Can SPARQL Talk in Portuguese? Answering Questions in Natural Language Using Knowledge Graphs

Elbe Alves Miranda and **Daniel de Oliveira** and **Aline Paes**
Institute of Computing, Universidade Federal Fluminense, Niterói, RJ, Brazil
elbemiranda@id.uff.br, {danielcmo,alinepaes}@ic.uff.br

Abstract

Knowledge Graph Question Answering (KGQA) aims to retrieve answers to natural language questions from a Knowledge Graph (KG), allowing users to obtain responses even without expertise in a KG query language like SPARQL. Most existing solutions focus on training Machine Learning (ML) models to convert questions in English into a specific query language. Only a few initiatives have been made for languages other than English, e.g. Portuguese, although it is the eighth most spoken language in the world and presents its linguistic challenges. Moreover, the number of datasets and examples in them to train ML models in other languages is also limited. This paper introduces $KQGA_{PT}$, a system that relies on low-resource-based techniques to answer questions posed in Portuguese from KGs. Instead of training an entirely end-to-end solution, our system is built upon five components: (i) question analysis, (ii) question classification, (iii) phrase mapping, (iv) query generation, and (v) query ranking. Our contributions include trained models for question classification and query ranking specifically customized for the Portuguese language, offering a comprehensive solution for answering questions in Portuguese from KGs. The results are promising: requiring only a few examples, they outperform a baseline method that translates the input question from Portuguese to English. To the best of our knowledge, this is the first KGQA solution designed for Portuguese that uses the standard QALD dataset.

1 Introduction

Pretrained language models have achieved state-of-the-art to answer questions based on textual information (Pang et al., 2022)[1]. However, despite recent progress with augmented information retrieval models (Lewis et al., 2020), they still struggle to answer certain questions that require factual knowledge adherence. Knowledge Graph Question Answering (KGQA) systems are designed to answer queries posed in natural language while leveraging rich factual information of Knowledge Graphs (KG) (Momtazi and Abbasiantaeb, 2022), such as Freebase[2], DBPedia[3], ConceptNet[4], among others. Contextualized and related information in the KG enhances the accuracy and quality of generated answers.

KGQA systems usually convert natural language questions to a KG query language (Momtazi and Abbasiantaeb, 2022), for example, SPARQL (W3C Semantic Web Standards, 2023). They may leverage machine learning (ML) models to find a mapping function that converts a natural language question to a SPARQL query. The ability to transform natural language into a query is crucial for individuals who work with structured data. Moreover, using explicit queries allows for clarity and transparency, benefiting explainability.

Standard datasets for training ML models are QALD[5] and LCQuAD[6]. QALD is a multilingual dataset, encompassing natural language questions in several languages, their corresponding SPARQL queries, and possible answers. In contrast, the LCQuAD dataset features complex questions, but only in English, each also paired with SPARQL query and possible answers.

However, even leveraging those datasets, learning that mapping is challenging, as it requires a precise alignment between the question tokens and the entities and relations within the KG. Existing solutions usually follow two approaches: (i) to end-to-end train an ML model that directly translates

[1] https://nyu-mll.github.io/quality/

[2] https://developers.google.com/knowledge-graph

[3] https://www.dbpedia.org/

[4] https://conceptnet.io/

[5] https://github.com/ag-sc/QALD

[6] https://github.com/AskNowQA/LC-QuAD

the natural language question to a query language, or (ii) to break down the conversion process into multiple steps to reduce the complexity of the task (Purkayastha et al., 2022). While the former approach requires more robust methods, hence more examples, the latter demands effective solutions for subproblems, such as correctly linking entities and relations in the question to the relevant resources within the KG (Momtazi and Abbasiantaeb, 2022).

This way, KGQA solutions have primarily been developed for the English language, benefiting from the abundance of resources available for training models and the maturity of the underlying tasks. Only a limited number of initiatives have been proposed for other languages (Momtazi and Abbasiantaeb, 2022). For instance, despite Portuguese being the eighth most spoken language in the world, with over 263 million speakers (Eberhard et al., 2023), and ranking as the fifth most used language on the Internet, with over 171 million users (Internet World Stats, 2023), there are very few initiatives addressing KGQA for Portuguese. To illustrate this, when examining the 76 proposed solutions for the KGQA task using the LcQuAD-v1 dataset (Trivedi et al., 2017) and DBPedia, only ten are designed for languages other than English, and just one is tailored explicitly for Portuguese[7] (Perevalov et al., 2022). In the case of the QALD-9 dataset, which employs DBPedia as the KG, all 49 of the proposed solutions were exclusively for English[8] (Perevalov et al., 2022). This underscores the need to develop KGQA solutions for languages beyond English, including Portuguese.

Furthermore, many languages typically present unique challenges. For example, the Portuguese language presents a multitude of verb tenses, each with distinct conjugation forms for different persons and numbers, which can introduce complexity when attempting to directly translate a question written in Portuguese into a SPARQL query. Consider, for instance, the question written in Portuguese: "*Quais filmes foram dirigidos por Quentin Tarantino?*" (Which movies were directed by Quentin Tarantino?). When translating this to SPARQL, the challenge lies in identifying the appropriate property within the KG for the phrase "*dirigidos por*" (directed by). This complexity arises from the myriad of possible conjugations of the verb "*dirigir*" (to direct) in Portuguese.

