Team Details	
Name	Nihar Mandahas, Pratheek Rao M P, Manvith L B
Email	niharmandahas.cs22@rvce.edu.in pratheekrmp.cs22@rvce.edu.in manvithlb.cs22@rvce.edu.in
University	RV College of Engineering (affiliated to VTU)
Course Pursued	B.E. (CSE)
Country of Residence	India
Date of Graduation	May 2026
RVCE Mentors	
Dr Shobha G, Professor, R V College of Engineering, Bengaluru, India shobhag@rvce.edu.in, +919480280273	

Jyoti Shetty,

Assistant Professor,

R V College of Engineering,

Bengaluru, India,

jyothis@rvce.edu.in, +919900052901

Project Proposal

Title: Enhancing Legal Assistance through Data Enrichment with HPCC Systems

The legal system needs to undergo a radical makeover as a result of the rising crime rates, and integrating artificial intelligence (AI) will be essential to this shift. Legal professionals may effectively handle and analyse huge volumes of legal data and derive useful insights essential for judicial studies by utilising cutting-edge big data technology like HPCC Systems. The system's cognitive powers are further boosted by the addition of outside knowledge sources, such Wiki Data, giving it more understanding and decision-making power.

Deliverables:

1. The core objective is to deliver a software that searches keywords in the input document and then shows most related legal references to the user.

2. Documentation of the results.

Wish list:

1. Expanding the project scope involves incorporating a user feedback mechanism to iteratively refine the relevance ranking of legal cases.

Introduction:

The legal system is based on precedents, with earlier court rulings serving as models for current cases. Finding analogous cases and spotting hidden relationships becomes essential in this situation for deciphering legal nuances and gathering pertinent data. But as legal professionals sift through the massive collection of cases, they frequently find intricate patterns and relationships that aren't always obvious. Graph-based solutions are useful in the legal field because they make it possible to depict complex relationships. The ideal choice for this application is to use **HPCC systems because they can manage the large-scale data processing** and storage needs of creating and querying a legal ontology on a legal data corpus, allowing for effective retrieval of context-aware information.

The proposed algorithm uses an existing legal ontology defined in **RDF triplets** to locate pertinent legal precedents for a particular input legal document. The HPCC cluster is sprayed with the ontology in RDF format and converted to XML for computation.

The Input Keywords variable contains the keywords that the algorithm first extracts from the input legal document using common NLP techniques like tokenization, stemming, and lemmatization. Kl. K2.. K6 in Fig. 1 stands for extracted Keywords. The ontology is queried in the first iteration for each keyword in the input keywords, and when the object part of an RDF triplet matches the keyword, the subject component of the triplet is retrieved.

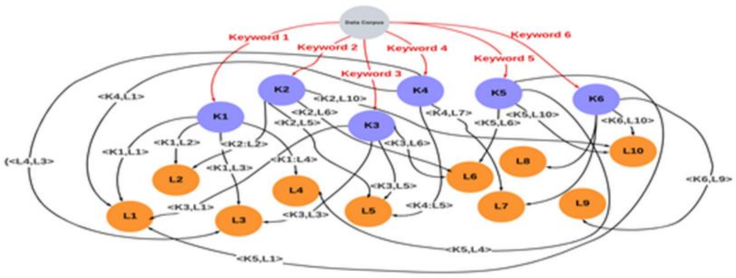


Fig. 1: **Representation of extracted keywords**

DBpedia:

DBpedia is a crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects. This structured information resembles an **open knowledge graph (OKG)** which is available for everyone on the Web. A knowledge graph is a special kind of database which stores knowledge in a machine-readable form and provides a means for information to be collected, organised, shared, searched and utilised. DBpedia extracts structured information from Wikipedia, interlinks it with other knowledge bases and freely publishes the results on the Web using Linked Data and **SPARQL.**

