Improving Question-Answering for Portuguese Using Triples Extracted from Corpora

Ricardo Rodrigues[1,2(✉)] and Paulo Gomes[1]

[1] Centre for Informatics and Systems of the University of Coimbra, Coimbra, Portugal
{rmanuel,pgomes}@dei.uc.pt
[2] College of Education of the Polytechnic Institute of Coimbra, Coimbra, Portugal

Abstract. We present here an evolution of a QA system for Portuguese that uses *subject-predicate-object* triples extracted from sentences in a corpus. The system is supported by indices that store those triples, related sentences and documents. It processes the questions and retrieves answers based on the triples.

For purposes of testing and evaluation, we have used the CHAVE corpus, used in multiple editions of the CLEF multilingual QA tracks. The questions from those editions were used to query and benchmark our system. Currently, the system manages to answer up to 42 % of those questions. This document describes the modules that compose the system and how they are combined, providing a brief analysis on them, and also current results, as well as some expectations regarding future work.

Keywords: Question Answering · Open information extraction · Triple extraction · Portuguese

1 Introduction

The quest for information is a quintessential human endeavour. And as soon as computers came into play, they immediately started to be used for storing and retrieving information, most of which in the form of natural language. Not long after, there were attempts to use computers in tasks related to natural language processing (NLP), trying to make sense of all the data described using natural language, which keeps increasing by the day. However, it is not enough to store and retrieve documents, being needed tools that can process them in order to retrieve just what the user wants or needs, instead of just a list of documents.

This issue is addressed by question answering (QA) systems [25], which allow the user to interact with those systems by means of natural language, and process documents whose contents are specified also using natural language.

In this context, we present RAPPort, a system that addresses QA for Portuguese that uses triples extracted from sentences in a corpus, much like open information extraction performs, that are then used to present "short answers" (passages), alongside the sentences and documents they belong to.

J. Silva et al. (Eds.): PROPOR 2016, LNAI 9727, pp. 25–37, 2016.
DOI: 10.1007/978-3-319-41552-9_3

In the remaining document, we present a brief contextualization on QA, address related work, describe the overall used approach and each of its modules, and draw some conclusions and reflections about future work.

2 Question Answering

QA, much like other subfields of information retrieval (IR), may include techniques such as: named entity recognition (NER) or semantic classification of entities, relation extraction between entities, and selection of semantically relevant sentences or chunks [16], beyond customary sentence splitting, tokenization, lemmatization, and part-of-speech (POS) tagging. QA can also address a restricted set of topics, in a closed domain, or forgo that restriction, operating in an open domain. Focusing specifically on open domain QA, it can consist of fundamentally two distinct approaches: IR-based QA or knowledge-based QA [11].

QA systems based on IR typically follow the framework depicted in Fig. 1, where the processing stages are made at run-time, except for document indexing. Knowledge-based QA systems, although sharing some similarities, tend to adopt logical representations of facts, for instance, through the use triples (*subject*, *predicate* and *object*) backed up by ontologies, often implemented by means of RDF triple stores, using SPARQL to query them [26], or similar data repositories.

