Пояснительная записка

Тема проекта: Разработка и тестирование приложения для анализа текста в промышленности с использованием машинного обучения.

Наименование проекта: Интеллектуальный помощник взаимодействия с обществом для Федерального центра прикладного развития искусственного интеллекта Минпромторга России

Автор: Мухина Мария Дмитриевна

Москва, 2023 г.

Оглавление

Введение

Предпосылки

ФГАУ «Федеральный центр прикладного развития искусственного интеллекта» (далее – ФЦПР), учрежденный Министерством промышленности и торговли Российской Федерации, имеет более 7 стратегических направлений деятельности, ключевым из которых является развитие Центра коллективного пользования ИИ Минпромторга России.

С целью сближения ФЦПР и потенциальных партнеров, заказчиков, экспертов ведутся работы по запуску масштабного сайта учреждения.

С целью повышения качества взаимодействия с пользователями сайта учреждения стоит задача внедрения на сайт интеллектуального помощника в формате чат-бота. Задачами функционирования такого чата-бота должны стать:

- Помощь пользователям в ориентации по сайту (не столько по наименованию страниц и разделов, сколько по контенту).
- Семантический анализ вводимых пользователями запросов и предоставление релевантных ответов.

Разработка такого чат-бота для промышленного внедрения выполняется специализированной командой разработчиков ФЦПР.

Однако в рамках выполнения настоящей аттестационной работы будут разработаны альтернативные решения, которые:

- Позволят создать инструмент для дополнительного тестирования решения, разрабатываемого ФЦПР;
- Позволит создать дополнительные дата-сеты;
- Позволит сформулировать критерии оценки качества разрабатываемой ФЦПР альтернативной модели;
- Позволит сформировать у автора данной отчетной работы необходимые компетенции для обоснованного проведения тестирования и приемо-сдаточных испытаний аналогичных систем и моделей.

Тема проекта «Разработка и тестирование приложения для анализа текста в промышленности с использованием машинного обучения» будет детализирована до разработки и тестирования интеллектуального помощника взаимодействия с обществом для Федерального центра прикладного развития искусственного интеллекта Минпромторга России.

Таким образом, реализация настоящей работы будет иметь практический эффект и будет использоваться в реальной деятельности автора.

Цели и задачи

Целью настоящей работы являлась разработка и тестирование приложения для анализа текста в промышленности с использованием машинного обучения.

Для реализации данной цели требовалось решить следующие задачи:

- Разработать приложения для анализа текста с использование машинного обучения;
- Протестировать приложение для анализа текста с использованием машинного обучения.

С целью реализации данных задач, максимально эффективным для автора настоящей работы методом – возможное практические применение разрабатываемых решений – было принято решение реализовать приложение в формате чата-бота, отвечающего на вопросы по реальным данным.

Предварительный состав работ выполнения настоящей аттестации:

- Формулирование задачи;
- Предварительный анализ существующих решений, доступных наборов данных;
- Формирование наборов данных;
- Подготовка программной среды;
- Обработка данных под задачу;
- Обработка данных перед загрузкой в модель;
- Проведение тренировки алгоритма машинного обучение;
- До настройка алгоритма;
- Оценка реализованного алгоритма;
- Визуализация результатов работы;
- Сохранение данных для использования.

Реализация

Постановка задачи

Создать чат-бот, который будет отвечать на вопросы, в соответствии с подготовленным текстом на русском языке.

Описание решения: причинно-следственная связь, логический вывод, Natural Language Inference, NLP. Настройка классификации текстовых пар аналогична существующим задачам логического вывода (NLI)

Тип задачи: Логика, Commonsense.

Предварительный анализ существующих решений, доступных наборов данных

Так как реализация планируется по только что созданной странице сайта, понятно, что исходных доступных наборов данных в открытых источниках нет.

Необходимо создать все наборы данных.

В ходе прохождения обучения, стало понятно, что существует весьма ограниченное число решений по анализу текста на русском языке.

Безусловно существуют коммерческие решения.

Однако в ходе реализации настоящей работы исследовались следующие решения:

1. https://github.com/Koziev/NLP_Datasets
2. https://natasha.github.io/
3. https://www.kaggle.com/datasets/stackoverflow/stacksample?select=Answers.csv
4. https://www.kaggle.com/datasets/stanfordu/stanford-question-answering-dataset
5. https://colab.research.google.com/drive/1tIC6zvx8KYCz_4zTQ0fKnmjleXbKeJDY?usp=sharing

Особое внимание было уделено данному кейсу: https://russiansuperglue.com/ru/tasks/task_info/DaNetQA .

Формирование наборов данных

В ходе реализации работы были сформировано несколько документов и наборов данных.

