Università degli Studi di Genova

Dipartimento di Matematica

Corso di studi in

Matematica

Anno accademico 2023/2024

Tesi di Laurea

Named Entity Recognition per anonimizzazione e clustering di documenti

Relatori
Curzio Basso
Silvia Villa

Correlatori
Cesare Molinari

Candidato
Giulia Sacco

Indice

Introduzione

La tesi si concentra sull'analisi automatica di documenti tramite tecniche di Natural Language Processing (NLP) e Machine Learning. Questa tesi è stata svolta in collaborazione con l'azienda Camelot Biomedical Systems. In particolare, l'applicazione riguarda l'analisi di un dataset di curricula vitae.
Uno dei compiti del NLP consiste nell'estrazione di informazioni dai testi. Tra le varie informazioni che si possono estrarre da un testo si è interessati alle "named entities" o entità denominate. Un'entità denominata è qualsiasi cosa a cui si possa fare riferimento con un nome proprio come ad esempio una persona, un luogo, un'organizzazione. Tuttavia, il termine entità denominata è comunemente esteso per includere cose che non sono entità di per sé, tra cui espressioni temporali come date e orari e persino espressioni numeriche come i numeri di telefono, codici fiscali e molto altro. Il processo di identificazione e classificazione delle entità è detto Named Entity Recognition (NER).
Tra le entità che si possono estrarre, di particolare importanza è la categoria dei dati sensibili. Per questo è stato fondamentale, nella prima fase, effettuare l'anonimizzazione di dati come nomi, indirizzi, codici fiscali e indirizzi e-mail, attraverso l'utilizzo di tecniche di Named Entity Recognition.
L'anonimizzazione dei dati sensibili nei curriculum vitae ha diversi scopi importanti. Innanzitutto, contribuisce a ridurre i pregiudizi nei processi di selezione del personale, promuovendo una valutazione più imparziale e concentrata sulle competenze e qualifiche professionali dei candidati. Inoltre, garantisce la protezione della privacy, un aspetto fondamentale nell'era digitale, soprattutto quando i CV vengono analizzati da modelli di linguaggio pre-addestrati, come quelli forniti da OpenAI o altri provider esterni.

Oltre all'anonimizzazione dei dati sensibili all'interno dei CV, un altro obiettivo di questa tesi consiste nell'effettuare clustering dei curricula per ottenere una descrizione dei profili professionali dei candidati presenti nel dataset. A questo scopo, è stato impiegato l'algoritmo *k-means*, un metodo di clustering non supervisionato che consente di raggruppare i CV in base a caratteristiche comuni. Durante questo processo, è stato utilizzato anche il NER per estrarre informazioni chiave, come la professione e la ragione sociale, che sono state integrate nel clustering al fine di migliorare la precisione nella categorizzazione dei curricula.

La tesi è articolata in quattro capitoli. Il primo capitolo offre una panoramica teorica del problema: inizia con l'introduzione alla formulazione matematica

del supervised learning, nel quale rientra il Named Entity Recognition, trattato come un problema di classificazione multiclasse, e prosegue con una descrizione dell'algoritmo *k-means* e delle metriche utilizzate per valutare le prestazioni di un algoritmo di clustering. Il secondo capitolo, intitolato Text Representation, approfondisce le tecniche di rappresentazione del testo, analizzando i metodi tradizionali come one-hot encoding e feature vectors, fino ad arrivare alle moderne tecniche di embedding, che permettono di rappresentare una parola mantenendo il suo significato semantico in base al contesto. Il terzo capitolo, dedicato al Named Entity Recognition, esplora le metodologie più comuni per l'estrazione delle entità, passando dai metodi tradizionali basati su regole fisse, ideali per estrarre entità con una struttura definita (come numeri di telefono o codici fiscali), ai più avanzati approcci basati sul machine learning, che utilizzano le rappresentazioni contestuali delle parole. Infine, nel quarto capitolo vengono presentati i risultati ottenuti applicando le tecniche di anonimizzazione e clustering sul dataset di curricula, descrivendo nel dettaglio il metodo proposto e le performance raggiunte.

Capitolo 1

Machine Learning

1.1 Introduzione al Machine Learning

L'intelligenza artificiale (IA) è un campo dell'informatica dedicato alla creazione di sistemi capaci di eseguire compiti che normalmente richiederebbero l'intelligenza umana. Questi compiti includono mansioni specifiche come il riconoscimento del linguaggio naturale, il riconoscimento di immagini e molto altro. Il nucleo matematico dell'IA è costituito dal Machine Learning (ML) che si occupa della progettazione e dello sviluppo di algoritmi in grado di apprendere da un insieme di esempi invece che essere esplicitamente programmati. Infatti, a differenza dei tradizionali approcci basati su regole fissate a priori, il machine learning permette ai computer di migliorare le loro prestazioni attraverso l'esperienza e l'analisi dei dati. In sostanza, l'obiettivo del Machine Learning è quello di estrarre informazioni utili da grandi quantità di dati, costruendo algoritmi e modelli predittivi. Questo approccio permette ai sistemi informatici di identificare schemi e prendere decisioni autonome senza l'intervento umano.
In base alla tipologia di dati a disposizione, il ML può essere suddiviso in tre principali categorie:

- **Supervised learning** : in questo approccio, gli algoritmi vengono addestrati su un insieme di dati etichettati, ovvero dati per i quali sono già note le risposte corrette. Più precisamente, l'insieme dei dati è costituito da coppie (x_i, y_i) con $i = 1, ..., n$ dove x_i e y_i sono detti input e output rispettivamente. L'obiettivo dell'apprendimento supervisionato è costruire un modello che sia in grado di relazionare gli input agli output corretti, imparando una funzione che possa prevedere l'etichetta corretta per nuovi dati non etichettati. All'interno di questo contesto si possono distinguere due problemi differenti: regressione e classificazione. La differenza tra queste due categorie consiste nella tipologia del dato di output: nella regressione gli output a disposizione sono variabili come potrebbero essere funzioni o vettori. Il termine classificazione invece fa rifermento alla determinazione di una classe. In questo caso gli output da determinare appartengono ad

un insieme finito di elementi che corrispondono alle varie categorie a disposizione. Nel caso di classificazione binaria risulta $y \in \{-1, 1\}$ altrimenti, per quanto riguarda la classificazione multiclasse, $y \in \{1, ..., T\}$ con T il numero delle classi.

- **Unsupervised learning** : in questo caso si ha a disposizione solo un insieme $\{x_1, ..., x_n\}$ di dati non "etichettati" ovvero non sono presenti esempi di risposta (output) che si vogliono ottenere. Un esempio comune di apprendimento non supervisionato è il clustering, dove l'obiettivo è estrarre una rappresentazione efficiente dei dati raggruppandoli in cluster omogenei. Un metodo molto utilizzato per il clustering è l'algoritmo k-means.

- **Semisupervised learning**: rappresenta una situazione intermedia tra le due precedentemente descritte in cui solo una parte dei dati di input presenta gli output corrispondenti (supervised), mentre la restante parte risulta non etichettata (unsupervised). Riassumendo in formule l'insieme dei dati è: $\{(x_i, y_i)\} \cup \{x_{n+1}, ..., x_{n+m}\}$.

Nell'ambito di questa tesi, il problema della classificazione è centrale nel tema nella Named Entity Recognition (NER) ovvero il processo di identificazione e classificazione delle entità nominate in un testo come nomi di persone, organizzazioni, località, date, e altre categorie rilevanti. Il clustering è invece utilizzato per determinare dei gruppi di profili professionali simili in base ai loro curricula. Prima di affrontare le metodologie tipiche per gli algoritmi di classificazione e clustering viene presentato nello specifico il problema dell'apprendimento supervisionato e non supervisionato.

1.2 Supervised Statistical Learning

Si considerino i seguenti insiemi:

- $\mathcal{X}$ spazio di input: l'insieme di tutti i possibili valori che le variabili di input possono assumere. Nel caso considerato, $\mathcal{X} \subseteq \mathbb{R}^d$.
- $\mathcal{Y}$ spazio degli output: l'insieme di tutti i possibili valori che le variabili di output possono assumere. Nel caso considerato $\mathcal{Y} \subseteq \mathbb{R}^t$. In particolare, in base alla tipologia di problema affrontato si ha:
  - $\mathcal{Y} \subseteq \mathbb{R}$ in caso di regressione;
  - $\mathcal{Y} \subseteq \mathbb{R}^t$ in caso di regressione vettoriale;
  - $\mathcal{Y} = \{-1, 1\}$ in caso di classificazione binaria;
  - $\mathcal{Y} = \{1, 2, \ldots, T\}$ in caso di classificazione multiclasse.
- $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ spazio dei dati: l'insieme delle coppie input/output.

Definizione 1.1. Sia $(\mathcal{X}, \mathcal{A})$ uno spazio misurabile e $\mathcal{Y} \subset \mathbb{R}^t$. Allora una funzione $l : \mathcal{Y} \times \mathbb{R} \to [0, \infty)$ si dice *loss function* se è misurabile.

Si riportano ora alcuni esempi di funzioni loss.

Esempio 1.1 (Loss per regressione). Spesso in questo contesto si utilizzano "distance-based loss functions" della forma:

$$l(t, y) = V(t - y)$$

con $V : \mathcal{Y} \longrightarrow [0, \infty)$. Nel caso della regressione si ricordi che $\mathcal{Y} \subseteq \mathbb{R}^t$. Si possono distinguere:

- loss quadratica : $l(t, y) = (t - y)^2$
- absolute loss : $l(t, y) = |t - y|$
- ϵ-insensitive : $l(t, y) = \max\{|y - t| - \epsilon, 0\}$

Esempio 1.2 (Loss per classificazione binaria). Queste funzioni non si basano sulla distanza ma sul prodotto. Si ricordi che in questo caso $\mathcal{Y} = \{-1, 1\}$;di conseguenza, per determinare se t e y hanno lo stesso segno, queste funzioni sono basate sul prodotto fra t e y e non più sulla distanza. Vengono infatti denominate "margin-based loss" per la loro forma:

$$l(t, y) = \sigma(ty)$$

Si possono distinguere:

- loss 0-1 : $l(t, y) = \begin{cases} 1 & se\ ty < 0 \\ 0 & se\ ty > 0 \end{cases}$
- Hinge loss: $l(t, y) = |1 - ty|_+ = \max\{1 - ty, 0\}$

- quadratica: $l(t,y) = (t-y)^2 = y^2(y-t)^2 = (y^2 - yt)^2 = (1-ty)^2$
- logistica: $l(t,y) = \ln(1+e^{-yt})$

Definizione 1.2. Sia $l : \mathcal{Y} \times \mathbb{R} \to [0,\infty)$ una loss function e sia P una misura di probabilità su $\mathcal{X} \times \mathcal{Y}$. Allora, per una funzione misurabile $f : \mathcal{X} \to \mathbb{R}$, il rischio atteso $L(f)$ è definito da:

$$L(f) := \int_{\mathcal{X}\times\mathcal{Y}} l(y, f(x))\, dP(x,y).$$

Osservazione 1.1. Si noti che la funzione $(x,y) \mapsto l(y,f(x))$ è misurabile per le assunzioni su f e, poiché è anche non negativa, l'integrale su $\mathcal{X} \times \mathcal{Y}$ esiste sempre, anche se non è necessariamente finito.

Osservazione 1.2. Il rischio atteso $L(f)$ dipende dalla distribuzione P che nella realtà non è nota.

Definizione 1.3. Sia $l : \mathcal{Y} \times \mathbb{R} \to [0,\infty)$ una loss function, P una misura di probabilità su $\mathcal{X} \times \mathcal{Y}$ e $\mathcal{M} = \{f : \mathcal{X} \to \mathcal{Y} | f \text{ misurabile}\}$. Allora il rischio minimo è definito come:

$$L^* := \min_{f\in\mathcal{M}} L(f).$$

Inoltre, una funzione misurabile $f^* : \mathcal{X} \to \mathbb{R}$ tale che $L(f^*) = L^*$ è chiamata funzione target.

Definizione 1.4. Il problema del supervised learning è il calcolo di:

$$\min_{f\in\mathcal{M}} L(f)$$

avendo accesso solo a $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ con $i = 1, ..., n$.

Osservazione 1.3. Il rischio atteso dipende da P che nella realtà è non nota. Di conseguenza, se i dati $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ con $i = 1, ..., n$ non sono rappresentativi di P, risulta poco chiaro come risolvere il problema di learning.

Definizione 1.5. L'insieme $\mathcal{X}_{\text{train}} = \{(x_1, y_1), ..., (x_n, y_n)\}$ dove $x_i \in \mathcal{X}$ rappresenta l'input e $y_i \in \mathcal{Y}$ l'output corrispondente viene detto training set.

La statistical learning theory assume l'esistenza di un modello probabilistico che permetta di descrivere l'incertezza dei dati a cui si ha accesso.

Siano $\mathcal{X}$ e $\mathcal{Y}$ lo spazio degli input e degli output rispettivamente. Si consideri lo spazio di probabilità $(\Omega, \mathcal{A}, \mathbb{P})$, si definisce la seguente variabile aleatoria:

$$(X,Y) : \Omega \longrightarrow \mathcal{X} \times \mathcal{Y}$$

$$w \mapsto (X(w), Y(w))$$

con legge di probabilità P e si indica con $(X,Y) \sim P$.

La Statistical Learning Theory assume che le coppie (x_i, y_i) siano le realizzazioni di (X_i, Y_i) copie indipendenti di (X,Y).

Osservazione 1.4. Nel caso dell'apprendimento supervisionato, si assume che la legge P di (X, Y) coincida con quella della definizione del rischio atteso.

Definizione 1.6. Una funzione $\mathcal{A}$ che prende in input il training set $\mathcal{X}_{\text{train}}$ e restituisce uno stimatore $\hat{f} : \mathcal{X} \longrightarrow \mathcal{Y}$ viene detto algoritmo di apprendimento.

Osservazione 1.5. Un algoritmo di apprendimento restituisce una stima della funzione target a partire dai dati di training a disposizione.

Definizione 1.7. Si definisce *excess risk* la seguente quantità:

$$L(\hat{f}) - L(f^*)$$

Osservazione 1.6. Lo stimatore $\hat{f}$ è una variabile aleatoria attraverso la sua dipendenza dai dati e conseguentemente l'excess risk risulta una variabile aleatoria reale positiva [8].

Una proprietà fondamentale di un algoritmo di apprendimento è che l'excess risk converga a zero quando il numero di dati n tende all'infinito. Questa proprietà è detta consistenza e può essere definita in diversi modi.

Definizione 1.8. Si dice che l'excess risk converge in probabilità a zero se:

$$\forall P,\ \forall \varepsilon > 0,\ \lim_{n\to\infty} P\big(L(\hat{f}) - L(f^*) \geq \varepsilon\big) = 0$$

Un altro modo per definire la consistenza è tramite la convergenza in media.

Definizione 1.9. Si dice che l'excess risk converge in media a zero se:

$$\forall P\ \lim_{n\to\infty} \mathbb{E}\big[L(\hat{f}) - L(f^*)\big] = 0$$

Osservazione 1.7. Nell'approccio del Supervised Statistical Learning, l'obiettivo è identificare una relazione $f : \mathcal{X} \to \mathcal{Y}$ tale che $f(x)$ sia una buona approssimazione della risposta possibile y per un dato x. La misura utilizzata per valutare la qualità di una risposta stimata $f(x)$ quando la coppia di input e output vera è (x, y) è rappresentata dal valore della loss-function $l(y, f(x))$ che risulta più piccolo quando la risposta stimata $f(x)$ è migliore. Il termine loss, infatti, indica che si è interessati a valori piccoli di l. Per valutare la qualità di una funzione appresa f, non è sufficiente conoscere il valore $l(y, f(x))$ per una particolare scelta di (x, y) e nella teoria dell'apprendimento statistico si considera solitamente il rischio atteso di f. Si considera quindi che una funzione sia migliore quanto più piccolo è il suo rischio atteso $L(f)$. Si vuole quindi trovare una funzione $f : X \to \mathcal{Y}$ che minimizzi (approssimativamente) il rischio $L(f)$ che dipende da P. Nel caso in questione, tuttavia, la distribuzione P è sconosciuta e si ha a disposizione solo $\{(x_1, y_1), \ldots, (x_n, y_n)\}$. A partire da questo insieme di dati, si vuole quindi costruire una funzione $\hat{f} : \mathcal{X} \to \mathcal{Y}$ il cui rischio $L(\hat{f})$ sia vicino al rischio minimo $L^* = L(f^*)$.

1.2.1 ERM regolarizzato

Si ricordi il problema di learning definito in 1.4. Poichè la legge P non è nota e l'unica informazione a disposizione su P consiste nel training set $\{(x_1, y_1), ..., (x_n, y_n)\}$, si devono effettuare alcune approssimazioni:

1. si rimpiazza il rischio atteso $L(f)$ con il rischio empirico $\hat{L}(f)$ definito da:

$$\hat{L}(f) = \frac{1}{n} \sum_{i=1}^{n} l(f(x_i), y_i)$$

2. anzichè minimizzare L su $\mathcal{M}$, si minimizza $\hat{L}$ su $\mathcal{H} \subseteq \mathcal{M}$ spazio di funzioni (detto spazio delle ipotesi) dove f dipende da alcuni parametri $\theta \in \mathbb{R}^p$ ovvero $f = f(x, \theta)$ che devono essere ottimizzati durante il processo di training. Questo rende il problema trattabile dal punto di vista numerico.

Osservazione 1.8. Ci sono tante scelte possibili per $\mathcal{H}$. La più semplice è:

$$\mathcal{H} = \{f : \mathbb{R}^d \longrightarrow \mathbb{R} |\ \exists\, w \in \mathbb{R}^d : f(x) = < w, x >\}$$

oppure

$$\mathcal{H} = \{\text{reti neurali}\}$$

Nella pratica, inoltre, si aggiunge anche un termine di regolarizzazione per evitare l'overfitting. Il problema di learning si riduce quindi a studiare:

$$\min_{\mathcal{H}} \hat{L}(f) + \lambda R(f) =$$

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} l(f(x_i, \theta), y_i) + \lambda R(f) \tag{1.1}$$

Teorema 1.1 (Consistenza rischio empirico regolarizzato)**.** *Supponiamo che la loss function sia di Lipschitz con costante G e convessa rispetto al secondo argomento, e sia $\mathcal{H} = \{f : \mathbb{R}^d \longrightarrow \mathbb{R} |\ \exists\, \theta \in \mathbb{R}^d : f(x) = < \theta, \varphi(x) >$ dove $\|\varphi(x)\|_2 \leq R$ quasi sicuramente$\}$. Sia $\hat{\theta}_\lambda \in \mathbb{R}^d$ il minimizzatore del rischio empirico regolarizzato nell'equazione 1.1; allora*

$$\mathbb{E}\big[L(\hat{\theta}_\lambda)\big] \leq \inf_{\theta \in \mathbb{R}^d} \left\{ L(\theta) + \frac{\lambda}{2} \|\theta\|_2^2 \right\} + \frac{24 G^2 R^2}{\lambda n}.$$

Per la dimostrazione si veda [13].

1.3 Unsupervised learning

1.3.1 Clustering

Il clustering è il processo di suddivisione di un insieme di oggetti in gruppi (cluster) in modo tale che gli oggetti appartenenti allo stesso gruppo siano più simili tra loro rispetto a quelli di gruppi diversi. È una tecnica di unsupervised learning utilizzata nell'analisi esplorativa dei dati per comprendere la loro struttura e identificare gruppi significativi. Valutare l'efficacia di un clustering è complesso, poiché non esiste un criterio univoco per determinare quale sia il 'clustering corretto". Lo stesso dataset può essere raggruppato in modi diversi a seconda degli obiettivi specifici dell'analisi.

In questa sezione, si vuole affrontare il metodo k-means, approfondendone il funzionamento e discutendo alcune metriche utili per valutare la qualità dei risultati di clustering.

1.3.2 K-means

Come già accennato, l'obiettivo del clustering è suddividere un insieme di punti in sottoinsiemi disgiunti, formando gruppi composti da elementi simili tra loro e distinti dai punti di altri cluster.
È fondamentale chiarire cosa si intenda per "similarità" tra gli oggetti. Intuitivamente, un cluster è ritenuto valido quando i punti che ne fanno parte sono tutti vicini a un punto centrale, detto centroide.
Si introduce ora la cosiddetta "cost function" su cui si basa il metodo k-means.

