МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

02.03.03 Математическое обеспечение и администрирование информационных систем

ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

БАКАЛАВРСКАЯ РАБОТА

на тему Разработка системы подбора товаров с использованием нейронной сети Хэмминга

Студент А.Н. Иванов _____

Руководитель В.С. Климов _____

Допустить к защите
Заведующий кафедрой, к.тех.н, доцент, А.В. Очеповский _____

«_____»_____2016 г.

Тольятти 2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**
Кафедра **«Прикладная математика и информатика»**

УТВЕРЖДАЮ
Зав.кафедрой «Прикладная математика и информатика»
_____А.В.Очеповский

«____»_____2016 г.

ЗАДАНИЕ
на выполнение бакалаврской работы

Студент Иванов Алексей Николаевич

1. Тема Разработка системы подбора товаров с использованием нейронной сети Хэмминга

2. Срок сдачи студентом законченной выпускной квалификационной работы 25 мая 2016 г.

3. Исходные данные к выпускной квалификационной работе: рекуррентная нейронная сеть Хэмминга, скриптовый язык общего назначения PHP, система управления базами данных MySQL.

4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов, разделов): анализ состояния вопроса; разработка алгоритма подбора товаров; программная реализация предложенных решений; апробация предложенных решений на реальных данных; выводы по работе.

5. Ориентировочный перечень графического и иллюстративного материала Блок-схемы поясняющие, работу системы подбора товаров; набор формул, объясняющих математический аппарат; демонстрация работы алгоритма на

реальном наборе данных; графики и диаграммы, поясняющие результат работы системы.

6. Дата выдачи задания « 11 » января 2016 г.

Инженер-программист технической поддержки системы поиска и бронирования туров ООО «Ехать»	_____	С.Н. Иванов
Руководитель выпускной квалификационной работы	_____	В.С. Климов
Задание принял к исполнению	_____	А.Н. Иванов

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**
Кафедра **«Прикладная математика и информатика»**

УТВЕРЖДАЮ
Зав.кафедрой «Прикладная
математика и информатика»
_____А.В.Очеповский

«____»_____2016 г.

КАЛЕНДАРНЫЙ ПЛАН
выполнения бакалаврской работы

Студента Иванов Алексей Николаевич
по теме Разработка системы подбора товаров с использованием нейронной сети Хэмминга

Наименование раздела работы	Плановый срок выполнения раздела	Фактический срок выполнения раздела	Отметка о выполнении	Подпись руководителя
1. Анализ состояния вопроса	1.04.2016	1.04.2016	выполнено	
2. Разработка алгоритма подбора товаров	11.04.2016	11.04.2016	выполнено	
3. Программная реализация предложенных решений	25.04.2016	25.04.2016	выполнено	
4. Апробация предложенных решений на реальных данных	9.05.2016	9.05.2016	выполнено	
Оформление пояснительной записки	07.04.2016	07.04.2016	выполнено	
Подготовка доклада и графического	14.04.2016	14.04.2016	выполнено	

материала к защите				
Предварительная защита бакалаврской работы	20.05.2016-20.06.2016	20.05.2016-20.06.2016	выполнено выполнено	
Проверка ВКР в системе «Антиплагиат.ВУЗ»	27.05.2016	27.05.2016	выполнено	
Сдача пояснительной записки ВКР	20.06.2016	20.06.2016	выполнено	

Руководитель выпускной квалификационной работы _____ В.С. Климов

Задание принял к исполнению _____ А.Н. Иванов

Аннотация

Тема данной выпускной квалификационной работы: Разработка системы подбора товаров с использованием нейронной сети Хэмминга.

Данная работа посвящена изучению и разработки новых подходов к поиску информации. В работе предложен алгоритм поиска ближайших объектов к заданным параметрам. Для этого используется математический аппарат нейронной сети Хэмминга. Разработанные в работе подходы по поиску ближайших объектов могут применяться, например, в сфере электронной коммерции с целью подбора товара, наиболее близкого к предпочтениям покупателя.

Таким образом, целью данной выпускной квалификационной работы является повышение удобства использования поиска товаров в интернет-магазинах, путем интеграции интеллектуальной системы подбора товаров основанных на нейронной сети Хэмминга.

Для достижения цели данной выпускной квалификационной работы были поставлены и решены следующие задачи:

– был произведен анализ недостатков существующих методов поиска товара;

– были разработаны подходы для реализации система подбора товаров с использованием нейронной сети Хэмминга;

– практическая реализация предложенных была показана на примере интернет-магазина с настройками фильтрации.

При разработке системы подбора товаров были использованы такие инструменты разработки как скриптовый язык JavaScript, интерпретируемый язык программирования PHP, система управления базами данных MySQL, язык гипертекстовой разметки HTML и веб-сервер Apache.

Выпускная квалификационная работа содержит пояснительную записку объемом 40 страниц, включая 14 рисунков, одну таблицу и 10 формул, одно приложение и список литературы из 23 источников.

Оглавление

Введение 4
1 Анализ состояния вопроса 6
1.1 Анализ существующих разработок для поиска товаров по заданным характеристикам 6
1.2 Актуальность разработки 9
2 Разработка алгоритма подбора товаров 11
2.1 Применение нейронной сети Хэмминга для подбора товаров по заданным характеристикам 11
2.2 Хранение, представление и нормализация данных для использования алгоритмом подбора товаров 19
3 Программная реализация предложенных решений 27
3.1 Выбор программных средств для разработки и тестирования 27
3.2 Реализация системы подбора товаров 32
Заключение 39
Список используемой литературы 40
Приложение А Программный код алгоритмов системы подбора товаров 43

Введение

В сегодняшнее время все полки магазинов перегружены огромным количеством товаров. Увеличение ассортимента, множество параметров, малейшие отличия одного продукта от другого – всё это усложняет поиск необходимого товара. Даже с имеющимися настройками поиска в каталогах перебор всего множества представленного товара в выведенном списке утомляет. После многоминутного поиска и подбора параметров может окончательно пропасть желание покупки.

Объектом исследования выпускной квалификационной работы являются интернет-магазины.

Предметом исследования является улучшение методов поиска и подбора товаров в интернет-магазинах.

Целью выпускной квалификационной работы будет являться повышение удобства использования и эффективности представления ассортимента торговых площадок, путем создания интеллектуальной системы подбора для покупателей товаров наиболее близких к их запросам.

Исходя из цели данной выпускной квалификационной работы, было предпринято решить следующие задачи:

– анализ недостатков существующих методов поиска товара;

– проектирование и разработка системы подбора товаров с использованием нейронной сети Хэмминга;

– разработка тестовой веб-страницы интернет-магазина и внедрение в него разработанной системы подбора товаров.

В ходе выполнения выпускной квалификационной работы будет создана система подбора товаров по заданным характеристикам, в основе которого будет использован алгоритм работы нейронной сети Хэмминга. Системой подбора товара будут являться функции, подключаемые к

странице поиска товаров и базе данных, обрабатывающие пользовательский запрос.

В первой главе будет произведен анализ текущего положения инструментов и способов подбора товаров по запросам пользователей. Будут описаны их недостатки и преимущества разрабатываемой системы.

Во второй главе произойдет описание структуры работы искусственной нейронной сети Хэмминга. В данной главе также будет обоснована пригодность данной нейронной сети для алгоритма системы подбора товаров.

Третья глава будет посвящена разработке веб-страницы на которой можно будет отобразить работу полученных алгоритмов и выбору программных средств для реализации системы подбора товаров.

1 Анализ состояния вопроса

1.1 Анализ существующих разработок для поиска товаров по заданным характеристикам

Анализом пригодности использования (usability) интернет-магазинов занимается независимый исследовательский институт Baymard, расположенный в Копенгагене. Специалисты из этого института занимаются исследованиями, направленными на улучшение качества поиска и фильтрации товаров в интернет-магазинах, выпускают статьи об проведенных опросах и наблюдениях, а так же статьи, описывающие то, как можно улучшить существующие методы поиска интернет товаров [1].

Специалисты из института Baymard так же занимаются исследованием, так называемого опыта пользователя (UX – user experience) – это восприятие и ответные реакции, возникающие при использовании или предстоящем использовании какой-либо информационной системы, это определение включает в себя также все эмоции, предпочтения, убеждения и ощущение во время использования продукции, услуги или системы [2].

Их недавнее исследование крупнейших интернет-магазинов показало, что 16% из них имеют достаточно хорошую систему подбора товаров [3]. Это масштабное исследование проходило на протяжении девяти месяцев. Они оценивали не только сами сайты, но и то, как пользователи ищут на них товары.

Несмотря на то, что тестам подверглись магазины с многомиллионными товарными списками, было выявлено более 700 проблем, связанных с использованием списка товаров, их фильтрацией и сортировкой. Все подмеченные проблемы были проанализированы и разделены на 93 кратких руководства по пригодности использования

товарных списков, 35 из которых специально посвящены доступности фильтров, их дизайну и логике.

В процессе тестирования 50 самых известных и быстрорастущих американских магазинов по 93 факторам было обнаружено, что фильтрация на них настроена не самым лучшим образом. Анализ по 1750 специальным параметрам, относящимся к доступности фильтрации, логики фильтрации и интерфейсам самих фильтров показал следующее:

– 34% интернет-магазинов обладают бедной системой фильтров, сильно ограничивают пользователя в поиске товаров;

– 50% интернет-магазинов имеют удовлетворительные фильтры, неплохие, но все же есть свойства, которые стоило бы улучшить;

– только 16% интернет-магазинов могли предоставить удобную систему фильтрации товаров, но даже в этой группе большинству сайтов есть что улучшить.

Сложность создания фильтров заключается не только в сборе всех необходимых для поиска параметров, но и в реализации интерфейса, с помощью которого было бы удобно и интуитивно понятно осуществлять поиск товаров.

Например, на рисунке 1 представлено огромное количество возможных настроек и фильтров поиска, но на данном сайте нет возможности производить сортировку по выбранным параметрам. При виде такого большого количества параметров для поиска может понадобиться много времени на их изучение, но все они бесполезны, если они будут отрицательно влиять на качество поиска.

