Санкт-Петербургский государственный университет
Кафедра системного программирования

Разработка системы для универсальной классификации текстов

Михаил Владимирович Выродов

Научный руководитель: Ю.В. Литвинов, старший преподаватель каф. СП

Консультант: Е.С. Спирин, старший программист-разработчик ООО “ВК”

Рецензент: А.Д. Земеров, разработчик АО “Точка”

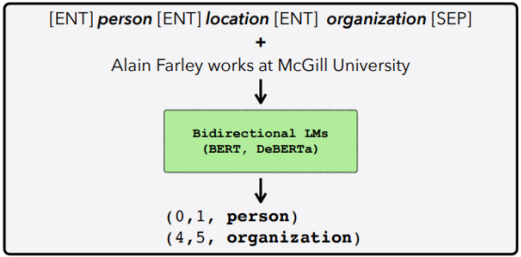
Санкт-Петербург
2025

Введение

- Социальные сети стали неотъемлемой частью нашей жизни
- Возникает широкий спектр задач классификации:
  - модерация контента
  - аналитика трендов
  - категоризация постов по темам
  - ...
- Необходимо уметь быстро получать прототипы классификации, чтобы эффективно проверять гипотезы о данных

Введение

- Для этого необходимо подобрать эффективный алгоритм универсальной классификации
- Успешный пример — GLiNER [1]. Универсальная модель для задачи именованного распознавания сущностей в тексте (NER)
  - Обучалась на данных, сгенерированных ChatGPT
  - Работает лучше, чем UniNER-7B[1] (прошлый state-of-the-art)
  - >10 раз меньше, чем UniNER-7B



[1] https://huggingface.co/Universal-NER/UniNER-7B-all

Постановка задачи

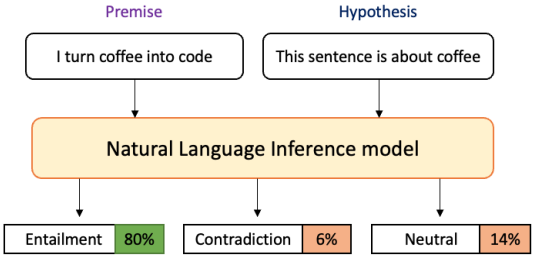
Цель: создание системы для универсальной автоматической обработки запросов на классификацию данных из различных доменов

Задачи:

1. Изучить существующие подходы к универсальной классификации, провести их сравнительный анализ. Определить алгоритм работы универсального классификатора и выбрать наборы данных для его обучения и валидации
2. Сформировать обучающую выборку и провести обучение универсального классификатора на подготовленной выборке
3. Разработать систему для автоматической обработки запросов на классификацию данных, используя обученную модель
4. Провести апробацию разработанной системы на реальных задачах классификации, оценить её практическую эффективность

Подходы к универсальной классификации

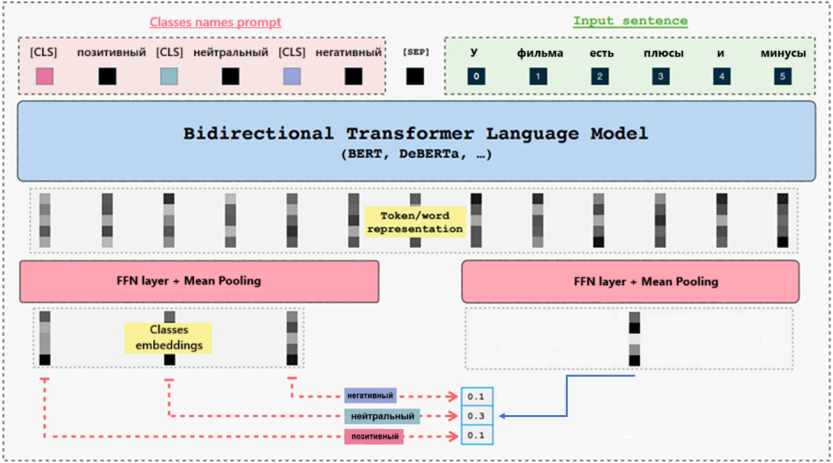
- Модели-кодировщики, обученные на NLI данных
  - MoritzLaurer/deberta-v3-large-zeroshot-v2.0[2]



- БЯМ (например, GPT-4o [2])
- БЯМ требуют много ресурсов, а скорость NLI моделей линейно зависит от количества классов в задаче

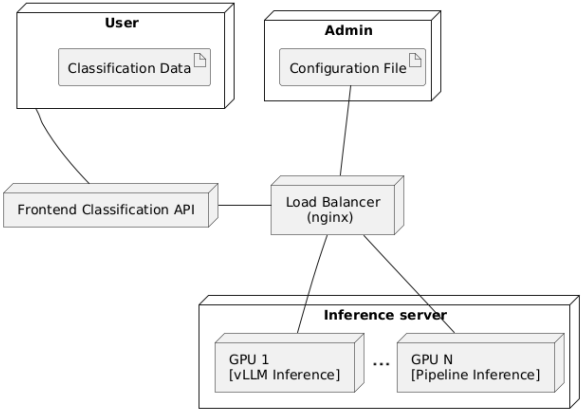
[2] https://huggingface.co/MoritzLaurer/deberta-v3-large-zeroshot-v2.0

