**Министерство науки и высшего образования Российской Федерации**
Федеральное государственное бюджетное образовательное учреждение высшего образования
**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**
**(ФГБОУ ВО «АмГУ»)**

Институт компьютерных и инженерных наук
Кафедра математического анализа и моделирования
Направление подготовки – 01.04.02 Прикладная математика и информатика
Направленность (профиль) образовательной программы – Математическое и программное обеспечение информационных систем

ДОПУСТИТЬ К ЗАЩИТЕ
И.о. зав. кафедрой
_____ Н.Н. Максимова
«_____»_____ 2024 г.

# МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Эволюционный алгоритм построения нейронной сети в приложении к гравитационному методу в золотодобыче

Исполнитель
студент группы 2101ом _____ (подпись, дата) А.И. Перепёлкин

Руководитель
канд. физ.-мат. наук _____ (подпись, дата) Л.И. Мороз

Руководитель научного содержания программы магистратуры
профессор, д-р физ.-мат. наук _____ (подпись, дата) А.Г. Масловская

Нормоконтроль
старший преподаватель _____ (подпись, дата) А.Н. Дудин

Рецензент
доцент, канд. тех. наук _____ (подпись, дата) С.Г. Самохвалова

Благовещенск 2024

**Министерство науки и высшего образования Российской Федерации**
Федеральное государственное бюджетное образовательное учреждение высшего образования
**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**
**(ФГБОУ ВО «АмГУ»)**

Институт компьютерных и инженерных наук
Кафедра математического анализа и моделирования

УТВЕРЖДАЮ
И.о. зав. кафедрой
_____ Н.Н. Максимова
«_____»_____2024 г.

# З А Д А Н И Е

К выпускной квалификационной работе студента Перепёлкина А.И.

1. Тема выпускной квалификационной работы: «Эволюционный алгоритм построения нейронной сети в приложении к гравитационному методу в золотодобыче» (утверждена приказом от 14.05.2024 № 1219-уч).
2. Срок сдачи студентом законченной работы (проекта) 20.06.2024 г.
3. Исходные данные к выпускной квалификационной работе: отчет по преддипломной практике, отчеты по научно-исследовательской работе, C#, TensorFlow, OpenNMT-py.
4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов): обзор технологий и оборудования для промышленной золотодобычи, разработка эволюционного алгоритма построения нейронной сети, настройки режимов работы концентрационного стола на основе нейросетевого подхода, разработка пользовательского приложения.
5. Перечень материалов приложения: листинг мобильного приложения, пользовательский интерфейс.
6. Консультанты по выпускной квалификационной работе (с указанием относящихся к ним разделов): рецензент – доцент, канд. тех. наук, доцент кафедры «Информационные и управляющие системы» АмГУ Самохвалова Светлана Геннадьевна; нормоконтроль – Дудин А.Н., старший преподаватель.
7. Дата выдачи задания: 29.02.2024 г.

Руководитель выпускной квалификационной работы: Мороз Любовь Игоревна канд. физ.-мат. наук, ведущий научный сотрудник.

Задание принял к исполнению (29.02.2024): _____ Перепёлкин А.И.

# РЕФЕРАТ

Магистерская диссертация содержит 79 с., 15 рисунков, 31 источник.

НЕЙРОННАЯ СЕТЬ, ЭВОЛЮЦИОННЫЙ АЛГОРИТМ, МНОГОСЛОЙНЫЙ ПЕРЦЕПТРОН, ЗОЛОТОДОБЫЧА, ГРАВИТАЦИОННЫЙ МЕТОД

Методы машинного обучения ежегодно находят все более широкое применение. Одним из приложений нейронных сетей является добыча полезных ископаемых. Например, методы машинного обучения используют для настройки эксплуатационных параметров оборудования. Особенно это актуально с постоянно запаздывающей относительно изменения характеристик подаваемой на переработку руды информацией. Одним из преимуществ нейросетевого подхода является устойчивость нейросетевой модели по отношению к ошибкам во входных данных.

Несмотря на все достоинства данного подхода, существующие нейронные сети не всегда показывают высокую точность результатов. Поэтому актуальным является построение нейронной сети с более высокими показателями точности вычислений.

Научно-исследовательская работа направлена на поиск оптимальной конфигурации настройки работы концентрационных столов с помощью методов машинного обучения, а также на разработку инструмента для минимизации человеческих трудозатрат при подборе структуры нейронной сети и создания удобного пользовательского приложения.

Объектом исследования является режим работы концентрационного стола, предметом исследования – алгоритмы поиска структуры искусственной нейронной сети для настройки оптимальных параметров режимов работы концентрационного стола.

**Цель** научных исследований связана с применением нейронных сетей для получения оптимальных параметров режимов работы концентрационного стола в золотодобыче.

В магистерской диссертации представлен эволюционный алгоритм построения архитектуры нейронной сети, базирующийся на внедрении «мутаций» и отборе более оптимальной структуры нейронной сети. Первоначально апробация алгоритма была проведена на примере обыкновенных дифференциальных уравнений, а затем адаптирована для настройки работы концентрационного стола, использующегося в золотодобыче. Разработано программное обеспечение в Visual C#.

Результаты научно-исследовательской работы докладывались и обсуждались на одной научной конференции вузовского уровня. За время работы над темой научного исследования были опубликованы 2 научные работы и поданы документы на регистрацию программы.

СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

**Актуальность и проработанность темы.**

В последнее время, с ростом вычислительных мощностей и накопленных данных, наблюдается увеличение количества приложений нейросетевого подхода в различных направлениях науки, техники и промышленности. Одним из таких приложений является золотодобыча. Так, с помощью нейронных сетей проводят исследования, связанные с поиском геологических зон залегания металла, распознаванием частиц золота по фотоснимках проб отложений, моделированием экологических последствий при золотодобыче и т.д. Кроме того, с помощью нейронных сетей можно производить настройку эксплуатационных параметров оборудования, в частности концентрационных столов, или даже целых предприятий.

На сегодняшний день установить правильный режим работы концентрационного стола можно только опытным путем, принимая во внимание состав исходного сырья, а также требования, которые предъявляются к готовому продукту. Самое важное – задать правильный режим рыхления состава, корректируя подачу воды и возвратно-поступательные движения рабочей поверхности, а также обеспечить по возможности предварительную классификацию по крупности исходного питания (руда поступает на концентрационный стол в виде пульпы через бункер питания, поэтому поступающий материал называют питанием). Оператор может самостоятельно менять как поперечный, так и продольный наклон, длину и частоту хода. Кроме того, технологические характеристики рабочего процесса напрямую зависят от частоты и амплитуды колебаний стола и расхода воды.

Оптимальная работа добывающих механизмов и устройств позволяет увеличить количество добываемого золота, в том числе, тонкого, мелкого золота (ТМЗ), что соответствующим образом влияет на прибыль компании. Поэтому настройка работы концентрационных столов на основе нейросетевого подхода и разработка соответствующего программного обеспечения является актуальной

задачей. В качестве обучающей выборки были использованы данных технологических процессов журналов гравитационной добычи золота в Амурской области компании ООО «Атлант».

**Научно-исследовательская работа** направлена на поиск оптимальной конфигурации настройки работы концентрационных столов с помощью методов машинного обучения, а также на разработку инструмента для минимизации человеческих трудозатрат при подборе структуры нейронной сети и создания удобного пользовательского приложения.

**Объектом** исследования является режим работы концентрационного стола, **предметом** исследования – алгоритмы поиска структуры искусственной нейронной сети для настройки оптимальных параметров режимов работы концентрационного стола.

**Цель** научных исследований связана с применением нейронных сетей для получения оптимальных параметров режима работы концентрационного стола в золотодобыче.

Для достижения поставленной цели были сформулированы следующие **задачи**:

– исследование нейросетевых методик машинного обучения;

– подготовка данных для настройки и обучения нейросети;

– разработка эволюционного алгоритма построения нейронной сети;

– программная реализация;

– верификация и анализ полученных результатов по настройке работы режимов концентрационного стола.

Для выполнения исследований, представленных в работе, были использованы следующие **методы** и **программные средства**: методы теории нейронных сетей, C#, Maple, TensorFlow, OpenNMT-py, JavaScript.

**Новизна** полученных результатов заключается в идее использования эволюционного подхода в генерации структур нейронных сетей для минимизации человеческих трудозатрат и получение максимально

оптимизированной нейронной сети в приложении к настройке работы концентрационного стола.

**Теоретическая значимость** работы заключается в развитии подхода в построение структур нейронных сетей.

**Практическая значимость** работы состоит в возможности использования разработанных алгоритмических и программных средств для оперативной настройки режима работы концентрационных столов в золотодобыче.

Результаты работы были **апробированы** на XXXIII научной конференции «День науки», г. Благовещенск, 2024 г. (очное участие).

Работа также была представлена в 2022-2024 гг. на научных семинарах кафедры математического анализа и моделирования.

За время работы над темой научного исследования **опубликованы** 2 научные работы и получено 1 свидетельство о регистрации программы, библиографическое описание которых приведено в общем списке литературы.

**Структура работы.** Работа состоит их введения, трех глав, заключения, библиографического списка и двух приложений. В первой главе подставлены основные понятия золотодобычи. Во второй главе рассказывается о нейронных сетях и их видах. В третьей главе описан эволюционный механизм подбора структуры нейронной сети с последующим приложением в программе для настройки оптимальных параметров режима работы концентрационного стола.

# 1 ЗОЛОТОДОБЫВАЮЩАЯ ПРОМЫШЛЕННОСТЬ: ОСНОВНЫЕ МЕТОДЫ

На сегодняшний день золото является одним из самых ценных в мире металлов, нашедшее широкое применение в промышленном производстве, ювелирном деле, сохранении капитала и инвестировании. Первыми шагами перед получением результата являются разведка; нахождение места добычи; определение состава породы – самородки или руда и подсчет затрат (сколько людей и оборудования потребуется для получения металла) [1].

Любой из известных способов добычи драгоценного металла является трудоемким в физическом и технологическом плане процессом, поиск самородков, извлечение золота из руды требует большого количества затрат. После получения положительных разведданных и проведения опытно-промышленных разработок, определения строения рудных тел, выбирается способ добычи, а также условия переработки, обогащения, производится примерный расчет количества получаемого металла.

Самородное золото, залегающее в земной коре, имеет примеси в виде серебра (до 40-45%), а также железа, меди, марганца и свинца. Цвет такого металла может быть бледно-жёлтым, красно-жёлтым и зеленоватым. Самым большим в мире самородком является так называемая «Плита Холтермана». Этот кусок золота был найден в Австралии в 1872 году. Его вес составил более 90 кг, но, к сожалению, самородок был пущен на переплавку и до наших дней не сохранился.

В современном мире самородное золото встречается редко. Как правило, для получения чистого благородного металла применяют разнообразные методы, основываясь на его физических и химических свойствах. Существует 4 основных способа получения золота: промывка, амальгамация, цианирование и регенерация.

Промывка относится к ручному способу добычи, используется с самых древних времен и получила распространение благодаря высокой плотности золота. Минералы с меньшей плотностью просто смываются в потоке воды, в

результате чего образуется шлих, состоящий из частиц песка и золота. Если объёмы небольшие, применяется промывочный лоток-лентяйка, их часто используют старатели для отработки небольших россыпных месторождений.

В крупных масштабах для получения золота используют драги (горно-обогатительные агрегаты), механическое оборудование способное работать под водой и при необходимости раскапывать дно реки. Промышленные приборы разной мощности для добычи золота также являются очень востребованными механическими устройствами, используемые старателями [2-3].

Полученные шлихи в этом случае могут содержать ряд других тяжёлых металлов, кроме золота. Под шлихом понимается концентрат тяжелых металлов и минералов, добываемых из природных отложений или горных пород, которые остаются в воде после промывки. Для их выведения применяют другие способы получения золота, например амальгамацию.

Так как ртуть при взаимодействии с золотом благодаря диффузии способна притягивать частицы драгоценного металла не растворяя её, а образовывая жидкий сплав, который называются амальгамой. Увлажнённую раздробленную породу смешивают с ртутью и оставляют на несколько часов для амальгации, а затем удаляют остатки шлака. Нагревая амальгаму, отделяют ртуть от золота, получившийся шлам промывают и используют для дальнейшей обработки. Наибольший эффект достигается при максимально очищенных от масел и налетов золотых частиц, поэтому перед амальгацией желательно провести химическую и механическую очистку.

Цианирование – гидрометалургический химико-технологический процесс извлечения золота из кремнистых и серебряных руд, основан на факте, что золото и серебро хорошо растворяются в цианидах (цианистый калий и цианистый натрий). Реакция золота с цианидами происходит в присутствии кислорода, содержащегося в воздухе, после получения нужного раствора драгоценный металл осаждают металлическим цинком.

Данный способ включает в себя два этапа:

1. Обработка золотоносной породы раствором цианида натрия (менее 1%):

$4Au + 8NaCN + O2 + 2H2O \rightarrow 4Na[Au(CN)2] + 4NaOH;$

2. Осаждение золота из цианоаурата натрия при помощи цинковой пыли:

$2Na[Au(CN)2] + Zn \rightarrow Na2[Zn(CN)2] + 2Au\downarrow.$

Цианирование становится невозможным, если руда содержит большое количество сульфидов или арсенидов, поскольку цианиды дают реакцию с этими веществами. Для отделения сульфидов от золота успешно применяют метод флотации, когда с помощью водного раствора выделяют частицы золота из рудной массы. Технологический процесс происходит в специальных флотационных установках и применяется формула раздела фаз, когда плохо смачиваемые водой твердые частицы драгоценных металлов отделяются от хорошо смачиваемых сульфидов.

### 1.1 Золотодобыча в России

Первые золотые прииски в России были открыты на Урале только в 1814 году, но настоящая золотая лихорадка охватила страну в 1830-1860 годах. С открытием сибирского золота Россия быстро обогнала все другие страны по его добыче. Весь процесс добычи проходил в то время вручную. Этим способом могли заниматься и несколько человек, и одиночки, поскольку в технике тогда не было необходимости, так как золото лежало на поверхности. Промывая гальку в речных долинах в поисках ценного песка, старатели положили начало тем способам золотодобычи, которые применяются сегодня.

Масштабную добычу золота производили посредством специальных желобов, которые устилались изнутри хворостом или шкурами животных. В желоба направляли воду, туда же загружали и пески, где их и промывали.

В XIX веке для извлечения драгоценного металла из россыпных месторождений создали многочисленные конструкции золотоизвлекающих машин: бутару, вашгерд, водобои, землечерпалки, драги.

Золота в общей массе земной коры – 0,000001%, оно является составляющим многих химических и физических веществ, присутствует даже в человеческом теле, однако, в земле находится в распылённом виде. Месторождения представляют собой места с наибольшей концентрацией, их подразделяют на

коренные и россыпные. Россыпные месторождения сформировались вследствие различного рода воздействий химического или физического характера на коренные залежи.

При разработке таких месторождений применяют открытый способ, имеющий ряд существенных преимуществ: более высокая степень извлечения полезных ископаемых, относительная простота добычи и низкая себестоимость разработки месторождений. Сегодня в России открытым способом добывают более 60% объёма руд цветных металлов [3-5].

### 1.2 Современные машины для добычи золота

Добыча золота – это особая технология с применением разнообразного специализированного оборудования. Сейчас в России и за рубежом производят многочисленные модели установок, позволяющих вести высокопроизводительную золотодобычу.

В России и странах СНГ накоплен значительный опыт применения пластинчатых грохотов.

1.2.1 Грохот.

Один из достойных внимания – ГПГМ – грохот промышленный гидромеханический, который выпускает Челябинское производственное объединение «Уральские технологии». Грохот ГПГМ предназначен для промывки и обогащения золотосодержащих песков при бульдозерной разработке продуктивных пластов песков россыпных валунистых месторождений.

Прибор востребован в Якутии, Бурятии, на Чукотке, в Иркутской, Читинской, Магаданской, Амурской области, Хабаровском крае, Казахстане, Таджикистане, Узбекистане и пр.

Грохот представляет собой комплект мобильных агрегатов и узлов, каждый из которых имеет собственную ходовую часть – сани, позволяющие транспортировать части прибора в отдельности после разборки соединяющих коммуникаций. Грохот промышленный гидромеханический ГПГМ представлен в двух модификациях: с шириной полотна 3 м и 5 м. Конкретную комплектацию грохота ГПГМ определяет заказчик в зависимости от предполагаемых горно-

геологических условий эксплуатации. Возможна установка грохота непосредственно в забое полигона, при этом хвосты промывки периодически убирает бульдозер, либо на отвале вскрыши. Подача песков может осуществляться бульдозером, погрузчиком, автосамосвалом.

ГПГМ представляет собой сваренную из толстостенного материала ванну, установленную на санях с углом наклона 12°. Внутри ванны расположены катки, на них опирается тележка, несущая колёса. На колёсах лежит пластинчатое полотно – основной рабочий орган грохота. Оно состоит из цепей оригинальной конструкции и пластин из износостойкой стали, крепящихся к цепям высокопрочными болтами. Снизу пластины усилены резьбовыми планками. Пластины полотна располагаются с зазором 30-50 мм, определяемым размером обогащаемой фракции. Цепи состоят из усиленных звеньев простой геометрии на шарнирных соединениях, запрессованных в отверстия звеньев с большим натягом, и фиксирующих крепёжных накладок. Полотно свободно провисает между колесами, оба его конца закреплены на торцевых стенках ванны. При помощи гидроцилиндра тележка совершает возвратно-поступательные движения, а полотно создаёт эффект «бегущей волны». В результате подаваемые бульдозером или погрузчиком пески подвергаются различным манипуляциям, в том числе и ворошению – грохочению при интенсивном орошении водой. Эфельная фракция поступает в приёмный бункер-лоток через поперечное окно в днище ванны, далее попадает в шлюз. Хвосты со шлюза самотёком направляются в отвал. Надрешётный материал, так называемая галя, под действием эффекта «бегущей волны» сбрасывается с заднего торца ванны грохота и удаляется бульдозером в отвал [6-7].

1.2.2 Концентрационные столы.

Концентрационный стол предназначен для сортировки раздробленной руды. С помощью него отделяют ценную руду цветных, черных, редких и драгоценных металлов от пустой породы. За час концентрационный стол может переработать до 10 тонн раздробленной породы. Аппараты применяются в горнодобывающей промышленности для получения руды высокой концентрации.

Концентрационный стол состоит из следующих частей:

- Рама из П-образного металлического профиля. На ней крепятся все элементы конструкции.

- Дека (стол). На ней происходит процесс разделения породы на составляющие. Дека представляет собой прямоугольный стол. Верх стола закрыт накладками из дерева, алюминия, пластика, или резины. Накладки имеют рифленую поверхность. Высота рифелей от 0,1 до 5 мм. Дека подвешена на раме на мощных, тугих пружинах.

- Электрический двигатель с приводным механизмом служит для придания деке возвратно-поступательного движения.

- Заливной желоб. Закреплен с боковой или торцевой стороны деки. В него поступает струя воды с дробленными минералами (пульпа). Желоб распределяет напор пульпы на всю ширину стола.

