Agenzia nazionale per le nuove tecnologie, l'energia e lo sviluppo economico sostenibile

Ministero della Transizione Ecologica

Tecniche di Machine Learning, Big Data Analytics e Natural Processing con applicazione all'analisi di Social Media

B. Di Martino, A. Esposito, M. Graziano, L. Colucci Cante, G. J. Pezzullo, V. Bombace

Università degli Studi della Campania *Luigi Vanvitelli*
Dipartimento di Ingegneria

Report RdS/PTR2021/054

TECNICHE DI MACHINE LEARNING, BIG DATA ANALYTICS E NATURAL PROCESSING CON APPLICAZIONE ALL'ANALISI DI SOCIAL MEDIA

B. Di Martino, A. Esposito, M. Graziano, L. Colucci Cante, G. Pezzullo, V. Bombace (Dipartimento di Ingegneria Università degli studi della Campania "L. Vanvitelli")

Dicembre 2021

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero della Transizione Ecologica- ENEA
Piano Triennale di Realizzazione 2019-2021 – III annualità
Obiettivo: Tecnologie
Progetto: Tecnologie per la penetrazione efficiente del vettore elettrico negli usi finali
Work package: Local Energy District
Linea di attività: Definizione di Tecniche di Sentiment Analysis e Big Data Analytics basate
Su NLP applicati alla Twitter Analysis
Responsabile del Progetto: Claudia Meloni, ENEA
Responsabile del Work package: Claudia Meloni, ENEA

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione
"Tecniche di Sentiment Analysis e Big Data Analytics per Twitter Analysis"
Responsabile scientifico ENEA: ENEA: Dr. Gregorio D' Agostino
Responsabile scientifico UniCampania - Dip. Ingegneria: Prof. Beniamino Di Martino

Indice

Sommario

Le molteplici attività svolte vengono riportate organicamente nel presente rapporto. Sono state affrontate problematiche di ricerca e definite soluzioni concrete per la realizzazione di una "pipeline" in grado di esaminare notizie nella lingua italiana (la letteratura e le realizzazioni esistenti sono più abbondanti nel caso della lingua inglese) al fine di estrarre informazioni legate al mondo delle "comunità energetiche". In particolare, sono state analizzate le pubblicazioni relative al concetto di "energy community" e la ricognizione di eventi specifici che le riguardano.

Le tecniche di analisi NLP (Natural Language Programming) utilizzate hanno anche consentito di valutare la predisposizione (stato d'animo) degli autori nei confronti della tematica, ovvero si è realizzata la cosiddetta "sentiment analysis".

Le tecniche poste in atto non sono ottimizzate per eseguire ad una analisi dei dati raccolti in tempo reale (in streaming), ma in modalità asincrona successiva al processo di acquisizione. Comunque, gli algoritmi ed il software messo a punto si prestano ad applicazioni asincrone. L'intera architettura della pipeline può considerarsi a tuti gli effetti "stream ready", ovvero pronta per le applicazioni in tempo reale.

Metodologie di "reti neurali" e "Word Embedding" sono state principalmente utilizzate per classificare i testi, mentre alcune tecniche di "machine learning" sono state preferite per l'analisi dei testi poco strutturati.

La base fondamentale per l'addestramento dei sistemi è stata il "corpus linguistico" fornito dall'ENEA.

L'applicazione delle stesse tecniche di big data e machine learning ha anche consentito la validazione dell'ontologia fornita da ENEA.

Tutto il software è stato sviluppato in "container" (docker) e utilizzando una struttura di progettazione modulare per consentirne la eventuale portabilità sulla piattaforma (kubernetes) ECListener di ENEA e l'eventuale integrazione selettiva nel sistema attualmente in produzione.

1 Introduzione

Il presente descrive le attività di ricerca, nonché le soluzioni architetturali e implementative attuate per la creazione di una pipeline in grado di esaminare notizie, raccolte da fonti testuali variegate in lingua italiana, che fosse in grado, non solo di selezionare esclusivamente i contenuti relativi al tema delle **Comunità Energetiche**, ma anche recuperare informazioni relative agli eventi descritti e di comprendere i toni con cui la notizia stessa è stata data (Di Martino et al., 2020).
Detta pipeline permette, non solo di monitorare quanto e come le notizie relative alle Comunità Energetiche circolino nella rete informativa italiana, ma anche se esse siano viste di buon occhio e presentate in maniera positiva ai lettori. Inoltre, il software che sarà descritto dal presente rapporto potrà fornire indicazioni utili per individuare tutti gli eventi che ruotano intorno alla realtà delle Comunità Energetiche, così da definire una sorta di "mappa di interesse" sul tema, sebbene limitata allo Stato Italiano.
Nel rispetto degli obiettivi originali su cui si basa la linea di attività, il presente documento non è focalizzato su un'analisi streaming delle informazioni prelevate dai Tweet, ma in Batch, andando ad esaminare testi di news scaricate in momenti diversi della giornata. Tuttavia, tutte le attività di ricerca svolte nell'ambito della linea di attività e descritte nel rapporto e le tecnologie impiegate per attuare processi di Natural Language Processing, Opinion mining e Sentiment Analysis, sono completamente compatibili con un'analisi Streamin; la stessa architettura ideata per realizzare il software a corredo è completamente Stream-ready, e ha le necessarie caratteristiche di scalabilità e adattabilità per poter essere utilizzata anche nell'ambito dell'analisi di Tweet. Come risulterà evidente dalla descrizione della pipeline utilizzata per descrivere le diverse attività svolte nell'ambito della linea assegnata, nonché delle tecniche impiegate per implementare i componenti software associati a tali attività, sono state applicate sia metodologie appartenenti al mondo Big Data, sia derivanti dal Machine Learning. L'uso delle tecnologie Big Data è necessario in quanto le News presentano caratteristiche complesse, sono essenzialmente testi non strutturati che necessitano di azioni di pre normalizzazione e cura, e possono avere dimensioni anche importanti, soprattutto se scaricati in batch. Reti neurali e Word Embedding sono stati principalmente utilizzati per classificare i testi in ingresso al sistema, validare le diverse Ontologie e individuare cluster di termini che fossero descrittivi del dominio analizzato.
La principale problematica che si riscontra nell'esaminare le notizie relative alle Comunità Energetiche sta nella relativa difficoltà che si incontra nel definirle in maniera completamente univoca. Il concetto è ancora in continua evoluzione, e certamente l'idea che c'è alla base continuerà a mutare nel tempo. Da qui la necessità di costruire dei sistemi automatici che possano rilevare le notizie relative, utilizzando sia concetti che notoriamente sono legati a quello di Comunità Energetica, sia estrapolati da analisi successive dei testi già esaminati e ritenuti pertinenti.
A tale scopo, il software sviluppato segue contemporaneamente più approcci, sia per l'analisi di inerenza del testo delle news, sia per l'estrazione degli eventi e delle relative informazioni, che per la Sentiment Analysis relativa.
Il presente rapporto è strutturato nel seguente modo: vengono innanzitutto presentati e riassunti gli obiettivi generali del software, di cui viene anche fornita una descrizione dell'architettura e dei relativi componenti. Ogni componente viene, in seguito, descritto nelle sue funzionalità e nei suoi obiettivi specifici, esplicitando le tecnologie utilizzate per realizzarlo.
Il deliverable si chiude con considerazioni su quanto è stato portato a termine, e offre spunti per ulteriori lavori futuri.

2 Obiettivi

In questo paragrafo sono identificati gli obiettivi specifici che i task portati avanti nell'ambito del deliverable e il software sviluppato a corredo, tutti descritti nel presente documento, si prefiggono. Essi riprendono ed arricchiscono quanto già detto in fase introduttiva.

Identificazione di News Inerenti il contesto delle Comunità Energetiche
Per poter analizzare in maniera significativa le news, occorre poter discernere tra quelle che effettivamente ricoprono il tema analizzato, e quelle che invece non sono affatto di interesse. Di seguito, indicheremo questa attività come "**Analisi di Pertinenza**" e il componente che utilizzeremo verrà indicato come "**Validatore**". L'identificazione delle news è certamente una attività preliminare necessaria all'analisi successiva, e comprende anche una prima fase di recupero e ripulitura dei testi da inviare ai successivi analizzatori, come verrà esplicitamente descritto nella sezione dedicata al componente specifico.

Recupero di informazioni relative ad eventi correlati alle Comunità Energetiche
Diverse sono le informazioni che ci interessano per quanto riguarda l'analisi delle Comunità Energetiche. Nello specifico, siamo interessati ad individuare nelle varie notizie già identificate nel componente di validazione, le seguenti tipologie di evento:

- **Convegni, conferenze, workshop** che indicano partecipazione, da parte della comunità scientifica nazionale, alla ricerca sul tema. Si tratta inoltre di attività di disseminazione, che spesso prevedono anche l'accesso del pubblico, e che possono rappresentare delle ottime occasioni per far conoscere il tema anche ai non addetti ai lavori, e a sensibilizzare l'opinione pubblica a riguardo. Anche per questo motivo siamo interessati alla Sentiment Analysis, così da comprendere come la notizia venga presentata.
- **Emanazione di decreti e ordinanze** che riguardino le Comunità energetiche. Questo tipo di evento è importante per mantenersi aggiornati continuamente riguardo l'evoluzione della legge relativamente al tema, e alle decisioni prese dall'amministrazione pubblica, sia a livello nazionale che locale.
- **Luoghi e date dei predetti eventi.** Senza questa informazione, l'identificazione degli eventi non avrebbe il giusto impatto e non potrebbe essere sfruttata a dovere.

Sentiment analysis relativa alle News scaricate dalla rete
Come spiegato nell'introduzione, oltre ad identificare notizie ed eventi relativi alle Comunità Energetiche, è necessario poterne comprendere l'opinione espressa. Per tale motivo, viene eseguita una **Sentiment analysis** relativa alle News scaricate dalla rete. Lo specifico componente che si occupa di effettuare questa analisi cattura le opinioni espresse nelle news, che spesso non riportano i fatti in maniera puramente oggettiva, ma contengono dati più soggettivi e legati al modo che hanno gli autori delle notizie di presentare l'argomento. C'è anche da aggiungere che gli articoli possono presentare stralci di interviste e commenti, che possono fornire un po' il polso della situazione. Questo tipo di analisi va certamente integrata con quella relativa ad altre fonti di informazioni, come quella dei Tweet, che sono una cartina al tornasole dell'opinione pubblica più di quanto non lo siano i testi delle News, per quanto questi ultimi siano certamente più complessi e corposi.
Verifica e popolamento di opportune ontologie
Al momento di iniziare lo sviluppo del software descritto in questo deliverable, era disponibile l'ontologia ARAKNE, contenente concetti presi dal corpus documentale di Wikipedia e utilizzati per descrivere il tema delle Comunità Energetiche. Si tratta di un'ontologia sicuramente molto ricca, sebbene priva di istanze, ma dotata solo di classi e relazioni tra esse. Tra gli obiettivi è stato preso in

considerazione quello di esaminare l'ontologia messa a disposizione, in relazione ad un nutrito corpus documentale costituito da circa 2400 testi (forniti da ENEA) ritenuti già inerenti il tema delle comunità energetiche, al fine di validare l'ontologia stessa e verificare l'effettiva aderenza dei concetti in essa descritti al tema trattato. L'ontologia non è tuttavia utile ai fini della descrizione di eventi relativi al tema delle Comunità Energetiche, e pertanto è stata definita un'ulteriore ontologia, in fase embrionale, che possa descrivere al meglio i temi di interesse per il presente deliverable. Tale ontologia è stata anche popolata con le informazioni derivanti dalle analisi ottenute dai componenti di verifica di pertinenza ed analisi degli eventi.

Visualizzazione e fruizione user-friendly delle informazioni ottenute

La raccolta di informazioni eseguita nei vari stadi del software sviluppato perde di utilità se tali informazioni non sono facilmente fruibili dagli utenti. Sono stati definiti diversi meccanismi di visualizzazione delle informazioni estratte, tutti basati su tecnologie standard e tecniche di visualizzazione note, che verranno ulteriormente dettagliate nel seguito.

3 Workflow e attività

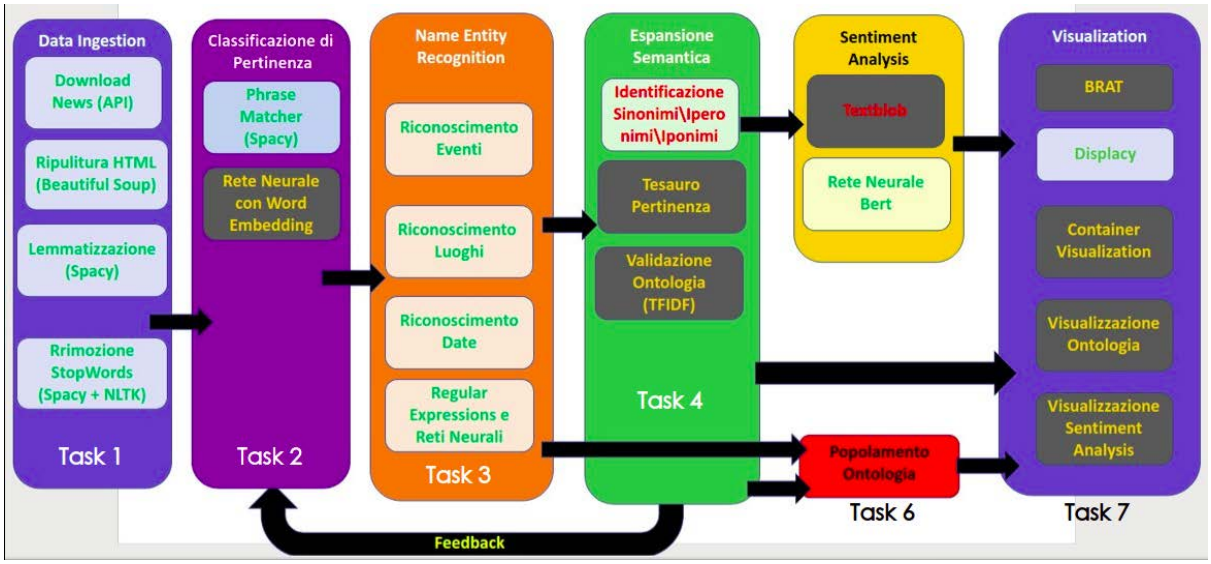

Figure 1.Workflow (modello Pipeline)

Per l'organizzazione delle attività di ricerca legate al presente Deliverable, è definito un modello a pipeline, mostrato in Figura 1, che mostra il flusso dei diversi task seguendo uno stile tipicamente Big Data.

Nella pipeline, i dati fluiscono da un componente iniziale, che si occupa essenzialmente della **Data Ingestion** delle news e della loro prima analisi e pulizia, per poi proseguire verso l'analisi di **Pertinenza** vera e propria, fino ad arrivare alla **Visualizzazione** finale dei risultati e delle informazioni. Nello specifico, la pipeline rappresenta il flusso informativo dei dati e delle informazioni, e si riflette poi nei diversi task descritti di seguito, e nei componenti software che sono stati definiti per realizzare tali task, nonché nell'architettura complessiva che sarà descritta in seguito. I colori utilizzati hanno un importante significato: in **verde** sono riportate tutte le metodologie e le tecniche che, al momento della consegna del rapporto, funzionavano correttamente e sono state debitamente testate attraverso opportuni componenti software, pronti all'uso; in **giallo** sono riportate le tecniche sperimentali, che sono state parzialmente testate e che hanno fornito comunque dei risultati soddisfacenti, sebbene non definitivi; in **rosso** le tecniche provate e poi abbandonate, in quando non fornivano i risultati attesi o si sono rivelate poco efficaci ed efficienti.

I diversi task definiti all'interno della Pipeline sono di seguito elencati brevemente.

- **Task 1: Data Ingestion**, che si occupa di recuperare i dati relative alla News in italiano, e di eseguire una serie di operazioni di Curation dei dati stessi. Il testo delle News contiene infatti una strutturazione HTML che non interessa ai fini dell'analisi, oltre a necessitare di un pretrattamento che ne permetta il corretto utilizzo ed esame attraverso le varie librerie sfruttate per la creazione dei diversi componenti.
- **Task 2: Classificazione di Pertinenza**, di cui abbiamo già accennato in precedenza, che si occupa di determinare se le news scaricate sono effettivamente inerenti il tema di interesse. Nella pipeline, si nota come si sia sviluppato il componente, di cui si tratterà meglio nella sezione dedicata, utilizzando due tecniche differenti: una basata sulla libreria Spacy, l'altra focalizzata sull'uso di Word Embedding e reti neurali.
- **Task 3: Name Entity Recognition (NER)**, focalizzata sul riconoscimento di eventi di interesse, nonché delle informazioni ad essi relative. Diversi approcci sono stati utilizzati in questa fase, dalla libreria Spacy alle Regular Expressions scritte appositamente per integrare tool pre-esistenti.
- **Task 4: Espansione Semantica**, la quale si è focalizzata sull'analizzare i termini presenti nell'ontologia e nei testi del corpus, già pre-esistente e considerato inerente il tema delle comunità energetiche, per poter esaminare l'ontologia e fornire una validazione. I dati sono stati inizialmente incrociati con quelli derivanti dall'analisi degli eventi provenienti dallo stadio precedente, utilizzando un meccanismo basato sull'uso di Wordnet e sulla ricerca di **sinonimi, iperonimi e iponimi**, ma questa tecnica non ha portato a risultati apprezzabili. In seguito è stata applicata la tecnica nota come **TF-IDF (term frequency–inverse document frequency)** per poter validare le ontologie fornite, in particolare in relazione alla pertinenza dei concetti in esse contenuti. Le informazioni ottenute, in particolare tramite espansione semantica dei termini contenuti nell'ontologia, sono stati sfruttati per fornire un feedback al Task 1 di analisi di pertinenza, soprattutto per l'approccio basato su reti neurali.
- **Task 5: Sentiment Analysis** si è focalizzata essenzialmente sulla **opinion mining** dei testi analizzati, a cui si era specificatamente interessati in termini di Positività, Neutralità e Negatività. Diverse tecniche sono state esaminate e poi adottate e testate. Due nello specifico sono state esplicitate nella pipeline: **Textblob**, che si è

  rilevato però poco efficiente, e una rete neurale basata su **Bert (Bidirectional Encoder Representations from Transformers)**, che invece ha fornito risultati molto migliori.
- **Task 6: Popolamento dell'Ontologia**, task che procede di pari passo con la Sentiment Analysis, dato che non ha bisogno del suo output per poter operare sulle ontologie stesse. L'obiettivo di questo task è inserire le informazioni estratte dai testi esaminati dai task precedenti e popolare l'ontologia, per poi procedere ad una visualizzazione delle informazioni.
- **Task 7: Visualization** finale è fondamentale per la fruizione delle informazioni. Come descritto nell'apposita sezione che si occupa di definire tutte le modalità di visualizzazione offerte, diverse sono le tecnologie che sono state esaminate, tutte basate su standard e su librerie note.

