Vysoké učení technické v Brně
Fakulta informačních technologií
Brno 2023

Software pro kombinování strukturovaných informací a textu, reprezentaci významu a odpovídání na otázky

Technická dokumentace

T A

Č R

Tento projekt je spolufinancován se státní podporou Technologické agentury ČR v rámci Programu průmyslového výzkumu a experimentálního vývoje TREND, Podprogram 1 – Technologičtí lídři, projekt FW03010656: MASAPI – Multilingvální asistent pro hledání, analýzu a zpracování informací a podporu rozhodování.

Úvod

Odpovídání na otázky je náročný úkol v oboru zpracování přirozeného jazyka. Systém musí disponovat konkrétními znalostmi, aby byl schopen na danou otázku odpovědět, a musí být schopen rozeznat relevanci faktů k otázce. K tomu je nutné zvolit vhodnou reprezentaci, která je schopna co nejlépe zachovat význam otázky a textu, obsahujícího odpověď. Dnes se k odpovídání na otázky používají především metody založené na modelech umělých neuronových sítí, schopných vytvářet kontextové reprezentace.

Tato technická dokumentace popisuje systém, který je schopen odpovídat na otázky, kladené na specifikovanou množinu dokumentů, a i na otázky, ke kterým takovéto dokumenty dodány nebyly (tzv. open-domain). Umožňuje také zpracovat strukturované informace a transformovat je do podoby, vhodné k trénování použitých neuronových sítí.

Analýza funkčních požadavků

V této části je uveden stručný souhrn vytyčených požadavků, které vyplynuly z konzultací s hlavním řešitelem projektu – firmou Lingea, jakož i kolegy z ÚFAL MFF UK.

Hlavním cílem bylo vytvořit systém, který bude schopen odpovědět na otázky v anglickém jazyce. Na tyto otázky bude poskytovat stručné odpovědi a bude možné určit, zda se má vygenerovat extraktivní či abstraktní odpověď. V případě extraktivní odpovědi se jedná o krátký podřetězec nacházející se v podkladových textech. Naopak abstraktní odpověď se v podkladových textech nemusí nacházet přímo v té konkrétní podobě, jak byla odpověď vygenerována, tedy může být například parafrázována.

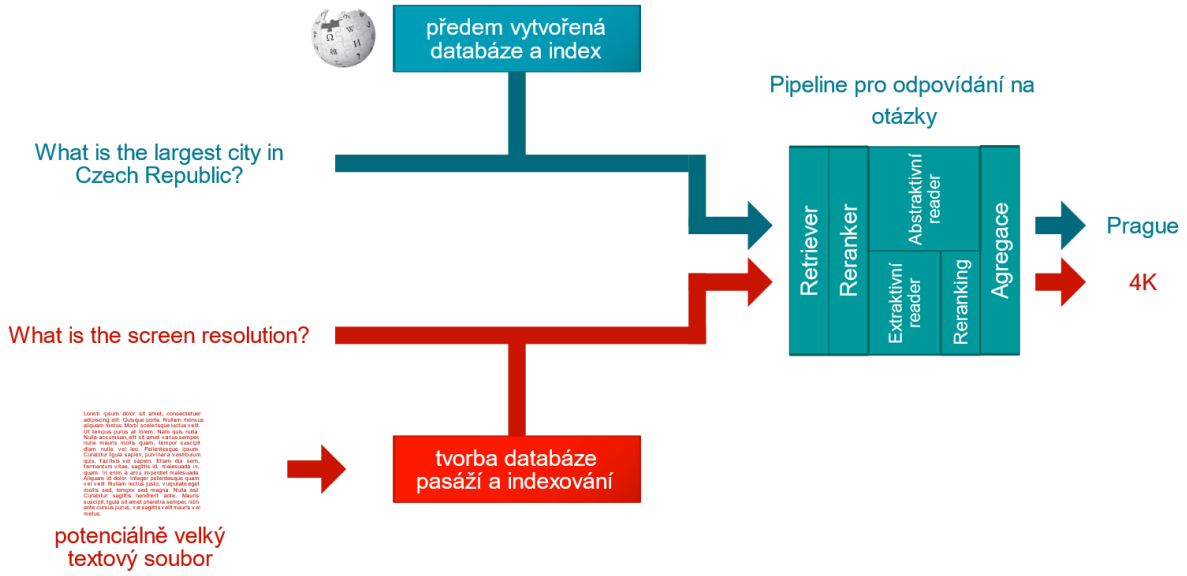
Otázky mohou být kladeny jak s poskytnutou množinou dokumentů, která definuje zdroj znalostí, tak musí umět i pracovat v tzv. režimu open-domain, kdy tyto dokumenty systému nejsou poskytnuty.

