UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

Dipartimento di Informatica

Corso di Laurea in Informatica

Tesi di laurea in
Basi di Dati

Generazione di un Knowledge Graph in ambito giuridico per i casi di violenza sulle donne

Relatrice:
Claudia d'Amato

Laureando:
Giuseppe Rubini

Anno Accademico 2022/2023

Sommario

Tra le problematiche attuali, una delle più gravi è rappresentata dalla persistente presenza della violenza sulle donne, manifestata in vari contesti e modi, ma presente in tutte le società.

La presente tesi si propone di affrontare questa complessa problematica sviluppando un processo mirato alla creazione di un knowledge graph, il quale rispetti i principi di Linked Open Data. Tale processo avrà come punto di partenza le pronunce emanate dalla Corte Europea dei Diritti dell'Uomo (ECHR), reperibili nel sito istituzionale https://hudoc.echr.coe.int/, riguardanti specificamente casi di violenza sulle donne.

L'obiettivo principale consiste nel costruire un knowledge graph che, seguendo il metodo proposto, funga da rappresentazione strutturata e interconnessa di dati. Inoltre, applicando i principi dei LOD, si può fornire una conoscenza in formato aperto e liberamente interrogabile.

La giustizia predittiva è un campo emergente dell'intelligenza artificiale, che si pone come obiettivo di suggerire esiti a casi legali. Proprio in questo ambito, il knowledge graph creato e reso pubblicamente disponibile, potrebbe costituire una preziosa risorsa per future applicazioni, offrendo un quadro informativo dettagliato e interrelato sui casi trattati dalla Corte.

In questo modo, la ricerca si propone di contribuire a una migliore comprensione dei fenomeni di violenza di genere e fornire supporto per iniziative orientate alla prevenzione e alla giustizia.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

La violenza sulle donne è un problema globale che persiste in tutte le società, affermando la sua presenza come una delle sfide più urgenti dell'era moderna. Questa forma di violenza, perpetrata in vari contesti, fisici ed emotivi, ha un impatto devastante sulle vite delle donne, compromettendo la loro salute fisica e mentale, la loro sicurezza e la loro dignità. La violenza di genere rappresenta una chiara violazione dei diritti umani e una manifestazione inaccettabile dell'ineguaglianza tra i sessi.

1.1 Contesto

Questa tesi si sviluppa nel contesto del progetto Horizon Europe Seeds dell'Università degli Studi di Bari Aldo Moro, con la finalità di creare una collezione di dati e soluzioni di intelligenza artificiale nel contesto della giustizia predittiva, nello specifico per i casi di violenza sulle donne. La collezione di dati è generata partendo dal contenuto di sentenze e atti normativi a livello europeo, condivisa in un formato aperto e pubblicamente accessibile.

Con giustizia predittiva si intende un campo emergente dell'intelligenza artificiale che ha lo scopo di suggerire esiti a casi legali. Per fare questo, però, è necessaria una grande mole di dati riguardanti casi passati affinché i modelli di machine learning possano effettuare delle previsioni e, soprattutto, farlo con maggiore precisione.

1.2 Motivazioni

L'obiettivo di questa tesi è quello di definire un processo, oltre che fornire un'implementazione, che permetta di creare un knowledge graph che rispetti i principi di Linked Open Data, partendo dalle pronunce emesse dalla Corte Europea dei Di-

ritti dell'Uomo (abbreviato in inglese con ECHR) contenute nel sito istituzionale https://hudoc.echr.coe.int/, le quali trattano casi di violenza sulle donne.

La creazione di LOD sotto forma di KG ha il vantaggio di fornire dati in formato aperto e liberamente interrogabili, con la possibilità di interconnessione con altri dati esistenti o aggiunti successivamente, offrendo agli utenti una base per lo sviluppo di applicazioni.

1.3 Organizzazione della tesi

I capitoli sono organizzati come segue. Nel capitolo 2 sono fornite le definizioni fondamentali utili alla comprensione della tesi. Prima di tutto è definito il concetto di knowledge graph e di ontologia, in quanto rappresenta la base del lavoro svolto. Successivamente, si introducono gli standard utilizzati nel web semantico per la creazione di risorse. Infine, si delinea il significato di linked data e LOD, illustrando i relativi principi utilizzati per la pubblicazione di risorse sul web semantico.

Il capitolo 3 è dedicato all'analisi dello stato dell'arte in merito alle ontologie giuridiche e allo sviluppo di knowledge graph, soprattutto nel dominio preso in esame, valutandone punti di forza e di debolezza.

Il capitolo 4 descrive l'approccio proposto per creare un KG partendo dalle pronunce della Corte e, successivamente, presenta l'applicazione dello stesso ad un insieme scelto di sentenze.

Nel capitolo 5 si valuta quanto sviluppato in questa tesi e del suo impatto nel dominio preso in considerazione.

Infine, nel capitolo 6 sono descritte le sfide ancora aperte, oggetto di futuri sviluppi.

Capitolo 2

Definizioni preliminari

In questo capitolo sono fornite le definizioni alla base per comprendere questa tesi. Prima di tutto, nella sezione 2.1, è definito il modello di rappresentazione della conoscenza utilizzato in questo lavoro, ovvero i knowledge graph. Successivamente, nella sezione 2.2, si parla di ontologie, ovvero la rappresentazione formale dei termini utilizzati, nel caso specifico, all'interno del knowledge graph. Sia i knowledge graph che le ontologie sono alla base del web semantico, estensione del web utile per creare e condividere conoscenza, concetto introdotto nella sezione 2.3, in cui si definiscono anche i linguaggi di rappresentazione standard nel web semantico. Quando i dati presenti nel web semantico sono interconnessi si parla di linked data o di linked open data, a seconda se siano pubblicamente disponibili o meno, presentati nella sezione 2.4 assieme ai principi su cui si basano.

2.1 Knowledge Graph

Un knowledge graph è un modello di rappresentazione della conoscenza, utilizzato per definire le relazioni tra diverse entità.

Formalmente un knowledge graph è una tripla $\mathbf{G} = (V, E, L)$ dove V è l'insieme dei nodi, L è l'insieme delle etichette utilizzate per gli archi e $\mathbf{E} \subseteq V \times L \times V$ è l'insieme degli archi.[17]. In altre parole, un KG è formato da un insieme di nodi collegati tra loro da archi orientati ed etichettati con un certo valore. In particolare i nodi e gli archi hanno un significato specifico:

- i nodi rappresentano delle entità del mondo che si vuole rappresentare;
- gli archi rappresentano una relazione binaria tra le entità che collega.

Nella figura 1 è rappresentato un esempio di knowledge graph che mostra le relazioni tra il concetto DBpedia limone, il suo genere e la sua famiglia dal punto di vista

biologico, i quali sono rappresentati come nodi. Prendendo in considerazione gli archi che coinvolgono *dbr:Lemon*, è descritto che ha come genere *dbr:Citrus* e come famiglia *dbr:Rutaceae*. Ognuno di queste entità e relazioni rappresentano delle risorse presenti su DBpedia. Inoltre, *dbr:Lemon* ha due attributi in cui i valori sono definiti da valori letterali, ovvero *dbo:calciumMg* e *rdfs:label*: al primo è assegnato un valore intero, al secondo una stringa seguita dall'identificatore della lingua. In modo simile *dbr:Citrus* appartiene alla famiglia *dbr:Rutaceae* oltre che *dbr:Aurantioideae*, e ne viene definita una etichetta in lingua inglese. Di un KG è possibile definire uno schema, ovvero la

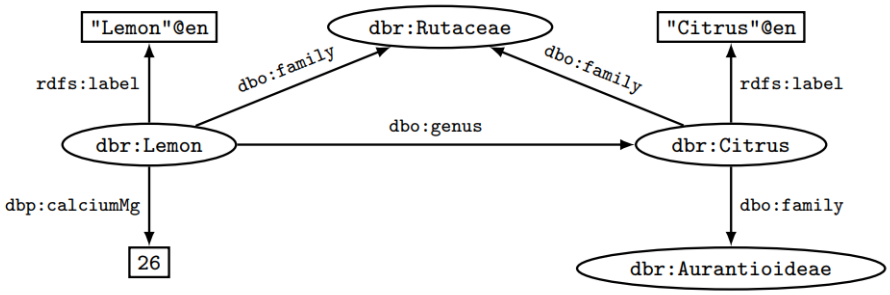

Figura 1: Esempio di un knowledge graph[16]

rappresentazione formale della struttura e delle relazioni all'interno del knowledge graph stesso. Per fare ciò si definiscono le ontologie.

2.2 Ontologia

In informatica un'ontologia è una concettualizzazione formale del significato dei termini utilizzati in un particolare dominio[17][21][23], con lo scopo di definire un insieme di primitive, quali classi, attributi o relazioni, con cui modellare un dominio. Delle primitive si specificano le informazioni sul loro significato oltre ai vincoli sulla loro applicazione[20].

Nel web semantico le ontologie sono utilizzate per creare vocabolari concettuali standard in modo tale da garantire l'interoperabilità di più sistemi[21][20]. Se sistemi diversi utilizzano un modo ben definito per rappresentare la conoscenza, questi possono interpretare e utilizzare le informazioni scambiate senza ambiguità.

2.3 Linguaggi di rappresentazione standard

Con web semantico si intende un'estensione del web utile per creare e condividere contenuti facilmente leggibili dalle macchine [16]. Con la nascita del web semantico si è assistiti anche allo sviluppo dei linguaggi di rappresentazione della conoscenza che, successivamente, sono stati standardizzati dal W3C, portando all'interoperabilità.

Nelle sezioni successive sono introdotti gli standard utilizzati per rappresentare le informazioni, per la definizione degli schemi, per la creazione di ontologie e per l'interrogazione.

2.3.1 Resource Description Framework (RDF)

Lo standard RDF definisce il framework per la rappresentazione delle informazioni utilizzato nel web semantico, basato sul modello dei dati a grafo[25]. Alla base di RDF ci sono i **termini** i quali possono fare riferimento a delle risorse. Questi possono essere[16][25]:

- **URI (Uniform Resource Identifier)**: identifica una risorsa presente sul web;
- **letterali**: valori lessicali, come una stringa, i quali possono avere un tipo di dato associato;
- **blank node**: definiscono l'esistenza di una risorsa senza fare riferimento a URI o letterali.

Partendo dai termini si possono creare delle triple per rappresentare la conoscenza: il primo elemento è il soggetto, il secondo è il predicato, il terzo è il soggetto.

Le triple devono rispettare delle regole relative al tipo di termine in base alla posizione[16]:

- **soggetto**: può essere un URI o un blank node. Definisce la risorsa descritta dalla tripla;

- **predicato**: deve essere un URI. Identifica la relazione che intercorre tra soggetto e oggetto;
- **oggetto**: può essere un qualsiasi tipo di termine. Rappresenta il valore della relazione.

