OTUS

Онлайн образование

otus.ru

Меня хорошо видно && слышно?

Защита проектной работы

Тема: «*Построение системы интеллектуального поиска по порталу СберСпасибо с применением технологий глубокого обучения*»

Кравченя Павел

Доцент кафедры ЭВМ и систем
Факультет электроники и вычислительной техники
ФГБОУ ВО «Волгоградский государственный технический университет»

План защиты

- Постановка задачи
- Используемые технологии и инструменты
- Архитектура системы
- Выполнение парсинга веб-страниц
- Векторная база данных Weaviate
- Метрики качества поиска
- Тестирование нагрузочной способности
- Выводы и результаты работы
- Планы по развитию

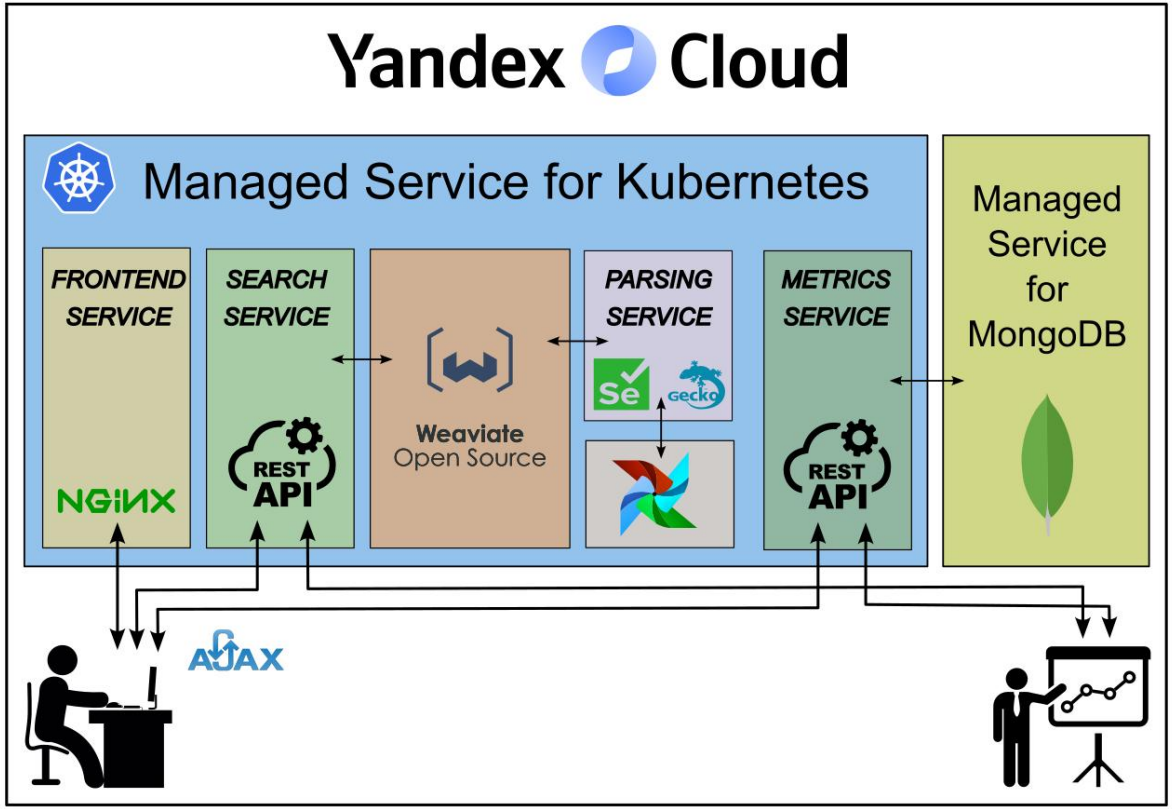
Постановка задачи

1. Реализовать систему «умного» поиска по порталу СберСпасибо
2. Поиск должен выполняться с учетом смысла содержимого
3. Архитектурно решение должно быть развернуто на кластере Kubernetes и выдерживать не менее 50 запросов в секунду
4. Требуется продумать метрики качества поисковой выдачи
5. Требуется предложить решение проблемы с некачественными текстовыми описаниями объектов

Используемые технологии и инструменты

1.	Kubernetes	— Разворачивание сервисов системы
2.	Weaviate	— Векторная БД для работы с эмбеддингами
3.	Selenium	— Парсинг динамического контента веб-страниц
4.	Airflow	— Запуск задач парсинга по расписанию
5.	MongoDB	— NoSQL БД для хранения информации о кликах пользователя
6.	NGINX	— Веб-сервер для интерфейса системы
7.	wrk	— Инструмент тестирования производительности

Архитектура системы



Выполнение парсинга веб-страниц

Delivery Club

www.delivery-club.ru

Об оформлении стола с легкостью позаботится Delivery Club. На выбор тысячи ресторанов и кафе на любой вкус и бюджет. Осталось только выбрать, и любимая еда уже на пути к вам.

