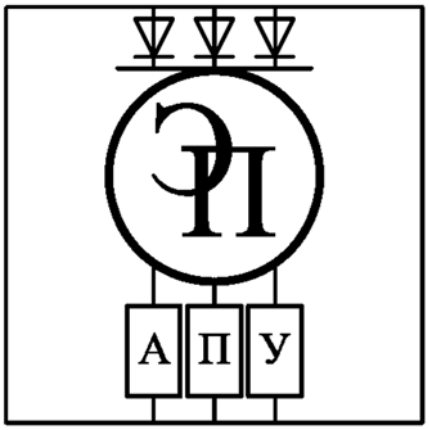
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Электропривод и автоматизация промышленных установок»

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ В АВТОМОБИЛЬНЫХ СИСТЕМАХ

Методические рекомендации к лабораторным работам для студентов направления подготовки 13.03.02 «Электроэнергетика и электротехника» дневной формы обучения



Могилев 2023

УДК 007:621.33
ББК 32.973.2:39.33
И75

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Электропривод и автоматизация промышленных установок» «31» августа 2023 г., протокол № 1

Составитель ст. преподаватель А. С. Третьяков

Рецензент канд. техн. наук, доц. С. В. Болотов

Методические рекомендации предназначены для студентов направления подготовки 13.03.02 «Электроэнергетика и электротехника» дневной формы обучения. Даны необходимые сведения для выполнения лабораторных работ.

Учебное издание

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ В АВТОМОБИЛЬНЫХ СИСТЕМАХ

Ответственный за выпуск	А. С. Коваль
Корректор	А. А. Подошевко
Компьютерная верстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 36 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Введение

Основной целью дисциплины является формирование представления об основных методах теории интеллектуальных систем, изучение знаний и моделирования рассуждений.

Цель лабораторных работ по данной дисциплине – получение навыков и знаний по построению простейших систем искусственного интеллекта (ИИ).

Ввиду большого объема материала и количества лабораторных работ, а также ограниченности объема данного учебного издания, в каждой работе представлен только подробно рассмотренный пример и индивидуальное задание для выполнения. Полные версии методических рекомендаций по каждой лабораторной работе выложены в а. 207 / к. 2.

1 Техника безопасности

Во время работы с персональным компьютером (ПК) студент обязан:

– соблюдать требования охраны труда;

– содержать в порядке и чистоте свое рабочее место;

– соблюдать оптимальное расстояние от экрана монитора до глаз.

Работу за экраном монитора следует периодически прерывать на регламентированные перерывы, которые устанавливаются для обеспечения работоспособности и сохранения здоровья, или заменять другой работой с целью сокращения рабочей нагрузки у экрана.

Продолжительность непрерывной работы с ПК без регламентированного перерыва не должна превышать 2 ч.

Не следует оставлять оборудование включенным без наблюдения. При необходимости прекращения на некоторое время работы корректно закрываются все активные задачи и оборудование выключается.

При работе с ПК не разрешается:

– при включенном питании прикасаться к панелям с разъемами оборудования, разъемами питающих и соединительных кабелей, экрану монитора;

– загромождать верхние панели оборудования, рабочее место бумагами, посторонними предметами;

– производить переключения, отключение питания во время выполнения активной задачи;

– допускать попадание влаги на поверхность оборудования;

– включать сильно охлажденное (принесенное с улицы в зимнее время) оборудование;

– производить самостоятельно вскрытие и ремонт оборудования;

– вытирать пыль на включенном оборудовании;

– допускать нахождение вблизи оборудования посторонних лиц.

В аварийных (экстремальных) ситуациях необходимо:

1) при повреждении оборудования, кабелей, проводов, неисправности заземления, появлении запаха гари, возникновении необычного шума и других неисправностях немедленно отключить электропитание оборудования и сообщить о случившемся непосредственному руководителю и лицу, осуществляющему техническое обслуживание оборудования;

2) в случае сбоя в работе оборудования ПК или программного обеспечения вызвать специалиста организации, осуществляющего техническое обслуживание данного оборудования, для устранения неполадок;

3) при возгорании электропроводки, оборудования и тому подобных происшествиях отключить электропитание и принять меры по тушению пожара с помощью имеющихся первичных средств пожаротушения, сообщить о происшедшем непосредственному руководителю.

2 Лабораторная работа № 1. Составление модели семантической сети для автомобильной системы

Цель работы: получение навыков по составлению модели семантической сети для автомобильной системы.

Методические рекомендации по проведению исследований

Рассмотрим пример построения семантической сети.

Этапы построения:

1) определение предметной области семантической сети;

2) определение понятий семантической сети;

3) определение связей между понятиями семантической сети.

В качестве примера была создана семантическая сеть, описывающая предметную область легковых автомобилей B-класса.

Были выделены три типа понятий:

1) сущность: автомобили B-класса, Volkswagen, Renault, KIA;

2) экземпляр: Volkswagen Polo 1.4 TSI MT Connect, Volkswagen Polo 1.4 TSI MT Highline, Volkswagen Polo 1.4 TSI DSG Connect, Volkswagen Polo 1.6 MPI MT Comfortline, Volkswagen Polo 1.6 MPI MT Trendline, Volkswagen Polo 1.6 MPI MT Allstar, Volkswagen Polo 1.6 MPI MT Comfortline, Volkswagen Polo 1.6 MPI MT Trendline, Volkswagen Polo 1.2 TSI DSG Comfortline 3dr., и т. д.;

3) свойство: тип привода, объем двигателя, разгон, цена до 700 тыс. р., от 700 тыс. р до 1 млн р., больше 1 млн р., коробка передач ручная, автоматическая и т. д.

На основании полученных зависимостей построена семантическая сеть (рисунок 1).

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает предметную область легковых автомобилей.

Студент должен сделать следующее.

1 Сформировать не менее трех понятий (сущность, экземпляр, свойства).

2 Сопоставить полученные понятия, получив взаимосвязи.

3 Составить и оформить семантическую сеть.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) семантическая сеть автомобилей определенной марки;

3) выводы.

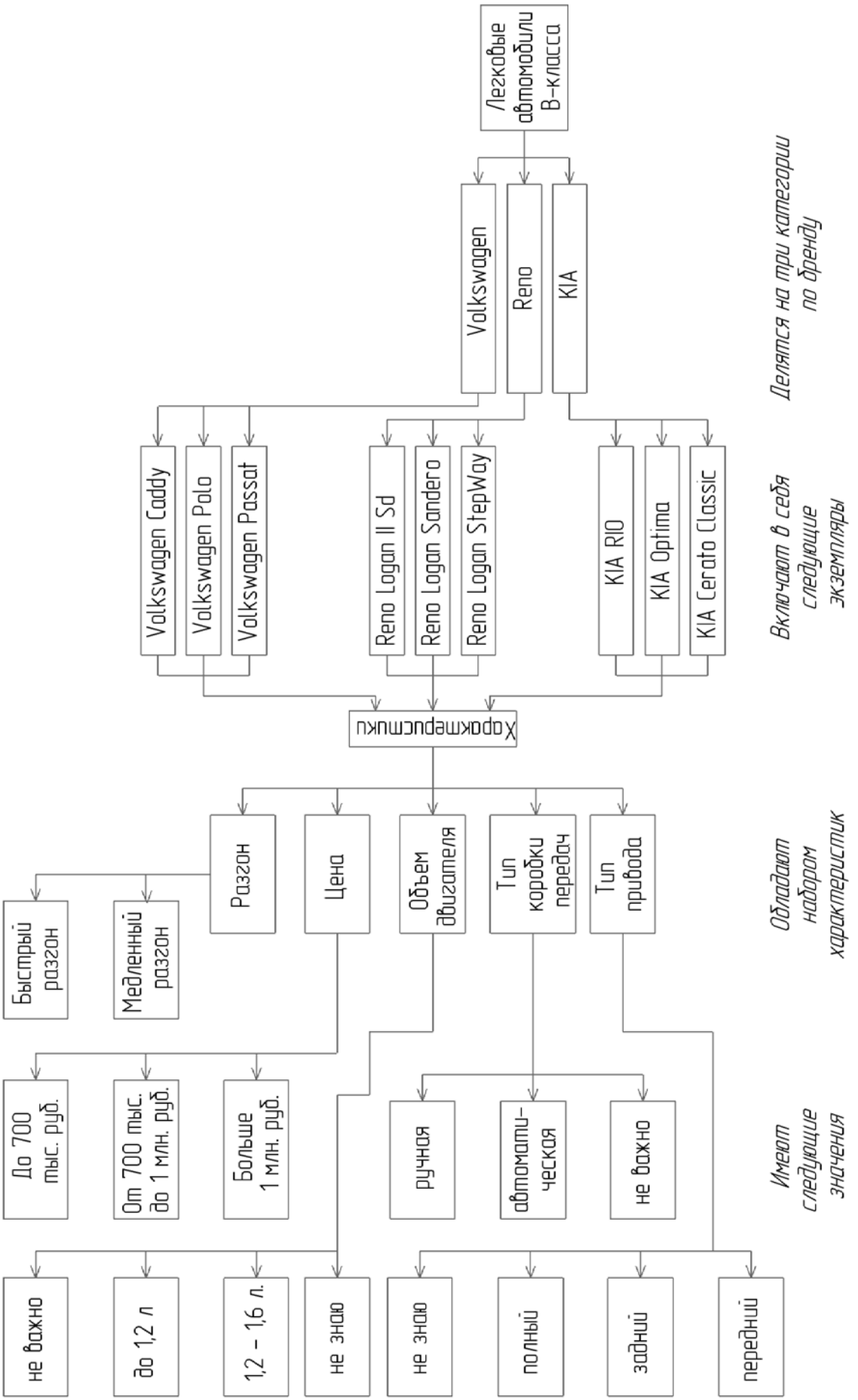


Рисунок 1 – Семантическая сеть понятий системы

Контрольные вопросы

1 Какую сеть можно назвать семантической?
2 Какие есть виды семантических сетей?
3 Что такое тезаурус?
4 Что такое онтология?
5 Что собой представляет машина обработки знаний пользовательского интерфейса?
6 Какие есть семантические отношения?
7 Какой электромобиль можно назвать умным?
8 Какие функции выполняет искусственный интеллект в умном автомобиле?
9 Какие есть виды умных автомобилей?
10 Как можно составить семантическую сеть по отношению к умному автомобилю?

3 Лабораторная работа № 2. Изучение среды Python для решения задач систем искусственного интеллекта

Цель работы: получение навыков по программированию на языке Python.

Методические рекомендации по проведению исследований

Запускаем IDLE (рисунок 2).

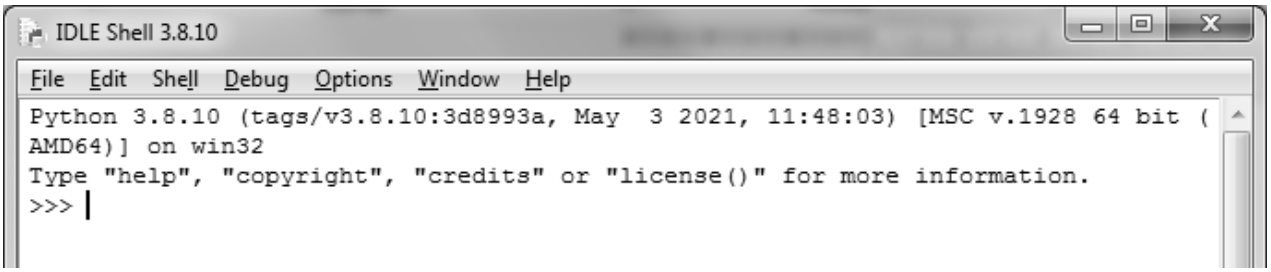


Рисунок 2 – Скриншот окна IDLE *Python*

Напишем первую программу в виде фразы «hello world».
Чтобы написать «hello world» на Python, достаточно всего одной строки:

```
print("Hello world!")
```

Вводим этот код в IDLE и нажимаем Enter. Результат виден на рисунке 3.
Такая работа проводится в интерактивном режиме. Но, всё-таки, интерактивный режим не будет являться основным. В основном программный код сохраняется в файл и запускается уже файл.
Для того чтобы создать новое окно, в интерактивном режиме IDLE выберите File → New File (рисунок 4).

