COMP6714 Assignment 1

Yao Yuan z5092195

Q1.

(1).

According to the question, word appears at least once in each of all N documents.

Let x = number of bits a word requires for its postings.

$$x = N \cdot \log_2 N \cdot \text{(bits)}$$

Because each posting requires $\log_2 N$ bits to represent.

And there are N docs in total.

By applying Elias-γ encoding.

Assume the id of one doc is k, $k \in [1, N]$

Therefore, the number of bits required for this doc is $2\lfloor \log_2 k \rfloor + 1$

Since there are N docs, let y_γ be the number of bits for this word by using Elias-γ encoding.

$$y_\gamma = \sum_{k=1}^{N} \left(2\lfloor \log_2 k \rfloor + 1\right)$$

By applying Elias-δ encoding.

Assume the id of one doc is k, $k \in [1, N]$

Therefore, the number of bits required for this doc is $2\log_2 \log_2 k + \log_2 k$

Since there are N docs, let y_δ be the number of bits for this word by using Elias-δ encoding.

$$y_\delta = \sum_{k=1}^{N} \left(2\log_2 \log_2 k + \log_2 k\right)$$

Since, compression ratio $= \frac{\text{Uncompressed size}}{\text{compressed size}}$

$$\text{Elias-}\gamma \text{ compression ratio} = \frac{x}{y_\gamma} = \frac{N\log_2 N}{\sum_{k=1}^{N}(2\lfloor\log_2 k\rfloor+1)}$$

$$\text{Elias-}\delta \text{ compression ratio} = \frac{x}{y_\delta} = \frac{N\log_2 N}{\sum_{k=1}^{N}(2\log_2\log_2 k + \log_2 k)}$$

(2). According to the question, we are using gaps and term frequency in our postings; and term frequency for each doc is known to be 10.

Let x = number of bits a word requires for its postings.

$$x = \log_2 N + \left(\frac{N}{5}-1\right)\log_2 5 + \frac{N}{5}\cdot\log_2 10. = \log_2 N + N - 2$$

Because each posting requires its "Head", which takes $\log_2 N$ bits.
the following gap is $\log_2 5$ and there are $\left(\frac{N}{5}-1\right)$.
And there are term frequency for every docID.
So bits for term frequency are $\frac{N}{5}\cdot\log_2 10$.

By applying Elias-γ encoding.
Assume the id of the first doc is k, $k \in [1, 5]$
Therefore, the number of bits required for this doc is $2\lfloor\log_2 k\rfloor+1$
Since we are using gaps, let y_γ be the number of bits for this word by using Elias-γ encoding.

$$y_\gamma = 2\lfloor\log_2 k\rfloor+1 + \left(2\lfloor\log_2 5\rfloor+1\right)\cdot\left(\frac{N}{5}-1\right) + \frac{N}{5}\cdot\left(2\lfloor\log_2 10\rfloor+1\right)$$
$$= 2\lfloor\log_2 k\rfloor + \frac{12}{5}N - 4$$

where $2\lfloor \log_2 5 \rfloor + 1$ is the number of bits required for each gap

$2\lfloor \log_2 10 \rfloor + 1$ is the number of bies for each term frequency

By applying Elias-δ encoding.

Assume the id of the first doc is k, $k \in [1, 5]$

Therefore, the number of bits required for this doc is $(2\log_2\log_2 k + \log_2 k)$

Since we are using gaps, let y_δ be the number of bits for this word by using Elias-δ encodin.

← gaps → | ← term freq → |

$$y_\delta = \underbrace{\left(2\log_2\log_2 k + \log_2 k\right)}_{\text{Head}} + \left(\frac{N}{5} - 1\right)\cdot\left(2\log_2\log_2 5 + \log_2 5\right) + \frac{N}{5}\cdot\left(2\log_2\log_2 10 + \log_2 10\right)$$

$$= 2\log_2\log_2 k + \log_2 k + 2N - 4$$

Since, compression ratio $= \dfrac{\text{uncompressed size}}{\text{compressed size}}$

$$\text{Elias-}\gamma \text{ compression ratio} = \frac{x}{y_\gamma} \approx \frac{\log_2 N + N - 2}{2\lfloor \log_2 k \rfloor + \frac{12}{5}N - 4}$$

$$\text{Elias-}\delta \text{ compression ratio} = \frac{x}{y_\delta} \approx \frac{\log_2 N + N - 2}{2\log_2\log_2 k + \log_2 k + 2N - 4}$$

Q2.

(1). Assume after all logarithmic merges have finished, there will be n number of sub-indexes formed in disk.

Therefore, we can have $2^n = t$.

Because for each merge operation, we merge 2 into 1.

And there are t sub-indexes in total.

Since $2^n = t$,

$n = \lceil \log_2 t \rceil$ for n to be an integer.

(2). From the proof of (1), we know that for t sub-indexes, there will be at most $\log_2 t$ sub-indexes in disk.

Therefore, for each sub-index in t, it will be called at most $\log_2 t$ times for merge operation.

Since for each sub-index, it has M pages. The read/write cost for it would be M. for each merge operation.

There are t sub-indexes in total.

Therefore the I/O cost for the whole merge operation will be bounded by $2 \cdot t \cdot M \cdot \log_2 t$ for read and write.

∴ the total I/O cost of the logarithmic merge is $O(t \cdot M \cdot \log_2 t)$.

Q3.

(1). The formula for recall is $\frac{TP}{TP + FN}$.

By using stemming, the original TP will not reduce because the stemming results must include the original correct words.

The original FN may reduce because we may include actual positives For example.

When the query term is "automate". and when using Boolean query retrival, docs having "automates" will not be considered; But it should consider.

By applying stemming, both "automate" and "automates" are "automa". Therefore, "automates" docs are considered.

Therefore FN decreases.

Therefore, we can conclude that stemming will not hurt recall.

(2). The formula for F_1 is

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

We know that when FN decrease by x, TP would increse by x.
when FP increase by y, TN would decrease by y.

From (1), we know that stemming will always derease FN by an amount. Assume it is x, and $x \geq 0$.

And we know that stemming will always increase FP by an amount. Assume it is y, and $y \geq 0$.

$$\therefore \text{new } F_1 = \frac{TP + x}{TP + x + \frac{1}{2}(FP + y + FN - x)} = \frac{TP + x}{TP + \frac{1}{2}(FP + FN) + \frac{1}{2}(x + y)}$$

Let $TP = a$, $TP + \frac{1}{2}(FP + FN) = b$, $b \geq a$

$\therefore F_1 = \frac{a}{b}$, new $F_1 = \frac{a+x}{b+\frac{1}{2}(x+y)}$

If $F_1 = \text{new } F_1$,

Then. $\frac{a}{b} = \frac{a+x}{b+\frac{1}{2}(x+y)}$

$$\frac{a}{2}(x+y) = bx$$

$$ax + ay = 2bx$$

$$ay = (2b-a)x$$

$$\frac{y}{x} = \frac{2b-a}{a}$$

Therefore. when $\frac{y}{x} > \frac{2b-a}{a}$, F_1 will decrease after stemming

when $\frac{y}{x} < \frac{2b-a}{a}$, F_1 will increase after stemming.

Intuitively, when the stemming word includes a lot of words that are not "same" as query word, F_1 score will be dominant by increasing in FP and will hurt F_1 score.
In opposite, if the stemming word include a lot of words that are "same" as query word, F_1 score will be dominant by decreasing in FN and will help F_1 score.

In conclusion, it is wrong that stemming always helps or hurts the F_1 score.