4 Architettura del Sistema

Per descrivere al meglio la struttura e il comportamento del sistema sviluppato dividiamo l'architettura in :

- Architettura logica
- Architettura Fisica

4.1 Architettura logica

L'architettura logica in grado di soddisfare i task menzionati nella sezione Workflow e Attività è strutturata nel seguente modo (Figura 2.):

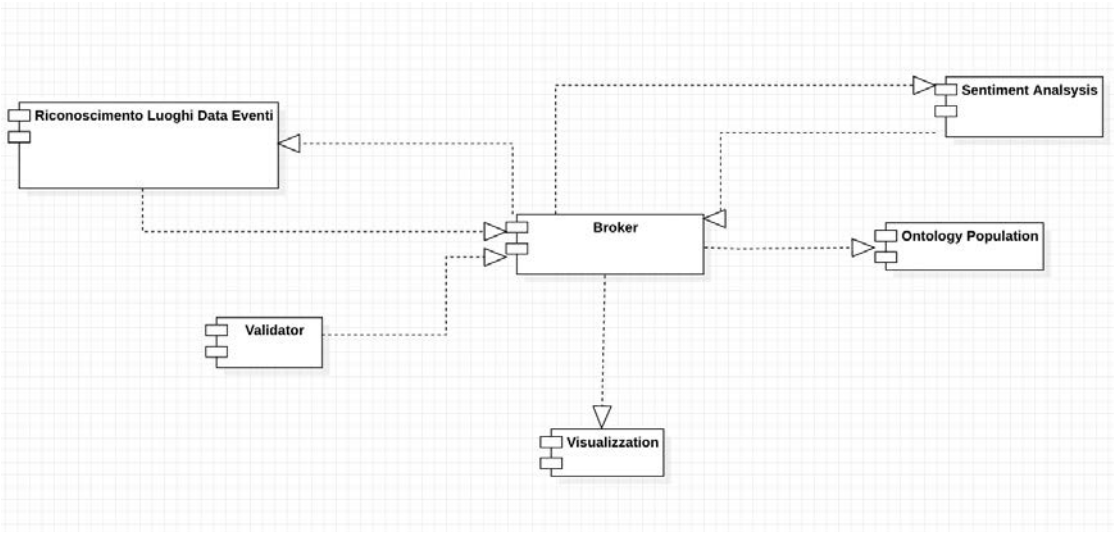

Figure 2. Component Diagram

I componenti dell'intero sistema sono sei:

- **Validator**: Questo componente si occupa di prelevare i dati dalle News Api e di analizzare la pertinenza rispetto al dominio applicativo. In particolare soddisfando il Task 1 e il Task 2
- **Riconoscimento Luoghi Data Event**i: Questo componente si occupa di estrapolare dai testi le informazioni inerenti ad eventi, specificando luoghi e date di questi ultimi se presenti. In particolare soddisfando il Task 3
- **Sentiment Analysis**: Questo componente si occupa di analizzare il sentimento sui dati che soddisfano parzialmente o completamente il Task 3 e va a svolgere le attività menzionate nel Task 5
- **Ontology Population:** si occupa di prelevare gli eventi estrapolati dal component precedente per poter popolare l'ontologia. Svolge, dunque, le attività del Task 6
- **Visualizzation:** preleva tutti i risultati prodotti dai componenti "Sentiment Analysis" e, tramite diverse interfacce grafiche, permette una visualizzazione dei risultati, eseguendo le operazioni menzionate nel Task 7
- **Broker:** è colui che si occupa della gestione del flusso dei dati; più nello specifico, che aiuta tutti i componenti a scambiarsi i dati tra di loro.

Per quanto riguarda il Task 4 non è stato creato un componente poiché i risultati ottenuti sono stati insufficienti; di conseguenza quest'ultimo non avrebbe giovato all'interno del progetto finale

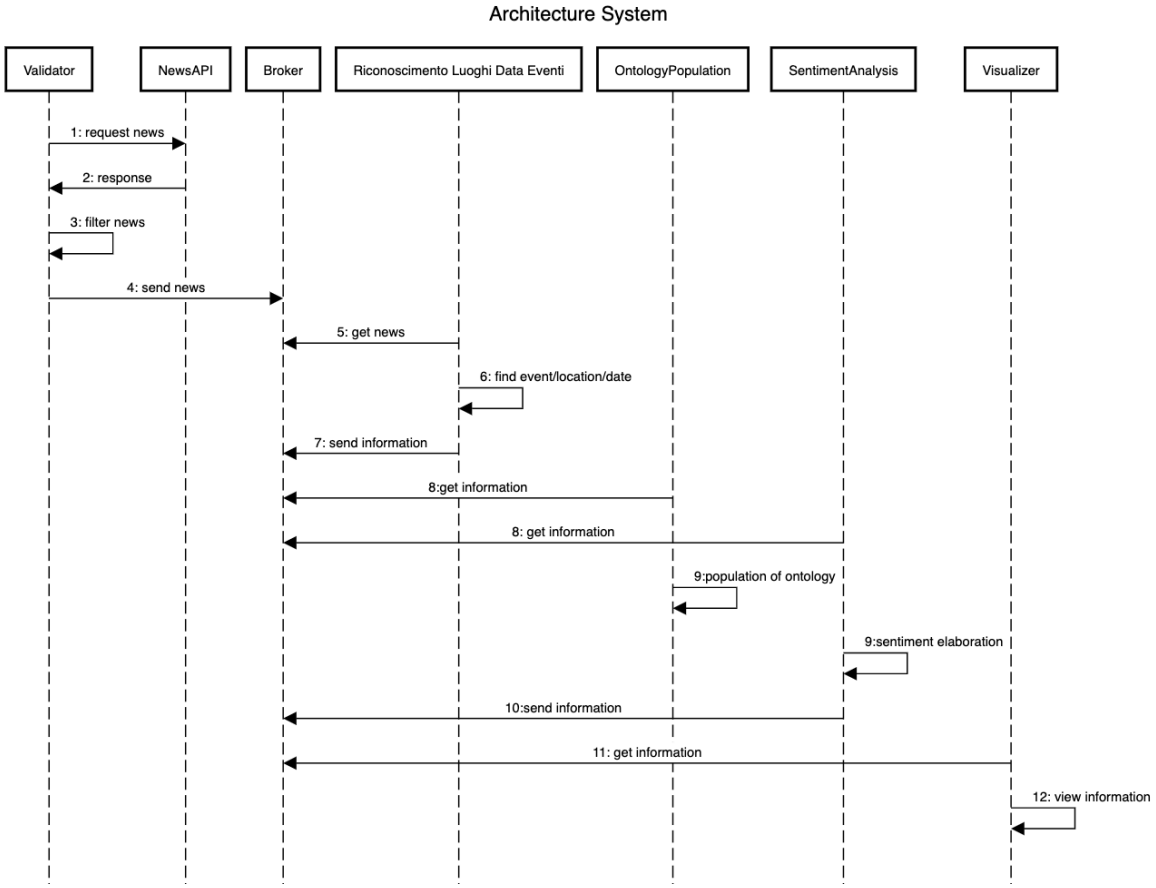

Figure 3. Sequence Diagram

La logica di funzionamento della pipeline, come illustrato dal sequence diagram Figura 3, si divide in 4 step:

1. Il primo a partire è il validator che preleva gli articoli dalle news, applica i filtri di pertinenza ed invia i dati al broker.

2. Il Riconoscimento Luoghi-Data-Eventi preleva gli articoli dal broker, estrae gli eventi ed eventuali luoghi e date associate dal testo e invia i dati al broker.

3. A questo punto sia il component "Ontology Population" che ''Sentiment Analysis"prelevano i dati dal broker, eseguono le loro operazioni e inviano i risultati al broker

4. Infine il visualization preleva dal broker i dati e li mostra a video.

4.2 Requisiti non funzionali

L'architettura di questo progetto è basata su una struttura a *microservizi*, un modello di progettazione realizzato tramite componenti indipendenti che eseguono ciascun processo finalizzato. Quello che distingue l'architettura basata su microservizi dagli approcci monolitici tradizionali è la suddivisione dell'applicazione nelle sue funzioni di base. Ciascuna funzione, denominata servizio, può essere compilata e implementata in modo indipendente. Pertanto, i singoli servizi possono funzionare, o meno, senza compromettere gli altri. Nelle architetture monolitiche tutti i processi sono strettamente collegati tra loro e vengono eseguiti come un singolo servizio. Aggiungere o migliorare una funzionalità dell'applicazione monolitica diventa più complesso, in quanto sarà necessario aumentare la base di codice. Tale complessità limita la sperimentazione e rende più difficile implementare nuove idee. Le architetture monolitiche rappresentano un ulteriore rischio per la disponibilità dell'applicazione, poiché la presenza di numerosi processi dipendenti e strettamente collegati aumenta l'impatto di un errore in un singolo processo. Un'architettura basata su microservizi è realizzata da componenti indipendenti che eseguono ciascun processo applicativo come un servizio. Tali servizi comunicano attraverso un'interfaccia ben definita che utilizza API leggere. Ogni servizio esegue una sola funzione poiché eseguito in modo indipendente; ciascun servizio può essere aggiornato, distribuito e ridimensionato per rispondere alla richiesta di funzioni specifiche di un'applicazione.

I requisiti non funzionali che l'architettura deve soddisfare sono:

- **Agilità**: I team di sviluppo agiscono in contesti ridotti e ben delineati così che possano lavorare in modo più indipendente e rapido. Ciò riduce i tempi del ciclo di sviluppo.
- **Scalabilità e Flessibilità:** I microservizi consentono di scalare ciascun servizio in modo indipendente per rispondere alle funzionalità richieste per l'applicazione
- **Semplicità e Distribuzione:** I microservizi supportano l'integrazione continua e la distribuzione continua, così da poter provare nuove idee in modo più semplice e ripristinare impostazioni precedenti quando qualcosa non funziona.
- **Libertà tecnologica:** Le architetture basate su microservizi non applicano un unico approccio all'intera applicazione. I team di sviluppo hanno la libertà di scegliere gli strumenti migliori per risolvere i loro problemi specifici. Di conseguenza, i team che costruiscono i microservizi possono scegliere lo strumento migliore per ciascun lavoro.
- **Codice riutilizzabile:** dividere il software in moduli piccoli e ben definiti permette la completa riutilizzabilità del codice.
- **Resilienza:** L'indipendenza dei servizi aumenta la resilienza di un'applicazione in caso di errori. In un'architettura monolitica, un errore in un unico componente potrebbe avere ripercussioni sull'intera applicazione. Con i microservizi, le applicazioni possono gestire completamente gli errori di un servizio isolando la funzionalità senza bloccare l'intera applicazione.

4.3 Architettura Fisica

Per realizzare questo tipo di struttura che soddisfi i requisiti non funzionali descritti in precedenza, ci siamo serviti di Docker, una piattaforma software che permette di creare, testare e distribuire applicazioni con la massima rapidità. Docker raccoglie il software in unità standardizzate chiamate container che offrono tutto il necessario per la loro corretta esecuzione, incluse librerie, strumenti di sistema, codice runtime.
Ogni microservizio è situato all'interno di un ***container***, il concetto di container è molto simile a quello di macchina virtuale ma i container sono più "leggeri" rispetto alle macchine virtuali, richiedono poche risorse (aggiuntive) di CPU e possono essere attivati in pochi istanti. Questo li rende particolarmente adatti a situazioni in cui il carico di lavorazione da sostenere è altamente variabile nel tempo e ha picchi imprevedibili.

Il flusso di dati di questa pipeline sarà gestito da **Apache Kafka** che è un sistema distribuito composto da server e client che comunicano tramite un protocollo di rete TCP ad alte prestazioni.
Kafka ha un'architettura basata sulla logica produttore, consumatore.
Ecco alcuni concetti con cui è opportuno familiarizzare riguardo Kafka:

- **Broker**: il broker Kafka riceve messaggi dai produttori e li archivia in base a un offset unico. Il broker consentirà inoltre ai consumatori di recuperare i messaggi per topic, partizione e offset
- **Messaggio**: È un'unità di dati in Kafka. Si può pensare a ogni messaggio come a un record di un database
- **Topics e Partizioni**: Ogni topic è un flusso di messaggi con nome. Un topic è composto da una o più partizioni. Le partizioni permettono a Kafka di scalare orizzontalmente distribuendo i dati tra i broker
- **Producer**: entità che invia i messaggi al broker kafka
- **Consumer:** entità che riceve i messaggi dal broker kafka

La pipeline sarà ricostruita fisicamente con i container, utilizzando la logica descritta in Figura 3: ogni Task comunicherà con gli stadi adiacenti mediante il Kafka Broker così che ogni servizio avrà una coda specifica in cui potrà inserire i propri risultati.

5 Metodologie definite, tecnologie utilizzate e realizzazione del Sistema

In questa sezione sono riportati tutti gli aspetti metodologici e realizzativi dei task precedentemente descritti nella sezione Workflow e attività, scendendo nel dettaglio relativamente ai singoli obiettivi e confrontando le diverse tecniche applicabili nei differenti contesti.

5.1 Task 1: Data Ingestion

L'obiettivo principale del Task 1 consiste nel definire delle tecniche per il recupero di dati, sotto forma di notizie, provenienti da diverse fonti, purché tutte rigorosamente in italiano, e di eseguire un primo filtraggio su di esse, in modo da poter ridurre la quantità di dati in ingresso alla Pipeline, e mantenendo il focus sul contesto di interesse delle Comunità Energetiche.
Per far ciò, è possibile operare in diversi modi:

- Effettuare un crawling di diversi siti di notizie, individuando quelle inerenti il dominio di interesse
- Richiedere dei Digest periodici delle news, limitatamente ad un certo argomento o insieme di argomenti, a siti in grado di operare da collettori (**Google News** [1] opera in tal senso).
- Utilizzare API pubbliche per la raccolta delle news inerenti un argomento specifico.

Il primo sistema è stato considerato troppo complesso da realizzare, soprattutto in relazione alle effettive necessità di ricerca.
Il secondo approccio, benché molto più semplice da realizzare, richiedeva una certa dipendenza rispetto alle tempistiche dei Digest fornite da Google, e in più imponeva il dover comunque analizzare delle e-mail inviate dal servizio, contenenti i link alle notizie vere e proprie.
Il terzo approccio invece ha dimostrato di avere, all'atto pratico, la flessibilità necessaria a realizzare il componente di Data Ingestion.
Applicando questo terzo approccio, è stato sviluppato un modulo che, usando un insieme di parole chiave predefinite, sfruttate per ridurre in prima battuta il numero di notizie esaminate, scarica le news di interesse, effettua le dovute operazioni di pulizia del testo scaricato, e poi invia le informazioni estratte al componente di Validazione, sviluppato come parte del Task 2.

Vediamo nel dettaglio come questo approccio è stato effettivamente implementato nel nostro sistema.

Il componente che scarica le news, realizzato in Python, utilizza le API messe a disposizione da **News API** [2], che permette di recuperare notizie da fonti differenti e varie lingue, impostando un insieme di parole chiave. Nella versione "Free", le API possono essere richiamate 100 volte ogni ventiquattro ore, mentre nelle versioni a pagamento è possibile chiamare l'API un numero maggiore di volte.

Le parole chiave utilizzate per recuperare le news sono attualmente le seguenti 18:

"autoconsumo elettrico","comunità energetica","comunità energetiche","cruise ENEA","dhmous ENEA","direttiva redII","economia circolare","energia collettiva","energy communities","energy community","energy sharing","living lab","recon ENEA","progetto cruise","progetto domus","progetto recon","smart community","energy prosumer".

Le API sono invocate ogni volta su una parola chiave differente, per cui ogni ciclo di download delle notizie richiede solo 18 invocazioni alle API. Nel complesso, il ciclo di download viene eseguito tre volte in un giorno, per un totale di 54 chiamate, ben al di sotto del limite giornaliero.

Una tipica chiamata alla API è stata costruita nel modo seguente:

"https://newsapi.org/v2/everything?q="+key+"&country=it&apiKey=ValoreAPIKey"

In questa chiamata di esempio:

- "everything" indica che si vuole scaricare qualsiasi articolo (altre opzioni quali "topheadlines" sono descritte al sito di newsapi.org)
- "country" indica il paese di provenienza, nel nostro caso "it" per Italia
- "key" è la parola chiave scelta, una delle 18 selezionate in precedenza
- "apiKey" è la API associata all'account registrato presso newsapi.org, ed è unica per ogni utilizzatore

Ogni invocazione della API restituisce un Array JSON, in cui ogni elemento è costituito da tre item di interesse:

- "url", cioè l'indirizzo a cui è stata trovata la notizia
- "title", il titolo della notizia
- "Description", una breve descrizione del testo della notizia, composto da meno di 2000 caratteri.

Title e Description non sono sufficienti a comprendere l'esatto contenuto della notizia, che in genere è molto più lunga e complessa.

Tuttavia grazie all'url, possiamo scaricare l'intero HTML della pagina da cui la notizia è stata scaricata, come se fosse del banale testo. Ciò viene eseguito usando la libreria Python "urllib", che semplicemente scarica l'HTML della url fornita, e poi sfruttando il modulo "**BeautifulSoup**" [3] della libreria Python "bs4", che consente di ripulire il testo da elementi indesiderati, nel nostro caso tag HTML e codice Javascript annesso al testo.

Una volta scaricato e ripulito il testo delle varie pagine contenenti le notizie, queste possono essere passate al secondo modulo che le analizza nel dettaglio e ne rileva la pertinenza.

5.2 Task 2: Validazione di Pertinenza

Per validazione di Pertinenza si intende una classificazione di tipo binario, in cui il testo analizzato viene inserito in una classe "**Pertinente**" o "**Non Pertinente**". Solo i testi identificati come Pertinenti sono trasferiti ai successivi stadi della Pipeline, attraverso quella che, nella sezione Architettura, viene identificata come un coda gestita da Kafka; gli altri semplicemente sono scartati.

Come per il Task 1, anche in questo caso sono stati esaminati diversi possibili approcci.

Il primo consiste nell'esaminare, utilizzando tecniche di Natural Language processing presenti in librerie predefinite, il contenuto dei testi e di determinarne dunque l'effettiva pertinenza al contesto.
Il secondo approccio esaminato è più specifico del Machine Learning, e prevede che si sfruttino dei classificatori di tipo binario, quali Decision Tree, Support Vector Machine o reti Neurali. Data la natura testuale della fonte di dati analizzata, il meccanismo più appropriato è senza dubbio quello di associare Reti Neurali a tecniche di Word Embedding, per le quali è necessario operare delle operazioni preliminari di trasformazione sul testo.
I due approcci qui identificati presentano vantaggi e svantaggi, che li rendono più o meno preferibili a seconda delle situazioni.

- L'approccio basato su librerie di NLP non necessita di un training, ma può essere immediatamente applicato al problema, purché vengano fornite delle caratteristiche del testo da esaminare che possano portare alla corretta classificazione. Ad esempio, conoscere delle parole chiave da riscontrare nel testo, che possono essere eventualmente arricchite ed estese man mano che i testi stessi sono esaminati, può rappresentare certamente una soluzione semplice ed efficace, ma che va valutata.
- L'approccio che si basa su Machine Learning richiede che il classificatore, nello specifico una rete neurale, venga addestrato utilizzando un corpus di testi predefinito. Inoltre, esso richiede generalmente una quantità di risorse di calcolo superiori alla semplice applicazione delle funzioni di una libreria, per quanto complessa quest'ultima sia. Il vantaggio principale consiste nel non legarsi in modo esplicito ad un insieme di parole ma, attraverso il Word Embedding, nel creare dei cluster significativi di termini utili alla classificazione.

Utilizzare la libreria Spacy fornisce certamente due vantaggi. Il primo è che non occorre addestrare alcun algoritmo, ma è sufficiente usare un insieme di parole chiave per ottenere i risultati desiderati, anche con una buona accuratezza. Il secondo, è che non si necessita di particolari risorse di calcolo per poter effettuare quella che, a tutti gli effetti, è una classificazione binaria.
Lo svantaggio principale è che si parte da un insieme di parole chiave predefinito, e non scoperto dal testo esaminato in maniera intelligente. Sebbene ciò venga parzialmente superato tramite la fase di espansione semantica, che fornisce un feedback allo strato di validazione di pertinenza, sono stati cercati altri sistemi che potessero in qualche modo slegarsi completamente dall'uso di parole chiave.
Il meccanismo previsto è stato quello di creare una rete semantica che, utilizzando il Word Embedding, potesse analizzare un corpus documentale, i cui elementi appartenessero ad un insieme indicato come **Pertinente**, oppure ad un altro indicato come **Non Pertinente.**
La differenziazione è necessaria per l'addestramento corretto di una rete, e nel nostro caso è stato possibile sfruttare sia il corpus documentale fornitoci in partenza, i cui circa 2500 testi erano considerati già pertinenti, a cui è stato affiancato un insieme di testi scaricati usando le News Api descritte inizialmente, ma utilizzando parole chiave completamente sconnesse al tema delle Comunità Energetiche. In questo modo sono stati ricavati circa 500 testi differenti, che sono stati etichettati in maniera negativa rispetto alla pertinenza.