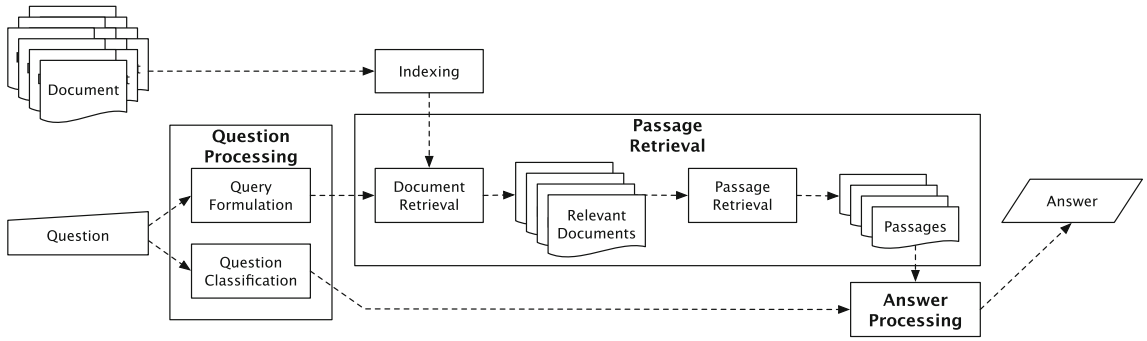

Fig. 1. A typical framework for a IR-based QA system (reproduced [11])

Regarding specific approaches to Portuguese, we present next the most relevant works whose results are compared against our work later on this document.

2.1 Senso

The Senso Question Answering System [22] (alias PTUE [20]) uses a local knowledge base, providing semantic information for text search terms expansion. It is composed of five major modules: *query* (for question analysis), *libs* (for corpora management), *ontology* (for knowledge representation), *solver* (for answer searching), and *web interface*. After all modules are used, the results are merged for answer list validation, to filter and adjust answers weight, ranking them.

2.2 Esfinge

Esfinge [5] is a general domain QA system that tries to take advantage of the great amount of information existing in the Web. Esfinge relies on pattern identification and matching. For each question, a tentative answer beginning is created. Then the probable answer beginning is used to search the corpus, through a search engine, in order to find possible answers that match the same pattern. In the remaining stages of the process, *n-grams* are scored and NER is performed in order to improve the performance of the system.

2.3 RAPOSA

The RAPOSA Question Answering System [24] tries to provide a continuous online processing chain from question to answer, combining stages from information extraction and retrieval. The system involves expanding queries for event-related or action-related factoid questions, using a verb thesaurus automatically generated using information extracted from large corpora. RAPOSA consists of six modules more or less typical on QA systems: a *question parser*, a *query generator*, a *snippet searcher*, an *answer extractor*, *answer fusion*, and an *answer selector*. It deals with two categories of questions: definitions and factoids.

2.4 IdSay

IdSay: Question Answering for Portuguese [3,4] uses mainly techniques from the area of IR, where the only external information that it uses, besides the text collections, is lexical information for the Portuguese language. IdSay uses a conservative approach to QA, being its main stages: *question analysis*, *set Wikipedia answer (SWAN)*, *document retrieval*, *passage retrieval*, *answer extraction* and *answer validation*. IdSay starts by performing document analysis and then proceeding to entity recognition. After that, the system makes use of patterns to define the type of the questions and expected answers. However, contrary to most QA systems, it does not store passages in the IR module, but documents, with the passages being extracted in real time, allowing for more flexibility.

2.5 QA@L^{2}F

QA@L^{2}F [15], the QA system from L^{2}F, INESC-ID, is a system that relies on three main tasks: *information extraction*, *question interpretation* and *answer finding*. The system starts by processing and analyzing the text sources in order to extract potentially relevant information (such as named entities or relations between concepts), which is stored into a knowledge base. Then, the questions are also processed and analyzed, selecting which terms should be used to build a query to search the database. Finally, the retrieved records are then processed, selecting the answer according to the question type and other strategies.

2.6 Priberam

Priberam's Question Answering System for Portuguese [2] is divided in five major modules: *indexation*, *question analysis*, *document retrieval*, *sentence retrieval*, and *answer extraction*. It starts by processing the documents, mainly at sentence level, and storing related data (lemmas, heads of derivation, named entities and fixed expressions, question categories and ontology domains) in different indices. Then each question is processed, extracting and expanding pivots for querying the indices. The resulting queries are used first for retrieving documents based on their scores (using lexical frequency, document frequency, and weighted POS tags) and then for selecting the sentences and extracting the answers, according to matches against the pivotal words in the questions.