In addition, training Machine Learning models by consuming datasets with a limited number of examples, as seen in the case of QALD7 with only 215 training examples, poses substantial challenges. The primary obstacle arises from the insufficient variety and diversity in the training data. When examples are scarce, the model may face difficulties learning patterns and applying that knowledge to new examples, in our case, translating a natural language question into a SPARQL query.

This paper proposes a system to address KGQA task in Portuguese, named KGQA$_{\text{PT}}$. KGQA$_{\text{PT}}$ faces the resources limitation by converting questions to SPARQL queries through five components: (i) Question Analysis, (ii) Question Type Classification, (iii) Phrase Mapping, (iv) Query Generation, and (iv) Query Ranking. With KGQA$_{\text{PT}}$, we aim to investigate the performance of a component-based system that tackles the KGQA task for the Portuguese language and which one of its steps fails at most. In addition to the proposed system, we contribute with a question and relation classifier that could be adapted to other tasks.

The experimental results show that KGQA$_{\text{PT}}$ outperforms a baseline method that translates the input question from Portuguese to English. To the best of our knowledge, this is the first KGQA solution designed for Portuguese that uses the standard QALD dataset.

2 Background

A **Knowledge Graph (KG)** is a data structure $KG = (V, E, R)$ where V is the set of entities, R is the set of relation types, and $E = (h, r, t)$ is an edge representing a fact, with $h, t \in V$, also called subject/object, or head/tail, and $r \in R$, also called as predicate. KGs provide a structured representation of the semantic relationships between entities in the real world. Let us assume that we want to represent in a KG the answer to the following question *"In which Formula 1 championship was Ayrton Senna a champion?"*. The answer would be in a subgraph $KG' = (V', E', R')$, where $KG' \subset KG$. Figure 1 exhibits a diagram that illustrates KG'.

There are several types of KG, either holding general concepts or specific knowledge. For example, DBpedia (Auer et al., 2007) is a general information KG representing Wikipedia texts in a structured format. The Wikipedia page about Ayr-

[7] https://github.com/KGQA/leaderboard/blob/gh-pages/dbpedia/lcquad.md#lc-quad-v1

[8] https://github.com/KGQA/leaderboard/blob/gh-pages/dbpedia/qald.md

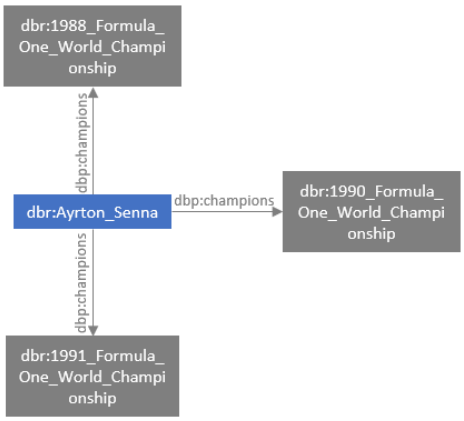

Figure 1: An example of a Knowledge Graph.

ton Senna[9] is represented on DBpedia as a node, with edges connecting it to other nodes that represent information such as date of birth and achievements.

KGs are usually represented using the Resource Description Framework (RDF) format and SPARQL as query language. **RDF** is a format for directed and labeled graph data that stores facts in the form of triples (h, r, t), where h is the subject, r is the predicate, and t is the object. Subjects are represented by resources and are identified using Uniform Resource Identifiers (URIs), which can represent real-world entities, abstract concepts, documents, and more. Predicates represent the properties or attributes of these resources, while objects represent the values of these properties or the relationships with other resources.

SPARQL (SPARQL Protocol and RDF Query Language) is a standardized query language designed for retrieving information from RDF data sources. It enables querying knowledge graphs that adhere to the RDF model, easing data retrieval and access to structured information. SPARQL allows users to perform complex queries on RDF graphs, combining search criteria, filtering, joining, and aggregation. The language supports triple patterns, graph queries, variable-based queries, and conditional expressions. SPARQL is widely adopted and standardized by the World Wide Web Consortium (W3C Semantic Web Standards, 2023), serving as a fundamental technology for accessing and exploring data in RDF-based KGs.

For instance, consider the following input question in Portuguese: "*Em quais campeonatos de Fórmula 1 Ayrton Senna foi campeão?*" (In which Formula 1 championship was Ayrton Senna a champion?) and the KG represented in Figure 1. A query that correctly answers the question is: "`SELECT ?resp WHERE {?resp dbp:champions dbr:Ayrton_Senna}`". The `SELECT` clause indicates the data we want to retrieve from the query. The variable `?resp` gets the response for the question. The `WHERE` clause specifies the search pattern. In this case, we are looking for an RDF triple where the variable `?resp` is related to the property `dbp:champions` and the resource `dbr:Ayrton_Senna`. A KGQA task evaluation must compare the returned answer with the answer annotated in the dataset.

