МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет
«Московский институт электронной техники»

Факультет микроприборов и технической кибернетики
Кафедра информатики и программного обеспечения вычислительных систем

Казначеев Александр Александрович

Бакалаврская работа
по направлению 09.03.04 «Программная инженерия»

Разработка программного модуля анализа данных для веб-сайтов с использованием технологий нейронных сетей

Студент _____ Казначеев А.А.
Научный руководитель,
к. т. н, доцент _____ Слюсарь В.В.

Москва 2016

Содержание

Перечень сокращений 4
Введение 5
1 ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ 6
1.1 Актуальность выбранной темы 6
1.2 Анализ существующих программных решений аналогичного функционала. 6
1.3 Цель и задачи выпускной квалификационной работы 8
1.4 Исследование предметной области 9
1.4.1 Архитектура сетей 11
1.4.2 Представление знаний 12
1.4.3 Алгоритм обратного распространения ошибки 14
1.4.4 Нормализация входных данных 16
1.4.5 Целевые значения 16
1.4.6 Скорость обучения 17
1.5.7 Преимущества и ограничения обучения методом обратного распространения 17
1.4.8 Вычислительная эффективность алгоритма обратного распространения 18
1.5 Концептуальная модель предметной области 19
1.6 Потребности потенциальных потребителей 20
1.7 Функциональные требования, предъявляемые к ПМ АДН 20
Выводы исследовательского раздела 21
2 КОНСТРУКТОРСКИЙ РАЗДЕЛ 22
2.1 Структура входных и выходных данных 22
2.2 Выбор инструментальных средств разработки 22
2.2.1 Выбор языка программирования. 22
2.2.2 Выбор системы управления содержимым. 24
2.2.3 Выбор среды разработки и отладки. 26
2.3 Программная архитектура и алгоритм работы 28
2.4 Проектирование БД для ПМ АДН 35
2.4.1 Инфологическая модель предметной области 35
2.4.2 Даталогическая модель 37
2.5 Требования к надежности 38
2.6 Требования к информационной и программной совместимости 38

2.7 Разработка пользовательского интерфейса ... 39
Выводы конструкторского раздела ... 43
3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ ... 44
3.1 Технологии, лежащие в основе веб-сайтов ... 44
3.1.1 Всемирная паутина ... 44
3.1.2 Язык разметки HTML ... 45
3.1.3 JavaScript и его особенности ... 46
3.1.4 Каскадные таблицы стилей ... 47
3.2 Системы управления содержимым ... 48
3.2.1 Основные сведения ... 48
3.2.2 Преимущества и недостатки использования систем управления содержимым ... 49
3.3 Язык PHP и его особенности ... 49
3.3.1 Основы синтаксиса ... 49
3.3.2 Использование ООП в PHP ... 50
3.4 Система «1С-Битрикс» ... 51
3.4.1 Программная платформа «Bitrix Framework» ... 51
3.4.2 Хранение данных в системе «1С-Битрикс». Информационные блоки ... 53
3.4.3 Взаимодействие с базами данных в системе «1С-Битрикс» ... 53
3.4.4 Кеширование в системе «1С-Битрикс» ... 54
3.4.5 Особенности установки модуля в системе «1С-Битрикс» ... 56
3.5 Построение графика функции и прогнозируемого значения ... 56
3.6 Тестирование и отладка ПМ АДН ... 57
3.6.1 Выбор метода тестирования ... 58
3.6.2 Алгоритм тестирования модуля ... 60
3.6.3 Сценарий тестирования «задача-XOR» ... 62
Выводы технологического раздела ... 64
Заключение ... 65
Список используемой литературы ... 66
Приложение 1 ... 71
Приложение 2 ... 123

Перечень сокращений

БД – база данных;

ВКР – выпускная квалификационная работа;

ПМ – программный модуль;

ООП – объектно-ориентированное программирование;

СУБД – система управления базами данных;

API – Application Programming Interface;

CMS – Content Management System;

FANN – Fast Artificial Neural Network;

GPL – General Public License

UML – Unified Modeling Language.

Введение

С каждым годом веб-сфера получает все большее распространение, соответственно растут и объемы обрабатываемой информации. Чаще всего получаемые от пользователей данные содержат шумы и трудно выявляемые закономерности. Подобные проблемы удобно решать с помощью нейронных сетей, но существующие библиотеки имеют либо большую стоимость, либо долгий процесс установки, либо неудобный интерфейс использования.

Актуальность задачи заключается в потребности наличия простого и удобного в использовании инструмента для работы с нейронными сетями интегрированного в систему управления сайтом.

Цель выполнения данной работы в улучшении эффективности анализа данных веб-сайтов путем реализация программного модуля, основанного на технологии нейронных сетей.

Практическая значимость данной разработки - создание решения, которое позволит ускорить и упростить применение технологий нейронных сетей в системе управления содержимым.

Пояснительная записка состоит из введения, исследовательского, конструкторского и технологического разделов, заключения, списка литературы и приложений.

В исследовательском разделе рассматриваются актуальность выбранной темы, исследование предметной области, анализ существующих программных решений.

В конструкторской части были проанализированы функциональные требования к программному модулю анализа данных для веб-сайтов с использованием технологий нейронных сетей (ПМ АДН), разработаны структуры входных и выходных данных, разработаны архитектуры и алгоритм работы программного модуля, проведен обзор пользовательского интерфейса, выбраны инструменты разработки ПМ АДН.

В технологическом разделе рассматриваются: особенности программирования, архитектура взаимодействия, способы отладки и тестирования.

В приложении 1 содержится программный код, в приложении 2 руководство программиста.

1 ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ

В исследовательском разделе рассматриваются актуальность выбранной темы, исследование предметной области, анализ существующих программных решений.

1.1 Актуальность выбранной темы

Технологии анализа данных с каждым днем набирают все большие обороты. Для анализа увеличивающихся объемов данных существует два подхода: «жесткий», основными технологиями которого являются теория автоматов и алгоритмов, и «мягкий», объединяющий в общий класс неточные, приближённые методы решения задач. «Мягкие» методы имеют преимущества при решении задач со слабо структурированной информацией, какой в большинстве своем являются данные получаемые веб-сайтом от пользователя.

В настоящее время получили популярность технологии нейронных сетей, которые входят в «мягкие» методы анализа данных. Их основные преимущества: решение задач при неизвестных закономерностях, устойчивость к шумам, адаптирование под изменения окружающей среды, отказоустойчивость. Все эти особенности удачно вписываются в концепцию современного интернета. Используя преимущества нейронных сетей, становится возможным создать эффективный инструмент для решения задач классификации и прогнозирования данных [19].

1.2 Анализ существующих программных решений аналогичного функционала.

«FANN» представляет собой библиотеку, которая реализована на языке C, для быстрого построения многослойных нейронных сетей, является продуктом с открытым и свободно распространяемым кодом. К преимуществам можно отнести: кроссплатформенность, поддержку плавающей и фиксированной точки, простоту в использовании, универсальность, хорошую документацию, высокую скорость работы. Реализация библиотеки доступна больше, чем на 20 языках. Содержит большое количество примеров. К минусам можно отнести сложность установки, связанную с необходимостью поддерживать большое количество платформ [44].

«ANN» реализует топологию нейронной сети многослойный персептрон для среды программирования PHP. Исходный код основан на работе Эдди Янг в 2002 г. С последующей доработкой Томасом Вином. К плюсам данного решения можно отнести графическое представление топологии сети, систему вывода данных сети, а именно весов и ошибок. Основная задача данного продукта – это классификация данных [41].

«1С-Битрикс BigData» - облачный сервис персонализации, являющийся составной частью платформы «1С-Битрикс». Сервис повышает качество управления, уровень продаж и конверсию в интернет-магазине. Обладает достаточно небольшим количеством команд, в связи со своей ориентированностью на работу интернет – магазина [39].

Обзор выше перечисленных программных решений представлен в таблице 1.

Таблица 1 - Обзор существующих программных решений

Параметры	FANN(Fast Artificial Neural Network)[1]	ANN[2]	Big Data Bitrix[3]	ПМ АДН
Необходимость установки	+	–	–	–
Модуль для CMS	–	–	+	+
Поддержка языков	ruby,php,python,c # и др.	PHP	PHP	PHP
Решаемые задачи	Классификация и прогнозирование	Классификация	Предоставление персональных рекомендаций	Классификация и прогнозирование

Продолжение таблицы 1

Параметры	FANN(Fast Artificial Neural Network)[1]	ANN[2]	Big Data Bitrix[3]	ПМ АДН
Наличие визуального отображения	+	+	–	–
Возможность получения весов сети и ошибок нейронов	+	+	–	+

Источники:

[1] http://leenissen.dk/fann

[2] http://ann.thwien.de/

[3] http://dev.1c-bitrix.ru

Условные обозначения:

+ указанная возможность присутствует

– указанная возможность отсутствует

В итоге было принято решение написать модуль, реализующий функции нейронных сетей, что позволит совместить все преимущества рассмотренных решений, а именно совместить легкость использования встроенного решения Big Data Bitrix и предоставляемые функции из сторонних библиотек [1].

1.3 Цель и задачи выпускной квалификационной работы

Цель ВКР: повышение эффективности анализа данных веб-сайтов путем реализация программного модуля, основанного на технологии нейронных сетей.

Задачи ВКР:

1) исследование предметной области;
2) сравнительный анализ существующих программных решений;

3) выбор инструментальных средств и среды разработки;
4) разработка схемы данных ПМ АДН;
5) разработка схем алгоритмов ПМ АДН;
6) программная реализация ПМ АДН;
7) разработка пользовательского интерфейса ПМ АДН;
8) отладка и тестирование ПМ АДН;
9) разработка руководства программиста.

1.4 Исследование предметной области

Нейронные сети построены на основе человеческого мозга, представляющий собой чрезвычайно сложный, нелинейный, параллельный компьютер. Он обладает способностью организовывать свои структурные компоненты, называемые нейронами, так, чтобы они выполняли конкретные задачи во много раз быстрее, чем могут позволить самые быстродействующие современные компьютеры.

Нейронная сеть – это распределенный параллельный процессор, состоящий из элементарных единиц обработки информации, накапливающих экспериментальные знания и предоставляющих их для последующей обработки. Нейронная сеть сходна с мозгом с двух точек зрения.

1) Знания поступают в нейронную сеть из окружающей среды и используются в процессе обучения.
2) Для накопления знаний применяются связи между нейронами, называемые синаптическими весами.

Процедура, используемая для процесса обучения, называется алгоритмом обучения. Эта процедура выстраивает в определенном порядке синаптические веса нейронной сети для обеспечения необходимой структуры взаимосвязей нейронов.

Нейрон представляет собой единицу обработки информации в нейронной сети. В модели нейрона три основных элемента:

1) Набор синапсов или связей, каждый из которых характеризуется своим весом или силой.
2) Сумматор складывает входные сигналы, взвешенные относительно соответствующих синапсов нейрона.

3) Функция активации ограничивает амплитуду выходного сигнала.

В математическом представлении функционирование нейрона можно описать следующей парой уравнений:

$$v_k = \sum_{j=1}^{m} w_{kj} x_j, \tag{1}$$

$$y_k = \varphi(v_k) \tag{2}$$

где $x_1, x_2, \dots, x_m -$ входные сигналы; $w_{k1}, w_{k2}, \dots, w_{km}, -$ синаптические веса нейрона k; $v_k -$ линейная комбинация входных воздействий; $\varphi(v) -$функция активации; $y_k -$выходной сигнал нейрона.

Функции активации, представленные в формулах как $\varphi(v)$, определяют выходной сигнал нейрона в зависимости от v. Можно выделить три основных типа функций активации [13]:

1) Функция единичного скачка

$$\varphi(v) = \begin{cases} 1, \text{если } v \geq 0 \\ 0, \text{если } v < 0 \end{cases} \tag{3}$$

2) Кусочно-линейная функция

$$\varphi(v) = \begin{cases} 1, \ v \geq +\frac{1}{2} \\ |v|, +\frac{1}{2} > v > -\frac{1}{2} \\ 0, \ v \leq -\frac{1}{2} \end{cases} \tag{4}$$

3) Сигмоидальная функция

$$\varphi(v) = \frac{1}{1+e^{-av}} \tag{5}$$

Где $a -$параметр наклона сигмоидальной функции,

Область значений функций активации, определенных этими формулами, представляет собой отрезок от 0 до +1. Однако иногда требуется функция активации,

имеющая область значений от -1 до +1. В этом случае функция активации должна быть симметрична относительно начала координат. В таких случаях следует использовать сигмоидальную функцию в виде гиперболического тангенса:

$$\varphi(v) = \mathrm{th}(v) \tag{6}$$

1.4.1 Архитектура сетей

Для решения задач прогнозирования и классификации широко применяются следующие архитектуры нейронных сетей:

1) Однослойные сети прямого распространения. В многослойной сети нейроны располагаются по слоям. В простейшем случае в такой сети существует входной слой узлов источника, информация от которого передается на выходной слой нейронов, но не наоборот. Такая сеть называется сетью прямого распространения или ацикличной сетью. Также сеть, состоящая из одного уровня нейронов, называется однослойной сетью, при этом под единственным слоем подразумевается слой вычислительных элементов (нейронов). При подсчете числа слоев мы не принимаем во внимание узлы источника, так как они не выполняют никаких вычислений.
2) Многослойные сети прямого распространения. Другой класс нейронных сетей прямого распространения характеризуется наличием одного или нескольких скрытых слоев, узлы которых называются скрытыми нейронами, или скрытыми элементами. Функция последних заключается в посредничестве между внешним входным сигналом и выходом нейронной сети. Добавляя один или несколько скрытых слоев, мы можем выделить статистики высокого порядка. Такая сеть позволяет выделять глобальные свойства данных с помощью локальных соединений за счет наличие дополнительных синаптических связей и повышения уровня взаимодействия нейронов[38]. Способность скрытых нейронов выделять статистические зависимости высокого порядка особенно существенна, когда размер входного слоя достаточно велик. Узлы источника входного слоя сети формируют соответствующие элементы шаблона активации (входной вектор), которые составляют входной сигнал, поступающий на нейроны (вычислительные

элементы) второго слоя. Выходные сигналы второго слоя используются в качестве входных для третьего слоя и т.д. Набор выходных сигналов нейронов выходного слоя сети определяет общий отклик сети на данный входной образ, сформированный узлами источника входного слоя. Нейронная сеть считается полносвязной, если все узлы конкретного слоя соединены со всеми узлами смежных слоев [2].

3) Особое семейство образуют сети с радиальной базисной функцией (РБФ), в которых нейроны реализуют функции, радиально изменяющиеся вокруг выбранного центра и принимающие ненулевые значения только в окрестности этого центра [18]. Сети РБФ имеют ряд преимуществ перед рассмотренными многослойными сетями прямого распространения. Во-первых, они моделируют произвольную нелинейную функцию с помощью всего одного промежуточного слоя, тем самым, избавляя разработчика от необходимости решать вопрос о числе слоев. Во-вторых, параметры линейной комбинации в выходном слое можно полностью оптимизировать с помощью хорошо известных методов линейной оптимизации, которые работают быстро и не испытывают трудностей с локальными минимумами, так мешающими при обучении с использованием алгоритма обратного распространения ошибки. Поэтому сеть РБФ обучается очень быстро - на порядок быстрее, чем с использованием алгоритма ОР (обратного распространения). Недостатки сетей РБФ: данные сети обладают плохими экстраполирующими свойствами и получаются весьма громоздкими при большой размерности вектора входов [16].

Для решения поставленной задачи следует использовать многослойный персептрон, так как он представляет собой универсальное решение, а также имеет хорошо изученный и легкий в использовании алгоритм обучения.

1.4.2 Представление знаний

Основной задачей нейронной сети является наилучшее обучение модели окружающего мира для решения поставленной задачи. Знания о мире включают два типа информации:

1) Известное состояние окружающего мира, представленное имеющимися в наличии достоверными фактами. Такая информация называется априорной.
2) Наблюдения за окружающим миром (измерения), полученные с помощью сенсоров, адаптированных для конкретных условий, в которых должна функционировать данная нейронная сеть. Обычно такие измерения в значительной мере зашумлены, что потенциально может стать источником ошибок. В любом случае измерения, полученные таким способом, формируют множество информации, примеры из которого используются для обучения нейронной сети.

Примеры могут быть маркированными и немаркированными. В маркированных примерах входному сигналу соответствует желаемый отклик. Немаркированные примеры состоят из нескольких различных реализаций одного входного сигнала. В любом случае набор примеров, будь то маркированных или нет, представляет собой знания об интересующей предметной области, на основании которых и проводится обучение нейронной сети.

Множество пар сигналов вход-выход, каждая из которых состоит из входного сигнала и соответствующего ему желаемого выхода, называются обучающими данными или обучающей выборкой.

Самым важным свойством нейронных сетей является их способность обучаться на основе данных окружающей среды и в результате обучения повышать свою производительность. Повышение производительности происходит со временем в соответствии с определенными правилами. Обучение нейронной сети происходит посредством интерактивного процесса корректировки синаптических весов. В идеальном случае нейронная сеть получает знания об окружающей среде на каждой итерации процесса обучения.

Обучение это процесс, в котором свободные параметры нейронной сети настраиваются посредством моделирования среды, в которую эта сеть встроена. Тип обучения определяется способом подстройки этих параметров.

Это определение процесса обучения предполагает следующую последовательность событий:

1) в нейронную сеть поступают стимулы из внешней среды.
2) в результате этого изменяются свободные параметры нейронной сети.

3) после изменения внутренней структуры нейронная сеть отвечает на возбуждения уже иным образом.

Данный список называется алгоритмом обучения.

1.4.3 Алгоритм обратного распространения ошибки

Предложенный в середине 1980х годов алгоритм обратного распространения ошибки стал одним из ведущих факторов развития технологий нейронных сетей до современного состояния, т.к. являлся эффективным способом обучения искусственной нейронной сети достаточно произвольной структуры. Для сети архитектуры многослойный персептрон приведем шаги алгоритма [11]:

1) Инициализация весов (веса всех связей инициализируются случайными небольшими значениями).
2) До тех пор пока условие прекращения работы алгоритма неверно, выполняются шаги 3 — 8.
3) Для каждой пары {данные, целевое значение} выполняются шаги 4 — 7.
4) Каждый входной нейрон отправляет полученный сигнал x_i всем нейронам в следующем слое (скрытом). Каждый k-ый нейрон суммирует взвешенные входящие сигналы:

$$v_k = \sum_{j=1}^{m} w_{kj} x_j, \tag{7}$$

где $x_1, x_2, \dots, x_m$ – входные сигналы; $w_{k1}, w_{k2}, \dots, w_{km}$, – синаптические веса нейрона k; v_k – линейная комбинация входных воздействий. И применяет активационную функцию:

$$y_k = \varphi(v_k) \tag{8}$$

где $\varphi(v)$ – функция активации; y_k – выходной сигнал нейрона. После чего посылает результат всем элементам следующего слоя.

5) Каждый выходной нейрон получает целевое значение — то выходное значение, которое является правильным для данного входного сигнала, и вычисляет ошибку:

$$\sigma_k = (t_k - y_k) * \varphi'(v_k) \tag{9}$$

где σ_k −ошибка k-го нейрона; t_k −ожидаемый выход нейрона. Так же нейрон вычисляет величину, на которую изменится вес связи:

$$\Delta w_{jk} = \mu * \sigma_k * x_j \tag{10}$$

где Δw_{jk} −величина, на которую следует изменить величину веса j-го входа k-го нейрона; μ − велечина, характеризующую скорость изменения весов. Ошибка σ посылается нейронам на предыдущем слое.

6) Каждый скрытый нейрон суммирует входящие ошибки (от нейронов в последующем слое) и вычисляет величину ошибки, умножая полученное значение на производную активационной функции:

$$\sigma_j = \varphi'(v_k) \sum_{k=1}^{n} \sigma_k * w_{jk} \tag{11}$$

так же вычисляет величину, на которую изменится вес связи:

$$\Delta w_{jk} = \mu * \sigma_j * x_j \tag{12}$$

7) Каждый выходной и скрытый нейрон изменяет веса своих связей с элементом смещения и скрытыми нейронами:

$$w_{jk} = w_{jk} + \Delta w_{jk}. \tag{13}$$

8) Проверка условия прекращения работы алгоритма. Условием прекращения работы алгоритма может быть как достижение суммарной квадратичной ошибкой результата на выходе сети

предустановленного заранее минимума в ходе процесса обучения, так и выполнения определенного количества итераций алгоритма.

1.4.4 Нормализация входных данных

Нормализация входных данных - это процесс, при котором все входные данные проходят процесс "выравнивания", т.е. приведения к интервалу [0,1] или [-1,1]. Если не провести нормализацию, то входные данные будут оказывать дополнительное влияние на нейрон, что приведет к неверным решениям [17]. Так как результат функции активации лежит в интервале от -1 до 1, то и результат работы нейронной сети мы можем получать только в этом интервале, следовательно, мы должны также проводить обратную операцию с данными. Воспользуемся формулой из [17] для получения числа после нормализации:

$$u = \frac{(x - x_{min}) * (d_2 - d_1)}{x_{max} - x_{min}} + d_1. \quad (14)$$

где $u -$ необходимое нам значение, $x -$ значение, подлежащее нормализации, $x_{max} -$максимальное значение x, $x_{min} -$ минимальное значение x, $[d_{1,} d_2] -$интервал, к которому будет приведено значение x.

1.4.5 Целевые значения

Целевые значения должны выбираться из области значений функции активации. Более точно, желаемый отклик нейрона выходного слоя многослойного персептрона должен быть смещен на некоторую величину от границы области значений функции активации в сторону ее внутренней части. В противном случае алгоритм обратного распространения будет модифицировать свободные параметры сети, устремляя их в бесконечность, замедляя, таким образом, процесс обучения и доводя скрытые нейроны до предела насыщения.

1.4.6 Скорость обучения

Алгоритм обратного распространения обеспечивает построение в пространстве весов "аппроксимации" для траектории, вычисляемой методом наискорейшего спуска. Чем меньше параметр скорости обучения, тем меньше корректировка синаптических весов, осуществляемая на каждой итерации, и тем более гладкой является траектория в пространстве весов. Однако это улучшение происходит за счет замедления процесса обучения. С другой стороны, если увеличить параметр для повышения скорости обучения, то результирующие большие изменения синаптических весов могут привести систему в неустойчивое состояние. Простейшим способом повышения скорости обучения без потери устойчивости является изменение формулы корректировки весов за счет добавления к нему момента инерции:

$$\Delta w_{jk}(n) = \alpha \Delta w_{jk}(n-1) + \mu * \sigma_j(n) * x_j * (n) \tag{14}$$

где α, как правило, положительное значение на отрезке от 0 до 1, называемое постоянной момента.

1.5.7 Преимущества и ограничения обучения методом обратного распространения

Алгоритм обратного распространения является самым распрастраненным среди алгоритмов обучения многослойного персептрона с учителем. Он представляет собой градиентный метод, а не метод оптимизации. Процедура обратного распространения обладает двумя основными свойствами:

1) процедура легко рассчитывается локально;
2) реализует градиентный спуск в пространстве весов (при котором синаптические веса обновляются для каждого примера).

Эти два свойства обучения методом обратного распространения в контексте многослойного персептрона и определяют его преимущества и ограничения.

Алгоритм обратного распространения является примером парадигмы, согласно которой возможности нейронной сети по обработке информации реализуются за счет локальных вычислений. Эта форма вычислительных ограничений называется

ограничением локальности в том смысле, что вычисления в каждом нейроне, на который воздействуют другие нейроны, выполняются обособленно. Использование локальных вычислений в контексте искусственных нейронных сетей объясняется тремя принципиальными причинами:

1) Искусственные нейронные сети, осуществляющие локальные вычисления, часто выступают в качестве модели биологических нейронных сетей.
2) Использование локальных вычислений вызывает плавное снижение производительности при сбоях в аппаратном обеспечении, что обеспечивает отказоустойчивость таких систем.
3) Локальные вычисления хорошо подходят для параллельной архитектуры, которая применяется в качестве эффективного метода реализации искусственных нейронных сетей.

Характерной особенностью, которая влияет на производительность алгоритма обратного распространения, является наличие не только глобального, но и локальных минимумов (т.е. изолированных впадин). Так как в процессе обучения методом обратного распространения направление спуска определяется направлением вектора градиента (наибольшего наклона поверхности). Всегда существует риск попадания в точку локального минимума, в которой любое приращение синаптических весов будет приводить к увеличению функции стоимости. Однако при этом в другой области пространства весовых коэффициентов могут существовать другие множества синаптических весов, для которых функция стоимости имеет меньшее значение, чем в данной точке локального минимума. Понятно, что остановка процесса обучения в точке локального минимума является крайне нежелательной, если эта точка находится выше точки глобального минимума.

1.4.8 Вычислительная эффективность алгоритма обратного распространения

Вычислительная сложность алгоритма обычно измеряется в терминах количества операций сложения, умножения и хранения, используемых в его реализации. Алгоритм обучения считается вычислительно эффективным, если его вычислительная сложность является полиномиальной по отношению к числу настраиваемых параметров, которые должны изменяться на каждой итерации. Основываясь на этом, можно утверждать, что

алгоритм обратного распространения является вычислительно эффективным. В частности, при использовании этого алгоритма для обучения многослойного персептрона, содержащего W синаптических весов, его вычислительная сложность линейно зависит от W.