Исходный текст, по которому требовалось создать решение:

«Центр коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта» (далее – ЦКП МПТ ИИ, Центр, Центр коллективного пользования) – структурное подразделение ФГАУ «ФЦПР ИИ» (Федеральное государственное автономное учреждение «Федеральный центр прикладного развития искусственного интеллекта») создано с целью повышения доступности аппаратного обеспечении, необходимого для решения задач в области ИИ, а также апробации отечественных АПК.

ЦКП МПТ ИИ по направлению создания модели «Искусственный интеллект, как сервис», предоставляет следующие решения:

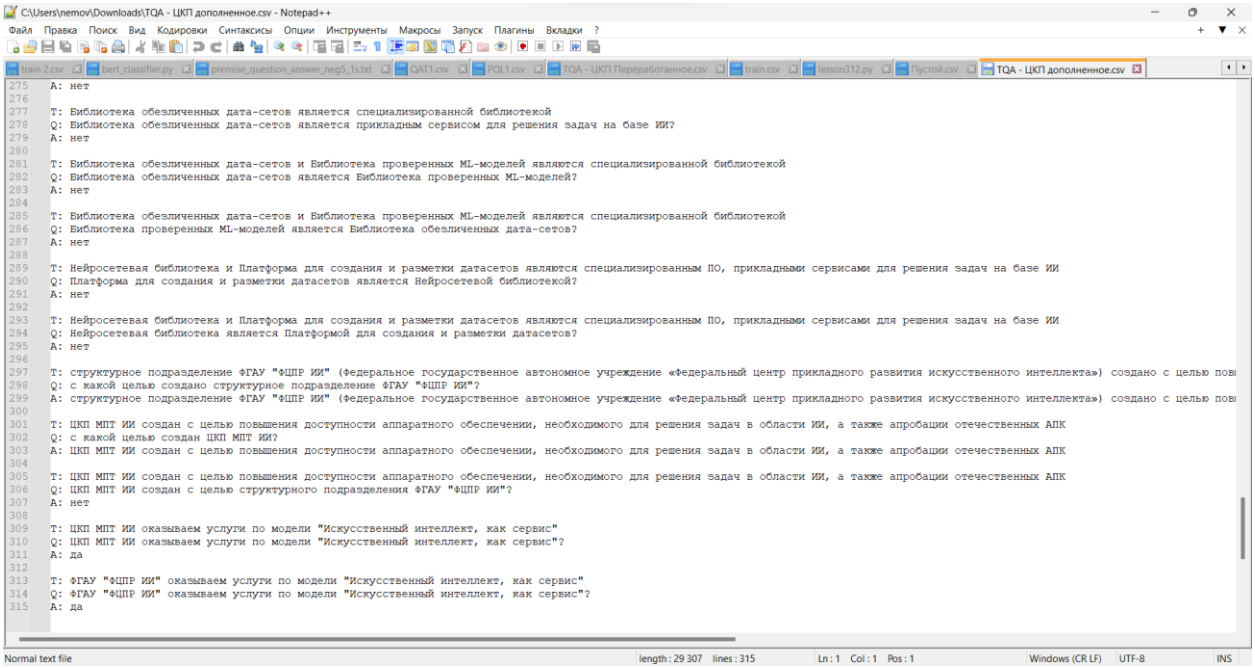
1. *Специализированное ПО, прикладные сервисы для решения задач на базе ИИ, а именно:*
   - *Платформа для создания и разметки датасетов;*
   - *Нейросетевая библиотека.*
2. *Специализированные библиотеки, а именно:*
   - *Библиотека проверенных ML-моделей;*
   - *Библиотека обезличенных дата-сетов;*
3. *Платформа взаимодействия с экспертами.*

https://aigov.ru/centre »

Текст для аттестационной рабо

Обучающий набор данных в формате Текст-Вопрос-Ответ в первой версии:

