Негосударственное частное образовательное учреждение
высшего образования
«Технический университет УГМК»

ЗАДАНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ СТУДЕНТОВ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ *ПО МОДУЛЮ* МОДУЛЬ 5 ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ УПРАВЛЕНИЯ ЭЛЕКТРОХОЗЯЙСТВОМ ПРЕДПРИЯТИЙ

Направление подготовки	***13.04.02 Электроэнергетика и электротехника***
Направленность (профиль)	***Управление и устойчивое развитие электрохозяйства предприятия***
Уровень высшего образования	***магистратура*** *(бакалавриат, специалитет, магистратура)*
Квалификация выпускника	***магистр***

Автор - разработчик: доктор техн. наук, профессор Карякин А.Л.
Рассмотрено на заседании кафедры энергетики
Одобрено Методическим советом университета 30 июня 2021 г., протокол № 4

г. Верхняя Пышма
2021

Методические рекомендации для магистрантов по выполнению курсовой работы составлены в соответствии с рабочей программой модуля «Модуль 5 Интеллектуальные системы управления электрохозяйством предприятий».

Курсовая работа по модулю Модуль 5 Интеллектуальные системы управления электрохозяйством предприятий предусмотрен в 5 семестре. Он является составной частью самостоятельной работы магистрантов.

Курсовое проектирование имеет целью закрепление магистрантами полученных на лекциях теоретических знаний и практического опыта, приобретенного на практических занятиях, путем самостоятельной работы под руководством преподавателя.

1. Тебования к содержанию и оформлению пояснительной записки к курсовой работы.

В пояснительной записке объемом 20-30 страниц текста, включая необходимые иллюстрирующие материалы (чертежи, схемы, графики, рисунки) излагается идеи и существо работы, приводятся результаты теоретических расчетов, приводят выводы.

При написании записки студент обязан дават сылки на автора и источники, откуда он заимствует материал или отдельные результаты. В тексте пояснительной записки недопустимыми являются орфографические и синтаксические ошибки и описки, небрежное оформление рисунков, таблиц, схем.

Пояснительная записка курсовой работы должна содержать следующие структурные части:

- Титульный лист;
- Задание;
- Содержание;
- Обозначения и сокращения;
- Введение;
- Основная часть;
- Заключение;
- Список использованных источников;
- Приложения.

2. Содержание пояснительной записки

Структурные части пояснительной записки начинают с нового листа, заголовки не нумеруют и размещают по центру строки. Исключение — заголовки основной части и приложения. Заголовки основной части начинают с абзацного отступа. Пояснительная записка переплетается либо вставляется в стандартные папки для дипломных работ. Пояснительная записка подписывается студентом на титульном листе с указанием даты окончания работы.

2.1. Титульный лист

Титульный лист является началом пояснительной записки. Пример оформления титульных листов приведены в приложении №1. На титульном листе указываются: полное наименование учебного заведения, структурного подразделения, оценка работы, тема работы, инициалы и фамилия студента и руководителя, город и год выполнения. На титульном листе должны быть подписи всех вышеуказанных лиц с указанием даты.

2.2. Задание

Задание на курсовой проект составляется по установленной форме, руководителем работы и студентом и помещается на странице, следующей за титульным листом. Задание не нумеруется.

2.3. Содержание

Содержание должно включать: введение, наименование всех разделов, подразделов, пунктов, заключение, список использованных источников и наименование приложений с указанием номеров страниц (ГОСТ 7.32-2017), с которых начинаются эти элементы.

2.4. Обозначения и сокращения

Раздел должен содержать перечень обозначений и сокращений, применяемых в пояснительной записке. Запись обозначений и сокращений проводят в порядке приведения их в тексте записки с необходимой расшифровкой и пояснениями. Допускаются определения, обозначения и сокращения приводить в одном структурном элементе «Определения, обозначения и сокращения».

2.5. Введение

Введение к пояснительной записке должно содержать актуальность выполняемого проекта.

2.6. Основная часть

Основной текст пояснительной записки, определяющий ее содержание, должен излагаться в строгой логической последовательности. Независимо от разнообразия задач и методов их решения основная часть пояснительной записки должна содержать следующие разделы, представляющие задания по темам Модуля:

- Моделирование систем электроснабжения
- Smart Grid предприятия
- Учет и качество электрической энергии

2.7. Заключение

В разделе должны отражаться основные результаты проделанной работы, оценка полноты решений поставленных задач, рекомендации по практическому использованию полученных результатов. Объем заключения должен составлять не более 1 страницы.

2.8. Список использованных источников

В списке указываются все источники, использованные в процессе работы. На них должны иметься соответствующие ссылки в тексте пояснительной записки. Источники следует располагать в порядке появления ссылок в тексте записки, нумеровать арабскими цифрами без точки. Сведения об источниках, включенных в список, необходимо давать в соответствии с требованиями ГОСТ 7.1-2003 и ГОСТ Р 7.0.9-2009.

Приложение 1 к методическим рекомендациям для магистров
по выполнению курсовой работы по модулю
Модуль 5 Интеллектуальные системы управления электрохозяйством предприятий

Негосударственное частное образовательное учреждение
высшего образования
«Технический университет УГМК»

Задание
на курсовую работу (проект)

по модулю _____

студента_____группы_____

специальность/направление подготовки_____

1.Тема курсовой работы (проекта)_____

_____

_____

2.Содержание (индивидуальное задание) курсовой работы (проекта), в том числе состав графических работ и расчетов _____

_____

3. 3. Структура работы:

_____

4. График работы _____

Наименование элементов проектной работы	Сроки	Примечания	Отметка о выполнении

Руководитель _____/И.О. Фамилия/

Негосударственное частное образовательное учреждение высшего образования
«Технический университет УГМК»

Кафедра_____*

КУРСОВАЯ РАБОТА (ПРОЕКТ)

по модулю «Интеллектуальные системы управления электрохозяйством предприятий»

Тема_____

Студент (ка)_____
ФИО

Группа _____

Руководитель _____
ФИО

_____
ученая степень, ученое звание

_____ _____
оценка подпись

Дата сдачи _____ _____20____г.

г. Верхняя Пышма
20___г.

*На титульном листе указывается название кафедры, за которой закреплен данный модуль.

Модуль 5 Интеллектуальные системы управления электрохозяйством предприятий

Негосударственное частное образовательное учреждение
высшего образования
«Технический университет УГМК»

РЕЦЕНЗИЯ
на курсовую работу (проект)

Студента _____ группы _____
(фамилия имя отчество)

Тема курсовой работы (проекта):
_____
_____
_____

Модуль 5 Интеллектуальные системы управления электрохозяйством предприятий
_____

1. Соответствие результатов выполнения работы целям и задачам курсовой работы (проекта), результатам обучения по дисциплине_____
_____
_____

2. Оригинальность и самостоятельность выполнения работы_____
_____
_____
_____

3. Полнота и глубина проработки разделов _____
_____

4. Общая грамотность и качество оформления текстового документа и графических материалов_____
_____
_____

5. Вопросы и замечания _____
_____

6. Общая оценка работы _____
_____
_____

Сведения о рецензенте:

Ф.И.О. _____
Уч. звание _____ Уч. степень _____

Дата ____ _____20____г. _____И.О. Фамилия
подпись

Негосударственное частное образовательное учреждение
высшего образования
«Технический университет УГМК»

Задание
на курсовую работу

по модулю №5 «Интеллектуальные системы управления электрохозяйством предприятий»

студента Артамонова М.Н. группы_Эн-1915з.

направление подготовки электроэнергетика

1.Тема курсовой работы: программирование алгоритма прогнозирования временного ряда методом искусственных нейронных сетей

2.Содержание (индивидуальное задание) курсовой работы, в том числе состав графических работ и расчетов _____

_____

1. Структура работы:
 - Устойчивое развитие электрохозяйства предприятия;
 - Методы эффективного прогнозирования потребления электроэнергии предприятий;
 - Управление энергоэффективностью предприятий

5. График работы _____

Наименование элементов проектной работы	Сроки	Примечания	Отметка о выполнении
Устойчивое развитие электрохозяйства предприятия			
Методы эффективного прогнозирования потребления электроэнергии предприятий			
Управление энергоэффективностью предприятий			

Руководитель _____/А.Л. Карякин/

Содержание

ВВЕДЕНИЕ. 10
Методы обучения синаптических весов нейрона. 11
Сигмоидная функция активации. 12
Проблема полноты. 12
Проблема исключающего «ИЛИ». 12
Многослойные нейронные сети. 13
Вычислительные возможности НС. 13
Многослойные НС. 14
Улучшение сходимости и качества градиентного обучения. 14

ОСНОВНАЯ ЧАСТЬ. 17
Описание объекта. 17
Постановка задачи. 18
Описание алгоритма. 18
Описание результатов. 23

ЗАКЛЮЧЕНИЕ. 29

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 30

Введение.