Un insieme di triple RDF è rappresentabile come un grafo in cui ogni soggetto e ogni oggetto è rappresentato da un nodo etichettato con il relativo valore. Gli archi sono orientati e collegano i nodi, da soggetto a oggetto, secondo quanto indicato dalla tripla, ed etichettati con il valore del predicato.[16]

Esistono diverse sintassi per serializzare i dati RDF in un file, ma quella utilizzata in questa tesi è stata Turtle[6], in quanto mette a disposizione delle abbreviazioni che rendono il contenuto più human-readable. Segue un esempio di triple scritte in Turtle, che descrivono il grafo rappresentato nella figura 1[16]:

```
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix dbo: <http://dbpedia.org/ontology/> .
@prefix dbp: <http://dbpedia.org/property/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

dbr:Lemon    rdfs:label        "Lemon"@en ;
             dbp:calciumMg     26 ;
             dbo:family        dbr:Rutaceae ;
             dbo:genus         dbr:Citrus .

dbr:Citrus   rdfs:label        "Citrus"@en ;
             dbo:family        dbr:Rutaceae ,
                               dbr:Aurantioideae .
```

Nella prima parte si può vedere una prima abbreviazione messa a disposizione dalla sintassi, ovvero la definizione dei prefissi, i quali permettono di associare dei namespace a delle stringhe più brevi, le quali possono essere richiamate successivamente. Successivamente si trova la definizione delle triple. Si nota come *Lemon* e *Citrus*

sono soggetti di diverse triple e che questa sintassi permette di evitare la ripetizione dei soggetti in comune utilizzando il punto e virgola alla fine della tripla. Una cosa simile è presente nell'ultima riga, in cui si utilizza una virgola per segnalare che gli oggetti hanno in comune sia il soggetto che il predicato.

2.3.2 RDF Schema (RDFS)

Il vantaggio principale dell'utilizzo dei grafi come modello dei dati rispetto a modelli più diffusi come quello relazionale è la su flessibilità nella definizione dello schema.

Lo standard offerto dal W3C per la definizione di uno schema semantico per i grafi RDF è RDFS [4] il quale permette di creare delle gerarchie di classi e proprietà utilizzando, rispettivamente, i termini *subClassOf* e *subPropertyOf*. La prima è una relazione che collega due classi, indicando che la classe di destinazione è una sotto-classe della classe di origine. Similmente la seconda indica che la proprietà di destinazione è sotto-proprietà di quella di origine. Questo fornisce un modo per rappresentare concetti più specifici o specializzati in relazione a concetti più generali o astratti, contribuendo ad organizzare e a dare significato ai dati semantici nell'ambito del Web semantico.

Inoltre, RDF permette di definire *dominio* e *range* di una proprietà: con dominio si intende la classe di appartenenza del soggetto, con range (o codominio), invece, quella dell'oggetto. Se una classe è specificata come dominio di una proprietà, ciò implica che tutte le sue sottoclassi possono anch'esse essere utilizzate come soggetti per quella proprietà. Stessa cosa vale per il codominio e gli oggetti della proprietà. In altre parole, dominio e range si applicano alla classe specificata e a tutte le sue sottoclassi nella gerarchia.

L'utilizzo di questi costrutti permette di ottenere degli schemi con della semantica, a differenza di RDF che permette di rappresentare solo fatti.

2.3.3 Web Ontology Language (OWL)

OWL[3] è un'ulteriore estensione ai linguaggi precedenti, in quanto è un linguaggio di rappresentazione della conoscenza utilizzato per descrivere ontologie, cioè rappre-

sentazioni formali di concetti, relazioni e regole all'interno di un dominio specifico. Le ontologie OWL consentono di definire in modo chiaro e strutturato[16]:

- **classi**: definizione di classi di oggetti o concetti all'interno di un dominio;
- **proprietà**: definizione di relazioni tra classi di oggetti, consentendo di specificare legami tra concetti;
- **individui**: creazione di istanze specifiche di classi o concetti.

A differenza di RDFS offre una maggiore espressività per la definizione delle relazioni tra concetti. Per fare ciò sono introdotti i seguenti costrutti[24][16]:

- **equivalentClass**: permette di definire l'equivalenza tra classi, rappresentando due classi diverse che sono sinonimi tra loro, quindi che le istanze della prima lo sono anche della seconda e viceversa;
- **disjointWith**: permette di definire che due classi hanno intersezione vuota, in modo tale da trovare inconsistenza se esiste un individuo che appartiene alle due classi contemporaneamente;
- **equivalentProperty**: permette di definire l'equivalenza tra proprietà, rappresentando proprietà sinonime;
- **inverseOf**: permette di definire che due proprietà sono una l'inversa dell'altra
- **TransitiveProperty**: permette di definire la transitività di una proprietà. Se una proprietà P è transitiva, e abbiamo le coppie (x,y) e (y,z), istanze di P, allora anche la coppia (x,z) è un'istanza di P;
- **SymmetricProperty**: permette di definire una proprietà simmetrica, quindi bidirezionale. Se P è una proprietà simmetrica e la coppia (x,y) è un'istanza di P, allora anche (y,x) è una istanza di P;
- **sameAs**: permette di definire che due individui sono la stessa cosa, permettendo di creare risorse diverse che si riferiscono ad uno stesso individuo;

- **differentFrom**: permette di definire che due individui sono obbligatoriamente diversi;
- **FunctionalProperty**: permette di indicare che la cardinalità massima della proprietà è uno;
- **InverseFunctionalProperty**:come il precedente ma indica che anche la proprietà inversa ha uno come cardinalità massima.

Creare un'ontologia OWL e sfruttare dei reasoner, strumenti software progettati per eseguire il ragionamento, permette di eseguire diverse procedure di ragionamento. Tra queste le principali sono[16]:

- **inferenza della classe**: inferire che un'istanza o entità è di un tipo specifico (classe asserita) e, allo stesso tempo, è anche di un tipo più generale (superclasse) all'interno della gerarchia delle classi;
- **instance checking**: verificare se un individuo soddisfa le condizioni specificate dalle classi a cui appartiene;
- **consistenct checking**: verificare se l'ontologia è consistente, quindi che non ci sono contraddizioni.

In conclusione, OWl ricopre un ruolo fondamentale nel web semantico grazie alla sua capacità di rappresentare ontologie in modo formale, e alla possibilità di effettuare ragionamento automatico. Ciò consente alle macchine di inferire nuove relazioni e comprendere il significato implicito nei dati, migliorando l'automazione e l'interpretazione semantica.

2.3.4 SPARQL

Per quanto riguarda l'interrogazione, il W3C fornisce come standard il linguaggio SPARQL[5] (SPARQL Protocol and RDF Query Language), il quale permette formulare query per filtrare, estrarre e combinare informazioni da grafi RDF, i quali

possono anche essere contenuti in fonti diverse. SPARQL è anche adatto per interrogare dati RDF che seguono schemi ontologici come RDFS e OWL. La query può contenere delle variabili, le quali sono precedute da un punto interrogativo seguito dall'identificativo scelto. Segue un esempio di query, la quale interroga il grafo rappresentato nella figura 1:

```
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbo: <http://dbpedia.org/ontology/>

FROM <http://dbpedia.org/data/Lemon.xml>

SELECT DISTINCT ?genus ?order

WHERE {
    dbr:Lemon dbo:genus ?genus ;
              dbo:order ?order .
}

LIMIT 2
```

Si può vedere come una query è suddivisa in cinque parti principali[16]:

1. **Dichiarazione di prefissi**: permette di definire prefissi per le URI, utilizzabili come abbreviazioni nella query. Per la definizione si usa la keyword *PREFIX* seguita dall'abbreviazione scelta, i due punti e l'URI interessato;
2. **Clausola del dataset**: specifica la porzione di dati da interrogare. È formata dalla parola chiave *FROM* seguita dall'URI del grafo di default a cui porre le query;
3. **Clausola di risultato**: definisce il tipo di query da eseguire e il tipo di risultato restituito. I tipi possibili sono quattro:
    - *SELECT*: ritorna la lista dei valori associati alle variabili. Se si aggiunge *DISTINCT* non ammette associazioni duplicati;

- *ASK*: ritorna un valore booleano riguardante l'esistenza di un'associazione per le variabili;
- *CONSTRUCT*: crea un grafo in cui le triple sono il risultato della query;
- *DESCRIBE*: ritorna una descrizione RDF di un particolare termine.

4. **Clausola di interrogazione**: è quasi sempre preceduta dalla parola chiave *WHERE* seguita da parentesi graffe al cui interno è descritto il pattern da rispettare, dando la possibilità di inserire delle variabili. Si possono anche porre delle condizioni che il risultato deve rispettare, utilizzando la parola chiave *FILTER* utilizzando, ad esempio, operatori logici per vincolare il valore di una variabile;

5. **Modificatore della soluzione**: in questa ultima parte si definiscono le clausole utilizzate per modificare il modo in cui i risultati di una query sono presentati o ordinati. In particolare troviamo le keyword:

   - *ORDER BY*: permette l'ordinamento lessicografico del risultato rispetto una variabile specificata. Questo può essere crescente (*ASC*) o decrescente (*DESC*);
   - *LIMIT*: seguito da un numero intero, specifica il numero massimo di risultati da ritornare;
   - *OFFSET*: simile al precedente ma definisce il numero di risultati da ignorare.

Per la risoluzione della query, SPARQL utilizza un meccanismo di pattern matching[5] [17]. Questo consiste nell'identificare corrispondenze tra pattern di triple specificati nella clausola di interrogazione e quelle effettivamente presenti nel grafo. Come visto nell'esempio precedente, i pattern possono includere variabili che fungono da segnaposto per i valori cercati.

Le ultime versioni di SPARQL permettono di operare sotto entailment regimes [28], abilitando l'uso di ragionatori per la risoluzione della query. In questo modo si può tener conto delle relazioni ontologiche definite nei grafi, consentendo di dedurre

ulteriori fatti o inferenze logiche durante il processo di interrogazione, ampliando le capacità del linguaggio.

2.4 Linked Data e Linked Open Data

Linked Data è un principio e un approccio di progettazione per la pubblicazione e l'interconnessione di dati sul web semantico in modo standardizzato e interoperabile. Il W3C definisce dei principi da rispettare per la creazione di risorse come linked data. Questi sono [2]:

1. usare URI come nomi delle cose;
2. usare URI HTTP in modo da dereferenziare quei nomi;
3. quando qualcuno cerca una URI, fornire informazioni utili, utilizzando gli standard (come RDF);
4. includere collegamenti ad altre URI in modo che si possano scoprire ulteriori informazioni.

Inoltre, se si tratta di Linked Open Data, quindi rilasciati sotto licenza libera per permetterne il riuso, sono state anche definiti ulteriori principi chiamati *"Linked Open Data 5 Star"*[2]:

1. dati disponibili nel web con licenza libera;
2. pubblicare dati strutturati, facilmente leggibili dalle macchine;
3. utilizzare formati non proprietari;
4. utilizzare gli standard W3C per identificare gli oggetti;
5. collegare i propri dati con dati di altri per dare contesto.

I LOD sono stati la prima forma di dati interconnessi e pubblicamente accessibili, standardizzati nel tempo in forma di knowledge graph. Quindi, per lo sviluppo di KG, è opportuno basarsi sui principi di LOD e utilizzare i linguaggi standard del web semantico.

