Risanje sinusoid

David Rubin (david.rubin@student.um.si)

6. junij 2019

1 Uvod

Implementirajte program za popravljanje pravopisnih napak. Program naj vsebuje naslednje korake:

- Ustvarite slovar, ki za vsako besedo vsebuje frekvenco pojavitev v učni množici. Vse besede so predstavljene z malimi črkami.
- Napišite funkcijo, ki za podano besedo vrne vse besede na razdalji urejanja 1. Operacije urejanja so brisanje, transpozicija, zamenjava in vstavljanje.
- Napišite funkcijo, ki za podano besedo vrne vse besede na razdalji urejanja 2. Pri tem uporabite funkcijo za vračanje besed, ki so na razdalji 1.
- Napišite funkcijo, ki za podano besedo vrne vse besede iz slovarja na razdalji urejanja 2.
- Napišite funkcijo, ki za podano besedo vrne seznam kandidatk in njihovih verjetnosti urejanja. Uporabimo poenostavljen model, ki uporablja naslednja pravila:
  - Če je podana beseda že v slovarju, je ta beseda prva in zadnja v seznamu kandidatk. Njena verjetnost urejanja je 1.
  - Če v slovarju ni podana besede, v seznam kandidatk dodamo besede, ki so v slovarju na razdalji urejanja 1. Njihove verjetnosti urejanja se normalizirajo glede na frekvence v slovarju in vsoto vseh frekvenc kandidatk.
  - Če v slovarju ni podane besede in kandidatk na razdalji urejanja 1, v seznam kandidatk dodamo besede, ki so v slovarju na razdalji urejanja 2. Njihove verjetnosti urejanja se normalizirajo glede na frekvence v slovarju in vsoto vseh frekvenc kandidatk.
  - Če ni izpolnjen nobeden od predhodnih pogojev, je kandidatka podana beseda z verjetnostjo urejanja 1.
- Napišite funkcijo, ki bo z uporabo jezikovnega modela, ki ste ga implementirali na eni od prejšnjih vaj, popravila podano poved s pomočjo naslednjih pravil:
  - Izračunamo verjetnost povedi brez uporabe kandidatk.

– Izračunamo verjetnost povedi, pri kateri uporabimo le eno od kandidatk. Ko računamo verjetnost n-grama, verjetnost množimo z verjetnostjo urejanja kandidatke.
– Za kandidatke, ki niso v slovarju uporabimo verjetnost 1/V.
– Rezultat je poved, ki ima maksimalno verjetnost. Tukaj si pomagamo z logaritmi.

2 Poročilo

Rešitev sem implementiral kot Python program (za delovanje potrebuje kneser_ney iz prejšnje naloge), ki je priložen poročilu (*spellcheck.py*). V sklopu ocenjevanja, sem uporabil 100 povedi iz priloženega *holbrook.dat*, kjer se pojavi 1 napaka. Po zagonu programa se generirajo datoteke, ki vsebujejo dejanske povedi (razbrane iz podanega korpusa), povedi popravljenje s pomočjo Kneser-Ney glajenja in povedi, kjer se za besedo kjer je definirana napaka uporabi tista z največjo verjetnostjo urejanja. Rezultati so sledeči:

Metoda	Uspešnost	Št. zadetih besed
s KN	0.267	27/101
verjetnost urejanja	0.198	20/101

V tabeli rezultatov je KN metoda, kjer verjetnost povedi ocenjujemo s pomočjo Kneser-Ney, pri verjetnosti urejanja pa le določimo, katera beseda je najverjetnejša za podano napako. Uspešnost meri koliko napak je uspešno popravila metoda, Število razlik pa pove, koliko besed se loči od pravilne povedi (torej lahko se zgodi, da je pri Kneser-Ney ocenila, da so še druge besede pri pravilno zapisanih verjetnejše).