В наше время успешном становится то предприятие, которое постоянно работает над повышением своей энергоэффективности. В особенности это касается крупных предприятий с высоким уровнем потребления энергоресурсов. Одним из основополагающих аспектов эффективности служит грамотное планирование потребления энергоресурсов. Оно позволяет предприятиям повысить качество управления финансами, а также избежать переплат за невыполнение графика нагрузок. Наиболее перспективным инструментом прогнозирования являются искусственные нейронные сети, модели которых мы рассмотрим в данной работе и выполним программирование алгоритма прогнозирования временного ряда методом искусственных нейронных сетей, для \ использования при прогнозировании потребления на сутки вперед для приобретения электроэнергии на оптовом рынке.

В 1943 по образу и подобию естественного нейрона математики МакКаллок и Питтс создали математическую модель.

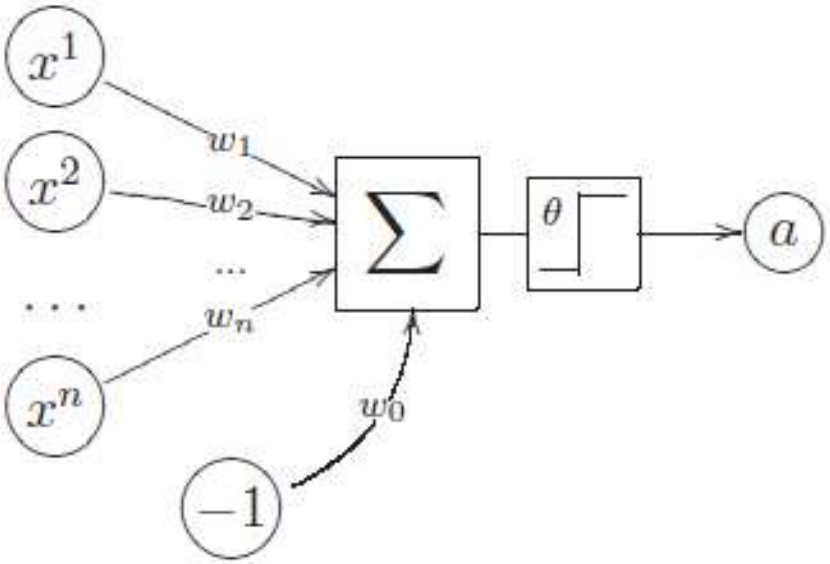


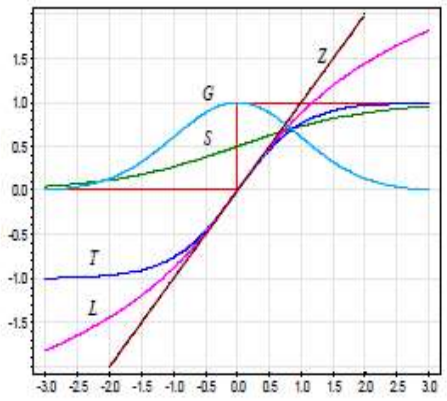
Рисунок 1

Модель имеет n входных каналов данных со своими признаками. То есть математический нейрон имеет n синапсов по которым поступают импульсы с определенными величинами. Каждый входной канал складывается с соответствующим весом. В зависимости от значения веса синапс может быть возбуждающим (+) и тормозящим (-). На рисунке 1 можно увидеть еще один входной параметр - ω_0, это порог активности. Если сумма n импульсов превышает порог, то происходит возбуждение и на выходе получаем 1, если порог не превышается получаем 0 и нейрон тормозит. Функцию нейрона можно представить в виде:

$$a(x) = \varphi\left(\sum_{j=1}^{n} w_j x^j - w_0\right),$$

Функция φ преобразует сумму импульсов в выходное значение, она называется функцией активации.

В последствии данная модель была обобщена. Наиболее часто используемые функции приведены на рисунке 2.



$\theta(z) = [z \geqslant 0]$	ступенчатая функция Хэвисайда;
$\sigma(z) = (1 + e^{-z})^{-1}$	сигмоидная функция (S);
$\mathrm{th}(z) = 2\sigma(2z) - 1$	гиперболический тангенс (T);
$\ln(z + \sqrt{z^2 + 1})$	логарифмическая функция (L);
$\exp(-z^2/2)$	гауссовская функция (G);
z	линейная функция (Z);

Рис. 2. Функции активации.

Методы обучения синаптических весов нейрона.

Персептрон Розенблата. Розенблат предложил эвристический алгоритм обучения нейрона. Принцип заключается в тренировке синаптических связей. Чем чаще синапс угадывает правильный ответ, тем сильнее становится связь соседних нейронов и таким образом происходит запоминание информации. В математике память — это вектор синапсических весов ω. Перед обучением векторам весов присваиваются значения, нулевые либо случайные. Затем обучающие объекты x_i поочередно подаются на вход модели МакКаллока-Питтса, и полученные ответы сравниваются с правильными.

Если ответ $a(x_i)$ совпадает с y_i – вектор весов не изменяется. В случае если $a(x_i)=0$ и $y_i=1$ увеличивается вектор весов ω. На результат влияет увеличение весов, соответствующих ненулевым компонентам x_i. Исходя из этого получаем:

$$w := w + \eta x_i,$$

Η-темп обучения

При $a(x_i)=1$ и $y_i=0$ вектор весов уменьшается

Оба случая можно объединить в формулу

$$w := w - \eta\big(a(x_i) - y_i\big)x_i.$$

Обучение повторяется до тех пор, пока веса меняются.

Метод стохастического градиента.

Метод поиска весов может быть основан на поиске вектора ω, доставляющего минимум функционалу качества:

$$Q(w) = \sum_{i=1}^{\ell} \mathscr{L}\big(a(x_i), y_i\big) \to \min_w,$$

$\mathscr{L}(a, y)$ - функция потерь, характеризует величину ошибки, *a* при правильном ответе *y*.

Правило изменения вектора весов на каждой итерации будет иметь вид:

$$w := w - \eta\frac{\partial Q}{\partial w},$$

Η>0 – величина шага.

Градиент будет иметь вид:

$$w := w - \eta \sum_{i=1}^{\ell} \mathscr{L}'_a\big(a(x_i), y_i\big)\varphi'(\langle w, x_i\rangle)x_i.$$

Здесь каждый обучающий элемент вносит свой вклад в обучение, но вектор изменяется только после предъявления всех объектов. Можно применить другой подход – для каждого объекта обновлять вектор. Когда на каждой итерации алгоритма из обучающей выборки случайным образом выбирается только один объект. Таким образом вектор w настраивается на каждый вновь выбираемый объект- метод называется стохастическим градиентом. Объекты перебираются в случайном порядке для каждого подбирается градиентный шаг и вектор весов:

$$w := w - \eta \mathscr{L}'_a\big(a(x_i), y_i\big)\varphi'(\langle w, x_i\rangle)x_i.$$

Такой процесс сходится быстрее градиентного метода.

Сигмоидная функция активации.

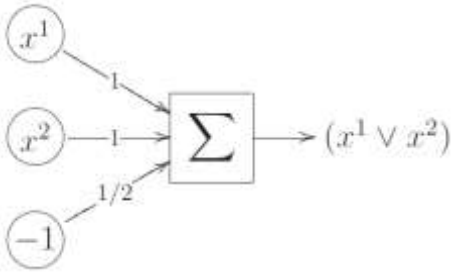
Используется для классификации. Нейрон с данной функцией вычисляет вероятность принадлежности объекта какому-либо классу.

Проблема полноты.

Возможности однослойного нейрона ограничены линейностью, что делает его плохо применимым в практических задачах.

Проблема исключающего «ИЛИ».

В случае с бинарными переменными легко построить логические функции И, ИЛИ, НЕ.



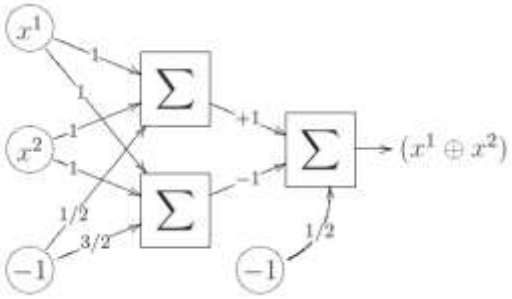
Однако функцию исключающую ИЛИ невозможно реализовать одним нейроном в двумя входами, так как множество нулей и единиц в этой функции линейно не разделимы.

1 путь решения проблемы- дополнить признаки, подавая на вход преобразованные исходные признаки. Если разрешить преобразовывать произведения исходных признаков, то нейрон будет строить не линейную, а полиномиальную поверхность. В случае исключающего ИЛИ достаточно добавить один вход x^1x^2, что бы множества нулей и единиц оказались линейно разделимыми:

$$x^1 \oplus x^2 = \left[x^1 + x^2 - 2x^1x^2 - \tfrac{1}{2} > 0\right].$$

Продляема в том, что подбор нужных преобразований является нерешенной задачей.