2.7 GistSumm

Brazil's Núcleo Interinstitucional de Lingüística Computacional (NILC) had built previously a summarization system, dubbed GistSum [19], that has been adapted for use in the task of monolingual QA for Portuguese texts. NILC's system comprises three main processes: *text segmentation*, *sentence ranking*, and *extract production* [6], associating sentences to a topic. The questions are then matched against the sentences and associated summaries, with the highest scored sentences being used to produce an answer.

3 RAPPort

Our system adheres to most of the typical framework for a QA system, combining aspects from both IR-based QA and knowledge-based QA. It does also improve on some techniques that differ from other approaches to Portuguese.

One of the most identifying elements of RAPPort is the use of triples as the basic unit of information regarding any topic, represented by a *subject*, a *predicate* and an *object*, and then using those triples as a basis for answering questions. This approach also possesses somes characteristics from open information extraction, regarding the extraction and storage of information in triples [8].

The system depends on a combination of four major modules for addressing information extraction, storage, querying and retrieving, namely:

- triple extraction (performed offline);
- triple storage (performed offline);
- data querying (performed online);
- and answer retrieving (performed online).

Each of these modules is described next, specifying the main tasks that compose them. An overview of the modules can also be seen in Fig. 2.

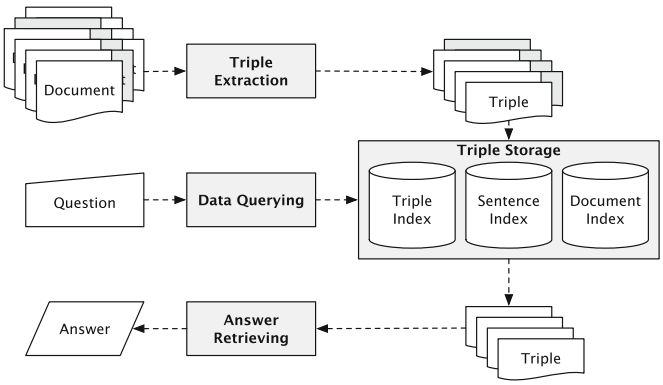

Fig. 2. Our system's general approach

3.1 Triple Extraction

This module processes the contents of the corpus, picking each of the documents, selecting sentences and extracting triples. It includes multiple tasks, namely sentence splitting, phrase chunking, tokenization, POS tagging, lemmatization, dependency parsing, and NER. Except for lemmatization and dependency parsing, these tasks are done using the *Apache OpenNLP toolkit*[1], with some minor tweaks for better addressing Portuguese, and with the models used for chunking and in being specifically created, as there was no available pre-built models.

For the lemmatization process, *LemPORT* [21], a Portuguese specific lemmatizer was used. For dependency parsing, it was used *MaltParser* [18], with a model trained on Bosque 8.0[2] [1]. The output of MaltParser is also further processed in order to group the tokens around the *main* dependencies, such as: subject, root (verb), and objects, among others.

Triple extraction is performed using two complementary approaches, both depending on named entities for determining which triples are of use. The triples are defined by *subject*, *predicate*, and *object*, that are obtained either through the proximity relations between phrase chunks, or through the analysis of the dependencies in sentences. Only the triples with entities in the *subject* or in the *object* are stored for future querying. Also, the predicate has the verb stored in its lemmatized form in order to facilitate later matches.

In the triples that are based on the proximity between chunks, most of the predicates comprehend, but are not necessarily limited to, the verbs *ser* (to be), *pertencer* (to belong), *haver* (to have), and *ficar* (to be located). For instance, if two noun phrase (NP) chunks are found sequentially, and the first chunk contains a named entity, it is highly probable that it is further characterized by the second chunk. If the second chunk starts with a determinant or a noun, the predicate of the future triple is set to *ser*; if it starts with the preposition *em* (in), it is used the verb *ficar*; if it starts with the preposition *de* (of), it is used the verb *pertencer*; and so on. An algorithm describing the process is found in Algorithm 1.

