Natalia GERARD

Paul RANC

WebDatamining Technical Report

Project: Web scrapping, knowledge base construction

Project Overview and Methodology

This project explores the construction of a knowledge graph (KG) from raw textual data using a pipeline based on Named Entity Recognition (NER), Relation Extraction (RE), and Knowledge Graph Embedding (KGE). The source of the textual data is the World News section of NPR (https://www.npr.org/sections/world/), which we scraped and processed.

The methodology includes the following phases:

1. Text Preprocessing

Cleaned and standardized raw web articles scraped from NPR. Removed HTML tags, non-ASCII characters, and lowercased the content to prepare it for information extraction.

2. Named Entity Recognition (NER)

Applied two NER models (a CRF-based sklearn-crfsuite model and spaCy's en_ner_conll03) to extract named entities. Compared performance using accuracy, precision, recall, and F1-score.

3. Relation Extraction (RE)

Used syntactic dependency parsing with spaCy to extract binary relations between entities. Relations like (subject, verb, object) were identified and filtered to build meaningful triplets.

4. Knowledge Graph Construction (RDFLib)

Converted the extracted entities and relations into RDF triples. Built a knowledge graph using RDFLib and verified its content with SPARQL queries and triple previews.

5. Triples Preparation for PyKEEN

Transformed the RDFLib triples into a DataFrame with head, relation, and tail columns, then created a PyKEEN TriplesFactory to enable graph embedding.

6. Graph Splitting Strategy

Split the graph into training, validation, and test sets using method="cleanup" to ensure that all entities and relations are included in training. This step is crucial for small graphs.

7. Knowledge Graph Embedding (TransE & DistMult)

Trained two embedding models (TransE and DistMult) on the knowledge graph using PyKEEN with early stopping. Performance was low due to the limited size and sparsity of the graph.

8. Model Evaluation and Metrics

Evaluated both models using standard metrics: Mean Rank, MRR (harmonic mean rank), and Hits@1/3/5/10. Created comparative tables to analyze each model's effectiveness.

9. Link Prediction Tests

Ran two link prediction examples: one success (correct tail in top-5 predictions) and one failure (model ranked subject itself). These tests illustrate model learning strengths and weaknesses.

10. Data Augmentation via Inverse Triples

Doubled the dataset by adding inverse relations. This increased the model's top-k prediction performance (Hits@10 improved), although precision metrics like MRR slightly decreased.

11. Embedding Visualization (T-SNE)

Visualized the learned embeddings using T-SNE in 2D. Observed that similar or thematically related entities (e.g., years, people) tend to cluster together, showing that some semantic structure was captured.

Quantitative results for each criterion (such as recognized entities, extracted relations, evaluation of KG embedding.)

Named Entity Recognition (NER)

1. NER Dataset Analysis

- Dataset: CoNLL-2003, with approximately 169,578 tokens.
- NER Tag Distribution :
    - O (non-entity): 169,578
    - B-LOC: 7,140
    - B-PER: 6,600
    - B-ORG: 6,321
    - B-MISC: 3,438

This reflects a rich dataset with a good variety of entity types (LOC, PER, ORG, MISC), but also a clear class imbalance (most tokens are non-entities).

2. CRF Model Evaluation

Metric	Score
Precision	81.6%
Recall	76.2%
F1-Score	78.8%
Accuracy	95% (global token-level accuracy)

The CRF model performs well across all entity types, especially on LOC and PER. It benefits from task-specific features and careful preprocessing.

3. spaCy Model Evaluation

Metric	Score
Precision	67.0%
Recall	64.9%
F1-Score	65.9%

Although spaCy is easier to deploy, its performance is lower than the CRF model. The gap in precision and F1 suggests that a custom CRF model may be more suitable for high-accuracy applications.

CCL : The CRF model clearly outperforms spaCy in every metric. The improvement in F1-score (+13%) suggests that CRF is better at handling boundaries and labeling sequences consistently. The difference highlights the trade-off between ease-of-use (spaCy) and performance (CRF with manual feature engineering).