Это важное свойство алгоритма обратного распространения может быть проверено с помощью прямого подсчета количества операций, осуществляемых при прямом и обратном проходах. При прямом проходе синаптические веса задействованы только для вычисления индуцированных локальных полей отдельных нейронов сети, эти вычисления линейны по синаптическим весам. При обратном проходе синаптические веса используются при вычислении локальных градиентов скрытых нейронов и при изменении самих синаптических весов.

Таким образом, все вычисления линейны по синаптическим весам сети. Отсюда следует вывод, что вычислительная сложность алгоритма обратного распространения является линейной по W, Т.е. составляет O(W) [38].

1.5 Концептуальная модель предметной области

В результате исследования можно выделить четыре основных сущности:

1) Нейронная сеть
2) Слой нейронной сети, так как выбран в качестве архитектуры многослойный персептрон
3) Нейрон, в качестве сумматора с функцией активации
4) Вес связи между нейронами

В качестве отношения между сущностями, можно обозначить, что каждая последующая сущность включается в предыдущую, то есть по методологии UML они связаны отношением агрегации. Размерность связи составляет один ко многим. Кроме этого, следует заметить необходимую для нормализации сущность, а именно входной параметр нейронной сети, которая включает в себя максимальное и минимальное значение для входа или выхода нейронной сети.

1.6 Потребности потенциальных потребителей

Основываясь на [3] и [4], [40] приведем основные сферы использования нейронных сетей:

1) разработка интеллектуальных систем, основанных на знаниях;
2) распознавание образов;
3) компьютерная лингвистика;
4) защита от вредоносных программ;
5) интеллектуальное математическое моделирование;
6) прогнозирование на фондовом рынке;
7) системы слежения за состоянием системы или оборудования.

На основе анализа программных решений и обзора комментариев, отзывов к программным средствам, были сформулированы основные потребности потенциальных потребителей:

1) легкость в использовании
2) высокая скорость обработки данных
3) быстрая настройка
4) управление из панели администрирования системы управления сайтом

1.7 Функциональные требования, предъявляемые к ПМ АДН

Разрабатываемое ПО должно обеспечить выполнение основных функций нейронной сети:

1) Создавать нейронную сеть, а именно, многослойный персептрон;
2) Обучать сеть, с помощью данных получаемых из системы 1С-Битрикс;
3) Классифицировать данные подаваемые на вход нейронной сети;
4) Прогнозировать результат на основании входных данных;

А также иметь отдельный интерфейс создания и настройки нейронной сети с возможностью управления администратором системы.

Выводы исследовательского раздела

В результате исследовательского раздела проведено сравнение существующих аналогов, что выявило необходимость создания ПМ АДН.

Для решения поставленной проблемы выполнено исследование предметной области, которое заключалось в определение понятия нейронной сети, основных положениях раскрывающих её природу, а также выборе функции активации и архитектуры нейронной сети.

Все это позволило выявить концептуальную модель предметной области, необходимую для продолжения работы над модулем в конструкторском разделе. Также были рассмотрены основные потребности потенциальных покупателей.

2 КОНСТРУКТОРСКИЙ РАЗДЕЛ

В конструкторской части были проанализированы функциональные требования к программному модулю анализа данных для веб-сайтов с использованием технологий нейронных сетей (ПМ АДН), разработаны структуры входных и выходных данных, разработаны архитектуры и алгоритм работы программного модуля, проведен обзор пользовательского интерфейса, выбраны инструменты разработки ПМ АДН.

2.1 Структура входных и выходных данных

Входные данные поступают, от пользователей сайта с помощью компонентов встроенных в модуль. Входные данные представляют собой список сигналов на входе нейронной сети.

Выходные данные представляют собой список сигналов, сформированный на выходе нейронной сети.

2.2 Выбор инструментальных средств разработки

Для разработки программного модуля нейронных сетей для системы управления сайтом необходимы следующие элементы:

1) язык программирования;
2) система управления содержимым
3) среда разработки и отладчик.

2.2.1 Выбор языка программирования.

Для выбора языка следует провести сравнение языков Python, Ruby, PHP и C#.

Python – это язык общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция (возможность определить тип

и структуру объекта во время выполнения программы [7]), механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты) [42].

Ruby — динамический, интерпретируемый высокоуровневый язык программирования для объектно-ориентированного программирования. Язык обладает независимой от операционной системы реализацией многопоточности, строгой динамической типизацией, сборщиком мусора [24]. К минусам можно отнести сложность в обучении, отсутствие документации на русском языке, малая доля потенциальных потребителей.

PHP — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов [50]. PHP является языком программирования с динамической типизацией, не требующим указания типа при объявлении переменных, равно как и самого объявления переменных. Преобразования между скалярными типами зачастую осуществляются неявно без дополнительных усилий. Большим недостатком является отсутствие механизма многопоточности, но с помощью сторонних библиотек это можно исправить.

C# - объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework. Из плюсов можно отметить большую и подробную документацию на русском языке, удобный инструментарий, собранный в программном продукте Microsoft Visual Studio [32].

Сравнительная таблица характеристик языков представлена в таблице 2.

Таблица 2 - Сравнение языков программирования

Параметры	Python[1]	Ruby[2]	C#[3]	PHP[4]
Опыт использования	нет	нет	6 мес.	10 мес.

Продолжение таблицы 2

Параметры	Python[1]	Ruby[2]	C#[3]	PHP[4]
Динамическая типизация	+	+	-	+
Многопоточность	+	+	+	+
Наличие документации на русском языке	+	-	+	+
Сборщик мусора	+	+	+	+
Интроспекция	+	-	+	+

Источники:

[1] https://blog.udemy.com/modern-language-wars

[2] Е. А. Роганов, Н. А. Роганова. Программирование на языке Ruby

[3] Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5

[4] http://www.tiobe.com/tiobe_index

Условные обозначения:

+ указанная возможность присутствует

– указанная возможность отсутствует

В результате для разработки модуля ПМ АДН был выбран язык программирования PHP.

2.2.2 Выбор системы управления содержимым.

WordPress — система управления содержимым с открытым исходным кодом, написанная на PHP. В качестве сервера базы данных выступает MySQL. Она выпущена

под лицензией GNU GPL версии 2. Сфера применения — от блогов до достаточно сложных новостных ресурсов и интернет-магазинов. Встроенная система «тем» и «плагинов» вместе с удачной архитектурой позволяет конструировать проекты широкой функциональной сложности [29].

Joomla — система управления содержимым (CMS), написанная на языках PHP и JavaScript, использующая в качестве хранилища базы данных СУБД MySQL или другие стандартные промышленные реляционные СУБД. Является свободным программным обеспечением, распространяемым под лицензией GNU GPL [20].

1С-Битрикс – система управления содержимым, в которой для хранения данных используется файловая система сервера и реляционная СУБД. Поддерживаются следующие СУБД: MySQL, Oracle, MS SQL. Продукт работает на Microsoft Windows и UNIX-подобных платформах, включая Linux [33].

Drupal — система управления содержимым, используемая также как каркас для веб-приложений, написанная на языке PHP и использующая в качестве хранилища данных реляционную базу данных (поддерживаются MySQL, PostgreSQL и другие [39]). Drupal является свободным программным обеспечением, защищённым лицензией GPL, и развивается усилиями программистов со всего мира [29].

Обзор систем управления содержимым представлен в таблице 3.

Таблица 3 – Сравнение систем управления содержимым

Параметры	WordPress[1]	Joomla[2]	1С-Битрикс[3]	Drupal[1]
Цена минимальная редакции), руб.	0	0	1990	0
Готовность к большим объемам контента	-	-	+	-
Встроенный инструмент для мультиязычности	+	+	+	+

Продолжение таблицы 3

Параметры	WordPress[1]	Joomla[2]	1С-Битрикс[3]	Drupal[1]
Поддержка мультимедийного контента	-	-	+	+
Наличие поддержки	-	-	+	-
Возможность поддерживания одновременно нескольких сайтов	+	+	+	+
Процент рынка	31,06%	24,40%	8,20%	4,90%

Источники:

[1] https://i-market.ru/news/tablitsa-sravneniya-cms/

[2] http://joomla.ru/docs/administrator/

[3] http://www.1c-bitrix.ru/products/cms/

Условные обозначения:

+ указанная возможность присутствует

– указанная возможность отсутствует

В итоге, для реализации была выбрана система управления содержимым «1С-Битрикс».

2.2.3 Выбор среды разработки и отладки.

PHP Development Tools — интегрированная среда разработки приложений на языке программирования PHP, разработанная на основе Eclipse, распространяемая на условиях лицензии Eclipse Public License. Первый релиз состоялся 18 сентября 2007 года. К преимуществам относятся возможности сворачивания кода, генерации кода, поддержка PHP 4 и PHP 5, иерархическое представление классов и методов, возможность отладки PHP-скриптов, поддержка HTML, CSS, JavaScript [28].

NetBeans IDE — свободная интегрированная среда разработки приложений (IDE) на языках программирования Java, Python, PHP, JavaScript, C, C++ и ряда других.

Проект NetBeans IDE поддерживается и спонсируется компанией Oracle, однако разработка NetBeans ведётся независимым сообществом разработчиков (NetBeans Community) и компанией NetBeans Org.

Последние версии NetBeans IDE поддерживают рефакторинг, профилирование, выделение синтаксических конструкций цветом, автодополнение набираемых конструкций на лету и множество предопределённых шаблонов кода.

Начиная с версии 6.0 NetBeans поддерживает разработку для PHP без установки дополнительных компонентов.

К преимуществам можно отнести поддержку стилей массивов, такие как разыменование массивов и краткий синтаксис массивов. IDE NetBeans также распознает признаки и анонимные переменные объектов, и PHP 7.3 включает дополнительные функции, такие как вызываемые типы подсказок, двоичное представление. В IDE NetBeans для PHP реализована поддержка непрерывной интеграции. Непрерывная интеграция представляет собой процесс разработки программного обеспечения, включающий управление версиями и специализированный сервер. На специализированном сервере запускаются тесты PHPUnit и покрытия кода на этом программном обеспечении.

Результаты тестов связаны с данными системы управления версиями, что позволяет разработчикам быстро и легко находить дефекты в программах [23]. Также в данном инструменте встроен отладчик кода «Xdebug», который позволяет проверять локальные переменные, устанавливать точки останова и просматривать выполнение кода в реальном времени.

JetBrains PhpStorm — коммерческая кросс-платформенная интегрированная среда разработки для PHP [49]. Разрабатывается компанией JetBrains на основе платформы IntelliJ IDEA.

PhpStorm представляет собой интеллектуальный редактор для PHP, HTML и JavaScript с возможностями анализа кода на лету [48], предотвращения ошибок в коде и автоматизированными средствами рефакторинга для PHP и JavaScript. Автодополнение кода в PhpStorm поддерживает спецификацию PHP 5.3, 5.4, 5.5 и 5.6 (современные и традиционные проекты), включая генераторы, сопрограммы, пространства имен, замыкания, типажи и синтаксис коротких массивов [46]. Имеется полноценный SQL-редактор с возможностью редактирования полученных результатов запросов [45].

PhpStorm разработан на основе платформы IntelliJ IDEA, написанной на Java. Пользователи могут расширить функциональность среды разработки за счет установки плагинов, разработанных для платформы IntelliJ, или написав собственные плагины.

Обзор сред разработки представлен в таблице 4.

Таблица 4 – Сравнение сред разработки[1]

Характеристики	Eclipse PDT	NetBeans IDE	PhpStorm
Подсветка синтаксиса	+	+	+
Встроенные функции PHP	+	+	+
Быстрый переход к ошибке	+	+	+
Сворачивание блоков кода	+	+	+
Поддержка версии 7.0	-	+	+
Безопасное удаление	-	+	+
Встроенный отладчик	-	+	+
Цена, руб.	0	0	6930

Источники:

[1] http://www.simplecoding.org/sravnenie-php-ide.html

Условные обозначения:

+ указанная возможность присутствует

– указанная возможность отсутствует

В результате анализа для создания модуля был выбран инструмент разработки NetBeans IDE.

Таким образом, для реализации поставленной задачи в качестве языка программирования был выбран PHP, в качестве системы управления сайтом 1С-Битрикс, а в качестве среды разработки NetBeansIDE.

2.3 Программная архитектура и алгоритм работы

В результате исследовательской работы была рассмотрена модель нейронной сети и её структура, нетрудно заметить, что концепция ООП хорошо подходит для реализации поставленной задачи.

Сущности, участвующие в построении и работе нейронных сетей:

1) Вес отдельной связи между нейронами.
2) Нейрон, представляет собой сумматор, с нелинейной функцией на выходе.
3) Уровень нейронной сети, набор из одинаковых нейронов, имеющих одинаковые входные данные.
4) Нейронная сеть, совокупность слоев, реализующих все функции по построению и решению задач, предъявляемых функциональными требованиями.

Под каждую сущность нейронной сети создается класс, за исключением весов, так как они не производят собственных вычислений. Для обеспечения возможности дальнейшего развития и реализации модуля ПМ АДН в других системах управления содержимым, следует сделать функции построения и вычисления выходов нейронной сети независимыми от реализации модуля в «1С-Битрикс». Для этого реализуется отдельная библиотека, которая будет осуществлять построение, обучение и использование нейронной сети.

Каждый класс библиотеки будет представлять собой сущность нейронной сети, это упростит задачу построения и позволит легко представлять структуру в удобном для понимания виде. Для более подробного описания приведем диаграмму классов по методологии UML:

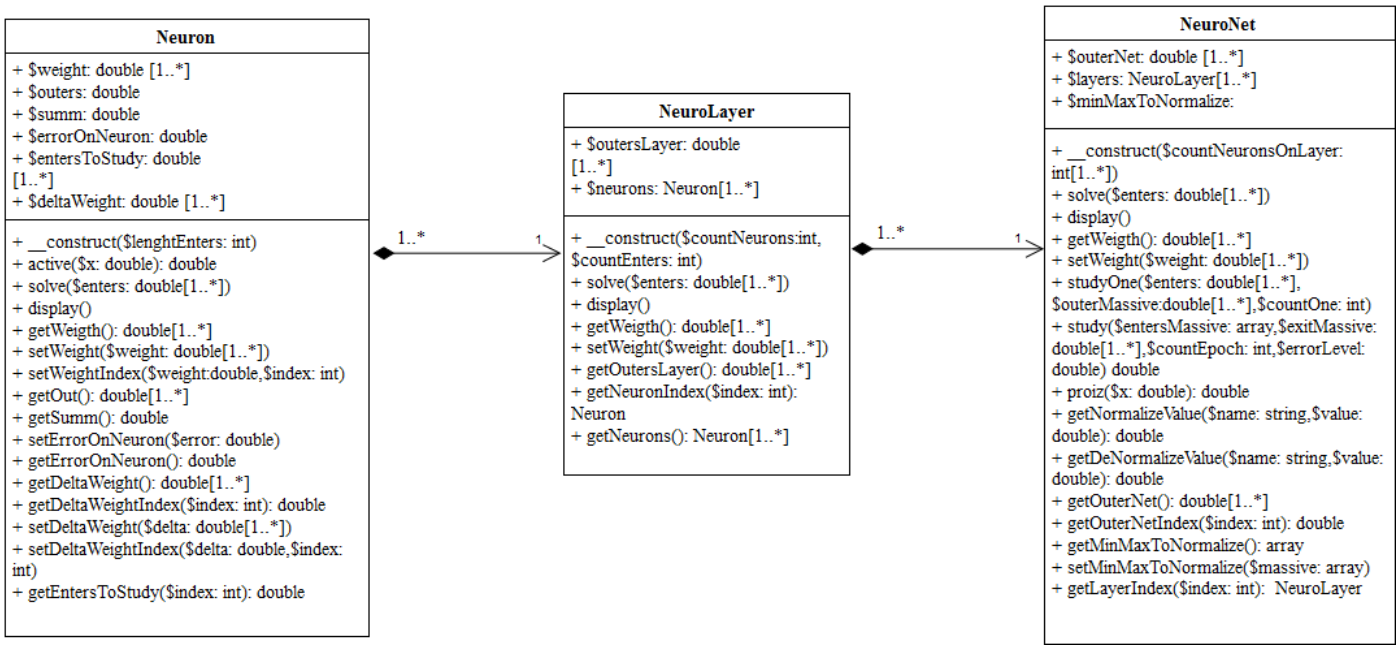

Рисунок 1 – Диаграмма классов библиотеки ПМ АДН по методологии UML

Рассмотрим подробнее каждый класс. Основой работы библиотеки является класс, элементами которого являются основные составляющие нейронной сети – нейроны. Этот класс содержит следующие поля:

1) $weight – массив весов сети;
2) $outers – результат работы нейрона;
3) $summ – выход с сумматора, результат суммирования произведений всех входов на их веса;
4) $errorOnNeuron – переменная, хранящая ошибку данного нейрона, заполняется во время работы алгоритма обучения сети;
5) $entersToStudy – массив входов нейронной сети, необходимые для обучения данные.

Решение нейроном основной задачи, суммирования и применения функции активации, обеспечивает метод «solve», вспомогательным методом является функция «active», которая вычисляет функцию активации. Для удобства представления нейронной сети введена функция «display», а также добавлены функции «getWeight» и «setWeight» для сохранения и загрузки весов.

Единичные нейроны не обеспечивают достаточный функционал для решения практически значимых задач, поэтому они объединяются в слои. Для этого используется класс «NeuroLayer». Свойства класса:

1) $outersLayer – массив выходов нейронов после выполнения функции «solve», объединённых в один слой;
2) $neurons – массив нейронов содержащихся в слое.

Методы слоя нейронов выполняют те же функции, что и в отдельных нейронах.

Слои уже могут решать определенные задачи, но для использования полного преимущества нейронных технологий составим из нейронных слоев сеть – класс «NeuroNet».

Поля нейронной сети:

1) $outerNet – массив выходов сети;
2) $layers – массив слоев нейронов содержащихся в сети.

Функции аналогичны функциям сети, за исключением функции обучения «studyOne», данная функция осуществляет обучение сети по методу обратного распространения ошибки. На вход подается массив входных сигналов и ожидаемый выход

нейронной сети. Функция предназначена для выполнения одного этапа обучения нейронной сети по методу обратного распространения ошибки и возвращает ошибку, полученную до редактирования весов нейронной сети.

На основании предыдущего метода составлена функция обучения на большом числе примеров «study». На вход подается массив примеров, а также максимальное значение ошибки и максимальное число эпох, допустимые для решения нашей задачи. Если процесс обучения проходит максимальное число эпох, то возвращается величина ошибки, если же во время обучения была достигнута требуемая погрешность, тогда возвращается количество пройденных эпох. В итоге работы будет получена обученная сеть, которая сможет выполнять заданные в требованиях функции.

Данная библиотека реализует все основные функции многослойной нейронной сети прямого распространения и при этом не зависит от структуры системы управления сайтом, что позволяет в перспективе внедрить эту библиотеку не только в «1С-Битрикс», но и в другие системы.

Система управления сайтом «1С-Битрикс» имеет модульную структуру. Каждый модуль отвечает за управление определенными элементами и параметрами сайта: информационным наполнением и структурой сайта, форумами, рекламой, рассылкой, распределением прав между группами пользователей, сбором статистики посещений, оценкой эффективности рекламных кампаний и т.д. Модуль - это модель данных и API для доступа к этим данным. Статические методы классов модуля могут вызываться в компонентах, шаблонах, других модулях. Также внутри контекста Bitrix Framework могут создаваться экземпляры классов [39]. Список имеющихся модулей представлен в разделе настройки на странице настройки продукта в списке «Модули». Для создания своего модуля мы должны придерживаться структуры, заданной разработчиками систем 1С-Битрикс:

1) admin/ - каталог с административными скриптами модуля;
    a) menu.php - файл с административным меню модуля;
2) classes/ - скрипты с классами модуля;
    a) general/ - классы модуля, не зависящие от используемой базы данных;
3) lang/ID языка/ - каталог с языковыми файлами скриптов модуля;
4) install/ - каталог с файлами, используемыми для инсталляции и деинсталляции модуля;

a) admin/ - каталог со скриптами, подключающими административные скрипты модуля (вызывающие скрипты);
b) js/ - каталог с js-скриптами модуля.

5) images/ - каталог с изображениями используемыми модулем; после инсталляции модуля они должны быть скопированы в каталог /bitrix/images/ID модуля/;
6) components/пространство имен/имя компонента/ - каталог с компонентами модуля;
7) panel/имя_модуля/ - содержит css и картинки для стилей административной панели, если модуль в таковых нуждается;
8) index.php - файл с описанием модуля;
9) version.php - файл с номером версии модуля. Версия не может быть равной нулю;
10) include.php - данный файл подключается в тот момент, когда речь идет о подключении модуля в коде, в нем должны находиться включения всех файлов с библиотеками функций и классов модуля;
11) default_option.php - содержит массив с именем $ID_модуля_default_option, в котором заданы значения по умолчанию для параметров модуля.

Придерживаясь данной структуры, составляется модуль, который будет устанавливаться из административной панели. Также он будет иметь страницу настроек, на которой можно будет выставить общие настройки сетей и отдельную страницу для создания и настройки каждой сети в отдельности.

Для внедрения нашей библиотеки классы размещаются в директории classes/general, а также создается класс «prepareForCMS» для взаимодействия библиотеки с системой управления сайтом. Он будет обеспечивать сохранение и загрузку весов в базу данных, а также получение выборок из базы данных для ввода их в алгоритм обучения нейронной сети.

Данная логика позволяет использовать функции библиотеки и логику модуля, но 1С-Битрикс содержит также компоненты, которые позволяют ещё больше упростить работу с нейронными сетями.

Компонент - это логически завершённый код, предназначенный для извлечения информации из информационных блоков, одна из основных сущностей системы 1С-

Битрикс, представляет собой уровень абстракции над обычными таблицами СУБД, и других источников и преобразования её в HTML-код для отображения в виде фрагментов веб-страниц. Состоит из собственно компонента (контроллер) и шаблона (представление). Компонент, с помощью API одного или нескольких модулей, манипулирует данными. Шаблон компонента выводит данные на страницу [39].

Компоненты реализуют паттерн проектирования Carrier-Rider-Mapper [39].

1) Carrier. Носитель любой информации, к которой могут иметь доступ несколько клиентов одновременно.
2) Rider (Reader либо Writer)- объекты, посредством которых Carrier предоставляет доступ к хранимой в нём информации. Клиенты считывают и записывают информацию, хранимую в Carrier исключительно только посредством объектов типа Reader и Writer. Таким образом, Reader и Writer - интерфейсы доступа к информации.
3) Mapper (Scanner либо Formatter) - объекты обёртки над Reader либо Writer соответственно. Мапперы отвечают за преобразование форматов данных в удобные для клиентов форматы.

При размещении компонента на странице пользователь задаёт параметры, с которыми её программный модуль будет вызван на данной конкретной странице. Набор параметров (включая их типы) перечисляется в файле параметров компонента в виде специального хэш-массива.

На странице сайта может быть размещено несколько компонентов. Один и тот же компонент может использоваться на разных страницах сайта и может использоваться на любом из сайтов внутри данной установки продукта [39].

Компонент представляет собой связь между пользователем и модулем, скрывающую работу функций библиотеки и позволяющую проводить настройки непосредственно на странице сайта. Значительным плюсом данного подхода является простота редактирования, как визуальной составляющей, путем редактирования шаблона компонента, так и, в случае необходимости, самой логики вызова функций, с помощью редактирования самого компонента.

Для реализации поставленных в требованиях функций создадим два компонента: компонент, реализующий функцию классификации, и компонент, реализующий функцию предсказания.

Задача классификации представляет собой процесс, состоящий из нормализации входных данных, обучения сети и ввода параметры объекта классификации. Прогнозирование представляет собой более сложную структуру, входные данные должны подаваться в виде наборов размерности соответствующей количеству входов нейронной сети, а также они должны представлять собой пересекающиеся последовательности данных, к примеру, для сети с количеством входов 3, и обучающего массива: 1,2,3,4,5,6,7. Наборы данных должны выглядеть как входной массив 1,2,3 ожидаемое значение 4, входной массив 2,3,4 ожидаемое значение 5 и т.д. После обучения, на вход сети подается последний набор, для нашего примера это массив 5,6,7, и на выходе сети считывается предсказанное значение.

На основе исследовательской части была разработана схема данных и схема алгоритма[12]. Схемы данных и алгоритма ПМ АДН представлены на рисунках 2 и 3.