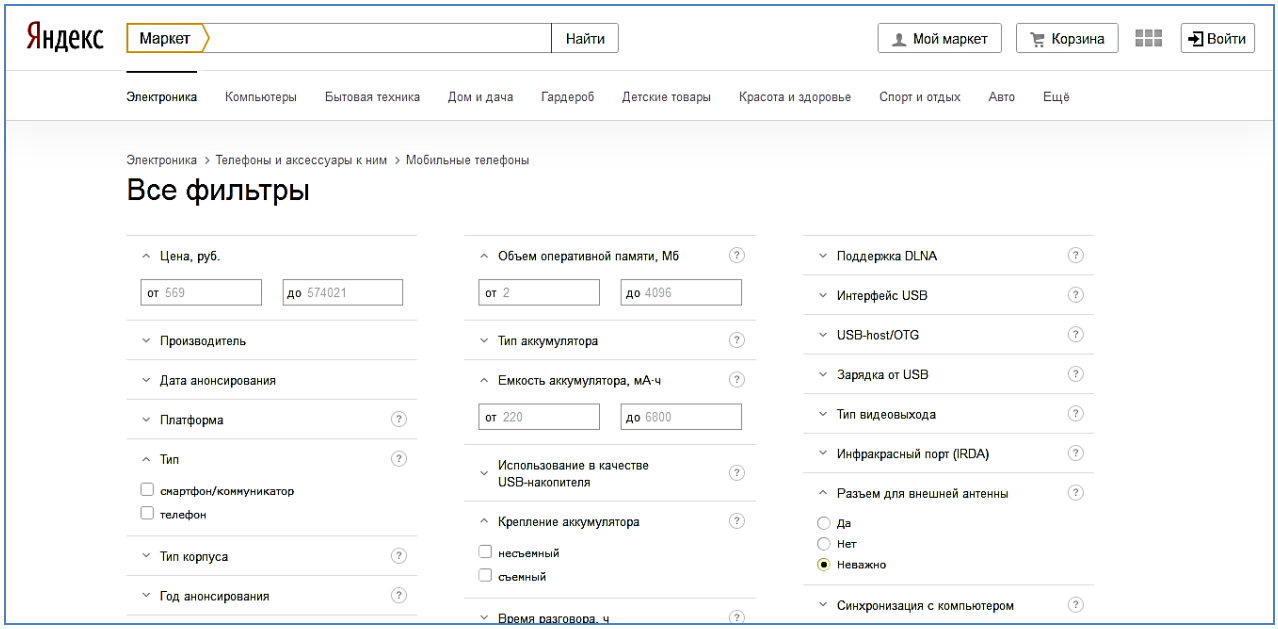

Рисунок 1 – Список фильтров в категории «мобильные телефоны» на сайте market.yandex.ru

Опишем проблему, возникающую при фильтрации списка товаров. Наиболее распространена фильтрация с жестким ограничением по заданному диапазону, такой тип фильтрации не может отобразить того, что вышло за рамки поиска. Демонстрируются только те товары, которые соответствуют четко заданному пользователем диапазону характеристик. Это значит, что товар, который всего лишь на один рубль дороже не будет показан пользователю, и он не сможет его увидеть до тех пор, пока не будут изменены условия поиска, даже если он отлично подходил по другим параметрам. А увеличение диапазона поиска может привести к появлению лишних для пользователя товаров. По сравнению с разрабатываемой системой, такой тип фильтрации не позволяет подбирать товары с параметрами, близкими к запросам пользователя.

Еще одна проблема чаще всего связана с сортировкой списка товаров. Например, пользователь ищет велосипед стоимостью не дороже двадцати тысяч рублей, выставить ограничение по цене не составляет труда в большинстве интернет-магазинов с помощью фильтрации по диапазону. Но когда пользователю будет необходимо найти велосипед с наименьшим весом

и ценою не больше двадцати тысяч, то в большинстве интернет-магазинов у него возникнут трудности.

Другой проблемой существующих способов подбора товара являются сложные запросы к реляционным базам данных. При изменении структуры базы данных или введении нового параметра, программисту приходится разрабатывать сложную систему запросов, учитывать каждый параметр, по которому нужно будет производить поиск. Для реализации всевозможных настроек фильтрации и сортировок программисту, разрабатывающему такую систему, требуется писать и проверять ни один десяток функций.

1.2 Актуальность разработки

С увеличением количества предоставляемых товаров и огромным разнообразием характеристик поиск усложняется, а время, затраченное на поиски искомого товара, может превысить десятки минут. С помощью использования фильтров, размещенных на подавляющем большинстве интернет магазинов, даже с удачным подбором параметров поиска не исключается возможность упущения товаров, которые вам могли бы подойти, но по каким-либо причинам немного вышли за рамки заданных вами параметров. А чрезмерное увеличение диапазона поиска может привести к тому, что будет отображено большое количество неподходящего товара.

Проанализировав существующие способы и методы подбора товаров, были выявлены следующие недостатки:

– фильтрация, которая может исключить товар, если он вышел за пределы диапазона, хотя одного из параметров;

– сортировка, которая производится по одному или двум параметрам, но не учитывает все их одновременно;

– сложные запросы выборки из базы данных – для поддержки достаточного качества применения фильтрации и сортировку к списку товаров необходимо создавать сложную систему запросов.

Поэтому целью разработки будет повышение удобства использования и эффективности представления ассортимента торговых площадок, путем создания интеллектуальной системы подбора товаров для покупателей. Разрабатываемая система будет учитывать все введённые параметры, чтобы на их основе вывести пользователю товары, наиболее близкие к его запросам, при этом областью поиска будут все представленные каталогом товары.

Система подбора товаров, так же как и любая другая программа работает со структурами данных и выполняет алгоритм, который сможет заменить человека в решении определенной задачи. В данном случае интеллектуальную систему подбора товаров можно сравнить с продавцом-консультантом, к которому вы подходите в магазине с просьбой найти товаров подходящий под ваше описание. Сначала продавец вам показывают тот продукт, который наиболее близок к вашему описанию, после чего может предложить те продукты, которые вас бы могли заинтересовать.

В качестве алгоритма реализующего возможность поиска товаров наиболее близких к запросу пользователя была выбрана искусственная нейронная сеть Хэмминга [4].

Разрабатываемая система должна обеспечивать удобство и простоту использования, поддерживать как числовые (например, цена, ширина и вес), так и категориальные (например, тип корпуса, цвет или материал) параметры товара. При необходимости иметь возможность применения четкой фильтрации к списку товаров. Поддерживать работу с большим количеством товаров.

2 Разработка алгоритма подбора товаров

2.1 Применение нейронной сети Хэмминга для подбора товаров по заданным характеристикам

В этом параграфе будет говориться о структуре искусственной нейронной сети Хэмминга и о возможности применения алгоритмов сети для поиска товаров.

Искусственная нейронная сеть (ИНС) – это математическая модель, упрощенно описывающая работу биологических нейронных сетей, ИНС также называют её программную и аппаратную реализацию [5].

Работы по искусственным нейросетевым моделям имеют длинную историю. Разработка детальных математических моделей началась более 70 лет назад с работами таких математиков, как МакКалох и Питтс, Хэбб, Розенблатт.

Более поздние работы Хопфилда, Румельхарта и МакКлелланда, Сежноуски, Фельдмана, Кроссберга и других привела к новому возрождению в этой области. Этот возобновившийся интерес обусловлен развитием новых топологий и алгоритмов, появлением новых методов реализации аналоговых сверхбольших интегральных схем, некоторыми занимательными выступлениями, а также растущим интересом к изучению функционирования человеческого мозга [6].

ИНС широко применяются для распознавания образов, задач классификации, задач оптимизации, прогнозирования, анализа данных, принятия решений и адаптивного управления.

Нейронная сеть Хэмминга была предложена Ричардом Липпманом в 1987 году [7]. Её можно рассматривать как развитие сети Хопфилда [8].Доктор Липпман является сотрудником лаборатории Массачусетского технологического института (MIT), в настоящее время он сосредоточен на

разработке показателей безопасности, с помощью которых можно будет точно оценить уровень опасности исходящий от возможных уязвимостей сетевых устройств. Многие из его разработок в области систем обнаружения вторжений, описанных еще в 1998 и 1999 годах, используются до сих пор, а данные полученные при исследованиях продолжают использоваться для разработок новых систем. В 2011 году получил награду от лаборатории Линкольна за развитие инструментов информационной безопасности, за огромный вклад в области распознавания образов и речи, и изучение искусственных нейронных сетей. Липпман является автором и соавтором более 100 научных работ, отчетов и книг. За статью «Введение в вычисления с нейронными сетями» получил первую премию от института инженеров электротехники и электроники (IEEE) [9].

Нейронная сеть Хэмминга названа именем американского математика Ричарда Уэсли Хэмминга, его называют гением одной идеи за то, что он изобрел коды, которые сами корректировали ошибки, возникающие при передаче данных. Хэмминг вёл активные исследования в области информации и телекоммуникации, но именно эта идея, опубликованная в его единственной научной статье в 1950 году, получила мировую известность. Помимо самокорректирующих кодов им так же было сформулировано расстояние Хэмминга, которое используется в нейронной сети предложенной Липпманом.

Идея функционирования ИНС Хэмминга заключается в выдаче одного из эталонных образцов, который наиболее близок к образу, подаваемому на вход нейронной сети. Притом выдается не сам образ, а лишь его номер. Нейронная сеть работает как классификатор, сравнивая вектор, который поступил на вход, с эталонными векторами, хранящимися в памяти. В качестве меры сравнения используется расстояние Хэмминга. Набор эталонных образов и образ, который подается на вход сети, задаются в виде

бинарных векторов [10]. В качестве образов могут храниться изображения, аудиозаписи или любая другая информация [11].

Расстояние Хэмминга между двумя векторами одинаковой размерности определяется количеством не совпадающих компонент в этих векторах (рисунок 2) [12]. На вход ИНС Хэмминга поступает вектора с биполярными компонентами.

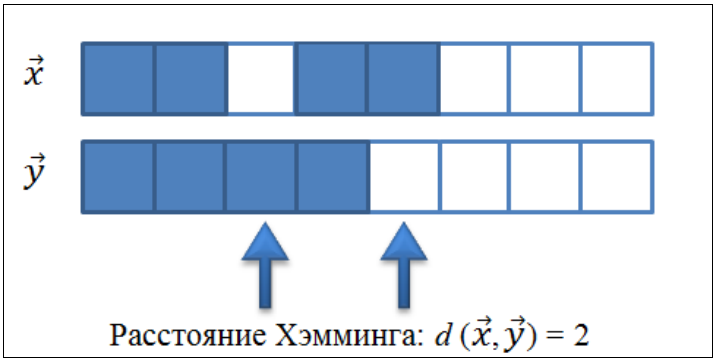

Рисунок 2 – Расстояние Хэмминга между двумя векторами

Как видно на рисунке 3, нейронная сеть Хэмминга состоит из четырех слоев нейронов, в некоторых источниках описывается, что сеть состоит из двух или трёх слоев, такое противоречие возникает из-за того, что входной и выходной слои считаются условными и относятся к другому слою, поэтому могут не обозначаться [7-8]. В литературе нет определенной договоренности относительно того, как считать число слоев в многослойных нейронных сетях. В вычислениях участвуют два слоя, поэтому в дальнейшем будем называть ИНС Хэмминга двухслойной.

