



Министерство науки и высшего образования Российской  
Федерации  
Калужский филиал федерального государственного автономного  
образовательного учреждения высшего образования  
«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(КФ МГТУ им. Н.Э. Баумана)

---

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК5 «Системы обработки информации»

## ЛАБОРАТОРНАЯ РАБОТА №2.2

### «МОДЕЛИ ТРАНСФОРМЕРОВ. ЯЗЫКОВАЯ МОДЕЛЬ GPT И ЕЕ РЕАЛИЗАЦИЯ НА ЯЗЫКЕ PYTHON. БИБЛИОТЕКА OpenAI»

по дисциплине: «Методы глубокого обучения»

Выполнил: студент группы ИУК5-21М

(Подпись)

А. Э. Дармограй

(И.О. Фамилия)

Проверил:

(Подпись)

Ю. С. Белов

(И.О. Фамилия)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Калуга, 2025

**Цель:** получение практических навыков работы с языковой моделью GPT.

**Задачи:**

- Ознакомление с методами работы модели GPT (ответы на вопросы, генерация статей, синтетических данных, программного кода);
- Реализация программ, использующих языковую модель GPT.

**Результатами работы являются:**

- Программа, использующая языковую модель GPT;
- Подготовленный отчет.

**Вариант 2.**

**Задание:**

Организуите ИИ для распознавания речи и ответов на вопросы по тематике: Нейронные сети.

## Выполнение работы

Код доступен в репозитории GitHub:

[https://github.com/Dariarty/Deep\\_Learning\\_Methods](https://github.com/Dariarty/Deep_Learning_Methods)

Данную лабораторную работы выполнял на Python версии 3.9.13

Код лабораторной работы №2.2:

[https://github.com/Dariarty/Deep\\_Learning\\_Methods/blob/main/src/LAB\\_2\\_2/](https://github.com/Dariarty/Deep_Learning_Methods/blob/main/src/LAB_2_2/)

В данной работе разработал приложение – голосовой чат-бот для общения с Chat GPT на тематику «нейронные сети».

### 1. Создание ключа OpenAI API

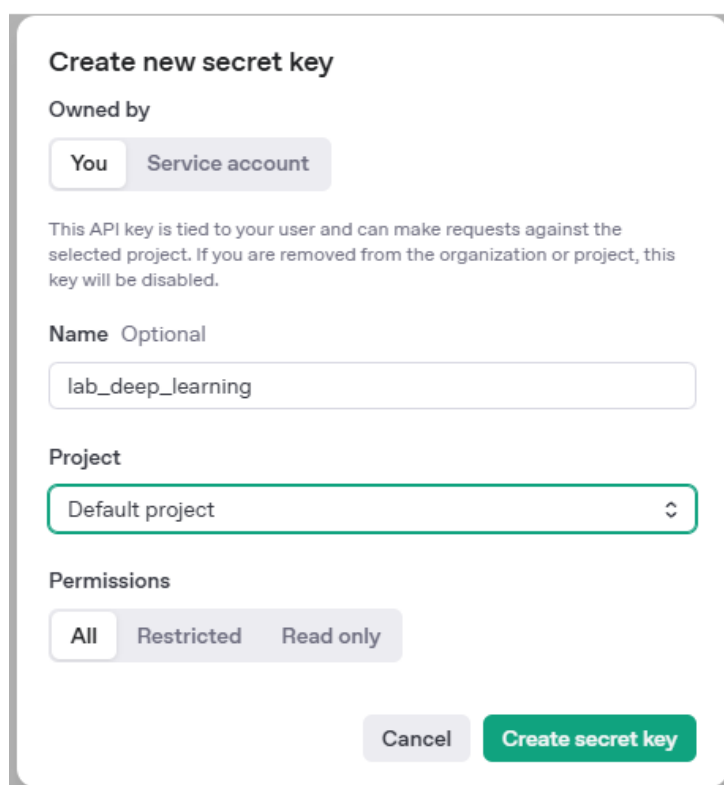
The image shows a screenshot of the 'Create new secret key' dialog box in the OpenAI interface. The dialog has a title 'Create new secret key' and a section 'Owned by' with two buttons: 'You' (selected) and 'Service account'. Below this is a warning: 'This API key is tied to your user and can make requests against the selected project. If you are removed from the organization or project, this key will be disabled.' There is a 'Name' field with the value 'lab\_deep\_learning' and a note 'Optional'. The 'Project' is set to 'Default project' in a dropdown menu. Under 'Permissions', there are three buttons: 'All' (selected), 'Restricted', and 'Read only'. At the bottom right are 'Cancel' and 'Create secret key' buttons.