[1] http://incubator.apache.org/opennlp/.
[2] http://www.linguateca.pt/floresta/BibliaFlorestal/completa.html.

```
Data: Corpus documents
Result: Triple list
Read documents;
foreach document do
    Split sentences;
    foreach sentence do
        Tokenize, POS tag, lemmatize;
        Extracts phrase chunks and dependency chunks;
        Extract named entities;
        foreach phrase chunk do
            if chunk contains any entity then
                if neighbouring chunk has a specific type then
                    Create triple relating both chunks, depending on the
                    neighbouring chunk type and contents;
                    Add it to the triple list;
                end
            end
        end
        foreach dependency chunk do
            if chunk contains any entity and is a subject or an object then
                Create triple using the subject or object, the root, and
                corresponding object or subject, respectively;
                Add it to the triple list;
            end
        end
    end
end
```

Algorithm 1. Triple Extraction Algorithm

As an example, the sentence "Mel Blanc, o homem que deu a sua voz a o coelho mais famoso de o mundo, Bugs Bunny, era alérgico a cenouras."[3] yields distinct triples, such as: "{*Bugs Bunny*} {*ser*} {*o coelho mais famoso do mundo*}" and "{*Mel Blanc*} {*ser*} {*o homem que deu a sua voz ao coelho mais famoso do mundo*}", both using the proximity approach, and "{*Mel Blanc*} {*ser*} {*alérgico a cenouras*}", using the dependency approach.

3.2 Triple Storage

After triple extraction is performed, *Lucene* [14] is used for storing the triples, the sentences where the triples are found, and the documents that, by their turn, contain those sentences. For that purpose, three indices were created:

– the triple index stores the triples (subject, predicate and object), their *ids*, and the *ids* of the sentences and documents that contain them;

[3] Loosely translated as: "Mel Blanc, the man who lent his voice to the world's most famous rabbit, Bugs Bunny, was allergic to carrots.".

– the sentence index stores the sentences *ids* (a sequential number representing their order within the document), the tokenized text, the lemmatized text and the documents *ids* they belong to;
– the document index stores the data describing the document, as found in CHAVE (number, *id*, date, category, author, and original text);

Although each index is virtually independent from the others, they can refer one another by using the *ids* of the sentences and of the documents. That way, it is easy to determine the relations between documents, sentences, and triples. These indices (mainly the sentence and the triple indices) are then used in the next steps of the presented approach.

3.3 Data Querying

In a similar way to the sentences in the corpus, the questions are processed in order to extract tokens, lemmas and named entities, and identify their types, categories and targets (although the last three tasks are not currently performed).

For building the queries, the system starts by performing NER and lemmatizing the questions. The lemmas are useful for broadening the matches and results that could be found only by using the tokens. The queries are essentially built on the lemmas found in the questions. All the query elements are, by default, optional, except for named entities. If no entities are present in the questions, proper nouns are made mandatory; by its turn, if there are also no proper nouns, (common) nouns replace them as mandatory keywords in the queries.

For instance, in order to retrieve the answer to the question "*A que era alérgico Mel Blanc?*"[4], the Lucene query will end up being defined by five terms: "*+Mel_Blanc a que ser alérgico*". We have chosen to keep all the lemmas because Lucene scores higher the hits with the optional lemmas, and virtually ignores them if they are not present. The query is then applied to the sentence index. When a match occurs, the associated triples are retrieved, along with the document data. In the same step, when applicable, and for the moment, just synonyms for the verb are added to query as optional items, using the synonymy relations defined in PAPEL [10].

The triples that are related to the sentence are then processed, checking for the presence of the question entities in either the *subject* or the *object* of the triples, for selecting which triples are of interest.

3.4 Answer Retrieving

After a sentence matches a query, as stated before, the associated triples and document data are retrieved — and this goes for all the sentences matching that query. As the document data is only used for better characterizing the answers, let us focus on the triples.