Definizione 1.10. Sia $k \in \mathbb{N}$ e siano $c_1, ..., c_k \in \mathbb{R}^d$. Si definisce la cost function nel seguente modo:

$$\varphi : \mathbb{R}^d \times \cdot \times \mathbb{R}^d \longrightarrow [0, +\infty)$$
$$(c_1, \ldots, c_k) \longmapsto \sum_{x \in X} \min_{1 \leq j \leq k} \|x - c_j\|_2^2$$

I punti $c_1, \ldots, c_k$ vengono chiamati centri ed ogni scelta di centri $c_1, \ldots, c_k$ identifica un clustering definito da:

$$C = (C_1, \ldots, C_k)$$

dove:

$$C_i = \{x \in X : ||x - c_i|| < ||x - c_j|| \quad \forall j \neq i\}$$

Osservazione 1.9. Se un punto del dataset è equidistante da due o più centri, allora viene assegnato in modo casuale a uno dei cluster.

Siano:

- $\mathcal{X} = \{x_1, ..., x_n\} \subset \mathbb{R}^d$ insieme di dati da raggruppare;
- k numero di cluster in cui si vuole suddividere X;
- $c_1, ..., c_k \in \mathbb{R}^d$ k centri;

Definizione 1.11. Il clustering k-means, denotato con C_{OPT}, è quello dato dai centroidi che minimizzano φ:

$$(c^1_{\text{OPT}}, \ldots, c^k_{\text{OPT}}) \in \arg\min_{c_1,\ldots,c_k} \varphi(c_1, \ldots, c_k)$$

I cluster ottimali sono denotati $(C^1_{\text{OPT}}, \ldots, C^k_{\text{OPT}})$.

Si osserva che trovare la soluzione ottimale di k-means è computazionalmente difficile (il problema è NP-hard) [9]. Per risolvere questo problema, viene comunemente utilizzato un semplice algoritmo iterativo, noto come algoritmo di Lloyd, che però non ha garanzia di trovare il clustering ottimale.

1.3.3 Lloyd algorithm

L'algoritmo di Lloyd si definisce nel seguente modo:

- **Input:** $\mathcal{X} \subset \mathbb{R}^d$; numero di cluster k
- **Inizializzazione:** Scegliere casualmente i centroidi iniziali $c^0_1, \ldots, c^0_k$
- per $t = 0, ..., T$
  - **Step 1:** per $i = 1, \ldots, k$, si pone:

$$C^t_i = \{x \in X : d(x, c^t_i) < d(x, c^t_j) \quad \forall j \neq i\}$$

  - **Step 2:** per $i = 1, \ldots, k$, si pone $c^{t+1}_i = \frac{1}{|C^t_i|} \sum_{x \in C^t_i} x$ come il centro di massa di C_i;
- **Return:** $C^T_1, \ldots, C^T_k$

Teorema 1.2. *L'algoritmo di Lloyd termina in un numero finito di passi.*

Prima di procedere con la dimostrazione, si vuole osservare un fatto fondamentale che sarà utile per la dimostrazione del teorema.

Osservazione 1.10. Sia $C \subset \mathbb{R}^d$ un insieme finito di punti, e μ il suo centro di massa:

$$\mu = \frac{1}{|C|} \sum_{x \in C} x$$

Allora, per ogni punto $c \in \mathbb{R}^d$, si ha l'identità:

$$\sum_{x \in C} \|x - c\|_2^2 = \sum_{x \in C} \|x - \mu\|_2^2 + |C| \cdot \|c - \mu\|_2^2$$

Dimostrazione. Si definisce la funzione obiettivo:

$$\phi(c_1, \ldots, c_k, C_1, \ldots, C_k) = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - c_i\|_2^2$$

Si vuole mostrare che, quando i centri vengono aggiornati, allora ϕ decresce strettamente, e ciò può avvenire al massimo k^n volte (il numero di raggruppamenti possibili).

Si ricordino i due step dell'algoritmo definiti in 1.3.3 e si denoti lo "stato" dell'algoritmo come segue:

- $c_1^0, \ldots, c_k^0, C_1^0, \ldots, C_k^0$ all'inizio dell'iterazione.
- $c_1^0, \ldots, c_k^0, C_1^1, \ldots, C_k^1$ dopo il primo step.
- $c_1^1, \ldots, c_k^1, C_1^1, \ldots, C_k^1$ dopo il secondo step.

Si osservi che:

$$\phi(c_1^0, \ldots, c_k^0, C_1^0, \ldots, C_k^0) \geq \phi(c_1^0, \ldots, c_k^0, C_1^1, \ldots, C_k^1)$$

Questo è vero per definizione di $C_1^1, \ldots, C_k^1$.

Si vuole ora provare che, $\forall (c_1, ..., c_k)$ tale che $\exists i$ con $c_i' \neq c_i$ si ha:

$$\phi(c_1, \ldots, c_k, C_1', \ldots, C_k') > \phi(c_1', \ldots, c_k', C_1', \ldots, C_k')$$

Consideriamo un i tale che $c_i' \neq c_i$ e si ricordi che c_i' è il centro di massa di C_i'. Per l'osservazione 1.10,

$$\sum_{x \in C_i'} \|x - c_i\|_2^2 = \sum_{x \in C_i'} \|x - c_i'\|_2^2 + |C_i'| \cdot \|c_i - c_i'\|_2^2 > \sum_{x \in C_i'} \|x - c_i'\|_2^2$$

Per la definizione di φ ,sommando su tutti i cluster si ottiene che ϕ decresce strettamente.

Inoltre, si osserva che ϕ può decrescere solo un numero finito di volte. Infatti, ϕ è una funzione dell'attuale partizionamento $C_1, \ldots, C_k$, che può assumere al

massimo k^n valori distinti. Pertanto, se l'algoritmo eseguisse più di k^n iterazioni, ripeterebbe lo stesso partizionamento due volte. Questo implica che ϕ assume lo stesso valore in due iterazioni distinte (escludendo l'ultima). Ciò è assurdo, poiché ϕ diminuisce sempre [9].

Questo completa la dimostrazione del teorema.

□

Osservazione 1.11. Il fatto che ϕ diminuisca indica che si ottiene un clustering migliore di quello da cui siamo partiti.

Nell'algoritmo k-means, i centroidi iniziali vengono scelti casualmente dal dataset, il che può influire significativamente sulla convergenza.

Per migliorare l'efficienza e la qualità delle soluzioni trovate, è possibile utilizzare metodi alternativi di inizializzazione, come k-means++. Questo approccio, introdotto da Arthur e Vassilvitskii (2007) in [14] ,sceglie i centri iniziali casuali con probabilità molto specifiche. In particolare, si sceglie un punto c come centro con una probabilità proporzionale al contributo di tale punto c al potenziale complessivo. Per ulteriore dettagli su questa particolare scelta si veda [14].

1.4 Metriche per valutare clustering

Le metriche per la valutazione di un algoritmo di clustering si dividono in:

- esterne;
- interne.

Le metriche esterne si utilizzano quando è disponibile una Ground Truth, cioè quando si hanno le etichette reali dei dati. Al contrario, le metriche interne si impiegano quando non si dispone di una partizione reale degli oggetti.

Indipendentemente dalla presenza di una Ground Truth, è importante ricordare che gli algoritmi di clustering appartengono alla categoria dell'apprendimento non supervisionato (unsupervised learning). In questo caso, si ha a disposizione soltanto l'insieme dei dati di input $x_1, ..., x_n$, che non sono etichettati.

1.4.1 Metriche esterne

Come precedentemente accennato, le metriche esterne si basano sul confronto tra le etichette predette dal modello di clustering e le etichette "reali" ovvero quelli della Ground Truth. Le metriche esterne hanno infatti lo scopo di verificare quanto la partizione ottenuta dall'algoritmo di clustering e la partizione reale della Ground Truth siano simili.

Alcune metriche esterne sono le seguenti:

- Adjusted Rand Index;
- Fowlkes-Mallows Index;
- Homogeneity;
- Completeness;
- V-measure.

Adjusted Rand Index

Definizione 1.12. Dato $\mathcal{X} = \{x_1, ..., x_N\}$ l'insieme dei dati di input e due partizioni di X:

- $C = \{C_i | i = 1, ..., n\}$ corrispondente alle etichette reali;
- $K = \{K_i | i = 1, ..., m\}$ rappresentante i cluster generati dall'algoritmo.

si definiscono le seguenti quantità:

- $a = \#\{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} | x_i, x_j \in C_k \text{ e } x_i, x_j \in K_p \text{ per qualche } k \in \{1, ..., n\} \text{ e } p \in \{1, ..., m\}\}$.
- $b = \#\{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} | x_i \in C_k, x_j \in C_q \text{ e } x_i \in K_l, x_j \in K_p \text{ per qualche } q, k \in \{1, ..., n\} \text{ con } q \neq k \text{ e } l, p \in \{1, ..., m\} \text{ con } l \neq p\}$

- $c = \#\{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} | x_i, x_j \in C_k \text{ e } x_i \in K_l, x_j \in K_p$ per qualche $k \in \{1, ..., r\}$ e $l, p \in \{1, ..., s\}$ con $l \neq p\}$
- $d = \#\{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} | x_i \in C_k, x_j \in C_q \text{ e } x_i, x_j \in K_p$ per qualche $q, k \in \{1, ..., r\}$ con $q \neq k$ e $p \in \{1, ..., s\}\}$

Il Rand Index è definito come segue:

$$RI = \frac{a+b}{a+b+c+d}$$

Osservazione 1.12. Si osservi che:

- $a = TP$ True Positive : il numero di coppie di punti che sono raggruppati insieme sia nella partizione reale (Ground Truth) che in quella predetta.
- $b = TN$ True Negative: il numero di coppie di punti che non sono raggruppati insieme né nella partizione reale (Ground Truth) né in quella predetta.
- $c = FN$ False Negative: il numero di coppie di punti che sono raggruppati insieme nella partizione vera ma non in quella predetta.
- $d = FP$ False Positive: il numero di coppie di punti che sono raggruppati insieme nella partizione predetta ma non in quella vera.

Osservazione 1.13. L'indice Rand ha un valore compreso tra 0 e 1, dove 0 indica che le due partizioni non concordano su nessuna coppia di punti mentre 1 indica che sono esattamente uguali.

Osservazione 1.14. Intuitivamente, il numeratore $a + b$ può essere considerato come il numero di accordi tra C e K mentre il valore $c + d$ come il numero di disaccordi tra C e K. Inoltre, poiché il denominatore è il numero totale di coppie, l'indice Rand rappresenta la frequenza di accordi sul totale delle coppie, ovvero la probabilità che C e K concordino su una coppia casuale.

Definizione 1.13. Siano $\mathcal{X}, C, K$ della definizione 1.12. La tabella di contingenza è definita come segue:

	K_1	K_2	$\cdots$	K_m	sums
C_1	n_{11}	n_{12}	$\cdots$	n_{1m}	a_1
C_2	n_{21}	n_{22}	$\cdots$	n_{2m}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
C_n	n_{n1}	n_{n2}	$\cdots$	n_{nm}	a_n
sums	b_1	b_2	$\cdots$	b_m	

dove $n_{ij} = |C_i \cap K_j|$.

L'Adjusted Rand Index si definisce come segue:

$$\text{ARI} = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

Osservazione 1.15. La tabella di contingenza riassume la sovrapposizione tra le due partizioni C e K.

Fowlkes-Mallows

Definizione 1.14. L'indice di Fowlkes-Mallows è definito come la media geometrica [1] di Precision e Recall:

$$FM = \frac{a}{\sqrt{(a+d)(a+c)}}$$

dove a, c, d sono definiti in 1.12.

Osservazione 1.16. Il punteggio varia da 0 a 1. Un valore elevato indica una buona somiglianza tra le due partizioni (clustering) [7].

Homogeneity

Un risultato di clustering soddisfa il criterio di omogeneità quando ciascun cluster contiene esclusivamente punti appartenenti a una singola classe. In altre parole, un clustering si considera omogeneo se la distribuzione delle classi all'interno di ogni cluster mostra una netta predominanza di una sola classe, idealmente una sola. In questo caso, si dice ad "entropia zero". [6].

Definizione 1.15. Sia $\mathcal{X}, C, K$ come nella definizione 1.12 e sia $A = \{a_{i,j}\}$ la tabella di contingenza dove $a_{i,j} = |C_i \cap K_j|$. L'Homogeneity è definita come segue:

$$\text{Homogeneity} = \begin{cases} 1 & \text{se } H(C) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & \text{altrimenti} \end{cases}$$

dove:

- $H(C|K) = -\sum_{k=1}^{|K|} \sum_{c=1}^{|C|} \frac{a_{ck}}{N} \log \frac{a_{ck}}{\sum_{c=1}^{|C|} a_{ck}}$.
- $H(C) = -\sum_{c=1}^{|C|} \frac{\sum_{k=1}^{|K|} a_{ck}}{n} \log \frac{\sum_{k=1}^{|K|} a_{ck}}{n}$

Osservazione 1.17. Si osservi che:

- $H(C|K)$ rappresenta l'entropia condizionale delle classi C date le etichette di cluster K. Questa misura quanto sono distribuite le classi nei cluster. Questo valore è 0 quando ogni cluster contiene solo membri di una singola classe ovvero nel caso di un clustering perfettamente omogeneo. Di conseguenza, in questo caso homogeneity = 1.
- $H(C)$ coincide con massima riduzione dell'entropia che le informazioni di clustering possono fornire [6]. $H(C) = 0$ corrisponde al caso in cui c'è solo un'unica classe.

[1] Dati $\{x_1, ..., x_n\}$ si definisce la media geometrica come $\left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \sqrt[n]{x_1 x_2 \dots x_n}$

Completeness

Un risultato di clustering soddisfa la completezza se tutti i punti che appartengono a una determinata classe sono elementi dello stesso cluster. In altre parole, un clustering completo deve garantire che ogni classe di dati sia rappresentata integralmente in un unico cluster. Per valutare la completezza, si esamina la distribuzione delle assegnazioni dei cluster K all'interno di ciascuna classe C, al contrario invece di qunato accadeva nel caso dell'homogeneity dove si misurava la distribuzione delle classi C fissati i cluster K.

Definizione 1.16. Siano $\mathcal{X}, C, K$ come nella definizione precedente. L'indice "completeness" viene definito come segue [6]:

$$\text{Completeness} = \begin{cases} 1 & \text{se } H(K) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & \text{altrimenti} \end{cases}$$

dove:

- $H(K|C) = -\sum_{c=1}^{|C|} \sum_{k=1}^{|K|} \frac{a_{ck}}{N} \log\left(\frac{a_{ck}}{\sum_{k=1}^{|K|} a_{ck}}\right)$
- $H(K) = -\sum_{k=1}^{|K|} \frac{\sum_{c=1}^{|C|} a_{ck}}{n} \log\left(\frac{\sum_{c=1}^{|C|} a_{ck}}{n}\right)$

Osservazione 1.18. Si osservi che:

- $H(K|C)$ rappresenta l'entropia condizionale della distribuzione dei cluster predetti, date le classi dei punti. Nel caso di perfetta "completeness" questo valore è pari a zero. Nel caso peggiore ovvero quando ogni classe è rappresentata da ogni cluster con una distribuzione uguale alla distribuzione delle dimensioni dei cluster, $H(K|C)$ è massima ed è uguale a $H(K)$.
- $H(K) = 0$ corrisponde al caso in cui c'è solo un singolo cluster.

V-measure

Definizione 1.17. Il valore di *v-measure* si calcola come la media armonica [2] di *Homogeneity* e *Completeness*.

Più precisamente:

$$\text{v-measure} = 2 \times \frac{\text{Homogeneity} \times \text{Completeness}}{\text{Homogeneity} + \text{Completeness}}$$

[2] La media armonica di n numeri reali positivi $x_1, ..., x_n$ si definisce come $\frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$

1.4.2 Metriche interne

Le metriche interne, come precedentemente accennato, si utilizzano per valutare le perforance di un algoritmo di clustering quando non si ha a disposizione la "Ground Truth". La valutazione del clustering in questo caso di basa su due concetti contrastanti:

- *coesione interna* o *compattezza*: entità nello stesso cluster devono essere simili le une alle altre;
- *isolamento esterno* o *separazione*: entità molto dissimili devono collocarsi in cluster diversi.

Alcune metriche interne sono:

- inerzia;
- Silhouette score;
- Davies-Bouldin index;
- Calinski-Harabasz.

Osservazione 1.19. In questa sezione è disponibile solo la partizione predetta dall'algoritmo di clustering che verrà ora indicata con la notazione $C = \{C_1, ..., C_k\}$ che precedentemente indicava quella della partizione reale.

Inerzia

Definizione 1.18. Sia $\mathcal{X} = \{x_1, ..., x_N\}$ l'insieme dei dati di input suddivisi in k cluster $C_1, C_2, ..., C_k$ con rispettivi centroidi $\mu_1, ..., \mu_k$. L'inerzia o *Sum of Squared Errors* (SSE) misura la somma delle distanze quadrate tra ciascun punto e il suo centroide assegnato. In formule:

$$\text{SSE} = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - \mu_j\|^2$$

Osservazione 1.20. SSE coincide con la ϕ definita nel teorema 1.2.

Un valore più basso di SSE indica che i punti dati all'interno di ciascun cluster sono più vicini ai rispettivi centroidi, suggerendo una soluzione di clustering migliore.

Silhouette Coefficient

Definizione 1.19. Sia $\mathcal{X} = \{x_1, ..., x_N\}$ l'insieme dei dati di input suddivisi in k cluster $C_1, C_2, ..., C_k$. Per ogni $x_i \in C_i$ si definiscono:

- $a_i = \frac{1}{|C_i|-1} \sum_{x_j \in C_i, j \neq i} ||x_i - x_j||$
- $b_i = \min_{k \neq i} \frac{1}{|C_k|} \sum_{x_j \in C_k} ||x_i - x_j||$

Osservazione 1.21. Si osservi che :

- a_i coincide con distanza media tra x_i e tutti gli altri punti della stessa classe C_i;
- b_i coincide con la distanza media tra x_i e tutti gli altri punti nel cluster più vicino.

Definizione 1.20. Il Silhouette Score per un singolo punto $x_i \in \mathcal{X}$ si definisce come:

$$S_i = \frac{b_i - a_i}{\max\{a_i, b_i\}}$$

Il Coefficiente di Silhouette per un insieme di campioni è dato dalla media del Coefficiente di Silhouette relativo ai singoli campioni [7]. In formule:

$$\text{Silhouette Coefficient} = \frac{1}{N} \sum_{i=1}^{N} S_i$$

Osservazione 1.22. Questa metrica misura la somiglianza e la differenza di ogni punto dati rispetto al proprio cluster e a tutti gli altri cluster. I valori di questa metrica variano da -1 a $+1$. Un valore elevato indica un clustering migliore.

Davies-Bouldin Index

Definizione 1.21. Sia $C = \{C_1, ...C_k\}$ l'insieme dei cluster ottenuti dall'algoritmo e siano c_i con $i = 1, ..., k$ i rispettivi centroidi. Si definisce il diametro del cluster C_i con $i = 1, ..., k$ la seguente quantità:

$$s_i = \frac{1}{|C_i|} \sum_{x \in C_i} ||x - c_i||$$

Osservazione 1.23. Il diametro del cluster C_i è definito come la distanza media tra ciascun punto nel cluster C_i e il centroide c_i.

L'indice di Davies-Bouldin è definito come la similarità media tra ciascun cluster C_i per $i = 1, ..., k$ e il suo cluster più simile C_j [7].

Nel contesto di questo indice, la similarità R_{ij} è definita come una misura che bilancia:

- s_i, il diamentro del cluster C_i;
- $d_{ij} = ||c_i - c_j||$, la distanza tra i centroidi dei cluster i e j

Definizione 1.22. La similarità R_{ij} è definita come:

$$\frac{s_i + s_j}{d_{ij}}$$

Definizione 1.23. Sia R_{ij} la similarità definita in 1.22. L'indice di Davies-Bouldin si definisce come segue:

$$\text{Davies-Bouldin} = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij}$$

Osservazione 1.24. Un valore più basso dell'indice di Davies-Bouldin è associato a un modello con una migliore separazione tra i cluster. Zero è il punteggio più basso possibile. Valori più vicini a zero indicano una migliore partizione.