Рис. 1. Создание нового ключа

Для обращения к API OpenAI необходим ключ, сгенерировал новый ключ. Считывать его буду из текстового файла.

## 2. Реализация распознавания голоса

В ходе выполнения работы возникли трудности к обращению по API к AssemblyAI. С 2024 года сервис требует обязательной привязки международной карты даже для бесплатного лимита запросов по API.

Было принято решение использовать Whisper от OpenAI. Whisper — это открытая нейросетевая модель от OpenAI, предназначенная для автоматического преобразования аудио в текст (ASR — automatic speech recognition).

К сожалению, модель Whisper от OpenAI предназначена только для пакетной обработки аудиофайлов, и на данный момент не поддерживает потоковое распознавание речи. Модель принимает только завершённые аудиофайлы, в своей программе я буду отправлять файлы .wav.

В рамках лабораторной работы была реализована логика:

- Записи аудиофайла до окончания речи (определяется по длительности тишины в течение нескольких секунд)
- После завершения записи — передача целого файла в модель Whisper для расшифровки.

После ответа от Whisper, расшифрованный текст отправляется в качестве запроса модели Chat GPT.

Кроме этого, в рамках работы для гибкости была реализована возможность как использовать локальную (заранее загруженную) модель Whisper medium, так и удаленно через API OpenAI.

При запуске программы у пользователя есть выбор — работать с удаленной моделью через API или локальной.

### 3. Код программы с пояснениями

#### Файл `openai_helper.py`

Модуль для обращения к API OpenAI:

- `ask_computer(prompt)` — отправляет текстовый запрос в модель `gpt-3.5-turbo` через API OpenAI.

В запрос встроено системное сообщение, ограничивающее тему на область нейронных сетей и глубокого обучения.

- `transcribe_audio_remote(file)` — отправляет аудиофайл на сервер OpenAI для расшифровки с помощью модели `whisper`.
- чтение API-ключа OpenAI из файла `openai_key.txt` (ключ не хранится в коде),
- базовая защита от отсутствия файла с ключом.

```
import openai
import os
from pathlib import Path

# Путь к файлу относительно местоположения скрипта
KEY_FILE = "openai_key.txt"

BASE_DIR = os.path.dirname(os.path.abspath(__file__))
KEY_PATH = os.path.join(BASE_DIR, KEY_FILE)

# Чтение ключа
try:
    with open(KEY_PATH, "r") as key_file:
        openai.api_key = key_file.read().strip()
except FileNotFoundError:
    raise RuntimeError(f"Файл {KEY_PATH} не найден. Убедитесь, что он существует и содержит API-ключ OpenAI.")

def ask_computer(prompt):
    response = openai.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "system", "content": "Ты — эксперт по нейронным сетям. Отвечай только по теме нейронных сетей, глубокого обучения и машинного обучения. Если вопрос не по теме, вежливо сообщи, что он выходит за рамки твоей компетенции и предложи пообщаться на тему глубокого обучения."},
            {"role": "user", "content": prompt}
        ],
        temperature=0.7,
        max_tokens=1000
    )
```

```
        return response.choices[0].message.content.strip()

def transcribe_audio_remote(recorded_voice_file):
    with open(recorded_voice_file, "rb") as audio_file:
        response = openai.audio.transcriptions.create(
            model="whisper-1",
            file=audio_file,
            response_format="json"
        )
    return response.text
```

System - роль сообщает модели, что она должна отвечать только по теме нейронных сетей и вежливо отказываться от других тем.

В данной работе буду использовать модель gpt-3.5-turbo.