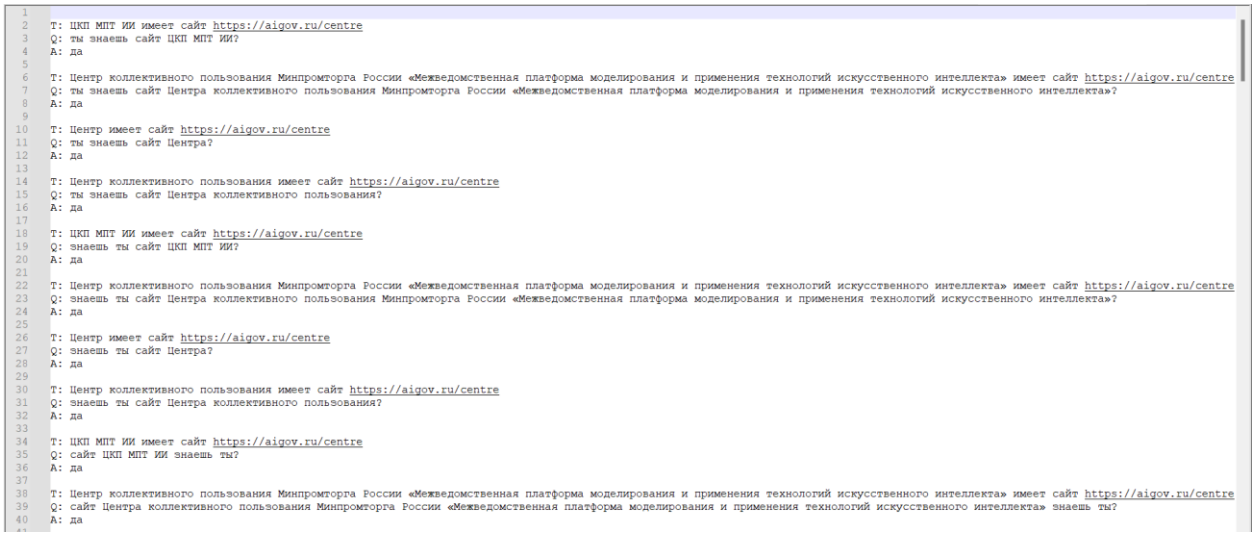
TQA - ЦКП дополненное.txt



Исходный состав набора данных включал ответа в формате «да» / «нет», а также развернутые ответы.

Кроме того, в исходном наборе данных были подготовлены примеры с

изменяющейся последовательностью слов.

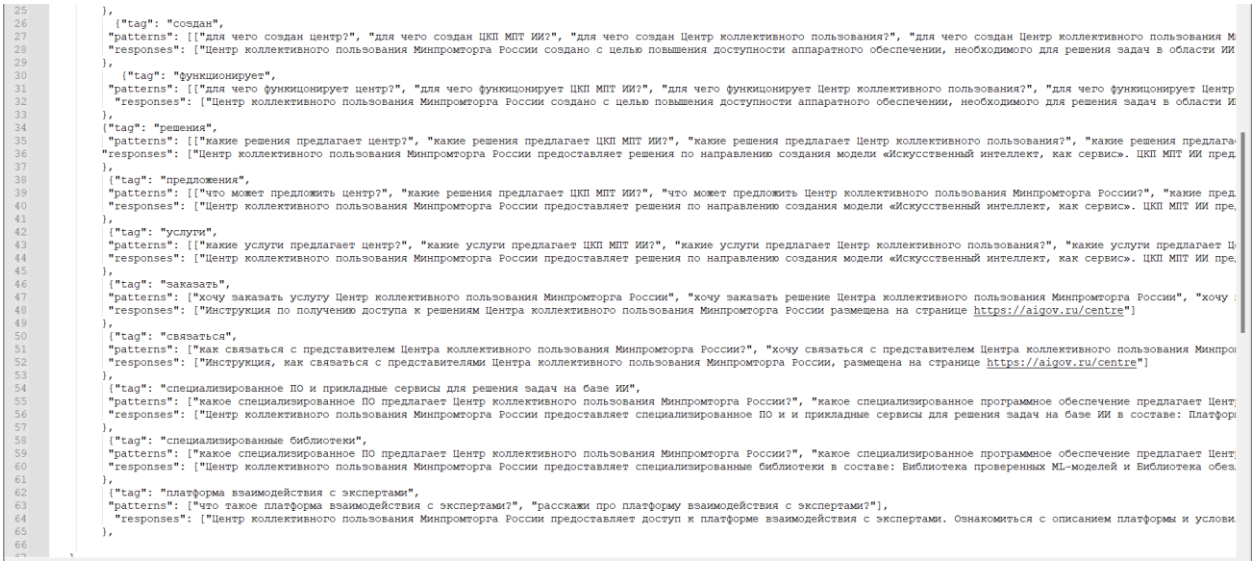


```
T: ЦКП МПТ ИИ имеет сайт https://aigov.ru/centre
Q: ты знаешь сайт ЦКП МПТ ИИ?
A: да

T: Центр коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта» имеет сайт https://aigov.ru/centre
Q: ты знаешь сайт Центра коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта»?
A: да

T: Центр имеет сайт https://aigov.ru/centre
Q: ты знаешь сайт Центра?
A: да

T: Центр коллективного пользования имеет сайт https://aigov.ru/centre
Q: ты знаешь сайт Центра коллективного пользования?
A: да

T: ЦКП МПТ ИИ имеет сайт https://aigov.ru/centre
Q: знаешь ты сайт ЦКП МПТ ИИ?
A: да

T: Центр коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта» имеет сайт https://aigov.ru/centre
Q: знаешь ты сайт Центра коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта»?
A: да

T: Центр имеет сайт https://aigov.ru/centre
Q: знаешь ты сайт Центра?
A: да

T: Центр коллективного пользования имеет сайт https://aigov.ru/centre
Q: знаешь ты сайт Центра коллективного пользования?
A: да

T: ЦКП МПТ ИИ имеет сайт https://aigov.ru/centre
Q: сайт ЦКП МПТ ИИ знаешь ты?
A: да

T: Центр коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта» имеет сайт https://aigov.ru/centre
Q: сайт Центра коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта» знаешь ты?
A: да
```