3 Related Work

Ketsmur et al. (2017) proposed a KGQA system that relies on DBpedia as the KG and SPARQL as the query language to answer factual questions in Portuguese. The system first identifies the question type (causal, list, or definition). Then, it determines the expected DBpedia classes as potential answers (Person, Agent, Place, Game, *etc*). Following, it performs a morphosyntactic analysis of the question. The next step is Entity Linking using the BabelNet system. The fifth step, Relation Linking, involves getting all the properties linked to the entities identified in the previous step and comparing their names with synsets extracted from BabelNet. Finally, it builds the SPARQL query using the entities and relations linked in the previous steps. The system is evaluated on a dataset of 22 factoid questions generated by the authors. However, only 15 had corresponding responses in DBpedia, from which the system generated a correct response in 10 cases. While the obtained result is promising, it is worth noting that the authors used a private dataset with only a few examples, preventing reproducibility.

More recently, de Sousa et al. (2020) proposed an ontology-based approach to answer questions in Portuguese about facts stored in a KG. The authors first execute the Entity Linking step by comparing the question terms with the ontology labels. The second step is the Relation Extraction, where the nodes of the question syntactic tree are compared with the indexed nodes of the ontology. After Entity and Relation linking, the SPARQL queries are built and ranked. The answers are presented as data visualizations, including bar plots, showing the answer to the initial question and other expanded responses. The authors built a movies-and-series dataset of

[9] https://en.wikipedia.org/wiki/Ayrton_Senna

Portuguese questions from QALD to evaluate the method. The dataset contains 150 questions with classes and individuals mentioned in QALD linked to classes and individuals of IMDb. The system achieved an F1-score of 57%. Although the aforementioned approaches propose solutions for the Portuguese language, they built their own datasets for testing and did not evaluate their methods with standard KGQA datasets, impairing an agnostic evaluation.

Given the vast availability of examples in English, previous work leveraged training sequence-to-sequence models (seq2seq) models to convert a question in natural language to a SPARQL query. For example, Rony et al. (2022) achieved an F1-score of 67.82% and Purkayastha et al. (2022), an F1-score of 55.3% in the English QALD-9 dataset. More recent approaches have also leveraged large language models to the task, primarily GPT. However, they do not guarantee better performance: GPT-4 achieved an F1-score of 57.2%, compared to 46.19% for GPT-3.5v3 and 38.54% for GPT-3 on the QALD-9 dataset (Tan et al., 2023).

Even in English, other approaches can be less data-intensive by dividing the solution into smaller parts, each solving a specific subtask. For example, Liang et al. (2021) proposed a modular architecture to address the KGQA task, where each component is responsible for solving a specific part of the task. The system comprises five components: Question Analysis, Question Type Classification, Phrase Mapping, Query Builder and Query Ranking. Using the QALD dataset in English, the result was an F1-score of 63.9%, while for LCQuAD the F1-score was 68%. We adopt the same strategy in this paper.

4 KGQA$_{PT}$: a KGQA System for Portuguese

Training a seq2seq model that converts natural language questions in Portuguese to a structured language is challenging, given the low availability of examples. In this way, in this paper, we adopted the component-based strategy proposed by Liang et al. (2021) and adapted each component to Portuguese. The five components are responsible for (i) Question Analysis, (ii) Question Type Classification, (iii) Phrase Mapping, (iv) Query Generation, and (v) Query Ranking[10]. Dividing the solution into subcomponents provides the additional advantage of improving each component separately, potentially enhancing the overall system.

To illustrate our approach, consider, for example, the following input question: *"Em quais campeonatos de Fórmula 1 Ayrton Senna foi campeão?"* (In which Formula 1 championships was Ayrton Senna a champion?). First, the *Question Analysis* component extracts morphosyntactic elements, such as POS-Tagging and Stemming. Next, the *Question Type Classification* component categorizes the question into one of three types: (i) Boolean, (ii) Count, or (iii) List. In the case of the aforementioned question, the classification would be "List". The *Phrase Mapping* component handles Entity Linking, linking the phrase "Fórmula 1" to the DBPedia resource "dbr:Formula_One" and the phrase "Ayrton Senna" to the resource "dbr:Ayrton_Senna". It also performs Relation Linking, associating the term "campeão" with the property "dbp:champions".

Based on the information from the preceding components, the *Query Generation* component generates a list of candidate queries in the SPARQL language. The *Query Ranking* component then arranges these queries according to similarity criteria, ultimately selecting the highest-ranked query as the answer. In our example, the chosen query is: "`SELECT ?resp WHERE {?resp dbp:champions dbr:Ayrton_Senna}`". Figure 2 illustrates our method based on this example. Next, we detail the five components, highlighting how each was adapted to the task in Portuguese.

4.1 Question Analysis

This component extracts morphosyntactic features from the input question, aiding the subsequent components. These features are derived from tokenization, lemmatization, stemming, POS-tagging, and syntactic dependency trees. First, KGQA$_{PT}$ tokenizes the input question with SPACY[11] (Honnibal et al., 2020), using the PT_CORE_NEWS_SM model (Rademaker et al., 2017). Next, each token is annotated with its *POS-Tag*, also obtained with SPACY. Moreover, we also leverage SPACY to acquire the syntactic dependency tree associated with the input question. For the Lemmatization, we use the SIMPLEMMA system[12] (Barbaresi, 2023), as they have achieved good accuracy in Portuguese. For Stem-

