FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

Semestrální práce KIV/UIR

Petr Kraus
krauspetr97@gmail.com
Celková doba řešení: 40 hodin

22. května 2019

Obsah

1 Zadání

Ve zvoleném programovacím jazyce navrhněte a implementujte program, který umožní automaticky určit, zda je ve větě pojmenovaná entita, či nikoli. V případě, že je, určete její typ. Pojmenované entity jsou významová slova a spojení ve větě jako např. jména osob, institucí, geografické názvy, časové údaje, atd.

Implementujte alespoň dva různé algoritmy (z přednášek i vlastní) pro tvorbu příznaků reprezentující text a dvě různé metody klasifikace dle vlastní volby s využitím metody reprezentace textu. Vyhodnoťte kvalitu klasifikačního algoritmu na anotovaných datech. Použijte metriky přesnost (accuracy). Otestujte všechny konfigurace programu.

2 Analýza problému

2.1 Popis problémů

Úloha skrývá několik problémů, prvním problémem je reprezentace textu. Čím lepší reprezentace zpráv bude, tím rychlejší a přesnější průběh detekčních algoritmů bude.

Druhým problémem je vhodné zvolení výpočtu vzdáleností mezi dvěma body, které jsou kritické pro detekčních metod.

Posledním problémem je výběr samotných detekčních algoritmů. Pokud jsou zvoleny nevyhovující algoritmy, přesnost anotací dramaticky klesne.

2.2 Reprezentace textu

Abychom mohli text reprezentovat, je potřeba vytvořit příznaky.

Pro tvorbu příznaků zle požít model **Bag of Words** (BoW). V tomto modelu je text reprezentován jako "taška"nebo "batoh"slov, ve kterém jsou všechna slova.

Dalšími algoritmy pro tvorbu příznaků jsou **Term Frequency** (TF) a **Term Frequency - Inverse Document Frequency** (TF-IDF). TF vyjadřuje, jak často se slovo v textu vyskytuje. Hodnota TF pro slovo v dokumentu se vypočítá pomocí vzorce číslo 1.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}} \tag{1}$$

Metoda TF-IDF je spojení TF a Inverse Document Frequency, jejíž hodnota reprezentuje "důležitost"slova. Čím vícekrát se slovo vyskytuje v dokumentech, tím méně je důležité. Hodnota IDF pro slovo v dokumentu se vypočítá pomocí vzorce číslo 2, kde čitatel je velikost databáze dokumentů a jmenovatel je počet dokumentů, které obsahují slovo, pro které se IDF hodnota počítá.

$$idf_i = \log \frac{|D|}{|\{j : t_i \in d_i\}|} \tag{2}$$

Další metoda pro vytvoření příznaků textu se nazývá **N-gram**. N-gram je definován jako sled N po sobě jdoucích položek z dané posloupnosti, pro tuto úlohu by to byla posloupnost slov.

2.3 Výpočet vzdálenosti mezi dvěma body

Pro výpočet vzdálenosti mezi dvěma body lze využít několik metod, nejznámější a nejpoužívanější jsou **Euklidovská a Menhattanská vzdálenost**.

Euklidovská vzdálenost je vzdálenost mezi dvěma body v euklidovském prostoru, tedy "vzdušnou čarou". Vzorec pro výpočet euklidovské vzdálenosti je vzorec č. 3:

$$d(x, y) = \|x - y\| = \sqrt{\sum_{n=1}^{n} (x_i - y_i)^2} \tag{3}$$

Mahattonská vzdálenost je součet absolutních rozdílů zadaných vektorů v kartézké soustavě souřadnic. Vzorec pro výpočet manhattanské vzdálenosti mezi dvěma doby je zobrazen jako vzorec č. 4

$$d_1(p, q) = \|p - q\|_1 = \sum_{i=1}^{n} |p_i - q_i| \tag{4}$$

2.4 Detekční algoritmy

2.4.1 Detekční algoritmy s učitelem

Mezi tyto algoritmy patří například N-nejbližší sousedé (**K-NN**), nebo **rozhodovací strom** (Desition tres).

K-NN při spuštění rozdělí trénovací data do předpřipravených skupin, následně se do algoritmu přidají neanotovaná data. Samotný průběh algoritmu spočívá ve vypočítání vzdálenosti neanotovaného prvku se všemi anotovanými. Algoritmus následně vybere K nejbližšíc sousedů a podle nich se určí třída, do které má být neanotovaný prvek zařazen.

Rozhodovací strom podle názvu vytváří strom. Uzly tohoto stromu reprezentují rozhodování podle jedné vybrané vlastnosti objektu. Tento strom se musí vytvořit z množiny daných objektů, které poskytuje učitel nebo jiný algoritmus. Největším problémem tohoto algoritmu je vytvoření samotného stromu. Strom totiž musí co nejlépe odlišovat objekty, aby byl algoritmus užitečný.