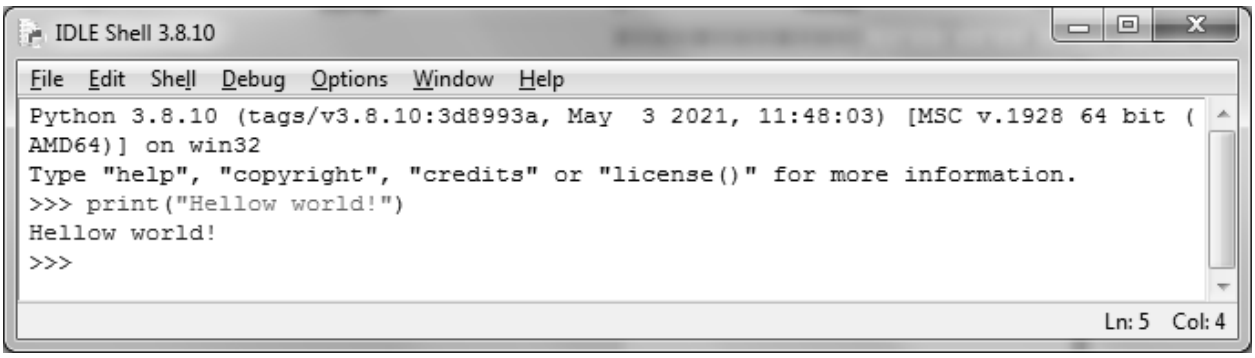


Рисунок 3 – Результат выполнения программы

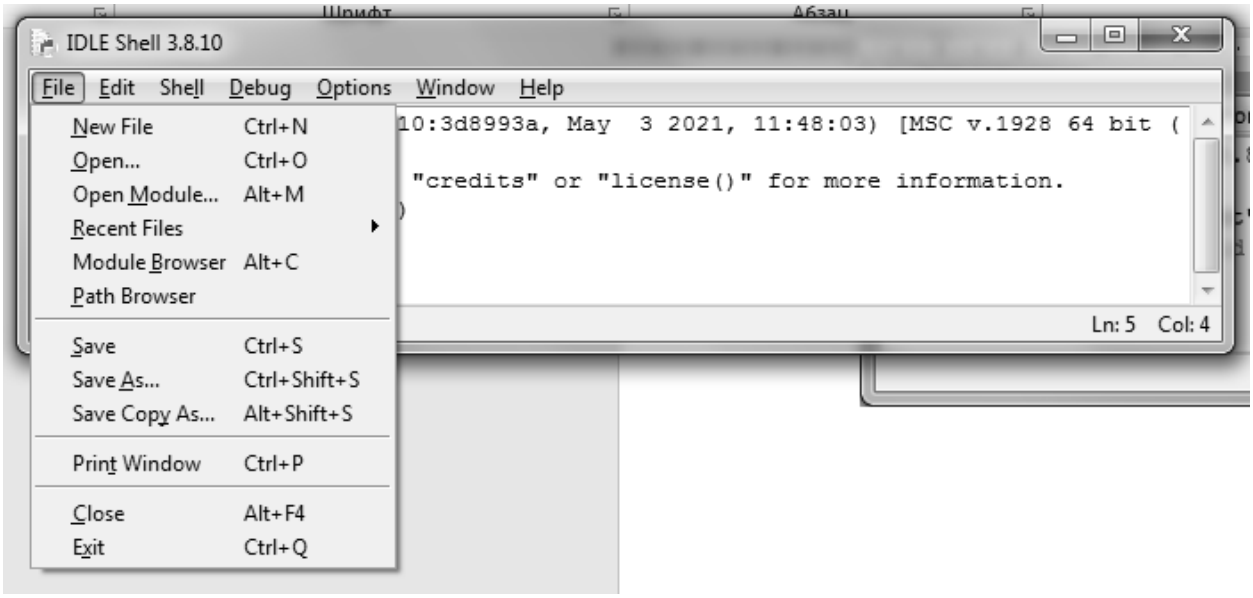


Рисунок 4 – Создание нового файла

В открывшемся окне введите следующий код:

```
name = input("Как Вас зовут? ")
print("Привет,", name)
```

Первая строка печатает вопрос («Как Вас зовут?»), ожидает, пока вы не напечатаете что-нибудь и не нажмёте Enter и сохраняет введённое значение в переменной name.

Во второй строке мы используем функцию print для вывода текста на экран, в данном случае для вывода «Привет», и того, что хранится в переменной «name».

Теперь нажмём F5 (или выберем в меню IDLE Run → Run Module) и убедимся, что то, что мы написали, работает. Перед запуском IDLE предложит нам сохранить файл. Сохраним туда, куда вам будет удобно, после чего программа запустится.

Вы должны увидеть что-то наподобие этого (рисунок 5).

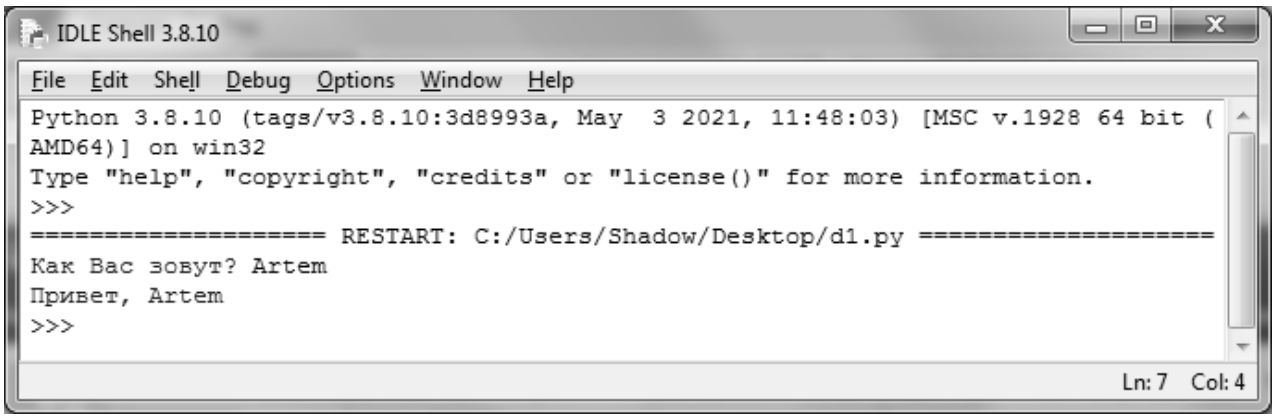


Рисунок 5 – Результат выполнения программы

Конструкция if-elif-else.

Условная инструкция if-elif-else (её ещё иногда называют оператором ветвления) – основной инструмент выбора в Python. Проще говоря, она выбирает, какое действие следует выполнить, в зависимости от значения переменных в момент проверки условия.

Сначала записывается часть if с условным выражением, далее могут следовать одна или более необязательных частей elif, и, наконец, необязательная часть else. Общая форма записи условной инструкции if выглядит следующим образом:

```
if test1:
   state1
elif test2:
   state2
else:
   state3
```

Простой пример (напечатает 'true', т. к. 1 – истина):

```
if 1:
    print('true')
else:
    print('false')
```

```
true
```

Чуть более сложный пример (его результат будет зависеть от того, что ввёл пользователь):

```
a = int(input())
if a < -5:
   print('Low')
elif -5 <= a <= 5:
   print('Mid')
else:
   print('High')
```

Конструкция с несколькими elif может также служить отличной заменой конструкции switch – case в других языках программирования.

Следующая инструкция:

```
if X:
   A = Y
else:
   A = Z
```

довольно короткая, но, тем не менее, занимает целых четыре строки. Специально для таких случаев и было придумано выражение if/else:

```
A = Y if X else Z
```

В данной инструкции интерпретатор выполнит выражение Y, если X истинно, в противном случае выполнится выражение Z.

```
A = 't' if 'spam' else 'f'
A
't'
```

Циклы for и while, операторы break и continue, оператор else

Цикл while.

While – один из самых универсальных циклов в Python, поэтому довольно медленный. Выполняет тело цикла до тех пор, пока условие цикла истинно.

```
i = 5
while i < 15:
    print(i)
    i = i + 2
...
5
7
9
11
13
```

Цикл for.

Цикл for уже чуточку сложнее, чуть менее универсальный, но выполняется гораздо быстрее цикла while. Этот цикл проходится по любому итерируемому объекту (например, строке или списку), и во время каждого прохода выполняет тело цикла.

```
for i in 'hello world':
print(i * 2, end='')
...
hheellllloo  wwoorrlldd
```

Оператор continue.

Оператор continue начинает следующий проход цикла, минуя оставшееся тело цикла (for или while).

```
for i in 'hello world':
    if i == 'o':
        continue
    print(i * 2, end='')

hheellll  wwrrlldd
```

Оператор break.

Оператор break досрочно прерывает цикл.

```
for i in 'hello world':
    if i == 'o':
        break
    print(i * 2, end='')

hheellll
```

Оператор else.

Оператор else, примененный в цикле for или while, проверяет, был ли произведен выход из цикла инструкцией break, или же «естественным» образом. Блок инструкций внутри else выполнится только в том случае, если выход из цикла произошел без помощи break.

```
for i in 'hello world':
    if i == 'a':
        break
else:
    print('Буквы а в строке нет')

Буквы а в строке нет
```

Математические операции над числами.

Числа в Python 3 ничем не отличаются от обычных чисел. Они поддерживают набор самых обычных математических операций (таблица 1).

Списки.

Списки в Python – упорядоченные изменяемые коллекции объектов произвольных типов (почти как массив, но типы могут отличаться).

Чтобы использовать списки, их нужно создать. Создать список можно несколькими способами. Например, можно обработать любой итерируемый объект (например, строку) встроенной функцией list:

```
 list('список')
['с', 'п', 'и', 'с', 'о', 'к']
```

Таблица 1 – Математические операции в Python

Операция	Название операции
x + y	Сложение
x – y	Вычитание
x * y	Умножение
x / y	Деление
x // y	Получение целой части от деления
x % y	Остаток от деления
–x	Смена знака числа
abs(x)	Модуль числа
divmod(x, y)	Пара (x // y, x % y)
x ** y	Возведение в степень
pow(x, y[, z])	xy по модулю (если модуль задан)

Еще один способ создать список – это генераторы списков. Генератор списков – способ построить новый список, применяя выражение к каждому элементу последовательности. Генераторы списков очень похожи на цикл for.

```
 c = [c * 3 for c in 'list']
 c
['lll', 'iii', 'sss', 'ttt']
```

Функции.

Функция в Python – объект, принимающий аргументы и возвращающий значение. Обычно функция определяется с помощью инструкции def.

Определим простейшую функцию:

```
def add(x, y):
   return x + y
```

Инструкция return говорит, что нужно вернуть значение. В нашем случае функция возвращает сумму x и y.

Теперь мы ее можем вызвать:

```
add(1, 10)
11
add('abc', 'def')
'abcdef'
```

Функция, может быть любой сложности и возвращать любые объекты (списки, кортежи и даже функции!):

```
def newfunc(n):
    def myfunc(x):
        return x + n
```

```
    return myfunc

new = newfunc(100)  # new - это функция
new(200)
300
```

Функция может и не заканчиваться инструкцией return, при этом функция вернет значение None:

```
def func():
    pass

print(func())
None
```

Аргументы функции.

Функция может принимать произвольное количество аргументов или не принимать их вовсе. Также распространены функции с произвольным числом аргументов, функции с позиционными и именованными аргументами, обязательными и необязательными.

```
def func(a, b, c=2): # c - необязательный аргумент
    return a + b + c

func(1, 2)  # a = 1, b = 2, c = 2 (по умолчанию)
5
func(1, 2, 3)  # a = 1, b = 2, c = 3
6
func(a=1, b=3)  # a = 1, b = 3, c = 2
6
func(a=3, c=6)  # a = 3, c = 6, b не определен
Traceback (most recent call last):
  File "", line 1, in
    func(a=3, c=6)
TypeError: func() takes at least 2 arguments (2 given)
```

Функция также может принимать переменное количество позиционных аргументов, тогда перед именем ставится *:

```
def func(*args):
    return args

 func(1, 2, 3, 'abc')
(1, 2, 3, 'abc')
 func()
()
 func(1)
(1,)
```

Как видно из примера, args – это кортеж из всех переданных аргументов функции, и с переменной можно работать также, как и с кортежем.