Устройство модели



Архитектура системы

- Система должна
  - Быть распределенной
  - Поддерживать использование других классификаторов
  - Работать по REST API



Реализация

- Код обучения был написан с помощью PyTorch-Lightning[3]
  - GeraclCore – весь пайплайн алгоритма
  - Geracl – все, что нужно для обучения
  - DataModule – подготовка данных для обучения
    - Предобработка и токенизация
    - Формирование промпта
    - Создание батча
- Скрипты для подбора гиперпараметров с помощью WandB sweeps[4] и для автоматической валидации на RU-MTEB
- Код обучения доступен в репозитории[5] (≈2K строк кода)
- Код системы доступен в репозитории[6] (≈500 строк кода)

[3] https://lightning.ai/docs/pytorch/stable/
[4] https://docs.wandb.ai/guides/sweeps/
[5] https://github.com/deepvk/geracl
[6] https://github.com/VyrodovMikhail/zero_shot_classification

Апробация системы

- Для валидации были выбраны:
  - 5 задач из RU-MTEB (от 2 до 29 классов). Один текст имеет один класс
  - Продуктовый набор данных на детекцию разных видов токсичности (5 классов). Один текст имеет несколько классов

Модель	Тип	Размер	Ru-MTEB (Acc.)	Токсичность (F1)	Сред.
mDeBERTa-v3-base-mnli-xnli	NLI кодировщик	279M	0.45	0.17	0.31
bge-m3-zeroshot-v2.0	NLI кодировщик	568M	0.60	**0.32**	**0.46**
Qwen2.5-1.5B-Instruct	БЯМ	1.5B	0.56	0.13	0.35
Qwen2.5-3B-Instruct	БЯМ	3B	**0.63**	**0.24**	**0.44**
USER-bge-m3	Кодировщик	359M	0.53	0.15	0.34
GeRaCl-USER-bge-m3	Кодировщик	367M	0.62	0.18	0.40
USER2-base	Кодировщик	149M	0.52	0.11	0.32
GeRaCl-USER2-base[7]	Кодировщик	155M	**0.64**	**0.24**	**0.44**

[7] https://huggingface.co/deepvk/GeRaCl-USER2-base

Результаты

- Выполнен обзор и сравнительный анализ популярных БЯМ и кодировщиков, подтвердивший эффективность компактных универсальных классификаторов
- Было проведено обучение классификатора GeRaCl-USER2-base на основе модели USER2-base на наборе данных, собранном с помощью meta-llama/Llama-3.3-70B-Instruct и задач классификации из RU-MTEB
- Была разработана распределенная система для классификации текстов с помощью nginx, vLLM и HuggingFace, позволяющая использовать обученный классификатор либо произвольную БЯМ
- В ходе оценки системы установлено, что после обучения точность работы моделей выросла в среднем на 28 %, при увеличении их размера в среднем на 4 %. Во время оценки БЯМ и обученный классификатор использовались через разработанную систему

Построение обучающей выборки

- За основу взято 100,000 текстов из **allenai/c4**[8]
- В ходе экспериментов были проверены следующие подходы
  - Unsupervised — классы полностью сгенерированы meta-llama/Llama-3.3-70B-Instruct [3]
  - Weakly-Supervised — классы взяты из задач из RU-MTEB [4]
  - Самая эффективная версия[9] — Unsupervised + Weakly-supervised в пропорции $\approx 1/2$

Текст	**Классы**	**Правильный класс**
улучшает состояние прядей, за счет наличия в составе активных добавок	защита волос, оздоровление кожи, укрепление ногтей	защита волос
	медицина, математика, обществознание ...	медицина

[8] https://huggingface.co/datasets/allenai/c4
[9] https://huggingface.co/datasets/deepvk/GeRaCl_synthethic_dataset

Детали построения обучающей выборки

- Сначала каждый текст в allenai/c4 делился на сегменты и после этого в обучающий набор данных попадал случайный сегмент из текста
- Тексты из данного набора были разделены по предложениям на сегменты со случайным количеством слов от 20 до 150
- Разделение на предложение было сделано с помощью библиотеки razdel[10]

[10] https://github.com/natasha/razdel/

Список литературы

[1] Urchade Zaratiana и др. *GLiNER: Generalist Model for Named Entity Recognition using Bidirectional Transformer*. 2023. URL: https://arxiv.org/abs/2311.08526.

[2] OpenAI и др. *GPT-4o System Card*. 2024. URL: https://arxiv.org/abs/2410.21276.

[3] Aaron Grattafiori и др. *The Llama 3 Herd of Models*. 2024. URL: https://arxiv.org/abs/2407.21783.

[4] Artem Snegirev и др. *The Russian-focused embedders' exploration: ruMTEB benchmark and Russian embedding model design*. 2025. URL: https://arxiv.org/abs/2408.12503.