

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Государственное образовательное учреждение  
высшего профессионального образования  
«Сибирский государственный аэрокосмический университет  
имени академика М.Ф. Решетнева»

Институт информатики и телекоммуникации

Кафедра информатики и вычислительной техники

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
К КУРСОВОМУ ПРОЕКТУ ПО ДИСЦИПЛИНЕ  
«АДМИНИСТРИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ»**

Тема: Разграничение прав доступа для пакета нейросетевого моделирования

Выполнил: ст. гр. БИС-81  
Истомина Ю.А.

Проверил: к.т.н., доцент каф. ИВТ  
Моргунова О.Н.

Красноярск 2011г.

<b>I. ДОКУМЕНТ-КОНЦЕПЦИЯ.....</b>	<b>3</b>
1. ВВЕДЕНИЕ.....	3
1.1 Цель документа-концепции .....	3
1.2 Назначение и общая характеристика продукта .....	3
1.3 Ссылки и использованная литература .....	3
2. ОПИСАНИЕ ПОЛЬЗОВАТЕЛЕЙ.....	3
2.1 Виды пользователей и их краткие описания .....	3
2.2 Среда пользователя .....	4
2.3 Основные потребности пользователя .....	4
3. Состояние рынка и конкурирующие продукты.....	4
4. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ (ФУНКЦИИ ПРОДУКТА) .....	12
4.1 Обязательные функции для первой версии .....	12
4.2 Дополнительные функции для первой версии .....	13
4.3 Будущие функции .....	13
5. ОСНОВНЫЕ СПОСОБЫ ИСПОЛЬЗОВАНИЯ ПРОГРАММНОГО ПРОДУКТА И СЦЕНАРИИ РАБОТЫ С НИМ .....	13
5.1 Простой способ использования программного продукта .....	13
5.2 Полномасштабный способ использования программного продукта .....	13
6. НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ.....	13
6.1 Требования к программному продукту.....	13
6.2 Организационные требования.....	14
6.3 Внешние требования .....	14
6.4 Прочие требования .....	14
7. ТРЕБОВАНИЯ К ДОКУМЕНТАЦИИ .....	14
7.1 Руководство пользователя.....	14
7.2 Руководство программиста .....	14
7.3 Интерактивная подсказка.....	14
7.4 Руководства по установке и конфигурированию и файл ReadMe .....	14
7.5 Маркировка и упаковка .....	14
8. ГЛОССАРИЙ И СПИСОК СОКРАЩЕНИЙ .....	15
8.1 Глоссарий.....	15
8.2 Список сокращений .....	15
<b>II. АРХИТЕКТУРА И СИСТЕМНЫЕ ТРЕБОВАНИЯ.....</b>	<b>15</b>
1. АРХИТЕКТУРА ПРОГРАММНОГО ПРОДУКТА И МОДУЛЬНЫЙ СОСТАВ ПОДСИСТЕМ .....	15
<b>III. ТЕХНИЧЕСКИЙ ПРОЕКТ .....</b>	<b>17</b>
<b>1.    ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ .....</b>	<b>17</b>
1. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ .....	17
2. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ .....	20
3. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ.....	23
<b>2.    РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА .....</b>	<b>27</b>
1. СТРУКТУРА ПРОГРАММНОГО ПРОДУКТА .....	27
2. РЕАЛИЗАЦИЯ БИЗНЕС-ПРАВИЛ .....	27
3. РУКОВОДСТВО ПРОГРАММИСТА.....	29
4. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ .....	31

## I. ДОКУМЕНТ-КОНЦЕПЦИЯ

### 1. Введение

#### 1.1 Цель документа-концепции

Цель данного документа состоит в сборе и анализе исходной информации для разработки, определении высокоуровневых потребностей пользователей и формулировании функций продукта.

#### 1.2 Назначение и общая характеристика продукта

Программный продукт представляет базу данных для хранения информации о создаваемых нейронных сетях, обучающих выборок и систему разграничения прав доступа средствами СУБД. Программный продукт предназначен для использования в программе, реализующей алгоритм обратного распространения ошибки многослойного персептрона.

#### 1.3 Ссылки и использованная литература

1. Уоссермен, Ф. Нейрокомпьютерная техника : Пер. с англ. М. : Мир, 1992. – 184 с.
2. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы [Текст] : Пер. с польск. / Д. Рутковская, М. Пилиньский, Л. Рутковский. М. : Горячая линия – Телеком, 2004. – 452 с.
3. Дж. Уросли, PostgreSQL для профессионалов [Текст]- М. – Питер 2003 г. – 498 с.
4. Описание библиотеки классов FCL [Электронный ресурс] Режим доступа: [mds.microsoft.com](http://mds.microsoft.com).

## 2. Описание пользователей

Программу предполагается использовать для научных целей, либо для решения прикладных задач.

### 2.1 Виды пользователей и их краткие описания

**Пользователь** проходит регистрацию в программе. Доступ к таблицам data, network только к своим записям. Создает новые записи, редактирует старые, удаляет. Имеет возможность заполнения базы данных из файла.

**Администратор** программы просматривает все события в программе. Доступ к таблицам data, network logging. Может удалять сети и данные.

**Администратор** сервера баз данных – создает новые таблицы, новые группы пользователей, новых пользователей.

**Продвинутый пользователь** – имеет доступ к сетям и выборкам, созданными другими пользователями. Доступ к таблицам data, network к чужим записям только на просмотр.

Для удобства были созданы следующие групповые роли Рис. 1.

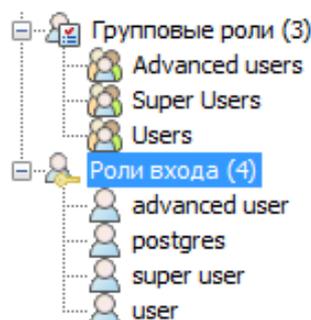


Рис. 1. Групповые роли пользователей.

## 2.2 Среда пользователя

Пользователь работает в ОС Windows 7, используя СУБД Postgresql версии 9.0.4. Приложение разработано с помощью Microsoft Visual Studio 2010.

## 2.3 Основные потребности пользователя

Пользователю необходима система аутентификации и хранилище данных.

## 3. Состояние рынка и конкурирующие продукты

Стандартный нейропакет включает в себя ряд следующих основных компонентов. Система проектирования позволяет создавать нейронные сети различных архитектур. Система экспорта и импорта данных – загружать/сохранять данные в разных форматах, а также проводить предварительную обработку данных. Система обучения позволяет подобрать алгоритм настройки параметров нейронной сети, соответствующий выбранной архитектуре. Система графического представления предоставляет пользователю возможность просматривать данные и результаты работы сети.

Для проведения исследований были выбраны следующие программы:

1. MatLab
2. Neuro Office
3. NeuroShell

Представляется, что при оценке конкретного программного средства автоматизации нейросетевого моделирования в первую очередь необходимо руководствоваться следующими критериями:

- возможность его использования совместно с другими программными приложениями;
- достаточное количество предлагаемых алгоритмов, архитектур, активационных функций;
- наличие средств визуального представления данных;
- разделение пользователей по группам в соответствии с уровнем знаний в области нейросетевого моделирования;
- понятность интерфейса для начинающего пользователя;
- простота подготовки исходных данных, наличие средств для предварительной обработки исходных данных;
- наличие справочного сопровождения, описывающего последовательность действий пользователя, необходимых для получения результата;
- присутствие средств анализа созданных нейронных сетей, например, оценка времени работы сети, анализ вычислительной эффективности сети;
- возможность создавать сети с собственной архитектурой, создавать новые алгоритмы обучения;
- простота и наглядность процесса формирования нейронной сети и настройки разнообразных параметров.

### *MatLab*

Matlab Neural Network Tools – пакет для проектирования нейронных сетей. Данная программа имеет простой интерфейс Рис. 2, легко задаются входные и выходные данные для обучения сети. Язык интерфейса английский.

