Лабораторная работа 4. HF tune Transformers

Машинное обучение, часть 2, весна 2025

Название проекта: Генератор демотиваторов

Состав команды:

- Пенкина Дарья
- Коломникова Дарья
- Горобец Иван
- Пушкарев Дмитрий

Описание проекта:

Система генерирует демотиваторы на основе пользовательских промптов (начал анекдотов) по следующему конвейеру:

1. **Генерация анекдотов**: Модель Qwen на основе промпта создает несколько вариантов завершения анекдота.
2. **Выбор лучшего**: Модель Bert ранжирует варианты и выбирает наиболее удачный.
3. **Генерация изображения**: Модель Kandinsky или Stable Diffusion создает иллюстрацию к анекдоту.
4. **Сборка демотиватора**: Изображение и текст объединяются в формате демотиватора с белой рамкой.

Целевая метрика успеха: Пользователь хихикает.

Ссылка на гитхаб:

https://github.com/T3FiO/funny_jokes
Ссылка на сайт в разделе readme.

Работа с данными

В ходе исследования выявилось, что для обучения моделей не существует удобных русскоязычных датасетов формата *«начало шутки*

— *конец шутки»*, которые позволили бы эффективно обучать модели генерации юмористического контента.

Хотя аналогичные англоязычные датасеты доступны, их прямое использование для русскоязычных задач **нецелесообразно** из-за различий в языковых паттернах и культурном контексте.

Теоретически, для структурирования данных можно было бы применить LLM (например, GPT), однако это потребовало бы:

- Дополнительных вычислительных ресурсов
- Сложной постобработки результатов
- Решения нетривиальной задачи выделения структурных элементов шуток

Сбор собственного датасета

В связи с отсутствием готовых решений был реализован парсинг сайта ***анекдоты.ру***, в результате чего получен датасет, содержащий:

- 230 000 шуток
- Их рейтинги

Как работает парсер:

1. Загрузка данных:
    - Использована библиотека `requests` для получения HTML-кода страниц
2. Парсинг:
    - Применена библиотека `BeautifulSoup` для анализа структуры страницы
3. Извлечение данных:

    Пример извлекаемого элемента с сайта (текст шутки -> Joke, рейтинг шутки -> 694):

```
<div class="topicbox" id="10" data-id="1516482" data-t="j" "="">
        <div class="text">Joke</div>
        <div class="rates" data-id="1516482" data-r="694;840;767;73"></div>
</div>
```

Подробная реализация доступна в файле parser.py.

Предобработка данных

Были проведены два типа предобработки:

1. Приведение к нормальному распределению
    - Устранение выбросов
    - Оптимизация для алгоритмов, чувствительных к распределению данных
2. Стандартизация (среднее = 0, дисперсия = 1)
    - Унификация масштаба признаков
    - Повышение стабильности обучения

Результат:
Лучшие показатели достигнуты при использовании стандартизированных данных.

