Esercitazione 3

L'obiettivo della terza esercitazione è implementare un sistema di riassunto automatico, e applicarlo su 5 articoli provando diversi approcci per il rank dei paragrafi e diverse compressioni. I risultati vengono confrontati con i riassunti di riferimento fatti manualmente calcolando due misure per le performance per "precision e recall".

utlis

La cartella utils contiene i file necessari all'esecuzione ed è strutturata come segue

- docs: contiene i 5 file .txt dei testi da riassumere forniti a lezione
- NASARI_vectors: contiene il file dd-small-nasari-15.txt contenente 13.084 vettori con 15 feature ciascuno.
- reference_summary: contiene 5 file di testo, ossia i riassunti fatti manualmente come elenco di parole importanti dell'articolo
- riassunti: cartella contenente i file riassunti, output del programma.
- babelnet.txt: dato che babelnet limita gli accessi a 1000 al giorno per ogni chiave, ci salviamo esempi e definizione di ogni babelnet synset per evitare di fare richieste ogni esecuzione.Questo file è necessario per disambiguare i termini durante l'esecuzione. Il file è stato scritto dal programma java in allegato.
- cue_words.txt: elenco di cue words utilizzate per l'implementazione del secondo metodo di rank dei paragrafi, descritto in seguito
- stop_words.txt: elenco di stop words utilizzato durante il processo di disambiguazione delle parole.

riassunto.py

file da eseguire per la computazione e la creazione dei file di output. Vengono inoltre stampati a terminale i valori delle misure delle performance.

I metodi per il rank dei paragrafi implementati sono due

- title: in questo approccio prendiamo semplicemente la prima riga significativa dei documenti, ossia il titolo. per ogni parola (dopo lemmatizzazione e disambiguazione) prendiamo il vettore di nasari corrispondente. Visto che sono poche parole, per ogni termine di ogni vettore prendiamo anche il relativo vettore.
- cue words: il contesto è creato prendendo i paragrafi del documento che contengono cue words presenti nel file di testo. Per ognuno dei termini dei paragrafi prendiamo il vettore corrispondente.

I metodi "create_context_titolo" e "create_context" si occupano di creare quindi il contesto. Dopo aver creato il contesto facciamo una classifica (ranking) dei paragrafi calcolando il Weighted Overlap del vettore del contesto e del vettore del paragrafo. Il metodo "unify_vet" si occupa di unire la lista di vettori in uno singolo, e il metodo " weigthed_overlap" si occupa del calcolo vero è proprio.

$$WO(v_1, v_2) = \frac{\sum_{q \in O} \left(rank(q, v_1) + rank(q, v_2)\right)^{-1}}{\sum_{i=1}^{|O|} (2i)^{-1}}$$

A questo punto abbiamo la classifica dei paragrafi, ordinata in base al WO.

Quindi scriviamo nei file di riassunto tre diversi riassunti con compressione del 10, 20 e 30%.

Infine calcoliamo le misure di performance "blue" e "rogue" per ogni articolo (e le 3 compressioni) e stampiamo nel terminale i risultati.

Nota: disambiguazione
Il file mini nasari contiene diversi vettori ambigui, per cui prima di scegliere un vettore per un termine, è necessario prima disambiguare. Questo passo è fatto in modo simile alla prima esercitazione con un metodo bag-of-words, utilizzando come contesto i termini dell'articolo e come signature dei sensi esempi e definizione estratti da babelnet e salvati precedentemente nel file "babelnet.txt".

Conclusioni
Il tempo di esecuzione è quasi istantaneo per ogni articolo e un esempio di risultato per l'articolo "Andy-Warhol" è il seguente

metodo di rank	compressione	blue	rogue
cue words	10%	0.05454545454545454	0.8076923076923077
cue words	20%	0.05847953216374269	0.7692307692307693
cue words	30%	0.05592105263157895	0.6538461538461539
title	10%	0.05454545454545454	0.8076923076923077
title	20%	0.06051873198847262	0.8076923076923077
title	30%	0.06557377049180328	0.7692307692307693

Come ci aspettavamo la misura blue è molto bassa, questo perché i riassunti sono semplici liste di parole, quindi la precision è giustamente pessima, ma possiamo vedere che per il metodo del titolo aumenta sempre, mentre per il metodo delle cue word no.
Dall'altra parte rogue è più significativa e riguarda la recall. Con il metodo delle cue word vediamo che però diminuisce sempre rispetto alla compressione, mentre per il metodo del titolo vediamo che è leggermente migliore, e nel passaggio da 10 a 20 rimane uguale.
Quindi per questo articolo possiamo concludere che il metodo del titolo è migliore rispetto alle cue words.
Per gli altri articoli i risultati sono variabili ma molto vicini tra loro. Possiamo comunque concludere che il metodo del titolo è leggermente meglio nei nostri esempi, ma probabilmente dipende semplicemente dai singoli articoli e non vi è uno migliore in assoluto.