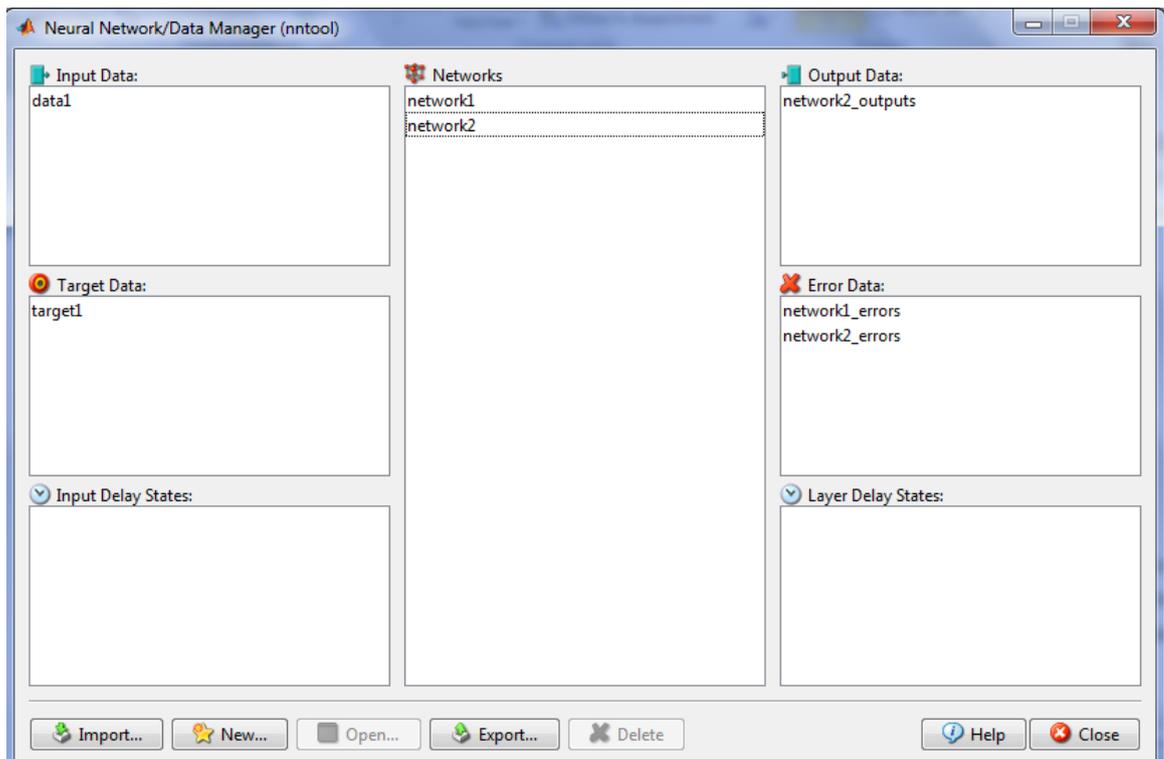


Рис. 2. Интерфейс приложения Neural Network Tools

Данные можно задавать, используя язык матлаба, что удобно для исследований в математике. Так же данные можно импортировать и экспортировать, причем они должны быть сохранены с расширением mat. Программа имеет внушительную базу разновидностей сетей, активационных функций и дополнительных настроек. Пакет предлагает 19 типов нейронных сетей с различными архитектурами. Для каждого типа нейронной сети предусмотрены индивидуальные настройки. Программный продукт предлагает 16 видов тренировочных функций.

После создания сети пользователь имеет возможность посмотреть её архитектуру Рис. 3. А также существует возможность посмотреть изменение ошибки в течении выполнения алгоритма обучения.

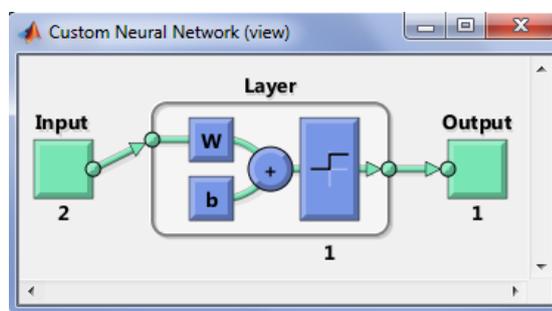


Рис. 3. Структура нейронной сети

Оценка программного продукта представлена в Табл. 1.

Табл. 1. Оценка программного пакета

Достоинства	Недостатки
Простой интерфейс легко работать с простыми примерами	Трудная синхронизация с файлами других форматов
Большая библиотека алгоритмов и параметров	Для сетей более сложной структуры нет соответствующей документации. Процесс настройки и использования сети усложняется.
Существуют средства визуализации результатов работы с программой.	Только англоязычный интерфейс.
От пользователя не требуется глубоких знаний языка матлаба	Нет подробного описания к средствам визуализации.

### *Neuro Office*

Neuro Office - пакет для построения и исследования работы нейронных сетей. Язык интерфейса английский/русский. Пакет основан на двух исполняемых файлах:

- NView.exe - редактор структурных моделей и топологий;
- NEmul.exe - эмулятор ядерных нейронных сетей.

Программа сопровождается хорошей справочной системой.

NView.exe позволяет создавать нейронную сеть произвольной топологии или сеть на основе существующих топологий. Пользователь настраивает количество нейронов в каждом слое и указывает связи между нейронами слоев. В программе представлено 8 различных архитектур. После создания сети или открытия существующей, пользователь может посмотреть оценку эффективности сети. Пример представлен на Рис. 4. Вычислительная эффективность вычисляется по следующему алгоритму: количество операций умножения полносвязной сети делится на количество операций данной сети. После создания сети пользователь может менять ее параметры, а также оптимизировать архитектуру нейронной сети.

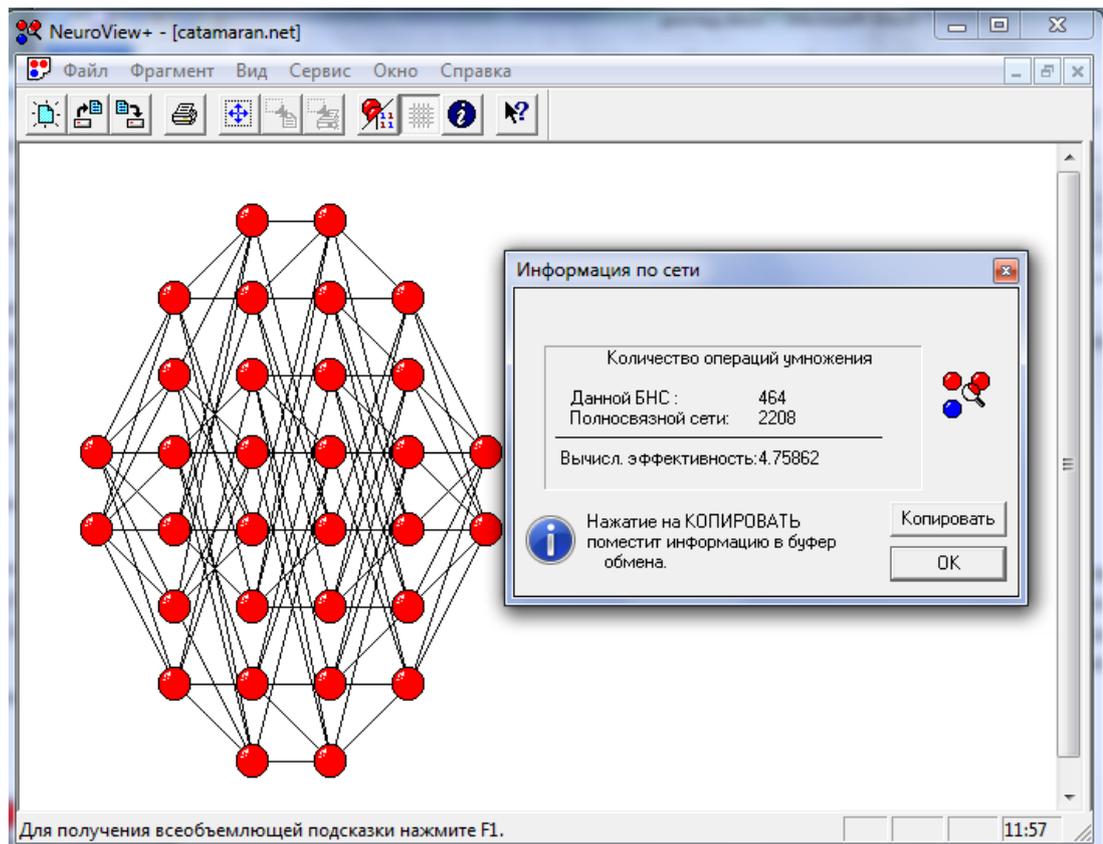


Рис. 4. Вычислительная эффективность сети

NEmul.exe позволяет проводить эксперименты с сетью. В эмуляторе сети возможно изменять матрицу весов каждого нейрона, также существует возможность заполнять веса случайными цифрами в выбранном диапазоне и выбирать активационную функцию. Используются следующие активационные функции:

- Сигмоидальная;
- Гиперболический тангенс;
- Синусоидальная;
- Линейная;
- Логическая -1,+1;
- Логическая 0,+1.

Интерфейс предусматривает все возможные настройки алгоритма обучения, однако используется только один алгоритм «Алгоритм обратного распространения ошибки». Одним из достоинств программы является возможность экспортировать и импортировать данные в excel, в базу данных и в обычный файл. В программе существует возможность измерить время работы сети. Последняя версия программы была выпущена в 1999 году.

