Relazione
Esercitazione 2

Summarizzation with NASARI vectors

Introduzione

L'obbiettivo di questa esercitazione è stato di implementare un algoritmo di automatic summarization usando la risorsa vettoriale NASARI.

L'algoritmo riceve in input una lista di documenti di testo

[*'Andy-Warhol.txt', 'Ebola-virus-disease.txt', 'Life-indoors.txt', 'Napoleon-wiki.txt', 'Trump-wall.txt'*]

e produce, per ogni documento, un riassunto estrattivo del documento originale.

Nello specifico, nel testo fornito in output (ovvero nel riassunto), troveremo un sottoinsieme dei paragrafi presenti nel testo iniziale. Questa riduzione è più o meno elevata a seconda di determinati tassi di compressione: 10, 20, e 30% i quali esprimono il rapporto tra la lunghezza dei documenti.

Uno dei punti più importanti per l'implementazione dell'algoritmo riguarda la scelta delle metodologie e di criteri necessari per l'individuazione di un topic, ovvero la porzione del testo del documento originale nella quale risiedono le frasi e i termini più rilevanti, perciò una volta determinato, è possibile selezionare le porzioni del testo originale che ne hanno una maggiore affinità per poi restituirle in output come sintesi del documento.

In questa esercitazione sono stati scelti, come criteri per l'individuazione del topic, due approcci

- il metodo del titolo, dove i termini più importanti sono le parole di contenuto presenti nel titolo, ovvero nella prima riga utile del documento.
- il metodo delle cue phrases è un criterio di estrazione del topic per trovare i termini più rilevanti nel documento. Le *bonus words* forniscono al paragrafo in questione un punteggio positivo al contrario le *stigma words* forniscono un punteggio negativo. Infatti, molti dei termini rilevanti sono nei paragrafi dove lo score delle bonus words è maggiore, e all'opposto quello delle stigma words è inferiore.

Infine, una volta ottenuto il testo riassunto, si procede ad una fase di valutazione andando a misurare l'output con le metriche BLEU e ROUGE che riguardano la precision e la recall rispettivamente.

Struttura del codice

Il codice è diviso in quattro file

- **`utils.py`** file che contiene metodi di utilità tra cui

    - *`bag_of_words`* metodo che data una frase in input consente di estrapolare le *bag of words* significative della frase. Per la generazione di quest'ultime, vengono richiamati, tramite una pipeline, altri metodi necessari per:
        - rimozione della punteggiatura
        - lemmatizzazione
        - rimozione delle stop words

    - *`parse_nasari_vectors`* metodo per la parsificazione dei vettori di Nasari, che va a definire quest'ultimi attraverso una struttura a dizionario per un accesso facilitato alle informazioni

    - *`parse_document`* metodo per la parsificazione dei documenti dati in input, che va rimuovere eventuali caratteri sporchi, link mantenendo nel documento solo il titolo e i paragrafi

    - *`get_gold_terms`* metodo per ottenere i termini rilevanti di un documento dato in input. Viene definito prima un contesto dato dalla parte iniziale e finale del documento, a questo punto si scorrono i paragrafi e si individuano quelli che fanno parte del contesto prima definito e si ritornano i termini dei paragrafi individuati rimuovendo eventuali duplicati.

    - *`get_summary_gold_terms`* metodo simile al precedente ma utilizzato per individuare le bag of words del riassunto dato in input.

- **`summary.py`** file che contiene il metodo principale che effettua il riassunto (*`summarization`*) e le sue funzioni di appoggio

    - *`search_lemma_in_values`* censente di cercare se esiste il lemma passato in input con i lemmi contenuti nel dizionario Nasari

    - *`define_context`* consente di definire un contesto di un testo dato in input.

    - *`rank`* ritorna il rango di una feature (lemma) all'interno di un vettore di Nasari

- `weighted_overlap` metrica di similarità dei vettori di Nasari. È definita considerando le features (lemmi) in comune tra due vettori in input

- `best_similarity` metodo che calcola la massima similarità tra i vettori di Nasari dei due testi dati in input

- `cue_phrases_method` metodo che assegna uno score ad ogni paragrafo di un documento dato in input sulla base delle bonus (+1) e stigma (-1) words. Tale metodo si avvale della funzione `parse_bonus_stigma_words` per recuperare i documenti contenenti le parole prima indicate (bonus-stigma)

- **`metrics.py`** definisce le due metriche implementate

  - `blue_metric`
  - `rouge_metric`

- **`main.py`** file principale dove vengono presi in input e ciclati i documenti e chiamata la funzione di `summarizzation` per poi stampare in output la tabella con i risultati finali delle metriche.

Implementazione

Inizialmente viene parsificato il file *dd-nasari.txt* (oppure lo *small* Nasari eventualmente) andando a definire un dizionario con chiave *title_ word* e valore un dizionario di coppie *lemma:score*.

Es. `'year': {'calendar': '748.9', 'day': '607.11',...}`

Al seguito vengono parsificati singolarmente i singoli documenti testuali, in andando a mantenere solo i paragrafi (compreso il titolo) rimuovendo eventuali link, caratteri sporchi (hashtag, ecc..).