- Сборники породы. В них собирается разделенное на фракции сырье. Как правило, сырье разделяется на 3 части (тяжелое, среднего веса и пустая порода), но бывают концентрационные столы, в которых исходное сырье разделяется по удельному весу на 4 и более частей.

- Механизм регулировки наклона стола. Регулируя наклон деки стол настраивают на разделение сырья разного веса и зернистости.

Во время работы смесь воды и дробленой породы подается в закрепленный сверху деки желоб. Длинный желоб с отверстиями внизу распределяет поток смеси по поверхности стола. Вода, попадая на верхний край наклонного стола, стекает с него вниз, унося легкие частицы породы в сборник. Тяжелые частицы породы опускаются на поверхность стола, зацепляются за канавки и движутся вдоль них по наклонной плоскости перемещаясь постепенно во второй сборник. Двигатель с помощью эксцентрикового механизма привода встряхивает стол, способствуя движению тяжелой породы в сторону сборника. Таким образом, смесь измельченной породы разделяется на два потока, унося легкие частицы в один сборник, а тяжелые в другой.

Концентрационные столы разделяются по нескольким показателя.

- По виду разделяемого сырья разделяют на песковые, шламовые и тонкошламовые столы. От этого показателя зависит, для разделения какого диаметра частиц предназначен стол. Песковые столы рассчитаны на разделение крупных частиц, на шламовых промывают пылевидную породу.

- По виду исполнения разделяют столы с левыми и правыми деками. От этого показателя зависит, на какую сторону будет выводиться разделенная порода.

- По количеству дек разделяют однодечные и многодечные системы. На небольших столах устанавливают одну деку, на мощные производительные разделительные станции устанавливают до 12 дек.

- По расположению промывочных столов разделяют одно, двух и трех ярусные аппараты [8].

Основное назначение концентрационных столов – разделение твердых частиц в породе по их плотности. При этом материалы разной плотности раскладывается в разные контейнеры. На большинстве столов материал делится на три и более фракции.

Концентрационные столы применяются в горнодобывающей промышленности для обогащения руды (отделяя из общей массы пустую породу) и выделения из потока породы металлов с определенной плотностью (редкие металлы). В золотодобывающей промышленности на концентрационных столах из пульпы выделяют золото, сульфидные материалы и пустую породу. На угледобывающих предприятиях разделяют уголь и каменную крошку.

1.2.3 Виброшлюзы.

На сегодняшний день многие золотодобывающие предприятия сталкиваются с проблемами извлечения ТМЗ (тонкое, мелкое золото).

Россыпные месторождения с крупным золотом истощаются, и золотодобытчики вынуждены разрабатывать месторождения с ТМЗ, которое тяжело поддается гравитационному извлечению и, как правило, при обогащении золотосодержащих песков происходят большие потери золота, что сильно сказывается на рентабельности золотодобывающих предприятий. При разработке

месторождений с рабочим золотом доля ТМЗ бывает очень высокой, и тогда предприятие несет убытки.

В мире существует очень много различного оборудования и технологий для улавливания ТМЗ, для различного вида золота и разнообразных горно-геологических условий. Определенными технологическими цепочками, состоящими из сортировочного и обогатительного оборудования, можно добиться извлечения до 90%. Есть изготовители оборудования, которые утверждают, что на их оборудовании можно «поймать» 100%, но опыт предприятий показывает, что это невозможно. 100% извлечения драгоценного металла можно получить только в лабораторных условиях, и только с применением химических реагентов. На полигоне при производительности хотя бы 50 кубов в час извлечь 100% невозможно, так как обязательно найдутся факторы, мешающие сбору, начиная от человеческого фактора и кончая горно-геологическими условиями. Довольно редко можно встретить на золотодобывающем предприятии в штате обогатителя, который производит определенные виды работ для контроля и подсчета потерь на обогатительном оборудовании.

На золотодобывающих предприятиях для гравитационного улавливания ТМЗ применяют отсадочные машины, центробежные концентраторы, концентрационные столы, гидроциклоны, винтовые концентраторы и т.д. Все это очень хорошее оборудование, но оно работает эффективно только в определенных условиях с определенными требованиями к сырью. Кроме того, стоимость данного оборудования высока. На стационарных обогатительных фабриках данное оборудование работает отлично, т. к. там можно создать все необходимые условия для эффективной работы по извлечению золота. Это и хорошая дезинтеграция, и сортировка, и стадии сокращения материала, равномерность подачи сырья, плотность воды и самое главное профессионализм обслуживающего персонала. На месте разработок создать условия как на обогатительной фабрике очень сложно, и не всегда эффективно и экономически невыгодно.

На многих золотодобывающих предприятиях знают про свои потери ТМЗ, но предпочитают бросать данное золото и не тратить на его улавливание деньги

и время, так как прекрасно понимают, что традиционным гравитационным способом его не уловить, а строить эстакады из множества этажей накладно. Кроме того, стоимость оборудования вычтет из бюджета предприятия немало средств. На сегодняшний день рентабельность золотодобывающих предприятий упала и не каждое предприятие сможет вложить инвестиции в долго окупаемый проект, который зависит от многих факторов, начиная с запасов месторождения и кончая политикой государства Российского. Но прежде всего у золотодобытчиков присутствует недоверие в получении большого эффекта от данного оборудования и многие используют только ручной труд. Из этого следует, что золотодобывающим предприятиям необходимо обогатительное оборудование недорогое, но эффективное, пригодное для ремонта в полевых условиях, без сложных узлов, способное к быстрому демонтажу и не требующее специализированного персонала. Соответственно, нужно найти ту золотую середину, когда можно извлечь максимум золота при минимальных затратах.

Коллектив ООО РМЗ Сталькомплект (г. Белово) 7 лет занимался поиском решения по извлечению золота с минимальными затратами, но с высоким коэффициентом улавливания. Были проведены полноценные испытания по всем существующим на сегодняшний день обогатительным оборудованием (отсадочные машины, концентрационные столы, центробежные концентраторы и т.д.).

Все усилия по данному решению вылились в создание вибропромывочного шлюза (ВПШ) или как его еще называют, качающийся шлюз. Качающийся шлюз был придуман в 19 веке и наши специалисты, по сути, занимались не его изобретением, а модернизацией и адаптацией к существующим условиям. Модернизация заключалась, прежде всего, в поиске эффективного логического алгоритма амплитуды промывочного шлюза, длины хода, надежного привода с опорами движения и самое главное в строении и форме трафаретов.

За 7 лет были проведены масштабные испытания на различных типах песков. В результате испытаний была выбрана оптимальная форма промывочного шлюза и трафаретов. Виброшлюз показал себя достойно везде, где бы его ни применяли. В результате испытаний пришлось провести много исследовательских

работ по строению и форме трафаретов. И данные работы вылились в математическое решение, учитывающее в своих формулах основные показатели сырья при работе виброшлюза с учетом определенных технических параметрах. Созданные на заводе трафареты назвали ХАПИ. Особенностью трафаретов является, прежде всего, зона разряжения в турбулентном потоке, который создается верхним спойлером и боковым планпотом, который в свою очередь создает направленное турбулентное движение пульпы для верхнего слоя потока.

При обтекании рифли трафарета специальной формы пульпа подвергается поперечной деформации, что приводит к изменению скорости, давления потока. Таким образом, около поверхности рифли трафарета создается область переменных скоростей и давлений потоков пульпы. Наличие различных по величине давлений у поверхности рифли трафарета приводит к возникновению гидродинамических сил и моментов. Распределение этих сил зависит от амплитуды, наклона виброшлюза и конфигурации рифли и трафарета, формы твердых вмещающих тел. Рифля трафарета в потоке пульпы создает довольно резкое изменение направления движения потока в обтекании трафарета, торможение потока перед ним, поджатие струек у краев и образование, непосредственно за краем верхнего спойлера, зоны разряжения и больших вихрей. Вихри заполняют всю область за рифлей трафарета и заставляют золото осесть в ячейку резинового коврика. Перед рифлей трафарета давления потока всегда больше, чем в невозмущенном потоке, а за рифлей, вследствие разряжения, давление уменьшится. Данное действие заставляет частицы золота, которые прилипли к пленке натяжения воды в верхнем слое потока и к воздушным пузырькам, находящимся в потоке, взаимодействовать механически с планпотом. Пленка разрушается, частицы золота освобождаются и оседают в ячейку коврика. Более легкие частицы за счет закона о неразрывности суй проходят дальше по шлюзу и уходят в эфель [9-10].

Виброшлюза были поставлены почти на все промприборы производства ООО РМЗ СтальКомплект, такие как Мулевка-2М, Касьма-7, Мунгай-15: 30: 50, МСК-10,30:50 (мульти-спиральный концентратор), Бирюля-30, Бирюля-50.

Также был сделан отдельный приставкой виброшлюз к промприборам ГГМ-3, ПГШ-50, Ортон-100, 150, к скруббер-бутарам различной производительностью.

Именно совокупность всех показателей виброшлюза и является успехом для извлечения ТМЗ, что прекрасно продемонстрировал промывочный виброшлюз на россыпных месторождениях, начиная с Камчатки и заканчивая Уралом и бывшими республиками СССР.

Преимущества промывочного виброшлюза.

- низкая энергоемкость.

- хорошее улавливание золота.

- при длине виброшлюза условно 3м. пульпа, двигаясь синусоидально по шлюзу преодолевает расстояние 4-5м. в зависимости от режима работы виброшлюза.

- регулировка скорости потока.

- взаимодействию с трафаретом подвергаются все слои потока пульпы.

- постель виброшлюза всегда работает, находится в движении и не дает сырью штыбоваться.

- удобство съема концентрата.

- регулировка амплитуды виброшлюза.

- регулировка угла наклона виброшлюза.

- ремонтно-пригодность.

- удобство быстрого демонтажа.

- при работе виброшлюза происходит дополнительная дезинтеграция песков.

- не требователен к переменам погоды.

- обучаемость персонала за 2 часа.

Недостатки

- много подшипников

- своя технология съема концентрата

- нужно готовить надежную площадку под раму виброшлюза.

- высота распределителя потока

Очень хорошо промывочный виброшлюз зарекомендовал себя на россыпных месторождениях, где в песках присутствует много тяжелых шлихов (магнетит, гематит, пирит, гранат и т.д.) затрудняющих улавливание золота и способствующих большому количеству потерь. Из-за постоянного движения постели виброшлюза, тяжелые шлихи не укладываются в ячейку коврика, а сдвигаются вниз шлюза потоком воды, освобождая ячейку для золота. Золото, как правило, садится в ячейку на границе срыва потока в зоне разряжения. Регулировкой амплитуды виброшлюза достигается необходимая продольная волна, которая многократно изменяет направление движения пульпы. Схожие движения есть у старателя при работе лотком, когда он делает просадку. На виброшлюзе просадка идет непрерывно, что обуславливает хорошее улавливание золота на данном оборудовании.

Регулировка амплитуды при разной производительности шлюзов устанавливается по определенному алгоритму, разработанному конструкторами ООО РМЗ СтальКомплект.

Промывочный виброшлюз – это и есть та золотая середина, которая дает золотодобывающим предприятиям необходимый экономический эффект при малых затратах. Простота и удобство промывочного виброшлюза признана многими золотодобывающими предприятиями.

Для максимизации выхода золота при промывке были использованы концентрационные столы, которые устанавливались после кассет с ковриками, на которых из породы отбиралось крупное золото, для сбора шлихового золота были применены концентрационные столы. Что при небольших дополнительных затратах на электричество и воду показало свою высокую эффективность. Однако, концентрационные столы требуют достаточно тонкую настройку, в зависимости от размера фракции, подаваемого питания, состава и прочее. Данная работа направлена на построение алгоритма настройки концентрационных столов с использованием нейросетевого подхода.

# 2 ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

Формализация понятия нейронных сетей впервые приведена в работе У. Маккалока и У. Питтса 1943 года. Авторы утверждали, что практически любую функцию можно реализовать с помощью нейронной сети. В свою очередь в 1948 году Н. Виннер вместе с соавторами издает работу, в которой говорится, что сложные биологические процессы можно описать математическими моделями. Спустя год Д. Хебб предлагает первый алгоритм обучения, основывающийся на изменении силы синаптических связей. В 60-х годах XX века Ф. Розенблатт изобрел однослойный перцептрон и продемонстрировал его способность решать задачи классификации. Период затишья в развитии нейронных сетей пришелся на 1968 – 1985 гг. С появлением высокопроизводительных персональных компьютеров стало доступным моделировать нейронные сети. Настольной книгой специалистов, интересующихся теорией нейронных сетей, стала работа Ф. Уоссермена «Нейрокомпьютерная техника».

После появления в 1982 г. работы Д. Хопфилд, интерес к нейронным сетям возрос. Хопфилд показал, что проблемы нейронов можно свести к обобщению многих моделей, разработанных в физике неупорядоченных систем. Затем в 1986 году появилась работа Уильямса, Румельхарта и Хинтона, которая ответила на важный вопрос поведении многослойных нейронных сетей во время обучения. После этого алгоритм, предложенный Хинтоном, претерпел множество изменений [9]. В 80-е годы постепенно сформировался прочный теоретический фундамент, на основе которого сегодня создается большинство сетей. Разработанная теория широко стала применяться в последние два десятилетия для решения прикладных задач. Стали появляться фирмы, занимающиеся разработкой программного обеспечения для конструирования искусственных нейронных сетей. В 90-е годы нейронные сети стали использоваться в бизнесе, где они показали колоссальную эффективность при решении многих задач - от предсказания спроса на продукцию до анализа платежеспособности клиентов банка. В 2007 в университете Торонто Джеффри Хинтон создал алгоритмы глубокого изучения

нейронных сетей. При обучении нижних слоев сети, Хинтон использовал ограниченную машину Больцмана, которая представляет стохастическую рекуррентную нейронную сеть. После обучения сети полученное приложение могло быстро решать поставленную задачу (например, поиск лиц на фотографии). Эта функция на сегодняшний день встроена во все цифровые фотоаппараты. Подобная технология используется интернет – поисковиками при классификации картинок [10]. По оценкам специалистов, в области проектирования нейронных сетей и нейрокомпьютеров ожидается технологический рост. Немало новых возможностей было открыто за последние годы, а работы в данной области становятся важным вкладом в науку, технологии, экономику. Несмотря на то, что изучение нейронного моделирования ведется уже более шестидесяти лет, нет ни одной области мозга, где процесс обработки информации был бы ясен до конца. Также нет ни одного нейрона, для которого можно было бы определить код передачи информации в виде последовательности импульсов. В настоящее время существует большое число конфигураций нейронных сетей, которые отличаются по принципам функционирования, и, следовательно, направлены на разные задачи. Будущее нейрокомпьютерных технологий будет связано с новыми открытиями в области нейронного моделирования – как только удастся разгадать тайну функционирования хотя бы одной области мозга, сразу станет ясно многое о других его областях.

### 2.1 Принципы работы нейронных сетей

Под *искусственной нейронной сетью* понимается математическая модель, а также ее программная и аппаратная реализация, построенная по принципу биологических нейронных сетей – нервных клеток живого организма. Это понятие возникло при попытке смоделировать процессы, протекающие в мозге человека [11].

Искусственная нейронная сеть представляет собой систему простых процессоров (искусственных нейронов), соединенных и взаимодействующих между собой. Каждый из процессоров сети имеет дело с сигналами, которые периодически поступают или передаются другим процессорам. Большая сеть способна

решать сложнейшие задачи в самые короткие сроки. С математической точки зрения нейронные сети представляют собой способ решения нелинейных задач оптимизации. Кибернетика использует теорию нейронных сетей в решении задач адаптивного управления, построении алгоритмов для робототехники [13]. В программировании нейронная сеть один из способов решения проблемы эффективного параллелизма. Программирование нейронных сетей подразумевает именно обучение сети, а не написание программного кода. Именно благодаря обучению сеть способна выявлять зависимости между данными (входными и выходными), обобщать, упрощать результаты, использовать знания для разбиения сложных задач на более простые.

*Искусственный интеллект* – это обширная отрасль компьютерных наук, сосредоточенная на создании машин, способных выполнять интеллектуальные задачи.

*Машинное обучение*, являясь подразделом искусственного интеллекта, предназначено для решения задачи не прямым способом, а путем поиска закономерностей в данных после обучения алгоритма на множестве примеров. Нейросети являются подразделом машинного обучения.

В программировании нейронная сеть один из способов решения проблемы эффективного параллелизма. Программирование нейронных сетей подразумевает именно обучение сети, а не написание программного кода. Именно благодаря обучению сеть способна выявлять зависимости между данными (входными и выходными), обобщать, упрощать результаты, использовать знания для разбиения сложных задач на более простые.

2.1.1 Модель одного нейрона.

Мозг человека и его нервная система состоят из нейронов, соединенных нервными волокнами. Между нейронами передаются электрические импульсы с помощью нервных волокон. Все действия, которые происходят с живым организмом, все раздражения кожи, глаз, боль, процессы мышления – есть взаимодействие между нейронами. Строение биологического нейрона представлено на рисунке 2.1 [11].

*Дендриты* принимают импульсы нейрона. *Аксон* передает импульс нейрона. *Синапсы* – образования, влияющие на силу импульса, для контакта аксона и дендрита.

