Aplikace pro doporučování podobných veřejných zakázek

## Historie dokumentu

|  |  |  |
| --- | --- | --- |
| Datum | Autor | Změna |
| - | Milan | Inicializace (abstrakt, data, techstack) |
| 8.1.2020 | Milan | Update (abstract, data, techstack) |
| 8.1.2020 | Milan | Změna struktury a přidání obsahu |
| 8.1.2020 | Milan | Přidání logického plánu |
| 8.1.2020 | Milan | Přidání časového plánu |
| 10.1.2020 | Milan | Doplnění textu k jednotlivým kapitolám |
| 15.1.2020 | Milan | Přidání Treexu u FEA04 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Autor: Milan Vancl

Vedoucí: Jaroslav Kuchař

Oponent: Marek Sušický

Obsah

[Historie dokumentu 1](#_Toc29571919)

[Abstract 4](#_Toc29571920)

[Úvod 4](#_Toc29571921)

[Data 4](#_Toc29571922)

[Technologie 4](#_Toc29571923)

[Plán 5](#_Toc29571924)

[Logický pohled 5](#_Toc29571925)

[Časově sousledný pohled 6](#_Toc29571926)

[STH00 – Stakeholders 7](#_Toc29571927)

[STH01 – Průzkum uživatelů a jejich požadavků 7](#_Toc29571928)

[DAT00 - Data 7](#_Toc29571929)

[DAT01 - Dokumenty veřejných zakázek 7](#_Toc29571930)

[DAT02 - Kategorie produktů a služeb 8](#_Toc29571931)

[DAT03 - Katalog produktů a služeb 8](#_Toc29571932)

[DAT04 – Data o lokacích 8](#_Toc29571933)

[FEA00 - Extrakce feature z dokumentů 8](#_Toc29571934)

[FEA01 - Klasifikace dokumentů 8](#_Toc29571935)

[FEA01a - Výběr metody a algoritmu 8](#_Toc29571936)

[FEA01b - Výběr technologie 8](#_Toc29571937)

[FEA01c - Návrh procesu učení 9](#_Toc29571938)

[FEA01d - Učení modelu 9](#_Toc29571939)

[FEA01e -Sestavení komponenty pro klasifikaci 9](#_Toc29571940)

[FEA02 - Document embedding 9](#_Toc29571941)

[FEA02a - Výběr metody a algoritmu 9](#_Toc29571942)

[FEA02b - Výběr technologie 9](#_Toc29571943)

[FEA02c - Sestavení komponenty pro extrakci embeddingu 9](#_Toc29571944)

[FEA03 - Metrika podobnosti dokumentů 9](#_Toc29571945)

[FEA03a - Výběr algoritmu 9](#_Toc29571946)

[FEA03b - Výběr technologie 9](#_Toc29571947)

[FEA03c - Sestavení komponenty pro porovnávání dokumentů 9](#_Toc29571948)

[FEA04 - Extrakce předmětu 10](#_Toc29571949)

[FEA04a - Výběr metody a algoritmu 10](#_Toc29571950)

[FEA04b - Výběr technologie 10](#_Toc29571951)

[FEA04c – Návrh procesu extrakce předmětu 10](#_Toc29571952)

[FEA04d - Sestavení komponenty pro extrakci předmětu 10](#_Toc29571953)

[REC00 - Doporučovací systém 10](#_Toc29571954)

[REC01 - Výběr doporučovací metody 10](#_Toc29571955)

[REC02 - Výběr technologie 10](#_Toc29571956)

[REC03 - Návrh systému 10](#_Toc29571957)

[REC04 – Sestavení 10](#_Toc29571958)

[APP00 - Aplikace 10](#_Toc29571959)

[APP01 - Výběr technologie 10](#_Toc29571960)

[APP02 - Sestavení prototypu 11](#_Toc29571961)

[APP03 - Integrace s ostatními komponentami 11](#_Toc29571962)

[APP04 - Nasazení? 11](#_Toc29571963)

[DIP00 - Psaní diplomového dokumentu 11](#_Toc29571964)

# Abstract

Návrh, sestavení a naučení modelu pro extrakci předmětu (produkt, služba) z textu dokumentace veřejných zakázek.

Vytvoření aplikace pro vyhledávání (doporučování) podobných zakázek podle předmětu, lokace, atd.

# Úvod

## Data

Data pro učení modelu – otevřená data veřejných zakázek v ČR. Implikuje český jazyk – nutnost naučení modelu NLP pro češtinu.