Работа с файлами.

Прежде, чем работать с файлом, его надо открыть. С этим замечательно справится встроенная функция open:

```
f = open('text.txt', 'r')
```

У функции open много параметров, важны три аргумента: первый – это имя файла. Путь к файлу может быть относительным или абсолютным. Второй аргумент – это режим, в котором мы будем открывать файл (таблица 2).

Таблица 2 – Режимы открытия файла

Режим	Обозначение
'r'	Открытие на чтение (является значением по умолчанию)
'w'	Открытие на запись, содержимое файла удаляется, если файла не существует, создается новый
'x'	Открытие на запись, если файла не существует, иначе исключение
'a'	Открытие на дозапись, информация добавляется в конец файла
'b'	Открытие в двоичном режиме
't'	Открытие в текстовом режиме (является значением по умолчанию)
'+'	Открытие на чтение и запись

Режимы могут быть объединены, т. е., к примеру, 'rb' – чтение в двоичном режиме. По умолчанию режим равен 'rt'.

И последний аргумент, encoding, нужен только в текстовом режиме чтения файла. Этот аргумент задает кодировку.

Чтение из файла.

Для этого есть несколько способов, но большого интереса заслуживают лишь два из них.

Первый – метод read, читающий весь файл целиком, если был вызван без аргументов, и n символов, если был вызван с аргументом (целым числом n).

```
f = open('text.txt')
f.read(1)
'H'
f.read()
'ello world!\nThe end.\n\n'
```

Ещё один способ сделать это – прочитать файл построчно, воспользовавшись циклом for:

```
f = open('text.txt')
for line in f:
    line
...
'Hello world!\n'
'\n'
'The end.\n'
'\n'
```

Запись в файл.

Теперь рассмотрим запись в файл. Попробуем записать в файл вот такой вот список:

```
l = [str(i)+str(i-1) for i in range(20)]
l
['0-1', '10', '21', '32', '43', '54', '65', '76', '87', '98', '109', '1110', '1211', '1312', '1413', '1514', '1615', '1716', '1817', '1918']
```

Откроем файл на запись:

```
f = open ('text.txt', 'w')
```

Запись в файл осуществляется с помощью метода write:

```
for index in l:
    f.write(index + '\n')
...
4
3
3
3
3
```

После окончания работы с файлом его обязательно нужно закрыть с помощью метода close:

```
f.close()
```

Теперь попробуем воссоздать этот список из получившегося файла. Откроем файл на чтение (надеюсь, вы поняли, как это сделать?), и прочитаем строки.

```
f = open('text.txt', 'r')
l = [line.strip() for line in f]
l
['0-1', '10', '21', '32', '43', '54', '65', '76', '87', '98', '109', '1110', '1211', '1312', '1413', '1514', '1615', '1716', '1817', '1918']
f.close()
```

Мы получили тот же список, что и был. В более сложных случаях (словарях, вложенных кортежей и т. д.) алгоритм записи придумать сложнее. Но это и не нужно. В Python уже давно придумали средства, такие как pickle или json, позволяющие сохранять в файле сложные структуры.

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по решению задач на Python.

Студент должен сделать следующее.

1 Решить задачу, связанную с типовыми математическими операциями.
2 Решить задачу, связанную с циклами.
3 Решить задачу, связанную с работой над файлами.
4 Оформить отчет.

Содержание отчета

Отчет должен содержать следующее:

1) цель работы;
2) листинг программного кода и результаты работы;
3) выводы.

Контрольные вопросы

1 Что собой представляет среда Python и какие имеет возможности?
2 Что такое нейрон?
3 Какими характеристиками описывается нейрон?
4 Как программируется нейрон?
5 Что такое нейронная сеть?
6 Какими характеристиками обладает нейронная сеть?
7 Из каких частей состоит нейронная сеть?
8 Как запрограммировать нейронную сеть?
9 Как проводить обучение нейронной сети?
10 Как минимизировать потери нейронной сети?

4 Лабораторная работа № 3. Моделирование экспертной системы электромобиля

Цель работы: получение навыков по моделированию экспертной системы электромобиля.

Индивидуальное задание

Студент должен сделать следующее.

1 Запустить экспертную систему МЭС 2.0.

2 Создать базу знаний на тему согласно заданию преподавателя, проводящего лабораторное занятие.

3 Заполнить базу знаний, используя не менее пять-семь объектов.

4 Провести консультации для всех объектов базы знаний, сохранить их протоколы.

5 Повторить пп. 1–4, создав еще пять баз знаний.

6 Провести консультации для полученных баз знаний, сохранить их протоколы.

7 Сравнить вероятности исходов в окне «Сортировка результатов» для каждой базы знаний, сделать выводы об их абсолютных величинах (количестве знаков после запятой) и сумме всех вероятностей для каждой базы знаний.

8 Записать в **Выводах по работе** результаты каждой консультации во всех базы знаний с указанием того, какой именно объект определялся и какие объекты, в результате, распознались, указать вероятности достоверности исходов.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) модель экспертной системы электромобиля определенной марки;

3) выводы.

Контрольные вопросы

1 Что такое экспертные системы?

2 Какими возможностями обладают экспертные системы?

3 Какими характеристиками обладают экспертные системы?

4 Какие компоненты включают в себя экспертные системы?

5 Какими режимами обладают экспертные системы?

6 Какие есть современные широко распространенные экспертные системы, применяемые в промышленности, медицине и т. д.?

7 Из каких этапов состоит создание и моделирование экспертных систем?

8 Опишите последовательность моделирования экспертных систем.

9 Опишите анализ полученных результатов.

5 Лабораторная работа № 4. Изучение пакета FuzzyLogic Toolbox в среде MATLAB

Цель работы: изучение пакета FuzzyLogic Toolbox в среде MATLAB.

Методические рекомендации по проведению исследований

Кинематика является наукой о движении. В роботизированной руке, учитывая углы соединений, уравнения кинематики дают местоположение кисти руки. Инверсная кинематика относится к противоположному процессу. Учитывая желаемое местоположение роботизированной руки, должны быть известны углы соединений, чтобы определить местоположение руки. Обычно существует более, чем одно решение такой задачи.

Это – типичная проблема в робототехнике, которая должна быть решена, чтобы управлять роботизированной рукой. В 2-мерном пространстве данная проблема уменьшается до нахождения двух углов: первый угол theta1между первой рукой и землей (или независимо от того, что это присоединено), второй угол theta1 между первой рукой и второй рукой (рисунок 6).

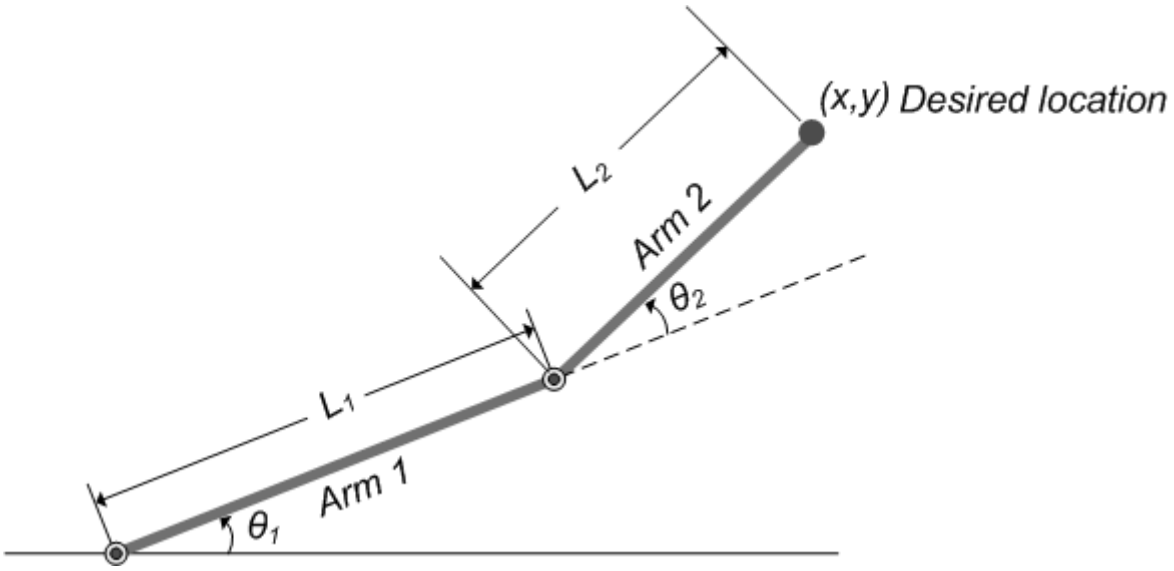


Рисунок 6 – Размеры роботизированной руки с двумя углами theta1 и theta2

Поскольку прямые формулы кинематики для роботизированной руки известны, координаты X и Y кисти руки выведены для целой области значений углов вращения двух соединений. Координаты и углы сохранены, чтобы использоваться в качестве обучающих данных, чтобы обучить ANFIS (адаптивная нейронечеткая система вывода) сеть.

Во время обучения сеть ANFIS учится сопоставлять координаты (X, Y) к углам (theta1, theta2). Обученная сеть ANFIS затем используется в качестве части большей системы управления, чтобы управлять роботизированной рукой.

Зная желаемое местоположение роботизированной руки, система управления использует обученную сеть ANFIS, чтобы вывести угловые положения соединений и прикладывает силу к соединениям роботизированной руки, соответственно, чтобы переместить его в желаемое местоположение.

ANFIS обозначает адаптивную нейронечеткую систему вывода. Это гибридный нейронечеткий метод, который приносит способности к обучению нейронных сетей к нечетким системам вывода. Алгоритм обучения настраивает функции принадлежности Sugeno-типа с помощью учебных данных о вводе/выводе.

В этом случае данные о вводе/выводе относятся к набору данных «координат/углов». Координаты действуют как вход к ANFIS и угловому действию как выход. Алгоритм обучения учит, что ANFIS, чтобы сопоставить координаты с углами посредством процесса вызвал обучение. В конце обучения обученная сеть ANFIS изучила бы карту ввода-вывода и была готова быть развернутой в большее решение для системы управления.

Генерация данных.

Пусть theta1 будет углом между первой рукой и землей. Пусть theta2 будет углом между первой и второй рукой (см. рисунок 6), длина первой руки будет L1, а второй – L2.

Примем, что первое сочленение ограничило свободу вращения, и оно может вращаться между 0° и 90°. Примем, что второе сочленение ограничило свободу вращения и может вращаться между 0° и 180°. Следовательно, 0<=theta1<=pi/2 и 0<=theta2<=pi (рисунок 7).

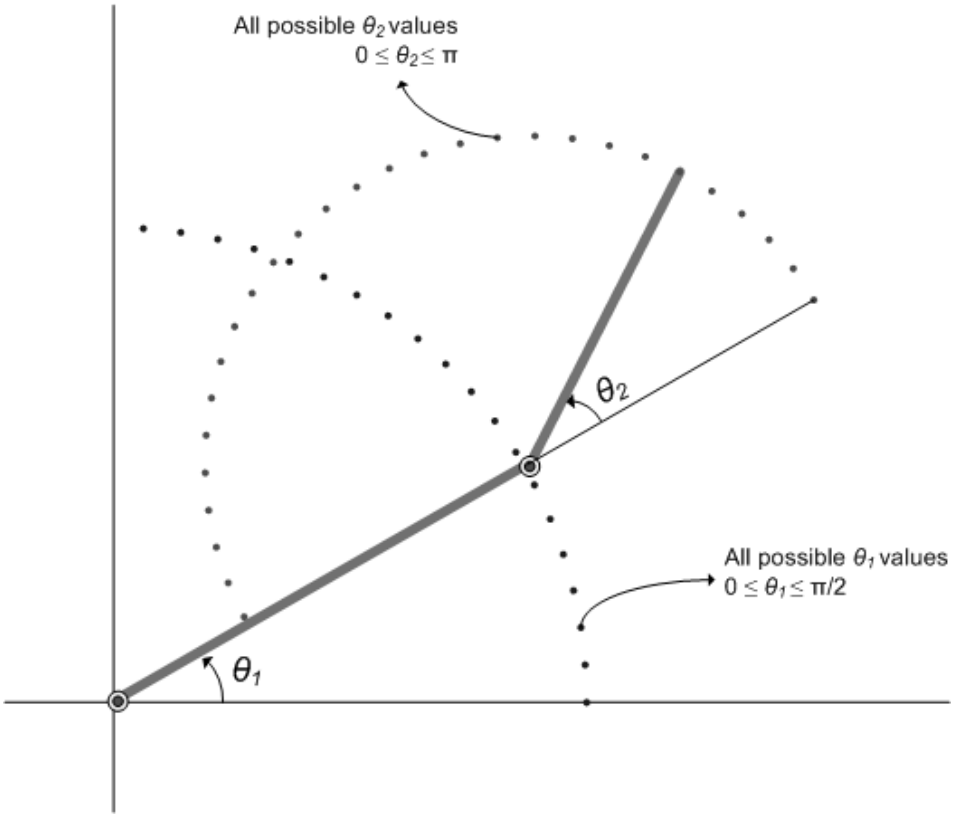


Рисунок 7 – Диапазоны значений theta1 и theta2

Теперь для каждой комбинации theta1 и theta2 значения координаты X и Y выведены с помощью прямых формул кинематики.

