Michal Belák
NPRG041 Programování v C++
MFF UK 2018

Dokumentácia zápočtového programu
Rozpoznávanie jazyka pomocou n-gramov

1. Špecifikácia

Cieľom bolo implementovať algoritmus na rozpoznávanie jazyka vstupného textu pracujúci na princípe počítania výskytu písmenných n-gramov založený na článku (Cavnar & Trenkle, 1996).

2. Používanie programu

2.1 Konzolové rozhranie

Program využíva konzolové rozhranie a rozoznáva nasledovné konzolové argumenty: `-a, --analyze <filepath>` - analyzuje súbor daný ako filepath. `-R, --refrank <dirpath>` - načíta referenčné jazyky z daného adresára ako poradie najčastejších n-gramov. Vstupné súbory sú chápané ako výsledky predchádzajúcej analýzy. `-r, --reftext <dirpath>` - načíta referenčné jazyky z daného adresára ako zdrojové texty. Vstupné súbory musia prejsť analýzou. Pri použití tohto argumentu je možné dodať ďalší argument -S <dirpath>, ktorý zabezpečí uloženie všetkých analyzovaných jazykov do adresára dirpath. `-m, --match` – text analyzovaný v argumente –a je porovnaný s referenčným jazykmi získanými argumentmi –r alebo –R. Program vypíše výsledky na jednotlivé riadky vo formáte <názov jazyka><TAB><skóre>.
`-w, --write` – vypíše na štandardný výstup výsledky analýzu textu, teda najčastejšie n-gramy nájdené v texte vo formáte jeden n-gram na riadok. `-W, --writeall` – vypíše na štandardný výstup všetky nájdené n-gramy a ich počty na samostatné riadky vo formáte `<n-gram><TAB><počet>`. `-s, --save <filepath>` - uloží výsledky analýzy do súboru filepath. Tento argument má ďalšie možnosti. Arguement `-c, --count <count>` určí, koľko maximálne najčastejších n-gramov bude uložených. Argument `-n, --name <langname>` určí, pod akým menom bude cieľový jazyk uložený (pre účely ďalšieho použitia ako referenčný jazyk).

2.2 Príklady použitia

`$>Zapoctak.exe -r "../RefTexts" -S "../RefRanks"`
Týmto príkazom boli generované referenčné poradia n-gramov (viď sekcia 6).

`$>Zapoctak.exe -R "../RefRanks" -m "../test.txt"`
Vypočíta podobnosť vzorky `../test.txt` so všetkými referenčnými jazykovými profilmi nájdenými v priečinku `../RefRanks`.

3. Vstupy a výstupy programu

Všetky textové vstupy a výstupy používajú a predpokladajú enkódovanie vo formáte UTF-8. Súbor obsahujúci referenčný jazyk môže ako prvý znak obsahovať #. Potom celé prvé *slovo* (súvislá postupnosť znakov po prvý whitespace) bude interpretované ako názov tohto jazyka. Inak bude názov použitý názov súboru. Vstupný text rozdeľujeme na slová, z ktorých následne odstraňujeme všetky nepísmenné znaky okrem apostrofov.

Výstupy programu závisia od použitých argumentov a sú popísané v sekcii 2 pri jednotlivých argumentoch.

4. Algoritmus

Algoritmus je len mierne pozmenený oproti pôvodnemu algoritmu (Cavnar & Trenkle, 1996).

4.1. Analýza vstupného textu

Referenčný text rozdelíme na jednotlivé slová (súvislé postupnosti nebielych znakov). Následne z týchto slov odstránime všetky nepísmenné znaky a všetky písmená znaky zmeníme na malé písmená. Z každého tokenu vytvoríme všetky možné písmenné n-gramy pre n = 1, 2, 3. Pre n = 1 sú to jednoducho všetky písmená slova. Pre n > 1 slovo ešte pred tvorením n-gramov rozšírime o znaky začiatku („^") a konca („$") slova. Potom z nich vytvoríme všetky možné substringy dĺžky n. Napríklad pre slovo `text` vytvoríme nasledovné bi-gramy (n = 2): `^t`, `te`, `ex`, `xt`, `t$` a tri-gramy (n = 3): `^te`, `tex`, `ext`, `xt$`.

Následne nájdeme v hashovacej tabuľke počítadlo pre každý vzniknutý n-gram a inkrementujeme ho. Tieto záznamy zoradíme zostupne podľa počtu výskytov a zoberieme určitý počet (v našom programe defaultne 300) najčastejších n-gramov. Toto poradie je *profil dokumentu.*

4.2 Zistenie jazyka na základe analýzy

Pre zistenie jazyka dokumentu v neznámom jazyku ho porovnáme s profilmi dokumentov v známych jazykoch (*referenčné profily*) a pre každý vypočítame *skóre*. Pre každý n-gram v referenčnom profile nájdeme jeho poradie v profile neznámeho jazyka a ku skóre pričítamé absolútnu hodnotu rozdielu ich poradí. V prípade, že sa daný n-gram v profile neznámeho jazyka nenachádza, pripočítame k skóre konštantu *K*.