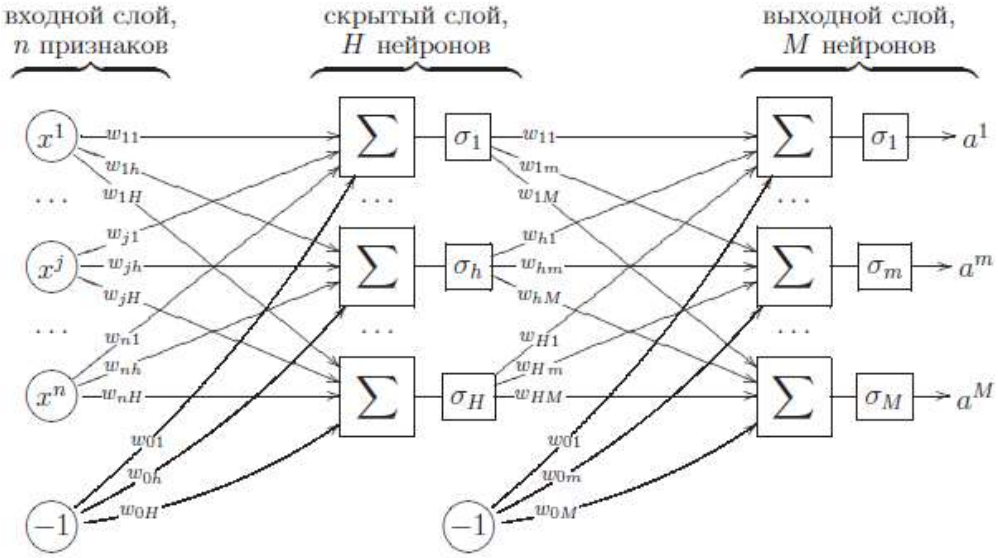
2 путь решения проблемы – построить конструкцию из нескольких нейронов. Например, исключающее ИЛИ можно построить, подав выходы И-нейрона и ИЛИ-нейрона на вход еще одному ИЛИ-нейрону,



$$x^1 \oplus x^2 = \left[(x^1 \vee x^2) - (x^1 \wedge x^2) - \tfrac{1}{2} > 0\right].$$

Этот принцип – основа построения многослойных нейронных сетей, которые уже напоминают естественные нейронные сети.

Многослойные нейронные сети.



На рисунке изображена двухслойная сеть, в которой n всех признаков подаются на вход всех нейронов первого стоя Н. затем их выходные значения подаются в слой М, который является выходным. Сеть может содержать любое количество слоев, все слои кроме выходного называются скрытыми.

Вычислительные возможности НС.

1. Любую Булеву функцию можно представить в виде двухслойной сети;
2. Двухслойная НС позволяет выделит в пространстве произвольный выпуклый многогранник. Трехсло
3. йная сеть позволяет аппроксимировать любые области с непрерывной границей, а также аппроксимировать любые непрерывные функции.
4. Двухслойная НС позволила ответит на вопрос – возможно ли произвольную непрерывную функцию n аргументов представить в виде суперпозиции функции меньшего количества аргументов;
5. Любую непрерывную функцию n переменных можно равномерно приблизить полиномом с любой степенью точности.

НС являются универсальным аппроксиматором функций. Возможности НС возрастают с увеличением числа слоев. Как правило, 2-3 слоев бывает достаточно для решения большинства задач.

Многослойные НС.

Метод обратного распространения ошибок.

Плюсы метода обратного распространения ошибок.

1. Достаточно высокая эффективность;
2. Легко реализуется на вычислительных устройствах с параллельной архитектурой;
3. Высокая степень общности позволяет записать алгоритм для любого числа слоев и размерности входов и выходов, любой функции потерь и активации.

Минусы метода обратного распространения

1. Не всегда сходится, часто, для улучшения сходимости применяются эвристические ухищрения;
2. Процесс градиентного спуска часто застревает в локальных минимумах функционала;
3. Приходится заранее фиксировать число нейронов скрытого слоя;
4. При слишком большом увеличении весов происходит переобучение;
5. При применении функций активации с горизонтальными асимптомами возможно попадание в состояние паралича сети.

Улучшение сходимости и качества градиентного обучения.

Нормализация данных.

Различия в масштабах измерения признаков может парализовать сеть. Поэтому необходима предварительная нормализация признаков:

$$x^j := \frac{x^j - x^j_{\min}}{x^j_{\max} - x^j_{\min}}, \quad \text{либо} \quad x^j := \frac{x^j - x^j_{\text{ср}}}{x^j_{\text{ско}}}, \quad j = 1, \ldots, n,$$

Где x^j_{min}, x^j_{max}, $x^j_{ср}$, $x^j_{ско}$ –соответственно минимальное, максимальное, среднее значения и среднеквадратичное отклонение признака x^j.

Выбор функции активации.

Сигмойдная функция $\sigma(z) = (1+e-z)-1$ применяется для классификации, она дает возможность оценить вероятность принадлежности объекта какому-либо классу, эффективность вычисления производной, ограниченность выходного значения. Применение нечетных функций, таких как $th(z) = 2\sigma(2z) – 1$ увеличивает скорость сходимости примерно в 1,5 раза.

Применение функционала среднеквадратичной невязки в задачах классификации несет скрытую опасность. Если значения меток классов {−1,+1} совпадают со значениями горизонтальных асимптот функции активации выходного слоя, то процесс оптимизации будет стремиться к неограниченному увеличению аргумента функции активации, то есть к параличу выходного нейрона. Значения меток должны классов должны находиться внутри области значений функции активации.

Выбор начального приближения**.**

Для исключения паралича сети синаптические веса должны активироваться небольшими по модулю значениями. Значения скалярных произведений не должны выпадать из рабочей зоны.

Для формирования начального приближения сначала настраиваются нейроны первого слоя по отдельности, как *H* однослойных персептронов. Затем настраиваются нейроны второго слоя, на вход которых подаются векторы значений первого слоя. Что бы сеть не выродилась,

нейроны первого слоя должны существенно отличаться. В идеале они должны приближать целевую зависимость, тогда нейроном второго слоя останется только усреднить результат первого слоя, сгладив ошибки. Для этого нейроны первого слоя нужно обучать на случайных подвыборках, либо подавать им на вход различные случайные подмножества признаков. При формировании начального приближения не требуется высокая точность, поэтому нейроны возможно обучать простыми градиентными методами.

Порядок предъявления объектов.

1. Метод перетасовки объектов. Смысл заключается в попеременном предъявлении объектов из разных классов, объекты одного класса с большей вероятностью окажутся схожими.
2. При условии, что массив данных не имеет выбросов, можно применить порядок предъявления, при котором в первую очередь предъявляются объекты на которых была допущена ошибка. Вероятность появления каждого объекта вычисляется в соответствии с величиной ошибки сети на данном объекте.
3. Задается порог ошибки на каком-либо объекте, если ошибка меньше порога, то вектор весов не модифицируется, в обратном случае выполняется обратный ход, вычисляется градиент и измеряются веса. Если объект уже неплохо классифицируется, то менять веса не нужно.

Сокращение весов

Добавляя штрафное слагаемое *Q(w)* к минимизируемому функционалу ограничивается рост абсолютных значений весов получаем:

$$Q_\tau(w) = Q(w) + \frac{\tau}{2}\|w\|^2.$$

Изменение функционала приводит к появлению аддитивной поправки у градиента:

$$\frac{\partial Q_\tau(w)}{\partial w} = \frac{\partial Q(w)}{\partial w} + \tau w.$$

При этом правило обновления весов принимает вид:

$$w := w(1 - \eta\tau) - \eta\frac{\partial Q(w)}{\partial w}.$$

Алгоритм заключается в появлении неотрицательного множества $(1-\eta\tau)$ приводящего к постоянному уменьшению весов. Отсюда и название – сокращение весов.

Метод легко реализуется, предотвращает паралич сети и повышает устойчивость. В конечном итоге сокращение весов способствует повышению обобщающей способности алгоритма и снижению риска переобучения.

Параметр τ позволяет найти компромисс между точностью настройки на конкретную выборку и устойчивостью весов.

Недостаток метода заключается в значительных затратах времени.

Сокращение весов уменьшает эффективную сложность сети, но число параметров остается прежним. Ресурс на вычисление используется неэффективно.

Выбор величины шага.

1. Градиентные методы сходятся при уменьшении шага вместе с ростом числа итераций;
2. метод скорейшего градиентного спуска приводит к выбору адаптивного шага η исходя из решения одномерной задачи минимизации

$$Q\left(w - \eta\frac{\partial Q}{\partial w}\right) \to \min_\eta.$$

Выбивание сети из локальных минимумов.

Суть заключается в том, чтобы при каждой стабилизации функционала ошибки осуществлять случайные модификации вектора весов в достаточно большом диапазоне значений и запуске градиентного спуска из новых точек. Этот метод называется – потряхиванием коэффициентов.

Выбор критерия останова.

Функционал Q оценивается приближенно, как экспоненциальное скользящее среднее ошибок Q_i, допущенных на объектах x_i, при этом наибольший вес получают объекты предъявленные последними.

Ранний останов.

Слишком глубокая оптимизация может привести к переобучению. Для предотвращения создается внешний критерий и, если он начинает возрастать, процесс обучения прекращается.