Следующий фрагмент кода показывает, как данные сгенерированы для всей комбинации значений theta1 и theta2 и сохраняет их в матрицу, которая будет использоваться в качестве обучающих данных.

```
l1 = 10; % length of first arm
l2 = 7; % length of second arm
theta1 = 0:0.1:pi/2; % all possible theta1 values
theta2 = 0:0.1:pi; % all possible theta2 values
[THETA1,THETA2] = meshgrid(theta1,theta2); % generate grid of angle values
X = l1 * cos(THETA1) + l2 * cos(THETA1 + THETA2); % compute x coordinates
Y = l1 * sin(THETA1) + l2 * sin(THETA1 + THETA2); % compute y coordinates
data1 = [X(:) Y(:) THETA1(:)]; % create x-y-theta1 dataset
data2 = [X(:) Y(:) THETA2(:)]; % create x-y-theta2 dataset
```

Следующий график показывает все точки данных X–Y, сгенерированные путем циклического повторения через различные комбинации theta1 и theta2, и выведение X и Y (рисунок 8). График может быть сгенерирован при помощи следующего кода. График аннотируется далее для более легкого понимания.

```
plot(X(:),Y(:),'r.');
axis equal;
xlabel('X','fontsize',10)
ylabel('Y','fontsize',10)
title('X-Y coordinates for all theta1 and theta2 combinations','fontsize',10)
```

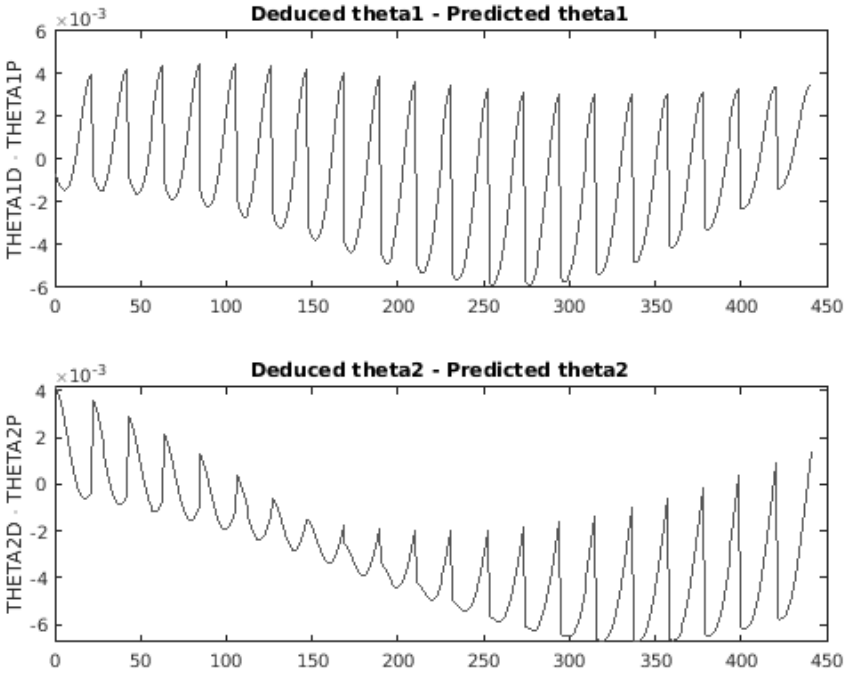


Рисунок 8 – Координаты X–Y для всех комбинаций theta1 и theta2

Создание сети ANFIS.

Один подход к созданию решения ANFIS должен создать две сети ANFIS: один, чтобы предсказать theta1 и другой, чтобы предсказать theta2.

Для сетей ANFIS, чтобы смочь предсказать углы они должны быть обучены с демонстрационными данными ввода-вывода. Первая сеть ANFIS будет обучена с координатами X и Y для угла theta1. Матричный data1 содержит x-y-theta1 набор данных, требуемый для обучения первой сети ANFIS. Поэтому data1 будет использоваться в качестве набора данных, чтобы обучить первую сеть ANFIS.

Точно также вторая сеть ANFIS будет обучена с координатами X и Y для угла theta2. Матричный data2 содержит x-y-theta2 набор данных, требуемый обучать вторую сеть ANFIS. Поэтому data2 будет использоваться в качестве набора данных, чтобы обучить вторую сеть ANFIS.

Чтобы обучить сеть ANFIS, сначала задайте опции обучения с помощью команды anfisOptions. Задайте объект FIS с 7 функции принадлежности для каждой входной переменной. Обучите систему 150 эпох:

```
opt = anfisOptions;
opt.InitialFIS = 7;
opt.EpochNumber = 150;
opt.DisplayANFISInformation = 0;
opt.DisplayErrorValues = 0;
opt.DisplayStepSize = 0;
opt.DisplayFinalResults = 0;
```

Обучите систему ANFIS с помощью первого набора обучающих данных, data1:

```
disp('--> Training first ANFIS network.')
--> Training first ANFIS network.
anfis1 = anfis(data1,opt);
```

Измените количество входных функций принадлежности и обучите систему ANFIS с помощью второго набора обучающих данных, data2:

```
disp('--> Training second ANFIS network.')
--> Training second ANFIS network.
opt.InitialFIS = 6;
anfis2 = anfis(data2,opt);
```

В данном примере количество входных функций принадлежности и учебных эпох было выбрано на основе экспериментирования с различными потенциальными ценностями.

anfis1 и anfis2 представляют собой две обученных сети ANFIS, которые будут развернуты в большей системе управления.

Если обучение завершено, две сети ANFIS научились аппроксимировать углы (theta1, theta2) в зависимости от координат (X-Y).

Проверка сети ANFIS.

Обучение сетей – важный шаг для проверки сети, чтобы определить, как хорошо сети ANFIS отработали бы в большей системе управления.

Примем, что для сетей ANFIS важно иметь низкие ошибки в рабочем диапазоне $0 < x < 2$ и $8 < y < 10$.

```
x = 0:0.1:2; % x coordinates for validation
y = 8:0.1:10; % y coordinates for validation
```

Значения углов theta1 и theta2 выведены математически из координат X и Y с помощью формул инверсной кинематики.

```
[X,Y] = meshgrid(x,y);

c2 = (X.^2 + Y.^2 - l1^2 - l2^2)/(2*l1*l2);
s2 = sqrt(1 - c2.^2);
THETA2D = atan2(s2,c2); % theta2 is deduced

k1 = l1 + l2.*c2;
k2 = l2*s2;
THETA1D = atan2(Y,X) - atan2(k2,k1); % theta1 is deduced
```

THETA1D и THETA2D – переменные, которые содержат значения theta1 и theta2.

theta1 и theta2 – значения, предсказанные обученными сетями ANFIS, получены при помощи команды evalfis, которая оценивает FIS для данных входных параметров. Здесь, evalfis используется, чтобы узнать выходные параметры FIS для тех же X–Y значений, используемых ранее в формулах инверсной кинематики.

```
XY = [X(:) Y(:)];
THETA1P = evalfis(anfis1,XY); % theta1 predicted by anfis1
THETA2P = evalfis(anfis2,XY); % theta2 predicted by anfis2
```

Теперь мы видим, как близки выходные параметры FIS относительно выведенных значений.

```
theta1diff = THETA1D(:) - THETA1P;
theta2diff = THETA2D(:) - THETA2P;
subplot(2,1,1)
plot(theta1diff)
ylabel('THETA1D - THETA1P')
title('Deduced theta1 - Predicted theta1')
subplot(2,1,2)
plot(theta2diff)
ylabel('THETA2D - THETA2P')
title('Deduced theta2 - Predicted theta2')
```

Figure contains 2 axes objects. Axes object 1 with title Deduced theta1 - Predicted theta1 contains an object of type line. Axes object 2 with title Deduced theta2 - Predicted theta2 contains an object of type line.

Создание решения вокруг обученных сетей ANFIS.

Теперь, учитывая задачу, такую как роботы, берущие объект в сборочном конвейере, большая система управления будет использовать обученные сети ANFIS в качестве ссылки, во многом как интерполяционная таблица, чтобы определить, каковы углы рук должны быть, учитывая желаемое местоположение для кисти руки. Зная желаемые углы и текущие углы соединений, система будет прикладывать силу соответственно на соединениях рук, чтобы переместить их к желаемому местоположению.

Команда invkine запускает графический интерфейс пользователя, который показывает, как две обученных сети ANFIS моделируют выполнение эллипса – фигуры, которая получается из траектории движения конца второго манипулятора (рисунок 9).

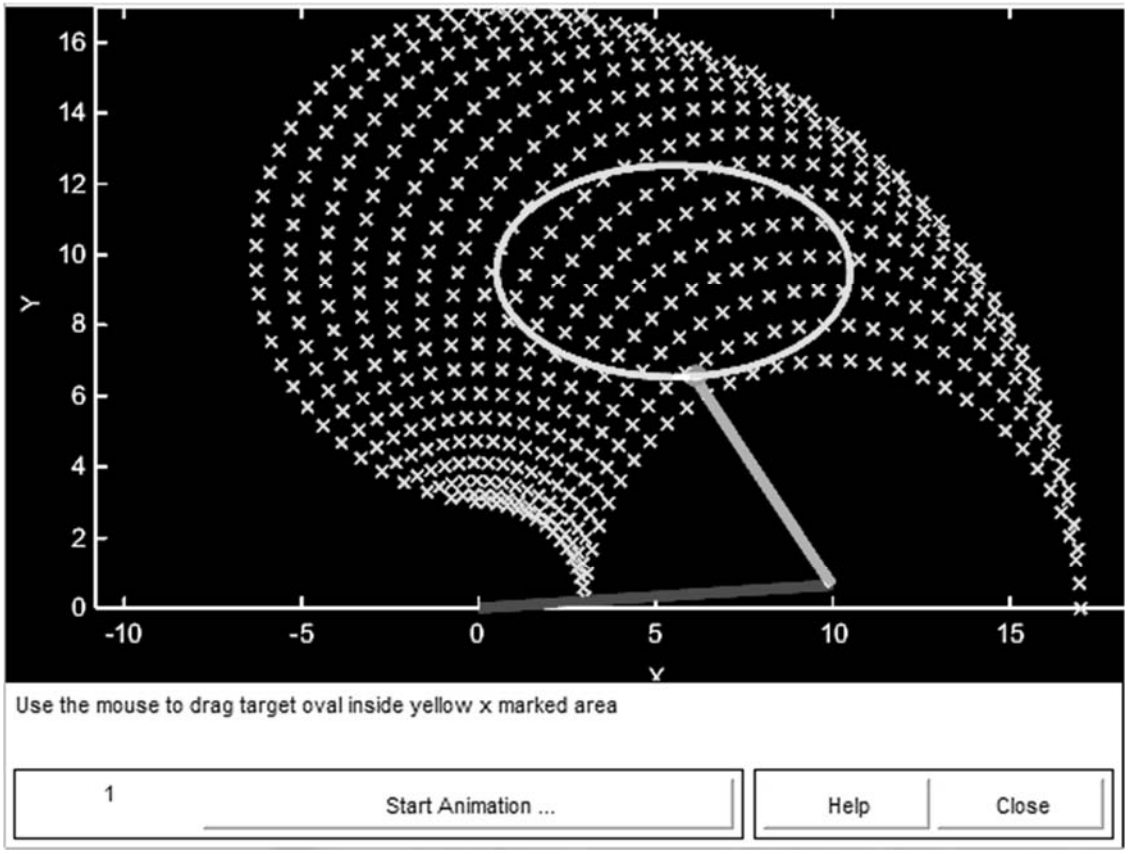


Рисунок 9 – Графический интерфейс пользователя для моделирования инверсной кинематики

Две сети ANFIS, используемые в примере, были предварительно обучены и развертываются в большую систему, которая управляет кистью манипулятора, чтобы проследить эллипс во входном пространстве.

