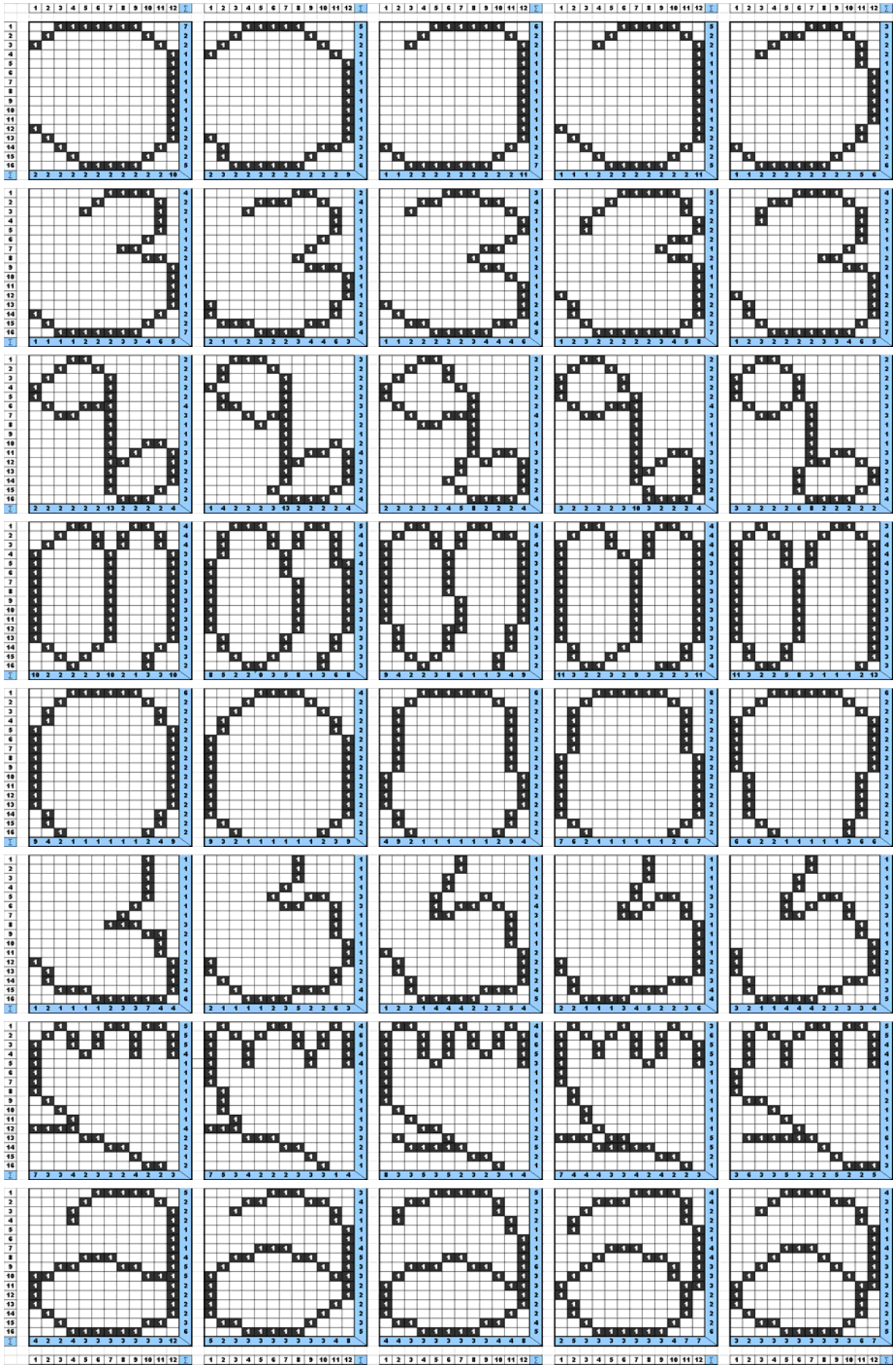
Примеры введенных эталонных символов “O” – “V” базы данных.



Примеры введенных эталонных символов “W” – “ᘯ” базы данных.



Примеры введенных эталонных символов “Ə” – “ə” базы данных.



Примеры введенных эталонных символов “б” – “ъ” базы данных.

Примеры введенных эталонных символов “ʒ” – “ɔ” базы данных.

Примеры введенных эталонных символов “Ѵ” – “ȝ” базы данных.

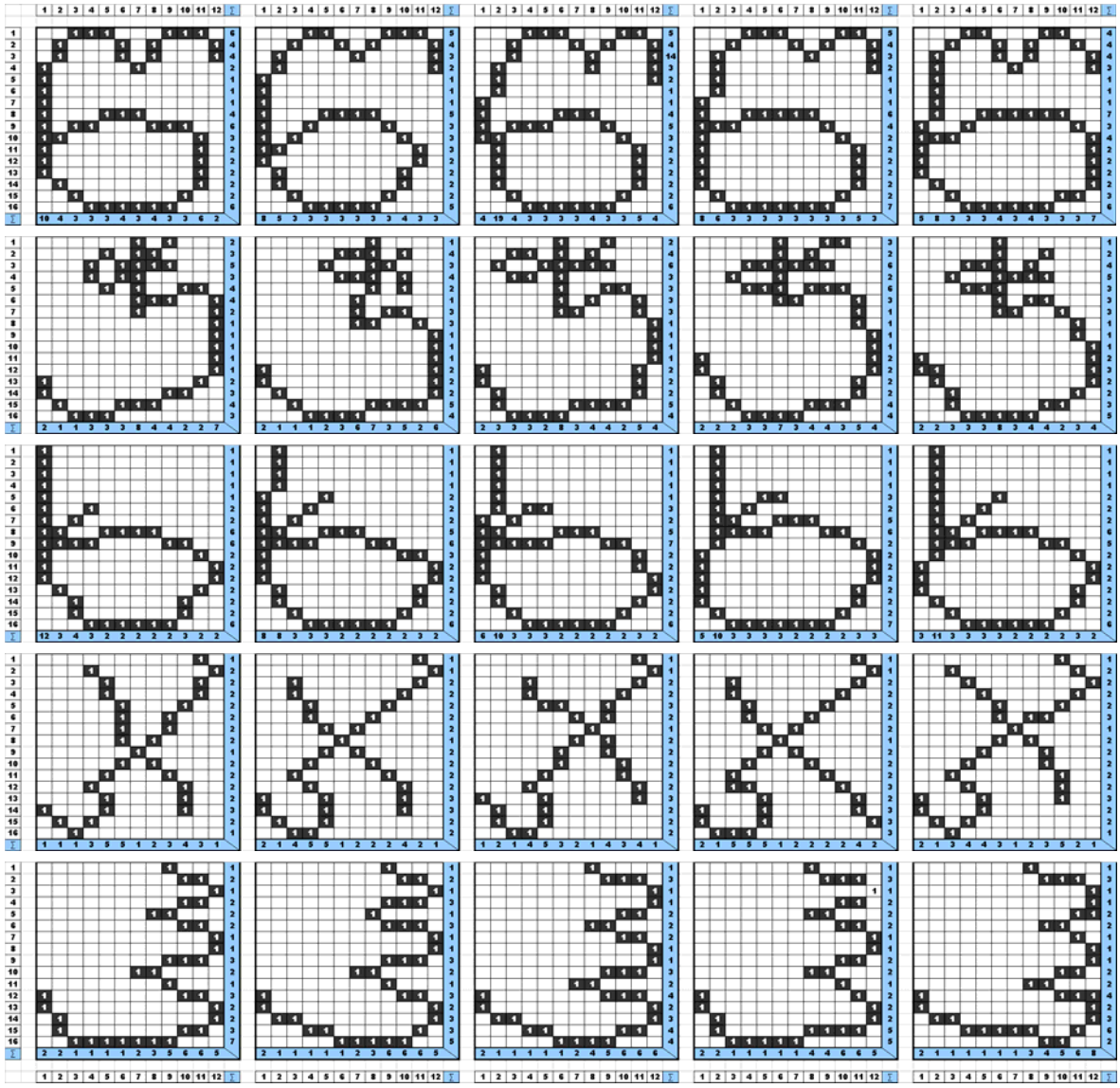


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	23	39	38	34	31	29	24	23	25	35	35	20	356
2	31	5	2	6	9	10	15	17	15	12	8	27	157
3	13	3	1	0	0	1	1	0	0	5	12	24	60
4	2	2	2	0	0	0	0	0	0	10	18	13	47
5	0	0	0	0	0	0	0	0	4	11	18	7	40
6	0	0	0	0	0	0	0	3	12	26	11	3	55
7	0	0	0	0	0	0	4	6	8	18	9	0	45
8	0	0	0	1	4	4	11	17	18	17	10	5	87
9	0	0	1	2	4	12	16	16	21	16	4	4	96
10	0	0	0	0	4	10	18	19	16	9	8	5	89
11	0	0	0	1	2	12	11	14	11	2	2	0	55
12	0	0	1	1	6	10	12	10	10	3	0	0	53
13	0	0	1	1	10	11	12	18	2	0	0	0	55
14	0	1	1	7	12	8	7	7	0	0	0	0	43
15	0	2	4	4	13	7	5	5	0	0	0	0	40
16	0	4	3	5	13	5	5	5	0	0	0	0	40
∑	69	56	54	62	108	119	141	160	142	164	135	108	

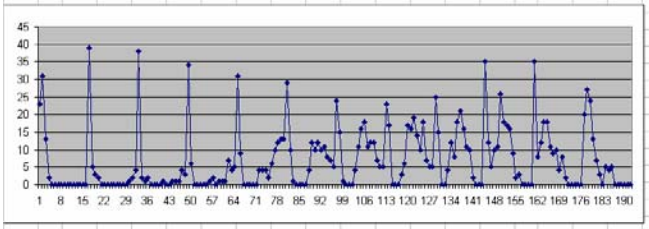
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.58	0.98	0.95	0.85	0.78	0.73	0.60	0.58	0.63	0.88	0.88	0.50	8.90
2	0.78	0.13	0.05	0.15	0.23	0.25	0.38	0.43	0.38	0.30	0.20	0.68	3.93
3	0.33	0.08	0.03	0.00	0.00	0.03	0.03	0.00	0.00	0.13	0.30	0.60	1.50
4	0.05	0.05	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.45	0.33	1.18
5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.28	0.18	1.00
6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.30	0.65	0.28	0.08	1.38
7	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.15	0.20	0.45	0.23	0.00	1.13
8	0.00	0.00	0.00	0.03	0.10	0.10	0.28	0.43	0.45	0.43	0.25	0.13	2.18
9	0.00	0.00	0.02	0.05	0.10	0.20	0.40	0.40	0.53	0.40	0.10	0.10	2.40
10	0.00	0.00	0.00	0.00	0.10	0.25	0.45	0.48	0.40	0.23	0.20	0.13	2.23
11	0.00	0.00	0.00	0.03	0.05	0.30	0.28	0.35	0.28	0.05	0.05	0.00	1.38
12	0.00	0.00	0.02	0.03	0.15	0.25	0.30	0.25	0.25	0.08	0.00	0.00	1.33
13	0.00	0.00	0.03	0.03	0.25	0.28	0.30	0.45	0.05	0.00	0.00	0.00	1.38
14	0.00	0.03	0.03	0.18	0.30	0.20	0.18	0.18	0.00	0.00	0.00	0.00	1.08
15	0.00	0.05	0.10	0.10	0.33	0.18	0.13	0.13	0.00	0.00	0.00	0.00	1.00
16	0.00	0.10	0.08	0.13	0.33	0.13	0.13	0.13	0.00	0.00	0.00	0.00	1.00
∑	1.73	1.40	1.35	1.55	2.70	2.98	3.53	4.00	3.55	4.10	3.38	2.70	

Статистические данные

Количество точек матрицы	192
Сумма весов матрицы	1,918
Сумма вероятностей матрицы	32.96
Среднее значение матрицы	5.86
Медиана весов матрицы	3.50
Вероятности:	Кол. точек
[0.00]	69
диапазон (0.00-0.25]	71
диапазон (0.25-0.50]	35
диапазон (0.50-0.75]	9
диапазон (0.75-1.00)	7
[1.00]	0