Calinski-Harabasz Index

L'indice di *Calinski-Harabasz*, noto anche come *Variance Ratio Criterion*, può essere utilizzato per valutare il modello di clustering.

Definizione 1.24. Sia $\mathcal{X} = \{x_1, ..., x_N\}$ l'insieme dei dati di input suddivisi in k cluster $C_1, C_2, ..., C_k$ e $c_1 ..., c_k$ i rispettivi centroidi. L'inidce Calinski-Harabasz si definisce come segue:

$$CH = \frac{\text{tr}(B_K)}{\text{tr}(W_K)} \cdot \frac{N - k}{k - 1}$$

dove:

- $\text{Tr}(B_K)$ è la traccia della matrice di dispersione inter-cluster, definita come:
$$B_K = \sum_{q=1}^{k} n_q (c_q - c_X)(c_q - c_X)^T$$
con n_q numero di elementi nel cluster q e C_X centro di X.
- $\text{Tr}(W_K)$ è la traccia della matrice di dispersione intra-cluster, definita come:
$$W_K = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T$$

Un valore più alto di CH indica cluster più ben definiti, con una maggiore separazione tra i cluster e una coesione interna più forte [7].

Capitolo 2

Text representation

Uno dei campi più rilevanti dell'Intelligenza Artificiale negli ultimi anni è l'elaborazione del linguaggio naturale, anche noto come "Natural Language Processing", abbreviato NLP. L'obiettivo principale è consentire ai computer di comprendere, interpretare e generare il linguaggio naturale in modo efficace. Il punto di partenza per gli algoritmi di NLP sono i dati testuali. Di conseguenza, è necessario convertire queste informazioni in un formato che il computer possa processare. Esistono diverse tecniche di codifica che permettono di trasformare i dati testuali in una rappresentazione comprensibile per i computer. La trasformazione dei dati testuali può avvenire a livello di parola, a livello di frase o a livello di documento in base a quale applicazione si è interessati. Prima di procedere con tali argomenti, verranno introdotti alcuni concetti fondamentali nell'ambito del Natural Language Processing.

2.1 Strumenti di NLP utili

Prima di poter applicare un qualsiasi algoritmo di NLP (in questo caso di NER) su un testo , esso deve essere normalizzato. Normalizzare un testo significa convertirlo in una forma più conveniente ed adatta per eseguirne l'analisi e l'elaborazione [1].

Alcuni compiti che rientrano comunemente nel processo di normalizzazione sono i seguenti:

- tokenizzazione;
- chunking;
- lemmatizzazione;
- POS tagging;
- dependency parser.

Si vuole ora formalizzare questi concetti.

Definizione 2.1. Un testo T da processare è una stringa ovvero una sequenza ordinata di caratteri $[c_1, ..., c_{|T|}]$ dove $|T|$ è la lunghezza del testo corrispondente al numero totale di caratteri presenti.

2.1.1 Tokenizzazione

La tokenizzazione è il processo di suddivisione di un testo in unità più piccole, chiamate "token". Formalmente, dato un testo T , la tokenizzazione restituisce una sequenza ordinata di token $< t_1 >, ..., < t_n >$ dove ciascun token t_j è una sottostringa di T.

I token possono essere:

- parole;
- sotto-parole (in questo caso i token corrispondono ad unità più piccole rispetto alle parole intere, come prefissi, radici e suffissi);
- caratteri.

Il numero di token n in cui viene suddiviso un testo dipende quindi dalla scelta della tipologia di token da utilizzare.

Osservazione 2.1. La tokenizzazione a livello di parola è quella più utilizzata. Quella a livello di sottoparola viene spesso utilizzata in modelli di deep learning per apprendere rappresentazioni delle parole (embedding) che verranno discussi in seguito. Infine l'approccio basato sulla suddivisione in caratteri viene utilizzato in alcune applicazioni speciali come il processamento di lingue complesse e basate su simboli come ad esempio il cinese.

2.1.2 Chunking

Il chunking è una tecnica che permette di suddividere un testo in blocchi di testo, detti chunk, più piccoli. Formalmente, l'obiettivo del chunking è suddividere un testo T in m chunk $C_1, ..., C_m$ dove ogni chunk C_j è una sottostringa di T e $|C_j| << |T| \ \ \forall j$.

I chunk possono essere:

- paragrafi;
- singole frasi;

Nella creazione dei chunk si possono specificare alcuni parametri (dipendenti dalla tipologia di applicazione in questione) come:

- dimensione massima del chunk;
- sovrapposizione, ovvero il numero di caratteri che si possono sovrapporre tra un chunk e quello successivo;
- separatori, ovvero caratteri di punteggiatura (".","!","?") che indicano i punti in cui un testo può essere frammentato.

In alcune applicazioni, oltre ai parametri sopra elencati, è necessario garantire la coerenza semantica, suddividendo il testo in chunk che riflettano concetti correlati e che rispettino i cambi di argomento.

Osservazione 2.2. Il chunking è particolarmente utile per gestire testi lunghi, poiché molti modelli di NLP, (compresi i modelli di NER) hanno un limite sul numero di token in input (ad esempio 512 o 1024 token). In questi casi, i testi devono essere segmentati in chunk, consentendo l'applicazione degli algoritmi su ciascun blocco ed evitando così la perdita di informazioni che altrimenti verrebbero troncate oltre il limite massimo di token.

2.1.3 Tagging delle Parti del Discorso (POS Tagging)

Il part-of-speech tagging è il processo di assegnazione di una categoria grammaticale (anche detta Part-Of-Speech) a ogni parola (token) presente nel testo.

Definizione 2.2. Si definisce la funzione POS nel seguente modo:

$$POS : T \longrightarrow P$$

$$t \longmapsto POS(t) = p$$

dove:

- $T = \{\text{t}|\ \text{ t token}\}$ l'insieme di tutti i token;
- $P = \{\text{p}|\ \text{ p parte del discorso}\}$ è l'insieme delle etichette grammaticali. Ad esempio p può essere nome, aggettivo, verbo...

Un algoritmo di POS tagging prende in input una sequenza di token $< t_1 >, ..., < t_n >$ e restituisce una sequenza di pari dimensioni

$$(POS(t_1), ..., POS(t_n))$$

L'algoritmo di POS tagging associa a ogni token t_i una specifica etichetta $POS(t_i)$ che indica la categoria grammaticale del token all'interno della frase.

2.1.4 Lemmatizzazione

La lemmatizzazione è un processo che riduce una parola alla sua forma base o lemma.

Definizione 2.3. Si definisce una funzione LEMMA:

$$\text{LEMMA} : T \longrightarrow L$$

$$t \longmapsto \text{LEMMA}(t) = l$$

dove:

- $T = \{t \mid t \text{ token}\}$ l'insieme di tutti i token;

- $L = \{l \mid l \text{ lemma}\}$ è l'insieme dei lemmi delle parole.

Il processo di lemmatizzazione prende in input una sequenza di token

$$< t_1 >, \ldots, < t_n >$$

e restituisce una sequenza di pari dimensioni

$$(\text{LEMMA}(t_1), \ldots, \text{LEMMA}(t_n)$$

ovvero associa a ciascun token t_j il suo lemma, basato sulla radice comune delle varianti morfologiche.

Esempio 2.1. Le parole am, are e is vengono associate al lemma comune be.

Osservazione 2.3. L'obiettivo della lemmatizzazione è trattare forme morfologicamente diverse di una stessa parola in modo uniforme, riconducendole a una forma comune. Questo processo permette di determinare se parole con differenze superficiali condividono la stessa radice.

2.1.5 Dependency parsing

Il dependency parsing (analisi delle dipendenze) è una tecnica che analizza le relazioni di dipendenza tra le parole di una frase come, per esempio, le relazioni sintattiche tra soggetto-oggetto.

Tra ogni parola di una frase esistono delle connessioni che formano la cosiddetta impalcatura della frase. Attraverso il dependency parser, la struttura sintattica di una frase viene descritta esclusivamente in termini di relazioni grammaticali binarie dirette tra le parole. Questi legami vengono rappresentati in una struttura ad albero in cui ogni parola dipende da un'altra parola detta head. [1]

2.2 Word representation

In questa sezione verrà introdotto il concetto di word representation (rappresentazione delle parole). Inizialmente verrà descritta la codifica one-hot-enconding basata sulla posizione delle parole all'interno di un vocabolario. Tuttavia, si osserverà subito che essa presenta significative limitazioni, come l'elevata dimensionalità, la sparsità e l'incapacità di cogliere la semantica delle parole. Per superare tali ostacoli, verranno quindi introdotte altre tecniche di rappresentazione come i feature vectors e le rappresentazioni distribuite o vettori semantici (anche noti come embeddings), che mirano a ottenere una rappresentazione vettoriale delle parole in grado di mantenere il loro significato semantico.

2.2.1 One-hot-encoding

Definizione 2.4. Sia $V = \{w_1, ..., w_{|V|}\}$ un vocabolario che rappresenta l'insieme di tutte le parole uniche presenti nel corpus di documenti da analizzare. La rappresentazione One-Hot-encoding per una parola w_i è il vettore v_{w_i} di n componenti con $n = |V|$ e:

$$(v_{w_i})_j = \begin{cases} 1 & i = j \\ 0 & \text{altrimenti} \end{cases}$$

Esempio 2.2. Si consideri il seguente testo: Sara gioca a tennis. Francesca gioca a pallavolo e a basket . Il vocabolario risulta essere :

$$V = \{\text{a, basket, e, Francesca, gioca, pallavolo, Sara, tennis}\}$$

Il vocabolario V ha cardinalità pari a 8, pertanto, ogni token verrà codificato con un vettore di dimensione 8. Le codifiche risultanti sono le seguenti:

$$Sara \longrightarrow (0,0,0,0,0,0,1,0)$$

$$gioca \longrightarrow (0,0,0,0,1,0,0,0)$$

$$a \longrightarrow (1,0,0,0,0,0,0,0)$$

$$tennis \longrightarrow (0,0,0,0,0,0,0,1)$$

$$Francesca \longrightarrow (0,0,0,1,0,0,0,0)$$

$$gioca \longrightarrow (0,0,0,0,1,0,0,0)$$

$$a \longrightarrow (1,0,0,0,0,0,0,0)$$

$$pallavolo \longrightarrow (0,0,0,0,0,1,0,0)$$

$$e \longrightarrow (0,0,1,0,0,0,0,0)$$

$$a \longrightarrow (1,0,0,0,0,0,0,0)$$

$$basket \longrightarrow (0,1,0,0,0,0,0,0)$$

Osservazione 2.4. Si noti che, a scopo illustrativo, in questo esempio il vocabolario ha dimensioni ridotte; tuttavia, generalmente contiene dalle 10.000 alle 50.000 parole.

Osservazione 2.5. La rappresentazione one-hot encoding presenta delle notevoli limitazioni. In primo luogo, la rappresentazione one-hot non è in grado di catturare la correlazione semantica tra le parole. In secondo luogo, si tratta di una rappresentazione sparsa ad alta dimensione, molto inefficiente. In terzo luogo, la rappresentazione one-hot non è flessibile per gestire nuove parole, il che richiede l'assegnazione di nuovi indici per le nuove parole e cambierebbe le dimensioni della rappresentazione [2]

Questa rappresentazione, inoltre, non considera le caratteristiche che distinguono una parola dall'altra, aspetto che invece si cerca di cogliere tramite i feature vectors descritti nella sezione successiva.

2.2.2 Feature vector representation

In questo approccio le parole vengono codificate attraverso un vettore di features ovvero caratteristiche/attributi accuratamente progettate per rappresentare ogni parola. Il *features vector* è costituito da valori booleani, numerici o nominali che catturano diverse caratteristiche della parola da rappresentare. Di seguito una spiegazione formale:

Definizione 2.5. Data una parola w, la rappresentazione mediante il features vector si definisce come un vettore:

$$v_w = (f_1(w), ..., f_n(w))$$

dove:

- n è il numero totale di features scelte per descrivere la parola.
- $f_i(w)$ rappresenta il valore associato alla i-esima caratteristica. Possono essere valori booleani, numerici o nominali. Le componenti $f_i(w)$ possono includere caratteristiche:
  - lessicali: ad esempio

$$f(w) = \text{lunghezza}(w)$$

$$f(w) = \begin{cases} True & \text{se } w \text{ è maiuscola} \\ False & \text{altrimenti} \end{cases}$$

  - morfologiche : ad esempio specificano la categoria grammaticale (Part of Speech) o il lemma:

$$f(w) = \text{POS}(w)$$

$$f(w) = \text{LEMMA}(w)$$

– semantiche che indicano per esempio la presenza in dizionari:

$$f(w) = \begin{cases} 1 & \text{se } w \text{ è presente nel dizionario} \\ 0 & \text{altrimenti} \end{cases}$$

– statistiche come frequenza di occorenza nel documento:

$$f(w) = \frac{n}{T}$$

dove n indica il numero di occorrenze della parola w nel documento e T è il numero totale di parola presenti nel documento.

I feature vector offrono un'interpretazione chiara, poiché le features che li compongono sono comprensibili e possono quindi aiutare a interpretare le decisioni del modello in cui vengono utilizzati.

Inoltre, possono essere personalizzati in base al compito specifico, come nel caso del Named Entity Recognition (NER) in cui si selezionano sottogruppi delle caratteristiche $f_i(w)$ che sono più rilevanti. Ad esempio, il vettore di features utilizzato per il NER può enfatizzare caratteristiche morfologiche e semantiche, come il POS e la presenza in dizionari. Tuttavia, questo approccio richiede una progettazione accurata delle features (feature engineering manuale), un processo spesso lungo e che necessita di competenze specifiche nel dominio. Per questi motivi, si introducono ora le cosiddette *rappresentazioni distribuite* (embedding).

2.2.3 Rappresentazioni distribuite

Le rappresentazioni descritte in questa sezione si basano sulla *distributional hypothesis*, secondo cui le parole che appaiono in contesti simili tendono ad avere significati simili [1]. Più precisamente, due parole che si trovano in distribuzioni molto simili (ovvero le cui parole vicine sono simili) hanno significati simili. L'idea della semantica vettoriale è quella di rappresentare una parola (detta target) come un vettore numerico in uno spazio semantico multidimensionale appreso direttamente dalle distribuzioni delle parole vicine alla parola target. Questi vettori numerici rappresentano i dati in uno spazio continuo e ridotto, preservando le relazioni semantiche tra le parole.

Queste rappresentazioni delle parole sono un esempio di representation learning, ovvero di apprendimento automatico di rappresentazioni utili del testo. Questo costituisce un grande vantaggio, poiché consente di evitare la feature engineering fondamentale invece nei feature vectors. Esse vengono utilizzate in tutte le applicazioni di NLP che richiedono l'interpretazione del significato.
In questa sezione verranno presentate due tipologie di rappresentazioni distribuite:

- vettori di co-occorrenza;
- embeddings.

Vettori di co-occorrenza

I modelli vettoriali presentati qui si basano sulle cosiddette matrici di co-occorrenza. Verranno analizzate due tipi di matrici di co-occorrenza: la **term-term matrix** e la **term-document matrix** che danno origine a due diverse rappresentazioni delle parole.

Nel primo caso la matrice di co-occorrenza "term-term matrix" fornisce una rappresentazione vettoriale di ciascuna parola basata sulle sue co-occorrenze con altre parole nel vocabolario. Di seguito una spiegazione formale.

Definizione 2.6. Sia V il vocabolario del corpus. Una term-term matrix anche detta word-word matrix è una matrice $C \in \mathbb{R}^{|V|\times|V|}$ dove $C_{i,j}$ rappresenta il numero di volte in cui la parola w_i co-occorre con la parola w_j in un determinato contesto. Le entrate vengono calcolate su un corpus di addestramento.

Il contesto può essere:

- documento: in questo caso $C_{i,j}$ rappresenta il numero di volte in cui w_i e w_j appaiono nello stesso documento;
- finestra di contesto di k parole: in questo caso $C_{i,j}$ rappresenta il numero di volte in cui w_j appare entro una finestra di $\pm k$ parole intorno alla parola target w_i.

Ogni parola $w_i \in V$ è quindi rappresentata da un vettore $v_{w_i} \in \mathbf{R}^{|V|}$ corrispondente alla riga i-esima di C che contiene le frequenze di co-occorrenza della parola target w_i con tutte le altre parole del vocabolario in un determinato contesto.

Esempio 2.3. Si mostra una porzione della word-word matrix per quattro parole del corpus di Wikipedia utilizzando come contesto una finestra di ± 4 parole.

	aardvark	...	computer	data	result	pie	sugar
cherry	0	...	2	8	9	442	25
strawberry	0	...	0	0	1	60	19
digital	0	...	1670	1683	85	5	4
information	0	...	3325	3982	378	5	13

Tabella 2.1: Word-word sub-matrix

I vettori di co-occorrenza delle quattro parole corrispondono alle corrispondenti righe di tale matrice. Per esempio, la parola *cherry* viene rappresentata dal seguente vettore:

$$cherry = (0, \ldots, 2, 8, 9, 442, 25)$$

Si noti che vengono riportate solo sei delle dimensioni mentre un vettore reale avrebbe molte più dimensioni e quindi sarebbe molto più sparso.

Osservazione 2.6. Questo metodo dà luogo a vettori molto lunghi. Si noti, infatti, che la dimensionalità del vettore $|V|$ corrisponde alla dimensione del

vocabolario, spesso compresa tra 10.000 e 50.000 parole. Inoltre, dato che la maggior parte delle parole non compare mai nel contesto di altre, la maggior parte di questi numeri è pari a zero e si tratta quindi di vettori sparsi.

Un altro tipo di rappresentazione basata su matrici di co-occorrenza utilizza invece la term-document matrix.

Definizione 2.7. Sia V il vocabolario del corpus e D l'insieme dei documenti nel corpus. La term-document matrix è una matrice $T \in \mathbb{R}^{|V|\times|D|}$ in cui l'i-esima riga rappresenta una parola target $w_i \in V$ mentre la j-esima colonna rappresenta un documento $d_j \in D$. L'entrata $T_{i,j}$ della matrice rappresenta il numero di volte in cui la parola w_i appare nel documento d_j.
In questa rappresentazione ogni parola w_i è descritta da un vettore riga $v_{w_i} \in \mathbb{R}^{|D|}$, che corrisponde alla i-esima riga della matrice T.

Osservazione 2.7. Questa matrice term-document, letta per colonne può essere utilizzata per la rappresentazione di documenti e non di parole come vedremo nella sezione 2.3.

Le entrate della term-term matrix e della term-document matrix rappresentano le frequenze delle parole rispetto ad altre parole e ai documenti, rispettivamente. Tuttavia, la frequenza non è la misura più efficace per valutare l'associazione tra parole, poiché la frequenza grezza tende a essere distorta e non molto discriminante [1]. Si consideri il seguente esempio.

Esempio 2.4. Se si vuole sapere quali tipi di contesti sono condivisi da ciliegia e fragola, ma non da digitale e informazione, non si ottiene una buona discriminazione da parole come il, esso o essi, che ricorrono frequentemente con tutti i tipi di parole e non sono informativi su nessuna parola in particolare.

Non si intende analizzare nel dettaglio le soluzioni a questo problema, ma è opportuno menzionare che si utilizzano tecniche di ponderazione, come l'algoritmo PPMI (Positive Pointwise Mutual Information) [1], comunemente usato quando le dimensioni del vettore sono altre parole e la tecnica TF-IDF discussa in 2.3.2, tipicamente impiegata quando le dimensioni del vettore di rappresentazione delle parole sono documenti.

Embeddings

Nella sezione precedente le parole sono state rappresentate attraverso i vettori di co-occorenza, vettori sparsi e di grandi dimensioni, con un numero di dimensioni corrispondenti alla cardinalità del vocabolario V o al numero di documenti D di una raccolta. Ora si vuole introdurre una rappresentazione più avanzata delle parole: gli embeddings ovvero vettori numerici densi a bassa dimensionalità. Gli embeddings sono corti, con un numero di dimensioni d che va da 50 a 1000, in contrasto con le dimensioni elevate del vocabolario e del corpus di documenti. A differenza dei vettori sparsi, i vettori di embedding non presentano valori nulli predominanti, ma contengono numeri reali, inclusi valori negativi. Inoltre, le dimensioni d degli embedding non hanno un'interpretazione semantica immediata, ma derivano dall'addestramento del modello.