Dataset pro učení klasifikátoru – opendata VZ + olabelování (Hlídač Státu, @Michal Bláha)

## Technologie

NLP – python, spaCy?, deeppavlov

Download dat – java (projekt public-contracts)

Infrastruktura – vm-opendata (hlavní databáze, integrace), lokální dev. Prostředí, prof-gpu (výpočetní server)

Aplikace: web app (python(django/flask))

# Plán

## Logický pohled

1. STH00 – Stakeholders
   1. STH01 – Průzkum uživatelů a jejich požadavků
2. DAT00 - Data
   1. DAT01 - Dokumenty veřejných zakázek
   2. DAT02 - Kategorie produktů a služeb
   3. DAT03 - Katalog produktů a služeb
   4. DAT04 – Data o lokacích
3. FEA00 - Extrakce feature z dokumentů
   1. FEA01 - Klasifikace dokumentů
      1. FEA01a - Výběr metody a algoritmu
      2. FEA01b - Výběr technologie
      3. FEA01c - Návrh procesu učení
      4. FEA01d - Učení modelu
      5. FEA01e -Sestavení komponenty pro klasifikaci
   2. FEA02 - Document embedding
      1. FEA02a - Výběr metody a algoritmu
      2. FEA02b - Výběr technologie
      3. FEA02c - Sestavení komponenty pro extrakci embeddingu
   3. FEA03 - Metrika podobnosti dokumentů
      1. FEA03a - Výběr algoritmu
      2. FEA03b - Výběr technologie
      3. FEA03c - Sestavení komponenty pro porovnávání dokumentů
   4. FEA04 - Extrakce předmětu
      1. FEA04a - Výběr metody a algoritmu
      2. FEA04b - Výběr technologie
      3. FEA04c – Návrh procesu extrakce předmětu
      4. FEA04d - Sestavení komponenty pro extrakci předmětu
4. REC00 - Doporučovací systém
   1. REC01 - Výběr doporučovací metody
   2. REC02 - Výběr technologie
   3. REC03 - Návrh systému
   4. REC04 - Sestavení
5. APP00 - Aplikace
   1. APP01 - Výběr technologie
   2. APP02 - Sestavení prototypu
   3. APP03 - Integrace s ostatními komponentami
   4. APP04 - Nasazení?
6. DIP00 - Psaní diplomového dokumentu

## Časově sousledný pohled

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Leden | | | | Únor | | | | Březen | | | | Duben | | | | Květen 7. 5. deadline | | | |
| STH01 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DAT01 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DAT02 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DAT03 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DAT04 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| FEA01 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| FEA02 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| FEA03 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| FEA04 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| REC01 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| REC02 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| REC03 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| REC04 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| APP01 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| APP02 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| APP03 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| APP04 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DIP00 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# STH00 – Stakeholders

## STH01 – Průzkum uživatelů a jejich požadavků

Pokusit se zkontaktovat nějaké zastupitele obcí ohledně jejich názor a požadavky na systém?

Marek má asi kontakt na starostu ve Slaném, se kterým se o projektu bavil, tak možná by se dalo s ním něco probrat.

# DAT00 - Data

## DAT01 - Dokumenty veřejných zakázek

Dataset dokumentů veřejných zakázek (dále VZ) vytvářím stahováním dokumentací VZ zadavatelů uveřejněných ve věstníku veřejných zakázek (dále VVZ). Používám k tomu open-source stahovátko *public-contracts* (dále PC)*,* vyvíjeného v rámci projektu *opendatalab*.

PC pracuje s centrální Postgres databází, kam ukládá veškeré záznamy.

PC nalezne všechny zadavatele VZ v systému VVZ scrappováním jejich webového vyhledávače profilů. Ke každému zadavateli ukládá url adresu jeho profilu, kde jsou zveřejňované zadavatelovy VZ.

Existuje několik portálů s profily zadavatelů. Mezi největší patří:

* [www.vhodne-uverejneni.cz](http://www.vhodne-uverejneni.cz)
* [www.e-zakazky.cz](http://www.e-zakazky.cz)
* [www.egordion.cz](http://www.egordion.cz)
* nen.nipez.cz
* [www.tenderarena.cz](http://www.tenderarena.cz)
* [www.profilzadavatele.cz](http://www.profilzadavatele.cz)

Každý z těchto portálů podle zákona podporuje XML API, přes které jsou dostupná základní data o všech zakázkách daného profilu. Mimo jiné jsou v těchto datech také odkazy na jednotlivé dokumenty k zakázkám.

Pro dokumenty již portály speciální API nemají, takže pro získání každého dokumentu je potřeba opět scrappovat jejich webová rozhraní.

Stahované dokumenty jsou ukládané v plain-text formátu do databáze. O extrakci textu se stará knihovna Apache Tika s TesseractOCR. Vzhledem k tomu, že Tika nepodporuje rozpoznávání českého jazyka, v některých případech špatně zpracovává PDF dokumenty. TesseractOCR češtinu podporuje a bez problému čte i dokumenty, se kterýma má Tika problém. Kvůli tomu převádím PDF dokumenty do obrazové reprezentace a zpracovávám samotným OCR.

