TendeR-Sims
Similarity Retrieval System for Public Tenders

Guilherme Q. Vasconcelos[1], Guilherme F. Zabot[1], Daniel M. de Lima[1],
José F. Rodrigues Jr.[1], Caetano Traina Jr.[1], Daniel dos S. Kaster[2], Robson L. F. Cordeiro[1]
[1]*Institute of Mathematics and Computer Sciences, University of São Paulo, São Carlos, Brazil,*
[2]*Computer Science Department, State University of Londrina, Paraná, Brazil*

Keywords: Relational Databases, Division Operator, Similarity Comparison, Complex Data, Ontology, Public Tendering.

Abstract: TendeR-Sims (*Tender Retrieval by Similarity*) is a system that helps to search for satisfiable request for tender's lots in a database by filtering irrelevant lots, so companies can easily discover the contracts they can win. The system implements the Similarity-aware Relational Division Operator in a commercial Relational Database Management System (RDBMS), and compares products by combining a path distance in a preprocessed ontology with a textual distance. Tender-Sims focuses on answering the following query: select the lots where a company has a similar enough item for each of *all* required items. We evaluated our proposed system employing a dataset composed of product catologs of Brazilian companies in the food market and real requests for tenders with known results. In the presented experiments, TendeR Sims achieved up to 66% cost reduction at 90% recall when compared to the ground truth.

1 INTRODUCTION

The initial step of a public tendering process is to publish a Request For Tender (RFT) document outlining the agency's needs, requirements, criteria and instructions. When the RFT is designed to acquire goods, the later are usually grouped in lots, which can have the requirement that a bidding company should provide all the goods listed. RFTs are semi-structured documents, as their structure may vary for each specific request and agency, and product/service descriptions are usually written in natural language. Moreover, these documents are commonly organized by grouping products and/or services in lots to which a business must be able to supply all of them to qualify for the tender. These characteristics makes challenging to automatically match companies to tenders.

It is interesting to note that the Division operator from Relational Databases is well-suited to find out whether or not a given business can provide all products of a lot, since this operator was designed to answer queries with the idea of "for all", such as "*Which companies have all products of a given lot?*". However, one cannot expect that the product descriptions presented in a lot will be exactly identical to its counterparts in a business catalog, thus, comparing descriptions by identity (=) is useless for this type of data.

Whenever identity comparison cannot be used in a given setting, comparing the data by similarity may be the best solution (Zezula et al., 2010). Similarity queries, in essence, analyze data objects represented by feature vectors extracted from intrinsic data information. For example, images are usually compared against each other using numeric features that represent color, texture or shape patterns extracted from their visual content. For textual data, on the other hand, a *bag-of-words* representation defines a feature vector where each element indicates the presence or absence of a specific word in the sentence; a TF-IDF representation (Rajaraman and Ullman, 2011) augments the bag-of-words by making each value quantify the frequency of a word in the sentence normalized by its frequency in all sentences; and so on. A distance function is then used to measure similarity between elements of a dataset using the chosen representation.

The inclusion of similarity queries is usually done by means of extending operators of the Relational Algebra, such as the Relational Division operator. This operator was extended to support similarity comparison (Gonzaga and Cordeiro, 2017) and it is well-

Vasconcelos, G., Zabot, G., de Lima, D., Rodrigues Jr., J., Traina Jr., C., Kaster, D. and Cordeiro, R.
TendeR-Sims.
DOI: 10.5220/0006697601430150
In *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018)*, pages 143-150
ISBN: 978-989-758-298-1

suited to answer queries with an idea of "candidate elements and requirements" involving complex data objects that despite of being distinct, can be similar to each other — which is the case in our application over product descriptions (semi/non-structured text).

In this context, we propose the Tender Retrieval By Similarity system — TendeR-Sims, a system capable of comparing product descriptions in request for tender lots with those in a business catalog in order to filter out irrelevant lots — those not satisfiable by the company. TendeR-Sims takes advantage of the aforementioned Similarity-aware Relational Division operator to answer queries with the idea of "for all" in textual data. The similarity is achieved by comparing elements from RFT lots and business catalogs against elements in an ontology that organize terms according to their semantics in a given tender domain. Elements from the RFT lots and business catalogs are said similar if their closest element in the ontology belongs to the same group.