Линейный график точечных весов



3D-Колонная диаграмма точечных вероятностей

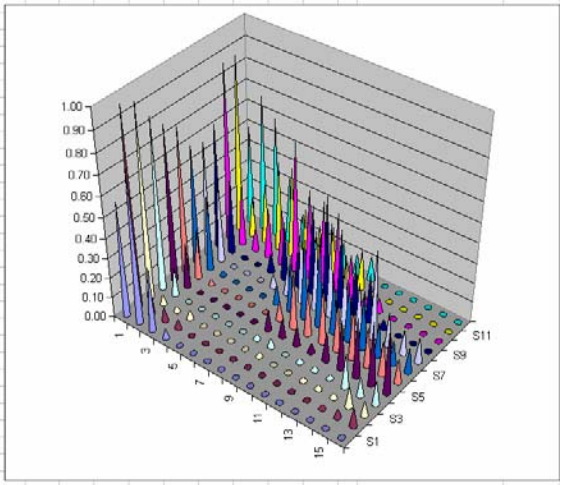


График сумм зондов по горизонтали

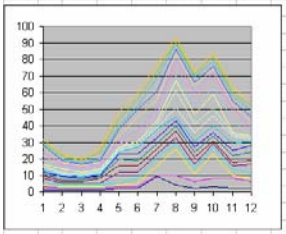


График сумм зондов по вертикали

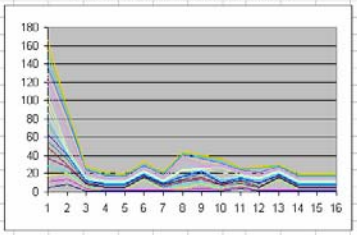


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0	0	2	20	39	40	40	36	27	2	0	0	206
2	0	2	25	19	1	0	0	4	12	29	3	0	95
3	2	19	17	2	0	0	0	0	1	14	23	3	81
4	14	21	5	0	0	0	0	0	0	2	19	19	80
5	27	12	1	0	0	0	0	0	0	0	11	29	80
6	34	6	0	0	0	0	0	0	0	0	3	37	80
7	34	6	0	0	0	0	0	0	0	0	7	34	81
8	29	11	0	0	0	0	0	0	0	3	8	33	84
9	20	12	8	5	3	2	2	3	4	5	21	29	114
10	5	19	12	5	3	4	3	4	8	18	26	25	132
11	1	6	18	14	14	14	15	14	19	15	17	24	171
12	4	1	4	14	17	16	16	17	9	7	15	20	140
13	16	5	1	2	3	4	4	2	2	9	19	11	78
14	13	23	5	1	0	0	0	1	7	13	18	5	86
15	2	13	26	12	4	3	3	9	24	18	4	0	118
16	0	1	9	27	36	37	37	30	11	3	0	0	191
∑	201	157	133	121	120	120	120	120	124	138	194	269	

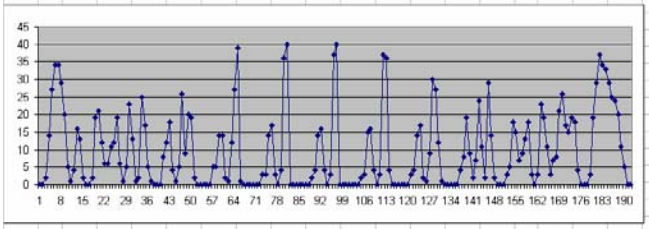
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.00	0.00	0.05	0.50	0.98	1.00	1.00	0.90	0.68	0.05	0.00	0.00	5.15
2	0.00	0.05	0.63	0.48	0.03	0.00	0.00	0.10	0.30	0.73	0.08	0.00	2.38
3	0.05	0.48	0.43	0.05	0.00	0.00	0.00	0.00	0.03	0.35	0.58	0.08	2.03
4	0.35	0.53	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.48	0.48	2.00
5	0.68	0.30	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.28	0.73	2.00
6	0.85	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.93	2.00
7	0.85	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.18	0.85	2.03
8	0.73	0.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.83	2.10
9	0.50	0.30	0.20	0.13	0.08	0.05	0.05	0.08	0.10	0.13	0.53	0.73	2.85
10	0.13	0.48	0.30	0.13	0.08	0.10	0.08	0.10	0.20	0.45	0.65	0.63	3.30
11	0.03	0.15	0.45	0.35	0.35	0.35	0.38	0.35	0.48	0.38	0.43	0.60	4.28
12	0.10	0.03	0.10	0.35	0.43	0.40	0.40	0.43	0.23	0.18	0.38	0.50	3.50
13	0.40	0.13	0.03	0.05	0.08	0.10	0.10	0.05	0.05	0.23	0.48	0.28	1.95
14	0.33	0.58	0.13	0.03	0.00	0.00	0.00	0.03	0.18	0.33	0.45	0.13	2.15
15	0.05	0.33	0.65	0.30	0.10	0.08	0.08	0.23	0.60	0.45	0.10	0.00	2.95
16	0.00	0.03	0.23	0.68	0.90	0.93	0.93	0.75	0.28	0.08	0.00	0.00	4.78
∑	5.03	3.93	3.33	3.03	3.00	3.00	3.00	3.00	3.10	3.45	4.85	6.73	

Статистические данные

Количество точек матрицы	192
Сумма весов матрицы	1,917
Сумма вероятностей матрицы	45.43
Среднее значение матрицы	5.45
Медиана весов матрицы	4.00
Вероятности:	Кол. точек
[0.00]	65
диапазон (0.00-0.25]	64
диапазон (0.25-0.50]	43
диапазон (0.50-0.75]	18
диапазон (0.75-1.00)	10
[1.00]	2

Линейный график точечных весов



3D-Колонная диаграмма точечных вероятностей

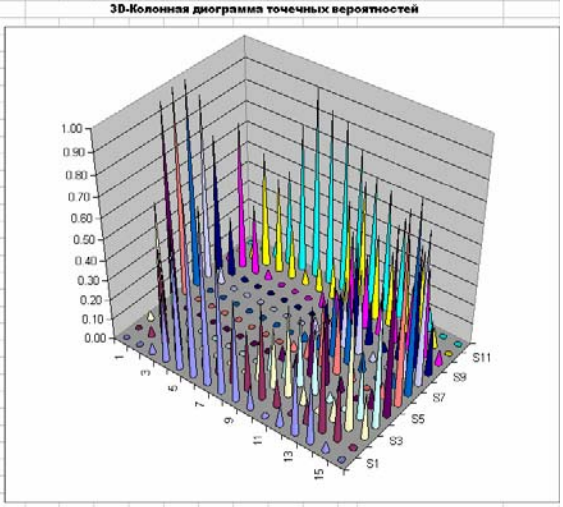


График сумм зондов по горизонтали

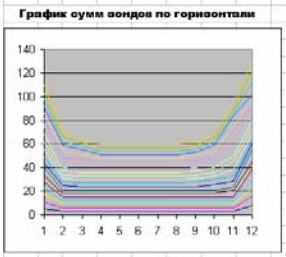


График сумм зондов по вертикали

Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	8	12	10	4	0	0	1	1	1	4	7	2	50
2	9	12	10	4	0	1	1	1	0	6	8	3	55
3	12	10	11	2	1	1	1	0	4	6	10	3	61
4	15	10	10	2	1	1	0	2	2	7	18	3	71
5	17	12	7	2	1	0	2	0	2	9	19	4	75
6	18	16	4	1	0	2	0	0	2	12	20	4	79
7	24	12	2	0	2	0	0	0	2	13	21	4	80
8	26	11	1	2	0	0	0	0	2	13	21	4	80
9	30	12	13	10	10	10	10	10	12	23	24	12	176
10	18	22	17	18	17	17	17	18	18	22	31	17	232
11	7	7	10	10	11	11	11	10	12	21	24	11	145
12	1	2	2	2	2	2	2	2	4	13	23	6	61
13	0	0	0	0	0	0	0	0	2	14	20	4	40
14	0	0	0	0	0	0	0	0	2	14	20	4	40
15	0	0	0	0	0	0	0	0	2	14	20	4	40
16	0	0	0	0	0	0	0	1	9	29	26	21	86
∑	185	138	97	57	45	45	45	45	76	220	312	106	

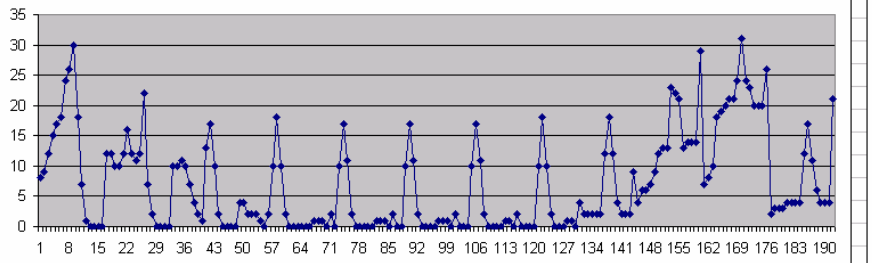
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.20	0.30	0.25	0.10	0.00	0.00	0.03	0.03	0.03	0.10	0.18	0.05	1.25
2	0.23	0.30	0.25	0.10	0.00	0.03	0.03	0.03	0.00	0.15	0.20	0.08	1.38
3	0.30	0.25	0.28	0.05	0.03	0.03	0.03	0.00	0.10	0.15	0.25	0.08	1.53
4	0.38	0.25	0.25	0.05	0.03	0.03	0.00	0.05	0.05	0.18	0.45	0.08	1.78
5	0.43	0.30	0.18	0.05	0.03	0.00	0.05	0.00	0.05	0.23	0.48	0.10	1.88
6	0.45	0.40	0.10	0.03	0.00	0.05	0.00	0.00	0.05	0.30	0.50	0.10	1.98
7	0.60	0.30	0.05	0.00	0.05	0.00	0.00	0.00	0.05	0.33	0.53	0.10	2.00
8	0.65	0.28	0.03	0.05	0.00	0.00	0.00	0.00	0.05	0.33	0.53	0.10	2.00
9	0.75	0.30	0.33	0.25	0.25	0.25	0.25	0.25	0.30	0.58	0.60	0.30	4.40
10	0.45	0.55	0.43	0.45	0.43	0.43	0.43	0.45	0.45	0.55	0.78	0.43	5.80
11	0.18	0.18	0.25	0.25	0.28	0.28	0.28	0.25	0.30	0.53	0.60	0.28	3.63
12	0.03	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.10	0.33	0.58	0.15	1.53
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.10	1.00
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.10	1.00
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.10	1.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.23	0.73	0.65	0.53	2.15
∑	4.63	3.45	2.43	1.43	1.13	1.13	1.13	1.13	1.90	5.50	7.80	2.65	

Статистические данные

Колличество точек матрицы	192
Сумма весов матрицы	1,371
Сумма вероятностей матрицы	34.28
Среднее значение матрицы	7.14
Медиана весов матрицы	4.00
Вероятности:	Кол. точек
[0.00]	50
диапазон (0.00-0.25]	84
диапазон (0.25-0.50]	42
диапазон (0.50-0.75]	15
диапазон (0.75-1.00)	1
[1.00]	0

Линейный график точечных весов



3D-Колонная диограмма точечных вероятностей

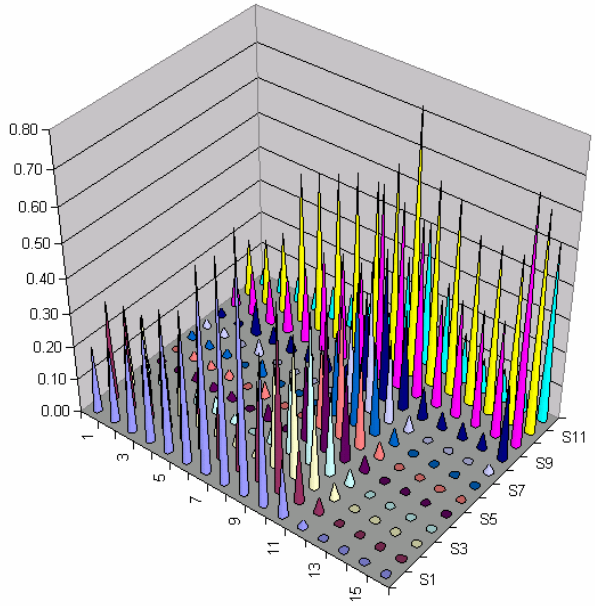


График сумм зондов по горизонтали

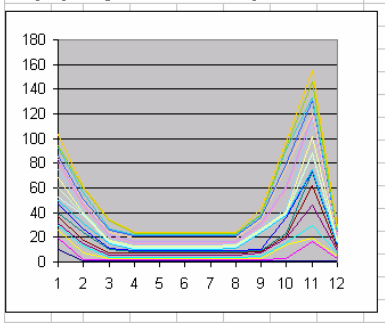


График сумм зондов по вертикали

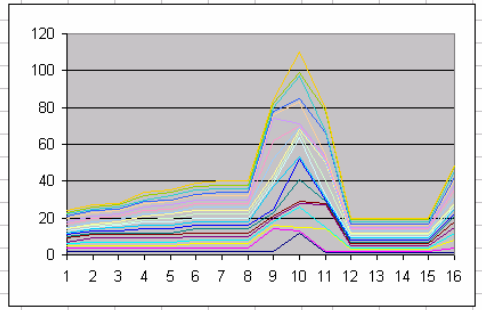


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	20	5	9	0	0	0	0	0	0	5	7	24	70
2	5	21	4	7	0	0	0	0	5	4	24	4	74
3	4	11	14	6	5	0	0	4	4	22	6	4	80
4	0	7	13	12	7	1	2	4	17	12	5	0	80
5	0	1	11	14	10	6	2	14	15	7	0	0	80
6	0	0	3	16	11	11	7	16	13	2	0	0	79
7	0	0	1	5	17	13	13	16	6	1	0	0	72
8	0	0	0	2	13	19	24	10	1	0	0	0	69
9	0	0	0	5	5	22	26	14	0	0	0	0	72
10	0	0	2	2	13	21	17	19	5	0	0	0	79
11	0	2	2	7	15	13	8	14	15	2	0	0	78
12	1	1	6	15	12	6	0	11	12	15	1	0	80
13	1	6	16	12	5	1	1	1	12	18	8	0	81
14	6	14	15	4	2	0	0	1	2	12	21	4	81
15	10	20	4	2	0	0	0	0	1	2	10	16	74
16	29	5	2	0	0	0	0	0	0	1	1	32	70
∑	76	93	102	109	115	113	100	124	108	103	92	84	