### Файл main.py

Основной управляющий скрипт, реализующий:

- record\_until\_silence() — захват аудио с микрофона до окончания фразы (определяется по звуку ниже уровня тишины в течение нескольких секунд),
- transcribe\_audio\_local() — локальное распознавание с помощью модели whisper-medium;
- выбор режима работы пользователем при старте (локальный Whisper или удалённый через OpenAI API)
- отправку распознанного текста в GPT
- получение и отображение ответа.
- диалоговый цикл с обработкой KeyboardInterrupt. После получения ответа можно отправить новый запрос. В любой момент можно завершить выполнение, нажав ctrl+c

```

import wave
import pyaudio
import os
import audioop
import whisper

import warnings
warnings.filterwarnings("ignore")

from openai_helper import ask_computer
from openai_helper import transcribe_audio_remote

# Параметры
SCRIPT_DIR = os.path.dirname(os.path.abspath(__file__))
FILENAME = os.path.join(SCRIPT_DIR, "recorded.wav")
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 16000
FRAMES_PER_BUFFER = 1024
SILENCE_THRESHOLD = 1000
MAX_SILENCE_SECONDS = 8 # Сколько секунд тишины в конце запроса

MAX_SILENCE_BUFFERS = int(RATE / FRAMES_PER_BUFFER * MAX_SILENCE_SECONDS)

# Запись аудио с автоостановкой по тишине
def record_until_silence():
    audio = pyaudio.PyAudio()
    stream = audio.open(format=FORMAT, channels=CHANNELS, rate=RATE, input=True,
frames_per_buffer=FRAMES_PER_BUFFER)

    print("Говорите. Запись остановится после паузы...")
    frames = []
    silence_count = 0

    while True:
        data = stream.read(FRAMES_PER_BUFFER, exception_on_overflow=False)
        frames.append(data)
        sound_volume = audioop.rms(data, 2) # Оценка громкости

        #Если громкость меньше порога, длительность тишины увеличивается
        if sound_volume < SILENCE_THRESHOLD:
            silence_count += 1
        else:
            silence_count = 0

        if silence_count > MAX_SILENCE_BUFFERS:
            break

    #Окончание записи запроса
    stream.stop_stream()
    stream.close()
    audio.terminate()

    with wave.open(FILENAME, 'wb') as wf:
        wf.setnchannels(CHANNELS)
        wf.setsampwidth(audio.get_sample_size(FORMAT))
        wf.setframerate(RATE)
        wf.writeframes(b''.join(frames))

```

```

    print("Запись завершена.")

#Распознавание речи через локальный Whisper medium модели
def transcribe_audio_local(recorded_voice_file, whisper_model):
    result = whisper_model.transcribe(recorded_voice_file)
    return result["text"]

if __name__ == "__main__":
    model = None
    use_remote = input("Использовать удалённую версию Whisper через OpenAI API?
y/n: ").strip().lower() == "y"
    if use_remote:
        print("Используется удаленная версия Whisper")
    else:
        print("Загрузка локальной модели Whisper medium...")
        model = whisper.load_model("medium")

    try:
        while True:
            #Запись до окончания речи
            print("Запись голосового запроса... (Ctrl+C для выхода)")
            record_until_silence()

            #Расшифровка сообщения
            if use_remote:
                text = transcribe_audio_remote(FILENAME)
            else:
                text = transcribe_audio_local(FILENAME, model)
            print("Вы сказали:", text)

            #Ответ от Chat GPT
            response = ask_computer(text)
            print("Ответ:")
            print(response)

            #Повторный запрос
            print("Нажмите Enter, чтобы спросить еще раз... (Ctrl+C для выхода)")
            input()
    except KeyboardInterrupt:
        print("Завершение работы.")

```



#### 4. Зависимости и запуск приложения

Зависимости проекта:

1. Установка необходимых библиотек

```
pip install torch numpy pyaudio openai os pathlib warnings
```

torch необходим для работы whisper

2. Установка whisper.

```
pip install git+https://github.com/openai/whisper.git
```

3. Также для работы whisper требуется установка внешней зависимости - ffmpeg.

На ОС семейства Windows:

```
winget install ffmpeg
```

На ОС семейства Linux:

```
sudo apt install ffmpeg
```

4. Необходим файл с ключом API OpenAI в директории проекта openai\_key.txt

5. Подключенный к устройству микрофон. Запись ведется с системного устройства по умолчанию.

#### 5. Работа приложения

Пример работы через API:

```
(.venv) C:\Work\repo\Deep_Learning_Methods>
c:/Work/repo/Deep_Learning_Methods/.venv_python_3.9.13_whisper/Scripts/python.exe c:/Work/repo/Deep_Learning_Methods/src/LAB_2_2/main.py
Использовать удалённую версию Whisper через OpenAI API? y/n: y
Используется удаленная версия Whisper
Запись голосового запроса... (Ctrl+C для выхода)
```

Говорите. Запись остановится после паузы...

