МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теории функций и стохастического анализа

# НЕЙРОННЫЕ СЕТИ В ЗАДАЧАХ УПРАВЛЕНИЯ РОБОТОМ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 412 группы
направления 01.03.02 — Прикладная математика и информатика
механико-математического факультета
Терехова Павла Олеговича

Научный руководитель
доцент, к. ф.-м. н. _____ Л. В. Борисова

Заведующий кафедрой
д. ф.-м. н., доцент _____ С. П. Сидоров

Саратов 2019

# ВВЕДЕНИЕ

**Актуальность** работы обусловлена огромной популярностью нейронных сетей. Они получили широкое распространение. Создание нейронных сетей вызвано попытками понять принципы работы человеческого мозга и, без сомнения, это будет влиять и на дальнейшее их развитие. Однако, в сравнении с человеческим мозгом нейронная сеть сегодня представляют собой весьма упрощенную модель, но несмотря на это весьма успешно используются при решении самых различных задач: автоматизация процесса классификации, автоматизация прогнозирования, автоматизация процесса распознавания, автоматизация процесса принятия решений; управление, кодирование и декодирование информации; аппроксимация зависимостей и др. Хотя решение на основе нейронных сетей может выглядеть и вести себя как обычное программное обеспечение, они различны в принципе, поскольку большинство реализаций на основе нейронных сетей «обучается», а «не программируется»: сеть учиться выполнять задачу, а не программируется непосредственно.

**Целью бакалаврской работы**, является создание автономной системы управления искусственным агентом, а также самого управляемого агента.

**Объектом** исследования являются виды и методы обучения искусственных нейронных сетей, для решения задач распознавания и классификации объектов.

**Предмет исследования** - особенности реализации методов обучения нейронных сетей.

Для достижения поставленных целей в работе необходимо создать робота и программное обеспечение для управления:

— придумать и реализовать схему механизма;
— написать программное обеспечение для каждого компонента схемы;
— написать управляющий алгоритм;
— установить устойчивую связь между всеми комплектующими механизма;
— создать и настроить безопасную, устойчивую локальную сеть для взаимодействия механизма и управляющего алгоритма;

Для этого необходимо решить следующие **задачи**:

— определить необходимые комплектующие механизма;
— определить необходимые для реализации технологии;
— создать и обучить нейронную сеть;
— придумать концепцию расчета траектории движения в неопределенных условиях;

**Теоретической основой** послужила совокупность научных работ западных ученых и инженеров конца 20 начала 21 века.

— F.Rosenblatt: The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain (1958)
— Gradient-Based Learning Applied to Document Recognition by Yann LeCun, Leon Botton, Yoshua Bengio and Patrick Haffner
— Extracting and Composing Robust Features with Denoising Autoencoders. Pascal Vincent, Hugo Larochelle, Yoshua Bengio, Pierre-Antoine Manzagol, Universite de Montreal, Dept. IRO, CP 6128, Succ. Centre-Ville, Montral, Qubec, H3C 3J7, Canada
— Auto-Encoding Variational Bayes. Diederik P. Kingma , Max Welling Machine Learning Group Universiteit van Amsterdam. arXiv:1312.6114v10 [stat.ML] 1 May 2014.

**Основное содержание работы**

Выпускная квалификационная работа состоит из введения, трех теоретических разделов и одного практического, заключения, списка использованных источников и четырех приложений.

**Введение** содержит основные положения: обоснование актуальности темы работы, формулируется цель, объект и предмет исследования.

В **первом** разделе, рассматриваются основные понятия связанные с искусственной нейронной сетью:

— определение искусственной нейронной сети;
— некоторых ее элементов, в частности нейронов;
— определение функции активации;
— определение некоторых видов нейронных сетей;

Определяется понятие искусственного нейрона, сумматора, взвешенной суммы, веса связи и тд. Рассматривается *сигмоидальная функция активации* как важный элемент в структуре нейронной сети. Определяются ее достоин-

ства и недостатки. Вычисляется ее производная.

Во **втором** разделе, рассматриваются основные понятия связанные с обучением искусственных нейронных сетей:

— определение обучения с учителем;

— определение обучения без учителя;

— рассматривается один из наиболее популярных методов обучения сетей, а именно метод обратного распространения ошибки;

— определяется алгоритм метода обратного распространения ошибки;

*Обучение с учителем* — один из способов машинного обучения, в ходе которого испытуемая система принудительно обучается с помощью примеров «объект-ответ». Между входами и ожидаемыми выходами (объект-ответ) может существовать некоторая зависимость, но она неизвестна. Известна только конечная совокупность - пары «объект-ответ», называемая обучающей выборкой. На основе этих данных требуется восстановить зависимость (построить модель отношений объект-ответ, пригодных для прогнозирования), то есть построить алгоритм, способный для любого объекта выдать достаточно точный ответ. Для измерения точности ответов, может вводиться функционал качества.