Выбор градиентного метода оптимизации.

1. При большом объеме выборки или требуется классификация, то можно использовать метод стахостического градиента с адаптивным шагом;
2. Диагональный метод Левенберга – Марквардта в несколько раз быстрее. Величина шага вычисляется индивидуально для каждого весового коэффициента, при этом используется только один диагональный элемент матрицы вторых производных:

$$\eta_{jh} = \frac{\eta}{\frac{\partial^2 Q}{\partial w_{jh}^2} + \mu},$$

Где η остается глобальным параметром темпа обучения, μ – новый параметр, предотвращающий обнуление знаменателя и неограниченное увеличение шага. $\eta/\ \mu$ – темп обучения на ровных участках функционала.

3. При небольшом объеме выборки и если решается задача регрессии, то лучше использовать адаптированные методы сопряженных элементов. Смысл в том, что объекты предъявляются не по одному, а пакетами. Состав пакета формируется случайно. Для каждого пакета минимизируемый функционал остается фиксированным, что позволяет использовать метод сопряженных градиентов.

Оптимизация структуры сети.

Выбор структуры сети является наиболее сложной проблемой. Существуют несколько стратегий поиска оптимальной структуры: постепенное наращивание, построение слишком сложной сети с последующим упрощением, поочередное наращивание и упрощение.

Проблема связана с переобучением и недообучением.

Выбор числа слоев.

Чем больше слоев, тем более богатый функционал сети, однако тем хуже сходятся градиентные методы и сложнее обучение.

Выбор числа нейронов в скрытом слое.

1. Визуальный. Если кривая регрессии слишком сглажена, то сеть слишком проста, нужно добавлять число нейронов. Если колебания регрессии слишком сильные, есть выбросы, то число нейронов нужно сократить. Этот метод подходит для задач с небольшим количеством выбросов.
2. Оптимизация по внешнему критерию. По критерию скользящего контроля или средней ошибки на независимой контрольной выборке. Недостаток метода в том, что приходится несколько раз строить сеть при различном количестве нейронов.

Динамическое добавление нейронов.

Сначала сеть обучается при заведомо недостаточном количестве нейронов скрытого слоя, до того, как ошибка начнет убывать. Тогда добавляются нейроны. Веса новых связей инициализируются небольшими случайными числами.

После добавления нейронов ошибка, как правило, возрастает, затем быстро снижается.

При постепенном наращивании нейронов нужно наблюдать за динамикой какого-либо внешнего критерия. Прохождение значения $Q(X^k)$ через минимум является надежным критерием останова, так как указывает на переобученность по причине усложнения сети.

Удаление избыточных связей.

Метод оптимального усечения сетиудаляет связи к которым наименее чувтвителен функционал Q. Уменьшение числа весов снижает склонность к переобучению.

Метод основан на предположении, что после стабилизации функционала ошибки Qвектор весов находится в локальном минимуме, где функционал может быть аппроксимирован квадратичной формой:

$$Q(w+\delta) = Q(w) + \tfrac{1}{2}\delta^{\mathrm{T}} H(w)\delta + o(\|\delta\|^2),$$

$$H(w) = \left(\frac{\partial^2 Q(w)}{\partial w_{jh}\partial w_{j'h'}}\right)$$

гессиан, матрица вторых производных.

Предполагается, что диагональные элементы доминируют в гессиане, а остальными частными производными можно пренебречь, приняв их равными нулю. Это предположение вводится для того, чтобы избежать трудоемкого вычисления всего гессиана.

Если гессиан диагонален, то:

$$\delta^{\mathrm{T}} H(w)\delta = \sum_{j=0}^{J}\sum_{h=1}^{H} \delta_{jh}^2 \frac{\partial^2 Q(w)}{\partial w_{jh}^2}.$$

Обнуление веса w_{jh} эквивалентно условию $w_{jh+}\ \delta jh=0$. Введем величину значимости синаптической связи, равную изменению функционала $Q(w)$ при обнулении веса:

$$S_{jh} = w_{jh}^2 \frac{\partial^2 Q(w)}{\partial w_{jh}^2}.$$

Смысл метода заключается в удалении из сети d синапсов, соответствующих наименьшим значениям Sjh. Здесь d – параметр метода настройки. После удаления- цикл итераций до стабилизации функционала Q. Процесс упрощения сети прекращается, когда внутренний критерий стабилизируется, либо, когда внешний начинает возрастать.

Основная часть.

Описание объекта.

Для разработки алгоритма прогнозирования выбран технологический процесс производства черновой меди металлургическим цехом филиала «Производство полиметаллов» АО «Уралэлектромедь» г.Кировград. Металлургический цех работает в режиме непрерывного производства в трехсменном графике, однако для прогнозирования мы располагаем только суточными показателями. Ремонтных смен нет, оборудование взаимозаменяемое, в течение года цех 2 раза останавливается на капитальный ремонт в конце августа и сентября. Для прогнозирования будут

использованы следующие суточные показатели: расход природного газа [м^3]; расход воздуха высокого давления [м^3]; расход питательной воды [м^3]; потребление электроэнергии [кВт*ч]; температура воздуха [°C]; выработка черновой меди [тн], архив за 2020г. (365 дней).

Постановка задачи.

Требуется по набору суточных данных о работе предприятия в течение заданного периода времени, обычно не менее одного года, составить нейросетевую модель суточного потребления электроэнергии с целью использования при прогнозировании.

Для этого необходимо решить следующие задачи:

1) чтение требуемых данных из файла в формате Excel и формирование массива входных данных и вектора выходной переменной, которой является величина суточного потребления электроэнергии; 2) просмотр таблицы входных данных и построение графика вектора значений суточного потребления электроэнергии;

3) нормирование значений и удаление выбросов в данных;

4) формирование обучающего, контрольного и тестового множеств;

5) составление описания и задание параметров модели ИНС и обучение;

6) проверочные расчеты по модели ИНС для обучающего, тестового и контрольного множеств с целью определения погрешности модели;

7) оценка погрешности модели ИНС для выборочного диапазона входных данных;

8) построение графиков действительных и прогнозных значений потребления электроэнергии и построение гистрограмм (распределений) погрешности прогнозирования;

9) запись параметров модели в файл с целью дальнейшего использования для прогнозирования.

Описание алгоритма.

Описание выполнено по модели ИНС в пакете *Matlab*.

1. Чтение требуемых данных из файла в формате Excel и формирование массива входных данных и вектора выходной переменной, которой является величина суточного потребления электроэнергии.

```
clear
%clf_all; %
global tableData
LoadDataFlag=1;
if LoadDataFlag
 cd c:/
 cd ./ann9
%Matlab
        N = xlsread('month_data_1', 'a2:k1117'); % массив данных из таблицы
        % Octave
   pkg load io
   pkg load nnet
          N = xlsread('month_data_1.xlsx', 'a2:k1117'); % массив данных из таблицы
end
disp('Размер данных в таблице Excel');      %На русском только Matlab,
 [n,m]=size(N)
```

2. Просмотр полной таблицы входных данных

```
% построение окна в таблицей данных для просмотра
% только Matlab
f=figure('Position', [100 100 752 500]);
```

```
t = uitable('Parent', f, 'Position', [25 25 700 200]);
t1 = uitable('Parent', f, 'Position', [25 250 700 200]); set(t, 'Data', N);
name_str=[]; %текстовая строка номеров столцов
for i=1:m
        name_str(i)=sprintf('%s', i);
end
        set(t, 'ColumnName', {name_str}); %отобразить номера столбцов в таблице % set(t,
        'ColumnName', {'Day', 'Smena', ' ', 'W', ...
% 'X2',});
```

3. Определение размера обучающего, контрольного и тестового множеств

```
Ndata=n; disp('Количество строк данных тестового множества');
Ntest=fix(n/3) %на больших объемах данных (более 10000) может не работать (n/1, n/2)
 %тогда задавать значение явно
disp('Количество строк данных контрольного множества'); NChk=fix(Ntest/2) %1/2 Ntest

NX=6; %количество входных переменных !!!
N_hist_array=2; % количество измерений назад для прогноза !!!
disp('Количество строк последовательности данных');
NtestSer=NChk %количество точек прогноза для последовательности данных (за месяц)
Ninput=NX+N_hist_array;
Inputs(:,:)=zeros(Ninput, Ndata);

%Названия и номер столбцов данных в таблице для справки, необходимо при указании номера
столбца, из которого выполняется запись
% День(1),  Расход ПГ(м3) (2),  Расход ВВД, (м3)(3),  Удельн. ЭЭ, (кВт*ч/тн(4)), Расход ПТВ
(м3)(5),  Удельн. ПТВ (м3/тн)(6),
% Расход эл.эн, (кВт*час)(7)          Удельн. эл.эн, (кВт*час/тн)(8)       темп НВ (С)(9),
% Удельн. ПГ, (м3/тн)(10)              Выработка, (тн)(11)
```

4. Формирование массива входных факторов Inputs путем выборки только необходимых столбцов из полного массива N и нормирование значений, и удаление выбросов в данных в процедуре *breakdata*;

```
Lambda=3.0; %показатель нормального распределения для уровня значимости 99,5%
viewhist=0;
Inputs(1,:)=breakdata(N(:,1)', Lambda, 'Days', viewhist);
Inputs(2,:)=breakdata(N(:,2)', Lambda, 'PG', viewhist);
Inputs(3,:)=breakdata(N(:,3)', Lambda, 'VVD', viewhist);
Inputs(4,:)=breakdata(N(:,5)', Lambda, 'PTB', viewhist);
Inputs(5,:)=breakdata(N(:,9)', Lambda, 'T, C', viewhist);
Inputs(6,:)=breakdata(N(:,11)', Lambda, 'P, tonn', viewhist);
```

Описание процедуры *breakdata* – смотри текст программы.