5.2.1 Uso di librerie per il Natural Language Processing.

Per eseguire la classificazione tramite NLP, il modulo realizzato utilizza la libreria "**Spacy**"[3] e le sue funzionalità di Natural Language Processing per analizzare il testo nel dettaglio, ripulendolo da segni di punteggiatura, stopword ed effettuando una lemmatizzazione dei vocaboli. Spacy può utilizzare, nello specifico, le stopword della lingua italiana ottenute tramite un'altra libreria, "**NLTK**" [4], che fornisce un set predefinito di tali parole.
Una volta impostato per effettuare la lemmatizzazione con vocabolario italiano e definite le stopword adatte al testo da trattare, tramite Spacy è possibile determinare la presenza di opportune parole chiave all'interno dello stesso testo. In particolare il modulo "**PhraseMatcher**" esegue questa ricerca

nel testo, indicando uno o più insiemi di parole su cui Spacy esegue le stesse operazioni di lemmatizzazione operate precedentemente sul testo, per poi verificarne la presenza. È possibile definire più insiemi di parole, indicati come "Pattern" dalla libreria Spacy, e il Matcher individua queste parole nel testo indicando sia a quale insieme appartengono, sia la posizione iniziale e finale della parola trovata nel testo.
Il meccanismo applicato è molto semplice ma alquanto efficace, ed è stato testato per verificarne l'effettiva accuratezza. I risultati sperimentali ottenuti sono riportati nella sezione **Risultati Sperimentali.**

5.2.2 Uso di rete Semantica con Word Embedding

La rete neurale definita per eseguire la classificazione è stata costruita usando un modello sequenziale, al quale sono stati aggiunti uno strato di **Embedding**, uno di **Flattening**, e uno finale **Dense** (fully connected), con funzione di attivazione **Sigmoidale.** L'ottimizzatore sfruttato per la rete è il classico **Adam**, noto per avere una buona rapidità di convergenza e una forte stabilità.

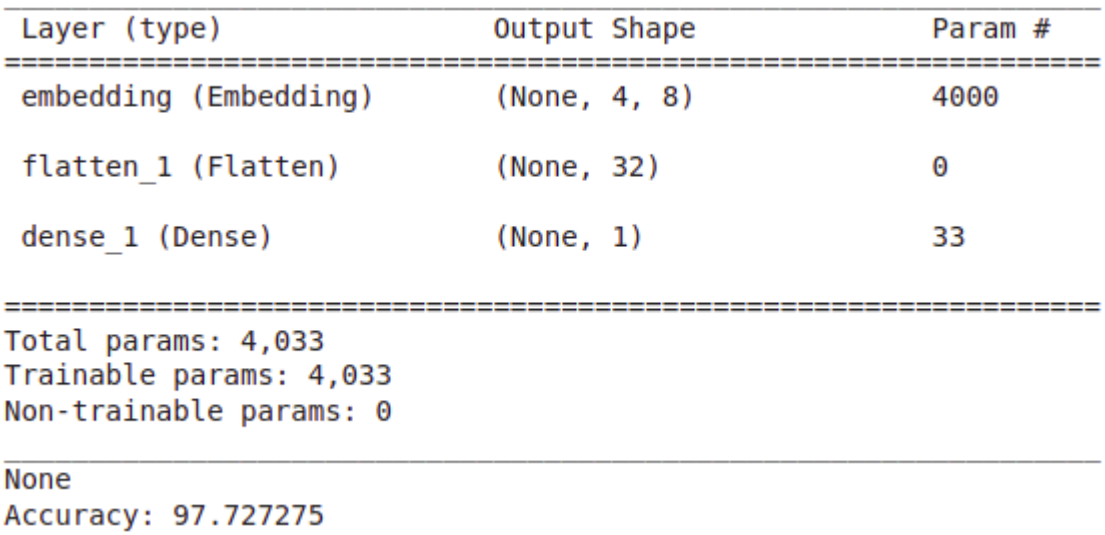

Figure 4. Schema della rete neurale con Word Embedding

La rete, mostrata in Figura 4, presenta circa 4000 parametri liberi, ed è stata addestrata per un numero di epoche pari a 50, raggiungendo un'accuratezza attesa del 97%. Si tratta tuttavia di risultati preliminari, certamente incoraggianti, chei devono essere ancora ben validati su ulteriori testi, prima di poter utilizzare la rete al posto del semplice meccanismo offerto da Spacy. La rete neurale inoltre ci permetterebbe sì di slegarci da un insieme di parole predefinito, ma avrebbe anche uno svantaggio: la potenza computazionale richiesta sarebbe sicuramente superiore, rendendo eventuali container piuttosto pensati dal punto di vista delle risorse necessarie. È tuttavia possibile fare alcune interessanti considerazioni, andando a proiettare su uno spazio tridimensionale i cluster di parole ottenuti utilizzando la tecnica del Word Embedding.
Innanzitutto risulta evidente, dalla Figura 5, in cui il clustering è stato effettuato applicando un algoritmo di Principal Component Analysis (PCA), che si formino degli aggregati di termini molto prossimi l'uno all'altro, e che dunque fanno presupporre la prevalenza di un insieme di componenti rispetto ad altri.

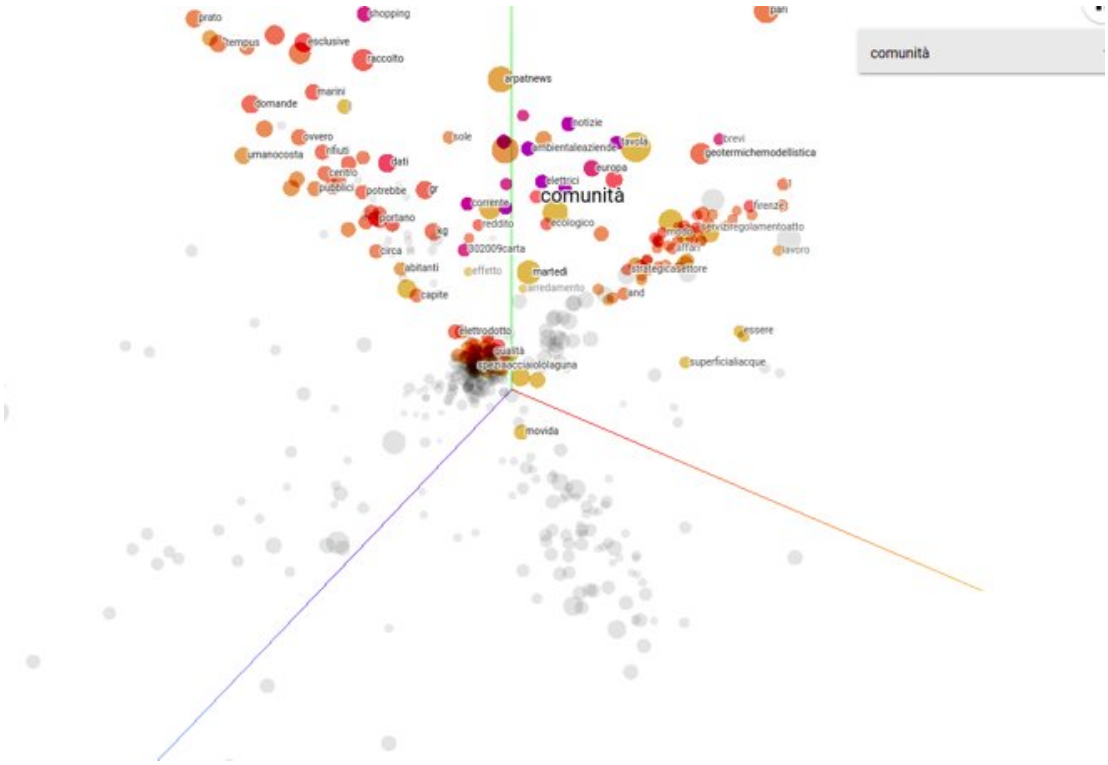

Figure 5. Cluster di Embedding con PCA

Se tuttavia i cluster in Figura 6 sono piuttosto sparpagliati e poco evidenti,
osservando la stessa rappresentazione ottenuta tramite la tecnica TSNE, mostrata in Figura 5, appare evidente come esistano cluster molto meglio definiti e piuttosto distanti di termini, che caratterizzano il corpus documentale esaminato.

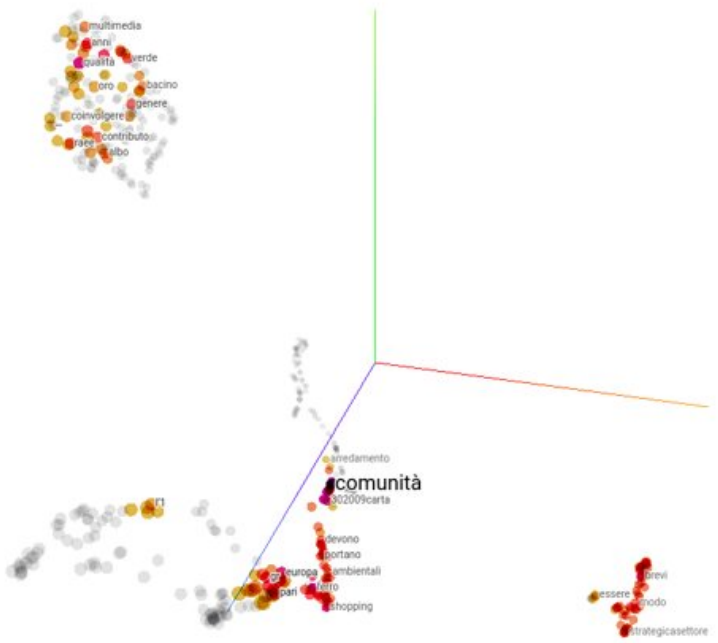

Figure 6. Cluster di Embedding con TSNE

Considerato il numero relativamente esiguo di testi esaminati, soprattutto per quanto riguarda l'aspetto della Non Pertinenza, appare evidente che è possibile migliorare ulteriormente i risultati ottenuti dal Word Embedding, ottenendo così cluster più significativi e rappresentativi.

5.3 Task 3: Riconoscimento Luoghi/Date/Eventi

L'obiettivo di questa parte del lavoro è stato quello di ricercare, nella maniera più precisa possibile, per ogni testo preso in input, se al suo interno fossero presenti riferimenti ad eventi specifici (come ad esempio: workshop, conferenze o riunioni) e anche se a questi ultimi fossero associati dei luoghi e/o delle date. Il lavoro è stato suddiviso in 5 attività principali:

- Ricerca Eventi
- Ricerca Luoghi
- Ricerca Date
- Relazioni Eventi/Data/Luogo
- Analisi e Selezione dei Risultati

Le principali librerie utilizzate nella realizzazione di tale script sono state SPACY[4] e NLTK[5].
Dato per assunto che i testi ricevuti fossero già filtrati, e quindi inerenti al contesto, sono state effettuate le seguenti operazioni:

5.3.1 Ricerca Eventi

Il primo passo è stato quello di ricercare gli eventi desiderati (inseriti manualmente in un vettore dall'utente); Come primo passo è stata effettuata una espansione semantica degli eventi. Quindi, per ogni evento inserito, sono stati trovati tutti i sinonimi tramite il comando wordnet.synsets della libreria NLTK[5]; successivamente tutti i sinonimi individuati sono stati inseriti all'interno di un vettore usato

per la ricerca degli eventi nel testo mediante il comando “PhraseMatcher” (già menzionato nella Task 2) della libreria Spacy[4]. Ad ogni match venivano prelevati e salvati in un secondo vettore sia il testo che la posizione all’interno del documento

5.3.2 Ricerca Luoghi

Dopo gli eventi si è passati alla ricerca dei luoghi all’interno del testo. Luoghi intesi sia come località geografiche sia come piattaforme online. Per ciascuna delle due ricerche sono stati utilizzati differenti approcci:

5.3.2.1 Ricerca Località Geografiche

Per ciò che concerne la ricerca di località geografiche sono stati utilizzati due diversi approcci:

- Inizialmente si è provato l’uso di librerie gratuite, come ad esempio la libreria GeoText, il cui funzionamento si limitava, preso un testo in input, a cercare e salvare all’interno di un vettore eventuali luoghi. Tuttavia tale metodologia non ha restituito i risultati attesi.

- Successivamente è stata utilizzata la tecnologia NER fornita da Spacy [4]. In particolare, dato un testo, una volta tokenizzato, sono stati salvati all’interno di un vettore posizione e testo del token di tutte le entità che avessero come label la stringa ‘LOC’. Quest’ultima è stata quella che ha prodotto risultati migliori (mostrati nel capitolo “Valutazione Qualitativa dei Risultati”) e di conseguenza quella che è stata scelta

5.3.2.2 Ricerca Piattaforme Online

Per quanto riguarda la ricerca di piattaforme online come Teams, Zoom, Skype, etc… l’idea di fondo è stata quella di utilizzare ancora una volta il comando PhraseMatcher [4]. La logica di funzionamento è stata equivalente alla ricerca degli eventi, dunque è stato tokenizzato il testo e si sono ricercate le keyword inerenti a tale contesto. Nel caso specifico dei test effettuali che possono essere visualizzati nel Capitolo “Valutazione Qualitativa dei Risultati” il set di parole chiave è stato il seguente:

```
loc_online=["online","remoto","Teams","Zoom","Skype","videoconferenza"]
```

5.3.3 Ricerca Date

Per quel che concerne la ricerca delle date invece sono emerse alcune difficoltà. Il primo approccio è stato quello di utilizzare la stessa metodologia usata per la ricerca delle località geografiche (NER), tuttavia il riconoscimento da parte di Spacy dell’entità DATA era possibile soltanto per la lingua inglese e non per quella italiana. Si è dunque passati all’utilizzo di altre librerie ad esempio datefinder o dateparser ma, in questo caso, i risultati sono stati di fatto poco precisi. Un'altra strada intrapresa è stata il tradurre il testo in lingua inglese e, successivamente, applicare la NER con un dizionario di inglese. Nell’utilizzo di tale metodo ha creato delle difficoltà nella traduzione poiché tutte le librerie reperite (googletrans, translate, deep_translator) consentivano la traduzione di un numero limitato di testi. Dopo una prima analisi del problema e i tentativi sopra elencati, le soluzioni escogitate sono state le seguenti:

- **Traduzione Testo:** Utilizzando il comando split() il testo analizzato è stato suddiviso in N elementi posizionati all'interno di un vettore, dove N è il numero delle parole presenti all'interno di ogni testo. Per ogni elemento del vettore, tramite il comando wordnet.synsets si è verificata l'esistenza di un corrispettivo sinonimo in inglese e, in caso positivo, il primo sinonimo è stato sostituito con l'elemento corrente; in caso contrario si è passato all'elemento

  successivo. Successivamente, tramite il comando join(), si è ricomposto lo scritto ottenendo un testo parzialmente tradotto (utile solo al fine di ricercare date) su cui applicare la NER di Spacy in lingua inglese.
- **Espressioni Regolari:** Infine, un ultimo approccio è stato l'utilizzo di espressioni regolari (RegEx) [12] per la ricerca non di entità ma di date in un formato standard. Con la regola definita veniva coperto un buon gruppo di date scritte in diversi formati. Questo approccio è stato già utilizzato in un altro progetto in cui andavamo ad individuare proprio queste entità caratterizzate da pattern ben definiti con espressioni regolari(ad esempio l'individuazione dei numeri di patente e carte di identità in sentenze giuridiche per poterle anonimizzare). Questo ci ha permesso di individuare con maggiore precisione le date e annotare i token corrispondenti come date. Il passo successivo sarà quello di provare ad estendere il modello di Spacy per la lingua italiana anche alle date utilizzando il corpus annotato come training set.

Tra tutti gli approcci testati l'ultimo è quello che ha prodotto i risultati migliori (Risultati visibili nel capitolo "Valutazione Qualitativa dei Risultati"). Di conseguenza questo è stato utilizzato per i test sui vari match.

5.3.4 Ricerca Relazioni Evento Luogo Data

Il passo successivo della ricerca è stata l'individuazione di un'ipotetica relazione doppia o tripla tra i due/tre elementi (tra un evento e un luogo, un evento e una data, un evento una data e un luogo). Per l'identificazione di queste relazioni ci si è avvalsi dell'utilizzo di due metodologie: un approccio legato alla distanza dei token individuati ed un altro che ha previsto l'utilizzo di pattern. Approfondiamo la natura di tali relazioni.

- **Distanza Token:** Con il primo approccio la ricerca di una relazione evento-data risulta la medesima di quella relativa al binomio evento-luogo. Per ogni evento all'interno del testo viene analizzata la posizione del token e, a partire da quest'ultima, si cerca in un intorno fissato se vi è una data o un luogo utilizzando, quindi, tutte le posizioni salvate nel modo precedentemente spiegato. Per la ricerca della tripla evento-data-luogo si preleva la posizione di ogni evento, si ricerca in un intorno fissato di questo se è presente un luogo. Per ogni luogo associato all'evento si ricerca in un intorno fissato se è presente una data, in caso positivo viene restituita la tripla, in caso negativo viene effettuata una seconda ricerca nello stesso intorno fissato ma a partire dall'evento. Dopo una serie di test l'intorno che ha prodotto i migliori risultati risulta il seguente [val-6;val+15] con val=posizione_token_di_riferimento

- **Pattern:** Con il secondo approccio, che prevede l'utilizzo di pattern, viene utilizzata la libreria Spacy [4]. Per la ricerca del luogo il pattern è strutturato nel modo seguente: per ogni verbo all'interno di un periodo si verifica se tale verbo è legato da una relazione di dipendenza all'evento e se lo stesso verbo è legato anche ad un luogo tramite il comando DependencyMatcher. Per l'identificazione dell'evento all'interno del pattern viene utilizzato l'attributo ORTH passando tutte le keywords degli eventi; per quanto riguarda i luoghi invece si utilizza l'attributo ENT_TYPE passando 'LOC'. Per

le date la metodologia utilizzata è la stessa ma, in questo caso, ci ritroviamo il problema dell'assenza dell'entità 'DATE' nella lingua italiana. La soluzione consiste nell'utilizzare, anche per le date, l'attributo ORTH passando per ogni evento il giorno, il mese e l'anno in successione; nel caso in cui si evidenzia un match sarà salvato in un vettore l'evento e la data completa. Infine, il pattern che ricerca evento-data-luogo verificherà se un dato verbo è in relazione con la tripla identificata tramite gli attributi ORTH e ENT_TYPE mediante lo stesso procedimento applicato ai binomi

Per ogni metodologia descritta sono poi stati effettuati dei test per misurare i parametri di Precisione, Recall e Accuratezza. Tutti risultati sono riportati nel capitolo "Valutazione Qualitativa dei Risultati".

5.3.4. Analisi e Selezione dei Risultati

Arrivati fino a questo punto ciò che abbiamo ottenuto fino ad ora è un tool in grado di analizzare un testo e restituire i seguenti dati:

- Luoghi Fisici individuati (1 tipologia di ricerca)
- Luoghi Online individuati (1 tipologia di ricerca)
- Date individuate (3 tipologie di ricerca)
- Eventi individuati (1 tipologia di ricerca)
- Relazioni Evento/Data individuate (2 tipologie di ricerca)
- Relazioni Evento/Luogo individuate (2 tipologie di ricerca)
- Relazioni Evento/Luogo/Data individuate (2 tipologie di ricerca)
- Relazioni Evento/Luogo Online individuate (2 tipologie di ricerca)
- Relazioni Evento/Luogo Online/Data individuate (2 tipologie di ricerca)

Osservando tali risultati possiamo fare le seguenti considerazioni:

1. Nella ricerca delle date la seconda tipologia di espressione regolare risulta essere quella che restituisce i migliori risultati.
2. Il metodo della distanza dei token restituisce risultati migliori rispetto che l'utilizzo dei pattern.
3. In quasi tutti i risultati la Recall restituisce dei valori accettabili. Non possiamo dire lo stesso per la precisione.

I primi 2 punti evidenziano la necessità di utilizzare le rispettive funzioni citate per ottenere risultati migliori.
Il 3o punto, invece, è principalmente causato dal fattore che tutte le funzioni vengono lanciate nel main senza uno schema al fine di analizzare i risultati sperimentali ottenuti nella maniera più precisa possibile. Tuttavia, facendo sempre riferimento al nostro obiettivo, a noi non interessa avere tanti risultati con un' elevata precisione che debbano poi essere analizzati, ma uno solo che mi indichi i possibili eventi collegati ad possibili date, a possibili luoghi o ad entrambi. Quindi l'ultima implementazione è stata quella di identificare una logica di fondo necessaria per restituire, non tutti gli eventi, luoghi, date e match trovati, ma quelli più probabili. Dopo aver capito (osservando i risultati dei test) quali funzioni utilizzare è stato dunque necessario cercare di capire come richiamarle per ottenere i risultati migliori. La logica secondo cui si è pensato di richiamare le funzioni è la seguente:
Come primo punto si ricerca all'interno del testo se è presente un evento desiderato (Nel nostro caso: workshop, conferenza, convegno, incontro).
Successivamente,in caso di esito positivo, si ricerca un match sia per la data che per un "luogo online".
Da qui ci possono essere 4 possibili scenari:

1. **Match Evento/Luogo Online TROVATO,Match Evento/Data TROVATO:**
   ricerco se esiste una tripla Evento/Luogo Online/Data. In caso positivo restituisco la tripla, in caso negativo restituisco le 2 doppie
2. **Match Evento/Luogo Online NON TROVATO, Match Evento/Data TROVATO:**
   Ricerco un match Evento/Luogo e a questo punto possono verificarsi 2 casistiche:
   a. **Match Evento/Data TROVATO, Match Evento/Luogo TROVATO:**

Ricerco una possibile tripla Evento/Data/Luogo. In caso positivo restituisco il risultato, in caso negativo restituisco le 2 doppie

b. **Match Evento/Data TROVATO, Match Evento/Luogo NON TROVATO:** restituisco il match Evento/Data

3. **Match Evento/Luogo Online TROVATO,Match Evento/Data NON TROVATO:** restituisco il match Evento/Luogo Online
4. **Match Evento/Luogo Online NON TROVATO,Match Evento/Data NON TROVATO:** ricerco se esiste un match Evento/Luogo. In caso di esito positivo restituisco la doppia, in caso contrario non restituisco nulla

Il risultato (identificato come final_match) è stato testato circa 100 testi e i risultati sono i seguenti: **Final Match:** Precisione=71%, Recall=83%, Accuratezza=82.