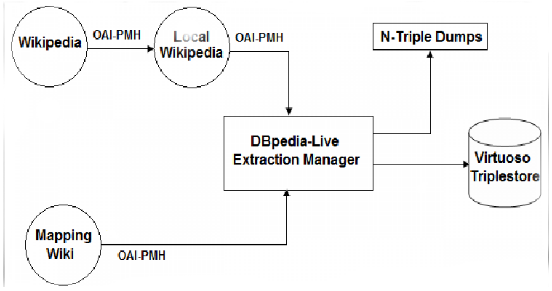
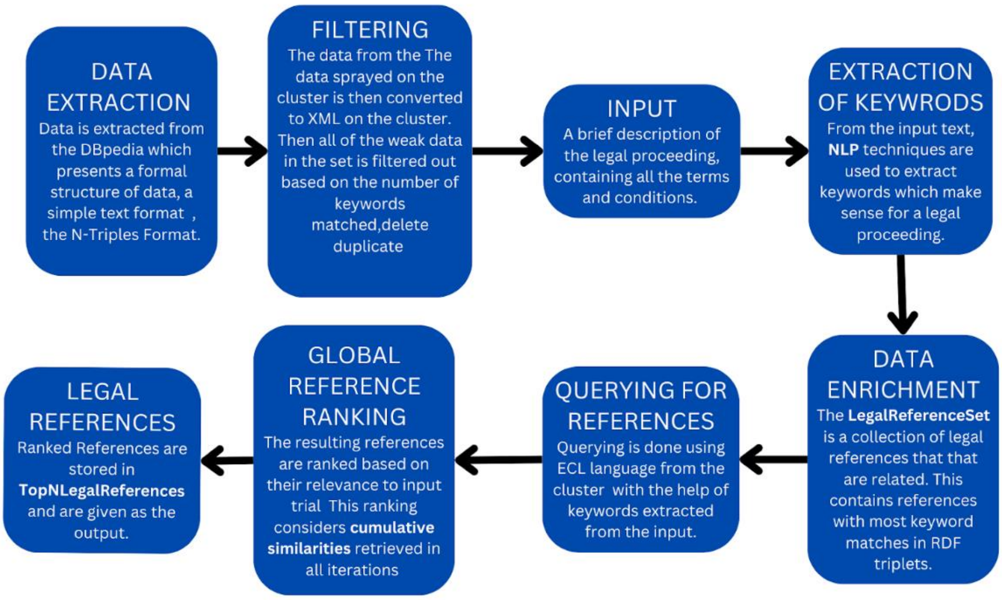


Fig-2: DBpedia Extraction Mechanism

Methodology for proposed recommendation system:


DATA EXTRACTION
Data is extracted from the DBpedia which presents a formal structure of data, a simple text format , the N-Triples Format.
FILTERING
The data from the The data sprayed on the cluster is then converted to XML on the cluster. Then all of the weak data in the set is filtered out based on the number of keywords matched,delete duplicate
INPUT
A brief description of the legal proceeding, containing all the terms and conditions.
EXTRACTION OF KEYWRODS
From the input text, NLP techniques are used to extract keywords which make sense for a legal proceeding.
DATA ENRICHMENT
The LegalReferenceSet is a collection of legal references that that are related. This contains references with most keyword matches in RDF triplets.
QUERYING FOR REFERENCES
Querying is done using ECL language from the cluster with the help of keywords extracted from the input.
GLOBAL REFERENCE RANKING
The resulting references are ranked based on their relevance to input trial This ranking considers cumulative similarities retrieved in all iterations
LEGAL REFERENCES
Ranked References are stored in TopNLegalReferences and are given as the output.

Dataset and Performance Analysis:

Example: M. C. Mehta v. Kamal Nath: ***M. C. Mehta v. Kamal Nath*** was a landmark case in Indian environmental law. In the case, the **Supreme Court of India** held that the **public trust doctrine** applied in India.

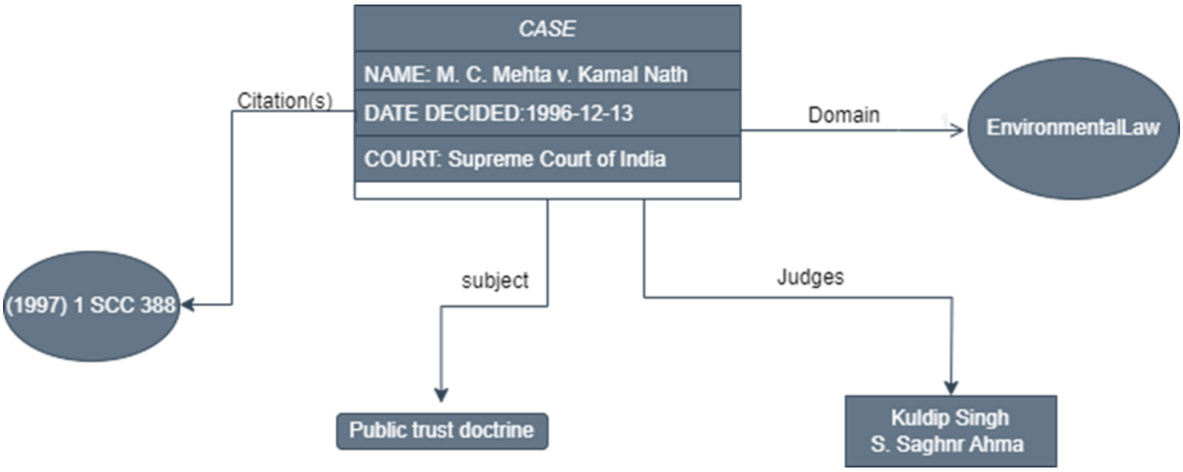


Fig - 3: Figurative Representation of subject, predicate and object

Predicates are specific resources which can be used to describe attributes or relations of other resources, in fig-4 the judges, domain are all Predicates. Statements are used to describe properties of a subject in the format subject, predicate, object. In fig-4 Our subject is the case M. C. Mehta v. Kamal Nath, predicates are citation, domain, subject and judges and their respective objects are given in the figure.