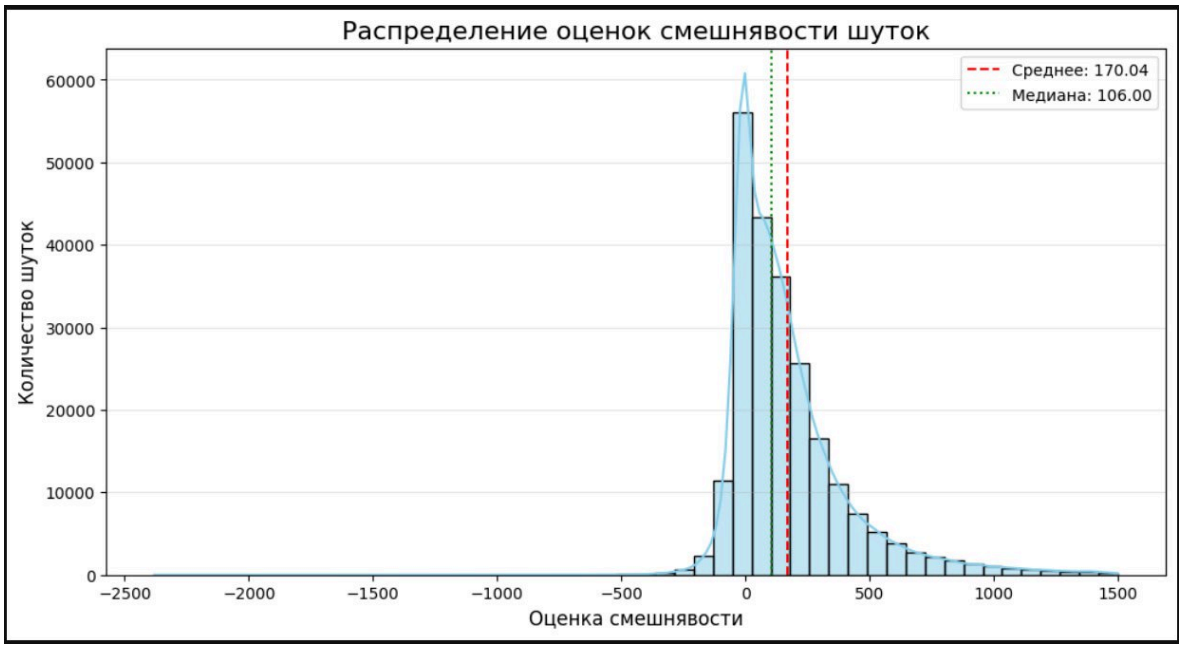


Рисунок 1. Изначальное распределение данных

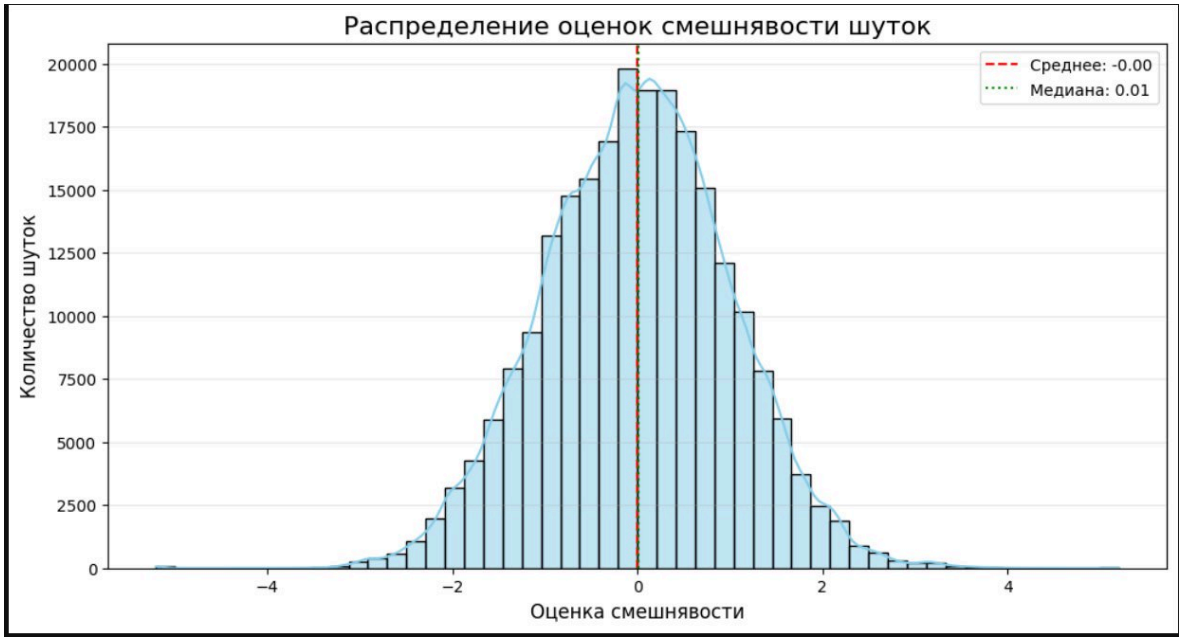


Рисунок 2. Приведенные к нормальному распределению данные

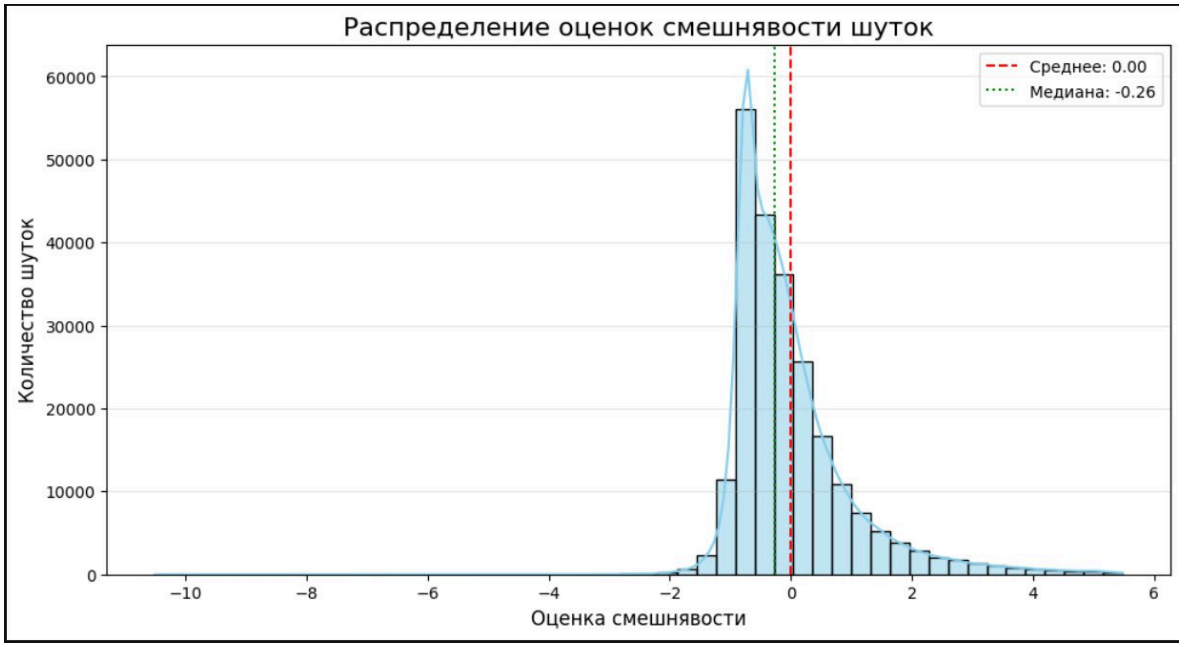


Рисунок 3. Приведенные к нулевому среднему и единичной дисперсии данные

Эксперименты

Хохотали всей маршруткой над генерируемым.

Технические характеристики решения

Для генерации анекдота используется модель Qwen. В процессе работы были дообучены две модели - на датасете с HF и на нашем датасете. Качество шуток в обоих случаях сопоставимы, поэтому для инференса остановились на модели, дообученной на датасете с анекдоты.ру.

Первоначально для оценки качества шуток обучали модель RuBert, получили train loss = 0.9461, test loss = 0.9092, веса модели имели размер 600 MB. Из-за большого размера весов попробовали модель, уменьшенную с помощью дистилляции, RuDistillBert. На ней получили train loss = 0.9699, test loss = 0.9565, при этом веса уменьшились в 3 раза. В качестве loss функции брался MSE.

	MSE	MAE	R^2
RuBert	0.9092	0.6835	0.0000
RuDistillBert	0.9699	0.6953	0.0381

RuBert плохо объясняет данные, это происходит из-за недообученности и зашумленных данных. RuDistillBert в данном случае объясняет данные лучше. Данные могут считаться зашумленными из-за того, что:

- один и тот же анекдот может казаться смешным одним людям и несмешным другим,
- вкусовые предпочтения сильно различаются в зависимости от возраста, культуры и чувства юмора,
- возможны артефакты (Пример: “Перестаньте учить своих детей, что война – это слава и героизм. Научите их, что настоящая слава – в предотвращении войн, и что герои – это те, кто может это сделать.” - это даже не анекдот, но имеет высокий рейтинг),
- анекдот популярный 10 лет назад, сейчас может казаться не смешным.

Система работает по следующему принципу:

1. На основе заданного промпта генерируется N вариантов продолжения,
2. Из полученных вариантов с помощью модели RuDistillBERT выбирается наиболее удачный.

Хотя технически возможно выводить все сгенерированные варианты с рейтингами, на практике это часто приводит к появлению странных или неуместных шуток, поэтому используется отбор лучшего варианта.

Стоит отметить, что модель Qwen демонстрирует значительное время генерации - от 5 секунд до 2 минут при работе на CPU.

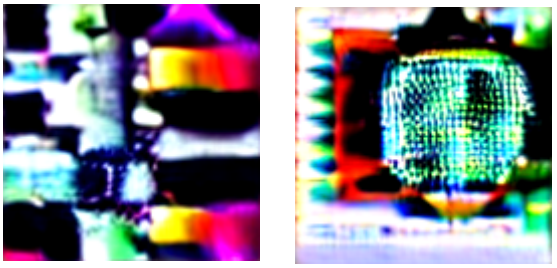
Для модели генерации изображений были протестированы две нейронные сети для создания иллюстраций:

- Kandinsky,
- Stable diffusion (SD).