Capitolo 3

Stato dell'arte

In questo capitolo viene introdotto dapprima il processo di sviluppo di knowledge graph[29], nella sezione 3.1. Successivamente nella sezione 3.2 sono descritte le ontologie giuridiche attualmente in uso nell'Unione Europea. Infine, nella sezione 3.3 vengono illustrate delle soluzioni che sviluppano dei knowledge graph in ambito giuridico[1][27][30].

3.1 Processo di sviluppo dei knowledge graph

Nell'articolo di Gytundefined Tamašauskaitundefined e Paul Groth[29] è definito in metodo generale per il processo di sviluppo dei knowledge graph. Come prima cosa vengono definite due metodologie di sviluppo:

- **Top-down**: si definisce un'ontologia e basandosi su questa si estrae la conoscenza dai dati;
- **Bottom-up**: si estrae la conoscenza dai dati e, successivamente, si crea un'ontologia che rappresenti i dati estratti.

Successivamente si definisce il processo di sviluppo, illustrato nella figura 2. Come

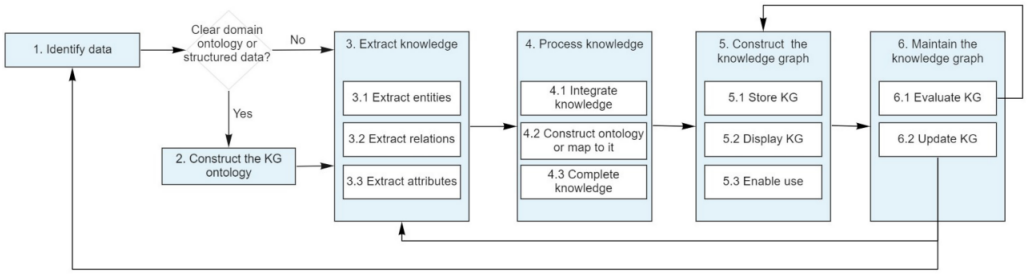

Figura 2: Processo di sviluppo di Tamašauskaitundefined e Groth[29]

si può vedere il metodo proposto si suddivide in sei passi:

1. **Identificazione dei dati**: in questo primo passo si definisce il dominio di interesse e si identificano le sorgenti da cui raccogliere i dati necessari. Come risultato di questa fase si saranno acquisiti i dati, che siano essi strutturati, semi-strutturati o non strutturati, da cui estrarre la conoscenza.

2. **Costruzione dell'ontologia**: *solo nel caso in cui si scelga una tipologia di sviluppo top-down*, si definiscono i tipi di entità e le relazioni che intercorrono tra esse usando delle ontologie di utilizzo comune come FOAF, ontologie esistenti rilevanti per il dominio di interesse, oppure definendole con linguaggi come RDFS o OWL.

3. **Estrazione della conoscenza**: partendo dai dati raccolti l'obiettivo di questo passo è quello di estrarre entità, relazioni e attributi. Questa fase è a sua volta suddivisa in tre parti:

   (a) *Estrazione delle entità*: si utilizzano metodi come il Named Entity Recognition il quale trova e classifica le entità in categorie predefinite;

   (b) *Estrazione delle relazioni*: dopo aver estratto le entità queste devono essere collegate tra loro con delle relazioni. Nel caso di dati non strutturati si utilizzano metodi come il Natural Language Processing. Inoltre, se è disponibile un'ontologia (quindi si sta seguendo un approccio top-down), si cercano le informazioni che la ricalcano;

   (c) *Estrazione degli attributi*: si cercano le informazioni che permettono di descrivere meglio le entità.

   Alla fine di questa fase è possibile costruire le triple utili per creare il knowledge graph.

4. **Processing della conoscenza**: si applicano dei metodi per garantire una buona qualità della conoscenza estratta, quali:

- *Integrazione*: eliminazione di ridondanza, contraddizione e ambiguità, sia per le entità che per le relazioni. Si valuta se diverse entità si riferiscono ad uno stesso oggetto del mondo reale, collegandole tra di loro. Verificare che tutte le entità siano identificate in modo univoco, ad esempio con un URI;
- *Costruzione ontologia*: *se si è seguito un approccio bottom-up*, e quindi non si ha ancora a disposizione l'ontologia, questo è la fase in cui si crea il modello della struttura del knowledge graph;
- *Completamento della conoscenza*: l'obiettivo di questa fase è di inferire nuova conoscenza, partendo da quanto è già rappresentato, e di ottimizzazione del grafo, ad esempio eliminando quelle parti superflue per il dominio di interesse.

5. **Costruzione del knowledge graph**: lo scopo di questa fase è quello di rendere il knowledge graph utilizzabile. Per fare questo prima di tutto deve essere memorizzato in un formato appropriato. Successivamente questo deve essere facilmente visualizzabile e interrogabile.

6. **Manutenzione**: si utilizza il feedback degli utenti per identificare problemi o mancanze del knowledge graph. Inoltre deve essere permesso l'aggiornamento del grafo, aggiungendo nuova conoscenza se ci sono nuovi dati da rappresentare, partendo dalle sorgenti già considerate o da nuove.

3.2 Ontologie giuridiche

In questa sezione sono illustrate le ontologie giuridiche messe a disposizione dall'Unione Europea: ELI[11][12] ed ECLI[8][9]. Successivamente viene anche introdotto il tesauro EuroVoc[13].

3.2.1 European Law Identifier (ELI)

ELI [14][11][12] è uno standard creato per identificare i documenti legislativi degli stati europei, fornendo un'ontologia e dei metadati. Questo garantisce un accesso,

uno scambio e un riutilizzo semplificato della legislazione per utenti esperti del dominio legale o ai cittadini, ponendosi come base per una rappresentazione delle Gazzette Ufficiali degli stati membri nel web semantico[10]. Per quanto riguarda l'ontologia si ha che:

- *eli:LegalResource* è una creazione intellettuale distinta come un atto legale, di uno specifico *eli:type_document*, ad esempio una direttiva, che è realizzato da un *eli:LegalExpression*;

- *eli:LegalExpression* ha un *eli:title* ed *eli:realizes* la versione base in una lingua particolare (*eli:language*) di un *eli:LegalResource*. È pubblicato in un *eli:Format* che è la rappresentazione fisica come HTML o PDF.

ELI segue i principi dell'ontologia Functional Requirements for Bibliographic Records, abbreviata FRBR, utilizzata per le pubblicazioni bibliografiche [12][14]: una risorsa legale è sottoclasse di *Work*, un'espressione legale è sottoclasse di *Expression* e il formato è sottoclasse di *Manifestation*.

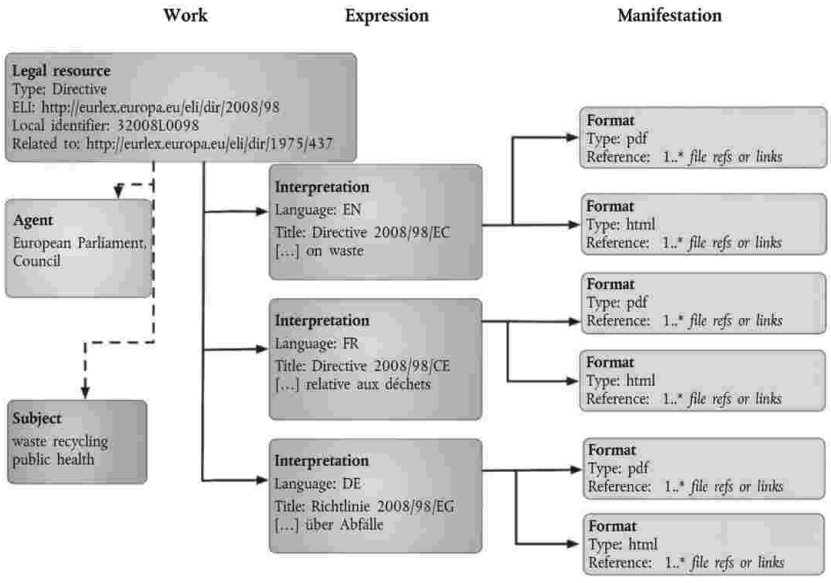

Figura 3: Schema dell'ontologia fornita da ELI[12]

3.2.2 European Case Law Identifier (ECLI)

ECLI[14][9][8] definisce un identificatore standard per la giurisprudenza europea, oltre ad un insieme minimo di metadati. L'identificatore fornito da ECLI ha lo scopo di avere un codice di identificazione la cui struttura è comune tra tutti gli stati membri. In particolare, questo è suddiviso in cinque parti separate dai due punti come nel seguente esempio:

ECLI:CE:ECHR:2022:0210JUD007397516

Ognuna delle sezioni ha un significato preciso:

- sigla ECLI, uguale per tutti i documenti;
- codice dello stato o dell'organizzazione internazionale in cui è stata emessa la decisione;
- codice della corte che ha preso la decisione;
- anno della decisione
- numero univoco ordinale della decisione

Per quanto riguarda i metadati vengono utilizzate le proprietà definite nell'ontologia Dublin Core Metadata Initiative[7], senza aggiungerne di nuove ma raccomandando l'uso delle seguenti:

- **dcterms:identifier**: URL in cui è possibile recuperare la risorsa
- **dcterms:isVersionOf**: indica che una risorsa è una versione di un'altra risorsa, usando un ECLI
- **dcterms:creator**: nome completo del tribunale decidente
- **dcterms:coverage**: indica il paese in cui ha sede la corte o il tribunale
- **dcterms:date**: la data in cui è stata emessa una decisione (nel formato ISO 8601[19])

- **dcterms:language**: la lingua in cui è scritto (abbreviata secondo ISO 639[18])
- **dcterms:publisher**: l'organizzazione responsabile della pubblicazione del documento
- **dcterms:accessRights**: definisce chi può accedere alla risorsa, quindi se è pubblica o privata
- **dcterms:type**: definisce il tipo di decisione presa

3.2.3 Relazione tra ELI e ECLI

Citando letteralmente il tredicesimo paragrafo delle *"Conclusioni del Consiglio che invitano all'introduzione dell'identificatore della legislazione europea (ELI)"*[12], contenuto nella Gazzetta ufficiale dell'Unione Europea si afferma che:

> *≪L'identificatore europeo della giurisprudenza (ECLI), applicabile su base volontaria, fornisce già un sistema europeo per l'identificazione della giurisprudenza. ELI identifica testi legislativi aventi caratteristiche diverse e più complesse, e i due sistemi sono complementari.≫*

Quindi, in generale, si deve preferire l'utilizzo di ECLI nel caso in cui si ha a che fare con decisioni giuridiche, mentre si deve scegliere di utilizzare ELI nel caso di testi legislativi, come ad esempio una legge.

3.2.4 EuroVoc

Il thesaurus EuroVoc[14][13] è un thesaurus multidominio e multilingue fornito dall'Ufficio delle pubblicazioni dell'Unione europea utilizzato per classificare i documenti dell'UE in categorie per facilitare la ricerca delle informazioni. Si basa sullo standard Simple Knowledge Organization System (SKOS)[26], utilizzato per rappresentare e organizzare concetti e vocabolari. In particolare:

- ogni termine di EuroVoc è di tipo *skos:Concept*
- più termini possono essere aggregati in un *skos:ConceptScheme*;

- i concetti sono collegati con *skos:narrower* e *skos:broader* per rappresentare la struttura gerarchica
- si usa *skos:related* per le relazioni associative;
- ogni concetto ha un termine preferito e altri alternativi, indicate con *skos:prefLabel* e *skos:altLabel*.