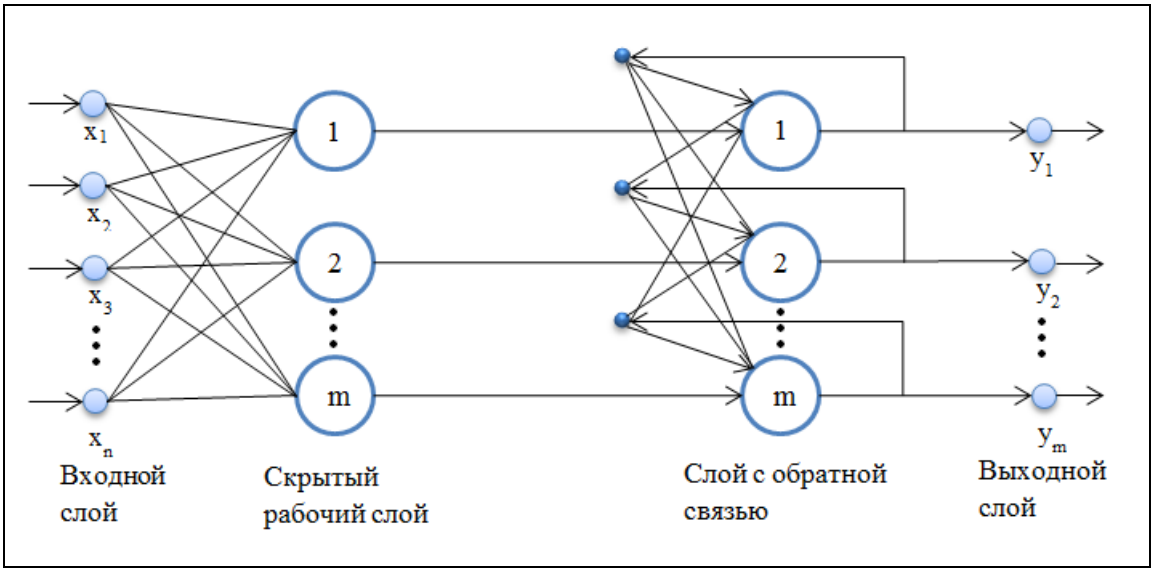

Рисунок 3 – Структура искусственной нейронной сети Хэмминга

Количество нейронов входного слоя n – соответствует количеству бинарных признаков рассматриваемых образов. В первом и втором слоем ИНС количество нейронов равно m, количеству хранимых в сети образов. Например, для поиска в нейронной сети из тысячи товаров, будет использованы слои размерностью в тысячу нейронов. Количество нейронов выходного слоя также равно количеству хранимых эталонных образов [13].

На рисунке 4 более подробно показано устройство первого слоя участвующего в вычислениях. Сигнал от вектора, поступившего на вход, проходит через синапс, который хранит информацию о той же компоненте вектора одного из эталонных образов. Затем происходит суммирование всех поступивших сигналов прошедших через синапсы. После чего полученная сумма поступает в функцию активации, которая рассчитывается по формуле (1). Используется линейная функция активации с насыщением, порог функции берется достаточно большим для того чтобы сигналы с наибольшим значением не сравнялись между собой, так как это может привести к возникновению трудностей в вычислениях второго слоя [7]. Так как порог активационной функции нейрона устанавливается достаточно большим, то иногда в вычислениях его не указывают.

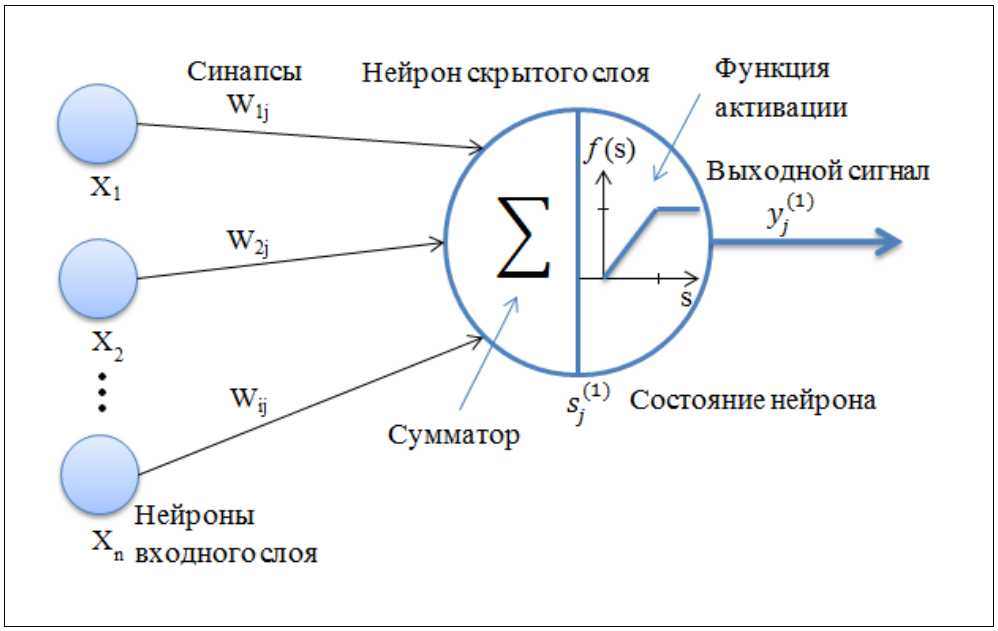

Рисунок 4 – Элемент первого слоя нейронной сети Хэмминга

$$f(s) = \begin{cases} 0, \text{если } s \leq 0, \\ s, \text{если } 0 < s < F, \\ F, \text{если } s \geq F, \end{cases} \qquad (1)$$

где f(s) – функция активации нейрона,

s – состояние нейрона,

F – порог активации нейрона.

Смещение (порог нейрона) используется для того чтобы образы с которыми связь окажется недостаточно сильной не выбывали из вычислений слишком быстро. Определяется по формуле (2), в некоторых моделях нейронных сетей не используется.

$$T = \frac{n}{2}, \qquad (2)$$

где T – смещение,

n – размерность входного сигнала.

Первый слой нейронной сети служит для вычисления расстояния Хэмминга, учитывается количество отличающихся компонент входного вектора от компонент, хранящихся в базе образов [14]. Чем меньше расстояние Хэмминга, тем сильнее получается сигнал. Расчет состояния нейронов первого слоя производится по формуле (3). Верхний индекс указывает на номер слоя в нейронной сети.

$$s_j^{(1)} = \frac{1}{2}\sum_{i=0}^{n-1} w_{ij}\, x_i + T = a, \qquad (3)$$

где $s_j^{(1)}$ – состояние j-го нейрона первого слоя , j = 0, …, m-1,

m – количество хранимых образов,

n – количество компонент входного вектора,

w_{ij} – весовой коэффициент i-го компонента, j-го образа,

x_i – i-ая компонента входного вектора,

T – смещение, по формуле (2),

a – количество одинаковых компонент векторов.

Расчет выходных сигналов нейронов первого слоя производится по формуле (4), эти же значения сигналов становятся состояниями нейронов и значениями выходов второго слоя первой итерации.

$$y_j^{(1)} = s_j^{(2)}(1) = y_j^{(2)}(1) = f\left(s_j^{(1)}\right), \qquad (4)$$

где $y_j^{(1)}$ – выходное значение j-го нейрона первого слоя, j = 0, …, m-1,

m – количество хранимых образов,

$s_j^{(2)}(1)$ – состояние j-го нейрона второго слоя первой итерации,

$y_j^{(2)}(1)$ – выходное значение j-го нейрона второго слоя первой итерации,

f – функция активации, по формуле (1),

$s_j^{(1)}$ – значение состояния j-го нейрона первого слоя, по формуле (3).

Обратная связь в нейронной сети Хэмминга формирует ассоциативную память. Ассоциативность заключается в укреплении связи с теми эталонными образами, которые наиболее похожи на подаваемый в сеть вектор.

На рисунке 5 более подробно отображен второй вычислительный слой нейронной сети Хэмминга. Данный слой реализует нейронную сеть MAXNET, которая находит нейрон с наибольшим сигналом. Каждый нейрон связан с другими нейронами отрицательными синаптическими связями,

синапс с положительным сигналом связывается со своим же нейроном. Такие связи образуются через особый элемент нейрона, называемый точкой ветвления. Отличие этого слоя, от нейронной сети Хопфилда в том, что у нейрона имеется связь с собственным входом.

Коэффициент тормозящих синапсов выбирается из диапазона по формуле (5). Чем больше этот коэффициент, тем быстрее происходит вычисление второго слоя нейронной сети за счет уменьшения количества итераций. Но при этом, большое значение коэффициента тормозящих синапсов ведет к уменьшению точности вычислений.

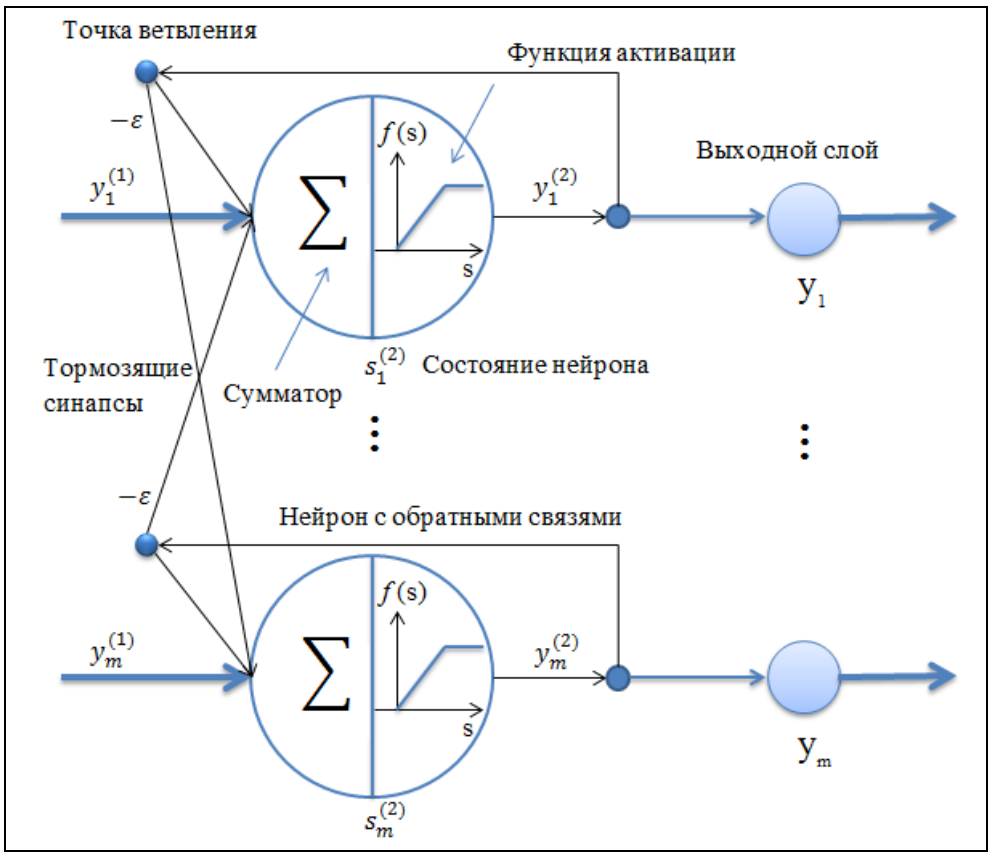

Рисунок 5 – Структура второго вычислительного слоя нейронной сети Хэмминга

где ε – коэффициент тормозящих синапсов,

m – количество хранимых образов в сети.

Расчет состояния нейронов второго слоя производится по формуле:

$$s_j^{(2)}(p+1) = y_j^{(2)}(p) - \varepsilon \sum_{\substack{i=1 \\ j \neq i}}^{m} y_j^{(2)}(p), \qquad (6)$$

где $s_j^{(2)}(p+1)$ – новое состояние j-го нейрона второго слоя,

p – номер итерации,

$y_j^{(2)}(p)$ – предыдущее состояние j-го нейрона второго слоя,

ε – коэффициент тормозящих синапсов, по формуле (5),

m – количество нейронов второго слоя.

