Enhancing Document Ranking with Knowledge Graphs

Nicholas Valle Gross - 122101661

November 2025

1 Introduction

The objective of this project is to measure the effect of using knowledge graphs(KG) for query expansion on the quality of document retrieval when the underlying mechanism is BM25. Hypothesis: expansion by KG increases recall and can increase ranking metrics (MAP, NDCG), especially in queries with synonyms/entities; results depend on the quality of the entity linking and the type of expansion. Proposal: compare retrieval using BM25 and BM25 + query expansion with KG. like KG.

Note: Throughout the report Knowledge graphs will be referred as KG.

2 Theoretical Background

Probabilistic models like BM25 or TF-IDF evaluate the relevance of documents based on the exact frequency of words. The problem with this approach is that the inherent semantic relationships between the document entities end up being ignored. So, for example, if the query is "capital of France" and a document mentions "Paris," BM25 will not see a match between these entities despite their high semantic similarity. This happens because these models do not understand the meaning of words, only their occurrences.

2.1 Knowledge Graphs

Knowledge Graphs are structured representations of information in which entities are represented as nodes and semantic relationships between these entities are represented as edges.

These graphs allow the modeling of relationships such as synonymy, hierarchy, and association, enabling the representation of semantic knowledge that goes beyond simple lexical matching.

In this project, KG's are used as a mechanism to enrich user queries by adding related entities that may not be explicitly present in the original query.

2.2 Query expansion

Query expansion is the reformulation of a query to obtain better results in the retrieval process. For this purpose, we can:

- Find synonyms for words
- Find semantically related words
- Add new terms
- Use stemming
- Correct grammatical errors

In the context of KG, query expansion serves to enrich the query by encompassing related terms, thereby retrieving more relevant documents and consequently increasing recall.

Example of query expansion: "pulmonary diseases" → "pulmonary diseases, respiratory diseases, bronchitis, asthma"

2.3 Semantic similarity

In the base article[1] the concept of relatedness is used, which is a metric to measure the degree of association between two entities within a KG. It measures the topological proximity between two entities, that is, how much their neighbors (incoming edges) overlap. In other words, two entities are more "related" if they share connections with many other similar entities in the graph.

Originally, relatedness was a metric used to assess the similarity between 2 Wikipedia pages[2], more specifically how many external links they have in common.

The relatedness of two entities is defined as follows:

$$R(E_1, E_2) = \frac{log(max(|in(E_1)|, |in(E_2)|)) - log(|in(E_1) \cap in(E_2)|)}{log(W) - log(min(|in(E_1)|, |in(E_2)|))}$$

where $in(E1)$ is the set of incoming edges of $E1$ and W is the total number of vertices of the KG.

QDR (Query Document Relatedness) measures precisely the degree of association between a document and a query. It compares each entity in the document with each entity in the query pairwise. It is defined as:

$$QDR = \sum_{i=0}^{n} \frac{\sum_{j=0}^{m} R(E_{q_i}, E_{d_j})}{m}$$

being n the number of entities in the query and m the number of entities of the document

However, when I ran some tests using the relatedness formula given in the article, I obtained terrible results, with the similarity going to 0, or becoming

very small. This most likely happened because all entities have the same weight and the graphs became very sparse, causing less relevant entities to generate noise in the similarity calculation.

Given this problem, I decided to use a different approach. Since some entities are inherently more relevant than others in a document, we can use pagerank to assign a greater weight to these entities. As a result, less relevant entities will have less weight and will generate less noise.

Another choice that was made is to disregard the direction and label of the edges in the similarity calculation, in order to obtain better results. Furthermore, in the similarity calculation we don't just look at the immediate neighborhood of the entity, but rather at a maximum degree of separation, which is defined beforehand.

Therefore, we define the similarity between two entities as:

$$sim(e_1, e_2) = \begin{cases} 1 - \frac{d(e_1,e_2)}{D} \ se \ d(e_1, e_2) \leq D \\ 0 \ caso \ contrario \end{cases}$$

where $d(e1, e2)$ is the minimum distance between entities $e1$ and $e2$ and D is the maximum degree of separation that we allow.

