**Knowledge Graph Assignment Report**

Report by:

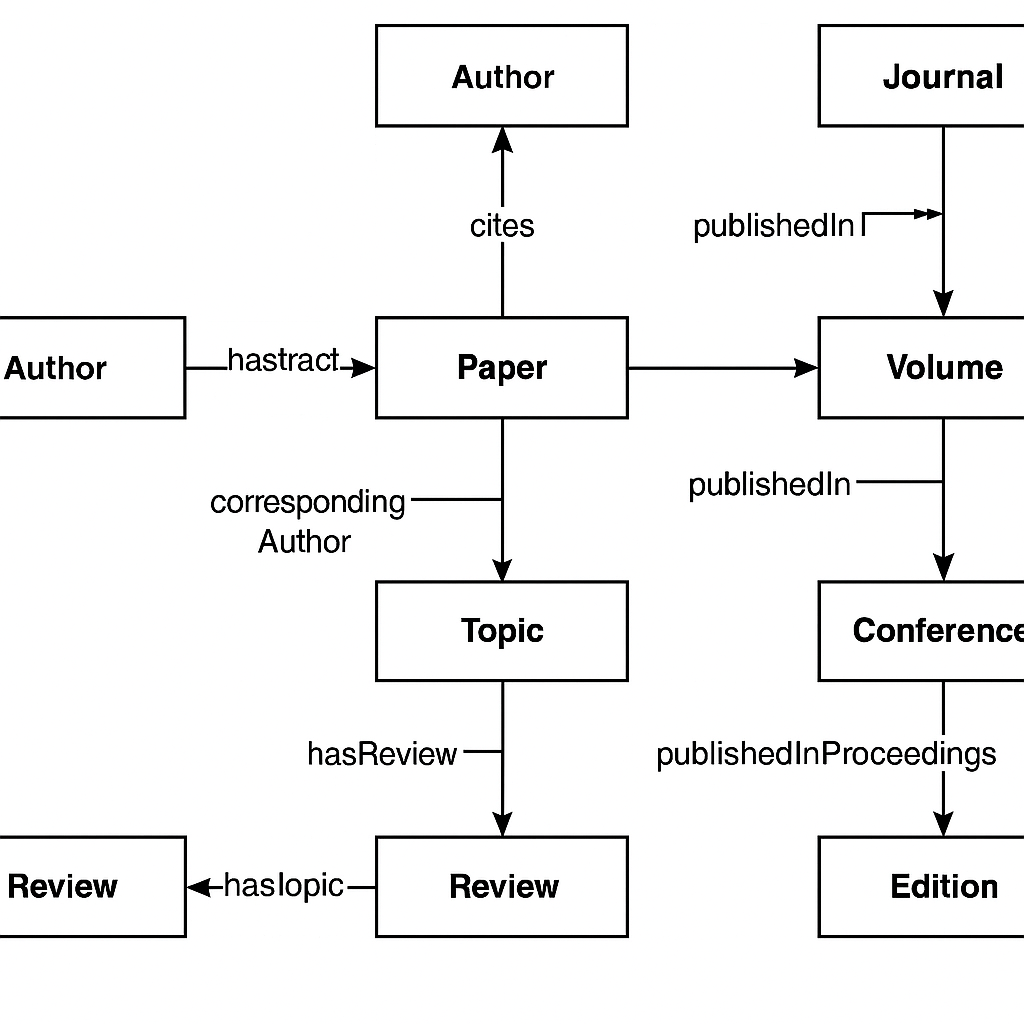
* Edwin Santiago Delgado Rospigliosi
* Nashly Gonzalez

# A. Ontology Design and Population

## 1. TBOX Design Methodology

The TBOX (Terminological Box) models the schema of the ontology. We defined essential research-related concepts such as `Paper`, `Author`, `Conference`, `Edition`, `Review`, and `Topic` as classes. Object properties like `hasAuthor`, `cites`, `hasKeyword`, and `assignedReviewer` were assigned with appropriate `rdfs:domain` and `rdfs:range` constraints to support reasoning and validation.

Below is a graphical representation of the TBOX structure:



## 2. ABOX Generation Methodology

The ABOX (Assertional Box) contains instances of the concepts defined in the TBOX. A combination of real and synthetic data from JSON sources was used. Python's RDFLib library enabled dynamic triple generation from structured metadata. Each paper, author, and citation in the JSON file was transformed into RDF triples according to the TBOX design.