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по моделированию инверсной кинематики в роботизированной руке.

Студент должен сделать следующее.

1 Сгенерировать исходные данные для решения задачи.

2 Создать сеть ANFIS.

3 Обучить сеть ANFIS.

4 Создать общее решение вокруг обученных сетей ANFIS.

5 Оформить отчет.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) листинг выполненного индивидуального задания;

3) выводы.

Контрольные вопросы

1 Дайте понятие Fuzzy Logic.

2 Что такое нечеткие множества?

3 Назовите виды функций принадлежности.

4 Перечислите операции над нечеткими множествами.

5 Какие есть лингвистические переменные?

6 Что такое нечеткий логический вывод?

7 Какие есть базы нечетких правил?

8 Какие есть алгоритмы нечеткого логического вывода?

9 Для чего создан пакет Fuzzy Logic Toolbox?

10 Какие основные программные средства входят в пакет Fuzzy Logic Toolbox?

11 Опишите работу с редактором систем нечеткого вывода FIS.

12 Опишите работу с редактором функций принадлежности.

13 Опишите работу с редактором правил системы нечеткого вывода.

14 Опишите работу с программой просмотра правил системы нечеткого вывода.

15 Опишите работу с программой просмотра поверхности системы нечеткого вывода.

6 Лабораторная работа № 5. Изучение пакета Neural Network Toolbox в среде MATLAB

Цель работы: получение навыков по работе в пакете Neural Network Toolbox в среде MATLAB.

Методические рекомендации по проведению исследований

Рассматриваемый пример демонстрирует следующие операции:
– загрузка и исследование изображений;
– определение сетевой архитектуры;
– задание опций обучения;
– обучение сети;
– предсказание метки новых данных и вычисление точности классификации.

Загрузка и исследование данных изображений.
Загрузите выборочные данные цифры как datastore изображений. ImageDatastore автоматически помечает изображения на основе имен папок и хранит данные как ImageDatastore объект. Datastore изображений позволяет сохранить большие данные изображения, включая данные, которые не умещаются в памяти, и эффективно считают пакеты изображений во время обучения сверточной нейронной сети.

```
digitDatasetPath = fullfile(matlabroot,'toolbox','nnet','nndemos', ...
    'nndatasets','DigitDataset');
imds = imageDatastore(digitDatasetPath, ...
    'IncludeSubfolders',true,'LabelSource','foldernames');
Отобразите некоторые изображения в datastore.
figure;
perm = randperm(10000,20);
for i = 1:20
    subplot(4,5,i);
    imshow(imds.Files{perm(i)});
end
```

Вычислите количество изображений в каждой категории. labelCount – таблица, которая содержит метки и количество изображений, имеющих каждую метку. Datastore содержит 1 000 изображений для каждой из цифр 0–9 для в общей сложности 10 000 изображений. Можно задать количество классов в последнем полносвязном слое сети как OutputSize аргумент.

```
labelCount = countEachLabel(imds)
labelCount=10×2 table
    Label    Count

    _____    _____

     0       1000
```

```
1    1000
2    1000
3    1000
4    1000
5    1000
6    1000
7    1000
8    1000
9    1000
```

Необходимо задать размер изображений во входном слое сети. Проверяйте размер первого изображения в digitData. Каждое изображение 28 28 на 1 пиксель.

```
img = readimage(imds,1);
size(img)
ans = 1×2

     28    28
```

Задание наборов обучения и валидации.

Разделите данные на наборы данных обучения и валидации так, чтобы каждая категория в наборе обучающих данных содержала 750 изображений, и набор валидации содержал остающиеся изображения от каждой метки. splitEachLabel разделяет datastore digitData в два новых хранилища данных, trainDigitData и valDigitData.

```
numTrainFiles = 750;
[imdsTrain,imdsValidation] = splitEachLabel(imds,numTrainFiles,'randomize');
```

Архитектура сети Define.

Задайте архитектуру сверточной нейронной сети.

```
layers = [
   imageInputLayer([28 28 1])

   convolution2dLayer(3,8,'Padding','same')
   batchNormalizationLayer
   reluLayer

   maxPooling2dLayer(2,'Stride',2)

   convolution2dLayer(3,16,'Padding','same')
   batchNormalizationLayer
   reluLayer

   maxPooling2dLayer(2,'Stride',2)

   convolution2dLayer(3,32,'Padding','same')
   batchNormalizationLayer
   reluLayer
```

```
    fullyConnectedLayer(10)
    softmaxLayer
    classificationLayer];
```

Задание опций обучения.

После определения сетевой структуры задайте опции обучения. Обучите сеть с помощью стохастического градиентного спуска с импульсом (SGDM) с начальным темпом обучения 0,01. Определите максимальный номер эпох к 4. Эпоха является полным учебным циклом на целом обучающем наборе данных. Контролируйте сетевую точность во время обучения путем определения данных о валидации и частоты валидации. Переставляйте данные каждую эпоху. Программное обеспечение обучает сеть на обучающих данных и вычисляет точность на данные о валидации равномерно во время обучения. Данные о валидации не используются, чтобы обновить сетевые веса. Включите график процесса обучения и выключите командное окно выход.

```
options = trainingOptions('sgdm', ...
    'InitialLearnRate',0.01, ...
    'MaxEpochs',4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',imdsValidation, ...
    'ValidationFrequency',30, ...
    'Verbose',false, ...
    'Plots','training-progress');
```

Обучение сети, используя обучающие данные.

Обучите сеть с помощью архитектуры, заданной layers, обучающие данные и опции обучения.

График процесса обучения показывает мини-пакетную потерю и точность, и потерю валидации, и точность. Потеря является перекрестной энтропийной потерей. Точность является процентом изображений, которые сеть классифицирует правильно.

```
net = trainNetwork(imdsTrain,layers,options);
```

Классификация изображений, валидация и вычисление точности классификации.

Предскажите метки данных о валидации с помощью обучившего сеть, и вычислите итоговую точность валидации. Точность является частью меток, которые сеть предсказывает правильно. В этом случае больше, чем 99 % предсказанных меток совпадают с истинными метками набора валидации.

```
YPred = classify(net,imdsValidation);
YValidation = imdsValidation.Labels;
```

```
accuracy = sum(YPred == YValidation)/numel(YValidation)
    accuracy = 0.9988
```

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по работе в пакете Neural Network Toolbox в среде MATLAB.

Студент должен сделать следующее.

1 Загрузить и исследовать изображения.

2 Задать опции обучения.

3 Обучить сети.

4 Предсказать метки новых данных и вычислить точность классификации.

5 Оформить отчет.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) листинг выполненного индивидуального задания;

3) выводы.

Контрольные вопросы

1 Что такое нейронная сеть?

2 Какими характеристиками обладает нейронная сеть?

3 Из каких частей состоит нейронная сеть?

4 Как запрограммировать нейронную сеть?

5 Как проводить обучение нейронной сети?

6 Как минимизировать потери нейронной сети?

7 Для чего необходим пакет Neural Network Toolbox в среде MATLAB?

8 Какие функции выполняет пакет Neural Network Toolbox в среде MATLAB?

9 Что такое персептрон?

10 Как создать новую нейронную сеть в Neural Network Toolbox?

11 Как настроить новую нейронную сеть в Neural Network Toolbox?

12 Как обучить нейронную сеть в Neural Network Toolbox?

13 Как получить результаты адаптации нейронной сети в Neural Network Toolbox?

7 Лабораторная работа № 6. Лингвистический анализатор системы искусственного интеллекта для электромобиля

Цель работы: получение навыков по работе с лингвистическим анализатором системы ИИ для электромобиля.

Методические рекомендации по проведению исследований

Рассмотрим несколько задач, связанных с лингвистическим анализатором.

Задача подсчета частоты употребления определенных букв в английских и русских текстах.

```
Импорт и начальные переменные
import matplotlib.pyplot as plt; plt.rcdefaults()
import numpy as np
import matplotlib.pyplot as plt
from tkinter import *
from tkinter.filedialog import *
from tkinter.messagebox import *
import fileinput
import matplotlib as mpl

mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

Открытие файла с английским текстом

def w_open_ing():
    aa=ord('a')
    bb=ord('z')
    op = askopenfilename()
    main(op,aa,bb)

Открытие файла с русским текстом

def w_open_rus():
    aa=ord('а')
    bb=ord('ё')
    op = askopenfilename()
    main(op,aa,bb)

Универсальная обработка данных для обоих языков

def main(op,aa,bb):
    alpha = [chr(w) for w in range(aa,bb+1)]   #обратное преобразование кода в символы
    f = open(op , 'r')
    text = f.read()
    f.close()
```

```
        alpha_text = [w.lower() for w in text if w.isalpha()]    #выбор только букв и привидение их к нижнему регистру
        k={}      #создание словаря для подсчета каждой буквы
        for i in alpha:     #заполнение словаря
            alpha_count =0
            for item in alpha_text:
                if item==i:
                    alpha_count = alpha_count + 1
            k[i]= alpha_count
        z=0
        for i in   alpha:      #графическая визуализация данных в поле формы
            z=z+k[i]
        a_a=[]
        b_b=[]
        t= ('|\tletter\t|\tcount\t|\tpercent,%\t\n')
        txt.insert(END,t)
        t=('|---------------------------|---------------------------|-------------------------|\n')
        txt.insert(END,t)
        for i in  alpha:       #графическая визуализация данных в поле формы
            persent = round(k[i] * 100.0 / z,2)
            t=( '|\t%s\t|\t%d\t|\t%s\t\n' % (i, k[i], persent))
            txt.insert(END,t)
            a_a.append(i)
            b_b.append(k[i])
        t=('|---------------------------|---------------------------|-------------------------|\n' )
        txt.insert(END,t)
        t=('Total letters: %d\n' % z)
        txt.insert(END,t)
        people=a_a     #подготовка данных для построения диаграммы
        y_pos = np.arange(len(people))
        performance =b_b     #подготовка данных для построения диаграммы
        plt.barh(y_pos, performance)
        plt.yticks(y_pos, people)
        plt.xlabel('Quantity(amount) of the uses of the letter in the text')
        plt.title('The letters of the alphabet')
        plt.show()       #визуализация диаграммы

    Очистка поля

    def clear_text():
        txt.delete(1.0, END)

    Запись данных из поля в файл

    def save_file():
        save_as = asksaveasfilename()
        try:
            x =txt.get(1.0, END)
            f = open(save_as, "w")
            f.writelines(x.encode('utf8'))
            f.close()
        except:
```

```
            pass
```

Закрытие программы

```
def close_win():
    if askyesno("Exit", "Do you want to quit?"):
        tk.destroy()
```

Стандартный интерфейс ткинтер

```
tk= Tk()
main_menu = Menu(tk)
tk.config(menu=main_menu)
file_menu = Menu(main_menu)
main_menu.add_cascade(label="Aphabet", menu=file_menu)
file_menu.add_command(label="English text", command= w_open_ing)
file_menu.add_command(label="Russian text", command= w_open_rus)
file_menu.add_command(label="Save file", command=save_file)
file_menu.add_command(label="Cleaning", command=clear_text)
file_menu.add_command(label="Exit", command=close_win)
txt = Text(tk, width=72,height=10,font="Arial 12",wrap=WORD)
txt.pack()
tk.mainloop()
```

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по работе с лингвистическим анализатором.