For each triple, it is retrieved each of its components: if the best match against the query is found in the *subject*, the *object* is returned as being the

[4] Loosely translated as: "What was Mel Blanc allergic to?".

answer; if, on the other hand, the best match is found against the *object*, it is the *subject* that is returned. This candidate answer, before being presented to the user, is ordered against other candidate answers. For that, the triples are used once again, as the candidate answers are ordered against the number of triples they are found in. An algorithm describing both data querying and this process is found in Algorithm 2.

```
Data: Question &Indices
Result: Answers
Create query using named entities (or, if inexistent, proper nouns, or nouns) as
mandatory, and the remaining lemmas from the question as optional;
Run query against sentence index;
foreach sentence hit do
    Retrieve triples related to the sentence hit;
    foreach triple do
        if subject contains named entities from question then
            Add object to answers and retrieve sentence and document
            associated with the triple;
        end
        else if object contains named entities from question then
            Add subject to answers and retrieve sentence and document
            associated with the triple;
        end
    end
end
Order answers based in the number of triples they belong to;
```

Algorithm 2. Answer Retrieval Algorithm

Continuing with the example provided earlier, after the correct sentence is retrieved, of the three corresponding triples, the one that best matches the question is "{*Mel Blanc*} {*ser*} {*alérgico a cenouras*}" — there is a match on the *predicate* and the named entity is found in the *subject*. Removing from the triple the terms found in the question, what remains must yield the answer: "[a] cenouras". Besides that, as the named entity, Mel Blanc, is found in the *subject* of the triple, the answer is most likely to be found in the *object*, and so retrieved.

4 Experimentation Results

For the experimental work, we have used the CHAVE corpus [23], a collection of 1456 editions of newspapers "Público" and "Folha de São Paulo", from 1994 and 1995, with each of the editions comprehending about one hundred articles, identified by *id*, number, date, category, author, and the text of article itself.

CHAVE was used in the Cross Language Evaluation Forum (CLEF) multilingual QA tracks for Portuguese [7,9,12,13,27], although in the editions of 2007

and 2008 a dump of the Portuguese Wikipedia was also used in addition — that is the reason, in the present paper, for just being addressing the 2004, 2005 and 2006 campaigns for evaluation purposes.

Nearly all of the questions used in each of the CLEF editions (200 for each language), and respective answers, are known. It is also known the results of each of the contestant systems. The questions used in CLEF adhere to the following criteria [13]: they can be *list* questions, *embedded* questions, *yes/no* questions (although none was found in the questions used for Portuguese), *who*, *what*, *where*, *when*, *why*, and *how* questions, and definitions.

For reference, in Table 1 there is a summary of the best results for the Portuguese QA tasks on CLEF from 2004 to 2008 (abridged [7,9,12,13,27]), alongside with the arithmetic mean for each system comprehending the editions where they were contenders. At the end of the table, it is also shown the current results of our system, for a maximum of ten answers per question.

Table 1. Comparison of the Results at CLEF 2004 to 2008

Approach	Overall Accuracy (%)						
	2004	2005	2006	2007	2008	*(2004–06 Avg)*	*(2004–08 Avg)*
Esfinge	15.08	23.00	24.5	8.0	23.5	*(20.86)*	*(18.82)*
Senso	28.54	25.00	—	42.0	46.5	*(26.77)*	*(35.51)*
Priberam	—	64.50	67.0	50.5	63.5	*(65.75)*	*(61.34)*
NILC	—	—	1.5	—	—	*(1.5)*	*(1.5)*
RAPOSA	—	—	13.0	20.0	14.5	*(13.0)*	*(18.83)*
QA@L^{2}F	—	—	—	13.0	20.0	—	*(16.5)*
IdSay	—	—	—	—	32.5	—	*(32.5)*
RAPPort	41.21	45.00	38.50	—	—	*(41.57)*	—

As already mentioned, we are only addressing the questions for Portuguese used in CLEF in 2004, 2005 and 2006. As such, a grand total of 599 questions[5] were used for testing our system, of which 10 % don't have an answer in the corpus — being 'NIL' the expected answer in that case. That is the reason for considering the average result of our system in Table 1 just for the years 2004 to 2006, and omitting the results for the years 2007 and 2008.

