TransE-Based Knowledge Graph Embedding:
Model Explanation and Evaluation

Rishikesh Yadav || Pruthvi Vishwakarma || Samuel Lee

6987520 4054853 4025098

Paderborn University

Abstract. This study explores the structural and semantic composition of the DBpedia knowledge base to support subsequent machine learning model development. Using SPARQL queries and the Protégé ontology editor, we identified over 400 distinct entity classes and more than 3,000 unique properties, illustrating DBpedia's rich and diverse schema. Our queries further revealed the dataset's emphasis on biographical, geographical, and cultural information. By combining query analysis with schema-level inspection, we delve into DBpedia's ontological consistency, interlinking structure.

Keywords: DBPedia · Knowledge Graphs · SparQL · TransE.

1 Introduction

Knowledge graphs like DBpedia offer structured and interlinked representations of information, making them fundamentally necessary to create better semantic web applications and AI systems. A clear understanding of DBpedia and its underlying structure is essential. This paper begins by examining DBpedia through the lens of entity types, properties, and their distributions. Using SPARQL for data-level querying and Protégé for schema-level inspection, we explain how the dataset is organized, its ontology design, and patterns that form.

2 Data Analysis

First we identify the different entity (node) types, the range of property relations (edge types), and the overall distribution. This exploration was initiated using SPARQL, the RDF query language, to extract the various classes assigned to entities. For instance, the following query retrieves a list of distinct RDF types used within the dataset:

```
SELECT DISTINCT ?type
WHERE { ?entity a ?type } LIMIT 100
```

Executing this query yielded around 400 distinct classes, including `dbo:Person`, `dbo:Place`, `dbo:Organisation`, `dbo:Work`, and `dbo:Species`. These figures align closely with the findings of Lehmann et al. (2015) [8], who also highlight the rich and diverse taxonomy encoded in DBpedia. Using the Protégé ontology editor, we visualized the hierarchical organization of these classes. For example, `dbo:Athlete` is a subclass of `dbo:Person`, which reflects a structured schema that supports property inheritance and type inference.

To explore object and data properties further, we used Protégé's interface to examine commonly used properties like `dbo:birthDate`, `dbo:populationTotal`, `dbo:areaTotal`,

and dbo:abstract. This analysis was complemented by the inclusion of FOAF properties such as foaf:name, often used for identifying people and organizations. Our SPARQL-based query:

```
SELECT DISTINCT ?property
WHERE { ?subject ?property ?object } LIMIT 100
```

revealed over 3,000 distinct properties across multiple namespaces (e.g., RDF Schema, FOAF, OWL, and DBpedia-specific vocabularies). This diversity demonstrates DBpedia's adherence to semantic web best practices (Auer et al., 2007; [1] Lehmann et al., 2015 [8]).

According to publicly available DBpedia statistics (Lehmann et al., 2015 [8]), the dataset encompasses approximately 6.6 million unique entities and more than 1.2 billion RDF triples. These are distributed across named graphs and subdomains, offering granularity of the view of data extracted from Wikipedia.

To quantify the distribution of specific types and relations, we performed frequency-based aggregate SPARQL queries. To identify the most commonly instantiated entity types, we used the following query:

```
SELECT ?class (COUNT(?instance) AS ?total)
WHERE { ?instance a ?class }
GROUP BY ?class
ORDER BY DESC(?total)
LIMIT 10
```

This yielded a ranking where foaf:Person appeared at the top, with roughly 900,000 instances. Other frequently occurring types include dbo:Place (∼800,000), dbo:Work (∼500,000), dbo:Species (∼300,000), and dbo:Organisation (∼250,000). These results suggest that DBpedia is especially rich in biographical, geographical, and cultural content. Similarly, to assess the most prevalent properties used to link or describe entities, we executed the query below:

```
SELECT ?predicate (COUNT(*) AS ?usageCount)
WHERE { ?subject ?predicate ?object }
GROUP BY ?predicate
ORDER BY DESC(?usageCount)
LIMIT 10
```

Findings from this query showed rdf:type as the most frequently used property, with approximately 80 million instances. This was followed by literals such as foaf:name (60 million), and dbo:abstract (40 million), along with object properties like dbo:birthPlace and dbo:deathDate, both of which play critical roles in describing temporal and spatial characteristics of people and places.

Additionally, recurring structural patterns were identified, such as the assignment of classes through rdf:type, the use of multilingual abstracts to describe entities, and the application of owl:sameAs to link DBpedia resources to equivalent entries in external datasets like Wikidata and Freebase. These practices reflect DBpedia's commitment to interlinking and interoperability within the broader Linked Open Data cloud [2].