Когда есть текстовое описание

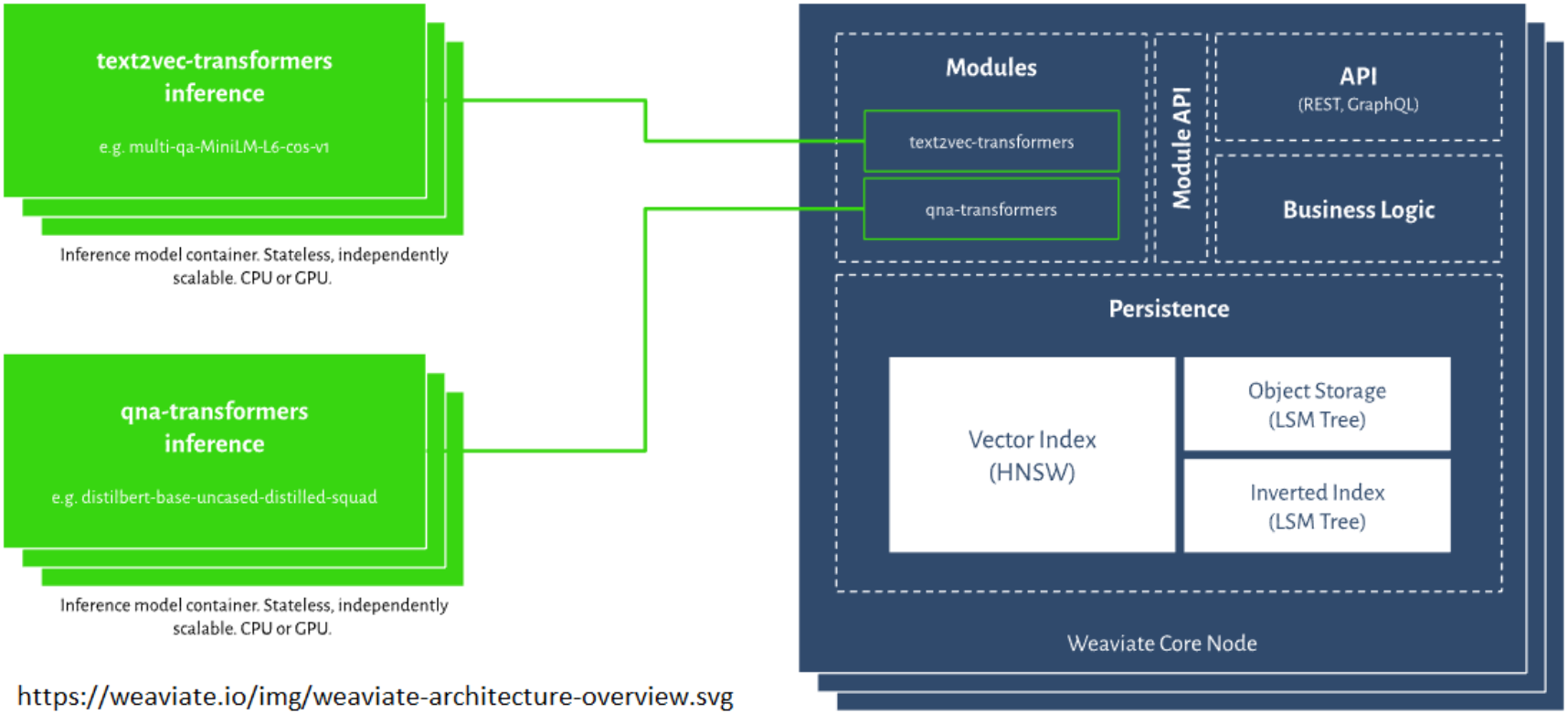
Ремонтиста

www.remontista.ru

и когда его нет

Накопить бонусы

Векторная база данных Weaviate



https://weaviate.io/img/weaviate-architecture-overview.svg

Метрики качества поиска

1. Качество поиска определяется соответствием найденных результатов ожиданиям пользователей