5.4 Task 4: Espansione Semantica

L'obiettivo consiste nell'analizzare i termini presenti nell'ontologia e nei testi del corpus, inerenti al tema delle comunità energetiche, così da poter fornire una validazione dell'ontologia ARAKNE.

Gli approcci utilizzati per questo task sono stati due:

- Confronto Sinonimi/Iperonimi/Iponimi;
- Media IDF/T.

5.4.1 Confronto Sinonimi/Iperonimi/Iponimi

Innanzitutto sono stati prelevati i termini dell'ontologia mediante l'uso della libreria **Owlready2** [6], di ogni termine si è generato una lista di sinonimi con il comando **Synset,** una lista di iponimi di ogni termine e una lista di iperonimi. Successivamente sono stati inseriti all'interno dell'algoritmo di ricerca di eventi in modo da poter valutare l'inerenza di questi termini all'interno del corpus e nel caso avessero prodotto risultati positivi, sfruttarli anche per ampliare lo spazio di ricerca eventi.
i risultati di questa metodologia non hanno portato vantaggi in quanto sia i sinonimi che gli iponimi e gli iperonimi si discostano troppo dal significato reale dei termini iniziali, producendo gli stessi risultati prodotti dai singoli eventi selezionati in precedenza.

5.4.2 Media TF-IDF

La seconda metodologia è stata quella di utilizzare le tecniche di TF-IDF per poter capire l'inerenza dei termini dell'ontologia rispetto al corpus.
In particolare quest'analisi si è concentrata sulla funzione IDF che indica l'importanza generale del termine *i* nella collezione:

$$idf = D/N$$

dove D è il numero di documenti che contengono il termine *i*, N invece è il numero di documenti.
Prima di procedere con il calcolo dell'IDF si va creare un dizionario di tutte le parole che costituiscono gli articoli, ad ogni parola viene associato il proprio numero di occorrenza all'interno dell'articolo.

Successivamente viene calcolato l'IDF di ogni termine con la formula scritta in precedenza, se ne fa una media e viene confrontato con l'idf dei termini dell'ontologia. Per ogni termine se l'idf è maggiore della media di tutti i termini l'ontologia viene considerata valida.

5.5 Task 5: Sentiment Analysis

L'obiettivo consiste nell'individuazione ed estrazione di opinioni, associando un sentiment(positivo ,negativo o neutro) ai testi presi in input dagli stadi precedenti della pipeline.

5.5.1 Metodologia

Le metodologie applicate alla risoluzione di questo task sono state:

1. L'approccio iniziale è stato quello di utilizzare la libreria **TextBlob** che ci permetteva di calcolare la polarità degli articoli presi in analisi in maniera molto semplice e veloce, questo metodo però produceva risultati poco accurati in quanto la libreria aveva una scarsa compatibilità sulla lingua italiana.

2. Il secondo approccio è stato quello di utilizzare una rete neurale, basata sul modello BERT (Bidirectional Encoder Representations from Transformers), messa a disposizione dalla piattaforma "huggingface.co". Inoltre per una maggiore precisione nelle misura la sentiment analysis non è stata applicata su tutto il testo, ma in un intorno delle keyword ricercate tramite il PhraMatcher. Per questo valido motivo la S.A. è stata posizionata subito dopo il Task 4 così da poter prelevare le posizioni degli eventi senza doverli ricalcolare. Per effettuare questa analisi in primo luogo è servito dichiarare un modello e un tokenizer che facesse riferimento alla rete citata in precedenza. Successivamente viene richiamata la funzione "Sentiment_score" che tokenizza il testo in analisi generando un tensore di tipo 'torch',una matrice multidimensionale contenente elementi di un singolo tipo di dati. Tale risultato viene dato in pasto alla rete neurale, che una volta elaborato restituisce un valore successivamente normalizzato in un range da 1 a 5 ; i valori 1 e 2 sono stati associati a un sentimento negativo, il valore 3 va a identificare un sentimento neutro e i valori 4 e 5 un sentimento positivo.

5.5.2 Tecnologie

Le tecnologie utilizzate all'interno di questo task sono:

1. **TextBlob:** una libreria python per l'elaborazione di dati testuali.Fornisce una semplice API per effettuare comuni attività di NLP come codifica di parti del discorso,sentiment analysis, classificazione e molto altro.

2. **BERT model:** uno dei modelli che ha riscontrato più successo nell'ambito del NLP negli ultimi anni e che è capace di incorporare il significato e le relazioni tra le parole stesse tramite una rappresentazione vettoriale,rilasciato da Google nel 2018.
   Le novità indotte da BERT sono varie. Innanzi tutto BERT utilizza un modello bidirezionale. É quindi in grado di rappresentare una parola in base al contesto della frase, sia che le informazioni rilevanti si trovino a destra che a sinistra, a differenza delle architetture 'left-to-right', le quali guardavano solo alle parole precedenti a quella che si sta processando. Una delle novità principali sta nel fatto che riesce a fare ciò senza l'utilizzo di layer ricorrenti. Bensì utilizza un meccanismo chiamato Self-Attention.Questo permette al modello, nel momento in cui sta processando un determinato token, di andare a guardare agli altri elementi della frase

di input, alla ricerca di quelli più rilevanti, i quali possono aiutare a creare un encoding migliore del token in questione.

5.6 Task 6: Ontology Population

Per "Ontology Population" si intende l'attività di popolamento di un'ontologia con opportune istanze e proprietà.
Come riportato nella sezione Obiettivi, l'obiettivo che si vuole perseguire è quello di popolare un'ontologia con delle istanze significative prelevate dalle precedenti attività di text extraction. Sono stati provati due approcci di ontology population:

1) **Ontology Population con Tecniche di Word Embedding**: l'ontologia ARAKNE viene popolata a partire da concetti individuati dal corpus documentale sfruttando opportune metriche di Text Similarity per riconoscere entità "vicine" alle classi dell'ontologia;
2) **Ontology Population con Entità Riconosciute dalla NER**: i risultati ottenuti dalla NER descritta nella sezione Riconoscimento Luoghi/Date/Eventi vengono utilizzati per popolare una mini-ontologia costruita ad-hoc;

Di seguito vengono presentati entrambi gli approcci, anche se la strada che è stata perseguita e portata poi avanti è la seconda.

5.6.1 Ontology Population con Tecniche di Word Embedding

Un primo approccio che abbiamo perseguito ai fini della ontology Population è stata l'applicazione di tecniche di Word Embedding (Di Martino et al., n.d.) al corpus documentale costituito dai 2161 testi inerenti alle comunità energetiche. L'idea di partenza era quella di provare ad identificare cluster di concetti presenti nei testi "affini" alle classi dell'ontologia di partenza ARAKNE, popolando tali classi con i concetti di questi cluster che risultassero più adeguati. Per realizzare un modello di Word Embedding è stato utilizzato il **modulo FastText** [11] integrato all'interno della libreria **Gensim**. La spiegazione di tale lavoro, per fornire maggiore chiarezza, viene suddivisa per step:

5.6.1.1 Step 1: Preprocessing di testi con tecniche di NLP

Siccome il modello di Word Embedding dovrà essere allenato su un corpus sufficientemente grande di testi puliti, è stato necessario attuare tutta una serie di procedure di pulizia mediante tecniche di Natural Language Processing. Esse vengono illustrate di seguito:

- **Estrazione di testi dall'HTML**: siccome alcuni testi di partenza contenevano residui di HTML dopo il processo di Web Scraping con il quale sono stati estratti, usando la libreria BeautifulSoup [3] sono stati puliti tali testi;
- **Lemmatizzazione**: tutti i token sono stati riportati al loro lemma, e per fare ciò si è ricorso alla libreria Spacy [4];
- **LowerCase Conversion**: tutti i caratteri contenuti nei token sono stati convertiti in minuscolo;
- **Rimozione di Stopword**: tutti i token il cui lemma fosse contenuto all'interno di liste di stopwords della lingua italiana sono stati rimossi;

- **Replacement di Caratteri Accentati**: tutti i caratteri accentati sono stati sostituiti con il corrispettivo carattere non accentato;
- **Eliminazione Caratteri di Punteggiatura**: sono state preparate apposite regex [12] per tale operazione;
- **Eliminazione di spazi multipli, tabulazioni e Andate a Capo**: sono state preparate apposite regex per tale operazione;

Nella Figura 7 viene mostrato il contenuto di un testo prima (in ROSSO) e dopo (in BLU) l'applicazione del modulo di preprocessing:

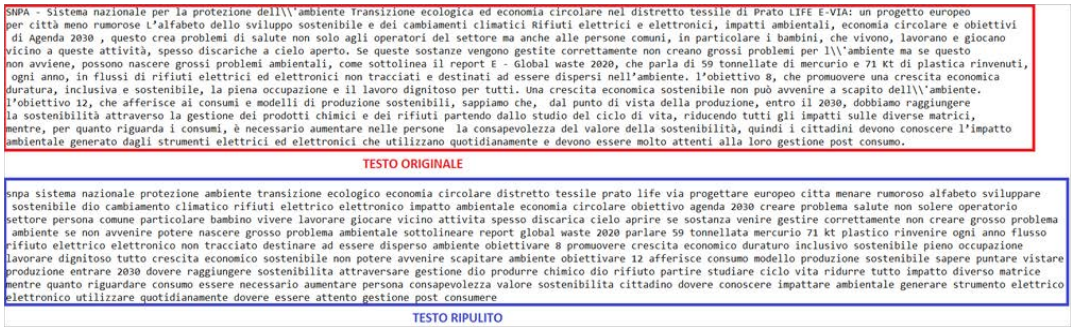

Figure 7. Testo prima e dopo la ripulitura.

5.6.1.2 Step 2: Creazione Modello di Word Embedding

Il testo preprocessato ottenuto nello step precedente è stato utilizzato come input per allenare un modello di Word Embedding. Di seguito vengono mostrati i parametri che sono stati settati per la realizzazione del modello:

- **embedding_size = 60**
- **window_size = 40**
- **min_word = 5**
- **down_sampling = 1e-2**
- **iterations = 100**

La fase di training ha richiesto diverse ore di elaborazione con sole 100 epoche su un computer dotato di una RAM di 12 GB; ultimato il training il modello ottenuto è stato salvato in maniera tale da averlo a disposizione per lo step successivo in qualunque momento. Il modello ottenuto contiene ben **491431 vettori di embedding**, ciascuno con un embedding_size pari a 60. A titolo di esempio, nella Figura 8 viene mostrato il vettore di embedding della parola "*sistema*":

```
[-0.4392273   0.01375031  0.25572848  0.06300752 -0.5894828   0.02212149
  0.24024265 -0.79320127  0.82834154 -0.20694213 -0.11108725  0.63546544
  0.02654576 -0.08182479 -0.10722093  0.07785691  0.12555468  0.09670032
 -0.18507946  0.25949907  0.06579361  0.03175073 -0.58212787  0.06422701
 -0.02515568  0.26130828 -0.15813445  0.18761185  0.26807255  0.01570615
 -0.29057875  0.21201925 -0.67683727  0.09383904 -0.06239376 -0.0129848
 -0.13851097  0.3291073   0.00335186 -0.04525982 -0.9435601  -0.26970044
  0.34703836  0.39187548  0.31972232 -0.11578809 -0.08726925 -0.1779774
 -0.04431435 -0.34699863 -0.14268182 -0.21605158 -0.1807494   0.4243169
 -0.22046745 -0.12681654  0.05592278  0.68148184 -0.0182415  -0.38050005]
```

Figure 8. Vettore di Embedding della parola "sistema".

5.6.1.3 Step 3: Utilizzo del Modello

Grazie alla vettorizzazione numerica prodotta dal modello di Word Embedding, diventa possibile rappresentare tutte le 491431 parole presenti nel modello all'interno di uno spazio vettoriale, e mediante apposite metriche è possibile andare a "misurare" quanto due parole sono vicine (affini) tra loro. In questo lavoro si è scelto di utilizzare la **distanza coseno** [13], che definiti due vettori "a" e "b", ricordiamo essere:

$$CosSimilarity(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a}^t \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|}$$

Figure 9. Formula distanza coseno.

Il modello mette a disposizione il metodo **most_similar()** grazie al quale è possibile individuare le N parole più vicine ad una determinata parola all'interno dello spazio vettoriale. Il confronto può essere effettuato anche con una parola non presente nel modello, in quanto quest'ultima verrà prima trasformata in un vettore di embedding, poi

viene rappresentata nello spazio vettoriale, ed infine mediante la distanza coseno vengono individuati gli N vettori ad essa più vicini nello spazio. Per dare un'idea dell'utilizzo del modello, viene mostrato nella Figura 10 un esempio nel quale viene chiesto al modello di individuare le 30 parole nel corpus più vicine alla parola "incentivo":

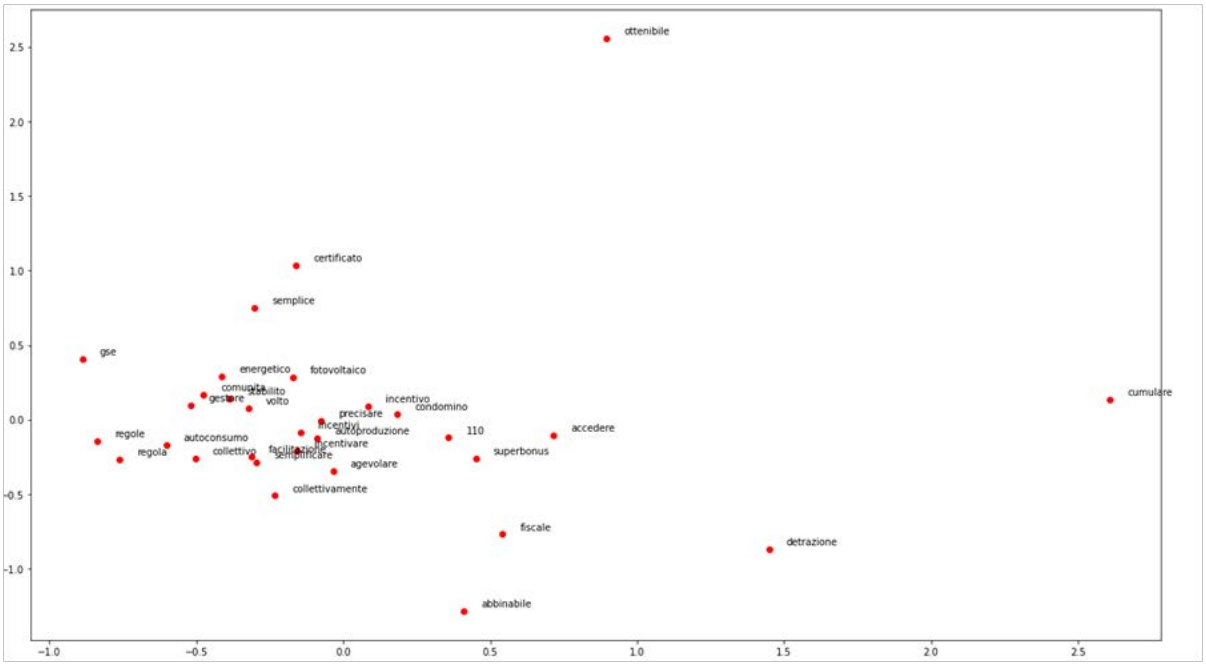

Figure 10. Cluster individuato a partire dalla parola "incentivo".

Trattandosi di un problema multi-dimensionale, per ottenere una rappresentazione grafica in 2-D siamo ricorsi ad una **PCA (Principal Component Analysis)** [14]. Una volta comprese le potenzialità della tecnica, abbiamo provato ad individuare le 10 parole più vicine nello spazio vettoriale alle classi dell'ontologia ARAKNE, in maniera tale da ottenere i cluster di concetti affini ad ogni classe, ricercando tali concetti però solo all'interno del corpus. Essendo 46 le classi dell'ontologia ARAKNE, vengono allora individuati i seguenti 46 cluster:

1. **impianto**:['fotovoltaico', 'installare', 'rinnovabile', 'fonte', 'installazione', 'impiantire', 'potenza', 'energia', 'realizzazione', 'produzione']
2. **investimento**:['investimenti', 'investire', 'concretare', 'miliardo', 'investment', 'fondo', 'dipendere', 'milione', 'stanziare', 'risorsa']
3. **consumo**:['riduzione', 'traguardo', 'prefiggere', 'emissione', 'innalzamento', 'serrare', 'monitorare', 'raggiungere', 'energetico', 'intero']
4. **efficienza**:['efficientamento', 'pa', 'semplificazioni', 'energetico', 'climatizzazione', 'certificazioni', 'riqualificazione', 'rendimento', 'norme', 'servizio']
5. **sostenibilità**:['sostenibilita', 'sostenibile', 'finance', 'modello', 'innovazione', 'economico', 'finanza', 'sociale', 'ambientale', 'morellino']
6. **edificio**:['edilizio', 'installazione', 'immobiliare', 'immobile', 'residenziale', 'riqualificazione', 'intervento', 'efficientamento', 'parcare', 'pubblico']
7. **gestione**:['progettazione', 'garantire', 'grado', 'pianificazione', 'caratteristico', 'vario', 'facilitare', 'funzionale', 'punto', 'altissimo']
8. **strategia**:['crescita', 'profilare', 'attuale', 'chiave', 'cruciale', 'sviluppare', 'globale', 'opportunita', 'competitivo', 'rischio']
9. **fonte**:['rinnovabile', 'energia', 'energetico', 'comunita', 'produzione', 'pulito', 'impianto', 'condivisione', 'ad', 'autoconsumo']
10. **economia**:['circolare', 'riciclare', 'riciclato', 'uneconomia', 'circolarita', 'sostenibilita', 'tessile', 'circolari', 'chimico', 'circular']
11. **settore**:['crescita', 'ricercare', 'industriare', 'emergere', 'globale', 'chiave', 'mercato', 'tendenza', 'dimensione', 'impattare']
12. **produzione**:['consumere', 'produrre', 'modellare', 'fonte', 'energia', 'utilizzare', 'rappresentare', 'ad', 'passare', 'esempio']
13. **energia**:['rinnovabile', 'energetico', 'comunita', 'fonte', 'elettrico', 'pulito', 'produrre', 'fotovoltaico', 'produzione', 'rete']
14. **tecnologia**:['tecnologico', 'artificiale', 'ricercare', 'smart', 'sviluppare', 'futuro', 'innovazione', 'intelligente', 'settore', 'analisi']
15. **obiettivo**:['obiettivare', 'raggiungere', '2030', 'raggiungimento', 'ambizioso', 'emissione', 'europeo', 'target', 'unione', 'serrare']
16. **intervento**:['riqualificazione', 'miglioramento', 'efficientamento', 'realizzazione', 'sismico', 'architettonico', 'edilizio', 'previsto', 'complesso', 'riqualificare']
17. **costo**:['ridurre', 'bolletta', 'abbattere', 'benefico', 'condizione', 'creazione', 'cittadino', 'riduzione', 'prioritariamente', 'rischiare']
18. **mercato**:['globale', 'tendenza', 'dimensione', 'analisi', 'previsione', 'chiave', 'rapportare', 'crescita', '2026', 'fattore']
19. **sviluppo**:['imprese', '2021', '2027', 'recere', 'regionale', 'dipartimento', 'assessore', 'nazionale', 'disegnare', 'panoramico']
20. **fotovoltaico**:['impianto', 'installare', 'pannello', 'energia', 'impiantire', 'solare', 'rinnovabile', 'accumulare', 'installazione', 'energetico']