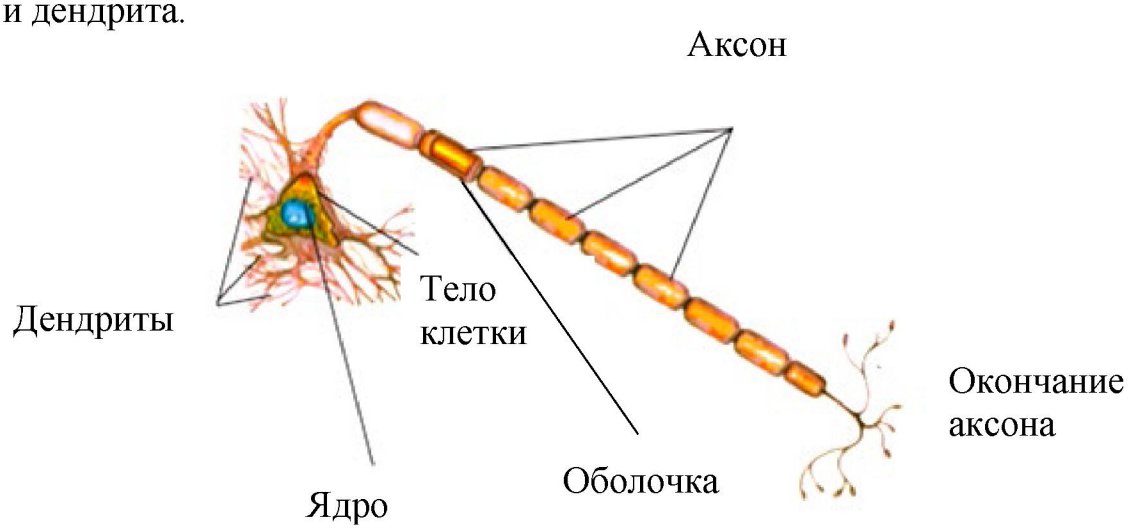



Рисунок 2.1 – Биологический нейрон [12, 13]

При прохождении синапса сила импульса меняется в определенное число раз (вес синапса). Когда к нейрону по нескольким дендритам поступают импульсы, то они суммируются. Если у суммарного импульса превышен порог, то нейрон переходит в состояние возбуждения, формирует собственный импульс и посылает его далее по аксону. Поведение соответствующего нейрона может меняться, так как веса синапсов имеют свойство меняться со временем. Математическая модель описанного процесса представлена на рисунке 2.2:

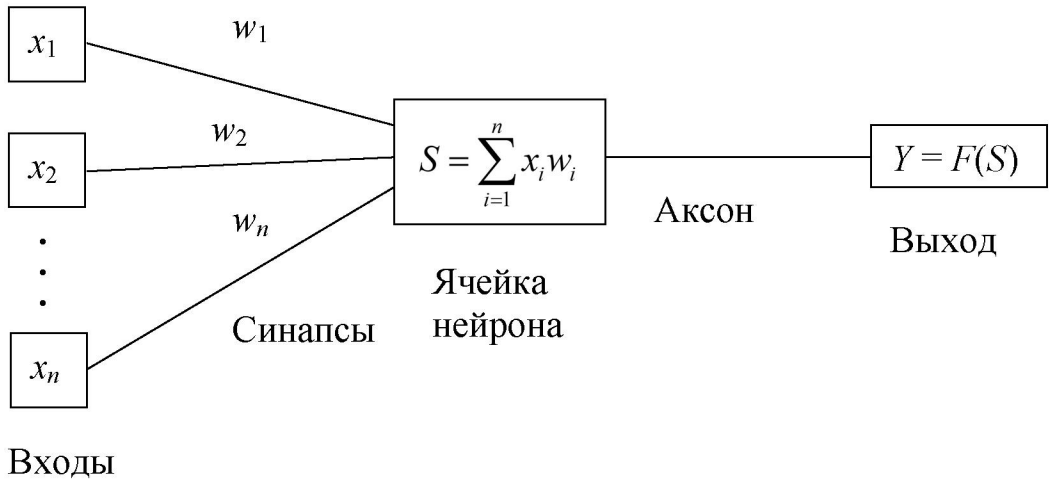


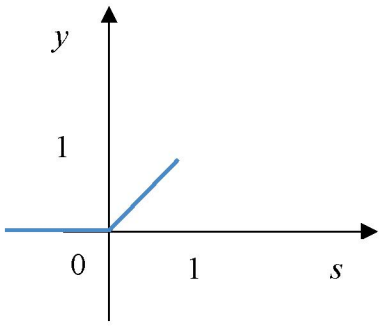

Рисунок 2.2 – Математическая модель нейрона и его активации [13]

На входной вектор $X\{x_1, x_2, x_3, ..., x_n\}$ подается некоторое значение входных сигналов от других нейронов или сенсоров. Каждый из этих входов имеет свой весовой коэффициент $w$ или мы просто умножаем входной вектор на матрицу коэффициентов $W\{w_1, w_2, ..., w_n\}$, а затем суммируем результат. Данный процесс называется взвешенным суммированием. После этого полученная сумма преобразуется в выходной сигнал с помощью выходной функции $Y = F(S)$. Получившийся сигнал преобразуется обычной линейной функцией, называемой *активационной F*.

2.1.2 Функции активации.

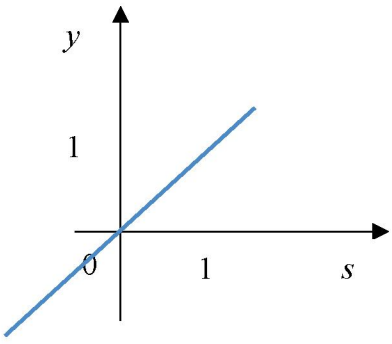
Функция активации может принимать различный вид, но есть несколько общепринятых вариантов, рисунок 2.3.

*а)*



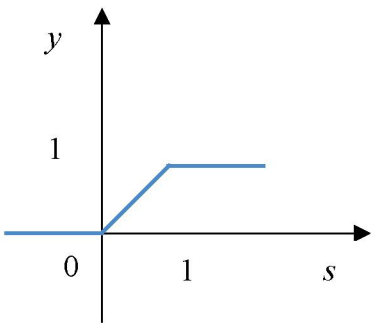

$$y = \begin{cases} 0, & s \le 0 \\ s, & s > 0 \end{cases}$$

*б)*



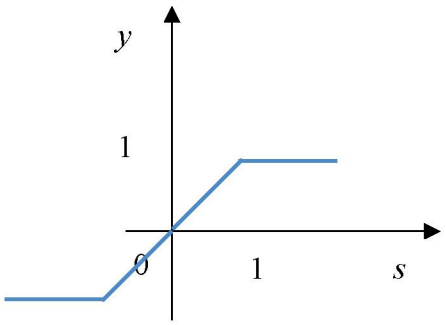

$$y = s$$

*в)*




$$y = \begin{cases} 0, & s \le 0 \\ s, & 0 < s < 1 \\ 1, & s \ge 1 \end{cases}$$

*г)*




$$y = \begin{cases} -1, & s \le -1 \\ s, & -1 < s < 1 \\ 1, & s \ge 1 \end{cases}$$

Рисунок 2.3 – Некоторые виды функции активации

Более широкий ряд функций активаций представлен в таблице 2.1.

Таблица 2.1 – Основные функции активации нейронов [13, 14]

| **Название** | **Формула** | **Область значений** |
|---|---|---|
| Линейная | $f(x) = kx$ | $(-\infty, \infty)$ |
| Полулинейная | $f(x) = \begin{cases} kx, & x > 0 \\ 0, & x \le 0 \end{cases}$ | $(0, \infty)$ |
| Сигмоид | $f(x) = \frac{1}{1 + e^{-x}}$ | $(0,1)$ |
| Гиперболический тангенс | $f(x) = \frac{e^{x} - e^{-x}}{e^{x} - e^{-x}}$ | $(-1,1)$ |
| Экспоненциальная | $f(x) = e^{-x}$ | $(0, \infty)$ |
| Квадратичная | $f(x) = x^2$ | $(0, \infty)$ |
| Знаковая | $f(x) = \begin{cases} 1, & x > 0 \\ -1, & x \le 0 \end{cases}$ | $(-1,1)$ |

## 2.2 Классификация нейронных сетей

В зависимости от критерия классификации, нейронные сети можно разделить на различные типы. Приведем классификацию искусственных нейронных сетей по следующим признакам:

1. по способу обучения;
2. по топологии;
3. по модели;
4. по способу настройки весовых коэффициентов;
5. по задачам, решаемым при помощи нейросетей.

В таблице 2.2 показана наиболее универсальная классификация по первым четырем признакам [15, 16].

При обучении с учителем предполагается, что есть внешняя среда, которая предоставляет обучающие примеры (значения входов и соответствующие им значения выходов) на этапе обучения или оценивает правильность функционирования нейронной сети и в соответствии со своими критериями меняет состояние нейронной сети или поощряет (наказывает) нейронную сеть, запуская тем самым механизм изменения ее состояния [16].

Остановимся подробнее на классификации нейронных сетей по топологии, рисунок 2.4.

Каждая нейронная сеть включает в себя первый слой нейронов, называемый входным. Этот слой не выполняет каких-либо преобразований и вычислений, его задача состоит в том, чтобы: принимать и распределять входные сигналы по остальным нейронам. И этот слой единственный, являющийся общим для всех типов нейросетей, а критерием для деления является уже дальнейшая структура.

Таблица 2.2 – Универсальная классификация нейронных сетей

| Способ обучения | Топология | Модель нейронной сети | Насойка весовых коэффициентов |
|---|---|---|---|
| – с учителем;<br>– без учителя;<br>– с подкреплением | – однослойные;<br>– многослойные;<br>– слабосвязные;<br>– полносвязные. | – прямого распросанения;<br>– рекуретные нейронные сети;<br>– радиально-базисные функции;<br>– нейронные сети Кохонена | – фиксированные коэффициенты;<br>– динамические коэффициенты |

*Однослойная структура нейронной сети.* Представляет собой структуру взаимодействия нейронов, в которой сигналы с входного слоя сразу направляются на выходной слой, который, преобразует сигнал, и сразу же выдаёт ответ. Входные нейроны являются объединёнными с основным слоем с помощью синапсов с разными весами, обеспечивающими качество связей.

*Многослойная нейронная сеть.* В сети помимо выходного и входного слоёв, имеются ещё несколько скрытых промежуточных слоёв. Число этих слоёв зависит от степени сложности нейронной сети.

В *слабосвязных нейронных сетях* нейроны располагаются в узлах прямоугольной или гексагональной решетки. Каждый нейрон связан с четырьмя в

окрестности фон Неймана, шестью в окрестности Голея или восемью в окрестности Мура своими ближайшими соседями.

*Полносвязанные нейронные сети*, в которых каждый нейрон передает свой выходной сигнал остальным нейронам, в том числе и самому себе. Все входные сигналы подаются всем нейронам. Выходными сигналами сети могут быть все или некоторые выходные сигналы нейронов после нескольких тактов функционирования сети. Примером полносвязанной сети является сеть Хопфилда, она является сетью с симметричной матрицей связей. Во время получения входных данных каждый узел является входом, в процессе обучения он становится скрытым, а затем становится выходом.

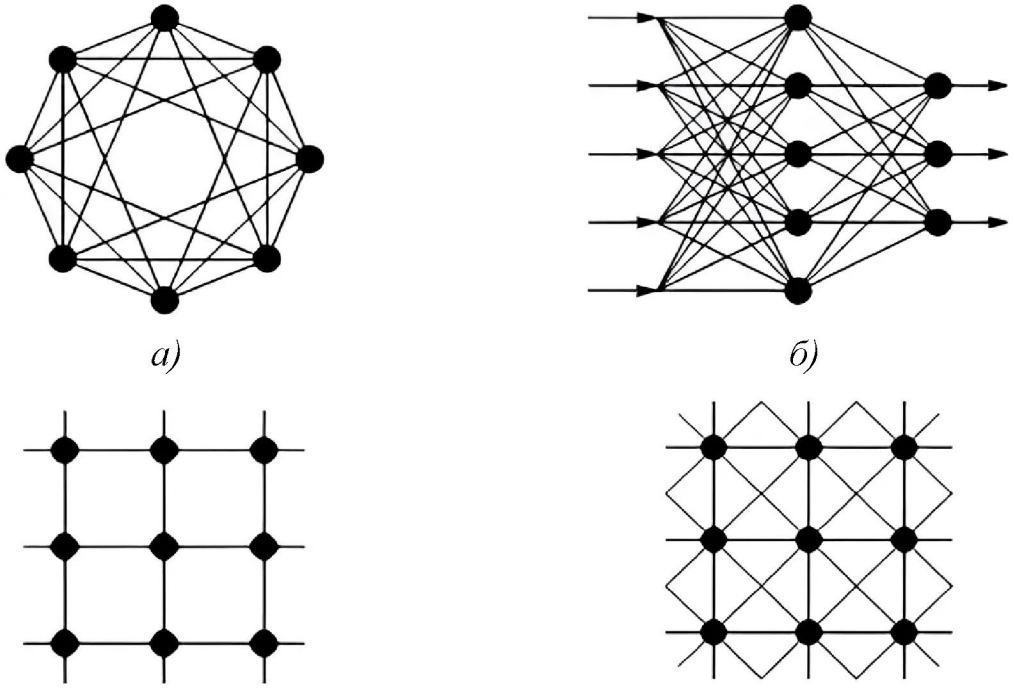



Рисунок 2.4 – Классификация нейронных сетей: полносвязные – *а*, многослойные – *б*, слабосвязные – *в*

Кроме количества слоёв, нейронные сети можно классифицировать по направлению распределения информации по синапсам между нейронами:

1. *Нейросети прямого распространения* (однонаправленные). В этой структуре сигнал перемещается строго по направлению от входного слоя к

выходному. Движение сигнала в обратном направлении не осуществляется и в принципе невозможно.

2. *Рекуррентные нейронные сети* (с обратными связями). Здесь сигнал двигается и в прямом, и в обратном направлении. В итоге результат выхода способен возвращаться на вход. Выход нейрона определяется весовыми характеристиками и входными сигналами, плюс дополняется предыдущими выходами, снова вернувшимися на вход. Этим нейронным сетям присуща функция кратковременной памяти.

Рекуррентные нейросети делятся на следующие виды:

– *слоисто-циклические*, отличающиеся тем, что слои замкнуты в кольцо: последний слой передает свои выходные сигналы первому; все слои равноправны и могут, как получать входные сигналы, так и выдавать выходные.

– *слоисто-полносвязные* состоят из слоев, каждый из которых представляет собой полносвязную сеть, а сигналы передаются как от слоя к слою, так и внутри слоя; в каждом слое цикл работы распадается на три части: прием сигналов с предыдущего слоя, обмен сигналами внутри слоя, выработка выходного сигнала и передача к следующему слою.

– *полносвязно-слоистые*, по своей структуре аналогичные слоисто-полносвязным, но функционирующим по-другому: в них не разделяются фазы обмена внутри слоя и передачи следующему, на каждом такте нейроны всех слоев принимают сигналы от нейронов, как своего слоя, так и последующих.

*Самоорганизующиеся карты*, называемые *картами Кохонена* – это одна из разновидностей нейронных сетей, однако они принципиально отличаются от рассмотренных выше, поскольку используют неконтролируемое обучение.

Классификация нейронных сетей по типу решаемой задачи представлена на рисунке 2.5.

Здесь также показана взаимосвязь между решаемой задачей и используемым типом нейронной сети.

*Перцептрон* – простейший вид нейронных сетей. В основе лежит математическая модель восприятия информации мозгом, состоящая из сенсоров, ассоциативных и реагирующих элементов.

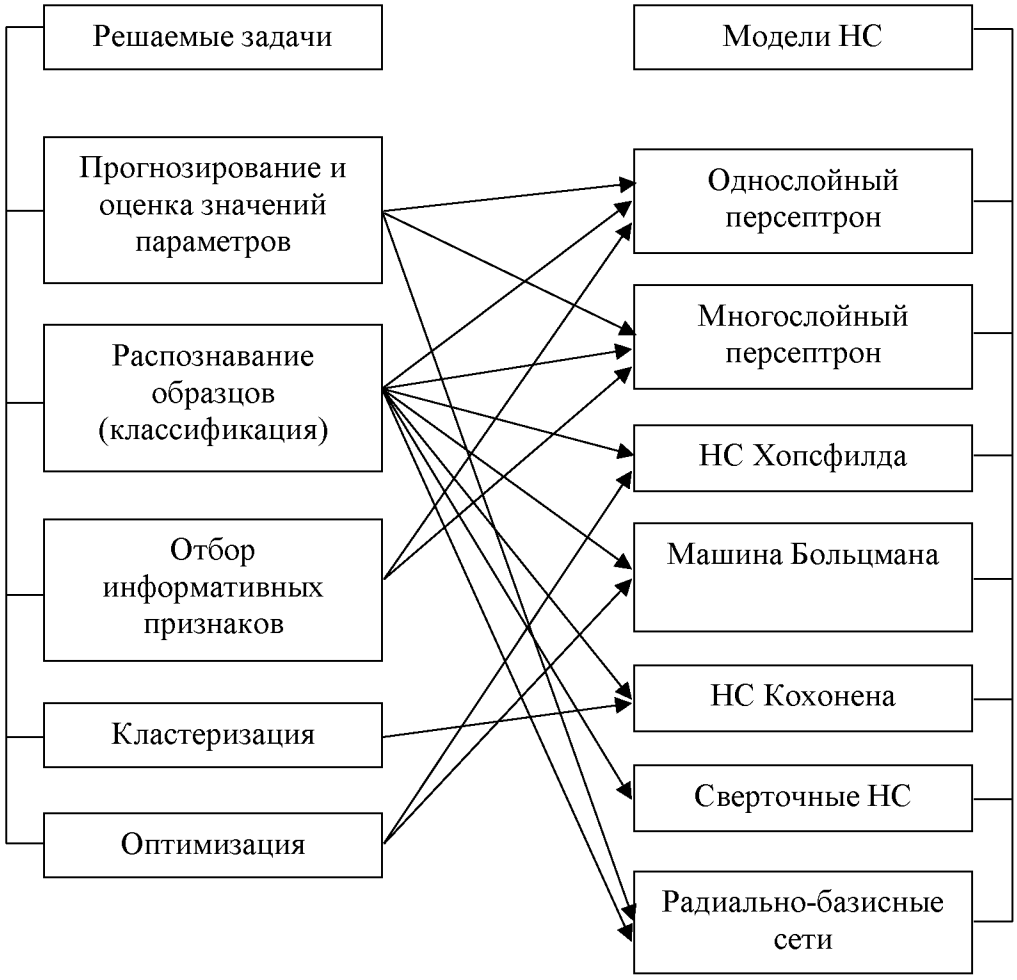



Рисунок 2.5 – Классификация нейронных сетей по типу решаемой задачи

Перцептрон является одной из первых моделей нейросетей, рисунок 2.6.