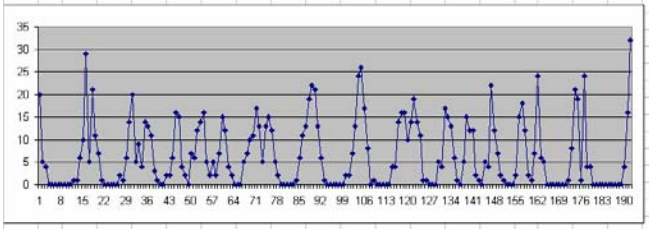
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.50	0.13	0.23	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.18	0.60	1.75
2	0.13	0.53	0.10	0.18	0.00	0.00	0.00	0.00	0.13	0.10	0.60	0.10	1.85
3	0.10	0.28	0.35	0.15	0.13	0.00	0.00	0.10	0.10	0.55	0.15	0.10	2.00
4	0.00	0.18	0.33	0.30	0.18	0.03	0.05	0.10	0.43	0.30	0.13	0.00	2.00
5	0.00	0.03	0.28	0.35	0.25	0.15	0.05	0.35	0.38	0.18	0.00	0.00	2.00
6	0.00	0.00	0.08	0.40	0.28	0.28	0.18	0.40	0.33	0.05	0.00	0.00	1.98
7	0.00	0.00	0.03	0.13	0.43	0.33	0.33	0.40	0.15	0.03	0.00	0.00	1.80
8	0.00	0.00	0.00	0.05	0.33	0.48	0.60	0.25	0.03	0.00	0.00	0.00	1.73
9	0.00	0.00	0.00	0.13	0.13	0.55	0.65	0.35	0.00	0.00	0.00	0.00	1.80
10	0.00	0.00	0.05	0.05	0.33	0.53	0.43	0.48	0.13	0.00	0.00	0.00	1.98
11	0.00	0.05	0.05	0.18	0.38	0.33	0.20	0.35	0.38	0.05	0.00	0.00	1.95
12	0.03	0.03	0.15	0.38	0.30	0.15	0.00	0.28	0.30	0.38	0.03	0.00	2.00
13	0.03	0.15	0.40	0.30	0.13	0.03	0.03	0.03	0.30	0.45	0.20	0.00	2.03
14	0.15	0.35	0.38	0.10	0.05	0.00	0.00	0.03	0.05	0.30	0.53	0.10	2.03
15	0.25	0.50	0.10	0.05	0.00	0.00	0.00	0.00	0.03	0.05	0.48	0.40	1.85
16	0.73	0.13	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.03	0.80	1.75
∑	1.90	2.33	2.55	2.73	2.88	2.83	2.50	3.10	2.70	2.58	2.30	2.10	

Статистические данные

Количество точек матрицы	192
Сумма весов матрицы	1,219
Сумма вероятностей матрицы	30.48
Среднее значение матрицы	6.35
Медиана весов матрицы	4.00
Вероятности:	Кол. точек
[0.00]	61
диапазон (0.00-0.25]	75
диапазон (0.25-0.50]	45
диапазон (0.50-0.75]	10
диапазон (0.75-1.00)	1
[1.00]	0

Линейный график точечных весов



3D-Колонная диаграмма точечных вероятностей

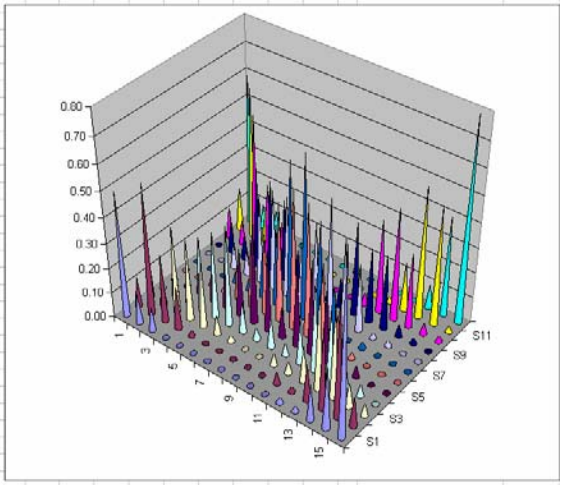


График сумм зондов по горизонтали

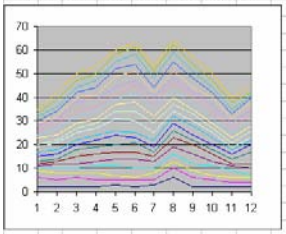


График сумм зондов по вертикали

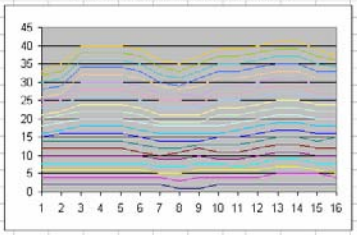


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	29	35	36	35	32	24	18	11	4	0	0	0	224
2	8	26	19	7	8	14	19	17	13	6	0	0	137
3	1	23	15	2	0	2	3	13	20	18	10	0	107
4	0	23	15	2	0	0	0	3	15	19	10	0	87
5	0	24	14	2	0	0	0	0	0	1	10	29	80
6	2	22	14	2	0	0	0	0	0	0	1	39	80
7	2	23	14	1	0	0	0	0	0	0	1	39	80
8	2	26	11	1	0	0	0	0	0	0	0	40	80
9	3	26	11	0	0	0	0	0	0	0	0	40	80
10	4	29	7	0	0	0	0	0	0	0	0	40	80
11	4	33	3	0	0	0	0	0	0	0	1	39	80
12	4	34	2	0	0	0	0	0	0	0	6	34	80
13	7	31	2	0	0	0	0	0	0	4	21	19	84
14	14	25	1	0	0	0	0	2	7	23	14	5	91
15	22	20	1	0	0	3	9	16	27	12	5	0	115
16	40	38	39	40	40	37	31	22	6	2	0	0	295
∑	142	438	204	92	80	80	80	81	80	81	88	334	

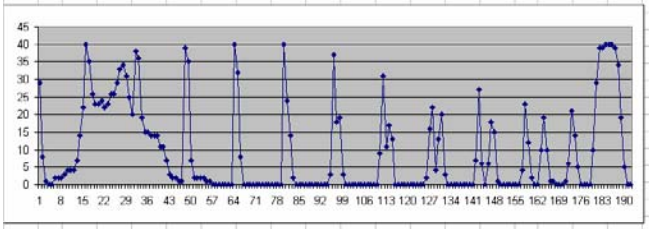
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.73	0.88	0.90	0.88	0.80	0.60	0.45	0.28	0.10	0.00	0.00	0.00	5.60
2	0.20	0.65	0.48	0.18	0.20	0.35	0.48	0.43	0.33	0.15	0.00	0.00	3.43
3	0.03	0.58	0.38	0.05	0.00	0.05	0.08	0.33	0.50	0.45	0.25	0.00	2.68
4	0.00	0.58	0.38	0.05	0.00	0.00	0.00	0.00	0.08	0.38	0.48	0.25	2.18
5	0.00	0.60	0.35	0.05	0.00	0.00	0.00	0.00	0.00	0.03	0.25	0.73	2.00
6	0.05	0.55	0.35	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.58	2.00
7	0.05	0.58	0.35	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.98	2.00
8	0.05	0.65	0.28	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00
9	0.08	0.65	0.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00
10	0.10	0.73	0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00
11	0.10	0.83	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.98	2.00
12	0.10	0.85	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15	0.85	2.00
13	0.18	0.78	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.53	0.48	2.10
14	0.35	0.63	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.18	0.58	0.35	0.13	2.28
15	0.55	0.50	0.03	0.00	0.00	0.00	0.03	0.23	0.40	0.68	0.30	0.13	0.00	2.88
16	1.00	0.95	0.98	1.00	1.00	0.93	0.78	0.55	0.15	0.05	0.00	0.00	7.38
∑	3.55	###	5.10	2.30	2.00	2.00	2.00	2.03	2.00	2.03	2.20	8.35	

Статистические данные

Количество точек матрицы	192
Сумма весов матрицы	1,788
Сумма вероятностей матрицы	44.69
Среднее значение матрицы	9.27
Медиана весов матрицы	2.00
Вероятности:	Кол. точек
[0.00]	82
диапазон (0.00-0.25]	46
диапазон (0.25-0.50]	25
диапазон (0.50-0.75]	18
диапазон (0.75-1.00)	15
[1.00]	6

Линейный график точечных весов



3D-Колонная диаграмма точечных вероятностей

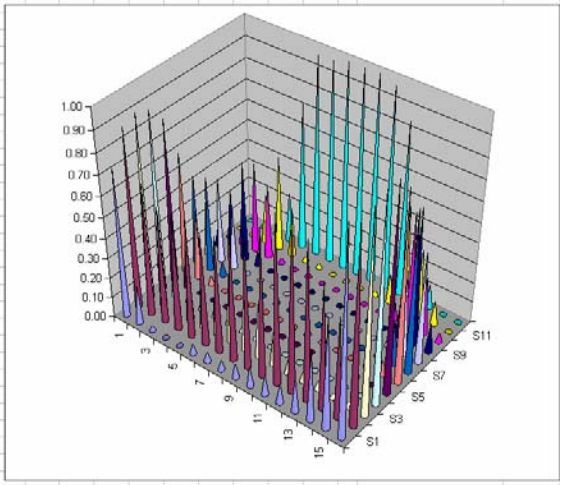


График сумм зондов по горизонтали

График сумм зондов по вертикали

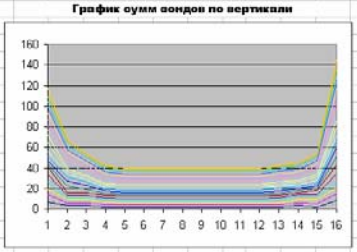


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	13	23	19	9	5	3	0	0	0	0	6	34	112
2	13	19	27	11	6	3	0	0	0	0	6	34	119
3	13	13	22	18	11	1	2	0	0	0	6	34	120
4	13	12	17	20	19	3	2	0	0	0	9	34	129
5	13	13	11	13	21	12	2	0	0	0	11	30	126
6	13	18	8	10	11	17	5	0	0	0	14	28	124
7	14	17	6	5	15	16	14	0	0	0	16	25	128
8	15	22	3	3	7	13	19	5	0	1	22	20	130
9	16	21	3	1	3	13	16	10	1	5	22	17	128
10	23	18	2	1	0	7	21	15	5	9	20	16	137
11	28	12	3	0	0	1	13	16	10	12	17	13	125
12	27	14	3	0	0	0	11	14	14	19	15	14	131
13	29	11	1	0	0	0	4	16	17	19	14	13	124
14	32	10	1	0	0	0	2	10	20	22	16	13	126
15	33	8	0	0	0	0	1	7	14	23	21	13	120
16	34	6	0	0	0	0	1	4	13	22	21	12	113
∑	329	237	126	91	98	89	113	97	94	132	236	350	

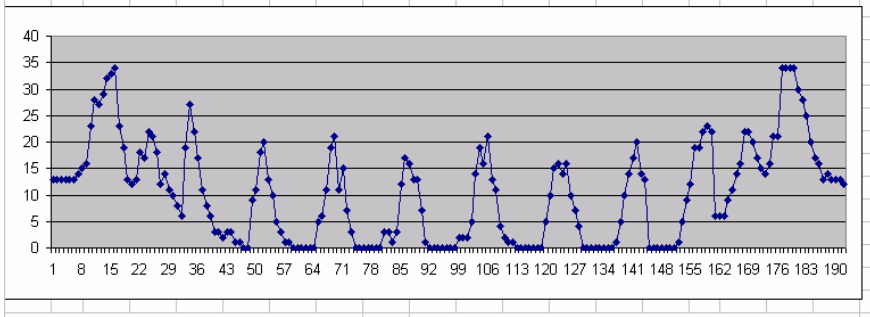
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.33	0.58	0.48	0.23	0.13	0.08	0.00	0.00	0.00	0.00	0.15	0.85	2.80
2	0.33	0.48	0.68	0.28	0.15	0.08	0.00	0.00	0.00	0.00	0.15	0.85	2.98
3	0.33	0.33	0.55	0.45	0.28	0.03	0.05	0.00	0.00	0.00	0.15	0.85	3.00
4	0.33	0.30	0.43	0.50	0.48	0.08	0.05	0.00	0.00	0.00	0.23	0.85	3.23
5	0.33	0.33	0.28	0.33	0.53	0.30	0.05	0.00	0.00	0.00	0.28	0.75	3.15
6	0.33	0.45	0.20	0.25	0.28	0.43	0.13	0.00	0.00	0.00	0.35	0.70	3.10
7	0.35	0.43	0.15	0.13	0.38	0.40	0.35	0.00	0.00	0.00	0.40	0.63	3.20
8	0.38	0.55	0.08	0.08	0.18	0.33	0.48	0.13	0.00	0.03	0.55	0.50	3.25
9	0.40	0.53	0.08	0.03	0.08	0.33	0.40	0.25	0.03	0.13	0.55	0.43	3.20
10	0.58	0.45	0.05	0.03	0.00	0.18	0.53	0.38	0.13	0.23	0.50	0.40	3.43
11	0.70	0.30	0.08	0.00	0.00	0.03	0.33	0.40	0.25	0.30	0.43	0.33	3.13
12	0.68	0.35	0.08	0.00	0.00	0.00	0.28	0.35	0.35	0.48	0.38	0.35	3.28
13	0.73	0.28	0.03	0.00	0.00	0.00	0.10	0.40	0.43	0.48	0.35	0.33	3.10
14	0.80	0.25	0.03	0.00	0.00	0.00	0.05	0.25	0.50	0.55	0.40	0.33	3.15
15	0.83	0.20	0.00	0.00	0.00	0.00	0.03	0.18	0.35	0.58	0.53	0.33	3.00
16	0.85	0.15	0.00	0.00	0.00	0.00	0.03	0.10	0.33	0.55	0.53	0.30	2.83
∑	8.23	5.93	3.15	2.28	2.45	2.23	2.83	2.43	2.35	3.30	5.90	8.75	