Данный слой работает до тех пор, пока в сети не останется один активный нейрон, имевший наибольший сигнал на входе. При работе второго слоя нейронной сети слабые сигналы постепенно обнуляются и становятся не активными, а в выходном слое активируется только один нейрон, принадлежащий тому эталонному образу, чей вектор наиболее близок по расстоянию Хэмминга к вектору образа, поступившего на вход.

Значения выходных сигналов нейронов второго слоя определяется по формуле:

$$y_j^{(2)}(p) = f\left(s_j^{(2)}(p)\right), \qquad (7)$$

где $y_j^{(2)}(p)$ – выходное значение j-го нейрона первого слоя, j = 0, …, m-1,

m – количество хранимых образов,

p – номер итерации,

$s_j^{(2)}(p)$ – состояние j-го нейрона второго слоя первой итерации, по формуле (6)

f – функция активации, по формуле (1).

После рассмотрения структуры нейронной сети Хэмминга стало ясно, что нейронная сеть Хэмминга сначала может выдавать товары, которые наименее похожи с заданным прототипом, постепенно приближаясь к товару с наибольшим сигналом. Таким образом, можно помещать в стек номера образов постепенно выбывающих из вычислений, а последним элементом в

стеке окажется номер образа, который наиболее близок к искомому пользователем товару.

Преимуществами ИНС Хэмминга над другими нейронными сетями является то, что данная сеть не требует много итерационного обучения, по одной и той же обучающей выборке всегда получаются одни и те же весовые коэффициенты.

2.2 Хранение, представление и нормализация данных для использования алгоритмом подбора товаров

Все веб-сайты хранят в себе определенную информацию, для интернет-магазинов просто необходимо иметь базу данных, в которой находятся сведения о товарах.

Компьютерная база данных (БД) представляет собой хранилище объектов. В одной базе данных может содержаться несколько таблиц. БД это средство сбора, хранения и организации информации, в базе данных могут содержаться различные сведения о продуктах [15].

Для поддержания целостности и безопасности данных, а так же возможности их изменения, управления и переноса удобнее всего использовать специально для этого разработанные системы.

Система управления базами данных (СУБД) – это совокупность программных и языковых средств, которая осуществляет доступ к данным, позволяет их создавать, менять и удалять, обеспечивает безопасность и целостность данных. СУБД позволяет создавать базы данных и манипулировать информацией содержащейся в них. Осуществляет доступ к данным СУБД посредством специальных языков, например SQL (от англ. Structured Query Language – язык структурированных запросов) [16].

К основным функциям СУБД относятся:

– определение данных;

– обработка данных;

– управление данными.

Любая СУБД позволяет выполнять следующие операции с данными:

– добавление записей в таблицу;

– удалений записей из таблицы;

– обновление значений в полях одной или нескольких записей;

– поиск в таблице записей удовлетворяющих заданному условию.

По способу доступа СУБД делятся на несколько типов архитектур:

– файл-серверные – файлы данных располагаются непосредственно на файл-сервере, а система управления базами данных установлена на каждом клиентском компьютере. Доступ к файл-серверу осуществляется с помощью компьютерной сети, а чтение и обновление происходит с помощью файловых блокировок. Положительной стороной этой архитектуры является невысокие требования к серверу за счет низкой нагрузки на его процессор. При интенсивном обмене данных снижаются такие характеристики, как надежность, доступность и безопасность, поэтому используются в небольших локальных сетях. Одним из примеров файл-серверной СУБД является Microsoft Access от корпорации Microsoft. Считается устаревшей на данный момент;

– клиент-серверные – архитектуры, в которых СУБД располагается на сервере вместе с базой данных. Клиент может выполнять некую предварительную обработку данных перед отправкой их на сервер, но основная его функция это обеспечение доступа пользователя к серверу. Все запросы пользователей обрабатываются на сервере, поэтому имеются повышенные требования к производительности и надежности сервера, на котором будет определена архитектура данного типа.

– встраиваемые – архитектура систем управления базами данных, в которых СУБД является составной частью прикладной программы, работает на том же компьютере, где и установлена. Чаще всего встраиваемые СУБД рассчитаны на локальное использование данных. Используются там, где

требуется хранить достаточно большие массивы данных, при этом открытый доступ к хранимым данным с остальных компьютеров не обязателен.

Так как на веб-сайт интернет-магазина, потенциальные пользователи могут попадать на него случайно, использование каких-либо сторонних программ или приложений для использования функций поиска не представляется возможным, поэтому чаще всего используется клиент-серверная СУБД. Клиентом для пользователей веб-сайта послужит их браузер, а сервером будет совокупность аппаратного и программного обеспечения установленного на оборудовании владельца сайта.

Многие базы данных изначально представляют собой список в текстовом процессоре или электронной таблице. По мере того как список разрастается, в нем накапливаются излишние и противоречивые данные. В форме списка эти данные становится все труднее понять, а возможности поиска или извлечения подмножеств данных для просмотра весьма ограничены. Поэтому для более рационального и эффективного хранения информации используются модели данных.

Модель данных – это концептуальное описание предметной области. Она включает в себя определения сущностей и атрибутов. Например, сущность – телефон, а атрибутами у него будет считаться такая информация как – производитель, размер экрана и вес. В классической теории баз данных, модель данных есть формальная теория представления и обработки данных в системе управления базами данных (СУБД).

Для представления о товарах используется БД, основанная на реляционной модели данных. Реляционная база данных представляет собой множество взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного вида. Все данные представляются в виде набора простых таблиц, которые описывают отдельные общие параметры объекта, например в отдельную таблицу можно вынести информацию о производителе товара.

Немаловажным параметром базы данных является её целостность, общие правила определения целостности БД отсутствуют. В некоторых системах поддерживаются ограничения уникальности значений некоторых полей, но в основном вся поддержка целостности данных возлагается на прикладную программу.

Для идентификации записей используется первичный ключ. Первичным ключом называется набор полей таблицы, комбинация значений которых однозначно определяет каждую запись в таблице.

По внешнему виду таблица базы данных сходна с электронной таблицей, в которой данные располагаются в строках и столбцах. Поэтому электронные таблицы обычно легко импортируются в таблицы базы данных. Основное различие между хранением данных в электронной таблице и в базе данных – способ организации данных. Как правило, хранение эквивалентной информации в БД займет гораздо меньше объёма памяти, чем это бы занимало в электронных таблицах.

Для алгоритма подбора товаров будет формироваться специально созданная таблица, в которой будут храниться только те данные, которые необходимы в вычислениях ИНС Хэмминга. Такой подход к хранению информации называется хранилищем данных. Разработанная БД будет использоваться специально для вычислений системы подбора товаров. Данные будут храниться в нормализованном виде.

Как было сказано ранее для классификации производимой нейронной сетью ИНС Хэмминга работает с векторами с биполярными компонентами входных данных.

Точность распознавания определяется количеством компонент, которые хранят информацию в нормализованном виде. Чем больше компонент может храниться в векторе, тем большее количество значений он может охарактеризовать. Для того чтобы каждый параметр имел одинаковое влияние на процесс распознавания производится нормализация входных

данных. Например, если один параметр товара будет измеряться в диапазоне от 3 до 7, а другой от одной до ста тысячи тысяч, то без проведения нормализации данных второй параметр будет всегда иметь большее влияние на процесс распознавания. Формула нормализации входных данных в общем виде:

$$y = \frac{[(x - x]_{\min})(d_1 - d_2)}{x_{max} - x_{\min}} + d_1, \qquad (8)$$

где y – значение в нормализованном виде,

x – значение которое нужно нормализовать,

$x_{\min}$ – минимальное значение из диапазона параметра x,

d_1 – минимальное значение диапазона, к которому приводится нормализация,

d_2 – максимальное значение диапазона, к которому приводится нормализация,

x_{max} – максимальное значение из диапазона параметра x.

Данный тип нормализации подходит для числовых параметров, на рисунке 6 отображен пример результата нормализации числового параметра диапазоном от одной до одиннадцати тысяч в диапазон от нуля до восьми, который затем отображен в виде множества биполярных компонент.

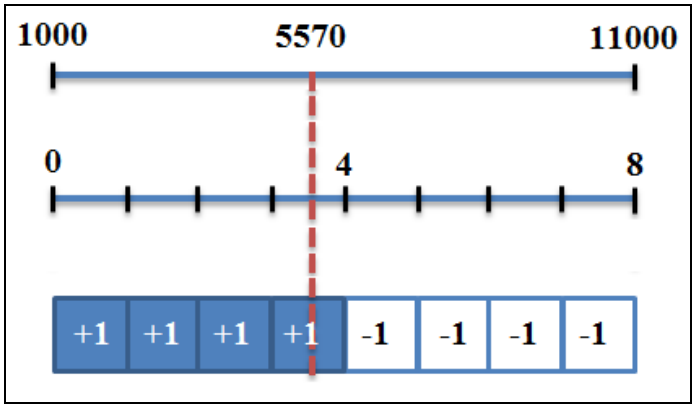

Рисунок 6 – Пример нормализации входных данных

Для хранения информации об определенных признаках для работы с ИНС их можно представить в виде матрицы, где строки это определенные признаки, а столбцы это их значения в виде вектора с биполярными

элементами которые используются как весовые коэффициенты для нейронной сети.

Категориальные признаки определяются другим способом, каждому признаку выделяется определенная компонента вектора, которой присваивается положительное значение. Если размерность вектора превышает количество компонент, то для хранения признака можно выделить две и более компоненты. Наибольшая точность поиска в таком случае достигается, если количество признаков кратно размерности вектора. Если кратность не получается, то необходимо найти наибольший общий делитель числа признаков и размерности вектора и выделить получившееся число на каждый имеющийся категориальный признак, а значения оставшихся компонент для всех оставить одинаковыми.

В ходе нормализации данных о товаре формируется матрица биполярных компонент, в которой хранятся все весовые коэффициенты параметров, используемые для поиска системой подбора товаров (рисунок 7).

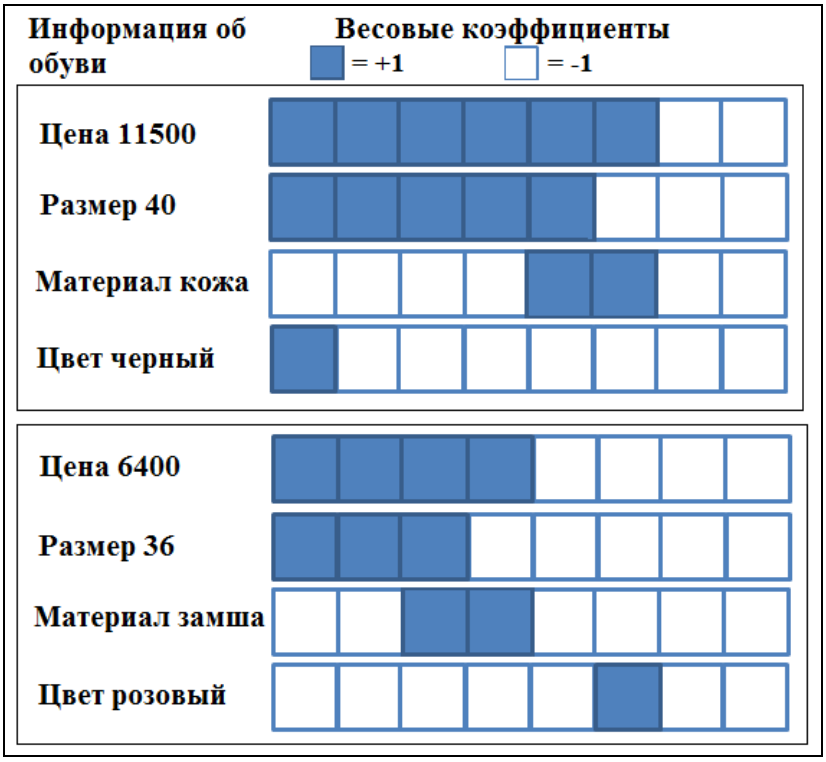

Рисунок 7 – Пример хранения информации в виде матрицы

Для того чтобы каждый параметр одинаково влиял на распознавание образа количество компонент выделенное для каждого параметра должно быть одинаковым. Для хранения значений параметров в бинарном виде может понадобиться большое количество столбцов, что не очень удобно для вычисления значений первого слоя нейронной сети, так как необходимо перебирать сравнивать каждую компоненту, а уменьшение количества столбцов приведет к уменьшению точности работы алгоритмов системы.