5. Формирование вектора выходной переменной Output и построение графиков

```
Output=breakdata(N(:,7)', Lambda, 'A', viewhist); %целевое множество, потребление электроэнергии Output_all=Output;
tdays=1:Ndata;
%plot(tdays, Inputs(1,:), '-rs'); %plot(tdays, Inputs(1,:), '-r');
figure; plot(tdays, Output, '-r'); title('W target');
```

6. Вычисляем базисные значения и нормируем данные for j=1:NX
```
 InpBase(j)=max(abs(Inputs(j,:)));
 InpBase(j)=ceil(InpBase(j)); Inputs(j,:)=Inputs(j,:)./InpBase(j);
end

InpBase(NX+1)=max(abs(Output));
InpBase(NX+1)=ceil(InpBase(NX+1));
Output=Output./InpBase(NX+1);
Output_all_oe=Output;

disp('Базисное значение расхода энергии'); InpBase(NX+1)
disp('Базисные значения входных переменных');
Base=cell(NX,2); col2=1; for j=1:NX
 InpBase(j);
 Base{j}=j; Base{col2*NX+j}=InpBase(j); end
```

7. Создать вектор значений энергии и векторы историй от 1 до 5 суток.

Пока не рассматриваем.

Это алгоритм расширения набора данных путем добаления данных за предыдущие сутки в массив факторов для прогноза.

8. Формирование обучающего, контрольного и тестового множеств.

```
Inp_all=Inputs;
%данные последовательные для тестирования реальной последовательности
TestInp1=zeros(Ninput,NtestSer);
TestOut1=zeros(1,NtestSer);
TimeOffset=5;
TestInp1(:,1:NtestSer)=Inputs(:,TimeOffset:TimeOffset+NtestSer-1);
TestOut1(1,1:NtestSer)=Output(1,TimeOffset:TimeOffset+NtestSer-1);
Inputs1(:,1:Ndata-NtestSer-TimeOffset)=Inputs(:,TimeOffset+NtestSer:Ndata-1);
Output1(1,1:Ndata-NtestSer-TimeOffset)=Output(1,TimeOffset+NtestSer:Ndata-1);

% выборка случайная
TestInp=zeros(Ninput,Ntest);
TestOut=zeros(1,Ntest);

% как связать данные после удаления данных случайным образом?
% Перенос образов из обучающего в тестовое множество for k=1:Ntest
 Index=1+int16(rand*(Ndata-k)); % Случайное целое число от 1 до Ndata-k
 EdInp(:,k)=Inputs(:,Index); % Данные по строкам
 EdOut(:,k)=Output(:,Index); % Данные по строкам
 %Заменим взятые значения "пустыми"
 Inputs(:,Index)=[]; Output(:,Index)=[];
end

%создание контрольного множества, которое должно быть
%представлено структурой VV, поля P и T входных и выходных значений [11].

% выборка случайная
```

```
ChInp=zeros(Ninput,NChk); % Создание нулевой матрицы входов контрольного множества
ChOut=zeros(1,NChk); % Создание нулевой матрицы выхода контрольного множества for
k=1:NChk
Index=1+int16(rand*(Ntest-k)); % Случайное целое число от 1 до Ndata-k
% Запись элементов контрольного множества
ChInp(:,k)=EdInp(:,Index);
ChOut(:,k)=EdOut(:,Index);
% Удаление элементов контрольного множества из обучающего
EdInp(:,Index)=[]; EdOut(:,Index)=[];
end
```

9. Создание структуры данных в соответствии с описанием функции и построение ИНС

```
VV=struct('P',ChInp,'T',ChOut); % Создание структуры контрольного множества
%NEWFF(P,T,S,TF,BTF,BLF,PF,IPF,OPF,DDF) takes,
% P - RxQ1 matrix of Q1 representative R-element input vectors.
% T - SNxQ2 matrix of Q2 representative SN-element target vectors.
% Si - Sizes of N-1 hidden layers, S1 to S(N-1), default = [].
% (Output layer size SN is determined from T.)
% TFi - Transfer function of ith layer. Default is 'tansig' for % hidden layers, and 'purelin' for output
layer.
% BTF - Backprop network training function, default = 'trainlm'.
% BLF - Backprop weight/bias learning function, default = 'learngdm'.
% PF - Performance function, default = 'mse'.
% IPF - Row cell array of input processing functions.
% Default is {'fixunknowns','remconstantrows','mapminmax'}.
% OPF - Row cell array of output processing functions.
% Default is {'remconstantrows','mapminmax'}.
% DDF - Data division function, default = 'dividerand'; % and returns an N layer feed-forward back-
prop network. for j=1:Ninput PInp(j,:)=[0 1]; end

% Создание многослойной нейронной сети прямого распространения
%Matlab
Net = newff(PInp,[15 1],{'logsig' 'logsig'},'trainlm','learngdm','sse');
%Octave
Net = newff(PInp,[15 1],{'logsig' 'logsig'},'trainlm','learngdm','mse');
% 15 – число нейронов в первом слое
% 1 – число нейронов во втором слое
% logsig – функция активации
% 'trainlm','learngdm','ss – см. описание функции
Net.trainParam.epochs = 1000; % Максимальное число циклов обучения
Net.trainParam.goal = 0.001; % Целевое значение функции ошибки
%Octave
Net.trainParam.show = 25;          %Отображение через 25 эпох
Net.trainParam.time = 5*60;        %Время счета, секунд
```

10. Обучение ИНС

```
Net = train(Net,EdInp,EdOut,[],[],VV); % Обучение нейронной сети
```