For verifying if the retrieved answers match the expected answers, the answers must contain the already known answers, and the corresponding document *ids* must also match those of the known answers.

Using the set of questions from 2004 to 2006, which were known to have their answers found on CHAVE, we were able to find the answers to 41.57 % of the questions (249 in 599), grouping all the question from the already identified editions of CLEF, with a limit of ten answers for each question. (If that limit is

[5] In 2004, one of the questions was unintentionally duplicated, hence 599 and not 600.

relinquished, the number of answered questions rises to 67.61 %, which may lead to the conclusion that one of the big issues to be further addressed is to improve the ordering and selection of the answers.)

For comparison purposes, a previous version of RAPPort (whose main differences to the current version was not using verb synonyms, and mainly the ranking of the answers, which was then directly related to the score of each Lucene match of the sentences housing the triples against the query generated from the question), for a limit of ten answers per question, achieved 20.75 % of right answers, and without a limit, 43.33 % of right answers.

On the answers that have not been found, we have determined that in a few cases the fault is due to questions depending on information contained in other questions or their answers. There are certainly also many shortcomings in the creation of the triples, mainly on the phrase chunks that are close together, as opposed to the dependency chunks, that should and must be addressed, in order to improve and create more triples. Furthermore, there are questions that refer to entities that fail to be identified as such by our system, an so no triples were created for them when processing the sentences.

5 Conclusions and Future Work

We have come to the conclusion that using triples as a means of representing and storing information found in corpora has strong advantages, besides allowing the exploration of a different way of supporting QA systems for Portuguese.

Firstly, the use of triples, restricting them to those containing named entities, provides a way of selecting which information should really be stored (just the triples and associated sentences), instead of, for instance, storing and indexing all text as a source for providing answers, having to processed the text later. Secondly, triples, being composed mostly of small chunks, already contain in themselves (in the *subject*, *predicate* or *object*) the passage that will be used as the short answer to a question.

Earlier experiments have shown us that trying to store every single bit of information regarding texts in corpora — such as using ontologies for storing syntactic and semantic data, or indexing and storing all and whole sentences — creates considerable overhead and noise, besides having its toll on performance. Using triples in the way described here helps to mitigate these problems.

Although the proposed system scores a strong second place for the three years considered (using solely CHAVE), the use of triples keeps proving to be a promising way of selecting the right and shorter answers to most of the questions addressed. However, there is still a lot that can be improved.

Triples could be improved, namely those that are built from the relations of proximity between chunks, so the system is able to have a number of retrieved triples on par with the sentences that contain the answers (and the triples). Another boost to the approach would be to properly differentiate the queries accordingly to the types of the named entities found in the questions, and improve NER, both on questions an on corpus sentences.

Another aspect that should be considered is the use of coreference resolution in order to increase the number of extracted triples by means of replacing, for instance, pronouns with the corresponding, if any, named entities.

And the system has yet to properly address NIL answers, as it currently provides almost always an answer, even if the match when querying the indices has an extremely low score.

We believe that expanding the queries using the above techniques, together with the creation of better models to extract triples and coreference resolution, will achieve better results in a short time span.

Finally, the next major goal is to use the Portuguese Wikipedia as a repository of information, either alongside CHAVE, to address the latter editions of CLEF, or by itself, as it has happened in Págico [17].