The similarity between query and document will be defined as:

$$\mathrm{Sim}(Q, D) = \frac{\sum_{e_q \in Q} \sum_{e_d \in D} \mathrm{PR}(e_q)\ \mathrm{PR}(e_d)\ \mathrm{sim}_{\mathrm{ent}}(e_q, e_d)}{\sum_{e_q \in Q} \sum_{e_d \in D} \mathrm{PR}(e_q)\ \mathrm{PR}(e_d)}$$

Finally, we will calculate the final score for each document as the sum of its BM25 score and the semantic similarity by KG multiplied by a factor α.

$$Score(q, d) = BM25(q, d) + \alpha sim(q, d)$$

I chose 0.85 for the value of α, which is the traditional value adopted by the IR community.

3 Implementation

Next, I will discuss the steps that make the implementation phase.

3.1 KG construction

The first stage of implementation is to build the KG for each document, including the query. This stage is divided into 2 sub-stages: entity extraction and semantic relation extraction. For this, I used spaCy[3], which is a natural language processing (NLP) library for Python.

3.1.1 Entity extraction

The name of this step is self-explanatory. We need to extract the entities from a text. The problem with spaCy is that it only performs Named Entity Recognition (NER), meaning it only recognizes named entities.

Named entities are real-world objects that can be denoted by a proper name, such as people, places, organizations, etc. So, spaCy would recognize "Rio de Janeiro" and "Michael Jackson" as entities, but it wouldn't recognize "frog", for example, since "frog" is a generic concept.

Therefore, we need to divide entity extraction into 3 classes:

- Named entities
- Composite entities
- Nouns

When spaCy processes text, in addition to extracting named entities, it performs tokenization, and for each token it associates attributes such as grammatical class (Part-of-Speech or POS) and syntactic dependency (function of the word in the sentence or DEP). These attributes can be used to extract composite entities.

Extracting composite entities can be divided into 3 types: by chunk, by adjectival modifier (noun + adjective), and by preposition (noun + preposition + noun). The first can be done using `noun_chunks` method from spaCy, which separates the text into chunks.

In the second step, for each noun, we check if any of its children in the parse tree are adjectives (token_dep=amod), if so we extract the noun and its modifier as an entity

In the third step, for each noun, we check if any of its children are nouns and if their syntactic dependency is a nominal modifier (child_dep=amod), we extract its children that are prepositions, and finally we concatenate noun, preposition, and noun.

Furthermore, I only recognized composite entities that had a Wikipedia page. I made this choice because the model often recognized composite entities that don't have much semantic value. For example, in the document "The rat gnawed the king of Rome's clothes", "Romes's clothes" was being recognized as an entity, despite having no semantic relevance within the document.

Finally, we extract the common nouns that are neither in named entities nor in compound entities.

Furthermore, we lemmatize the entities, which converts each word to its canonical form, avoiding ambiguity.

3.1.2 Relation extraction

After extracting the entities from the text, the next step is to identify how these entities relate to each other. This task is known as relation extraction. The recognition of semantic relations has been divided into 2 types:

- **SVO**: subject-verb-object relations identified via syntactic dependencies
- **Nominal relations**: such as "X of Y" or "X from Y", extracted by analysis of nmod, prep, pobj dependencies.

The result of extracting relations is a set of triplets (sro) or (subject-relation-object). Then, for each triplet, we take the entities corresponding to the subject and object in the KG and create an edge between them with the label r. Although the edge label is being ignored in this work, it may be useful in future applications.

3.1.3 Query expansion

Before calculating the similarity between the query and the documents, we perform query expansion. For this, I used the Wikipedia API[4] to search for related terms for each entity in the query. And for each related term found, we add an edge between the entity and the term with a generic label.

3.1.4 Semantic similarity

With the KG built and PageRank calculated, the following is computed:

1. distances between each entity-query × entity-document pair;
2. weighted similarities;
3. final weighted average

3.2 BM25

BM25 was implemented using the traditional formula:

$$BM25(q,d) = \sum_{t \in q} \log\left(\frac{N - n_t + 0.5}{n_t + 0.5}\right) \cdot \frac{(k_1+1) f_{t,d}}{k_1(1 - b + b\frac{|d|}{|d|}) + f_{t,d}}$$

with standard parameters k1 = 1 and b = 0.75. In addition, we the normalize the BM25 ranking to generate consistent results.

4 Experimentation

For the experimentation phase, I chose the Wikipedia API again to extract pages and use them as documents. I made this choice for practicality and to have more control over the experimentation environment, where I could choose exactly which related documents would go into the test set.