21. **riscaldamento**:['raffrescamento', 'climatizzazione', 'raffreddamento', 'ibridare', 'palazzoitalia', 'expo', 'raffreddare', 'aicarr', 'aria', 'journal']
22. **fattore**:['tendenza', 'mercato', 'globale', 'chiave', 'crescita', 'emergere', 'attore', 'prospettivo', 'dimensione', 'analisi']
23. **società**:['societaassociazioni', 'politichedellenergia', 'primapagina', 'attiamministrativi', 'attivitaparlamentare', 'internazionali', 'distribuzionee', 'prezzi', 'petroliferied', 'elettrici']
24. **rete**:['distribuzione', 'elettrico', 'energia', 'sistemare', 'prelevare', 'distribuire', 'stoccaggio', 'questa', 'collegare', 'internazionali']
25. **incentivo**:['110', 'incentivare', 'cumulare', 'energetico', 'collettivo', 'autoconsumo', 'facilitazione', 'autoproduzione', 'superbonus', 'gse']
26. **impatto**:['crescita', 'globale', 'analisi', 'impattare', '2026', 'rapportare', 'tendenza', 'dimensione', 'mercato', 'profilare']
27. **biomassa**:['termico', 'geotermico', 'biogas', 'manufatto', 'organico', 'digestione', 'galleggiare', 'idoneo', 'fertilizzare', 'inserimento']
28. **riduzione**:['emissione', 'serrare', 'co2', 'obiettivo', 'ridurre', 'aumentare', 'raggiungere', 'consumo', 'obiettivare', 'ambientale']
29. **misura**:['scarso', 'piano', 'adottare', 'raccomandare', 'rilevanza', 'prevenzione', 'emanare', 'spesare', 'governo', 'parlamento']
30. **risparmio**:['bolletta', 'cittadino', 'vantaggio', 'risparmiare', 'consigliere', 'riscossione', 'autoprodurre', 'orlando', 'abbattere', 'notevole']
31. **valore**:['concretamente', 'sviluppare', 'tutto', 'presupporre', 'economico', 'impattare', 'azienda', 'risultare', 'etico', 'lineare']
32. **fabbisogno**:['144', 'cento', 'soddisfare', 'caseario', 'surplus', 'termoelettrico', 'eccesso', 'sassano', 'protagonisti']
33. **prezzo**:['produrre', 'costare', 'mercato', 'fornito', 'venire', 'quotare', 'lordare', 'flood', 'trend', 'inoltre']
34. **valutazione**:['identificare', 'analisi', 'previsione', 'dimensione', 'quotare', 'filo', 'metodologia', 'fattore', 'approfondito', '2026']
35. **distribuzione**:['rete', 'elettrico', 'energia', 'esempio', 'collegare', 'sistemare', 'utilizzare', 'gestione', 'distribuire', 'dalla']
36. **petrolio**:['nucleare', 'internazionali', 'nazionali', 'carburanti', 'petroliferied', 'prezzi', 'rubricare', 'consumi', 'produzioneidrocarburi', 'approvvigionamentiraffinazione']
37. **emissione**:['co2', 'serrare', 'riduzione', '2030', 'obiettivo', 'raggiungere', '2050', 'target', 'carbonio', 'obiettivare']
38. **elettricità**:['elettrici', 'approvvigionamentiraffinazione', 'energiaelettrica', 'naturalegpl', 'produzioneidrocarburi', 'prezzi', 'fontiefficienza', 'ambientesicurezza', 'consumi', 'idrici']
39. **combustibile**:['fossile', 'idrogenare', 'carburare', 'vergine', 'isolato', 'trimestre', 'carbonio', 'carbone', 'inquinare', 'centrale']
40. **quantità**:['quanti', 'giocattolo', 'bocca', 'venduto', 'riparare', 'quantita', 'elettronico', 'smaltire', 'pila', 'vestito']
41. **certificazione**:['certificazioni', 'certificare', 'bim', 'iot', 'prestazione', 'remoto', 'miglioramento', 'obbligatorio', 'rilasciare', 'costruzioni']
42. **certificato**:['bianco', 'portale', 'gorizia', 'engie', 'ampliare', 'bosch', 'monetizzare', 'aren', 'elettricita', 'neutralita']
43. **carbone**:['fossile', 'cerano', 'miniera', 'protesta', 'centrale', 'riminese', 'protestare', 'gasdotto', 'brindisi', 'out']
44. **gas**:['metano', 'eni', 'centrale', 'serrare', 'carbone', 'interconnessione', 'snam', 'flessibilita', 'stoccaggio', 'questi']
45. **servizo**:['accessibilita', 'accessibile', 'orientato', 'collaborativo', 'processo', 'digitalizzazione', 'innovativo', 'offrire', 'urbano', 'dedicare']

46. **reattore**:['lezione', 'approfondimento', 'redattore', 'didattico', 'psicologico', 'laboratorio', 'acceleratore', 'evento', 'video', 'confrontarsi']

Nella Figura 11 viene anche riportata una rappresentazione grafica di tutti i concetti individuati in ciascuno dei 46 cluster:

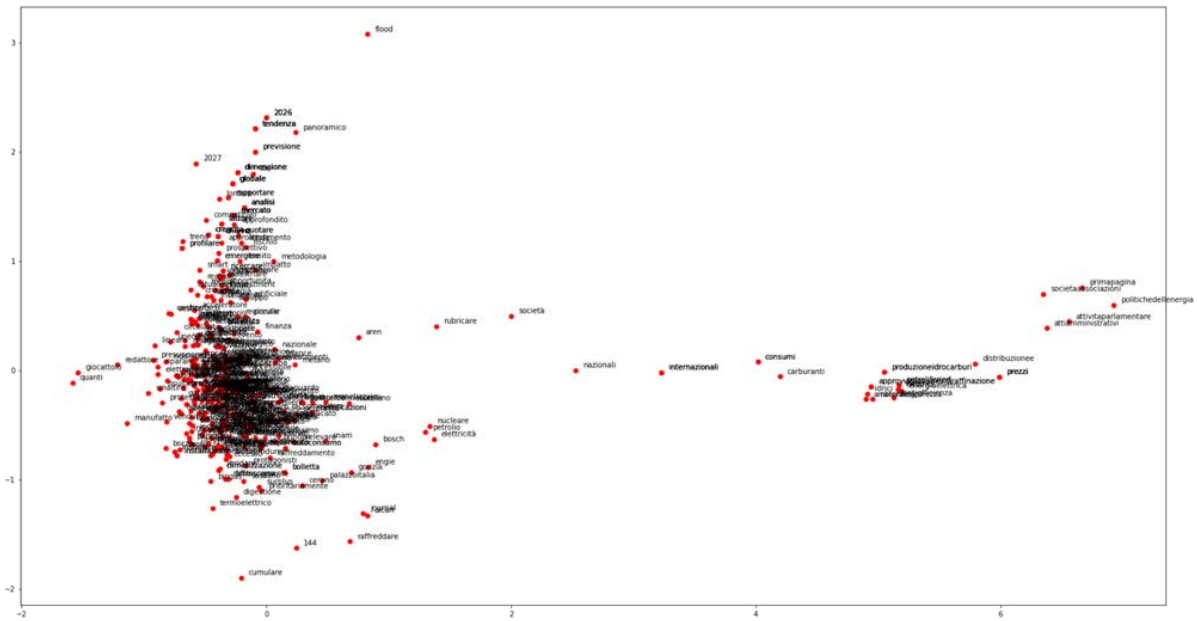

Figure 11. Rappresentazione nello Spazio Vettoriale dei 46 cluster individuati..

L'approccio del Word Embedding è stato accantonato in quanto con esso non si è riuscito a trovare cluster di inerenza adatti al popolamento dell'ontologia, quelli individuati si sono rivelati troppo generici, per questo motivo, abbiamo scelto di perseguire un approccio migliore basato sull'individuazione di entità all'interno dei testi.

5.6.2 Ontology Population con Entità Riconosciute dal Testo

Una delle attività intraprese in questo progetto è stata la Ontology Population, ovvero il popolamento di un ontologia utilizzando le entità riconosciute nei testi negli step precedenti. In un primo momento si è scelto di "semplificare" questa attività andando, invece che a popolare l'ontologia ARAKNE, a creare una nuova mini-ontologia OWL che contiene solo le tre classi di tipologie di entità riconosciute nel testo, ovvero:

- Date di Eventi;
- Luoghi di Eventi;
- Tipologie di Eventi (workshop, conferenze, etc).

Partendo dai risultati ottenuti dalla precedente fase di Name Entity Recognition, è stato realizzato un apposito modulo incaricato di creare la mini-ontologia e popolarla con le entità. Essendo il lavoro in questione realizzato interamente in Python, per soddisfare tali compiti la scelta progettuale è ricaduta sull'utilizzo delle API dell'OWL per il Python, che prendono il nome di **Owlready2** [6]. Si tratta di una libreria Python nata dagli sforzi della Paris University durante il progetto di ricerca VIIIP finanziato dall'ANSM, l'Agenzia francese del farmaco, infatti tutti gli esempi disponibili nella documentazione si riferiscono a farmaci (Lamy, 2017). Essa contiene differenti funzioni e classi in grado di:

- Importare e gestire ontologie OWL 2.0 come oggetti Python;
- Modificare ontologie esistenti;
- Creare nuove ontologie;
- Aggiungere nuove classi, istanze e proprietà;
- Definire relazioni inverse, funzionali, funzionali inverse e transitive;
- Eseguire reasoning su ontologie tramite un reasoner **HermiT** già incluso nel pacchetto;
- Eseguire query SPARQL;

Anche se si tratta di un pacchetto nato alcuni anni dopo rispetto le più famose API di OWL per Java (**OWL API**) (Di Martino & Graziano, 2021) (Di Martino et al., 2021, #), esso presenta alcune limitazioni come ad esempio:

1. L'impossibilità di rinominare con la stessa label entità di tipo diverso (ad esempio non è possibile creare un'istanza che abbia lo stesso nome di una proprietà esistente). Per assicurarci che tale errore non venisse mai commesso dal modulo è stato aggiunto il seguente controllo: "if(str(type(onto[token])) == "<class 'NoneType'>"):", il quale impedisce di creare una nuova istanza se il suo nome coincide con quello di qualunque altro elemento dell'ontologia;
2. Le istanze con le quali si popolano le varie classi dell'ontologia devono avere dei nomi "puliti", ovvero privi di alcuni caratteri speciali che possono rendere difficoltoso il processing dell'ontologia da parte di tool come **Protegè**. Anche per questo punto è stata adottata un'apposita soluzione, che consiste nel ripulire la stringa corrispondente al nome dell'istanza da aggiungere all'ontologia mediante una serie di espressioni regolari;

Il modulo di Ontology Population effettua anche un arricchimento degli attributi delle entità riconosciute negli step precedenti; in particolare, vengono aggiunte due informazioni di vitale importanza ai fini della visualizzazione delle entità (riconosciute dalla NER) all'interno di un testo, che sono lo **Start Index** e lo **Stop Index** di ogni entità all'interno della stringa testuale. Tali attributi sono stati ottenuti ricorrendo alla libreria **Spacy** [4] nel seguente modo:

- Lo Start Index si ottiene banalmente richiamando l'attributo "**idx**" sull'entità in analisi;
- Lo Stop Index si ottiene sommando allo Start index la lunghezza del testo del token in analisi;

L'informazione disponibile in partenza per ogni entità riconosciuta riguardava invece la **Token Position**, grazie alla quale è stato semplice ed immediato ricreare il modello Spacy che occorreva per prelevare queste informazioni. Dunque per ogni entità riconosciuta nel testo è stata creata un'apposita istanza, alla quale sono state associate le seguenti data properties:

- **id_alert**: identificativo del testo nel quale quell'entità occorre almeno una volta;
- **token_position**: posizione del token all'interno del modello di Spacy;
- **start_index**: posizione del carattere iniziale del token all'interno del testo;
- **stop_index**: posizione del carattere finale del token all'interno del testo;
- **has_label**: label con la quale è stato etichettato quel token ("LUOGO", "EVENTO", "DATA");
- **token_content**: testo associato all'entità;
- **has_Description**: testo completo da cui sono state estratte le entità di riferimento.

Il risultato sperimentale di questo task è mostrato nella Figura 12, nella quale viene messo in evidenza il popolamento della mini-ontologia con un luogo riconosciuto in un testo:

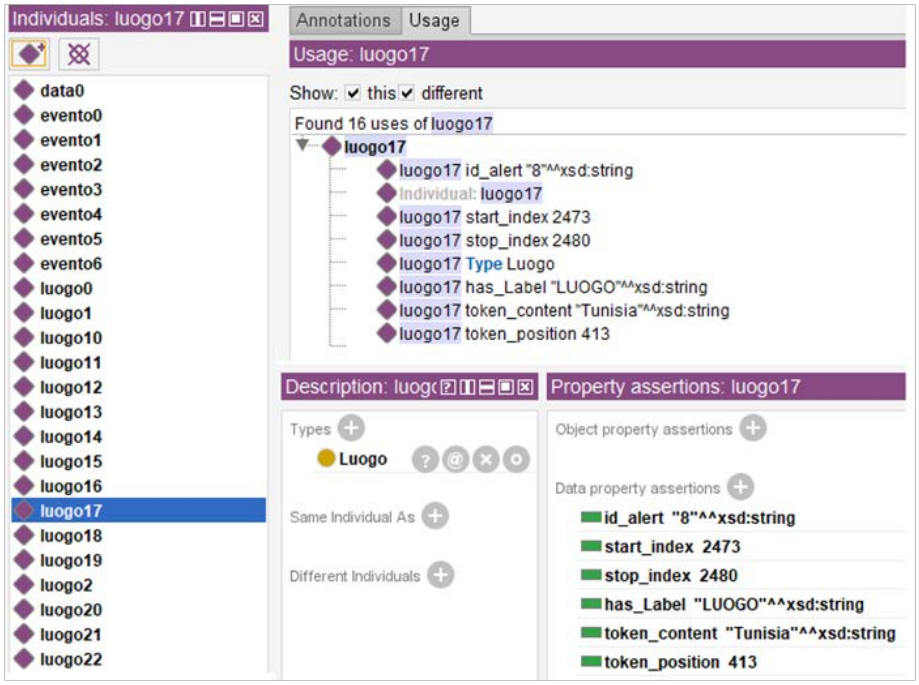

Figure 12. Esempio di Popolamento della Mini-Ontologia .

Come si può osservare dalla Figura 12, le istanze della stessa classe sono rinominate con il nome della classe seguito da un numero progressivo; l'istanza in particolare riferita in Figura 12 è "l*uogo17*", che occorre almeno una volta nel testo con **ID_Alert 8**, precisamente nella posizione [2473:2480] del testo originale (non pulito).

La mini-ontologia realizzata è stata popolata con ben 1345 individui.

5.7 Task 7: Visualization

Una volta che la NER realizzata negli step precedenti ha riconosciuto le entità, e che queste ultime sono state utilizzate per popolare una mini-ontologia, il passo successivo è stato quello di investigare le possibili strategie per la visualizzazione delle entità all'interno del testo. In particolare, sono stati percorsi tre possibili approcci:

5.7.1 Visualizzazione con il modulo Displacy di Spacy

La libreria Spacy di Python mette a disposizione un utilissimo modulo di visualizzazione dei risultati di una NER, chiamato **Displacy** [7]. Esso prende in ingresso un oggetto di tipo "**Doc**" ottenuto dal processing del testo mediante uno Spacy model, e visualizza tutte le entità riconosciute dalla NER all'interno del testo mediante delle apposite label colorate. Siccome però le entità sono state riconosciute in uno step differente, non si dispone di un modello di NER Spacy nell'attuale step, tuttavia però è stato molto semplice creare un Blank Spacy model e riempirlo con tanti oggetti di tipo Span quante sono le entità riconosciute nello step precedente. A titolo di esempio, nella Figura 13 viene mostrato l'utilizzo di displacy con il testo con Id_Alert 8:

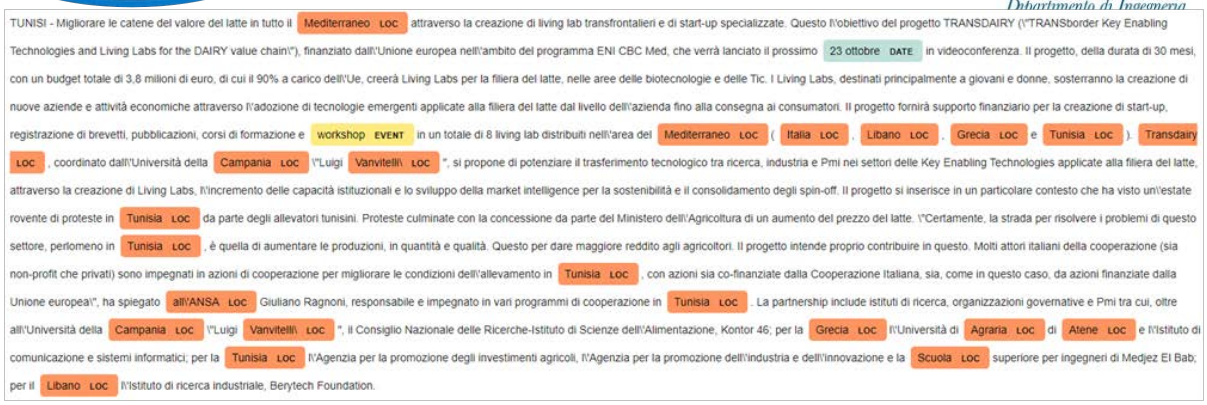

Figure 13. Esempio di Visualizzazione di Entità in un Testo con il Modulo Displacy di Spacy.

Avendo alla base un modello Spacy, questo approccio soffre di un grande problema, noto come **SpaCy overlapping entities**: non è possibile effettuare annotazioni multiple, o per meglio dire, non è possibile etichettare un token o parte di un token con label differenti. Ciò significa che bisogna essere molto accorti quando si affida ad un programma automatizzato il compito di aggiungere entità ad una NER in Spacy, in quanto è molto facile incappare in questo errore. Il vantaggio di tale approccio è la semplicità e l'immediatezza con il quale la visualizzazione può essere ottenuta. Inoltre questa visualizzazione può essere facilmente integrata all'interno di un notebook Colab o Jupyter settando il parametro **jupyter=True** del metodo **render()** di Dispacy.

5.7.2 Visualizzazione con Displacy-ent Javascript

Dato il grande successo che ha riscontrato Displacy, con il tempo sono state realizzate diverse librerie open-source che hanno esteso il modulo Displacy di Spacy mettendolo a disposizione di chiunque. Quella che ha avuto maggiore successo è la libreria JavaScript **Dispacy-ent** [8]. Si tratta di una libreria leggera ed estensibile che recupera le annotazioni di entità da un formato JSON e le trasforma in HTML. Displacy-ent fornisce delle API che consentono ad un programmatore web, con poche righe di codice, di fare il rendering grafico di un'analisi generata dai servizi di spaCy a partire da un qualunque testo in qualunque lingua riconosciuta da Spacy. Se invece si possiedono già dei risultati di una NER, è possibile costruire un proprio Parser che, utilizzando le sole tecnologie HTML, CSS, e Javascript, consente di visualizzare le entità nel testo. Nella Figura 14 è mostrato il suo utilizzo utilizzando come esempio lo stesso testo di prima:

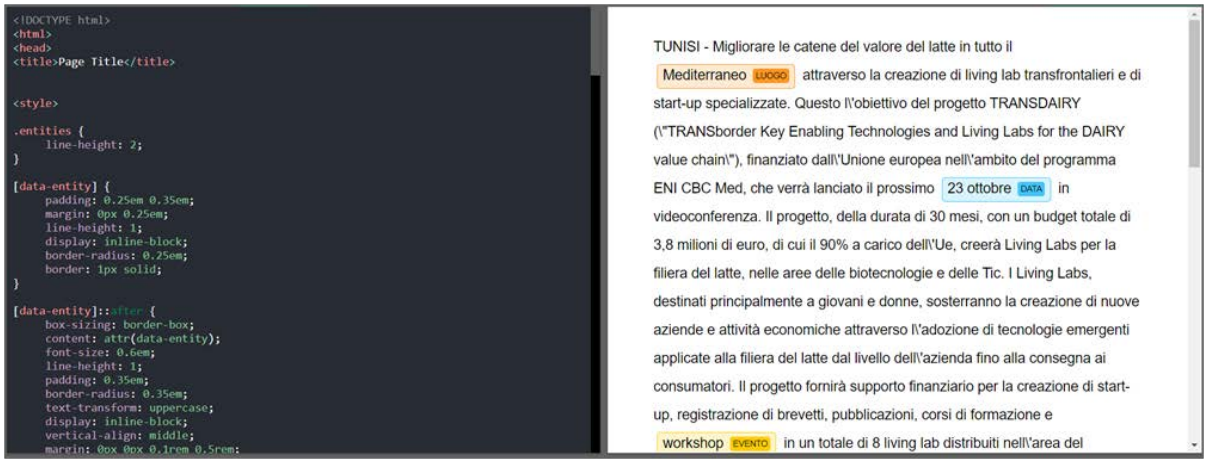

Figure 14. Esempio di Visualizzazione di Entità in un Testo con il Modulo Displacy-ent.