3.3 Knowledge graph in ambito giuridico

Negli anni recenti, le tecnologie legate al web semantico hanno introdotto un nuovo approccio alla condivisione delle informazioni anche nel campo legale. Nello specifico, nel settore giuridico, tali tecnologie potrebbero portare a una rivoluzione nella gestione delle informazioni legali[1].

In letteratura ci sono diversi sistemi per la creazione di knowledge graph, ma pochi lo fanno in ambito giuridico, e nessuno nello specifico per quanto riguarda casi di violenza sulle donne.

I sistemi presi in considerazione[1][27][30] utilizzano dati diversi da cui estrarre la conoscenza ma come si potrà notare, il processo utilizzato si può ricondurre a quanto descritto nella sezione 3.1.

3.3.1 KG da Istituto Poligrafico Zecca dello Stato

Nel sistema creato da Anelli et al.[1] si crea una pipeline per estrarre le informazioni dai documenti prodotti dall'Istituto Poligrafico Zecca dello Stato, rendendoli facilmente interrogabili. Il sistema è formato dai seguenti moduli:

1. **Law description extractor**: carica la banca dati dell'IPZS;
2. **Law description converter**: i dati vengono convertiti in triple, usando prefissi e predicati usati in linked open data;
3. **Triplestore**: carica le triple in un grafo RDF dando la possibilità di interrogarlo e aggiornarlo;

4. **Dereferencer**: permette di pubblicare i dati in formato RDF;
5. **Graph browser**: permette di navigare la struttura del grafo;
6. **SPARQL endpoint**: fornisce l'API e l'interfaccia per interrogare il grafo;
7. **Dashboard**: come il precedente ma mette a disposizione query preimpostate per chi non ha familiarità con i linguaggi di interrogazione.

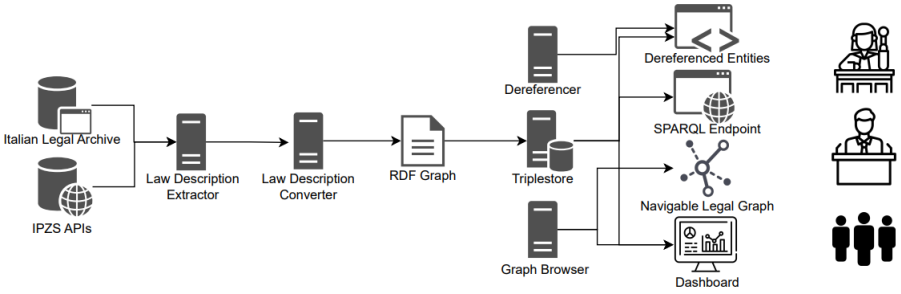

Figura 4: Architettura del sistema di Anelli et al.[1]

3.3.2 Legal KG-based QA

Il sistema creato da Sovrano et al.[27] ha lo scopo di rispondere a domande poste per cercare informazioni legate a specifici argomenti, concetti o entità. Per estrarre le informazioni contenute nei documenti si seguono i seguenti passi:

1. **KG extraction**: si estraggono concetti e relazioni dal testo, si assegnano URI e label RDF ai nodi. Si aggiungono anche triple speciali per tenere traccia dei frammenti di testo in cui sono stati estratti i concetti;
2. **Taxonomy construction**: si estrae la tassonomia di tipi/classi dei concetti rappresentati attraverso Formal Concept Analysis;
3. **Legal ODP alignment**: si allinea la struttura a design pattern di ontologie;
4. **Question answering**: dato una domanda in linguaggio naturale restituisce i risultati rilevanti.

3.3.3 Constructing a KG for Vietnamese legal cases

In questo ultimo sistema creato da Vuong et al.[30] si crea un knowledge graph eterogeneo partendo da leggi e casi legali vietnamiti.

Formalmente un knowledge graph eterogeneo è una quadrupla $\mathbf{G} = (V, E, L, l)$ dove V è l'insieme dei nodi, L è l'insieme delle etichette utilizzate per gli archi, $\mathbf{E} \subseteq V \times L \times V$ è l'insieme degli archi e $l : V \rightarrow L$ assegna ad ogni nodo un'etichetta[17]. In altre parole si ha che ogni nodo è associato un tipo specifico, definito dall'etichetta. I passi utilizzati per la creazione del KG sono i seguenti:

1. **Data crawling**: si ricercano leggi e casi legali su siti governativi;
2. **Information Extraction**: si estraggono le entità e le relazioni dai file raccolti;
3. **Knowledge graph deployment**: si crea il grafo, i cui nodi possono essere solo di tipo:
   - *case*: rappresenta una sentenza;
   - *domain*: contiene informazioni relative al tipo di decisione;
   - *court*: rappresenta la corte decidente;
   - *law*: contiene il nome di una legge.

   Anche le relazioni seguono uno schema predefinito:
   - *decide*: collega *court* e *case*, indica che una determinata corte si è occupata di un determinato caso;
   - *belongTo*: collega *case* e *domain*, indica il dominio del caso;
   - *basedOn*: collega *case* e *law*, indica quali leggi sono state prese in considerazione per prendere una decisione sul caso.

Nella figura 5 sono elencati gli attributi definiti ed utilizzati per ognuno dei tipi di entità che formano il grafo.

Entity	Attributes	Description
Case	case_id	id of the case
	case_number	number of the case (e.g 577/2022/HC-PT)
	document_type	type of the document (Verdict or Decision)
	case_level	level of the court (Trial, Appellate, and Cassation/Reopening)
	case_content	basic information of the case
	case_text	full content of the case
	date	documented and relevant dates
	court_id	id of the court
	domain_id	id of the case's domain
Domain	domain_id	id of the domain
	domain_name	type of the case (e.g Criminal, Civil, etc)
	subdomain	crimes, legal relations in the domain
Court	court_id	id of the court
	court_name	name of the court (e.g Hanoi Supreme People's Court)
	court_level	level of the court (e.g Provincial People's Court)
Law	law_id	id of the law
	law_name	name of the law (e.g Criminal Code, Civil Code)

Figura 5: Attributi per ogni tipo di entità, nel sistema di Vuong et al.[30]

3.3.4 Analisi dei KG in ambito giuridico

I tre sistemi descritti nelle precedenti sezioni utilizzano fonti normative differenti, nonostante lavorino su uno stesso dominio. Anelli et al. [1] trattano quanto contenuto nella Gazzetta Ufficiale della Repubblica Italiana, pubblicata dall'Istituto Poligrafico Zecca dello Stato, i quali possono trattare diverse tipologie di atti normativi ed amministrativi, non strettamente leggi[15]. Invece, Sovrano et al.[27] creano un knowledge graph partendo da dei regolamenti, nello specifico da: Regolamento CE Roma I n. 593/2008, Regolamento CE Roma II n. 864/2007 e Regolamento UE Bruxelles I bis n. 1215/2012. A differenza dei due precedenti i quali trattano docu-

menti validi all'interno dell'Unione Europea, nel sistema di Vuong et al.[30] si tratta la giurisprudenza e la legislazione in vigore in Vietnam.

Oltre al tipo di documento trattato, questi sistemi differiscono anche per le funzionalità offerte all'utente. Se nell'ultimo citato[30] l'ultimo passo è la creazione del KG, i primi due forniscono dei servizi ulteriori: nel primo[1] si permette di esplorare la struttura del KG oltre che di effettuare query preimpostate o personalizzate attraverso l'endpoint SPARQL; nel secondo[27] si fornisce la possibilità di restituire i risultati rilevanti a delle domande poste in linguaggio naturale.

Dal punto di vista dell'estrazione della conoscenza, quanto definito da Vuong et al.[30] presenta un'ulteriore differenza. Se nei sistemi presentati in [1] e [27] si estrae quanta più conoscenza possibile, in questo sono definite delle categorie precise di entità, di relazioni e di attributi associati alle entità (sintetizzati nella figura 5).

Quanto analizzato in questo capitolo fornisce una base di partenza per l'approccio proposto nel capitolo successivo. Alcuni degli step sono in comune, in quanto sono parte fondamentale della creazione di un knowledge graph. Tra questi ci sono la raccolta dei dati, l'estrazione della conoscenza e la generazione del KG stesso: ottenere il KG è l'obiettivo finale del lavoro e, per arrivare a questo punto, bisognerà estrarre la conoscenza presente nei dati, che prima però devono essere individuati.

A differenza dei sistemi presentati, nel capitolo successivo viene posta maggiore attenzione alla creazione dell'ontologia, seguendo un metodo di sviluppo di ontologie.

Capitolo 4

Approccio proposto

L'obiettivo di questo lavoro è quello di generare un knowledge graph per i casi di violenza sulle donne. Per fare questo è stata creata una metodologia bottom-up da seguire nello sviluppo. Il processo definito nella sezione 3.1 ha fornito, con opportune modifiche, il modello da seguire.

Per la generazione del knowledge graph, come primo passo è stato necessario individuare le pronunce dell'ECHR che trattano casi di violenza sulle donne. Quanto contenuto sul sito della Corte è analizzato nella sezione 4.1.

Una volta raccolti tutti i dati, segue la fase di estrazione della conoscenza, il cui risultato finale è quello di ottenere delle triple, seguendo gli standard del web semantico e i principi dei LOD, introdotti nelle sezioni 2.3 e 2.4.

Successivamente, viene posta enfasi sulla creazione dell'ontologia (concetto definito in 3.2), usando il metodo di sviluppo di ontologie basato su competency questions[22], estendendo quanto presente nell'ambito giuridico, descritto nella sezione 3.2.

Il knowledge graph ottenuto rappresenta una collezione di dati utilizzabile per task di giustizia predittiva ed, eventualmente, può essere interrogato mettendo a disposizione un endpoint SPARQL, linguaggio trattato nella sezione 2.3.4.

Questo capitolo è organizzato come segue. Nella sezione 4.1 è analizzato quanto contenuto nel sito dell'ECHR. Successivamente, nella sezione 4 è definito il processo di generazione del knowledge graph, seguito, nella sezione 4.3, dall'applicazione della metodologia proposta, in cui si trattano gli aspetti più pratici.