При хранении числовых параметров в виде вектора с биполярными компонентами положительные компоненты расположены непрерывно, поэтому для вычисления состояния входного сигнала в скрытый слой ИНС Хэмминга можно использовать формулу:

$$vn_i = n - 2|x_i - w_{ij}|, \qquad (9)$$

где vn_i – значение сигнала от i-го входного числового признака, i = 0,…, n-1,

n – количество компонент входного вектора,

x_i – количество положительных компонент i-го входного числового признака,

w_{ij} – количество положительных компонент i-го признака образа, j-го нейрона, j = 0,…, m,

m – количество нейронов первого слоя.

Для категориальных признаков значение сигнала можно определить по формуле:

$$vp_i = \begin{cases} n - k(l-1), \text{если } w_{ij} \in X, \\ n - k(l+1), \text{если } w_{ij} \notin X, \end{cases} \qquad (10)$$

где vp_i – значение сигнала от i-го входного категориального признака, i = 0,…,n-1,

n – количество компонент входного вектора,

k – количество компонент на определенный признак,

l – количество выбранных пользователем категорий,

w_{ij} – индекс категории i-го признака образа j-го нейрона, j = 0,…, m-1,

X – множество выбранных пользователем признаков категории.

Таким образом, с помощью вычисления формул для хранения информации о каждом признаке товара потребуется не более одного столбца в таблице весовых коэффициентов. Для хранения информации о том, к какому типу описываемой информации относится определенное поле, можно будет создать отдельную таблицу или файл, при чтении которого алгоритм будет выбирать необходимую формулу для вычисления весовых коэффициентов параметров товара.

Отсутствие стохастичности в процессе обучения позволяет, создав один раз матрицу весовых коэффициентов шаблонов не изменять её до тех пор, пока не появятся данные, которые вышли за рамки максимумов и минимумов закодированных ранее значений.

В таблице 1 отображены основные типы для хранения числовых параметров в СУБД MYSQL [16].

Таблица 1 – Требования к памяти для числовых типов MySQL

Тип столбца	Размер, байт	Диапазон
TINYINT	1	от минус 128 до 128 или от 0 до 255
SMALLINT	2	от минус 32768 до 32676 или от 0 до 65535
MEDIUMINT	3	от минус 8388608 до 8388607 или от 0 до 16777215
INT	4	от минус 2147483648 до 2147483647 или от 0 до 4294967295
BIGINT	8	от минус 9223372036854775808 до 9223372036854775807. Диапазон без знака от 0 до 18446744073709551615.
FLOAT(X)	4	от минус 3,402823466E+38 до минус 1,175494351E-38, 0, и от 1,175494351E-38 до 3,402823466E+38
DOUBLE	8	от минус 1,7976931348623157E+308 до минус 2,2250738585072014E-308, 0, и от 2,2250738585072014E-308 до 1,7976931348623157E+308

Как видно из данных таблицы 1, наиболее экономичным типом для хранения нормализованных данных будет являться тип TINYINT. При выборе без знакового диапазона можно хранить информацию для одного числового признака эквивалентную вектору размерностью в 256 компонент. Этого диапазона так же должно хватить для хранения информации о категориальных признаках, так как показывает опыт исследования интернет-

магазинов количество вариантов у категориального признака редко превышает 20.

В процессе выбора средств для хранения данных решено использовать СУБД MySQL с системой хранения данных MyISAM, тип полей для хранения нормализованных данных TINYINT.

3 Программная реализация предложенных решений

3.1 Выбор программных средств для разработки и тестирования

Разработка и тестирование системы подбора товаров будет производиться на наборе дистрибутивов и программной оболочке Денвер (от сокр. ДНВР — джентльменский набор веб-разработчика).

Денвер является WAMP (акроним от Windows, Apache, MySQL и PHP) дистрибутивом, т.е. это комплектация пакетов программ, под соответствующую операционную систему, в данном случае это Microsoft Windows, которая содержит в себе все необходимые для работы связанной с веб-разработкой утилиты. Системы WAMP поставляются в форме пакетов, связывающих компоненты таким образом, чтобы их не нужно было загружать и настраивать по отдельности. Это позволяет в кратчайшие сроки поставить и запустить разработочный сервер с минимальными затратами времени.

В первую очередь этот набор для веб-разработчика предназначен для тестирования и отладки веб-страниц на локальном персональном компьютере (ПК) [17]. Один из больших плюсов этого набора заключается в том, что подключение к интернету для работоспособности сервера не требуется. Последняя версия Денвера поддерживает установку и работу со съемного запоминающего устройства. Это очень удобно, когда нет возможности установить веб-сервер непосредственно на персональный компьютер пользователя.

Веб страницы и другие веб-файлы обслуживаются веб-серверами – специальным программным и аппаратным обеспечением, которое доставляет содержимое веб-сайтов клиентам по веб-протоколам. На сегодняшний день наиболее распространенными считаются данные сетевые протоколы передачи данных:

– HTTP (аббревиатура от англ. Hyper Text Transfer Protocol – протокол передачи гипертекста) – самый старый и наиболее известный протокол, используемый в большинстве стандартных веб-сайтов;

– HTTPS (аббр. от англ. Hyper Text Transfer Protocol Secure – безопасный протокол передачи гипертекста) – является расширением протокола HTTP, безопасность за счет поддержки шифрования по криптографическому протоколу SSL (аббр. от англ. Secure Socket Layer – уровень защищенных сокетов) или TLS (аббр. от англ. Transport Layer Security – безопасность транспортного уровня). Используется там, где требуется защита от перехвата данных, например, при аутентификации пользователя к своей учетной записи;

– SPDY (читается как speedy – быстрый) – относительно новый и безопасный, совместимый с HTTP протокол, разработанный корпорацией Google. Поддерживается большинством современных веб-браузеров. SPDY протокол позиционируется как замена некоторых частей протокола HTTP, отвечающих за управление соединениями и формата передачи данных. Основной задачей данного протокола является снижение времени загрузки веб-страниц и их элементов за счет объединения нескольких потоков данных в один, для уменьшения количества соединений для каждого клиента.

Сразу после установки Денвера предоставляется полностью функционирующий HTTP-сервер Apache. Основными его достоинствами являются надежность и гибкость конфигурации. На веб-сервере Apache может работать практически неограниченное количество сайтов.

Также Apache обладает рядом механизмов обеспечения безопасности, такие как:

– ограничение доступа к определенным директориям и файлам, или же всему серверу;

– запрет на получение доступа к обозначенным типам файлов для всех или определенной группы пользователей;

– ограничение доступа по IP-адресу пользователей;

– использование шифрования данных, передаваемых между пользователем и сервером с помощью механизма SSL. Это криптографический протокол, подразумевающий более безопасную связь. Использует среду протоколов с несколькими слоями, что обеспечивает безопасность обмена информации. Конфиденциальность общения работает за счет того, что безопасное соединение открыто только целевыми пользователями.

В базовый пакет Денвера входит интерпретатор PHP с поддержкой MySQL. PHP является одним из наиболее популярных языков программирования, используемый программистами и веб-разработчиками. Он также является одним из самых простых в освоении языков программирования. PHP имеет ряд особенностей, которые хорошо работают вместе, они включают сбор информации, динамический ввод и низкий уровень абстракции, что делает доступным данный язык.

PHP обрабатывается на стороне сервера и является HTML-встроенным скриптовым языком, это означает, что при выборе PHP в качестве языка реализации появится возможность создавать динамически генерируемые страницы быстро и легко.

Базовым компонентом Денвера также является СУБД MySQL – это самая популярная в мире база данных с открытым кодом, позволяющая экономично предоставлять надежные, высокопроизводительные и масштабируемые веб-приложения и встроенные приложения базы данных.

Сравнительный анализ нескольких БД проведенный журналом eWEEK показал, что MySQL и Oracle получили примерно равное количество голосов за лучшую производительность и высокий показатель масштабируемости [18].

СУБД MySQL отлично справляется с запросами любой сложности. Самая распространенная функция запросов – извлечение определенных данных из таблиц. Данные, которые необходимо просмотреть, как правило, находятся в нескольких таблицах, а запросы позволяют представить их в одной таблице.

Для удобства управления СУБД MySQL используется панель для администрирования phpMyAdmin – это веб-приложение с открытым кодом, написанное на языке PHP, позволяющие администрировать сервер MySQL через браузер. phpMyAdmin имеет простой и эргономичный интерфейс.

Наиболее распространенными системами хранения данных в СУБД MySQL являются MyISAM и InnoDB [16].

Таблицы систем InnoDB поддерживают транзакции – блок операторов запроса SQL, который в случае ошибки выполнения какого-либо запроса возвращает таблицу к предыдущему состоянию. В InnoDB имеется поддержка внешних ключей, что позволяет поддерживать целостность связи родительской и дочерней таблицы. Система InnoDB обладает высокой надежностью хранения данных и высокой скоростью восстановления после сбоя.

Таблицы MyISAM не поддерживают автоматический контроль ссылочной цельности, в отличие от InnoDB в которой имеется поддержка внешних ключей. Поэтому для поддержания ссылочной цельности в таблицах MyISAM программисту приходится самостоятельно разрабатывать алгоритмы для проверки целостности. При изменении строки блокируется вся таблица, в InnoDB блокируется только строка, которая подвергается

изменению. В MyISAM вставки в таблицу производятся быстрее, чем в InnoDB.

В таблицах InnoDB могут возникнуть блокировки, когда несколько процессов будут находиться в состоянии бесконечного ожидания ресурса, в MyISAM такого не происходит.

Недостатком MyISAM является то, что в случае сбоя таблица может отказаться работать до выполнения функции восстановления. Таблицы InnoDB обладают высокой надежностью, но медленнее выполняют операции чтения.

При создании новой таблицы в MySQL можно указать, какой тип таблицы использовать. По умолчанию в MySQL создаются таблицы типа MyISAM, для того, чтобы создать таблицу другого типа необходимо указывать дополнительные параметры [19].