[10]The source code is available in https://github.com/elbemiranda/KGQApt

[11]https://spacy.io/

[12]https://github.com/adbar/simplemma

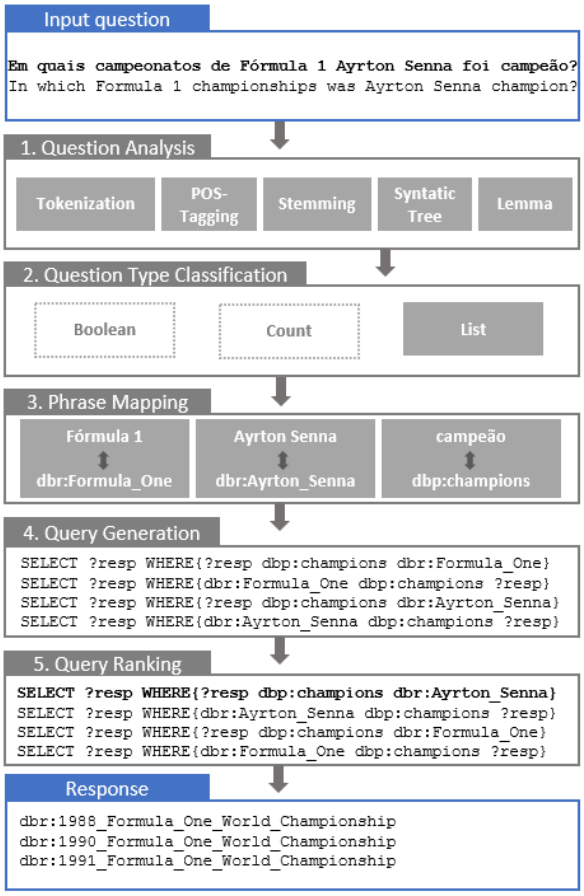

Figure 2: An illustration of KGQA$_{PT}$.

ming, we use RSLPSTEMMER from the NLTK[13] (Bird et al., 2009) Python library. All the features extracted here serve as inputs for the subsequent components. Specifically, the syntactic dependency tree and the POS-tag representation will be inputs for the Query Ranking component, while lemmas feed the Question Type classifier training and stems, the Phrase Mapping component.

4.2 Question Type Classification

The SPARQL language provides various constructs to ease question answering. For instance, to answer a question that demands binary responses, like "*Was Ayrton Senna a Formula 1 driver?*", the SPARQL query must incorporate the `ASK` clause. Some questions require a list of values as the answer, such as "*In which Formula 1 championship was Ayrton Senna a champion?*" In these instances, the most suitable construct is the `SELECT` clause. Other questions expect a numerical response, like "*How many times was Ayrton Senna a Formula 1 champion?*" In such situations, two constructs are needed: the `SELECT` and `COUNT` clauses.

While the SPARQL language encompasses a variety of other clauses like `FILTER`, `ORDER BY`, `OFFSET`, and `LIMIT`, KGQA$_{PT}$ focuses on only four: `SELECT`, `COUNT`, `WHERE`, and `ASK`. This way, to ensure the accurate construction of the SPARQL query, it is essential to pre-identify the question type that corresponds to each query construct. In this context, likewise Liang et al. (2021), we consider three types: Boolean - requiring the use of `ASK`, List - using `SELECT`, or Count - involving `SELECT COUNT`.

We trained ternary classifiers to automatically identify the appropriate SPARQL construct for a query. To that, we automatically annotate the LCQuAD dataset, as follows: if the target query included the `ASK` clause, it was annotated as Boolean; if it included the `SELECT` and `COUNT` clauses, it was annotated as Count, and otherwise, it was annotated as List. We leverage three algorithms for that task: Random Forest (RF), Support Vector Machines (SVM), and Multilayer Perceptron (MLP). For simplicity and focusing on low-resource demands, we leveraged only TF-IDF and fastText embeddings (Joulin et al., 2017) as feature representations. Moreover, since the question type classification problem did not present significant difficulty, there was no need to utilize more complex features. Both vector representations are generated from the lemmatized input questions. We conducted experiments with each combination to determine the one yielding the most accurate predictive results.

4.3 Phrase Mapping

The Phrase Mapping component associates entities or relations identified in the input question with the resources, classes, and properties within the KG. This process goes beyond merely detecting entities and relations in the input question. Instead, it entails recognizing the concepts in the input question and linking them to their corresponding resources in the KG.

For instance, consider the following input question in Portuguese: "*Em quais campeonatos de Fórmula 1 Ayrton Senna foi campeão?*" (In which Formula 1 championship was Ayrton Senna a champion?). In this case, it is insufficient to merely identify "*Ayrton Senna*" and "*Formula 1*" as entities and "*champion*" as a relation. The crucial step is establishing the correct links between these entities and relations and the appropriate classes and proper-

[13] https://github.com/nltk/nltk

ties within the KG. In the example provided, "*Ayrton Senna*" must be linked to "*dbr:Ayrton_Senna*", "*Formula* 1" to "*dbr:Formula_One*", and "*champion*" to "*dbp:champions*".

This component consists of two primary tasks: Entity Linking (EL) and Relation Linking (RL). EL encompasses two subtasks: Named Entity Recognition (NER) and Entity Disambiguation (ED). Due to the limited availability of annotated data for training models to these specific tasks, we aimed to use existing RL and EL methods for the Portuguese language as much as possible.

