1 Témátko č. 1: Konec zdlouhavému psaní zpráv?

1.1 Získání dat uživatele

První krok k personalizovanému nabízení slov je zpracování dat uživatele. Data jsou v našem případě věty a slova, které uživatel napsal. Kde však seženeme dostatečně velké množství gramaticky správných dat, aby se systém mohl učit?

Jedna možnost je vytvořit systém tak, aby zpracovával data okamžitě po jejich napsání a učil se tak "za běhu". Variací tohoto systému existuje v dnešním světě řada, primárně na mobilních zařízeních.

Nevýhoda však vězí v tom, že zpočátku se jedná o systém velice hloupý a až při jeho dlouhodobějším používání se stává použitelný. Pokud tedy chceme vybudovat systém tak, aby byl použitelný okamžitě, musíme nejdříve získat uživatelem napsané věty.

V dnešní době drtivá většina z nás používá nějakou formu sociálních médií, což je pro náš případ velice vhodné. Naštěstí pro nás je získání těchto dat z většiny populárnějších služeb přímočaré:

- **Facebook:** Po přihlášení na Facebook je v Nastavení pod záložkou Vaše informace na facebooku možnost Stažení vašich informací.
- **Gmail:** Stačí navštívit Google Takeout a z produktů ke stažení vybrat Gmail.
- **Instagram:** V nastavení v sekci Soukromí a Bezpečnost si lze data vyžádat po kliknutí na podsekci Stažení dat.

1.2 Zpracování dat z Facebooku

V mém případě je nejlepší volba Facebook, jelikož službu Messenger již několik let aktivně používám.

Ve složce *dat* je několik scriptů na zpracování dat z Facebooku. Po umístění stažených zpráv z Facebooku do této složky pod jménem *messages* dělají scripty na zpracování dat následující:

- *clean_facebook_data.py*: odstraní nepotřebné soubory (fotky, videa...) a prázdné adresáře z adresáře *dat/messages* a všech jeho podadresářů.
- *generate_messages.py*: vygeneruje zprávy daného uživatele v určeném časovém intervalu ze všech JSON souborů adresáře *dat/messages* a všech jeho podadresářů.

I přes to, že máme zprávy stažené může nastat problém gramatických chyb, díky kterým budou naše data vadná. Tento problém řeší aplikace *correct_messages.py*, ve které uživatel opravuje slova s gramatickými chybami vygenerovaného souboru zpráv. Aplikace rovněž odstraňuje nečeská slova a věty, URL, emailové adresy a další nechtěné části textu.

setup.py pouze po pořadě spouští všechny ostatní scripty.

1.3 Metody nabízení textu

1.3.1 Doplňování pomocí trie

Jednou z možností, jak slova uživateli nabízet je pomocí trie slov (postavenou z každého slova z uživatelových dat).

Po získání posledního slova[1] textu, který chceme upravovat přetraverzujeme trii po písmenech tohoto slova. Po přetraverzování získáme všechna slova tvořená zbytkem trie. Tato slova setřídíme podle četnosti jejich výskytu v trii a několik nejčetnějších nabídneme uživateli.

Metoda je implementována funkcí *getPredictionsFromTrie()*.

1.3.2 Opravy z možných úprav

Pokud metoda doplňování pomocí trie nebyla aplikovatelná, protože se slovo v trii nevyskytovalo, další možná metoda je generování nejbližších obdob slova.

Má inspirace pro tuto metodu pochází ze skvělého článku *How to Write a Spelling Corrector*, který tuto metodu vysvětluje.

Princip je v tom, že se vygenerují všechna slova, která se liší od námi upravovaného slova několika[2] úpravami. Poté odebereme ta slova, která nejsou v námi vybudované trii a několik v trii nejčetnějších nabídneme uživateli.

Metoda je implementována funkcí *getWordCorrections()*.

1.3.3 Doporučení dalšího slova

Pokud je slovo v trii a potřebujeme doporučit další po něm jdoucí, ani jedna z výše uvedených metod není aplikovatelná.

Trii proto zkonstuujeme tak, abychom kromě slov v textu přidávali do trie i slova, která na tato slova ve větách navazují. Uživateli poté doporučíme několik nejčetnějších slov, která na naše slovo bezprostředně navazují.

Dalším možným vylepšením by bylo "rozšířit kontext" a brát i slova, která bezprostředně navazují na n-tice slov. Náš model by byl přesnější na úkor rychlosti, paměti zabírané trií a množství potřebných dat.

Metoda je implementována v rámci funkce *textboxChanged()*.

[1] Např. regulárním výrazem.

[2] Pro náš případ právě jednu, pro více by byl náš program pomalý.

1.4 Závěr

V repozitáři je přiložen soubor *sample_messages.txt*, který je připraven ke zpracování výše popsaným modelem. Po spuštění aplikace *autocomplete.py* stačí do horního textboxu zadat relativní (nebo absolutní) umístění tohoto souboru, stisknout Generate a poté psát do spodního textboxu věty, které má model doplňovat/upravovat.