Esistono anche delle versioni online [9], come quello mostrato nella Figura 15, che mettono a disposizione queste API di visualizzazione, oltre a dei modelli di NER già addestrati per differenti lingue, che possono essere utilizzati per provare questa visualizzazione.

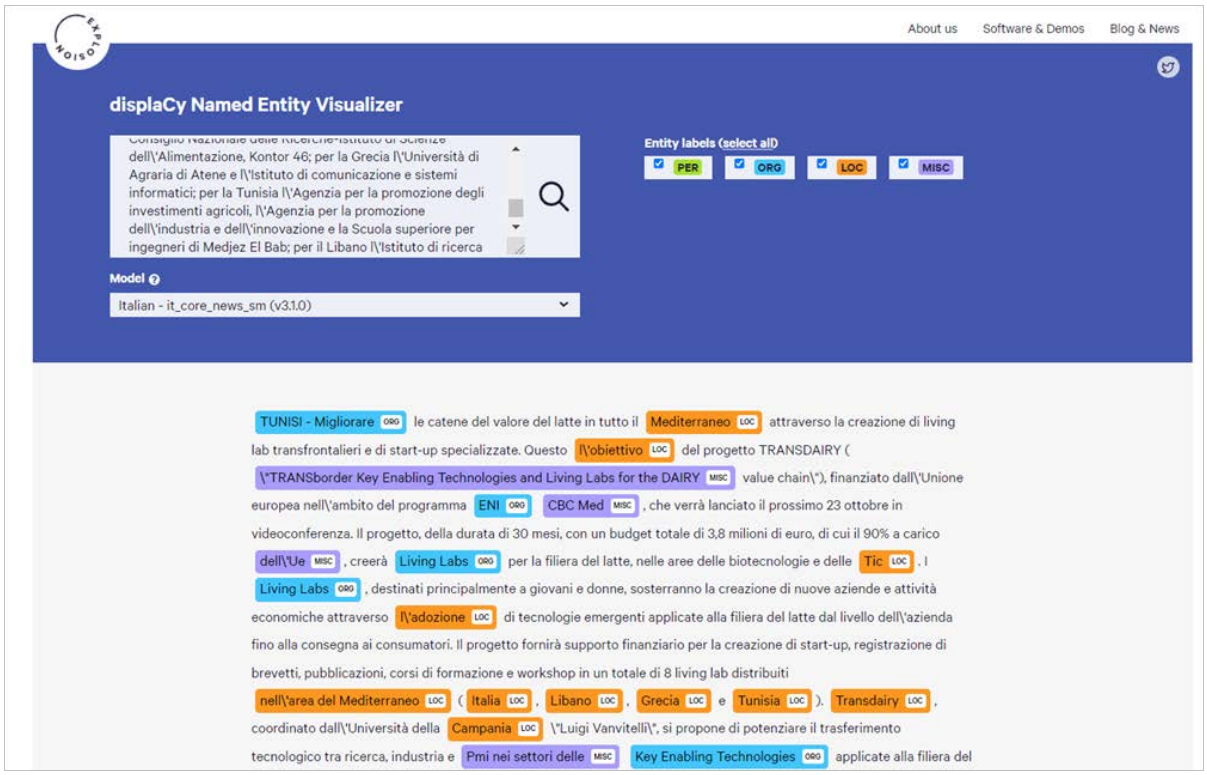

Figure 15. NER online messa a disposizione da Displacy-ent.

Purtroppo, come è osservabile dalla figura, i modelli già pre-allenati per la lingua italiana non sono molto efficienti, ed inoltre, le label utilizzate per annotare le entità sono davvero poche e non molto utili ai fini degli obiettivi di questo lavoro. Vengono mostrati alcuni PRO e CONTRO di questo approccio:

- **PRO**: non c'è il problema dello **SpaCy overlapping entities**, dunque è possibile applicare label multiple sulla stessa entità;

- **PRO**: è molto più leggero di BRAT, infatti richiede solo poche righe di codice CSS e Javascript per la configurazione, mentre invece BRAT richiede più di 500 righe di codice CSS, oltre ad 11 dipendenze JavaScript, incluso jQuery con due plug-in;
- **CONTRO**: non ha la potenza espressiva di BRAT;
- **CONTRO**: non è semplicissimo costruire un parser per utilizzare questo modulo nelle proprie applicazioni;

5.7.3 Visualizzazione con Brat

Brat [10] è un tool web-based per l'annotazione del testo, cioè per aggiungere etichette a documenti di testo esistenti.
Questo tool è disponibile sia in versione online che in versione desktop, e viene messo a disposizione a tutti gli utenti che desiderano annotare/validare testi manualmente (Di Martino et al., 2021). Brat mette a disposizione una visualizzazione web-based molto potente ed intuitiva a livello grafico, ed essendo il progetto open-source, è stato possibile "estrarre" questo sistema di visualizzazione in maniera da utilizzarlo in questo lavoro.
A titolo di esempio, nella Figura 16, è mostrato con una visualizzazione stile Brat lo stesso testo analizzato in precedenza:

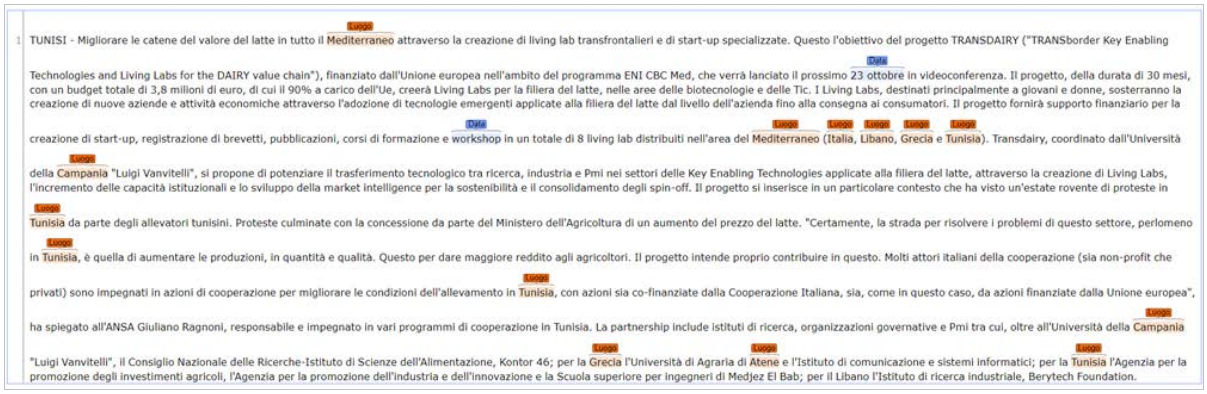

Figure 16. Esempio di Visualizzazione di Entità in un Testo con con stile Brat .

Come è possibile osservare dalla Figura 16, le etichette colorate con le annotazioni vengono inserite al di sopra della parola a cui si riferiscono, delimitandole mediante una parentesi graffa. Per ottenere una visualizzazione di questo tipo, occorre che il file contenente i risultati della NER sia opportunamente strutturato con un formato standard chiamato **CoNLL-U**, il quale deve contenere necessariamente i seguenti campi:

- ID Progressivo dell'annotazione;
- Nome dell'entità annotata;
- Indice iniziale e finale dell'entità nel testo;
- Label utilizzata per annotare l'entità;

Per utilizzare questo tipo di visualizzazione, sarà necessario realizzare un parser che, a partire da un file in formato CoNLL-U, costruirà l'HTML che permetterà di visualizzare il testo con le entità opportunamente taggate attraverso una visualizzazione brat-like. Il file formato CoNLL-U potrebbe essere ottenuto dagli step precedenti strutturando opportunamente i risultati della NER.

5.7.4 Visualizzazione Integrata Ontologia e Testo Annotato

Stiamo lavorando in fase sperimentale, anche sulla possibilità di realizzare un modulo che vada ad integrare in un'unica interfaccia grafica, la visualizzazione della tassonomia di un'ontologia da un lato e dall'altro lato il testo con i tag visualizzati con una delle modalità viste in precedenza, sperabilmente Brat oppure Dispacy-ent. L'idea che stiamo sperimentando si prefissa come obiettivo quello di navigare tra le classi o istanze dell'ontologia ed "evidenziarle" nel testo mostrando le annotazioni corrispondenti nel seguente modo:

- In caso si selezioni un'istanza, viene visualizzata nel testo, se presente, l'occorrenza di essa (mostrando la parola etichettata);
- In caso si selezioni una classe, vengono visualizzate nel testo tutte le parole presenti in esso che sono state etichettate con la label corrispondente a quella classe (Data, Luogo, oppure Evento);

Per quanto riguarda la visualizzazione e la navigazione dell'ontologia, è stato sviluppato un apposito modulo che si intende "integrare" all'interno del sistema in esame, ed "affiancare" ad un testo visualizzato. Anche per questo secondo punto sarà necessario preparare ed integrare un ulteriore modulo di parsing per la creazione dinamica dei files HTML (con i testi annotati) da mandare in visualizzazione. La versione attuale del modulo di navigazione dell'ontologia è realizzato mediante una combinazione di Java, PHP, e tecnologie Web varie (HTML, CSS, JavaScript, JQuery, librerie JavaScript). La Figura 17 mostra il mockup di ciò che si vorrà realizzare:

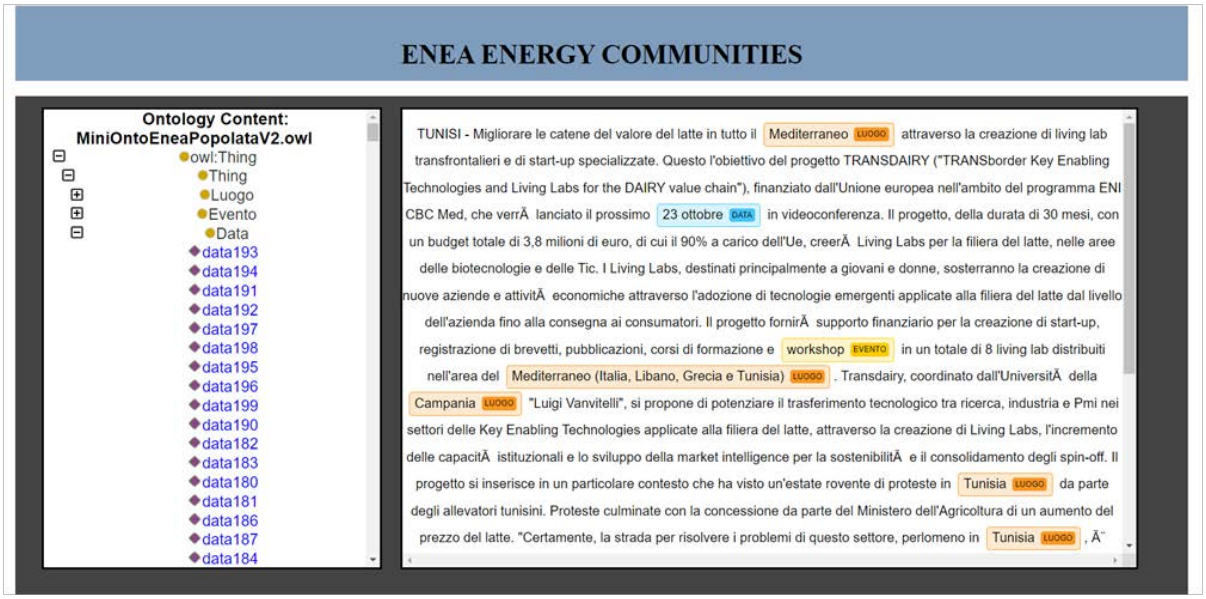

Figure 17. Vista Integrata Ontologia + Testo Annotato con Entità dell'ontologia.

5.7.5 Visualizzazione Geografica

L'obiettivo in questo caso è stato quello di realizzare una visualizzazione geografica in grado di dare all'utente un overview generale di tutti gli eventi posizionandoli all'interno di una mappa.

In particolare l'idea di fondo è stata quella di associare, ad ogni relazione evento/luogo/data un marker, posizionato sulla mappa tramite le coordinate associate al luogo della tripla. Il marker può essere di 2 colori:

- Rosso se la Sentiment Analysis ha restituito un risultato negativo
- Azzurro se la Sentiment Analysis ha restituito un risultato positivo

Successivamente, al click di un marker, viene mostrata:

- un'anteprima con il titolo dell'articolo
- una sezione "INFO" per accedere a tutte le informazioni in modo più dettagliato. Cliccando su "INFO" si viene poi reindirizzati ad un' altra schermata con **TUTTE** le informazioni rilevate sia dalla Sentiment Analysis che dalla Ricerca Eventi/Data/Luoghi
- cliccando su "brat" si accede alla visualizzazione del testo in formato brat

Per la realizzazione è stata sviluppata una web app in flask. Il **front-end** è stato quindi scritto tramite linguaggi HTML/CSS, mentre per quanto riguarda il **back-end** è stato utilizzato Python.
Per la localizzazione e posizionamento degli eventi sulla mappa è stata utilizzata la libreria **folium** che mette a disposizione sia la mappa che la possibilità di aggiungere dei marker. Tuttavia per il posizionamento dei marker tale libreria necessita di prendere in input non un testo ma delle coordinate. Per la risoluzione di questo problema è stata utilizzata un altra libreria, **geocoder,** che, dato una stringa restituisce le coordinate associate al luogo. Prima di aggiungere il marker alla mappa, come possiamo notare in Figura 18, i marker avranno il colore rosso se il sentiment è negativo, azzurro in caso di sentimento positivo.

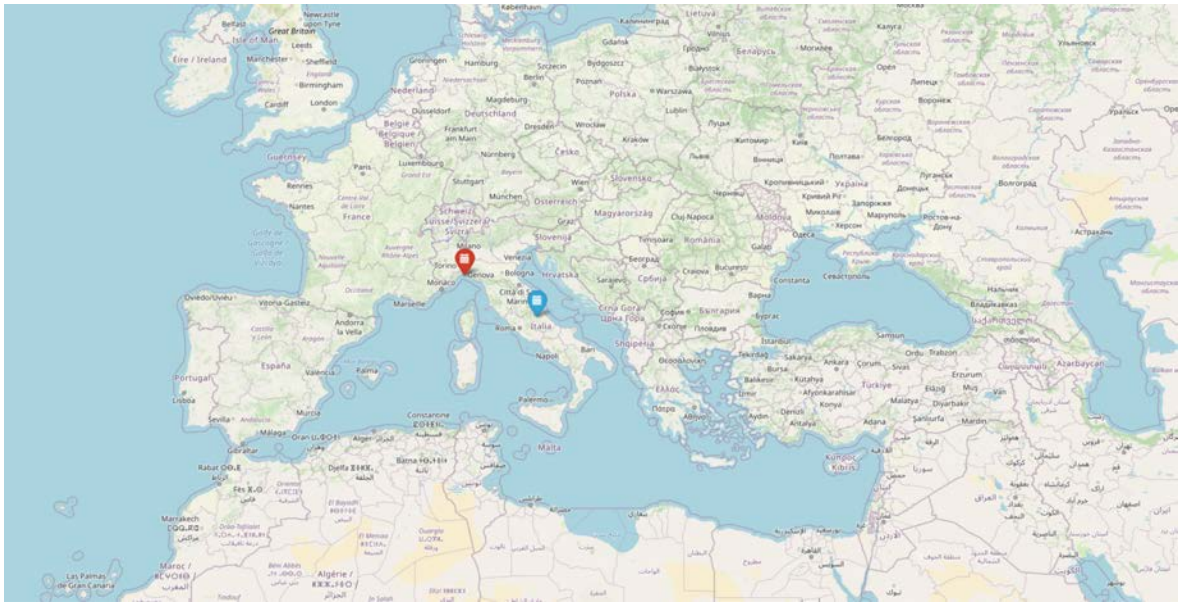

Figura 18. Vista di Eventi Posizionati sulla mappa con marker colorati in base alla S.A.

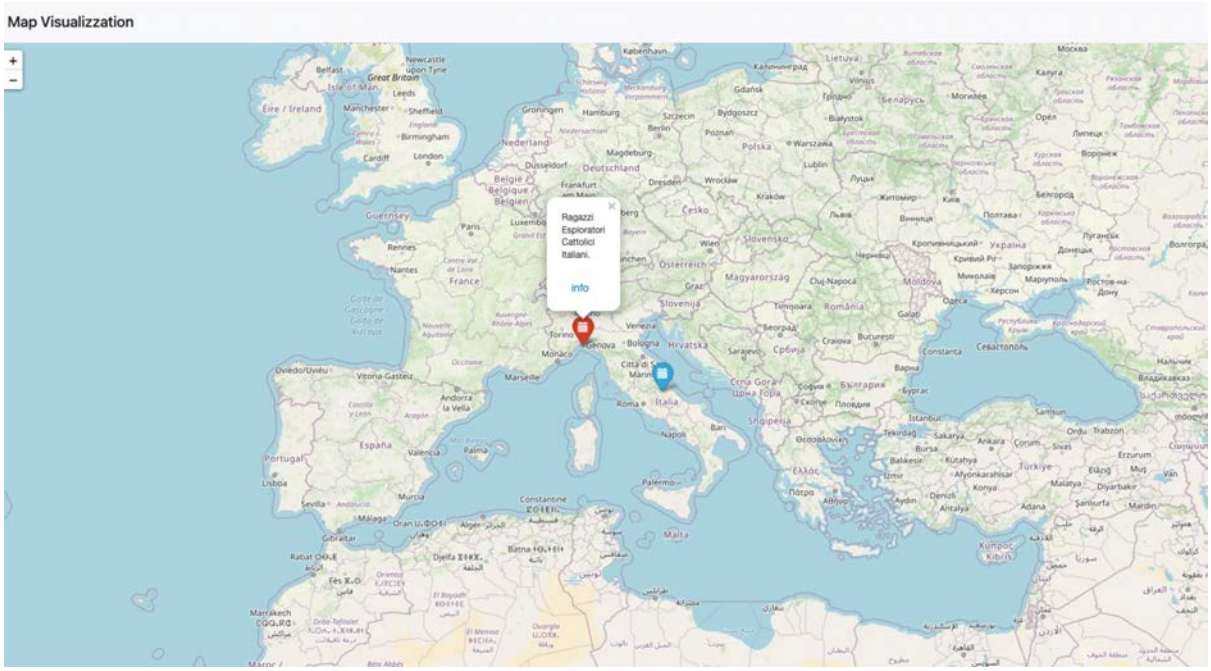

Figura 19: Vista Espansione Marker

Il reindirizzamento alla schermata “info” Figura 20 per visualizzare i dettagli di ogni evento è possibile grazie ad un id univoco che da la possibilità di associare a ciascuno le rispettive informazioni:

- Titolo
- Descrizione
- Relazioni Rilevate
- Sentiment Analysis

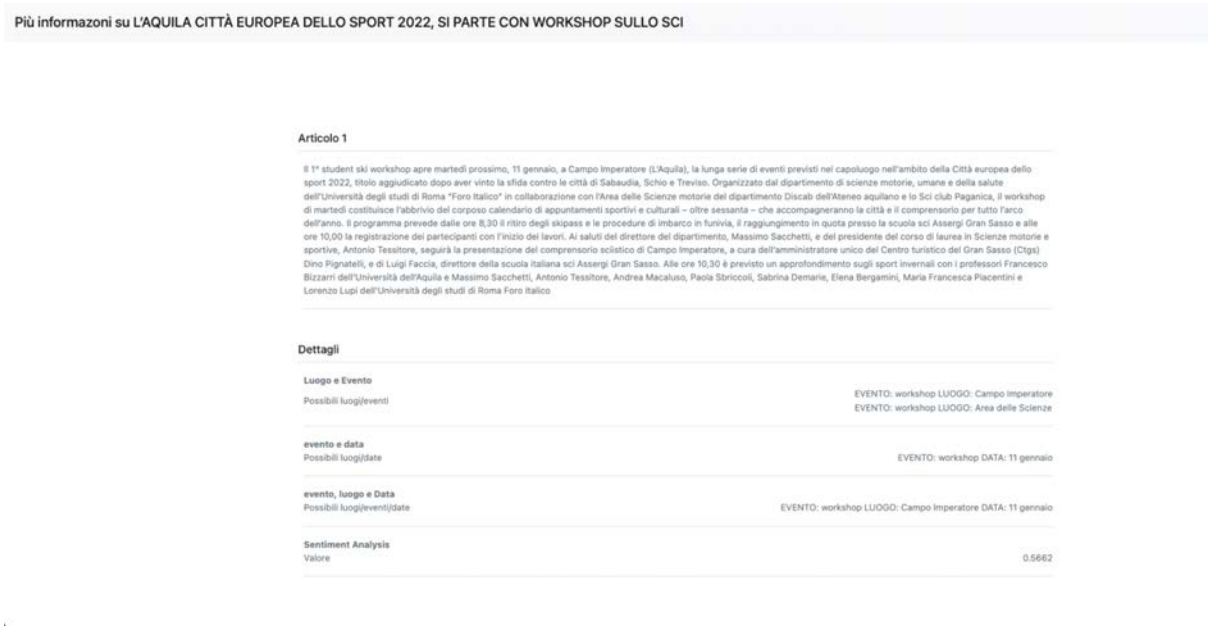

Figura 20: Vista schermata “INFO”

Si sta lavorando, in fase sperimentale, anche sulla possibilità di effettuare un filtraggio non solo di tipo spaziale, ma anche di tipo temporale. In particolare l'idea sarebbe quella di inserire all'interno della barra superiore un modulo date-picker dalla libreria bootstrap che permetta di selezionare un range di date di interesse. In questo modo l'utente avrebbe la possibilità di filtrare contemporaneamente sia per data che per luogo. Inoltre ricollegandoci a ciò che è stato discusso nei sottocapitoli precedenti si sta lavorando alla possibilità di inserire non solo la sezione "info" nel pop-up dei marker, ma anche un reindirizzamento ad una visualizzazione tramite BRAT associata al rispettivo articolo come mostrato in Figura 21.