In English, several systems present good results in Entity Linking, as evidenced by various papers (Ferragina and Scaiella, 2010; Mendes et al., 2011; Brank et al., 2017; Dubey et al., 2018; Sakor et al., 2019). However, for Portuguese, the options are limited, with only two well-known systems available: DBpedia Spotlight[14] (Mendes et al., 2011) and Wikifier[15] (Brank et al., 2017). KGQA$_{PT}$ combines these two systems by merging the Entity Linking (EL) outputs from DBpedia Spotlight and Wikifier while eliminating duplicate entries. This process results in a set, denoted as $V' \subset V$, comprising entities within the KG.

By combining the results from both EL models, we enhance the ability of the proposed approach to identify entities within the text correctly. Consequently, this approach allows us to provide accurate answers to input questions. For instance, consider the input question: "*In which Formula 1 championship was Ayrton Senna a champion?*" Suppose DBpedia Spotlight links only the entity "*Formula 1*", while Wikifier only identifies "*Ayrton Senna*". If we rely on just one of these EL systems, we might overlook essential entities crucial for constructing a query that can accurately answer the question. By combining the outputs of both models, we increase the likelihood of accurately mapping the entities required to answer the question correctly.

While EL is responsible for linking entities in the question to resources within the KG, Relation Linking (RL) maps the text strings representing relations in the question to their corresponding relations and properties in the Knowledge Graph. RL models are less common compared to EL models, even for English. However, in English, a few models are still available (Dubey et al., 2018; Singh et al., 2018; Sakor et al., 2019). Unfortunately, none of them have a Portuguese version.

To address this issue, we adapted RNLIWOD[16] (Singh et al., 2018) to work with Portuguese, as it is a simple and straightforward open-source model. The adaptation includes translating the labels of the property dictionary that RNLIWOD uses to Portuguese using Google Translate API [17]. Additionally, we replaced its Stemmer with RSLPStemmer, which performs better for Portuguese.

We also developed a new RL model called PTRL. It first removes from the input text all entities identified by the EL model, *stop words* and interrogative pronouns, such as "*where*", "*who*", "*when*". The hypothesis of PTRL is that only the text referring to the relation will remain by removing those elements from the input question. For example, in the question "*Onde nasceu Ayrton Senna?*" (Where was Ayrton Senna born?), by removing the text referring to the entity "*Ayrton Senna*" and the pronoun *Onde (where)*, we are left with only "*nasceu*", which is the relation we need to map. Then, PTRL computes the fastText embedding of the remaining text. This embedding vector is compared using cosine similarity with the embeddings of DBpedia property dictionary labels. The properties with the top three cosine similarity values are selected and mapped as candidate relations. The Figure 3 illustrates the PTRL method. The results from both RNLIWOD and PTRL are combined by a union operator to generate a set, denoted as $R' \subset R$, comprising relations within the KG.

4.4 Query Generation

Once the Question Type Classification component has categorized the question into one of the three types, and the Phrase Mapping component has associated the entities and relations to the KG resources, the Query Generation component uses this information to formulate the queries sent to the KG.

The initial section of the SPARQL query can encompass the `ASK`, `SELECT`, or `SELECT COUNT()` clauses, depending on the classification output from the Question Type Classification component. The subsequent part of the query incorporates the `WHERE` SPARQL clause, primarily comprised of one or more triples in the subject-predicate-object (h-r-t) format. Consequently, the primary goal of the Query Generation component is to establish a

[14] https://api.dbpedia-spotlight.org/pt/annotate

[15] http://www.wikifier.org/annotate-article

[16] https://github.com/semantic-systems/NLIWOD

[17] https://cloud.google.com/translate

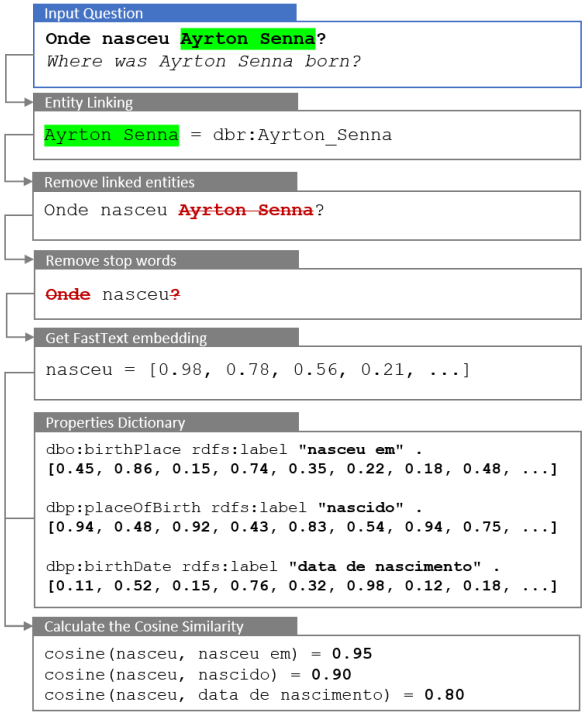

Figure 3: An illustration of the PTRL method for Relation Linking.

list of triples to construct the WHERE clause of the SPARQL query.

We employed the SQG (SPARQL Query Generator) method proposed by Zafar et al. (2018) to construct the SPARQL query. This method assembles a list of entities and relations mapped in the KG by the Phrase Mapping component. From this list, it generates a set of triples used to construct the WHERE clause of the SPARQL query. The method operates under the assumption that the formal representation of the input question is represented as a path within the KG. This path comprises only the set of mapped entities (V') and the set of mapped relations (R'). The path leads to the answer nodes. Valid answer paths within the KG are identified by initiating the search from a particular entity ($e \in V'$) and navigating through the relations ($r \in R'$) within KG. The triples used to create the queries constitute a set denoted as $T = (e, r, v)$, where $v \in V$ is a virtual entity positioned at a one-hop distance from the entity e and represents a potential answer to the question.