To validate our proposals, we performed a case study using data from real RFTs and business catalogs in the context of the Brazillian food market. In this study, our TendeR Sims system was highly effective with up to 66% cost reduction at a 90% recall threshold when searching request for tenders where a given business is most likely to be able to attend. Our main contributions in this work are as follows.

1. **Prospective Client Identification** – We carefully developed an automatic mediator system between public organizations and private companies targeted at reducing the efforts of the latter when selecting RFTs to bid for.
2. **Case Study** – We validated our system with real data in the context of food RFTs in Brazil. To make it possible, we also developed and published:

2.1. A novel food ontology in Brazilian Portuguese;

2.2. An example dataset of Brazilian RFTs and food business catalogs.

The remainder of this article is organized as follows: Section 2 presents the background word and related work, Section 3 focuses the development methodology of our system proposal, Section 4 reports the results obtained in our case study with Brazilian RFTs, and Section 5 concludes this article.

2 BACKGROUND AND RELATED WORK

This section presents the background on similarity queries, the division operator and the related work on computer solutions to recover and filter data from tendering process and the inclusion of similarity queries on RDBMS.

2.1 Similarity Queries

Complex data like images, audio, fingerprints, textual descriptions and graphs are typically manipulated by similarity. The queries require the dataset represented as feature vectors and a distance function to evaluate the dissimilarity between pairs of data elements.

The inclusion of similarity queries in RDBMS can be performed either by extending the core of the system or by implementing external resources that execute similarity-aware operations and return information to the core. For instance, **PostgreSQL-IE** (Guliato et al., 2009) and **Kiara** (Oliveira et al., 2016) encapsulate similarity query algorithms in User Defined Functions in PostgreSQL while **SimDB** (Silva et al., 2010) is an extension of PostgreSQL's core. **Siren** (Barioni et al., 2006), **SimbA** (Bedo et al., 2014) are similarity retrieval middlewares that recognizes an extended SQL syntax by intercepting the canonical query, executing similarity operations externally and rewriting the query with stardard SQL statements. Finally, **FMI-SiR** (Kaster et al., 2010) is a module that use interface and resource extensions provided by Oracle to implement DBMS-linked data types, feature extractors, distance functions and indexes for similarity-based retrieval.

Most researchers have been extending relational operators to support similary queries based on two approches (Zezula et al., 2010): a) the *range query*, which selects all elements that are similar enough to a query center element, up to a predefined threshold; and b) *k-Nearest Neighbors* (*k*-NN) query, which returns the *k* most similar elements to the query center.

One of the extended operators is the **Similarity-aware Division** (Gonzaga and Cordeiro, 2017). This new operator is able to evaluate complex data and answers queries with the concept of "for all". As we discussed in the introductory section, the Similarity-aware Division is a basis to our work, so we detail it in the next subsection.

2.2 Similarity-Aware Division

The Division operation from the Relational Algebra directly corresponds to the Universal Quantification from the Relational Calculus (Codd, 1972). It allows writing queries involving the concept of "*for all*" in a simple and intuitive manner, such as "*Which companies can provide all products listed in a request for tender's lot?*"

To illustrate the division, let us consider this query about RFT lots and business catalogs, as well as the example relations shown in Table 1. In this setting, relations `CompanyProducts` and `RFTLotProducts` list, respectively, the products in the catalog of four distinct businesses (Table 1a) and the products in a given tender lot (Table 1b). The relational division between these two relations produces a third relation, Company (Table 1c), which contains the companies that can provide all products required in the lot, containing exactly the same descriptions, and, therefore, can bid for that lot.

However, the description of each product or service listed in a RFT rarely matches exactly the corresponding description found in business catalogs. This happens for a number of reasons, like synonym representation, mispelling and the addition of extra information such as trademarks or technical specifications. So, the relational division operator may miss companies that fully match an RFT lot but are discarded due to small differences in product/service descriptions.

As we mentioned earlier, long texts are commonly compared by similarity and not equality, thus the similarity-aware division is better suited to answer such queries. The **Similarity-aware Division** operator (Gonzaga and Cordeiro, 2017) relaxes attribute comparisons that are intrinsic to the traditional division, originally performed by identity (=), by redefining them to be compared by similarity according to an user-defined distance function and a threshold. In this setting, tuples of relation `CompanyProducts` (lines 4-6 in Table 1a) match by similarity tuples of relation `RFTLotProducts` (Table 1b), thus providing a more complete answer by adding company *Company #2* to the result (Table 1c).