Запись завершена.

Вы сказали: Что такое сверточная нейронная сеть?

Ответ:

Сверточная нейронная сеть (Convolutional Neural Network, CNN) – это тип нейронной сети, который обычно используется для анализа визуальных данных, таких как изображения. Основная особенность сверточных нейронных сетей – использование сверточных слоев для извлечения признаков из входных данных. Эти слои позволяют сети автоматически изучать различные характеристики изображений, такие как углы, текстуры, цвета и т.д.

Сверточные нейронные сети обычно состоят из нескольких слоев: сверточные слои, слои подвыборки (pooling), полносвязанные слои и слои активации. Они широко применяются в задачах компьютерного зрения, таких как классификация изображений, обнаружение объектов, сегментация изображений и других.

Если у вас есть дополнительные вопросы о сверточных нейронных сетях или машинном обучении в целом, не стесняйтесь спрашивать.

Нажмите Enter, чтобы спросить еще раз... (Ctrl+C для выхода)

Запись голосового запроса... (Ctrl+C для выхода)

Говорите. Запись остановится после паузы...

Запись завершена.

Вы сказали: Для чего нужен градиентный спуск.

Ответ:

Градиентный спуск используется в машинном обучении для минимизации функции потерь, что позволяет оптимизировать параметры модели. Он помогает находить локальный минимум функции путем итеративного обновления параметров модели в направлении, противоположном градиенту функции потерь. Таким образом, градиентный спуск позволяет модели учиться на данных и постепенно улучшать свои предсказательные способности.

Нажмите Enter, чтобы спросить еще раз... (Ctrl+C для выхода)

Завершение работы.

### Пример работы с локальной версией Whisper medium:

```
(.venv) C:\Work\repo\Deep_Learning_Methods>
c:/Work/repo/Deep_Learning_Methods/.venv_python_3.9.13_whisper/Scripts/p
ython.exe c:/Work/repo/Deep_Learning_Methods/src/LAB_2_2/main.py
Использовать удалённую версию Whisper через OpenAI API? y/n: n
Загрузка локальной модели Whisper medium...
Запись голосового запроса... (Ctrl+C для выхода)
Говорите. Запись остановится после паузы...
Запись завершена.
Вы сказали: Чем отличается глубокое обучение от машинного обучения?
Ответ:
Глубокое обучение является подмножеством машинного обучения, которое
использует искусственные нейронные сети для извлечения сложных
закономерностей из данных. Основное отличие глубокого обучения от
классических методов машинного обучения заключается в том, что глубокие
нейронные сети способны автоматически извлекать признаки из данных на
разных уровнях абстракции, что позволяет им эффективно решать сложные
задачи, такие как распознавание образов, обработка естественного языка, и
многое другое.
Нажмите Enter, чтобы спросить еще раз... (Ctrl+C для выхода)
Завершение работы.
```

### Пример вопроса не по теме:

```
(.venv)
C:\Work\repo\Deep_Learning_Methods>c:/Work/repo/Deep_Learning_Meth
ods/.venv_python_3.9.13_whisper/Scripts/python.exe
c:/Work/repo/Deep_Learning_Methods/src/LAB_2_2/main.py
Использовать удалённую версию Whisper через OpenAI API? y/n: y
Используется удаленная версия Whisper
Запись голосового запроса... (Ctrl+C для выхода)
Говорите. Запись остановится после паузы...
Запись завершена.
Вы сказали: Сколько планет в Солнечной системе?
```

Ответ:

Извините, но ваш вопрос не относится к моей области знаний. Я могу помочь вам с вопросами о нейронных сетях, глубоком обучении и машинном обучении. Если у вас есть интересные вопросы на эти темы, не стесняйтесь спрашивать!

Нажмите Enter, чтобы спросить еще раз... (Ctrl+C для выхода)

Завершение работы.

**Вывод:** в ходе выполнения лабораторной работы были сформированы практические навыки работы с языковой моделью GPT. Были изучены методы работы модели GPT, а также разработана программа, использующая языковую модель GPT, а также модель Whisper для обработки голосовых запросов.