2. Оценить ожидания пользователей можно, анализируя статистику кликов на выданных системой результатах

3. Для сбора статистики и расчета метрик был реализован сервис, работающий в паре с системой поиска, но независимый от нее

4. Были рассчитаны «кликовые» метрики:
   а) **CTR** (Click-Through Rate)
   б) **AHC** (Average Highest Click)

$$\mathrm{CTR}_i = \frac{N_i}{N} \qquad \mathrm{AHC} = \frac{1}{N}\sum_{j=1}^{N} pos_j$$

N_i — количество результатов поиска, для которых был хотя бы один клик по первым ***i*** ссылкам;

N — общее количество результатов поиска;

pos_j — наивысшая позиция клика для j-того результата поиска.

Тестирование нагрузочной способности

Число реплик:
weaviate: 1, transformers: 2, search: 4

```
$ docker run --rm williamyeh/wrk -t6 -c60 -d10s http://51.250.20.120/search/?q=...
Running 10s test @ http://51.250.20.120/search/?q=...
  6 threads and 60 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   809.25ms  566.37ms   1.72s    54.17%
    Req/Sec     5.01        5.60         30.00    92.50%
  174 requests in 10.03s, 172.64KB read
  Socket errors: connect 0, read 0, write 0, timeout 150
Requests/sec:     17.34
Transfer/sec:     17.21KB
```

Число реплик:
weaviate: 1, transformers: 2, search: 8

```
$ docker run --rm williamyeh/wrk -t6 -c60 -d10s http://51.250.20.120/search/?q=...
Running 10s test @ http://51.250.20.120/search/?q=...
  6 threads and 60 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.16s   590.27ms   2.00s    59.32%
    Req/Sec     5.70      6.66         40.00    92.04%
  164 requests in 10.04s, 162.72KB read
  Socket errors: connect 0, read 0, write 0, timeout 105
Requests/sec:     16.33
Transfer/sec:     16.20KB
```

Число реплик:
weaviate: 1, transformers: 4, search: 8

```
$ docker run --rm williamyeh/wrk -t6 -c60 -d10s http://51.250.20.120/search/?q=...
Running 10s test @ http://51.250.20.120/search/?q=...
  6 threads and 60 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.04s   652.81ms   1.83s    50.00%
    Req/Sec     4.85      6.01         30.00    95.77%
  100 requests in 10.06s, 99.22KB read
  Socket errors: connect 0, read 0, write 0, timeout 80
Requests/sec:     9.94
Transfer/sec:     9.86KB
```

Число реплик:
weaviate: 4, transformers: 4, search: 8

```
$ docker run --rm williamyeh/wrk -t6 -c60 -d10s http://51.250.20.120/search/?q=...
Running 10s test @ http://51.250.20.120/search/?q=...
  6 threads and 60 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.07s   517.68ms   1.90s    58.82%
    Req/Sec     4.59      6.17         30.00    91.78%
  97 requests in 10.03s, 96.24KB read
  Socket errors: connect 0, read 0, write 0, timeout 63
Requests/sec:     9.67
Transfer/sec:     9.60KB
```

Демонстрация работы системы

Выводы и результаты работы

1. Разработана система для выполнения интеллектуального поиска по порталу СберСпасибо: https://github.com/kpdphys/MLOps/tree/main/SberProject

2. При поиске учитывается смысл содержимого за счет использования эмбедддингов, сформированных трансформерной нейросетью

3. Сервис развернут в Kubernetes, созданы и отлажены конфигурационные файлы. Обеспечена совместная работа всех сервисов в системе

4. Предложен подход к анализу качества работы поисковой системы, создан сервис для учетов кликов пользователей и вычисления метрик качества

5. Протестирована пропускная способность сервиса, результаты неудовлетворительные, система не масштабируется горизонтально.

Планы по развитию

1. Реализовать систему на основе другой векторной базы данных с целью увеличения пропускной способности. При этом могут потребоваться серьезные изменения в архитектуре проекта
2. Попробовать использовать трансформерную модель, обученную на русскоязычных текстах, с целью достижения лучшего качества поиска. Однако, построение такой системы может потребовать значительного количества вычислительных ресурсов.
3. Реализовать CI\CD в проекте

Спасибо за внимание!