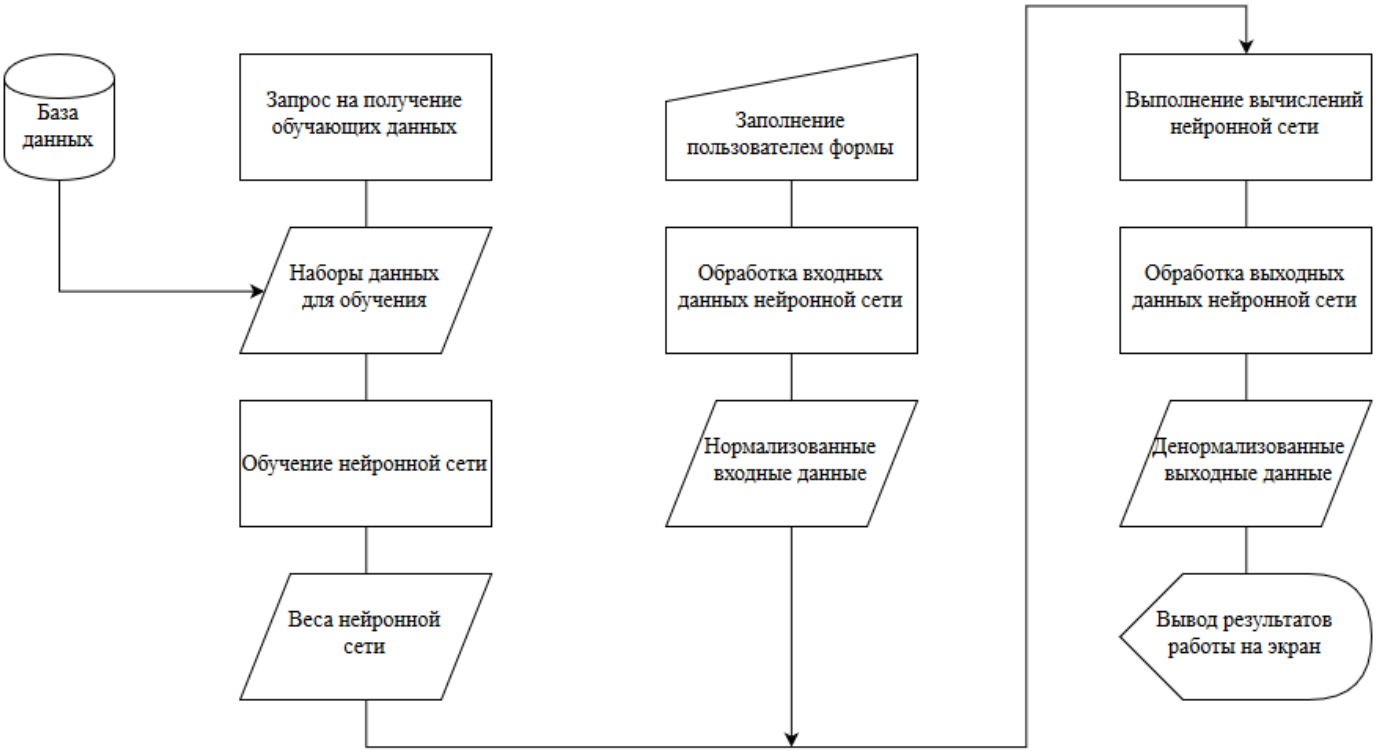

Рисунок 2 – Схема данных ПМ АДН

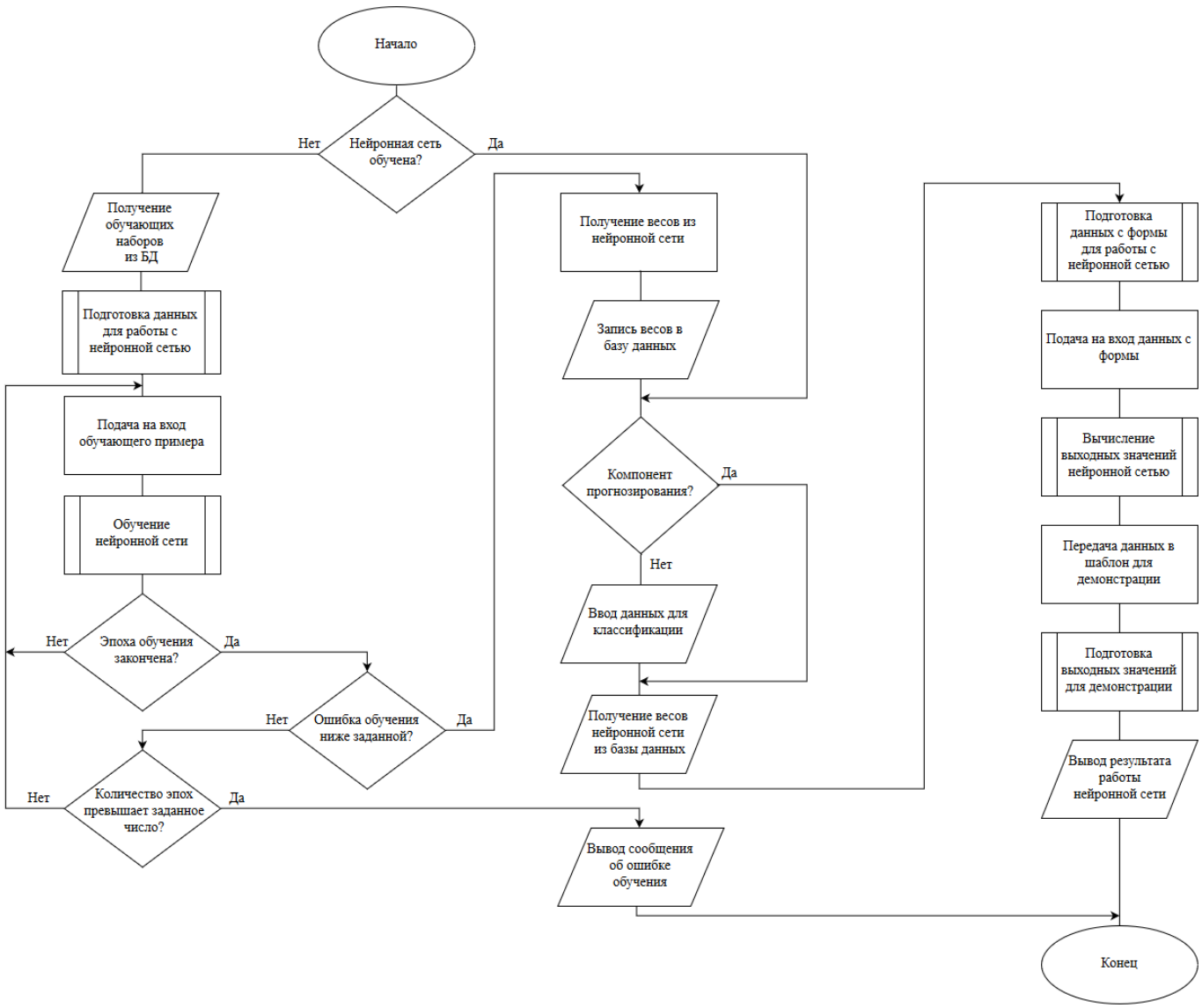

Рисунок 3 – Схема алгоритма ПМ АДН

2.4 Проектирование БД для ПМ АДН

2.4.1 Инфологическая модель предметной области

При анализе библиотеки были выделены все сущности необходимые для построения инфологической модели предметной области. А именно: нейронная сеть, уровень нейронной сети, нейрон и вес отдельной связи между нейронами. На их основе была построена инфологическая модель предметной области. Результат построения представлен на рисунке 4.

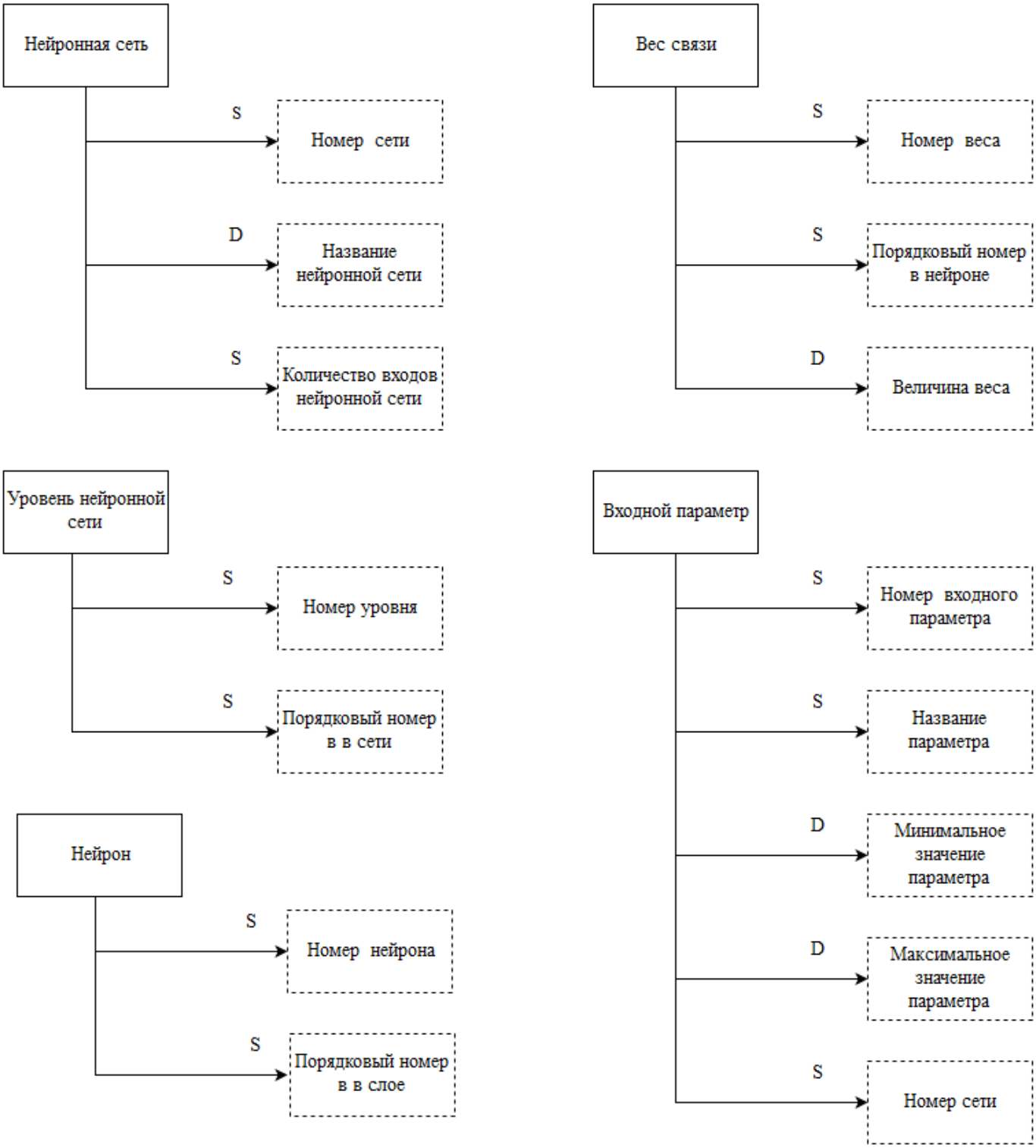

Рисунок 4 – Описание объектов

Связи между объектами отражаются на диаграмме ER-типа рисунок 5.

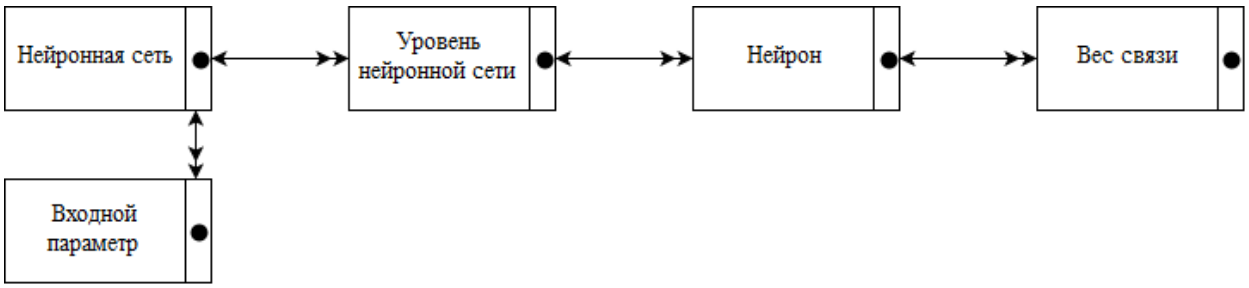

Рисунок 5 – Диаграмма ER-типа

2.4.2 Даталогическая модель

Шаги построения даталогической модели:

1) рассмотрение каждой пары сущностей и формирование таблиц, необходимых для их описания;
2) анализ полученных таблиц, для уменьшения их количества;
3) формирование логической структуры базы данных.

На основании полученной инфологической модели была построена схема базы данных (таблица 5) и схема связей таблиц в проектируемой БД (рисунок 6):

Таблица 5 – Схема базы данных

Таблица БД	Атрибут	Тип	Размер	Значение по умолчанию
a_list_neuro_nets	NET_ID	int(11)	11	
	NAME	varchar(100)	100	
	COUNT_ENTERS	int(11)	11	0
a_list_neuro_layers	LAYER_ID	int	11	
	NUMBER	int	11	0
	NET_ID	int	11	0
a_list_neuro_neurons	NEURON_ID	int	11	
	NUMBER	int	11	0
	LAYER_ID	int	11	0
a_list_neuro_weights	WEIGHT_ID	int	11	
	NUMBER	int	11	0
	WEIGHT	double		0
	NEURON_ID	int	11	0
a_list_neuro_normalized	NORM_ID	int	11	
	NORM_NAME	varchar(100)	100	
	NORM_MIN	double		0
	NORM_MAX	double		0
	NET_ID	int	11	0

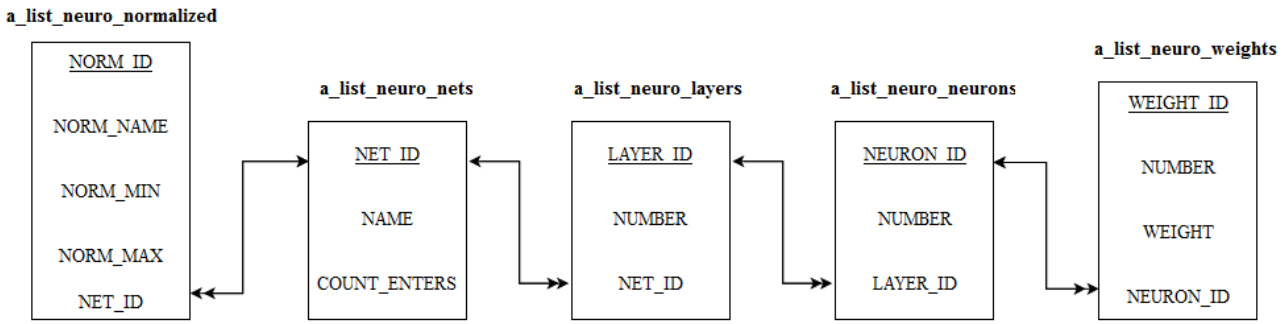

Рисунок 6 – Схема связей таблиц в проектируемой БД

2.5 Требования к надежности

Разрабатываемое ПО, предназначенное для работы с веб-серверами, должно соответствовать следующим требованиям:

1) Объем занимаемого дискового пространства не больше 50 Гигабайт;
2) Объем занимаемой оперативной памяти не больше 1 Гигабайта;
3) Возможность прерывания в случае невозможности обучить архитектуру, либо другое решение, позволяющие сохранить работоспособность сервера в подобной ситуации.

2.6 Требования к информационной и программной совместимости

Требования к составу и параметрам технических средств представлены в таблицах 5 и 6.

Таблица 5 – Минимальный состав технических средств и их технические характеристики

Процессор	Intel Xeon E3-1225 v3 3.2 Ггц
RAM (оперативная память)	1 Гбайт
OS (операционная система)	Windows 7, Unix
HDD (объем свободного места на жестком диске)	50 Гб

Таблица 6 – Рекомендуемый состав технических средств и их технические характеристики

Процессор	Intel Xeon E5-2650v3 2,3 Ггц
RAM (оперативная память)	16 Гбайт
OS (операционная система)	Windows 8, Unix
HDD (объем жесткого диска)	500 Гб

Программное обеспечение должно работать под операционными системами windows начиная с версии 7.0 и выше, язык разработки – PHP(версия 5.4), среда разработки – NetBeans IDE 8.1, система управления сайтом – «1С-Битрикс: Управление сайтом» начиная с версии 14.0 и выше.

2.7 Разработка пользовательского интерфейса

Веб-интерфейсы имеют ряд различий с привычными для пользователей приложений операционной системы Windows интерфейсами:

1) реализация через язык разметки HTML, что с одной стороны позволяет осуществлять более полное редактирование интерфейса, а с другой стороны значительно усложняет работу над ним;
2) интерфейс можно разделить на две части, пользовательскую часть должны видеть и использовать большое количество человек, в свою очередь администрированием занимается только определенная группа людей, что означает необходимость разделения прав доступа;
3) интерфейс должен быть понятным пользователю, то есть содержать описание полей, кнопок;
4) реализация должна предусматривать модификацию интерфейса.

В большинстве систем управления сайтом реализован интерфейс административной части, в частности в «1С-Битрикс» расположены:

1) настройки модулей;
2) данные из базы данных;
3) настройки различных инструментов работы с сайтом.

Для модуля ПМ АДН требуется реализовать панель создания нейронных сетей, так как это позволит использовать одну и ту же обученную нейронную сеть многократно и с разными настройками. Также интерфейс административной панели «1С-Битрикс» обеспечивает безопасность данного раздела. Тем не менее, необходимо составить удобный для пользователя программный интерфейс созданий нейронных сетей.

Входные параметры, необходимые для создания сети:

1) название сети;
2) количество входов;
3) добавление и настройка уровней.

Так как каждый уровень состоит из нейронов, следовательно, при создании сети и добавлении нового уровня мы должны выставлять количество нейронов. На основании этих требований сформируем интерфейс списка нейронных сетей, представленный на рисунке 7, а также интерфейс создания нейронной сети, изображенный на рисунке 8:

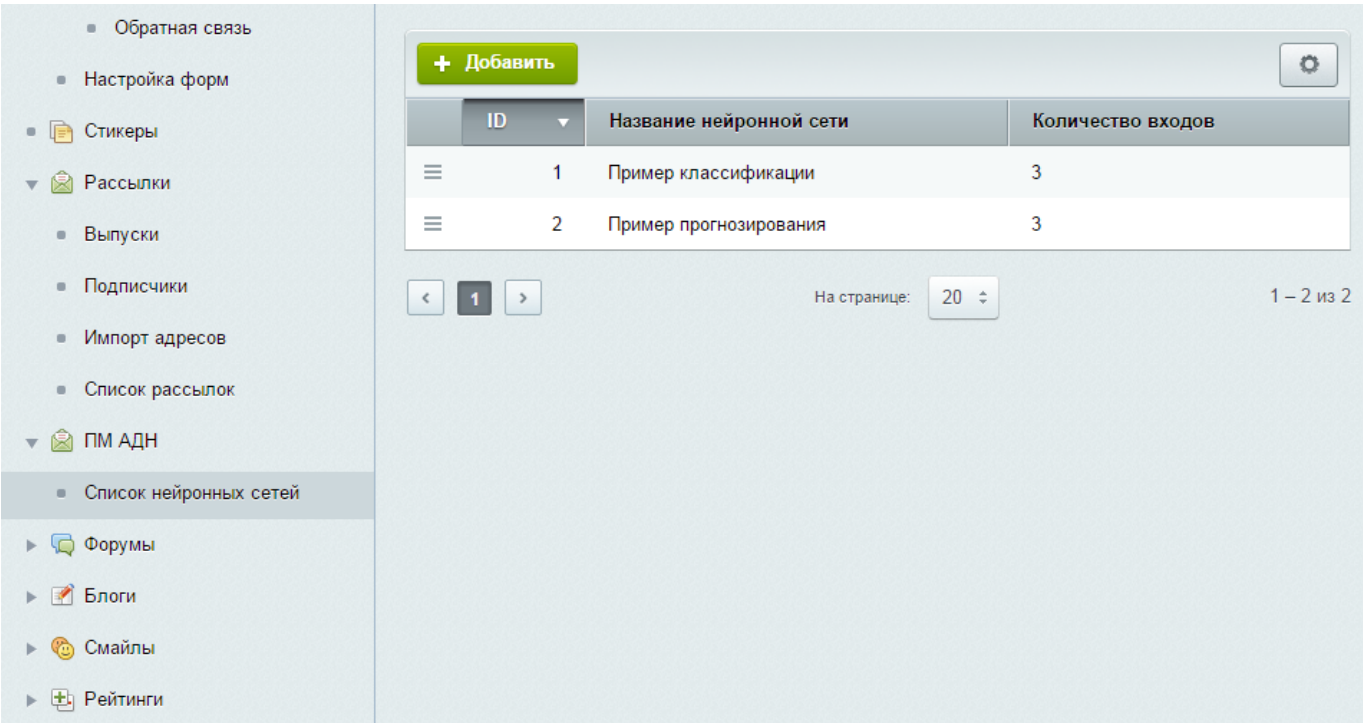

Рисунок 7 – Экранная форма страницы со списком нейронных сетей

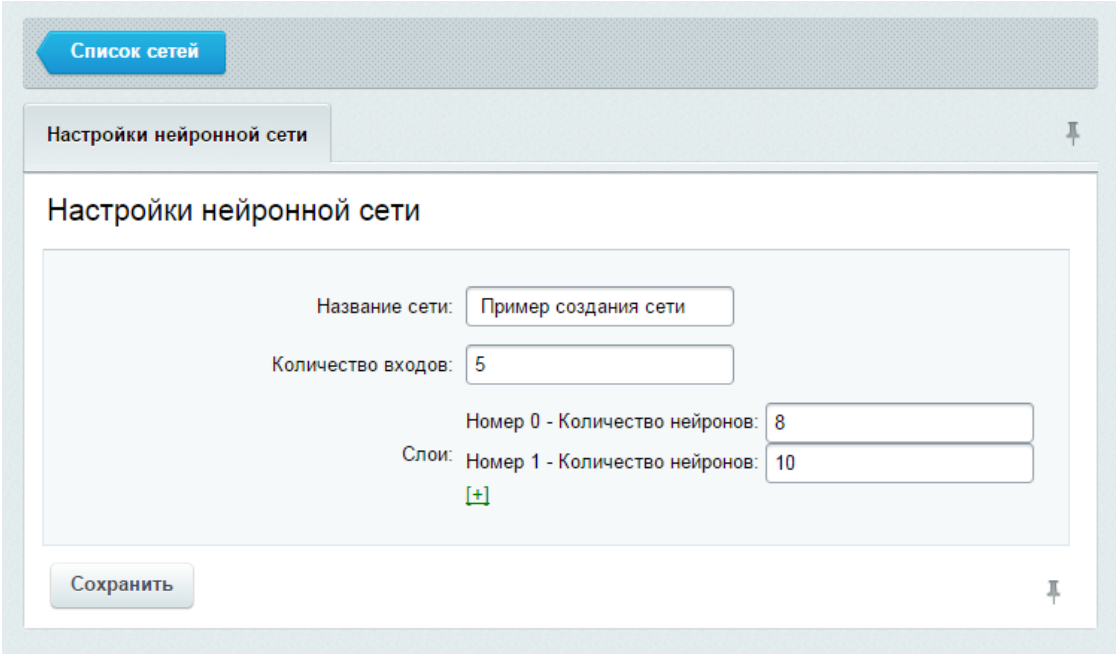

Рисунок 8 – Экранная форма страницы настроек нейронной сети

Помимо настроек модуля компонент системы «1С-Битрикс» также должен иметь свое меню настроек, которое должно содержать следующие поля: выбор сети, выбор инфоблока (специальной структуры хранения данных в «1С-Битрикс») для обучения, выбор полей подаваемых на входы и выходы, кнопка включения и выключения режима обучения сети. Разработанный интерфейс представлен на рисунке 9.

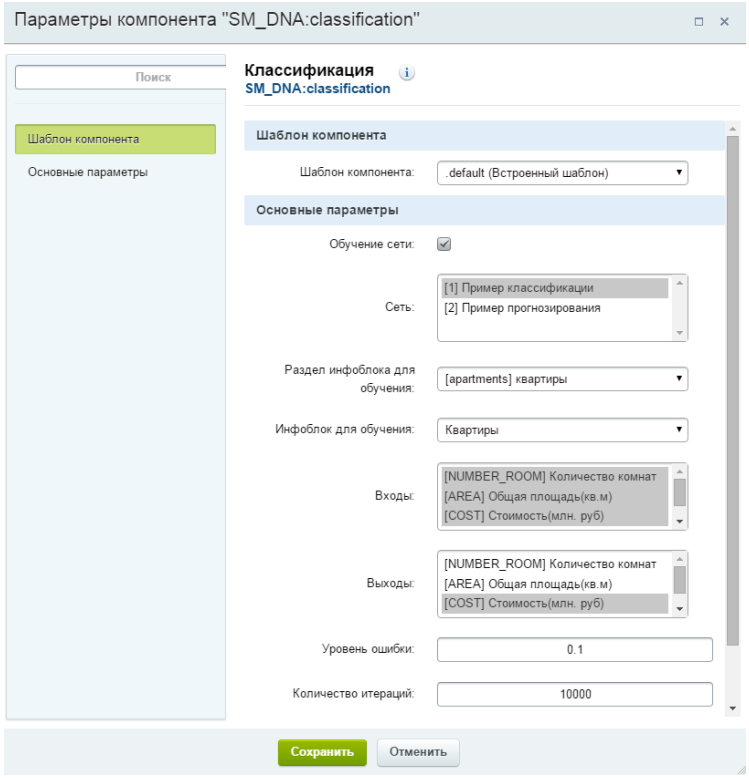

Рисунок 9 – Экранная форма настройки компонента модуля ПМ АДН

Как уже было сказано раннее компонент системы управления сайтом «1С-Битрикс» создается из двух частей: первая - это логика компонента находящаяся в файле «component.php», и вторая – это шаблон компонента, реализующий его визуальное представление, расположен в файле «template.php». Зачастую стандартный вид компонента не подходит представление внешнего вида сайта, и поэтому шаблон подвергается изменению. Кроме этого применение нейронной сети в качестве классификатора и предсказателя возможно и без визуальной составляющей, либо может содержать какие-либо особые структуры для обработки входных данных. В связи с этим шаблон компонента должен представлять максимально простую структуру, демонстрирующую пример работы с модулем. Данный пример пользовательского интерфейса компонента, реализующего функцию классификации, представлен на рисунке 10.