2.3 Computer-aided Tendering Processes

In several countries, whenever a public sector agency — such as the Legislative and Judicial Branch, the Public Ministry, the Court of Accounts, etc. — needs to buy goods or request a service, it must open a tendering process. This administrative procedure is public and garantees market competition, as any private business can participate in, and isonomy as the business that is best-suited to attend a tendering will be the one selected (Batista and Prestes, 2004).

After the companies submit their proposal to be evaluated, each tender will be verified and the best proposal wins. There are four types of tenders in Brazil (Mazza, 2011): *lowest price*, *best technique*, *technique and price* and *highest bid*. The types may vary from scenario to scenario. For instance, the Ministry of Defence usually demands the best technique over price, whereas a school requiring a catering company will probably go for the lowest price.

In regards to information systems for consulting Brazilian tenders, the Ministry of Planning, Budget and Management released, in 2007, the **Comprasnet** system (Ministério do Planejamento Desenvolvimento e Gestão do Governo Federal do Brasil, 2007). It is an online tool developed to publish information about tendering processes and hirings promoted by the Federal Brazilian Government. An interesting feature is the eletronic tender dealing. It also aims to provide easy access to request for tenders documents so that companies are encouraged participate in.

Following the same perspective, in 2017, an online governmental system named **Painel de Preços** (Ministério do Planejamento Desenvolvimento e Gestão do Governo Federal do Brasil, 2017) has been released. It allows the storage, analysis and comparison of reference prices in the acquisition of goods and contracting services for public administration. The tool allows anyone to compare prices for a particular item, possibly filtering the data by date or country regions, displaying details of previous purchases for the purpose of establishing a reference price.

These systems are focused on presenting information regarding RFTs for the general audience and lack more advanced features like manipulating or filtering the requirements themselves. As the authors did not find another tendering application in Brazil with more advanced features, we propose TendeR-Sims. TendeR-Sims is focused on filtering Request for Tender documents to reduce amount of effort needed to search for RFTs to bid in, thus helping prospective companies to find contracts more easily.

3 METHODOLOGY

This section details our proposal, the *TendeR-Sims* system – Tender Retrieval by Similarity. Provided that textual product descriptions are better compared by similarity and such form of manipulation is still not properly supported by RDBMS, it becomes necessary to implement new functionalities in the RDBMS environment in order to execute queries that use the similarity-aware division operator.

We acquired request for tender documents and business product catalogs from online websites and created a food ontology to semantically group different representations for similar products.

Then, we carefully designed a novel distance function well-suited to compare product descriptions by similarity, based on the ontology and the Jaccard

Table 1: Example of division to select companies able to provide all products listed in an RFT lot. Notice that CompanyProducts (a) with a darker shade are not present in the lot, so they are not matched to any product in the table RFTLotProducts (b).

(a) `CompanyProducts`

Company	Product
Company #1	bread
Company #1	butter
Company #1	milk
Company #2	white bread
Company #2	salted butter
Company #3	light butter
Company #3	tomato ketchup
Company #4	bread
Company #4	butter
Company #4	popcorn
Company #4	raw fish

$\div$ (b) `RFTLotProducts`

Product
bread
butter

$=$ (c) `Companies`

Company
Company #1
Company #4

distance. It is worth to notice that similarity measure developed, in which we use the bag-of-words representation and Jaccard distance, was chosen for simplicity. Any other text representation model and similarity metric can be used in TendeR-Sims. The text representation and distance function is effectively a parameter of the SQL query. Finally, we implemented the similarity-aware division operator in a commercial DBMS by extending the FMI-SiR similarity retrieval module [1].

3.1 Data Acquisition

The tender documents were collected from Brazilian governmental agencies and non-governmental organizations websites: the Social Commerce Service of the state of Paraná (SESC·PR)[2], the Regional Council of Engineering and Agronomy of the State of Paraná (CREA·PR)[3], and the Camboriu/SC City Administration website[4] — all written in Brazilian Portuguese.

These websites provide tender documents in Adobe's Portable Document Format (PDF), in such a way that any one can have download these documents freely. Only food-related tenders that have been already completed were collected, that is, those that have already been taken by some company. A total of nine RFT documents were obtained in this process and 18 companies catalogs were collected.