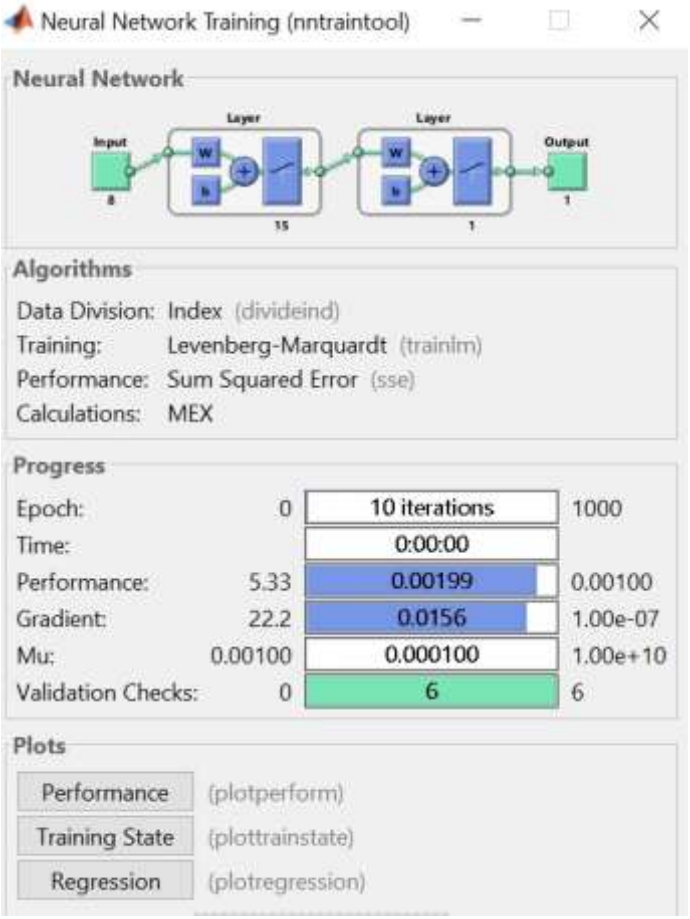
11. Получение результатов расчета по модели ИНС, определение абсолютной погрешности deflection, и среднеквадратичной ошибки rezsim=sim(Net,EdInp); deflection=(rezsim-EdOut); % Отклонения на обучающем множестве mistake(1)=sqrt(sum(deflection.^2)/length(EdOut)); % СКО stdW=std(deflection);

```
rezsimCh=sim(Net,VV.P);
deflection1=(rezsimCh-VV.T); % Отклонения на контрольном множестве mistake(2)= sqrt(sum(deflection1.^2)/length(VV.T)); % СКО stdWch=std(deflection1);
%clc % Очистка экрана перед выводом результатов
fprintf('%-60s %7.4f процентов\n%-60s %7.4f процентов\n\n',...
'Среднее квадратическое отклонение на обучающем множестве:',100.*mistake(1),... 'Среднее квадратическое отклонение на контрольном множестве:',100.*mistake(2)...
)
% Вывод результатов

tm1=1:length(rezsim);
figure; plot(tm1, 0, tm1, rezsim, 'b', tm1, EdOut, 'r'); title('ANN simulation education set'); figure; hist(deflection, 10); stdWstr=num2str(stdW, '%7.3f'); title(stdWstr); tm1=1:length(rezsimCh);
figure; plot(tm1, rezsimCh, 'b', tm1, VV.T, 'r'); title('ANN simulation check set'); figure; hist(deflection1, 10); stdWstr=num2str(stdWch, '%7.3f'); title(stdWstr);


%последовательные (рабочие) данные из массива данных

rezsimSer=sim(Net,TestInp1); %
deflectionSer=(rezsimSer-TestOut1); % Вектор-строка отклонений
mistake(3)=sqrt(sum(deflectionSer.^2)/length(rezsimSer)) % Целевая функция ошибки
stdWSer=std(deflectionSer) tm1=1:length(rezsimSer);
figure; plot(tm1, rezsimSer, 'b', tm1, TestOut1, 'r'); title('ANN simulation seriasly data'); figure;
plot(tm1, deflection, 'b'); title('ANN simulation. Deflection seriasly data'); figure; hist(deflection, 10);
stdWstr=num2str(stdWSer, '%7.3f'); title(stdWstr);

%вывод в таблицу на экран
set(t1, 'Data', mistake); set(t1, 'ColumnName', {'СКО обуч', 'СКО контр', 'СКО послед', ...
 });
%запись в таблицу Excel
key=input('Сохранить модель в файл? Y/N [Y]: ', 's'); if isempty(key)
 key = 'Y'; end
if key=='Y' || key=='y'

x1=mistake; x2=InpBase(1:NX); x3=InpBase(NX+1); rowmax=25; colmax=20; C = cell(rowmax, colmax);
%1 4 индекс в таблице
%2 5
%3 6
%запись в столбцы через столбец, 0 2 4 6

clk = clock; C{1} = date;
C{2} ='hour'; C{3} = clk(4); C{4} ='minute'; C{5} = clk(5);
if 0 % убрано, т. к. запись в строках 224...226 for k = 1:size(s1'), s1(k) C{k} = s1(k); end end %
ncol=2;
```

```
C{ncol*rowmax+1}='СКО модели';
for k = ncol*rowmax+2:ncol*rowmax+1+size(x1'),
 C{k} = x1(k-ncol*rowmax-1);
end ncol=4;
C{ncol*rowmax+1}='Хбаз';
for k = ncol*rowmax+2:ncol*rowmax+1+size(x2'),
 C{k} = x2(k-ncol*rowmax-1);
end ncol=6;
C{ncol*rowmax+1}='Wбаз';
for k = ncol*rowmax+2:ncol*rowmax+1+size(x3'),
 C{k} = x3(k-ncol*rowmax-1);
end

s = xlswrite('model_param.xls', C, 'Neural Network Parameters', 'A1'); if s==1
 disp('Результаты прогноза сохранены в файл model_param.xls'); else
 disp('Сохранение результатов прогноза не выполнено (возм., открыт файл для просмотра)'); end

xbase=InpBase(1:NX); ybase=InpBase(NX+1); save('model_param', 'NX', 'mistake', 'xbase', 'ybase',...
 'Inp_all', 'Net','Output_all','Output_all_oe' );
disp('Модель и результаты сохранены в файлы model_param.xls и model_param.mat'); else
disp('Сохранение модели и результатов не выполнено'); end % if key=='Y'

end
```

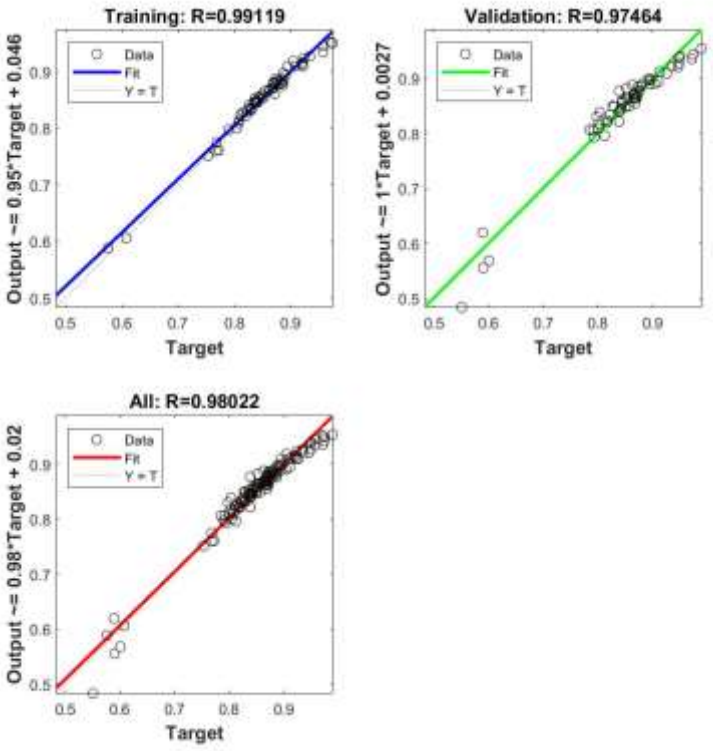
Описание результатов.

Обучение нейронной сети.



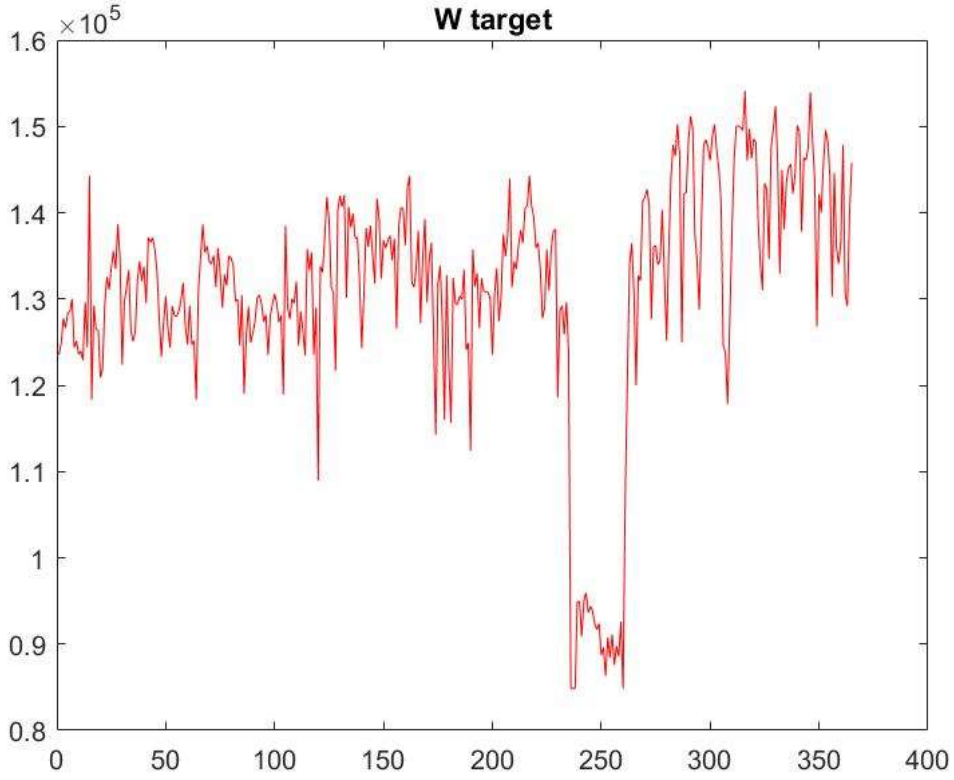
Сформирована двухслойная нейронная сеть. В первом слое 15 нейронов, скрытый слой содержит 1 нейрон. Количество переменных – 8. Алгоритм обучения прошел за 10 итераций.

Коэффициенты корреляции.