По итогу первых попыток работы с таким набором данных, стало ясно, что был выбран не корректный подход.

Тогда набор данных был скорректирован в формате «Tag-Patterns- Responses». Были добавлены только развернутые ответы, способствующие побуждению будущего пользователя чат-бота к диалогу. Кроме того, стало понятно, что время на перестановку слов в предложениях при формировании предыдущего набора данных, было потрачено впустую.

TQA - ЦКП Переработанное.csv



```
    ),
     {"tag": "создан",
     "patterns": [["для чего создан центр?", "для чего создан ЦКП МПТ ИИ?", "для чего создан Центр коллективного пользования?", "для чего создан Центр коллективного пользования М
     "responses": ["Центр коллективного пользования Минпромторга России создано с целью повышения доступности аппаратного обеспечении, необходимого для решения задач в области ИИ
    ),
      {"tag": "функционирует",
     "patterns": [["для чего функционирует центр?", "для чего функционирует ЦКП МПТ ИИ?", "для чего функционирует Центр коллективного пользования?", "для чего функционирует Центр
      "responses": ["Центр коллективного пользования Минпромторга России создано с целью повышения доступности аппаратного обеспечении, необходимого для решения задач в области И
    ),
    {"tag": "решения",
     "patterns": [["какие решения предлагает центр?", "какие решения предлагает ЦКП МПТ ИИ?", "какие решения предлагает Центр коллективного пользования?", "какие решения предлага
    "responses": ["Центр коллективного пользования Минпромторга России предоставляет решения по направлению создания модели «Искусственный интеллект, как сервис». ЦКП МПТ ИИ пред
    ),
     {"tag": "предложения",
     "patterns": [["что может предложить центр?", "какие решения предлагает ЦКП МПТ ИИ?", "что может предложить Центр коллективного пользования Минпромторга России?", "какие пред
     "responses": ["Центр коллективного пользования Минпромторга России предоставляет решения по направлению создания модели «Искусственный интеллект, как сервис». ЦКП МПТ ИИ пре
    ),
     {"tag": "услуги",
     "patterns": [["какие услуги предлагает центр?", "какие услуги предлагает ЦКП МПТ ИИ?", "какие услуги предлагает Центр коллективного пользования?", "какие услуги предлагает Ц
     "responses": ["Центр коллективного пользования Минпромторга России предоставляет решения по направлению создания модели «Искусственный интеллект, как сервис». ЦКП МПТ ИИ пре
    ),
     {"tag": "заказать",
     "patterns": ["хочу заказать услугу Центр коллективного пользования Минпромторга России", "хочу заказать решение Центра коллективного пользования Минпромторга России", "хочу
     "responses": ["Инструкция по получению доступа к решениям Центра коллективного пользования Минпромторга России размещена на странице https://aigov.ru/centre"]
    ),
     {"tag": "связаться",
     "patterns": ["как связаться с представителем Центра коллективного пользования Минпромторга России?", "хочу связаться с представителем Центра коллективного пользования Минпро
     "responses": ["Инструкция, как связаться с представителями Центра коллективного пользования Минпромторга России, размещена на странице https://aigov.ru/centre"]
    ),
     {"tag": "специализированное ПО и прикладные сервисы для решения задач на базе ИИ",
     "patterns": ["какое специализированное ПО предлагает Центр коллективного пользования Минпромторга России?", "какое специализированное программное обеспечение предлагает Цент
     "responses": ["Центр коллективного пользования Минпромторга России предоставляет специализированное ПО и и прикладные сервисы для решения задач на базе ИИ в составе: Платфор
    ),
     {"tag": "специализированные библиотеки",
     "patterns": ["какое специализированное ПО предлагает Центр коллективного пользования Минпромторга России?", "какое специализированное программное обеспечение предлагает Цент
     "responses": ["Центр коллективного пользования Минпромторга России предоставляет специализированные библиотеки в составе: Библиотека проверенных ML-моделей и Библиотека обез
    ),
     {"tag": "платформа взаимодействия с экспертами",
     "patterns": ["что такое платформа взаимодействия с экспертами?", "расскажи про платформу взаимодействия с экспертами?"],
      "responses": ["Центр коллективного пользования Минпромторга России предоставляет доступ к платформе взаимодействия с экспертами. Ознакомиться с описанием платформы и услови
    ),
```

Все текстовые примеры были собраны вручную в соответствии с методологией сбора оригинального датасета.