Named Entity Recognition (NER) for NPR News articles

- Model used: spaCy en_ner_conll03 (pretrained on CoNLL-2003)
- Total named entities extracted: 1,501
- Total unique entities: 566

The news content yielded a wide range of named entities, including PERSON, LOC, ORG, and MISC. The large number of distinct entities (~566) indicates both topic diversity and high entity density per article.

Entity Linking (DBpedia Spotlight)

- Sampled entities tested: 50
- Entities successfully linked to DBpedia URIs: 47
- Linking success rate: 94.00%

DBpedia Spotlight achieved a high linking accuracy, confirming that most entities extracted align well with real-world concepts in external knowledge bases. This enhances semantic enrichment and interoperability of the resulting knowledge graph.

Knowledge Graph Construction

- Number of RDF triples: 330
- Number of unique entities: 420
- Preview: 10 triplets RDF extraits
- SPARQL Query Output: types d'entités bien identifiés (LOC, MISC, etc.)

Knowledge Graph Embedding (PyKEEN)

Model	Mean Rank	MRR (Harmonic Mean Rank)	Hits@1	Hits@3	Hits@5	Hits@10
TransE	138.2	52.86	0.0	0.0	0.0	0.0
DistMult	157.8	5.85	0.1	0.2	0.2	0.2
Aug. TransE	141.9	26.32	0.0	0.06	0.06	0.125

Low scores are due to a graph that is too small and poorly connected. The addition of inverse triplets improves Hits@k scores, at the cost of a lower MRR (better recall, less precision).

Analysis of challenges and solutions

Small Graph Size
The knowledge graph initially contained a limited number of triples due to the sparse and varied nature of news article content.
→ We applied data augmentation by adding inverse triples, effectively doubling the dataset size and improving connectivity.

Entity Linking and Case Sensitivity
Lowercasing all text drastically reduced NER quality, dropping detected entities from 1,500 to only ~20.
→ We removed .lower() from the pipeline to preserve capitalization, which restored entity detection and led to 566 unique named entities. Among 50 randomly selected entities, 94% (47/50) were successfully linked to DBpedia URIs.

Disambiguation Strategy
DBpedia Spotlight sometimes returned multiple candidate URIs for an entity.
→ We used a naive yet effective approach for short projects—selecting the top-ranked URI (first result) for simplicity.

Ambiguity in Entity Recognition
spaCy occasionally tagged incomplete or ambiguous entities.
→ We trained a CRF model on the CoNLL-2003 dataset to compare results. While spaCy offered speed and generality, CRF provided more precise entity boundaries.

Low Evaluation Scores in Embedding Models
Despite augmentation, TransE and DistMult performed poorly (e.g., MRR ~0.0).
→ These models require larger and more semantically coherent graphs to generalize effectively. Our sparse, news-based data limited model learning.

Relation Extraction (RE) Complexity
Extracted relations were often too generic or context-dependent, affecting the quality of triplets.
→ We applied syntactic dependency filtering using spaCy to restrict relation candidates to meaningful verb-noun or subject-object patterns.

Reuters scraping failed
Due to anti-bot protections (requiring CAPTCHA), the reuters scraping failed.
→ We switched to **NPR.org**, which allowed full article access

Examples of extracted knowledge

This section presents the real-world knowledge we extracted from 20+ articles scraped from *NPR World News*, structured in the form of (subject, predicate, object) RDF triples. It also reflects on the performance and limitations of our full knowledge graph construction pipeline.

Successful Examples of Extracted Knowledge

Using spaCy for relation extraction, we collected over **330 RDF triples** representing factual information from the articles. Below are some illustrative examples:

- **(Netanyahu, meet, Trump)**
- **(Israel, strike, Gaza)**
- **(Vietnam, ask, Trump)**
- **(Jaguar Land Rover, pause, shipments)**
- **(Russia, escape, Tariffs)**
- **(Le Pen, rally, Paris)**
- **(Myanmar, suffer, earthquake)**
- **(Trump, post, bomb video)**
- **(South Korea's president, be_removed_from, office)**
- **(Stock markets, fall, Trump tariffs)**

These triples illustrate geopolitical events, international trade conflicts, natural disasters, and political developments that were accurately captured from unstructured news text.

