Asdos PJ: Muhammad Faisal Adi Soesatyo
CSCE604231: PENGOLAHAN BAHASA MANUSIA (NLP)
September 3, 2023

Tugas Individu 2 (TI 2)
Eksperimen Pengembangan N-gram *Language Model*
Deadline Jumat, 6 Oktober 2023 jam 22:00

Pada Tugas Individu 2 ini, peserta diminta untuk melakukan eksperimen pengembangan N-gram *Language Model* serta mengukur kualitas model yang dibangun menggunakan metrik *perplexity*.

1. Apa yang Perlu Dipahami?

Untuk menyelesaikan tugas ini, peserta diharapkan telah memahami konsep dari n-gram *language model* serta metrik evaluasi *perplexity*.

1.1. Apa itu N-gram Language Model.

Language Model (LM) merupakan model yang memberikan nilai probabilitas pada urutan kata. Salah satu model bahasa yang paling sederhana adalah **n-gram**. Suatu *n-gram* adalah sebuah urutan yang terdiri dari n kata. Misal, suatu model bahasa 2-gram atau dikenal dengan bigram merupakan suatu urutan dua kata dari kumpulan kata.

Esensi dari *N-gram Language Model* adalah menghitung probabilitas kemunculan suatu urutan kata dalam sebuah kalimat. Untuk menghitung probabilitas tersebut memanfaatkan aturan rantai (*The chain rule*) yang kemudian disederhanakan menggunakan *Markov Assumption*. Dengan mengasumsikan kehadiran antar kata di dalam suatu kalimat sebagai komponen independen maka muncul beragam jenis *n-gram model*, seperti 1-gram (*unigram*), 2-gram (*bigram*), 3-gram (*trigram*), etc.

1.2. Menghitung Probabilitas Menggunakan N-gram.

Salah satu pendekatan yang dilakukan adalah menggunakan ***Maximum Likelihood Estimate*** (MLE).

Contoh perhitungan yang digunakan adalah menggunakan 2-gram (*bigram*), di mana suatu probabilitas $P(w_i|w_{i-1})$ dapat diaproksimasi sebagai berikut,

$$P_{MLE}(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

di mana,

- count(w_{i-1}, w_i): jumlah kemunculan pasangan kata w_{i-1} dan w_i secara berurut di sebuah kalimat.
- count(w_{i-1}): jumlah kemunculan kata (w_{i-1}) di sebuah kalimat.

tetapi, apa yang akan terjadi apabila count(w_{i-1}, w_i) bernilai 0? Tentu probabilitas dari pasangan kata w_{i-1} dan w_i akan bernilai 0, yang berakibat pada nilai 0 untuk keseluruhan probabilitas kalimat tersebut. Untuk menangani isu tersebut, dilakukanlah tahap *smoothing*, bernama *add-one (laplace) smoothing*.

Laplace smoothing diaproksimasi sebagai berikut,

$$P_{laplace}(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i) + 1}{count(w_{i-1}) + |V|} \tag{1}$$

di mana,

- $|V|$: jumlah kosakata unik yang terdapat di dalam corpus

2. Apa yang Harus Dikerjakan?

Data yang digunakan pada tugas individu ini berasal dari https://dumps.wikimedia.org/idwiki/latest/idwiki-latest-pages-articles.xml.bz2. Konten dataset tersebut merupakan potongan kalimat dari artikel yang terdapat pada laman wikimedia.

Setelah memahami informasi di atas, berikut adalah beberapa hal yang perlu dilakukan.

1) *Clone*/unduh repository GitHub ini yang berisi kode program dari eksperimen pengembangan *n-gram language model* ini.
2) Membaca file `README.md` untuk mendapatkan penjelasan yang lebih *straightforward*.

3) Lengkapi *method* yang diberikan komentar `TODO` pada kode program tersebut.
4) Untuk menjalankan keseluruhan program, Anda cukup menjalankan file `main.py`.
5) Di akhir masing-masing python file kecuali `main.py` terdapat *conditional statement* yang dapat Anda *uncomment* apabila ingin *debug* pada file tersebut.
6) Perlu diperhatikan bahwa proses pembangunan corpus pada file `build_corpus.py` memakan waktu yang cukup lama, yakni sekitar 7-10 menit tergantung RAM serta kekuatan processor Anda. Maka dari itu, command `build_corpus.main()` pada file `main.py` cukup dijalankan sekali, kemudian Anda *comment* atau hapus.
7) Jumlah keseluruhan kalimat pada corpus sebesar 490,000 kalimat dengan rasio train:dev:test sebesar 8:1:1. Apabila untuk memproses sebanyak 490,000 kalimat dinilai cukup berat, Anda dipersilakan untuk mengatur jumlah kalimat serta rasio tran:dev:test yang disesuaikan dengan kemampuan mesin Anda.
8) Lakukan eksperimen serta eksplorasi sebebas Anda, asalkan model dapat berjalan dan memberikan nilai *perplexity* yang menurut Anda optimal.

3. Objektif TI 2

1) Lakukan eksplorasi pada pengaturan ukuran dataset, *data preprocessing*, kombinasi n-gram model, pengaturan *threshold* token, dsb hingga mencapai nilai *perplexity* serendah atau se-optimum mungkin.

4. Luaran TI 2

Kode program hasil pengembangan dapat Anda unggah di github masing-masing dan pastikan *visibility* dari *repository* tersebut sudah *private*. *Invite* github `ialfina` serta `faisaladisoe` sebagai collaborator. Mengingat ukuran dataset yang cukup besar, maka dari itu, dataset **tidak perlu** Anda *upload* ke github.

Anda diharapkan membuat laporan hasil eksperimen pengembangan n-gram language model yang berisi informasi seperti,

1) Tahap *data preprocessing* yang dilakukan termasuk pemilihan jumlah kalimat serta konfigurasi train:dev:test.
2) Implementasi logic n-gram model serta perhitungan *perplexity*.
3) Nilai *perplexity* dari model Anda serta usaha yang dilakukan hingga mendapatkan nilai perplexity tersebut.
4) Link github yang berisi program hasil eksperimen Anda.

5. Pengumpulan Tugas

File yang dikumpulkan:

1) *File* laporan dalam format PDF dengan aturan nama `NLP-TI2-[NamaMahasiswa].pdf`. Contoh: `NLP-TI2-CommanderKowalski.pdf`.