While straightforward questions may lead to the answer node through a single hop, more complex questions often require traversing the graph beyond a single hop away from the entities ($e \in V'$). To address the complexity of such questions, KGQA$_{PT}$ allows for traversal of the graph by one additional hop from the virtual entity (v), using the relations ($r \in R'$) until reaching other virtual entities ($v' \in V$) that might also serve as potential answer nodes. This process can be extended by further expanding the paths with additional virtual entities. However, creating more virtual entities with each step makes the process more computing-intensive.

The process yields a subgraph G comprising the entities ($e \in V'$), relations ($r \in R'$), and the newly introduced virtual entities ($v, v' \in V$). Subsequently, the task is to extract from G the potential paths that can provide answers to the question. To achieve this, we regard all virtual entities (v) as potential answer nodes, but only if they form part of a valid path within the graph. A path is deemed valid if it includes all entities and relations identified in the question through the phrase mapping, besides other possible nodes. Any valid path can become the basis for the SPARQL query. Several queries can be formed by including the possible combinations of nodes in the path. Consequently, a ranking process is required to decide which query is the most suitable for answering the question.

4.5 Query Ranking

The core premise of the Query Ranking component is that the queries most likely to answer a question are those whose tree structure closely resembles the syntactic dependency tree of the question itself. This similarity is evaluated using a Tree-LSTM model, as described by Tai et al. (2015). In this approach, the tree representation of the input question is compared with the tree derived from the generated query. A Tree-LSTM model shares many similarities with the traditional LSTM (Long Short-Term Memory) model, with the difference being its ability to consider the tree structure of the words within a text, not just their sequence in the sentence.

A tree representation is constructed for each generated query using all the triples contained within the query. The underlying concept is that the properties (relations) within the query are converted into parent nodes, and the children of these nodes consist of variables or resources (entities). To illustrate this, consider the example shown in Figure 2, where multiple queries were generated to address the question: "*In which Formula 1 championship was Ayrton Senna a champion?*" The tree representing the query that correctly answers this question would have the property *dbp:champions* at the root, with the entity *dbr:Ayrton_Senna* and the vari-

able *?resp* as its children, as depicted in Figure 4. In cases where the query comprises more than one triple, the process is repeated, starting from the non-variable node, in this instance, *dbr:Ayrton_Senna*. This process involves replacing the *Ayrton_Senna* node with the element from the new triple representing a relation.

```
?resp dbp:champions dbr:Ayrton_Senna
```

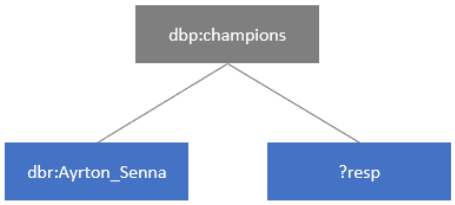

Figure 4: Tree representation of a triple.

The syntactic dependency tree, previously generated by the Question Analysis component, is fed into a Tree-LSTM to create a vectorized representation of the question. Simultaneously, the tree derived from the generated queries is also processed by the Tree-LSTM model to calculate their embedding vectors. With these representations at hand, a neural network predicts a similarity score that considers both the distance and angle between the pairs of vector representations, as detailed in (Tai et al., 2015). The query corresponding to the highest similarity value with the original question is then selected and employed to answer the question.

5 Experimental Evaluation

5.1 Datasets

We evaluate our approach with two KGQA benchmarks. The QALD (Question Answering over Linked Data) (Usbeck et al., 2017) is an initiative of the scientific community to promote the development of practical KGQA systems. QALD includes a diversity of questions and languages, a variety of linked data, and periodic updates. The QALD-7 consists of 215 questions and their respective SPARQL queries. QALD-7 includes translations in eight languages, including Portuguese. The LC-QuAD (Largescale Complex Question Answering Dataset) (Trivedi et al., 2017) has two versions: LCQuAD v1, with 5,000 question examples in English and their corresponding SPARQL queries using only DBPedia as the KG, and LCQuAD v2, containing 30,000 examples, covering queries for both DBPedia and Wikidata.

Due to the complexity of the questions in LCQuAD and the lack of examples in Portuguese that would impair reproducibility and potentially introduce misguiding results, the LCQuAD v1 dataset was used only for training the Question Type Classifier and the Tree-LSTM model. On the other hand, QALD was exclusively used as the test dataset for the final solution, as it includes Portuguese examples. Since LCQuAD v1 dataset only had questions in English, the questions were translated to Portuguese using Google Translate.

5.2 Results

Question Type Classifier We trained the Question Type Classifier with three classification algorithms: Random Forest (RF), Support Vector Machines (SVM), and Multilayer Perceptron (MLP). The representation methods are TF-IDF and fastText embeddings. Table 1 shows that RF with TF-IDF achieved the best result among all combinations. Due to that, it becomes part of the final solution.

