

Metodologia di test

Sebastiano Caccaro

14 giugno 2021

1 Introduzione

Questo documento ha lo scopo di definire le metodologie di test per valutare l'efficacia di alcuni dei metodi di OCR post-processing presenti in letteratura. Il documento è così strutturato:

- Nella sezione 3 torefsec:dataset è descritto il dataset usato per il training e/o la valutazione dei vari approcci.
- Nella sezione 3 viene fornita una breve overview per ognuno degli approcci scelti. Vengono inoltre definite le metodologie di allenamento e/o di test per ognuno degli approcci.
- Nella sezione 4 sono definite le misure e le metriche attraverso le quali gli approcci scelti verranno valutati.

2 Dataset

Il dataset usato è ricavato da un insieme di documenti proveniente dagli archivi vaticani. I documenti comprendono trascrizioni di discorsi, encicliche, lettere e altri tipi di testo che spaziano dal 6 Luglio 1439 al 22 Aprile 2021. Ogni documento contiene dei metadati che specificano dettagli come data, lingua, titolo e autore. Inoltre, per ogni documento sono presenti sia il testo originale che lo stesso testo già diviso in paragrafi.

Il testo estratto può essere considerato corretto sia a livello di singolo token, che a livello di segmentazione.

Da questi documenti viene dunque costruito un dataset nelle seguenti fasi:

- Spezzettamento
- Perturbazione

2.1 Frammentazione

La fase di frammentazione consiste nel ricavare da ogni paragrafo uno o più frammenti di testo che andranno a costituire dei data point nel dataset. Le regole per la frammentazione sono quelle descritte in [1]:

- Ogni frammento deve avere al più 50 caratteri;
- Quando possibile, la frammentazione avviene su segni di punteggiatura che sanciscono la fine di una frase (.?!:);
- Se il frammento in questione risultasse più lungo di 50 caratteri, la frammentazione avviene sui numeri;
- Se ancora il frammento risultasse più lungo di 50 caratteri, la frammentazione avviene al 50° carattere.

Nel frammentare i paragrafi, quando possibile, viene favorita la creazione di frammenti con un numero di caratteri il più vicino possibile a 50.

Applicando questo approccio, è stato possibile ottenere 2827879 frammenti distinti.

2.2 Perturbazione

I frammenti fin'ora individuati sono considerati corretti, ovvero non contengono errori. Per testare la capacità dei vari approcci individuati di correggere errori OCR, è necessario associare ad ogni frammento corretto c , un frammento c' così definito:

$$c' = f_{err}(c) \quad (1)$$

dove f_{err} è una funzione che introduce degli errori in c . Per testare l'efficacia di ogni approccio su testi con diversi livelli di intensità di errore, vengono prodotte più versioni dello stesso dataset. Ogni versione del dataset è ottenuta utilizzando una diversa funzione f_{err} .

Utilizzando le definizioni di Pipeline e SuperPipeline date in ([Inserire ref qui](#))

è possibile associare ad ogni versione del dataset una funzione f_{err} corrispondente ad una diversa SuperPipeline.

Sono quindi individuate due categorie di Pipeline:

- **Di segmentazione:** sono pipeline che introducono unicamente errori che intaccano la segmentazione del testo. Sono composti dai seguenti moduli:

- Split
- AddPunct
- MergeHypen
- SplitComma

Esempio di perturbazione: **Inserire esempio**

- **Di token:** sono pipeline che introducono unicamente errori che intaccano i singoli token, senza danneggiare la segmentazione. Sono composti dai seguenti moduli:

- SubChar

Esempio di perturbazione: **Inserire esempio**

- **Miste:** sono pipeline che introducono sia errori a livello di token, che di segmentazione. Sono composte dalla concatenazione di una pipeline di segmentazione concatenata ad una pipeline di token.

Esempio di perturbazione: **Inserire esempio**

Sono quindi definite le seguenti Pipeline, dove ogni ad ogni modulo è associata la sua probabilità:

Nome	Split	AddPunct	MergeHypen	SplitComma	SubChar
s1	0.0025	0.005	0.001	0.001	/
s2	0.008	0.025	0.001	0.002	/
s3	0.05	0.1	0.01	0.02	/
t1	/	/	/	/	0.1
t2	/	/	/	/	0.3
t3	/	/	/	/	0.8

Sono denotate con tx le pipeline di token, e con sx le pipeline di segmentazione. Maggiore è x, maggiore è l'intensità di errore delle pipeline. È quindi possibile ricavare le seguenti pipeline miste:

- $m1 = s1 + t1$
- $m2 = s2 + t2$
- $m3 = s3 + t3$

Sono poi definite le seguenti SuperPipeline. Ogni superpipeline è composta da una serie di pipeline, ad ognuna delle quali è associato un peso:

Nome	s1	s2	s3	t1	t2	t3	m1	m2	m3
S1	6	4	1	/	/	/	/	/	/
S2	2	8	1	/	/	/	/	/	/
S3	1	6	4	/	/	/	/	/	/
T1	/	/	/	6	4	1	/	/	/
T2	/	/	/	2	8	1	/	/	/
T3	/	/	/	1	4	4	/	/	/
M1	/	/	/	/	/	/	6	4	1
M2	/	/	/	/	/	/	2	8	1
M3	/	/	/	/	/	/	1	6	4

Sono quindi presenti 9 versioni diverse del dataset, ognuna delle quali è composta da coppie (c, c') dove $c' = f_{err}(c)$, con una diversa funzione $f_{err} \in \{S1, S2, S3, T1, T2, T3, M1, M2, M3\}$ a seconda della versione.

3 Approcci

4 Misure e metriche di valutazione

Riferimenti bibliografici

- [1] Vivi Nastase and Julian Hirschler. Correction of ocr word segmentation errors in articles from the acl collection through neural machine translation methods. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.