Все вопросы были написаны автором без каких-либо искусственных ограничений.

Ответы на вопросы были сформированы вручную на основании имеющей экспертизы. При этом в качестве дополнительного контроля были привлечены коллеги в роли асессоров, которые оценили качество ответов в ручном режиме.

Исходные данные были сформированы в формате .csv.

В последствии для обучения модели данные были переведены в формат .json.

Пример:

{

"tag": "специализированные библиотеки",

"patterns": ["какое специализированное ПО предлагает Центр коллективного пользования Минпромторга России?", "какое специализированное программное обеспечение предлагает Центр коллективного пользования Минпромторга России?", "какое специализированное ПО есть в Центре коллективного пользования Минпромторга России?", "какое специализированное программное обеспечение есть в Центре коллективного пользования Минпромторга России?", "какое специализированное ПО можно получить в Центре коллективного пользования Минпромторга России?", "какое специализированное программное обеспечение можно получить в Центре коллективного пользования Минпромторга России?","к какому специализированному ПО можно получить доступ в Центре коллективного пользования Минпромторга России?", "к какому специализированному программному обеспечению можно получить доступ в Центре коллективного пользования Минпромторга России?", "какое специализированное ПО предлагается в Центре коллективного пользования Минпромторга России?", "какое специализированное программное обеспечение предлагается в Центре коллективного пользования Минпромторга России?"],

"responses": ["Центр коллективного пользования Минпромторга России предоставляет специализированные библиотеки в составе: Библиотека проверенных ML-моделей и Библиотека обезличенных дата-сетов. Ознакомиться с описанием библиотек и условиями подключения можно на странице https://aigov.ru/centre"]

}

Подготовка программной среды

Для реализации решения использовались инструменты, работе с которыми автор научился в процессе обучения, а именно:

1. Jupiter notebook;
2. NotePad++;
3. Visual Studio Code;
4. Github.

Обработка данных

До старта обучения алгоритма была произведена дополнительная обработка данных:

1. Стемминг — нахождение основы слова для заданного исходного слова (при этом изначально планировалось использовать лемматизацию, которая ранее была протестирована в домашних работах с документами на английском языке, но в итоге на русском языке стемминг сработал качественнее, чем лемматизация).
2. Токенизация – разделение текста на токены (слова, предложения или другие более мелкие единицы), чтобы сделать текст более структурированным для последующей обработки
3. Очистка текста перед анализом: удаление стоп-слов и дополнительных лишних символов.
4. Преобразование текста: перевод в нижний регистр.
5. Попытка обработки естественного языка с помощью глубокого обучения (Deep Learning NLP) - Применение глубоких нейронных сетей для вопросно-ответной системы.

До настройка программной среды

В дополнение к этому была произведена до настройка программной среды и до обработка данных:

1. Добавление возможности работы с json файлами
2. Добавление возможности чтения запись моделей
3. Добавление возможности работы с массивами
4. Подключение дополнительной нейронной сети Sequential для работы с текстом
5. Подключение дополнительных слоев для нейронной сети: Dense, Activation, Dropout
6. Подключение оптимизатора для нейронной сети: SGD
7. Добавление возможности работы со строками
8. Добавление возможности загрузки модели для разбиения текста на слова: nltk.download('punkt'), nltk.download('averaged_perceptron_tagger' .

Обработка и очистка данных:

```
# подключение готовых библиотек для обработки и очистки данных
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.stem.snowball import SnowballStemmer
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.corpus import stopwords
import json
import pickle
import numpy as np
```

```
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from keras.optimizers import SGD
import random
import string

nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('stopwords')

lemmatizer = WordNetLemmatizer()
stemmer = SnowballStemmer("russian")
words=[]
classes = []
documents = []
ignore_words = ['?', '!', '«','»']
stop_words = set(stopwords.words('russian'))

data_file = open('/content/drive/MyDrive/datasets/questions.json').read()
intents = json.loads(data_file)

for intent in intents['intents']:
    for pattern in intent['patterns']:

        #tokenize each word
        w = nltk.word_tokenize(pattern)
        words.extend(w)
        #add documents in the corpus
        documents.append((w, intent['tag']))

        # add to our classes list
        if intent['tag'] not in classes:
            classes.append(intent['tag'])

words_lower = [word.lower() for word in words ]
# lemmatize, lower each word and remove duplicates
words = [stemmer.stem(w) for w in words_lower if w not in stop_words and w
not in string.punctuation and w not in ignore_words]
#stemmer.stem(w)
words = sorted(list(set(words)))
# sort classes
classes = sorted(list(set(classes)))
# documents = combination between patterns and intents
print (len(documents), "documents")
# classes = intents
print (len(classes), "classes", classes)
# words = all words, vocabulary
print (len(words), "unique stemmed words", words)

pickle.dump(words,open('words.pkl','wb'))
pickle.dump(classes,open('classes.pkl','wb'))
```