Based on the defined scope of the tenders, a survey of the companies participating in the bidding process was carried out in order to create a database from their products. We did not have access to the internal database of the companies participating in the tenders, so we decided to carry out the extraction of the product information from their websites. For this, *web-crawler* clients were used, capable of capturing and extracting the information presented in their product catalog. Data was then processed to remove information related to brands, trademarks, units of measure, quantity as well as symbols, stopwords and any other information that was irrelevant about the product itself. Part of the normalized data was used to build the ontology and the other part was used in our experiments.

3.2 Ontology

The leaf nodes represent unique product categories and the parent nodes are generalizations of these categories, creating a hierarchical structure in which products share common nodes. we defined categories and subcategories to represent products of a food domain in Brazilian Portuguese in a hierarchical structure, because we did not found any such an ontology available in this language. In Figure 1, the leaves are illustrated as rectangles and circles represent leaf nodes[5]. Thus, "turkey fillet" and "chicken fillet" are placed in different leaf nodes, however, both are types of fowl, so their ontology relation branches on the fowl node, meaning they are different subcategories of fowl products. In the leaf nodes there are many textual representation of the product. This representations are used to compare similarity between products. As shown in figure 1, the leaf product "whole grain" contains following representations: "whole grain bread vitãogrão 12 grains", "supreme whole grain bread apple and nut" and "whole oatmeal bread".

We preprocessed the item descriptions in both the RFTs and business catalogs to generate the ontology,

[1]To access the source code visit https://bitbucket.org/tendersimsproject/tendersims/

[2]http://www.sescpr.com.br/sesc-parana/licitacoes

[3]http://www.crea-pr.org.br/ws/licitacoes-do-crea-pr

[4]http://www.camboriu.sc.gov.br/licitacoes_compra.php

[5]Our ontology is in Portuguese, but we translated to English in the figure for easier explanation.

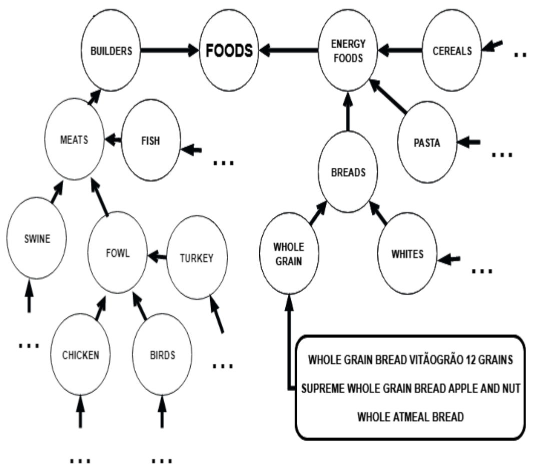

Figure 1: Overview of our ontology with its classes and subclasses.

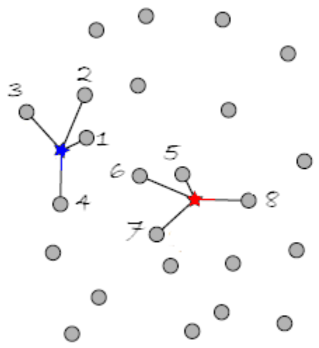

Figure 2: Illustration of the elements of the ontology (in gray), the RFT's text (in blue) and the business catalog's (in red). Elements labeled from 1 to 4 are the 4-nearest neighbors of the RFT's element and the elements labeled from 5 to 8 are the 4-nearest neighbors of the business catalog's element.

which is one of the knowledge bases of our method. The ontology provides useful semantics when dealing with synonyms with different spellings, e.g., "whole grain bread vitãograão 12 grains" and "white bread bauducco", which could be considered similar snacks in an university restaurant, but different in a more restrictive diet.

The creation of the categories represented in Figure 1 is based on the food pyramid, in which the food domain is divided into four large groups: a) the energetics – the bases of human nutrition such as bread, pasta and other products rich in carbohydrates; b) the builders – protein-rich foods; c) the regulators – known for their high content of vitamins and minerals; and d) the extra energetics – sugars, oils and fats. However, we also inserted new nodes to represent products that do not fit into the four large groups, such as industrialized products, as is the case with canned food, colorants and flavoring agents. Each product description were manually added to an initial ontology category according to their position in the food pyramid.

An example is the breads subcategory, which derives from the energetics. Based on the ontology created, methods that use comparison by similarity can be implemented using the hierarchy present in the ontology to improve the result returned by the comparison.