Принцип работы перцептрона:

1) Первыми в работу включаются *S*-элементы. Они могут находиться либо в состоянии покоя (сигнал равен 0), либо в состоянии возбуждения (сигнал равен 1);

2) Далее сигналы от *S*-элементов передаются *A*-элементам по так называемым *S*-*A* связям. Эти связи могут иметь веса, равные только -1, 0 или 1;

Затем сигналы от сенсорных элементов, прошедших по *S*-*A* связям, попадают в *A*-элементы, которые еще называют ассоциативными элементами;

Одному *A*-элементу может соответствовать несколько *S*-элементов. Если сигналы, поступившие на *A*-элемент, в совокупности превышают некоторый его порог θ, то этот *A*-элемент возбуждается и выдает сигнал, равный 1; В противном случае (сигнал от *S*-элементов не превысил порога А-элемента), генерируется нулевой сигнал;

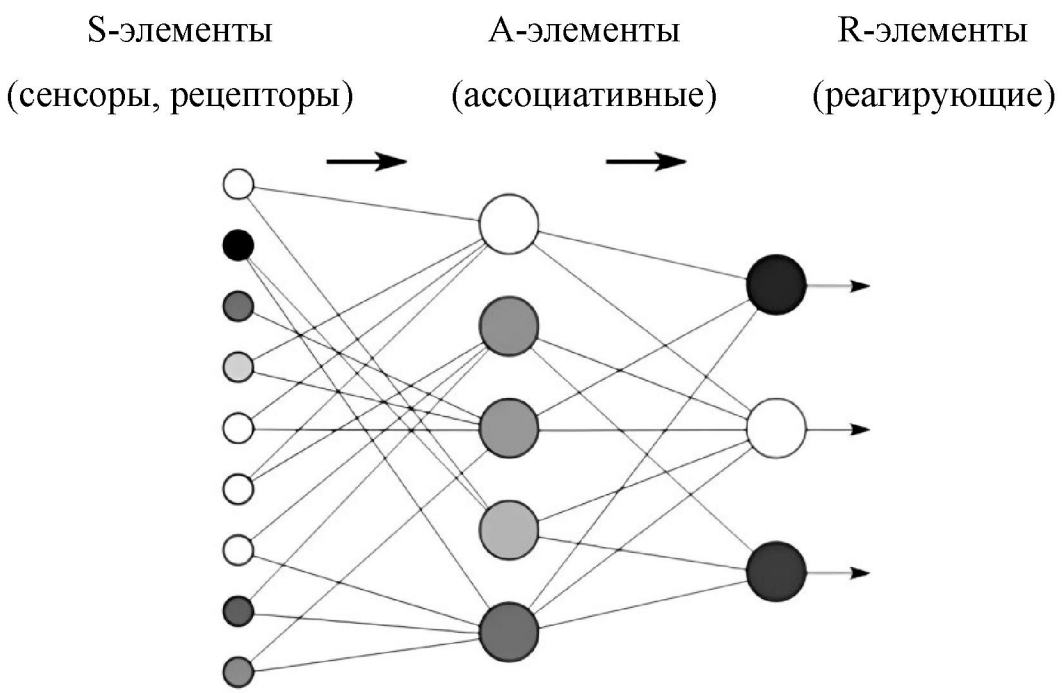



Рисунок 2.6 – Схема перцептрона

3) Далее сигналы, которые произвели возбужденные *A*-элементы, направляются к сумматору (*R*-элемент), действие которого нам уже известно. Однако, чтобы добраться до *R*-элемента, они проходят по *A*-*R* связям, у которых тоже есть веса (которые уже могут принимать любые значения, в отличие от *S*-*A* связей);

4) *R*-элемент складывает друг с другом взвешенные сигналы от *A*-элементов, а затем если превышен определенный порог, генерирует выходной сигнал, равный 1; если порог не превышен, то выход перцептрона равен -1.

Для элементов перцептрона используют следующие названия:

- *S*-элементы называют сенсорами;
- *A*-элементы называют ассоциативными;
- *R*-элементы называют реагирующими.

*Классификация перцептронов:*

*Перцептрон с одним скрытым слоем* – перцептрон, у которого имеется только по одному слою *S*, *A* и *R* элементов.

*Однослойный персептрон* – перцептрон, каждый *S*-элемент которого однозначно соответствует одному *A*-элементу, *S-A* связи всегда имеют вес 1, а порог любого *A*-элемента равен 1. Часть однослойного персептрона соответствует модели искусственного нейрона.

Его ключевая особенность состоит в том, что каждый S-элемент однозначно соответствует одному A-элементу, все S-A связи имеют вес, равный +1, а порог *A* элементов равен 1. Часть однослойного перцептрона, не содержащая входы, соответствует искусственному нейрону, как показано на картинке. Таким образом, однослойный перцептрон – искусственный нейрон, который на вход принимает только 0 и 1.

Однослойный персептрон также может быть и элементарным персептроном, у которого только по одному слою *S*, *A*, *R*-элементов.

*Многослойный перцептрон по Розенблатту* – перцептрон, который содержит более 1 слоя *A*-элементов.

*Многослойный перцептрон по Румельхарту* – частный случай многослойного персептрона по Розенблатту, с двумя особенностями:

- *S-A* связи могут иметь произвольные веса и обучаться наравне с *A-R* связями;

- Обучение производится по специальному алгоритму, который называется обучением по методу обратного распространения ошибки.

*Машина Больцмана* – вид стохастической рекуррентной нейронной сети, изобретенной Джеффри Хинтоном и Терри Сейновскив 1985 году [17]. Машина Больцмана может рассматриваться как стохастический генеративный вариант сети Хопфилда. Специалисты по статистике называют такие сети случайными марковскими полями.

Эта сеть использует для обучения алгоритм имитации отжига и оказалась первой нейронной сетью, способной обучаться внутренним представлениям, решать сложные комбинаторные задачи. Несмотря на это, из-за ряда проблем, машины Больцмана с неограниченной связностью не могут использоваться для

решения практических проблем. Если же связность ограничена, то обучение может быть достаточно эффективным для использования на практике. В частности, из каскада ограниченных машин Больцмана строится так называемая глубокая сеть доверия.

*Нейронная сеть Хопфилда* – полносвязная нейронная сеть с симметричной матрицей связей. В процессе работы динамика таких сетей сходится (конвергирует) к одному из положений равновесия. Эти положения равновесия определяются заранее в процессе обучения, они являются локальными минимумами функционала, называемого энергией сети (в простейшем случае – локальными минимумами отрицательно определённой квадратичной формы на n-мерном кубе). Такая сеть может быть использована как автоассоциативная память, как фильтр, а также для решения некоторых задач оптимизации. В отличие от многих нейронных сетей, работающих до получения ответа через определённое количество тактов, сети Хопфилда работают до достижения равновесия, когда следующее состояние сети в точности равно предыдущему: начальное состояние является входным образом, а при равновесии получают выходной образ [17]. Ее вариацией является Нейронная сеть Хэмминга.

## 2.3 Обучение нейронных сетей. Общие проблемы при обучении нейронной сети

В теории нейронных сетей термин «обучение» был определён Дж. Менделем и Р. Маклареном как процесс, в котором параметры сети настраиваются посредством воздействий, оказываемых средой, в которой находится эта сеть [16]. Способ настройки определяется типом обучения. На практике передаточные функции нейронов в многослойной сети фиксированы, поэтому настройка параметров сети сводится к итеративному процессу корректировки синаптических весов, состоящему из следующих этапов.

1. В нейронную сеть поступают обучающие данные.

2. Сеть на их основе формирует выходные сигналы.

3. Происходит оценка выходных сигналов (в различных алгоритмах обучения оценивание проводится разными методами).

4. По результатам оценки перенастраиваются синаптические веса нейронной сети, после чего сеть формирует выходные сигналы иным образом.

Обучение прекращается, когда, оценив выходные сигналы сети после очередной итерации, можно сказать, что сеть решает поставленную перед ней прикладную задачу максимально приближенно к наилучшему решению. Таким образом в процессе обучения нейронная сеть автоматически приобретает новые знания, однако следует учитывать, что из обученной сети невозможно извлечь правила её работы, поэтому обученная сеть представляет собой «чёрный ящик» и её тестирование производится в соответствии с этим, рисунок 2.5.

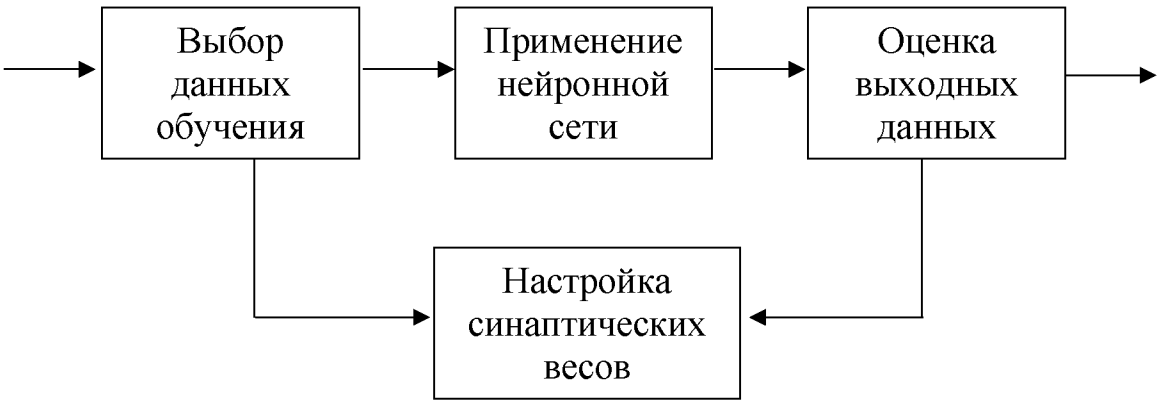



Рисунок 2.5 – Процесс обучения нейронной сети [17]

Говоря о проблемах обучения нейронных сетей, нужно обратиться к понятию градиента в нейронных сетях – вектор частных производных функции потерь по весам нейронной сети. Он указывает на направление наибольшего роста этой функции для всех весов по совокупности. Градиент считается в процессе тренировки нейронной сети и используется в оптимизаторе весов для улучшения качества модели.

В процессе обратного распространения ошибки при прохождении через слои нейронной сети в элементах градиента могут накапливаться большие значения, что будет приводить к сильным изменениям весов. Это в свою очередь может сделать нестабильным алгоритм обучения нейронной сети. В таком случае элементы градиента могут переполнить тип данных, в котором они хранятся.

Существует аналогичная обратная проблема, когда в процессе обучения при обратном распространении ошибки через слои нейронной сети градиент становится все меньше. Это приводит к тому, что веса при обновлении изменяются на слишком малые значения, и обучение проходит неэффективно или останавливается, то есть алгоритм обучения не сходится. Это явление называется затухающим градиентом (англ. vanishing gradient).

Таким образом, увеличение числа слоев нейронной сети с одной стороны увеличивает ее способности к обучению и расширяет ее возможности, но с другой стороны может порождать данную проблему. Поэтому для решения сложных задач с помощью нейронных сетей необходимо уметь определять и устранять ее.

Такая проблема может возникнуть при использовании нейронных сетях классической функцией активации $f(x) = \frac{1}{1+e^{-x}}$. Эта функция часто используется, поскольку множество ее возможных значений – отрезок [0,1] – совпадает с возможными значениями вероятностной меры, что делает более удобным ее предсказание. Также график сигмоиды соответствует многим естественным процессам, показывающим рост с малых значений, который ускоряется с течением времени, и достигающим своего предела (например, рост популяции).

В таком случае оценка элементов градиента растет экспоненциально при рассмотрении частных производных по весам слоев в направлении входа в нейронную сеть (уменьшения номера слоя). Это в свою очередь может приводить либо к экспоненциальному росту градиента от слоя к слою, когда входные значения нейронов – числа, по модулю большие 1, либо к затуханию, когда эти значения – числа, по модулю меньшие 1.

Однако, входные значения скрытых слоев есть выходные значения функций активаций предшествующих им слоев. В частности, сигмоида насыщается при стремлении аргумента к $+\infty$ или $-\infty$, то есть имеет там конечный предел. Это приводит к тому, что более отдаленные слои обучаются медленнее, так как увеличение или уменьшение аргумента насыщенной функции вносит малые

изменения, и градиент становится все меньше. Это и есть проблема затухающего градиента.

*Способы определения:*

1) Взрывающийся градиент

Возникновение проблемы взрывающегося градиента можно определить по следующим признакам:

- Модель плохо обучается на данных, что отражается в высоком значении функции потерь.

- Модель нестабильна, что отражается в значительных скачках значения функции потерь.

- Значение функции потерь принимает значение NaN.

Более непрозрачные признаки, которые могут подтвердить возникновение проблемы:

- Веса модели растут экспоненциально.

- Веса модели принимают значение NaN.

2) Затухающий градиент

Признаки проблемы затухающего градиента:

- Точность модели растет медленно, при этом возможно раннее срабатывание критерия останова, так как алгоритм может решить, что дальнейшее обучение не будет оказывать существенного влияния.

- Градиент ближе к концу показывает более сильные изменения, в то время как градиент ближе к началу почти не показывает никакие изменения.

- Веса модели уменьшаются экспоненциально во время обучения.

- Веса модели стремятся к 0 во время обучения.

Способы устранения:

1)Использование другой функции активации

2) Изменение модели

Для решения проблемы может оказаться достаточным сокращение числа слоев. Это связано с тем, что частные производные по весам растут экспоненциально в зависимости от глубины слоя.

В рекуррентных нейронных сетях можно воспользоваться техникой обрезания обратного распространения ошибки по времени, которая заключается в обновлении весов с определенной периодичностью.

3) Регуляризация весов

Регуляризация заключается в том, что слишком большие значения весов будут увеличивать функцию потерь. Таким образом, в процессе обучения нейронная сеть помимо оптимизации ответа будет также минимизировать веса, не позволяя им становиться слишком большими.

4) Обрезание градиента

Обрезание заключается в ограничении нормы градиента. То есть если норма градиента превышает заранее выбранную величину.

## 2.4 Основные сведения о нейронных сетях TensorFlow, OpenNMT-py и пакете Maple

Maple – программный пакет, система компьютерной алгебры (точнее, система компьютерной математики). Является продуктом компании Watcom Products Inc, которая с 1982 года выпускает программные продукты, ориентированные на сложные математические вычисления, визуализацию процессов и моделирование систем. Система Maple предназначена для символьных вычислений, хотя имеет ряд средств и для численного решения дифференциальных уравнений и нахождения интегралов. Обладает развитыми теоретически механическими средствами. Имеет собственный интерпретируемый язык программирования, синтаксисом частично напоминающий Паклин [19].

OpenNMT (Open Neural Machine Translation) – открытая система машинного перевода, использующая методы машинного обучения. Для построения нейронной сети проект использует возможности библиотеки глубинного машинного обучения Torch. Код, развиваемый проектом OpenNMT модулей для Torch написан на языке Lua и распространяется под лицензией MIT. Для упрощения распространения продукта проектом также развивается самодостаточный вариант транслятора на языке C++.

TensorFlow – открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронной сети с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия. Применяется как для исследований, так и для разработки собственных продуктов Google. Основной API для работы с библиотекой реализован для Python, также существуют реализации для R, C#, C++, Haskell, Java, Go и Swift. Является продолжением закрытого проекта DistBelief. Изначально TensorFlow была разработана командой Google Brain для внутреннего использования в Google, в 2015 году система была переведена в свободный доступ с открытой лицензией Apache 2.0 [21].

# 3 ПРИЛОЖЕНИЕ НЕЙРОННЫХ СЕТЕЙ К ОПТИМИЗАЦИИ РАБОТЫ КОНЦЕНТРАЦИОННЫХ СТОЛОВ

Для разработки и верификации эволюционного алгоритма построения нейронной сети необходима достаточна большая база данных для обучения. В качестве таких данных было принято решение использовать аналитические решения для обыкновенных дифференциальных уравнений первого и второго порядков. Также было проведено сравнение полученных результатов с уже существующими нейронными сетями OpenNMT и Tensorflow. Для создания обучающей базы данных использовался пакет Maple. Полученные результаты помогли прийти к выводу о целесообразности разработки своей нейронной сети.

## 3.1 Нейронные сети OpenNMT и Tensorflow

1. Создание базы с аналитическим решением обыкновенных дифференциальных уравнений для обучения нейронной сети было сделано с помощью пакета Maple. На рисунках 3.1 и 3.2 представлены примеры получения общего решения ОДУ первого и второго порядков в пакете Maple.

С помощью пакета Maple было сгенерировано 150 000 уравнений с решениями для обучения нейронной сети и 3000 уравнений для проверки результатов после обучения нейронной сети.

Для автоматизации процесса генерации дифференциальных уравнений было разработано программное обеспечение, которое создавало ОДУ первого порядка и ОДУ второго порядка из заранее определенных функций в случайной рекомбинации с условием уникальности каждого нового сгенерированного уравнения. В свою очередь для получения потоковых решений без использования графического интерфейса Maple и без необходимости участия оператора в данном процессе было разработано программное обеспечение на языке C#, которое посредством API Maple получало решение и записывало его в базу данных.

2. Проведем сравнительный анализ качества полученных решений с помощью рекурентной нейронной сети [22-23] OpenNMT и классического многослойного перцептрона Tensorflow.

2.1 OpenNMT

Подготавливаем полученную базу уравнений и их решений для использования в обучении рекурентной нейронной сети OpenNMT, как представлено на рисунке 3.1 и рисунке 3.2.

Для нашей задачи в OpenNMT была сконфигурирована шестислойная рекуррентная нейронная сеть, содержащая по 512 нейрона на слой.

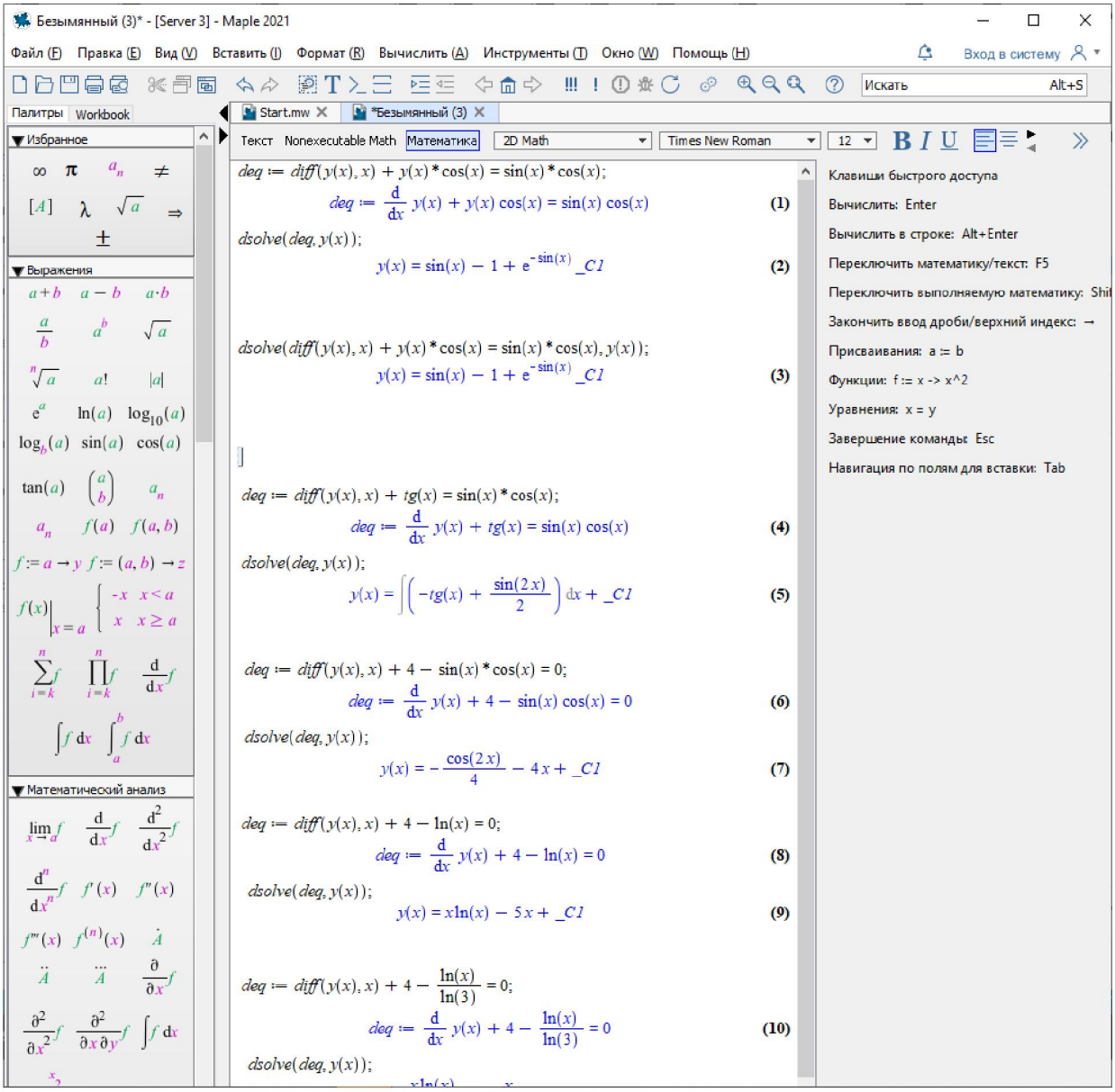



Рисунок 3.1 – Примеры решения ОДУ первого порядка в Maple

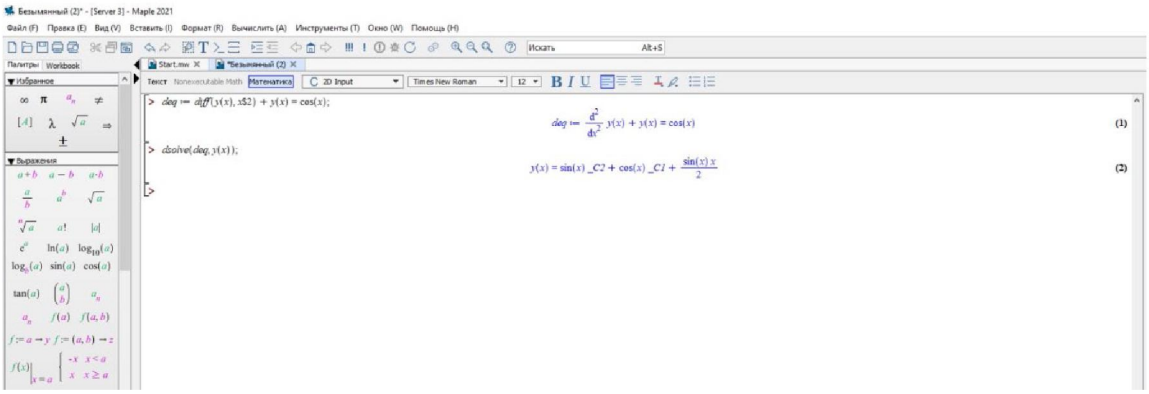



Рисунок 3.2 – Примеры решения ОДУ второго порядка в Maple

Предварительно была проведена токенизация базы дифференциальных уравнений и их решений. Здесь под токенизацией понимаем отделение каждой функции и спецсимвола пробелами. Использовался скрипт от разработчиков OpenNMT (https://github.com/ManiaCello/en-ru-onmt/tree/master/tools).