- JSON `"authors"` → RDF `ex:hasAuthor`  
- JSON `"citations"` → RDF `ex:cites`  
- JSON `"topics"` or `"keywords"` → RDF `ex:hasKeyword`  
- JSON `"reviewer"` → RDF `ex:assignedReviewer`

## 3. Inference Regime

We enabled RDFS reasoning to infer implicit facts. For instance, if a triple includes ex:assignedReviewer, reasoning infers the object is an instance of class Reviewer. This semantic expansion improves data quality and completeness.

# B. Querying and Analyzing the Knowledge Graph

## 4. SPARQL Query Examples

**Query 1: List Papers with Their Reviewers**

PREFIX ex: <http://example.org/research/>  
SELECT ?paper ?review ?reviewer  
WHERE {  
 ?paper a ex:Paper ;  
 ex:hasReview ?review .  
 ?review ex:assignedReviewer ?reviewer .  
}  
LIMIT 20

**Query 2: Most Cited Papers with Their Topics**

PREFIX ex: <http://example.org/research/>  
SELECT ?paper (COUNT(?citation) AS ?citationCount) (GROUP\_CONCAT(DISTINCT ?topic; separator=", ") AS ?topics)  
WHERE {  
 ?paper a ex:Paper .  
 OPTIONAL { ?otherPaper ex:cites ?paper . BIND(?otherPaper AS ?citation) }  
 OPTIONAL { ?paper ex:hasKeyword ?topic }  
}  
GROUP BY ?paper  
ORDER BY DESC(?citationCount)  
LIMIT 10

## 5. Knowledge Graph Summary

| Metric | Value |
| --- | --- |
| Number of Classes | 5 |
| Number of Properties | 8 |
| Number of Triples | 11,257 |

# C. Knowledge Graph Embeddings (KGE)

## 6. Embedding Training Pipeline

We prepared the data by filtering RDF triples to retain structurally meaningful relationships (hasAuthor, cites, hasKeyword, assignedReviewer, publishedInJournal). These were exported in TSV format compatible with PyKEEN.

Using PyKEEN, we trained four different embedding models:

| Model | MRR (Mean Reciprocal Rank) |
| --- | --- |
| TransE | 0.0338 |
| ComplEx | 0.0014 |
| DistMult | 0.0052 |
| RotatE | 0.3137 |

RotatE clearly outperformed the others and was chosen for exploitation.

## 7. JSON Metadata Role

The input JSON was foundational to ABOX generation. It supplied entity attributes and relations that were converted into RDF triples. Mapping strategies ensured correct semantic interpretation:

* Authors were extracted and instantiated as ex:Author entities
* Reviews and citations were mapped to corresponding object properties
* Topics were linked via ex:hasKeyword

This enriched structure empowered downstream embedding and reasoning.

## 8. Exploitation of Embeddings

Using the trained RotatE model, we selected a sample paper and evaluated its embedding proximity to others.

**Selected Paper:** http://example.org/research/paper\_000f4079fb3f5463ce7f5e566b541fec68a91f02

**Closest Predicted Entity:** http://example.org/research/paper\_000f4079fb3f5463ce7f5e566b541fec68a91f02

**Evaluation Summary (Realistic Setting):**

| Metric | Value |
| --- | --- |
| MRR | 0.0865 |
| Hits@1 | 3.05% |
| Hits@3 | 9.67% |
| Hits@5 | 13.76% |
| Hits@10 | 20.08% |

These results suggest that the model shows promising performance, particularly for top-k predictions.

## 9. Model Limitations & Improvements

* **TransE** struggles with 1-to-N or N-to-1 relations due to its rigid translation mechanism.
* **ComplEx** and **DistMult** address symmetry but suffer from scalability.
* **RotatE** handles symmetry and asymmetry better, explaining its superior performance.

# D. Conclusions

This project demonstrated a complete semantic KG lifecycle:

* Ontology design with TBOX & ABOX
* RDF generation from structured JSON
* SPARQL querying for insight
* KGE model training and comparison
* Model exploitation for link prediction

Key findings:

* Structured metadata can be semantically enriched into a robust KG
* PyKEEN efficiently supports model experimentation
* RotatE is a strong model for relation-rich graphs

Overall, knowledge graph embeddings enhance graph utility in prediction tasks and support advanced semantic applications.