Статистические данные

Колличество точек матрицы	192
Сумма весов матрицы	1,992
Сумма вероятностей матрицы	49.80
Среднее значение матрицы	10.38
Медиана весов матрицы	11.00
Вероятности:	Кол. точек
[0.00]	44
диапазон (0.00-0.25]	51
диапазон (0.25-0.50]	69
диапазон (0.50-0.75]	21
диапазон (0.75-1.00)	7
[1.00]	0

Линейный график точечных весов



3D-Колонная диограмма точечных вероятностей

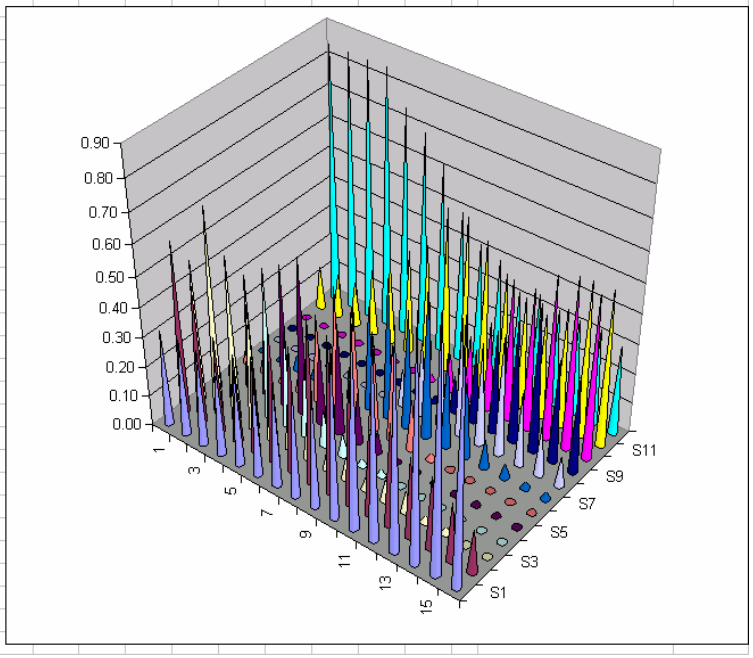


График сумм зондов по горизонтали

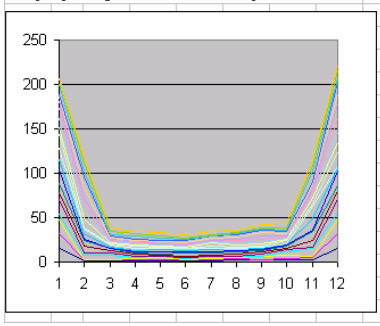


График сумм зондов по вертикали

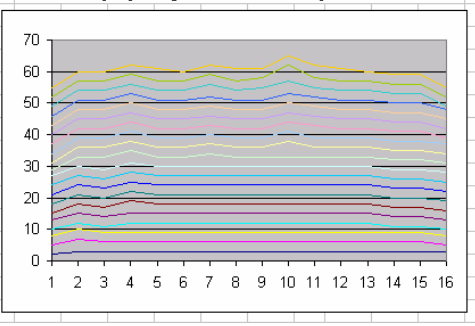


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0	5	19	29	26	12	2	8	23	35	26	0	185
2	5	16	13	9	12	16	16	17	14	5	16	24	163
3	17	11	9	3	4	13	21	14	7	1	5	35	140
4	11	8	6	1	1	7	19	12	1	1	4	36	107
5	2	0	0	0	0	1	4	1	1	0	2	38	49
6	0	0	0	0	0	0	0	0	0	0	3	37	40
7	0	0	0	0	0	0	0	0	0	0	1	39	40
8	0	0	4	14	25	27	15	3	1	0	6	34	129
9	0	6	21	18	14	12	20	23	6	1	19	21	161
10	5	22	8	7	1	1	5	11	19	8	28	13	128
11	22	11	6	1	0	0	0	3	12	17	37	6	115
12	28	10	2	0	0	0	0	0	3	15	37	6	101
13	24	14	2	0	0	0	0	0	0	18	19	3	80
14	12	24	4	0	0	0	0	0	7	16	17	0	80
15	2	14	25	15	3	0	6	18	24	20	0	0	127
16	0	2	12	25	37	40	34	22	9	0	0	0	181
∑	128	143	131	122	123	129	142	132	127	137	220	292	

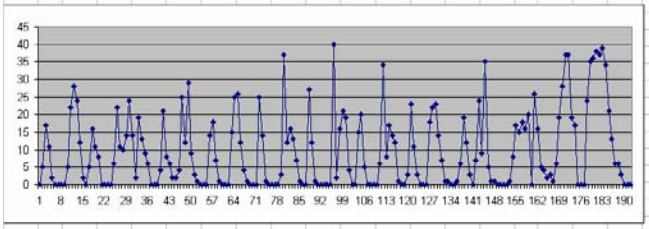
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.00	0.13	0.48	0.73	0.65	0.30	0.05	0.20	0.58	0.88	0.65	0.00	4.63
2	0.13	0.40	0.33	0.23	0.30	0.40	0.40	0.43	0.35	0.13	0.40	0.60	4.08
3	0.43	0.28	0.23	0.08	0.10	0.33	0.53	0.35	0.18	0.03	0.13	0.88	3.50
4	0.28	0.20	0.15	0.03	0.03	0.18	0.48	0.30	0.03	0.03	0.10	0.90	2.68
5	0.05	0.00	0.00	0.00	0.00	0.03	0.10	0.03	0.03	0.00	0.05	0.95	1.23
6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.93	1.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.98	1.00
8	0.00	0.00	0.10	0.35	0.63	0.68	0.38	0.08	0.03	0.00	0.15	0.85	3.23
9	0.00	0.15	0.53	0.45	0.35	0.30	0.50	0.58	0.15	0.03	0.48	0.53	4.03
10	0.13	0.55	0.20	0.18	0.03	0.03	0.13	0.28	0.48	0.20	0.70	0.33	3.20
11	0.55	0.28	0.15	0.03	0.00	0.00	0.00	0.08	0.30	0.43	0.93	0.15	2.88
12	0.70	0.25	0.05	0.00	0.00	0.00	0.00	0.00	0.08	0.38	0.93	0.15	2.53
13	0.60	0.35	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.45	0.48	2.00
14	0.30	0.60	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.18	0.40	0.43	2.00
15	0.05	0.35	0.63	0.38	0.08	0.00	0.15	0.45	0.60	0.50	0.00	0.00	3.18
16	0.00	0.05	0.30	0.63	0.93	1.00	0.85	0.55	0.23	0.00	0.00	0.00	4.53
∑	3.20	3.58	3.28	3.05	3.08	3.23	3.55	3.30	3.18	3.43	5.50	7.30	

Статистические данные

Количество точек матрицы	192
Сумма весов матрицы	1,026
Сумма вероятностей матрицы	46.66
Среднее значение матрицы	5.51
Медиана весов матрицы	6.00
Вероятности:	Кол. точек
[0.00]	58
диапазон (0.00-0.25]	59
диапазон (0.25-0.50]	42
диапазон (0.50-0.75]	21
диапазон (0.75-1.00)	11
[1.00]	1

Линейный график точечных весов



3D-Колонная диаграмма точечных вероятностей

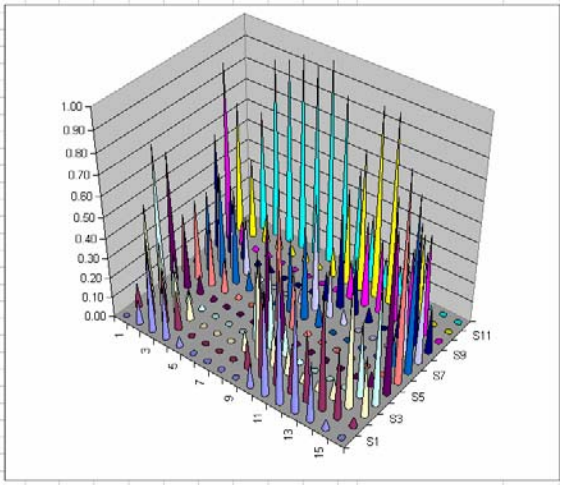


График сумм зондов по горизонтали

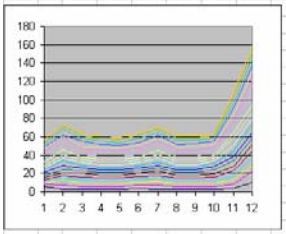


График сумм зондов по вертикали

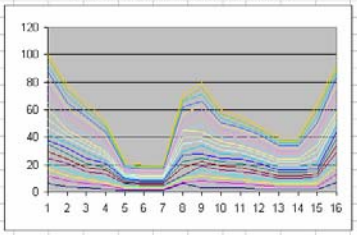


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	15	18	7	0	0	0	0	0	0	0	0	0	40
2	15	18	7	0	0	0	0	0	0	0	0	0	40
3	15	22	3	0	0	0	0	0	0	0	0	0	40
4	16	21	3	1	3	3	0	0	0	0	0	0	47
5	18	22	1	4	15	12	1	0	0	0	0	0	73
6	19	21	5	16	13	7	1	0	0	0	0	0	82
7	22	23	15	14	12	11	10	6	1	0	0	0	114
8	23	29	26	19	18	22	22	16	14	7	0	0	196
9	22	28	22	20	12	8	8	15	21	15	8	0	180
10	24	24	10	7	5	0	0	3	4	15	26	7	125
11	28	14	6	2	0	0	0	0	0	3	9	30	92
12	28	13	2	0	0	0	0	0	0	0	1	39	83
13	20	18	2	0	0	0	0	0	0	0	7	33	80
14	8	19	13	0	0	0	0	0	0	4	31	8	83
15	0	10	19	14	2	0	0	1	7	27	10	0	90
16	0	0	10	26	38	40	40	39	33	10	0	0	236
∑	274	300	151	123	118	103	82	80	80	81	92	117	

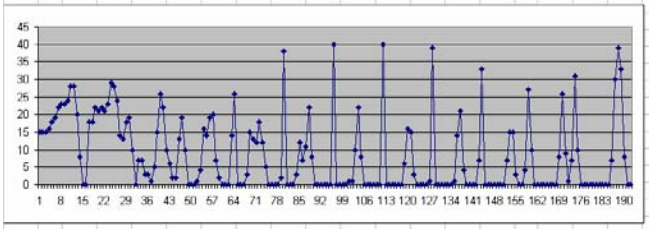
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.38	0.45	0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
2	0.38	0.45	0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
3	0.38	0.55	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
4	0.40	0.53	0.03	0.03	0.03	0.03	0.00	0.00	0.00	0.00	0.00	0.00	1.13
5	0.45	0.55	0.03	0.10	0.38	0.30	0.03	0.00	0.00	0.00	0.00	0.00	1.83
6	0.48	0.53	0.13	0.40	0.33	0.18	0.03	0.00	0.00	0.00	0.00	0.00	2.05
7	0.55	0.58	0.38	0.35	0.30	0.28	0.25	0.15	0.03	0.00	0.00	0.00	2.85
8	0.58	0.73	0.65	0.48	0.45	0.55	0.55	0.40	0.35	0.18	0.00	0.00	4.90
9	0.55	0.70	0.55	0.50	0.30	0.20	0.20	0.38	0.53	0.38	0.20	0.00	4.50
10	0.60	0.60	0.25	0.18	0.13	0.00	0.00	0.08	0.10	0.38	0.65	0.18	3.13
11	0.70	0.35	0.15	0.05	0.00	0.00	0.00	0.00	0.00	0.08	0.23	0.75	2.30
12	0.70	0.33	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.98	2.08
13	0.50	0.45	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.18	0.83	2.00
14	0.20	0.48	0.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.78	2.08
15	0.00	0.25	0.48	0.35	0.05	0.00	0.00	0.03	0.18	0.68	0.25	0.00	2.25
16	0.00	0.00	0.25	0.65	0.95	1.00	1.00	0.98	0.83	0.25	0.00	0.00	5.90
∑	6.85	7.50	3.78	3.08	2.95	2.58	2.05	2.00	2.00	2.03	2.30	2.93	

Статистические данные

Количество точек матрицы	192
Сумма весов матрицы	1,001
Сумма вероятностей матрицы	46.03
Среднее значение матрицы	5.34
Медиана весов матрицы	5.00
Вероятности:	Кол. точек
[0.00]	84
диапазон (0.00-0.25]	44
диапазон (0.25-0.50]	33
диапазон (0.50-0.75]	23
диапазон (0.75-1.00)	6
[1.00]	2

