Relazione TLN - Prima parte

Ilaria Figliuzzi

Questa relazione tratta dell'implementazione dell'esercizio La magia NER nascosta.

Struttura del progetto -

Nella cartella del progetto si potrà visualizzare:

- La cartella **data** sono presenti:
  - Le cartelle:
    - ita (con il file test, train e val)
    - eng (con il file test, train e val)
  - Il file
    - valutazione_frasiDelProf → che sarà utile nella quarta parte, in quanto sono presenti alcune frasi da valutare
- I file python:
  - La_magia_NER_nascosta.py → è il file main che richiama tutti gli altri file python
  - corpusAlgoritmo.py → è il file utile per la fase di decoding
  - faseDiLearning.py → è il file utile per la fase di learning
  - faseDiSmoothing.py → è il file utile per "gestire" le parole sconosciute

Infine, la struttura di questa relazione verrà così suddivisa:

- Pre-compilazione
- Fase di learning
- Fase di decoding
- Post-compilazione e valutazione

FASE DI PRE-COMPILAZIONE

La prima interazione con il progetto si ha con la domanda che ci pone il compilatore: “In quale lingua mi vuoi istruire?” A questa domanda bisogna rispondere `it o eng` e tramite questa risposta si vanno a settare i file corretti di train, test e val nell’opportuna lingua.

FASE DI LEARNING

La fase di learning è condotta dalla funzione che prende il suo nome, ossia la funzione `fase_di_learning`. Essa si divide in due fasi:

- Conteggi → C(ti | wi) e C(ti-1 | ti)
- Probabilità di emissione e transizione → P(ti | ti-1) e P(wi | ti)

Conteggi

Tale parte è fondamentale per svolgere il secondo punto della fase di learning.

C(ti | wi)

Attraverso la funzione `riempi_dict_quando_non_è_una_riga_vuota`, si va a riempire/incrementare un dizionario, in modo tale che possano contare per ogni parola quali e quanti i tag vengono attribuiti.

In dettaglio:

- se è la prima parola della frase, viene dato il tag First ed incrementato il conteggio di questo tag
- per le successive parole si fa il conteggio di quante volte alla parola W corrente viene dato un tag T

C(ti-1 | ti)

Attraverso la funzione `contatore_di_tag_e_parole`, si va a riempire/incrementare un dizionario, in modo tale che si possa contare quanti tag sono preceduti in un particolare tag T corrente.

Inoltre, viene introdotto anche il tag LAST che rappresenta l'ultima parola della frase.

Tutti questi dati, come richiesto, verranno salvati in forma logaritmica, per evitare in fase di decoding il fenomeno di underflow.

Probabilità di emissione e transazione

Nella seconda parte della funzione `fase_di_learning`, infine, recuperando conteggi precedenti si può procedere alla creazione del modello probabilistico, calcolando perciò le due probabilità.

P(wi | ti) → probabilità che un tag t sia attribuito ad una parola w

P(ti | ti-1) → probabilità che compaia un tag T dato un tag precedente ti-1

Parole sconosciute (smoothing)

In questa fase, come richiesto, si procede a settare un tag a delle parole. Questa esigenza nasce nel caso in cui alcune parole non siano presenti nel training test. Per far ciò sono state create delle probabilità sotto elencate per fare smoothing assumptions.

1. P(unk|O) = Ad una parola sconosciuta si attribuisce il tag O con una probabilità pari ad 1
2. P(unk|O) = P(unk|B-MISC) = Ad una parola sconosciuta si attribuisce il tag O con una probabilità pari ad 0.5
3. P(unk|ti) = 1/#(PoS_TAGs) = in questo caso ad ogni parola sconosciuta viene associata una probabilità uniforme.
4. Ed infine “statistica TAG sul development set: parole che compaiono 1 sola volta” che sarà svolta dalla funzione `smoothingFour`.

   In questa funzione, verranno individuate tutte le parole che compaiono una volta. Per ogni tag, successivamente viene calcolata la probabilità con il rapporto tra il numero totale di occorrenze del tag T per ogni parola con il numero di parole presenti una volta sola.

FASE DI DECODING

L'ultima fase, ovvero quella di decoding, è svolta nel file *corpusAlgoritmo.py*

In questa classe tra i metodi principali possiamo trovare lo sviluppo:

- Dell'algoritmo di Viterbi
- Baseline.

Algoritmo di viterbi

Nell'algoritmo di Viterbi è possibile associare un tag utilizzando la programmazione dinamica e perciò senza valutare ogni combinazione dei tag. Infatti, visto che vogliamo la sequenza massima dovremmo provare tutte le possibili sequenze di tag associate ad una frase, calcolare la probabilità e scegliere infine quella massima. Come si può intuire facilmente tale problema esplode in maniera esponenziale, ma grazie all'algoritmo di Viterbi ciò non succede.

L'algoritmo di Viterbi si basa su tre punti:

1. Si inizia con il "creare" la prima colonna, con la somma delle probabilità (emissione e transazione) tag T sia preceduto da ogni tag FIRST. Se non esiste viene messa ad una probabilità nulla.
2. La fase centrale dell'algoritmo è contraddistinta da dei passi ripetitivi: ossia per ogni singola parola per tutti i possibili TAG vengono calcolate tutte le somme delle probabilità più la somma della probabilità della parola attuale.
3. Infine, nella fase finale si va a completare l'ultima colonna analogamente alla prima facendo riferimento questa volta al tag LAST.