References

1. Afonso, S., Bick, E., Haber, R., Santos, D.: Floresta sintá(c)tica: a treebank for portuguese. In: Rodríguez, M.G., Araujo, C.P.S. (eds.) Proceedings of LREC 2002, The Third International Conference on Language Resources and Evaluation, pp. 1698–1703. ELRA, Paris (2002)
2. Amaral, C., Figueira, H., Martins, A., Mendes, A., Mendes, P., Pinto, C.: Priberam's question answering system for portuguese. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 410–419. Springer, Heidelberg (2006)
3. Carvalho, G., de Matos, D.M., Rocio, V.: IdSay: question answering for portuguese. In: Peters, C., Deselaers, T., Ferro, N., Gonzalo, J., Jones, G.J.F., Kurimo, M., Mandl, T., Peñas, A., Petras, V. (eds.) CLEF 2008. LNCS, vol. 5706, pp. 345–352. Springer, Heidelberg (2009)
4. Carvalho, G., Matos, D.M., Rocio, V.: Robust Question Answering. In: PhD and MSc/MA Dissertation Contest of the of the 10th International Conference on Computational Processing of the Portuguese Language (PROPOR 2012), Coimbra, Portugal, April 2012
5. Costa, L.F.: Esfinge – a question answering system in the web using the web. In: Proceedings of the Demonstration Session of the 11th Conference of the European Chapter of the Association for Computational Linguistics, pp. 410–419. Association for Computational Linguistics, Trento, Italy, April 2006
6. Filho, P.P.B., de Uzêda, V.R., Pardo, T.A.S., das Graças Volpe Nunes, M.: Using a Text Summarization System for Monolingual Question Answering. In: CLEF 2006 Working Notes (2006)
7. Forner, P., Peñas, A., Agirre, E., Alegria, I., Forăscu, C., Moreau, N., Osenova, P., Prokopidis, P., Rocha, P., Sacaleanu, B., Sutcliffe, R., Tjong Kim Sang, E.: Overview of the CLEF 2008 multilingual question answering track. In: Peters, C., Deselaers, T., Ferro, N., Gonzalo, J., Jones, G.J.F., Kurimo, M., Mandl, T., Peñas, A., Petras, V. (eds.) CLEF 2008. LNCS, vol. 5706, pp. 262–295. Springer, Heidelberg (2009)
8. Gamallo, P.: An overview of open information extraction. In: Pereira, M.J.V., Leal, J.P., Simões, A. (eds.) Proceedings of the 3rd Symposium on Languages, Applications and Technologies (SLATE 2014), pp. 13–16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik Dagstuhl Publishing, Germany (2014)