2.4.2 Detekční algoritmy bez učitele

Mezi tyto algoritmy patří například **K-means**. K-means předpokládá, že shlukované objekty lze chápat jako body v euklidovském prostoru a že počet shluků K je předem dán. Shluky jsou definovány svými centroidy, což je střed vypočítaný ze všech bodů v jednom shluku. Algoritmus postupuje iterativně tak, že se vyjde z nějakých centroidů, přiřadí do nich body a přepočítá centroidy. Toto trvá do doby, než se poloha centroidů neustálí.

3 Návrh řešení

Jako parametrizační metody jsem zvolil BoW a TF-IDF. Pro výpočet vzdáleností mezi dvěma vektory jsem využil Euklidovskou vzdálenost, která je přesnější při výpočet vzdálenosti vektorů. K-Means a K-NN (pro K = 1) jsem zvolil jako detekční algoritmy.

4 Popis řešení

4.1 Popis tříd

4.1.1 App

App je hlavní třída, která řídí celý program. Nejdříve zkontroluje vstupní parametry, na jejich základě zvolí parametrizační a detekční algoritmus, a vstupní soubor. Pokud jsou zvolené špatné vstupní parametry, vypíše jakým způsobem je nastavit a zastaví běh programu. Při správně zadaných parametrech načte vstupní soubor, dle zvoleného parametrizačního algoritmu uloží do slovníku a vytvoří vektory pro jednotlivé věty. Následuje rozdělení vět do tříd pomocí zvoleného detekčního algoritmu, výsledek uloží do souboru a vypíše přesnost.

4.1.2 Cluster

Třída, která reprezentuje jednotlivé shluky. Má dva důležité atributy a to vektor, který reprezentuje střed shluku a list vět, které patří do clusteru.

4.1.3 DetectionAlgs

DetectionAlgs je třída poskytující detekční algoritmy K-means a K-NN. Dále pomocné metody k těmto algoritmům, konkrétně metodu getMin(), která vybere minimum ze zadaného listu a metodu euclideanDistance(), která vypočte vzdálenost mezi dvěma zadanými vektory.

4.1.4 EntityType

EntityType je enum obsahující názvy tříd pojmenovaných entit a jejich zkratky. Konkrétně:

- PERSONAL_NAME("po") - jména osob
- INSTITUTION("i") - názvy institucí a organizací
- GEOGRAPHICAL_NAME("g") - geografické názvy
- TIME_EXPRESSIONS("t") - časové údaje
- ARTIFACT_NAMES("o") - označení objektů, produktů, dokumentů a staveb
- AMBIGUIUS("a") - nejednoznačné
- ERROR("e") - chyba
- NOT_ENTITY("O") - není pojmenovaná entita

4.1.5 ParamArgs

Třída poskytující parametrizační algoritmy BoW a TF-IDF, a k nim pomocné metody tfIdfFindWords(), idfCompute(), idCompute().

4.1.6 Quality

Quality slouží pro výpočet kvality metod, konkrétně používá metriku přesnost (precision).

4.1.7 Sentence

Sentence reprezentuje větu. Obsahuje list slov, ze kterých se skládá, dále atributy se skutečným a určeným typem pojmenované entity, a vektor.

4.1.8 Word

Reprezentuje slovo. Kromě Stringu se samotným slovem obsahuje atributy, které reprezentují počet výskytů v textu a větách, které jsou potřeba pro paramametrizační algoritmy.

5 Uživatelská dokumentace

Program se spustí příkazem *java -jar UIR_SP2019.jar <datový soubor><parametrizační algoritmus><detekční algoritmus>*.

Formát vstupního souboru se skládá z řádků ve formátu *<slovo>_ _ <Třída pojmenované entity>*, který definuje jedno slovo, vynechaný řádek znamená ukončení věty a začátek nové. *<slovo >* je jakýkoliv řetězec, reprezentuje slovo ve větě. Třída pojmenované entity může být *"O"* nebo *"<I/B>-<zkratky pojmenovaných entit viz 4.1.4>"*.

Jako parametrizační algoritmus lze zvolit "bow"- Bag of words, či "tfidf"- Term Frequency - Inverse Document Frequency.

Detekčním algoritmem může být "kmeans", nebo "onenn".

Program po spuštění zkontroluje vstupní argumenty, pokud nebyly zadány správně, vypíše jak správně program spustit a ukončí se. Pokud byly argumenty správné, zvolené algoritmy určí jednu z 8 tříd pojmenovaných, či nepojmenovaných entit z vstupního souboru. Poté vypíše přesnost určení a výsledek uloží do souboru ve formátu *<určená třída>-<skutečná třída>-<věta>* . Název výstupního souboru je ve formátu *results-<parametrizační alg.>-<detekční alg.>.txt*.

6 Závěr

Aplikaci jsem testoval na souboru *test.conll*, trénovací soubor pro K-NN byl *my-data.conll*. Kombinací algoritmů BoW a K-means jsem dosáhl přesnosti 0,242; TF-IDF a K-means přesnost 0,142; Bow a K-NN 0,065; TF-IDF a K-NN 0,242. Vzorec výpočtu přesnosti je $Acc = TP/(TP + FP)$, kde TP je správně určený pozitivní příklad a FP správně odhadnutý negativní případ. Výsledky algoritmů jsou uložené v souborech s názvem *results-<bow/tfidf>-<kmeans/onenn>.txt*.