Si distinguono in embedding statici e dinamici:

- **Embedding statici**: In questo caso, per ogni parola del vocabolario viene appreso un singolo embedding, indipendentemente dal contesto in cui la parola appare. Di conseguenza, ogni parola è rappresentata sempre dallo stesso vettore, anche se può assumere significati diversi in contesti differenti. Gli embedding statici sono generalmente pre-addestrati tramite algoritmi non supervisionati, come i modelli di Continuous Bag of Words (CBOW) e Continuous Skip-Gram [3]. Tra i modelli più comuni per la generazione di embedding statici troviamo Google Word2Vec, Stanford GloVe, Facebook fastText e SENNA.

- **Embedding dinamici o contestuali**: Gli embedding dinamici, o contestuali, generano vettori diversi per una stessa parola quando essa appare in contesti differenti. Questo approccio utilizza reti neurali avanzate, come i modelli basati sui transformers, che catturano le variazioni di significato delle parole in base al contesto. Un esempio di modello a embedding contestuali è BERT (Bidirectional Encoder Representations from Transformers).

2.3 Rappresentazioni di frasi e documenti

In alcuni compiti di NLP, come il clustering di documenti, è fondamentale ottenere una rappresentazione a livello di documento piuttosto che limitarsi a una rappresentazione a livello di parola. Procediamo ora con la descrizione di alcuni metodi per rappresentare un corpus di documenti, denotato come $D = \{d_1, ..., d_r\}$. Le tecniche che verranno presentate si basano sull'approccio di trattare documenti e frasi come insiemi di parole, combinando quindi le rappresentazioni sviluppate per singole parole per ottenere una rappresentazione estesa al livello di frase o di documento. Verranno descritte le seguenti tecniche:

- Bag of Words;
- TF-IDF;
- embeddings.

2.3.1 Bag of Words

Bag of Words o BoW è una forma di codifica utilizzata per rappresentare documenti. In questo caso la rappresentazione riflette la frequenza con cui una parola appare nel documento. Di seguito una spiegazione formale.

Definizione 2.8. Sia $V = \{w_1, ..., w_n\}$ un vocabolario di parole con $n = |V|$. La rappresentazione Bag of Words di un documento d è un vettore

$$v_d = (f(w_1), ..., f(w_n))$$

dove $f(w_i)$ indica la frequenza assoluta della parola w_i nel documento d.

Osservazione 2.8. La rappresentazione "Bag of Words" può essere vista in termini di rappresentazione One-Hot-encoding 2.4. Infatti si osservi che $v_d = \sum_{i \in I} v_{w_i}$ dove $I \subseteq V$ è l'insieme delle parole che compongono il documento e v_{w_i} è la rappresentazione one-hot-encoding della parola w_i descritta precedentemente nella sezione 2.2.1.

Osservazione 2.9. Si noti che questo tipo di rappresentazione equivale a interpretare la matrice term-document descritta in 2.7 leggendo le sue colonne. Di seguito è riportato un esempio di matrice term-document, in cui ciascuna colonna rappresenta un documento corrispondente alla rappresentazione Bag of words.

	Documento 1	*Documento 2*	*Documento 3*	*Documento 4*
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Tabella 2.2: Term-Document Matrix

Nella rappresentazione Bag of Words si utilizza una frequenza assoluta. Una versione "normalizzata" di questa rappresentazione è descritta di seguito.

2.3.2 TF-IDF encoding

TF-IDF è l'acronimo di Term Frequency-Inverse Document Frequency ed è una statistica numerica che riflette l'importanza di una parola in un documento di una raccolta o di un corpus e si ottiene come prodotto di due termini:

- Frequenza dei termini (TF): Misura la frequenza con cui un termine (parola) compare in un documento. Più frequente è il termine, più alto è il TF.
- Frequenza inversa del documento (IDF): Misura l'importanza di un termine nell'intero corpus. Se un termine appare frequentemente in molti documenti, ha un IDF più basso.

Definizione 2.9. Sia $V = \{w_1, ..., w_n\}$ un vocabolario e $D = \{d_1, ..., d_r\}$ un corpus di r documenti. Si consideri una matrice $F \in \mathbb{R}^{|V|\times|D|}$ con $F_{i,j} = TF(i,j) \times IDF(i)$ dove:

- $TF(i,j) = \frac{n_{i,j}}{|d_j|}$ dove $n_{i,j}$ è la frequenza della parola w_i nel documento d_j mentre $|d_j|$ è il numero di parole di cui è composto il documento d_j.
- $IDF(i) = \log_{10}(\frac{r}{df(i)})$ con r il numero totale di documenti nel corpus D e $df(i) = |\{d|w_i \in d\}|$

La rappresentazione TF-IDF del j-esimo documento d_j è la seguente:

$$v_{d_j} = (F_{1j}, F_{2j}, ..., F_{nj})$$

Osservazione 2.10. Le righe della matrice F rappresentano le parole del vocabolario, mentre le colonne corrispondono ai documenti, analogamente alla term-document matrix descritta in 2.7. Tuttavia, a differenza di quest'ultima, la matrice F non registra le frequenze assolute delle parole, bensì i valori TF-IDF per ciascun termine.

Nel contesto dell'elaborazione del linguaggio naturale (NLP), TF-IDF è spesso utilizzato per l'estrazione di testi, il recupero di informazioni e la categorizzazione di testi. Aiuta a identificare le parole importanti in un documento rispetto a un corpus più ampio, consentendo di cogliere l'unicità e il significato dei termini in un determinato contesto [4].

Osservazione 2.11. I metodi Bag of Words e TF-IDF appena visti però non tengono conto della semantica. Quindi anche nel caso della rappresentazione di documenti si vogliono introdurre gli embedding.

2.3.3 Embedding

I metodi precedenti per rappresentare documenti o frasi non catturano la semantica del testo. Per ottenere una rappresentazione semantica, possiamo utilizzare gli embedding delle parole, che preservano il loro significato, e combinarli per formare embedding a livello di frase o documento. Mentre gli embedding a livello di parola forniscono una rappresentazione per ogni token, gli embedding di frasi e documenti offrono una sintesi in un'unica rappresentazione di lunghezza fissa, riflettendo il significato complessivo della sequenza. Questo processo di aggregazione è noto come pooling [5].

In termini più formali si ha:

Definizione 2.10. Sia $f =< t_1 >, ..., < t_n >$ una frase composta da n token. Siano $e_1, ..., e_n$ dove $e_i \in \mathbb{R}^k$ rappresenta l'embedding di dimensione k dell'i-esimo token della sequenza di input. Allora l'embedding rappresentativo della frase f è dato da:

$$\mathbf{e}_f = h(e_1, ..., e_n)$$

dove h è una funzione di pooling. Le funzioni di pooling possono essere:

- Mean pooling:
$$h(e_1, ..., e_n) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{e}_i$$
- Weighted-mean pooling:
$$h(e_1, ..., e_n) = \sum_{i=1}^{n} \alpha_i \mathbf{e}_i$$
- Max-pooling:
$$(\mathbf{e}_f)_j = \max_{i=1}^{n} (\mathbf{e}_i)_j \quad \text{con } j = 1, ..., k$$
- mean sqrt len pooling:
$$(\mathbf{e}_f)_j = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{w}_i)_j$$
- Il "CLS pooling" consiste nell'inserire un token speciale, < CLS >, all'inizio di ogni sequenza. Questo token raccoglie informazioni da tutti i token della frase, e il suo embedding è progettato per condensare il contenuto globale della sequenza.

Osservazione 2.12. Le strategie *max pooling* e *mean sqrt len pooling* sono quelle utilizzate meno frequentemente. In generale non c'è un consenso definitivo riguardo al metodo di pooling da utilizzare; la scelta dipende dal contesto specifico e dall'applicazione in esame.

Si può notare che la rappresentazione di un documento d può essere ottenuta in modo analogo a quanto descritto, semplicemente sostituendo gli embedding a livello di parola con quelli a livello di frase nella definizione 2.10.

2.3.4 Similarità Coseno

Per misurare la similarità tra due parole oppure tra due documenti rappresentati tramite un embedding, solitamente la misura di similarità più comunemente usata è la similarità coseno che misura l'angolo tra i due vettori.

Definizione 2.11. Dati $v_1, v_2 \in \mathbb{R}^n$ si definisce la similarità coseno come:

$$sim(v_1, v_2) = \frac{< v_1, v_2 >}{||v_1|| \cdot ||v_2||}$$

Il valore della similarità coseno varia da 1 per i vettori con stessa direzione, a 0 per i vettori ortogonali, a -1 per i vettori con direzioni opposte.

Si osservi che se i vettori sono normalizzati, la distanza euclidea fornisce lo stesso ordinamento della misura di similarità coseno.

Dimostrazione. Siano $v_1, v_2 \in \mathbb{R}^n$. Si ha che:

$$\begin{aligned}
d\left(\frac{\mathbf{v}_1}{\|\mathbf{v}_1\|}, \frac{\mathbf{v}_2}{\|\mathbf{v}_2\|}\right) &= \sqrt{\sum_{i=1}^{n} \left(\frac{(\mathbf{v}_1)_i}{\|\mathbf{v}_1\|} - \frac{(\mathbf{v}_2)_i}{\|\mathbf{v}_2\|}\right)^2} \\
&= \sqrt{\frac{1}{\|\mathbf{v}_1\|^2} \sum_{i=1}^{n} (\mathbf{v}_1)_i^2 - \frac{2}{\|\mathbf{v}_1\| \cdot \|\mathbf{v}_2\|} \sum_{i=1}^{n} (\mathbf{v}_1)_i (\mathbf{v}_2)_i + \frac{1}{\|\mathbf{v}_2\|^2} \sum_{i=1}^{n} (\mathbf{v}_2)_i^2} \\
&= \sqrt{\frac{1}{\|\mathbf{v}_1\|^2} \cdot \|\mathbf{v}_1\|^2 - \frac{2}{\|\mathbf{v}_1\| \cdot \|\mathbf{v}_2\|} \cdot \langle \mathbf{v}_1, \mathbf{v}_2 \rangle + \frac{1}{\|\mathbf{v}_2\|^2} \cdot \|\mathbf{v}_2\|^2} \\
&= \sqrt{2(1 - \mathrm{sim}(\mathbf{v}_1, \mathbf{v}_2))}.
\end{aligned}$$

Questo dimostra che, se i vettori sono normalizzati, una maggiore similarità rispetto alla similarità coseno implica una minore distanza euclidea tra i vettori. □

Capitolo 3

Named Entity Recognition

Uno dei compiti del NLP consiste nell'estrazione di informazioni dai testi che permette di svolgere mansioni del NLP ancora più complessi come la traduzione automatica, la creazione di sistemi di domanda/risposta e molto altro.
Tra le varie informazioni che si possono estrarre da un testo si è interessati alle "named entities" o entità denominate. Un'entità denominata è qualsiasi cosa a cui si possa fare riferimento con un nome proprio come ad esempio una persona, un luogo, un'organizzazione. Tuttavia, il termine entità denominata è comunemente esteso per includere cose che non sono entità di per sé, tra cui espressioni temporali come date e orari e persino espressioni numeriche come i numeri di telefono, codici fiscali e molto altro. D'ora in avanti faremo riferimento alle "entità denominate" semplicemente come entità.

Il processo di riconoscere queste entità e classificarle in categorie predefinite, permettendo l'estrazione di informazioni rilevanti dai testi, viene detto "Named Entity Recognition", abbreviato NER. Ad esempio, un sistema di NER applicato alla frase "Mario Rossi ha lavorato per Mediaset a Milano" è in grado di classificare le seguenti entità:

- "Mario Rossi" - PERSONA
- "Mediaset"- ORGANIZZAZIONE
- "Milano"- LUOGO

In questo capitolo si intende formalizzare il concetto del NER e fornire una panoramica delle metodologie utilizzate, partendo dagli approcci tradizionali basati su regole rigide che non necessitano di dati annotati fino ad arrivare a quelli basati sul machine learning, in cui sarà invece necessario illustrare come rappresentare gli esempi (x_i, y_i) a partire dal dato grezzo, ovvero il testo.

3.1 Formalizzazione NER

L'obiettivo del NER è individuare ed etichettare specifiche entità all'interno di un testo. Questo compito rientra in una vasta categoria di problemi di NLP chiamata sequence tagging ovvero etichettatura di sequenze. Il Named Entity Recognition prende in input una frase o un testo T che viene trasformato in una sequenza di token:

$$(t_1, ..., t_n)$$

e restituisce in output una sequenza di etichette:

$$(y_1, ..., y_n)$$

dove:

- $t_i \in X = \{t |\ t \text{ token}\}$;
- $y_i \in Y = \{y |\ y \ \text{ entità}\}$

Si deve quindi definire una funzione f che leghi i token alle entità:

$$f :\ X \longrightarrow Y$$

$$t \longmapsto f(t) = y$$

Ogni token t_j della sequenza di input deve essere associato all'etichetta corrispondente y_j. Se un token non corrisponde a nessuna entità viene comunque classificato con il tag "no-entity". Le etichette utilizzate per i modelli NER vengono discusse nella sezione 3.3.

Osservazione 3.1. L'input di un modello di NER consiste in una sequenza di token, poiché gli approcci utilizzati per classificare le entità sfruttano le dipendenze dal contesto fornito dai token circostanti.

Inoltre, per ogni entità $y \in Y$, l'algoritmo restituisce sia i confini dell'entità nel testo, in termini di caratteri, sia il punteggio ad essa associato ovvero un valore di confidenza che riflette la probabilità stimata dal modello per quella particolare entità.

Con l'introduzione della funzione f, il compito di sequence tagging viene riformulato come un problema di classificazione multiclasse. Esistono diversi metodi per effettuare NER che verranno descritti nella sezione successiva e la funzione f dipende proprio dall'approccio utilizzato.

3.2 Metodologie NER

Le metodologie per eseguire il riconoscimento di entità nominate (NER) si distinguono sia in approcci tradizionali sia in approcci innovativi. Di seguito vengono elencate le principali categorie:

- strategie basate su regole tra cui:
  - dizionari di entità
  - espressioni regolari (dette anche RegEx)
  - regole basate su pattern
  - regole basate sul contesto
- tecniche di machine learning.

Nel caso degli approcci basati su regole, la funzione f è definita da regole predeterminate che non vengono apprese ma sono programmate e vengono applicate in modo fisso. Al contrario, nei metodi basati su tecniche di machine learning, f corrisponde a un modello che deve essere appreso dai dati attraverso un processo di addestramento.

Un'ulteriore importante differenza tra i due approcci riguarda l'input utilizzato: nei metodi basati su regole, il riconoscimento delle entità avviene direttamente sui token, ovvero sulla sequenza di parole grezze. Nei modelli di machine learning, invece, è necessario trasformare i token in rappresentazioni più avanzate che catturano informazioni semantiche e sintattiche più profonde, per permettere al modello di apprendere in modo più efficiente. Si vuole ora descrivere questi approcci più nello specifico.

3.2.1 Approcci basati su regole

Il NER basato su regole (Rule-Based NER) è uno dei metodi più tradizionali per il riconoscimento delle entità nominate all'interno di un testo. Questo approccio si basa sull'utilizzo di un insieme di regole predefinite molto accurate che sfruttano conoscenze linguistiche, lessicali e sintattiche per individuare pattern/schemi specifici che corrispondono a determinate categorie di entità.
Queste regole vengono necessariamente progettate manualmente in base alla tipologia (dominio) dei testi da analizzare e di conseguenza non sono adattabili in maniera efficiente a testi con domini differenti e non necessitano di dati annotati. Questi sistemi richiedono sia l'ingegno e la progettazione umana sia abilità di programmazione molto elevate. Inoltre, queste regole possono sfruttare alcuni risultati delle tecniche di NLP analizzate precedentemente come POS tagging e lemmatizzazione oltre alla tokenizzazione.

Tra questi metodi si trovano:

- dizionari di entità;
- espressioni regolari (dette anche RegEx);

- regole basate su pattern;
- regole basate sul contesto.

Dizionari di entità

L'approccio basato su dizionari è il metodo più semplice e intuitivo per il riconoscimento delle entità (NER). In questo sistema, si utilizzano dizionari ovvero elenchi di vocaboli che rappresentano esempi di una specifica entità. In questo approccio viene utilizzato un algoritmo di string matching per confrontare ogni token nel testo con i termini presenti nei dizionari. Se un token coincide con una parola nel dizionario, il sistema lo classifica con l'entità corrispondente.

Formalmente, sia Y l'insieme delle entità da riconoscere. Per ogni $entity \in Y$ si definisce un dizionario D_{entity} ovvero un insieme di vocaboli associati all'entità $entity$.

Sia T un testo tokenizzato a livello di parola $T =< t_1 > ... < t_n >$.
La regola basata su dizionari è quindi formalizzata come segue:

$$\mathrm{f}(t_j) = \begin{cases} entity, & \text{se } \exists\, entity \in Y \text{ tale che } t_j \in D_{entity} \\ O & \text{altrimenti} \end{cases}$$

dove O indica che il token non è stato assegnato a nessuna entità.

Esempio 3.1. Se lo scopo è identificare le professioni e i mestieri, si potrebbe definire un dizionario come segue:

$$D_{PROFESSIONE} = \{\text{“ingegnere”, “avvocato” ,“insegnante”, “cameriere”}\}$$

Data la frase: “Dopo una prima esperienza di cameriere, Mario ha deciso di diventare cuoco”, il testo viene suddiviso in token a livello di parola, e ogni token viene confrontato con i termini presenti nel dizionario. In questo esempio, l'algoritmo NER basato su dizionari riconosce correttamente solo la parola “cameriere” come appartenente alla categoria PROFESSIONE, ma non riesce a identificare la professione “cuoco” poiché non è inclusa nel dizionario.

Osservazione 3.2. Dall'esempio precedente emergono chiaramente alcuni svantaggi legati a questo approccio. In primo luogo, sono necessari aggiornamenti costanti dei dizionari affinché il modello NER funzioni in modo efficace. Inoltre, i dizionari tendono a essere poco generalizzabili a domini diversi. Ad esempio, se si desidera riconoscere nomi di medicinali, sarà necessario creare dizionari specifici per quel settore, i quali non saranno adeguati o utili per altri contesti.

Espressioni Regolari - (Regex)

L'espressione regolare (o RegEx) è un modello utilizzato per identificare entità costituite da sequenze di caratteri specifici. Formalmente, un'espressione regolare è una notazione algebrica per descrivere un insieme di stringhe [1].Queste espressioni permettono di identificare i token che soddisfano i criteri stabiliti

dallo schema definito dalla regex e di assegnare a ciascun token l'entità corrispondente. Si rivelano particolarmente utili per effettuare ricerche testuali di entità con una struttura ben definita, come indirizzi email, date o codici fiscali.

Regole basate su pattern - (Pattern matching)

I metodi basati su pattern in NLP si riferiscono a tecniche più sofisticate che non si limitano a una semplice corrispondenza tra stringhe e pattern predefinito come avviene nelle espressioni regolari, ma sfruttano anche informazioni lessicali, sintattiche o grammaticali per riconoscere schemi più complessi. Queste informazioni vengono estratte tramite alcune tecniche di NLP come POS tagging, lemmatizer, e Dependency Parsing. Un esempio di algoritmo di riconoscimento di entità (NER) basato su pattern grammaticale, che sfrutta i risultati del POS tagging per identificare nomi di persona è il seguente.

Esempio 3.2. Dato un testo tokenizzato a livello di parola:

$$T =< t_1 >, \ldots, < t_n >$$

si applica un algoritmo di POS tagging che genera una sequenza di tag $< p_1 > ... < p_n >$ dove p_j rappresenta il tag grammaticale associato al token t_j. La regola per l'assegnazione della categoria NOME è la seguente: se $p_j = p_{j+1} = PROPN$ (proper name) per qualche $j \in \{1, ..., n-1\}$, allora si assegna:

$$f(t_j) = f(t_{j+1}) = NOME$$

Regole basate sul contesto

Le regole basate sul contesto si fondano sul significato e sulla posizione della parola nel documento. Come suggerisce il nome, queste regole sono progettate per identificare entità in base al contesto circostante. Per esempio, si può definire un contesto di parole chiave per la determinazione di alcune entità come mostrato di seguito.

Esempio 3.3. Se si desidera identificare un numero di telefono, è possibile definire un contesto di parole chiave, come "cellulare", "numero", "telefono", "contatti", che aumentano il punteggio dell'entità quando si trovano nelle vicinanze della possibile entità.