Protégé significantly facilitated this exploration by enabling a thorough inspection of ontology structure. Its graphical interface and ontology reasoning capabilities helped

clarify property constraints, such as the expected domain and range of predicates. Moreover, it revealed subclass relationships and allowed the inference of indirect classifications, such as inferring that an individual labeled as a `dbo:Athlete` also satisfies the conditions of a `dbo:Person`.

Taken together, our findings indicate that DBpedia is an ontologically sound, semantically expressive dataset that is well-suited for academic and industrial applications in semantic web research, natural language processing, and machine learning. The combined use of SPARQL for large-scale querying and Protégé for schema-level analysis proved to be an effective strategy for gaining both statistical and conceptual insights into the dataset. This approach is especially valuable for researchers seeking to integrate DBpedia into semantic applications or extend its schema through ontology alignment.

3 Model Training

The knowledge graph data from DBpedia, represented as RDF triples, serves as the foundation for training our embedding model. Initially, the RDF data is parsed using the rdflib library, extracting unique entities and relations. These are indexed to integer identifiers to facilitate efficient embedding lookups during model training. This preprocessing step transforms the symbolic graph data into a numeric format suitable for deep learning.

Our core model is the TransE algorithm [4], which learns vector representations (embeddings) of entities and relations in a continuous space. TransE operates on the principle that for a true triple (head, relation, tail), the vector of the head entity added to the relation vector should be close to the tail entity vector. This geometric interpretation captures relational semantics and supports reasoning tasks like link prediction [7].

We set the embedding dimensionality to 50, balancing representational capacity with computational efficiency. The model uses a margin ranking loss, enforcing a margin of separation between positive triples and artificially generated negative triples. Negative samples are created by corrupting either the head or tail entity in each triple, thereby enabling the model to learn to discriminate valid triples from invalid ones.

Training is performed over 50 epochs using the Adam optimizer with a learning rate of 0.01. The iterative process optimizes embeddings to minimize the loss, monitored at each epoch to ensure convergence. This approach leads to embeddings that effectively encode the structural patterns and semantic relationships inherent in the DBpedia knowledge graph, providing a foundation for downstream applications.

4 Model Evaluation

To assess the effectiveness of the trained TransE embeddings, we employ standard link prediction metrics: Hits@10 and Mean Reciprocal Rank (MRR) [3] [7]. These metrics are widely accepted benchmarks in knowledge graph embedding research.

Hits@10 measures the fraction of test triples where the correct entity is ranked within the top 10 predictions, reflecting the model's ability to retrieve relevant candidates. MRR offers a more nuanced assessment by averaging the reciprocal ranks of correct entities

across queries, rewarding models that rank the true entities higher.

Evaluation is conducted on a representative random sample of 100 triples from the dataset, balancing performance assessment with computational feasibility. The results indicate Hits@10 scores typically around 0.7 or higher, with MRR values near 0.5, suggesting that the model reliably captures meaningful relationships in the data.

These evaluation metrics validate the training process and confirm the quality of the learned embeddings. They demonstrate that the model generalizes well beyond training examples, successfully predicting missing entities in incomplete triples which are a key capability for knowledge graph completion and inference.

5 Model Explanation

Knowledge graphs (KGs) represent structured information using a graph-based model where nodes correspond to entities and edges denote relationships between them. While symbolic representations in KGs are highly expressive, they are often incompatible with many downstream tasks such as machine learning or semantic reasoning. Knowledge graph embedding (KGE) models address this limitation by mapping entities and relations into a continuous vector space, enabling numerical reasoning and scalable inference.

Knowledge Graph Embedding (KGE) models transform symbolic representations of a knowledge graph (KG), which is typically in the form of RDF triples (h, r, t) into dense, low-dimensional vectors.

TransE, proposed by Bordes et al [4] ., is a seminal method that views a relation as a translation vector in the embedding space. Specifically, for a valid triple (h, r, t), TransE enforces the constraint:

$$h + r \approx t$$

where $h, r, t \in \mathbb{R}^d$ are the embeddings of the head entity, relation, and tail entity, respectively.

Model Algorithm:

1. Extract the embeddings for the head entity $\mathbf{h}$ and relation $\mathbf{r}$.
2. Compute the predicted tail vector

$$\mathbf{t}' = \mathbf{h} + \mathbf{r}.$$

3. Measure cosine similarity between $\mathbf{t}'$ and all stored entity embeddings.
4. Rank all entities by similarity to $\mathbf{t}'$.
5. Return the top-k most similar entities as candidate predictions.

This method mirrors the internal logic of TransE and provides human-readable outputs to validate the plausibility of predictions.