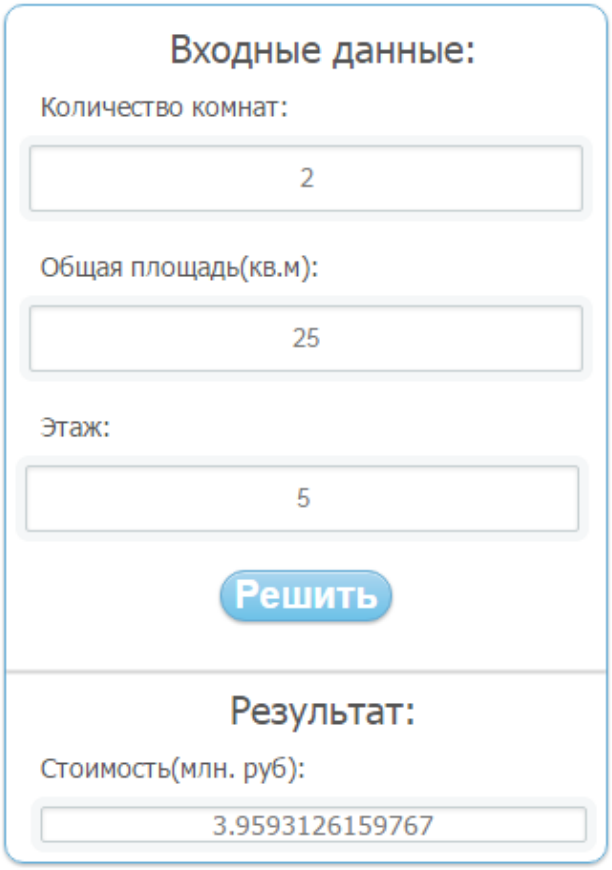

Рисунок 10 – Пример экранной формы пользовательского интерфейса компонента, реализующего функцию классификации

Компонент, реализующий функции прогнозирования, не имеет отличий в настройках, но при этом должен наглядно демонстрировать прогнозирование зависимости. Для этого необходимо внедрить библиотеку для построения графиков, где последней

точкой на графике будет искомый прогноз. Пример реализации пользовательского интерфейса для компонента, реализующего функции прогнозирования представлен на рисунке 11.

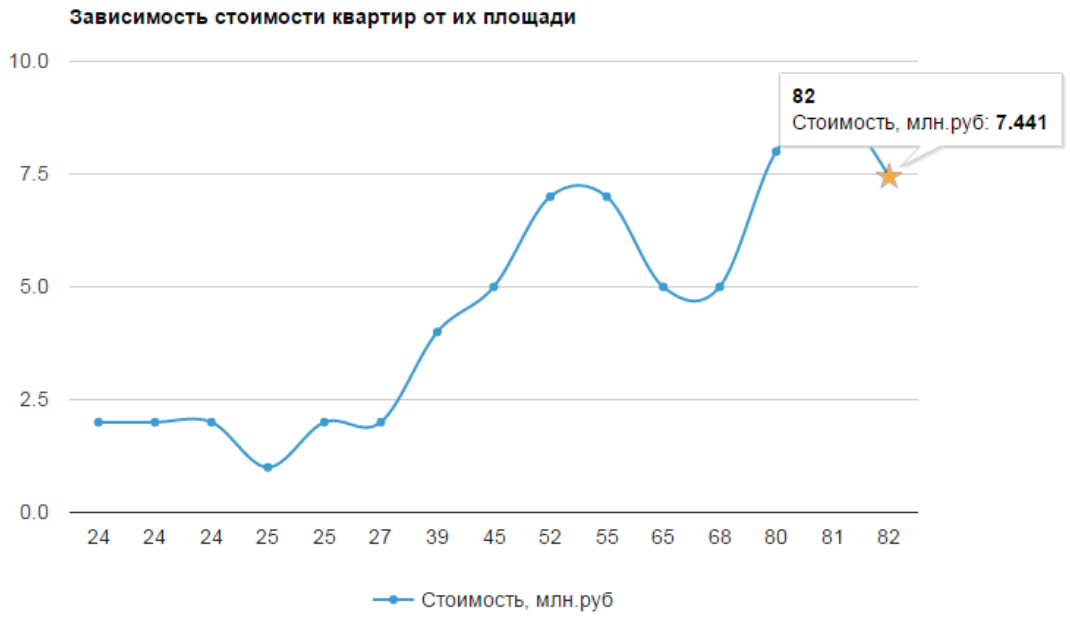

Рисунок 11 – Пример экранной формы пользовательского интерфейса компонента, реализующего функцию прогнозирования

Выводы конструкторского раздела

В результате конструкторского раздела были выбраны: в качестве языка разработки – PHP, в качестве системы управления содержимым – «1С-Битрикс», в качестве среды – разработки NetBeans IDE.

Концептуальная модель, созданная в исследовательском разделе, переведена в систему классов библиотеки для модуля ПМ АДН. Это обеспечивает возможность дальнейшего переноса модуля в другую систему управления содержимым.

Сформированы таблицы базы данных, которые автоматически должен создавать модуль. Это позволяет пользователю не заботиться о внутренних составляющих модуля. Разработаны схемы данных и алгоритма ПМ АДН. Они показывают общий алгоритм работы модуля. Также разработан пользовательский интерфейс для двух компонентов системы «1С-Битрикс», основанный на алгоритме работы модуля.

3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ

В технологическом разделе рассматриваются: особенности программирования, архитектура взаимодействия, способы отладки и тестирования.

3.1 Технологии, лежащие в основе веб-сайтов

3.1.1 Всемирная паутина

Всемирная паутина представляет собой систему, предоставляющую доступ к связанным между собой документам, и расположенную на распределенных в пространстве вычислительных узлах. В основе работы системы лежит технология гипертекста, которая представляет собой добавление возможности раскрытия дополнительного содержания материала с помощью перехода по ссылкам. Для разработки документов в данном формате создан специальный язык написания документов HTML.

Содержание веб-сайта определяется особенностями фирмы, для которой создается сайт, и тем, есть ли на этом сайте интересная, заслуживающая внимания информация. Неинтересный сайт никому не нужен, не будет посещаться, обречен на постепенное угасание.

Структура веб-сайта должна быть понятна каждому посетителю. Имея перед глазами документ, пользователь интуитивно должен представлять себе способ получения из него необходимой информации. Однажды запутавшись в джунглях сайта, клиент повторно на него не придет. И напротив – ясная и прозрачная структура сайта, в которой поиск данных не представляет труда, обязательно привлечет дополнительных посетителей.

Новизна информации на веб-сайте оказывает сильное влияние на его эффективность. Если информация не обновляется, то после второго посещения сайт перестанет быть интересным. Информацию необходимо обновлять, а старую – переносить в архив. Доступной должна быть и старая, и новая информация. Показать посетителям, что материалы обновлены, можно, указывая дату создания и обновления каждого документа.

Достоверность информации определяет авторитетность сайта. Размещать на сайте нужно только проверенные материалы, не содержащие ошибок. Каждый документ на сайте должен иметь конкретного автора – с именем, рабочим адресом, телефоном и электронной почтой, включенными в документ или вынесенными на отдельную страницу с

обязательной гипертекстовой ссылкой на нее. Этим подтверждается, что есть человек, который несет ответственность за публикуемые материалы (при этом фирма, в которой он работает, также отвечает за его действия, поскольку он является ее сотрудником).

Стиль веб-сайта должен быть уникальным. Сайт должен быть всегда узнаваемым, независимо от того, виден на экране логотип или нет. Стиль создается различными способами, в том числе – манерой изложения информации [22].

3.1.2 Язык разметки HTML

Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986—1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности путём определения небольшого набора структурных и семантических элементов — дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже [31].

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи. Например, тег <table> предназначен для создания в документах таблиц, но часто используется и для оформления размещения элементов на странице. С течением времени основная идея платформонезависимости языка HTML была принесена в жертву современным потребностям в мультимедийном и графическом оформлении [37].

Язык HTML является международным стандартом, поэтому все гипертексты, единым образом воспринимаются и единым образом отображаются на всех персональных компьютерах во всем мире [9].

3.1.3 JavaScript и его особенности

В ПМ АДН для построения графика в компоненте, предсказывающим значение, используется библиотека, написанная на JavaScript.

Для придания интерактивности веб-страницам широко используется язык JavaScript. Программы, написанные на этом языке, называются скриптами. В браузере они подключаются напрямую к HTML и выполняются, как только загружается веб-страница. Современный JavaScript – это «безопасный» язык программирования общего назначения. Он не предоставляет низкоуровневых средств работы с памятью, процессором, так как изначально был ориентирован на браузеры, в которых это не требуется.

Что же касается остальных возможностей – они зависят от окружения, в котором запущен JavaScript. В браузере JavaScript умеет делать всё, что относится к манипуляции со страницей, взаимодействию с посетителем: создавать новые HTML-теги, удалять существующие, менять стили элементов, прятать, показывать элементы, реагировать на действия посетителя, обрабатывать клики мыши, перемещения курсора, нажатия на клавиатуру, посылать запросы на сервер и загружать данные без перезагрузки страницы, получать и устанавливать cookie, запрашивать данные, выводить сообщения, и др.

JavaScript – быстрый и мощный язык, но браузер накладывает на его исполнение некоторые ограничения. Это сделано для безопасности пользователей, чтобы злоумышленник не мог с помощью JavaScript получить личные данные или как-то навредить компьютеру пользователя.

Этих ограничений нет там, где JavaScript используется вне браузера, например на сервере. Кроме того, современные браузеры предоставляют свои механизмы по установке плагинов и расширений, которые обладают расширенными возможностями, но требуют специальных действий по установке от пользователя.

JavaScript не может читать/записывать произвольные файлы на жесткий диск, копировать их или вызывать программы. Он не имеет прямого доступа к операционной системе.

Современные браузеры могут работать с файлами, но эта возможность ограничена специально выделенной директорией – «песочницей». Возможности по доступу к устройствам также прорабатываются в современных стандартах и частично доступны в некоторых браузерах.

JavaScript, работающий в одной вкладке, не может общаться с другими вкладками и окнами, за исключением случая, когда он сам открыл это окно или несколько вкладок из одного источника (одинаковый домен, порт, протокол).

Есть способы это обойти, и они раскрыты в учебнике, но они требуют специального кода на оба документа, которые находятся в разных вкладках или окнах. Без него, из соображений безопасности, залезть из одной вкладки в другую при помощи JavaScript нельзя.

Из JavaScript можно легко посылать запросы на сервер, с которого пришла страница. Запрос на другой домен тоже возможен, но менее удобен, т. к. и здесь есть ограничения безопасности [6].

3.1.4 Каскадные таблицы стилей

Для отображения компонентов модуля ПМ АДН широко применяются каскадные таблицы стилей.

Каскадные таблицы стилей это язык стилей, определяющий отображение HTML-документов. Он работает со шрифтами, цветом, полями, строками, высотой, шириной, фоновыми изображениями, позиционированием элементов и др. Язык разметки HTML может использоваться для оформления сайта, но каскадные таблицы стилей предоставляет большие возможности и более точны и проработаны, а также поддерживаются всеми браузерами [34].

Преимущества каскадных таблиц стилей [25]:

1) код легче поддерживать,
2) он быстрее загружается,
3) лучше оптимизирован для поисковых систем,
4) представляет собой модульную структуру,

5) правила стиля могут применяться к множеству страниц, что обеспечивает единообразный дизайн и более легкую поддержку, теги HTML больше не используются не по назначению.

К недостаткам можно отнести различное отображение вёрстки в различных браузерах (особенно устаревших), которые по-разному интерпретируют одни и те же данные каскадных таблиц стилей [43].

3.2 Системы управления содержимым

3.2.1 Основные сведения

Любой веб-сайт состоит из набора страниц, а различия заключаются лишь в том, как они были созданы – заверстаны вручную (статическая верстка) или сформированы динамически (с помощью программного кода). В первом случае специалисты, отвечающие за создание и поддержку сайта, пишут в HTML-форме каждую в отдельности страницу, включая ее оформление и содержание. Во втором – в основе любой веб-страницы лежит шаблон, определяющий расположение в окне веб-браузера всех компонентов страницы, и вставка конкретной информации производится с использованием стандартных средств, не требующих от участника процесса знания языка HTML и достаточно сложных для неспециалиста процедур публикации веб-страницы.

Если сайт состоит из множества страниц или он должен часто обновляться, то преимущество динамической организации становится очевидным. Разработчикам веб-сайта не надо переписывать всю страницу при изменении ее информационного наполнения или дизайна. Странички не хранятся целиком, а формируются «на лету» при обращении к ним.

Таким образом, отделение дизайна от контента является главной отличительной особенностью динамических сайтов от статических. На этой основе возможны дальнейшие усовершенствования структуры сайта, а самое главное, контроль поступающей на сайт информации. Для создания динамического сайта возможны два пути. Во-первых, это написание собственных программ, отвечающих за создание нужных шаблонов и поддерживающих необходимые функции. При этом созданная система будет полностью отвечать потребностям, однако возможно потребует больших программистских усилий и

времени. Второй путь - это воспользоваться уже существующими системами, которые называются системами управления содержимым [27].

3.2.2 Преимущества и недостатки использования систем управления содержимым

Преимуществами использования подобных систем является [21]:

1) уменьшение стоимости и времени разработки веб-сайта;
2) легкость добавления, хранения и использования информации;
3) обслуживание сайта может осуществлять низко квалифицированный персонал;
4) возможность расширения функционала сайта.

К его недостаткам можно отнести:

1) снижение гибкости отображения каждой конкретной страницы;
2) предоставление недостаточного набора возможностей редактирования;
3) безопасность сайта зависит от разработчиков системы;
4) за счет универсальности решения, наличие большого количества не задействованных участков кода.

3.3 Язык PHP и его особенности

3.3.1 Основы синтаксиса

Рассмотрим процесс выполнения php-сценария при обращении браузера к серверу. Итак, вначале браузер запрашивает страницу с расширением .php, после чего web-сервер пропускает программу через машину PHP и выдаёт результат в виде html-кода. Причем, если взять стандартную страницу HTML, изменить расширение на .php и пропустить её через машину PHP, последняя просто перешлёт её пользователю без изменений. Чтобы включить в этот файл команды PHP, необходимо заключить команды PHP в специальные теги,[32[36] такие как <?php и ?>, которые указывают PHP, когда начинать и заканчивать обработку кода между ними. Подобный способ обработки позволяет PHP внедряться во все виды различных документов, так как всё, что находится вне пары открывающих и закрывающих тегов, будет проигнорировано парсером PHP [26].

3.3.2 Использование ООП в PHP

PHP до недавнего времени обеспечивал лишь некоторую поддержку ООП. Однако после выхода PHP5 поддержка ООП в PHP стала практически полной. Стратегию ООП лучше всего описать как смещение приоритетов в процессе программирования от функциональности приложения к структурам данных. Это позволяет программисту моделировать в создаваемых приложениях реальные объекты и ситуации. Технология ООП обладает тремя главными преимуществами [47]:

1) она проста для понимания: ООП позволяет мыслить категориями повседневных объектов;
2) повышенная надежность и простота сопровождения — правильное проектирование обеспечивает простоту расширения и модификации объектно-ориентированных программ. Модульная структура позволяет вносить независимые изменения в разные части программы, сводя к минимуму риск ошибок программирования;
3) ускоряет цикл разработки — модульность и здесь играет важную роль, поскольку различные компоненты объектно-ориентированных программ можно легко использовать в других программах, что уменьшает избыточность кода и снижает риск внесения ошибок при копировании.

Специфика ООП заметно повышает эффективность труда программистов и позволяет им создавать более мощные, масштабируемые и эффективные приложения. Описание классов в PHP начинаются служебным словом class:

```
Class Имя_класса X {
     // описание членов класса - свойств и методов для их
обработки
}
```

Для объявления объекта необходимо использовать оператор new:

```
Объект = new Имя_класса;
```

Основы работы ООП в PHP схожи с аналогичными в других языках программирования за исключением некоторых особенностей [47]:

1) Конструктор, ранее совпадавший с названием класса, теперь необходимо объявлять как `__construct()`, что позволит легче перемещать классы в иерархиях. Конструкторы в классах-родителях не вызываются автоматически.

Чтобы вызвать конструктор, объявленный в родительском классе, следует обратиться к методу `parent::__construct()`. Аналогично, для деструктор необходимо объявлять как `__destruct()`.

2) В определения классов теперь можно включить константы, и ссылаться на них, используя объект. Константы также могут быть объявлены и в пределах одного класса. Отличие переменных и констант состоит в том, что при объявлении последних или при обращении к ним не используется символ $. Как и свойства и методы, значения констант, объявленных внутри класса, не могут быть получены через переменную, содержащую экземпляр этого класса.
3) PHP 5 предоставляет механизм итераторов для получения списка всех свойств какого-либо объекта, например, для использования совместно с оператором foreach. По умолчанию, в итерации будут участвовать все свойства, объявленные как public.
4) В PHP 5 сравнение объектов является более сложным процессом, чем в PHP 4, а также процессом, более соответствующим идеологии объектно-ориентированного языка. При использовании оператора сравнения (==), свойства объектов просто сравниваются друг с другом, а именно: два объекта равны, если они содержат одинаковые свойства и одинаковые их значения и являются экземплярами одного и того же класса. С другой стороны, при использовании оператора идентичности (===), свойства объекта считаются идентичными тогда и только тогда, когда они ссылаются на один и тот же экземпляр одного и того же класса.

3.4 Система «1С-Битрикс»

3.4.1 Программная платформа «Bitrix Framework»

Bitrix Framework - это созданная на основе PHP платформа для разработки веб-приложений. На этой платформе компанией «1С-Битрикс» созданы два популярных продукта: «1С-Битрикс: Управление сайтом» и «1С-Битрикс: Корпоративный портал».

при разворачивании Bitrix Framework мы получаем не только набор классов, но и развитый интерфейс администрирования.

В базовой поставке идёт большой набор компонентов, и именно он обеспечивает быстрое развёртывание и внедрение проектов.

Архитектурой платформы является архитектура модель-представление-контроллер.

Модуль - это модель данных и API для доступа к этим данным. Статические методы классов модуля могут вызываться в компонентах, шаблонах, других модулях. Также внутри контекста Bitrix Framework могут создаваться экземпляры классов.

Несколько десятков модулей системы содержат набор функций, необходимых для реализации какой-то глобальной, большой задачи: веб-формы, работа интернет-магазина, организация социальной сети и другие. Модули также содержат инструментарий для администратора сайта для управления этими функциями.

Компонент - это контроллер и представление для использования в публичном разделе. Компонент с помощью API одного или нескольких модулей манипулирует данными. Шаблон компонента (представление) выводит данные на страницу.

Компоненты входят в состав модулей, но решают более узкую, частную задачу — например, выводят список новостей или товаров. Вносить свои изменения в код продукта рекомендуется на уровне компонентов. Программист может модифицировать их как угодно, использовать свои наработки и использовать неограниченное число шаблонов на каждый из компонентов. На одной странице сайта может располагаться несколько компонентов, кроме того, их можно включать в шаблон сайта. Таким образом, программист имеет возможность собрать сайт как конструктор, после чего доработать необходимые компоненты для получения желаемого результата, как в функциональном, так и в визуальном плане.

Файлы можно править как по FTP, так и в SSH, не прибегая к дополнительным инструментам СУБД. Их легко копировать, перемещать, делать резервные копии и т.п. Строго говоря, вы можете весь контент хранить в БД. Но для простых статичных сайтов это будет явное усложнение и замедление.

Реализация на файлах кажется проблематичной в том плане, что от такой системы ожидается десятки тысяч файлов на диске. Обычно это не так. Динамическая информация (новости, каталог товаров, статьи) сохраняются в БД модулем Информационные блоки. Тогда для вывода, например, десятка тысяч товаров в интернет-магазине используется одна единственная физическая страница (файл). В этом файле вызывается компонент инфоблоков, который в свою очередь выбирает и выводит товары из базы данных.

3.4.2 Хранение данных в системе «1С-Битрикс». Информационные блоки

Информационные блоки - модуль, позволяющий каталогизировать и управлять различными типами (блоками) однородной информации. С помощью информационных блоков может быть реализована публикация различных типов динамической информации: каталоги товаров, блоки новостей, справочники и т.д.

Информационные блоки - ключевой момент каркаса системы 1С-Битрикс. Практически всё, что делается в системе в той или иной мере завязано на этот модуль, даже если это и не отображается явно. Информационные блоки представляют собой очередной уровень абстракции над обычными таблицами СУБД, своеобразная "база данных в базе данных". Поэтому к ним частично применимы все те правила, которых придерживаются при проектировании БД.

Инфоблоки - сущность, которая в физической структуре БД создает 4 таблицы, не меняющиеся при изменении структуры данных: типы объектов, экземпляры объектов, свойства объектов и значения свойств объектов [10].

Плюсы:

1) удобный контроль над данными такой структуры из своего приложения;
2) универсальность методов;
3) общая структура данных для любого проекта;
4) возможность многократно менять типы данных для полей без уничтожения самих данных.

Минусы:

1) повышенные требования к производительности;
2) непрозрачность при прямом доступе к данным.

3.4.3 Взаимодействие с базами данных в системе «1С-Битрикс»

Работа с базами данных в системе управления сайтом «1С-Битрикс» несколько отличается от аналогичной работы с БД с помощью только функций PHP. Соединение с базой данных обеспечивает система управления сайтом, поэтому для работы используются стандартные функции «1С-Битрикс».

Общая архитектура классов API для работы с базами данных:

1) Список соединений Bitrix\Main\Data\ConnectionPool управляет соединениями, параметры которых настроены в файле настроек .settings.php. В списке есть соединение по умолчанию, и может быть набор дополнительных именованных соединений. Например, соединений к другой базе данных.
2) Классы соединений наследуют абстрактный класс Bitrix\Main\DB\Connection. Для разных типов баз данных реализованы разные конкретные классы.
3) Классы формирования SQL запросов, наследующие Bitrix\Main\DB\SqlHelper. Они помогают сформировать запрос, не опускаясь до синтаксиса конкретной базы данных.
4) Классы для работы с результатом выполнения запроса, наследующие Bitrix\Main\DB\Result.

Эти классы позволяют работать с базами данных на низком уровне, но это необходимо в небольшом числе случаев.

3.4.4 Кеширование в системе «1С-Битрикс»

При большом объеме базы данных может возникнуть проблема производительности. Связано это со следующими причинами:

1) обращения к этому массиву информации на чтение или на запись порождают конкурентные запросы;
2) большое количество запросов к базе данных, создание очереди запросов;
3) запросы выполняются с невысокой скоростью и имеют большой вес, а также часто повторение одинаковых запросов.

Именно для разгрузки наиболее загруженных как по ресурсам, так и по времени, мест, и применяют многоуровневое кеширование. Каждую из технологий кеширования можно применять для каждого компонента в отдельности, выбирая оптимальный вариант для конкретного случая.

Кеширование - технология, позволяющая кешировать результаты работы редко обновляемых и ресурсоемких кусков кода (например, активно работающих с базой данных).

Система Bitrix Framework включают в себя разные технологии кеширования:

1) Кеширование компонентов (или Автокеширование) - все динамические компоненты, которые используются для создания веб-страниц, имеют встроенную поддержку управления кешированием. Для использования этой технологии достаточно включить автокеширование одной кнопкой на административной панели. При этом все компоненты, у которых был включен режим автокеширования, создадут кеши и полностью перейдут в режим работы без запросов к базе данных.
2) Неуправляемое кеширование - возможность задать правила кеширования ресурсоемких частей страниц. Результаты кеширования сохраняются в виде файлов в каталоге /bitrix/cache/. Если время кеширования не истекло, то вместо ресурсоемкого кода будет подключен предварительно созданный файл кеша. Кеширование называется неуправляемым, поскольку кеш не перестраивается автоматически после модификации исходных данных, а действует указанное время после создания, которое задается в диалоге Параметры компонента. Правильное использование кеширования позволяет увеличить общую производительность сайта на порядок. Однако необходимо учитывать, что неразумное использование кеширования может привести к серьезному увеличению размера каталога /bitrix/cache/.
3) Управляемый кеш - автоматически обновляет кеш компонентов при изменении данных. Для часто обновляемого большого массива данных использование управляемого кеша неоправданно.
4) HTML кеш лучше всего включить на какой-нибудь редко изменяющийся раздел с регулярным посещением анонимных посетителей. Технология проста в эксплуатации, не требует от пользователя отслеживать изменения, защищает дисковой квотой от накрутки данных и самовосстанавливает работоспособность при превышении квоты или изменении данных. (Считается устаревшей, рекомендуется использовать технологию Композитный сайт.) В плане многосайтовости поддерживает только многосайтовость на одном домене.
5) Кеширование меню. Для кеширования меню применяется специальный алгоритм, который учитывает тот факт, что большая часть посетителей - это незарегистрированные пользователи. Кеш меню управляемый и обновляется

при редактировании меню или изменении прав доступа к файлам и папкам через административный интерфейс и API.

3.4.5 Особенности установки модуля в системе «1С-Битрикс»

Каждый модуль должен быть корректно описан в системе для того, чтобы система знала, как с этим модулем работать. Некорректно описанные модули могут привести к полной или частичной неработоспособности системы (например, может не работать система обновлений).

Основным файлом, используемым системой для манипуляции модулем является /bitrix/modules/ID модуля/install/index.php. Основное назначение этого файла - это размещение в нем класса с именем, совпадающем с ID модуля.