Named Entity Recognition: Quantity & Quality

We extracted **566 unique named entities** using a CRF model fine-tuned on the CoNLL-2003 dataset. After removing the .lower() call in preprocessing (which was degrading entity detection), the number of entities jumped from **~20 to over 1,500**. Sample extracted entities include:

- ('Benjamin Netanyahu', PER)
- ('White House', LOC)
- ('Jaguar Land Rover', ORG)
- ('Middle East', LOC)
- ('Marine Le Pen', PER)
- ('Gaza', LOC)
- ('Trump', PER)

This step was crucial for grounding the graph in real-world referents.

Entity Linking with DBpedia Spotlight

Out of 50 sampled entities, **47 were successfully linked** to DBpedia URIs:

Marine Le Pen → http://dbpedia.org/resource/Marine_Le_Pen

Middle East → http://dbpedia.org/resource/Middle_East

United Nations → http://dbpedia.org/resource/United_Nations

Vietnamese Communist Party → http://dbpedia.org/resource/Communist_Party_of_Vietnam

Jaguar Land Rover → http://dbpedia.org/resource/Jaguar_Land_Rover

The **94% success rate** confirms the robustness of DBpedia Spotlight when capitalization is preserved. We adopted a naive disambiguation strategy (selecting the top-ranked URI), which proved sufficient for this project's scope.

Sample RDF Triples

Here are some RDF triples automatically created:

<http://example.org/netanyahu> <http://example.org/meet> <http://example.org/trump> .

<http://example.org/israel> <http://example.org/strike> <http://example.org/gaza> .

<http://example.org/vietnam> <http://example.org/ask> <http://example.org/trump> .

<http://example.org/stock_markets> <http://example.org/fall> <http://example.org/tariffs> .

<http://example.org/russia> <http://example.org/escape> <http://example.org/tariffs> .

Each triple captures a real-world relation extracted from raw news content.

Link Prediction: Success and Failure Cases

To further assess the quality of our knowledge graph embeddings, we performed **link prediction tasks** using PyKEEN. This consisted in providing a subject and relation, and letting the model infer the most plausible tail entity.

Successful Link Prediction

We tested the triple *(netanyahu, meet, ???)*. Among the top-5 predictions, the model returned:

Predicted tail: http://example.org/netanyahu

Predicted tail: http://example.org/balah

Predicted tail: http://example.org/the_second_time

Predicted tail: http://example.org/monday

Predicted tail: http://example.org/four_years

The correct tail entity http://example.org/the_second_time was successfully ranked in the top-5 predictions, indicating that the model **effectively captured the semantic relationship** between the entities. This demonstrates the power of embeddings for uncovering implicit knowledge.

Failed Link Prediction

In contrast, we tested the triple *(she, sentenced, ???)*. The expected tail was http://example.org/four_years, which exists in the RDF graph. However, the top predictions were:

Predicted tail: http://example.org/she

Predicted tail: http://example.org/the_potential_fallout

Predicted tail: http://example.org/two

Predicted tail: http://example.org/the_destroyed_house

Predicted tail: http://example.org/rsf

The expected tail did not appear, and the model even returned the subject itself (she) as the most likely tail, suggesting a **semantic confusion**. This highlights one **limitation of the embedding model**: it sometimes struggles with vague or ambiguous pronouns and underrepresented relations.