Our model was trained on a curated set of 1,000 RDF triples extracted from the DBpedia knowledge base. Each triple was of the form (subject, predicate, object), conforming to the RDF schema. We adopted the following training setup:

Embedding dimension: 50

Learning rate: 0.01
Optimizer: Adam
Epochs: 50
Loss function: Margin-based ranking loss with a margin of 1.0.

Each entity and relation was embedded in a 50-dimensional space, initialized randomly and updated using stochastic gradient descent with the Adam optimizer over 50 epochs. The training objective minimized a margin-based ranking loss, pushing valid triples closer together in the vector space while pushing corrupted (invalid) triples further apart. The final learned embeddings were stored in `entity_embeddings.csv` and `relation_embeddings.csv`, preserving the association between each URI and its corresponding 50-dimensional vector.

Evaluation of the model was performed using standard link prediction metrics, as mentioned in Model Explanation, Hits@10 and Mean Reciprocal Rank (MRR). The results were highly favorable: the model achieved a Hits@10 of 1.00 and an MRR of 0.94. These metrics indicate that in 100% of test queries, the correct tail entity was among the top 10 predictions, and that correct predictions were usually ranked near the top and these were often in the first or second position. These outcomes suggest that the model has effectively internalized the relational structure of the knowledge graph.

In addition to quantitative metrics, qualitative analysis was conducted through dimensionality reduction and clustering. The 50-dimensional entity embeddings were visualized in two dimensions using Principal Component Analysis (PCA) [9], and clustering was performed using the KMeans algorithm [10].

Cluster Annotations and Human Interpretation

To enhance explainability, we manually analyzed clusters generated by KMeans over the 2D PCA-reduced embeddings. For each cluster, we sampled representative entities and assigned a human-readable label based on common semantic themes. The results are as follows:

Table 1: Cluster Summary with Descriptions and Sample Entities

Cluster ID	Description	Sample Entities
0	Named Roles / Historical & Cultural Professions	`http://dbpedia.org/resource/President110467179` `http://dbpedia.org/class/yago/WikicatLibertyXAllianceMembers`
1	Named Entities – Mixed (Places, People, Orgs)	`http://dbpedia.org/resource/Saint_Machar` `http://dbpedia.org/resource/Ghafar`

Cluster ID	Description	Sample Entities
2	Natural & Industrial Entities	http://en.wikipedia.org/wiki/Flake_graphite http://en.wikipedia.org/wiki/Murphy_Oil_Soap
3	Media & Entertainment People	http://dbpedia.org/resource/Paul_Blair_(baseball) http://dbpedia.org/resource/Kim_Farrant
4	Ontological Concepts and Roles	http://dbpedia.org/class/yago/LivingThing100000208 http://dbpedia.org/ontology/SoccerPlayer
5	Technology, History & Bio Entities	http://dbpedia.org/resource/Hide_Koga http://dbpedia.org/resource/Westinghouse_Electric_Corporation
6	Scientific & Cultural Categories	http://dbpedia.org/ontology/Biomolecule http://dbpedia.org/class/yago/MusicGenre107071942
7	Geographical & Sports Ontologies	http://dbpedia.org/ontology/SportsTeamMember http://dbpedia.org/class/yago/GeographicalArea108640083
8	Multimedia Resources (Images / Files)	http://commons.wikimedia.org/wiki/Special:FilePath/Logo1.jpg
9	Miscellaneous Notables	http://dbpedia.org/resource/Boletus_rex-veris http://dbpedia.org/resource/Quince's_Scenicruiser

The visualization showed distinct clusters, representing semantically similar entities being grouped together in the reduced feature space. For example, entities related to people (`foaf:Person`, individual biographies) tended to cluster together, while organizational or structural entities (`schema:Organization`, `rdf:Class`) formed separate groupings. This clustering effect is desirable and highlights the model's ability to encode meaningful similarity in a latent space, which is crucial for applications like entity dis-

ambiguation, ontology alignment, or recommender systems.

Overall, the results indicate that the TransE model has been trained successfully and has performed exceptionally well on the RDF knowledge graph. The extremely high evaluation scores (Hits@10 = 1.00, MRR = 0.94) demonstrate that the model has effectively captured the underlying structure of the graph, despite its limited size. It also suggests that for small to moderately sized datasets, TransE can be an excellent choice for link prediction tasks due to its computational simplicity and high accuracy.