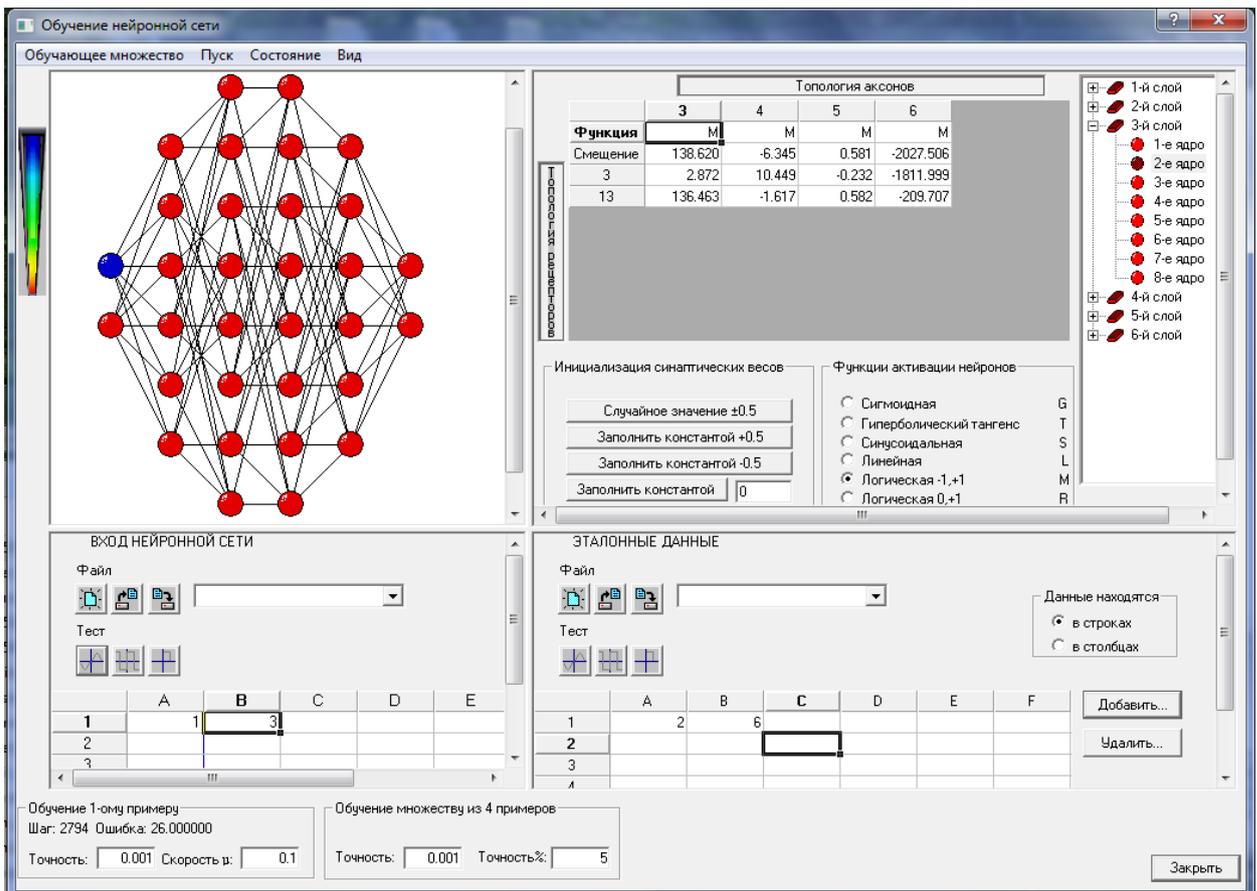


Рис. 5. Интерфейс приложения Neuro Office

Оценка программного продукта представлена в Табл..

Табл. 2. Оценка программного пакета

Достоинства	Недостатки
Возможность сохранять и загружать файлы в формате excel и возможность работы с БД	Использован один алгоритм обучения.
Программа подходит для начинающего пользователя, существующая база созданных нейронных сетей позволяет легко понять принципы работы программы	Последняя версия программы была выпущена довольно давно.
Хорошая справочная система	Трудно использовать данную программу совместно с другой.
	Время обучения сети 1 примеру с точностью 0,001 занимает более часа. Параметры сети 6 слоев, до 8 нейронов в каждом слое.

### Neuro Shell

Neuro Shell – самый популярный пакет для использования нейронных сетей. Для использования данного программного обеспечения необходимо иметь задачу, при создании новой задачи пользователю предоставляется сделать выбор своего уровня:

- Нейронные сети для начинающего пользователя – тренировка и применение простой архитектуры.
- Нейронные сети для профессионалов – тренировка и применение сложных архитектур.
- Средства автономного использования - создание автономных версий обученных сетей.

Система для начинающего содержит упрощённый набор функций. Используется только алгоритм обратного распространения ошибки, все параметры настраиваются по умолчанию.

В системе для профессионала существует большой набор различных настроек: от установки правил преобразования файла в исходные данные до проектирования сети.

При проектировании пользователь имеет возможность оперировать различными архитектурами и выбирать активационные функции. Интерфейс программы представлен на Рис. 6.

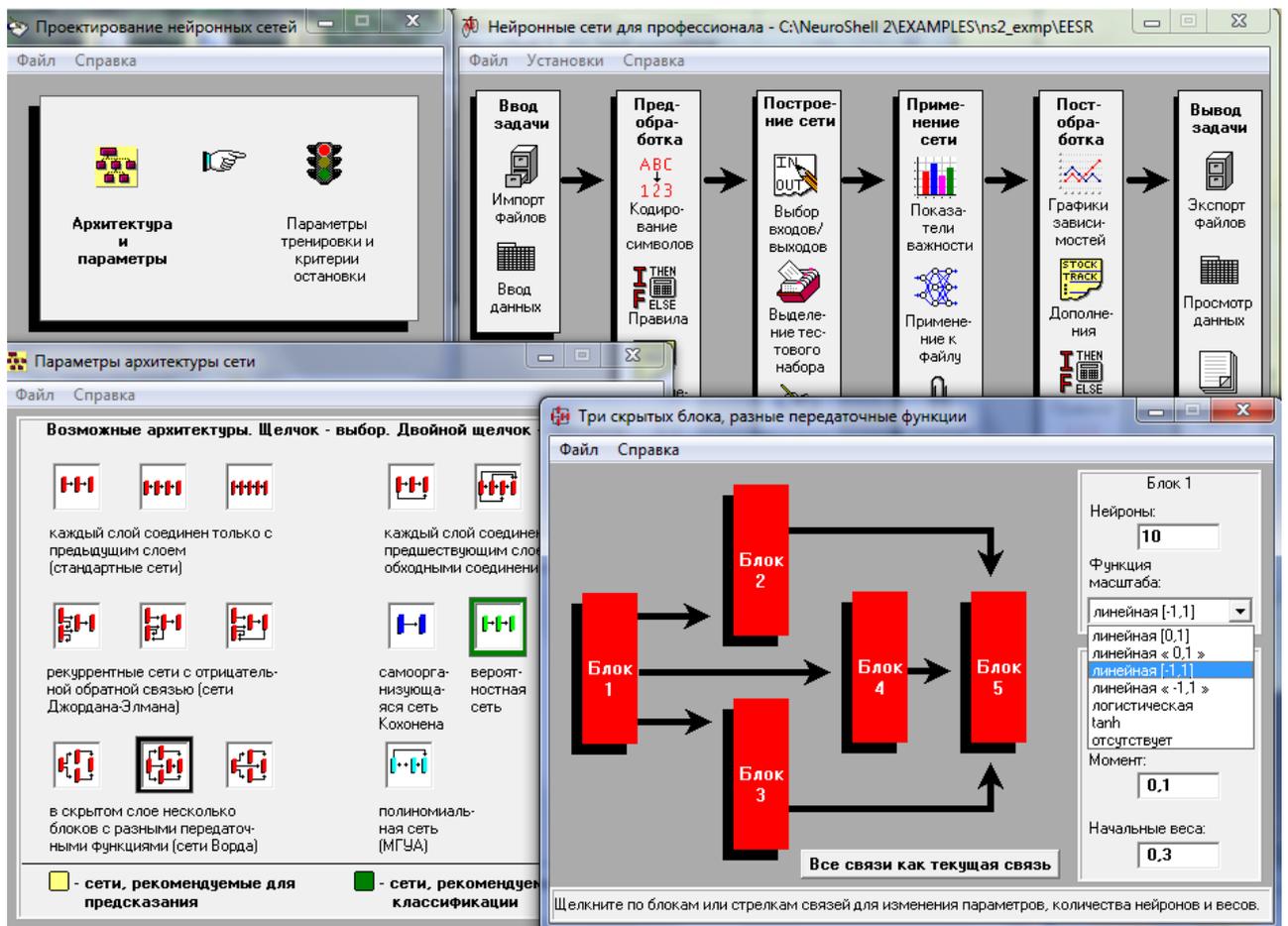


Рис. 6. Интерфейс программы Neuro Shell

Программа представляет возможность пользователю выбирать различные архитектуры нейронных сетей. Возможные архитектуры представлены в Табл. 3.

Табл. 3. Архитектуры НС

Название	Описание
Стандартная сеть	сеть с обратным распространением, в котором каждый слой связан только с непосредственно предшествующим слоем.
Сети с обходными соединениями.	каждый слой связан со всеми предшествующими слоями
Рекуррентные сети	применяются для финансовых биржевых предсказаний, поскольку эти сети могут запоминать последовательности, т. е. обладают долговременной памятью
Сети Ворда	способны выделять различные свойства в данных, благодаря наличию в скрытом слое нескольких блоков, каждый из которых имеет свою передаточную функцию.
Сети Кохонена	в процессе тренировки не требуется сообщать сети правильный ответ при предъявлении примера. Эти сети способны находить распределение данных на различные категории (классы).
Вероятностные нейронные сети	обучаются на ограниченных наборах данных, причем для обучения нейросети достаточно однократного предъявления тренировочного набора
Нейронные сети с общей регрессией	однократное предъявление тренировочных данных. Однако, в отличие от сетей ВНС, которые классифицируют данные, сети НСОР способны предсказывать выходы с непрерывной амплитудой.
Полиномиальные сети	содержит в связях полиномиальные выражения и использует в некотором смысле аналогичный генетическим алгоритмам механизм принятия решения о том, сколько слоев необходимо построить. Результатом тренировки является возможность представить выход как полиномиальную функцию всех или части входов.

Далее перечислим средства визуального представления данных:

1. График зависимости переменной (переменных) по всем примерам.

Позволяет построить зависимость одной или нескольких переменных по всем примерам в файле, даже если переменные имеют различные типы. Этот тип графиков полезен для анализа временных зависимостей.

2. График зависимости внутри примера

Используется, если все переменные в примере одного типа.

### 3. Корреляционная точечная диаграмма.

Позволяет построить зависимость одной переменной от другой для всех примеров, что демонстрирует корреляцию между ними. На этом графике отображается также линейный коэффициент корреляции между этими двумя переменными.

### 4. График High-Low-Close.

Позволяет выбрать переменные из файла данных, чтобы построить график в переменных максимальная цена (high) - минимальная цена (low) - цена закрытия (close), используемых для рыночных предсказаний.

### 5. Графики тренировки.

NeuroShell позволяет также отображать графики зависимостей средней ошибки на тренировочном и тестовом наборах от времени тренировки в процессе тренировки сетей с обратным распространением ошибки и другими алгоритмами.

Оценка программного продукта представлена в Табл. 4.