Príklad:

Profil jazyka L: [a, e, s, i, de]
Profil neznámeho jazyka: [e, o, a, i, u]

Potom skóre bude 2 + 1 + K + 0 + K.

Tieto skóre nasledovne zoradíme vzostupne a ako našu predikciu jazyka neznámeho dokumentu zoberieme ten referenčný jazyk, pri porovnaní s ktorým sme dostali najnižšie skóre.

5. Implementácia

V našom programe používame pre uchovávanie všetkých textových premenných typ `std::wstring` (so znakmi typu `wchar_t`), pretože (aspoň vo Visual C++) sľubuje lepšiu podporu non-ASCII znakov oproti bežnejšiemu `std::string`.

Analýzu vstupných textov má na starosti trieda `TextAnalyzer`. Dostáva objekt typu `std::wistream` (prostredníctvom konštrukturu alebo metódy `set_input()`). V metóde `analyze()` číta z daného vstupu a analyzuje ho tak, ako je popísané v časti 4.1. V tejto časti tiež prečítame zo vstupného textu jeho jazyk, ak je nájdený (viď časť 3). Návratová hodnota metódy `analyze()` je objekt typu `LangProfile`.

Trieda `LangProfile` uchováva hashovaciu tabuľku (typ `std::unordered_map<std::wstring, size_t>`), kde kľúčom je n-gram a hodnotou je

počet jeho výskytov v analyzovanom texte. Preťažená metóda `update_profile()` prijíma v rôznych podobách ďalšie n-gramy, ktoré sú následne zahrnuté v sledovaných počtoch. Metóda `get_ranking()` vracia objekt typu `NgramRanking` obsahujúci poradie najčastejších n-gramov. Berie argument typu `size_t` (s defaultnou hodnotou 300), ktorý určuje, koľko najviac najčastejších n-gramov požadujeme.

Trieda `NgramRanking` uchováva pole n-gramov (typ `std::vector<std::wstring>`), ktoré boli pri konštrukcii objektu zoradené zostupne podľa početnosti ich výskytu v analyzovanom texte. Metóda `get_rank(const std::wstring & ngram)` pre daný n-gram `ngram` nájde a vráti jeho poradie (rank) medzi uloženými n-gramami. Pokiaľ daný n-gram nebol nájdený vracia číslo *K* (viď 4.1), ktorá je v našej implementácií definované ako 1.2-násobok počtu všetkých uložených n-gramov.

Trieda `ProfileMatcher` vlastní zoznam referenčných jazyky (typ `std::vector<NgramRanking>`) a stará sa o porovnávanie jazykov a určovanie najpodobnejšieho jazyka k danému jazyku. Túto funkčnosť zabezpečuje hlavná metóda `match_prof(const NgramRanking &)`, ktorá vráti zoznam mien jazykov a ich príslušných skóre (typ `std::vector<std::wstring, size_t>`) zoradený od najmenšieho skóre.

6. Testovacie dáta

Testovacie dáta obsahujú texty, ktoré sú dobre použiteľné pre naučenie referenčných jazykov. Ide o Všeobecnú deklaráciu ľudských práv (United Nations Human Rights | Office of the High Commissioner, 2018) v 27 rôznych jazykových verziách. Tieto texty sú priložené v 2 podobách: v zložke RefTexts nájdeme tieto texty v surovej podobe a v zložke RefLangs nájdeme našim programom predspracované poradie 300 najčastejších n-gramov z každého z týchto referenčných textov. Pre vyskúšanie tohto programu je možné použiť akýkoľvek text, je však z implementačných dôvodov nutné, aby bol uložený vo formáte UTF-8.

Zložka `Test` obsahuje v niekoľkých jazykoch úvodnú sekciu článku Wikipedie o krajine, z ktorej daný jazyk pochádza.

7. Záver

Výsledný program funguje možno až prekvapivo dobre vzhľadom na jednoduchosť algoritmu. Takmer vždy odhadne správny jazyk a pri testovaní som nenašiel nájsť dlhší text (cca >50 slov), pre ktoré by nebol správny jazyk aspoň medzi prvými tromi vybranými kanditátmi. Presnosť programu by mohla byť zvýšená zväčšením objemu textov, z ktorých sú generované referenčné profily.

Rýchlosť behu sa taktiež javí ako slušná, pri čom vytvorenie profilu pre 27 referenčných jazykov (veľkosti textu cca 10-20 kB v UTF-8) a 1 neznámeho textu spolu s následným zistením najpravdepodobnejšieho jazyka trvá v Release režime prekladu len zlomok sekundy.

8 .Referencie

Cavnar, W. B., & Trenkle, J. M. (1996). N-Gram-Based Text Categorization. *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, (pp. 161--175).

United Nations Human Rights | Office of the High Commissioner. (2018, 03 28). *OHCHR | Search by Translation*. Retrieved from OHCHR: http://www.ohchr.org/EN/UDHR/Pages/SearchByLang.aspx