# Relazione Progetto TLN: Mazzei

Andrea Senese, Lorenzo Favaro, Lorenzo Vercelli

25 aprile 2022

# Indice

# Capitolo 1

# Fase di Learning

La fase di learning si suddivide a sua volta in due sotto-fasi: la prima consiste nel costruire la matrice di transizione per calcolare le probabilità di transizione tra due tag qualsiasi, mentre nella seconda viene costruita la matrice di emissione contenente le singole probabilità di emissione di ogni parola dato l'insieme dei tag. Agli output di ciascuna di queste fasi è stata infine applicata la funziona logaritmica per migliorare le performance.

## 1.1 Matrice di transizione

Per costruire ogni probabilità di transizione abbiamo utilizzato la libreria `conllu`. Tramite essa abbiamo effettuato il parsing delle frasi del training set, le quali vengono trasformate in liste di token, ossia dei dizionari contenenti le chiavi *form* (per indicare la parola) e *lemma* (per indicare il NER-tag associato).

Dati due tag sequenziali $P(t_i)$ e $P(t_{i-1})$, abbiamo creato i dizionari `tag_counter` per tenere il conto delle occorrenze di ogni tag (ossia $C(t_{i-1})$) e `transition_counter` per contare le occorrenze della coppia (ossia $C(t_{i-1}, t_i)$). Innanzitutto, per ogni frase parsificata contiamo una volta il tag fittizio *START* e la transizione da *START* al primo tag della frase. Dopodiché, generiamo tutte le coppie sequenziali fino al penultimo token e incrementiamo di uno i valori delle relative chiavi dei due dizionari. Ad esempio, data la seguente frase:

Paolo\B-PER ama\O Francesca\B-PER dolcemente\O

le coppie di tag ricavate sono `[(B-PER,O), (O,B-PER), (B-PER,O)]`. Infine, contiamo manualmente l'occorrenza dell'ultimo tag.

Una volta che i dizionari sono stati popolati, per calcolare la probabilità di transizione $P(t_i \mid t_{i-1})$ abbiamo generato tutte le combinazioni di tag e in corrispondenza della riga *i* e della colonna *i-1* inseriamo il valore `transition_counter[(`$t_i$`, `$t_{i-1}$`)] / tag_counter[`$t_i$`]`

| | O | I-LOC | I-MISC | I-ORG | I-PER | B-LOC | B-MISC | B-PER | B-ORG |
|---|---|---|---|---|---|---|---|---|---|
| **O** | 0.87 | 0.00 | 0.00 | 0.00 | 0.00 | 0.03 | 0.02 | 0.02 | 0.01 |
| **I-LOC** | 0.75 | 0.24 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **I-MISC** | 0.46 | 0.00 | 0.54 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **I-ORG** | 0.59 | 0.00 | 0.00 | 0.41 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **I-PER** | 0.88 | 0.00 | 0.00 | 0.00 | 0.12 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-LOC** | 0.74 | 0.26 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-MISC** | 0.65 | 0.00 | 0.32 | 0.00 | 0.00 | 0.00 | 0.02 | 0.00 | 0.00 |
| **B-PER** | 0.36 | 0.00 | 0.00 | 0.00 | 0.64 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-ORG** | 0.50 | 0.00 | 0.00 | 0.49 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

## 1.2 Matrice di emissione

La costruzione della matrice di emissione passa attraverso due step principali: il primo consiste nel computare una volta sola le probabilità di emissione per le parole presenti nel training set e il secondo nel costruire la matrice vera a propria per ogni frase del test set, impiegando eventuali strategie di smoothing nel caso in cui una parola non abbia già una probabilità di emissione associata. Nel primo step abbiamo utilizzato un nuovo dizionario `word_tag_counter` per contare le occorrenze di ogni coppia $(w_i,t_i)$ (parola-tag), oltre a quello già utilizzato precedentemente `tag_counter`. La probabilità di emissione $P(w_i \mid t_i)$ è stata quindi calcolata come `word_tag_counter[(`$w_i$`,`$t_i$`)] / tag_counter[`$t_i$`]`.