Osservazione 3.3. Si osserva facilmente che questi metodi, se utilizzati singolarmente, non si rivelano particolarmente efficienti. Nei modelli di NER, infatti, è consueto combinare diversi approcci. Come già evidenziato, l'output di un algoritmo NER include anche un punteggio che indica la probabilità che un determinato token rappresenti un'entità. I punteggi ottenuti da ciascun metodo coinvolto vengono quindi combinati per determinare il risultato finale associato alle entità.

Tuttavia, anche la combinazione di questi metodi presenta delle notevoli limitazioni che verranno descritte nella seguente sezione.

Problematiche degli approcci basati su regole

I metodi basati su regole presentano significative limitazioni. Questi approcci si concentrano principalmente sul riconoscimento di pattern predefiniti o sulla ricerca di stringhe esatte, senza considerare il contesto in cui le parole appaiono. Di conseguenza, hanno difficoltà a gestire situazioni in cui una stessa parola può corrispondere a entità diverse a seconda del contesto in cui si trova, come si verifica nel seguente esempio.

Esempio 3.4. (Ambiguità di tipologia)
Si considerino le seguenti frasi:

1. "Washington è stato un grande leader nella storia americana"

2. "Sara ha visitato Washington"

Nella prima frase, la parola "Washington" si riferisce a un nome proprio di persona, mentre nella seconda indica un luogo.

I metodi basati su regole, come quelli precedentemente descritti, non sono in grado di differenziare tra i due significati della parola "Washington" senza un'analisi del contesto circostante. Questa mancanza di sensibilità semantica limita la loro capacità di riconoscere correttamente le entità in testi complessi o ambigui. In tali situazioni, è fondamentale comprendere il significato semantico dell'intera frase per poter identificare e assegnare correttamente le etichette alle entità.

Oltre al problema della **semantica**, questi approcci si basano su regole molto specifiche e rigide. Questa rigidità rende difficile l'adattamento a nuovi domini, poiché, come già osservato in 3.2, le regole devono essere costantemente aggiornate per garantire l'efficienza dell'algoritmo. Oltre all'ambiguità della tipologia come già descritto nell'esempio 3.4, esiste anche un'ambiguità dei confini: questi approcci infatti non performano bene nella gestione delle entità complesse, per esempio quelle costituite da più parole. Queste limitazioni mettono in evidenza la necessità di approcci più dinamici e adattabili, come quelli basati sul machine learning, che utilizzano dati annotati per addestrare modelli in grado di apprendere automaticamente. Tali approcci consentono una gestione più efficace di testi complessi e ambigui, superando le restrizioni imposte dalle regole fisse.

3.2.2 Approcci basati su Machine Learning

Nel caso degli approcci basati sul machine learning, il NER si configura come un problema di **supervised learning**. Si consideri la funzione f descritta in precedenza:

$$f : X \longrightarrow Y$$

dove:

- $X = \{t | t \text{ token}\}$, rappresenta l'insieme dei token di input;
- $Y = \{y \mid y \text{ etichetta}\}$, rappresenta l'insieme delle etichette.

A differenza degli approcci basati su regole predefinite, in questo contesto f è un modello parametrico di classificazione-multiclasse i cui parametri vengono appresi durante l'addestramento utilizzando dati annotati.

Inoltre, è importante osservare che, nei modelli di machine learning, i token di input t_i non vengono trattati come semplici sequenze di caratteri, ma devono essere trasformati in rappresentazioni più efficienti, che costituiscono l'input per il modello di classificazione. Questo processo di trasformazione è essenziale per consentire al modello di apprendere e generalizzare dai dati di addestramento.

Un modello di NER basato su ML può essere idealmente suddiviso in tre passaggi principali:

- **Rappresentazione dell'input**: i token vengono trasformati in vettori che possono essere *feature vectors* 2.2.2, *embedding statici* o *embedding dinamici* 2.2.3.
- **Codifica del contesto**: Questo secondo step è cruciale per ottenere rappresentazioni degli input che includano il contesto, consentendo di etichettare ogni parola tenendo conto dell'intera frase.

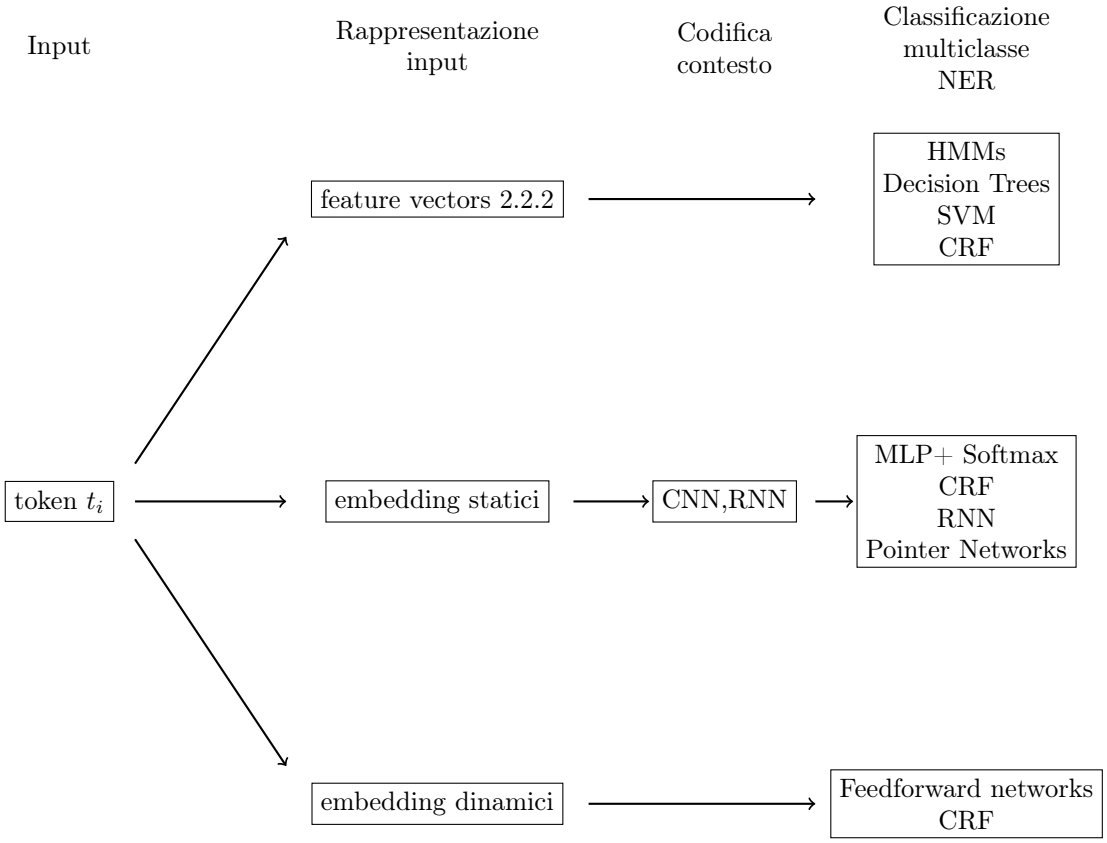
  Quando si utilizzano *embedding statici*, privi di informazioni contestuali (come descritto nella sezione 2.2.3), la codifica del contesto diventa indispensabile. In questi casi, si arricchisce la rappresentazione dell'input incorporando informazioni contestuali attraverso l'uso di reti come le **CNN** (Convolutional Neural Networks) o le **RNN** (Recurrent Neural Networks).

  Al contrario, se si utilizzano *embedding dinamici* come quelli generati da modelli come BERT, il contesto è già integrato nella rappresentazione dei token che può essere direttamente utilizzata come input nel modello di classificazione dello step successivo.
- **Classificazione multi-classe**: In questa parte si effettuano le previsioni delle entità per i token della sequenza di input a partire dalle rappresentazioni dipendenti dal contesto prodotte nello step precedente. In questa parte si utilizza il modello parametrico di classificazione f che viene addestrato a partire dai dati annotati minimizzando il rischio empirico, come descritto nella sezione 1.2:

  $$\min_{f \in \mathcal{H}} \hat{L}(f),$$

dove $\mathcal{H}$ rappresenta la famiglia di modelli disponibili, che può includere **CRFs** (Conditional Random Fields), **HMMs** (Hidden Markov Models), **RNNs** (Recurrent Neural Networks) o reti **feedforward**. MultiLayer Perceptron (MLP) e softmax, e Pointer Networks [3]. La scelta del modello f dipende dalla rappresentazione utilizzata per l'input e dalle caratteristiche del task.

Di seguito, uno schema riassuntivo delle tecniche di Machine Learning.



Nei primi approcci basati sul Machine Learning, i dati di input x_i erano rappresentati da *feature vectors*, come descritto nella sezione 2.2.2. La selezione delle feature che compongono questi vettori sfruttava frequentemente caratteristiche morfologiche, come il *POS tagging*, e semantiche, come le occorrenze multiple o la presenza in dizionari specifici.

Un vantaggio di questo approccio è che le feature estratte risultano facilmente interpretabili, offrendo una maggiore comprensione del modello. Tuttavia, questo metodo presenta anche delle limitazioni: richiede una progettazione accurata delle feature, un processo spesso complesso, lungo e che richiede competenze specifiche nel dominio di applicazione.

Per superare questi limiti, i modelli stato dell'arte per il Named Entity Recognition (NER) si basano su embedding dinamici appresi automaticamente attraverso algoritmi di *deep learning*, eliminando la necessità di progettare manualmente le features.

Osservazione 3.4. In generale, la scelta delle metodologie per il Named Entity Recognition dipende strettamente dal tipo di entità che si desidera individuare. Se le entità da rilevare hanno una struttura definita, come numeri di telefono o codici fiscali, approcci basati su regex, pattern matching e contesto risultano adeguati e spesso sufficienti. Tuttavia, quando è necessaria un'analisi semantica più approfondita, ad esempio per riconoscere entità che dipendono dal significato della frase o dal contesto, è preferibile ricorrere a tecniche di deep learning. In questa tesi viene adottata la piattaforma Presidio di Microsoft, che combina entrambe le metodologie: utilizza infatti regex e pattern matching per il riconoscimento di entità strutturate e approcci di deep learning per rilevare entità più complesse. In particolare, Presidio è stato integrato con un modello BERT.

3.3 Etichette

Come menzionato in precedenza, anche l'identificazione dei confini può dare origine a ambiguità, come illustrato nel seguente esempio.

Esempio 3.5. (Ambiguità dei confini)

Nella frase "Francesca lavora a New York", l'entità "New York" rappresenta un LUOGO, nonostante sia composta da due token. Questo genera un conflitto con la definizione di NER fornita in precedenza, secondo cui ogni singolo token viene assegnato a un sottoinsieme di etichette, escludendo quindi le entità multi-token.

Per gestire i casi di entità multi-token (come nell'esempio 3.5) rispettando la definizione di NER come compito di etichettatura token per token, si definisce l'insieme E dei tag in modo tale da catturare sia i confini delle entità sia il loro tipo, consentendo di identificare con precisione le entità denominate nel testo. [1]

Esistono diversi formati per rappresentare le entità nel NER. Uno dei più comuni è il formato BILUO. Esso tiene conto delle entità composte da più token catturando sia i confini delle entità sia il loro tipo.

In questo schema, a ogni entità viene assegnato un prefisso che indica la posizione del token all'interno dell'entità:

- B (Begin): primo token di un'entità multi-token
- I (In): token interno di un'entità multi-token
- L (Last): ultimo token di un'entità multi-token
- U (Unit): entità composta da un solo token
- O (Out): token che non fa parte di alcuna entità

Le entità vengono quindi etichettate con una categoria predefinita preceduta dal prefisso appropriato specificando quindi se si tratta di entità multi-token o no, e la posizione dei diversi token che costituiscono l'entità. In questo caso $|E| = 4l + 1$ dove l rappresenta il numero delle tipologie di entità ricercate e l'aggiunta di una unità si riferisce al prefisso O che indica l'assenza di entità.

Un secondo formato è lo schema IOB, una versione semplificata del BILUO. In questo caso, i prefissi indicano solo se un token si trova all'inizio o all'interno di un'entità multi-token:

- I : token all'interno di un'entità
- O: token che non rappresenta un'entità
- B: token all'inizio di un'entità

Una terza rappresentazione è il formato **jsonl**. In questo formato per ogni entità rilevata si indica la posizione nel testo (ovvero indice di inizio e fine dell'entità) e la categoria associata.

3.3.1 Strategia di Aggregazione per il Named Entity Recognition

Gli schemi BILUO e IOB descritti precedentemente, producono etichette che evidenziano la posizione del token rispetto all'entità rilevata. Risulta quindi necessario l'introduzione di una **strategia di aggregazione** per ricostruire le entità a partire dalle etichette predette.

Alcune strategie di aggregazione sono le seguenti:

- `''none''`: non viene effettuata alcuna aggregazione, restituendo semplicemente i risultati grezzi prodotti dal modello.
- `''simple''`: le entità vengono raggruppate seguendo uno schema predefinito. Per esempio:

  (A, B-TAG), (B, I-TAG), (C, I-TAG), (D, B-TAG2), (E, B-TAG2)

risulterà:

['word': 'ABC', 'entity': 'TAG', 'word': 'D', 'entity': 'TAG2', 'word': 'E', 'entity': 'TAG2'].

Si possono verificare situazioni di ambiguità, in cui una parola è suddivisa in più token e ciascun token riceve etichette diverse. L'ambiguità, quindi, si riferisce alla discrepanza nelle etichette assegnate ai token che compongono la stessa parola. In questi casi, è possibile applicare le strategie first, average e max per risolvere il problema.

- ''first'': utilizza la strategia `simple`, ma in caso di ambiguità, la parola assume semplicemente il tag del primo token.
- ''average'': utilizza la strategia `simple`, ma in caso di ambiguità, i punteggi vengono prima mediati tra i token, e successivamente la parola assume il tag con il punteggio medio massimo.
- ''max'': utilizza la strategia `simple`, ma in caso di ambiguità la parola assume semplicemente il tag del token con il punteggio massimo.

3.4 Valutazione del modello

La valutazione di un modello di riconoscimento di entità nominate (NER) si basa sul confronto tra l'output generato dall'algoritmo e i risultati prodotti da esperti linguistici. Nella verifica del funzionamento di un sistema NER entrano in gioco due aspetti fondamentali: l'identificazione dei confini dell'entità e l'assegnazione della tipologia.
Gli errori che possono verificarsi sono i seguenti:

- il sistema identifica un'entità dove non è presente;
- l'algoritmo non riconosce un'entità esistente;
- un'entità viene identificata con i confini corretti, ma viene assegnata un'etichetta sbagliata;
- un'entità viene riconosciuta correttamente, ma con confini errati;
- sia i confini sia l'etichetta assegnati sono errati.

Il NER consiste in un problema di classificazione multi-classe dove le classi corrispondono alle varie entità predefinite che si vogliono identificare.
Per la valutazione di tali modelli si introduce la seguente:

Definizione 3.1 (Confusion Matrix)**.** La Confusion Matrix corrisponde ad una tabella incrociata che registra il numero di occorrenze tra la classificazione vera/reale e la classificazione predetta. Le righe e le colonne di tale matrice rappresentano rispettivamente la classificazione reale e la previsione del modello.

	Valori Predetti			
	PERSONA	**LUOGO**	**ORGANIZZAZIONE**	**NON-ENTITÀ**
PERSONA - reale	120	5	3	12
LUOGO - reale	4	110	6	10
ORGANIZZAZIONE - reale	3	7	90	15
NON-ENTITÀ - reale	8	10	13	300

Tabella 3.1: Confusion matrix per la classificazione delle entità denominate

Si riporta di seguito un'esempio di tale matrice.

Osservazione 3.5. Le classi sono elencate nello stesso ordine nelle righe e nelle colonne; di conseguenza gli elementi classificati correttamente si trovano sulla diagonale principale.

Osservazione 3.6. L'ultima riga della Confusion Matrix indica se è stata assegnata un'entità dove non era prevista. L'ultima colonna, invece, tiene conto di quelle entità che il sistema avrebbe dovuto identificare ma per le quali non è stata predetta alcuna entità.

La Confusion Matrix permette di ricavare facilmente i seguenti valori necessari per la definizione delle metriche di valutazione:

- True Positive (TP): entità riconosciute correttamente dal modello NER;
- True Negative (TN): entità che il modello ha correttamente ignorato;
- False Positive (FP): entità erroneamente riconosciute dal modello;
- False Negative (FN): entità che il modello avrebbe dovuto riconoscere, ma non ha identificato.

Per ogni specifica entità si verifica che i TP corrispondono agli elementi sulla diagonale principale; i TN corrispondono alla somma delle entrate della matrice rimanenti escludendo la riga e la colonna dell'entità considerata; i FP corrispondono alla somma delle entrate della colonna dell'entità considerata escludendo la riga corrispondente dell'entità in questione ed infine, i FN corrispondono alla somma delle entrate della riga dell'entità escludendo la colonna dell'entità stessa.

Si vogliono ora introdurre alcune metriche classiche per la valutazione di un sistema di classificazione valide per ciascuna entità.

Definizione 3.2. La proporzione di entità predette correttamente rispetto al numero totale di entità riconosciute dal sistema viene chiamata **Precision**. Essa è la misura di quanto le entità predette siano corrette tra tutte quelle identificate dal modello. Si calcola attraverso la seguente formula:

$$Precision = \frac{\#TP}{\#TP + \#FP}$$

Definizione 3.3. La proporzione di entità correttamente riconosciute dal modello rispetto al numero totale di entità esistenti viene chiamata **Recall**. Questa misura indica quanto il modello sia riuscito a individuare tutte le entità presenti. Si calcola tramite la seguente formula:

$$Recall = \frac{\#TP}{\#TP + \#FN}$$

Definizione 3.4. L'**F-score** consiste in una metrica che combina Precision e Recall, calcolata come la loro media armonica [1]. Più precisamente:

$$\text{F-score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Definizione 3.5. L'accuratezza è la probabilità che la previsione del modello sia corretta. Essa restituisce una misura complessiva di quanto il modello predice correttamente sull'intero set di dati. Si calcola con la seguente formula:

$$Accuratezza = \frac{\#TP + \#TN}{\#TP + \#TN + \#FP + \#FN}$$

Inoltre, per avere una valutazione complessiva e riassumere la performance di tutte le entità predette dal modello, queste metriche possono anche essere calcolate globalmente, utilizzando due approcci:

- macro-media: si calcolano le metriche per ciascuna entità e si fa una media aritmetica di tutti i valori;
- micro-media: si calcolano i TP,TN,FP,FN come somma totale di quelli di ogni entità e si calcolano le metriche tramite le definizioni con questi nuovi valori totali.

Osservazione 3.7. La micro media si utilizza in caso di classi sbilanciate ovvero quando alcune classi sono molto più frequenti di altre.

Rimane da definire cosa si intenda per entità correttamente riconosciuta dal modello, ovvero come definire i True Positive (TP).
Due criteri giocano un ruolo cruciale nella valutazione di un modello NER:

- *tipologia*: la capacità del modello di assegnare l'etichetta corretta;
- *testo* : l'abilità di determinare correttamente i confini testuali a cui assegnare l'etichetta.

[1] La media armonica di n numeri reali positivi $x_1, ..., x_n$ si definisce come $\frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$

In generale, esistono due tipi di valutazione in base a come un'entità viene considerata correttamente riconosciuta, ovvero come vengono definiti i True Positive (TP): corrispondenza esatta o rilassata.

La "corrispondenza esatta" richiede che un'entità sia considerata correttamente riconosciuta solo se entrambi i criteri di tipologia e testo vengono soddisfatti in modo corretto.
Nella valutazione con corrispondenza rilassata, invece, tipologia e testo vengono considerati separatamente: la tipologia è considerata corretta se l'etichetta assegnata è giusta, indipendentemente dai confini, purché vi sia una sovrapposizione con i confini effettivi dell'entità; il testo è considerato corretto se i confini dell'entità sono individuati correttamente, indipendentemente dalla tipologia.

In generale, l'approccio utilizzato deve sempre far riferimento al contesto in cui si opera.

Capitolo 4

Applicazioni e Risultati ottenuti

L'obiettivo della tesi consiste nell'analisi automatica di documenti attraverso l'utilizzo di tecniche avanzate di Natural Language Processing (NLP). L'obiettivo principale è sviluppare un sistema in grado di elaborare e organizzare automaticamente documenti testuali mediante tecniche di rappresentazione e raggruppamento dei dati.