Пример токенизации:

из diff(y(x), x) + y(x)*cos(x) = sin(x)*cos(x) <-> y(x) = sin(x)-1+exp(-sin(x))*_C1

получаем y(x)' + y ( x ) * cos ( x ) = sin ( x ) * cos ( x )<->y(x) = sin ( x ) -1 + exp( -1 * sin ( x ) ) * _C1.

Параметры запуска обучения сгенерированной нейронной сети:

layers: 6 rnn_size: 512 word_vec_size: 512 normalization: tokens

Пример решения ОДУ первого порядка с помощью OpenNMT представлен на рисунке 3.3 и для ОДУ 2 порядка на рисунке 3.4.

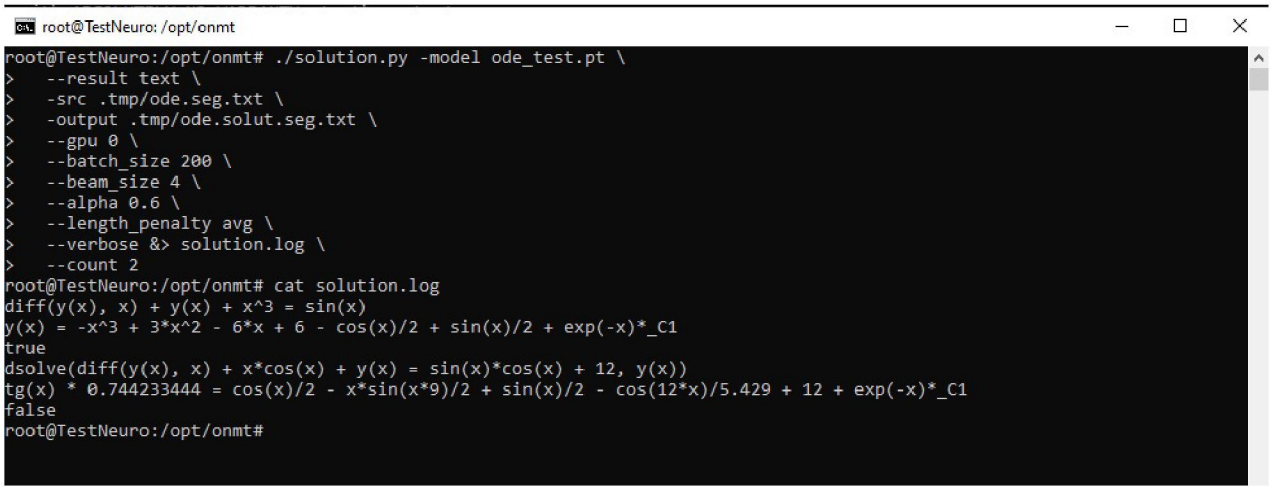



Рисунок 3.3 – Пример решения ОДУ первого порядка с помощью OpenNMT

После обучения на выборке из 150000 уравнений для обучения и 3000 уравнений для проверки нейронной сети показала результат точности в 47% для ОДУ первого порядка и 42% для ОДУ второго порядка.

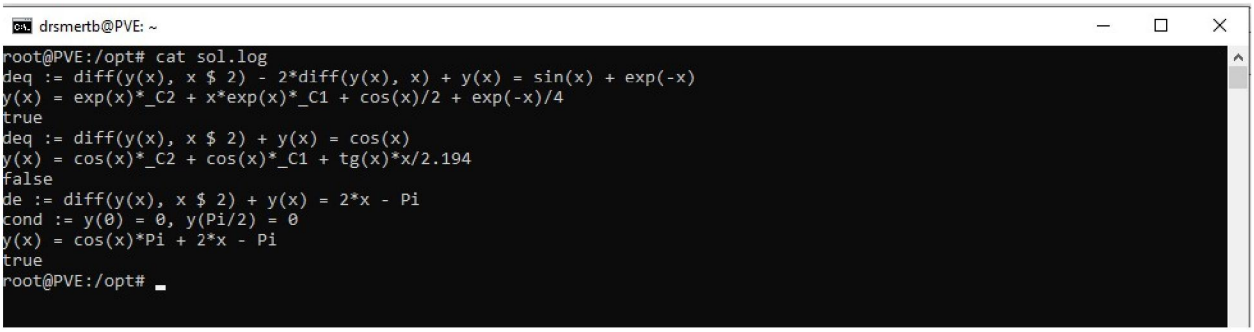



Рисунок 3.4 – Пример решения ОДУ второго порядка с помощью OpenNMT

Обучение проводилось на протяжении 25 эпох.

На 6-ти видеокартах (3*nVidia 1060 и 3* nVidia 3060).

Обучение заняло 72 и 74 часа, соответственно.

2.2 Tensorflow

Аналогичным образом, как и для OpenNMT, проведем подготовку обучающей выборки.

С изменением действий на этапе токенизации (требуется провести индексацию полученных токенов.

Пример:

y(x)' + y ( x ) * cos ( x ) = sin ( x ) * cos ( x )<->y(x) = sin ( x ) -1 + exp( -1 * sin ( x ) ) * _C1

после индексации стала иметь вид:

1 17 3 7 2 11 30 6 7 2 11 29 9 7 2 11 30 6 7 2 11

<->

3 7 2 11 29 9 7 2 11 18 221 17 32 7 18 221 30 9 7 2 11 11 30 222

В Tensorflow был сконфигурирован трёхслойный перцептрон, каждый слой которой содержал 1024 нейрона.

Параметры обучения нейронной сети:

n_input = 1024

n_hiden=1024

n_output=1024

Обучение проводилось до тех пор, пока погрешность решения между эпохами не составила $10^{-4}$.

После 32 эпох была получена погрешность 18% для ОДУ первого порядка и 25% для ОДУ второго порядка.

Обучение заняло 50 и 63 часа, соответственно.

Пример решения ОДУ первого порядка с помощью Tensorflow представлен на рисунке 3.5.

В процессе работы с нейросетью, было проведено обучение ещё на 150000 дифференциальных уравнениях для ОДУ первого порядка и 135000 для ОДУ второго порядка. На 300000 проверочных примерах нейронная сеть показала точность в 93% для ОДУ первого порядка и 87% для ОДУ второго порядка, что на 5,8% и 12% улучшило наши предыдущие результаты. Но все последующие попытки получить более хороший результат не дали положительного результат, т.к. последующее обучение сети только негативно сказывалось на результате. Для компенсации этой проблемы было принято решение менять структуру нейронной сети, т.к. смена гиперпараметров сети, увеличение количества материала для обучения и смена генератора псевдослучайных чисел (использовался для начального задания весов сети) не давали должного результата. Был проведен ряд экспериментов: добавлялись внутренние полносвязные слои, слои свёртки, количество нейронов входных и выходных слоёв, что давало только временный результат, больше похожий на погрешность, но достигнутый вариант улучшить не удалось.

В связи с этим было принято решение о разработки алгоритма автоматизированного изменения структуры сети на основе удачного\неудачного результата [25-28].

Таким образом, задача сводится к реализации подобия эволюционного процесса с генерацией и закреплением признака (в нашем случае новых слоёв, нейронов или связей). В случае улучшения результата – обучение на 100000

примерах обучающей выборки с последующей проверкой на 25000 тестовых задач. В противном случае – откат к первоначальной структуре [29-30].

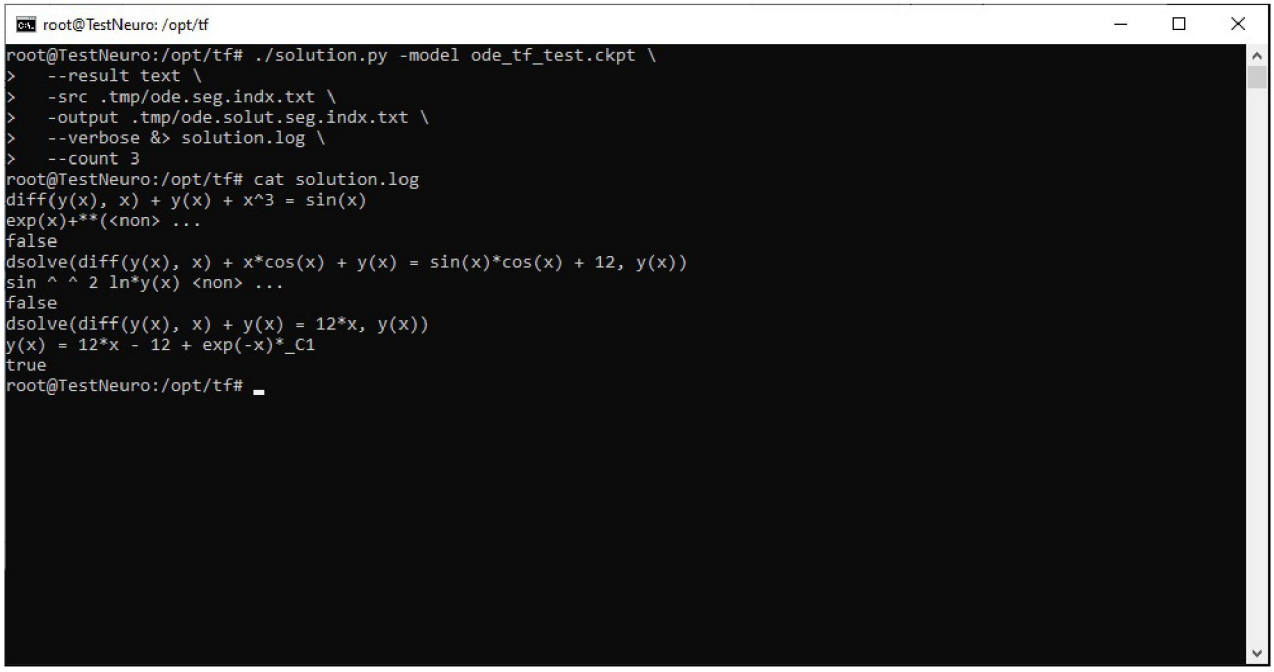



Рисунок 3.5 – Пример решения ОДУ первого порядка с помощью Tensorflow

### 3.2 Модификация структуры нейронной сети. Эволюционная генерация

Алгоритм выглядит следующим образом:

1. Обучаем «пустую» нейронную сеть на 100000 примерах для каждого вида уравнений обучающей выборки и делаем проверку на 25000 тестовой выборки и сохраняем результат обучения, а также сохраняем веса сети для отката в случае неудачи.

2. Генерируем 20 новых нейросетей на основе нейросети, полученной в п.1. В каждой нейросети алгоритм может с равной вероятностью внести до 3 изменений. Изменения могут выражаться в добавление элемента сети (нейрона, связи, слоя), либо в удаление элемента сети.

3. Проводим обучения новых 20 сетей на тех же условиях что и в п.1, но на новых данных.

4. Полученные результаты оцениваются, выбирается три наиболее «преуспевших» сети. Под «преуспевшими» будем понимать сети, показавшие

результат лучше, чем сеть, являющаяся «родителем» в плане точности и при этом не затратившая на обучение время больше 105% от родительской, либо сеть, показавшая результат хуже не более чем на 0,5% и время на 10% лучше, чем родительская сеть. Аналогичным образом сравниваем «сестринские» сети.

5. В итоге остаётся две наиболее преуспевших сети, которые смогут дать дочерние сети и повторить цикл.

6. Цикл продолжается пока одна из получившихся сетей не даст точность в 97%.

Для демонстрации процесса воспользуемся нейронной сетью на 5 полносвязных слоях, где каждый слой содержит по 15 нейронов, рисунок 3.6.

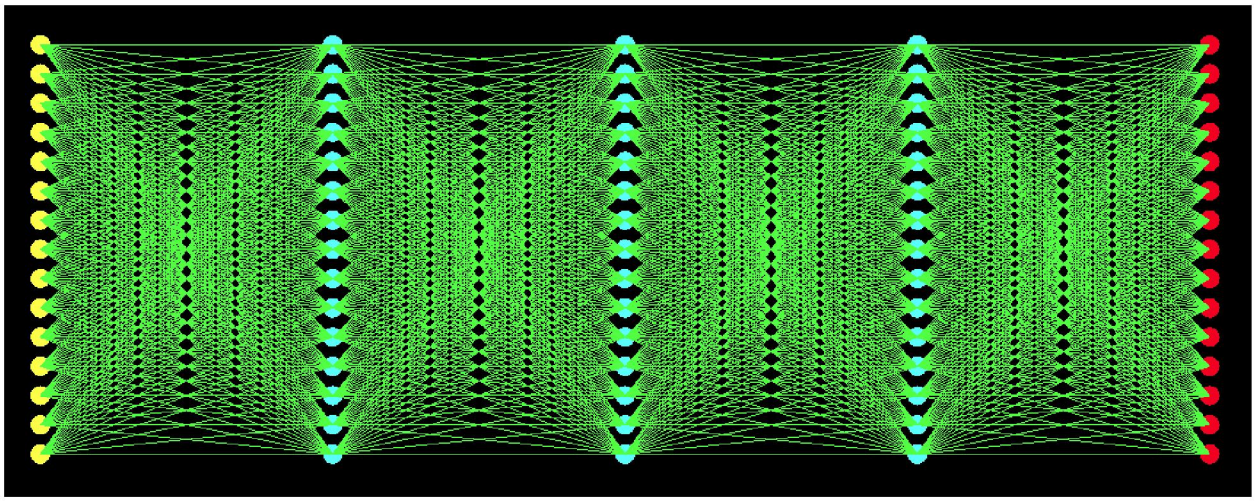


Рисунок 3.6 – Пример полученной нейронной сети с пятью слоями

На рисунках 3.7 – 3.9 продемонстрированы лучшие представители первого, пятого и сотого поколений:

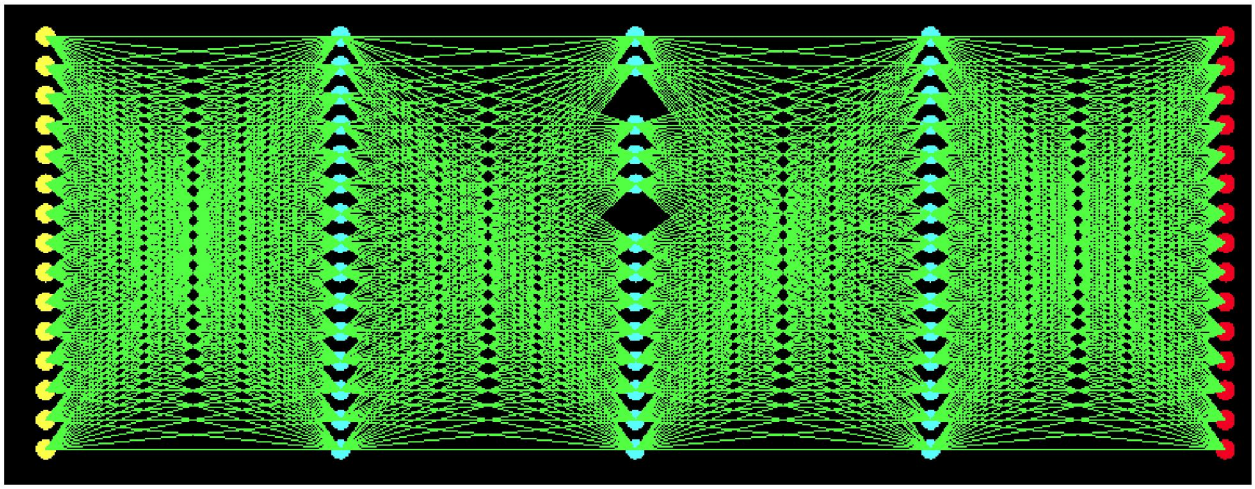


Рисунок 3.7 – Первое поколение нейронной сети с пятью слоями

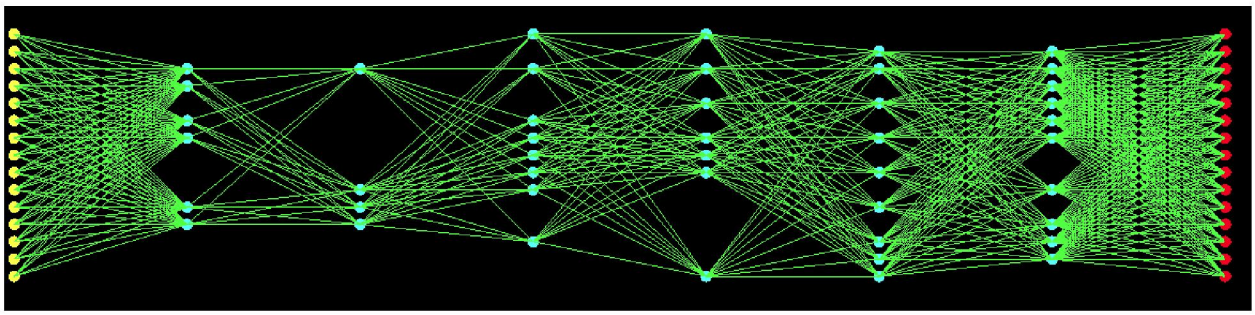


Рисунок 3.8 – Пятое поколение нейронной сети с пятью слоями

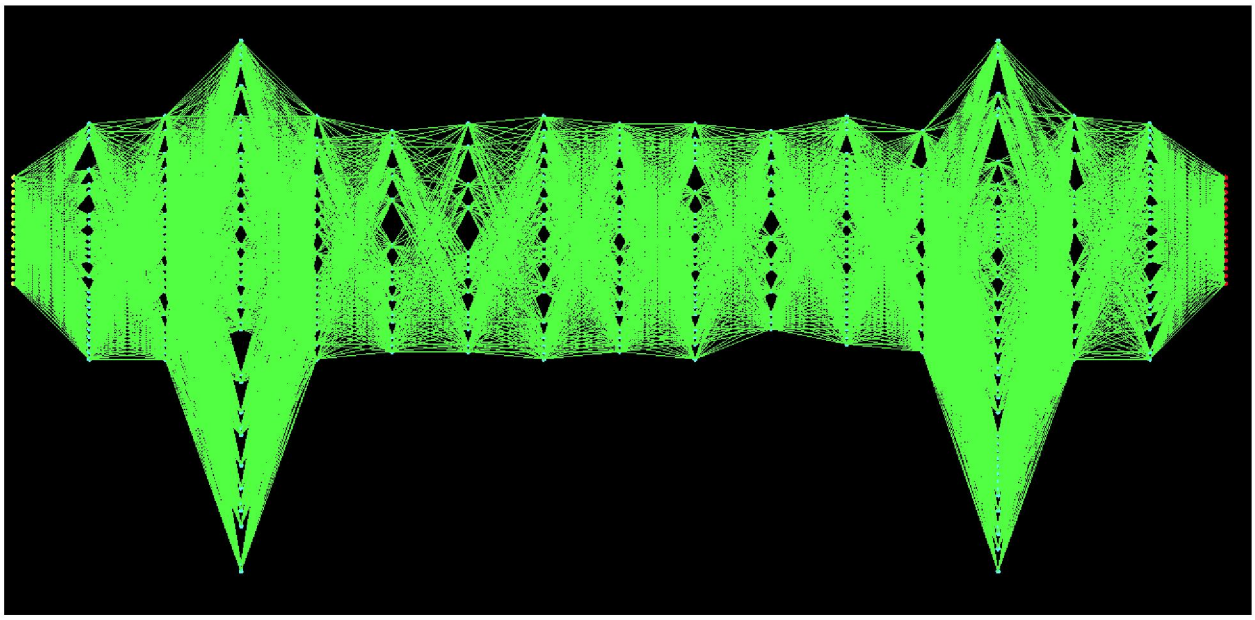


Рисунок 3.9 – Сотое поколение нейронной сети с пятью слоями

Таким образом, создан и реализован алгоритм по автоматической генерации нейронной сети. Это позволяет разработать нейронную сеть, максимально адаптированную к решению поставленной задачи.

### 3.3 Эволюционный алгоритм нейронной сети в приложении к поиску оптимальных параметров режима работы концентрационного стола

3.3.1 Структура концентрационного стола.

Стол для встряхивания, также называемый концентрационным столом, является одним из наиболее широко используемых гравитационных оборудований. Точность концентрирования и коэффициент обогащения концентрата высоки. Во время работы концентрационных столов минералы отделяются как движением поверхности стола, так и промывкой

горизонтального потока воды. Чтобы повысить эффективность концентрации, операторы должны понимать навыки работы и места обслуживания для более точной настройки процесса.

Во время работы концентрационных столов операторы должны наблюдать за перегородкой поверхности кровати концентрационных столов и тщательно контролировать продольный и поперечный наклон подходящей поверхности кровати. Кроме того, также необходимо учитывать такие факторы, как количество питания, концентрация питания, объем промывочной воды, ход и вторичный импульс в процессе сортировки. Схема концентрационного стола представлена на рисунке 3.10.

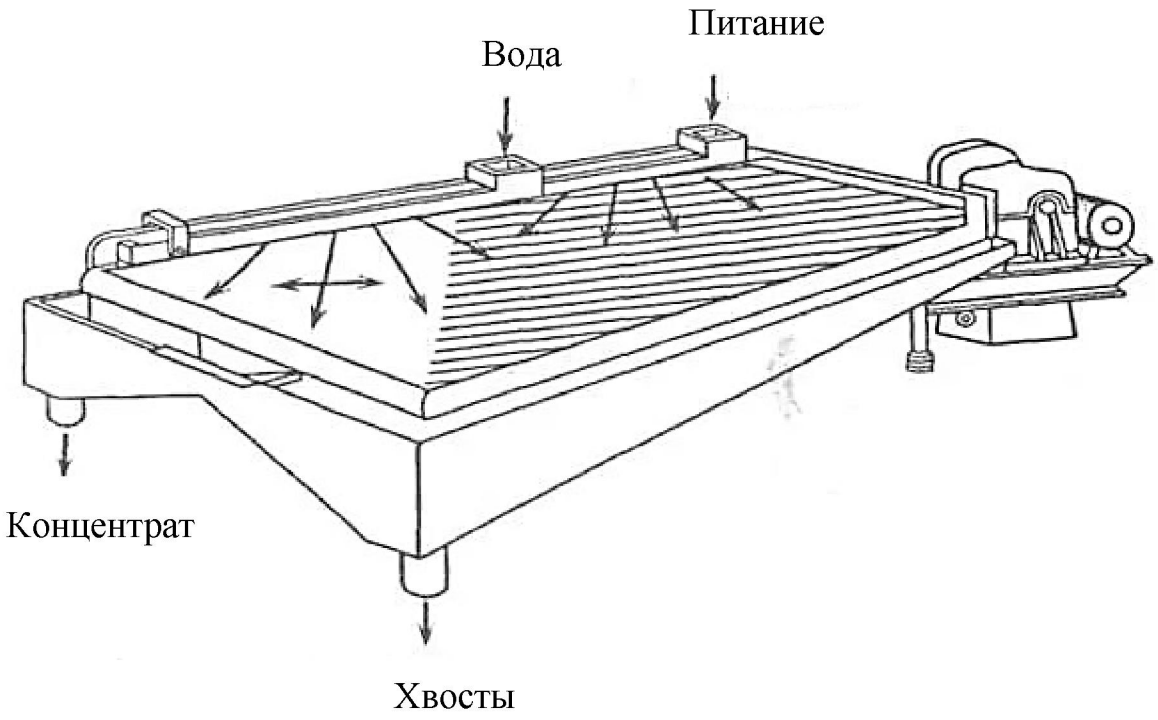



Рисунок 3.10 – Схема концентрационного стола [5]

В качестве входных и выходных параметров нейронной сети будут использоваться следующие показатели:

1. Ширина зон концентрации материала, мм:

- ширина зоны концентрации;

- ширина средней зоны:

- ширина зоны хвостохранилища;

- ширина зоны слизи.

2. Угол наклона стола, °;

3. Количество питания концентрационного стола, $м^3$;

4. Концентрация питания концентрационного стола, $м^3$;

5. Количество промывочной воды, $м^3$;

6. Длина хода концентрационного стола, мм;

7. Частота колебаний, 1/с;

8. Выход золота, г/ч;

9. Содержание золота в породе $г/м^3$.

Рассмотри более подробно каждый из параметров.

1. Поверхность перегородки

Поверхность ложа делится на зону концентрации, среднюю зону, зону хвостохранилища и зону слизи. В обычных условиях ширина слизи составляет 0,9 – 1,4 м. Горная измельченная порода подается так, чтобы не порождать пороги или явление рва. Толщина шва должна быть соответствующей и покрыта водой. Хвосты можно контролировать, регулируя концентрацию питания и песчаное отверстие канала подачи. В районе добычи выделяется средняя полоса, основной контроль, регулируя напор воды и поперечный уклон. Область выбора требует четкого зонирования различных минералов с удельным весом, образующих устойчивую и четкую границу между областью выбора и областью первичной концентрации, которая контролируется регулировкой объемов промывочной воды и бокового наклона.

2. Наклон концентрационных столов

При монтаже оборудования определяется продольный уклон. Область грубого песка имеет продольный наклон 1° – 2°, область мелкого песка имеет угол наклона равный 0,5° – 1,0°, а область грязи вообще не имеет наклона. Поверхность ложа наклонена в поперечном направлении от конца подачи к противоположной стороне при 1,5° – 5°. В реальной работе шейкера необходимо точно контролировать продольные и поперечные уклоны поверхности кровати. Как правило, если минеральные частицы являются крупными, горизонтальный

наклон поверхности слоя шейкера должен быть соответственно увеличен. Наоборот, если минеральные частицы более мелкие, необходимо уменьшить горизонтальный наклон поверхности слоя. Как правило, горизонтальные диапазоны уклона крупнозернистой песчаной зоны, мелкозернистой песчаной зоны и площади грязи составляют 2,5° – 4,5°, 1,5° – 3,5° и 1° – 2°, соответственно.

3. Количество питания концентрационных столов

Размер подачи руды связан с размером исходного зерна. Если зерна руды толще, требуется больше руды. Однако, если количество руды слишком велико, то это приведет к проблемам зонирования, и в этом случае необходимо переместить пластину перехватчика концентрата, увеличить промывочную воду и горизонтальный наклон поверхности слоя шейкера. Важно отметить, что как только правильное количество руды найдено, процесс подачи руды должен быть непрерывным и равномерным.

4. Концентрация кормления концентрационных столов

Во время процесса сортировки должна быть обеспечена соответствующая концентрация руды. В общем, концентрация грубых и мелких минералов составляет 20% – 30% и 15% – 25%, соответственно. При встряхивании проблема стягивания поверхности кровати стола может быть отрегулирована для увеличения концентрации корма; с другой стороны, если есть куча песка, то концентрацию руды необходимо уменьшить. Концентрацию корма можно контролировать, регулируя количество питающей воды.

5. Промывка водой концентрационных столов

Промывочная вода включает питательную взвесь и промывочную воду. Во время работы концентрационных столов, когда зона концентрата сужается и концентрат попадает в среднюю руду, количество промывочной воды необходимо уменьшить. И наоборот, если на поверхности слоя шейкера нет открытой пленки, необходимо увеличить подаваемое количество воды. При управлении объемом промывки следует тщательно следить за скоростью пульпы и шириной зоны концентрации.

6. Длина и частота хода концентрационных столов

На длину и частоту хода в основном влияют размер частиц минералов, нагрузка на пласт и плотность руды. При выборе материалов с крупным зерном и толстым слоем следует использовать большой ход и маленький ход. В противном случае вам нужно увеличить амплитуду и частоту колебаний. При возникновении неисправности необходимо проверить, подходит ли ход или нет.

7. Результативный выход готовой продукции (шлихового золота)

На результативность выхода золота влияют, как правильно подобранные параметры из предыдущих пунктов, так и содержание золота в золотоносной породе. Содержание определяется на участке путём химического травления и восстановления металлического золота из выборки после первичной обработки из выхода «колоды». Пробы берутся лаборантом и обрабатываются непосредственно на участке.

3.3.2 Настройка работы концентрационного стола.

Установить правильный режим работы концентрационного стола можно только опытным путем, принимая во внимание состав исходного сырья, а также требования, которые предъявляются к готовому продукту. Самое важное – задать правильный режим рыхления состава, корректируя подачу воды и возвратно-поступательные движения рабочей поверхности, а также обеспечить по возможности предварительную классификацию по крупности исходного питания. Оператор может самостоятельно менять как поперечный, так и продольный наклон, длину и частоту хода. Следует понимать, что технологические характеристики рабочего процесса напрямую зависят от частоты и амплитуды колебаний, расхода воды.

Выбор архитектуры и количества слоев для нейронной сети зависит от конкретной задачи, типа данных, доступных ресурсов и других факторов. Это процесс, который требует проведения некоторых экспериментов и оптимизации.

Для решения данной проблемы был разработан эволюционный алгоритм разработки структуры нейронных сетей, основанный на внедрение мутаций в

потомках и отборе наиболее оптимальной структуры нейронной сети по точности и временных затрат.

Для оперативной настройки параметров работы столов был применён эволюционный алгоритм обучения и изменения нейронной сети, взятый за основу многослойный полносвязный перцептрон содержащий 12 нейронов во входном и выходном слое, а также 8 внутренних слоёв по 128 нейронов каждый.

После циклов обучения и до обучения структура нейронной сети представляла из себя архитектуру, потерявшую полносвязность внутренних слоёв, а также приобретшая новые внутренние слои, которые содержат следующее количество нейронов: {91, 184, 67, 66, 104, 34, 63, 37, 17, 95, 47, 174, 196, 216, 166, 275, 94}.

Вычисления при данной архитектуре сети на мобильном устройстве составляют порядка 3-5 секунд, что позволяет оператору производить настройку стола в зависимости от меняющихся входных параметров в режиме реального времени.

На данный момент достигнуто оптимальное, по мнению владельцев и ответственных сотрудников, отделение шлихового золота от хвостов, порядка 80%, оператор производит настройки 1-3 раза в день, либо при глазометрической фиксации изменения состава подаваемой пищи стола, Приложение А.

Для работы в полевых условия было разработано мобильное приложение, обеспечивающее удобные и оперативные вычисления, не требующее дополнительных компетенций от оператора столов в сфере IT. Приложение было разработано с учётом частого отсутствия интернет-связи на участке. Для этого приложение хранит последнюю версию архитектуры нейронной сети, полученной с сервера, и производит вычисления с её помощью, при этом сохраняя входные параметры, вводимые оператором. Когда устройство с установленным приложением получает доступ к сети интернет, то оно отправляет все вводимые оператором данные на сервер, а также получает последнюю обученную версию нейронной сети.

Пользовательский интерфейс, в котором вводятся данные и выводятся рекомендуемые параметры, представлен в Приложение Б. Интерфейс разрабатывался в JavaScript-библиотеке React.js.

# ЗАКЛЮЧЕНИЕ

Таким образом, в работе рассмотрен нейросетевой подход к решению задачи обогащения золотосодержащих россыпей посредством оптимизации настройки концентрационных столов. Применение нейронной сети позволяет получить оптимальные настройки оборудования в кратчайшие сроки. На основе многослойного перцептрона построена модель нейросетевого решения задачи и алгоритм, реализующий оптимизацию технологического процесса.

Разработано программное обеспечение по подбору параметров настройки концентрационных столов с помощью нейронной сети с процедурной генерацией архитектуры нейронной сети.

Предложен эволюционный алгоритм по отбору оптимальной нейронной сети.

Подготовлено программное и аппаратное обеспечение для автоматизации развёртывания LXC контейнеров с дедупликацией оперативной памяти.

Проведена оцифровка данных технологических процессов журналов гравитационной добычи золота с пяти приисков за 3-5 летний период.

Разработано мобильное кроссплатформенное приложение для использования обученной нейронной сети оператором в полевых условиях в отсутствие интернет-соединения.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Серавина, Т.В. Условия локализации золото-серебро-полиметаллического оруденения Березовогорского рудного поля (Рудный Алтай) / Т.В. Серавина. – М : МИР, 2016. – 128 с.

2. Буляков, Г.Х. Закономерности распределения золота и особенности опробования даек на стадии поисково-оценочных работ/ Г.Х. Буляков. – М : Физмалит, 1988. – 216 с.

3. Мальцев, Е.Н. Нейросетевые технологии обработки данных для решения практических задач прогнозирования в ходе геолого-технологического моделирования / Е.Н. Мальцев // Золото и технологии, 2021. – №1. – С.152–162.

4. Мишулович, П.М. Методологические аспекты создания геолого-технологических моделей месторождений полезных ископаемых/ П.М. Мишулович, С.В. Петров // Вестник СПбГУ. Науки о Земле, 2019. – Т. 64. – Вып. 2. – С. 249–266.

5. Мостович, В.Я. Обработка золотосодержащих руд (амальгамация, цианирование, хлоринация) / В.Я. Мостович, П.В. Бузников. – М : Мир, 2000. – 109 с.

6. Караганов, В.В. Кучное выщелачивание золота - зарубежный опыт и перспективы развития / В.В. Караганов, Б.С. Ужкенов. – М : ВИЭМС, 2012. – 288 с.

7. Мостович, В.Я. Пробирное искусство. Методы сухого пути / В.Я. Мостович. – Государственное научно-техническое издательство литературы по черной и цветной металлургии, Ленинград-Свердловск, 1934. – 147 с.

8. Якубчук, А.С. Металлогения золота в геодинамике Центральной Азии / А.С. Якубчук. – М: Мир, 2023. – 179 с.

9. Annels, A.E. Classification and reporting of mineral resources for high-nugget effect gold vein deposits / A.E. Annels, S.C. Dominy, P.R. Stephenson. – Canada: Canadian Institute of Mining Metallurgy and Petroleum, 2003. – 19 с.

10. База и генератор образовательных ресурсов [Электронный ресурс]// МГТУ им. Н.Э. Баумана, кафедра САПР. 2003-2015. URL: http://bigor.bmstu.ru/?cnt/?doc=NN/base.cou (дата обращения 17.03.2024).

11. Шитиков, В.К. Методы системной идентификации / В.К. Шитиков, Г.С. Розенберг, Т.Д. Зинченко. – Тольятти: ИЭВБ РАН. 2003. – 463 с.

12. Введение в теорию нейронных сетей [Электронный ресурс]//Основные понятия нейросетей/ URL: http://www.orc.ru (дата обращения 12.03.2024)

13. Исмаилов, Ш-М.А. Математическая модель нейрона и возможности его технической реализации / Ш-М.А. Исмаилов, Н.В. Поздняков // Системные технологии, 2014. – № 12, С. 1–10.

14. Aarts, L.P. Neural Network Method for Solving Partial Differential Equations / L.P. Aarts, P. Veer // Neural Processing Letters, 2001 – V. 14. –P. 261–271.

15. Ростовцев, В.С. Искусственные нейронные сети / В. С. Ростовцев. – 4-е изд., стер. – Санкт-Петербург : Лань, 2024. – 216 с. – ISBN 978-5-507-47362-5. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/364517 (дата обращения: 25.03.2024).

16. Гафаров Ф.М Искусственные нейронные сети и приложения: учеб. пособие / Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Изд-во Казан. ун-та, 2018. – 121 с.

17. Данилов, В.В. Проектирование искусственных нейронных сетей : методические указания / В. В. Данилов. – Донецк : ДонНУ, 2020. – 133 с. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/179954 (дата обращения: 25.03.2024).

18. Горожанина, Е.И. Нейронные сети : учебное пособие / Е. И. Горожанина. – Самара : Поволжский государственный университет телекоммуникаций и информатики, 2017. – 84 с. – Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. – URL: https://www.iprbookshop.ru/75391.html (дата обращения: 25.03.2024).