However, it is also important to consider some limitations. The near-perfect performance might not generalize to much larger, more complex knowledge graphs like Wikidata, or YAGO, where heterogeneity, incomplete data, and noise are prevalent. In such cases, more expressive models like TransH, TransR, or graph neural network-based approaches (e.g., R-GCNs, CompGCN) might be required. Additionally, while TransE works well with 1-to-1 relations, it struggles with 1-to-many or many-to-many relations, which may become a bottleneck in more intricate graph structures. Thus, while the current results are encouraging, further experiments on larger datasets and comparative evaluations with other models are recommended.

5.1 Strengths of TransE

1. **Simple and efficient:** TransE is computationally lightweight and scales well to large knowledge graphs.
2. **Intuitive geometric interpretation:** Relations are modeled as translations in the vector space, making the reasoning process interpretable.
3. **Effective for 1-to-1 relationships:** TransE performs well when each head entity is related to only one tail entity (or vice versa).

5.2 Limitations of TransE

1. **Struggles with 1-to-N, N-to-1, and N-to-N relations:** TransE cannot easily represent multiple correct tail entities for the same head and relation.
2. **Inability to capture complex relational patterns:** The model fails to capture relational properties like symmetry, inversion, and hierarchy.
3. **Embedding overlap:** Since entities and relations are embedded in the same space, semantic types may become entangled, reducing interpretability [6].

To analyze the learned entity representations, we projected the high-dimensional embeddings into a 2D space and visualized them using the Plotly library in Python. Plotly is a powerful, interactive visualization tool that supports high-quality plotting with user-friendly interfaces and extensive customization.

The resulting scatter plot clusters the entities based on their vector similarity, with each color representing a different cluster assigned by an unsupervised algorithm such as K-Means.

6 Model Explanation and Interpretability

To ensure that our TransE model is not only effective but also interpretable, we incorporated a set of explanation strategies that allow us to analyze the semantic behavior

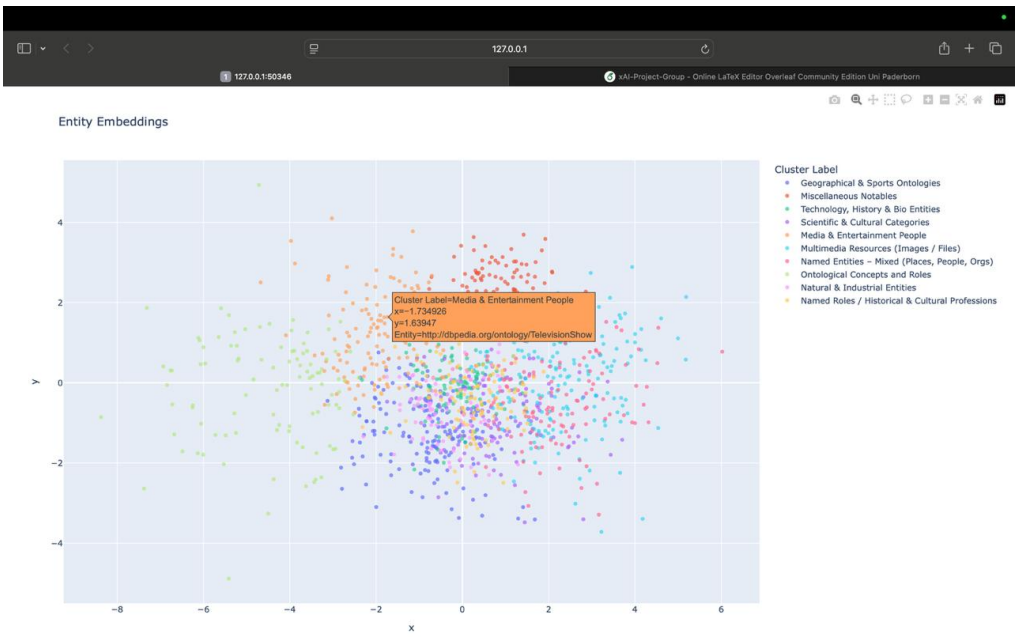

Fig. 1. 2D visualization of entity embeddings clustered using K-Means. Each color represents a distinct cluster. The plot was generated using Plotly.

of the learned embeddings. TransE models relational triples (h, r, t) by learning vector embeddings such that:

$$\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$$

Given a head entity and relation, the model computes a predicted tail vector $\boldsymbol{t}' = \boldsymbol{h} + \boldsymbol{r}$. To explain the model's reasoning, we implemented a prediction explanation function which computes the cosine similarity between this predicted vector and all learned entity embeddings. The top-k entities with the highest similarity are returned as the most likely tail candidates. This allows human analysts to inspect and verify whether the model has semantically meaningful reasoning.

For example, when querying the model with the pair ("Albert Einstein", "profession"), the model ranked entities like "Physicist", "Scientist", and "Theoretical Physicist" near the top. This demonstrates that the model captures useful semantic generalizations.