Студент должен сделать следующее.

1 С помощью нейросети загрузить и прочитать текст на английском и русском языках.

2 Проанализировать текст.

3 Выявить наиболее часто встречающиеся буквы.

4 Выявить самые массово встречающиеся и самые длинные слова.

5 Оформить отчет.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) листинг выполненного индивидуального задания;

3) выводы.

Контрольные вопросы

1 Что такое лингвистический анализатор?

2 Для чего необходим лингвистический анализатор?

3 Какие функции выполняет лингвистический анализатор?

4 Какие основные части содержит лингвистический анализатор?
5 Какие есть виды анализаторов?
6 Что такое токенизация текста?
7 Что такое стоп-слова?
8 Что собой представляет процесс нормализации?
9 Что такое стемминг?
10 Что такое лемматизация?
11 Что такое векторизация текста?

8 Лабораторная работа № 7. Обнаружение и отслеживание объектов системой искусственного интеллекта

Цель работы: изучение способов обнаружения и отслеживания ИИ объектов в пространстве на языке программирования Python.

Методические рекомендации по проведению исследований

Распознавание объектов

```
# USAGE
# Python real_time_object_detection.py --prototxt MobileNetSSD_deploy.prototxt.txt --model MobileNetSSD_deploy.caffemodel

# import the necessary packages
from imutils.video import VideoStream
from imutils.video import FPS
import numpy as np
import argparse
import imutils
import time
import cv2

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-p", "--prototxt", required=True,
  help="path to Caffe 'deploy' prototxt file")
ap.add_argument("-m", "--model", required=True,
  help="path to Caffe pre-trained model")
ap.add_argument("-c", "--confidence", type=float, default=0.2,
  help="minimum probability to filter weak detections")
args = vars(ap.parse_args())

# initialize the list of class labels MobileNet SSD was trained to
# detect, then generate a set of bounding box colors for each class
CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",
  "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
  "dog", "horse", "motorbike", "person", "pottedplant", "sheep",
  "sofa", "train", "tvmonitor"]
```

```
COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))

# load our serialized model from disk
print("[INFO] loading model...")
net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])

# initialize the video stream, allow the cammera sensor to warmup,
# and initialize the FPS counter
print("[INFO] starting video stream...")
vs = VideoStream(src=0).start()
time.sleep(2.0)
fps = FPS().start()

# loop over the frames from the video stream
while True:
  # grab the frame from the threaded video stream and resize it
  # to have a maximum width of 400 pixels
  frame = vs.read()
  frame = imutils.resize(frame, width=400)

  # grab the frame dimensions and convert it to a blob
  (h, w) = frame.shape[:2]
  blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),
          0.007843, (300, 300), 127.5)

  # pass the blob through the network and obtain the detections and
  # predictions
  net.setInput(blob)
  detections = net.forward()

  # loop over the detections
  for i in np.arange(0, detections.shape[2]):
          # extract the confidence (i.e., probability) associated with
          # the prediction
          confidence = detections[0, 0, i, 2]

          # filter out weak detections by ensuring the `confidence` is
          # greater than the minimum confidence
          if confidence > args["confidence"]:
                  # extract the index of the class label from the
                  # `detections`, then compute the (x, y)-coordinates of
                  # the bounding box for the object
                  idx = int(detections[0, 0, i, 1])
                  box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
                  (startX, startY, endX, endY) = box.astype("int")

                  # draw the prediction on the frame
                  label = "{}: {:.2f}%".format(CLASSES[idx],
                          confidence * 100)
                  cv2.rectangle(frame, (startX, startY), (endX, endY),
                          COLORS[idx], 2)
                  y = startY - 15 if startY - 15 > 15 else startY + 15
```

```
                cv2.putText(frame, label, (startX, y),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)

    # show the output frame
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

    # if the `q` key was pressed, break from the loop
    if key == ord("q"):
        break

    # update the FPS counter
    fps.update()

# stop the timer and display FPS information
fps.stop()
print("[INFO] elapsed time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()Распознавание лиц
# USAGE
# Python real_time_object_detection.py --prototxt MobileNetSSD_deploy.prototxt.txt --model
MobileNetSSD_deploy.caffemodel
```

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по изучению способов обнаружения и отслеживания ИИ объектов в пространстве на языке программирования Python.

Студент должен сделать следующее.

1 Создать нейросеть для работы с веб-камерой.

2 Запустить программу и убедиться в ее работоспособности.

3 Добавить функционал для различения предметов.

4 Добавить функционал для различения лиц.

5 Оформить отчет.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) листинг выполненного индивидуального задания;

3) выводы.

Контрольные вопросы

1 Что такое компьютерное зрение?
2 Какие функции преследует техническое зрение?
3 Как получить доступ к веб-камере/видеопотоку?
4 Как применить распознавание объекта для каждого кадра?
5 Как инициализировать список классов и набор цветов?
6 Как загрузить модель и настроить видеопоток?
7 Как работать с кадрами?
8 Как проводить фильтрацию объектов?
9 Как отобразить кадр?
10 Как запустить программу для обнаружения и слежения за объектами?

9 Лабораторная работа № 8. Моделирование мультиагентной системы искусственного интеллекта

Цель работы: получение навыков по моделированию мультиагентной системы.

Методические рекомендации по проведению исследований

Основными элементами мультиагентной системы являются Агенты. Мультиагентная платформа состоит из контейнеров, в которых «живут» Агенты. Контейнер может не содержать ни одного агента или содержать сколько угодно агентов. Каждый контейнер обладает своим сетевым адресом и именем. Есть особый контейнер – главный контейнер – без него платформа не работает. Главный контейнер имеет несколько особенностей. Перечислим их:

– он должен быть создан первым;

– он должен включать в себя двух специальных агентов:

1) AMS (agent management system) – агент, который управляет остальными агентами; он способен создавать и останавливать агентов;

2) DF (directory facilitation) – по сути представляет из себя желтые страницы, в которых записываются адреса, имена и возможности агентов в системе.

Для запуска платформы необходимо запустить Eclipse IDE, создать новую рабочую среду и в ней открыть новый Java проект (рисунок 10).

Далее переходим в окно создания проекта, задаем имя и нажимаем Next. В следующем окне необходимо подключить к нашему проекту библиотеки Jade (рисунок 11).

Для подключения библиотек необходимо нажать Add External JARs и указать путь к библиотекам. Они находятся в папке \lib, куда были скопированы все файлы архивов платформы Jade. После этого, нажимаем Finish.

Теперь к проекту подключены библиотеки JADE. Создаем нового Агента из примера «PingAgent». Для этого необходимо создать новый класс в пап-

ке \src (рисунок 12).

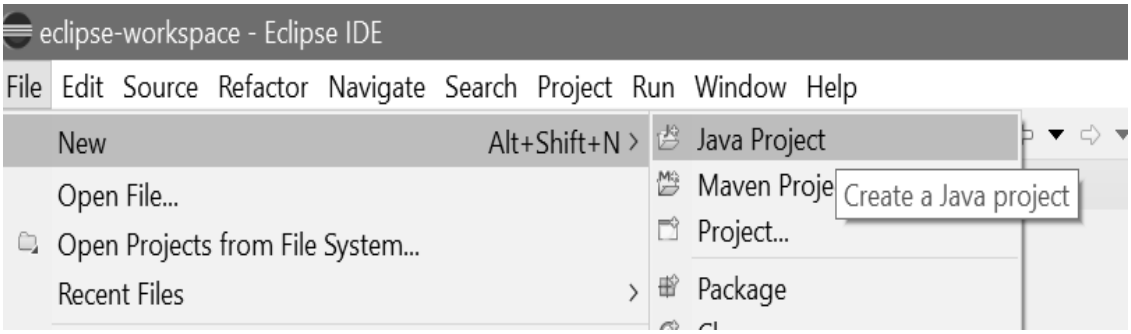


Рисунок 10 – Новый Java проект в Eclipse IDE

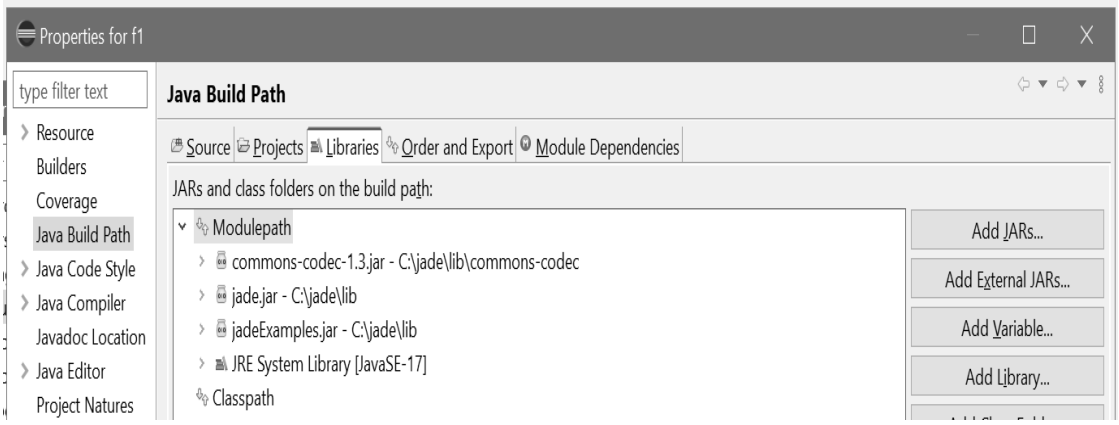


Рисунок 11 – Библиотеки Jade в проекте

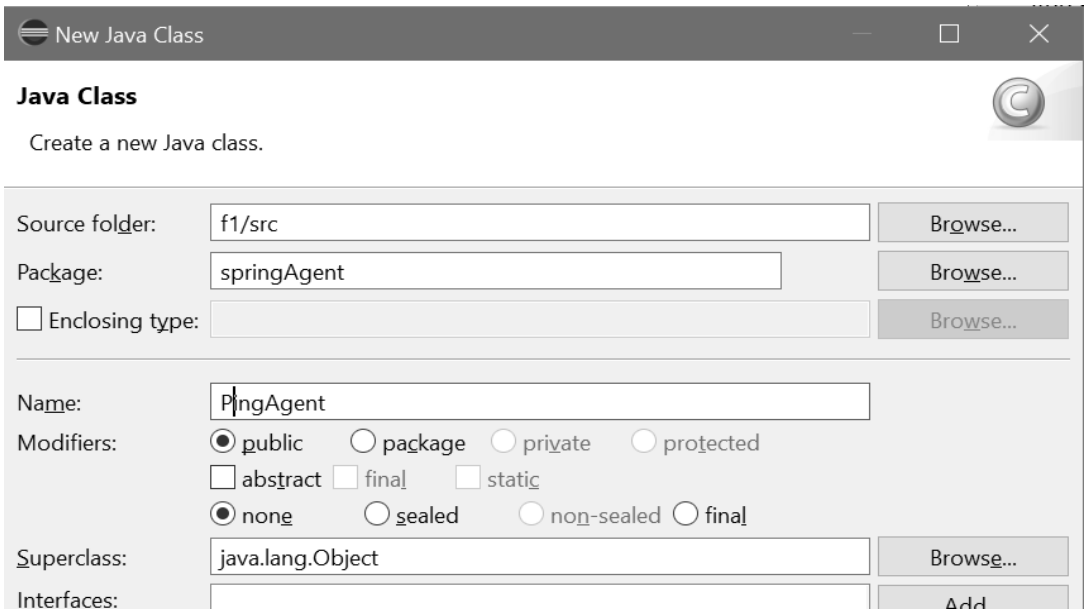


Рисунок 12 – Новый класс

Теперь вводим информацию, создаем новый класс в папку Project/src в пакете sprintintel с именем PingAgent. Также на этом этапе необходимо записать в поле «Superclass» следующую информацию: java.lang.Agent и нажать Finish.

Сейчас у нас есть новый Агент и подключены необходимые библиотеки. Необходимо задать программный код Агента. Для этого открываем PingAgent.java из папки examples\PingAgent, которая находится в директории, в которую распакованы архивы Jade. Копируем все содержимое примера нашего агента. Посмотрим на код нашего агента, в нем есть ошибки, которые необходимо исправить (рисунок 13).