Табл. 4. Оценка программного пакета

Достоинства	Недостатки
Предоставлен набор графических средств	Сложный и запутанный интерфейс
Предлагается деление пользователей на группы: начинающие пользователи, специалисты.	Программа распределена на множество отдельных модулей, что уменьшает надежность системы.
Обеспечена возможность встраивания созданных сетей в программу.	Нет сопровождения к системе, которое описывает последовательность действий пользователя
Программа адаптирована под прикладные применения.	

## 4. Функциональные требования (функции продукта)

### 4.1 Обязательные функции для первой версии

1. Система аутентификации. При запуске программы открывается окно аутентификации. Пользователь выбирает группу, к которой он относится, свое имя, вводит пароль и нажимает ОК. При неправильном вводе пароля появляется предупреждение и предлагается пользователю ввести пароль ещё раз, при правильном вводе пароля открывается главное окно программы.
2. Отображение всех таблиц и навигация в таблицах. В главном окне, при нажатии определенных кнопок, запрос отправляется к СУБД, и ответ отображается в таблице. К таблице присоединен специальный компонент, осуществляющий навигацию по набору данных.

3. Заполнение основных таблиц. Предполагается создать специальные формы для заполнения основных таблиц. Заполнение осуществляется автоматически, а пользователь работает с абстрактным представлением выборки и нейронной сети.

#### 4.2 Дополнительные функции для первой версии

4. Осуществить слияние программы с алгоритмом обратного распространения и применить для хранения и обмена данными. Необходимо дополнить разграничение прав доступа программными средствами.

#### 4.3 Будущие функции

### 5. Основные способы использования программного продукта и сценарии работы с ним

На Рис. 7. Диаграмма вариантов использования. представлена диаграмма вариантов использования программы.

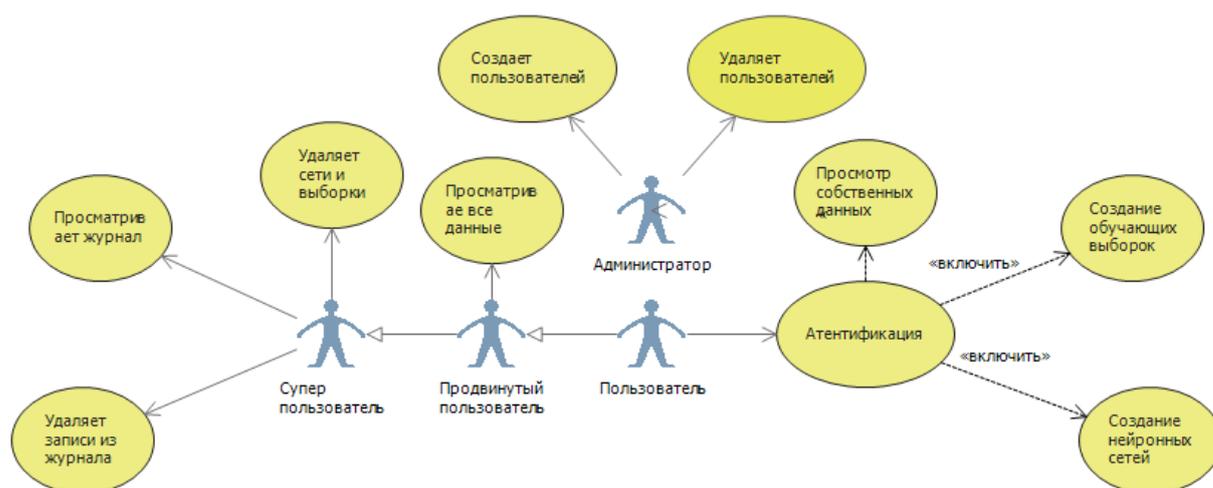


Рис. 7. Диаграмма вариантов использования.

#### 5.1 Простой способ использования программного продукта

Простота способа использования зависит от роли пользователя в системе. «Пользователь» имеет возможность самым простым образом использовать программный продукт.

#### 5.2 Полномасштабный способ использования программного продукта

См. варианты использования «Супер пользователя»

### 6. Нефункциональные требования

#### 6.1 Требования к программному продукту

Нет требований.

## 6.2 Организационные требования

Нет требований.

## 6.3 Внешние требования

Нет требований.

## 6.4 Прочие требования

Нет требований.

## 7. Требования к документации

### 7.1 Руководство пользователя

1. Должно содержать:
  - описание процедур работы с продуктом;
  - глоссарий.
2. Форма руководства – электронная, формат – *pdf*. Должна быть предусмотрена возможность вывода руководства на печать (полностью или отдельных разделов).

### 7.2 Руководство программиста

Должно содержать следующие сведения (но не обязательно ограничиваться ими):

- описание порядка установки программного продукта и его первичной
- настройки;
- описание модулей программы и их взаимосвязей;
- описание типовой структуры модуля и порядка его функционирования.

### 7.3 Интерактивная подсказка

### 7.4 Руководства по установке и конфигурированию и файл ReadMe

1. Установить PostgreSQL. Запустить приложение postgresql-9.0.4-1-windows.exe и следовать инструкциям установщика.
2. Создайте базу данных с именем nnbase. Выполнить команду для восстановления базы данных из файла nnbase\_copy.sql.
3. Установить драйвер для работы с PostgreSQL. Для этого копируем файлы PGOLEDB.DLL и LIBPQ.DLL в папку system32 и от администратора выполняем команду regsvr32 PGOLEDB.DLL.

### 7.5 Маркировка и упаковка

## **8. Глоссарий и список сокращений**

### **8.1 Глоссарий**

### **8.2 Список сокращений**

НС – нейронная сеть.

ИИ – искусственный интеллект.

ПП – программный продукт.

ПО – программное обеспечение.

БД – база данных.

СУБД – система управления базами данных.

## **II. АРХИТЕКТУРА И СИСТЕМНЫЕ ТРЕБОВАНИЯ**

### **1. Архитектура программного продукта и модульный состав подсистем**

На Рис. 8 представлена архитектура программного продукта. База Данных содержит рабочие таблицы, журнальные таблицы и некоторые системные.

СУБД взаимодействует со всеми таблицами, обеспечивая выполнение правил для заполнения журнальных таблиц. Также СУБД обеспечивает организацию групповых ролей для разграничения прав доступа пользователей. Пользовательское приложение взаимодействует с СУБД и предоставляет возможность просматривать таблицы и создавать новые записи. В пользовательском приложении БД позволяет упростить процедуру использования программиста. Использование БД позволяет улучшить использование алгоритма обратного распространения ошибки.

Модульный состав ПП можно посмотреть на Рис. 9.