Выбирать InnoDB необходимо когда: нужна высокая надежность хранения, требуются транзакции и таблица работает со смешанной нагрузкой. Систему хранения MyISAM лучше использовать когда: основной нагрузкой в таблице является операция чтения, данные таблицы не изменяются. Поэтому для хранилища данных используемых в вычислениях алгоритма подбора товаров лучше всего подойдет система хранения данных MyISAM.

Основное в разработке на языке PHP это то, что он прост в использовании. Кроме того, он доступен для большинства операционных систем и веб-серверов, а так же есть возможность получить доступ к наиболее распространенным базам данных, включая MySQL [20].

Другая немаловажная причина популярности MySQL заключается в том, что её создатели с самого начала разработки этой СУБД акцентировались на её быстродействие. Связка PHP с MySQL обеспечивает очень высокое быстродействие, которого очень трудно достичь другими средствами [21].

Поддержка MySQL входит в стандартную сборку PHP, поэтому проблем обращения к серверу MySQL из PHP-скриптов не возникает. Для обеспечения взаимодействия PHP с другими СУБД (PostgreSQL, Oracle и другие) приходится компилировать их самостоятельно из исходных кодов с дополнительными опциями [22].

Отличная производительность и высокая масштабируемость, тесная взаимосвязь с PHP, а так же то, что MySQL является продуктом свободного пользования, делает её одной из лучших СУБД на сегодняшний день [23].

Редактором для написания программного кода послужит Notepad++. Этот текстовый редактор относится к свободным программам, имеет удобную подсветку синтаксиса, которая определяет большинство самых популярных языков программирования и поддерживает более ста различных форматов файлов.

3.2 Реализация системы подбора товаров

Для реализации и отображения возможностей системы подбора товаров была разработана веб-страница, которая имеет боковую панель с настройками фильтрации товаров, а в центральной части происходит вывод полученных после обработки запроса товаров.

Процесс написание программного кода для отображения веб-страницы называется верстка. С помощью HTML, PHP, CSS и JavaScript кодов создается дизайн страниц сайта [17]. Каждый фрагмент кода отвечает за определенные элементы на странице, а так же их размещение.

Помимо PHP придавать интерактивность странице будет язык сценариев JavaScript. JavaScript используется в клиентской части веб-приложений клиент-серверных программ, в которых клиентом является браузер, а сервером — веб-сервер, имеющих распределённую между сервером и клиентом логику. Обмен информацией в веб-приложениях происходит по сети. Одним из преимуществ такого подхода является тот

факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются кроссплатформенными сервисами. Для дополнительно взаимодействия HTML с JavaScript будет использована библиотека JQuery. С помощью подключенной библиотекой JQuery реализован ползунок с выбором диапазона (рисунок 8).

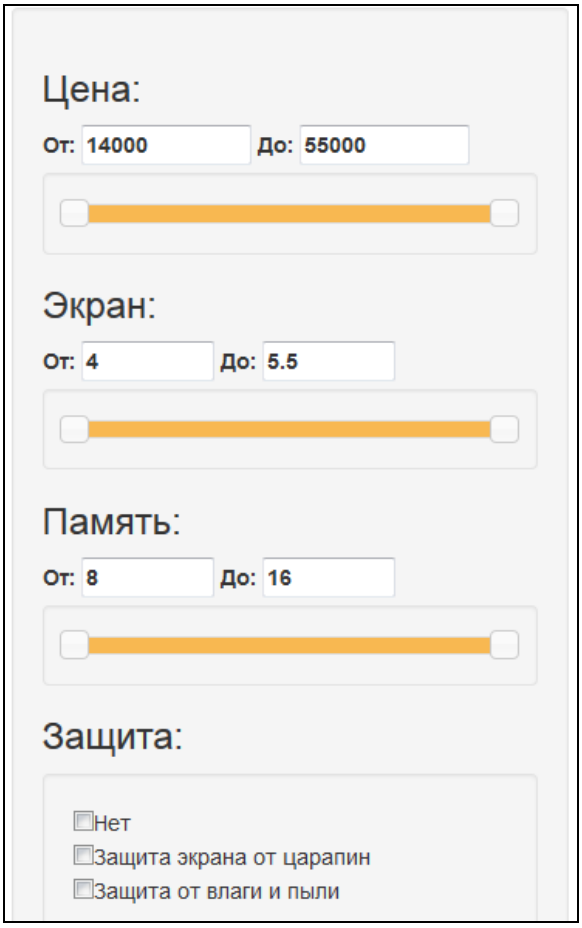

Рисунок 8 – Пример боковой панели с выбором параметров фильтрации

Перед тем как можно будет возможность выполнять сортировку товаров, необходимо создать таблицу хранилище, в которой все параметры, участвующие в поиске будут храниться в нормализованном виде

При создании таблицы, в которой будут находиться весовые коэффициенты необходимо заполнить имя таблицы и два столбца (рисунок 9). Первый столбец это имя поля таблицы, которое будет использоваться для поиска товаров, второй столбец это тип таблицы.

Всего предусмотрено три типа таблицы:

– тип 1 – это тип, который хранит информацию о первичном ключе;

– тип 2 – тип, использующийся для хранения значений категориальных параметров;

– тип 3 – тип для хранения значений числовых параметров.

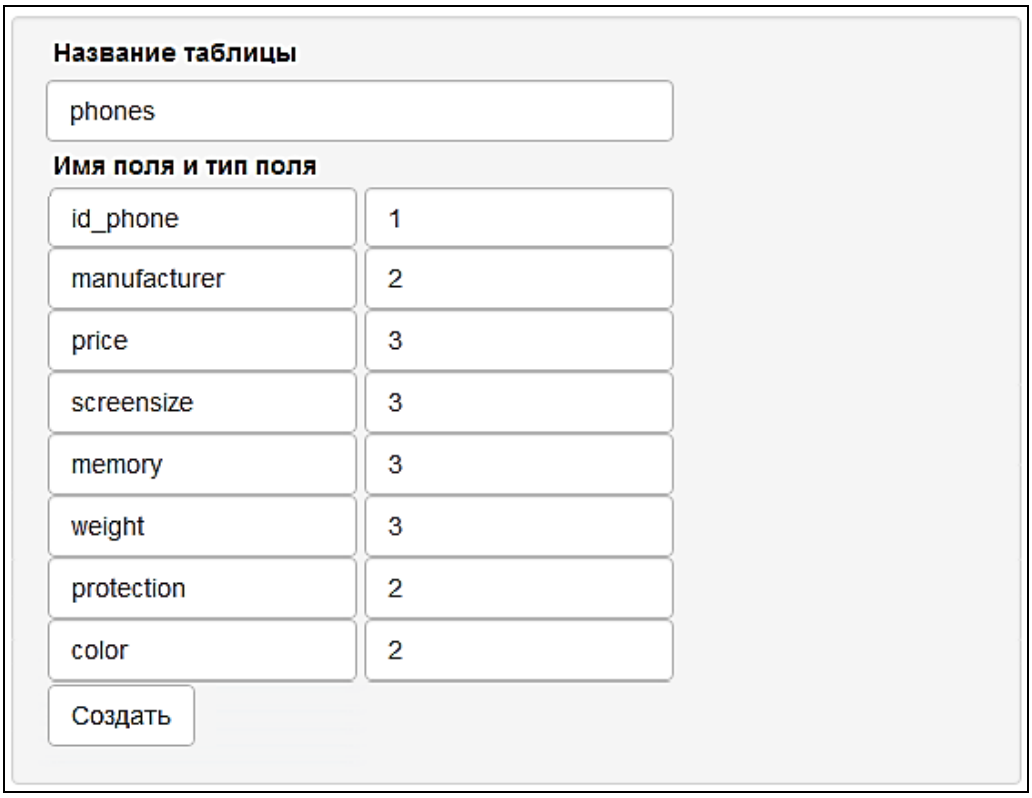

Рисунок 9 – Меню заполнения данных о нормализуемой таблице

После заполнения формы для создания хранилища весовых коэффициентов, созданная таблица появляется в списке БД (рисунок 10).

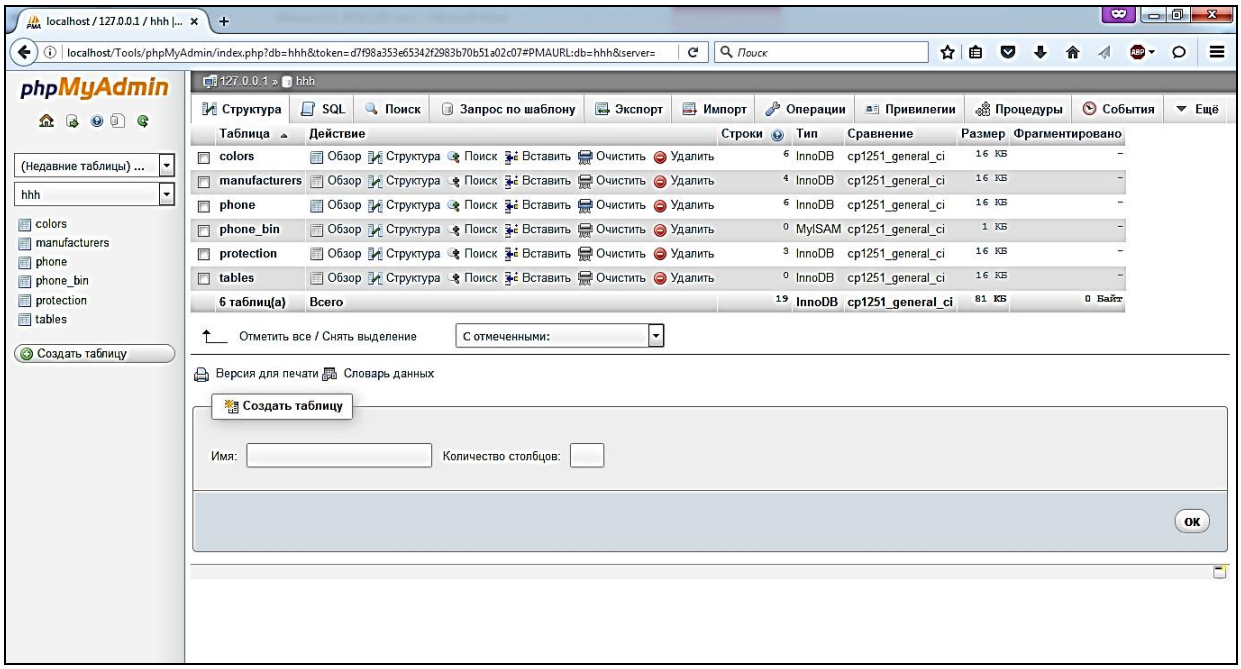

Рисунок 10 – Список таблиц в базе данных

На рисунке 11 изображена структура исходной таблицы БД на основе, которой формируется специальное хранилище весовых коэффициентов для вычислений нейронной сети.