19. Maple [Электронный ресурс], 2020. URL: https://www.maplesoft.com/products/Maple/ (дата обращения: 13.04.2023).

20. OpenMT [Электронный ресурс], 2022. URL: https://opennmt.net/ (дата обращения: 13.04.2023).

21. Tensorflow [Электронный ресурс], 2020. URL: https://www.tensorflow.org/?hl=ru (дата обращения: 13.04.2023).

22. Емельянов, В.В., Теория и практика эволюционного моделирования / В.В. Емельянов, В.В. Курейчик, В.М. Курейчик. – М.: Физматлит, 2003. – 432 с. – ISBN 5-9221-0337-7.

23. Курейчик, В.М. Поисковая адаптация: теория и практика / В.М Курейчик, Б.К. Лебедев, О.К. Лебедев. – М.: Физматлит, 2006. – 272 с.

24. Гладков, Л.А. Генетические алгоритмы / Л.А. Гладков, В.В. Курейчик, В.М. Курейчик. – М.: Физматлит, 2006. – 320 с.

25. Гладков, Л.А. Биоинспирированные методы в оптимизации: монография / Л.А. Гладков, В.В. Курейчик, В.М. Курейчик. – М.: Физматлит, 2009. – 384 с.

26. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы = Sieci neuronowe, algorytmy genetyczne i systemy rozmyte / Д. Рутковская, М. Пилиньский, Л. Рутковский. – 2-е изд. – М.: Горячая линия-Телеком, 2008. – 452 с.

27. Рутковский, Л. Методы и технологии искусственного интеллекта /Л. Рутковский. – М.: Горячая линия-Телеком, 2010. – 520 с. – ISBN 5-9912-0105-6.

28. Волович, И.В. О теории моделирования и гиперграфе классов / И.В. Волович, М.Н. Хохлова // Труды Математического института им. В.А. Стеклова. – 2004. – Т. 245. – С. 281–287.

29. Перепёлкин, А.И. Нейросетевой подход к решению дифференциальных уравнений. Автоматизация вычислительного процесса / А.И. Перепёлкин // XXXII научная конференция «День науки», 2024. – 78-79 С.

30. Перепёлкин, А.И. Алгоритм поиска аналитического решения обыкновенных дифференциальных уравнений с применением методов

нейросетевого обучения / А.И. Перепёлкин, Л.И. Мороз // Студенческий форум, 2024. – С.138-142.

31. Перепёлкин, А.И. Свидетельство о государственной регистрации программы для ЭВМ № 2024664016 «Программа для автоматического изменения структуры нейронной сети» / А.И. Перепелкин, Л.И. Мороз // зарег. Федеральной службой по интеллектуальной собственности и товарным знакам, 17.06.2024, г. Москва.

# ПРИЛОЖЕНИЕ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
import React, {Component} from 'react';
import {View, Text, StatusBar, TouchableOpacity, StyleSheet} from 'react-native';
import configureStore from './app/store/configureStore';
import Main from './app/Main';
import { Provider } from 'react-redux';
/*const createStoreWithMiddleware = applyMiddleware(thunk)(createStore);
 const reducer = combineReducers(reducers);
 const store = createStoreWithMiddleware(reducer); */
const store = configureStore();
export default class Index extends Component {
   constructor() {
      super();
      this.state = {
         isLoading: false,
         store: configureStore()
      };
   }
   render() {
      return (
         <Provider store={this.state.store}>
            <Main />
         </Provider>
      );
   }
}
import React, {Component} from 'react';
import MainStackRouter from './routers/MainStackRouter';
import {
   Asset,
   AppLoading,
} from 'expo';
class Main extends Component {
   constructor(props) {
      super(props);
      this.state = {
         downloadProgress: true
      };
   }
   componentWillMount() {
      this._cacheResourcesAsync();
   }
   render() {
      if (this.state.downloadProgress) {
         return <AppLoading />;
      }
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
      return <MainStackRouter />;
    }
    async _cacheResourcesAsync() {
      const videos = [
        require('../assets/C.bin'),
        require('../assets/D.bin'),
        require('../assets/P.bin'),
        require('../assets/hourglass.bin'),
        require('../assets/operator.bin'),
        require('../assets/table.bin'),
        require('../assets/tap_icon.bin'),
        require('../assets/R1.bin'),
        require('../assets/R2_111_121_notab.bin'),
        require('../assets/R2_111_121_tab.bin'),
        require('../assets/R2_111_122_notab.bin'),
        require('../assets/R2_111_122_tab.bin'),
        require('../assets/R2_112_121_notab.bin'),
        require('../assets/R2_112_121_tab.bin'),
        require('../assets/R2_112_122_notab.bin'),
        require('../assets/R2_112_122_tab.bin'),
        require('../assets/R3_111_121.bin'),
        require('../assets/R3_111_122.bin'),
        require('../assets/R3_112_121.bin'),
        require('../assets/R3_112_122.bin'),
        require('../assets/R51_121.bin'),
        require('../assets/R51_122.bin'),
        require('../assets/R53_111_121.bin'),
        require('../assets/R53_111_122.bin'),
        require('../assets/R53_112_121.bin'),
        require('../assets/R53_112_122.bin'),
        require('../assets/R54_111.bin'),
        require('../assets/R54_112.bin'),
      ];
      for (let video of videos) {
        await Asset.fromModule(video).downloadAsync();
      }
      this.setState({downloadProgress: false});
    }
}
export default Main;
import { createStore, applyMiddleware } from 'redux';
import thunk from 'redux-thunk';
import reducer from '../reducers';
export default function configureStore(initialState) {
  const store = createStore(
    reducer,
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
    initialState,
    applyMiddleware(thunk),
  );
  if (module.hot) {
    module.hot.accept(() => {
      const nextRootReducer = require('../reducers/index').default;
      store.replaceReducer(nextRootReducer);
    });
  }

  return store;
};
import Login from "../components/login/";
import Home from "../components/home/";
import CardAnswer from "../components/cardanswer/";
import TopRate from "../components/toprate/";
import Profile from "../components/profile/";
import Blank from "../components/blank/";
import About from "../components/about/";
import { StackNavigator } from "react-navigation";
export default (StackNav = StackNavigator({
  Login: { screen: Login },
  Home: { screen: Home },
  CardAnswer: { screen: CardAnswer },
  TopRate: { screen: TopRate },
  Profile: { screen: Profile },
  Blank: { screen: Blank },
  About: { screen: About },
}));
import * as types from '../actions/actionTypes';
import Immutable from 'immutable';
import data from '../data/data.json';
import {
  Platform
} from 'react-native';
import _ from 'lodash';
const db = Immutable.fromJS(data);
const initialState = Immutable.fromJS({
  answers: {
    "11": "",
    "12": "",
    "13": "",
    "21": "",
    "22": "",
    "31": "",      "32": "",      "33": "",
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
  },
  kurs: 0,
  ploshad: 10,
  cards: [],
  step: 1,
  info: "",
  background: 'R1',
  flagChangeTables: false,
  waiter: false,

  kv_number: 0,
  total: 10000,
  stPom: 0,
  table: 150,
  tables: 6,
  waiters: 1,
  P: 0,
  D: 0,
  Pr: 0,
  Val: 0,
  C: 0,
  Pribl: 0,
function calculate_P({tables, waiters}) {
  return 11*96 * tables / (0.25 * Math.exp(tables/waiters/2) + 40) * (1 - 0.25 *
Math.exp(tables/waiters/2)/120);
}
function calculate_D({table, answers}) {
  let vl = 36926058 / table**2.5;
  if(answers["12"] === "121") {
    vl = 36926058 / table**2.5 * 1.05;
  }
  if(answers["31"] === "311") {
    vl = vl * 1.1;
  }
  return vl;}
function calculate_Pr({D, P}) {
  return Math.min(D, P);}
function calculate_Val({Pr, table}) {
  return Pr * table * 0.75;}
function calculate_1_C({answers, waiters, Val, tables, Pr, koeff_2=1, koeff_3=1}) {
  if(answers["11"] === "111") {
    if(answers["12"] === "121") {
      return (50000/30)*koeff_3 + ((60000 + waiters * 40000) / 30 + Val * 0.07) * koeff_2 +
Val*0.06 + ((tables - 6)*6000/30)*koeff_3 + Pr * 25 + 567 + 61;
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
        }
        return (50000/30)*koeff_3 + ((60000 + waiters * 40000) / 30 + Val * 0.07) * koeff_2 +
Val*0.06 + ((tables - 6)*6000/30)*koeff_3 + Pr * 25 + 567;
    }
    return (50000/30)*koeff_3 + ((60000 + waiters * 40000) / 30 + Val * 0.07) * koeff_2 + Val*0.06
+ ((tables - 6)*6000/30)*koeff_3 + Pr * 25;
}
function calculate_2_C({...param}) {
    let {answers, Pr, kurs} = param;
    let vl = calculate_1_C({...param});
    if(answers["21"] === "211") {
        vl = vl + Pr * 20 * 0.15/4;
    }
    if(answers["21"] === "212") {
        if(kurs > 0) {
            vl = vl + Pr * 20 * kurs;
        }
        if(kurs < 0) {
            vl = vl + Pr * 20 * kurs;
        }
    }
    if(answers["21"] === "213") {
        vl = vl + Pr * 20 * 0.05;
        if(kurs < 0) {
            vl = vl + Pr * 20 * kurs;
        }
    }
    return vl;
}
function calculate_3_C({...param}) {
    let {answers} = param;
    let vl = calculate_2_C({...param});
    if(answers["31"] === "312") {
        let koeff_2 = 1.15;
        vl = calculate_2_C({...param, koeff_2});
    }
    if(answers["31"] === "311") {
        let koeff_2 = 1.27;
        vl = calculate_2_C({...param, koeff_2});
    }
    return vl; }
function calculate_4_C({...param}) {
    let {answers, tables} = param;
    let vl = calculate_3_C({...param});
    if(answers["32"] === "321" && answers["31"] === "311") {
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
      vl = calculate_3_C({...param}) + ((50 + (tables-6)*6)*75000 +
2250000)*((0.13/12*(1+0.13/12)**84)/((1+0.13/12)**84-1))/30
   }
   if(answers["32"] === "321" && answers["31"] === "312") {
      vl = calculate_3_C({...param}) + ((50 + (tables-6)*6)*75000 +
2250000)*((0.15/12*(1+0.15/12)**84)/((1+0.15/12)**84-1))/30
   }
   if(answers["32"] === "322"){
      let koeff_3 = 0;
      vl = calculate_3_C({...param, koeff_3}) + ((50 + (tables-6)*6)*18+100)*(50 + (tables-
6)*6)/30
   }
   return vl;
}
function calculate_Pribl({Val, C, answers}) {
   if(answers["11"] === "112") {
      return (Val - C) * 90 * 0.7;
   }
   return (Val - C) * 90;
}
function get_card(step, context) {
   const card = db.getIn(['steps', `${step}`]);
   const body = _.template(card.get('body'))(context);
   return card.merge({body})
}

function get_info(step, context) {
   const info = db.getIn(['vvedenie', `${step}`]);
   return _.template(info)(context)
}
export default function common(state = initialState, action = {}) {
   if(action.type === types.FIRST_KVARTAL) {
      const newstate = initialState;
      const data = Immutable.fromJS({
         kv_number: 1,
         total: 10000.0,
         kurs: 0,
         stPom: 0,
         C: 0,
         Pribl: 0,
         table: 150.0,
         tables: 6,
         waiters: 1,
         background: 'R1',
         flagChangeTables: false,
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
        cards: [
          get_card('11'),
          get_card('12'),
          get_card('13')
        ],
        info: [
          get_info('10')
        ]
      });
      return newstate.merge(data);
    }
    if(action.type === types.NEXT_KVARTAL) {
      const answers = state.get('answers').toJS();
      const total = state.get('total');
      const waiters = state.get('waiters');
      const tables = state.get('tables');
      const table = state.get('table');
      let kurs = Math.round(Math.random()*30-15)/100;
      const ploshad = 50 + (tables-6)*6;
      const stPom =  (50 + (tables-6)*6)*75000 + 2250000;
      let kursInfo;
      if(kurs<0) {
        kursInfo = 'понизился';
      }
      if(kurs===0) {
        kursInfo = 'остался неизменным';
      }
      if(kurs>0) {
        kursInfo = 'повысился';
      }
      const P = calculate_P({tables, waiters});
      const D = calculate_D({table, answers});
      const Pr = calculate_Pr({D, P, answers});
      const Val = calculate_Val({Pr, table, answers});
      const C = calculate_4_C({answers, waiters, Val, tables, Pr, kurs});
      const Pribl = calculate_Pribl({Val, C, answers});
      /* console.log('NNNNNNNN');
      console.log(P);
      console.log(D);
      console.log(Pr);
      console.log(Val);
      console.log(C);
      console.log(Pribl);*/
      let newtotal = total + Pribl;
      const flagChangeTables = state.get('flagChangeTables');
      const data_statistic = Immutable.fromJS({
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
        total: newtotal,
        flagChangeTables: false, // обнуляем флаг для следующего периода
        ploshad: ploshad,
        stPom: stPom,
        P,
        D,
        Pr,
        Val,
        C,
        Pribl
      });
      if(action.kv_number === 1) {
        let background = state.get('background');
        if (answers['11'] === '111' && answers['12'] === '121' && !flagChangeTables) background
= 'R2_111_121_notab';
        if (answers['11'] === '111' && answers['12'] === '121' && flagChangeTables) background
= 'R2_111_121_tab';
        if (answers['11'] === '111' && answers['12'] === '122' && !flagChangeTables) background
= 'R2_111_122_notab';
        if (answers['11'] === '111' && answers['12'] === '122' && flagChangeTables) background
= 'R2_111_122_tab';
        if (answers['11'] === '112' && answers['12'] === '121' && !flagChangeTables) background
= 'R2_112_121_notab';
        if (answers['11'] === '112' && answers['12'] === '121' && flagChangeTables) background
= 'R2_112_121_tab';
        if (answers['11'] === '112' && answers['12'] === '122' && !flagChangeTables) background
= 'R2_112_122_notab';
        if (answers['11'] === '112' && answers['12'] === '122' && flagChangeTables) background
= 'R2_112_122_tab';
        let info = [
          get_info('20', {Pribl: Pribl.toFixed(0)})
        ];
        if (answers['11'] === '111') info.push(get_info('21:111'));
        if (answers['11'] === '112') info.push(get_info('21:112'));
        if (answers['12'] === '121') info.push(get_info('22:121'));
        if (answers['12'] === '122') info.push(get_info('22:122'));
        const data = Immutable.fromJS({
          cards: [
            get_card('21', {sumPartii: (Pr*20*90).toFixed(0)}),
            get_card('22'),
          ],
          info: info,
          background: background,
          kv_number: 2
        });
        return state.merge(data_statistic).merge(data);
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
    }
    if(action.kv_number === 2) {
      let background = state.get('background');
      if (answers['11'] === '111' && answers['12'] === '121') background = 'R3_111_121';
      if (answers['11'] === '111' && answers['12'] === '122') background = 'R3_111_122';
      if (answers['11'] === '112' && answers['12'] === '121') background = 'R3_112_121';
      if (answers['11'] === '112' && answers['12'] === '122') background = 'R3_112_122';
      let info = [
        get_info('30', {kursInfo: kursInfo, Pribl: Pribl.toFixed(0)})
      ];
      const izderjkiInfo = kurs > 0 ? 'увеличились' : 'сократились';
      const izderjki = kurs*100;
      const izderjkiInfo2 = kurs > 0 ? 'увеличились' : 'сократились';
      if (answers['21'] === '211') info.push(get_info('31:211', {C: C.toFixed(0)}));
      if (answers['21'] === '212') info.push(get_info('31:212', {kursInfo: kursInfo, izderjkiInfo:
izderjkiInfo, izderjki: izderjki.toFixed(0)}));
      if (answers['21'] === '213') info.push(get_info('31:213', {kursInfo: kursInfo, izderjkiInfo2:
izderjkiInfo2, izderjki: izderjki.toFixed(0)}));
      const data = Immutable.fromJS({
        kurs: kurs,
        cards: [
          get_card('31'),
          get_card('32', {ploshad: ploshad, stPom: stPom.toFixed(0)}),
          get_card('33'),
        ],
        info: info,
        background: background,
        kv_number: 3
      });
      return state.merge(data_statistic).merge(data);
    }
    if(action.kv_number === 3) {
      let background = state.get('background');
      if (answers['11'] === '111' && answers['12'] === '121' && !flagChangeTables) background
= 'R2_111_121_notab';
      if (answers['11'] === '111' && answers['12'] === '121' && flagChangeTables) background
= 'R2_111_121_tab';
      if (answers['11'] === '111' && answers['12'] === '122' && !flagChangeTables) background
= 'R2_111_122_notab';
      if (answers['11'] === '111' && answers['12'] === '122' && flagChangeTables) background
= 'R2_111_122_tab';
      if (answers['11'] === '112' && answers['12'] === '121' && !flagChangeTables) background
= 'R2_112_121_notab';
      if (answers['11'] === '112' && answers['12'] === '121' && flagChangeTables) background
= 'R2_112_121_tab';
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
        if (answers['11'] === '112' && answers['12'] === '122' && !flagChangeTables) background
= 'R2_112_122_notab';
        if (answers['11'] === '112' && answers['12'] === '122' && flagChangeTables) background
= 'R2_112_122_tab';
        let info = [
          get_info('40', {Pribl: Pribl.toFixed(0)})
        ];
        if (answers['31'] === '311' && answers['32'] === '321')info.push(get_info('41:311:321'));
        if (answers['31'] === '312' && answers['32'] === '321')info.push(get_info('41:312:321'));
        if (answers['32'] === '322')info.push(get_info('41:322'));
        if (answers['31'] === '311')info.push(get_info('42:311'));
        if (answers['31'] === '312')info.push(get_info('42:312'));
        const data = Immutable.fromJS({
          cards: [
            get_card('41'),
          ],
          info: info,
          background: background,
          kv_number: 4
        });
        return state.merge(data_statistic).merge(data);
      }
      if(action.kv_number === 4) {
        let background = state.get('background');
        let info = [
          get_info('50')
        ];
        const best = 1000000;
        if (newtotal < 0 && answers['12'] === '121') background = 'R51_121';
        if (newtotal < 0 && answers['12'] === '122') background = 'R51_122';
        if (newtotal > 0 && newtotal < best && answers['11'] === '111' && answers['12'] ===
'121') background = 'R53_111_121';
        if (newtotal > 0 && newtotal < best && answers['11'] === '111' && answers['12'] ===
'122') background = 'R53_111_122';
        if (newtotal > 0 && newtotal < best && answers['11'] === '112' && answers['12'] ===
'121') background = 'R53_112_121';
        if (newtotal > 0 && newtotal < best && answers['11'] === '112' && answers['12'] ===
'122') background = 'R53_112_122';
        if (newtotal >= best && answers['11'] === '111') background = 'R54_111';
        if (newtotal >= best && answers['11'] === '112') background = 'R54_112';
        if(newtotal < 0) info.push(get_info('51:1'));
        if(newtotal == 0) info.push(get_info('51:2'));
        if(newtotal > 0) info.push(get_info('51:3'));
        if(newtotal > 1000000) info.push(get_info('51:4'));
        info.push(get_info('52', {
          chashek: (36926058/table**2.5).toFixed(0),
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
            viruchka: (Val*30).toFixed(0),
            Total: newtotal.toFixed(0),
          }));
          const data = Immutable.fromJS({
            cards: [],
            info: info,
            background: background,
            kv_number: 5
          });
          return state.merge(data_statistic).merge(data);
        }
        return state;
    }
    if(action.type === types.SETTINGS_SET) {
        const flagChangeTables = state.get('tables') === action.tables;
        const newstate = Immutable.fromJS({
          table: action.table,
           tables: action.tables,
           waiters: action.waiters,
           flagChangeTables: flagChangeTables  // изменилось ли количество столов в текущем периоде?
        });
        return state.merge(newstate);
    }
    if(action.type === types.ANSWER_SELECT) {
        return state.setIn(['answers', action.question], action.answer);
    }
    if(action.type === types.DROP_CARD) {
        const cards = state.get('cards').filter((card)=>  card.get('number') !== action.question);
        return state.merge({cards});
    }
    if(action.type === types.SCREENWAIT_VISIBLE) {
        return state.merge({waiter: action.visible});
    }
    if(action.type === types.TESTACTION) {
        const answers = state.get('answers').toJS();
        const flagChangeTables = state.get('flagChangeTables');
        let background = state.get('background');
        if (answers['11'] === '111' && answers['12'] === '121' && !flagChangeTables) background = 'R2_111_121_notab';
        if (answers['11'] === '111' && answers['12'] === '121' && flagChangeTables) background = 'R2_111_121_tab';
        if (answers['11'] === '111' && answers['12'] === '122' && !flagChangeTables) background = 'R2_111_122_notab';
        if (answers['11'] === '111' && answers['12'] === '122' && flagChangeTables) background = 'R2_111_122_tab';
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
      if (answers['11'] === '112' && answers['12'] === '121' && !flagChangeTables) background =
'R2_112_121_notab';
      if (answers['11'] === '112' && answers['12'] === '121' && flagChangeTables) background =
'R2_112_121_tab';
      if (answers['11'] === '112' && answers['12'] === '122' && !flagChangeTables) background =
'R2_112_122_notab';
      if (answers['11'] === '112' && answers['12'] === '122' && flagChangeTables) background =
'R2_112_122_tab';
      return state.merge({background});
   }
   return state;
}
import { combineReducers } from 'redux';
import common from './common';
import profile from './profile';
export default combineReducers({
   common,
   profile
})
import * as types from '../actions/actionTypes';
/* import fake from '../fake/fakeData'; */
import Immutable from 'immutable';
const ISDEBUG =  false;// __DEV__;
const initialState = Immutable.fromJS({
   username: 'Василий',
   tablename: 'Starbucks',
   sex: 'm'
});
export default function profile(state = initialState, action = {}) {
   switch (action.type) {
      case types.LOGIN:
         // console.log('UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU22333',
action.username, action.tablename);
         // __DEV__ && console.log(state.toJSON());
         const data = Immutable.fromJS({
            username: action.username,
            tablename: action.tablename,
            sex: action.sex,
         });
         return state.merge(data);
      default:
         return state;
   }
}
const data = [
   {
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
      type: "image",
      uri: "file:///storage/sdcard/DCIM/IMG_20161109_041654.jpg",
      isSel: false,
      isProcess: false
    },
    {
      type: "image",
      uri: "file:///storage/sdcard/DCIM/IMG_20161109_041658.jpg",
      isSel: false,
      isProcess: false
    },
    {
      type: "image",
      uri: "file:///storage/sdcard/DCIM/IMG_20161109_041707.jpg",
      isSel: false,
      isProcess: false
    }
];
export const DTP_STATUS_WAIT = 'DTP_STATUS_WAIT';
export const DTP_STATUS_PAYED = 'DTP_STATUS_PAYED';
export const CUSTOM_FONT = {
  fontFamily: 'sans-serif-thin',
  color: '#000'
};
export const STORAGE_KEY_PROFILE_ID = 'profileId';
import { StyleSheet } from 'react-native';
export default StyleSheet.create({
  scroll: {
    backgroundColor: 'white'
  },
  background: {
    padding: 20,
  },
  header: {
    alignItems: 'center',
    paddingBottom: 6,
  },
  headerText: {
    fontSize: 18,
    fontWeight: 'bold',
    textAlign: 'center'
  },
  paraghaph: {
    paddingBottom: 15
  },
  paraghaphImageContainer: {
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
      height: 100,
    },
    paraghaphImage: {
      flex:1,
      width: undefined,
      height: undefined
    },
    paraghaphText: {
      paddingBottom: 3,
      fontSize: 16,
    },
    paraghaphList: {
      paddingLeft: 4,
      paddingBottom: 3,
    },
    paraghaphListElement: {
    },
    paraghaphListElementText: {
      //padding: 2,
      fontSize: 16,
    },
});
import React, {Component} from 'react';
import {
  View
} from 'react-native';
import { Video } from 'expo';
import styles from "./styles";
const images = {
  'R1': require('../../../assets/R1.bin'),
  'R2_111_121_notab': require('../../../assets/R2_111_121_notab.bin'),
  'R2_111_121_tab': require('../../../assets/R2_111_121_tab.bin'),
  'R2_111_122_notab': require('../../../assets/R2_111_122_notab.bin'),
  'R2_111_122_tab': require('../../../assets/R2_111_122_tab.bin'),
  'R2_112_121_notab': require('../../../assets/R2_112_121_notab.bin'),
  'R2_112_121_tab': require('../../../assets/R2_112_121_tab.bin'),
  'R2_112_122_notab': require('../../../assets/R2_112_122_notab.bin'),
  'R2_112_122_tab': require('../../../assets/R2_112_122_tab.bin'),
  'R3_111_121': require('../../../assets/R3_111_121.bin'),
  'R3_111_122': require('../../../assets/R3_111_122.bin'),
  'R3_112_121': require('../../../assets/R3_112_121.bin'),
  'R3_112_122': require('../../../assets/R3_112_122.bin'),
  'R51_121': require('../../../assets/R51_121.bin'),
  'R51_122': require('../../../assets/R51_122.bin'),
  'R53_111_121': require('../../../assets/R53_111_121.bin'),
  'R53_111_122': require('../../../assets/R53_111_122.bin'),
```

Листинг программы мобильного приложения настройки работы концентрационного стола