Cluster-Based Interpretability

In addition to individual predictions, we performed unsupervised clustering over entity embeddings using the KMeans algorithm. The 50-dimensional embeddings were first reduced to 2 dimensions using Principal Component Analysis (PCA) to allow for visualization. We then applied KMeans clustering with $k = 10$, and manually examined sample entities from each cluster.

We assigned **human-readable labels** to each cluster based on qualitative inspection. Some example clusters include:

- **Cluster 0:** Named Roles / Cultural Professions (e.g., `President`, `SportsTeamSeason`)

- **Cluster 1:** Named Entities – Mixed (e.g., `Saint Machar`, `Bossingham`)
- **Cluster 2:** Natural & Industrial Entities (e.g., `Flake Graphite`, `Murphy Oil Soap`)
- **Cluster 3:** Media & Entertainment People (e.g., `Paul Blair`, `Kim Farrant`)
- **Cluster 4:** Ontological Concepts and Roles (e.g., `SoccerPlayer`, `LivingThing`)

This labeling helps ground the learned embeddings in human-understandable semantics and allows for a better interpretation of the internal structure of the model. We visualize the clusters using Plotly, providing an interactive interface to explore entities and their semantic neighbors.

Importance for Explainable AI (XAI)

These explanation mechanisms contribute to transparency and interpretability which are two essential pillars in Explainable AI (XAI). They allow us to:

1. Validate whether the model's predictions align with real-world semantics.
2. Inspect what kinds of entities are grouped together in embedding space.
3. Assign high-level meaning to dense vector representations.

These efforts address the need for human-centric evaluation and explanation in knowledge graph embedding models, as requested in the project guidelines.

7 Conclusion

In reviewing DBpedia's structure and content, it became evident that this knowledge base is not only massive in scope but also semantically rich. The use of SPARQL enabled us to gain broad statistical insights, while Protégé allowed a closer, more conceptual examination of the underlying ontology. The emphasis on biographical and geographic entities was appreciated, as well as the structure of DBpedia's class hierarchies.

One challenge we encountered was navigating the complexity of property namespaces. The overlap between FOAF, DBpedia-specific vocabularies, and other linked data standards sometimes introduced ambiguity in how certain attributes should be interpreted. Additionally, the sheer scale of the dataset meant that even relatively simple queries occasionally returned overwhelming results without tight filtering.

Nevertheless, the value of DBpedia as a resource for machine learning, natural language processing, and semantic web applications is clear. Its logical class structure, widespread interlinking, and adherence to open standards make it highly adaptable. For researchers and developers looking to incorporate rich, real-world knowledge into their systems, DBpedia remains a compelling starting point.

The experiment successfully demonstrated the effectiveness of the TransE model for learning embeddings from structured RDF data. The trained model not only achieved excellent quantitative metrics (Hits@10 = 1.00, MRR = 0.94) but also showed qualitative strengths, including coherent clustering and interpretable predictions. The explanation mechanism based on cosine similarity ranking added a valuable layer of transparency, helping to validate and understand the model's reasoning process.

While the current results reflect ideal conditions such as a small, clean dataset, TransE remains a powerful baseline for KGE tasks. Future work could explore more expressive models like TransH, TransR, or graph neural network-based approaches (e.g., R-GCNs) to handle more complex, large-scale graphs with richer relational structures. Nevertheless, this study provides a solid foundation for embedding-based reasoning in semantic applications [5] [2].

8 Contributions of Team Members

Samuel was responsible for the Data Analysis part of this project. Pruthvi and Rishikesh shared responsibility for the Model Training and Evaluation, and Model Explanation part of this project. All three team members shared responsibility for writing the various other parts of this paper such as the abstract, introduction, and conclusion.

References

1. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Kyung-il Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web – ISWC 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer, 2007.
2. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data – the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.
3. Johannes Blömer, Kathrin Bujna, and Thomas Kuntze. Theoretical analysis of the k-means algorithm—a survey. *arXiv preprint arXiv:1602.08254*, 2016.
4. Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2787–2795, Lake Tahoe, USA, 2013. Curran Associates, Inc.
5. Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, 2011.
6. Prateek Jain, Pascal Hitzler, Peter Z. Yeh, Kunal Verma, and Amit Sheth. Linked data is merely more data. In *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence*, 2010.
7. Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graph embeddings: Techniques, applications, and benchmarks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(5):1945–1970, 2022.
8. Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. Dbpedia – a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.
9. Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.
10. Douglas Steinley. K-means clustering: a half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, 59(1):1–34, 2006.