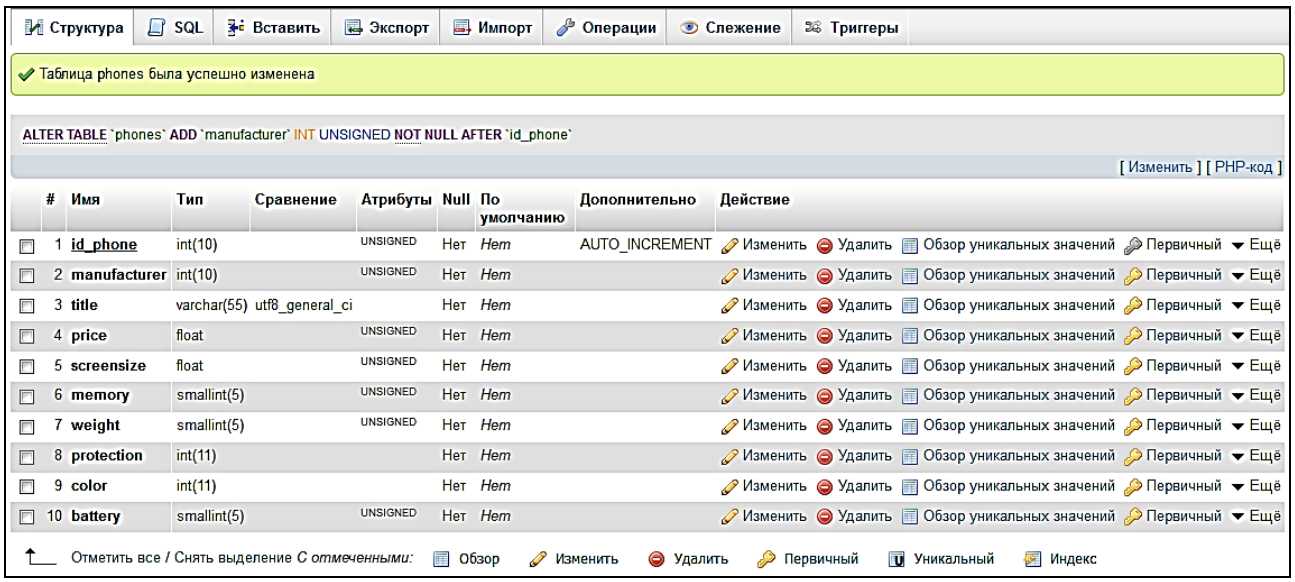

Рисунок 11 – Структура исходной таблицы товаров

В ходе выполнения функции, которая формирует хранилище весовых коэффициентов для дальнейшей нормализации, получается структура данных, отображенная на рисунке 12.

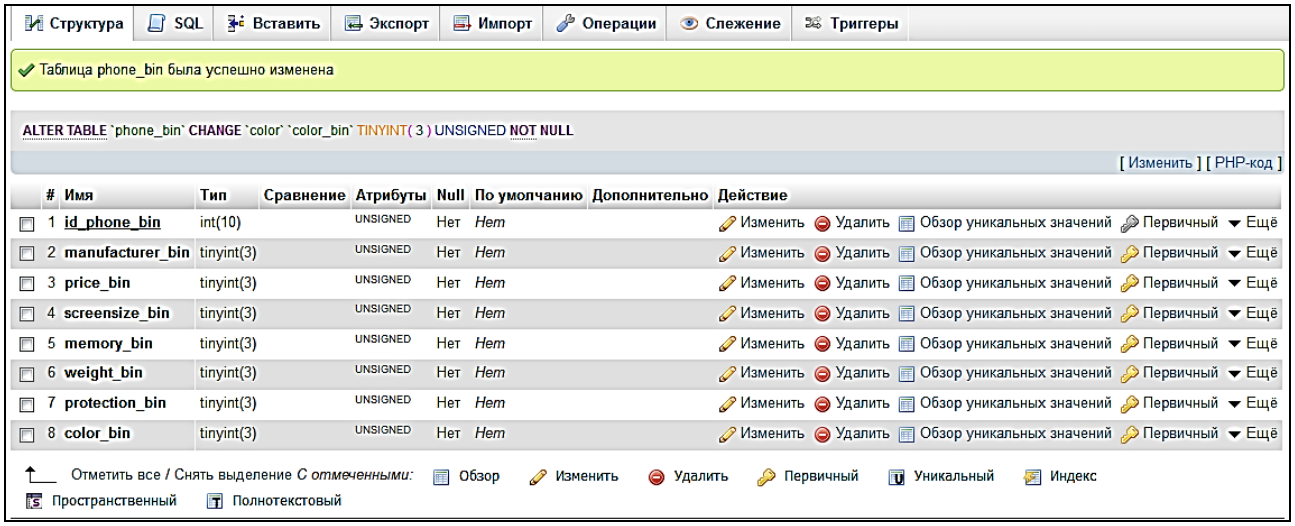

Рисунок 12 – Структура сформированной таблицы для весовых коэффициентов

В ходе реализации были проведены тесты различных типов таблиц в MySQL, которые показали абсолютное превосходство MyISAM перед InnoDB в скорости операций чтения и записи. Из-за систем поддерживающих высокую надежность InnoDB вставка в таблицы данного типа медленнее в более чем в 300 раз. Так же InnoDB использует большее количество памяти для хранения данных, таблицы MyISAM более чем в 4 раза эффективней в этом плане (рисунок 13).

	Таблица	Строки	Тип	Сравнение	Размер
FLOAT InnoDB	aaa	10,000	InnoDB	cp1251_general_ci	368 КБ
8xBIT(1) InnoDB	bbb	10,000	InnoDB	cp1251_general_ci	368 КБ
TINYINT InnoDB	bca	10,000	InnoDB	cp1251_general_ci	304 КБ
BIT(8) InnoDB	bin	10,001	InnoDB	cp1251_general_ci	304 КБ
TINYINT MyISAM	binmyisam	10,000	MyISAM	cp1251_general_ci	69.4 КБ
BIT(8) MyISAM	bitmyisam	10,000	MyISAM	cp1251_general_ci	69.4 КБ

Рисунок 13 – Различные типы данных и таблиц

На рисунке 14 отображен пример работы алгоритма системы подбора товаров, который использует нормализованные данные информации о квартирах.

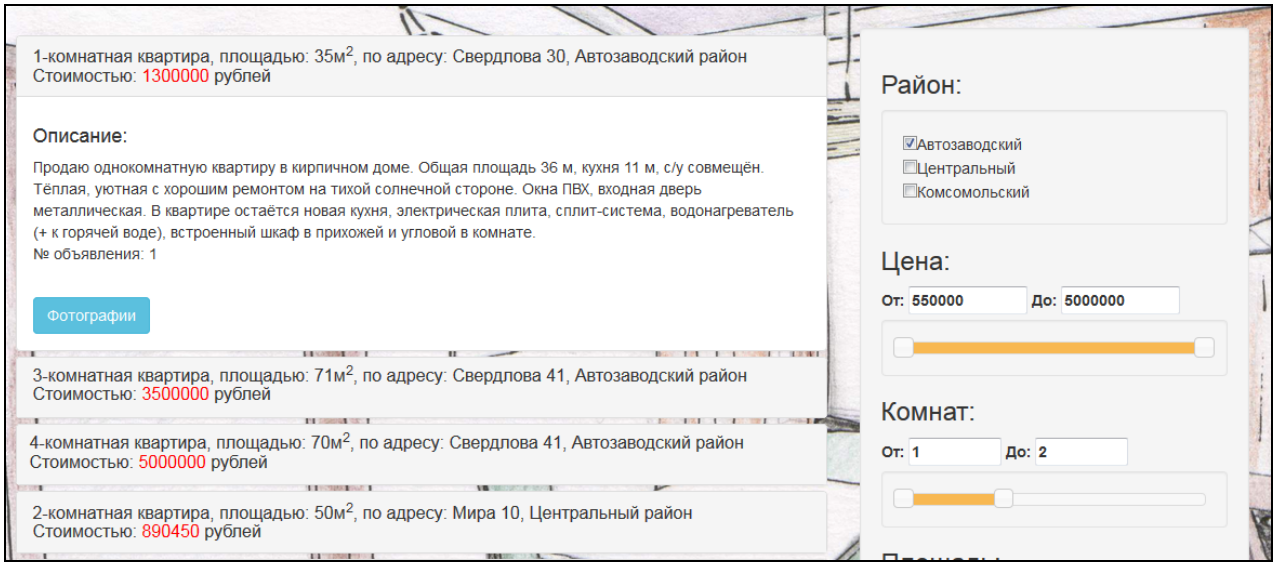

Рисунок 14 – Пример вывода товаров, отсортированных в порядке схожести с заданными характеристиками

Все параметры, имеющиеся для поиска в боковой панели фильтра, отправляются в функцию системы подбора товаров, которая проверяет какие именно параметры, были заданы пользователем. Если это диапазон, то в функцию надо передавать среднее значение выбранного диапазона, если это категориальный признак, то следует передавать массив с индексами выбранных параметров.

Система подбора товаров на основе запроса пользователя в реальном времени формирует ИНС Хэмминга, где количество нейронов входного слоя равно количеству заданных для поиска параметров, после чего с помощью формулы (9) и формулы (10) высчитываются значения состояний нейронов первого слоя, которые сразу же начинают исполнять алгоритм реализующий формулу (6). Индексы нейронов состояния, которых доходит до нуля записываются в массив. По окончанию вычислений второго слоя, массив в котором товары отсортированы от наименее подходящего до самого близкого к запросу пользователя, передается на страницу, где в зависимости от верстки сайта отображается вся необходимая информация о товаре.

В системе подбора товаров реализована функция, с помощью которой в интернет-магазин можно добавить кнопки рядом с описанием товара, по

нажатию которой будет произведен поиск товаров похожих. Программный код основных функций алгоритма написан в приложении.

При дальнейшем развитии данной работы возможно добавление функционала для более точного и быстрого поиска, улучшения интерфейса и удобства пользования. Например, добавления возможности регулировки приоритетов определенных параметров поиска – это непременно повысит точность нахождения именно тех товаров, которые необходимы пользователю, также можно добавить возможность применения жесткой фильтрации, если пользователю нужно будет строго определенное значение характеристики, например, необходим только один определенный производитель товара. Так же планируется указывать в процентах сходство найденного товара с прототипом, а так же помечать отличия от заданных характеристик.

Заключение

Целью данной выпускной квалификационной работы целью является повышение удобства поиска товаров по запросам пользователей.

Все поставленные задачи были выполнены, а именно:

– проведен анализ недостатков существующих методов поиска товара, в результате которого было установлено, что дальнейшее совершенствование алгоритмов поиска информации возможно за счет синтеза в них нейронных сетей искусственного интеллекта;

– спроектированы и разработаны подходы обеспечивающие поиск объектов с помощью нейронной сети Хэмминга. В том числе разработаны алгоритмы кодирования информации об объектах в биполярные матрицы;

– при программной реализации была смоделирована нейронная сеть с заданными параметрами и проверены все предложенные решения на примере интернет-магазина.

В ходе выполнения выпускной квалификационной работы были созданы функции, с помощью которых можно интегрировать систему подбора товаров в интернет-магазин использующий СУБД MySQL. Алгоритмы системы подбора товаров работающей с использованием ИНС Хэмминга были разработаны на скриптовом языке PHP. Для уменьшения объёма выборки из БД и повышения скорости выполнения запроса все данные необходимые для работы нейронной сети были помещены в специальную таблицу – хранилище данных информации о товарах. Для проведения тестирования разработанных алгоритмов была создана веб-страница с помощью, которой можно осуществлять поиск товаров по схожести.