Data Extraction:

The first step in this approach is to extract data from DBpedia and recognise keywords from the case summary. Extracted data is in the form of RDF triples. The data we extract contains only legal references from all parts of the world from DBpedia.

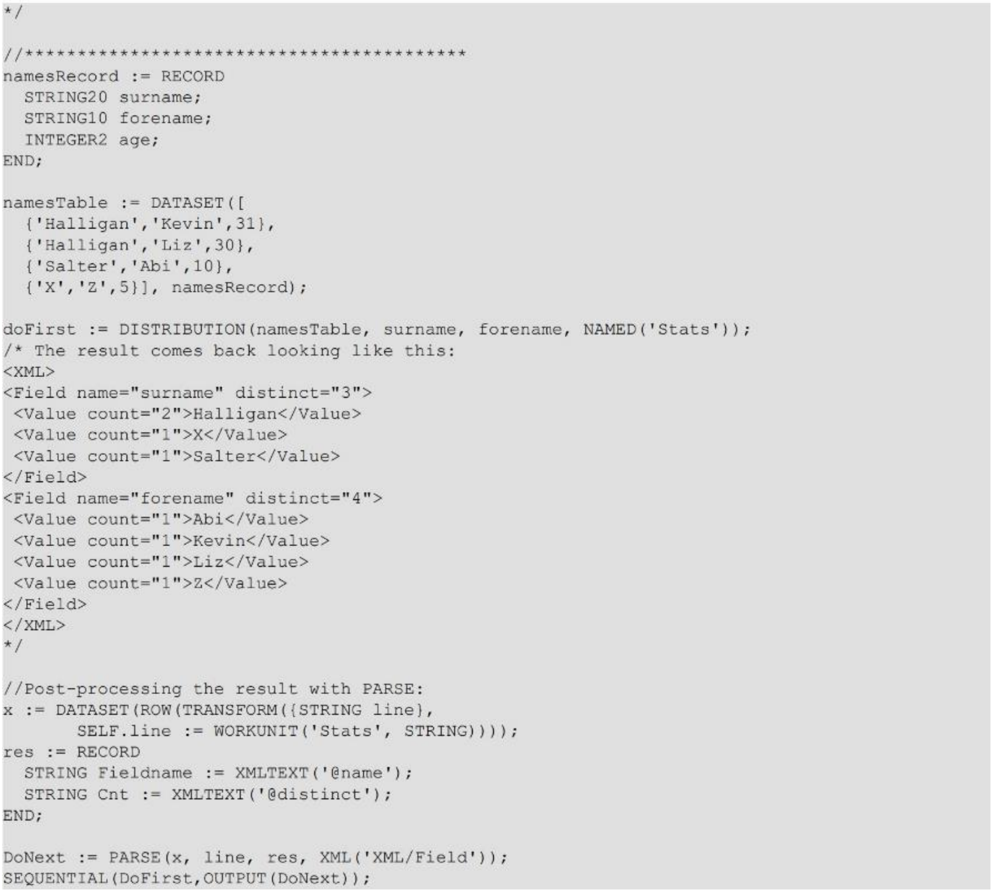
There are several formats for serialising RDF data. A simple text format is the N-Triples Format. RDF has several basic elements, namely resources (also called entities or subjects), properties (also predicates or roles), and statements (also triples or facts or objects) Subject comprise any concrete or abstract entities, e.g., a book or a person. Every Subject is identified by a Uniform Resource Identifier (URI) which is guaranteed to be globally unique. The extracted data is then sprayed on THOR for computation, this is done using rdf2hpcc.

Filtering the RDF file and formatting it to XML for computation:

The data sprayed on the cluster is then converted to XML on the cluster. We use rdf2hpcc to ingest rdf data, rdf2hpcc is based on Apache Jena which also has the functionality to convert it to XML. The cluster we are working with here is THOR. Then all of the weak data in the set is filtered out based on the number of keywords matched, delete duplicates etc. This is done using functions in ECL.

