Uživatelská dokumentace LANSHER

Vilém Zouhar

2017

1 Úvod

LANSHER (LANguage distinguiSHER) slouží k rozlišení jazyka na základě velkého množství trénovacích dat. Používá se ve dvou krocích:

1. Vytvoření databáze z JSON formátovaného souboru
2. Použití databáze k rozeznání jazyka vstupu

Přičemž není nutné pro každé použití absolvovat oba dva kroky. Zejména krok 1., výpočetně náročný, je možné provést jen jednou a používat tak vytvořenou databázi na rozeznávání jazyka. Je také možné použít již zpracovanou databázi dodávanou s programem.

2 Vytváření databáze

Databáze se vytváří z JSON souboru v daném formátu: (soubor *alenka.json*)

```
{
"cs":
        "Alenka ani chvíli nemeškala a vskočila za ním, aniž jen zdaleka
        pomyslila, jak se kdy opět dostane ven.",
"en":
        "In another moment down went Alice after it, never once considering
        how in the world she was to get out again.",
"de":
        "Den nächsten Augenblick war sie ihm nach in das Loch
        hineingesprungen, ohne zu bedenken, wie in aller Welt sie wieder
        herauskommen könnte.",
"pl":
        "Wczołgała się więc za nim do króliczej nory nie myśląc o tym, jak
        się póóniej stamtąd wydostanie."
}
```

Není třeba definovat všechny jazyky, a ani není nutné dodržovat dvouznakové značení. Pro přehlednost se v tomto dokumentu budeme dvou znaků držet. JSON vstup se do programu vkládá pomocí přepínače **-ls**, nebo **--language-samples**.

3 Ukládání databáze

Pokud do programu vkládáme text ke zpracování, což je obecně náročná výpočetní operace, můžeme si takto zpracovanou databázi uložit za pomocí přepínače **-sd**, nebo **--save-database**.

Pokud třeba načítáme soubor *alenka.json* a chceme zpracovanou verzi uložit do souboru *alenka.ld*, použijeme přepínače **-ls** a **-sd**:

```
./lansher.py -ls alenka.json -sd alenka.ld
```

4 Načítání databáze

Je-li k dispozici již vytvořená databáze, do programu ji načteme pomocí **-ld**, nebo **--load-database**. V příkladu:

```
./lansher.py -ld alenka.ld
```

Tento příkaz však sám o sobě nemá žádný výstup, neboť mu nebyla předán žádný přepínač pro výstupní operace.

5 Slučování databáze

Jelikož existují dvě možnosti jak vytvářet databázi, je možné je zkombinovat a vytvořit tak novou, která je sloučením obou dvou. Toho docílíme načtením již vytvořené databáze, zpracováním nového úseku textu (*kralik.json*) a uložením do souboru (*alenka_ kralik.ld*):

```
./lansher.py -ls kralik.json -ld alenka.ld -sd alenka_kralik.ld
```

6 Vstup vzorku

Má-li program načtenou databázi (např. z předem zpracovaného textu), můžeme mu vložit vzorek k rozeznání.

Pokud chceme předat vzorek z příkazové řádky, použijeme přepínač **-i**, nebo **--input**:

```
./lansher.py --load-database alenka.ld --input "Alenka v říši divů"
```

Výstup:

```
input text comparison:
cs (96.988%)
en (1.045%)
pl (1.023%)
de (0.944%)
```

Pokud chceme předat vzorek ze souboru, použijeme přepínač **-fi**, nebo **--file-input**. *vzorek_ textu.txt* obsahuje řetězec: *Sie könnte nicht mehr herauskommen.*

```
./lansher.py --load-database alenka.ld --file-input vzorek_textu.txt
```

Výstup:

```
input text comparison:
de (98.662%)
en (0.474%)
pl (0.443%)
cs (0.421%)
```

Databáze je v našem případě velmi malá. Pro přesnější rozlišování je třeba mít rozsah v rámci několika knih či slovníků.

7 Srovnání jazyků

Má-li program načtenou databázi, můžeme porovnat jazyky mezi sebou za pomocí přepínače **-lc**, nebo **--language-comparison**, což pro každý jazyk vypíše nejbližší jiný:

```
./lansher.py --load-database alenka.ld --language-comparison
```

Výstup:

```
cs: pl (97.92%)
de: en (97.33%)
pl: cs (97.66%)
en: de (96.97%)
```

8 Přehled příkazů

-i, **--input**	vzorek z příkazové řádky
-fi, **--file-input**	vzorek ze souboru
-ls, **--language-samples**	vytvoření databáze
-ld, **--load-database**	načtení připravené databáze
-sd, **--save-database**	uložení načtené databáze
-v, **--version**	aktuální verze programu
-h, **--help**	nápověda