With this in mind, I chose varied themes, such as Health, Philosophy, Technology, and for each theme I developed a query, selected related documents, and a "noise" document, which has no relation to the query, in order to test

the accuracy of the model. In addition, I manually determined the relevant documents for each query. Below is an example of a created test set.

To extract the documents, I wrote a script that consults Wikipedia, checks if the page exists, and extracts only the first paragraph of its introduction.

This decision is motivated by the fact that the introduction usually presents a concise and objective definition of the term, containing information without redundancies or excessive details. This text is then cleaned and stored as a document, replacing the original "titles" field with a "documents" field. In this way, each topic comes to have a set of real texts, from external sources, which are later used in tasks such as similarity, rankings, and knowledge graph construction.

Finally, for each test set, I ran the 2 approaches: BM25 and BM25+KG and calculated the performance metrics: accuracy, recall, MAP, and nDCG. For precision, recall, and MAP, I used the entire ranking, and for nDCG, I used the top 10 results from the ranking. In total, 20 test sets were tested.

5 Validation

The experiment was validated by:

- Comparing the 2 approaches (BM25 base and BM25 + KG) using performance metrics.
- Using of real documents (Wikipedia).
- Using aame set of documents for both approaches.
- Using short queries with varied subjects.

6 Results

Averages of the performance metrics were calculated for each test set, and these were the results:

	Precision	Recall	MAP	nDCG
BM25	0.383	0.729	0.746	0.849
BM25 + KG	0.433	0.784	0.753	0.843
Variation(%)	+13.05	+7.54	+0.94	-0.71

Figure 1: Averages of metrics

The results showed that:

- Accuracy has increased significantly. This indicates that more retrieved documents became relevant when semantic expansion via KG was added.
- The increase in recall confirms the project's main hypothesis: expanding queries by KG increases the set of relevant documents retrieved, bringing documents into the ranking that pure BM25 would not identify, especially when synonymy or lexical variability is present in the query domain.
- In some cases, relevant documents retrieved by the query expansion were positioned slightly lower than the top, or irrelevant documents gained points and "moved up," negatively impacting the initial BM25 ranking, hence the slight worsening in nDCG. This means that the weight of semantic similarity per KG still needs to be calibrated. Furthermore, the query expansion may have introduced noise, which influenced the ranking order of the documents.

Therefore, based on the results of the experiments performed, we can conclude that KG query expansion significantly improves the system's ability to retrieve relevant documents, although there may still be room to calibrate the influence of semantic similarity.

7 Future work

Although the developed system is capable of extracting entities and semantic relations in Portuguese and representing them in a knowledge graph, there are still several aspects that can be improved. Future work includes:

- Improve semantic coverage of relations. Current relation extraction is primarily based on SVO (Subject-Verb-Object) structures and simple nominal relations mediated by prepositions. The aim is to expand the rule set to include:
  - temporal relations (e.g., "during", "before", "after")
  - causal relations (e.g., "caused", "resulted in")
  - conditional and modal relations
- Incorporation of machine learning-based models. The current pipeline relies heavily on linguistic heuristics and syntactic analysis provided by spaCy. An advancement would be to integrate supervised models or transformers (e.g., BERTimbau or specialized LLMs) to:
  - to classify types of relationships with greater precision;
  - to detect implicit or anaphoric relationships;
  - to reduce errors in sentences with incomplete grammatical structure.
- Improve handling of co-referencing and composite entities. Although the system implements basic resolution, there are limitations when:

– entities are referred to by distant pronouns
– there are multiple entities with similar names;
– overlapping titles, roles or attributes (e.g., “President Lula”, “Federal Government”)

It is proposed to use more robust methods of co-reference resolution, as well as semantic mapping between synonyms and hypernyms.

References

[1] Boqi Chen, Kua Chen, Yujing Yang, Afshin Amini, Bharat Saxena, Cecilia Chávez-García, Majid Babaei, Amir Feizpour, and Dániel Varró. Towards improving the explainability of text-based information retrieval with knowledge graphs. *arXiv preprint arXiv:2301.06974*, 2023.

[2] Paolo Ferragina and Ugo Scaiella. Fast and accurate annotation of short texts with wikipedia pages, 2010.

[3] Explosion AI. spacy: Industrial-strength natural language processing in python. `https://spacy.io`, 2025. Acessado em: 27-Nov-2025.

[4] Wikipédia. Wikipédia api. `https://pypi.org/project/Wikipedia-API/`, 2025. Acessado em: 27-Nov-2025.