В данном случае человек (или компьютер) выступает в качестве эксперта, который формирует так называемое «обучающее множество», т.е. набор примеров и правильных ответов с экспертной точки зрения. Далее человек передает полученное множество с набором примеров, которые обозначим как $X$ и решений, которые обозначим как $Y$ в некий алгоритм, задача которого - найти некоторую функцию $f(X)$, преобразующую множество $X$ в множество $Y$

Далее, используя найденную функцию, алгоритм пытается найти ответ для примера, которого не было в обучающем множестве.

*Обучение без учителя* - отличается тем, что разметки для данных в этом случае нет. От этого образуются сразу несколько особенностей — во-первых это возможность использования несопоставимо больших объёмов данных, поскольку их не нужно будет размечать руками для обучения, а во-вторых это неясность измерения качества методов, из-за отсутствия таких же прямолинейных и интуитивно понятных метрик, как в задачах обучения

с учителем.

*Метод обратного распространения ошибки* - это итеративные градиентный алгоритм, который используется с целью минимизации ошибки работы многослойного прецептрона или, иными словами, искусственной нейронной сети и получения желаемого результата.

Использует обучение с учителем, для него требуется обучающее множество с заранее известными правильными ответами. Вводится мера ошибки, которая определяет, насколько сильно выходные значения сети отличаются от правильных ответов. Затем мера ошибки минимизируется путем изменения значений весов связей нейронной сети с помощью метода градиентного спуска.

Основная идея этого метода заключается в распространении сигналов ошибки от выходов сети к ее входам, в направлении, обратном, прямому распространению сигналов в обычном режиме работы сети. Для возможности применения данного метода, функция активации нейронов должна быть дифференцируемой.

Для того чтобы оценить, насколько сильно каждый вес влияет на выходное значение, рассчитываются частные производные ошибки по весам. Затем производится изменение весов на небольшие значение с учетом градиента. Так повторяется до тех пор, пока ошибка на выходе не сократится до допустимых значений. Начальные значения весов нейронов в сети задаются случайным образом.

В глубокой нейронной сети с несколькими скрытыми слоями производится расчет ошибки, которая передается от одного слоя к другому. На первом этапе рассчитывается значение ошибки на выходе нейронной сети, для которого мы знаем правильные ответы. Затем рассчитывается ошибка на входе в выходной слой сети, которая будет использоваться как ошибка на выходе скрытого слоя. Таким способом расчет продолжается до того момента, когда будет известна ошибка на входном слое. Именно поэтому алгоритм имеет название обратное распространение ошибки. Далее опишем алгоритм метода.

Определим обозначения:

1. $w_{ij}$ - вес связи соединяющем $i$-й и $j$-й узлы;

2. $\triangle w_{ij}$ - поправка;
3. $x_i$ - множество входных сигналов;
4. $O_i$ - множество выходных сигналов;
5. $Outputs$ - множество внутренних узлов;
6. $\delta_j = O_j(1 - O_j)(t_j - O_j)$ - поправка, вычисленная для нейрона на следующем уровне;

Сам алгоритм:

1. инициализировать $w_{ij}$ маленькими случайными значениями $\triangle w_{ij} = 0$;
2. повторить $N$ раз:

   Для всех $d$ от 1 до $m$:
   - на вход сети подать $x_i^d$ и подсчитать выходы $O_i$ для каждого нейрона;
   - для всех $k \in Outputs$ подсчитать $\delta_j = O_j(1 - O_j)(t_j - O_j)$;
   - для каждого слоя $L$ начиная с предпоследнего
     для каждого нейрона $j$ слоя $L$ вычислить:
     $\delta_j = O_j(1 - O_j) \sum_{k \in Outputs(j)} \delta_k w_{jk}$
   - для каждой связи сети $i, j$ вычислить:
     $\triangle w_{ij}(n) = \alpha \triangle w_{ij}(n-1) + (1 - \alpha)\eta\delta_j O_j$
     $w_{ij}(n) = w_{ij}(n-1) + \triangle w_{ij}(n)$
3. вывести значение $w_{ij}$;

В **третьем** разделе более подробно рассматривается такой вид нейронных сетей, как сверточные нейронные сети. Определяются и описываются различные слои этой сети и принципы их работы.