Dalším požadavkem je, aby bylo možné výsledný zdrojový kód začlenit jako modul do systému MASAPI. Za tímto účelem byl systém implementován jako balíček v jazyce Python (Python package), poskytující funkcionalitu pro případy užití systému s poskytnutými a neposkytnutými vstupními dokumenty.

Začlenění do systému MASAPI může také do budoucna znamenat, že zdrojové texty mohou být napsány v libovolném jazyce, budou přeloženy automatickým systémem překladu a budou začleněny do systému. Obdobným způsobem mohou být potom přeloženy otázky, případně odpovědi.

Technická dokumentace

Vytvořený systém očekává na vstupu otázku v angličtině a volitelně i anglický dokument, na který se dotazujeme. Pokud není dokument poskytnut, systém začne pracovat v režimu dotazování se nad otevřenou doménou a pokusí se k odpovědi využít předzpracovanou anglickou Wikipedii. Systém předpokládá otázky, na které je možné poskytnout stručnou odpověď.



Obrázek 1 - Horní cesta ukazuje použití systému v režimu dotazování se nad otevřenou doménou. Spodní pak nad dodaným dokumentem.

Řešení využívá modulární architekturu, skládající se z komponent, označovaných jako retriever, reranker, extraktivního reader, abstraktivního reader a z agregátorů, slučujících výsledky do finálního rozhodnutí. Ne všechny komponenty je nutné použít a je například možné vypnout abstraktivní reader, aby byla snížena výpočetní náročnost. Tento flexibilní přístup umožňuje adaptaci systému při nasazení. Bloková ilustrace – viz Obrázek 1. Následující odstavce popisují jednotlivé komponenty.

Komponenta Retriever se stará o výběr relevantních částí textu k dané otázce. Využívá k tomu databázi krátkých úseků textu, získaných z podkladových dat (Wikipedie či vstupní dokument), které jsou převedeny do vektorové reprezentace za pomocí BM25 nebo DPR (Karpukhin a další, 2020). Tyto vektorové reprezentace jsou použity k tvorbě indexu pro vyhledávání. Rozhodli jsme se použít oba přístupy (BM25 a DPR), protože BM25 může být vhodné v případech, kdy se jedná o data mimo doménu, na které nebylo natrénováno DPR, a nemáme dostatek dat pro dotrénování. Naopak DPR nalezne uplatnění při odpovídání nad otevřenou doménou, kde dosahuje lepších výsledků (Karpukhin a další, 2020).

Výsledky z retrieveru mohou být dále zpracovány pomocí rerankeru, který seřadí poskytnuté pasáže od retrieveru. Toto řazení je výpočetně náročnější než pomocí retrieveru, ale dovede zachytit jemnější interakce mezi tokeny pasáže a otázky.

Extraktivní reader příjme vybrané texty z předchozích kroků a extrahuje z nich odpověď v podobě krátkého, několikaslovného úseku textu. Tento modul je založen na modelu ELECTRA (Clark, Luong, Le, & Manning, 2020), který byl dotrénován pro odpovídání na otázky nad otevřenou doménou pomocí datové sady Natural Questions-Open (Lee, Chang, & Toutanova, 2019) za použití anglické Wikipedie.

Abstraktivní reader je komponenta, jež je schopna na základě dané otázky a vstupního textu vygenerovat odpověď, která se přímo v textu nemusí nacházet. Tento modul může být i používán pro vypočítání nového skóre odpovědi získané pomocí extraktivního readeru (Reranking blok ve schématu), a tak napomoci, podobně jako reranker pro pasáže, vybrat relevantnější odpověď.

Programátorská dokumentace

Systém je implementován jako modulární konfigurovatelná pipeline. Jeho schéma je možné vidět na Obrázek 1.

Implementace je psána v jazyce python. Pro tvorbu modelů byla použita knihovna PyTorch společně s knihovnou transformers od Hugging Face.

Systém je navržen jako python balík, aby mohl být snadno začleněn do systému MASAPI.