4.1 Analisi dei dati

Le pronunce emesse dall'ECHR sono raccolte nel sito istituzionale https://hudoc.echr.coe.int/. Prendendo come esempio la pronuncia intitolata *"CASE OF A AND B v. GEORGIA"*, possiamo notare la presenza di tre schede diverse, intitolate: *View*, *Case Details*, *Language Versions*, *Related*. Tra queste le due che hanno un'importanza maggiore per lo scopo di questa tesi sono:

- **View**: questa scheda contiene l'intero corpo del documento (vedi figura 6)
- **Case Details**: contiene delle informazioni semi strutturate riguardo al documento (vedi figura 7)

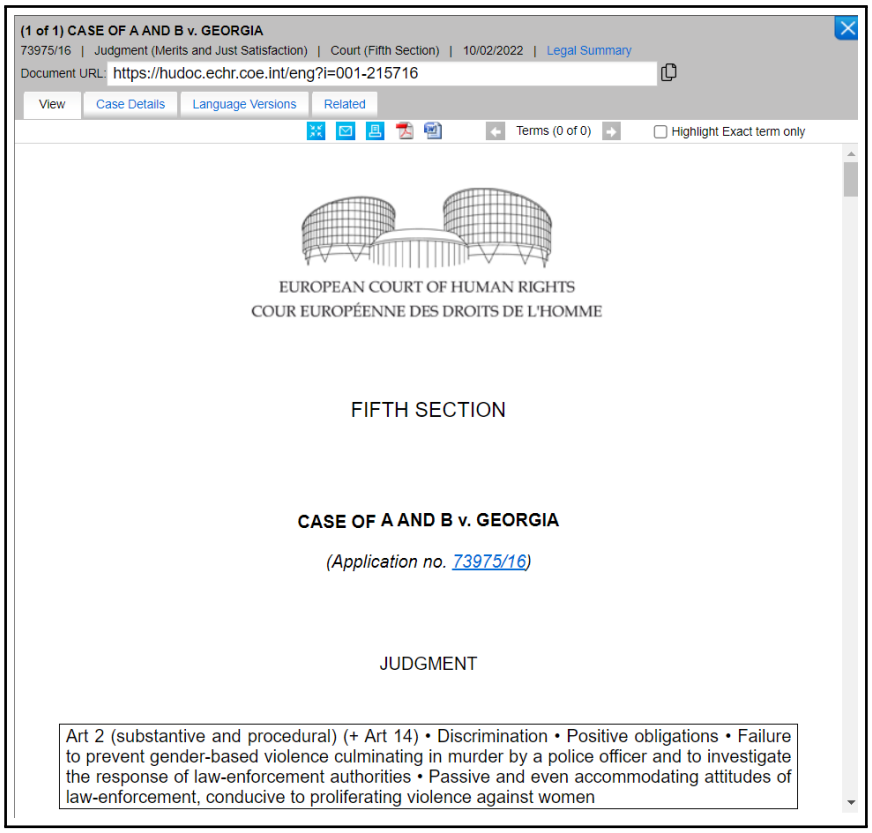

Figura 6: Scheda contenente la pronuncia

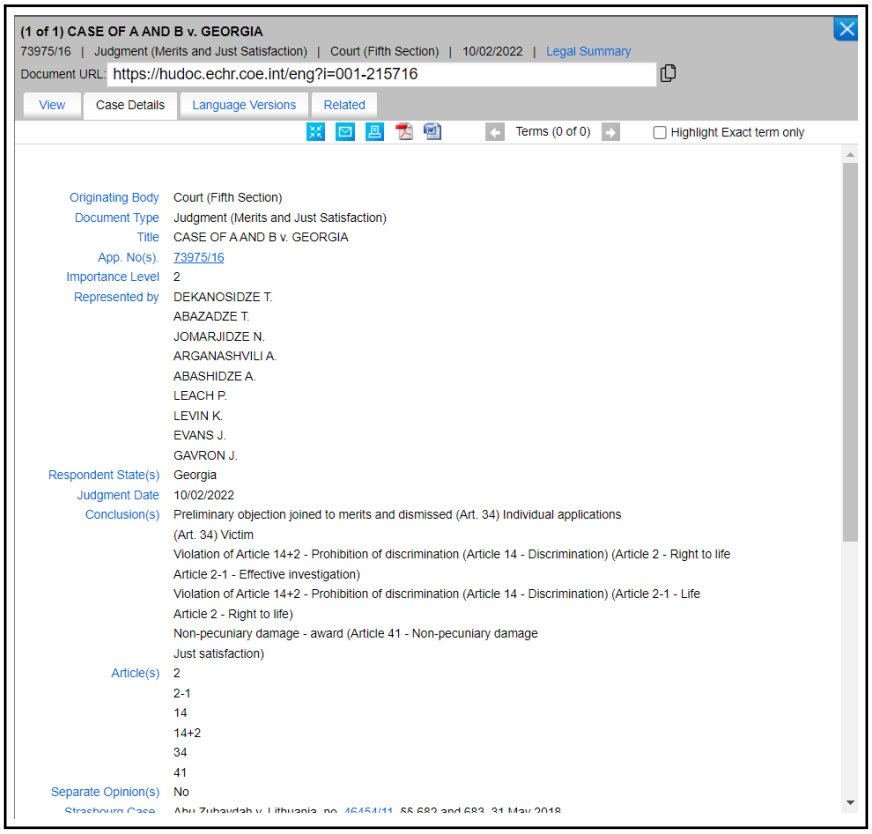

Figura 7: Scheda relativa ai "Case Detail"

Soffermandoci su quanto contenuto tra i "Case Detail", i dati presenti sono:

Case Detail	**Descrizione del contenuto**
Originating Body	Numero della sezione della corte
Document Type	Specifica se il documento è Decision o Judgment
Published in	Nome della raccolta in cui è pubblicato
Title	Titolo del documento
App. No(s).	Numero identificativo del ricorso
Importance Level	Livello di importanza del caso (da 1 a 3 e Key cases)

Represented by	Nomi degli avvocati di chi fa ricorso
Respondent State(s)	Stato convenuto
Introduction Date	Data di deposito del ricorso (solo per Decision)
Reference Date	Data in cui il caso è stato deferito a ECHR
Judgment Date	Data del giudizio
Decision Date	Data della seduta della decisione
Applicability	Presente se è stata sollevata una questione dell'applicabilità di un articolo della Convenzione
Conclusion(s)	Conclusioni tratte (ad esempio se ci sono danni morali)
Article(s)	Numero degli articoli della Convenzione interessati nel caso
Rules of Court	Numero della regola di tribunale
Separate Opinion(s)	Unanimità della sentenza. Se il valore è "Yes" al documento è aggiunta una sezione contenente l'opinione di chi non era a favore
Domestic Law	Leggi prese in considerazione dello stato di appartenenza di chi fa ricorso
Strasbourg Case-Law	Giurisprudenza ECHR presa in considerazione
International Law	Leggi internazionali prese in considerazione
Keywords	Parole chiave che rappresentano doc.
ECLI	Identificatore come definito in 3.2.2

Tabella 1: Descrizione dei dati nella sezione Case Detail

Alcune delle informazioni descritte nella tabella 1 non sono contenute nel corpo del documento. In particolare questi sono: *Published in*, *Importance Level* ed *ECLI*.

Di questi solo l'ultimo può essere derivato dalle informazioni contenute nel testo, in quanto ha un formato standard.

Come detto nella sezione 3.2.2 l'identificatore ECLI è suddiviso in cinque parti principali, le quali, nel caso specifico, hanno i seguenti valori:

1. **ECLI**: uguale per tutti gli identificatori per definizione;
2. **CE**: uguale per tutti i documenti e indica che la decisione è stata presa nella Comunità Europea;
3. **ECHR**: uguale per tutti i documenti e indica che la decisione è stata presa dalla Corte Europea dei Diritti dell'Uomo;
4. **anno**: contenuto nella data della decisione o del giudizio;
5. **numero univoco**: ha una forma particolare, in quanto è formato da; due cifre per il mese; due cifre per il giorno; JUD o DEC in base al tipo di documento; nove cifre in cui quelle più a destra contengono il numero identificativo del ricorso e, dato che questo ha una lunghezza inferiore a quella definita, è aggiunto un padding contenente zeri a sinistra, fino ad arrivare a nove cifre.

Riprendendo l'esempio fatto in precedenza *ECLI:CE:ECHR:2022:0210JUD007397516*, le prime quattro sezioni sono facilmente individuabili, successivamente abbiamo che *02* indica il mese, *10* il giorno, stiamo considerando un documento di tipo *Judgment* e, infine, ha come numero di ricorso *73975/16* con i precedenti *00* necessari per arrivare al numero prefissato di cifre di cui deve essere composto l'identificatore per la giurisprudenza.

4.2 Pipeline

Quanto descritto nella sezione 3.1 può essere una base di partenza per la definizione di un processo di generazione di un knowledge graph, nello specifico, per quanto riguarda le sentenze dell'ECHR.

Prima di tutto è stata fatta una scelta sulla metodologia di sviluppo, preferendo quella bottom-up. La scelta di questa metodologia, a discapito di quella top-down,

nasce dalla volontà di estrarre quanta più conoscenza possibile dai dati e, successivamente, mapparli ad una ontologia. Infatti, questa metodologia permette di scoprire eventuali relazioni emergenti tra le entità, le quali rimarrebbero non rappresentate seguendo il metodo top-down, in cui si segue uno schema ontologico predefinito.

L'approccio utilizzato per la generazione, di seguito descritto, presenta dei punti in comune e delle differenze rispetto a quanto detto nella sezione 3.1. I passi di raccolta dei dati e di estrazione della conoscenza sono in comune dal punto di vista concettuale, ma applicati al caso specifico. Dato che l'obiettivo primario è quello di estrarre conoscenza dai dati, questi passi sono fondamentali in qualsiasi sistema avente lo stesso scopo. Discorso simile vale per la fase di costruzione del knowledge graph. La differenza principale si trova nella maggiore importanza data per la creazione delle ontologie, in quanto viene suggerito l'uso di una metodologia di sviluppo che guidi la creazione, nello specifico quella basata su competency questions[22]. Inoltre, è stato dato maggiore spazio alla creazione di un endpoint SPARQL.

Il processo di sviluppo bottom-up di un knowledge graph partendo dalle sentenze emesse dall'ECHR è costituito dai seguenti passi:

1. **Raccolta dei dati**: individuazione da parte di esperti del dominio di quelle pronunce le quali trattano casi di violenze di genere e raccolta di riferimenti dal sito della Corte Europea dei Diritti dell'Uomo;

2. **Estrazione della conoscenza**: estrazione della conoscenza sotto forma di triple dalle pronunce raccolte, utilizzando tecniche e strumenti diversi in base alla tipologia di dati. Ad esempio, nel caso di dati non strutturati, quindi di testo libero, possono essere usati tecniche di Natural Language Processing o strumenti come Large Language Model. Nel caso di dati semi strutturati si possono utilizzare le espressioni regolari. Dato che, come descritto nella sezione 4.3.2, allo stato attuale è stato utilizzato solo il contenuto di Case Details, sono state utilizzate delle espressioni regolari. Come descritto nel capitolo 6, è stata presa in considerazione l'utilizzo di GPT per l'estrazione di conoscenza dal contenuto;

3. **Integrazione delle triple**: in questa fase si eliminano eventuali ridondanze e inconsistenze nelle triple ottenute. Inoltre si controlla che le entità siano identificate da URI. Come risultato si deve ottenere un insieme di triple rappresentate in un formato standard del web semantico, secondo i principi dei LOD;

4. **Creazione dell'ontologia**: si crea un'ontologia partendo da quelle esistenti nel dominio legale o di utilizzo comune ed espandendole, nel caso in cui si ha necessità di aggiungere nuovi concetti non rappresentati. In questo passo è opportuno essere guidati da una metodologia, come quella basata sulle competency questions[22],in cui si formulano delle domande a cui la conoscenza basata sull'ontologia deve poter rispondere;

5. **Costruzione del knowledge graph**: in questa fase si mettono insieme le triple ottenute da ognuno dei documenti presi in considerazione, in modo da ottenere il KG. Seguendo i principi dei LOD è importante che questo possa avere collegamenti verso altre risorse già esistenti. Inoltre, in questa fase si può permettere la visualizzazione del KG ottenuto;

6. **Endpoint SPARQL**: infine, come ultimo passo, opzionale ma fortemente consigliato, è quello della creazione di un endpoint SPARQL per l'interrogazione dei dati. In assenza di questo si disporrebbe comunque di una collezione di dati sotto forma di triple.