3.3 Similarity Comparison

We measure two types of similarity: lexical and semantical. To measure the lexical similarity, the leaves elements are represented as a bag-of-words in such a way that the text is represented as a sorted array with the length equal to the amount of distinct words. Then, the bag-of-words is evaluated using the Jaccard distance, which is the relation between the amount of exclusive words (set difference) and the overall amount of distinct words (set union). As an example, let us consider the comparison between "whole grain bread" and "whole grain bread loaf", with respective bag-of-words representations [1,1,1,0] and [1,1,1,1]. The distance between these two elements is 0.25, since there are three matching words out of four possible matches. The smaller the value, the more similar the pair of elements is.

This similarity calculus is represented in Figure 2. The element in blue is the text of a product in a RFT lot, the element in red is the text of the product in the business catalog and the gray dots are the representations of the ontology's leaf nodes elements. As seen in the picture, the four closest elements to the text of the product in the RFT lot are the elements 1, 2, 3 and 4 while the four closest elements to the text of the product in the business catalog are 5, 6, 7 and 8.

The semantical similarity is achieved by the ontologic categorization of the elements. If one of the k-nearest elements of the product's text in the RFT's lot and one of the k-nearest elements of the business catalog's text belongs to the same category, the ontology consider the product in a RFT lot and the product in the business catalog to be similar.

3.4 Implementing TendeR-Sims in a commercial RDBMS

To implement our proposed TendeR-Sims system, we extended the FMI-SiR similarity retrieval module. This module offers resources such as distance functions, feature extractors and database indexes for complex data. FMI-SiR is implemented in C++ as an Oracle library that is controled by the query processor.

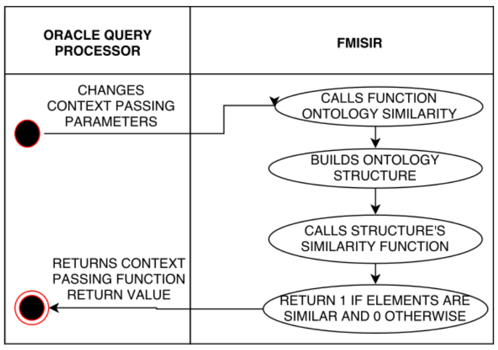
ORACLE QUERY PROCESSOR
FMISIR
CHANGES CONTEXT PASSING PARAMETERS
CALLS FUNCTION ONTOLOGY SIMILARITY
BUILDS ONTOLOGY STRUCTURE
CALLS STRUCTURE'S SIMILARITY FUNCTION
RETURNS CONTEXT PASSING FUNCTION RETURN VALUE
RETURN 1 IF ELEMENTS ARE SIMILAR AND 0 OTHERWISE

Figure 3: Sequence of operations to execute similarity-aware division. During the query execution, the DBMS query processor execute similarity operation externally.

This allows applications to manipulate complex data and use of Oracle database resources together.

Specifically, we inserted our ontology, the Jaccard distance function and the similarity function OntologySimilarity (which evaluates pairs of text segments) inside the module. To associate our external similarity function to the schema, one must run the folowing command on Oracle:

```
CREATE FUNCTION
OntologySimilarity
(text1 varchar2, text2 varchar2,
ontologyPathFile varchar2, k number)
RETURN float
AS
LANGUAGE C LIBRARY libfmisir
NAME "OntologySimilarity"
WITH CONTEXT;
```

This command creates a function that takes, as parameters, two texts and the number of neighbouring elements and a structured file containing the relationship between leaves elements and categories. The implementation is within the FMI-SiR module. When this function is invoked in a SQL query, the DBMS changes context from SQL to C++, as shown in Figure 3. The function loads the ontology into memory and evaluates whether or not the pair of product descriptions received is classified to the same node. The function returns one if a pair of elements is similar and zero otherwise. The pseudo-code this function is illustrated in Algorithm 1 and Algorithm 2. The later represents the function in the ontology structure that evaluates similarity between elements as descripted in section 3.3. This result is sent back to the RDBMS, which continues to process the query.