В отличие от обычной нейронной сети, слои сверточной нейронной сети состоят из нейронов, расположенных в 3-х измерениях: ширине, высоте и глубине, т. е. измерениях, которые формируют объем. Кроме того, к концу построения CNN мы преобразуем полное изображение в единый вектор оценок класса, расположенных по измерению глубины.

В **четвертом** разделе описывается практическая часть данной работы: Основной практической задачей данной работы было создание искусственного агента, который смог бы ориентироваться и перемещаться в пространстве в случайных условиях, как под управлением человека, так и под управлением специального набора программ, также разработанных в рамках данного

практического задания, а так же распознавать объекты повседневной жизни, находящиеся вокруг него.

Прежде всего, для того, чтобы модулю автопилота было чем управлять, необходимо было создать самого робота. Для выполнения этой задачи были использованы следующие комплектующие.

- Arduino Mega 2560;
- плата расширения Motor Shield Plus - драйвер моторов выполненный на базе двухканального H-моста L6206Q на полевых транзисторах. Микросхема рассчитана на два коллекторных мотора с напряжением питания от 8 до 52 вольт и током до двух с половиной ампер. На плате предусмотрены два контура питания:
  - силовой контур — для питания моторов от силовой части микросхемы H-моста L6206Q. Силовое питание подключается через клеммник PWR;
  - цифровой контур — для питания вспомогательной цифровой логики управления микросхемой L6206Q и светодиодов индикации. Цифровое питание поступает на плату расширения от пина 5V управляющей платы;
- Troyka Shield — это плата расширения, которая помогает подключать большое количество периферии вроде сенсоров;
- ультразвуковой дальномер HC-SR04;
- инфракрасный дальномер, модель GP2Y0A021 компании Sharp;
- Raspberry Pi 3 Model B - миниатюрный, полноценный компьютер, который может взаимодействовать с внешним миром при помощи предустановленного wifi модуля;
- гусеничная платформа - оснащена двумя электромоторами с напряжением питания: 7,2 В и потребляемым током: до 2,5 А. Максимальная скорость до 1 км/ч;
- батарейный отсек на 2 батареи для питания силового контура платы расширения Motor Shield Plus;
- внешний аккумулятор для питания цифрового контура робота;
- web-камера;

В качестве корпуса робота или иными словами основной несущей платформы используется гусеничная платформа. Чертеж которой представлен на рисунке 1.

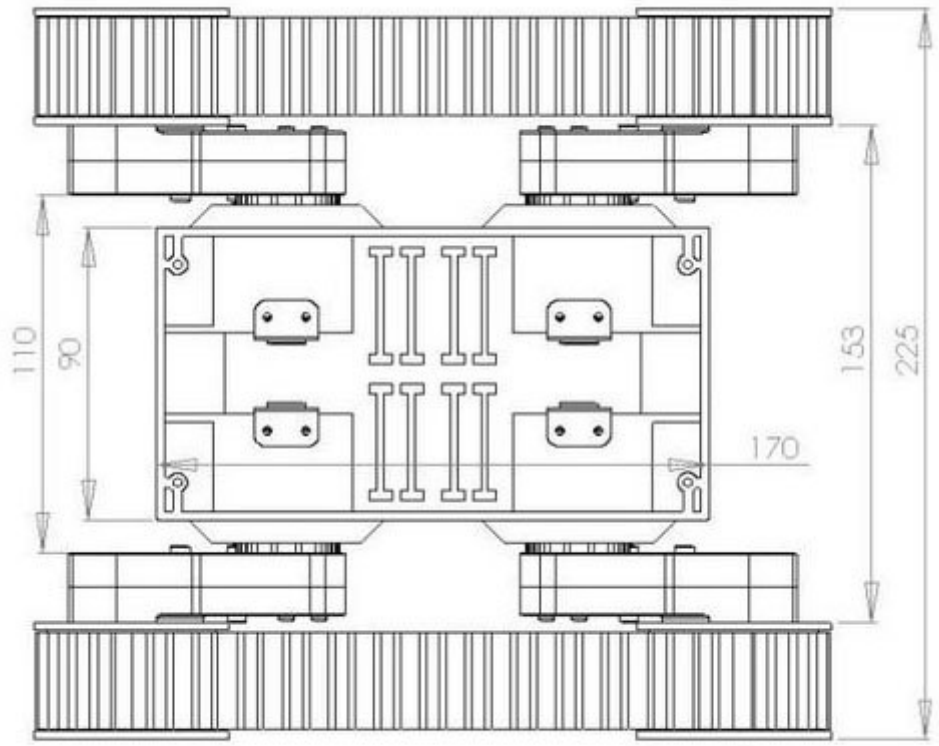


Рисунок 1 – Чертеж несущей платформы робота

Внутри нее располагается первый управляющий узел, состоящий из следующих компонентов, соединенных вместе по принципу слоеного пирога:

— Arduino Mega 2560;
— Motor Shield Plus;
— Troyka Shield;

А так же ультразвуковой и инфракрасный дальномеры, HC-SR04 и GP2Y0A021 соответственно, подключенные к плате расширения Troyka Shield. Схема подключения датчиков представлена на рисунке 2.