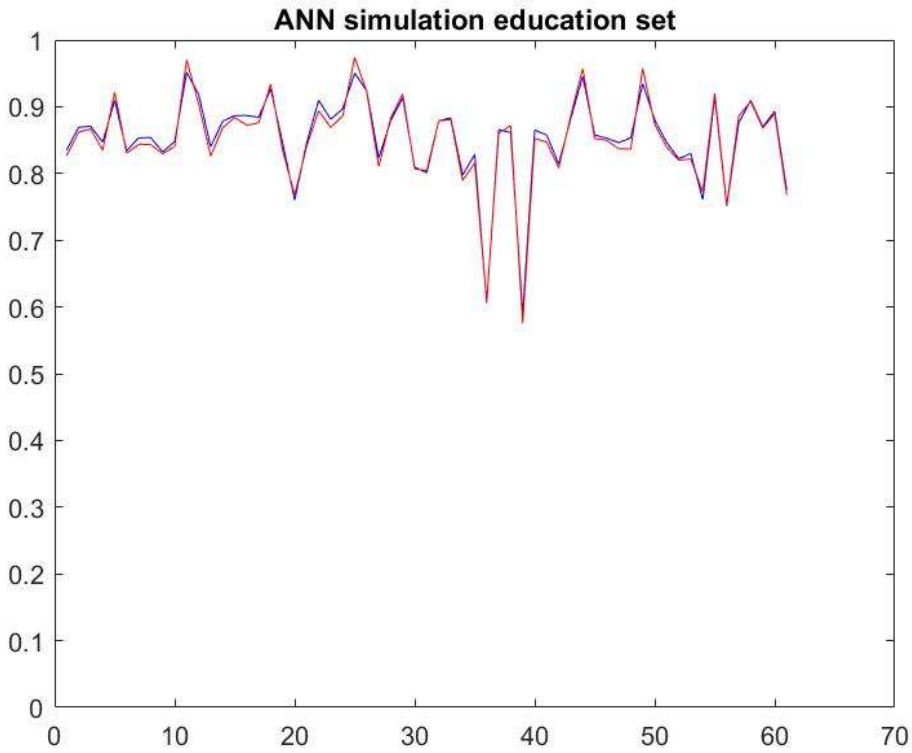


Коэффициент корреляции обучающего множества составил 0,99, контрольном множестве - 0,97, выходном множестве – 0,98.

График изменения целевого множества.

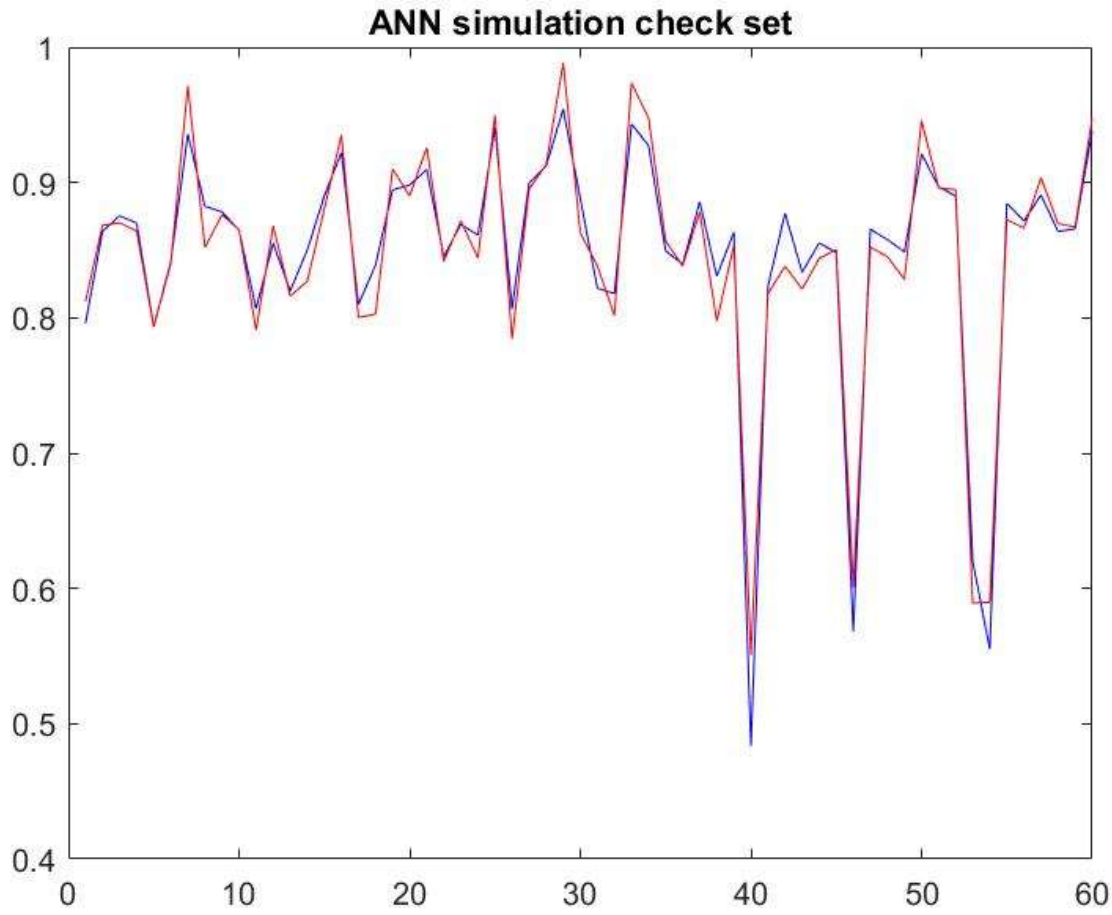


Моделирование процесса обучения.

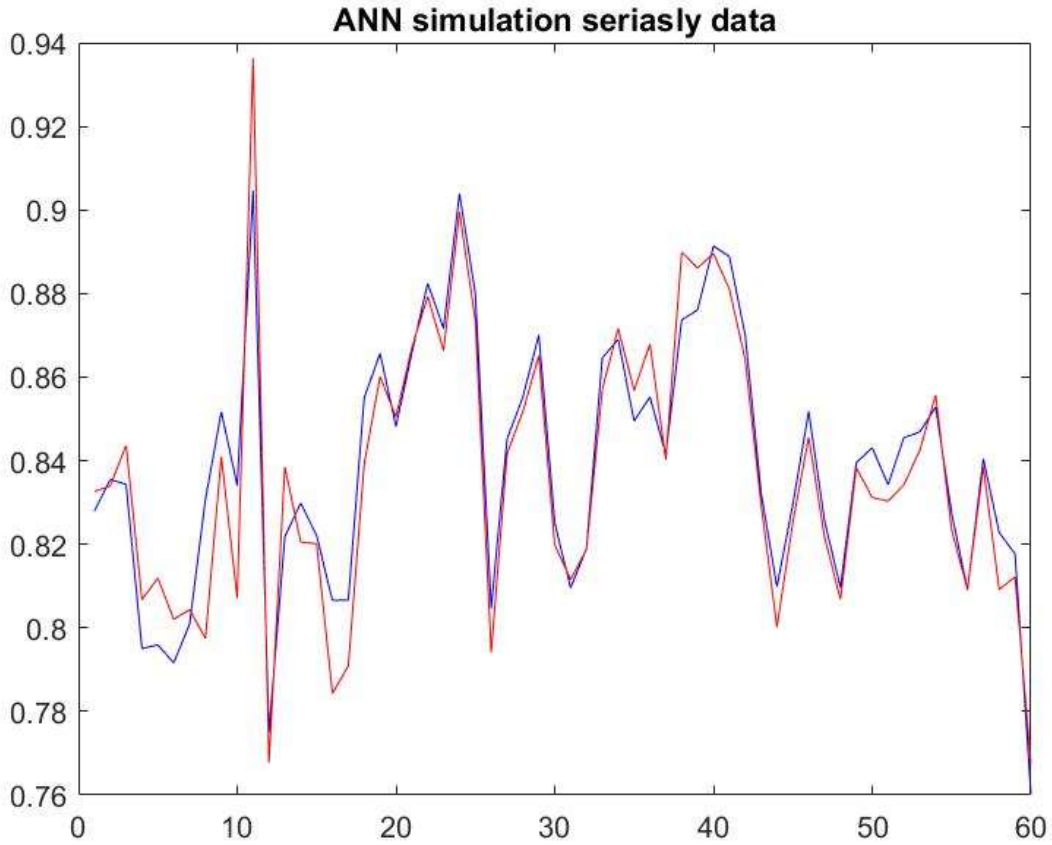


.

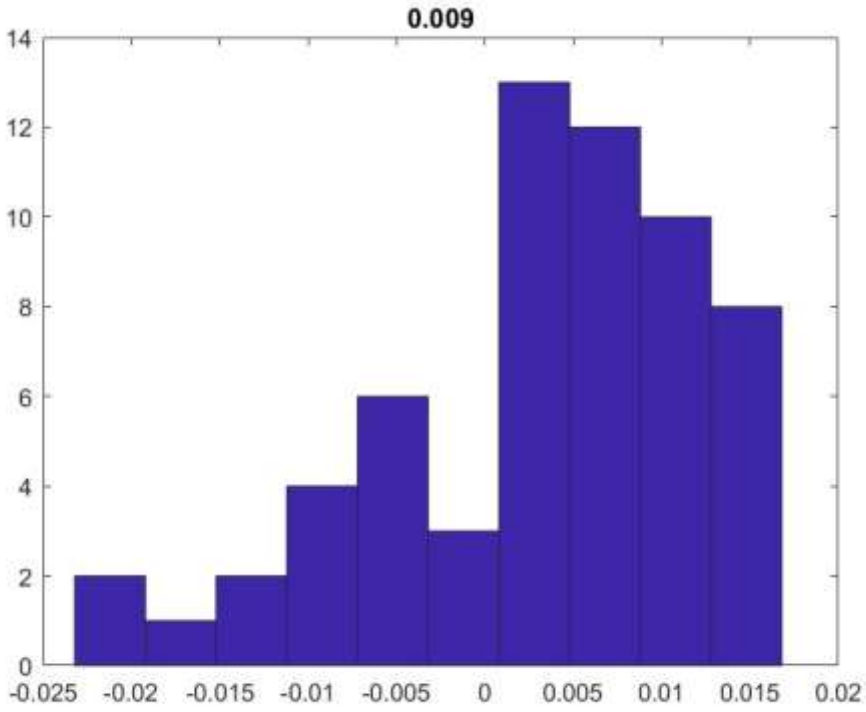
Моделирование процесса проверки.