Линейный график точечных весов



3D-Колонная диаграмма точечных вероятностей

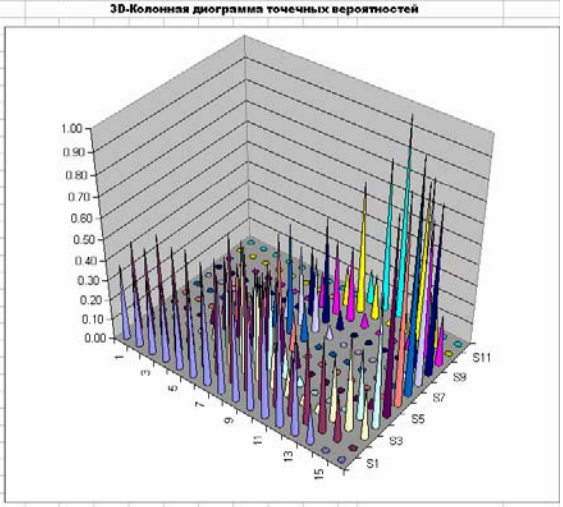


График сумм зондов по горизонтали

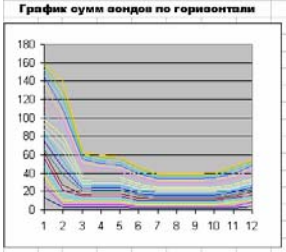


График сумм зондов по вертикали

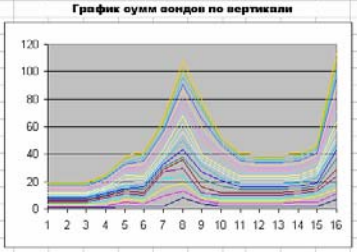


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0	19	31	23	10	3	8	23	39	30	7	0	193
2	19	17	8	14	15	15	19	16	1	10	26	7	167
3	30	9	1	4	13	18	17	5	0	0	11	29	137
4	34	6	0	2	10	15	11	3	0	0	0	40	121
5	31	9	0	1	6	19	8	6	0	0	6	34	120
6	29	9	2	0	4	10	15	8	11	10	20	15	133
7	19	16	4	1	1	5	8	17	13	18	14	3	119
8	10	18	11	5	0	1	7	6	10	9	3	1	81
9	5	15	15	9	6	0	0	5	6	3	1	0	65
10	4	12	15	15	9	8	0	0	0	0	0	0	63
11	4	5	13	16	11	6	8	1	0	0	0	0	64
12	12	13	12	19	10	7	5	8	2	0	0	0	88
13	5	11	13	13	23	21	15	7	9	2	0	0	119
14	0	1	4	6	6	12	18	19	7	7	2	0	82
15	0	0	0	1	1	2	2	9	19	17	7	1	59
16	0	0	0	0	0	0	1	1	7	16	19	11	55
∑	202	160	129	129	125	142	142	134	124	122	116	141	

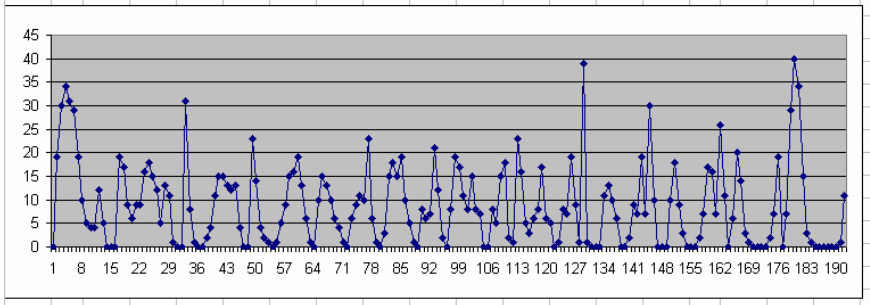
Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.00	0.48	0.78	0.58	0.25	0.08	0.20	0.58	0.98	0.75	0.18	0.00	4.83
2	0.48	0.43	0.20	0.35	0.38	0.38	0.48	0.40	0.03	0.25	0.65	0.18	4.18
3	0.75	0.23	0.03	0.10	0.33	0.45	0.43	0.13	0.00	0.00	0.28	0.73	3.43
4	0.85	0.15	0.00	0.05	0.25	0.38	0.28	0.08	0.00	0.00	0.00	1.00	3.03
5	0.78	0.23	0.00	0.03	0.15	0.48	0.20	0.15	0.00	0.00	0.15	0.85	3.00
6	0.73	0.23	0.05	0.00	0.10	0.25	0.38	0.20	0.28	0.25	0.50	0.38	3.33
7	0.48	0.40	0.10	0.03	0.03	0.13	0.20	0.43	0.33	0.45	0.35	0.08	2.98
8	0.25	0.45	0.28	0.13	0.00	0.03	0.18	0.15	0.25	0.23	0.08	0.03	2.03
9	0.13	0.38	0.38	0.23	0.15	0.00	0.00	0.13	0.15	0.08	0.03	0.00	1.63
10	0.10	0.30	0.38	0.38	0.23	0.20	0.00	0.00	0.00	0.00	0.00	0.00	1.58
11	0.10	0.13	0.33	0.40	0.28	0.15	0.20	0.03	0.00	0.00	0.00	0.00	1.60
12	0.30	0.33	0.30	0.48	0.25	0.18	0.13	0.20	0.05	0.00	0.00	0.00	2.20
13	0.13	0.28	0.33	0.33	0.58	0.53	0.38	0.18	0.23	0.05	0.00	0.00	2.98
14	0.00	0.03	0.10	0.15	0.15	0.30	0.45	0.48	0.18	0.18	0.05	0.00	2.05
15	0.00	0.00	0.00	0.03	0.03	0.05	0.05	0.23	0.48	0.43	0.18	0.03	1.48
16	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.03	0.18	0.40	0.48	0.28	1.38
∑	5.05	4.00	3.23	3.23	3.13	3.55	3.55	3.35	3.10	3.05	2.90	3.53	

Статистические данные

Колличество точек матрицы	192
Сумма весов матрицы	1,666
Сумма вероятностей матрицы	41.65
Среднее значение матрицы	8.68
Медиана весов матрицы	7.00
Вероятности:	Кол. точек
[0.00]	42
диапазон (0.00-0.25]	84
диапазон (0.25-0.50]	51
диапазон (0.50-0.75]	9
диапазон (0.75-1.00)	5
[1.00]	1

Линейный график точечных весов



3D-Колонная диограмма точечных вероятностей

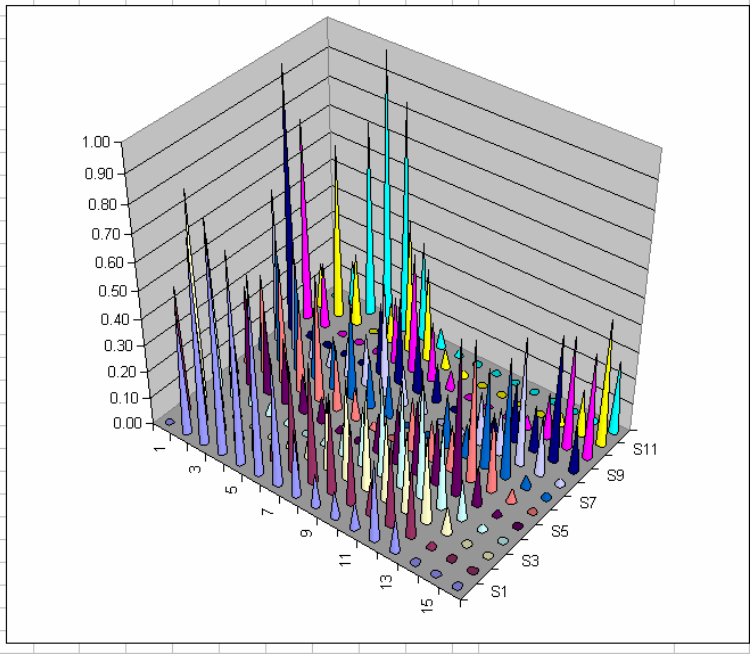


График сумм зондов по горизонтали

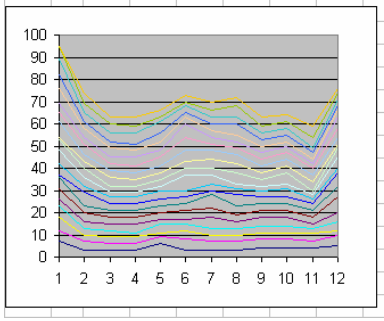
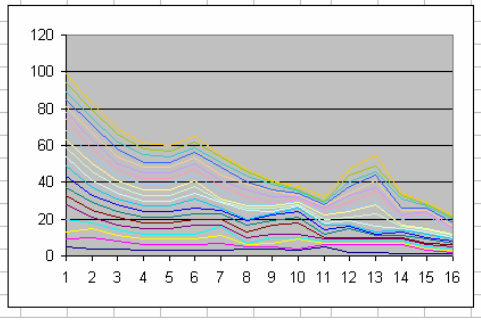


График сумм зондов по вертикали



Детектирование свойств цифр “3” и “5”.

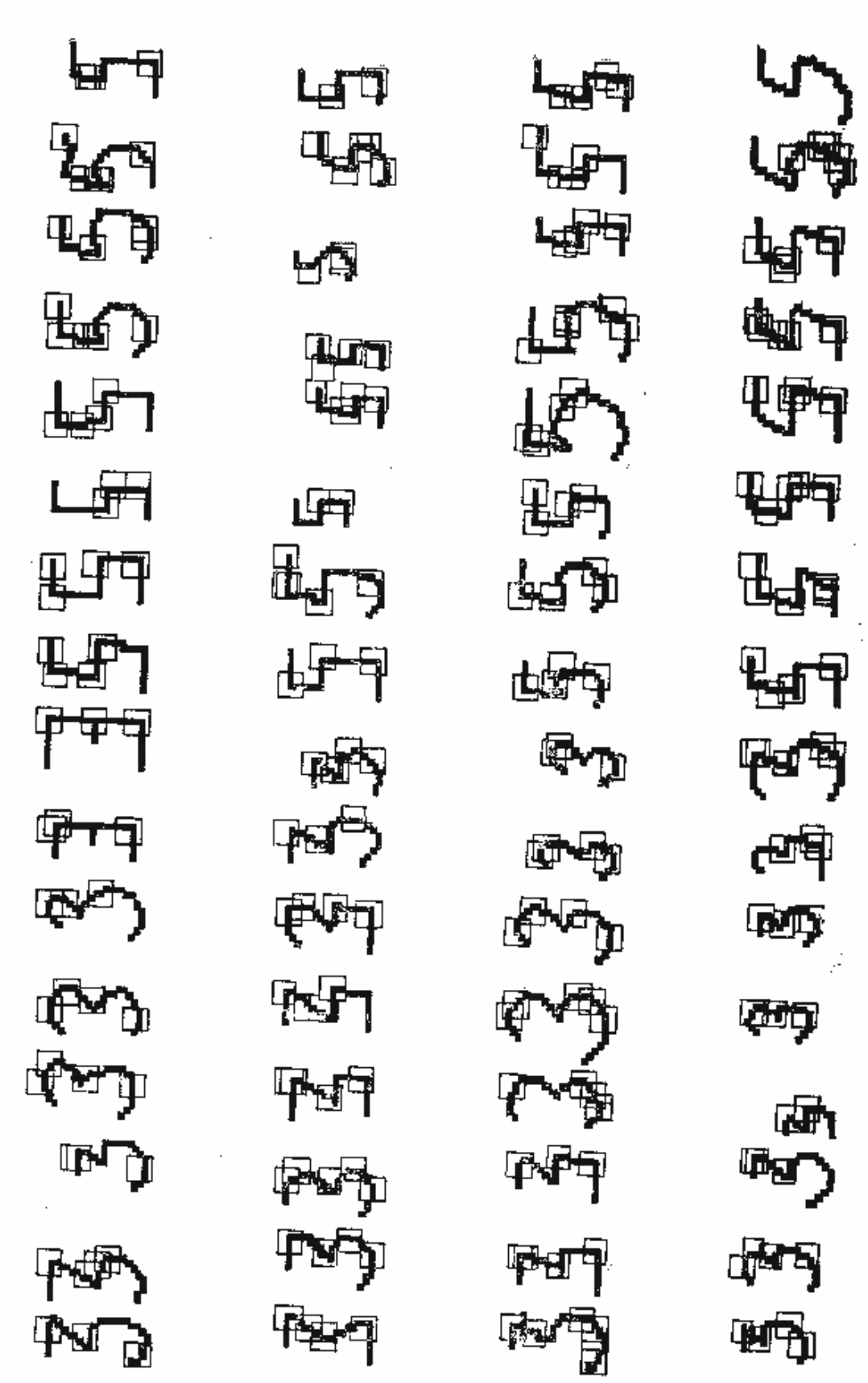


Таблица точечных весов

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0	0	3	14	19	27	32	35	36	30	11	2	209
2	0	8	21	19	20	13	8	5	3	8	21	11	137
3	8	24	14	7	1	0	0	0	1	2	12	15	84
4	32	7	2	0	0	0	0	0	0	0	3	3	47
5	34	6	0	0	0	0	0	0	0	0	0	0	40
6	21	15	7	0	0	0	0	0	0	0	0	0	43
7	7	15	18	16	7	3	1	1	1	0	0	0	69
8	0	7	13	19	23	20	14	8	5	2	1	0	112
9	0	0	2	5	9	15	20	22	16	12	5	1	107
10	0	0	0	0	1	2	5	6	15	17	16	6	68
11	0	0	0	0	0	0	0	3	4	6	11	22	46
12	9	1	0	0	0	0	0	0	0	3	6	33	52
13	16	4	0	0	0	0	0	0	1	3	9	31	64
14	17	15	3	0	0	0	2	3	6	9	16	16	87
15	12	15	13	5	3	6	7	10	11	19	16	0	117
16	2	12	24	35	37	34	31	27	22	10	0	0	234
∑	158	129	120	120	120	120	120	120	121	121	127	140	