Обязательные методы этого класса:

1) DoInstall - запускается при нажатии кнопки «Установить» на странице «Модули» административного раздела, осуществляет инсталляцию модуля.
2) DoUninstall - запускается при нажатии кнопки «Удалить» на странице «Модули» административного раздела, осуществляет деинсталляцию модуля.

3.5 Построение графика функции и прогнозируемого значения

Для создания компонента, реализующего функцию прогнозирования, требуется ввести дополнительную стороннюю библиотеку для построения графиков. Для решения данной задачи существует Google Chart API, данная библиотека предоставляет большое количество готовых к использованию типов диаграмм. От простых линейных диаграмм до построения иерархических деревьев. Самый распространённый способ использования данного решения – это интегрирование в страницу кода на языке javascript. Последовательность действий для построения диаграммы с помощью библиотеки Google Chart:

1) загрузка библиотеки с официального сайта;
2) составление набора входных данных;
3) настройка параметров диаграммы;
4) создание объекта диаграммы с идентификатором;

5) размещение на веб-странице элемента с ранее заданным идентификатором.

Внешний вид диаграмм легко настраивается под внешний вид сайта с помощью параметров диаграммы. Все данные передаются в диаграммы через класс «DataTable», что позволяет легко переключаться между типами диаграмм [51]. В случае с модулем ПМ АДН для демонстрации прогнозирования необходим линейный график, на котором будут выделены точки имеющихся обучающих данных, а также особо будет выделена точка предсказанного значения.

3.6 Тестирование и отладка ПМ АДН

Отладка модуля ПМ АДН осуществлялась с помощью инструмента Xdebug. Это расширение для PHP, написанное одним из разработчиков языка PHP для упрощения отладки PHP-скриптов. Основной целью расширения является максимально возможное упрощение отладки PHP-скриптов и добавление в разработку на PHP таких удобств, как точки останова, пошаговое выполнение и наблюдение за выражениями.

Помимо этого, расширение также позволяет выполнять профилировку приложения и находить те части, которые замедляют его работу. Поддерживается также выполнение произвольного кода на точке останова, а также и ряд других полезных при отладке функций. Установка осуществляется с помощью добавления в настройки сервера библиотеки. Благодаря специальному инструменту, размещённому на сайте разработчиков, для установки требуется только знание своих настроек PHP. Пример отладки представлен на рисунке 12.

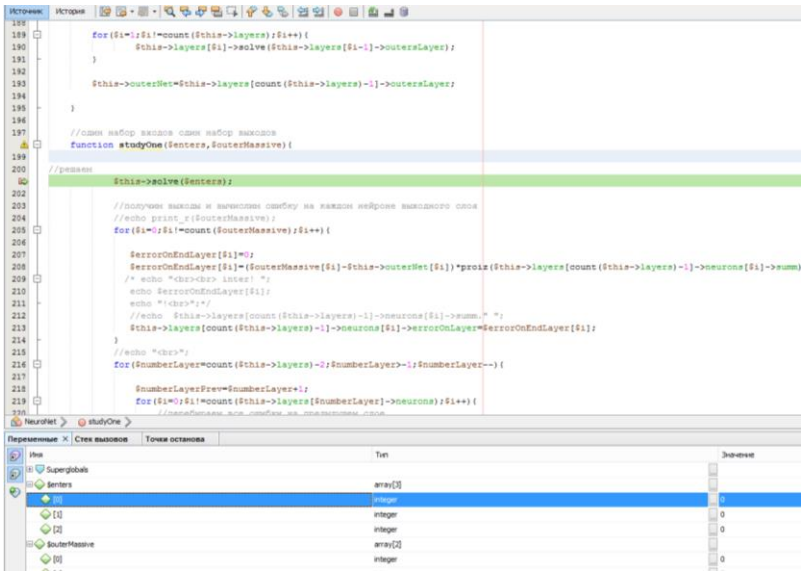

Рисунок 12 – отладка с помощью расширения Xdebug

Во время разработки программ также требуется анализировать характеристики программы, такие как: время выполнение, количество вызовов функций и т.д. Процесс сбора характеристик работы программы называется профилированием, а инструмент профилировщиком. В Xdebug присутствует встроенный профилировщик, но полученную в виде текстового файла информацию трудно анализировать, поэтому будем использовать специальную библиотеку Webgrind для представления данных в удобном для анализа виде [35], этот бесплатный продукт прост в установке и предоставляет удобный интерфейс для работы [5]. Пример работы представлен на рисунке 12.

webgrind v1.1
profiling in the browser

Show 100%
Hide PH

D:\OpenServer\OpenServer\domains\multiNeuroClass\index.php
cachegrind.out.multineuroclass_index_php @ 2016-05-05 01:23:52

Filter: (regex too)

Function	Invocation Count	Total Self Cost
(15)	2472	45.10
(9)	14861	16.10
(10)	4953	9.98
(13)	14837	5.80
(11)	2476	3.71
(8)	14861	3.49
(7)	14861	2.96
(1)	244977	0.90
(14)	49459	0.18
(12)	29675	0.13
(6)	14861	0.08
(15) NeuroNet->studyOne	1	0.02
(13) proiz	1	0.00
(17) {main}	1	0.00
(5)	0	0.00
(16)	3	0.00
(16) php::print_r	1	0.00
(2) php::rand	1	0.00
(14) php::abs	1	0.00
(12) php::cosh	1	0.00
(10) NeuroLayer->solve	1	0.00

Рисунок 103 – Представление профайлера Xdebug через библиотеку Webgrind

Как можно наблюдать, самая часто вызываемая функция из библиотеки модуля ПМ АДН это функция обучения «studyOne». Показатели, указанные в выделенном столбце, измеряются в процентах от общего времени выполнения, что говорит о достаточно высоком быстродействии. Однако следует заметить, что от строения сети и входных данных может сильно зависеть время работы, как функции вычисления, так и особенно функции обучения нейронной сети.

3.6.1 Выбор метода тестирования

По объекту тестирования различают:

1) функциональное тестирование;

2) тестирование производительности;
3) тестирование безопасности;
4) тестирование интерфейса пользователя.

Для тестирования ПМ АДН было выбрано функциональное тестирование, так как производительность следует тестировать совместно с системой «1С-Битрикс», для чего в системе присутствуют специальные инструменты, безопасность также обеспечивает система контроля содержимого, а тестировать пользовательский интерфейс имеет смысл только после внедрения в структуру сайта.

По знанию системы [30]:

1) Тестирование чёрного ящика. При использовании «черного ящика» тестировщик использует только внешние рычаги взаимодействия с программой: с помощью пользовательского интерфейса или подключившись к тестируемой системе [14].
2) Тестирование «белого ящика». При работе с «белым ящиком» тестировщик имеет доступ к коду, тем самым тестируя внутреннюю структуру программы.
3) Тестирование серого ящика [14].

Для проверки работоспособности модуля следует использовать тестирование «черного ящика». Используя метод «черного ящика», тестировщику не нужно знать внутреннее устройство программы. Объектами тестирования в этом случае являются потоки входных и выходных данных. Это позволяет определять «правильность» работы ПО в соответствии с функциональными требованиями к продукту.

По степени изолированности компонентов:

1) Модульное тестирование. Модульное (компонентное) тестирование проверяет функциональность и ищет дефекты в частях приложения, которые доступны и могут быть протестированы по-отдельности (модули программ, объекты, классы, функции и т.д.). Обычно компонентное (модульное) тестирование проводится, вызывая код, который необходимо проверить и при поддержке сред разработки. К преимуществам модульного тестирования относятся: упрощение интеграции, документирование кода, отделение интерфейса от реализации. К недостаткам можно отнести: невозможно найти ошибки, возникающие при взаимодействии модулей, неизвестен результат работы модуля в целом [15].

2) Интеграционное тестирование. Интеграционное тестирование - это фаза тестирования ПО, на которой отдельные программные модули комбинируются и тестируются в группе. Основной целью интеграционного тестирования является подтверждение того, что результаты взаимосвязи между двумя и более компонентами отвечают функциональным требованиям [8].
3) Системное тестирование. Системное тестирование предназначено для тестирования готового ПО в том состоянии, в котором оно будет внедряться в опытно-промышленную эксплуатацию. Системное тестирование позволяет обнаружить такие дефекты как выявление отсутствующего функционала в системе, некорректная работа функций системы, возникновение ошибок при использовании специфических тестовых данных или их комбинации, ошибки взаимодействия с другими системами.

Так как составные части нейронных сетей при каждом обучении могут выдавать значительно отличающийся результат, модульное тестирование не может обеспечить правильную работу модуля в целом. Интеграционное тестирование полезно на этапе отладки библиотеки, так как позволяет выявить ошибки уже в работающей системе, но в связи с тем, что внедрение библиотеки происходило исключительно в одну систему управления контентом и тестирование работы внутри системы имеет приоритетную цель, то было выбрано системное тестирование.

По степени автоматизации:

1) ручное тестирование;
2) автоматизированное тестирование;
3) полуавтоматизированное тестирование.

По степени автоматизации тестирование ПМ АДН проходило в ручном режиме. Системное тестирование для модуля в системе «1С-Битрикс» требует действий в панели администратора, что может блокироваться системой безопасности.

3.6.2 Алгоритм тестирования модуля

Модуль ПМ АДН представляет собой инструмент построения нейронных сетей, для проверки работоспособности модуля следует проверить результаты работы на сценарии

тестирования. Так как в результате предыдущего раздела было выбрано функциональное тестирование и в качестве метода «черный ящик», был разработан алгоритм сценария тестирования. На Рисунке 14 представлен алгоритм работы подобных сценариев.

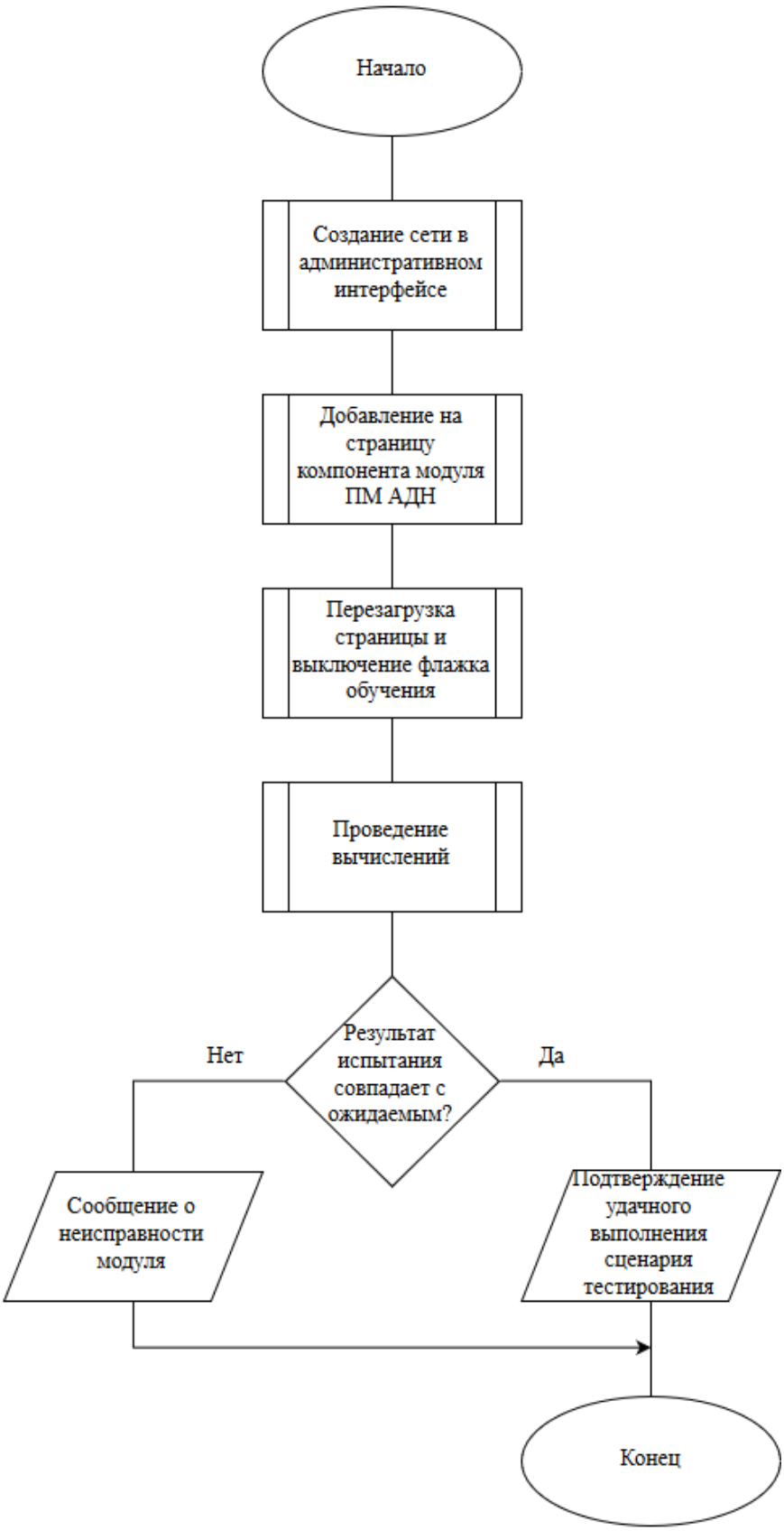

Рисунок 14. Схема алгоритма сценария тестирования

3.6.3 Сценарий тестирования «задача-XOR»

Для построения нейронных сетей классической задачей является распознавание сетью логического оператора «XOR». А именно входами сети должны быть (0,0), (0,1), (1,0), (1,1), для них соответственно должны быть выходы (0), (1), (1), (0). Подобную задачу должна решать сеть с одним скрытым слоем из двух нейронов и одним выходным нейроном [38]. Следовательно, для проверки правильности работы библиотеки можно реализовать данную структуру и оценить результаты работы. Данный сценарий тестирования представлен в таблице

Таблица 7 – Сценарий тестирования «задача-XOR»

Этап тестирования	Шаги тестирования	Ожидаемый результат	Полученный результат
1	Создание сети в административном интерфейсе с параметрами: название: «тестовая сеть 1»,количество входов: 2, слой номер 0: 2, слой номер 1: 1. Нажать кнопку «Сохранить».	Создана нейронная сеть с заданными параметрами	Создана нейронная сеть с заданными параметрами
2	Добавить на страницу компонент модуля ПМАД-Н со свойствами: Входы и выходы выбрать соответственно обучающим выборкам, для сети необходимой нам топологии это два поля для входов, и одно поле для выхода. Выставить флажок «обучение нейронной сети». Выбрать нейронную сеть «тестовая сеть 1». Оставить поля «Уровень ошибки» и «Количество итерации» по умолчанию, для параметра «Уровень ошибки» это свойство равно 0.1, а для «Количество итерации» содержит 10000 итераций.	На странице создан компонент с заданными параметрами	На странице создан компонент с заданными параметрами

Продолжение таблицы 7

Этап тестирования	Шаги тестирования	Ожидаемый результат	Полученный результат
3	Осуществить перезагрузку страницы для запуска обучения и убрать флажок «обучение нейронной сети».	Происходит ускорение работы страницы и флажок «обучение нейронной сети» в настройках не выставлен.	Происходит ускорение работы страницы и флажок «обучение нейронной сети» в настройках не выставлен.
4	Провести вычисления: Подать на вход (0,0). Подать на вход (0,1). Подать на вход (1,0). Подать на вход (1,1).	Для (0,0) -0.1<выход <0.1; Для (0,1) 0.9<выход <1.1; Для (1,0) 0.9<выход <1.1; Для (1,1) -0.1<выход <0.1;	Для (0,0) выход равен 0; Для (0,1) выход равен 0.9658383; Для (1,0) выход равен 0.9658158; Для (1,1) выход равен 0.0224677;

Все полученные результаты входят в интервалы погрешности, следовательно, тестовый сценарий выполнен успешно.

Выводы технологического раздела

В результате использования современных технологий можно создать простой и удобный инструмент для работы с нейронными сетями в интернет-сфере.

Современные средства организации объектно-ориентированного программирования на языке PHP позволяют реализовать все классы, запланированные в конструкторском разделе и использовать все преимущества инкапсуляции. Также современные библиотеки на языке JavaScript позволяют быстро и эффективно построить график по необходимым данным.

Система «1С-Битрикс» предоставляет достаточные возможности для обеспечения безопасности, простоты установки и использования модуля. Встроенные функции позволяют упростить вид sql-запросов к базе данных. Инструменты кеширования системы «1С-Битрикс» обеспечивают снижение нагрузки на веб-сервер.

Проведены отладка и тестирования с помощью специальных инструментов тестирования. Также, для дальнейшего упрощения тестирования, написан алгоритм тестирования и приведена хорошо известная задача в качестве сценария тестирования.

В итоге технологического раздела выполнены следующие работы:

1) изучены и применены технологии разработки;
2) проведены тестирование и отладка модуля ПМ АДН.

Заключение

Выпускная квалификационная работа посвящена разработке программного модуля анализа данных для веб-сайтов с использованием технологий нейронных сетей. В ходе ВКР были выполнены следующие задачи:

1) исследована предметная область;
2) проведен сравнительный анализ существующих аналогичных решений;
3) выбраны инструментальные средства и среды разработки;
4) разработана схема данных ПМ АДН;
5) разработана схема алгоритма ПМ АДН;
6) выполнена программная реализация ПМ АДН;
7) разработан пользовательский интерфейс ПМ АДН;
8) проведены отладка и тестирование ПМ АДН;
9) разработано руководство программиста;

Полученный программный модуль удовлетворяет всем требованиям технического задания. Использование современных технологий позволило создать удобный в обращении программный модуль.

В ходе разработки было выявлено, что для повышения эффективности требуется ввести параллельные вычисления для ускорения работы модуля, а также провести дополнительное исследование на тему эффективности внедрения для разных задач различных алгоритмов обучения.

Список используемой литературы

1. Гагарина Л.Г., Касимов Р.А., Коваленко Д.Г., Федотова Е.Л,Чжо Зо Е, Черников Б.В. Методические указания по подготовке выпускной квалификационной работы по направлению подготовки бакалавров 09.03.04 «Программная инженерия»/ Под редакцией Б.В. Черникова; М., МИЭТ, 2016 г., 20 с.
2. Аксенов С.В., Новосельцев В.Б. Организация и использование нейронных сетей (методы и технологии). – Томск: Изд-во НТЛ, 2006. – 128 с.
3. Боровиков В.П. Нейронные сети. STATISTICA Neural Networks: Методология и технологии современного анализа данных. – 2-е изд., перераб. и доп. – М.: Горячая линия – Телеком, 2008. – 392 с., ил.
4. Бэстенс Д.-Э., ван ден Берг В.-М., Вуд Д. Нейронные сети и финансовые рынки принятие решений в торговых операциях. – Москва: ТВП, 1997. – 236 с.
5. В поисках слабого звена: как найти узкие места в приложениях [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://xakep.ru/2011/03/22/55102/. – (Дата обращения: 03.05.2016).
6. Введение в JavaScript. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://learn.javascript.ru/intro – (Дата обращения: 21.05.2016).
7. Дэвид, Скляр PHP. Рецепты программирования. 3-е изд. – СПб.: Питер, 2015. – 784 с.
8. Интеграционное тестирование. - Электрон. текстовые дан. – Режим доступа: http://qatestlab.com/ru/services/Step-by-Step/Integration-Testing/ – (Дата обращения: 29.05.2016).
9. Интернет-технологии. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.tadviser.ru/index.php – (Дата обращения: 03.05.2016).
10. Инфоблоки [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://dev.1c-bitrix.ru/learning/course/index.php?COURSE_ID=43&CHAPTER_ID=04610. – (Дата обращения: 03.05.2016).
11. Каллан, Роберт Основные концепции нейронных сетей.: Пер. с англ. – М.: Издательский дом «Вильямс», 2001. – 287 с.

12. Колдаев, В.Д. Основы алгоритмизации и программирования: учебное пособие / под ред. проф. Л.Г. Гагариной. – М.: ИД «ФОРУМ»: ИНФРА-М, 2012. – 416 с.
13. Комашинский В.И., Смирнов Д.А. Нейронные сети и их применение в системах управления и связи. – М.: Горячая линия – Телеком, 2003. – 94 с.
14. Методами Черного ящика и Белого ящика [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://qalight.com.ua/baza-znanij/metodami-chernogo-i-belogo-yashchikov.html. – (Дата обращения: 02.05.2016).
15. Модульное тестирование. - Электрон. текстовые дан. – Режим доступа: http://citforum.ru/SE/testing/unit_testing/ – (Дата обращения: 29.05.2016).
16. Нейронные сети с радиальными базисными функциями [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://bibliofond.ru/view.aspx?id=445701. – (Дата обращения: 03.05.2016).
17. Нормализация входных данных [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://www.mql5.com/ru/articles/497. – (Дата обращения: 03.05.2016).
18. Общие сведения о нейронных сетях с радиальными базисными функциями [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://studopedia.org/1-12530.html. – (Дата обращения: 03.05.2016).
19. Осовский С. Нейронные сети для обработки информации.: Пер. с польского И.Д. Рудинского. – М.: Финансы и статистика, 2002. – 344с.: ил.
20. Первые шаги - Руководство Joomla 3.0 [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://joomla.ru/docs/administrator/joomla3-start. – (Дата обращения: 19.02.2016).
21. Плюсы и минусы CMS - движков. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http:/iq-project.ru/info/pros-and-cons-of-cms. – (Дата обращения: 03.05.2016).
22. Последовательность создания гипертекстовых систем. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.intuit.ru/studies/courses/3632/874/lecture/14329 – (Дата обращения: 21.05.2016).

23. Разработка на PHP [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://netbeans.org/features/php/index_ru.html. – (Дата обращения: 03.05.2016).
24. Роганов Е. А., Роганова Н. А.. Программирование на языке Ruby. Учебное пособие (PDF, 425 Кбайт). — М.: МГИУ, 2008. — 56 с.
25. Руководство по CSS для начинающих. - Электрон. текстовые дан. – Режим доступа: http://technologyweb.org/ – (Дата обращения: 29.05.2016).
26. Руководство по PHP. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://secure.php.net/manual/ru/. – (Дата обращения: 03.05.2016).
27. Системы управления контентом, их функции, требования предъявляемые к ним [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://joomla.ru/articles/site-development/482-cms-requirements. – (Дата обращения: 03.05.2016).
28. Сравнение PHP IDE [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.simplecoding.org/sravnenie-php-ide.html. – (Дата обращения: 20.02.2016).
29. Таблица сравнения CMS [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://i-market.ru/news/tablitsa-sravneniya-cms/.– (Дата обращения: 19.02.2016).
30. Тестирование программного обеспечения - основные понятия и определения. - Электрон. текстовые дан. – Режим доступа: http://www.protesting.ru/testing – (Дата обращения: 29.05.2016).
31. Титтел Эд, Джефф Ноубл. HTML, XHTML и CSS для чайников, 7-е издание – М.: Диалектика, 2011. — 400 с.
32. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5, 6-е издание: Пер. с англ. – М.: ООО «И.Д.Вильямс», 2013 – 1312 с.
33. Управление сайтом 1С-Битрикс[Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.1c-bitrix.ru/products/cms/.– (Дата обращения: 19.02.2016).
34. Урок 1: Что такое CSS?. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://ru.html.net/tutorials/css/lesson1.php – (Дата обращения: 29.05.2016).

35. Ускорение кода при помощи GNU-профайлера [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.ibm.com/developerworks/ru/library/l-gnuprof/. – (Дата обращения: 03.05.2016).
36. Учебник по PHP 4. [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.softtime.ru/bookphp/gl1_1.php. – (Дата обращения: 03.05.2016).
37. Фримен Э. Изучаем HTML, XHTML и CSS – СПб.: Питер, 2010. – 656 с.
38. Хайкин, Саймон Нейронные сети: полный курс, 2-е изд., испр.: Пер. с англ. – М.: ООО «И.Д.Вильямс», 2006. – 1104 с.
39. Центр поддержки разработчиков 1С-Битрикс [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://dev.1c-bitrix.ru.– (Дата обращения: 19.02.2016).
40. Ясницкий Л.Н. Введение в искусственный интеллект: учеб. пособие для студ. высш. заведений – 3-е издание – М.: Издательский центр «Академия». 2010 – 176 с.
41. ANN - Artificial Neural Network for PHP 5.x [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://ann.thwien.de/.– (Дата обращения: 19.02.2016).
42. Code Wars: Ruby vs Python vs PHP [Infographic] [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://blog.udemy.com/modern-language-wars/.– (Дата обращения: 19.02.2016).
43. CSS. - Электрон. текстовые дан. – Режим доступа: https://ru.wikipedia.org/wiki/CSS – (Дата обращения: 29.05.2016).
44. Fast Artificial Neural Network Library[Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://leenissen.dk/fann/wp/. – (Дата обращения: 19.02.2016).
45. JetBrains PhpStorm 5.0 Aligns To Symfony2 and Yii [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.drdobbs.com/tools/jetbrains-phpstorm-50-aligns-to-symfony2/240007578. – (Дата обращения: 02.05.2016).