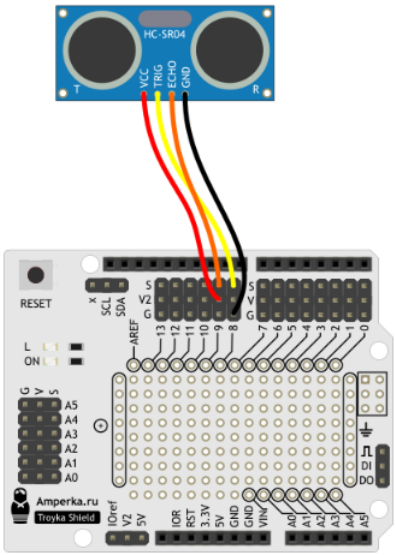


Рисунок 2 – Схема подключения дальномера

Затем управляющая плата была перепрошита, то есть в память контроллера была загружена разработанная управляющая программа, которая позволяет получать сигналы с датчиков, обрабатывать нажатия кнопок, общаться с различными устройствами через интерфейсы, управлять исполнительными процессами.

Далее была произведена сборка и настройка второго управляющего узла, состоящего из платы Raspberry Pi 3 Model B. Этот шаг был сложнее предыдущего, так как Raspberry Pi по сути является полноценным компьютером. Поэтому первым шагом в его настройке была установка соответствующей операционной системы. Вместо традиционного для обычных компьютеров жёсткого диска, Raspberry Pi использует microSD флеш-карту. Она должна быть предварительно подготовлена — на неё следует установить операционную систему. Мною была выбрана стандартная для таких контроллеров операционная система Raspbian. Плюс этой системы заключается не только в том, что это официальная поддерживаемая система, по сути являющаяся одной из огромного множества сборок Linux, но и в том, что она поставляется с предустановленным программным обеспечением для программирования, а именно, вместе с самой системой устанавливается Git, Python 3.7 и Java 8.

После подготовки второго управляющего модуля, была произведена его настройка. Так как робот управляется удаленно через WiFi сеть, необходимо

было настроить автоматическое определение точки доступа, для этого были произведены следующие действия:

— подключаем Raspberry к компьютеру, с которого осчуществляем настройку, при помощи патч-корда;
— подключаемся к Raspberry. По ssh, telnet, или другими путями. В нашем случае для первоначальной настройки используется подключение по ssh, а именно следующая команда: ssh pi@<IP адрес raspberry в нашей локальной сети>;
— sudo raspi-config. При выполнении этой команды откроется окно инструмента настройки Raspberry;
— переходим во вкладку Network Options > > Wi-Fi;
— указываем имя точки доступа и пароль;

Теперь, каждый раз при включении Raspberry автоматически указанную сеть и пытаться подключиться к ней. Далее запускается веб-сервер, что дает возможность возможность удаленного управления роботом.

На основном компьютере, с которого осуществляется управление, запускается клиент веб-сервера, который использует обученную нейронную сверточную сеть для идентификации обьектов на кадрах, полученных с веб камеры робота, в реальном времени. Сам клиент позволяет не только видеть то, что видит робот, но и управлять им, посредством нажатий соответствующих клавиш на клавиатуре.

— W - вперед;
— S - назад;
— A - поворот в направлении против движения часовой стрелки;
— D - поворот в направлении движения часовой стрелки;
— X - остановка моторов;

Ознакомиться со всеми программными кодами можно на страницах дипломной работы в соответствующих приложениях.

# ЗАКЛЮЧЕНИЕ

Разработка и использование нейронных сетей на сегодняшний день является одной из наиболее популярных и обширных отраслей индустрии информационных технологий. Количество и сложность задач, для решения которых создаются и применяются нейронные сети увеличивается с каждым годом. Очевидно, что в ближайшем будущем популярность данной технологии не снизится. В данной работе была изучена и решена одна из наиболее популярных задач, решаемых с помощью нейронных сетей, задача компьютерного зрения. Для ее решения был рассмотрен специально созданный для подобного рода задач вид сетей - сверточная нейронная сеть. Для применения этой сети был спроектирован и построен робот. Для которого также было написано собственное программное обеспечение. Таким образом, цель бакалаврской работы достигнута.