Таблица точечных вероятностей

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.00	0.00	0.08	0.35	0.48	0.68	0.80	0.88	0.90	0.75	0.28	0.05	5.23
2	0.00	0.20	0.53	0.48	0.50	0.33	0.20	0.13	0.08	0.20	0.53	0.28	3.43
3	0.20	0.60	0.35	0.18	0.03	0.00	0.00	0.00	0.03	0.05	0.30	0.38	2.10
4	0.80	0.18	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.08	1.18
5	0.85	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
6	0.53	0.38	0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.08
7	0.18	0.38	0.45	0.40	0.18	0.08	0.03	0.03	0.03	0.00	0.00	0.00	1.73
8	0.00	0.18	0.33	0.48	0.58	0.50	0.35	0.20	0.13	0.05	0.03	0.00	2.80
9	0.00	0.00	0.05	0.13	0.23	0.38	0.50	0.55	0.40	0.30	0.13	0.03	2.68
10	0.00	0.00	0.00	0.00	0.03	0.05	0.13	0.15	0.38	0.43	0.40	0.15	1.70
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.10	0.15	0.28	0.55	1.15
12	0.23	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.15	0.83	1.30
13	0.40	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.08	0.23	0.78	1.60
14	0.43	0.38	0.08	0.00	0.00	0.00	0.05	0.08	0.15	0.23	0.40	0.40	2.18
15	0.30	0.38	0.33	0.13	0.08	0.15	0.18	0.25	0.28	0.48	0.40	0.00	2.93
16	0.05	0.30	0.60	0.88	0.93	0.85	0.78	0.68	0.55	0.25	0.00	0.00	5.85
∑	3.95	3.23	3.00	3.00	3.00	3.00	3.00	3.00	3.03	3.03	3.18	3.50	


Линейный график точечных весов
40
35
30
25
20
15
10
5
0
1 8 15 22 29 36 43 50 57 64 71 78 85 92 99 106 113 120 127 134 141 148 155 162 169 176 183 190

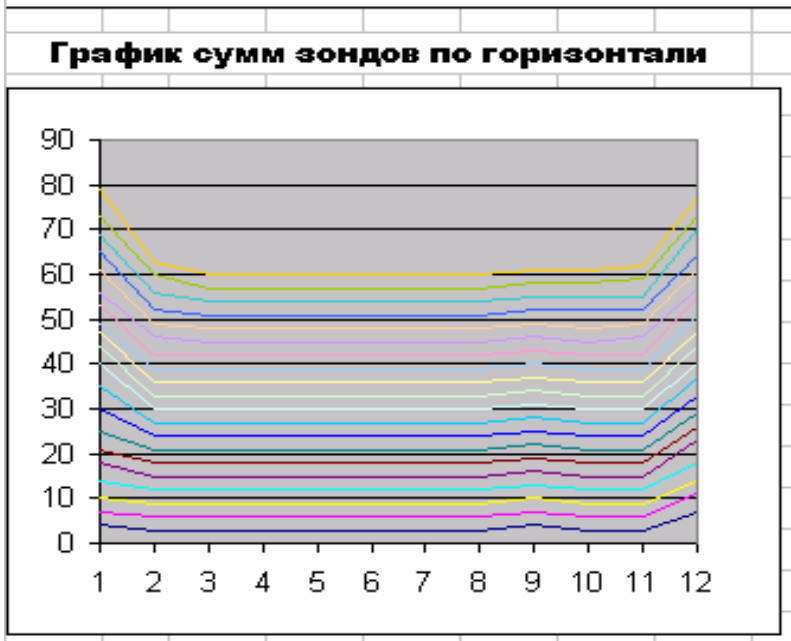


График сумм зондов по горизонтали
90
80
70
60
50
40
30
20
10
0
1 2 3 4 5 6 7 8 9 10 11 12

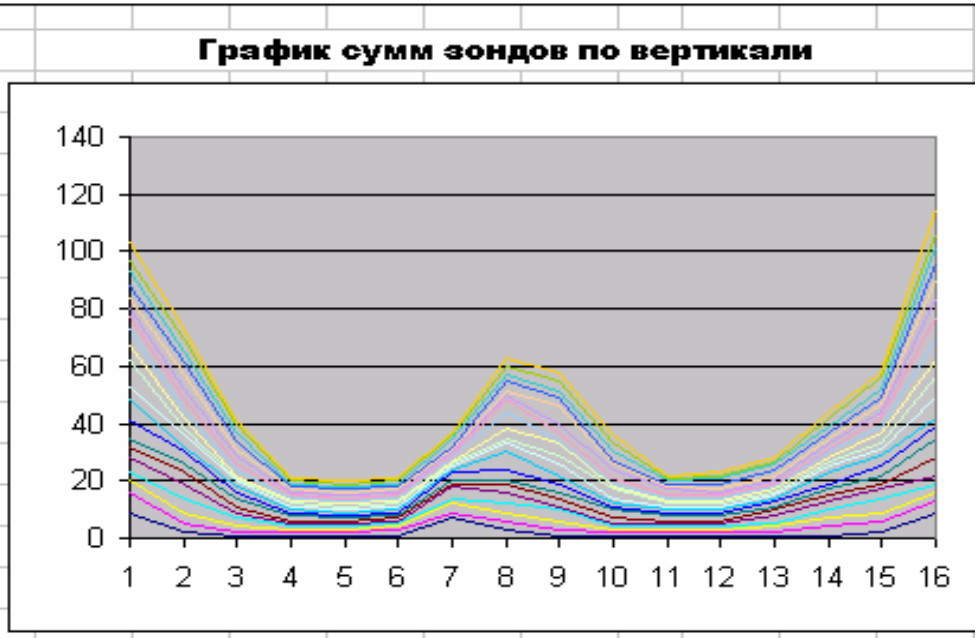
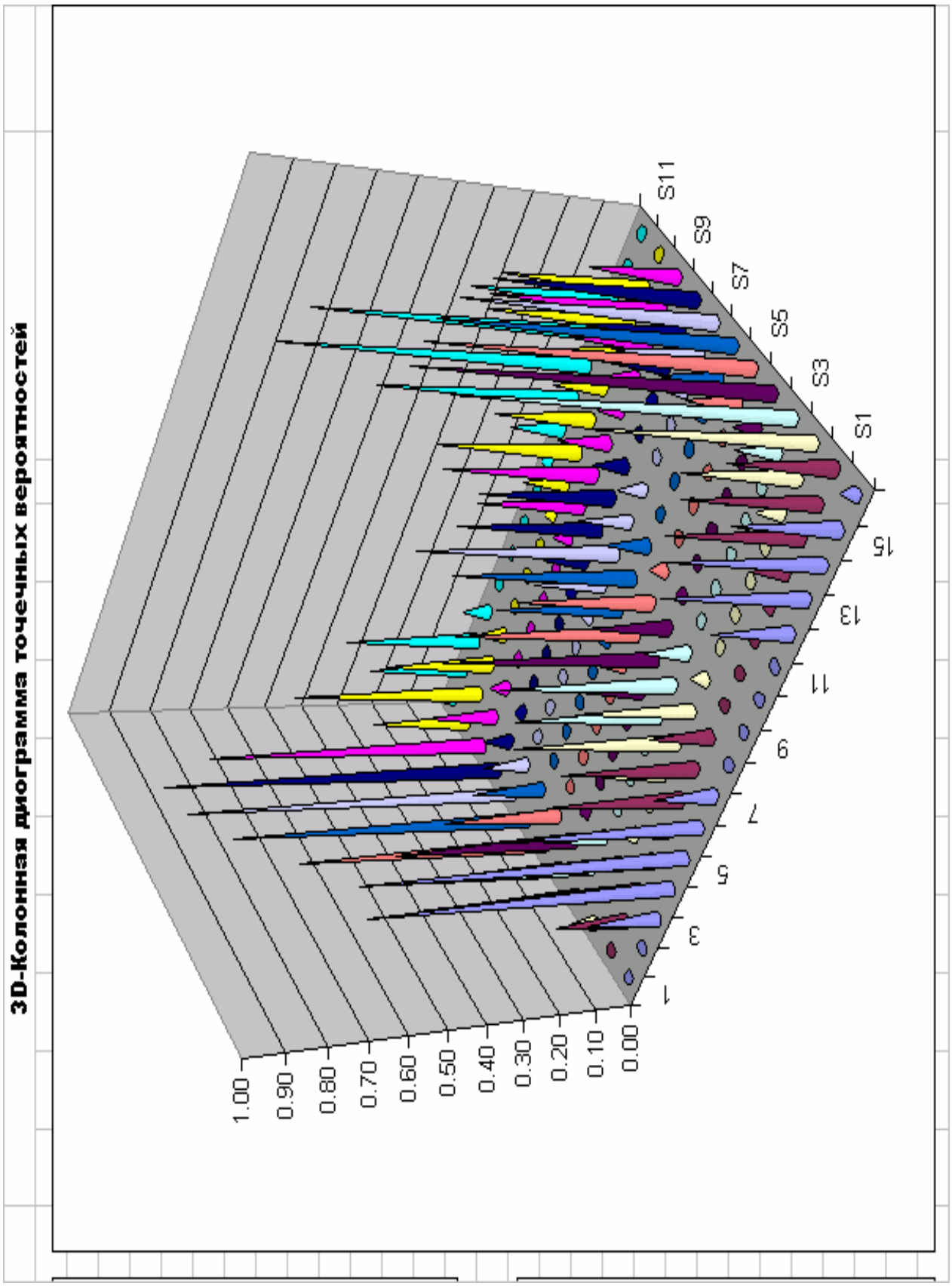
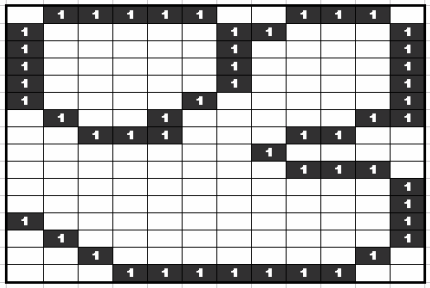
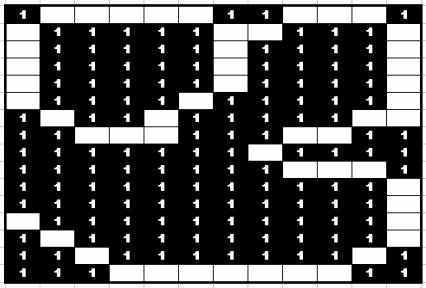


График сумм зондов по вертикали
140
120
100
80
60
40
20
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16


3D-Колонная диограмма точечных вероятностей
S11
S9
S7
S5
S3
S1
1
3
5
7
9
11
13
15
0.00
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00



52



P1

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.00	0.00	0.00	0.03	0.03	0.13	0.25	0.50	0.48	0.20	0.15	0.00	
2	0.00	0.00	0.03	0.05	0.15	0.40	0.33	0.53	0.48	0.08	0.15	0.00	
3	0.00	0.03	0.08	0.18	0.28	0.38	0.28	0.35	0.38	0.10	0.15	0.00	
4	0.03	0.10	0.20	0.25	0.28	0.30	0.08	0.28	0.40	0.08	0.15	0.00	
5	0.13	0.23	0.25	0.25	0.18	0.15	0.03	0.30	0.38	0.08	0.15	0.00	
6	0.25	0.20	0.20	0.18	0.05	0.13	0.05	0.28	0.40	0.05	0.15	0.00	
7	0.15	0.23	0.18	0.08	0.03	0.10	0.03	0.28	0.40	0.05	0.15	0.00	
8	0.28	0.18	0.08	0.00	0.00	0.10	0.03	0.28	0.40	0.05	0.15	0.00	
9	0.13	0.05	0.00	0.00	0.00	0.10	0.03	0.30	0.38	0.05	0.15	0.00	
10	0.05	0.00	0.00	0.00	0.00	0.10	0.03	0.30	0.38	0.05	0.15	0.00	
11	0.00	0.00	0.00	0.00	0.00	0.10	0.03	0.30	0.38	0.05	0.15	0.00	
12	0.00	0.00	0.00	0.00	0.00	0.10	0.03	0.30	0.38	0.05	0.15	0.00	
13	0.00	0.00	0.00	0.00	0.00	0.10	0.05	0.28	0.38	0.05	0.18	0.03	
14	0.03	0.03	0.03	0.00	0.00	0.10	0.05	0.30	0.38	0.10	0.18	0.05	
15	0.00	0.00	0.00	0.03	0.03	0.15	0.10	0.30	0.43	0.10	0.20	0.03	
16	0.10	0.13	0.28	0.55	0.68	0.80	0.83	0.80	0.88	0.90	0.90	0.90	
∑													