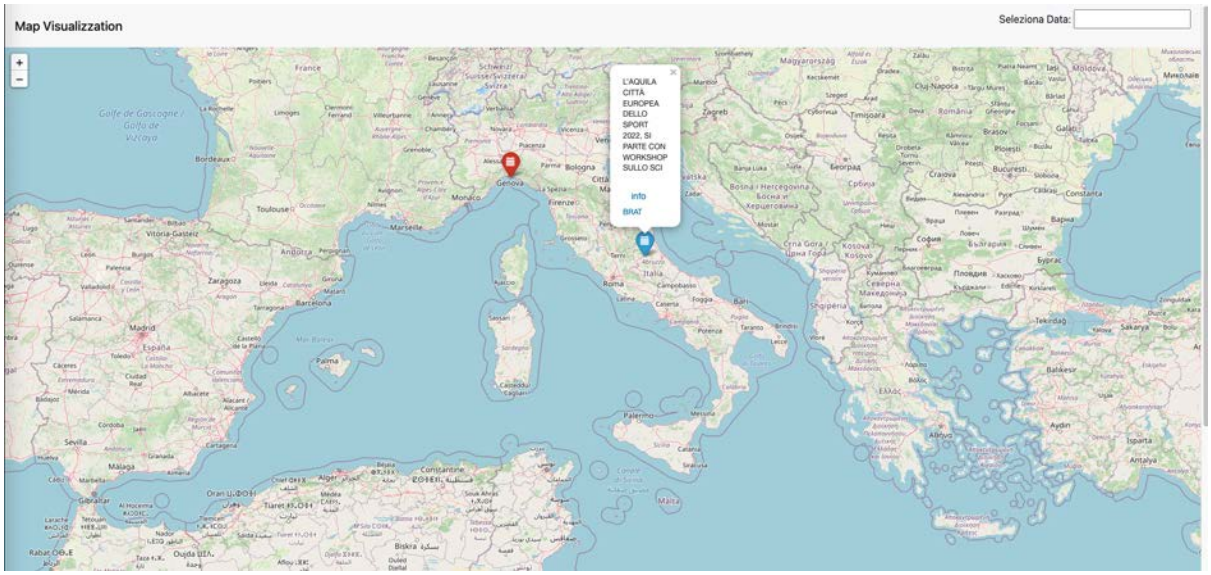

Figura 21: Prototipo Visualizzazione Spaziale-Temporale con collegamento a BRAT

5.7.6 Visualizzazione Dashboard

L'obiettivo di questa parte di ricerca è stata quello di raggruppare le caratteristiche più importanti di alcune tipologie di visualizzazione descritte in precedenza mediante l'uso di una dashboard.
Inoltre l'implementazione di quest'ultima ha permesso di poter realizzare appositi filtri inerenti agli elementi da visualizzare Figura 22
I filtri applicabili sono:

- Data Filter: possibilità di selezionare gli eventi a partire da un dato mese e/o un dato anno
- Location Filter: possibilità di selezionare l'evento in base alla tipologia di luogo in cui è svolto (online, presenza)
- Event Filter: possibilità di selezionare la tipologia di evento (workshop ,conferenze, ecc…)
-

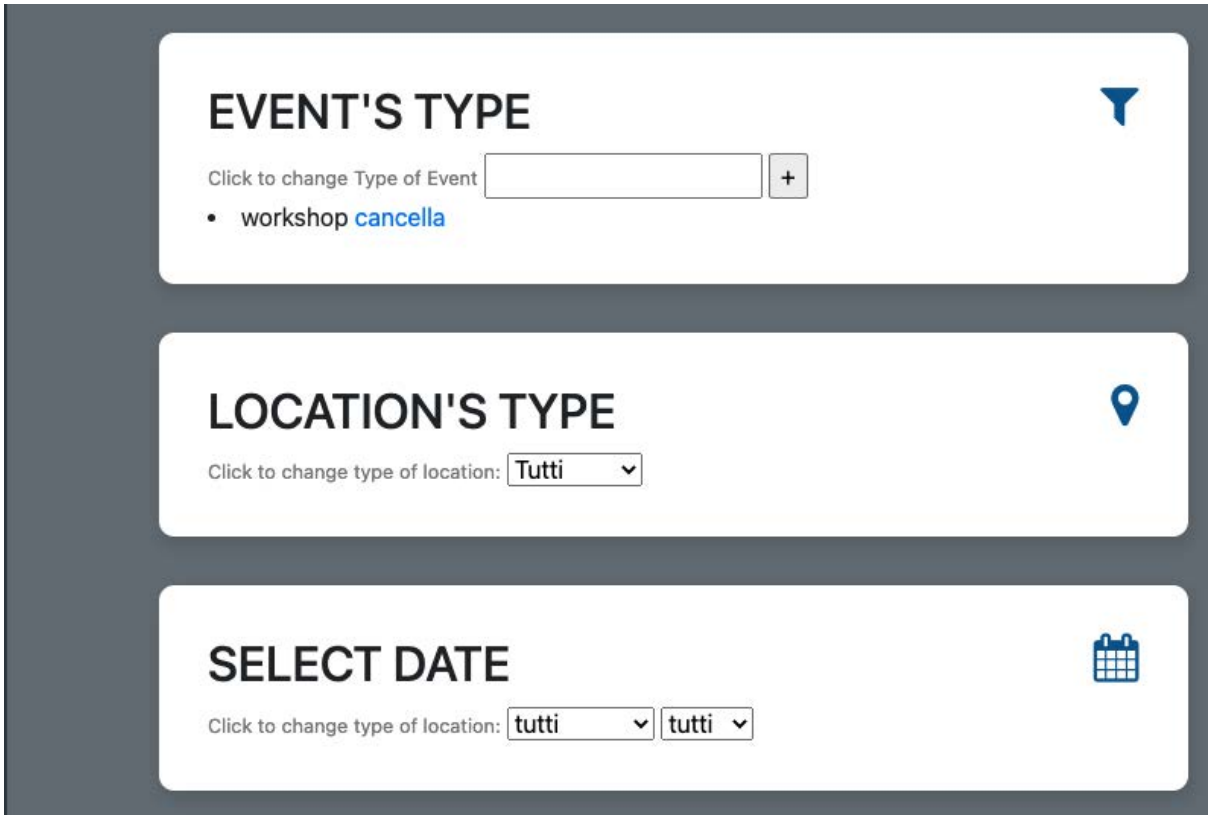

Figura 22: Sezione relativa al filtraggio in HTML

In generale la dashboard è strutturata in due macro aree:

- Una side-bar che aiuta l'utente nella navigazione(presente in qualsiasi pagina dell'applicazione).
- Il corpo della pagina(che varia a seconda della sezione in cui ci troviamo).

Fino ad ora sono state implementate 3 pagine e sono le seguenti:

1. Home: la pagina principale dove sono presenti sia i collegamenti alle visualizzazioni (MapView e ListView) che i selettori inerenti al filtraggio Figura 23
2. Map Vew: dopo aver applicato i filtri ci si potrà spostare all'interno della visualizzazione geografica così da poter visualizzare tutti i risultati ottenuti dal filtraggio. Figura 24
3. List View, dopo aver applicato i filtri ci si potrà spostare nella sezione list view che ci permetterà di visualizzare tutti i risultati ottenuti dal filtraggio sotto forma di lista cliccabile. Figura 25

Sia per la MapView che per la ListView si ha la possibilità di espandere l'evento ed essere reindirizzati ad una altra schermata contenente sia le informazioni aggiuntive (Sentiment Analysis, Match rilevati) che la visualizzazione del testo tramite Displacy.

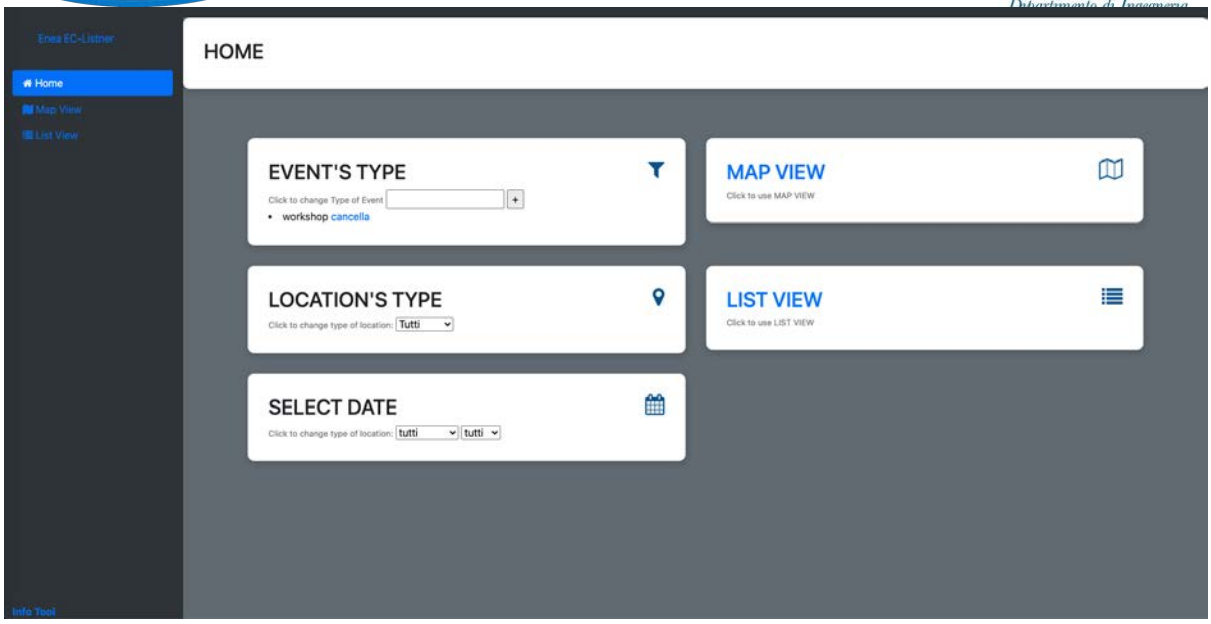

Figura 23 : Home page

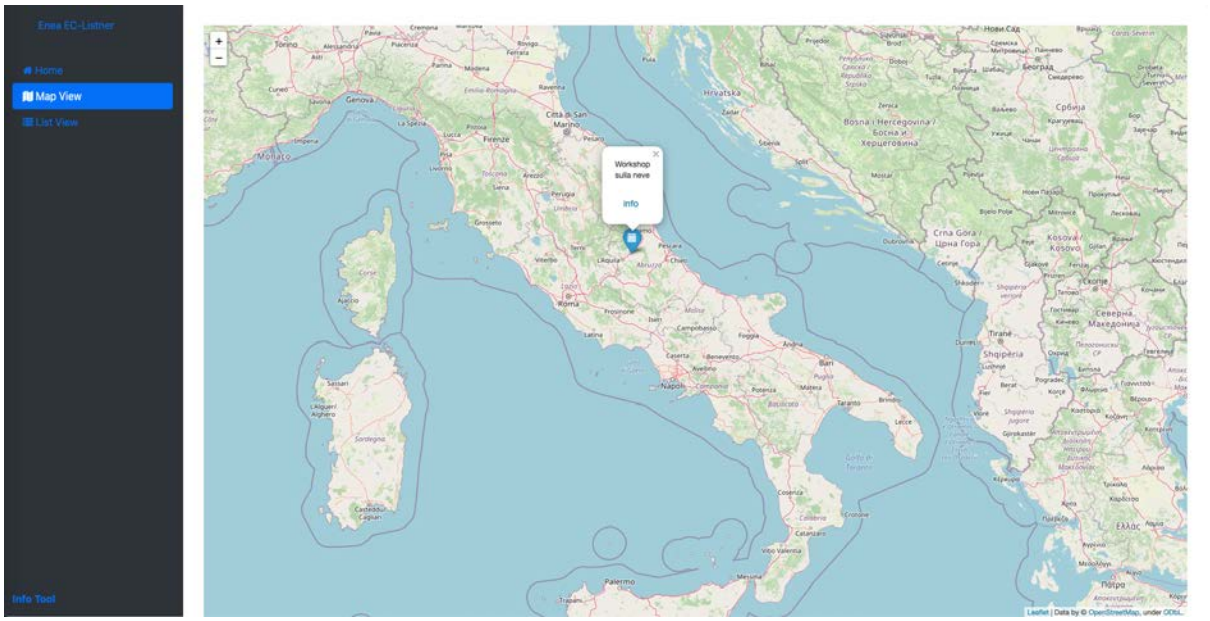

Figura 24: Sezione Map View

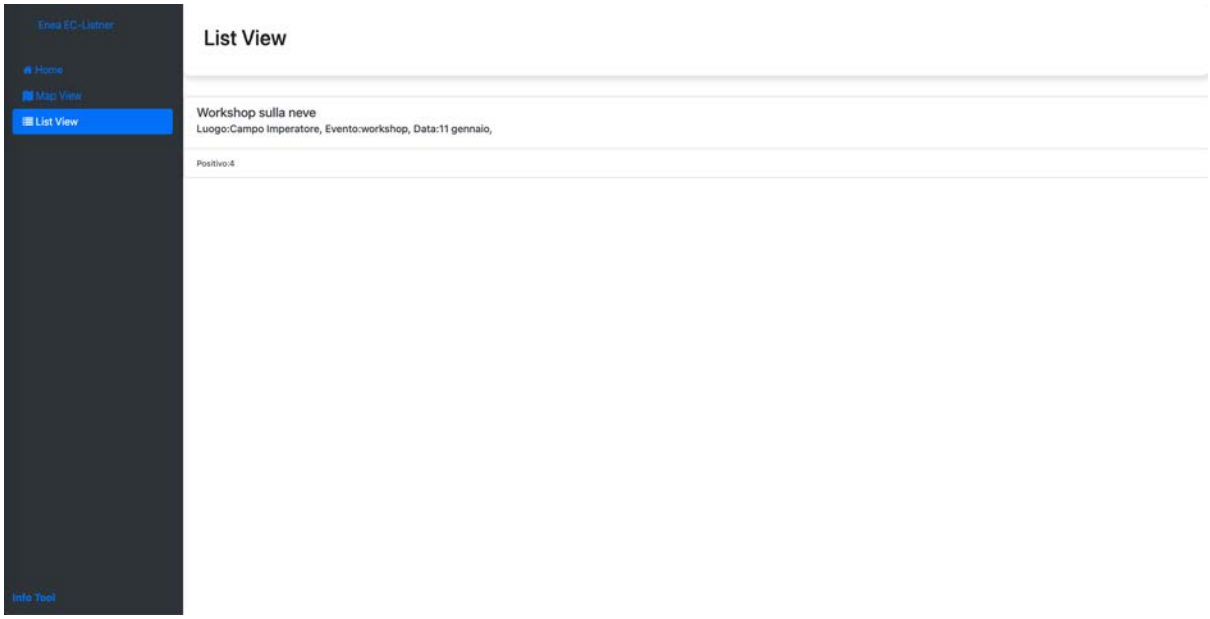

Figura 25: Sezione List View

Attualmente l'intero tool essendo in fase di sviluppo preleva i dati da un file di tipo JSON, tuttavia l'idea sarebbe certamente quella di prelevarli dall'ontologia.
L'idea, attualmente in fase di sviluppo, è quella di utilizzare delle query SPARQL specifiche per ogni filtraggio selezionato (utilizzando la libreria "owlready2" in Python). I risultati prodotti da questa query saranno successivamente adattati al metodo di visualizzazione selezionato e passati al HTML per visualizzarli.

6 Valutazione quantitativa dei risultati

In questa sezione sono riportati i risultati sperimentali e le relative valutazione di alcune delle attività descritte in precedenza, specificamente quelle per le quali è stato possibile utilizzare metriche di valutazione della efficacia delle tecniche utilizzate, e cioe' i tasks: T2-Validazione di Pertinenza e T3-Riconoscimento Luoghi/Date/Eventi.

6.1 Task 2: Validazione di Pertinenza

Per verificare l'efficacia dei meccanismi di classificazione applicati in questo task, sono stati applicati i classici parametri di valutazione utilizzati per gli algoritmi di Machine Learning e simili, quali il calcolo della Precisione, della Recall, e dell'accuratezza.
Nello specifico, il classificatore basato su Spacy è stato innanzitutto applicato ad un Corpus pre-esistente di notizie considerate inerenti il contesto delle Comunità Energetiche, costituito da 2348 testi differenti. Di questi, 2247 sono stati riconosciuti come correttamente inerenti il dominio di interesse (**TP**=2237), mentre 101 non sono stati riconosciuti (**FN**=101).
Sono state inoltre scaricate altre notizie, sicuramente non inerenti il dominio delle Comunità energetiche, e l'approccio è stato nuovamente applicato ai testi estratti.

Le parole chiave utilizzate per scaricare le notizie non inerenti sono state le seguenti:"ciclismo","pallavolo","medicina","arte","spettacolo","calcio","cinema","politica","informatica","psicologia","animali","salute","covid","decreto","moda","finanza","smart working","inflazione". Queste hanno permesso di identificare 350 testi differenti, esaminati poi a mano per verificare che effettivamente essi non fossero in alcun modo inerenti il contesto preso in esame. Il nostro modulo, applicato ai testi estratti, ha riscontrato 0 (zero) testi pertinenti (**FP**=0) e 350 testi non pertinenti (**TN**=350).
Su questa base risulta:
Precisione = TP/(TP+FP)*100=100%
Recall= TP/(TP+FN)*100=96%
Accuratezza=(TP+TN)/(TP+FP+FN+TN)=96%

Il sistema risulta dunque sufficientemente accurato da poter essere applicato al nostro problema reale.