Table 1: F1-score for Question Type Classifiers

	TF-IDF			*embeddings*		
	SVM	RF	MLP	SVM	RF	MLP
List	93.3	**94.3**	90.9	91.1	90.4	89.0
Boolean	70.8	**80.0**	60.4	69.1	0.0	61.2
Count	58.3	**63.6**	56.0	50.0	28.6	40.0
Macro Avg	74.2	**79.3**	69.1	70.1	39.7	63.4

Complete Solution We used the QALD-7 dataset to assess the complete solution, given the availability of examples in Portuguese. The dataset consists of 215 examples, of which 179 are of the List type, seven are of the Count type, and 29 are of the Boolean type. The results are evaluated with ranked-biased precision, recall, and F1. This way, precision and recall consider how many answers are correct according to the annotated dataset and also their positions according to the query ranking. F1 is computed as usual, the harmonic mean between precision and recall.

Since previous works that applied KGQA in Portuguese have not employed standard datasets such as QALD for evaluating the task, we could not compare KGQA$_{PT}$ with them. Then, to establish a baseline, we translated the questions in Portuguese to English and executed the system proposed by Liang et al. (2021), as KGQA$_{PT}$ was based on that

architecture. Note that the translation might not be perfect, which could introduce additional errors.

Table 2 brings the results of the complete solution. Out of its 215 examples, 43 did not have an answer in the KG, due to changes in DBpedia over time; therefore, we removed them from the set and computed the metrics for the 172 remaining. The table shows that KGQA$_{PT}$ achieved an overall F1-score of 41.9% in contrast to an F1 of 28.5% of the baseline. While the baseline got a better result with the count type, our system was better on both list and boolean types.

Further analyzing the results, we noticed that KGQA$_{PT}$ could not generate a single query for 114 questions. In 25 cases, the Entity Linking component failed to identify some entity. In 45 cases, the Relation Linking component failed to identify the relation. In 29 cases, both of them failed. Because of those misidentifications, KGQA$_{PT}$ could not find a subgraph containing the identified entities and/or relations, therefore, not generating the corresponding queries. Furthermore, in 15 cases, it was not possible to find a subgraph despite entities and relationships being identified. From the generated queries, 46 questions were answered correctly and 12 incorrectly. Four were due to incorrect entity mapping, five to incorrect relation identification, one to incorrect mapping of both entity and relation and two to incorrect question type classification.

Regarding the baseline, from the 172 questions, 29 were correctly answered, while one was incorrect. It was not possible to create a query for 142 questions, 27 because of invalid paths, and 115 due to failure in the phrase mapping, some of them due to translation mistakes.

Table 2: Evaluation of the Portuguese KGQA system on the **QALD-7** dataset

	P	R	F1
Liang *et al.* - List	89.7	18.4	30.5
Liang *et al.* - Boolean	100.0	3.5	6.7
Liang *et al.* - Count	100.0	42.9	**60.0**
Liang *et al.* - All	91.1	16.9	28.5
KGQA$_{PT}$ - List	80.5	32.4	**46.2**
KGQA$_{PT}$ - Boolean	75.0	10.3	**18.2**
KGQA$_{PT}$ - Count	66.7	28.6	40.0
KGQA$_{PT}$ - All	79.4	28.5	**41.9**

6 Conclusion

We proposed a solution for the KGQA task in Portuguese, adapting a model composed of five components to the specificities of the Portuguese language. We showed that adapting a solution originally developed for the KGQA task in English to Portuguese achieved an overall F1-score result of 41.9%. We emphasize that the lack of customized tools for performing Entity Linking and Relation Linking tasks greatly hinders the performance of the final solution, as they are crucial for generating queries that correctly answer the question. This way, future work should focus on enhancing the phrase mapping component, either with customized previous strategies (Gamallo and García, 2016) or developing zero-shot methods (Logeswaran et al., 2019; Wu et al., 2020) that demand less annotated data. Another suggestion for future work is to increase the number of examples, possibly with translation APIs, to train a seq2seq system.

Limitations

When interpreting the paper's results, one should consider its limitations. First, KGQA$_{PT}$ was not tested on the translated version of LCQuAD, a step that could have offered more insights into its performance. On the other hand, translations may introduce errors, misguiding the results. Additionally, the system was designed to handle only three types of queries. This focus might not cover the full spectrum of query types encountered in real-world scenarios. Lastly, the system lacks mechanisms to prevent responding to inappropriate questions, in case the knowledge bases contain such answers. This oversight could lead to ethical concerns that must be carefully considered in future investigations. These limitations underscore the need for further research and the importance of a thorough evaluation using a variety of datasets.

Acknowledgements

This research was financed by CNPq (National Council for Scientific and Technological Development), grants 311275/2020-6 and 315750/2021-9, FAPERJ - *Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro*, process SEI-260003/000614/2023, and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer.

Adrien Barbaresi. 2023. Simplemma: a simple multilingual lemmatizer for python [computer software] (version 0.9.1).

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Janez Brank, Gregor Leban, and Marko Grobelnik. 2017. Annotating documents with relevant wikipedia concepts. *Proceedings of SiKDD*, 472.