Balík je rozdělen do několika modulů a pod balíků. Následující seznam k nim poskytuje stručné informace pro rychlé zorientování se:

- data
  - obsahuje výchozí a ukázkový konfigurační soubor
- module
  - balík obsahující wrappery pro začlenění modelů do pipeline
- type
  - zde se nachází třídy pro zpracování konfigurací tvořených při běhu programu
- utility
  - různé pomocné funkce
- __main__.py
  - implementace vytyčených případů užití, evaluace a funkce pro tvorbu databáze a indexů
- checkpoint.py
  - modul pro práci s checkpointy modelů
- config.py
  - zpracování uživatelské konfigurace
- database.py
  - module pro práci s databází úryvků textů
- evaluate.py
  - modul s utilitami pro evaluaci
- responder.py
  - implementace pipeline

Kód je dostupný na adrese: https://github.com/KNOT-FIT-BUT/MASAPI_QA/

Popis ověření funkčnosti softwaru

Proběhlo manuální testování tohoto softwaru, které mělo za cíl vyzkoušet všechny základní případy užití. Systém byl otestován na ukázkových datech s dodanými dokumenty na vstupu a v režimu open-domain. Dále bylo otestováno několik základních druhu konfigurace jako například použití pipeline s a bez generativního readeru.

Výchozí konfigurace byla vyhodnocena v režimu open-domain na datasetu NQ-OPEN dev. Výsledky shrnuje Tabulka 1.

Exact match	48%
Průměrná doba odpovědi [s]	1,91
GPU paměť	cca 9GiB

Tabulka 1 - Vyhodnocení open-domain experimentu na datasetu NQ_OPEN dev. Exact match ukazuje procento odpovědí, které měli přesnou shodu alespoň na jedno referenční řešení.

V neposlední řadě došlo k testování samostatného použití retrievalu a tvorby databáze z datových sad Huggingface. Konkrétně se jednalo o datovou sadu cnn_dailymail.

Uživatelská příručka

Tato sekce popisuje instalaci a základní případy užití. Další informace týkající se konkrétních argumentů programu může uživatel získat vypsáním nápovědy pomocí parametru *-h*.

Instalace

Nejprve musíte mít nainstalovanou Javu, protože balíček Pyserini závisí na JDK 11. Nezapomeňte jej nainstalovat a nastavte JAVA_HOME na adresář JDK.

Dále nainstalujte faiss (faiss~=1.7.2). K tomu lze použít conda příkaz:

```
conda install -c conda-forge faiss-gpu
```

nebo

```
conda install -c pytorch faiss-gpu
```

Nakonec použijte standardní soubor requirements.txt pro instalaci dalších balíčků:

```
pip install -r requirements.txt
```

Podrobnosti týkající se instalace PyTorch lze nalézt zde https://pytorch.org/get-started/locally/.

Modely a zpracovaná Wikipedie

Chcete-li stáhnout natrénované modely a zpracovanou Wikipedii, která se používá pro odpovídání na otázky nad otevřenou doménou, stáhněte si tento soubor zip a rozbalte jej do požadovaného umístění:

```
r2d2.fit.vutbr.cz/checkpoints/nq-open/MASAPI_QA.zip
```

Použití

Model je natrénován tak, aby poskytoval stručné odpovědi na otázky pomocí dat z Wikipedie. Umožňuje však také použití vlastních dat. Tato část představuje několik případů použití pro hladký start se systémem.

Pokládání otázek nad otevřenou doménou

Chcete-li položit otázku bez udání vstupních dokumentů použijte následující příkaz:

```
./run.py ask "When Alan Turing was born?"
```

K zodpovězení této otázky bude použita předem vytvořená databáze pasáží z Wikipedie. Tuto databázi můžete změnit v konfiguračním souboru. Vizte část konfigurace.

Pokládání otázek nad poskytnutým textem

Pokud máte text, který chcete použít jako podklad pro zodpovězení otázky, můžete použít argument ask v kombinaci s argumentem *text* (zkratka *t*):

```
./run.py ask "When Alan Turing was born?" -t
bibliography_of_alen_turing.txt
```

nebo

```
./run.py ask "When Alan Turing was born?" -t "... Alan Mathison Turing
was born in 1912... "  --no_title
```

chcete-li přímo předat textový obsah. Volitelný argument *–no_title* říká, že první řádek neobsahuje titulek a měl by být brán jako běžná součást textu. Tento argument lze použít i se vstupem v podobě

souboru. Tedy systém defaultně očekává na prvním řádku titulek, je to z toho důvodu, aby byl schopen si tento titulek speciálně poznačit.

Konfigurace

Tato část popisuje konfiguraci systému pro odpovídání na otázky, která se používá při volání systému s argumentem ask.