Nella figura 8 è contenuta una rappresentazione grafica dell'approccio proposto.

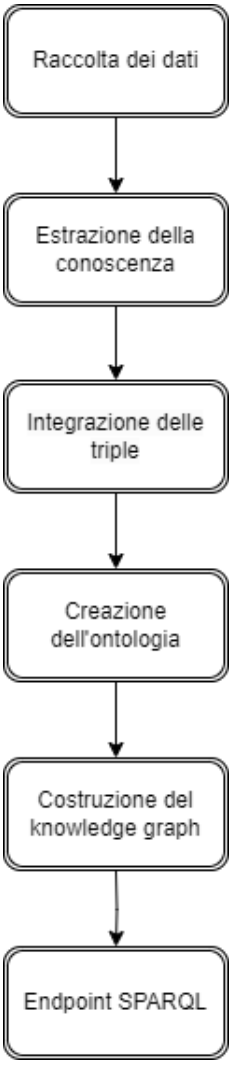

Figura 8: Diagramma della pipeline

4.3 Applicazione della Metodologia Proposta

Si è sviluppato un sistema che implementi quanto descritto in precedenza, scritto in Python e liberamente disponibile su GitHub[1]. Nelle sezioni successive viene spiegato come è stato affrontato ogni passo della pipeline.

[1]Repository: https://github.com/PeppeRubini/EVA-KG

4.3.1 Raccolta dei dati

I documenti presenti nel sito sono scritti in almeno una delle lingue ufficiali della Corte, ovvero inglese e francese. Per semplicità sono state state prese in considerazione solo quelli in lingua inglese. Le sentenze sono state scelte da esperti del dominio con competenze in diritto internazionale, membri del progetto citato nella sezione 1.1. I riferimenti a questi documenti sono stati successivamente raccolti in un unico documento [2]

Dati questi collegamenti alle pronunce, si è posto il problema di ottenere i file contenenti le informazioni necessarie. Per fare ciò è stata creata una funzione [3] che, dato un URL appartenente al dominio *https://hudoc.echr.coe.int/eng*, scarica automaticamente il file PDF, oltre che salvare il codice HTML, utilizzando la libreria Selenium [4] la quale permette di controllare un browser web in modo automatico e simulare azioni umane, come il clic del mouse su un elemento specifico.

Si è scelto di salvare anche l'HTML in modo tale che si possa facilmente accedere ai dati contenuti nei Case Details, in quanto non sono presenti nei file PDF.

Nel passo successivo ogni sentenza è processata singolarmente per l'estrazione della conoscenza.

4.3.2 Estrazione della conoscenza

Una volta raccolti tutti i dati individuati nel punto precedente, questi vengono processati per estrarre le triple.

In questa sezione sono illustrati i dettagli per l'applicazione della metodologia presentata e come questi sono stati realizzati nella soluzione implementata.

Questo passo procede considerando una sentenza per volta, creando triple dai metadati contenuti nella sezione Case Details del sito dell'ECHR, illustrati nella tabella 1. Allo stato non si riescono ad estrarre triple dal corpo dei documenti, ma questo fa parte degli obiettivi da raggiungere, descritti nella sezione 6

[2] https://github.com/PeppeRubini/EVA-KG/blob/main/data/mapping_doc_link.xlsx
[3] https://github.com/PeppeRubini/EVA-KG/blob/main/src/echr_scraper.py
[4] Selenium: https://selenium-python.readthedocs.io/

In particolare è stata creata una classe per rappresentare un documento della Corte Europea dei Diritti dell'Uomo, chiamata *ECHRDocument*[5]. Per l'inizializzazione, questa classe prende in input i percorsi delle directory in cui si trova il file PDF e l'HTML, oltre che il nome del file (per come è stato implementato lo script citato nel punto precedente i due file hanno lo stesso nome a meno dell'estensione).

Dato che la conoscenza estratta è quella contenuta nella sezione Case Detail (vedi immagine 7 e tabella 1), si è deciso di processare solo il file HTML, in quanto questi metadati sono presenti solo in questo file. Il file PDF può avere una maggiore importanza nell'estrazione della conoscenza dal contenuto in quanto la sua strutturazione (le diverse pagine o i comma) è più facilmente comprensibile, rispetto ai tag HTML presenti nella pagina web.

Prima di creare le triple, sono state raccolte le informazioni in modo che siano più facilmente processabili, attraverso il metodo *extract_case_detail_from_html()*. Questo metodo, partendo da quanto contenuto nel file HTML, avvalora un dizionario attributo di classe chiamato *_case_detail*, ponendo come chiavi le diverse intestazioni (la prima colonna della tabella 1) e come valori i rispettivi contenuti.

L'estrazione delle diverse informazioni è stata fatta utilizzando la libreria Beautiful Soup[6], la quale permette di analizzare codice HTML. In particolare, osservando il contenuto dei file si vede come nomi e valori siano contenuti in blocchi $<div>$ a cui è assegnata una classe specifica:

- per le intestazioni *span2 noticefieldheading*;
- per i valori *col-offset-2 noticefieldvalue*.

Tra i dati presi in considerazione è presente il nome dello stato convenuto. Invece di rappresentarlo come una stringa, questi valori sono stati sostituiti con l'URI Wikidata corrispondente: si effettua una query all'endpoint SPARQL messo a disposizione da Wikidata cercando un'istanza di *"sovereign state"* (identificato da *Q3624078*) avente come label principale o alternativa il valore contenuto nella stringa.

[5]https://github.com/PeppeRubini/EVA-KG/blob/main/src/ECHRDocument.py
[6]Beautiful Soup: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Un'altra modifica ai valori presenti è stata fatta per il campo Importance Level. Dato che i valori in origine vanno da 1 a 3 a cui si aggiunge anche "Key cases", per coerenza di tipo quest'ultimo è stato sostituito con il valore 4.

Quanto estratto finora può essere serializzato in un file JSON usando l'apposito metodo, oppure può essere usato per estrarre le triple. Per quest'ultima funzionalità è stato creato il metodo *extract_triples_from_case_detail()* il quale usa la libreria RDFLib[7] per creare triple in formato standard, e serializzate in file Turtle.

4.3.3 Integrazione delle triple e ridondanze

Per quanto riguarda questo passo, non è stato necessario svolgere alcun compito specifico, in quanto non è presente ridondanza nelle triple ottenute da ognuna delle sentenze.

Inoltre, quando successivamente mettiamo insieme le triple ottenute da ognuno dei documenti, RDFLib elimina automaticamente le eventuali ripetizioni contenute in file diversi.

4.3.4 Creazione dell'ontologia

Per la creazione dell'ontologia è stata seguito il metodo delle *competency questions*[22], il quale consiste nella formulazione di domande a cui la knowledge base basata sull'ontologia, nel nostro caso il knowledge graph, deve poter rispondere. Le domande poste sono:

CQ1: Quale tipo di documento stiamo trattando?

CQ2: Quando è datato il documento?

CQ3: Posso ritrovare le informazioni di un certo caso, dato un suo identificativo?

CQ4: Da chi è rappresentato chi fa ricorso?

CQ5: A quale stato europeo appartiene chi fa ricorso?

[7]RDFLib: https://rdflib.readthedocs.io/en/stable/

CQ6: Qual è stata la sentenza?

CQ7: La sentenza è stata presa all'unanimità?

CQ8: Quali articoli della Convenzioni sono stati presi in considerazione?

CQ9: Quali leggi sono state prese in considerazione per arrivare a tale conclusione?

CQ10: Qual è l'importanza della sentenza presa, rispetto a casi futuri?

CQ11: Il documento è pubblicamente accessibile?

CQ12: In quale lingua è scritto il documento?

CQ13: Dove posso consultare il documento?

Prima di creare un'ontologia partendo da zero, sono state analizzate quelle esistenti considerate nella sezione 3.2, facendo emergere che l'utilizzo di ECLI e dei suoi metadati sembrano essere più adatti a rappresentare le pronunce dell'ECHR e a rispondere alle precedenti domande. Non tutte, però, trovano una risposta, rendendo necessaria la definizione di ulteriori concetti e proprietà.

Partendo da quanto contenuto nella tabella 1 è stata creata una corrispondenza tra alcuni dei campi contenuti in Case Details e i metadati di ECLI:

Case Details	**Metadati ECLI**
Originating Body	dcterms:creator
Document Type	dcterms:type
Published in	dcterms:isPartOf
Title	dcterms:title
App. No(s).	-
Importance Level	-
Represented by	dcterms:contributor
Respondent State(s)	-

Introduction Date	-
Reference Date	-
Judgment Date	dcterms:date
Decision Date	dcterms:date
Applicability	-
Conclusion(s)	dcterms:abstract
Article(s)	-
Rules of Court	-
Separate Opinion(s)	-
Domestic Law	dcterms:references
Strasbourg Case-Law	dcterms:references
International Law	dcterms:references
Keywords	dcterms:description
ECLI	dcterms:isVersionOf

Tabella 2: Associazione campi Case Details-ECLI

Altri metadati messi a disposizione da ECLI possono essere avvalorati con conoscenza pregressa, in particolare:

- **dcterms:identifier**, con l'URL che contiene il documento;
- **dcterms:coverage**, con la città in cui ha sede l'ECHR, quindi Strasburgo;
- **dcterms:language**, nel caso specifico con il valore "en", dato che si sono presi in considerazione solo documenti in lingua inglese;
- **dcterms:publisher**, con il nome dell'organizzazione, quindi European Court of Human Rights;
- **dcterms:accessRights**, con il valore "public", in quanto le pronunce sono liberamente accessibili a chiunque.

Quanto definito nel precedente elenco puntato e nella tabella 2 definisce con quali dati presenti nei Case Details verranno avvalorati i metadati di ECLI. Inoltre, sono stati definiti anche domini e range di alcune delle proprietà importate, in modo tale da rappresentare in modo più accurato il dominio in esame.

Arrivati a questo punto, solo le domande 3, 7, 8 e 10 sono rimaste senza risposta. Questo sottolinea la necessità di dover creare nuovi concetti, in modo tale da permettere al KG di rispondere anche a queste domande. Quindi sono state definite classi, proprietà ed individui. Le classi create sono:

- **DomesticLaw**: sottoclasse dell'entità Wikidata *Q7748*, la quale indica diritto o legge;
- **InternationalLaw**: sottoclasse della stessa entità Wikidata precedente;
- **StrasbourgCaseLaw**: sottoclasse dell'entità Wikidata *Q11022655*, ovvero giurisprudenza.