Per quanto riguarda il secondo step, abbiamo definito una matrice le cui righe sono i tag e le cui colonne sono le parole delle frasi del test set. Qui utilizziamo l'output dello step precedente per generare tutte le combinazioni di coppie (parola-tag) per produrre le probabilità di emissione da associare agli elementi della matrice. Nel caso in cui la probabilità di emissione associata ad una parola non sia stata computata (in quanto non presente nel training set), la si ottiene utilizzando una delle strategie di smoothing descritte nella prossima sezione.

| | She | proceeded | to | the | University | of | the | Philippines | College | of | Law | . |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **O** | 0.00 | 0.00 | 0.02 | 0.06 | 0.00 | 0.03 | 0.06 | 0.00 | 0.00 | 0.03 | 0.00 | 0.05 |
| **I-LOC** | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.04 | 0.00 | 0.00 | 0.00 | 0.04 | 0.00 | 0.00 |
| **I-MISC** | 0.00 | 0.00 | 0.01 | 0.03 | 0.00 | 0.05 | 0.03 | 0.00 | 0.00 | 0.05 | 0.00 | 0.00 |
| **I-ORG** | 0.00 | 0.00 | 0.00 | 0.01 | 0.02 | 0.06 | 0.01 | 0.00 | 0.01 | 0.06 | 0.00 | 0.00 |
| **I-PER** | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-LOC** | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-MISC** | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-PER** | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **B-ORG** | 0.00 | 0.00 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

## 1.3 Strategie di smoothing

Le strategie di smoothing che abbiamo implementato sono raccolte in una classe enumeratore "SmoothingStrategy" utile a specificare quale venga utilizzata al momento dell'esecuzione. In questa sezione ci concentriamo sul descrivere l'ultima strategia di quelle elencate di seguito, in quanto l'implementazione delle altre è immediata.

1. `uniform`: $P(w_i \mid t_i) = \frac{1}{\#NER\_TAGs}$
2. `always_other`: $P(w_i \mid \texttt{O}) = 1$
3. `other_and_misc`: $P(w_i \mid \texttt{O}) = P(unknown \mid \texttt{B\_MISC}) = 0.5$
4. `one_shot_word`: $P(w_i \mid t_i) = \frac{C(w_i,t_i)}{\sum_{j \in len(T)} C(w_j,t_j)}$

Per l'implementazione di `one_shot_word` abbiamo utilizzato come dataset di riferimento il development set (`val.conllu`). Abbiamo popolato il dizionario `word_tag_counter`, per poi andare a calcolare per ogni tag $t_i$ la distribuzione delle probabilità di emissione di tutte le coppie $(w_i, t_i)$ la cui occorrenza è pari a 1. Quindi, durante la computazione della matrice di emissione, ad ogni coppia $(w_i, t_i)$ a cui non corrisponde nessuna coppia nel dizionario delle probabilità di emissione verrà assegnata la probabilità associata al tag calcolata rispetto alle parole con occorrenza singola.

# Capitolo 2

# Fase di Decoding

Nella fase di decoding, per ogni frase viene rilevato l'insieme dei tag reali (reference), assieme alla relativa matrice di emissione. Quindi, in base all'algoritmo di decoding selezionato precedentemente viene costruito l'insieme dei tag predetti dall'algoritmo. I risultati e le reference vengono quindi inseriti in due liste globali utilizzate per calcolare la bontà degli algoritmi. Le metriche impiegate per la valutazione sono state importate da Scikit Learn. Nelle seguenti sezioni vengono discussi gli algoritmi di decoding implementati.

## 2.1 Most Frequent Tag (MFT)

L'algoritmo "Most Frequent Tag" è una delle baseline implementate e restituisce come predizione la lista di tag più frequenti rispetto a una data sequenza di parole in input. Nello specifico, ad ogni parola viene associato il tag con maggior probabilità ($P(w \mid t)$) rispetto alla matrice di emissione calcolata sul training set.

## 2.2 Viterbi

L'implementazione dell'algoritmo di Viterbi segue da vicino quella vista a lezione, e fa uso di una matrice di Viterbi inizializzata a $-\infty$ (per via del passaggio ai logaritmi). È suddivisa in tre punti principali:

1. le probabilità iniziali vengono calcolate sommando (e non moltiplicando, per via dei logaritmi) i valori della prima colonna della matrice di emissione a quelli dell'array $\Pi$ (contenente i valori delle probabilità di transizione dal tag fittizio START ad ogni altro);
2. per il calcolo delle probabilità intermedie abbiamo ridefinito la funzione *argmax* in modo che restituisca sia l'indice del tag più frequente, utilizzato per il calcolo del backpointer, sia il suo valore di probabilità di transizione;

3. per l'ultima parola è sufficiente applicare l'argmax all'ultima colonna della matrice di viterbi e restituire l'indice del tag, che viene utilizzato come primo punto di partenza nella matrice backpointer per la ricostruzione del percorso migliore.

# Capitolo 3

# Esperimenti

Di seguito vengono discussi alcuni esperimenti che abbiamo effettuato nel corso dell'esercitazione, con le relative osservazioni. Abbiamo eseguito entrambi gli algoritmi di decoding applicando le varie strategie di smoothing implementate.