P0

	1	2	3	4	5	6	7	8	9	10	11	12	∑
1	0.00	0.00	0.00	0.03	0.03	0.13	0.00	0.00	0.48	0.20	0.15	0.00	Sigma
2	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.53	0.00	0.00	0.00	0.00	10.28
3	0.00	0.00	0.00	0.00	0.00	0.00	0.28	0.00	0.00	0.00	0.00	0.00	
4	0.03	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	Count
5	0.13	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	369.9
6	0.25	0.00	0.00	0.00	0.00	0.13	0.00	0.00	0.00	0.00	0.00	0.00	
7	0.00	0.23	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.15	0.00	
8	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.40	0.05	0.00	0.00	36
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.38	0.05	0.15	0.00	7.113
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7.113
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	
14	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	
16	0.00	0.00	0.00	0.55	0.68	0.80	0.83	0.80	0.88	0.90	0.00	0.00	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.13	0.38	0.55	0.75	0.83	0.85	0.88	0.78	0.43	0.28	0.00	
	0.10	0.28	0.18	0.23	0.08	0.08	0.15	0.08	0.15	0.35	0.20	0.25	
	0.20	0.28	0.18	0.13	0.03	0.08	0.03	0.05	0.08	0.20	0.28	0.40	
	0.25	0.28	0.15	0.13	0.03	0.05	0.00	0.00	0.08	0.18	0.18	0.58	
	0.23	0.20	0.13	0.03	0.03	0.00	0.00	0.00	0.05	0.18	0.15	0.63	
	0.13	0.08	0.08	0.05	0.00	0.00	0.00	0.00	0.08	0.15	0.25	0.53	
	0.08	0.00	0.05	0.08	0.00	0.00	0.00	0.10	0.10	0.28	0.23	0.48	
	0.03	0.03	0.00	0.00	0.03	0.03	0.13	0.08	0.20	0.25	0.18	0.35	
	0.00	0.00	0.00	0.03	0.03	0.15	0.10	0.25	0.18	0.25	0.13	0.20	
	0.00	0.00	0.03	0.05	0.15	0.08	0.28	0.20	0.25	0.15	0.10	0.13	
	0.00	0.00	0.08	0.13	0.08	0.28	0.15	0.15	0.10	0.08	0.08	0.03	
	0.00	0.08	0.15	0.10	0.25	0.20	0.20	0.15	0.08	0.05	0.00	0.03	
	0.08	0.25	0.28	0.43	0.30	0.18	0.13	0.08	0.05	0.03	0.03	0.08	
	0.45	0.35	0.40	0.28	0.20	0.15	0.05	0.00	0.00	0.00	0.08	0.23	
	0.65	0.25	0.30	0.25	0.33	0.30	0.35	0.35	0.30	0.20	0.10	0.43	
	0.63	0.90	0.80	0.75	0.70	0.73	0.70	0.65	0.73	0.80	0.80	0.60	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.13	0.38	0.55	0.75	0.83	0.00	0.00	0.78	0.43	0.28	0.00	Sigama
	0.10	0.00	0.00	0.00	0.00	0.00	0.15	0.08	0.00	0.00	0.00	0.25	15.78
	0.20	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.40	
	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.58	Count
	0.23	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.63	709.9
	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.53	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.23	0.48	
	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.20	0.25	0.00	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.00	0.00	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.15	0.10	45
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	13.65
	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	
	0.00	0.35	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.23	
	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	
	0.00	0.00	0.00	0.75	0.70	0.73	0.70	0.65	0.73	0.80	0.00	0.00	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.03	0.03	0.10	0.28	0.48	0.75	0.93	0.83	0.60	0.28	0.00	
	0.03	0.00	0.08	0.18	0.20	0.30	0.23	0.05	0.13	0.25	0.30	0.28	
	0.03	0.00	0.08	0.23	0.28	0.23	0.18	0.00	0.05	0.25	0.15	0.53	
	0.03	0.00	0.03	0.10	0.20	0.13	0.03	0.00	0.10	0.18	0.18	0.53	
	0.00	0.00	0.00	0.03	0.00	0.03	0.08	0.13	0.08	0.23	0.30	0.33	
	0.00	0.00	0.00	0.00	0.00	0.05	0.13	0.15	0.23	0.38	0.28	0.15	
	0.00	0.00	0.00	0.00	0.05	0.13	0.18	0.30	0.45	0.35	0.13	0.00	
	0.00	0.00	0.00	0.00	0.03	0.03	0.13	0.43	0.53	0.40	0.10	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.03	0.28	0.40	0.38	0.10	
	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.05	0.15	0.45	0.43	
	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.13	0.85	
	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.93	
	0.53	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.18	0.83	
	0.38	0.50	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.35	0.55	
	0.05	0.40	0.50	0.10	0.03	0.00	0.03	0.05	0.18	0.40	0.40	0.13	
	0.00	0.05	0.43	0.90	0.98	1.00	1.00	0.95	0.83	0.50	0.15	0.00	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.03	0.03	0.10	0.28	0.48	0.00	0.00	0.83	0.60	0.28	0.00	Sigama
	0.03	0.00	0.00	0.00	0.00	0.00	0.23	0.05	0.00	0.00	0.00	0.28	18.10
	0.03	0.00	0.00	0.00	0.00	0.00	0.18	0.00	0.00	0.00	0.00	0.53	
	0.03	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.53	Count
	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.33	832.6
	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.15	
	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.13	
	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.53	0.40	46
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.15	0.45	16.01
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.85	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.93	
	0.53	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.83	
	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.55	
	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	
	0.00	0.00	0.00	0.90	0.98	1.00	1.00	0.95	0.83	0.50	0.00	0.00	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.20	0.30	0.25	0.10	0.00	0.00	0.03	0.03	0.03	0.10	0.18	0.05	
	0.23	0.30	0.25	0.10	0.00	0.03	0.03	0.03	0.00	0.15	0.20	0.08	
	0.30	0.25	0.28	0.05	0.03	0.03	0.03	0.00	0.10	0.15	0.25	0.08	
	0.38	0.25	0.25	0.05	0.03	0.03	0.00	0.05	0.05	0.18	0.45	0.08	
	0.43	0.30	0.18	0.05	0.03	0.00	0.05	0.00	0.05	0.23	0.48	0.10	
	0.45	0.40	0.10	0.03	0.00	0.05	0.00	0.00	0.05	0.30	0.50	0.10	
	0.60	0.30	0.05	0.00	0.05	0.00	0.00	0.00	0.05	0.33	0.53	0.10	
	0.65	0.28	0.03	0.05	0.00	0.00	0.00	0.00	0.05	0.33	0.53	0.10	
	0.75	0.30	0.33	0.25	0.25	0.25	0.25	0.25	0.30	0.58	0.60	0.30	
	0.45	0.55	0.43	0.45	0.43	0.43	0.43	0.45	0.45	0.55	0.78	0.43	
	0.18	0.18	0.25	0.25	0.28	0.28	0.28	0.25	0.30	0.53	0.60	0.28	
	0.03	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.10	0.33	0.58	0.15	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.10	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.10	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.10	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.23	0.73	0.65	0.53	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.30	0.25	0.10	0.00	0.00	0.00	0.00	0.03	0.10	0.18	0.00	Sigama
	0.23	0.00	0.00	0.00	0.00	0.00	0.03	0.03	0.00	0.00	0.00	0.08	8.88
	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	
	0.38	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	Count
	0.43	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.10	363.9
	0.45	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.10	
	0.00	0.30	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.53	0.10	
	0.00	0.00	0.03	0.05	0.00	0.00	0.00	0.00	0.05	0.33	0.00	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.00	0.00	0.00	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.45	0.55	0.78	0.00	41
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.28	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15	6.998
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.23	0.73	0.00	0.00	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.00	0.03	0.23	0.53	0.70	0.53	0.33	0.45	0.65	0.75	0.55	
	0.00	0.00	0.13	0.23	0.55	0.40	0.48	0.68	0.55	0.38	0.25	0.08	
	0.00	0.00	0.18	0.38	0.38	0.10	0.03	0.00	0.00	0.00	0.00	0.00	
	0.00	0.03	0.20	0.48	0.33	0.03	0.00	0.00	0.00	0.00	0.00	0.00	
	0.00	0.05	0.18	0.53	0.28	0.00	0.00	0.03	0.03	0.00	0.00	0.00	
	0.03	0.03	0.23	0.50	0.33	0.28	0.40	0.48	0.45	0.15	0.00	0.00	
	0.03	0.03	0.33	0.60	0.53	0.38	0.50	0.43	0.35	0.48	0.23	0.00	
	0.03	0.05	0.23	0.50	0.33	0.35	0.10	0.08	0.18	0.28	0.45	0.23	
	0.00	0.00	0.00	0.08	0.20	0.03	0.00	0.00	0.00	0.10	0.35	0.63	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.88	
	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.90	
	0.35	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.28	0.70	
	0.33	0.38	0.10	0.00	0.00	0.00	0.00	0.00	0.03	0.20	0.28	0.50	
	0.18	0.40	0.30	0.10	0.00	0.00	0.00	0.05	0.20	0.30	0.25	0.33	
	0.05	0.15	0.40	0.38	0.20	0.10	0.08	0.25	0.33	0.30	0.28	0.03	
	0.03	0.08	0.20	0.53	0.80	0.90	0.93	0.70	0.45	0.20	0.03	0.00	

	1	2	3	4	5	6	7	8	9	10	11	12	∑
	0.00	0.00	0.03	0.23	0.53	0.70	0.00	0.00	0.45	0.65	0.75	0.00	Sigama
	0.00	0.00	0.00	0.00	0.00	0.00	0.48	0.68	0.00	0.00	0.00	0.08	15.60
	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	Count
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	37
	0.00	0.00	0.00	0.00	0.00	0.28	0.00	0.00	0.00	0.00	0.00	0.00	
	0.00	0.03	0.00	0.00	0.53	0.00	0.00	0.00	0.00	0.00	0.23	0.00	
	0.00	0.00	0.23	0.50	0.33	0.00	0.00	0.00	0.18	0.28	0.00	0.00	11.1
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	577.2
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.00	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.90	
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70	
	0.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	
	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.33	
	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.28	0.00	
	0.00	0.00	0.00	0.53	0.80	0.90	0.93	0.70	0.45	0.20	0.00	0.00	

Анализ и подсчеты гипер масс для сумматорных матриц (II).

1	2	3	4	5	6	7	8	9	10	11	12	∑
0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.50	0.00	0.00	0.00	0.00	Sum
0.00	0.00	0.03	0.05	0.15	0.40	0.00	0.00	0.48	0.08	0.15	0.00	1.80
0.00	0.03	0.08	0.18	0.28	0.38	0.00	0.35	0.38	0.10	0.15	0.00	
0.00	0.10	0.20	0.25	0.28	0.30	0.00	0.28	0.40	0.08	0.15	0.00	
0.00	0.23	0.25	0.25	0.18	0.15	0.00	0.30	0.38	0.08	0.15	0.00	
0.00	0.20	0.20	0.18	0.05	0.00	0.05	0.28	0.40	0.05	0.15	0.00	
0.15	0.00	0.18	0.08	0.00	0.10	0.03	0.28	0.40	0.05	0.00	0.00	
0.28	0.18	0.00	0.00	0.00	0.10	0.03	0.28	0.00	0.00	0.15	0.00	
0.13	0.05	0.00	0.00	0.00	0.10	0.03	0.00	0.38	0.05	0.15	0.00	
0.05	0.00	0.00	0.00	0.00	0.10	0.03	0.30	0.00	0.00	0.00	0.00	
0.00	0.00	0.00	0.00	0.00	0.10	0.03	0.30	0.38	0.05	0.15	0.00	
0.00	0.00	0.00	0.00	0.00	0.10	0.03	0.30	0.38	0.05	0.15	0.00	
0.00	0.00	0.00	0.00	0.00	0.10	0.05	0.28	0.38	0.05	0.18	0.00	
0.03	0.00	0.03	0.00	0.00	0.10	0.05	0.30	0.38	0.10	0.18	0.00	
0.00	0.00	0.00	0.03	0.03	0.15	0.10	0.30	0.43	0.10	0.00	0.03	
0.10	0.13	0.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.90	0.90	

1	2	3	4	5	6	7	8	9	10	11	12	∑
1	2	3	4					5	6	7	8	
0.00	2.00	2.00	2.03	2.03	2.13	2.03	2.13	2.03	2.20	2.15	0.00	10.28
2.00	0.00	0.03	0.05	0.15	0.40	0.15	0.40	0.15	0.08	0.15	2.00	
2.00	0.03	0.08	0.18	0.28	0.38	0.28	0.38	0.28	0.10	0.15	2.00	16
2.03	0.10	0.20	0.25	0.28	0.30	0.28	0.30	0.28	0.08	0.15	2.00	
2.13	0.23	0.25	0.25	0.18	0.15	0.18	0.15	0.18	0.08	0.15	2.00	1.80
2.25	0.20	0.20	0.18	0.05	2.13	0.05	2.13	0.05	0.05	0.15	2.00	
0.15	2.23	0.18	0.08	2.03	0.10	2.03	0.10	2.03	0.05	2.15	2.00	-7.53
0.28	0.18	2.08	2.00	2.00	0.10	2.00	0.10	2.00	2.05	0.15	0.00	
0.13	0.05	0.00	0.00	0.00	0.10	0.00	0.10	0.00	0.05	0.15	0.00	
0.05	0.00	0.00	0.00	0.00	0.10	0.00	0.10	0.00	2.05	2.15	0.00	
0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.10	0.00	0.05	0.15	2.00	
0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.10	0.00	0.05	0.15	2.00	
2.00	0.00	0.00	0.00	0.00	0.10	0.00	0.10	0.00	0.05	0.18	2.03	
0.03	2.03	0.03	0.00	0.00	0.10	0.00	0.10	0.00	0.10	0.18	2.05	
0.00	0.00	2.00	0.03	0.03	0.15	0.03	0.15	0.03	0.10	2.20	0.03	
0.10	0.13	0.28	2.55	2.68	2.80	2.68	2.80	2.68	2.90	0.90	0.90	