9. Giampiccolo, D., Forner, P., Herrera, J., Peñas, A., Ayache, C., Forascu, C., Jijkoun, V., Osenova, P., Rocha, P., Sacaleanu, B., Sutcliffe, R.F.E.: Overview of the CLEF 2007 multilingual question answering track. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152, pp. 200–236. Springer, Heidelberg (2008)
10. Oliveira, H.G., Santos, D., Gomes, P., Seco, N.: PAPEL: a dictionary-based lexical ontology for portuguese. In: Teixeira, A., de Lima, V.L.S., de Oliveira, L.C., Quaresma, P. (eds.) PROPOR 2008. LNCS (LNAI), vol. 5190, pp. 31–40. Springer, Heidelberg (2008)
11. Jurafsky, D., Martin, J.H.: Speech and Language Processing, 2nd edn. Pearson Education International Inc., Upper Saddle River (2008)
12. Magnini, B., Giampiccolo, D., Forner, P., Ayache, C., Jijkoun, V., Osenova, P., Peñas, A., Rocha, P., Sacaleanu, B., Sutcliffe, R.F.E.: Overview of the CLEF 2006 multilingual question answering track. In: Peters, C., Clough, P., Gey, F.C., Karlgren, J., Magnini, B., Oard, D.W., de Rijke, M., Stempfhuber, M. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 223–256. Springer, Heidelberg (2007)
13. Magnini, B., Vallin, A., Ayache, C., Erbach, G., Peñas, A., de Rijke, M., Rocha, P., Simov, K.I., Sutcliffe, R.F.E.: Overview of the CLEF 2004 multilingual question answering track. In: Peters, C., Clough, P., Gonzalo, J., Jones, G.J.F., Kluck, M., Magnini, B. (eds.) CLEF 2004. LNCS, vol. 3491, pp. 371–391. Springer, Heidelberg (2005)
14. McCandless, M., Hatcher, E., Gospodnetić, O.: Lucene in Action. Manning Publications Co., Greenwich (2010)
15. Mendes, A., Coheur, L., Mamede, N.J., Ribeiro, R., Batista, F., de Matos, D.M.: QA@L^{2}F, first steps at QA@CLEF. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152, pp. 356–363. Springer, Heidelberg (2008)
16. Moens, M.F.: Information Extraction: Algorithms and Prospects in a Retrieval Context. Springer, Heidelberg (2006)
17. Mota, C.: Resultados Págicos: Participação, Resultados e Recursos. Linguamática 4(1), April 2012
18. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryiğit, G., Kübler, S., Marinov, S., Marsi, E.: MaltParser: a language-independent system for data-driven dependency parsing. Nat. Lang. Eng. **13**(2), 95–135 (2007)
19. Pardo, T.A.S., Rino, L.H.M., Nunes, M.G.V.: GistSumm: a summarization tool based on a new extractive method. In: Mamede, N.J., Baptista, J., Trancoso, I., Nunes, M.G.V. (eds.) PROPOR 2003. LNCS, vol. 2721, pp. 210–218. Springer, Heidelberg (2003)
20. Quaresma, P., Quintano, L., Rodrigues, I., Saias, J., Salgueiro, P.: The University of Évora approach to QA@CLEF-2004. In: CLEF 2004 Working Notes (2004)
21. Rodrigues, R., Gonçalo-Oliveira, H., Gomes, P.: LemPORT: a high-accuracy cross-platform lemmatizer for portuguese. In: Pereira, M.J.V., Leal, J.P., Simões, A. (eds.) Proceedings of the 3rd Symposium on Languages, Applications and Technologies (SLATE 2014). pp. 267–274. Germany (2014)
22. Saias, J., Quaresma, P.: The senso question answering approach to portuguese QA@CLEF-2007. In: Nardi, A., Peters, C. (eds.) Working Notes for the CLEF 2007 Workshop, Budapest, Hungary, September 2007
23. Santos, D., Rocha, P.: The key to the first CLEF with portuguese: topics, questions and answers in CHAVE. In: Peters, C., Clough, P., Gonzalo, J., Jones, G.J.F., Kluck, M., Magnini, B. (eds.) Multilingual Information Access for Text, Speech and Images. LNCS, vol. 3491, pp. 821–832. Springer, Heidelberg (2005)

24. Sarmento, L., Oliveira, E.: Making RAPOSA (FOX) smarter. In: Nardi, A., Peters, C. (eds.) Working Notes for the CLEF 2007 Workshop, Budapest, Hungary, September 2007
25. Strzalkowski, T., Harabagiu, S. (eds.): Advances in Open Domain Question Answering, Text, Speech and Language Technology, vol. 32. Springer, Heidelberg (2006)
26. Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.C.N., Gerber, D., Cimiano, P.: Template-based question answering over RDF data. In: Proceedings of the 21st International Conference on World Wide Web (WWW 2012), pp. 639–648. ACM Press, Lyon, France, April 2012
27. Vallin, A., Magnini, B., Giampiccolo, D., Aunimo, L., Ayache, C., Osenova, P., Peñas, A., de Rijke, M., Sacaleanu, B., Santos, D., Sutcliffe, R.F.E.: Overview of the CLEF 2005 multilingual question answering track. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 307–331. Springer, Heidelberg (2006)