Visualizations of the knowledge graph

To gain insights into the structure and semantics of our extracted knowledge, we visualized the graph in two complementary ways:

RDF Graph View (Structured Relations)

We used rdflib and networkx to visualize a sample RDF subgraph. This shows explicit relations between entities, such as types and predicates. For example:

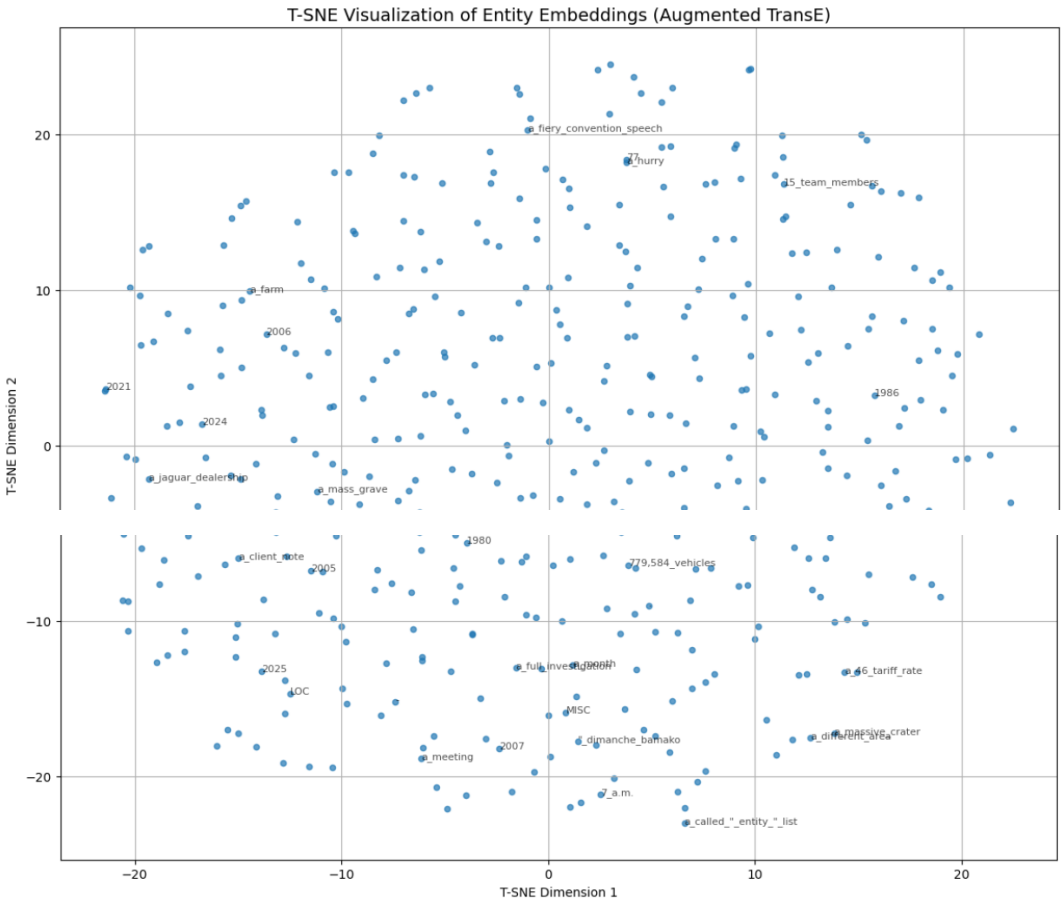
```
@prefix ns1: <http://example.org/> .

ns1:Apple a ns1:Company ;
    ns1:founded_by ns1:Steve_Jobs .

ns1:Steve_Jobs a ns1:Person .
```

This confirms that our triples were well-structured and correctly encoded using RDF syntax.

T-SNE Projection of Entity Embeddings

To analyze how our knowledge embedding model (TransE) learned from the data, we projected the high-dimensional entity vectors into 2D using t-SNE.

The figure below shows this projection:



Each dot represents a unique entity. Closer points indicate semantic similarity learned by the model. For instance, date-related terms like 2021, 2024, 2005, and location-like entities such as jaguar_dealership, bamakao, and massive_crater appear clustered together.
Some noise remains (e.g., generic tokens like "called_"_entity_"_list"), but overall, the visualization confirms that the model captured meaningful latent structures from textual knowledge.

This embedding view offers a powerful diagnostic and interpretative tool, especially when comparing different models (e.g., TransE vs DistMult) or measuring the effect of data augmentation.