Algoritmo di Baseline

Per quanto riguarda l'algoritmo di Baseline, si scorre ogni parola della frase e viene associato il tag più frequente per quella parola, nel caso in cui tale parola non è presente verrà dato il tag MISC.

FASE DI POST-COMPILAZIONE E VALUTAZIONE

Dopo che anche la fase di decoding si è conclusa:

1. Si stampano i risultati:
    a. Nel caso in cui i si era scelta la *lingua italiana* i risultati sono i seguenti:

        smoothing1: 68.57 %

        smoothing2: 68.57 %

        smoothing3: 76.56 %

        smoothing4: 76.87 %

        baseline: 94.01%

    b. Nel caso in cui i si era scelta la *lingua inglese* i risultati sono i seguenti:

        smoothing1: 65.96 %

        smoothing2: 65.96 %

        smoothing3: 75.05 %

        smoothing4: 75.16 %

        baseline: 92.89%

2. Infine si procede a chiedere all'utente se si vogliono testare le frasi di valutazione sotto elencate. Se si risponde in maniera affermativa, si procede stampando i risultati.

    Nel file *valutazione_frasiDelProf.conllu* ho inserito le frasi:

    - La vera casa di Harry Potter è il castello di Hogwards.
    - Harry le raccontò del loro incontro a Diagon Alley.
    - Mr Dursley era dire/ore di una di/a di nome Grunnings, che fabbricava trapani.

    In questo file ad ogni parola della frase ho inserito il TAG più corretto suggerito da un sito.

    Ho fatto ciò per consentire una valutazione più accurata. Il compilatore dovrà valutare le frasi autonomamente e dovrà poi confrontare i suoi risultati con i TAG del file *valutazione_frasiDelProf.conllu.*

    Ho inserito, infine, una tabella che riassume tali risultati.

    <u>Nel caso in cui si utilizza la lingua italiana come trainset di partenza:</u>

	La	vera	casa	di	Harry	potter	è	il	castello	di	Hogwarts	.
smoothing 1	B-MISC	I-MISC	B-LOC	B-LOC	B-PER	I-MISC	I-MISC	O	B-LOC	B-LOC	B-LOC	O
smoothing 2	B-MISC	I-MISC	B-LOC	B-LOC	B-PER	I-MISC	I-MISC	O	B-LOC	B-LOC	B-LOC	O
smoothing 3	B-MISC	I-MISC	B-LOC	I-LOC	B-PER	I-MISC	I-MISC	O	B-LOC	I-LOC	B-MISC	O
smoothing 4	B-MISC	I-MISC	B-LOC	I-LOC	B-PER	I-MISC	I-MISC	O	B-LOC	I-LOC	B-MISC	O
Baseline	O	O	O	O	B-PER	I-MISC	O	O	O	O	MISC	O

	Harry	le	raccontò	del	loro	incontro	a	Diagon	Alley	.
smoothing 1	B-PER	O	O	O	O	O	O	O	O	O
smoothing 2	B-PER	O	O	O	O	O	O	O	O	O
smoothing 3	B-PER	O	O	O	O	O	I-MISC	B-PER	I-PER	O
smoothing 4	B-PER	O	O	O	O	O	I-MISC	B-PER	I-PER	O
Baseline	B-PER	O	O	O	O	O	O	MISC	I-PER	O

	Mr	Dursley	era	direttore	di	una	ditta	di	nome	Grunnis	,	che	fabbricava	trapani	.
smoothing 1	B-MISC	I-MISC	O	O	O	O	O	O	O	O	O	O	O	O	O
smoothing 2	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
smoothing 3	B-MISC	I-MISC	O	O	O	O	O	O	I-MISC	B-ORG	I-MISC	I-MISC	B-ORG	B-MISC	O
smoothing 4	B-MISC	I-MISC	O	O	O	O	O	O	I-MISC	B-ORG	O	I-MISC	B-ORG	B-MISC	O
Baseline	B-MISC	MISC	O	O	O	O	O	O	O	MISC	O	O	MISC	MISC	O

- *I risultati ottenuti testando le tre frasi tra il tag ottenuto dagli algoritmi con la lingua di partenza in italiano e il file valutazione_frasiDelProf.conllu è il seguente:*

  smoothing1: 18.92 %

  smoothing2: 18.92 %

  smoothing3: 40.54 %

  smoothing4: 43.24 %

  smooth baseline: 59.46%

  Le compatibilità minori riscontrate tra TAG degli algoritmi e i tag del file *valutazione_frasiDelProf.conllu*, sono per smoothing1 e smoothing2.

- *I risultati ottenuti testando le tre frasi tra il tag ottenuto dagli algoritmi con la lingua di partenza in inglese e il file valutazione_frasiDelProf.conllu è il seguente:*

  smoothing1: 2.7 %

  smoothing2: 2.7 %

  smoothing3: 13.51 %

  smoothing4: 16.22 %

  smooth baseline: 18.92%

  Le compatibilità minori riscontrate tra TAG degli algoritmi e i tag del file *valutazione_frasiDelProf.conllu*, sono per smoothing1 e smoothing2.