A questo punto per ogni documento viene richiamato l'algoritmo `summarization` il quale prende in input

- documento parsificato (descritto poc'anzi)
- vettori di NASARI parsificati (descritto poc'anzi)
- reductions la lista contenente le percentuali di riduzione del documento (10, 20, 30%)

All'interno dell'algoritmo di `summarizzation`, vengono eseguiti in sintesi i seguenti passi

1. Individuazione del topic, può avvenire in due modalità
   a. Cue phrases vengono pre-processati i paragrafi del documento rimuovendo la punteggiatura e lemmatizzando i termini,

successivamente viene assegnato uno score ciascuno (descritto precedentemente) e ritornato il paragrafo con score maggiore.
   b. Titolo il titolo del documento viene pre-processato rimuovendo simboli di punteggiatura, spazi multipli, stop words e lemmatizzando i termini restanti, ottenendo un set di parole di contenuto

2. Definizione del contesto, il topic del documento viene analizzato attraverso la funzione `define_context` che genera un contesto, sfruttando la risorsa di Nasari, mediante due criteri:

   a. Salvare il vettore di Nasari la cui chiave matcha con il singolo lemma del topic
   b. Salvare il vettore di Nasari per il quale esiste un valore al suo interno che matcha con il singolo lemma del topic

3. Una volta ottenuto il contesto del topic vengono ciclati i paragrafi del documento andando a definire un contesto del singolo paragrafo richiamando la funzione appena descritta.

4. I vettori costruiti a partire dalle parole di contenuto presenti nei paragrafi vengono confrontati con i vettori del topic, usando la metrica di similarità Weighted Overlap.

5. A termine di tutti i confronti, le similarità trovate vengono sommate e divise per il numero di overlap trovati, ottenendo un valore di score che viene associato al paragrafo.

6. Una volta calcolate tutte le similarità con il contesto, i paragrafi vengono ordinati per valore di score decrescente di modo che quelli con il punteggio più alto saranno considerati come paragrafi più rilevanti del documento sintetizzato. (l'estrazione delle frasi per il riassunto avviene rispettando il loro ordinamento originale.)

7. Infine, in base alla percentuale di riduzione viene selezionato un sottoinsieme di $N - (N \times RP)$ paragrafi.
   Con $N =$ numero di paragrafi e $RP =$ percentuale di riduzione.

Per valutare la qualità del riassunto sono state utilizzate due metriche

- BLEU

$$\frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

- ROUGE

$$\frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

BLEU valuta la precision mentre ROUGE misura la Recall.

Le due metriche considerano l'intersezione tra i termini rilevanti estratti dal documento originale e le parole di contenuto presenti nel riassunto. Per individuare le parole più importanti presenti in tutto il documento si è usato un approccio semplicistico, andando a considerare tutte le keyword nel topic e nelle conclusioni.

Il risultato finale sarà un *bag of words* delle parole di tutte le frasi puntate dalle keywords. L'intersezione con il *bag of words* del documento riassunto calcola il numeratore di entrambe le metriche.

Il denominatore per BLEU sarà il numero di parole di contenuto nella sintesi mentre per ROUGE sarà il numero di parole rilevanti del documento originale.

Risultati

Di seguito una tabella che riporta i valori delle metriche implementate per i documenti in questione e sulle percentuali di riduzione utilizzate

/	Blue 10%	Rouge 10%	Blue 20%	Rouge 20%	Blue 30%	Rouge 30%
Andy-Warhol.txt	100.0	82.0	100.0	76.0	100.0	63.0
Ebola-virus-disease.txt	92.0	100.0	94.0	99.0	100.0	87.0
Life-indoors.txt	66.0	100.0	71.0	100.0	87.0	100.0
Napoleon-wiki.txt	90.0	86.0	95.0	86.0	96.0	82.0
Trump-wall.txt	100.0	100.0	100.0	95.0	99.0	79.0

Figura 1 Metodo del titolo

/	Blue 10%	Rouge 10%	Blue 20%	Rouge 20%	Blue 30%	Rouge 30%
Andy-Warhol.txt	100.0	87.0	100.0	82.0	100.0	73.0
Ebola-virus-disease.txt	92.0	100.0	91.0	82.0	89.0	69.0
Life-indoors.txt	66.0	100.0	71.0	100.0	87.0	100.0
Napoleon-wiki.txt	93.0	96.0	98.0	96.0	98.0	82.0
Trump-wall.txt	99.0	100.0	99.0	99.0	100.0	96.0

Figura 2 Metodo Cue Words

Conclusioni

Le schermate dei risultati delle metriche hanno lo scopo di mettere in evidenza le differenze tra i due approcci

Rispetto al metodo del titolo, le cue words riportano in media valori più alti per entrambe le metriche in ogni documento analizzato.

Nello specifico Bleu è pressappoco simile, per entrambe le modalità, in quanto molto bassa, questo perché i riassunti sono semplici paragrafi ben definiti, dunque la precisione è molto bassa.

La Rogue invece, segna una maggiore differenza. Questo è dato dal fatto che con la tecnica cue words si individuano più intersezioni con il documento sintetizzato, segno che nel riassunto il rapporto tra il numero di parole rilevanti trovate e il totale delle parole importanti è più alto. Dall'altra parte Rogue è più significativa in quanto riguarda la recall. Si nota inoltre che con il metodo delle cue word il valore diminuisce sempre rispetto alla compressione, mentre per con il titolo si evince che è leggermente migliore.

È interessante notare che nel testo '*Napoleon-wiki.txt'* il metodo delle cue words ha una predominanza nei valori di entrambe le metriche rispetto al metodo del titolo, questo è dovuto al fatto che il titolo di questo testo è poco informativo (viene solamente indicato "*Napoleone Bonaparte*") per cui le cue words riescono ad ottenere risultati migliori.

In ogni caso questi risultati sono soggetti a diversi fattori che entrano in gioco tra cui la tipologia del documento che si va a trattare (scientifico, letterale, ecc), e la tipologia di cue words scelte, per cui non vi è un metodo preferito in assoluto.