For example, DISTRIBUTION () is used to filter out duplicates from then data

In the below example the dataset consists of surname, forename and age. The name ‘Halligan’ is repeated twice. DISTRIBUTION function gives the count of each name i.e., if a name is repeated twice (‘Halligan’ in this case) count will be 2. This function helps us identify duplicates. This function also gives the number of unique records of each type. Transform and Parse functions are then used to format the output. Output will be as shown in the comments in the below code in the form of XML.



```
*/

//******************************************
namesRecord := RECORD
  STRING20 surname;
  STRING10 forename;
  INTEGER2 age;
END;

namesTable := DATASET([
  {'Halligan','Kevin',31},
  {'Halligan','Liz',30},
  {'Salter','Abi',10},
  {'X','Z',5}], namesRecord);

doFirst := DISTRIBUTION(namesTable, surname, forename, NAMED('Stats'));
/* The result comes back looking like this:
<XML>
<Field name="surname" distinct="3">
 <Value count="2">Halligan</Value>
 <Value count="1">X</Value>
 <Value count="1">Salter</Value>
</Field>
<Field name="forename" distinct="4">
 <Value count="1">Abi</Value>
 <Value count="1">Kevin</Value>
 <Value count="1">Liz</Value>
 <Value count="1">Z</Value>
</Field>
</XML>
*/

//Post-processing the result with PARSE:
x := DATASET(ROW(TRANSFORM({STRING line},
       SELF.line := WORKUNIT('Stats', STRING))));
res := RECORD
  STRING Fieldname := XMLTEXT('@name');
  STRING Cnt := XMLTEXT('@distinct');
END;

DoNext := PARSE(x, line, res, XML('XML/Field'));
SEQUENTIAL(DoFirst,OUTPUT(DoNext));
```

Fig-3: Code snippets used to filter out duplicates from the records. https://cdn.hpccsystems.com/releases/CE-Candidate-9.4.4/docs/EN_US/ECLLanguageReference_EN_US-9.4.4-1.pdf

Input and keyword searching:

Input will be a brief description of the legal proceeding. For example

In the legal case of **[Case Name]**, the plaintiff, **[Plaintiff Name]**, filed a lawsuit against the defendant, **[Defendant Name]**, alleging patent infringement. After careful deliberation, the court ruled in favour of the plaintiff, revoking the defendant's patent and awarding damages. This decision marks a significant victory for intellectual property rights. The input will be taken from the user with the help of python.

From this paragraph all the relevant keywords are extracted using NLP techniques. Some examples of these keywords above are ‘legal case’, ‘defendant’, ‘revoking’, ‘intellectual property rights. This will be done by embedding python NLP. The NLP model will also sort the keyword based on its relevance i.e., if it is year, type of case (criminal, civil, PIL, frauds). Then it is formatted as XML output.

Giving the keywords set to the cluster:

The XML output is compared with our dataset using ROXIE query as it faster than THOR. We then find matches of the keyword with our dataset. After matches are found we rank the found matches based on relevance.

Data Enrichment and Legal References:

The LegalReferenceSet has a group of legal citations that are associated with the keywords that were taken from the input (the LegalReferenceSet and input document share keywords). LegalReferenceSet is ordered according to how many times the appropriate legal papers mention each keyword from Input Key-Words. The TopNLegalReferenceSet contains the top-ranked references. The ontology is queried to match the keywords for each reference in the TopNLegalReferenceSet. ExtendedLegalReferenceSet is used to hold the outcomes.

Based on the number of references in TopNLegalReferenceSet that cite each reference in ExtendedLegalReferenceSet, ExtendedLegalReferenceSet is then rated. The TopNExtendedLegalReferenceSet is expanded with the high-ranking references. The most relevant legal references are in TopNExtendedLegalReferenceSet.