46. JetBrains PhpStorm 5.0 Provides New PHP Framework Support [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.eweek.com/c/a/Application-Development/JetBrains-PhpStorm-50-Provides-New-Framework-Support-560166. – (Дата обращения: 02.05.2016).
47. PHP[Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.php.su/learnphp/phpoo. – (Дата обращения: 02.05.2016).
48. PhpStorm review [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.pcadvisor.co.uk/review/programming-software/phpstorm-review-3331137. – (Дата обращения: 02.05.2016).
49. Slick PhpStorm Makes Editing JavaScript and PHP Fun [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.pcworld.com/article/248117/slick_phpstorm_makes_editing_javascript_and_php_fun.html. – (Дата обращения: 02.05.2016).
50. TIOBE Index for April 2016 [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: http://www.tiobe.com/tiobe_index. – (Дата обращения: 02.05.2016).
51. Using Google Charts [Электронный ресурс]. - Электрон. текстовые дан. – Режим доступа: https://developers.google.com/chart/interactive/docs/. – (Дата обращения: 03.05.2016).

Студент гр. МП – 44 _____/ Казначеев А.А./

«___»_____2016 г.

ПРИЛОЖЕНИЕ 1

ПРОГРАММНЫЙ МОДУЛЬ АНАЛИЗА ДАННЫХ ДЛЯ ВЕБ-САЙТОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ НЕЙРОННЫХ СЕТЕЙ (ПМ АДН) ПРОГРАММНЫЙ КОД

Москва, 2016

СОДЕРЖАНИЕ

1. Файлы установки и настройки параметров ПМ АДН 73
1.1 Файл include.php 73
1.2 Файл install/index.php 73
1.3 Файл install/version.php 77
1.4 Файл admin/menu.php 77
1.5 Файл admin/neuro_net.php 78
1.6 Файл admin/neuro_net_edit.php 83
2 Классы модуля ПМ АДН 90
2.1 Класс Neuron 90
2.2 Класс NeuroLayer 93
2.3 Класс NeuroNet 96
2.4 Функции работы с базой данных prepareForCMS 103
3 Компонент, реализующий функцию классификации 109
3.1 Файл .description.php 109
3.2 Файл .parameters.php 110
3.3 Файл component.php 113
3.4 Файл template.php 118
4 Компонент, реализующий функцию предсказания 123
4.1 Файл .description.php 123
4.2 Файл .parameters.php 123
4.3 Файл component.php 127
4.4 Файл template.php 131

1. Файлы установки и настройки параметров ПМ АДН

1.1 Файл include.php

```
<?
IncludeModuleLangFile(__FILE__);

//подключение библиотек
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/neuron.php');
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/neuroLayer.php');
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/neuroNet.php');
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/prepareForCMS.php');

CModule::AddAutoloadClasses(
	"SDAM_N",
	array(
		"neuron" => "classes/general/neuron.php",
		"neuroLayer" => "classes/general/neuroLayer.php",
		"neuroNet" => "classes/general/neuroNet.php",
		"prepareForCMS" => "classes/general/prepareForCMS.php"
	)
);
?>
```

1.2 Файл install/index.php

```
<?
/**
 * Класс установки модуля ПМ АДН
 */
class SDAM_N extends CModule
{
```

```
    var $MODULE_ID = "SDAM_N";
    var $MODULE_VERSION;
    var $MODULE_VERSION_DATE;
    var $MODULE_NAME;
    var $MODULE_DESCRIPTION;
    var $MODULE_GROUP_RIGHTS = "Y";

    /**
     * Конструктор класса SDAM_N
     */
    function SDAM_N()
    {
        $arModuleVersion = array();

        $path = str_replace("\\", "/", __FILE__);
        $path = substr($path, 0, strlen($path) - strlen("/index.php"));
        include($path."/version.php");

        $this->MODULE_VERSION = '1.0';
        $this->MODULE_VERSION_DATE = '5555-55-55 15:19:25';
        $this->MODULE_NAME = 'Программный модуль анализа данных с помощью
технологий нейронных сетей';
        $this->MODULE_DESCRIPTION = 'Программный модуль, позволяющий
создавать и использовать нейронные сети';
    }

    /**
     * Функция установки модуля SDAM_N
     */
    function DoInstall()
    {
```

```
global $APPLICATION,$DB;
//создание таблиц
//net
$results = $DB->Query("CREATE TABLE a_list_neuro_nets
(
    NET_ID        int(11)        NOT NULL auto_increment,
    NAME        VARCHAR(100)    NULL,
    COUNT_ENTER        int(11)        NULL,
    PRIMARY KEY (NET_ID)
);");
//layer
$results = $DB->Query("CREATE TABLE a_list_neuro_layers
(
    LAYER_ID        int(11)        NOT NULL auto_increment,
    NUMBER int(11) NULL,
    NET_ID        int(11)        NOT NULL,
    PRIMARY KEY (LAYER_ID)
);");
//neuron
$results = $DB->Query("CREATE TABLE a_list_neuro_neurons
(
    NEURON_ID        int(11)        NOT NULL auto_increment,
    NUMBER int(11) NULL,
    LAYER_ID int(11) NOT NULL,
    PRIMARY KEY (NEURON_ID)
);");
//weight
$results = $DB->Query("CREATE TABLE a_list_neuro_weights
(
    WEIGHT_ID        int(11)        NOT NULL auto_increment,
```

```
            NUMBER int(11) NULL,
            WEIGHT double NULL,
            NEURON_ID int(11) NULL,
            PRIMARY KEY (WEIGHT_ID)
        );");
        //normalize
        $results = $DB->Query("CREATE TABLE a_list_neuro_normalizes
        (
            NORM_ID        int(11)        NOT NULL auto_increment,
            NORM_NAME        VARCHAR(100)    NULL,
            NORM_MIN        double        NULL,
            NORM_MIN        double        NULL,
            NET_ID        int(11)        NOT NULL,
            PRIMARY KEY (NORM_ID)
        );");

        //копирование компонентов

CopyDirFiles($_SERVER["DOCUMENT_ROOT"]."/local/modules/SDAM_N/install/compone
nts",SERVER["DOCUMENT_ROOT"]."/local/components", true, true);
        RegisterModule('SDAM_N');
        COption::SetOptionInt('SDAM_N', 'installed', 1);
    }

    /**
     * Функция деинсталляции модуля SDAM_N
     */
    function DoUninstall()
    {
        global $APPLICATION,$DB;
        //удаление таблиц
```

```
        $results = $DB->Query("DROP TABLE a_list_neuro_nets");
        $results = $DB->Query("DROP TABLE a_list_neuro_layers");
        $results = $DB->Query("DROP TABLE a_list_neuro_neurons");
        $results = $DB->Query("DROP TABLE a_list_neuro_weights");
        $results = $DB->Query("DROP TABLE a_list_neuro_normalizes");
        //удаление компонентов
        DeleteDirFilesEx("/local/components/SDAM_N");
        UnregisterModule('SDAM_N');
        COption::SetOptionInt('SDAM_N', 'installed', 0);
    }
}
?>
```

1.3 Файл install/version.php

```
<?
$arModuleVersion = array(
     "VERSION" => "1.0",
     "VERSION_DATE" => "2016-05-20 12:00:00"
);
?>
```

1.4 Файл admin/menu.php

```
<?
IncludeModuleLangFile(__FILE__);

if($APPLICATION->GetGroupRight("SDAM_N")!="D")
{
    $aMenu = array(
```

```
        "parent_menu" => "global_menu_services",
        "section" => "SDAM_N",
        "sort" => 200,
        "text" => "ПМАД-Н",
        "title" => "ПМАД-Н",
        "icon" => "subscribe_menu_icon",
        "page_icon" => "subscribe_page_icon",
        "items_id" => "menu_SDAM_N",
        "items" => array(
            array(
                "text" => "Список нейронных сетей",
                "url" => "neuro_net.php?lang=".LANGUAGE_ID,
                "more_url" => array("neuro_net.php"),
                "title" => "Список нейронных сетей"
            )
        )
    );

    return $aMenu;
}
return false;
?>
```

1.5 Файл admin/neuro_net.php

```
<?
require_once($_SERVER["DOCUMENT_ROOT"]."/bitrix/modules/main/include/prolog_a
dmin_before.php");

global $DB, $DBType, $APPLICATION;
```

```
IncludeModuleLangFile(__FILE__);


$sTableID = "sdamn";

$oSort = new CAdminSorting($sTableID, "ID", "desc");

$lAdmin = new CAdminList($sTableID, $oSort);


if($_GET['DELETE']==1){

    deleteNet($_GET['ID']);

}


$strSql = "SELECT * FROM a_list_neuro_nets";

$rsData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

$rsData = new CAdminResult($rsData, $sTableID);

$rsData->NavStart();

$lAdmin->NavText($rsData->GetNavPrint(GetMessage("rub_nav")));


$lAdmin->AddHeaders(array(

    array(    "id"        =>"NET_ID",

        "content"    =>"ID",

        "sort"        =>"id",

        "align"        =>"right",

        "default"    =>true,

    ),

    array(    "id"        =>"NAME",

        "content"    =>"Название нейронной сети",

        "sort"        =>"name",

        "default"    =>true,

    ),

    array(    "id"        =>"COUNT_ENTER",
```

```
        "content"    =>"Количество входов",
        "sort"        =>"COUNT_ENTER",
        "default"    =>true,
    )
));

while($arRes = $rsData->NavNext(true, "f_")):
    $row =& $lAdmin->AddRow($f_NET_ID, $arRes);

    $arActions = Array();
    $arActions[] = array(
        "ICON"=>"edit",
        "DEFAULT"=>true,
        "TEXT"=>"Характеристики сети",
        "ACTION"=>$lAdmin-
>ActionRedirect("neuro_net_edit.php?ID=".$f_NET_ID."&SHOW=1")
    );
    $arActions[] = array(
        "ICON"=>"delete",
        "TEXT"=>"Удалить сеть",
        "ACTION"=>$lAdmin-
>ActionRedirect("neuro_net.php?ID=".$f_NET_ID."&DELETE=1")
    );

    $arActions[] = array("SEPARATOR"=>true);



    if(is_set($arActions[count($arActions)-1], "SEPARATOR"))
        unset($arActions[count($arActions)-1]);
    $row->AddActions($arActions);
```

```
endwhile;


$lAdmin->AddFooter(

    array(

        array("title"=>GetMessage("MAIN_ADMIN_LIST_SELECTED"),
"value"=>$rsData->SelectedRowsCount()),

        array("counter"=>true,
"title"=>GetMessage("MAIN_ADMIN_LIST_CHECKED"), "value"=>"0"),

    )

);




$aContext = array(

    array(

        "TEXT"=>GetMessage("MAIN_ADD"),

        "LINK"=>"neuro_net_edit.php?lang=".LANG,

        "TITLE"=>GetMessage("POST_ADD_TITLE"),

        "ICON"=>"btn_new",

    ),

);

$lAdmin->AddAdminContextMenu($aContext);

$lAdmin->CheckListMode();


$APPLICATION->SetTitle("Список нейронных сетей");


require($_SERVER["DOCUMENT_ROOT"]."/bitrix/modules/main/include/prolog_admin_
after.php");

?>


<?$lAdmin->DisplayList();?>
```

```
<?require($_SERVER["DOCUMENT_ROOT"]."/bitrix/modules/main/include/epilog_admi
n.php");?>


<?function deleteNet($idNet){

    global $DB;

    $strSql = "SELECT * FROM a_list_neuro_nets where NET_ID='".$idNet."';";

    $nets = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

    while($net = $nets->NavNext(true, "f_")){

        $strSql = "SELECT * FROM a_list_neuro_layers where
NET_ID='".$idNet."';";

        $layers = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

        while($layer = $layers->NavNext(true, "f_")){

            $strSql = "SELECT * FROM a_list_neuro_neurons where
LAYER_ID='".$layer['LAYER_ID']."';";

            $neurons = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);

            while($neuron = $neurons->NavNext(true, "f_")){

                //удаление весов

                $strSql = "DELETE FROM a_list_neuro_weights where
NEURON_ID='".$neuron['NEURON_ID']."';";

                $rsData = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);

            }

            //удаление нейронов

            $strSql = "DELETE FROM a_list_neuro_neurons where
LAYER_ID='".$layer['LAYER_ID']."';";

            $rsData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

        }

        //удаление слоев

        $strSql = "DELETE FROM a_list_neuro_layers where
NET_ID='".$idNet."';";
```

```
        $rsData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

    }

    //удаление сети

    $strSql = "DELETE FROM a_list_neuro_nets where NET_ID='".$idNet."';";

    $rsData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

}

?>
```

1.6 Файл admin/neuro_net_edit.php

```
<?

require_once($_SERVER["DOCUMENT_ROOT"]."/bitrix/modules/main/include/prolog_a
dmin_before.php");


IncludeModuleLangFile(__FILE__);

require($_SERVER["DOCUMENT_ROOT"]."/bitrix/modules/main/include/prolog_admin_
after.php");

$endlayers=0;

//если id то получаем данные

if($ID){

    //получаем наши данные

    $strSql = "SELECT * FROM a_list_neuro_nets WHERE NET_ID=$ID;";

    $rsData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

    //конечные данные

    $arRes = $rsData->NavNext(true, "f_");

    //получаем и слои

    $strSql = "SELECT * FROM a_list_neuro_layers WHERE NET_ID=$ID;";

    $layersData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);
```

```
    while($layersRes = $layersData->NavNext(true, "f_")){

        $strSql = "SELECT COUNT(NEURON_ID) as countNeurons FROM
a_list_neuro_neurons WHERE LAYER_ID='".$layersRes['LAYER_ID']."';";

        $neuronsData = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);

        $countNeurons = $neuronsData->NavNext(true, "f_");

        $layers[$layersRes['NUMBER']]=$countNeurons['countNeurons'];

        $endlayers++;

    }

}

//отступы

$aTabs = array(

    array("DIV" => "edit1", "TAB" => "Настройки нейронной сети",
"ICON"=>"main_user_edit", "TITLE"=>"Настройки нейронной сети"),

);

$tabControl = new CAdminTabControl("tabControl", $aTabs);

$aMenu = array(

    array(

        "TEXT"=>"Список сетей",

        "TITLE"=>"Список сетей",

        "LINK"=>"neuro_net.php?lang=".LANG,

        "ICON"=>"btn_list",

    )

);

$context = new CAdminContextMenu($aMenu);

$context->Show();

//если нажимаем применить то сохраняем данные

if($_POST['apply']){

    //изменения

    if($_POST['ID']){
```

```
        $strSql = "UPDATE a_list_neuro_nets SET NAME='".$_POST['nameNet']."'
WHERE NET_ID=$ID";

        $rsData = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

    }else{

        //создаем сеть

        $arFields = array(

            "NAME" => "'".trim($_POST['nameNet'])."'",

            "COUNT_ENTER" => "'".trim($_POST['countEnters'])."'"

            );

        $netID = $DB->Insert("a_list_neuro_nets", $arFields,
$err_mess.__LINE__);

        //создаем уровни

        foreach($_POST['layer'] as $number=>$count){

            $arFields = array(

                "NUMBER" => "'".trim($number)."'",

                "NET_ID" => "'".trim($netID)."'"

                );

            $layerID = $DB->Insert("a_list_neuro_layers", $arFields,
$err_mess.__LINE__);

            //создаем нейроны

            for($i=0;$i!=$count;$i++){

                $arFields = array(

                    "NUMBER" => "'".trim($i)."'",

                    "LAYER_ID" => "'".trim($layerID)."'"

                    );

                $neuronID = $DB->Insert("a_list_neuro_neurons", $arFields,
$err_mess.__LINE__);

                //создаем веса

                if($number==0){

                    //первый слой

                    echo "FIRSTLevel".$_POST['countEnters']."<br>";
```

```
                    for($j=0;$j!=$_POST['countEnters'];$j++){
                        echo $j." ".(rand(10, 30)/100)." ".$neuronID."<br>";
                        $arFields = array(
                            "NUMBER" => "'".trim($j)."'",
                            "WEIGHT" => "'".(rand(10, 30)/100)."'",
                            "NEURON_ID" => "'".trim($neuronID)."'"
                        );
                        $weightID = $DB->Insert("a_list_neuro_weights",
$arFields, $err_mess.__LINE__);
                    }
                }else{
                    //другие слои
                    for($j=0;$j!=$prevCount;$j++){
                        echo $j." ".(rand(10, 30)/100)." ".$neuronID."<br>";
                        $arFields = array(
                            "NUMBER" => "'".trim($j)."'",
                            "WEIGHT" => "'".(rand(10, 30)/100)."'",
                            "NEURON_ID" => "'".trim($neuronID)."'"
                        );
                        $weightID = $DB->Insert("a_list_neuro_weights",
$arFields, $err_mess.__LINE__);
                    }
                }
            }
            $prevCount=$count;
        }
    }
    LocalRedirect("neuro_net_edit.php?ID=".$netID."&SHOW=1");
}
?>
```

```
<form method="POST" Action="<?echo $APPLICATION->GetCurPage()?>"
ENCTYPE="multipart/form-data" name="post_form">

<?

$tabControl->Begin();

$tabControl->BeginNextTab();

?>

<tr>

    <td width="40%"><?="Название сети:"?></td>

    <td width="60%"><input type="text" name="nameNet" <?if($_GET['SHOW'])
echo 'disabled';?> value="<?=$arRes['NAME']?> "></td>

</tr>

<tr>

    <td width="40%"><?="Количество входов:"?></td>

    <td width="60%"><input type="text" name="countEnters" <?if($_GET['SHOW'])
echo 'disabled';?> value="<?=$arRes['COUNT_ENTER']?>"></td>

</tr>

<tr>

    <td><?="Слои:"?></td>

    <td>

        <div id="parentId">

            <div>

                <?foreach($layers as $number=>$countNeuron){

                    echo "Номер ".$number." - Количество нейронов: ";

                    if($_GET['SHOW'])

                    echo "<input name='layer[$number]' type='text' disabled
value='$countNeuron'/><br>";

                    else

                    echo "<input name='layer[$number]' type='text'
value='$countNeuron'/><br>";

                }?>

            </div>

        </div>
```

```
        <?if($_GET['SHOW']){?>

        <a style="color:red;" onclick="alert('Необходимо создать новую
сеть')" href="#">[+]</a>

        <?}else{?>

        <a style="color:green;" onclick="return addField()" href="#">[+]</a>

        <?}?>

    </td>

</tr>


<input type="hidden" name="ID" value="<?=$ID?>" >


</form>

<script>

var countOfFields = <?=$endlayers?>-1; // Текущее число полей

var curFieldNameId = <?=$endlayers?>-1; // Уникальное значение для атрибута
name

var maxFieldLimit = 25; // Максимальное число возможных полей

function deleteField(a) {

  if (countOfFields > 1)

  {

 // Получаем доступ к ДИВу, содержащему поле

 var contDiv = a.parentNode;

 // Удаляем этот ДИВ из DOM-дерева

 contDiv.parentNode.removeChild(contDiv);

 // Уменьшаем значение текущего числа полей

 countOfFields--;

 }

 // Возвращаем false, чтобы не было перехода по ссылке

 return false;

}

function addField() {
```

```
 // Проверяем, не достигло ли число полей максимума
 if (countOfFields >= maxFieldLimit) {
 alert("Число полей достигло своего максимума = " + maxFieldLimit);
 return false;
 }
 // Увеличиваем текущее значение числа полей
 countOfFields++;
 // Увеличиваем ID
 curFieldNameId++;
 // Создаем элемент ДИВ
 var div = document.createElement("div");
 // Добавляем HTML-контент с пом. свойства innerHTML
 div.innerHTML = "Номер "+curFieldNameId+" - Количество нейронов: <input
name='layer["+curFieldNameId+"]' type='text'/>";
 // Добавляем новый узел в конец списка полей
 document.getElementById("parentId").appendChild(div);
 // Возвращаем false, чтобы не было перехода по ссылке
 return false;
}
</script>
<?
// завершение формы без вывода кнопок
$tabControl->Buttons();
if(!$_GET['SHOW']){
?>
<input class="button"
       type="submit" name="apply"
       value="Сохранить"
       title="Моя кнопка для сохранения" />
<?}
```

```
$tabControl->End();

?>


<?

$tabControl->ShowWarnings("post_form", $message);

?>

<?require($_SERVER["DOCUMENT_ROOT"]."/bitrix/modules/main/include/epilog_admi
n.php");?>
```

2 Классы модуля ПМ АДН

2.1 Класс Neuron

```
<?

/**

 * Класс, реализующий модель нейрона

 */

class Neuron{


    public $weight;

    public $outers;

    public $summ;

    public $errorOnNeuron;

    public $entersToStudy;

    public $deltaWeight;



    /**

    * Конструктор класса

    *
```

```
* @param int $lenghtEntersy
*/
function __construct($lenghtEnters){
    for($i=0;$i!=$lenghtEnters;$i++){
        $this->weight[$i]= rand(10, 30)/100;
    }
}

/**
* Функция активации
*
* @param int $x
*
* @return int
*/
function active($x){
    //тангенциальная
    return tanh($x);
    //экспоненциальная
    //return 1/(1+exp(-$x));
}

/**
* Функция вычисления выхода
*
* @param array $enters
*/
function solve($enters){
    $this->entersToStudy=$enters;
    $this->summ=0;
```

```
    for($i=0;$i<count($enters);$i++){
        $this->summ+=$enters[$i]*$this->weight[$i];
    }
    $this->outers=$this->active($this->summ);
}

/**
* Функция вывода данных нейрона на экран
*/
function display(){
    echo " summ ";
    echo $this->summ;
    echo " outers ";
    echo $this->outers;
    echo " weidth ";
    echo print_r($this->weight);
    echo " enters ";
    echo print_r($this->entersToStudy);
    echo " error ";
    echo $this->errorOnNeuron;
    echo "<br>";
}

/**
* Функция получения весов нейрона
*
* @return array
*/
function getWidth(){
    $result;
```

```
        $count=0;
        foreach ($this->weight as $weightOne){
            $result[$count]=$weightOne;
            $count++;
        }
        return $result;
    }

    /**
    * Функция установки весов нейрона
    *
    * @param array $weight
    */
    function setWidth($weight){
        $this->weight=$weight;
    }
}
```

2.2 Класс NeuroLayer

```
<?
/**
 * Класс, реализующий модель слоя из нейронов
 */
class NeuroLayer{

    public $outersLayer;
    public $neurons;
```

```
/**
* Конструктор класса
*
* @param int $countNeurons
* @param int $countEnters
*/
function __construct($countNeurons,$countEnters){
    for($i=0;$i!=$countNeurons;$i++){
            $this->neurons[$i]=new Neuron($countEnters);
    }
}

/**
* Функция вычисления выхода
*
* @param array $enters
*/
function solve($enters){
    for($i=0;$i!=count($this->neurons);$i++){
            $this->neurons[$i]->solve($enters);
            $this->outersLayer[$i]=$this->neurons[$i]->outers;
    }

}

/**
* Функция вывода данных слоя на экран
*/
function display(){
    echo count($this->neurons)."<br>";
```

```
    foreach ($this->neurons as $neuron){
        $neuron->display();
    }
}

/**
* Функция получения весов нейронов данного слоя
*
* @return array
*/
function getWidth(){
    $result;
    $count=0;
    foreach ($this->neurons as $neuron){
        $result[$count]=$neuron->getWidth();
        $count++;
    }
    return $result;
}

/**
* Функция установки весов нейронов данного слоя
*
* @param array $weight
*/
function setWidth($weight){
    for($i=0;$i!=count($this->neurons);$i++){
        $this->neurons[$i]->setWidth($weight[$i]);
    }
```

```
    }
}
```