```
  'R54_112_121': require('../../../assets/R53_112_121.bin'),
  'R53_112_122': require('../../../assets/R53_112_122.bin'),
  'R54_111': require('../../../assets/R54_111.bin'),
  'R54_112': require('../../../assets/R54_112.bin'),
};
export default class Background extends Component {
  render() {
    const total = this.props.total;
    return (
      <View style={{flex:1}}>
        <Video
          rate={1.0}
          volume={0}
          muted={true}
          resizeMode="cover"
          shouldPlay
          isLooping
          style={styles.background}
          source={images[this.props.gifname]}
        />
        {this.props.children}
      </View>
    );
  }
};
import { StyleSheet, Dimensions } from 'react-native';
const { width: viewportWidth, height: viewportHeight } = Dimensions.get('window');
export default StyleSheet.create({
  background: {
    position: 'absolute', left: 0, right: 0, top: 0, bottom: 0, backgroundColor: '#cbd2d9'
  },
});
import React, {Component} from 'react';
import {
  Text,
  View
} from 'react-native';
import { connect } from "react-redux";
class BlankComponent extends Component {
  static navigationOptions = ({ navigation }) => ({
    title: `Blank with...`,
  });
  render() {
    return (
      <View>
        <Text>Пустое окно...</Text>
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
            </View>
        );
    }
}
function bindAction(dispatch) {
    return {
        setIndex: index => dispatch(setIndex(index)),
        openDrawer: () => dispatch(openDrawer())
    };}
const mapStateToProps = state => ({
    /* name: state.user.name*/
    data: 'Blank data'
});
const Blank = connect(mapStateToProps, bindAction)(BlankComponent);
export default Blank;
import React, {Component} from 'react';
import {
    Text,
    View,
    ScrollView
} from 'react-native';
import * as styles from "./styles";
import Modal from "react-native-modalbox";
import { FontAwesome } from '@expo/vector-icons';
import Touchable from '../../touchable/';
const AnswerButton = (props) => {
    return (
        <View style={styles.answers.buttonWrap}>
            <Touchable
                onPress={() => props.onPress(props.number, props.info, props.title, props.comments)}>
                <View style={styles.answers.button}>
                    <Text style={styles.answers.buttonText}>{props.title}</Text>
                </View>
            </Touchable>
        </View>
    ); };
const Comment = (props) => {
    return (
        props.type === 'minus' ?
            <View style={styles.card.comment}>
                <View style={styles.card.commentLabel}>
                    <View style={styles.card.commentLabelMinus}>
                        <FontAwesome name="minus" size={12} color="white"/>
                    </View>
                </View>
                <Text style={styles.card.commentText}>{ props.title }</Text>          </View>          :
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
        <FontAwesome name="plus" size={12} color="green"/>     ); };
export default class CardSelectQuestion extends Component {
  constructor(props) {
    super(props);
    this.state = {
      title: "",
      number: "",
      info: "",
      comments: []
    };    }    _reg(ref) {
    this._modal = ref;    }
  _onPressBtn(number, info, title, comments) {
    /* console.log('TTT', number); */
    this.setState({number, info, title, comments});
    this._modal.open();    }
  _onPressConfirm() {
    const navigation = this.props.navigation;
    const question = this.props.card.get('number');
    const answer = this.state.number;
    this.props.actions.selectAnswer(navigation, question, answer);    }
  _onPressCancel() {
    this._modal.close();    }
  render() {
    const card = this.props.card;
    return (
      <View style={styles.card.form}>
        <View style={styles.card.body}>
          <ScrollView>
            <Text style={styles.card.bodyText}>{ card.get('body') }</Text>
          </ScrollView>
        </View>
        <View style={styles.answers.background}>
          <View style={styles.answers.header}>
            <Text style={styles.answers.headerText}>Что выбираем?</Text>
          </View>
          <View style={styles.answers.container}>
            {
              card.getIn(['options','answers']).map((answer) => <AnswerButton
key={answer.get('number')} onPress={this._onPressBtn.bind(this)} {...answer.toJS()}/>)
            }          </View>        </View>        <Modal
          style={styles.card.modal}
          backdrop={true}
          swipeToClose={false}
          ref={this._reg.bind(this)}>
          <View style={styles.card.form}>
            <View style={styles.card.backlight}>
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
                    <View style={styles.card.header}>
                      <Text style={styles.card.headerText}>{ this.state.title }</Text>
                    </View>
                    <ScrollView>
                      <View style={styles.card.body}>
                        <Text style={styles.card.bodyText}>{ this.state.info }</Text>
                      </View>
                    </ScrollView>
                    {
                      this.state.comments ?
                        <View style={styles.card.commentContainer}>
                          {
                            this.state.comments.map((comment, number)=> <Comment
key={number} {...comment}/>)
                          }
                        </View>
                        :
                        undefined
                    }

                    <View style={styles.card.footer}>
                      <Touchable
                        onPress={this._onPressConfirm.bind(this)}>
                        <View style={styles.card.btnConfirm}>
                          <Text style={styles.card.btnText}>Подтвердить</Text>
                        </View>
                      </Touchable>
                      <Touchable
                        onPress={this._onPressCancel.bind(this)}>
                        <View style={styles.card.btnCancel}>
                          <Text style={styles.card.btnText}>Отмена</Text>
                        </View>
                      </Touchable>
                    </View>                    </View>                </View>                </Modal>
  </View>         );     } }
import React, {Component} from 'react';
import {
  Text,
  View,
  ScrollView
} from 'react-native';
import * as styles from "./styles";
import { FontAwesome } from '@expo/vector-icons';
import Touchable from '../../touchable/';
const ButtonOK = (props) => {
  return (
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
      <View style={styles.control.buttonWrap}>
        <Touchable
          onPress={props.onPress}>
            <View style={styles.control.button}>
              <Text style={styles.control.buttonText}>OK</Text>
            </View>          </Touchable>
      </View>    ); }; const WidgetControl = (props) => {
  return (
      <View style={props.last ? styles.control.widgetWrapL : styles.control.widgetWrap }>
          <View>
            <Text style={styles.control.widgetTitle}>{ props.title }</Text>
          </View>
          <View style={styles.control.widgetBody}>
            <Touchable
              underlayColor="#f1f1f1"
              onPress={()=>props.onPress(-1)}>
              <View style={styles.control.arrow}>
                <FontAwesome name="chevron-down" size={26} color="gray" />
              </View>
            </Touchable>
            <View style={styles.control.count}>
              <View>
                <Text style={styles.control.countText}>{ props.count }</Text>
              </View>
            </View>
            <Touchable
              underlayColor="#f1f1f1"
              onPress={()=>props.onPress(1)}>
              <View style={styles.control.arrow}>
                <FontAwesome name="chevron-up" size={26} color="gray" />
              </View>
            </Touchable>
          </View>
      </View>    ); };
export default class CardSettings extends Component {
  constructor(props) {
    super(props);
    this.state = {
      table: this.props.table,
      tables: this.props.tables,
      waiters: this.props.waiters,
    };    }    _reg(ref) {
    this._modal = ref;    }
  _onPressOK() {
    const navigation = this.props.navigation;
```

Продолжение ПРИЛОЖЕНИЯ А

Листинг программы мобильного приложения настройки работы концентрационного стола

```
    const question = this.props.card.get('number');
     const table = this.state.table;
     const tables = this.state.tables;
     const waiters = this.state.waiters;
     this.props.actions.setSettings(navigation, question, table, tables, waiters);    }
  _onPresstable(step) {
     const table = this.state.table + step*10;
     if(table >= 10 && table <= 1000) {
        this.setState({table});
     }    }
  _onPressTables(step) {
     const tables = this.state.tables + step;
     if(tables >= 6 && tables <= 100) {
        this.setState({tables});
     }    }
  _onPressWaiters(step) {
     const waiters = this.state.waiters + step;
     if(waiters >= 1 && waiters<= 10) {
        this.setState({waiters});
     }    }
  render() {
     const card = this.props.card;
     return (
        <View style={styles.card.form}>
          <View style={styles.card.backlight}>
            <View style={styles.card.header}>
              <Text style={styles.card.headerText}>Что делаем?</Text>
            </View>
            <ScrollView>
              <View style={styles.card.body}>
                <Text style={styles.card.bodyText}>
                </Text>
              </View>
            </ScrollView>
          </View>
          <View style={styles.control.background}>
            <View style={styles.control.header}>
              <WidgetControl title="Цена за кофе" onPress={this._onPresstable.bind(this)}
count={this.state.table}/>
              <WidgetControl title="Столики" onPress={this._onPressTables.bind(this)}
count={this.state.tables}/>
              <WidgetControl title="Персонал" onPress={this._onPressWaiters.bind(this)}
count={this.state.waiters} last/>
            </View>
            <ButtonOK onPress={this._onPressOK.bind(this)}/>
          </View>          </View>        );    } }
```

ПРИЛОЖЕНИЕ Б

Пользовательский интерфейс приложения настройки работы концентрационного стола

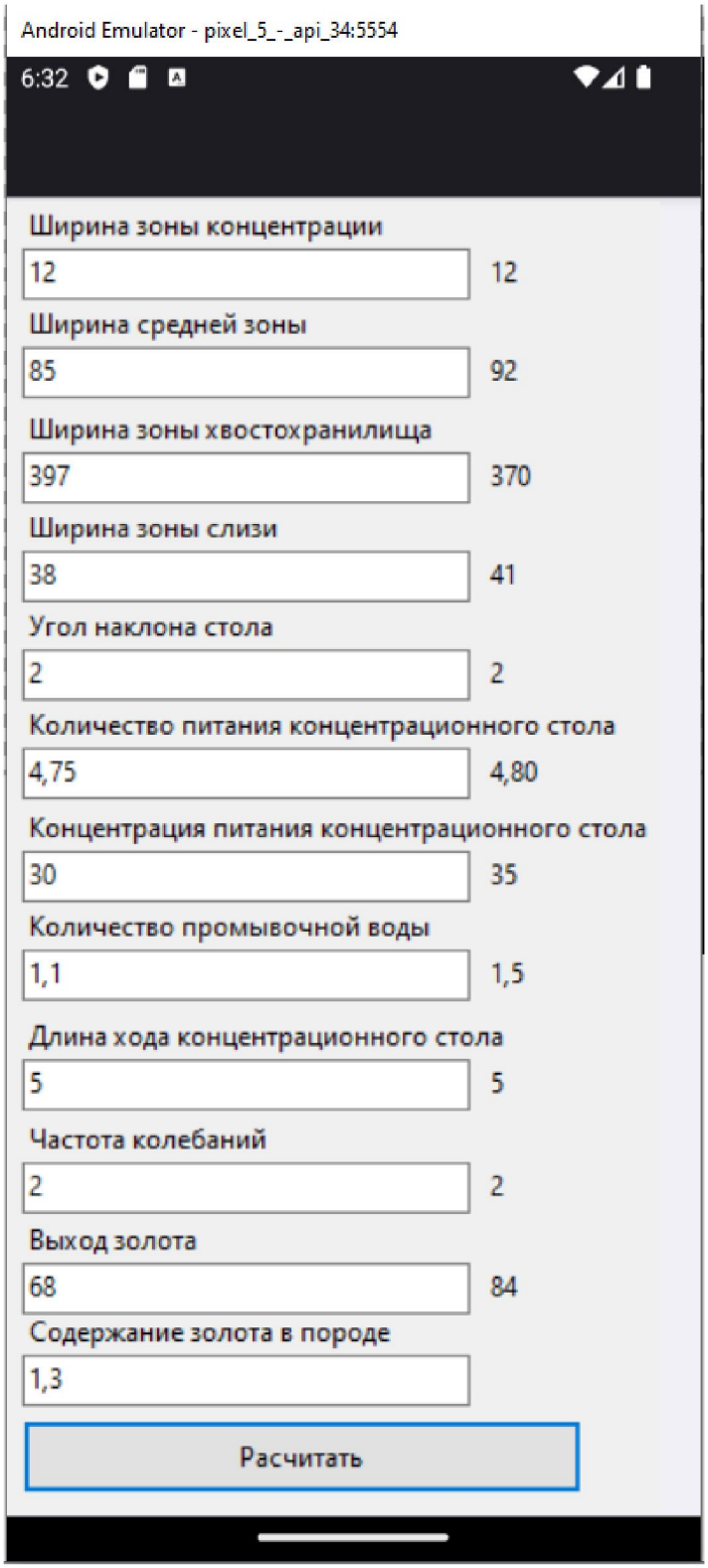



Рисунок Б.1 – Пользовательский интерфейс