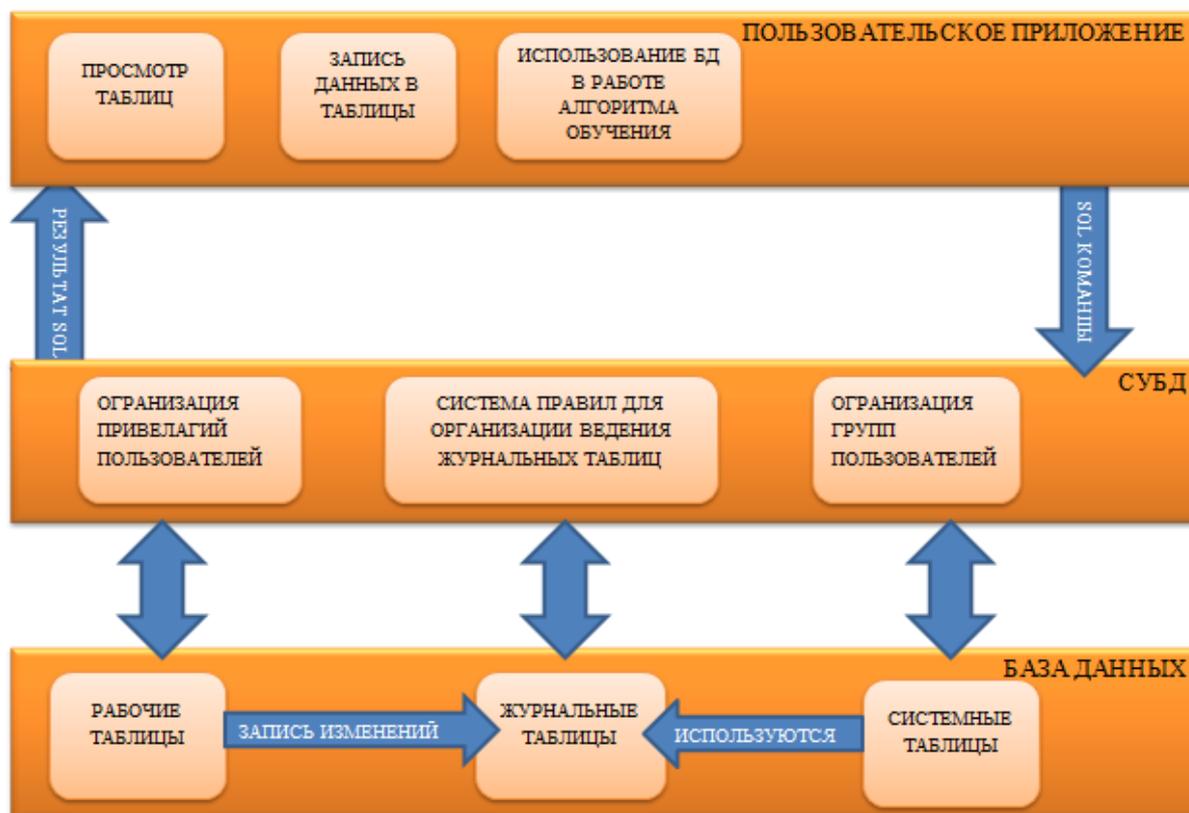


Рис. 8. Архитектура программного продукта

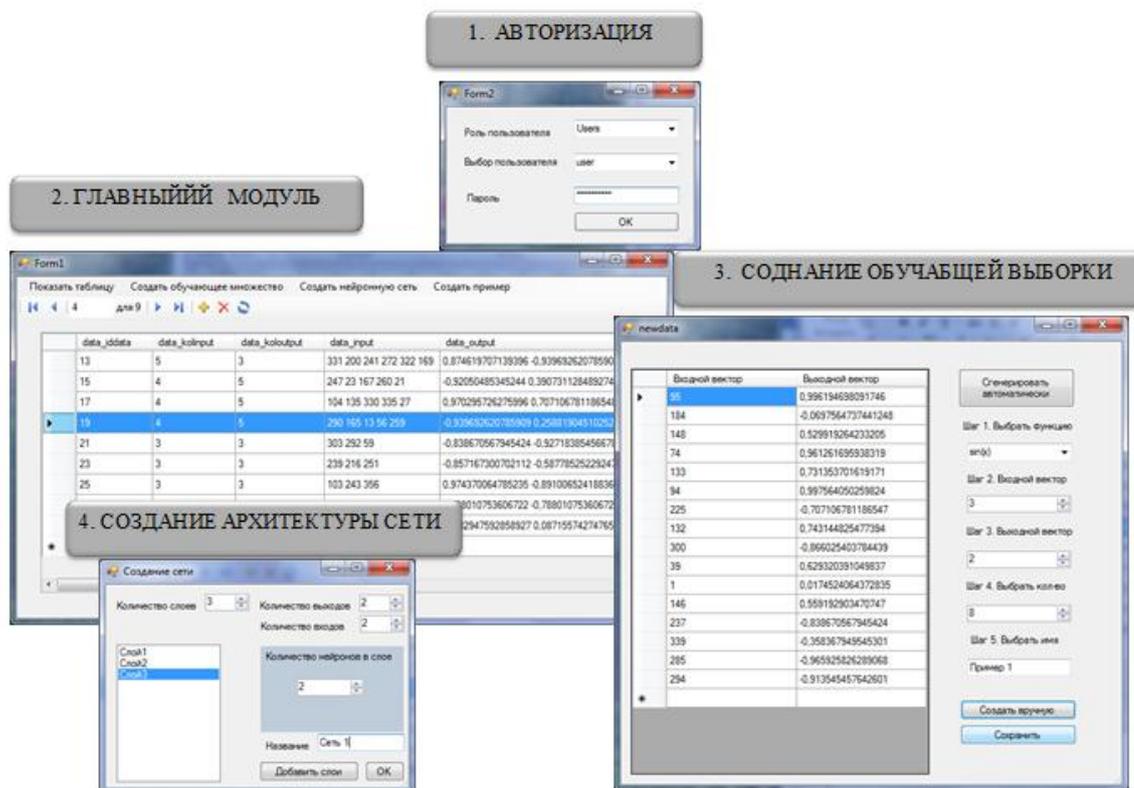


Рис. 9. Структура вызовов модулей пользовательской подсистемы

### III. ТЕХНИЧЕСКИЙ ПРОЕКТ

#### 1. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

##### 1. Концептуальное проектирование базы данных

Концептуальное проектирование базы данных – это процедура конструирования информационной модели предприятия, не зависящей от каких-либо физических условий реализации.

Каждая локальная концептуальная модель данных включает следующую информацию:

- типы сущностей
- типы связей
- атрибуты
- домены атрибутов
- потенциальные ключи
- первичные ключи

Типы сущностей – это объекты или концепции, которые характеризуются на данном предприятии как имеющие независимое существование.

Атрибут – это свойство типа сущности или типа связи.

Типы сущностей можно классифицировать как сильные и слабые.

Слабый тип сущности – это тип сущности, существование которого зависит от какого-то другого типа сущности.

Сильный тип сущности – это тип сущности, существование которого не зависит от какого-то другого типа сущности.

Доменом называется некоторый пул значений, элементы которого выбираются для присвоения значений одному или более атрибутам. Полностью разработанная модель данных должна включать домены для каждого из присутствующих в ней атрибутов.

Потенциальным ключом называется атрибут или минимальный набор атрибутов заданной сущности, позволяющей уникальным образом идентифицировать каждый ее экземпляр. Для некоторых сущностей возможно наличие нескольких потенциальных ключей. В этом случае среди них нужно выбрать один ключ, который будет называться первичным ключом.

Информация о содержании сущностей представлена в Таблице 1.1.

Таблица 5. Сведения о типах сущностей проекта

Имя сущности	Псевдонимы	Описание	Особенности использования(тип)
Нейронная сеть	network	Содержит информацию о структуре сети и синоптических весах	Сильный тип
Исходные данные	data	Содержит обучающие множество	Сильный тип
Рабочие примеры	example	Содержит нейронную сеть связанную с данными	Слабый тип
Пользователи	pg_shadow	Содержит пользователей системы	Сильный тип
Группы пользователей	pg_group	Содержит группы пользователей	Сильный тип

На Рис. 10 представлена модель «сущность-связь».

В качестве первичного ключа для всех таблиц выбран идентификатор записи каждой записи. Это было сделано для упрощения написания пользовательского приложения.

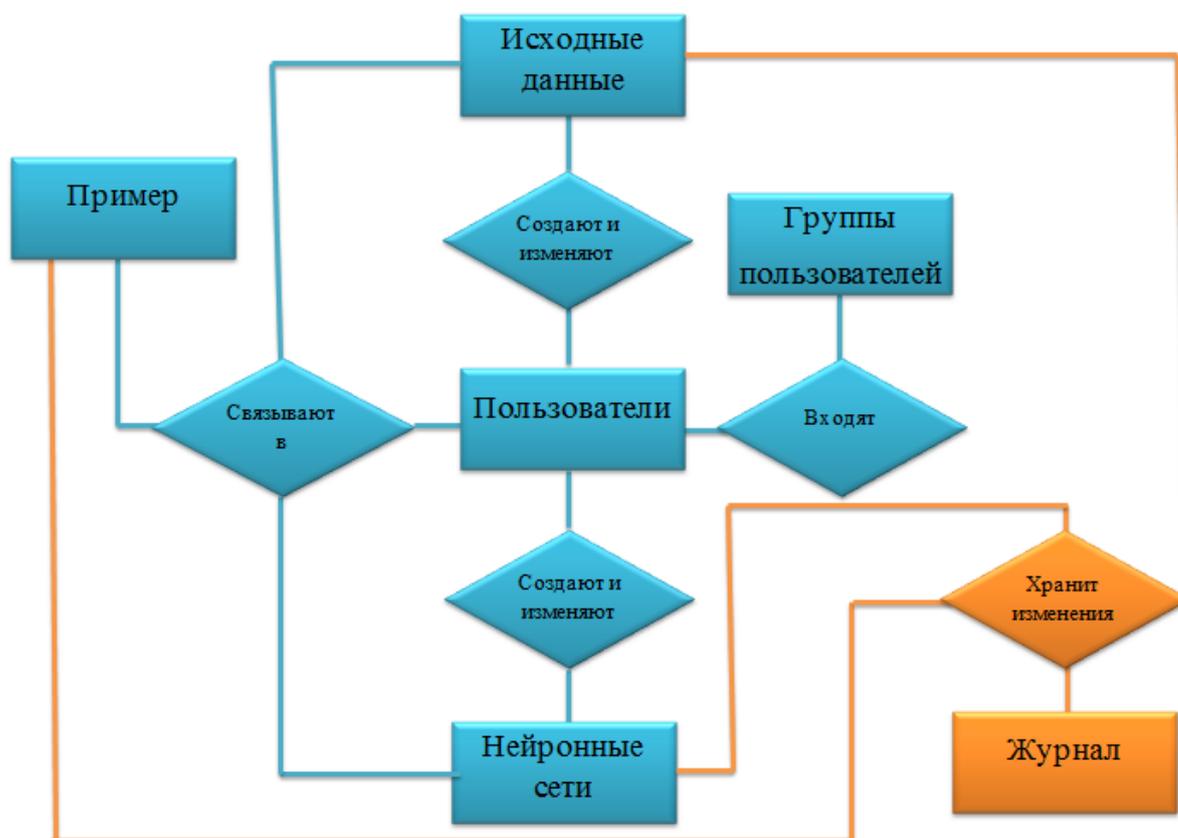


Рис. 10. Модель сущность-связь проекта

По модели «сущность-связь» для лучшего представления была составлена Табл. 6

Табл. 6. Сведения о типах связей проекта «Зарплата и сотрудники»

Тип сущности	Тип связи	Тип сущности	Кардинальность
Пользователь	Входит в	Группу пользователей	M:1
Пользователь	Создает и изменяет	Нейронные сети	1:M
Пользователь	Создает и изменяет	Исходные данные	1:M
Пользователь	Связывает нейронные сети и данные	Пример	1:M
Группы пользователей	Содержат	Пользователей	1:M
Журнал	Хранит изменения	Нейронные сети, Исходные данные, Пример	1:M

Описание атрибутов сущности и доменов атрибутов представлены в Табл. 7

Табл. 7. Атрибуты и домены атрибутов

Сущность	Атрибуты	Домены атрибутов
Данные	Код записи	Автоинкрементен
	Название	Строка
	Количество входов	Целое
	Количество выходов	Целое
	Входной вектор	Текст
	Выходной вектор	Текст
	Пользователь	Текст
	Дата создания	Дата/время
Примеры	Код записи	Автоинкрементен
	Название	Строка
	Код данных	Целое
	Код сети	Целое
	Пользователь	Текст
	Дата создания	Дата/время
Сети	Код записи	Автоинкрементен
	Название	Строка
	Структура сети	
	Весы сети	
	Вход сети	
	Выход сети	
	Пользователь	Текст
	Дата создания	Дата/время

В качестве сущностей Пользователи и Роли пользователей используются системные таблицы pg\_shadow и pg\_group.