0.00	0.00	0.00	0.00	0.00	0.00	0.85	0.88	0.00	0.00	0.00	0.00	Sum
0.00	0.28	0.18	0.23	0.08	0.08	0.00	0.00	0.15	0.35	0.20	0.00	6.10
0.00	0.28	0.18	0.13	0.03	0.08	0.00	0.05	0.08	0.20	0.28	0.00	
0.00	0.28	0.15	0.13	0.03	0.05	0.00	0.00	0.08	0.18	0.18	0.00	
0.00	0.20	0.13	0.03	0.03	0.00	0.00	0.00	0.05	0.18	0.15	0.00	
0.00	0.08	0.08	0.05	0.00	0.00	0.00	0.00	0.08	0.15	0.25	0.00	
0.08	0.00	0.05	0.08	0.00	0.00	0.00	0.10	0.10	0.28	0.00	0.00	
0.03	0.03	0.00	0.00	0.00	0.03	0.13	0.08	0.00	0.00	0.18	0.35	
0.00	0.00	0.00	0.03	0.03	0.15	0.10	0.00	0.18	0.25	0.13	0.20	
0.00	0.00	0.03	0.05	0.15	0.08	0.28	0.20	0.00	0.00	0.00	0.13	
0.00	0.00	0.08	0.13	0.08	0.28	0.15	0.15	0.10	0.08	0.08	0.00	
0.00	0.08	0.15	0.10	0.25	0.20	0.20	0.15	0.08	0.05	0.00	0.00	
0.00	0.25	0.28	0.43	0.30	0.18	0.13	0.08	0.05	0.03	0.03	0.00	
0.45	0.00	0.40	0.28	0.20	0.15	0.05	0.00	0.00	0.00	0.08	0.00	
0.65	0.25	0.00	0.25	0.33	0.30	0.35	0.35	0.30	0.20	0.00	0.43	
0.63	0.90	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.80	0.60	

0.00	2.13	2.38	2.55	2.75	2.83	0.85	0.88	2.78	2.43	2.28	0.00	15.78
2.10	0.28	0.18	0.23	0.08	0.08	2.15	2.08	0.15	0.35	0.20	2.25	
2.20	0.28	0.18	0.13	0.03	0.08	2.03	0.05	0.08	0.20	0.28	2.40	7
2.25	0.28	0.15	0.13	0.03	0.05	2.00	0.00	0.08	0.18	0.18	2.58	
2.23	0.20	0.13	0.03	0.03	0.00	2.00	0.00	0.05	0.18	0.15	2.63	6.10
2.13	0.08	0.08	0.05	0.00	2.00	0.00	0.00	0.08	0.15	0.25	2.53	
0.08	2.00	0.05	0.08	2.00	0.00	0.00	0.10	0.10	0.28	2.23	2.48	2.68
0.03	0.03	2.00	2.00	2.03	0.03	0.13	0.08	2.20	2.25	0.18	0.35	
0.00	0.00	0.00	0.03	0.03	0.15	0.10	2.25	0.18	0.25	0.13	0.20	
0.00	0.00	0.03	0.05	0.15	0.08	0.28	0.20	2.25	2.15	2.10	0.13	
0.00	0.00	0.08	0.13	0.08	0.28	0.15	0.15	0.10	0.08	0.08	2.03	
0.00	0.08	0.15	0.10	0.25	0.20	0.20	0.15	0.08	0.05	0.00	2.03	
2.08	0.25	0.28	0.43	0.30	0.18	0.13	0.08	0.05	0.03	0.03	2.08	
0.45	2.35	0.40	0.28	0.20	0.15	0.05	0.00	0.00	0.00	0.08	2.23	
0.65	0.25	2.30	0.25	0.33	0.30	0.35	0.35	0.30	0.20	2.10	0.43	
0.63	0.90	0.80	2.75	2.70	2.73	2.70	2.65	2.73	2.80	0.80	0.60	

0.00	0.00	0.00	0.00	0.00	0.00	0.75	0.93	0.00	0.00	0.00	0.00	Sum
0.00	0.00	0.08	0.18	0.20	0.30	0.00	0.00	0.13	0.25	0.30	0.00	1.68
0.00	0.00	0.08	0.23	0.28	0.23	0.00	0.00	0.05	0.25	0.15	0.00	
0.00	0.00	0.03	0.10	0.20	0.13	0.00	0.00	0.10	0.18	0.18	0.00	
0.00	0.00	0.00	0.03	0.00	0.03	0.00	0.13	0.08	0.23	0.30	0.00	
0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.15	0.23	0.38	0.28	0.00	
0.00	0.00	0.00	0.00	0.00	0.13	0.18	0.30	0.45	0.35	0.00	0.00	
0.00	0.00	0.00	0.00	0.00	0.03	0.13	0.43	0.00	0.00	0.10	0.00	
0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.28	0.40	0.38	0.10	
0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.43	
0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.13	0.00	
0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.00	
0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.18	0.00	
0.38	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.35	0.00	
0.05	0.40	0.00	0.10	0.03	0.00	0.03	0.05	0.18	0.40	0.00	0.13	
0.00	0.05	0.43	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15	0.00	

0.00	2.03	2.03	2.10	2.28	2.48	0.75	0.93	2.83	2.60	2.28	0.00	18.10
2.03	0.00	0.08	0.18	0.20	0.30	2.23	2.05	0.13	0.25	0.30	2.28	
2.03	0.00	0.08	0.23	0.28	0.23	2.18	0.00	0.05	0.25	0.15	2.53	6
2.03	0.00	0.03	0.10	0.20	0.13	2.03	0.00	0.10	0.18	0.18	2.53	
2.00	0.00	0.00	0.03	0.00	0.03	2.08	0.13	0.08	0.23	0.30	2.33	1.68
2.00	0.00	0.00	0.00	0.00	2.05	0.13	0.15	0.23	0.38	0.28	2.15	
0.00	2.00	0.00	0.00	2.05	0.13	0.18	0.30	0.45	0.35	2.13	2.00	10.43
0.00	0.00	2.00	2.00	2.03	0.03	0.13	0.43	2.53	2.40	0.10	0.00	
0.00	0.00	0.00	0.00	0.00	0.00	0.03	2.03	0.28	0.40	0.38	0.10	
0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.03	2.05	2.15	2.45	0.43	
0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.13	2.85	
0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	2.93	
2.53	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.18	2.83	
0.38	2.50	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.35	2.55	
0.05	0.40	2.50	0.10	0.03	0.00	0.03	0.05	0.18	0.40	2.40	0.13	
0.00	0.05	0.43	2.90	2.98	3.00	3.00	2.95	2.83	2.50	0.15	0.00	

0.20	0.00	0.00	0.00	0.00	0.00	0.03	0.03	0.00	0.00	0.00	0.05	Sum
0.00	0.30	0.25	0.10	0.00	0.03	0.00	0.00	0.00	0.15	0.20	0.00	7.13
0.00	0.25	0.28	0.05	0.03	0.03	0.00	0.00	0.10	0.15	0.25	0.00	
0.00	0.25	0.25	0.05	0.03	0.03	0.00	0.05	0.05	0.18	0.45	0.00	
0.00	0.30	0.18	0.05	0.03	0.00	0.00	0.00	0.05	0.23	0.48	0.00	
0.00	0.40	0.10	0.03	0.00	0.00	0.00	0.00	0.05	0.30	0.50	0.00	
0.60	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.05	0.33	0.00	0.00	
0.65	0.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.53	0.10	
0.75	0.30	0.33	0.25	0.25	0.25	0.25	0.00	0.30	0.58	0.60	0.30	
0.45	0.55	0.43	0.45	0.43	0.43	0.43	0.45	0.00	0.00	0.00	0.43	
0.18	0.18	0.25	0.25	0.28	0.28	0.28	0.25	0.30	0.53	0.60	0.00	
0.03	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.10	0.33	0.58	0.00	
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.00	
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	0.00	
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.00	0.10	
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.65	0.53	

0.20	2.30	2.25	2.10	2.00	2.00	0.03	0.03	2.03	2.10	2.18	0.05	8.88
2.23	0.30	0.25	0.10	0.00	0.03	2.03	2.03	0.00	0.15	0.20	2.08	
2.30	0.25	0.28	0.05	0.03	0.03	2.03	0.00	0.10	0.15	0.25	2.08	11
2.38	0.25	0.25	0.05	0.03	0.03	2.00	0.05	0.05	0.18	0.45	2.08	
2.43	0.30	0.18	0.05	0.03	0.00	2.05	0.00	0.05	0.23	0.48	2.10	7.13
2.45	0.40	0.10	0.03	0.00	2.05	0.00	0.00	0.05	0.30	0.50	2.10	
0.60	2.30	0.05	0.00	2.05	0.00	0.00	0.00	0.05	0.33	2.53	2.10	-9.25
0.65	0.28	2.03	2.05	2.00	0.00	0.00	0.00	2.05	2.33	0.53	0.10	
0.75	0.30	0.33	0.25	0.25	0.25	0.25	2.25	0.30	0.58	0.60	0.30	
0.45	0.55	0.43	0.45	0.43	0.43	0.43	0.45	2.45	2.55	2.78	0.43	
0.18	0.18	0.25	0.25	0.28	0.28	0.28	0.25	0.30	0.53	0.60	2.28	
0.03	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.10	0.33	0.58	2.15	
2.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	2.10	
0.00	2.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	0.50	2.10	
0.00	0.00	2.00	0.00	0.00	0.00	0.00	0.00	0.05	0.35	2.50	0.10	
0.00	0.00	0.00	2.00	2.00	2.00	2.00	2.03	2.23	2.73	0.65	0.53	

0.00	0.00	0.00	0.00	0.00	0.00	0.53	0.33	0.00	0.00	0.00	0.55	Sum
0.00	0.00	0.13	0.23	0.55	0.40	0.00	0.00	0.55	0.38	0.25	0.00	4.80
0.00	0.00	0.18	0.38	0.38	0.10	0.00	0.00	0.00	0.00	0.00	0.00	
0.00	0.03	0.20	0.48	0.33	0.03	0.00	0.00	0.00	0.00	0.00	0.00	
0.00	0.05	0.18	0.53	0.28	0.00	0.00	0.03	0.03	0.00	0.00	0.00	
0.00	0.03	0.23	0.50	0.33	0.00	0.40	0.48	0.45	0.15	0.00	0.00	
0.03	0.00	0.33	0.60	0.00	0.38	0.50	0.43	0.35	0.48	0.00	0.00	
0.03	0.05	0.00	0.00	0.00	0.35	0.10	0.08	0.00	0.00	0.45	0.23	
0.00	0.00	0.00	0.08	0.20	0.03	0.00	0.00	0.00	0.10	0.35	0.63	
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.88	
0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	
0.35	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.28	0.00	
0.00	0.38	0.10	0.00	0.00	0.00	0.00	0.00	0.03	0.20	0.28	0.00	
0.18	0.00	0.30	0.10	0.00	0.00	0.00	0.05	0.20	0.30	0.25	0.00	
0.05	0.15	0.00	0.38	0.20	0.10	0.08	0.25	0.33	0.30	0.00	0.03	
0.03	0.08	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	

0.00	2.00	2.03	2.23	2.53	2.70	0.53	0.33	2.45	2.65	2.75	0.55	15.60
2.00	0.00	0.13	0.23	0.55	0.40	2.48	2.68	0.55	0.38	0.25	2.08	
2.00	0.00	0.18	0.38	0.38	0.10	2.03	0.00	0.00	0.00	0.00	2.00	15
2.00	0.03	0.20	0.48	0.33	0.03	2.00	0.00	0.00	0.00	0.00	2.00	
2.00	0.05	0.18	0.53	0.28	0.00	2.00	0.03	0.03	0.00	0.00	2.00	4.80
2.03	0.03	0.23	0.50	0.33	2.28	0.40	0.48	0.45	0.15	0.00	2.00	
0.03	2.03	0.33	0.60	2.53	0.38	0.50	0.43	0.35	0.48	2.23	2.00	-4.20
0.03	0.05	2.23	2.50	2.33	0.35	0.10	0.08	2.18	2.28	0.45	0.23	
0.00	0.00	0.00	0.08	0.20	0.03	0.00	2.00	0.00	0.10	0.35	0.63	
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.00	2.00	2.13	0.88	
0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	2.90	
0.35	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.28	2.70	
2.33	0.38	0.10	0.00	0.00	0.00	0.00	0.00	0.03	0.20	0.28	2.50	
0.18	2.40	0.30	0.10	0.00	0.00	0.00	0.05	0.20	0.30	0.25	2.33	
0.05	0.15	2.40	0.38	0.20	0.10	0.08	0.25	0.33	0.30	2.28	0.03	
0.03	0.08	0.20	2.53	2.80	2.90	2.93	2.70	2.45	2.20	0.03	0.00	