Il primo esperimento è stato eseguito sulla frase "La vera casa di Harry Potter è il castello di Hogwarts".

| | **La** | **vera** | **casa** | **di** | **Harry** | **Potter** | **è** | **il** | **castello** | **di** | **Hogwarts** | **.** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | O | O | O | O | B-MISC | I-MISC | O | O | O | O | B-MISC | O |

I risultati ottenuti sono i seguenti:

**Viterbi**

| | **La** | **vera** | **casa** | **di** | **Harry** | **Potter** | **è** | **il** | **castello** | **Hogwarts** | **.** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | O | O | O | O | B-MISC | I-MISC | O | O | O | B-MISC | O |
| **Uniform** | O | B-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | O | O |
| **Always-other** | O | O | O | O | O | O | O | O | O | O | O |
| **Other-and-Misc** | O | B-MISC | B-MISC | I-MISC | O | O | O | O | B-MISC | O | O |
| **Oneshot-Word** | O | B-LOC | I-LOC | I-LOC | O | O | B-LOC | I-LOC | I-LOC | O | O |

In assenza di tecniche di smoothing si ottiene una soluzione più accurata, in quanto la categorizzazione di ogni parola avviene con il giusto tag. Utilizzando le varie tecniche di smoothing si presentano associazioni errate di tag in modo sparso.

**MFT**

| | La | vera | casa | di | Harry | Potter | è | il | castello | Hogwarts | . |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | B-MISC | O | I-MISC | I-LOC | B-MISC | I-MISC | O | O | B-LOC | B-MISC | O |
| **Uniform** | I-ORG | I-LOC | I-LOC | I-LOC | O | O | I-LOC | I-LOC | I-LOC | O | I-LOC |
| **Always-other** | B-MISC | O | I-MISC | I-LOC | O | O | O | O | B-LOC | O | O |
| **Other-and-Misc** | B-MISC | B-MISC | B-MISC | B-MISC | O | O | B-MISC | O | B-MISC | O | B-MISC |
| **Oneshot-Word** | I-ORG | I-PER | B-PER | I-LOC | O | O | I-PER | B-LOC | I-PER | O | I-PER |

Analogamente, utilizzando la baseline (MFT) si hanno risultati errati sia in presenza di smoothing che in assenza.

In termini di accuracy abbiamo riscontrato i seguenti risultati:

| | **Viterbi** | **MFT** |
|---|---|---|
| **No-smoothing** | 1.0 | 0.58 |
| **Uniform** | 0.16 | 0.00 |
| **Always-other** | 0.75 | 0.33 |
| **Other-and-Misc** | 0.33 | 0.08 |
| **Oneshot-Word** | 0.16 | 0.00 |

Rispetto alla baseline implementata, le performance dell'algoritmo di Viterbi sono risultate migliori in tutti i casi. A giudicare dai risultati ottenuti, le tecniche di smoothing utilizzate non sono così accurate. Possiamo vedere che la tecnica di smoothing 'Always-other' ottiene le performance più elevate; questo è probabilmente dovuto dallo sbilanciamento del NER-tag di tipo 'Other' presente sia nel training-set che nel test-set.

Di seguito riportiamo l'elenco degli ulteriori esperimenti effettuati con annessi risultati:

Secondo esperimento: "Harry le raccontò del loro incontro a Diagon Alley."

| | **Harry** | **le** | **raccontò** | **del** | **loro** | **incontro** | **a** | **Diagon** | **Alley** | **.** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Reference** | B-MISC | O | O | O | O | O | O | B-LOC | I-LOC | O |

**Viterbi**

| | **Harry** | **le** | **raccontò** | **del** | **loro** | **incontro** | **a** | **Diagon** | **Alley** | **.** |
|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | B-PER | O | O | O | O | O | O | B-PER | I-PER | O |
| **Uniform** | O | B-LOC | I-LOC | I-LOC | I-LOC | I-LOC | O | O | O | O |
| **Always-other** | O | O | O | O | O | O | O | O | O | O |
| **Other-and-Misc** | O | B-MISC | B-MISC | O | B-MISC | B-MISC | O | O | O | O |
| **Oneshot-Word** | O | B-PER | I-PER | O | B-PER | I-PER | I-PER | O | O | O |

**MFT**

| | Harry | le | raccontò | del | loro | incontro | a | Diagon | Alley | . |
|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | B-MISC | O | O | I-LOC | O | O | O | O | I-PER | O |
| **Uniform** | O | B-LOC | I-LOC | B-LOC | I-LOC | I-LOC | I-PER | O | O | I-LOC |
| **Always-other** | O | O | O | I-LOC | O | O | O | O | O | O |
| **Other-and-Misc** | O | B-MISC | B-MISC | I-LOC | B-MISC | B-MISC | B-MISC | O | O | B-MISC |
| **Oneshot-Word** | O | B-LOC | I-PER | B-LOC | I-PER | I-PER | I-PER | O | O | I-PER |

Terzo esperimento: "Mr Dursley era direttore di una ditta di nome Grunnings, che fabbricava trapani."