Tra le metodologie utilizzate sono presenti gli embedding 2.2.3 per la rappresentazione vettoriale del testo, il Named Entity Recognition (NER) 3 per l'identificazione di informazioni chiave, e il metodo k-means, basato sull'algoritmo di Lloyd 1.3.3 per effettuare il clustering.

L'applicazione specifica riguarda l'analisi e il clustering di curriculum vitae, con l'obiettivo di individuare automaticamente candidati simili. Il dataset utilizzato nel contesto di questa tesi è composto da circa 1700 curricula. Ogni riga rappresenta un curriculum individuale, mentre le colonne contengono informazioni relative a vari campi che possono essere a testo libero o no. In particolare, sono presenti quattro campi a testo libero:

- professional experience;
- professional skills;
- education;
- other information.

Per il clustering sarà considerato esclusivamente il campo "professional experience", mentre per l'anonimizzazione saranno presi in esame tutti e quattro i campi.

Nella sezione successiva verrà approfondita la parte dedicata al Named Entity Recognition (NER), seguita dall'analisi relativa al clustering, descritta dettagliatamente nella sezione 4.4.

4.1 NER

Come accennato in precedenza, il Named Entity Recognition (NER) consente di identificare diverse tipologie di entità all'interno di testi. Le entità più comunemente ricercate includono persone, luoghi e organizzazioni, ma, a seconda delle esigenze specifiche, il NER può essere adattato per rilevare entità più dettagliate e pertinenti al problema in esame.

In questo contesto, il NER è stato impiegato con due obiettivi principali:

1. **Anonimizzazione delle informazioni sensibili**: in una fase di preprocessing, il NER è stato utilizzato per identificare informazioni sensibili presenti nei curricula, come nomi, indirizzi, codici fiscali e indirizzi e-mail. Una volta rilevate, queste informazioni sono state anonimizzate. Gli scopi dell'anonimizzazione applicati ai curriculum sono vari:
   - riduzione dei pregiudizi nei processi di selezione, rimuovendo informazioni come nome, indirizzi...
   - focus sulle competenze e qualifiche per valutazione più equa
   - protezione della privacy, qualora i CV fossero elaborati da modelli LLM, come quelli forniti da OpenAI o altri provider esterni.
2. **Rilevamento di entità per il clustering**: successivamente, il NER è stato utilizzato per estrarre entità quali nomi di professioni e organizzazioni. Queste informazioni sono state impiegate in una fase successiva per eseguire il clustering dei curricula, con l'obiettivo di raggruppare i candidati in base a caratteristiche comuni. Il processo di clustering verrà approfondito nelle sezioni successive.

4.2 Presidio

Per il NER è stata utilizzata la libreria Microsoft Presidio [10], una libreria specializzata sia nel riconoscimento sia nell'anonimizzazione delle informazioni sensibili. Oltre ai riconoscitori di entità predefiniti, la libreria consente di configurare riconoscitori personalizzati e di integrare modelli esterni per estendere la gamma di entità rilevabili. Presidio è una libreria flessibile in quanto permette la personalizzazione sia della parte di identificazione sia della parte dell'anonimizzazione.
Come accennato esso è costituito da due parti:

- **Presidio Analyzer** per il riconoscimento;
- **Presidio Anonymizer** per l'anonimizzazione.

4.2.1 Presidio Analyzer

Questa parte si occupa dell'identificazione delle entità presenti nel testo. In generale Presidio utilizza diversi metodi per il riconoscimento tra cui:

- espressioni regolari (RegEx) 3.2.1;
- regole basate su contesto 3.2.1;
- regole basate su pattern 3.2.1;
- checksum;
- modelli NER basati su machine learning 3.2.2.

Presidio utilizza infatti una pipeline di NLP che svolge operazioni come la tokenizzazione, la lemmatizzazione, l'etichettatura delle parti del discorso (POS tagging) per processare il testo ed estrarre caratteristiche per gli approcci basati su regole. L'ultima componente della pipeline, invece, performa il NER attraverso metodi di machine learning.

La pipeline di default utilizzata da Presidio corrisponde ad un modello open source della libreria di NLP SpaCy (*en_core_web_lg*), ma è possibile inserire altri modelli che possono essere addestrati o scaricati da framework NLP esistenti come SpaCy, Stanza e Transformers.

Osservazione 4.1. Le espressioni regolari e gli approcci basati su regole vengono utilizzati per identificare entità che hanno una struttura specifica (come codice fiscale, numero di telefono) mentre i modelli NER basati su machine learning sono utili per ricercare le entità come nomi, luoghi che invece non hanno una struttura specifica e che possono essere quindi identificati in base al contesto della frase.

Presidio è in grado di riconoscere sia entità globali, ovvero valide per ciascuna lingua, sia entità riferite ad una specifica lingua. Si osservi che i curricula sono redatti in lingua italiana; di conseguenza, è stato necessario configurare Presidio per il riconoscimento in tale lingua. La lista delle entità riconosciute è riportata nelle tabelle 4.2 e 4.1.

Entità	**Descrizione**	**Metodo**
IT_FISCAL_CODE	Un codice fiscale italiano	Pattern match, contesto e checksum
IT_DRIVER_LICENSE	Un numero di patente italiana.	Pattern match e checksum
IT_VAT_CODE	Un numero di partita iva italiano.	Pattern match, contesto e contesto
IT_PASSPORT	Un numero di passaporto italiano.	Pattern match e contesto.
IT_IDENTITY_CARD	Un numero di carta d'identità italiano.	Pattern match e contesto.

Tabella 4.1: Entità predefinite di Presidio per italiano

Entità	Descrizione	Metodo
CREDIT_CARD	Il numero di una carta di credito è composto da 12 a 19 cifre.	Pattern match e checksum
CRYPTO	Un numero di portafoglio Crypto. Attualmente è supportato solo l'indirizzo Bitcoin.	Pattern match, contesto e checksum
DATE_TIME	Date assolute o relative, periodi o tempi inferiori al giorno.	Pattern match e contesto
EMAIL_ADDRESS	Un indirizzo e-mail identifica una casella di posta elettronica a cui vengono recapitati i messaggi e-mail.	Pattern match, contesto e validazione RCF-822
IBAN_CODE	L'International Bank Account Number (IBAN) è un sistema concordato a livello internazionale per identificare i conti bancari al di là dei confini nazionali.	Pattern match, contesto e checksum
IP_ADDRESS	Un indirizzo di protocollo internet (IPv4 o IPv6).	Pattern match, contesto e checksum
NRP	Nazionalità, religione o gruppo politico di una persona.	Logica e contesto personalizzati
LOCATION	Nome di una località politicamente o geograficamente definita (città, province, paesi, regioni internazionali, corpi idrici, montagne, ecc.)	Logica e contesto personalizzati
PERSON	Il nome completo della persona, che può includere nomi, secondi nomi o iniziali e cognomi.	Logica e contesto personalizzati
PHONE_NUMBER	Un numero di telefono.	Logica personalizzata, pattern match e contesto
MEDICAL_LICENSE	Numeri di licenza medica comuni.	Pattern match, contesto e checksum
URL	Un URL (Uniform Resource Locator), identificatore univoco utilizzato per localizzare una risorsa su Internet.	Pattern match e contesto

Tabella 4.2: Entità predefinite di Presidio globali

4.2.2 Presidio Anonymizer

Questa parte è dedicata all'anonimizzazione delle entità riconosciute da Presidio Analyzer. Presidio Anonymizer contiene diversi operatori ognuno dei quali può essere utilizzato per rendere anonime le entità in modo diverso.
Le opzioni di de-identificazione sono:

- **replace** sostituisce le informazioni sensibili con l'entità corrispondente;
- **redact** rimuove le informazioni sensibili rilevate dal testo;
- **highlight** mantiene le informazioni sensibili nel testo e inserisce la categoria/entità corrispondente;
- **mask** sostituisce informazioni sensibili con il carattere indicato;
- **hash** sostituisce informazioni con un codice;
- **encrypt** cripta le informazioni con una determinata chiave.

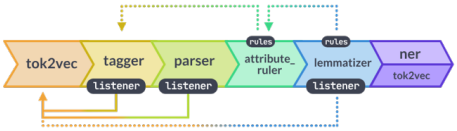
Osservazione 4.2. Nel contesto di questa tesi, l'algoritmo di selezione trasforma i curriculum in vettori di embedding, su cui viene poi eseguito il clustering. Pertanto, nel processo di anonimizzazione è necessario considerare questo aspetto. Le ultime tre opzioni non preservano il senso della frase, quindi l'opzione utilizzata è stata *replace*.

Come accennato precedentemente, Presidio è costituito da diversi riconoscitori tra cui quelli predefiniti, quelli personalizzati ed infine modelli esterni che si possono integrare. Grazie all'integrazione di modelli esterni all'interno di Presidio Analyzer, è possibile estendere il riconoscimento delle entità oltre quelle predefinite dalla piattaforma.
In questa tesi sono stati testati tre modelli distinti per identificare quello più efficiente. In particolare, sono state analizzate le seguenti opzioni:

1. Presidio con un modello di SpaCy $it\_core\_news\_lg$;
2. Presidio con un modello Transformer $DeepMount00/Italian\_NER\_XXL$;
3. utilizzo del modello $DeepMount00/Italian\_NER\_XXL$ senza Presidio.

Presidio con il modello di SpaCy it_core_news_lg

Inizialmente, Presidio è stato utilizzato non con la pipeline predefinita, addestrata in inglese, ma con un modello di Spacy specifico per la lingua italiana ovvero il modello $it\_core\_news\_lg$. In generale, la struttura delle pipeline di SpaCy è la seguente:



Questo modello è in grado di riconoscere tre tipologie di entità: nomi di persona (PER), di luoghi (LOC) e di organizzazioni (ORG).
Quando un modello esterno viene integrato, bisogna configurare Presidio ad utilizzarlo. La configurazione può avvenire sia tramite codice sia tramite un file di configurazione esterno. In particolare devono essere specificati alcuni parametri:

- **nlp_engine_name** : nome del sistema di NLP usato. Devono essere indicati anche:
  - **lang_code** : lingua utilizzata
  - **model_name**: nome del modello esterno scelto
- **ner_model_configuration**: in cui devono essere specificati:
  - **labels_to_ignore**: un elenco di etichette da ignorare. Ad esempio, 0 (nessuna entità) entità che non si è interessati a restituire.
  - **model_to_presidio_entity_mapping**: una mappatura tra le etichette del modello e i tipi di entità Presidio.
  - **low_confidence_score_multiplier**: un moltiplicatore da applicare al punteggio delle entità con bassa confidenza.

– **low_score_entity_names**: elenco di tipi di entità a cui applicare il moltiplicatore del punteggio di bassa confidenza.

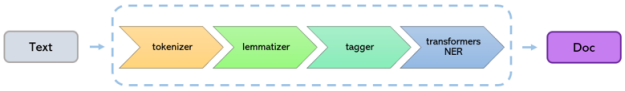
Nel caso in questione il file di configurazione utilizzato è rappresentato in figura 4.2.2.

```
nlp_engine_name: spacy
models:
  -
    lang_code: it
    model_name: it_core_news_lg

ner_model_configuration:
  model_to_presidio_entity_mapping:
    PER: PERSON
    LOC: LOCATION
    ORG: RAGIONE_SOCIALE
    MISC: MISC


  low_confidence_score_multiplier: 0.4
  low_score_entity_names:
  -
  labels_to_ignore:
  - MISC
```

Osservazione 4.3. Questo modello riconosce un numero limitato di entità. Di conseguenza, sono stati esaminati altri modelli capaci di riconoscere un numero maggiore di entità.

Presidio con il modello Transformer DeepMount00/Italian_NER_XXL

Successivamente, è stato scelto un modello di NER addestrato in italiano che fosse in grado di riconoscere un numero maggiore di entità. Il modello scelto è integrato nella libreria Transformers di Hugging Face [11] ed è basato sull'architettura dei transformer ovvero una tipologia di rete neurale. Il nome del modello è $DeepMount00/Italian\_NER\_XXL$.
Presidio offre la possibilità di integrare un modello della libreria di Hugging Face Transformers attraverso una pipeline SpaCy che incapsula tale modello al posto della componente NER di SpaCy, nel seguente modo:



Presidio utilizza anche altre informazioni fornite da SpaCy, come token, lemmi e parti del discorso. Di conseguenza, la pipeline restituisce sia i risultati del modello NER sia quelli delle altre componenti della pipeline.

Osservazione 4.4. Tuttavia, mentre la pipeline di SpaCy gestisce già la tokenizzazione, il modello NER basato su Transformer adotta una propria tokenizzazione, suddividendo le parole in sotto-token anziché considerarle come singoli token. Questo può generare dei warning relativi all'allineamento tra i token delle due differenti tokenizzazioni.

Come nel caso del modello italiano di SpaCy, bisogna configurare Presidio per l'integrazione con il modello transformer o tramite codice o tramite un file di configurazione.

In questo caso il modello transformer corrisponde alla componente NER di una pipeline Spacy; pertanto oltre al modello transformer di Hugging Face scelto, nella configurazione è necessario specificare la pipeline Spacy scelta e alcuni parametri aggiuntivi rispetto a quelli gia visti precedentemente.
Più precisamente:

- **aggregation_strategy**: La strategia da utilizzare quando si aggregano i risultati del modello dei transformer in quanto essi utilizzano lo schema IOB nell'assegnazione delle identità. Inoltre, come descritto in 3.3.1, la strategia di aggregazione permette di determinare i tag a livello di parola per i casi in cui le sotto-parole all'interno di una parola non ricevono lo stesso tag previsto.

- **stride**: rappresenta la lunghezza della sovrapposizione della finestra nei token del tokenizzatore del transformer. Per stride positivo o nullo, il testo viene elaborato in finestre sovrapposte dove il valore di stride specifica il numero di token sovrapposti tra le finestre. Se stride è None, il testo potrebbe essere troncato.

- **alignment_mode**: La strategia da utilizzare quando si allineano i risultati del modello dei trasformatori al testo originale. Essa determina come le previsioni del transformers vengono allineate ai limiti del token SpaCy ovvero come gli indici dei caratteri si agganciano ai confini dei token.
  Le possibili opzioni sono:
  - **strict**: nessun aggancio
  - **contract**: l'intervallo di tutti i token deve essere completamente all'interno dell'intervallo di caratteri
  - **expand** : l'intervallo di tutti i token deve essere almeno parzialmente coperto dall'intervallo di caratteri

Questo modello è in grado di riconoscere fino a 52 entità diverse. La lista completa delle entità riconosciute è riportata nella Tabella 4.3. Le entità più rilevanti selezionate per il caso di applicazione in questione sono le seguenti:

- NOME
- COGNOME
- NUMERO_TELEFONO
- EMAIL
- INDIRIZZO
- LUOGO
- RAGIONE_SOCIALE
- PROFESSIONE
- CODICE_FISCALE
- CODICE_POSTALE
- NUMERO_CARTA
- DATA
- P_IVA
- BANCA
- LEGGE

Tramite il file di configurazione 4.2.2, esse sono state mappate all'interno di Presidio. Dalla lista delle entità del modello DeepMount00/Italian_NER_XXL, riportata in 4.3, si può notare che molte di esse sono correlate all'ambito medico, riflettendo così il focus e il dataset utilizzato per l'addestramento del modello.

Nel file di configurazione del modello, mostrato in 4.2.2, si osserva che le entità NOME e COGNOME sono state mappate all'entità PERSON già predefinita in Presidio. Un ragionamento analogo è stato applicato alle entità DATA, DATA_NASCITA e DATA_MORTE, che sono state mappate all'entità DATE_TIME di Presidio per aumentare il numero di entità riconosciute.

Durante i test del codice, è emerso che le entità del modello esterno non vengono riconosciute se non sono specificate all'interno del parametro model_to_presidio_entity_mapping.

Entità	Descrizione
INDIRIZZO	Identifica un indirizzo fisico
VALUTA	Rappresenta una valuta
CVV	Codice di sicurezza della carta di credito
NUMERO CONTO	Numero di un conto bancario
BIC	Codice identificativo di una banca
IBAN	Numero di conto bancario internazionale
STATO	Identifica un paese o una nazione
NOME	Riferito al nome di una persona
COGNOME	Riferito al cognome di una persona
CODICE POSTALE	Codice postale di un'area geografica
IP	Indirizzo IP di un dispositivo in rete
ORARIO	Riferito a un orario specifico
URL	Indirizzo web
LUOGO	Identifica un luogo geografico
IMPORTO	Riferito a una somma di denaro
EMAIL	Indirizzo di posta elettronica
PASSWORD	Parola chiave per l'accesso a sistemi protetti
NUMERO CARTA	Numero di una carta di credito o debito
TARGA VEICOLO	Numero di targa di un veicolo
DATA NASCITA	Data di nascita di una persona
DATA MORTE	Data di decesso di una persona
RAGIONE SOCIALE	Nome legale di un'azienda o entità commerciale
ETA	Età di una persona
DATA	Riferita a una data generica
PROFESSIONE	Occupazione o lavoro di una persona
PIN	Numero di identificazione personale
NUMERO TELEFONO	Numero telefonico
FOGLIO	Riferito a un foglio di documentazione
PARTICELLA	Riferito a una particella catastale
CARTELLA CLINICA	Documentazione medica di un paziente
MALATTIA	Identifica una malattia o condizione medica
MEDICINA	Riferito a un farmaco o trattamento medico
CODICE FISCALE	Codice fiscale personale o aziendale
NUMERO DOCUMENTO	Numero di un documento ufficiale
STORIA CLINICA	Registro delle condizioni mediche di un paziente
AVV NOTAIO	Identifica un avvocato o notaio
P IVA	Partita IVA di un'azienda o professionista
LEGGE	Riferito a una legge specifica
TASSO MUTUO	Tasso di interesse di un mutuo
N SENTENZA	Numero di una sentenza legale
MAPPALE	Riferito a un mappale catastale
SUBALTERNO	Riferito a un subalterno catastale
REGIME PATRIMONIALE	Stato patrimoniale in ambito legale
STATO CIVILE	Stato civile di una persona
BANCA	Identifica una banca o istituto di credito
BRAND	Marchio o brand commerciale
NUM ASSEGNOBANCARIO	Numero di un assegno bancario
IMEI	Numero di identificazione internazionale di un dispositivo mobile
$N_LICENZA$	Numero di una licenza specifica
IPV6 1	Indirizzo IP versione 6
MAC	Indirizzo MAC di un dispositivo di rete
USER AGENT	Identifica il software usato per accedere a una rete
TRIBUNALE	Identifica un tribunale specifico
STRENGTH	Riferito alla forza o intensità di del medicinale
FREQUENZA	Riferito alla frequenza di un trattamento medico
DURATION	Durata di un evento o trattamento
DOSAGGIO	Quantità di un medicinale da assumere
FORM	Forma del medicinale, ad esempio compresse

Tabella 4.3: Entità predefinite di DeepMount00/Italian_NER_XXL

```yaml
nlp_engine_name: transformers
models:
  -
    lang_code: it
    model_name:
      spacy: it_core_news_lg
      transformers: DeepMount00/Italian_NER_XXL

ner_model_configuration:

  aggregation_strategy: max # "simple", "first", "average", "max"
  stride: 0
  alignment_mode: expand # "strict", "contract", "expand"
  model_to_presidio_entity_mapping:
    INDIRIZZO: INDIRIZZO
    CVV: CVV
    NUMERO_CONTO: NUMERO_CONTO
    BIC: BIC
    IBAN: IBAN_CODE
    STATO: LOCATION
    NOME: PERSON
    COGNOME: PERSON
    CODICE_POSTALE: CODICE_POSTALE
    URL: URL
    LUOGO: LOCATION
    IMPORTO: IMPORTO
    EMAIL: EMAIL_ADDRESS
    PASSWORD: PASSWORD
    RAGIONE_SOCIALE: RAGIONE_SOCIALE
    NUMERO_CARTA: CREDIT_CARD
    TARGA_VEICOLO: TARGA_VEICOLO
    DATA_NASCITA: DATE_TIME
    DATA_MORTE: DATE_TIME
    ETA: ETA
    DATA: DATE_TIME
    PROFESSIONE: PROFESSIONE
    NUMERO_TELEFONO: PHONE_NUMBER
    CODICE_FISCALE: IT_FISCAL_CODE
    NUMERO_DOCUMENTO: NUMERO_DOCUMENTO
    STORIA_CLINICA: STORIA_CLINICA
    AVV_NOTAIO: AVV_NOTAIO
    P_IVA: IT_VAT_CODE
    LEGGE: LEGGE
    TASSO_MUTUO: TASSO_MUTUO
    N_SENTENZA: N_SENTENZA
    MAPPALE: MAPPALE
    SUBALTERNO: SUBALTERNO
    REGIME_PATRIMONIALE: REGIME_PATRIMONIALE
    STATO_CIVILE: STATO_CIVILE
    BANCA: BANCA

  low_confidence_score_multiplier: 0.4

  low_score_entity_names:
  - ID
  labels_to_ignore :
  - USER_AGENT
```

Di seguito è riportato un'esempio dell'anonimizzazione con i due modelli precedentemente descritti:

Testo originale	**Testo anonimizzato**
Mi chiamo Sara Neri e sono laureata in ingegneria biomedica presso l'Università di Genova. Attualmente lavoro come ingegnere biomedico presso HealthTech a Genova, situata in via della Salute 10, dove sviluppo dispositivi medici innovativi e collaboro con un team di ricercatori per migliorare le tecnologie esistenti. Ho contribuito alla realizzazione di due brevetti nel campo delle protesi avanzate. Potete contattarmi via email all'indirizzo sara.neri@example.com o telefonicamente al numero +39 010 2345678. Risiedo a Genova in via delle Innovazioni 12 e sono in possesso di patente di guida categoria B. Il mio codice fiscale è NRISRA90L50H501X. Il mio IBAN è IT60X0542811101000000123456.	Mi chiamo *PERSON* e sono laureata in ingegneria biomedica presso l'*LOCATION*. Attualmente lavoro come ingegnere biomedico presso *LOCATION* a *LOCATION*, situata in via della *LOCATION*, dove sviluppo dispositivi medici innovativi e collaboro con un team di ricercatori per migliorare le tecnologie esistenti. Ho contribuito alla realizzazione di due brevetti nel campo delle protesi avanzate. Potete contattarmi via email all'indirizzo *EMAILADDRESS* o telefonicamente al numero *PHONENUMBER*. Risiedo a *LOCATION*in via delle Innovazioni 12 e sono in possesso di patente di guida categoria B. Il mio codice fiscale è *LOCATION IBANCODE*.