Výchozí konfiguraci najdete na:

```
masapiqa/data/default_config.py
```

Také je možné se podívat na ukázkovou konfiguraci:

```
masapiqa/data/example_config.py
```

Nyní následuje popis nejdůležitějších částí konfiguračního souboru:

- use_gpu – povolí/zakáže použití GPU
- cache_dir – cesta ke složce, která má být použita jako cache pro ukládání modelů a databází
- retriever_models – seznam retriver modelů, která mají být načteny
    - K dispozici je vícero retriverů:
        - R2D2EncoderFramework
            - Jedná se o implementaci DPR
            - Popis parametrů:
                - label – používá se pro identifikaci modelu
                - model – naučený encoder, který se používá k získání reprezentace otázky
                - tokenizer – používá se k tokenizaci vstupu
                - framework (R2D2EncoderFramework) - wrapper retriveru umožňující jej použít v pipelině
                - checkpoint – cesta k checkpointu obsahujícím natrénovaný model
                - database – soubor obsahující kontexty
                - index – embedované kontexty používající se k vyhledávání
        - PyseriniRetrieverFramework
            - Implementace používající Pyserini (Lucene)
            - Může se jednat o DPR nebo BM25.
            - Popis parametrů:
                - label – používá se pro identifikaci modelu
                - framework (PyseriniRetrieverFramework) - wrapper retriveru umožňující jej použít v pipelině
                - database – Cesta ke složce s databází. V tomto případě databáze také obsahuje index určující, zda bude použito BM25 nebo DPR.
        - FirstKFramework
            - Jednoduchá metoda, která vždy vezme top-k prvních záznamů z databáze
            - skóre každé pasáže je uměle nastaveno na 1
            - Popis parametrů:
                - label – používá se pro identifikaci modelu

        - framework (FirstKFramework) - wrapper retrieveru umožňující jej použít v pipelině
        - database – cesta ke složce s databází
- passage_reranker_models – seznam rerankerů, které mají být načteny
    - Pole jsou podobná těm, které jsou u R2D2EncoderFramework.
- reader_models – seznam readerů, které mají být načteny
    - Pole jsou podobná těm, které jsou u R2D2EncoderFramework.
- aggregation – Konfigurace agregační části, která se používá k agregaci skóre z retrieveru, rerankeru, extraktivního reader a generativního readeru.
    - Můžete jej použít k definování vah a biasu ke každému skóre [(retriver_label, reranker_label, ext_reader_label, abst_reader_used_for_reranking_label, {"w1": num, "w2": num, "w3": num, "w4": num, "bias": num})]
    - Pokud chcete definovat agregace pro konfigurace bez rerankeru nebo abstraktního readeru použijte None místo jejich značky (label).
- open_domain_config – konfigurace pro open-domain
    - Popis parametrů:
        - retriever
            - model – Značka retrieveru, který se má použít (definovaná v retriever_models).
            - top_k – kolik kontextů se má získat
        - passage_reranker
            - model – Značka rerankeru, který se má použít (definovaná v passage_reranker_models).
        - extractive_reader
            - model – Značka extraktivního readeru, který se má použít (definovaná v reader_models).
            - reader_top_k_answers – kolik nejlepších odpovědí má být vráceno
            - reader_max_tokens_for_answer – v úvahu budou brány pouze odpovědi mající maximálně daný počet tokenů
            - generative_reranking – aktivuje/deaktivuje reranking odpovědí pomocí abstraktním readeru (používejte pouze z abstraktním readerem)
        - abstractive_reader
            - Můžete definovat abstraktní reader, který se má použít společně s extraktivním.
            - label – Značka abstraktního readeru, který se má použít (definovaná v reader_models).
            - score_aggregation – aktivuje/deaktivuje agregaci skóre
- on_demand – konfigurace pro případ, kdy jsou poskytnuty kontexty
    - jsou použity stejná pole až na část retrieveru
    - Vzhledem k tomu, že index je vytvářen za běhu, není nutné poskytovat označení retrievera s vytvořeným indexem. Místo toho použijte pole modelu k určení, zda chcete použít DPR nebo BM25. Definujete jej zadáním řetězce "DPR" nebo "BM25".
    - Existují také dvě další pole, batch a threads. Batch definuje velikost dávky při použití DPR a threads definuje počet paralelních běhů při použití BM25.

Reference

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *International Conference on Learning Representations.*

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., . . . Yih, W.-t. (2020). *Dense Passage Retrieval for Open-Domain Question Answering.* Association for Computational Linguistics.

Lee, K., Chang, M.-W., & Toutanova, K. (2019). Latent Retrieval for Weakly Supervised. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.*