| | Mr | Dursley | era | direttore | di | una | ditta | nome | Grunnings | , | che | fabbricava | trapani | . |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Reference** | B-MISC | I-MISC | O | O | O | O | O | O | B-ORG | O | O | O | O | O |

**Viterbi**

| | Mr | Dursley | era | direttore | di | una | ditta | nome | Grunnings | , | che | fabbricava | trapani | . |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | B-MISC | I-MISC | O | O | O | O | O | O | O | O | O | O | O | O |
| **Uniform** | O | O | B-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | O | O | O |
| **Always-other** | O | O | O | O | O | O | O | O | O | O | O | O | O | O |
| **Other-and-Misc** | O | O | B-MISC | B-MISC | O | O | B-MISC | B-MISC | O | O | O | O | O | O |
| **Oneshot-Word** | O | O | B-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | O | O | O | O | O | O |

**MFT**

| | Mr | Dursley | era | direttore | di | una | ditta | nome | Grunnings | , | che | fabbricava | trapani | . |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **No-smoothing** | B-MISC | O | O | O | I-LOC | O | O | O | O | O | O | O | O | O |
| **Uniform** | O | O | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | I-LOC | O | I-LOC | I-LOC | O | O | I-LOC |
| **Always-other** | O | O | O | O | I-LOC | O | O | O | O | O | O | O | O | O |
| **Other-and-Misc** | O | O | B-MISC | B-MISC | B-MISC | B-MISC | B-MISC | B-MISC | O | B-MISC | B-MISC | O | O | B-MISC |
| **Oneshot-Word** | O | O | I-PER | I-PER | I-LOC | I-PER | I-PER | I-PER | O | B-LOC | I-PER | O | O | I-PER |

### 3.0.1 Test set : test.connlu

Abbiamo testato gli algoritmi di Viterbi e di baseline sul test-set italiano e inglese.

Le performance ottenute da Viterbi sono promettenti e si distinguono decisamente dalla baseline. Ciò indica che la fase di training è stata ottimale. Di seguito, riportiamo i risultati ottenuti:

Parametri:

- Algoritmo: Viterbi;
- Smoothing: None;
- Dataset: Inglese;

```
Total words: 267155
Accuracy: 96.32%

Classification Report
              precision    recall  f1-score   support

           O       0.83      0.76      0.79      5955
       I-LOC       0.72      0.64      0.68      4505
      I-MISC       0.80      0.70      0.75      3449
       I-ORG       0.89      0.78      0.83      5207
       I-PER       0.86      0.75      0.80      2063
       B-LOC       0.79      0.64      0.71      4897
      B-MISC       0.82      0.76      0.79      2622
       B-PER       0.91      0.84      0.87      3959
       B-ORG       0.98      1.00      0.99    234498

    accuracy                           0.96    267155
   macro avg       0.85      0.76      0.80    267155
weighted avg       0.96      0.96      0.96    267155


Confusion Matrix
              O  I-LOC  I-MISC  I-ORG  I-PER  B-LOC  B-MISC  B-PER  B-ORG
O        233468     41     283     98     79     99     230     76    124
I-LOC       206   1541      55     89     43     92      15     14      8
...
B-LOC       955     47      27     18     30   4523     107     77    171
B-MISC      946      7     244     14      7    177    2894    109    107
B-PER       767      5      31      1     83    135     109   4037     39
B-ORG       480      3      32     82      7    243     135     49   2418
```

Parametri:

- Algoritmo: Viterbi;
- Smoothing: None;
- Dataset: Italiano;

```
Total words: 313066
Accuracy: 97.44%

Classification Report
              precision    recall  f1-score   support

           O       0.90      0.84      0.87      9788
       I-LOC       0.81      0.66      0.73      2342
      I-MISC       0.88      0.83      0.85      2229
       I-ORG       0.95      0.86      0.91      8377
       I-PER       0.85      0.73      0.79      3034
       B-LOC       0.79      0.62      0.69      3331
      B-MISC       0.89      0.75      0.81      1045
       B-PER       0.94      0.89      0.91      7094
       B-ORG       0.98      1.00      0.99    275826

    accuracy                           0.97    313066
   macro avg       0.89      0.80      0.84    313066
weighted avg       0.97      0.97      0.97    313066


Confusion Matrix
              O  I-LOC  I-MISC  I-ORG  I-PER  B-LOC  B-MISC  B-PER  B-ORG
O        274857    163     315     16    105    196      78     73     23
I-LOC       472   2206      48     22     70    165      15     32      4
...
B-LOC      1283     87      13      2     31   8174      34     60    104
B-MISC      523      2      69      1      6     84    1543     57     57
B-PER       729     26      22      0    146    105      65   7242     42
B-ORG       205      1      19     10      5     65      60     19   1845
```