

# ИИ


- [Управление языковыми моделями](#)
- [Взаимодействие через python](#)
- [Книги](#)
- [Теория LLM](#)

# Управление языковыми моделями

## Технические требования

Настройка НА кластера - критичный раздел, но сейчас пока не актуально.

Мне хватило следующего ПК:

Процессор	Intel Xeon E5-2670 v3 @2.3GHz (даже не средний  ) Во время обработки грузился на 60%.
ОП	Всего 32 Gb На обученной модели во время обработки вопроса в пиках подскакивало только до 17 Gb Просто ollama в фоне - 11 Gb
SSD	Для размещения модели deepseek-r1:7b потребовалось 5 Gb
Видео	Не использовалось, слишком старая. Да, не особо быстро, иногда полного ответа нужно было ждать секунд 30.
ОС	Windows

Для построения векторного индекса по одному файлу word размером 100 страниц потребовалось 35 минут.

## Запуск модели

Использовал менеджер моделей Ollama [ollama.com](https://ollama.com) Установщик. Затем управление через cmd.

Команда	Описание
ollama run model_name	Скачать, установить и запустить модель <div><pre>ollama run deepseek-r1:7b</pre></div>
ollama list	Список установленных моделей

Команда	Описание
ollama rm model_name	Удаление модели

После запуска по умолчанию <http://localhost:11434/> запускается API.

# Взаимодействие через python

## Запрос - ответ в существующую модель

```
import ollama
import requests

def chat_with_deepseek(prompt, model="deepseek-r1:7b"):
    response = ollama.chat(
        model=model,
        messages=[{"role": "user", "content": prompt}]
    )
    return response["message"]["content"]

def chat_with_deepseek_api(prompt, model="deepseek-r1:7b"):
    url = "http://localhost:11434/api/chat"
    data = {
        "model": model,
        "messages": [{"role": "user", "content": prompt}],
        "stream": False # Отключить потоковый вывод
    }
    response_api = requests.post(url, json=data)
    return response_api.json()["message"]["content"]

# Пример использования
user_input = "Объясни, что такое ООП простыми словами."
#response = chat_with_deepseek(user_input)
response = chat_with_deepseek_api(user_input)
print("Ответ DeepSeek:", response)
```

[Источник RAG](#)

## Дообучение модели на собственных данных

Дополнительные пакеты

```
python -m pip install ollama llama-index transformers torch sentence-transformers llama-index-llms-ollama
```

Здесь должен был быть код, успешный результат (хотя бы результатик), но... Все уперлось в токенизацию. Напрямую 100 страничный файл оказался бессмысленным, узлов после 30 минутной обработки было создано 0. Причем различные варианты не помогли - простое предоставление файла в свободном форматировании оказалось бессмысленным занятием. Нужно погружаться как минимум в теорию Chunk'ов. Хотя скорее всего потребуется еще много чего.

# КНИГИ

# Теория LLM

Генеративный ИИ относится к алгоритмам, которые могут генерировать новый контент, в отличие от анализа существующих данных или воздействия на них, как более традиционные системы машинного обучения с прогнозированием или искусственного интеллекта.