Tabella 4.4: Anonimizzazione con modello SpaCy

Testo originale	**Testo anonimizzato**
Mi chiamo Sara Neri e sono laureata in ingegneria biomedica presso l'Università di Genova. Attualmente lavoro come ingegnere biomedico presso HealthTech a Genova, situata in via della Salute 10, dove sviluppo dispositivi medici innovativi e collaboro con un team di ricercatori per migliorare le tecnologie esistenti. Ho contribuito alla realizzazione di due brevetti nel campo delle protesi avanzate. Potete contattarmi via email all'indirizzo sara.neri@example.com o telefonicamente al numero +39 010 2345678. Risiedo a Genova in via delle Innovazioni 12 e sono in possesso di patente di guida categoria B. Il mio codice fiscale è NRISRA90L50H501X. Il mio IBAN è IT60X0542811101000000123456.	Mi chiamo NOME COGNOME e sono laureata in ingegneria biomedica presso l'Università di LUOGO . Attualmente lavoro come PROFESSIONE presso RAGIONE_SOCIALE a LUOGO situata in INDIRIZZO dove sviluppo dispositivi medici innovativi e collaboro con un team di ricercatori per migliorare le tecnologie esistenti. Ho contribuito alla realizzazione di due brevetti nel campo delle protesi avanzate. Potete contattarmi via email all'indirizzo EMAIL_ADDRESS o telefonicamente al numero NUMERO_TELEFONO . Risiedo a LUOGO in INDIRIZZO e sono in possesso di patente di guida categoria B. Il mio codice fiscale è IT_FISCAL_CODE Il mio IBAN è IBAN .

Tabella 4.5: Anonimizzazione con Presidio integrato con il modello DeepMount00/Italian_NER_XXL

Osservazione 4.5. Si osserva che il modello SpaCy riconosce l'entità "HealthTech" come luogo e non come organizzazione, mentre il secondo modello la riconosce correttamente.

DeepMount00/Italian_NER_XXL senza Presidio

Per quanto detto nell'osservazione 4.4 , è stato testato il modello Transformer senza l'integrazione con la libreria Presidio. Questo perché, per utilizzare un modello Transformer all'interno di Presidio, esso deve essere configurato come componente NER in una pipeline di SpaCy, il che causa una discrepanza tra le diverse tokenizzazioni (quella di SpaCy e quella del modello esterno).
L'obiettivo del test era verificare se l'assenza di questa integrazione comportasse differenze significative nei risultati.
In questo caso, utilizzando esclusivamente il modello di riconoscimento, il risultato consiste unicamente in una lista delle entità identificate. Di conseguenza, è stata implementata una funzione di anonimizzazione che sostituisce le informazioni sensibili con la rispettiva tipologia di entità riconosciuta.

4.3 Valutazioni NER

Per la valutazione dei modelli, sono stati presi in considerazione solo gli ultimi due scenari: uno in cui il modello $DeepMount00/Italian\_NER\_XXL$ è stato utilizzato autonomamente con l'implementazione di una funzione di anonimizzazione, e l'altro in cui lo stesso modello è stato integrato all'interno di Presidio. Il modello SpaCy non è stato valutato poiché era in grado di riconoscere un numero limitato di entità.

Il dataset di curriculum a disposizione comprende circa 1700 documenti.
Prima di procedere con la valutazione del modello utilizzando le metriche descritte nella sezione 3.4 su un campione di dati, è stata effettuata un'analisi delle entità riconosciute dal modello per identificare quelle più frequenti.
Nella tabella 4.6 sono riportate, per ciascuna colonna, le percentuali di ogni entità rispetto al totale delle entità rilevate all'interno del campo corrispondente. L'ultima colonna, invece, presenta le percentuali calcolate rispetto al totale complessivo di tutte le entità individuate nei quattro campi considerati. Questi valori sono stati calcolati sull'intero dataset, composto da 1700 curricula. L'ultima colonna della tabella 4.6 indica la percentuale di ciascuna entità sul totale delle entità rilevate. Si può osservare come le entità che sono state riconosciute maggiormente sono i nomi di persona (30% del totale delle entità rilevate), luoghi (30%) , ragioni sociali (7%) ed infine le date (20%).
Inoltre leggendo la tabella per colonne, si hanno le percentuali delle entità rilevate all'interno di ciascun campo a testo libero analizzato.

	Professional experience	**Professional skills**	**Education**	**Other information**	**Totale riga**
PERSON	5.9866	2.8846	11.9382	47.9211	24.9766
PHONE_NUMBER	0.3352	0	0.1404	0.6429	0.4306
EMAIL_ADDRESS	0	0	0	0.1286	0.0562
INDIRIZZO	4.7893	0.9615	2.3876	0.6858	2.5276
LOCATION	39.9904	45.6731	47.4719	20.7458	32.8029
RAGIONE_SOCIALE	11.7835	17.7885	6.6011	5.8294	8.7250
PROFESSIONE	3.7835	7.2115	3.7921	0.9001	2.6587
IT_FISCAL_CODE	0.0958	0	0.1404	0	0.0562
CODICE_POSTALE	1.7241	0	0.4213	0	0.7302
CREDIT_CARD	0	0	0	0	0
DATE_TIME	25.7663	20.6731	22.1910	17.2310	21.3630
IT_VAT_CODE	0.0479	0	0	1.071582	0.4868
BANCA	2.3946	1.9231	1.5449	2.1860	2.1719
LEGGE	3.3046	2.8846	3.3708	2.6575	3.0144
Totale colonna	100	100	100	100	100

Tabella 4.6: Distribuzione percentuale delle entità rilevate nei diversi campi dei curricula

Il controllo per la valutazione dei modelli è stato effettuato su un insieme di campionamento casuale di 50 curricula.
Come visto nella sezione 3.4, la valutazione di modelli di riconoscimento di entità nominate (NER), essendo modelli di classificazione multi-classe, si basa sul completamento delle "Confusion Matrix" che permettono la facile deduzione dei quattro valori (TP,TF,FP,FN) su cui sono definite le metriche Precision, Recall, e F-score utili per valutare la performance del modello.

Per ciascun curriculum è stata compilata la corrispondente "Confusion Matrix" relativa alle entità selezionate descritte precedentemente. Successivamente, è stata creata una "Confusion Matrix" complessiva che tenesse conto di tutti i valori ottenuti nei 50 curriculum analizzati. Essa è stata creata sommando i risultati di ciascuna Confusion Matrix dei curricula.

Si riportano di seguito i risultati ottenuti utilizzando Presidio con l'integrazione del modello Transformer. La Confusion Matrix complessiva è la seguente:

	PERSON	PHONE_NUMBER	EMAIL_ADDRESS	INDIRIZZO	LOCATION	RAGIONE_SOCIALE	PROFESSIONE	IT_FISCAL_CODE	CODICE_POSTALE	CREDIT_CARD	DATE_TIME	IT_VAT_CODE	BANCA	LEGGE	NON-ENTITÀ
PERSON	1202	0	0	3	1	0	0	0	0	0	0	0	0	0	6
PHONE_NUMBER	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0
EMAIL_ADDRESS	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0
INDIRIZZO	0	0	0	94	1	1	0	0	0	0	0	0	0	0	1
LOCATION	0	0	0	0	1465	0	0	0	0	0	0	0	0	0	8
RAGIONE_SOCIALE	9	0	0	1	45	379	0	0	0	0	1	0	55	0	65
PROFESSIONE	0	0	0	0	0	3	130	0	0	0	0	0	0	0	379
IT_FISCAL_CODE	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
CODICE_POSTALE	0	0	0	0	0	0	0	0	38	0	0	0	0	0	0
CREDIT_CARD	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DATE_TIME	0	5	0	0	3	0	0	0	0	0	1040	0	0	9	571
IT_VAT_CODE	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0
BANCA	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0
LEGGE	0	0	0	0	0	0	0	0	0	0	0	0	0	123	0
NON-ENTITÀ	37	14	0	33	90	74	10	3	1	0	26	6	49	20	0

Tabella 4.7: Matrice di confusione complessiva delle entità riconosciute

Come osservato nella sezione precedente, nella tabella le righe rappresentano le entità reali, mentre le colonne indicano le entità predette dal modello.
È importante notare che, oltre alle entità selezionate, sono state aggiunte una riga e una colonna etichettate come "NON-ENTITÀ".
Come già descritto in 3.1, l'ultima riga riporta le parole che, pur non appartenendo a nessuna delle entità elencate, sono state erroneamente classificate come tali dal modello. In particolare, l'ultima riga indica i casi in cui il modello ha applicato l'anonimizzazione in modo eccessivo.
Al contrario, l'ultima colonna evidenzia le parole che avrebbero dovuto essere anonimizzate come informazioni sensibili, ma che il modello non ha riconosciuto e quindi non ha anonimizzato. Si parla quindi di fallimento completo del modello nell'anonimizzazione.

Poiché l'obiettivo è escludere le informazioni sensibili dai curriculum, è fondamentale prestare particolare attenzione all'ultima colonna della tabella. Questa colonna rappresenta errori più critici rispetto ad altri, poiché il modello fallisce completamente nell'anonimizzazione delle entità sensibili, non applicando alcuna misura di anonimizzazione.
In questo contesto, quest'ultima tipologia di errore è più grave rispetto a quando il modello identifica erroneamente un'entità come un'altra. In questi casi, infatti, sebbene il modello commetta un errore di identificazione, riconosce comunque

che si tratta di un'informazione sensibile e applica l'anonimizzazione. Pertanto, anche se l'identificazione non è corretta, l'obiettivo finale di anonimizzare l'informazione viene rispettato.

Dalla tabella 4.7, si può facilmente dedurre che su questi 50 curricula di campione, le entità che maggiormente il modello non è stato in grado di riconoscere completamente sono le ragioni sociali, le professioni e le date.

Di seguito sono riportate, per ciascuna entità analizzata, le percentuali relative ai diversi esiti dell'anonimizzazione sul totale delle entità esistenti:

- la percentuale di casi in cui l'anonimizzazione è stata effettuata correttamente (equivalente al valore di Recall per l'entità specifica)
- la percentuale di casi in cui l'anonimizzazione è stata completata, ma con l'assegnazione di un'etichetta errata;
- la percentuale di casi in cui il modello ha completamente fallito nel processo di anonimizzazione.

I risultati sono riportati nella tabella 4.8.

	Anonimizzazione corretta	**Anonimizzazione con etichetta errata**	**Fallimento anonimizzazione**
PERSON	99.1749	0.3300	0.4951
PHONE_NUMBER	100	0	0
EMAIL_ADDRESS	100	0	0
INDIRIZZO	96.9072	2.0619	1.0309
LOCATION	99.4569	0	0.5431
RAGIONE_SOCIALE	68.2883	20	11.7117
PROFESSIONE	25.3906	0.5859	74.0234
IT_FISCAL_CODE	0	100	0
CODICE_POSTALE	100	0	0
CREDIT_CARD	0	0	0
DATE_TIME	63.8821	1.0442	35.0737
IT_VAT_CODE	100	0	0
BANCA	100	0	0
LEGGE	100	0	0

Tabella 4.8: Percentuali di anonimizzazione

Nel caso analizzato, l'interesse è quello di anonimizzare, quindi la seconda colonna non viene considerata un errore, ma anzi per avere una percentuale complessiva di quanto il modello ha funzionato nell'anonimizzazione si devono sommare le percentuali della prima e della seconda colonna.
Come già accennato precedentemente, si verifica che le entità che il modello fallisce maggiormente di anonimizzare sono le ragioni sociali con quasi il 12%, le professioni con circa il 74% e infine le date con circa il 35%. Si può osservare che le entità più rilevanti ai fini dell'anonimizzazione, come *nomi*, *codici fiscali*, *email*, *indirizzi* e *location*, sono state correttamente anonimizzate dal modello.

Inoltre, i pochi casi di errore riscontrati riguardano entità non sensibili, il cui mancato anonimato non compromette la riservatezza dei dati personali. Pertanto, lo scopo primario dell'anonimizzazione, ovvero garantire la protezione dei dati sensibili, è stato pienamente raggiunto.

Infine, vengono riportati i valori delle metriche Precision, Recall e F-score per ciascuna entità nella tabella 4.9.

	Precision	**Recall**	**F-score**
PERSON	0.9631	0.9917	0.9772
PHONE_NUMBER	0.1739	1	0.2963
EMAIL_ADDRESS	1	1	1
INDIRIZZO	0.7176	0.9691	0.8246
LOCATION	0.9128	0.9946	0.9519
RAGIONE_SOCIALE	0.8293	0.6829	0.7490
PROFESSIONE	0.9286	0.2539	0.3988
IT_FISCAL_CODE	0	0	0
CODICE_POSTALE	0.9744	1	0.9870
CREDIT_CARD	0	0	0
DATE_TIME	0.9747	0.6388	0.7718
IT_VAT_CODE	0.7308	1	0.8444
BANCA	0.0631	1	0.1186
LEGGE	0.8092	1	0.8945

Tabella 4.9: Metriche per entità

Si può notare che i valori di *Precision* per le entità PHONE_NUMBER e BANCA sono piuttosto bassi; ciò significa che, quando il modello individua queste entità, spesso lo fa in modo errato, riconoscendo qualcosa che non è effettivamente l'entità corretta.
I valori di *Precision* per INDIRIZZO, RAGIONE_SOCIALE, IT_VAT_CODE e LEGGE sono compresi tra il 70% e 80% circa; mentre per tutti gli altri valori si ottengono valori sopra al 90%.

Si osservi che la colonna *Recall* corrisponde alla prima colonna della tabella 4.8. Questa metrica, infatti, rappresenta la percentuale di entità classificate (e quindi anonimizzate) correttamente rispetto al totale delle entità reali, dato dalla somma dei valori di TP e FN.

I risultati di *Recall* sono molto elevati per la maggior parte delle entità, il che indica che il modello, in generale, effettua correttamente l'anonimizzazione, includendo l'assegnazione dell'etichetta corretta.
Considerando solo l'obiettivo dell'anonimizzazione, questi valori vengono ulteriormente aumentati includendo i casi in cui l'assegnazione dell'etichetta è stata errata, ma l'entità è stata comunque anonimizzata, come già menzionato in precedenza. Gli unici valori di *Recall* non elevati riguardano le entità RAGIONE_SOCIALE, PROFESSIONE e DATE_TIME.

L' *F-score* rappresenta una misura che considera sia la *Precision* sia il *Recall*, essendo la loro media armonica. Pertanto, è una metrica complessiva che offre una valutazione globale delle prestazioni del modello per una determinata entità.

Quando l'*F-score* è elevato, indica che il modello sta funzionando bene: non solo ha un'alta *Precision* (quindi, quando riconosce un'entità, lo fa correttamente), ma riesce anche a identificare la maggior parte delle entità presenti che corrisponde ad un alto valore di Recall.

Nel caso dell'entità PROFESSIONE, si osserva un *F-score* basso, pari a 0.3988, con una *Precision* molto alta di 0.9286 e *Recall* basso pari a 0.2539.

Questo suggerisce che, sebbene il modello riconosca correttamente le entità che individua, non riesce a identificare tutte quelle presenti.

Al contrario, per le entità PHONE_NUMBER e BANCA, l'*F-score* è basso a causa di un *Recall* elevato ma una *Precision* bassa, il che significa che il modello riconosce molte entità, ma compie frequenti errori, identificando spesso entità errate.

Per avere una metrica globale, facendo riferimento a quanto già osservato nella sezione 3.4, è stato scelto di effettuare una micro media in quanto le entità erano sbilanciate ovvero ci sono entità che sono state riconosciute molto più di altre.
I risultati sono riportati nella tabella 4.10 seguente:

Precision	**Recall**	**F-score**	**Accuracy**
0.8999	0.7941	0.8437	0.9802

Tabella 4.10: Metriche globali

Osservazione 4.6. Sono stati riportati solo i risultati di Presidio con l'integrazione del modello perchè i risultati del caso in cui è stato utilizzato solo il modello esterno senza Presidio erano analoghi e non hanno evidenziato differenze sostanziali.

4.4 Clustering

In questa sezione si intende illustrare l'approccio utilizzato per effettuare il clustering dei curricula. Lo scopo del clustering è ottenere una descrizione dei profili dei candidati che compongono il dataset.
Tradizionalmente, i metodi per il clustering di documenti testuali sfruttavano rappresentazioni come Bag of Words 2.3.1 o TF-IDF 2.3.2, che descrivono i testi in base alla frequenza delle parole presenti. Tuttavia, come già descritto in 2.3 tali approcci non considerano il significato semantico delle parole, riducendo l'efficacia in contesti più complessi.

Per superare questa limitazione, i modelli più avanzati utilizzano tecniche di embedding 2.3.3, che consentono di rappresentare i documenti catturando le relazioni semantiche tra le parole.

In questa tesi è stato adottato un metodo di clustering basato sull'algoritmo k-means, utilizzando una rappresentazione dei documenti che integra gli embedding con i risultati del riconoscimento delle entità (NER) ottenuti in precedenza.

4.5 Metodologia

Di seguito vengono presentati i diversi passaggi che caratterizzano il metodo utilizzato per il clustering dei curricula.

1. Suddivisione dei curricula in frasi.
2. Riconoscimento e anonimizzazione delle entità tramite Named Entity Recognition (NER).
3. Conteggio delle entità riconosciute per ogni frase.
4. Calcolo del peso associato a ciascuna frase.
5. Calcolo degli embedding delle frasi e combinazione tramite pooling.
6. Normalizzazione vettori di embedding
7. Riduzione della dimensionalità degli embedding.
8. Clustering mediante algoritmo k-means.

Suddivisione dei curricula in frasi

Sia d un documento (curriculum vitae). Il documento viene suddiviso in m frasi $\{s_1, s_2, \ldots, s_m\}$ utilizzando il modello di NLP `it_core_news_lg` di SpaCy, descritto in 4.2.2.

Osservazione 4.7. Il numero di frasi m varia in base alla lunghezza e alla complessità del documento d.