Подготовка данных для обучения – приведение данных в формат для нейронной сети:

```
# приведение данных в формат для нейронки
# create our training data (приведение данных в формат для нейронки)
training = []
# create an empty array for our output ()
output_empty = [0] * len(classes)
# training set, bag of words for each sentence
for doc in documents:
    # initialize our bag of words
    bag = []
    # list of tokenized words for the pattern
    pattern_words = doc[0]
    # lemmatize each word - create base word, in attempt to represent
related words
    pattern_words = [stemmer.stem(word.lower()) for word in pattern_words]
    # create our bag of words array with 1, if word match found in current
pattern
    for w in words:
        bag.append(1) if w in pattern_words else bag.append(0)

    # output is a '0' for each tag and '1' for current tag (for each
pattern)
    output_row = list(output_empty)
    output_row[classes.index(doc[1])] = 1

    training.append([bag, output_row])
# shuffle our features and turn into np.array (перемешивание слов в
случайном порядке)
random.shuffle(training)
training = np.array(training)
# create train and test lists. X - patterns, Y - intents
train_x = list(training[:,0])
train_y = list(training[:,1])
print("Training data created")
```

Отдельное внимание хочется обратить на применение функции «`random.shuffle`», благодаря которой не потребовалось для финального дата сета подбирать все возможные последовательности слов, как было сделано с первым набором данным.

Создание модели, создание слоев, связей между ними:

```
# Создание модели
# Create model - 3 layers. First layer 128 neurons, second layer 64
neurons and 3rd output layer contains number of neurons (создание модели -
3 слоя и связи между ними)
# equal to number of intents to predict output intent with softmax
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
```

```
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))

# Compile model. Stochastic gradient descent with Nesterov accelerated
gradient gives good results for this model (компиляция модели)
sgd = SGD(learning_rate=0.01, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd,
metrics=['accuracy'])

#fitting and saving the model #обучение модели (200 циклов по 5 штук)
hist = model.fit(np.array(train_x), np.array(train_y), epochs=200,
batch_size=5, verbose=1)
model.save('chatbot_model.keras', hist) #сохранение модели в файл для
альтернативного использования

print("model created")
```

Запуск работы нейронной сети и обученной модели:

```
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import pickle
import numpy as np

from keras.models import load_model
model = load_model('chatbot_model.keras')
import json
import random
intents =
json.loads(open('/content/drive/MyDrive/datasets/questions.json').read())
words = pickle.load(open('words.pkl','rb')) #подготовленные моделью слова
classes = pickle.load(open('classes.pkl','rb')) #подготовленные моделью
классы

def clean_up_sentence(sentence):
    # tokenize the pattern - split words into array (разбивает введенные
слова на массивы)
    sentence_words = nltk.word_tokenize(sentence)
    # stem each word - create short form for word (создает краткую
исходную форму введенного слова)
    sentence_words = [stemmer.stem(word.lower()) for word in
sentence_words]
    return sentence_words
# return bag of words array: 0 or 1 for each word in the bag that exists
in the sentence (присваевает слову значение 0 или 1)

def bow(sentence, words, show_details=True):
```

```
    # tokenize the pattern
    sentence_words = clean_up_sentence(sentence)
    # bag of words - matrix of N words, vocabulary matrix
    bag = [0]*len(words)
    for s in sentence_words:
        for i,w in enumerate(words):
            if w == s:
                # assign 1 if current word is in the vocabulary position
(значение 1 присваивается, если слово имеется в словаре)
                bag[i] = 1
                if show_details:
                    print ("found in bag: %s" % w)
    return(np.array(bag))

# извлечение ответа от нейронки
def predict_class(sentence, model):
    # filter out predictions below a threshold (фильтр прогнозов ниже
порогового значения)
    p = bow(sentence, words,show_details=False)
    res = model.predict(np.array([p]))[0]
    ERROR_THRESHOLD = 0.25
    results = [[i,r] for i,r in enumerate(res) if r>ERROR_THRESHOLD]
    # sort by strength of probability (сортировка по степени вероятности)
    results.sort(key=lambda x: x[1], reverse=True)
    return_list = []
    for r in results:
        return_list.append({"intent": classes[r[0]], "probability":
str(r[1])})
    return return_list

# подбор ответа нейронкой. random.choice применяется, если для введенного
текста возвожно несколько альтернативных ответов
def getResponse(ints, intents_json):
    tag = ints[0]['intent']
    list_of_intents = intents_json['intents']
    for i in list_of_intents:
        if(i['tag']== tag):
            result = random.choice(i['responses'])
            break
    return result

# печать ответа чат ботом
def chatbot_response(text):
    ints = predict_class(text, model)
    res = getResponse(ints, intents)
    return res

# Start chatbot (вечный цикл до  остановки)
while True:
    question = input('You: ')
    if question == 'quit':
```

```
        break
    answer = chatbot_response(question)
    print('Chatbot:', answer)
```