Табл. 8. Поля таблицы pg\_group

Имя	Тип	Ссылки	Описание
groname	имя		Название группы
grosysid	int4		Произвольное число идентифицировать этот GROUP
grolist	int4 []	pg_shadow. Usesysid	Массив, содержащий идентификаторы пользователей в этой группе

Табл.9. Поля таблицы pg\_shadow

Имя	Тип	Описание
username	Имя	Имя пользователя
usesysid	int4	Идентификатор пользователя (Произвольное количество используется для ссылки на этого пользователя)
usecreatedb	Bool	Май создания пользовательских баз данных
usesuper	Bool	Пользователь является администратором
usecatupd	Bool	Пользователь может обновлять системные каталоги
Passwd	текст	Пароль (возможно, с шифрованием)
valuntil	abstime	Пароль истечения времени (используется только для парольной аутентификации)
useconfig	текст	Сессия по умолчанию для времени выполнения переменных

## 2. Логическое проектирование базы данных

Логическое проектирование базы данных представляет собой процесс конструирования модели информационной структуры организации, выполняемый в соответствии с выбранной схемой организации информации (например, реляционной). Однако создаваемая логическая модель не зависит от особенностей конкретной СУБД и физических условий реализации.

Действия, необходимые для преобразования концептуальной модели данных в логическую модель данных, включают: удаление связей типа M:N, удаление сложных связей, удаление рекурсивных связей, удаление связей с атрибутами, удаление множественных атрибутов, перепроверка связей типа 1:1 и удаление избыточных связей.

Логическая модель данных может быть проверена с помощью методов нормализации, а также на возможность выполнения всех требуемых транзакций. Нормализация используется для общего улучшения характеристик модели, что достигается с помощью введения различных ограничений, позволяющих избежать дублирования данных. Прове-

дение нормализации позволяет получить уверенность в том, что результирующая модель более точно отражает особенности организации, обладает внутренней согласованностью, минимальной избыточностью и максимальной устойчивостью.

Логическая модель представлена на Рис. 11. При проектировании логической модели были убраны повторяющиеся данные, перепроектированы таблицы таким образом, чтобы не было связей M:N. Также были найдены и удалены избыточные данные.

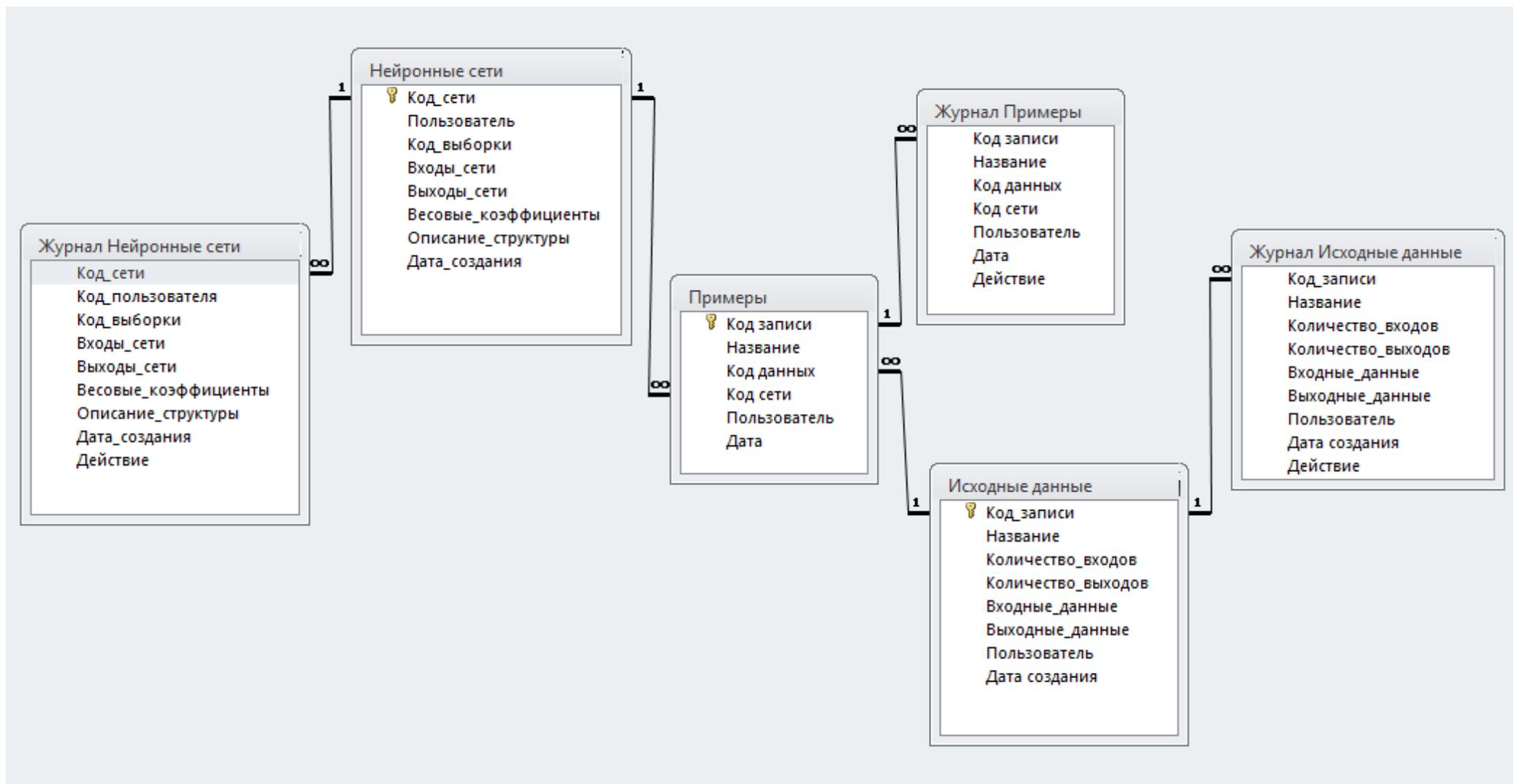


Рис. 11. Схема данных логической модели

### 3. Физическое проектирование базы данных

Физическое проектирование базы данных – это процесс создания описания конкретной реализации базы данных, размещаемой во вторичной памяти. Предусматривает описание структуры хранения данных и методов доступа, предназначенных для осуществления наиболее эффективного доступа к информации.

Фаза физического проектирования базы данных предусматривает принятие разработчиком окончательного решения о способах реализации создаваемой базы. Поэтому физическое проектирование обязательно производится с учетом всех особенностей используемой СУБД. Между фазами физического и логического проектирования всегда имеется определенная обратная связь, поскольку решения, принятые на этапе физического проектирования с целью повышения производительности разрабатываемой системы, могут потребовать некоторого пересмотра логической модели данных.

В Табл. 10 – Табл. 12 представлена информация о структуре таблиц спроектированной базы данных. На Рис. 12 изображена схема данных физической модели.

Для отслеживания действий пользователя созданы журнальные таблицы:

- Журнал данных(log\_data);
- Журнал сетей(log\_network);
- Журнал примеров(log\_example).

Поля журнальных таблиц полностью дублируют таблицы data, network и example, но добавляется дополнительное поле, которое содержит команду, выполняемую пользователем(INSERT, UPDATE, DELETE).

Журнальные таблицы необходимы для сохранения истории изменений данных, которая может понадобиться для восстановления данных.

Заполнение журнальных таблиц выполняется путем создания трех правил для операций вставки, обновления и удаления записей из рабочих таблиц. При выполнении операций с рабочей таблицей копии вставляемой, обновляемой или удаляемой записи вставляются в журнальную таблицу.