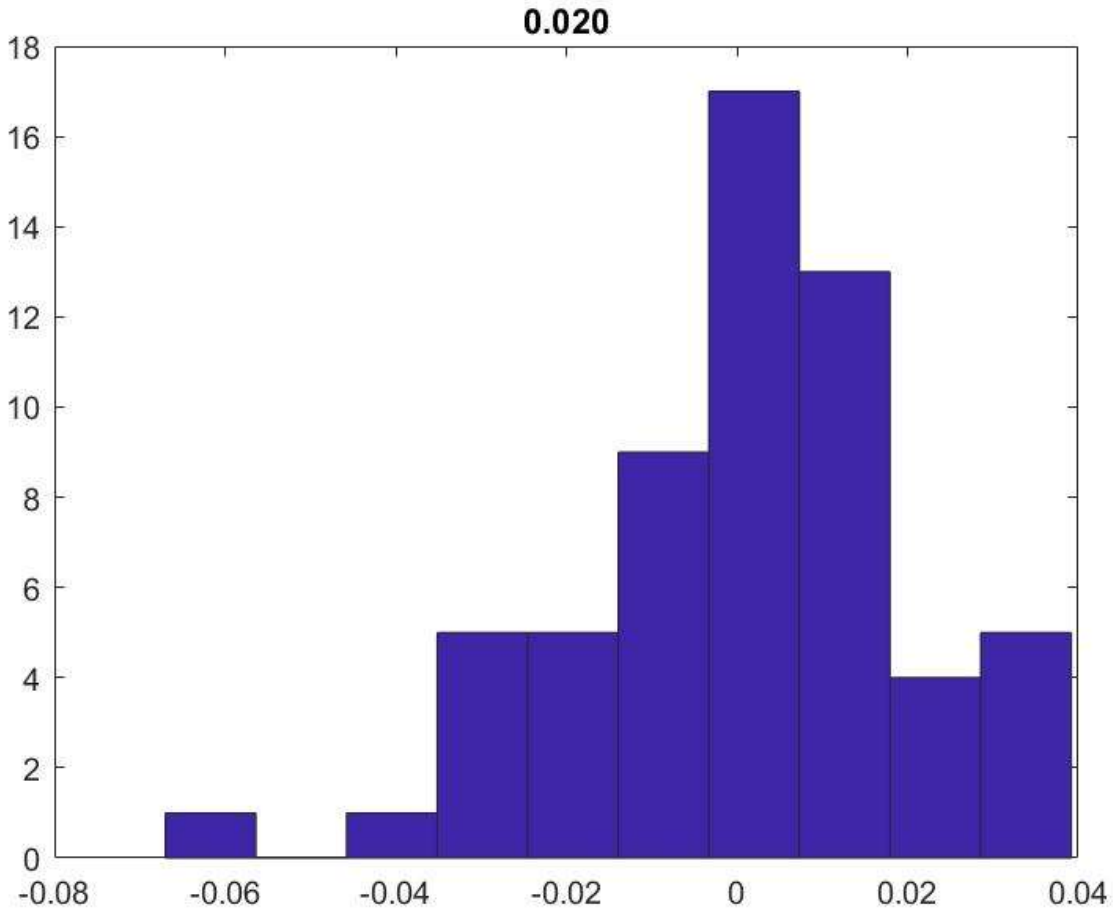


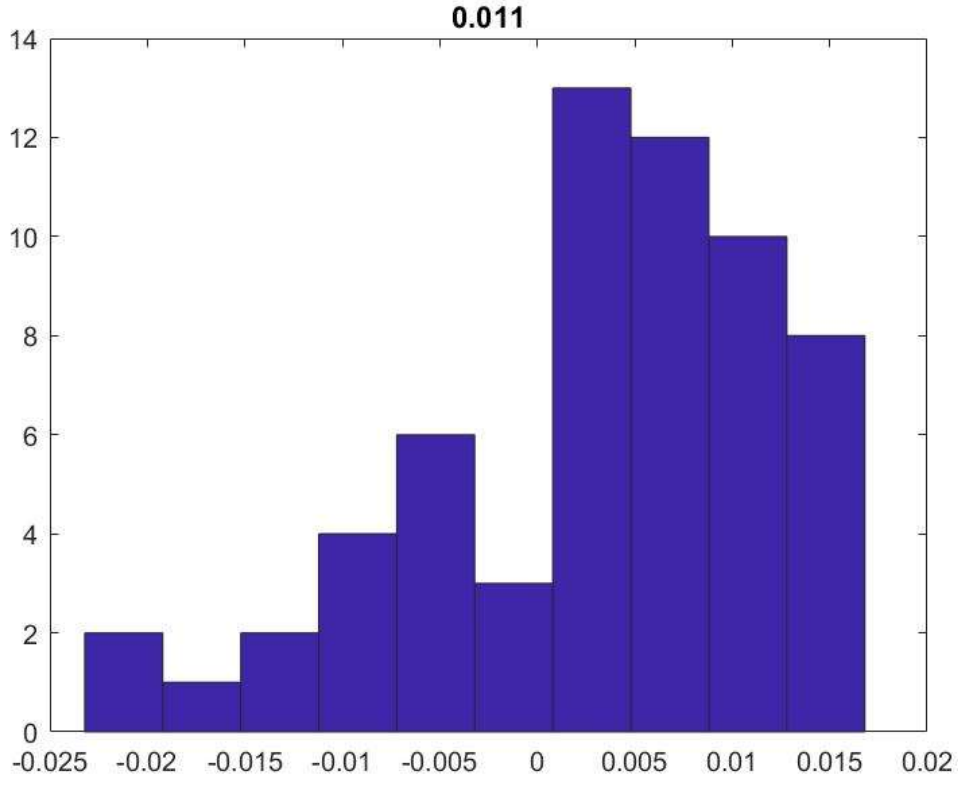
Серийные данные моделирования.



Распределение погрешности прогнозирования.




0.020
18
16
14
12
10
8
6
4
2
0
-0.08
-0.06
-0.04
-0.02
0
0.02
0.04


0.011
14
12
10
8
6
4
2
0
-0.025
-0.02
-0.015
-0.01
-0.005
0
0.005
0.01
0.015
0.02

В конце алгоритма результат прогнозирования записан в файл EXEL с целью дальнейшего прогнозирования.

28-Apr-2021		СКО модели		Хбаз		Wбаз
hour		0,009757		365		154117
20		0,019592		42511		
minute		0,01068		2224		
53				1222		
				39		
				4624		

Заключение.

В ходе работы была составлена нейросетевая модель суточного потребления электроэнергии для ее использования при прогнозировании.

Для этого выполнены следующие действия:

1. чтение требуемых данных из файла в формате Excel и формирование массива входных данных и вектора выходной переменной;
2. просмотр таблицы входных данных, исправление нечитаемых входных данных, и построение графика вектора значений суточного потребления электроэнергии;
3. нормирование значений и удаление выбросов в данных;
4. формирование обучающего, контрольного и тестового множеств;
5. составление описания и задание параметров модели ИНС и обучение;
6. проверочные расчеты по модели ИНС для обучающего, тестового и контрольного множеств с целью определения погрешности модели;
7. оценка погрешности модели ИНС для выборочного диапазона входных данных;
8. построение графиков действительных и прогнозных значений потребления электроэнергии и построение гистрограмм (распределений) погрешности прогнозирования;
9. запись параметров модели в файл с целью дальнейшего использования для прогнозирования.

Список использованной литературы

1. Головко В. А. Нейронные сети: обучение, организация и применение. М.: ИПР-ЖР, 2001.
2. Колмогоров А. Н. О представлении непрерывных функций нескольких переменных в виде суперпозиции непрерывных функций одного переменного // Докл. АН СССР. 1958. Т. 114, № 5._ С. 953–956.
3. Нейроинформатика / А. Н. Горбань, В.Л. Дунин-Барковский, А.Н. Кирдин, Е.М. Миркес, А.Ю. Новоходько, Д.А. Россиев, С.А. Терехов и др._ Новосибирск: Наука, 1998. - С. 296.
4. Тихонов А. Н., Арсенин В. Я. Методы решения некорректных задач. М.: Наука, 1986.
5. Яблонский С. В. Введение в дискретную математику. М.: Наука, 1986.