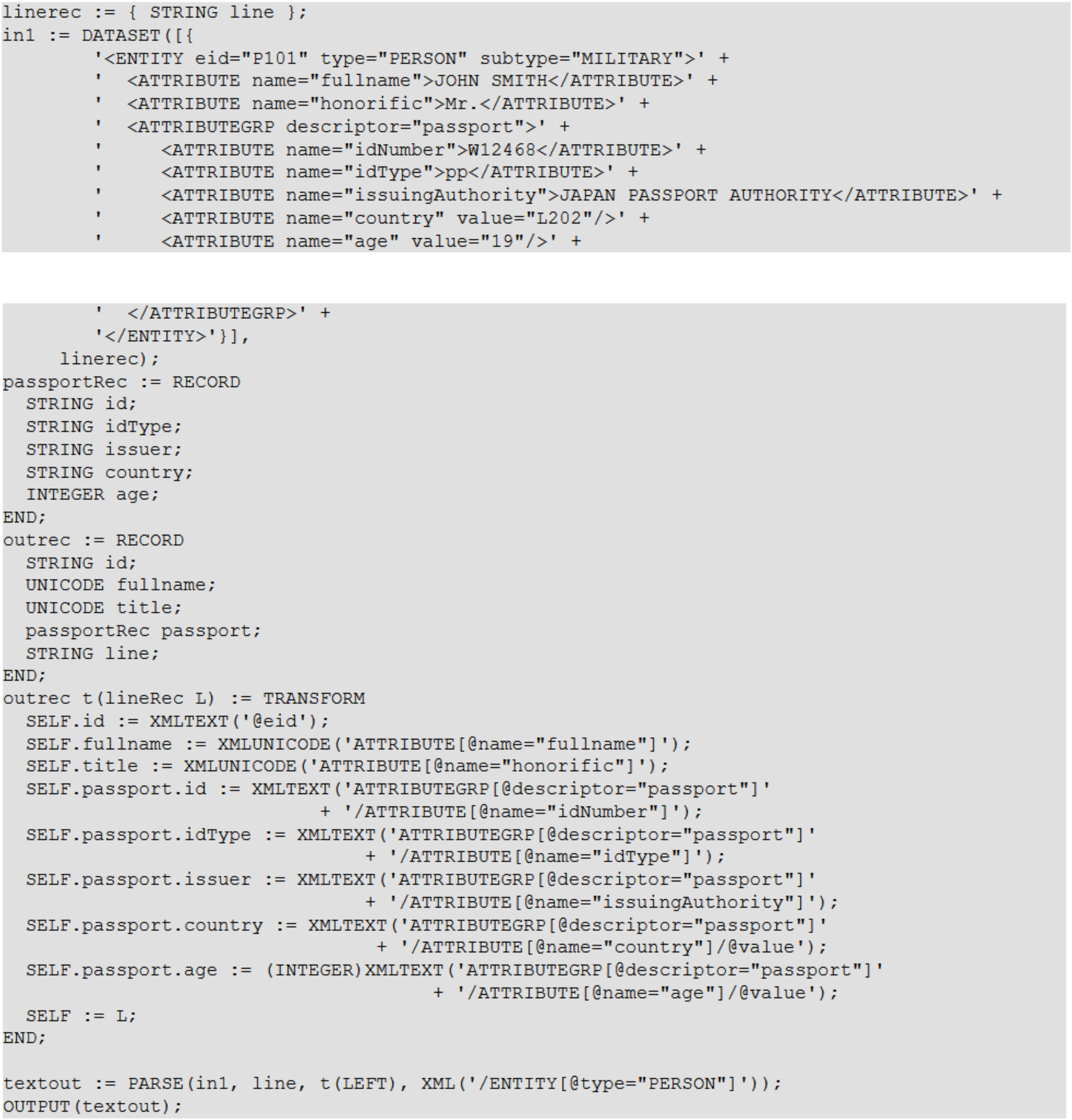
This permits data enrichment with terms not previously present in the InputKeyWords that were cited in legal papers in TopNExtendedLegalReferenceSet and Top- NReferenceSet. The procedure is repeated based on Depth, a user-defined parameter that regulates the algorithm's iterations rhythm. each iteration that follows. The new InputKeyWords are taken from the ExtendedKeywordSet. Then the procedure is repeated to access the ontology and gather ancillary data. The actions of the flowchart provide examples of the algorithm.

Querying:

From the formatted XML output we use the PARSE () function to find the matching entries with the keywords present in our database.

The PARSE () function operates on an XML dataset, parsing the XML data and creating a result set using the result parameters. The PARSE function implements the parsing operation.

It returns a record set that may then be post-processed as needed using standard ECL syntax, or simply output.



```
linerec := { STRING line };
in1 := DATASET([{
        '<ENTITY eid="P101" type="PERSON" subtype="MILITARY">' +
        '  <ATTRIBUTE name="fullname">JOHN SMITH</ATTRIBUTE>' +
        '  <ATTRIBUTE name="honorific">Mr.</ATTRIBUTE>' +
        '  <ATTRIBUTEGRP descriptor="passport">' +
        '     <ATTRIBUTE name="idNumber">W12468</ATTRIBUTE>' +
        '     <ATTRIBUTE name="idType">pp</ATTRIBUTE>' +
        '     <ATTRIBUTE name="issuingAuthority">JAPAN PASSPORT AUTHORITY</ATTRIBUTE>' +
        '     <ATTRIBUTE name="country" value="L202"/>' +
        '     <ATTRIBUTE name="age" value="19"/>' +
        '  </ATTRIBUTEGRP>' +
        '</ENTITY>'}],
     linerec);
passportRec := RECORD
  STRING id;
  STRING idType;
  STRING issuer;
  STRING country;
  INTEGER age;
END;
outrec := RECORD
  STRING id;
  UNICODE fullname;
  UNICODE title;
  passportRec passport;
  STRING line;
END;
outrec t(lineRec L) := TRANSFORM
  SELF.id := XMLTEXT('@eid');
  SELF.fullname := XMLUNICODE('ATTRIBUTE[@name="fullname"]');
  SELF.title := XMLUNICODE('ATTRIBUTE[@name="honorific"]');
  SELF.passport.id := XMLTEXT('ATTRIBUTEGRP[@descriptor="passport"]'
                          + '/ATTRIBUTE[@name="idNumber"]');
  SELF.passport.idType := XMLTEXT('ATTRIBUTEGRP[@descriptor="passport"]'
                              + '/ATTRIBUTE[@name="idType"]');
  SELF.passport.issuer := XMLTEXT('ATTRIBUTEGRP[@descriptor="passport"]'
                              + '/ATTRIBUTE[@name="issuingAuthority"]');
  SELF.passport.country := XMLTEXT('ATTRIBUTEGRP[@descriptor="passport"]'
                               + '/ATTRIBUTE[@name="country"]/@value');
  SELF.passport.age := (INTEGER)XMLTEXT('ATTRIBUTEGRP[@descriptor="passport"]'
                                    + '/ATTRIBUTE[@name="age"]/@value');
  SELF := L;
END;

textout := PARSE(in1, line, t(LEFT), XML('/ENTITY[@type="PERSON"]'));
OUTPUT(textout);
```