Результаты и выводы

Оценка реализованного алгоритма

Вывод о качестве модели можно сделать, проанализировав вывод из результатов её обучения:

При начальных циклах обучения, значение «**Loss**» (представляет собой сумму ошибок в модели) более 2.8 , а значение **Accuracy** (доля правильных ответов алгоритма) менее 0.03.

При этом к 100 циклу обучения, значения: loss: 0.2780 - accuracy: 0.8525.

А к 200 циклу обучения, значения: loss: 0.2072 - accuracy: 0.8770.

```
Epoch 1/200
25/25 [==============================] - 1s 2ms/step - loss: 2.8254 - accuracy: 0.0328
Epoch 2/200
25/25 [==============================] - 0s 2ms/step - loss: 2.5916 - accuracy: 0.1393
Epoch 3/200
25/25 [==============================] - 0s 2ms/step - loss: 2.4590 - accuracy: 0.1967
Epoch 4/200
25/25 [==============================] - 0s 2ms/step - loss: 2.2493 - accuracy: 0.2459
Epoch 5/200
25/25 [==============================] - 0s 2ms/step - loss: 2.1145 - accuracy: 0.3115
Epoch 6/200
25/25 [==============================] - 0s 2ms/step - loss: 2.0603 - accuracy: 0.3525
Epoch 7/200
25/25 [==============================] - 0s 2ms/step - loss: 1.7757 - accuracy: 0.4508
Epoch 8/200
25/25 [==============================] - 0s 3ms/step - loss: 1.6144 - accuracy: 0.5082
Epoch 9/200
25/25 [==============================] - 0s 2ms/step - loss: 1.5761 - accuracy: 0.5082
Epoch 10/200
25/25 [==============================] - 0s 2ms/step - loss: 1.3582 - accuracy: 0.5656
Epoch 11/200
25/25 [==============================] - 0s 2ms/step - loss: 1.2609 - accuracy: 0.5984
Epoch 12/200
25/25 [==============================] - 0s 2ms/step - loss: 1.1193 - accuracy: 0.6475
Epoch 13/200
25/25 [==============================] - 0s 2ms/step - loss: 0.9809 - accuracy: 0.6967
Epoch 14/200
25/25 [==============================] - 0s 2ms/step - loss: 1.1162 - accuracy: 0.6803
Epoch 15/200
25/25 [==============================] - 0s 2ms/step - loss: 0.9122 - accuracy: 0.6475
```

```
25/25 [==============================] - 0s 3ms/step - loss: 0.1979 - accuracy: 0.8770
Epoch 185/200
25/25 [==============================] - 0s 3ms/step - loss: 0.2063 - accuracy: 0.8689
Epoch 186/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1869 - accuracy: 0.8607
Epoch 187/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1669 - accuracy: 0.9016
Epoch 188/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1920 - accuracy: 0.8852
Epoch 189/200
25/25 [==============================] - 0s 3ms/step - loss: 0.2001 - accuracy: 0.8934
Epoch 190/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1830 - accuracy: 0.8934
Epoch 191/200
25/25 [==============================] - 0s 3ms/step - loss: 0.2438 - accuracy: 0.8852
Epoch 192/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1932 - accuracy: 0.8689
Epoch 193/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1587 - accuracy: 0.9016
Epoch 194/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1762 - accuracy: 0.8770
Epoch 195/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1738 - accuracy: 0.8607
Epoch 196/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1947 - accuracy: 0.8443
Epoch 197/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1780 - accuracy: 0.8689
Epoch 198/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1384 - accuracy: 0.9344
Epoch 199/200
25/25 [==============================] - 0s 3ms/step - loss: 0.1924 - accuracy: 0.8852
Epoch 200/200
25/25 [==============================] - 0s 3ms/step - loss: 0.2072 - accuracy: 0.8770
model created
```

Визуализация результатов работы

Результаты работа алгоритма визуализированы в формате чат-бота, доступного в веб-интерфейсе.