*Aktuální stav: 9. 1. 2020*

|  |  |
| --- | --- |
| *Počet zadavatelů* | *22888* |
| *Celkový počet zakázek (období leden - červen 2019)* | *136k* |
| *Počet postahovaných zakázek* | *17k* |
| *Celkový počet dokumentů k zakázkám* | *900k* |
| *Počet postahovaných dokumentů* | *135k* |

*Stahování trvá dlouho (cca 1000/den). Systém tenderarena.cz pravidelně odmítá službu kvůli obraně proti zahlcení, čímž tvoří “bottleneck” procesu.*

## DAT02 - Kategorie produktů a služeb

Jako kategorie (třídy) zakázek a jejich dokumentů chci využít CPV kódy. CPV kódy jsou (zřejmě) uváděny u všech zakázek ve VVZ. Na profilech zadavatelů už z pravidla nejsou a v XSD schématu API profilů položku ani nemají.

Potřebuji tedy CPV kódy přiřadit k zakázkám dodatečně. Zároveň je zde ale problém s identifikací zakázek. Ve VVZ existuje evidenční číslo zakázky (dále ECZ), které ale také není uvedeno u všech zakázek na profilech.

Získat CPV kódy by asi mohlo jít vyscrapováním VVZ, nebo z databáze systému, kde je mají. Jedním z takových systémů je Hlídač Státu . Existuje sice API ke hlídači, ale neposkytuje ke smlouvám CPV kódy.

*Aktuální stav: 9. 1. 2020*

|  |  |
| --- | --- |
| *Počet zakázek s ECZ* | *30k* |
| *Počet postahovaných zakázek s ECZ* | *9k* |
| *Počet postahovaných dokumentů k zakázkám s ECZ* | *88k* |

*Snažím se vyjednat se správcem Hlídače státu získání CPV kódů.*

## DAT03 - Katalog produktů a služeb

Pro přesnější extrakci předmětu zakázky (produktu/služby) by mohlo jít vyhledávat položky podle „katalogu“.

*Aktuální stav: 9. 1. 2020*

*Marek říkal něco o katalogu z Alzy. Nevím, jestli to nějak dopadne.*

*Zkoušel jsem dělat scrapper na heureka.cz, která má ale asi 22 milionů produktů a hodně omezené možnosti stránkování. Za předpokladu, že by mě nechali bez postihu scrappovat by to podle mého odhadu trvalo asi 300 hodin.*

## DAT04 – Data o lokacích

Pro zeměpisnou identifikaci zakázek je potřeba získat data o adresách zadavatelů/dodavatelů a jejich místním zaměření.

# FEA00 - Extrakce feature z dokumentů

## FEA01 - Klasifikace dokumentů

### FEA01a - Výběr metody a algoritmu

V tuhle chvíli (od června 2019) drží SOTA v NLP metoda XLNet, která rozšiřuje předešlou úspěšnou metodu BERT.

BERT byl vydaný v říjnu 2018 a od té doby se stal SOTA v mnoha NLP problémech. Stal se tak velice populárním a vzniklo z něj mnoho upravených modelů. Jeho sílou je množství před-trénovaných modelů, které jdou ladit pro specifické NLP úlohy. Nyní existují modely naučené pro desítky světových jazyků, včetně češtiny.

BERT, kvůli jeho veliké podpoře, volím pro své účely i já.

### FEA01b - Výběr technologie

Existují dva frameworky poskytující model s podporou češtiny:

* Transformers (huggingface) - BERT-Base, Multilingual Cased (New, recommended)
* deeppavlov - SlavicBERT, Slavic (bg, cs, pl, ru)

Zatím jsem vyzkoušel klasifikaci s deeppavlov modelem na kaggle\_insults datasetu.

### FEA01c - Návrh procesu učení

Mám k dispozici Profinití stroj: 2 x Intel(R) Xeon(R) Silver 4114 CPU, 2 x Nvidia GTX 2080 Ti, 64 GB DDR4  
Jeho využití je umožněno za použití dockeru.

*Aktuální stav 9. 1. 2020*

*Mám připravený vzorový (kaggle\_insults) docker pro učení deeppavlov modelu na obou CPU i GPU.*

### FEA01d - Učení modelu

Vybranou technologií naučím výsledný klasifikátor.

### FEA01e -Sestavení komponenty pro klasifikaci

Aby bylo možné naučený klasifikátor použít v aplikaci, sestavím komponentu, která k němu bude tvořit rozhraní a bude zaintegrovaná do aplikace.