Fig - 4: example of PARSE () function on XML data which is filtering entity of type='PERSON' https://cdn.hpccsystems.com/releases/CE-Candidate-9.4.4/docs/EN_US/ECLLanguageReference_EN_US-9.4.4-1.pdf

Global Reference Ranking:

A global rating of references, GlobalReferenceRank, is updated on each cycle to gauge how relevant each reference is to the incoming legal document. This ranking provides a thorough assessment of the importance of each reference by taking into account the cumulative similarity of the information retrieved over all rounds. The user is shown the GlobalReferenceRank when all iterations have been completed. The most pertinent legal antecedents are represented by the highest-ranking legal references in the GlobalReferenceRank, which are documents that are most similar to the input legal document. The algorithm allows for wide-ranging and thorough semantic-based depth traversal of the ontology. The user-defined parameter alone does not determine the depth of traversal.

Algorithm:

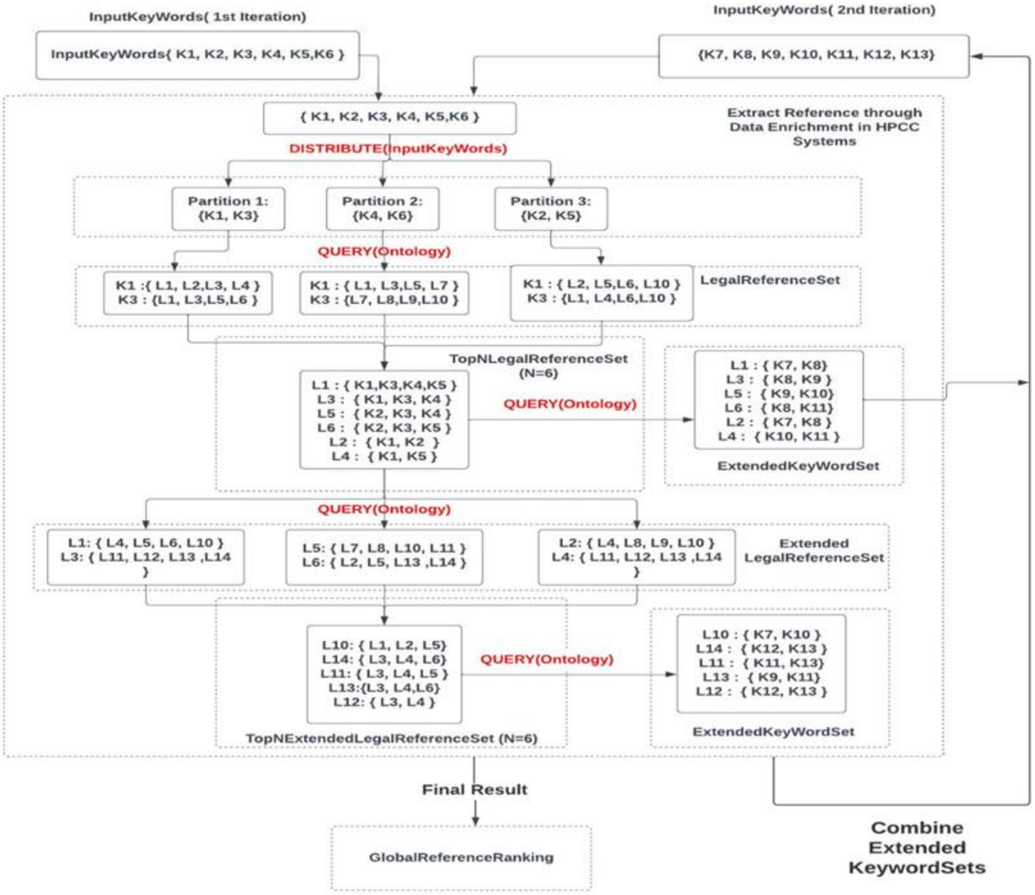

InputKeyWords(1st Iteration)
InputKeyWords{ K1, K2, K3, K4, K5,K6 }
InputKeyWords(2nd Iteration)
{K7, K8, K9, K10, K11, K12, K13}
{ K1, K2, K3, K4, K5,K6 }
Extract Reference through Data Enrichment in HPCC Systems
DISTRIBUTE(InputKeyWords)
Partition 1: {K1, K3}
Partition 2: {K4, K6}
Partition 3: {K2, K5}
QUERY(Ontology)
K1 :{ L1, L2,L3, L4 } K3 : {L1, L3,L5,L6 }
K1 : { L1, L3,L5, L7 } K3 : {L7, L8,L9,L10 }
K1 : { L2, L5,L6, L10 } K3 : {L1, L4,L6,L10 }
LegalReferenceSet
TopNLegalReferenceSet (N=6)
L1 : { K1,K3,K4,K5 } L3 : { K1, K3, K4 } L5 : { K2, K3, K4 } L6 : { K2, K3, K5 } L2 : { K1, K2 } L4 : { K1, K5 }
QUERY(Ontology)
L1 : { K7, K8} L3 : { K8, K9 } L5 : { K9, K10} L6 : { K8, K11} L2 : { K7, K8 } L4 : { K10, K11 }
ExtendedKeyWordSet
QUERY(Ontology)
L1: { L4, L5, L6, L10 } L3: { L11, L12, L13 ,L14 }
L5: { L7, L8, L10, L11 } L6: { L2, L5, L13 ,L14 }
L2: { L4, L8, L9, L10 } L4: { L11, L12, L13 ,L14 }
Extended LegalReferenceSet
L10: { L1, L2, L5} L14: { L3, L4, L6} L11: { L3, L4, L5 } L13:{L3, L4,L6} L12: { L3, L4 }
QUERY(Ontology)
L10 : { K7, K10 } L14 : { K12, K13 } L11 : { K11, K13} L13 : { K9, K11} L12 : { K12, K13 }
TopNExtendedLegalReferenceSet (N=6)
ExtendedKeyWordSet
Final Result
GlobalReferenceRanking
Combine Extended KeywordSets

Fig 5. Representation of the algorithm used

The algorithm in fig 3 accepts a case summary as input and discovers keywords—represented as K1, K2, etc.—in the input. The system then searches the DBpedia for cases that are the most comparable to the keywords and prioritises them from most to least important.