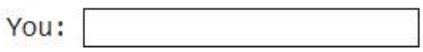
Пред настройка и обучение модели в коде занимает 5 шагов.

Визуализация чата в коде занимает 1 шаг.

Дополнительно под чат-ботом опубликованы 2 текстовых окна:

1. Первый – с информацией о том, как остановить работу чат-бота.
2. Второй – с информацией о том, какой текст лег в основу обучения чатбота.

Интерфейса чат бота на стартовом экране представляет собой диалоговое окно, куда пользователь может написать текст:



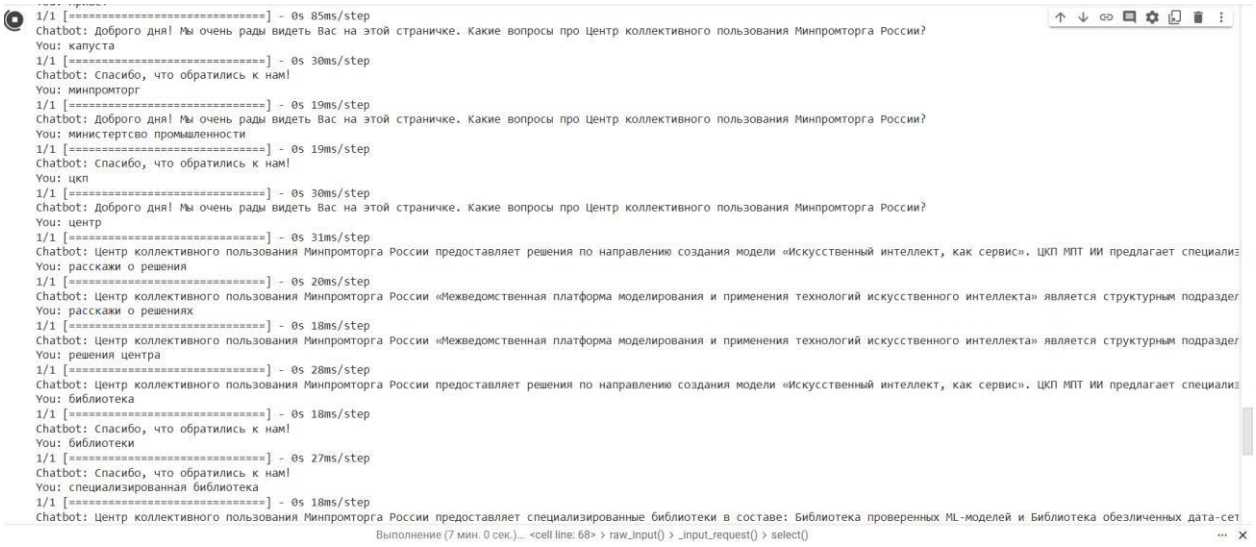
При введение пользователем запроса, который имеется в модели, чат-бот продолжает диалог, провоцируя пользователя на последующие вопросы:

```
You: Привет
1/1 [==============================] - 0s 85ms/step
Chatbot: Доброго дня! Мы очень рады видеть Вас на этой страничке. Какие вопросы про Центр коллективного пользования Минпромторга России?
You:
```

В случае, если пользователь ввел запрос, ответ на который отсутствует в модели, чат-бот благодарит пользователя за проявленный интерес:

```
You: Привет
1/1 [==============================] - 0s 85ms/step
Chatbot: Доброго дня! Мы очень рады видеть Вас на этой страничке. Какие вопросы про Центр коллективного пользования Минпромторга России?
You: капуста
1/1 [==============================] - 0s 30ms/step
Chatbot: Спасибо, что обратились к нам!
You:
```

Помимо диалога, система также подсвечивает пользователю скорость выдачи ответа нейронной сетью.



Сохранение данных для использования

Все данные сохранены и доступны для использования:

https://github.com/MukhinaMaria/FinalAttest

Обученная модель сохранена и доступна для использования:

https://github.com/MukhinaMaria/CB_CKP/blob/main/chatbot_model.keras

Набор данных сохранен и доступен для использования:

https://github.com/MukhinaMaria/CB_CKP/blob/main/questions.json

Инструкция работы с приложением также доступна по ссылке:

https://github.com/MukhinaMaria/CB_CKP/blob/main/Инструкция%20работы%20с%20приложением.docx

Планируемое развитие проекта

Уровень автора работы объективно далек от уровня профессиональных разработчиков нейронных сетей.

Однако данный проект является крайне важен для промышленной реализации интеллектуального помощника взаимодействия с обществом для Федерального центра прикладного развития искусственного интеллекта Минпромторга России, так как автор работы освоил необходимые навыки для качественной постановки задачи и качественной оценки результатов работ.

При это подготовленный набор данных ляжет в основу промышленной системы и будет существенно доработан.