2.3 Класс NeuroNet

```
<?
/**
 * Класс, реализующий нейронную сеть
 */
class NeuroNet{

    public $outerNet;
    public $layers;
    public $minMaxToNormalize;

    /**
    * Конструктор класса,принимает массив первым элементом которого является число входов, а последующие число нейронов на скрытых слоях
    *
    * @param int $countNeuronsOnLayer
    */
    function __construct($countNeuronsOnLayer){
        for($i=1;$i!=count($countNeuronsOnLayer);$i++){
            $this->layers[$i-1]=new NeuroLayer($countNeuronsOnLayer[$i],$countNeuronsOnLayer[$i-1]);
        }
    }

    /**
    * Функция производной функции активации
    *
    * @param int $x
```

```
*
* @return int
*/
function proiz($x){
        //тангенциальная
        return 1/(cosh($x)*cosh($x));
        //экспоненциальная
        //return exp(-$x)/((1+exp(-$x))*(1+exp(-$x)));
}

/**
* Функция вычисления выхода
*
* @param array $enters
*/
function solve($enters){
    $this->layers[0]->solve($enters);
    for($i=1;$i!=count($this->layers);$i++){
            $this->layers[$i]->solve($this->layers[$i-1]->outersLayer);
    }
    $this->outerNet=$this->layers[count($this->layers)-1]->outersLayer;
}

/**
* Функция обучения на одном наборе данных,один набор входов один набор
выходов
*
* @param array $enters
* @param array $outerMassive
* @param int $countOne
```

```
    *

    * @return double

    */

    function studyOne($enters,$outerMassive,$countOne){

        //решаем

        $this->solve($enters);

        $globalerror=0;

        //получим выходы и вычислим ошибку на каждом нейроне выходного слоя

        for($i=0;$i!=count($outerMassive);$i++){

            $errorOnEndLayer[$i]=0;

            $errorOnEndLayer[$i]=($outerMassive[$i]-$this-
>outerNet[$i])*$this->proiz($this->layers[count($this->layers)-1]-
>neurons[$i]->summ);

            $this->layers[count($this->layers)-1]->neurons[$i]-
>errorOnNeuron=$errorOnEndLayer[$i];

            if($globalerror<abs($outerMassive[$i]-$this->outerNet[$i])){

                $globalerror=abs($outerMassive[$i]-$this->outerNet[$i]);

            }

        }

        for($numberLayer=count($this->layers)-2;$numberLayer>-1;$numberLayer-
-){

            $numberLayerPrev=$numberLayer+1;

            for($i=0;$i!=count($this->layers[$numberLayer]->neurons);$i++){

                //перебираем все ошибки на предыдущем слое

                $errorOnNeuron[$i]=0;

                for($j=0;$j!=count($this->layers[$numberLayerPrev]-
>neurons);$j++){

                    $errorOnNeuron[$i]+=$this->layers[$numberLayerPrev]-
>neurons[$j]->errorOnNeuron*$this->layers[$numberLayerPrev]->neurons[$j]-
>weight[$i];

                }

                //конечная ошибка на каждом нейроне
```

```
                $errorOnNeuron[$i]=$errorOnNeuron[$i]*$this->proiz($this-
>layers[$numberLayer]->neurons[$i]->summ);

                $this->layers[$numberLayer]->neurons[$i]-
>errorOnNeuron=$errorOnNeuron[$i];

            }

        }

        //прямой проход меняем веса

        for($i=0;$i!=count($this->layers);$i++){

            for($j=0;$j!=count($this->layers[$i]->neurons);$j++){

                //найдем ню по хайникену стр 253, как обратную величину
квадратного корня из суммы весов, РАБОТАЕТ ПЛОХО

                for($k=0;$k!=count($this->layers[$i]->neurons[$j]-
>weight);$k++){

                    $this->layers[$i]->neurons[$j]-
>deltaWeight[$k]=0.4*$this->layers[$i]->neurons[$j]-
>deltaWeight[$k]+0.1*$this->layers[$i]->neurons[$j]->errorOnNeuron*$this-
>layers[$i]->neurons[$j]->entersToStudy[$k];

                    $this->layers[$i]->neurons[$j]->weight[$k]=$this-
>layers[$i]->neurons[$j]->weight[$k]+$this->layers[$i]->neurons[$j]-
>deltaWeight[$k];

                }

            }

        }

        return $globalerror;

    }


    /**

    * функция обучения на обучающей выборке

    *

    * @param array $entersMassive

    * @param array $exitMassive

    * @param int $countEpoch

    * @param int $errorLevel
```

```
    *
    * @return mixed
    */
    function study($entersMassive,$exitMassive,$countEpoch,$errorLevel){
        $gl=0;
        for($j=0;$j<$countEpoch;$j++){
            $gl=0;
            //берем все ключи
            $numberOfStudyArray=array_keys($entersMassive);
            //перемешиваем
            shuffle($numberOfStudyArray);
            for($i=0;$i!=count($entersMassive);$i++)
            {
                $err=$this->studyOne($entersMassive[$numberOfStudyArray[$i]],
$exitMassive[$numberOfStudyArray[$i]],$j);
                if($gl<$err){
                    $gl=$err;
                }
            }
            $massivetoGraf[]=[$j,$gl];
            if($gl<($errorLevel/1.2)){
                return $j;
                break;
            }
        }
        return $gl;
    }


    /**
    * Функция вывода данных сети на экран
```

```
*/
function display(){
    $count=0;
    foreach($this->layers as $layer){
        echo "<br>Layer ".$count."<br>";
        $layer->display();
        $count++;
    }
}

/**
* Функция получения весов нейронов
*
* @return array
*/
function getWidth(){
    $result;
    $count=0;
    foreach($this->layers as $layer){
        $result[$count]=$layer->getWidth();
        $count++;
    }
    return $result;
}

/**
* Функция установки весов нейронов
*
* @param array $weight
*/
```

```
    function setWidth($weight){
        for($i=0;$i!=count($this->layers);$i++){
            $this->layers[$i]->setWidth($weight[$i]);
        }
    }

    /**
    * функция нормализации
    *
    * @param string $name
    * @param int $value
    *
    * @return int
    */
    function getNormalizeValue($name,$value){
        $d1=-0.6;
        $d2=1;
        if (in_array($name,array_keys($this->minMaxToNormalize))){
            $localMinMax=$this->minMaxToNormalize[$name];
            $result=($d2-$d1)*($value-
$localMinMax["MIN"])/($localMinMax["MAX"]-$localMinMax["MIN"])+$d1;
            return $result;
        }else{
            return $value;
        }
    }

    /**
    * функция денормализации
    *
```

```
    * @param string $name
    * @param int $value
    *
    * @return int
    */
    function getDeNormalizeValue($name,$value){
        $d1=-0.6;
        $d2=1;
        if (in_array($name,array_keys($this->minMaxToNormalize))){
            $localMinMax=$this->minMaxToNormalize[$name];
            $result=($value-$d1)*($localMinMax["MAX"]-
$localMinMax["MIN"])/($d2-$d1)+$localMinMax["MIN"];
            return $result;
        }else{
            return $value;
        }
    }
}
```

2.4 Функции работы с базой данных prepareForCMS

```
<?
/**
 * Функция получения весов из базы данных
 *
 * @param int $idNet
 *
 * @return array
 */
function getWidthFromDB($idNet){
```

```
    global $DB;

    $resultMassive;

    $strSql = "SELECT * FROM a_list_neuro_nets where NET_ID='".$idNet."';";

    $nets = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

    while($net = $nets->NavNext(true, "f_")){

        /*echo "<pre>";

        echo print_r($net);

        echo "</pre>";*/

        $strSql = "SELECT * FROM a_list_neuro_layers where
NET_ID='".$idNet."';";

        $layers = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

        while($layer = $layers->NavNext(true, "f_")){

            /*echo "<pre>";

            echo print_r($layer);

            echo "</pre>";*/

            $strSql = "SELECT * FROM a_list_neuro_neurons where
LAYER_ID='".$layer['LAYER_ID']."';";

            $neurons = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);


            while($neuron = $neurons->NavNext(true, "f_")){

                /*echo "<pre>";

                echo print_r($neuron);

                echo "</pre>";*/

                $strSql = "SELECT * FROM a_list_neuro_weights where
NEURON_ID='".$neuron['NEURON_ID']."';";

                $weights = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);

                while($weight = $weights->NavNext(true, "f_")){

                    /*echo "<pre>";

                    echo print_r($weight);
```

```
                    echo "</pre>";*/

$resultWeight[$layer['NUMBER']][$neuron['NUMBER']][$weight['NUMBER']]=$weight
['WEIGHT'];

                }

            }

        }

    }

    return $resultWeight;

}


/**

 * Функция записи весов в базу данных

 *

 * @param int $idNet

 * @param array $weigthOut

 */

function setWidthFromDB($idNet,$weigthOut){

    global $DB;

    $strSql = "SELECT * FROM a_list_neuro_nets where NET_ID='".$idNet."';";

    $nets = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

    $countNet=0;

    while($net = $nets->NavNext(true, "f_")){

        $strSql = "SELECT * FROM a_list_neuro_layers where
NET_ID='".$idNet."';";

        $layers = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

        $countLayer=0;

        while($layer = $layers->NavNext(true, "f_")){

            $strSql = "SELECT * FROM a_list_neuro_neurons where
LAYER_ID='".$layer['LAYER_ID']."';";
```

```
            $neurons = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);


            $countNeuron=0;

            while($neuron = $neurons->NavNext(true, "f_")){

                $strSql = "SELECT * FROM a_list_neuro_weights where
NEURON_ID='".$neuron['NEURON_ID']."';";

                $weights = $DB->Query($strSql, false, "File:
".__FILE__."<br>Line: ".__LINE__);

                $countWeight=0;

                while($weight = $weights->NavNext(true, "f_")){


$weightLocal=$weigthOut[$countLayer][$countNeuron][$countWeight];

                    //echo "<br>".$countLayer." ".$countNeuron."
".$countWeight." = ".$weightLocal;

                    $strSql = "UPDATE a_list_neuro_weights SET
WEIGHT=$weightLocal where WEIGHT_ID='".$weight['WEIGHT_ID']."';";

                    $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

                    $countWeight++;

                }

                $countNeuron++;

            }

            $countLayer++;

        }

        $countNet++;

    }

}


/**

 * Функция получения параметров сети

 *

 * @param int $idNet
```

```
 */
function getNeuro($idNet){
    global $DB;
    $strSql = "SELECT * FROM a_list_neuro_nets where NET_ID='".$idNet."';";
    $nets = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);
    $net = $nets->NavNext(true, "f_");
    //поместим первый вход
    $result[]=$net["COUNT_ENTER"];
    echo "<pre>";
echo print_r($idNet);
echo "</pre>";
    //теперь слои
    $strSql = "SELECT * FROM a_list_neuro_layers where NET_ID='".$idNet."';";
    $layers = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);
    while($layer = $layers->NavNext(true, "f_")){
        $strSql = "SELECT count(NEURON_ID) FROM a_list_neuro_neurons where
LAYER_ID='".$layer['LAYER_ID']."';";
        $neurons = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);
        $neuronsCount = $neurons->NavNext(true, "f_");
        $result[]=$neuronsCount["count(NEURON_ID)"];
    }

    return $result;
}

/**
 * Функция получения параметров нормализации
 *
```

```
 * @param int $idNet
 *
 * @return array
 */
function getNormalizeFromBD($idNet){
    global $DB;
    $strSql = "SELECT * FROM a_list_neuro_normalizes where
NET_ID='".$idNet."';";
    $norms = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);
    while($norm = $norms->NavNext(true, "f_")){
        $result[$norm['NORM_NAME']]['MIN']=$norm['NORM_MIN'];
        $result[$norm['NORM_NAME']]['MAX']=$norm['NORM_MAX'];
    }
    return $result;
}


/**
 * Функция записи параметров нормализации в базу данных
 *
 * @param int $idNet
 * @param int $normilizesMassive
 */
function setNormalizeToBD($idNet,$normilizesMassive){
    global $DB;
    //для проверки на наличие
    $norms=getNormalizeFromBD($idNet);
    foreach($normilizesMassive as $name=>$normalizes)
    {
        if($norms[$name]){
```

```
            $strSql = "UPDATE a_list_neuro_normalizes SET
NORM_MIN='".$normalizes['MIN']."',NORM_MAX='".$normalizes['MAX']."' where
NET_ID='".$idNet." AND NORM_NAME=$name';";

            //echo $strSql;

            $DB->Query($strSql, false, "File: ".__FILE__."<br>Line:
".__LINE__);

        }

        else{

            $arFields = array(

                "NORM_NAME" => "'".trim($name)."'",

                "NORM_MIN" => "'".trim($normalizes['MIN'])."'",

                "NORM_MAX" => "'".trim($normalizes['MAX'])."'",

                "NET_ID" => "'".trim($idNet)."'"

            );

            $netID = $DB->Insert("a_list_neuro_normalizes", $arFields,
$err_mess.__LINE__);

        }

    }

}
```

3 Компонент, реализующий функцию классификации

3.1 Файл .description.php

```
<?

if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true) die();


$arComponentDescription = array(

    "NAME" => "Простая нейронная сеть",

    "DESCRIPTION" => "Многослойный перспетрон",

    "ICON" => "/images/icon.gif",

    "SORT" => 1,
```

```
    "PATH" => array(
        "ID" => "Обработка данных",
        "CHILD" => array(
            "ID" => "Neuro",
            "NAME" => "ПМАД-Н"
        )
    ),
);
?>
```

3.2 Файл .parameters.php

```
<?
if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true) die();


if(!CModule::IncludeModule("iblock"))
    return;


$arTypesEx = CIBlockParameters::GetIBlockTypes(array("-"=>" "));


$arIBlocks=array();
$db_iblock = CIBlock::GetList(array("SORT"=>"ASC"),
array("SITE_ID"=>$_REQUEST["site"], "TYPE" =>
($arCurrentValues["IBLOCK_TYPE"]!="-"?$arCurrentValues["IBLOCK_TYPE"]:"")));
while($arRes = $db_iblock->Fetch())
    $arIBlocks[$arRes["ID"]] = $arRes["NAME"];


$arProperty_LNS = array();
$rsProp = CIBlockProperty::GetList(array("sort"=>"asc", "name"=>"asc"),
array("ACTIVE"=>"Y",
"IBLOCK_ID"=>(isset($arCurrentValues["IBLOCK_ID"])?$arCurrentValues["IBLOCK_I
D"]:$arCurrentValues["ID"])));
```

```
while ($arr=$rsProp->Fetch())
{
    $arProperty[$arr["CODE"]] = "[".$arr["CODE"]."] ".$arr["NAME"];
    if (in_array($arr["PROPERTY_TYPE"], array("L", "N", "S")))
    {
        $arProperty_LNS[$arr["CODE"]] = "[".$arr["CODE"]."] ".$arr["NAME"];
    }
}

//получение сетей
global $DB;
$strSql = "SELECT * FROM a_list_neuro_nets;";
$nets = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line: ".__LINE__);
while($net = $nets->NavNext(true, "f_")){
    $listNet[]="[".$net["NET_ID"]."] ".$net["NAME"];
}

$arComponentParameters = array(
    "GROUPS" => array(
    ),
    "PARAMETERS" => array(
        "STUDY_NET" => array(
            "PARENT" => "BASE",
            "NAME" => "Обучение сети",
            "TYPE" => "CHECKBOX",
            "MULTIPLE" => "N",
            "DEFAULT" => "N"
        ),
        "NEURO_NET" => array(
            "PARENT" => "BASE",
```

```
        "NAME" => "Сеть",
        "TYPE" => "LIST",
        "MULTIPLE" => "Y",
        "VALUES" => $listNet
    ),
    "IBLOCK_TYPE" => array(
        "PARENT" => "BASE",
        "NAME" => "Раздел инфоблока для обучения",
        "TYPE" => "LIST",
        "VALUES" => $arTypesEx,
        "DEFAULT" => "news",
        "REFRESH" => "Y",
    ),
    "IBLOCK_ID" => array(
        "PARENT" => "BASE",
        "NAME" => "Инфоблок для обучения",
        "TYPE" => "LIST",
        "VALUES" => $arIBlocks,
        "REFRESH" => "Y",
    ),
    "ENTERS" => array(
        "PARENT" => "BASE",
        "NAME" => "Входы",
        "TYPE" => "LIST",
        "MULTIPLE" => "Y",
        "VALUES" => $arProperty_LNS
    ),
    "EXITS" => array(
        "PARENT" => "BASE",
        "NAME" => "Выходы",
```

```
            "TYPE" => "LIST",
            "MULTIPLE" => "Y",
            "VALUES" => $arProperty_LNS
        ),
        "LEVEL_ERROR" => array(
            "PARENT" => "BASE",
            "NAME" => "Уровень ошибки",
            "TYPE" => "TEXT",
            "DEFAULT" => 0.1
        ),
        "COUNT_EPOCH" => array(
            "PARENT" => "BASE",
            "NAME" => "Количество итераций",
            "TYPE" => "TEXT",
            "DEFAULT" => 10000
        ),

    )
);
```

3.3 Файл component.php

```
<?
if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true)die();


//подключаем библиотеки
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/neuron.php');
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/neuroLayer.php');
```

```
require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/neuroNet.php');

require_once($_SERVER['DOCUMENT_ROOT'].'/local/modules/SDAM_N/classes/general
/prepareForCMS.php');


//номер сети

$numberNet=$arParams["NEURO_NET"][0]+1;

//номер инфоблока откуда берем данные

$numberStudyIB=$arParams["IBLOCK_ID"];


//выбираем из бд сеть и её структуру

$neuroMlp=new NeuroNet(getNeuro($numberNet));


//записываем входы и выходы в нужном нам виде

foreach($arParams["ENTERS"] as $enter1)

    $entersMassive[]="PROPERTY_".$enter1;

foreach($arParams["EXITS"] as $exit1)

    $exitsMassive[]="PROPERTY_".$exit1;


$arResult["ENTERS"]=$entersMassive;

$arResult["EXITS"]=$exitsMassive;


//если обучение

if($arParams["STUDY_NET"]=="Y"){

    //забираем обучающие данные

    $arSelect = array_merge($entersMassive,$exitsMassive);

    $arFilter = Array("IBLOCK_ID"=>IntVal($numberStudyIB));

    $res = CIBlockElement::GetList(Array(), $arFilter, false,
Array("nPageSize"=>50), $arSelect);

    //добавляем _value

    foreach($entersMassive as $value){
```

```
    $entersNameWithValue[]=$value."_VALUE";
}
foreach($exitsMassive as $value){
    $exitNameWithValue[]=$value."_VALUE";
}
//а теперь перебираем и составляем наборы входы-выходы
while($ob = $res->GetNext()){
    foreach($ob as $name=>$value){
        $localEnters;
        $localExit;
        if (in_array($name, $entersNameWithValue)) {
            $localEnters[]=$value;
            $toNormalizeEnters[$name][]=$value;
        }
        if (in_array($name, $exitNameWithValue)) {
            $localExit[]=$value;
            $toNormalizeExits[$name][]=$value;
        }
    }
    $enters[]=$localEnters;
    unset($localEnters);
    $exit[]=$localExit;
    unset($localExit);
}

//нормализация
//найдем максимумы и минимумы
foreach($toNormalizeEnters as $name=>$value){
    $toNormalizeEntersMaxMin[$name]["MIN"]=min($value);
    $toNormalizeEntersMaxMin[$name]["MAX"]=max($value);
```

```
    }
    foreach($toNormalizeExits as $name=>$value){
        $toNormalizeExitsMaxMin[$name]["MIN"]=min($value);
        $toNormalizeExitsMaxMin[$name]["MAX"]=max($value);
    }
    $d1=-0.6;
    $d2=1;
    foreach($enters as $keyArray=>$enterMess){
        foreach($enterMess as $key=>$enterOne){

$localMinMax=$toNormalizeEntersMaxMin[$arResult["ENTERS"][$key]."_VALUE"];
            $newEntersMassive[$keyArray][$key]=($d2-$d1)*($enterOne-
$localMinMax["MIN"])/($localMinMax["MAX"]-$localMinMax["MIN"])+$d1;
        }
    }
    foreach($exit as $keyArray=>$exitMess){
        foreach($exitMess as $key=>$exitOne){

$localMinMax=$toNormalizeExitsMaxMin[$arResult["EXITS"][$key]."_VALUE"];
            $newExitMassive[$keyArray][$key]=($d2-$d1)*($exitOne-
$localMinMax["MIN"])/($localMinMax["MAX"]-$localMinMax["MIN"])+$d1;
        }
    }
    //наборы получены, теперь пора обучать


    //$mess=$neuroMlp-
>study($enters,$exit,$arParams["COUNT_EPOCH"],$arParams["LEVEL_ERROR"],$neuro
Mlp);
    $minMax=array_merge($toNormalizeExitsMaxMin,$toNormalizeEntersMaxMin);
    setNormalizeToBD($numberNet,$minMax);
    $mess=$neuroMlp-
>study($newEntersMassive,$newExitMassive,$arParams["COUNT_EPOCH"],$arParams["
LEVEL_ERROR"],$neuroMlp);
```

```
    //запишем в бд
    $weigthAfterStudy=$neuroMlp->getWidth();
    setWidthFromDB($numberNet,$weigthAfterStudy);
}


$minMax=getNormalizeFromBD($numberNet);
$neuroMlp->minMaxToNormalize=$minMax;
//решение
//в случае прихода сигнала с формы
foreach($_POST as $name=>$value){
    if(in_array($name, $entersMassive)){
        //порверяем на наличие в массиве нормализации и если надо нормализуем
        echo "<br>value ".$value." normalize ".$neuroMlp-
>getNormalizeValue($name."_VALUE",$value)."<br>";
        $resultEnterFromPost[]=$neuroMlp-
>getNormalizeValue($name."_VALUE",$value);
    }
}
if($resultEnterFromPost){
    //берем веса из бд
    $weightFromDb=getWidthFromDB($numberNet);
    //записываем в сеть
    $neuroMlp->setWidth($weightFromDb);
    $neuroMlp->solve($resultEnterFromPost);
    for($i=0;$i!=count($neuroMlp->outerNet);$i++){
        echo "<br>Exit from form ".$neuroMlp->outerNet[$i]."<br>";
        $neuroMlp->getDeNormalizeValue($exitsMassive[$i]."_VALUE",$neuroMlp-
>outerNet[$i]);


        $arResult["RESULT_SOLVE"][$exitsMassive[$i]]=$neuroMlp-
>getDeNormalizeValue($exitsMassive[$i]."_VALUE",$neuroMlp->outerNet[$i]);
```

```
    }
}

//получаем список русских названий
$rsProp = CIBlockProperty::GetList(array("sort"=>"asc", "name"=>"asc"),
array("ACTIVE"=>"Y", "IBLOCK_ID"=>($numberStudyIB)));
while ($arr=$rsProp->Fetch())
{
    $arResult["normalName"]["PROPERTY_".$arr["CODE"]]= $arr["NAME"];
}

$this->IncludeComponentTemplate();
?>
```

3.4 Файл template.php

```
<?if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true)die();?>



<br>
<div id="all_for_SDAM">
    <div id="header_for_SDAM">Входные данные:</div>
    <form method="POST"  action="">
        <?foreach($arResult["ENTERS"] as $enter){?>
                <div
id="label_for_SDAM"><?=$arResult["normalName"][$enter]?>:</div>
                <input id="input_for_SDAM" name="<?=$enter?>" type="text">
        <?}?>
        <br><input type="submit" id="button_for_SDAM" value="-ешить">
    </form>
```

```
<br>
<hr id="hr_for_SDAM">

<?if($arResult["RESULT_SOLVE"]){?>
<div id="header_for_SDAM">Результат:</div>
    <?foreach($arResult["RESULT_SOLVE"] as $name=>$exitSolve){?>
        <div id="label_for_SDAM"><?=$arResult["normalName"][$name]?>:</div>
        <div id="result_for_SDAM"><?=$exitSolve?></div>
    <?}?>
<?}?>
</div>

<style>
#header_for_SDAM{
    color: #555;
    margin-left: 18px;
    font-size: 20px;
    text-align:center;
}
#label_for_SDAM{
    color: #555;
    margin-left: 18px;
    padding-top: 10px;
    font-size: 14px;
    text-align:left;
}
#all_for_SDAM{
    color: #555;
    display: inline-block;
    margin-left: 18px;
```

```
    padding-top: 10px;

    font-size: 14px;

    text-align:center;


    border-radius: 10px;

    border: 1px solid #66add6;

    box-shadow: 0 1px 2px rgba(0, 0, 0, .3), inset 0 1px 0 rgba(255, 255,
255, .5);

}

#hr_for_SDAM{

    border-radius: 1px;

    box-shadow: 0 1px 1px rgba(0, 0, 0, .3), inset 0 1px 0 rgba(255, 255,
255, .5);

}

#result_for_SDAM{


    color: #777;

    padding-left: 10px;

    margin: 10px;

    margin-top: 12px;

    margin-left: 18px;


    border: 1px solid #c7d0d2;

    border-radius: 2px;

    box-shadow: inset 0 1.5px 3px rgba(190, 190, 190, .4), 0 0 0 5px #f5f7f8;

    -webkit-transition: all .4s ease;

    -moz-transition: all .4s ease;

    transition: all .4s ease;

}
```

```
input[id=button_for_SDAM] {

    margin-top: 10px;

    font-size: 20px;

    font-weight: bold;

    color: #fff;

    background-image: -webkit-gradient(linear, left top, left bottom,
from(#acd6ef), to(#6ec2e8));

    background-image: -moz-linear-gradient(top left 90deg, #acd6ef 0%,
#6ec2e8 100%);

    background-image: linear-gradient(top left 90deg, #acd6ef 0%, #6ec2e8
100%);

    border-radius: 30px;

    border: 1px solid #66add6;

    box-shadow: 0 1px 2px rgba(0, 0, 0, .3), inset 0 1px 0 rgba(255, 255,
255, .5);

    cursor: pointer;

    text-align:center;

}



input[type=name],

input[type=text],

input[type=password] {

    color: #777;

    margin: 10px;

    margin-top: 12px;


    width: 290px;

    height: 35px;

    border: 1px solid #c7d0d2;

    border-radius: 2px;
```

```
    box-shadow: inset 0 1.5px 3px rgba(190, 190, 190, .4), 0 0 0 5px #f5f7f8;
    -webkit-transition: all .4s ease;
    -moz-transition: all .4s ease;
    transition: all .4s ease;
        text-align:center;
}

input[type=name]:hover,
input[type=text]:hover,
input[type=password]:hover {
    border: 1px solid #b6bfc0;
    box-shadow: inset 0 1.5px 3px rgba(190, 190, 190, .7), 0 0 0 5px #f5f7f8;
}

input[type=name]:focus,
input[type=text]:focus,
input[type=password]:focus {
    border: 1px solid #a8c9e4;
    box-shadow: inset 0 1.5px 3px rgba(190, 190, 190, .4), 0 0 0 5px #e6f2f9;
}

#result_for_SDAM:hover {
    border: 1px solid #a8c9e4;
    box-shadow: inset 0 1.5px 3px rgba(190, 190, 190, .4), 0 0 0 5px #e6f2f9;
}

#result_for_SDAM:focus {
    border: 1px solid #a8c9e4;
    box-shadow: inset 0 1.5px 3px rgba(190, 190, 190, .4), 0 0 0 5px #e6f2f9;
}
```

```
</style>

</div>
```

4 Компонент, реализующий функцию предсказания

4.1 Файл .description.php

```
<?
if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true) die();

$arComponentDescription = array(
    "NAME" => "График предсказания",
    "DESCRIPTION" => "График предсказания нейронной сети",
    "ICON" => "/images/icon.gif",
    "SORT" => 1,
    "PATH" => array(
        "ID" => "Обработка данных",
        "CHILD" => array(
            "ID" => "Neuro",
            "NAME" => "ПМАД-Н"
        )
    ),
);
?>
```