References

[1] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z. (2007). *DBpedia: A Nucleus for a Web of Open Data*. In: Aberer, K., et al. The Semantic Web. ISWC ASWC 2007 2007. Lecture Notes in Computer Science, vol 4825. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-76298-0_52

[2] Y. Leng, Z. Chen, F. Zhong and H. Zhong, "*BRDPHHC: A Balance RDF Data Partitioning Algorithm Based on Hybrid Hierarchical Clustering,"* 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems, New York, NY, USA, 2015, pp. 1755-1760, doi: https://10.1109/HPCC-CSS-ICESS.2015.190.

[3]Lehmann, Jens et al. '*DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia*'. 1 Jan. 2015 : 167 – 195

[4]Passant, A. (2010). dbrec — *Music Recommendations Using DBpedia*. In: Patel-Schneider, P.F., *et al.* The Semantic Web – ISWC 2010. ISWC 2010. Lecture Notes in Computer Science, vol 6497. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-17749-1_14

[5]V. V. Kindratenko *et al.*, "GPU clusters for high-performance computing," *2009 IEEE International Conference on Cluster Computing and Workshops*, New Orleans, LA, USA, 2009, pp. 1-8, doi: https://10.1109/CLUSTR.2009.5289128

[6]Stranieri, A., & Zeleznikow, J. (2006). *Knowledge discovery from legal databases* (Vol. 69). Springer Science & Business Media.

[7] Jeremy J. Carroll and Patrick Stickler. 2004. RDF triples in XML. In Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters (WWW Alt. '04). Association for Computing Machinery, New York, NY,USA,412–413. https://doi.org/10.1145/1013367.1013501

[8]https://cdn.hpccsystems.com/releases/CE-Candidate9.4.4/docs/EN_US/ECLLanguageReference_EN_US-9.4.4-1.pdf

TIMELINE	
December	Engaging in the exploration of HPCC systems architecture and mastering the ECL programming language.
1st January - 10th January	Understanding the working of NLP and choosing the most suitable algorithm.
11th January - 30th January	Data Extraction from DBpedia and converting RDF to XML. Filtering the XML data.
31st January - 15th April	Development of NLP using Python.
16th April - 10th May	Development of Ranking System, Integrating NLP and Ranking System.
11th May - 20th May	Refining and fine-tuning the entire project through meticulous testing and optimization.
Till End	Writing Research Paper.

Knowledge and Skills
Familiarity with HPCC Architecture and ECL language. Familiarity with DBpedia, RDF, XML. Proficient in Python, Java.