Per quanto riguarda le proprietà, segue una tabella di quelle create contenente nome, dominio e range:

Proprietà	**Dominio**	**Range**
hasApplicationNumber	StrasbourgCaseLaw	stringa
importanceLevel	StrasbourgCaseLaw	numero intero (da 1 a 4)
respondentState	StrasbourgCaseLaw	stato sovrano in Wikidata (*Q3624078*)
involveConventionArticle	StrasbourgCaseLaw	stringa
unanimousDecision	StrasbourgCaseLaw	booleano

Tabella 3: Proprietà definite nell'ontologia

Quanto definito in questa tabella è anch'esso avvalorato con quanto contenuto nei Case Details o derivato da esso:

1. **hasApplicationNumber**: con il valore di "App. No(s).", ovvero il numero di ricorso, il quale ha la seguente forma *73975/15*;
2. **importanceLevel**: con il valore dell'omonimo campo;
3. **respondentState**: con la risorsa Wikidata ottenuta partendo da quanto contenuto nell'omonimo campo;
4. **involveConventionArticle**: con il valore di "Article(s)"
5. **unanimousDecision**: in base al valore di "Separate Opinon(s)". Se il valore è "No", allora nel KG sarà presente False, True altrimenti.

Infine, per quanto riguarda gli individui, è stato creato un URI per ognuna delle pronunce, identificate dal numero del ricorso, e dichiarate con il tipo *StrasbourgCaseLaw*, precedentemente definito.

4.3.5 Costruzione del KG e visualizzazione

Per ottenere il KG finale vengono unite le triple ottenute da ognuna delle pronunce. Il KG ottenuto presenta connessioni con altre risorse presenti sul web, nello specifico con quelle di Wikidata, come mostrato nella sezione precedente. Inoltre, sono state sviluppate tre soluzioni per la visualizzazione del KG. Queste sono indipendenti indipendenti tra loro ma convergono verso un risultato simile:

1. **PyVis**[8][9]: questa libreria Python permette di creare una rete, la quale viene salvata in un file HTML, visualizzabile da browser;
2. **RDF Grapher**[10][11]: è stata creata una funzione che dato il grafo che vogliamo rappresentare, serializzato in uno dei formati standard di RDF, effettua una

[8] PyVis: https://pyvis.readthedocs.io/en/latest/
[9] https://github.com/PeppeRubini/EVA-KG/blob/main/src/pyvis_utils.py
[10] RDF Grapher https://www.ldf.fi/service/rdf-grapher
[11] https://github.com/PeppeRubini/EVA-KG/blob/main/src/http_requests.py

richiesta HTTP POST al servizio, avente come payload l'insieme delle triple. La risposta dal server conterrà un'immagine (nel formato scelto) rappresentante il grafo;

3. **Neo4j**[12][13]: come ultima possibilità è stata utilizza libreria la libreria Neo4j, la quale crea basi di dati a grafo e ne permetta la visualizzazione e l'interrogazione, utilizzando il linguaggio Cypher.

4.3.6 Interrogazione tramite SPARQL Endpoint

Come ultimo passo, è stato creato un endpoint SPARQL locale [14] per permettere di interrogare il grafo ottenuto nel passo precedente. In particolare è stata utilizzata la libreria Flask [15] , la quale permette di gestire richieste HTTP, insieme alla libreria RDFLib per l'interrogazione del grafo.

Il server Flask è in ascolto su localhost alla porta 5000, rendendo possibile accedere al server attraverso il browser o effettuando richieste HTTP alla porta 5000 del localhost. Nello specifico il server gestisce un endpoint SPARQL accessibile tramite richieste a *http://localhost:5000/sparql*.

Una volta avviato il server, è possibile interrogare il grafo, inserendo una query dalla console. Successivamente si controlla la correttezza sintattica della stessa e, se è scritta correttamente, viene posta all'endpoint il quale ritorna i risultati come file JSON. È stato scelto questo formato perché è quello più utilizzato nell'ambito dello scambio dei dati sul web.

Segue un esempio di query in cui si chiede di trovare il riferimento a quei documenti aventi come valore per la proprietà *importanceLevel* almeno 3:

[12]Neo4j: https://neo4j.com/
[13]https://github.com/PeppeRubini/EVA-KG/blob/main/src/neo4j_utils.py
[14]https://github.com/PeppeRubini/EVA-KG/blob/main/src/endpoint.py
[15]https://flask.palletsprojects.com/en/3.0.x/

```
PREFIX eva: <https://github.com/PeppeRubini/EVA-KG/tree/main
    /ontology/ontology.owl#>
PREFIX dcterms: <http://purl.org/dc/terms/>

SELECT ?o
WHERE {
        ?s dcterms:identifier ?o .
        ?s eva:importanceLevel ?l .
        FILTER(?l >= 3)
}
```

Questa query ritorna una lista formata da trentasei elementi, a titolo esemplificativo ne viene mostrato solo uno :

{'o': {'type': 'uri', 'value': 'https://hudoc.echr.coe.int/eng?i=001-114397'}}

Come si può vedere, la risposta è un dizionario avente come chiave il nome della variabile specificata nella clausola di selezione e come valore un ulteriore dizionario contenente tipo e valore associato al nodo.

Capitolo 5

Valutazione

Questa tesi rappresenta un contributo significativo nel campo giuridico, affrontando con approccio innovativo la complessa tematica delle violenze sulle donne attraverso la creazione di un knowledge graph. La valutazione di tale lavoro si basa su diversi criteri chiave, evidenziando l'unicità del progetto, le dimensioni notevoli del knowledge graph e l'accessibilità dei dati.

Il knowledge graph sviluppato si distingue per la sua unicità nel contesto di documenti giuridici relativi alla violenza sulle donne, offrendo una rappresentazione interconnessa e facilmente interpretabile dalle macchine.

Il KG ottenuto ha permesso di rendere liberamente accessibili e interrogabili alcune delle sentenze della Corte Europea dei Diritti dell'Uomo avente come argomento la violenza di genere, suddivise come nella tabella 4 Nella tabella 5 sono indicati

Judgment	65
Decision	8
Totale	73

Tabella 4: Numero e tipi di sentenze ECHR rappresentate nel KG

alcuni dei valori relativi alle dimensioni del knowledge graph ottenuto. Questi valori indicano lo stato attuale, ma hanno la potenzialità di crescere, considerando ulteriori sentenze o estraendo ulteriore conoscenza, come indicato nel capitolo successivo.

Seguendo i principi di LOD, il KG utilizza risorse già esistenti nel web semantico, nello specifico ne utilizza 27 da Wikidata. Queste sono rappresentate nella tabella 7

Per quanto riguarda l'ontologia, la tabella 6 evidenzia come ad ognuna delle competency questions può ottenere una risposta.

Dato	Valore
Numero di triple totali	10325
Numero di nodi totali	5185
Numero di predicati distinti	22
Numero di soggetti distinti	1747
Numero di oggetti distinti	5108

Tabella 5: Dati relativi alla composizione del KG

Competency Question	Risposta
Quale tipo di documento stiamo trattando?	dcterms:type
Quando è datato il documento?	dcterms:date
Posso ritrovare le informazioni di un certo caso, dato un suo identificativo?	dcterms:isVersionOf per ECLI e hasApplicationNumber num. ricorso
Da chi è rappresentato chi fa ricorso?	dcterms:contributor
A quale stato europeo appartiene chi fa ricorso?	respondentState
Qual è stata la sentenza?	dcterms:abstract
La sentenza è stata presa all'unanimità?	unanimousDecision
Quali articoli della Convenzioni sono stati presi in considerazione?	involveConventionArticle
Quali leggi sono state prese in considerazione per arrivare a tale conclusione?	dcterms:references
Qual è l'importanza della sentenza presa, rispetto a casi futuri?	importanceLevel
Il documento è pubblicamente accessibile?	dcterms:accessRights
In quale lingua è scritto il documento?	dcterms:language
Dove posso consultare il documento?	dcterms:identifier

Tabella 6: Tabella delle competency questions

Dato	Valore
http://www.wikidata.org/entity/Q224	Croatia
http://www.wikidata.org/entity/Q34	Sweden
http://www.wikidata.org/entity/Q214	Slovakia
http://www.wikidata.org/entity/Q219	Bulgaria
http://www.wikidata.org/entity/Q3769186	judgment
http://www.wikidata.org/entity/Q218	Romania
http://www.wikidata.org/entity/Q6602	Strasbourg
http://www.wikidata.org/entity/Q230	Georgia
http://www.wikidata.org/entity/Q159	Russia
http://www.wikidata.org/entity/Q27	Republic of Ireland
http://www.wikidata.org/entity/Q36	Poland
http://www.wikidata.org/entity/Q37	Lithuania
http://www.wikidata.org/entity/Q40348	lawyer
http://www.wikidata.org/entity/Q122880	European Court of Human Rights
http://www.wikidata.org/entity/Q222	Albania
http://www.wikidata.org/entity/Q40	Austria
http://www.wikidata.org/entity/Q43	Turkey
http://www.wikidata.org/entity/Q229	Cyprus
http://www.wikidata.org/entity/Q29	Spain
http://www.wikidata.org/entity/Q145	United Kingdom
http://www.wikidata.org/entity/Q28	Hungary
http://www.wikidata.org/entity/Q29999	Kingdom of the Netherlands
http://www.wikidata.org/entity/Q327000	court decision
http://www.wikidata.org/entity/Q38	Italy
http://www.wikidata.org/entity/Q212	Ukraine
http://www.wikidata.org/entity/Q215	Slovenia
http://www.wikidata.org/entity/Q217	Moldova

Tabella 7: Tabella delle risorse di Wikidata

Capitolo 6

Conclusioni e sviluppi futuri

Negli ultimi anni si è assistito alla diffusione dell'intelligenza artificiale applicata a diversi domini, tra cui quello legale. Quanto sviluppato in questa tesi mira ad essere il punto di partenza di un più ampio progetto di giustizia predittiva, fornendo un modello da seguire per rappresentare la conoscenza utilizzando i grafi, oltre che un'implementazione dello stesso.

Nello specifico si è sviluppato un KG in ambito giuridico in particolare per trattare casi di violenza sulle donne a partire da sentenze esistenti. Il KG è stato sviluppato secondo una metodologia proposta, che riutilizza i principi esistenti e li specializza per l'applicazione al particolare dominio. Il KG realizzato è stato ottenuto esaminando 73 sentenze dando luogo ad una base di conoscenza costituita da 10325 triple, 5185 nodi, utilizzando 22 predicati differenti, ed interconnesso con Wikidata per la rappresentazione di informazioni principalmente geografiche (per identificare gli stati). Il KG è inoltre interrogabile mediante SPARQL endpoint appositamente realizzato.