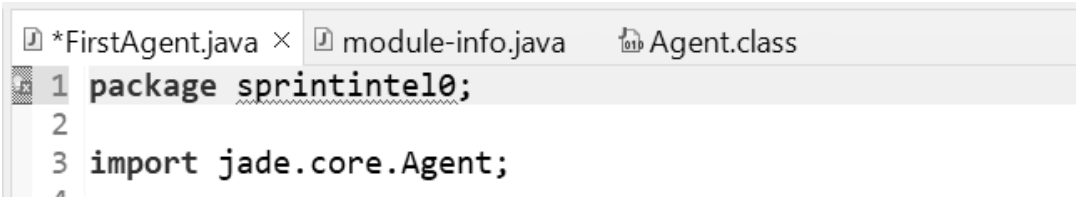


Рисунок 13 – Ошибка в коде Агента

Эта ошибка вполне понятна. Необходимо исправить имя пакета, который мы используем. В нашем случае – это sprintintel. Далее необходимо обязательно сохранить изменения в файле Агента.

Для того чтобы запустить Агента и мультиагентную платформу, переходим в меню Run Configuration. В нем добавляем новую конфигурацию запуска Java-приложения. На рисунках 14 и 15 показаны параметры главного класса (MainClass)–jade.Boot и видно название настоящего проекта. В поле Program Argument вкладки Arguments вводим значении «-gui». Таким образом, мы определили, что нам необходимо запустить платформу и главный контейнер, который уже содержит дополнительный класс нашего агента PingAgent.

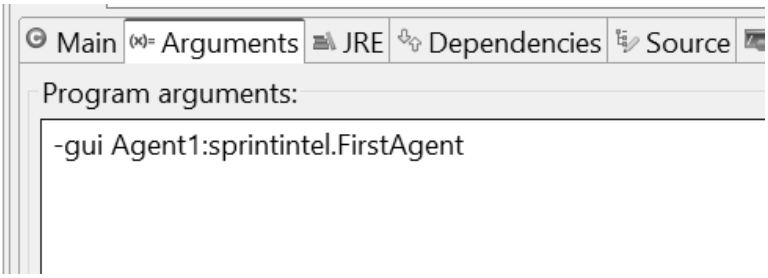


Рисунок 14 – Конфигурация запуска, вкладка Arguments

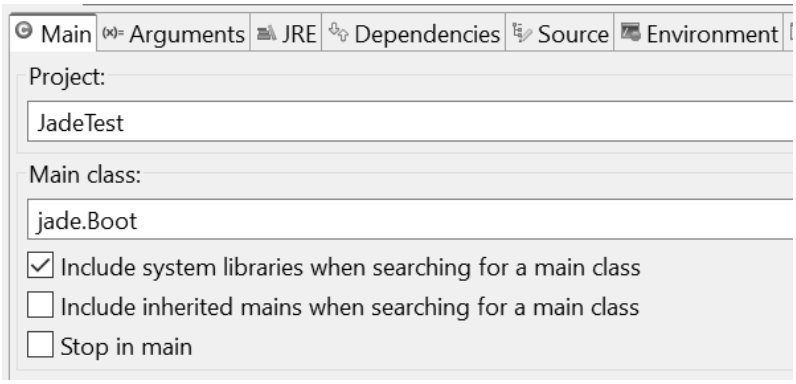


Рисунок 15 – Конфигурация запуска. Вкладка Main

Нажимаем на кнопку Run и запускаем нашу платформу. Платформа запущена и мы видим пользовательский интерфейс, в котором запущена платформа и два базовых агента AMS и DF. Чтобы добавить в главный контейнер нашего Агента «Ping», необходимо войти во вкладку Action и добавить класс нашего Агента (рисунки 16 и 17).

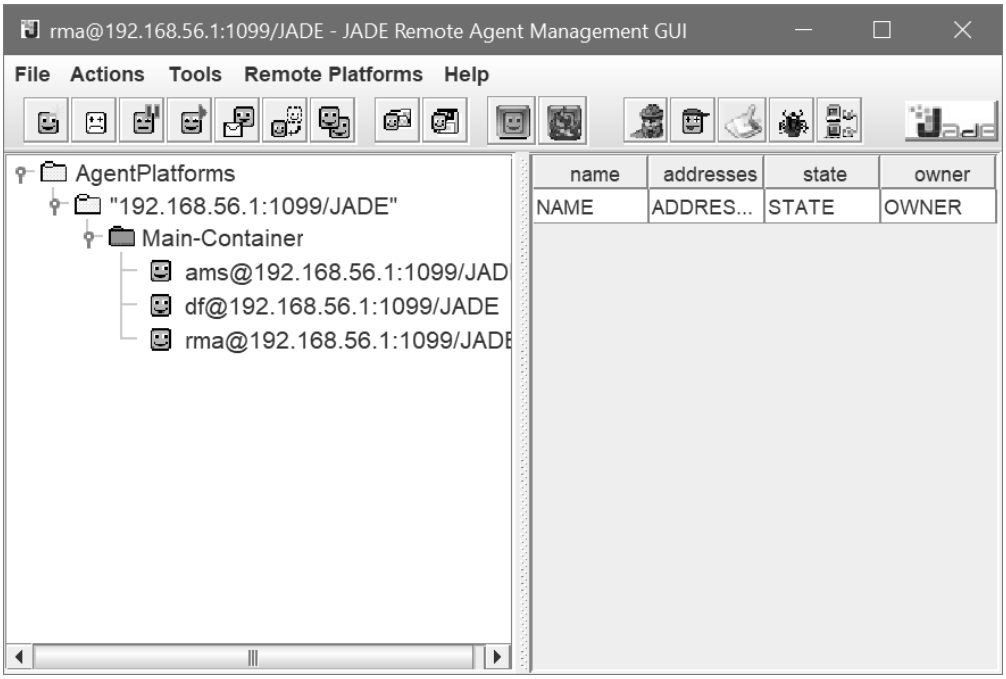


Рисунок 16 – Пользовательский интерфейс платформы Jade

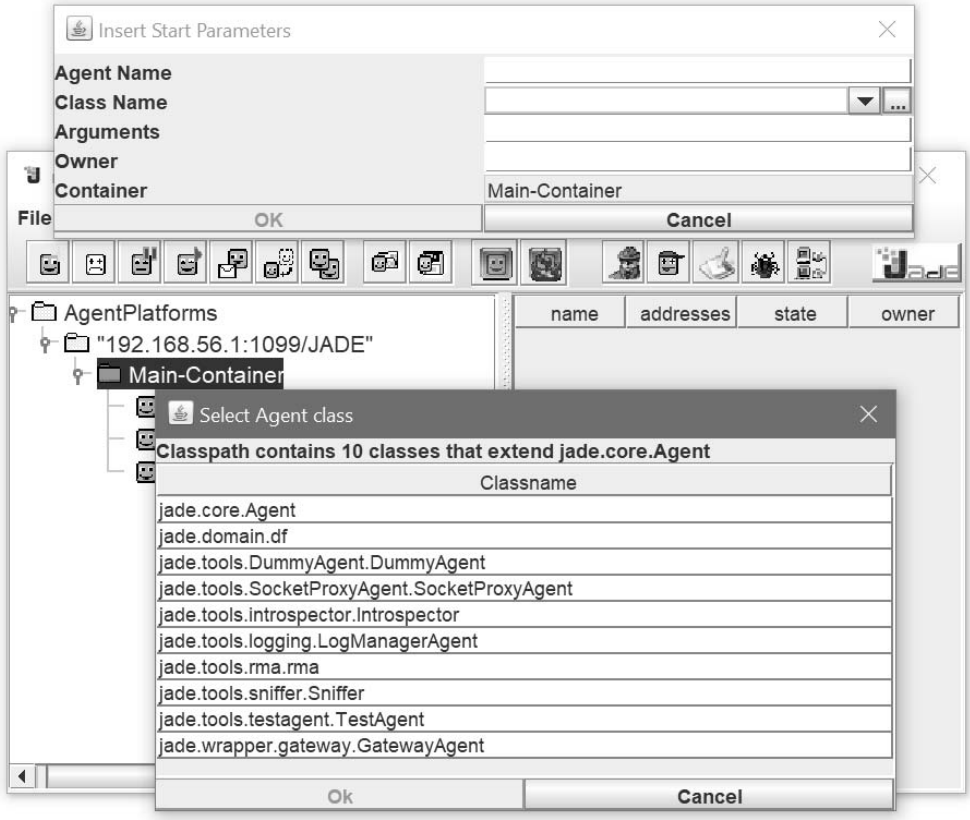


Рисунок 17 – Class Агента Ping Agent в пакете sprintintel

При запуске агента в главном контейнере он обращается в «желтые страницы» – агент DF, который записывает у себя адрес этого агента. Все запросы осуществляются через DF агента. Таким образом, в систему могут входить и выходить новые агенты и быть «на связи».

Теперь проверим работу нашего Агента и отправим ему запрос с помощью DummyAgent – готовый агент с пользовательским интерфейсом, посредством которого возможно отправлять сообщения на другие агенты и получать от них ответы. Для запуска DummyAgent необходимо нажать кнопку StartDummyAgent в ряду кнопок быстрого доступа пользовательского интерфейса платформы. В самом Агенте необходимо составить запрос в правильной форме, как это показано на рисунке 18.

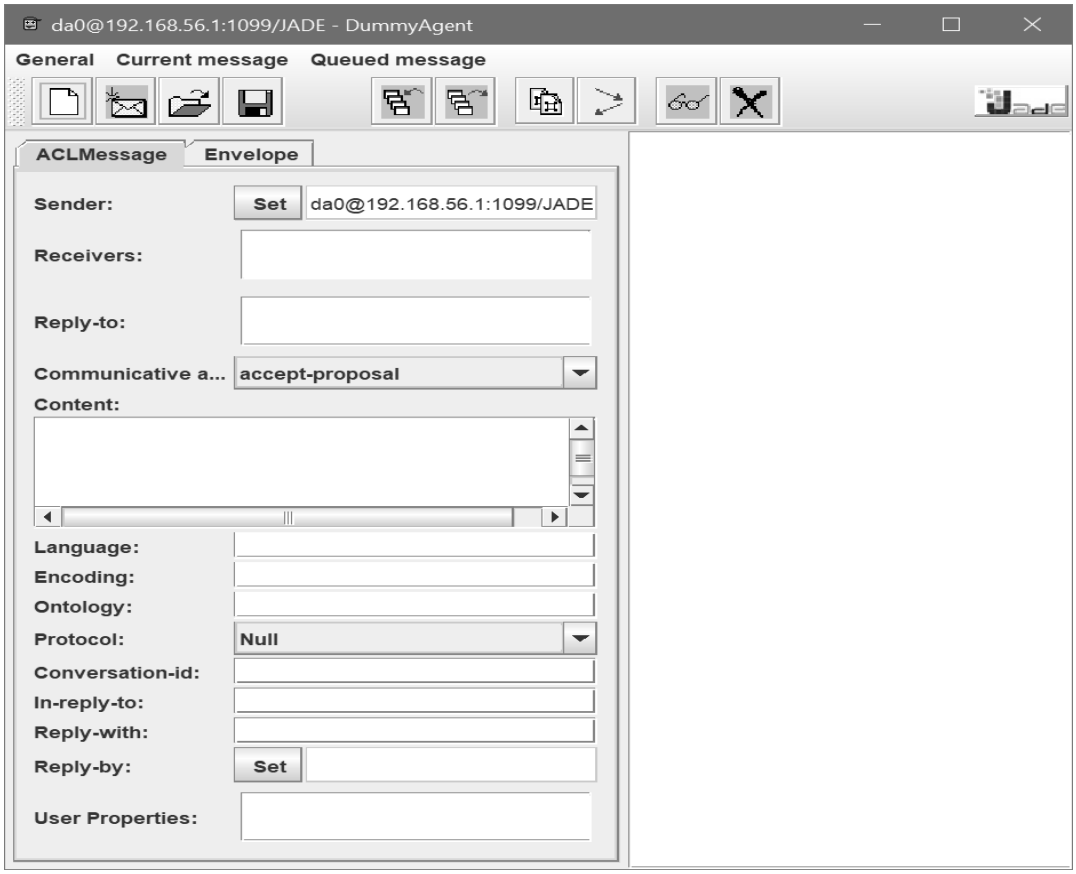


Рисунок 18 – DummyAgent, запрос на Ping агента

После правильного заполнения запроса необходимо нажать на кнопку «Отправить сообщение». DummyAgent позволяет отправляет запрос с одного точного адреса агента на точный адрес другого агента, даже если они находятся в различных контейнерах. Но эти агенты должны быть прописаны в одной мультиагентной системе (т. е. известны DF агентам). Справа, нажав на значок очки, в белом поле пользовательского интерфейса DummyAgent, можно увидеть, пришел ли ответ и какого он содержания.