Nihar Mandahas

RVCE

CONTACT

9482808895

niharmandahas.cs22@rvce.edu.in

Bengaluru

Nihar Mandahas

EDUCATION

Bachelor of Engineering in Computer Science and Engineering

- **R.V college of engineering 2022-present**

SKILLS

- Python
- ECl
- C programming
- Java Script
- Using creative software tools

PROFILE

Enthusiastic and dedicated B.E. in Computer Science student at RVCE, passionate about leveraging cutting-edge technology to drive innovation and solve real-world problems. Eager to apply my technical skills, creativity, and strong work ethic to contribute effectively to the ever-evolving world of computer science.

EXPERIENCE

During my work experience, I excelled in Python, ECL, and JavaScript, applying these skills to develop efficient solutions. Python enabled versatile problem-solving, while ECL optimized data processing. JavaScript empowered me to create user-friendly web applications, enriching project outcomes.

PROJECTS

- Interactive map-based solution using the Leaflet framework in js to display and navigate EV charging station locations.
- Leveraged NLP to seamlessly convert logic statements in Kannada into Python code, bridging language barriers in programming.
- Executed a Python project encompassing data scraping and visualization, offering valuable insights from movie data through informative visuals.
- Engineered a hardware project that involved extracting battery levels from a battery, followed by uploading this data to the cloud for in-depth analysis, enhancing efficiency and data-driven decision making.

REFERENCE

https://github.com/Pratheekrao/EV-Charger-Locator

LANGUAGES

English	● ● ● ● ○
Kannada	● ● ● ● ●
Hindi	● ● ● ○ ○

CONTACT

☎ 8660501765

✉ mppratheek@gmail.com

Bengaluru

in Pratheek Rao

EDUCATION

Bachelor of Engineering in Computer Science and Engineering

- R.V college of engineering 2022-present

SKILLS

- Java
- HTML and CSS
- C
- JavaScript
- ECL

Pratheek Rao MP

RVCE

PROFILE

As an ambitious and hard-working computer science engineering student, I am dedicated to pursuing excellence in both my academic and professional endeavors. With a passion for technology and a relentless drive to overcome challenges, I am committed to continuous learning and growth in the dynamic field of computer science. My strong work ethic, coupled with my determination to achieve goals, positions me to excel in collaborative projects and contribute meaningfully to the ever-evolving world of technology

EXPERIENCE

I have gained valuable experience in Java during my academic studies and practical projects. I am proficient in Java programming and have utilized it to develop various applications,My experience with JavaScript includes web development and front-end scripting. I am skilled in using JavaScript to enhance the user experience on websites, creating interactive and dynamic web applications.

PROJECTS

- Engineered a hardware project that involved extracting battery levels from a battery, followed by uploading this data to the cloud for in-depth analysis, enhancing efficiency and data-driven decision making.

- Interactive map-based solution using the Leaflet framework in js to display and navigate EV charging station locations in Bengaluru.

REFERENCE

- EV Charger Locator in Bengaluru
- Cloud Battery Management system

LANGUAGES

English	● ● ● ● ●
Kannada	● ● ● ● ○
Hindi	● ● ● ● ○

Manvith L B

Computer Science Undergrad

CONTACT

☎ 9426949530

in linkedin.com/in/manvithlb

✉ manvithlb.cs22@rvce.edu.in

📍 Bengaluru , Karnataka

EDUCATION

BE. Computer Science and Engineering

- RV College of Engineering
  2022-2026

12th CBSE

- ATOMIC ENERGY CENTRAL SCHOOL, MYSORE
  2021-2022

SKILLS

- Django
- Data visualization using Python
- ECL
- Javascript
- React

PROFILE

Complex problem-solver with analytical and driven mindset. Dedicated to achieving demanding development objectives according to tight schedules while producing impeccable code.

EXPERIENCE

During my work experience, I excelled in Python, ECL, and JavaScript, applying these skills to develop efficient solutions. Python enabled versatile problem-solving, while ECL optimized data processing. JavaScript empowered me to create user-friendly web applications, enriching project outcomes.

PROJECTS

- Interactive map-based solution using the Leaflet framework in js to display and navigate EV charging station locations.
- utilized Natural Language Processing (NLP) to effortlessly transform logic statements written in Kannada into Python code, effectively bridging language barriers in the field of programming.
- I successfully conducted a Python project that encompassed both data scraping and visualization, presenting valuable insights derived from inflation data of diverse countries through informative visual representations.
- designed a hardware project that entailed extracting battery levels from a battery and subsequently uploading this data to the cloud for thorough analysis. This initiative aimed to improve efficiency and facilitate data-driven decision-making.

REFERENCE

https://github.com/Pratheekrao/EV-Charger-Locator

LANGUAGES

English	● ● ● ● ○
kannada	● ● ● ● ○
hindi	● ● ● ● ●