4.2 Файл .parameters.php

```
<?
```

```
if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true) die();


if(!CModule::IncludeModule("iblock"))

    return;


$arTypesEx = CIBlockParameters::GetIBlockTypes(array("-"=>" "));


$arIBlocks=array();

$db_iblock = CIBlock::GetList(array("SORT"=>"ASC"),
array("SITE_ID"=>$_REQUEST["site"], "TYPE" =>
($arCurrentValues["IBLOCK_TYPE"]!="-"?$arCurrentValues["IBLOCK_TYPE"]:"")));

while($arRes = $db_iblock->Fetch())

    $arIBlocks[$arRes["ID"]] = $arRes["NAME"];


$arProperty_LNS = array();

$rsProp = CIBlockProperty::GetList(array("sort"=>"asc", "name"=>"asc"),
array("ACTIVE"=>"Y",
"IBLOCK_ID"=>(isset($arCurrentValues["GRAF_IBLOCK_ID"])?$arCurrentValues["GRA
F_IBLOCK_ID"]:$arCurrentValues["ID"])));

while ($arr=$rsProp->Fetch())

{

    $arProperty[$arr["CODE"]] = "[".$arr["CODE"]."] ".$arr["NAME"];

    if (in_array($arr["PROPERTY_TYPE"], array("L", "N", "S")))

    {

        $arProperty_LNS[$arr["CODE"]] = "[".$arr["CODE"]."] ".$arr["NAME"];

    }

}


//получение сетей

global $DB;

$strSql = "SELECT * FROM a_list_neuro_nets;";

$nets = $DB->Query($strSql, false, "File: ".__FILE__."<br>Line: ".__LINE__);
```

```
while($net = $nets->NavNext(true, "f_")){
    $listNet[]="[".$net["NET_ID"]."] ".$net["NAME"];
}

$arComponentParameters = array(
    "GROUPS" => array(
    ),
    "PARAMETERS" => array(
        "GRAF_STUDY_NET" => array(
            "PARENT" => "BASE",
            "NAME" => "Обучение сети",
            "TYPE" => "CHECKBOX",
            "MULTIPLE" => "N",
            "DEFAULT" => "N"
        ),
        "GRAF_NEURO_NET" => array(
            "PARENT" => "BASE",
            "NAME" => "Сеть",
            "TYPE" => "LIST",
            "MULTIPLE" => "Y",
            "VALUES" => $listNet
        ),
        "GRAF_IBLOCK_TYPE" => array(
            "PARENT" => "BASE",
            "NAME" => "Раздел инфоблока для обучения",
            "TYPE" => "LIST",
            "VALUES" => $arTypesEx,
            "DEFAULT" => "news",
            "REFRESH" => "Y",
        ),
```

```
"GRAF_IBLOCK_ID" => array(
    "PARENT" => "BASE",
    "NAME" => "Инфоблок для обучения",
    "TYPE" => "LIST",
    "VALUES" => $arIBlocks,
    "REFRESH" => "Y",
),
"GRAF_ENTERS" => array(
    "PARENT" => "BASE",
    "NAME" => "Обучающие наборы(строится по оси Y)",
    "TYPE" => "LIST",
    "MULTIPLE" => "N",
    "VALUES" => $arProperty_LNS
),
"GRAF_AXIS_X" => array(
    "PARENT" => "BASE",
    "NAME" => "Ось X",
    "TYPE" => "LIST",
    "MULTIPLE" => "N",
    "VALUES" => $arProperty_LNS
),
"GRAF_LEVEL_ERROR" => array(
    "PARENT" => "BASE",
    "NAME" => "Уровень ошибки",
    "TYPE" => "TEXT",
    "DEFAULT" => 0.1
),
"GRAF_COUNT_EPOCH" => array(
    "PARENT" => "BASE",
    "NAME" => "Количество итераций",
```

```
            "TYPE" => "TEXT",
            "DEFAULT" => 10000
        ),
        "GRAF_NAME_GRAPH" => array(
            "PARENT" => "BASE",
            "NAME" => "Название графика",
            "TYPE" => "TEXT"
        ),
        "GRAF_NAME_X" => array(
            "PARENT" => "BASE",
            "NAME" => "Название оси X",
            "TYPE" => "TEXT"
        ),
        "GRAF_NAME_Y" => array(
            "PARENT" => "BASE",
            "NAME" => "Название оси Y",
            "TYPE" => "TEXT"
        ),
    )
);
```

4.3 Файл component.php

```
<?
if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true)die();

//номер сети
$numberNet=$arParams["GRAF_NEURO_NET"][0]+1;

//номер инфоблока откуда берем данные
```

```
$numberStudyIB=$arParams["GRAF_IBLOCK_ID"];


//выбираем из бд сеть и её структуру

$neuroMlp=new NeuroNet(getNeuro($numberNet));

//окно

$window=count($neuroMlp->layers[0]->neurons[0]->weight);


//названия входных данных и x

$arResult["ENTER_NAME"]="PROPERTY_".$arParams["GRAF_ENTERS"];

$arResult["AXIS_X"]="PROPERTY_".$arParams["GRAF_AXIS_X"];


//забираем данные

$arFilter = Array("IBLOCK_ID"=>IntVal($numberStudyIB));

$res = CIBlockElement::GetList(Array(), $arFilter, false,
Array("nPageSize"=>50), array($arResult["ENTER_NAME"],$arResult["AXIS_X"]));

//добавляем value

$entersNameWithValue=$arResult["ENTER_NAME"];

$exitNameWithValue=$arResult["AXIS_X"];

while($ob = $res->GetNext()){

    //рабочий массив получаем всегда

    $enterY[]=$ob[$arResult["ENTER_NAME"]."_VALUE"];

    $enterX[]=$ob[$arResult["AXIS_X"]."_VALUE"];

}


//если обучаем то нормализуем и записываем веса в базу данных

if($arParams["GRAF_STUDY_NET"]=="Y"){

    //нормализуем

$minMax["PROPERTY_".$arParams["GRAF_ENTERS"]]["MAX"]=max($enterY)+max($enterY
)*0.5;
```

```
    $minMax["PROPERTY_".$arParams["GRAF_ENTERS"]]["MIN"]=min($enterY)-
min($enterY)*0.5;

    //записываем в таблицу нормализации

    setNormalizeToBD($numberNet,$minMax);

}

//забираем из таблицы нормализации

$minMax=getNormalizeFromBD($numberNet);


$neuroMlp->minMaxToNormalize=$minMax;


foreach($enterY as &$enterOne){

    $enterOne=$neuroMlp-
>getNormalizeValue("PROPERTY_".$arParams["GRAF_ENTERS"],$enterOne);

}


for($i=0;$i!=(count($enterY)-$window);$i++){

    for($j=0;$j!=$window;$j++){

        $enters[$i][]=$enterY[$i+$j];

    }

    $exit[$i][]=$enterY[$i+$window];

}


//если обучаем

if($arParams["GRAF_STUDY_NET"]=="Y"){

    //наборы получены, теперь пора обучать

    $mess=$neuroMlp-
>study($enters,$exit,$arParams["GRAF_COUNT_EPOCH"],$arParams["GRAF_LEVEL_ERRO
R"],$neuroMlp);

    //запишем в бд

    $weigthAfterStudy=$neuroMlp->getWidth();

    setWidthFromDB($numberNet,$weigthAfterStudy);
```

```
}
//решение
//берем веса из бд
$weightFromDb=getWidthFromDB($numberNet);

//записываем в сеть
$neuroMlp->setWidth($weightFromDb);

//вычисляем последнее окно без предсказанного решения
for($i=(count($enterY)-$window);$i!=count($enterY);$i++){
    $lastEnter[]=$enterY[$i];
}

$neuroMlp->solve($lastEnter);

//пошлем наши данные в график сфомируем нужный массив
for($i=0;$i!=count($enterY);$i++){
    $demonstrationMassive[]=array($enterX[$i],$neuroMlp-
>getDeNormalizeValue("PROPERTY_".$arParams["GRAF_ENTERS"],$enterY[$i]));
}
//добавляем предсказание
//вычисляем предсказанный x
$xPrepose=($enterX[count($enterY)-1]-$enterX[count($enterY)-
2])+$enterX[count($enterY)-1];
$demonstrationMassive[]=[$xPrepose,$neuroMlp-
>getDeNormalizeValue("PROPERTY_".$arParams["GRAF_ENTERS"],$neuroMlp-
>outerNet[0])];

//данные для передачи в шаблон
$arResult['dataMassive']=$demonstrationMassive;
$arResult[' GRAF_NAME_GRAF']=$arParams["GRAF_NAME_GRAPH"];
```

```
$arResult['GRAF_NAME_X']=$arParams["GRAF_NAME_X"];

$arResult['GRAF_NAME_Y']=$arParams["GRAF_NAME_Y"];


$this->IncludeComponentTemplate();

?>
```

4.4 Файл template.php

```
<?if (!defined("B_PROLOG_INCLUDED") || B_PROLOG_INCLUDED!==true)die();?>


<script type="text/javascript"
src="https://www.gstatic.com/charts/loader.js"></script>

    <script type="text/javascript">

        google.charts.load('current', {'packages':['corechart']});

        google.charts.setOnLoadCallback(drawChart);

        var ar = <?php echo json_encode($arResult['dataMassive']) ?>;

        var x=[];

        ar.forEach(function(item, i, ar) {

            if(i==0){

x[i]=[<?=$arResult['GRAF_NAME_X']?>,<?=$arResult['GRAF_NAME_Y']?>,{'type':
'string', 'role': 'style'}];

            }

            x[i+1]=[item[0],Number(item[1]),null];

        });

        //задаем вид последнего элемента

        x[x.length - 1][2]='point { size: 8; shape-type: star; fill-color:
#FFA940; }';


        function drawChart() {

            var data = google.visualization.arrayToDataTable(x);
```

```
            var options = {
                title: <?=$arResult['GRAF_NAME_GRAF']?>,
                curveType: 'function',
                legend: { position: 'bottom' },
                pointSize: 4,
                colors: ['#3D9AD1']
            };

            var chart = new
google.visualization.LineChart(document.getElementById('curve_chart'));

            chart.draw(data, options);
      }
    </script>
<div id="curve_chart" style="width: 900px; height: 500px"></div>
```

ПРИЛОЖЕНИЕ 2

ПРОГРАММНЫЙ МОДУЛЬ АНАЛИЗА ДАННЫХ ДЛЯ ВЕБ-САЙТОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ НЕЙРОННЫХ СЕТЕЙ (ПМ АДН) РУКОВОДСТВО ПРОГРАММИСТА

Москва, 2016

АННОТАЦИЯ

В данном программном документе приведено руководство программиста по использованию ПМ АДН, предназначенного для работы с нейронными сетями в рамках системы управления содержимым «1С-Битрикс».

В данном программном документе, в разделе «Назначение и условия применения программы» указаны назначение и функции, выполняемые программой, условия, необходимые для выполнения программы (объем оперативной памяти, требования к составу и параметрам периферийных устройств, требования к программному обеспечению и т. п.).

В разделе «Характеристика программы» приведено описание основных характеристик и особенностей программы (режим работы, средства контроля правильности выполнения и т. п.).

В данном программном документе, в разделе «Входные и выходные данные» приведено описание организации используемой входной и выходной информации.

В разделе «Сообщения » указаны тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Оформление программного документа «Руководство программиста» произведено по требованиям ЕСПД (ГОСТ 19.504-79).

АННОТАЦИЯ

В данном программном документе приведено руководство программиста по использованию ПМ АДН, предназначенного для работы с нейронными сетями в рамках системы управления содержимым «1С-Битрикс».

В данном программном документе, в разделе «Назначение и условия применения программы» указаны назначение и функции, выполняемые программой, условия, необходимые для выполнения программы (объем оперативной памяти, требования к составу и параметрам периферийных устройств, требования к программному обеспечению и т. п.).

В разделе «Характеристика программы» приведено описание основных характеристик и особенностей программы (режим работы, средства контроля правильности выполнения и т. п.).

В данном программном документе, в разделе «Входные и выходные данные» приведено описание организации используемой входной и выходной информации.

В разделе «Сообщения » указаны тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Оформление программного документа «Руководство программиста» произведено по требованиям ЕСПД (ГОСТ 19.504-79).

1 НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММЫ

1.1 Назначение программы

Технологии анализа данных с каждым днем набирают все большие обороты. Для анализа увеличивающихся объемов данных существует два подхода: «жесткий», основными технологиями которого являются теория автоматов, алгоритмов, и «мягкий», объединяющий в общий класс неточные, приближённые методы решения задач. «Мягкие» методы имеют преимущества при решении задач со слабо структурированной информацией, какой в большинстве своем являются данные получаемые веб-сайтом от пользователя.

В настоящее время получили популярность технологии нейронных сетей, которые входят в «мягкие» методы анализа данных. Их основные преимущества: решение задача при неизвестных закономерностях, устойчивость к шумам, адаптирование под изменения окружающей среды, отказоустойчивость – все эти особенности удачно вписываются в концепцию современного интернета. Используя преимущества нейронных сетей, становится возможным создать эффективный инструмент для решения задач классификации и прогнозирования данных.

Для реализации поставленной задачи создан модуль ПМ АДН. ПМ АДН предназначен для работы с нейронными сетями в рамках системы управления содержимым «1С-Битрикс».

1.2 Функции, выполняемые программой

Программный модуль ПМ АДН реализует функции организации структуры нейронной сети, обучения нейронной сети, вычисления выхода нейронной сети. А также на основании данных функций созданы два компонента, которые в свою очередь реализуют функции классификации и прогнозирования.

1.3 Условия, необходимые для выполнения программы

1.3.1 Объем оперативной памяти

Для выполнения своих функций ПМ АДН достаточно 1 Гбайт оперативной памяти. Но рекомендуется использовать модуль на устройстве имеющем ОЗУ более 16 Гбайт

1.3.2 Требования к составу периферийных устройств

Особых требований к составу периферийных устройств, модуль ПМ АДН не предъявляет.

1.3.3 Требования к параметрам периферийных устройств

Никаких требований к параметрам периферийных устройств, модуль не предъявляет.

1.3.4 Требования к программному обеспечению

Системные программные средства, используемые модулем ПМ АДН, должны быть представлены версиями операционной системы Windows 7, Windows 8 или Unix.

Модуль ПМ АДН предназначен для работы в системе управления содержимым «1С-Битрикс», поэтому перед началом работы необходимо её установить.

1.3.5 Требования к персоналу

Программист должен иметь минимум среднее техническое образование, а также пройти курсы 1С-Битрикс «Разработчик Bitrix Framework».

В перечень задач, выполняемых программистом, должны входить:

1) задача поддержания работоспособности системных программных средств – операционной системы и системы управления содержимым;
2) задача поддержания работоспособности модуля ПМ АДН.

2 ХАРАКТЕРИСТИКА ПРОГРАММЫ

2.1 Описание основных характеристик

2.1.1 Режим работы программы

Режим работы модуля ПМ АДН круглосуточный и непрерывный.

2.1.2 Контроль правильности выполнения программы

Работоспособность модуля ПМ АДН можно проверить с помощью сценария тестирования описанного в таблице 1:

Таблица 1 – Сценарий тестирования работоспособности

Этап тестирования	Шаги тестирования	Ожидаемый результат	Полученный результат
1	Создание сети в административном интерфейсе с параметрами: название: «тестовая сеть 1»,количество входов: 2, слой номер 0: 2, слой номер 1: 1. Нажать кнопку «Сохранить».	Создана нейронная сеть с заданными параметрами	

Продолжение таблицы 1

Этап тестирования	Шаги тестирования	Ожидаемый результат	Полученный результат
2	Добавить на страницу компонент модуля ПМАД-Н со свойствами: Входы и выходы выбрать соответственно обучающим выборкам, для сети необходимой нам топологии это два поля для входов, и одно поле для выхода. Выставить флажок «обучение нейронной сети» Выбрать нейронную сеть «тестовая сеть 1».Оставить «Уровень ошибки» и «Количество итерации» по умолчанию, для параметра «Уровень ошибки» это свойство равно 0.1, «Количество итерации» 10000 итераций.	На странице создан компонент с заданными параметрами	
3	Осуществить перезагрузку страницы для запуска обучения и убрать флажок «обучение нейронной сети».	Происходит ускорение работы страницы и флажок «обучение нейронной сети» не выставлен.	
4	Провести вычисления: Подать на вход (0,0). Подать на вход (0,1). Подать на вход (1,0). Подать на вход (1,1).	Для (0,0) -0.1<выход <0.1; Для (0,1) 0.9<выход <1.1; Для (1,0) 0.9<выход <1.1; Для (1,1) -0.1<выход <0.1;	

Если полученный результат совпадет с ожидаемым, это означает, что модуль работоспособен.

3 Обращение к программе

3.1 Установка модуля

Для установки программы необходимо:

1) установка модуля:
    a. запустить браузер;
    b. войти в административную панель системы «1С-Битрикс»;
    c. перейти на вкладку «Настройки»
    d. в разделе «Настройки продукта» выбрать «Модули»
    e. нажать кнопку «Установить» напротив «Программный модуль анализа данных с помощью технологий нейронных сетей»
2) создание нейронной сети:
    a. перейти на вкладку «Сервисы»
    b. перейти на вкладку «ПМ АДН»
    c. выбрать раздел «Список нейронных сетей»
    d. нажать кнопку «Добавить»
    e. ввести название нейронной сети в поле «Название сети»
    f. ввести количество входов нейронной сети в поле «Количество входов»
    g. с помощью кнопки «[+]» добавить необходимое количество слоев нейронной сети с вводом количество нейронов на каждом

3.2 Установка компонентов

Для установки компонента можно воспользоваться следующей последовательностью действий:

1) установка компонента:
    a. перейти на страницу сайта, где необходимо установить компонент
    b. нажать кнопку «Изменить страницу» на панели администратора
    c. среди имеющихся компонентов выбрать раздел «Обработка данных»
    d. в разделе выбрать модуль ПМ АДН

e. перенести компонент, выполняющий функции прогнозирования, или компонент выполняющий функции классификации на веб-страницу

2) настройка компонента

a. установить флажок «Обучение сети»
b. в списке «Сеть» выбрать созданную ранее сеть
c. выбрать раздел инфоблока для обучения
d. выбрать инфоблок для обучения
e. задать входы и выходы нейронной сети из предлагаемых свойств инфоблока
f. задать уровень ошибки обучения
g. задать количество эпох обучения
h. если выбран компонент прогнозирования, то следует установить ещё названия графика и его осей.
i. провести перезагрузку страницы
j. при достижении удовлетворительных результатов обучения, снять флажок «Обучение сети»

А также с помощью непосредственного внедрения на страницу кода следующего вида для компонента классификации:

```
<?$APPLICATION->IncludeComponent(
	"SM_DNA:classification",
	"",
	Array(
		"COMPONENT_TEMPLATE" => ".default",
		"COUNT_EPOCH" => "10000",
		"ENTERS" => array(NUMBER_ROOM, FLOOR, AREA),
		"EXITS" => array(COST),
		"IBLOCK_ID" => "22",
		"IBLOCK_TYPE" => "apartments",
		"LEVEL_ERROR" => "0.1",
		"NEURO_NET" => array("0"),
		"STUDY_NET" => "Y"
	)
);?>
```

где, пользователю необходимо задать параметры: "COUNT_EPOCH" –количество эпох обучения, "ENTERS" – входы нейронной сети, "EXITS" – выходы нейронной сети, "IBLOCK_ID" – идентификатор информационного блока, "IBLOCK_TYPE" – раздел, где расположен информационный блок, "LEVEL_ERROR" – уровень ошибки, "NEURO_NET" – номер нейронной сети, "STUDY_NET" – флаг обучения нейронной сети.

В свою очередь для компонента прогнозирования:

```
<?$APPLICATION->IncludeComponent(
	"SDAM_N:graf",
	"",
	Array(
		"COMPONENT_TEMPLATE" => ".default",
		"GRAF_AXIS_X" => "AREA",
		"GRAF_COUNT_EPOCH" => "10",
		"GRAF_ENTERS" => "COST",
		"GRAF_IBLOCK_ID" => "22",
		"GRAF_IBLOCK_TYPE" => "apartments",
		"GRAF_LEVEL_ERROR" => "0.01",
		"GRAF_NAME_GRAPH" => "Зависимость стоимости квартир от их
площади",
		"GRAF_NAME_X" => "Площадь, кв.м",
		"GRAF_NAME_Y" => "Стоимость, млн.руб",
		"GRAF_NEURO_NET" => array("1"),
		"GRAF_STUDY_NET" => "Y"
	));?>
```

где, пользователю необходимо задать параметры: "GRAF_AXIS_X" – названия свойства инфоблока, чьи значения будут по оси x," GRAF_COUNT_EPOCH" –количество эпох обучения, " GRAF_ENTERS" – входы нейронной сети, " GRAF_IBLOCK_ID" – идентификатор информационного блока, " GRAF_IBLOCK_TYPE" – раздел, где расположен информационный блок, " GRAF_LEVEL_ERROR" – уровень ошибки, "GRAF_NAME_GRAPH" – название графика, выводимого на страницу, "GRAF_NAME_X" – название оси x, "GRAF_NAME_Y" – название оси y, " GRAF_NEURO_NET" – номер нейронной сети, " GRAF_STUDY_NET" – флаг обучения нейронной сети.

3.3 Выполнение программы

3.3.1 Выполнение функции классификации

Для выполнения классификации необходимо:

1) запустить браузер;
2) перейти на страницу, содержащую компонент классификации модуля ПМ АДН;
3) ввести данные в форму компонента классификации модуля ПМ АДН.

3.3.2 Выполнение функции прогнозирования

Для выполнения классификации необходимо:

1) запустить браузер;
2) перейти на страницу, содержащую компонент классификации модуля ПМ АДН;
3) ввести данные в форму компонента классификации модуля ПМ АДН.

3.3.3 Выполнение функции организации нейронной сети

Для выполнения функции организации нейронной сети необходимо вызвать конструктор класс «NeuroNet»:

```
$neuroMlp = new NeuroNet($countNeuronsOnLayer);
```

где $countNeuronsOnLayer – массив, первым элементом которого является число входов, а последующие число нейронов на скрытых слоях, а переменная $neuroMlp – это переменная содержащая нейронную сеть заданной структуры.

3.3.4 Выполнение функции обучения нейронной сети

Для выполнения функции организации нейронной сети необходимо вызвать функцию класса «NeuroNet» «study»:

```
$mess=$neuroMlp->study($entersMassive, $exitMassive, $countEpoch, $levelError, $neuroMlp);
```

где $entersMassive – массив входов нейронной сети, $exitMassive – массив выходов нейронной сети, $countEpoch – количество эпох обучения, $errorLevel – максимально допустимый уровень ошибки, $neuroMlp – номер нейронной сети, которая проходит обучение.

3.3.5 Выполнение функции вычисления выхода нейронной сети

Для выполнения функции организации нейронной сети необходимо вызвать функцию класса «NeuroNet» «solve», а после вычисления получить выходы с помощью команды «getOuterNet()»:

```
$neuroMlp->solve($enters);
$exit=$neuroMlp->getOuterNet();
```

где $enters – массив входов нейронной сети, $exit – массив выходов нейронной сети.

3.4 Завершение программы

Для завершения работы программы следует удалить используемый компонент модуля с веб-страницы.

Для удаления модуля следует:

4) запустить браузер;
5) войти в административную панель системы «1С-Битрикс»;
6) перейти на вкладку «Настройки»
7) в разделе «Настройки продукта» выбрать «Модули»;
8) нажать кнопку «Удалить» напротив «Программный модуль анализа данных с помощью технологий нейронных сетей».

4 Входные и выходные данные

4.1 Организация используемой входной информации

Входная информация представляется в виде элементов информационных блоков в системе «1С-Битрикс»

4.2 Организация используемой выходной информации

Выходная информация представляется в виде вывода набора выходных значений для компонента классификации и графической информации для компонента прогнозирования.

5 СООБЩЕНИЯ

В таблице 2 указан перечень сообщений, которые появляются в случае неисправности.

Таблица 2 – Перечень возможных неисправностей

Наименование неисправностей	Вероятная причина	Способ устранения
При попытке добавить появляется сообщение «Необходимо создать новую сеть»	Сети после создания не подлежат редактированию	Создать новую сеть с большим количеством слоев

Продолжение таблицы 2

Наименование неисправностей	Вероятная причина	Способ устранения
«За установленное число итераций сеть не обучена, величина ошибки равна» …	Неправильно выбрана структура сети, малое количество итераций	Выбрать другую структуру сети, установить большее количество этапов обучения
«DB query error.»	Ошибка базы данных	Следует воспользоваться встроенным инструментом системы «1С-Битрикс» для проверки и восстановления базы данных запускается из административного раздела сайта Настройки -> Инструменты -> Диагностика -> Проверка БД
«MySQL Query Error: ….. [Out of memory restart server and try again (needed 65528 bytes)]»	Недостаточно объем памяти	Необходимо увеличить объем памяти в настройках MySQL.
«Error connecting to database»	Неправильные настройки подключения	Проверить параметры подключения к базе данных (файл /bitrix/php_interface/dbconn.php); проверить доступность базы данных.

Продолжение таблицы 2

500 - Internal Server Error	Превышение разрешенных прав на хостинге	Необходимо обратиться к веб-сервера
	Наличие лимита по времени на исполнение php-скриптов	
	нарушение конфигурации сервера или попытка использования неразрешенных инструкций	