Alysson Gomes de Sousa, Dalai dos Santos Ribeiro, Rômulo César Costa de Sousa, Ariane Moraes Bueno Rodrigues, Pedro Henrique Thompson Furtado, Simone Diniz Junqueira Barbosa, and Hélio Lopes. 2020. Using a domain ontology to bridge the gap between user intention and expression in natural language queries. In *Proceedings of the 22nd International Conference on Enterprise Information Systems, ICEIS 2020, Prague, Czech Republic, May 5-7, 2020, Volume 1*, pages 751–758. SCITEPRESS.

Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. 2018. EARL: joint entity and relation linking for question answering over knowledge graphs. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*, volume 11136 of *Lecture Notes in Computer Science*, pages 108–126. Springer.

David M Eberhard, Gary F Simons, and Charles D Fennig. 2023. Ethnologue: Languages of the world(22nd edn.). dallas, tx: Sil international. *Online version: http://www. ethnologue. com [01.09.2023]*.

Paolo Ferragina and Ugo Scaiella. 2010. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1625–1628. ACM.

Pablo Gamallo and Marcos García. 2016. Entity linking with distributional semantics. In *Computational Processing of the Portuguese Language - 12th International Conference, PROPOR 2016, Tomar, Portugal, July 13-15, 2016, Proceedings*, volume 9727 of *Lecture Notes in Computer Science*, pages 177–188. Springer.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Internet World Stats. 2023. Internet world stats. https://www.internetworldstats.com/stats7.htm.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Maksym Ketsmur, Mário Rodrigues, and António Teixeira. 2017. A question and answer system for factual queries in portuguese on DBPEDIA. In *Proceedings of the International Conference on WWW/Internet 2017 and Applied Computing 2017*, page 87 – 94.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Shiqi Liang, Kurt Stockinger, Tarcisio Mendes de Farias, Maria Anisimova, and Manuel Gil. 2021. Querying knowledge graphs in natural language. *J. Big Data*, 8(1):3.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS 2011, Graz, Austria, September 7-9, 2011*, ACM International Conference Proceeding Series, pages 1–8. ACM.

Saeedeh Momtazi and Zahra Abbasiantaeb. 2022. *Question Answering over Text and Knowledge Base*. Springer.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, and Samuel Bowman. 2022. QuALITY: Question answering with long input texts, yes! In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5336–5358, Seattle, United States. Association for Computational Linguistics.

Aleksandr Perevalov, Xi Yan, Liubov Kovriguina, Longquan Jiang, Andreas Both, and Ricardo Usbeck.

2022. Knowledge graph question answering leaderboard: A community resource to prevent a replication crisis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2998–3007, Marseille, France. European Language Resources Association.

Sukannya Purkayastha, Saswati Dana, Dinesh Garg, Dinesh Khandelwal, and G. P. Shrivatsa Bhargav. 2022. A deep neural approach to KGQA via SPARQL silhouette generation. In *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*, pages 1–8. IEEE.

Alexandre Rademaker, Fabricio Chalub, Livy Real, Cláudia Freitas, Eckhard Bick, and Valeria de Paiva. 2017. Universal dependencies for portuguese. In *Proceedings of the Fourth International Conference on Dependency Linguistics, Depling 2017, Pisa, Italy, September 18-20, 2017*, pages 197–206. Linköping University Electronic Press.

Md. Rashad Al Hasan Rony, Uttam Kumar, Roman Teucher, Liubov Kovriguina, and Jens Lehmann. 2022. SGPT: A generative approach for SPARQL query generation from natural language questions. *IEEE Access*, 10:70712–70723.

Ahmad Sakor, Kuldeep Singh, and Maria-Esther Vidal. 2019. FALCON: an entity and relation linking framework over dbpedia. In *Proceedings of the ISWC 2019 Satellite Tracks (Posters & Demonstrations, Industry, and Outrageous Ideas) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26-30, 2019*, volume 2456 of *CEUR Workshop Proceedings*, pages 265–268. CEUR-WS.org.

Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, Maria-Esther Vidal, Jens Lehmann, and Sören Auer. 2018. Why reinvent the wheel: Let's build question answering systems together. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1247–1256. ACM.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1556–1566. The Association for Computer Linguistics.

Yiming Tan, Dehai Min, Yu Li, Wenbo Li, Nan Hu, Yongrui Chen, and Guilin Qi. 2023. Can chatgpt replace traditional KBQA models? an in-depth analysis of the question answering performance of the GPT LLM family. In *The Semantic Web - ISWC 2023 - 22nd International Semantic Web Conference, Athens, Greece, November 6-10, 2023, Proceedings, Part I*, volume 14265 of *Lecture Notes in Computer Science*, pages 348–367. Springer.

Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. LC-QuAD: A corpus for complex question answering over knowledge graphs. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, volume 10588 of *Lecture Notes in Computer Science*, pages 210–218. Springer.

Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Bastian Haarmann, Anastasia Krithara, Michael Röder, and Giulio Napolitano. 2017. 7th open challenge on question answering over linked data (QALD-7). In *Semantic Web Challenges - 4th SemWebEval Challenge at ESWC 2017, Portoroz, Slovenia, May 28 - June 1, 2017, Revised Selected Papers*, volume 769 of *Communications in Computer and Information Science*, pages 59–69. Springer.

W3C Semantic Web Standards. 2023. Sparql 1.1 overview. https://www.w3.org/TR/sparql11-overview/.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

Hamid Zafar, Giulio Napolitano, and Jens Lehmann. 2018. Formal query generation for question answering over knowledge bases. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 714–728. Springer.