Обе модели продемонстрировали склонность к созданию абстрактных изображений. Однако SD показала следующие преимущества:

- более интересные результаты,
- меньшее ресурсопотребление,
- выше скорость работы.

В итоге был выбран SD в качестве основного генератора изображений. Для улучшения визуального качества изображение масштабируется в 3 раза (с исходного разрешения 128×128 пикселей) и добавляется белая рамка толщиной 2 пикселя.



Картинки, сгенерированные Stable Diffusion

Картинки, сгенерированные Kandinsky

Для реализации веб-интерфейса системы был выбран FastAPI -Python-фреймворк, обеспечивающий:

- простоту разработки API,
- поддержку асинхронных запросов.

Дизайн платформы в стиле минимализма и включает только необходимые элементы:

1. Основное поле ввода
    - расположено по центру страницы,
    - подсказка предлагает ввести начало анекдота.
2. Кнопка "Создать!",
3. Блок с результатом
    - Появляется после завершения генерации
    - Включает сгенерированное изображение с характерной белой рамкой и текст анекдота, расположенный под картинкой.

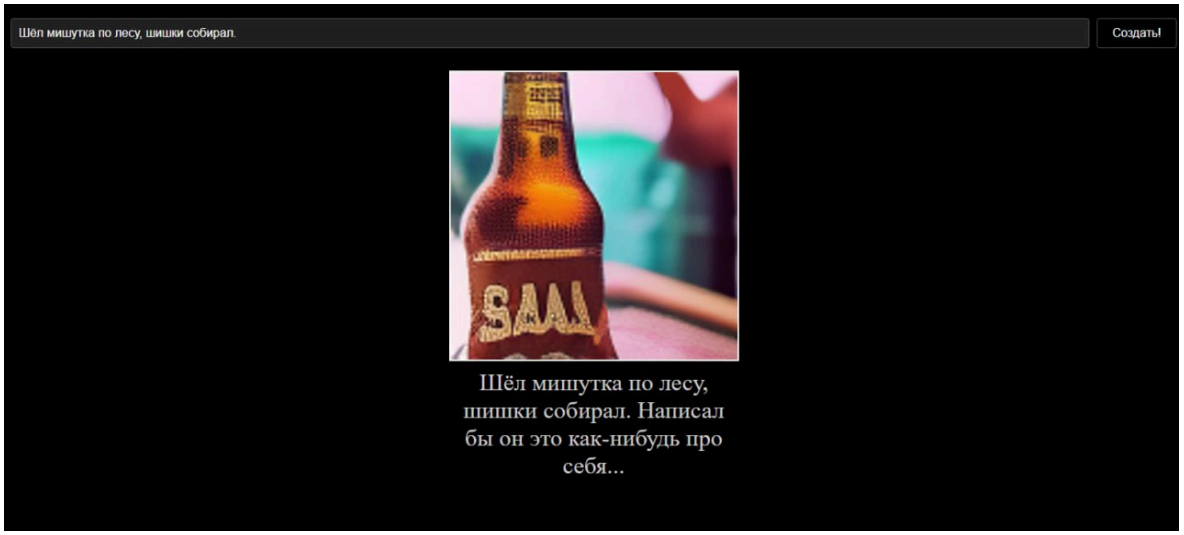


Рисунок 4. Пример интерфейса с корректно сгенерированной картинкой Kandinsky

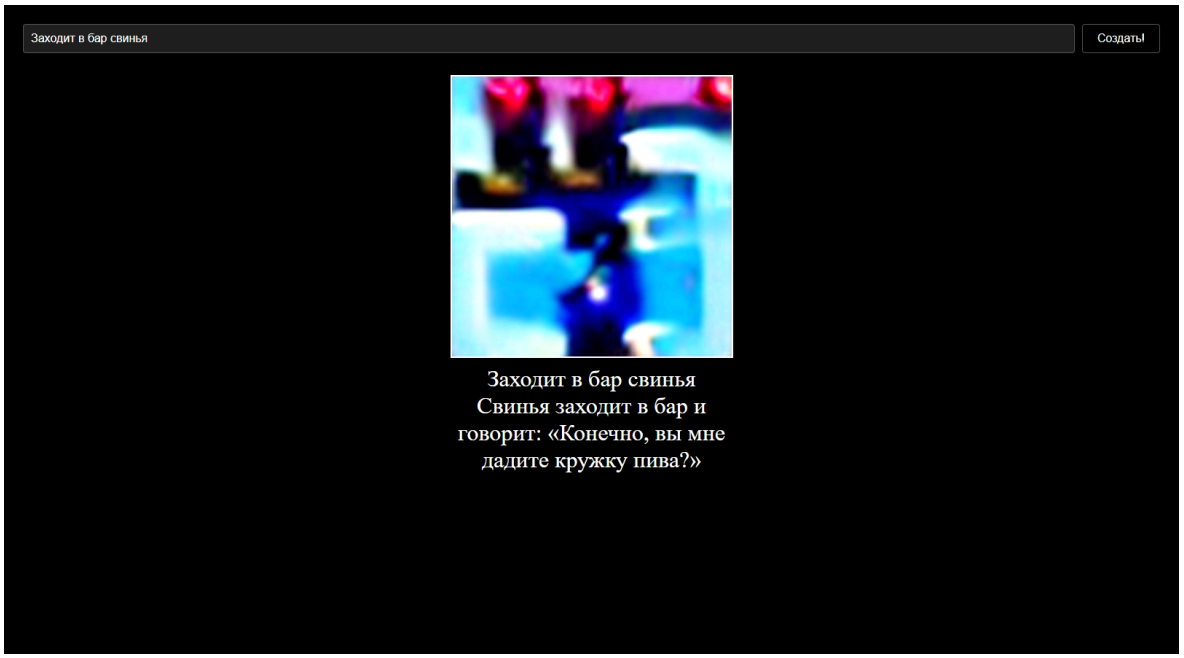


Рисунок 5. Пример интерфейса с корректно сгенерированной картинкой Stable Diffusion

Рисунок 6. Пример интерфейса, где Stable Diffusion посчитал анекдот nsfw

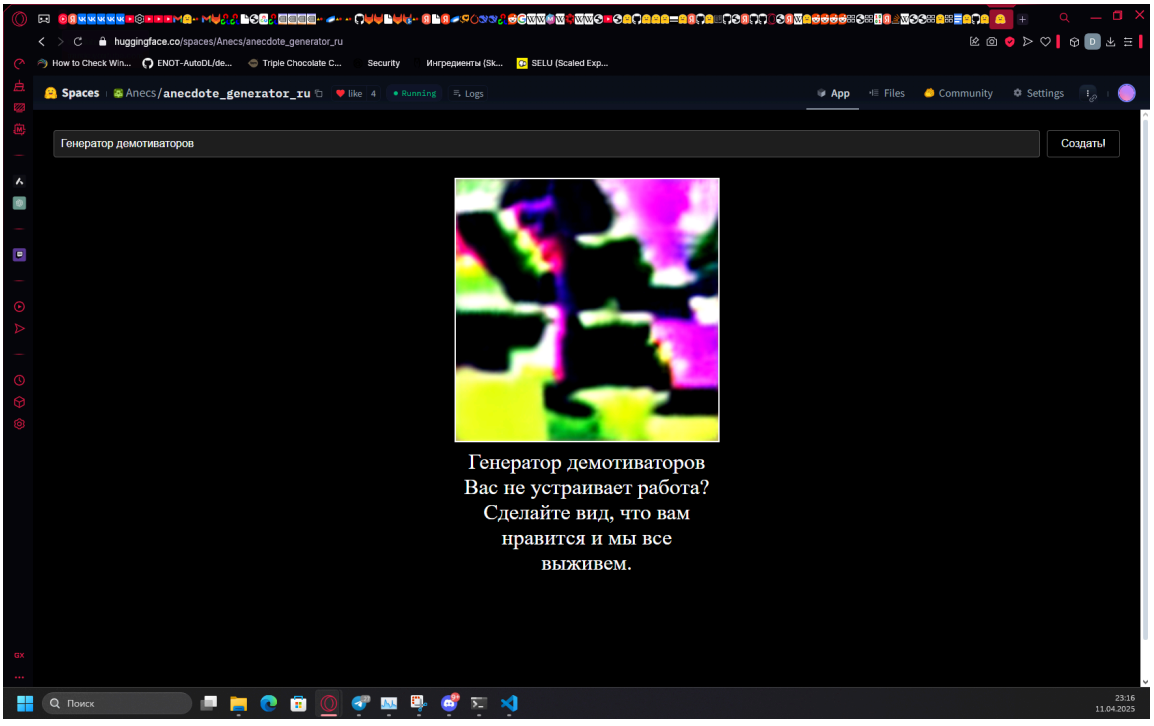


Рисунок 7. Финальный вариант на хостинге

Примечание

Генерация анекдота занимает около 3-х минут, картинки - около 5, подождите, пожалуйста, это всё работает на одном потоке бесплатного CPU :_)