Riconoscimento e anonimizzazione delle entità tramite Named Entity Recognition (NER)

A ciascuna frase s_i ($i = 1, \ldots, m$), viene applicato il riconoscimento delle entità tramite *Presidio*, integrato con il modello `DeepMount00/Italian_NER_XXL` ovvero il metodo già precedentemente selezionato in 4.2.2.

Le entità rilevate sono le stesse già considerate in 4.2.2. Esse sono state suddivise nelle seguenti tre classi:

- E_{maggiori} = {PROFESSIONE, RAGIONE_SOCIALE};
- E_{minori} = {PERSON, PHONE_NUMBER, EMAIL_ADDRESS, IT_FISCAL_CODE, CODICE_POSTALE, CREDIT_CARD, IT_VAT_CODE, BANCA, LEGGE}
- E_{uguali} = {INDIRIZZO, LOCATION, DATE_TIME}

Una volta rilevate, tutte le entità sono state anonimizzate, ad eccezione di quelle appartenenti all'insieme $E_{maggiori}$. Questa scelta è motivata dal fatto che si vuole che il processo di clustering si basi proprio su queste entità per differenziare i documenti; di conseguenza, anonimizzarle comprometterebbe l'efficacia del clustering. Ci si aspetta che le persone che svolgono la stessa professione siano nello stesso cluster. Si è scelto pertanto di favorire quetsto fatto pesando di più le frasi in cui è presente il riferimento alla professione e/o ragione sociale.

Conteggio delle entità riconosciute per ogni frase

Per ciascuna frase s_i, si calcolano:

- $N_{maggiori}(s_i) = |E_{\text{maggiori}}(s_i)|$
- $N_{minori}(s_i) = |E_{\text{minori}}(s_i)|$
- $N_{uguali}(s_i) = |E_{\text{uguali}}(s_i)|$

il conteggio delle entità rilevate nella frase s_i rispettivamente per le categorie E_{maggiori}, E_{minori}, e E_{uguali}.

Calcolo del peso associato a ciascuna frase

Il peso di ciascuna frase s_i, denotato con w_{s_i}, è calcolato secondo le seguenti regole:

$$w_{s_i} = \begin{cases} 3 & \text{se } |E_{\text{maggiori}}(s_i)| > 0, \\ 0.5 & \text{se } |E_{\text{minori}}(s_i)| > 0 \text{ e } |E_{\text{maggiori}}(s_i)| = 0, \\ 1 & \text{altrimenti.} \end{cases}$$

In particolare, se una frase contiene entità appartenenti alla categoria $E_{maggiori}$ (ad esempio, professione o ragione sociale), che sono fondamentali per distinguere i curricula, viene assegnato un peso elevato alla frase. In assenza di tali entità, si verifica la presenza di entità appartenenti alla categoria E_{minori} (come

person, phone number, email address); in questo caso, alla frase viene attribuito un peso inferiore. Infine, se non vengono rilevate entità appartenenti né a $E_{maggiori}$ né a E_{minori}, alla frase viene assegnato un peso pari a 1, garantendo che ogni frase contribuisca comunque alla rappresentazione finale del documento. Si assume quindi che le frasi contenenti numeri di telefono, indirizzi e simili siano di natura "introduttiva" o "conclusiva" nel contesto del curriculum. Di conseguenza, si ritiene che esse non includano informazioni più rilevanti per il clustering. Si tratta di un'assunzione e comunque non viene mai assegnato un peso nullo alle frasi.

Calcolo degli embedding delle frasi e combinazione tramite pooling

Per ogni frase s_i, si calcola l'embedding $\mathbf{e}_{s_i}$ sulla frase anonimizzata utilizzando il modello *efederici/sentence-bert-base*. Questo modello utilizza la struttura BERT 2.2.3 per creare embedding contestuali di dimensione 768. Gli embedding delle frasi di un documento d vengono combinati in un embedding unico $\mathbf{e}_d$ tramite una strategia di pooling pesata basata sui pesi w_{s_i}. Formalmente, l'embedding finale del documento è dato da:

$$\mathbf{e}_d = \frac{\sum_{i=1}^{m} w_{s_i} \cdot \mathbf{e}_{s_i}}{\sum_{i=1}^{m} w_{s_i}}.$$

Grazie a questo step è stata creata la matrice $X \in \mathbb{R}^{n \times p}$ dove n è il numero di curriculum e $p = 768$ dimensione dell'embedding che rappresenta il documento.

Normalizzazione

Quando si utilizzano gli embedding, la misura di similarità più comune è la similarità coseno. Come evidenziato in 2.3.4, esiste una relazione diretta tra la similarità coseno e la distanza euclidea per vettori normalizzati: una maggiore similarità coseno tra due vettori corrisponde a una minore distanza euclidea tra gli stessi vettori normalizzati. Di conseguenza, embedding con alta similarità coseno risultano vicini rispetto alla distanza euclidea, a condizione che i vettori siano normalizzati. Dato che l'algoritmo k-means utilizza la distanza euclidea, la matrice X è stata normalizzata.

Riduzione della dimensionalità

Prima di applicare l'algoritmo k-means 1.3.3, gli embedding dei documenti $\mathbf{e}_d$ ottenuti precedentemente sono stati ridotti ad una dimensionalità inferiore tramite una tecnica di riduzione chiamata UMAP (Uniform Manifold Approximation and Projection).

Clustering mediante algoritmo k-means

Infine, gli embedding ridotti sono clusterizzati utilizzando l'algoritmo k-means.

Nella sezione precedente sono stati descritti i passaggi chiave del metodo utilizzato. Si vuole ora porre particolare attenzione al processo di riduzione della dimensionalità. È infatti importante notare che il metodo adottato presenta una serie di parametri che influenzano sia la modalità di riduzione dei dati, sia il successivo processo di clustering.

Per valutare adeguatamente il comportamento del metodo proposto, è stata condotta un'analisi preliminare su un dataset fittizio composto da profili di candidati, per i quali sono disponibili le etichette di riferimento. Questo approccio ha permesso di esplorare e calibrare le impostazioni dei parametri in un contesto controllato.

In particolare è stato utilizzato un dataset fittizio di 100 curriculum divisi nelle seguenti 8 classi:

- economico;
- educativo;
- editoria;
- legale;
- ristorazione;
- scientifico;
- ingegneristico-tecnologico;
- data-science.

Questo dataset è stato utilizzato per individuare la configurazione dei parametri di UMAP che permette di ottenere cluster il più possibile coerenti con le etichette disponibili. Questo approccio ha consentito di affinare i parametri del metodo, garantendo una riduzione della dimensionalità che preserva le strutture dei dati rilevanti per il clustering.

I parametri di UMAP che devono essere definiti sono i seguenti:

- `n_neighbors`: Scegliere un valore piccolo implica un'interpretazione molto locale, capace di catturare i dettagli della struttura. Al contrario, optare per un valore grande comporta una stima basata su regioni più ampie, sacrificando alcuni dettagli della struttura locale [12].
- `min_dist`: definisce la distanza minima tra i punti nella rappresentazione a bassa dimensionalità, controllando il livello di compressione dei cluster.
- `n_components`: specifica la dimensione dello spazio di proiezione.

Per testare questi parametri, sono state definite delle liste di possibili valori per ciascun parametro. Successivamente, è stato eseguito l'algoritmo `k-means` per tutte le possibili combinazioni dei parametri.

Per ciascuna combinazione, sono state calcolate le seguenti metriche esterne al fine di valutare le performance del clustering associato a quella particolare combinazione di parametri:

- **Adjusted Rand Index** (1.4.1);
- **Fowlkes-Mallows** (1.4.1);
- **Homogeneity** (1.4.1);
- **Completeness** (1.4.1);
- **V-measure** (1.4.1).

La configurazione ottimale dei parametri è stata identificata come quella che massimizzava le metriche descritte sopra.

Dagli esperimenti risulta che la configurazione migliore e i corrispondenti valori delle metriche sono i seguenti:

k	n_neighbors	min_dist	dimensioni	adjusted_rand_score	fowlkes_mallows_score	homogeneity_score	completeness_score	v_measure_score
7	10	0.3	30	0.66	0.70	0.78	0.83	0.81

Tabella 4.11: Risultati delle metriche per la configurazione ottimale dei parametri.

Di seguito viene mostrata la distribuzione delle etichette nei cluster utilizzando la configurazione ottimale di parametri.

	Cluster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
Economico	15	0	0	0	0	0	3
Educativo	0	12	0	0	2	0	0
Editoria	0	0	9	0	0	0	0
Legale	8	1	0	0	0	0	0
Ristorazione	0	0	0	13	0	0	0
Scientifico	0	0	0	1	14	5	0
Ingegneristico-Tecnologico	0	0	0	0	0	9	0
Data-Science	0	0	0	0	0	0	9

Tabella 4.12: Distribuzione delle categorie nei cluster.

In una seconda fase, il clustering k-means è stato eseguito per diversi valori di k sul dataset reale, utilizzando come configurazione dei parametri di UMAP quella ottimale ottenuta precedentemente con il dataset fittizio. Nel trasferire la configurazione dei parametri corrispondente all'esperimento migliore ottenuto sul dataset fittizio a quello reale, si è assunto che i dataset fittizio e reale siano simili.

Per questa fase, è stato scelto un dataset ridotto a 100 curriculum, rispetto ai 1700 originali, per permettere una valutazione manuale dei risultati.

Il numero ottimale di cluster è stato determinato eseguendo il clustering per vari valori di k e valutando i risultati tramite le seguenti metriche interne:

- **Inerzia** (1.4.2);
- **Silhouette coefficient** (1.4.2);
- **Davies-Bouldin index** (1.4.2);
- **Calinski-Harabasz index** (1.4.2).

Di seguito si riportano i risultati delle metriche interne, per determinare il numero di cluster ottimale per il dataset reale.

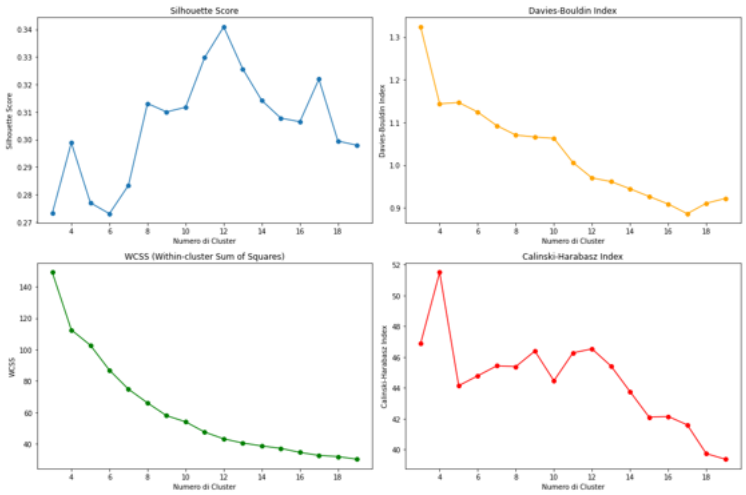


Figura 4.1: Metriche interne per scelta numero di cluster ottimale

Come già descritto in 1.4.2, valori elevati del Silhouette score e del Calinski-Harabasz index indicano una buona qualità del clustering, mentre per l'indice di Davies-Bouldin valori più bassi sono segno di un buon clustering. Per quanto riguarda l'inerzia, il numero ottimale di cluster si determina quando la curva smette di scendere in modo significativo.

Sulla base dell'analisi delle metriche riportata in 4.1, è stato scelto un numero di cluster pari a 12.

Di conseguenza, k-means è stato rieseguito con $k = 12$, e i risultati ottenuti sono presentati di seguito.

I risultati sono stati valutati "manualmente" e a ciascun cluster è pertanto stato assegnato un nome che rappresentasse gli elementi al suo interno. Ecco gli ambiti trovati:

- educativo-pedagogia-inclusione;
- polizia;
- ingegneria civile-architettura urbanistica;
- economico;
- progettazione e gestione progetti;
- ingegneria civile-architettura urbanistica;
- legale;

- ambito civile-amministrativo-urbanistica;
- polizia;
- informatico-scientifico;
- medico-sanitario;
- consulenza organizzativa-formazione per pubbliche relazioni

Si osserva che sono stati creati due cluster per l'ambito della polizia e tre cluster per il settore dell'ingegneria civile e urbanistica. In questo caso, il clustering risulta omogeneo, in quanto soddisfa il criterio di omogeneità (definito in 1.4.1), che valuta la distribuzione delle classi all'interno di ciascun cluster e considera un clustering omogeneo quando, all'interno di ogni cluster, è presente una sola classe; piuttosto che soddisfare il criterio di completeness1.4.1), che si concentra sulla distribuzione delle classi tra i cluster e richiede che ogni classe venga assegnata a un solo cluster. Di seguito vengono riportati i risultati per i cluster medico-sanitario, polizia, legale:

Cluster medico-sanitario
nessuna esperienza lavorativa
direttore medico e sanitario
Infermiere in unità di medicina generale e pronto soccorso/coordinatore caposala
Docente di igiene e medicina preventiva/assistente di laboratorio presso ospedali
Docente di igiene e medicina preventiva/assistente di laboratorio presso ospedali
ostetrica

Tabella 4.13: Cluster medico-sanitario

Cluster polizia
dipendente pubblico/funzionario di servizio di Polizia Locale
Commissario dirigente ufficio immigrazione
Dipendente amministrazione comunale in qualità di comandante di polizia
Agente di polizia /Docente corsi di polizia
Dirigente a tempo indeterminato di Polizia Locale
Comandante di polizia
Comandante polizia/direzione tecnico-operativa di personale di polizia
Polizia locale /istruttore formatore per la sicurezza
Commissario maggiore con mansioni di videosorveglianza/ infortunistica stradale Polizia municipale
Agente di polizia giudiziaria
Polizia locale/docente corsi per prevenzione frodi

Tabella 4.14: Cluster Polizia

Cluster legale
Avvocato/docente di diritto processuale/albo arbitri per la camera arbitrale della camera di commercio
Avvocato/docente di diritto processuale/albo arbitri per la camera arbitrale della camera di commercio
Pratica forense presso avvocato/agente polizia/ iscrizione agrotecnici e dottori agronomi
Iscritto albo avvocati/avvocato amministrativista che si occupa in particolar edi urbanistica, edilizia, lavori pubblici... attività seminariale per corso diritto amministrativo
Ricercatore in diritto amministrativo/avvocato in diritto amministrativo e civile/incarichi in qualità di esperto nelle materie urbanistica edilizia e diritto dell'ambiente
Socio fondatore avvocato operante nei settori di diritto dell'ambiente igiene salute e sicurezza sul lavoro e diritto amministrativo
Iscritto albo avvocati in diritto civile e processuale civile amministrazione di sostegno/mediatore civile
Fiscalista e avvocato mediatore civile
Avvocato in diritto penale diritto civile diritto amministrativo e diritto del lavoro

Tabella 4.15: Cluster Legale

Un aspetto importante da considerare nei risultati del clustering testuale è la non omogeneità interna dei curricula. Infatti, un candidato potrebbe aver intrapreso un percorso professionale in un settore specifico per poi cambiare

radicalmente ambito nel corso degli anni. Questo rende complessa la categorizzazione, poiché un singolo curriculum può contenere informazioni relative a più cluster. Inoltre, esistono casi particolari, come i curricula nel caso “nessuna esperienza lavorativa" in 4.13, i quali non si inseriscono naturalmente in alcun cluster, ma devono comunque essere associati forzatamente a uno di essi per necessità di categorizzazione. Questa eterogeneità può influire sulla coerenza e sull'interpretabilità dei cluster generati.

Capitolo 5

Conclusioni

In conclusione, in questa tesi sono stati analizzati i curricula vitae, utilizzando tecniche di Named Entity Recognition (NER) e di clustering. Il NER è stato impiegato per anonimizzare i dati sensibili ed estrarre informazioni rilevanti per il clustering, individuando entità come professioni e ragioni sociali. Il clustering, a sua volta, ha sfruttato le entità riconosciute dal NER per ottenere una descrizione significativa dei profili dei candidati.

Dall'analisi condotta, si evidenzia che il modello NER utilizzato ha identificato con maggiore frequenza entità quali nomi di persona, luoghi, ragioni sociali e date.

Le entità più rilevanti per l'anonimizzazione, come nomi, codici fiscali, email, indirizzi e luoghi, sono state gestite correttamente dal modello. Tuttavia, per le entità relative a professioni e ragioni sociali, è emerso che il modello, pur essendo accurato nel riconoscere le entità rilevate, non è riuscito a individuare tutte quelle effettivamente presenti nel testo.

Gli errori riscontrati, però, hanno riguardato principalmente entità non sensibili, il cui mancato anonimato non compromette la protezione dei dati personali. Pertanto, l'obiettivo principale dell'anonimizzazione, ossia garantire la riservatezza delle informazioni sensibili, è stato pienamente raggiunto. Per migliorare l'efficacia del riconoscimento delle entità legate a professioni e ragioni sociali, potrebbe essere utile affinare il training del modello utilizzando dataset più specifici, con una maggiore rappresentanza di tali categorie.

Per quanto riguarda il clustering, l'analisi ha permesso di ottenere una descrizione piuttosto accurata dei profili professionali. I cluster identificati riflettono categorie legate ad ambiti come medico-sanitario,educativo, legale,ingegneristico-urbanistico, economico, di progettazione e gestione progetti, polizia e di consulenza organizzativa/formazione per relazioni pubbliche. Si osserva, in particolare, la formazione di due cluster per il settore della polizia e di tre cluster per l'ambito ingegneristico-urbanistico. Questo evidenzia una parziale violazione del criterio di completezza, secondo il quale ogni classe dovrebbe essere rappresentata da un singolo cluster. Tuttavia, i cluster risultano omogenei, raggruppando profili appartenenti alla stessa categoria.

In conclusione, i cluster ottenuti sono coerenti e offrono una rappresentazione significativa dei profili analizzati. Tra le possibili aree di miglioramento si propone di perfezionare ulteriormente il modello per il riconoscimento di professioni e ragioni sociali e di testare nuove configurazioni dei parametri al fine di evidenziare con maggiore enfasi queste entità, che costituiscono la base fondamentale per il processo di clustering.

Bibliografia

[1] *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, Third Edition* ,Daniel Jurafsky,James H. Martin

[2] *Representation Learning for Natural Language Processing*, Zhiyuan Liu, Yankai Lin, Maosong Sun

[3] *A survey on Deep Learning for Named Entity Recognition*,IEEE transactions on knowledge and data engineering,2020, Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li

[4] *Explain TF-IDF in Natural Language Processing.*,Mudadla, S. (2023, November 10),Medium

[5] *The Art of Pooling Embeddings.*,Leys, Mathias. Medium, ML6team, 20 giugno 2022.

[6] *V-Measure: A conditional entropy-based external cluster evaluation measure*,Andrew Rosenberg and Julia Hirschberg Department of Computer Science Columbia University New York, NY 10027

[7] *Documentazione libreria Scikit-learn*, https://scikit-learn.org/1.5/modules/clustering.html

[8] *MIT 9.520/6.860: Statistical Learning Theory Lecture2 - Statistical Learning Theory* , Lorenzo Rosasco

[9] *K-means clustering slides*, Marco Bressan, Universit'a degli Studi di Milano April 20, 2021

[10] *Documentazione ufficiale Microsoft Presidio*, https://microsoft.github.io/presidio/

[11] *Modello DeepMount00/Italian NER XXL*, https://huggingface.co/DeepMount00/Italian_NER_XXL

[12] *Considerably Improving Clustering Algorithms Using UMAP Dimensionality Reduction Technique: A Comparative Study*,Mebarka Allaoui, Mohammed Lamine Kherfi and Abdelhakim Cheriet3

[13] *Learning Theory from First Principles*, Francis Bach, August 27, 2024

[14] *k-means++: The Advantages of Careful Seeding* David Arthur, Sergei Vassilvitskii

[15] *A survey on Named Entity Recognition — datasets, tools, and methodologies* Basra Jehangir, Saravanan Radhakrishnan, Rahul Agarwal

[16] *A survey of named entity recognition and classification*David Nadeau, Satoshi Sekine National Research Council Canada / New York University

[17] *Support Vector Machines*Ingo Steinwart, Andreas Christmann

[18] *Understanding Machine Learning- From Theory to Algorithms* Shai Shalev-Shwartz, Shai Ben-David

[19] *A Text Document Clustering Method Based on Weighted BERT Model* Yutong Li, Juanjuan Cai, Jingling Wang