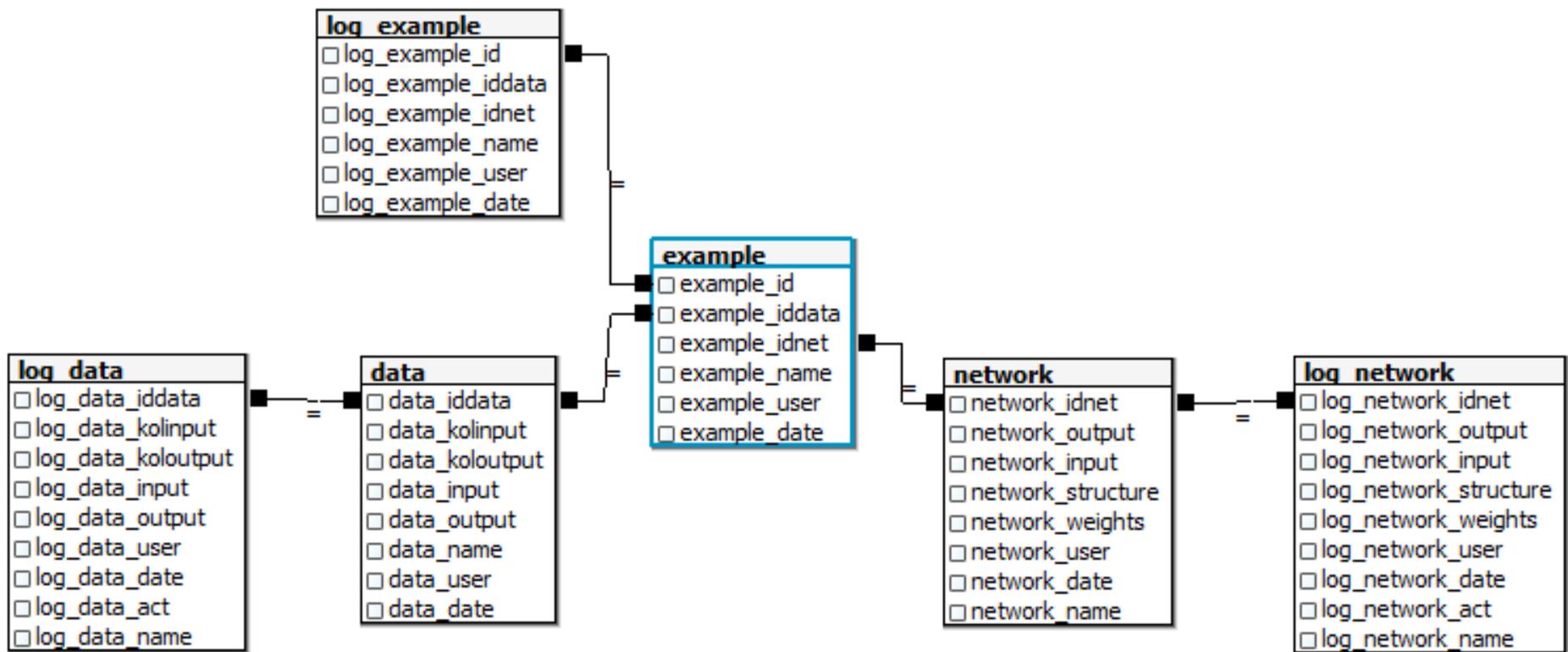


Рис. 12. Схема данных физической модели

Табл. 10. Структура таблицы Исходные данные (data)

Наименование поля	Тип поля	Размер поля	Ключевое или индексированное поле	Условие на значение	Содержание поля
data_iddata	serial		Ключевое	уникальное	Код записи
data_kolinput	integer				Количество входов
data_koloutput	integer				Количество выходов
data_input	text				Входной вектор
data_output	text				Выходной вектор
data_name	character varying	50			Имя данных
data_user	text				Владелец данных
data_date	timestamp				Дата создания

Табл. 11. Структура таблицы Примеры (example)

Наименование поля	Тип поля	Размер поля	Ключевое или индексированное поле	Условие на значение	Содержание поля
example_id	serial		Ключевое	уникальное	Код записи
example_iddata	integer				Код данных
example_idnet	integer				Код сети
example_name	character varying	50			Название примера
example_user	text				Владелец данных
example_date	timestamp				Дата создания

Табл. 12. Структура таблицы Нейронные сети (network)

Наименование поля	Тип поля	Размер поля	Ключевое или индексированное поле	Условие на значение	Содержание поля
network_idnet	serial		Ключевое	Уникальное	Код записи
network_output	text				Количество входов
network_input	text				Количество выходов
network_structure	text				Структура нейронной сети
network_weights	text				Веса нейронной сети
network_user	text				Владелец данных
network_date	timestamp				Дата создания
network_name	character varying	50			Имя сети

## 2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

### 1. Структура программного продукта

Программный продукт состоит из одного исполняемого файла, включающего в себя 5 форм. Более подробно структуру ПП см. Архитектура и системные требования

### 2. Реализация бизнес-правил

Бизнес-правила реализованы на основе SQL-запросов.

#### Запрос 1

```
SELECT groname FROM pg_group;
```

#### Запрос 2

```
SELECT r.rolname FROM pg_group g, pg_roles r  
WHERE g.groname = + Convert.ToString(comboBox2.SelectedItem)+  
AND r.oid = any( g.grolist );
```

+ – конкатенация;

Convert.ToString() – функция преобразования в строку;

comboBox2.SelectedItem – имя роли;

any() – функция для сопоставления кода роли с массивом ролей.

#### Запрос 3

```
SELECT DISTINCT network_name FROM network where network_user=+ User;  
DISTINCT – не повторяющиеся записи.
```

#### Запрос 4

```
SELECT DISTINCT data_name from data where data_user=+ User;
```

#### Запрос 5

```
SELECT g.groname FROM pg_group g, pg_shadow u  
WHERE u.username = + User + AND u.usesysid = any( g.grolist);
```

#### Запрос 6

```
SELECT data_iddata AS Код_записи,  
data_kolinput AS Количество_входов,  
data_koloutput AS Количество_выходов,  
data_input AS Входной_вектор,  
data_output AS Выходной_вектор,  
data_name AS Название_множества  
FROM data where data_user = + User;
```

User – поле класса, содержащее текущего пользователя.

### Запрос 7

```
SELECT network_idnet AS код_записи,  
       network_kolinput AS количество_входов,  
       network_koloutput AS количество_выходов,  
       network_output AS выходной_вектор,  
       network_input AS входной_вектор,  
       network_structure AS структура_сети,  
       network_weights AS веса_сети,  
       network_name AS название_сети  
FROM network where network_user = + User;
```

### Запрос 8

```
SELECT example_id AS код_записи,  
       data_name AS имя_данных,  
       network_name AS имя_сети,  
       example_name AS имя_примера,  
       example_train AS обучающий_вектор,  
       example_test AS тестовый_вектор  
FROM example, data, network  
WHERE example_iddata=data_iddata  
AND example_idnet=network_idnet  
AND example_user = + User;
```

### Запрос 9

```
SELECT count(data_iddata) FROM data  
WHERE data_name=+comboBox2.SelectedItem;  
comboBox2.SelectedItem – название данных.
```

### Запрос 10

```
SELECT data_iddata from data  
WHERE data_name=+ comboBox2.SelectedItem;
```

### Запрос 11

```
SELECT network_idnet FROM network  
WHERE network_name=+ comboBox1.SelectedItem;  
comboBox1.SelectedItem – название сети.
```

### Запрос 12

```
INSERT INTO example (example_id, example_idnet, example_iddata, example_name,  
example_train, example_test)  
VALUES (default, Convert.ToString(idnet), Convert.ToString(Arr[i]), textBox1.Text ,true,false);  
Idnet – код сети;  
Arr[i] – элемент массива с кодами данных;  
textBox1.Text – название примера.
```

### Запрос 13

```
INSERT INTO data(data_iddata, data_kolinput, data_koloutput, data_input,  
data_output,data_name)  
VALUES (default, Convert.ToString(kolin), Convert.ToString(kolout), In, Out, name);
```

Kolin – количество входов;

Kolout – количество выходов;

In – входной вектор;

Out – выходной вектор;

Name – название данных.

## Запрос 14

```
INSERT INTO network (network_idnet, network_structure, network_name, network_kolinput,  
network_koloutput)
```

```
VALUES (default, str, name, Convert.ToString(kolin), Convert.ToString(kolout));
```

Str – структура нейронной сети.

## 3. Руководство программиста

Программа предназначена для автоматизации процесса использования данных в алгоритме обратного распространения ошибки.

Возможности программы:

- Обеспечена возможность хранить в базе данных информацию.
- Обеспечена возможность ввода всех необходимых данных и их редактирования.
- Предусмотрены простые средства поиска в базе данных и выборки информации из нее.
- Обеспечен удобный логичный интерфейс. Все данные отображаются наглядно для пользователя.

### *Характеристики программы*

*Время загрузки программы:* не больше 10 секунд;

*Вход в программу:* осуществляется с помощью исполняемого модуля;

*Режим работы:* запуск по мере необходимости.

Диаграмма классов представлена на Рис. 13.

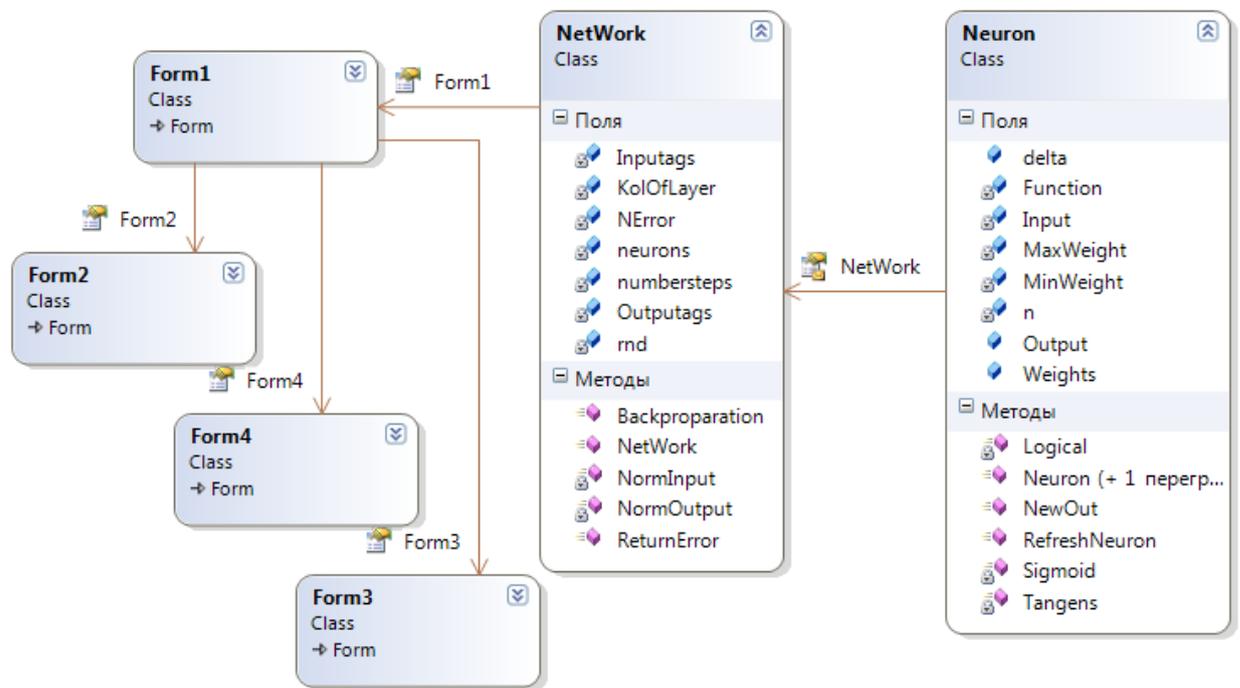


Рис. 13. Диаграмма классов

### *Поля и методы класса Network*

Поля класса:

- KolOfLayer – количество нейронов в слое;
- Inputtags – входные данные для НС;
- Outputtags – выходные данные НС;
- neurons – ступенчатый массив нейронов;

Методы класса:

- NetWork() – конструктор класса;
- Backproparation() – метод реализующий алгоритм обратного распространения;
- Train() – метод осуществляющий тестирование сети;
- NormInput() – метод нормализации входа;
- NormOutput() – метод нормализации выхода.