Per quanto riguarda l'estrazione della conoscenza, è stato preso in considerazione l'utilizzo dell'API di GPT. L'idea era quella di formulare un prompt contenete la richiesta di estrazione di triple dal testo della pronuncia.

Analizzando il funzionamento di questo LLM, si può vedere che alla base ci sono i token, ovvero unità di testo, che possono essere parole o altri elementi, come la punteggiatura, i quali formano sia la richiesta che la risposta generata. Il numero di token nel complesso, prendendo in considerazione sia quelli di input che quelli di output, è limitato e varia in base alla versione del modello utilizzato[16]. Ad esempio GPT 3.5 Turbo ha un limite di 4097 token. Questo rende impossibile pensare di fornire l'intera pronuncia in input, rendendo necessario suddividerla in sezioni mol-

[16]https://platform.openai.com/docs/guides/text-generation/managing-tokens

to piccole, portando alla perdita di contesto e di eventuale dipendenza tra sezioni diverse.

Dato questa forte limitazione è stato scelto di non proseguire su questa strada, ma nell'appendice A è stata fornita una presentazione del funzionamento dell'API, per eventuali sviluppi futuri.

Nonostante il lavoro svolto, ci sono delle sfide ancora aperte, le quali si possono riassumere nei seguenti punti:

- **estrazione di triple dal contenuto**: il contenuto dei documenti presi in considerazione rispettano una struttura specifica: la prima parte racconta i fatti accaduti ed oggetto del ricorso; successivamente è descritto il processo che si è tenuto nel Paese di appartenenza e, infine, la sentenza da parte della Corte Europea dei Diritti dell'Uomo. Utilizzando la conoscenza contenuta in ognuna di queste sezioni, sfruttando dei Large Language Model o tecniche di Natural Language Processing, si potrebbero ottenere delle triple da integrare nel grafo, in modo da renderlo più completo;

- **endpoint SPARQL pubblico**: allo stato attuale l'endpoint, una volta avviato il server, è utilizzabile solo all'interno della rete locale. Un miglioramento può essere apportato rendendo l'endpoint liberamente accessibile sul web, eventualmente fornendo in aggiunta un'interfaccia, in modo tale da migliorare anche l'interazione uomo-macchina;

- **valutazione da parte di esperti del dominio**: sottoporre quanto creato ad esperti del dominio giuridico, per individuare eventuali criticità o miglioramenti. Ad esempio, potrebbero porre delle competency question più precise data la conoscenza maggiore del dominio, in modo da migliorare l'ontologia;

- **applicazione in altri contesti giuridici**: l'approccio proposto potrebbe anche essere utilizzato per creare un knowledge graph per altre categorie di reati. In una visione molto ambiziosa, questi potrebbero portare ad ottenere una rappresentazione unica e facilmente interpretabile dalle macchine di leggi o giurisprudenza europea.

Appendice A

Utilizzo API GPT

In questa appendice viene fornita una breve panoramica sul funzionamento dell'API di GPT[17]. Prima di tutto, se stiamo usando Python come linguaggio di programmazione, dobbiamo installare la libreria, attraverso il comando *pip install openai* e successivamente importandola con *from openai import OpenAI* . Per utilizzare i servizi offerti, bisogna prima di tutto ottenere una chiave per l'autenticazione e, successivamente, specificarla con l'istruzione *openai.api_key* a cui assegniamo come valore la key ottenuta.

Il modo principale per lavorare con il testo è attraverso la *chat completions API*[18], le cui richieste hanno la seguente forma:

```
from openai import OpenAI
client = OpenAI()
response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a helpful
            assistant."},
        {"role": "user", "content": "Who won the World
            Series in 2020?"},
        {"role": "assistant", "content": "The Los Angeles
            Dodgers won the World Series in 2020."},
        {"role": "user", "content": "Where was it played?"}
    ]
)
```

[17]https://platform.openai.com/docs/api-reference/
[18]https://platform.openai.com/docs/guides/text-generation/chat-completions-api

La richiesta è formata da due parti principali:

- nella prima parte, con *model*, si va a specificare il modello che verrà utilizzato;
- successivamente, in *messages*, si imposta un dialogo tra l'utente e l'assistente, specificandone il ruolo e il contenuto. I possibili ruoli sono
  - **System**: permette di modificare il comportamento dell'assistente;
  - **Assistant**: contiene le risposte alle domande dell'utente;
  - **User**: contiene la domanda dell'utente. Se è l'ultimo tra i messaggi, questa rappresenta la domanda che andiamo a porre.

Ulteriori parametri opzionali sono:

- **Temperature**: valore tra 0 e 2. Un valore più alto comporta più casualità nella risposta, mentre un valore più basso la rende più deterministica;
- **Top_p**: valore tra 0 e 1. Considera i token che hanno una probabilità maggiore di quella fornita;
- **Max_token**: determina il numero massimo di token da ritornare.

Anche la risposta fornita ha un formato preciso, e si presenta come segue:


```
{
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "message": {
        "content": "The 2020 World Series was played in
           Texas at Globe Life Field in Arlington.",
        "role": "assistant"
      }
    }
```

```
  ],
  "created": 1677664795,
  "id": "chatcmpl-7QyqpwdfhqwajicIEznoc6Q47XAyW",
  "model": "gpt-3.5-turbo-0613",
  "object": "chat.completion",
  "usage": {
    "completion_tokens": 17,
    "prompt_tokens": 57,
    "total_tokens": 74
  }
}
```


Analizzando le parti principali si ha che:

- il campo *finish_reason* contenente la motivazione che ha portato alla terminazione. Questa può essere:
  - **stop**: l'API ha ritornato il messaggio completo;
  - **lenght**: l'output è incompleto a causa del parametro max_tokens o si è raggiunto il limite di token;
  - **function_call**: il modello ha deciso di chiamare una funzione;
  - **content_filter**: contenuto omesso a causa di un filtro;
  - **null**: risposta in generazione o incompleta.
- la risposta alla domanda è contenuta nel campo *content* di *messages*. Quindi per accedere alla risposta bisogna eseguire l'istruzione *response['choices'][0]['message']['content']*;
- in *usage* è descritto il conteggio di token utilizzati, sia di input che di output. Per quelli di input sono presi in considerazione tutti i messaggi presenti nel dialogo. Il valore di *total_tokens* può essere al massimo pari al valore limite di token del modello (ad esempio 4097 per GPT 3.5), rendendo necessario

un adeguato computo dei token di input ed eventualmente limitando quelli di output. Inoltre questo valore determina il costo del servizio[19].

[19]https://openai.com/pricing

Riferimenti bibliografici

[1] Vito Walter Anelli, Eros Brienza, Marco Recupero, Francesco Greco, Andrea De Maria, Tommaso Di Noia, and Eugenio Di Sciascio. Navigating the legal landscape: Developing italy's official legal knowledge graph for enhanced legislative and public services. *ceur-ws.org/Vol-3486*, 2022.

[2] World Wide Web Consortium. Linked data. `https://www.w3.org/DesignIssues/LinkedData.html`.

[3] World Wide Web Consortium. Owl. `https://www.w3.org/OWL/`.

[4] World Wide Web Consortium. Rdf schema. `https://www.w3.org/TR/rdf12-schema/`.

[5] World Wide Web Consortium. Sparql. `https://www.w3.org/TR/sparql11-query/`.

[6] World Wide Web Consortium. Turtle. `https://www.w3.org/TR/turtle/`.

[7] Dublin core metadata initiative (dcmi). `https://www.dublincore.org/specifications/dublin-core/dcmi-terms/`.

[8] European case law identifier (ecli). `https://e-justice.europa.eu/content_european_case_law_identifier_ecli-175-it.do`.

[9] Conclusioni del consiglio che invitano all'introduzione dell' european case law identifier (ecli) e di una serie minima di metadata uniformi per la giurisprudenza. `https://eur-lex.europa.eu/legal-content/IT/TXT/?uri=uriserv%3AOJ.C_.2011.127.01.0001.01.ITA&toc=OJ%3AC%3A2011%3A127%3AFULL`.

[10] Major progress towards transparency: a new european legislation identifier. `https://ec.europa.eu/commission/presscorner/detail/en/IP_12_1040`.

[11] European legislation identifier (eli). https://op.europa.eu/en/web/eu-vocabularies/eli.

[12] Conclusioni del consiglio che invitano all'introduzione dell'identificatore della legislazione europea (eli). https://eur-lex.europa.eu/legal-content/IT/TXT/?uri=CELEX%3A52012XG1026%2801%29.

[13] Eurovoc. https://op.europa.eu/en/web/eu-vocabularies/dataset/-/resource?uri=http://publications.europa.eu/resource/dataset/eurovoc. vedi sezione Documentation.

[14] Erwin Filtz, Sabrina Kirrane, and Axel Polleres. The linked legal data landscape: linking legal data across different countries. *Artificial Intelligence and Law*, 29(4):485–539, 2021.

[15] La gazzetta ufficiale. https://www.gazzettaufficiale.it/caricaHtml?nomeTiles=gazzettaUfficiale.

[16] Aidan Hogan. Linked data & the semantic web standards., 2014.

[17] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutiérrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. *Knowledge Graphs*. Number 22 in Synthesis Lectures on Data, Semantics, and Knowledge. Springer, 2021.

[18] Iso 639. https://www.iso.org/iso-639-language-codes.html.

[19] Iso 8601. https://www.iso.org/iso-8601-date-and-time-format.html.

[20] Ontology, by tom gruber, encyclopedia of database systems, ling liu and m. tamer Özsu (eds.), springer-verlag, 2008. http://web.dfc.unibo.it/buzzetti/IUcorso2007-08/mdidattici/ontology-definition-2007.htm.

[21] Gruber, t. r., a translation approach to portable ontology specifications. knowledge acquisition, 5(2):199-220, 1993. https://tomgruber.org/writing/ontolingua-kaj-1993.pdf.

[22] Ontology development 101: A guide to creating your first ontology. https://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html.

[23] Ontologia e informatica, wikipedia. https://it.wikipedia.org/wiki/Ontologia#Ontologia_e_informatica.

[24] Owl web ontology language overview. https://www.w3.org/TR/owl-features/.

[25] Rdf 1.2 concepts and abstract syntax. https://www.w3.org/TR/rdf12-concepts/.

[26] Skos simple knowledge organization system reference. https://www.w3.org/TR/skos-reference/.

[27] Francesco Sovrano, Monica Palmirani, and Fabio Vitali. Legal knowledge extraction for knowledge graph based question-answering. In *Legal Knowledge and Information Systems*, pages 143–153. IOS Press, 2020.

[28] Sparql 1.1 entailment regimes. https://www.w3.org/TR/sparql11-entailment/.

[29] Gytundefined Tamašauskaitundefined and Paul Groth. Defining a knowledge graph development process through a systematic review. *ACM Trans. Softw. Eng. Methodol.*, 32(1), feb 2023.

[30] Thi-Hai-Yen Vuong, Minh-Quan Hoang, Tan-Minh Nguyen, Hoang-Trung Nguyen, and Ha-Thanh Nguyen. Constructing a knowledge graph for vietnamese legal cases with heterogeneous graphs. *arXiv preprint arXiv:2309.09069*, 2023.