6.2 Task 3: Riconoscimento Luoghi/Date/Eventi

Dopo aver sviluppato il programma per validare sperimentalmente il suo effettivo funzionamento sono stati effettuati alcuni test.
Di seguito sono riportati i risultati di:

- Precisione (TP/(TP+FP)*100)
- Recall (TP/(TP+FN)*100)
- Accuratezza ((TP+TN)/(TP+FP+FN+TN))

di tutte le funzioni sviluppate su un centinaio di testi.
Ricerca Eventi: Precisione=89%, Recall=100%, Accuratezza=96%
Ricerca Luoghi: Precisione=56%, Recall =97%, Accuratezza=56%
Ricerca Luoghi (Online): Precisione=94%, Recall =100%, Accuratezza=98%
Ricerca Date

- **Date con traduzione testo**: Precisione=21%, Recall=53%, Accuratezza=28%
- **Date con espressione regolare 1o tipo:** Precisione=18%, Recall=82%, Accuratezza=22%
- **Date con espressione regolare 2o tipo:** Precisione=%62, Recall=88%, Accuratezza=68%

Match con distanza Token

- **Ricerca Evento/Luogo:** Precisione=61%, Recall=93%, Accuratezza=83%
- **Ricerca Evento/Data:** Precisione=73%, Recall=87%, Accuratezza=92%
- **Ricerca Evento/Luogo/Data:** Precisione=44%, Recall=91%, Accuratezza=84%
- **Ricerca Evento/Luogo Online:** Precisione=90%, Recall=90%, Accuratezza=97%
- **Ricerca Evento/Luogo/Data Online:** Precisione=%60, Recall=85%, Accuratezza=94%

Match con Pattern

- **Ricerca Evento/Luogo:** Precisione=49%, Recall=93%, Accuratezza=76%
- **Ricerca Evento/Data:** Precisione=42%, Recall=80%, Accuratezza=86%
- **Ricerca Evento/Luogo/Data:** Precisione=36%, Recall=81%, Accuratezza=81%
- **Ricerca Evento/Luogo Online:** Precisione=77%, Recall=100%, Accuratezza=97%
- **Ricerca Evento/Luogo/Data Online:** Precisione=42%, Recall=75%, Accuratezza=94%

Final Match: Precisione=71%, Recall=83%, Accuratezza=82%

7 Conclusioni e sviluppi futuri

Le attività di ricerca descritte nel presente deliverable hanno permesso di definire e sperimentare una serie di metodologie, poi tradotte in applicazione di tecniche e implementazione di software, per l'analisi di notizie scaricate in batch dalla rete, attraverso opportune API, e la loro elaborazione mediante tecniche di Natural Language Processing, Semantic Based, e orientate al Machine Learning.
In particolare, è stata definita una Pipeline Big Data le cui attività, suddivise in diversi Task oggetto di ricerca, rappresentano il movimento del flusso di dati in ingresso, dall'Ingestion alla Visualization delle informazioni finali. La Pipeline è stata pensata per un'analisi Batch di informazioni, costituite come detto da News, tuttavia essa è predisposta per input di tipo Streaming, quali ad esempio Tweet o commenti provenienti da altre piattaforme social, che possono essere analizzati con le medesime tecniche descritte nel presente deliverable, salvo operare in maniera differente nella fase di Ingestion vera e propria, dove possono essere previsti canali di ingresso dedicati e separati.
L'uso di tecnologie a Container ha permesso di realizzare un'architettura a Microservizi, con un elevato grado di scalabilità e flessibilità, e tale da permettere una facile integrazione tra differenti componenti software o la loro immediata sostituzione, qualora una o più delle tecniche presentate venga superata o se ne voglia fornire una implementazione differente.
Le diverse metodologie esaminate all'interno del presente rapporto sono state valutate così da poterne apprezzare efficacia ed efficienza; tuttavia ulteriori esperimenti sono necessari per poter confrontare le soluzioni adottate con altre, sia già introdotte nel presente deliverable, sia completamente nuove e non affrontate in esso.
L'uso di una rete neurale con strato di Word Embedding, affrontato come alternativo all'uso di librerie predefinite di Natural Language Processing per la classificazione binaria di pertinenza delle News, è un possibile fronte di ulteriore ricerca lungo cui muoversi,

considerando la necessità di testare ulteriormente la soluzione alternativa individuata, ma soprattutto tenendo in considerazione il consumo di risorse che tale tecnica comporterebbe. Il Word Embedding è stato esaminato anche nell'ambito della Ontology Population, e anche in questo caso sarebbe utile un approfondimento della tecnica, con test più approfonditi e corpus documentali corposi.
In tema Riconoscimento Luoghi/Date/Eveti una futura implementazione potrebbe essere la possibilità di poter estendere la ricerca degli eventi ad un contesto più generale. In particolare questo sarebbe possibile a partire dalla visualizzazione tramite Dashboard menzionata nel paragrafo 5.7.5. Lasciando invariato il front-end ma modificando in maniera sostanziale il back-end. Il tool non andrebbe quindi a verificare l'esistenza di eventi salvati all'interno dell'ontologia, ma a ricercarli esternamente.
In tema di Visualizzazione, sono state applicate diverse tecniche, da BRAT a Displacy, che permettono di osservare i dati recuperati dai testi originali con particolari etichettature e annotazioni. Poiché non tutte le viste mostrate sono state implementate ma, come già descritto nella sezione dedicata, alcune di esse sono state progettate ma non implementate e testate completamente, parte del lavoro futuro sarà concentrata sulla effettiva realizzazione di questi meccanismi di visualizzazione, nonché la loro estensione ed integrazione.

8 Riferimenti sitografici

[1] Google News: https://news.google.it/
[2] News API https://newsapi.org/
[3] BeautifulSoup: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
[4] Spacy: https://spacy.io/
[5] NLTK: https://www.nltk.org/
[6] Owlready2: https://owlready2.readthedocs.io/en/v0.35/
[7] Displacy: https://spacy.io/usage/visualizers
[8] Displacy ent: https://github.com/explosion/displacy-ent
[9] NER di Displacy ent: https://explosion.ai/demos/displacy-ent
[10] Brat Annotation Tool: https://brat.nlplab.org/
[11] Modulo FastText di Gensim: https://radimrehurek.com/gensim/models/fasttext.html
[12] Regex Python: https://docs.python.org/3/library/re.html
[13] Cosine Similarity: https://www.machinelearningplus.com/nlp/cosine-similarity/
[14] PCA: https://setosa.io/ev/principal-component-analysis/
[15]Bert-Model:https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment

9 Riferimenti bibliografici

Di Martino, B., Colucci Cante, L., Graziano, M., & Enrich Sard, R. (2020, giugno 11). Tweets Analysis with Big Data Technology and Machine Learning to Evaluate Smart and Sustainable Urban Mobility Actions in Barcelona. *CISIS 2020: Complex, Intelligent and Software Intensive Systems*.

Di Martino, B., Esposito, A., Marulli, F., Colucci Cante, L., Graziano, M., D'angelo, S., Lupi, P., & Cataldi, A. (n.d.). A Big Data Pipeline and Machine Learning for a Uniform Semantic Representation of structured Data and Documents from Information Systems of Italian Ministry of Justice. *International Journal of Grid and High Performance Computing*.

Di Martino, B., & Graziano, M. (2021, Maggio 12). Semantic techniques for discovering architectural patterns in building information models. *International Journal of Computational Science and Engineering*.

Di Martino, B., Graziano, M., & Cerullo, N. (2021, Giugno). Semantic Techniques for Automated Recognition of Building Types in Cultural Heritage Domain. *Complex, Intelligent and Software Intensive Systems*.

Di Martino, B., Marulli, F., Graziano, M., & Lupi, P. (2021, giugno 30). PrettyTags: An Open-Source Tool for Easy and Customizable Textual MultiLevel Semantic Annotations. *Complex, Intelligent and Software Intensive Systems*.

Lamy, J.-B. (2017, luglio). Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. *Artificial Intelligence in Medicine*.

TreeTagger - a part-of-speech tagger for many languages. (n.d.). TreeTagger - a part-of-speech tagger for many languages. Retrieved January 17, 2022, from https://www.cis.lmu.de/~schmid/tools/TreeTagger/

Di Martino, B., Cascone, D., Colucci Cante, L., & Esposito, A. (2021, July). Semantic Representation and Rule Based Patterns Discovery and Verification in eProcurement Business Processes for eGovernment. In *Conference on Complex, Intelligent, and Software Intensive Systems* (pp. 667-676). Springer, Cham.

Di Martino, B., Branco, D., Cante, L. C., Venticinque, S., Scholten, R., & Bosma, B. (2021). Semantic and knowledge based support to business model evaluation to stimulate green behaviour of electric vehicles' drivers and energy prosumers. *Journal of Ambient Intelligence and Humanized Computing*, 1-23.

Di Martino, B., Esposito, A., & Colucci Cante, L. (2021). Multi agents simulation of justice trials to support control management and reduction of civil trials duration. *Journal of Ambient Intelligence and Humanized Computing*, 1-13.
Di Martino, B., Cante, L. C., & Venticinque, S. (2020, July). An ontology framework for evaluating e-mobility innovation. In *Conference on Complex, Intelligent, and Software Intensive Systems* (pp. 520-529). Springer, Cham.

Martino, B. D., Cante, L. C., Esposito, A., Lupi, P., & Orlando, M. (2021). Temporal outlier analysis of online civil trial cases based on graph and process mining techniques. *International Journal of Big Data Intelligence*, *8*(1), 31-46.

Appendice

Beniamino Di Martino è Professore Ordinario presso l'Università della Campania "Luigi Vanvitelli" (già Seconda Università di Napoli), per il settore Scientifico Disciplinare INGINF/05 - Sistemi per l'Elaborazione dell' Informazione. E' Adjunct Professor presso la Asia University – Taiwan. E'/è stato Delegato del Rettore per il Coordinamento e Potenziamento delle Reti di Ateneo, per l' Informatica, per il CINECA e per il GARR. E' stato vice-Direttore del Dipartimento di Ingegneria Industriale e dell' Informazione. Dal 1995 al 1998 e' stato Ricercatore presso l' Institute for Software Technology and Parallel Systems dell'Università' di Vienna (Austria). Dal 1998 al 2002 è stato Ricercatore Universitario, e dal 2002 al 2005 Professore Associato, presso la Seconda Università di Napoli. Ha pubblicato 14 testi scientifici a diffusione internazionale ed oltre 300 pubblicazioni scientifiche a diffusione internazionale, di cui più di 80 articoli su riviste internazionali. Ha partecipato ed è stato responsabile scientifico a numerosi progetti di ricerca internazionali (EC Esprit, EC CEI-Pact, EC TMR, Austrian FWF), nazionali (PRIN, FAR) e regionali. E' stato Project Coordinator del Progetto Europeo (FP7-ICT) ***mOSAIC*** (su Cloud Computing), e Responsabile di Unità per i Progetti Europei FP7-SMARTCITIES ***CoSSMIC*** (su Smart Energy Grids) e JU-ARTEMIS ***Crystal*** (su sistemi software affidabili). E' stato responsabile scientifico di un progetto nazionale CNR (Agenzia 2000), di un progetto MUR FAR (Laboratori Pubblico-privati) e dell'unità di ricerca della Seconda Università di Napoli per i progetti europei e Nazionali: IST (FP5-ICT) Working Group APART, Thematic Network (FP5-ICT) OntoWEB, PRIN-MUR Cloud@Home. E' attualmente WorkPackage leader dei Progetti Europei H2020-ICT **Toreador**, su Big Data, ed H2020-MG **GreenCharge** su Smart, Green and Integrated Transport. E' Editor o Associate Editor di 7 riviste scientifiche internazionali (tra cui IEEE Transactions on Cloud Computing – TCC - ed IEEE Transactions on Parallel and Distributed Systems - TPDS), membro in numerosi editorial board di riviste internazionali e guest editor per numerose riviste internazionali. E' stato general e program chair di numerosi congressi internazionali, steering committee member di congressi internazionali, membro di numerosi comitati di programma di congressi internazionali. E' Vice Chair dell' IEEE Technical Committee on Scalable Computing (IEEE-TCSC); è stato membro dell' Executive Board of the IEEE CS Technical Committee on Supercomputing Applications (IEEE-TCSA). E' membro dell' IEEE

Standardization Working group su Cloud Interoperability, degli IEEE Technical Committees on Scalable Computing (TCSC), on Big Data (TCBD), on Data Engineering (TCDE), on Semantic Computing (TCSEM), on Services Computing (TCSVC), on Intelligent Informatics (TCII), on Pattern Analysis and Machine Intelligence (TCPAMI), on Software Engineering (TCSE), on Distributed Processing (TCDP), on Parallel Processing (TCPP), on Cloud Computing (TCCLD), del Cloud Standards Customer Council, dell' OMG – Cloud Working Group, del Future Cloud Experts' Group della Commissione Europea – Internet of Services, Software and Virtualization Unit, dell' Innovation Advisory Board (IAB) of VECMA H2020 EC Project, dell' Advisory Board of OntoChain H2020 EC Project, del Comitato di Indirizzo della Federazione IDEM (IDEntity Management). E' Chair del Nomination Committee for the ``IEEE TCSC Award of Excellence in Scalable Computing'', membro del Nomination Committee for the ``IEEE TCSC Award for Early and Medium Career Researchers", e membro dell' Award Committee dell' IEEE Technical Committee on Cloud Computing. E' revisore di progetti per la Commissione Europea, per i programmi ICT, ICT-PSP ed eInfrastructures, e per l' European Research Council (ERC). E' revisore di progetti scientifici per i Ministeri della Ricerca del Belgio, del Lussemburgo e del Cile. E' revisore di progetti industriali in ambito ICT per i Ministeri dell' Università e della Ricerca e dello Sviluppo Economico, per la Regione Piemonte, per la Regione Campania, per la Regione Calabria, per la Regione Lazio, per la Regione Puglia, per la Regione Toscana, per la Regione Sardegna e per numerosi enti pubblici locali. E' membro del Comitato "Agenda Digitale" ed è stato membro del "Comitato Tecnico per la predisposizione ed implementazione del Piano Strategico della Società dell'Informazione" della Regione Campania. E' stato membro di Commissione per l'abilitazione a ruoli di Senior Researcher e Professore Associato per l' Universita' di Cork (IR) e di Lille (Fr). E' stato valutatore di esami finali per il PhD per le Università di Oxford, Cipro (Cy), Sidney (Au), Vienna (A), La Laguna (Sp), Genova, Calabria, Politecnico di Torino, Roma Tor Vergata, Roma Tre, Pavia, Pisa, Napoli2. E' membro del Consiglio di Dottorato del Gran Sasso Science Institute. E' membro del Consiglio Direttivo del Consorzio Interuniversitario Nazionale per l' Informatica – CINI, membro del Consiglio Consortile del Consorzio Interuniversitario CINECA, e membro del Consiglio Scientifico del Consorzio Interuniversitario RIMIC; è stato membro del CdA del Consorzio Interuniversitario per l' Università a distanza NETTUNO. E' Direttore dei Nodi locali dei Laboratori Nazionali CINI "Artificial Intelligence and Intelligent Systems", "Big Data", CyberSecurity" and "Smart Cities and Communities". E' Responsabile dell'Unità della Università della Campania "Luigi Vanvitelli" del Gruppo Nazionale di Ingegneria Informatica - G.I.I. E' membro del Comitato di Indirizzo dell' Iniziativa Nazionale IDEM – Identity Management, e Responsabile IDEM per l' Ateneo. E' stato vice-Direttore del Dipartimento di Ingegneria Industriale e dell'Informazione dell' Università della Campania. Ha svolto attività di ricerca sui seguenti temi: Modelli, paradigmi, linguaggi ed ambienti di programmazione ed esecuzione avanzati per sistemi informatici Distribuiti, Cloud, BigData, Edge ed IoT, Tecniche e strumenti di Software Engineering e Program Comprehension, Semantic Web e Semantic Web Services, Natural Language Processing, Big Data Analytics, Deep Learning, Ingegneria della Conoscenza, Intelligenza Artificiale e Business Intelligence.

Antonio Esposito è attualmente Ricercatore presso il Dipartimento di Ingegneria dell'Università della Campania "Luigi Vanvitelli". La sua tesi di dottorato si è concentrata sul riconoscimento e l'applicazione di Design e Cloud Patterns allo sviluppo di Software in ambiente Cloud, con il supporto di Tecnologie Semantiche. È stato coinvolto nel progetto FP7-ICT finanziato dall'UE mOSAIC e nel progetto Horizon 2020 Toreador, ed è attualmente coinvolto nel progetto Horizon 2020 GreenCharge e nel progetto di ricerca applicata "Big data Giustizia e Datawarehouse" promosso dal Ministero italiano di Giustizia nell'ambito del Consorzio Interuniversitario Nazionale per l'Informatica (CINI). È inoltre responsabile del progetto Tenacious, finanziato nell'ambito del progetto cascata Horizon 2020 Ontochain. I suoi principali interessi sono l'ingegneria del software, il Cloud computing, i Design e i Cloud Pattern, e le tecnologie per il Web Semantico.

Mariangela Graziano è attualmente assegnista di ricerca presso il Dipartimento di Ingegneria dell'Università della Campania "Luigi Vanvitelli". Ha conseguito la laurea magistrale in ingegneria informatica nel 2020 con una tesi nell'ambito di Knowledge Engineering and Big Data Intelligence incentrata sull'applicazione di tecniche di rappresentazione semantica e inferenza logica applicate al domino BIM - Building Information Modeling. E' stata coinvolta nel progetto Horizon 2020 GreenCharge su Smart, Green and Integrated Transport. Attualmente partecipa al progetto di ricerca "Big data Giustizia e Datawarehouse", promosso dal Ministero Italiano di Giustizia nell'ambito del Consorzio Interuniversitario Nazionale per l'Informatica (CINI). I suoi principali interessi riguardano l'ingegneria del software, le tecnologie per il semantic web e il natural language processing.

Luigi Colucci Cante è attualmente Assegnista di Ricerca presso il Dipartimento di Ingegneria dell'Università della Campania "Luigi Vanvitelli". Egli si è laureato in Ingegneria informatica nel 2020 presso l'Università degli studi della Campania “Luigi Vanvitelli” con una tesi di laurea nell'ambito di Knowledge Engineering and Big Data Intelligence incentrata sull'applicazione di Tecniche di Process Mining e Simulazione Multi-Agente al Processo Civile Telematico Italiano. Attualmente egli è anche Research Assistant nel progetto ”Big data Giustizia e Datawarehouse” promosso dal Ministero della Giustizia Italiano e realizzato dal Consorzio Interuniversitario Nazionale per l’Informatica (CINI). Ha partecipato al progetto Horizon 2020 GreenCharge su Smart, Green and Integrated Transport.I suoi principali interessi di ricerca riguardano l'Ingegneria del Software, il Process Mining, e le Tecnologie Multi-Agente e Semantiche applicate nel campo dell’intelligenza Artificiale.

Gennaro Junior Pezzullo è vincitore del dottorato Nazionale in Intelligenza Artificiale (Area Salute e Scienze della Vita) presso l' Università “Campus Bio-Medico” di Roma, attività che sta attualmente svolgendo. Ha conseguito la laurea magistrale in Ingegneria Informatica presso l’ Università della Campania "Luigi Vanvitelli” nel 2021. Vincitore di due borse di Studio Erasmus:
Nel 2018, durante il percorso di laurea triennale, per un periodo di oltre 6 mesi, ha frequentato e superato diversi esami presso “Universidad de Málaga”, Malaga in Spagna.
Nel 2021, durante il percorso di laurea magistrale, per un periodo di oltre 3 mesi ha frequentato, superato esami e prodotto una parte del lavoro di tesi presso “AGH University of Science and Technology”, Cracovia in Polonia
Nel 2020 ha ottenuto la certificazione dei 24 CFU per l’insegnamento presso l’ Università della Campania "Luigi Vanvitelli” e nello stesso anno è stato assunto come docente di due corsi PON di informatica presso il liceo “Francesco Durante” per un periodo di 7 mesi.

Vincenzo Bombace è attualmente studente magistrale di Ingegneria Informatica dell'Università della Campania "Luigi Vanvitelli" e impegnato a svolgere una borsa di ricerca in ambito "Tecniche di Machine Learning ,Big Data Analytics e Natural Language Processing con applicazione all’analisi di Social Media”. Ha conseguito la laurea triennale in Ingegneria Informatica nel 2021 con una tesi nell’ambito di Ingegneria del Software incentrata sull’applicazione di Tecniche di Natural Language Processing ed Architetture a Containers per l’Analisi di Tweets.