### *Поля и методы класса Neuron*

Поля класса:

- n – количество входов нейрона;
- Input – подаваемый образ;
- Output - выход нейрона;
- Weights - веса нейрона;
- Function – активационная функция нейрона;
- delta - изменение весов нейрона.

Методы класса:

- Neuron() – конструктор класса;

- RefreshNeuron() – метод обновляющий нейрон;
- NewOut() – пересчет выхода нейрона;
- Sigmoid() – функция активации сигмоид;
- Tangens() – функция активации гиперболический тангенс;
- Logical() – логистическая функция активации.

#### 4. Руководство пользователя

Для использования программного продукта необходимо выполнить пункты, указанные в I.7.4.

Пользователь проходит регистрацию в программе Рис. 14.

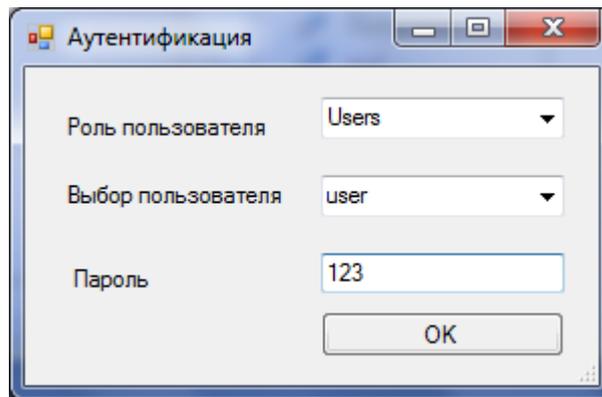


Рис. 14. Окно аутентификации

Пользователь имеет возможность создавать нейронные сети Рис. 15.

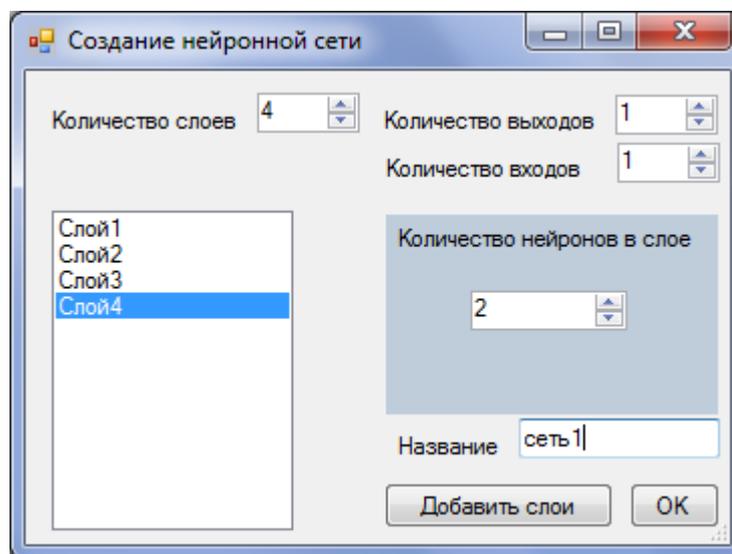


Рис. 15. Создание нейронной сети

Пользователь имеет возможность создавать обучающие множества Рис. 16.

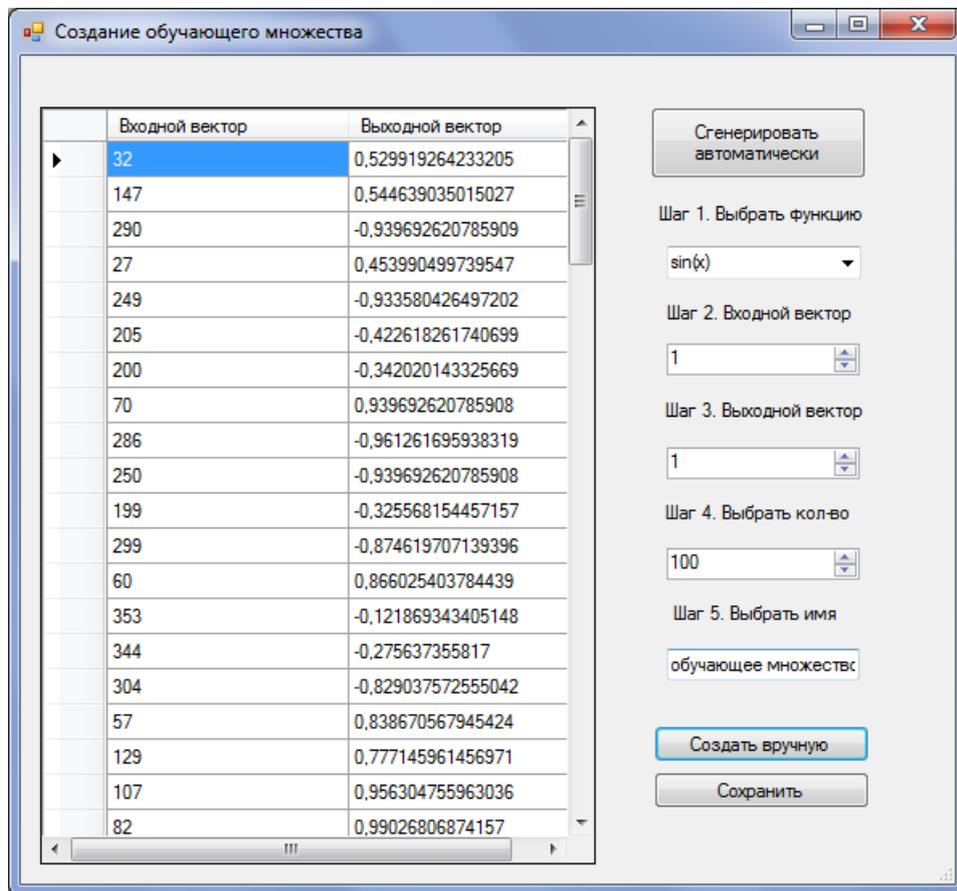


Рис. 16. Создание обучающего множества

Пользователь имеет возможность просматривать таблицы и соединять нейронную сеть с данными Рис 17.

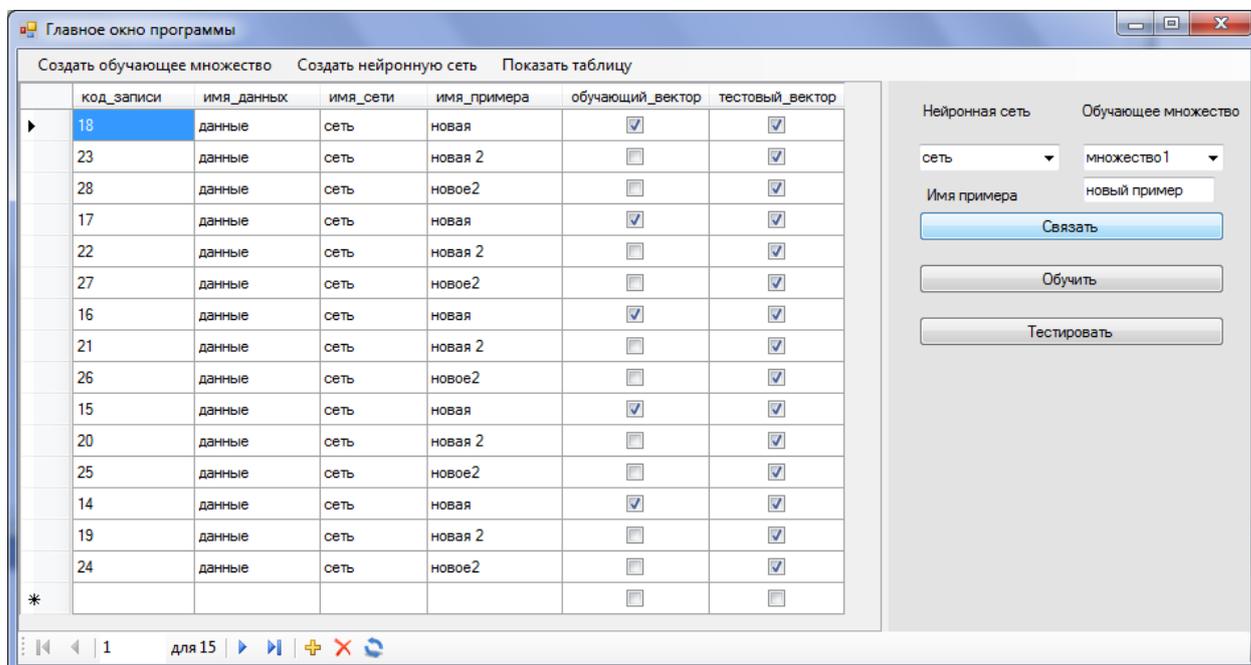


Рис 17. Главное окно программы