Список используемой литературы

1. Appleseed J. Category-Specific Sorting: A New Way to Sort Products [Electronic resource]: [UX Article] / J. Appleseed. – Electronic data. – Copenhagen: Baymard Institute, [2015]. – Mode of access: http://baymard.com/blog/category-specific-sorting

2. ГОСТ Р ИСО 9241-210–2012. Эргономика взаимодействия человек-система. Часть 210. Человеко-ориентированное проектирование интерактивных систем. – Введен впервые; введ. 2013-12-01. – М.: Стандартинформ, 2013. – 36 с.

3. Holst C. External Article: The Current State of E-Commerce Search [Electronic resource]: [Интернет журнал для веб-дизайнеров и разработчиков Smashing Magazine] / C. Holst. – Electronic data. – Freiburg, [2015]. – Mode of access: https://www.smashingmagazine.com/2015/04/the-current-state-of-e-commerce-filtering/

4. V.B. Kovacević, A.M. Gavrovska, M.P. Paskaš, «High-speed implementation of Hamming neural network», Neural Network Applications in Electrical Engineering (NEUREL), 2010 10th Symposium on, Sept. 2010, pp.167-170.

5. Искусственная нейронная сеть [Электронный ресурс]: Материал из Википедии — свободной энциклопедии : Версия 78351080, сохранённая в 14:01 UTC 14 мая 2016 / Авторы Википедии // Википедия, свободная энциклопедия. — Электрон. дан. — Сан-Франциско: Фонд Викимедиа, 2016. — Режим доступа: http://ru.wikipedia.org/?oldid=78351080

6. Galushkin, A. I. Neural Networks Theory / Alexander I. Galushkin. – Springer Berlin Heidelberg, 2007. – 396 p.

7. Бураков, М. В. Нейронные сети и нейроконтроллеры: учебное пособие / М. В. Бураков. – СПб.: Изд-во «ГУАП», 2013. – 284 с.

8. Нейросетевые технологии обработки информации: учебное пособие / Э.Д. Аведьян, А.И. Галушкин, Н.И. Червяков, П.А. Сахнюк. – Ставрополь: Изд-во «Агрус», 2013. – 292 с.

9. Richard P. Lippmann – Information [Electronic resource]: [Информация о докторе Ричарде Липпмане] . – Electronic data. – Cambridge: Massachusetts Institute of Technology. – Mode of access: https://www.ll.mit.edu/mission/cybersec/cybersec-bios/lippmann-bio.html

10. Хайкин С. Нейронные сети. Полный курс. 2-е издание / С. Хайкин; пер. с анг. Н.Н. Куссуль и А.Ю. Шелестова, под ред. Н.Н. Куссуль. – 2-е изд. – М.: «Вильямс», 2006. – 1104 с.

11. Koprinkova-Hristova, P. Artificial Neural Networks / Petia Koprinkova-Hristova, Valeri Mladenov, Nikola K. Kasabov. – Springer International Publishing, 2015. – 488 p.

12. Li, S.Z. Hamming Distance / S.Z. Li, A. Jain. – Encyclopedia of Biometrics, 2009. – 668 p.

13. Rigatos, G. G. Advanced Models of Neural Networks / Gerasimos G. Rigatos. – Springer Berlin Heidelberg, 2015. – 396 p.

14. F. Santoyo, E. Chávez, E.S. Téllez, «A Compressed Index for Hamming Distances», Similarity Search and Applications, Oct. 2014, pp. 113-126.

15. Ульман, Л. PHP и MySQL. Создание интернет-магазинов, 2-е изд. / Л. Ульман; пер. с анг. А. Сергеев. – 2-е изд. – М.: «Вильямс», 2015. – 544 с.

16. Бейли, Л. Изучаем SQL / Л. Бейли. – СПб.: Изд-во «Питер», 2012. – 573 с.

17. Прохоренок, Н.А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера / Н.А. Прохоренок, В.А. Дронов. – 4-е изд. – СПб.: Изд-во «БХВ-Петербург», 2015. – 766 с.

18. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript / Р. Никсон; пер. с анг. Н. Вильчинский. – 2-е изд. – СПб.: Изд-во «Питер», 2013. – 560 с.

19. Справочное руководство по MySQL [Электронный ресурс]: MySQL Manual. – Электрон. дан. – Режим доступа: http://www.mysql.ru/docs/man/

20. Руководство по PHP [Электронный ресурс]: PHP: Руководство по PHP – Manual / группа документирования PHP. – Электрон. дан. – [2016]. – Режима доступа: http://php.net/manual/ru/index.php

21. Колисниченко, Д.Н. PHP и MySQL. Разработка веб-приложений / Д.Н. Колисниченко. – СПб.: Изд-во «БХВ-Петербург», 2015. – 592 с.

22. Kroenke, David M. Database Processing: Fundamentals, Design, and Implementation / David M. Kroenke, David J. Auer. – 14th edition. – Pearson Education Limited, 2015. – 640 p.

23. Kromann, Frank M. PHP and MySQL Recipes: A Problem-Solution Approach / Frank M. Kromann. – 2nd edition. – Appress, 2015. – 700 p.

Приложение А

Программный код алгоритмов системы подбора товаров

```
<?php
// функция для создания таблицы эталонных образов в БД
function createTable($conn, $tableName, $p, $t)
{
	// p* - имя поля таблицы
	// t* - тип поля таблицы
	// используемые типы:
	// тип 1 - id primary key
	// тип 2 - категория / id foreign key
	// тип 3 - числовой
	$parameters = '';
	for($i = 0; $i < count($p); $i++)
	{
		if($t[$i] == 1)
		{	$parameters = $parameters.$p[$i].'_bin INT UNSIGNED, ';		}
		if($t[$i] == 2)
		{	$parameters = $parameters.$p[$i].'_bin TINYINT UNSIGNED, '; }
		if($t[$i] == 3)
		{	$parameters = $parameters.$p[$i].'_bin TINYINT UNSIGNED, '; }
	}
	$parameters = $parameters.'PRIMARY KEY('.$p[0].'_bin)';
	echo $parameters;
	echo "<br>";
	$sql = 'CREATE TABLE '.$tableName."_weight.' ('.$parameters.') TYPE = MYISAM';
	echo $sql;
	$conn->query($sql);
}
?>
<?php
// функция для заполнения таблицы образов
function completeTheTable($conn, $tableName)
{
	n = 255;

	$t = getTypes($tableName);
	$p = getParameters($tableName);

	$sql = 'SELECT * FROM '.$tableName;
	$result = $conn->query($sql);
	$row = $result->fetch_assoc();

	for ($i = 0; $i < count($p); $i++)
	{ $val[$i] = ""; }
```

```
	$col = "";
	for($i = 0; $i < count($p); $i++)
	{
		if($i == count($p) -1)
			{ $col = $col.$p[$i]; }
		else
			{ $col = $col.$p[$i]', '; }
	}
	while ($row = $result->fetch_assoc())
	{
		// считаем поля для определенного объекта
		for($i = 1; $i < count($p); $i++) // 0 - это id
		{
			if($t[$i] == 2)
			{
				$val[$i] = $row[$p[$i]];
			}
			if($t[$i] == 3)
			{
				$max = getMax($conn, $p[$i], $tableName);
				$min = getMin($conn, $p[$i], $tableName);
				{
					$val[$i] = ( ( $row[$p[$i]] - $min)*n ) / ($min - $max);
				}
			}
		}
		$values = ""; // обнуляем строку параметров перед каждой записью
		for($i = 1; $i < count($p); $i++)
		{
			if($i == count($p) -1)
				{ $values = $values.$val[$i]'_bin'; }
			else
				{ $values = $values.$val[$i]'_bin, '; }
		}
		// добавляем в таблицу объект
		$sql = 'INSERT INTO '.$tableName.'_bin ('.$col.') VALUES ('.$row[$p[0]].',
'.$values.')';
		$conn->query($sql);
	}
}
function normalize($conn, $tableName, $select)
{
	n = 255;

	$t = getTypes($tableName);
	$p = getParameters($tableName);

	for($i = 1; $i < count($p); $i++) // 0 - это id
	{
		if($t[$i] == 2)
```

```
                {
                        $val[$i] = $row[$p[$i]];
                }
                if($t[$i] == 3)
                {
                        $max = getMax($conn, $p[$i], $tableName);
                        $min = getMin($conn, $p[$i], $tableName);
                        {
                                $val[$i] = ( ( $row[$p[$i]] - $min)*n ) / ($min - $max);
                        }
                }
                $obt[$i] = val[$i];
        }
        return $obt;
}
?>
<?php
// функция поиска товаров
function search($conn, $tableName, $selection)
{
        $n = 256; // общее число

        $epsilon = getEpsilon($tableName)
        $t = getTypes($tableName);
        $p = getParameters($tableName);

        $search = normalize($conn, $tableName, $selection);

        // получаем необходимую таблицу весовых коэффициентов
        $sql = 'SELECT * FROM '.$tableName.'_bin';
        $result = $conn->query($sql);

        // считаем количество объектов
        $sql = 'SELECT COUNT('.$p[0].') FROM '.$tableName.'_bin';
        $resultcount = $conn->query($sql);
        $row = $resultcount->fetch_assoc();
        $m = $row['COUNT('.$p[0].')'];

        $sum; // для подсчета весовых коэффициентов
        $id;
        // первый слой сети
        while ($row = $result->fetch_assoc())
        {
                id[$i] = $row[$p[0].'_bin']; // запоминаем id
                for($i = 1; $i < count($p); $i++) // 0 - это id
                {
                        if($t[$i] == 2)
                        {
                                $l = howmany($p);
                                if(in_array($row[$p[$i].'_bin'], $search, true) )
                                        {$sum[$i] = $sum[$i] + n-2*($l-1);}
```

```
                        else
                                {$sum[$i] = $sum[$i] + n-2*($l+1);}
                }
                if($t[$i] == 3)
                {
                        $sum[$i] = $sum[$i] + n - 2*abs($row[$p[$i].'_bin'.] - $search[$i]);
                }
        }
}
$stack;
for($i = 0; $i < $m; $i++) // флаги для выхода из слоя
{
        $f[$i] = true;
}
for($k = 0; $k < $m; $k++) // сумма выходных состояний второго слоя
{
        $y = $y + $sum[$k];
}
for($j = $m; $j > 1; ) // до тех пор пока не останется победитель
{
        for($i = 0; $i < $m; $i++)
        {
                if($f[$i] == false)
                { continue; }
                if($sum[$i] <= 0)
                {
                        array_push($stack, id[$i]);
                        $j--;
                        $f[$i] = false;
                        $sum[$i] = 0;
                        continue;
                }
                else
                {
                        $sum[$i] = $sum[$i] - $epsilon*($y-$sum[$i]);
                }
        }
        $y = 0;
        for($k = 0; $k < $m; $k++) // сумма выходных состояний второго слоя
        {
                $y = $y + $sum[$k];
        }
}
for($k = 0; $k < $m; $k++) // ищем id победителя
{
        if($sum[$k] > 0);
                array_push($stack, id[$k]);
}
array_reverse($id); // делаем обратный порядок, чтобы первый элемент был самый
подходящий
return $id;
```

```
}
?>
```