Для того чтобы проще было отслеживать отправку сообщений, можно запустить еще одного агента Sniffer, в котором отражаются стрелочками отправленные сообщения между агентами (рисунок 19).

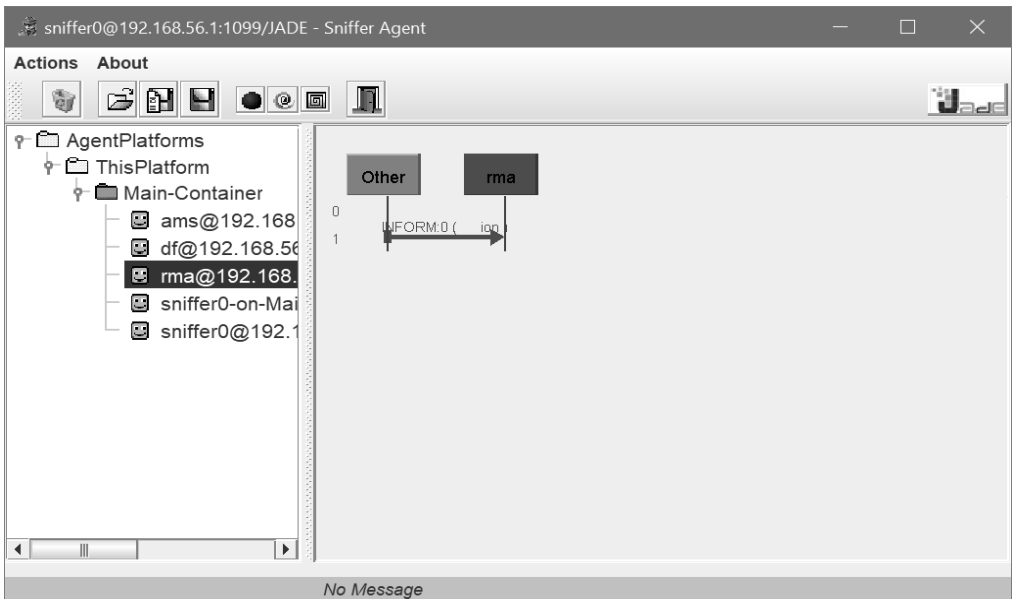


Рисунок 19 – Агент Sniffer

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по моделированию мультиагентной системы.

Студент должен сделать следующее.

1 Создать еще одного агента Ping1 и отправить запрос двум агентам одновременно.

2 Изменить текст сообщения запроса и ответа.

3 Создать Container1 (не главный) и создать в нем агента.

4 Запустить агентов в примере папки src/examples/bookTrading.

5 Оформить отчет.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) листинг выполненного индивидуального задания;

3) выводы.

Контрольные вопросы

1 Что такое мультиагентные системы?

2 Какие цели преследуют мультиагентные системы?

3 Какие задачи решаются с помощью мультиагентных систем?

4 Какие есть методы и технологии возможных решений создания мультиагентных моделей?

5 Какие есть международные подходы к созданию мультиагентных систем?

6 Из каких структурных частей состоят мультиагентные системы?

7 Опишите архитектуру мультиагентных систем.
8 Опишите этапы построения мультиагентных систем.

10 Лабораторная работа № 9. Аппаратные средства системы искусственного интеллекта для электромобиля

Цель работы: изучение аппаратных средств системы ИИ для электромобиля.

Методические рекомендации по проведению исследований

Пример кода для обучения нейросети распознавания букв:

```
import numpy as np
# visualizing the data, ploting A.

# A
a =[0, 0, 1, 1, 0, 0,
  0, 1, 0, 0, 1, 0,
  1, 1, 1, 1, 1, 1,
  1, 0, 0, 0, 0, 1,
  1, 0, 0, 0, 0, 1]
# B
b =[0, 1, 1, 1, 1, 0,
  0, 1, 0, 0, 1, 0,
  0, 1, 1, 1, 1, 0,
  0, 1, 0, 0, 1, 0,
  0, 1, 1, 1, 1, 0]
# C
c =[0, 1, 1, 1, 1, 0,
  0, 1, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0,
  0, 1, 1, 1, 1, 0]

# A
a1 =[1, 0, 1, 1, 0, 0,
  0, 1, 0, 0, 1, 0,
  1, 1, 0, 1, 1, 1,
  1, 0, 1, 0, 0, 1,
  1, 0, 1, 0, 0, 1]
# B
b1 =[0, 1, 1, 1, 1, 0,
  0, 1, 1, 0, 1, 0,
  0, 1, 1, 1, 1, 0,
  0, 1, 1, 0, 1, 0,
  0, 0, 0, 1, 1, 0]
# C
c1 =[0, 0, 1, 1, 1, 0,
```

```
   0, 1, 1, 0, 0, 0,
   0, 1, 1, 0, 0, 0,
   0, 1, 0, 1, 0, 0,
   0, 1, 1, 1, 1, 0]

# Creating labels
y =[[1, 0, 0],
   [0, 1, 0],
   [0, 0, 1]]


x =[np.array(a).reshape(1, 30), np.array(b).reshape(1, 30),
                        np.array(c).reshape(1, 30)]


x1 =[np.array(a1).reshape(1, 30), np.array(b1).reshape(1, 30),
                        np.array(c1).reshape(1, 30)]

# Labels are also converted into NumPy array
y = np.array(y)


# activation function

def sigmoid(x):
   return(1/(1 + np.exp(-x)))


# Creating the Feed forward neural network
# 1 Input layer(1, 30)
# 1 hidden layer (1, 5)
# 1 output layer(3, 3)

def f_forward(x, w1, w2):
   # hidden
   z1 = x.dot(w1)# input from layer 1
   a1 = sigmoid(z1)# out put of layer 2

   # Output layer
   z2 = a1.dot(w2)# input of out layer
   a2 = sigmoid(z2)# output of out layer
   return(a2)


# initializing the weights randomly
def generate_wt(x, y):
   l =[]
   for i in range(x * y):
      l.append(np.random.randn())
   return(np.array(l).reshape(x, y))

# for loss we will be using mean square error(MSE)
```

```
def loss(out, Y):
    s =(np.square(out-Y))
    s = np.sum(s)/len(y)
    return(s)

# Back propagation of error
def back_prop(x, y, w1, w2, alpha):

    # hidden layer
    z1 = x.dot(w1)# input from layer 1
    a1 = sigmoid(z1)# output of layer 2

    # Output layer
    z2 = a1.dot(w2)# input of out layer
    a2 = sigmoid(z2)# output of out layer
    # error in output layer
    d2 =(a2-y)
    d1 = np.multiply((w2.dot((d2.transpose()))).transpose(),
                         (np.multiply(a1, 1-a1)))

    # Gradient for w1 and w2
    w1_adj = x.transpose().dot(d1)
    w2_adj = a1.transpose().dot(d2)

    # Updating parameters
    w1 = w1-(alpha*(w1_adj))
    w2 = w2-(alpha*(w2_adj))

    return(w1, w2)

def train(x, Y, w1, w2, alpha = 0.01, epoch = 10):
    acc =[]
    losss =[]
    for j in range(epoch):
        l =[]
        for i in range(len(x)):
            out = f_forward(x[i], w1, w2)
            l.append((loss(out, Y[i])))
            w1, w2 = back_prop(x[i], y[i], w1, w2, alpha)
        print("epochs:", j + 1, "======== acc:", (1-(sum(l)/len(x)))*100)
        acc.append((1-(sum(l)/len(x)))*100)
        losss.append(sum(l)/len(x))
    return(acc, losss, w1, w2)


def predict(x, w1, w2):
    Out = f_forward(x, w1, w2)
    maxm = 0
    k = 0
    for i in range(len(Out[0])):
        if(maxm<Out[0][i]):
            maxm = Out[0][i]
```

```
        k = i
    if(k == 0):
        print("Image is of letter A.")
    elif(k == 1):
        print("Image is of letter B.")
    else:
        print("Image is of letter C.")

    print("Accuracy:")
    print("A: " + str(Out[0][0]))
    print("B: " + str(Out[0][1]))
    print("C: " + str(Out[0][2]))

    #plt.imshow(x.reshape(5, 6))
    #plt.show()


w1 = generate_wt(30, 5)
w2 = generate_wt(5, 3)

acc, losss, w1, w2 = train(x, y, w1, w2, 0.1, 100)

predict(x1[0], w1, w2)
predict(x1[1], w1, w2)
predict(x1[2], w1, w2)
```

Индивидуальное задание

Преподаватель, проводящий лабораторное занятие, выдает индивидуальное задание по изучению аппаратных средств системы ИИ для электромобиля.

Студент должен выполнить следующее.

1 Сформировать три матрицы размерности 6×5, в которых будут зашифрованы буквы русского алфавита следующим образом: 0 – белый пиксель; 1 – черный пиксель.

2 Полученные матрицы вставить в программный код, который на данный момент уже набран и отлажен.

3 Выполнить программный код.

4 Проанализировать работу кода, алгоритм распознавания букв и обучения нейросети.

Содержание отчета

Отчет должен содержать:

1) цель работы;

2) листинг выполненного индивидуального задания;

3) выводы.

Контрольные вопросы

1 Что такое технология AI (Artificial Intelligence)?
2 Что такое технология ML (Machine Learning)?
3 Что такое интернет вещей (IoT –Internet of Things)?
4 Что такое промышленный интернет вещей (IIoT – Industrial Internet of Things)?
5 Какую ценность приносят технологии AL и ML для IoT (IIoT)?
6 Что такое искусственный интеллект вещей (AIoT)?
7 Что такое искусственный интеллект?
8 Что такое машинное обучение?
9 Что такое глубокое обучение?
10 Какие есть практические области применения приложений IoT?
11 Какие есть аппаратные и программные требования к приложению IoT?
12 Какие есть инструменты разработки для приложений IoT?

Список литературы

1 **Андрейчиков, А. В.** Интеллектуальные информационные системы и методы искусственного интеллекта: учебник / А. В. Андрейчиков, О. Н. Андрейчикова. – Москва: ИНФРА-М, 2022. – 530 с.

2 **Пенькова, Т. Г.** Модели и методы искусственного интеллекта: учебное пособие / Т. Г. Пенькова, Ю. В. Вайнштейн. – Красноярск: Сиб. федер. ун-т, 2019. – 116 с.

3 **Варламов, О. О.** Основы создания миварных экспертных систем: учебное пособие / О. О. Варламов. – Москва: ИНФРА-М, 2021. – 267 с.

4 **Варламов, О. О.** 18 примеров миварных экспертных систем: учебное пособие / О. О. Варламов. – Москва: ИНФРА-М, 2021. – 630 с.

5 **Авдеенко, Т. В.** Введение в искусственный интеллект и логическое программирование. Программирование в среде Visual Prolog: учебное пособие / Т. В. Авдеенко, М. Ю. Целебровская. – Новосибирск: НГТУ, 2020. – 64 с.

6 **Манусов, В. З.** Применение методов искусственного интеллекта в задачах управления режимами электрических сетей Smart Grid : монография / В. З. Манусов, Н. Хасанзода, П. В. Матренин. – Новосибирск: НГТУ, 2019. – 240 с.

7 Применение методов искусственного интеллекта в задачах технической диагностики электрооборудования электрических систем: монография / Под общ. ред. В. З. Манусова. – Новосибирск: НГТУ, 2020. – 446 с.

8 **Пятаева, А. В.** Интеллектуальные системы и технологии: учебное пособие / А. В. Пятаева, К. В. Раевич. – Красноярск: Сиб. федер. ун-т, 2018. – 144 с.