## FEA02 - Document embedding

### FEA02a - Výběr metody a algoritmu

* FastText (jeví se nejlépe)
* Doc2vec,
* BERT (huggingface),
* Sentence-BERT,
  + Zřejmě SOTA, ale nepodporuje češtinu
* Word Mover’s Embedding,
* Universal Sentence Embedidng (nepodporuje češtinu),
* GloVe (nepodporuje češtinu)

Pravděpodobně bude nejlepší použít BERT embedding, když ho používám jako jazykový model.

### FEA02b - Výběr technologie

Viz. FEA01b

### FEA02c - Sestavení komponenty pro extrakci embeddingu

Obdobně jako FEA01e

## FEA03 - Metrika podobnosti dokumentů

### FEA03a - Výběr algoritmu

Nejlepší bude pravděpodobně metrika na základě embeddingu, kvůli zachycení sémantických podobností. Reálně si ale nejsem jistý, zdali to bude dostatečně fungovat. Kosinová vzdálenost by mohla dobře měřit podobnost dvou embeddingů.

Další zajímavou alternativou je WMD.

### FEA03b - Výběr technologie

Podle výběru algoritmu.

### FEA03c - Sestavení komponenty pro porovnávání dokumentů

Komponenta by měla sloužit jako metrika pro doporučovací engine.

## FEA04 - Extrakce předmětu

### FEA04a - Výběr metody a algoritmu

Potřeba zprocesovat text:

1. Tokenizace
2. Lemmatizace
3. POS tagování
4. Analýza závislostí slov

Podle Davida Šenkýře je nejspolehlivější explicitní návrh patternů větného rozboru, které se matchují v textu.

Alternativně by mohlo jít vyzkoušet sumarizaci dokumentu, ale pochybuji o výsledcích na extrakci předmětu.

#### Větný rozklad

4 úrovně jazykového rozkladu:

1. Slovní úroveň – prostý text (tokeny)
2. Morfologická úroveň – lemma, POS tagy
3. Analytická úroveň – syntaktická analýza (větná struktura)
4. Tektogramatická úroveň - sémantická anotace (vztahy mezi důležitými slovy)

*Aktuální stav: 15. 1.*

*Také bych se o tom chtěl pobavit na schůzce na ZČU.*

### FEA04b - Výběr technologie

Pro větný rozbor lze použít:

* Treex
  + má český model
  + poskytuje až tektogramatickou úroveň rozkladu, čímž se jeví být nejlepší
  + implementace v Perlu
    - slabší podpora
    - komplikovaná instalace
* UDPipe/Udapi
  + Používá Universal Dependencies anotační modely
  + Neposkytuje tektogramatickou úroveň
  + Krom jiných užití existují i jako knihovny pro python
    - Dokonce i api pro spacy, ovšem model na pozadí funguje stále stejně
* MorphoDiTa - má český model, poskytuje pouze morfologickou úroveň rozkladu
* Stanford NLP
* Možná TreeTagger, RFTagger

*Aktuální stav: 16. 1.*

### FEA04c – Návrh procesu extrakce předmětu

Podle vybrané metody a technologie.

### FEA04d - Sestavení komponenty pro extrakci předmětu

Komponenta by měla extrahovat z dokumentace zakázek předmět (produkt, službu) pro vyhledávání/feature extraction.

# REC00 - Doporučovací systém

Systém by měl doporučovat na základě vyhledávání (zadaná klíčová slova, dokument) podobné zakázky (kategorie, předmět, doba, lokalita).

Ještě je potřeba domyslet do jaké míry by mělo doporučování fungovat.

## REC01 - Výběr doporučovací metody

Pravděpodobně content-base algoritmus, využívající množinu feature zakázek.

Na collaborative-filtering nebude mít systém uživatelské vstupy.

## REC02 - Výběr technologie

Nutno prozkoumat.

## REC03 - Návrh systému

Závisí na REC01 a REC02.

## REC04 – Sestavení

Doporučovací systém bude jako další komponenta, která na vstupu bude očekávat feature extrahované z uživatelského vstupu. Na základě feature systém vypočte doporučené položky.

# APP00 - Aplikace

## APP01 - Výběr technologie

Pravděpodobně některý z Python web frameworků.

* Django,
* Flask

## APP02 - Sestavení prototypu

Jednoduché webové rozhraní s možností zadání vstupu pro vyhledávání a oblastí se zobrazením výstupu. Prototyp bude mít mockované komponenty.

## APP03 - Integrace s ostatními komponentami

Do prototypu aplikace zaintegrovat všechny funkční komponenty, aby byl celkový systém schopný fungovat.

## APP04 - Nasazení?

Bude-li to potřeba, připravit a alokovat infrastrukturu pro nasazení celého systému.

# DIP00 - Psaní diplomového dokumentu

Průzkum dosavadních systémů a řešení doporučování veřejných zakázek.

* Tenderman.cz- pricing až 10k Kč/měsíc
* Vhodne-uverejneni.cz