Provided that there is no explicit SQL implementation for the division, it must be represented through the combination of existing operators. We used double negation for this task, that is, we selected companies that do not have any product that are not similar to a given product from a given lot of a request for tender, i.e lot one of RFT of id one. The corresponding code is given as follows:

```
SELECT DISTINCT company
FROM CompanyProducts this
WHERE NOT EXISTS (
SELECT textDescription
FROM RFTLotProducts request
WHERE lot = 1 AND RFTID = 1 AND NOT EXISTS (
SELECT * FROM CompanyProducts product
WHERE product.company = this.company
AND OntologySimilarity(
request.textDescription,
product.textDescription,
"ontology_file", 2
) = 1
));
```

This concludes our methodology. The next Section will present and discuss the experimental evaluation of this method with the database we collected.

```
Data: text1, text2, k, ontologyFilePath
Result: a value representing if elements are
        similar or not
begin
    creates ontology structure based on
      relationships in the ontologyFilePath; if
      ontologyStructure.similar(text1,text2,k)
      then
        return 1;
    return 0;
```

Algorithm 1: Skeleton of the OntologySimilarity algorithm. text1 and text2 are the texts from the RFT lot's product and the text from the business catalog's product. This function is invoked by the query processor, builds ontology in memory and uses the structure's function "similar" to measure similarity between the elements.

4 EXPERIMENTS

To evaluate TendeR-Sims in a meaningful context, we performed a semi-supervised case study in a realistic setting: a food distributor looking for contracts in the public sector using the data acquired in section 3.1. Our case study focused on the product description for validation purposes, and we intend to consider other attributes in future work.

We randomly selected 20% of the RFT lots and manually labelled all the companies to *satisfy* (real positives) or *not satisfy* (real negatives) each RFT lot. This resulted in a set of 35 RFT lots which were used as our evaluation database, with a positive prevalence of 0.163, i.e., on average each company satisfies 16.3% of all lots.

```
Data: text1, text2, k
Result: a value representing if elements are
        similar or not
begin
    initialize an empty list of leaf's names
      and distances to text1;
    initialize an empty list of leaf's names
      and distances to text2;
    foreach leaf in ontologyStructure do
        foreach textRepresentation in leaf do
            calculate jaccardDistance
              between textRepresentation and
              text1;
            save value and leaf's name in the
              list for text1;
            calculate jaccard distance
              between textRepresentation and
              text2;
            save value and leaf's name in the
              list for text2;
    foreach item in the list for text1 do
        foreach item in the list for text2 do
            if leaf's name in item for text1 is
              equal to leaf's name in item for
              text2 then
                return 1;
    return 0;
```

Algorithm 2: Skeleton of the similarity evaluation algorithm. This function is implemented in the Ontology-Strucure class and it is invoked by Algorithm 1. It takes two texts and a number of neighboring elements as parameters. In the leaf nodes object there are information about the leaf's name and all the textual representations.

From this evaluation database, we selected two companies catalogs (11.1%) to "train" our model — only the products in these two catalogs were preprocessed and added to the ontology. A total of 3,068 products categories are present in this training list, and are further filtered by a more strict list of stopwords to better reflect the semantics of the product categories present in the RFT lots. The words removed includes numbers, manufacturer and product proper names. The resulting list has 2,229 product categories, which were then put into the ontology. The other companies were used as the test dataset.

For each one of the RFT lots in the test dataset, TendeR-Sims queries the companies satisfying all products in the lot. This behaviour make this system act as a *filter* which removes irrelevant items from the analytical workflow. Those items are the lots that cannot be satisfied by the company. The efficiency of this filter is measured by *how much* manual work can be avoided by the user when she uses TendeR-Sims.

The power of such a filter can be expressed by the measures summarized in Table 2. **Specificity** tells how many irrelevant lots are left out of the answer, which should be as high as possible in order to reduce the number of RFTs that will be analyzed manually by the user; **Recall** informs how many of relevant lots are returned in the query, which should be high in order to not ignore possibly satisfiable contracts; the value of **Precision** is how many of the returned lots are indeed satisfiable, which should be high, so the user does not spend time analyzing irrelevant lots; **F1-Score** is the harmonic average of Precision and Recall, which summarizes the overall predictive and selective power for the positive class.

Table 2: Evaluation results.

	$k = 1$	$k = 2$
Specificity	0.958	0.805
Precision	0.740	0.474
Recall	0.607	0.902
F1-score	0.667	0.621
ECR	0.762	0.656

According to the expectations in Table 2 at $k = 1$, if we use TendeR-Sims to filter these RFTs, the system will answer with ≈ 5 RFT lots on average, where 3 or 4 (≈ 3.7) of those can really be satisfied by the company. The Precision of this answer is $3.7 / 5 \approx 74\%$, and the Recall is $3.7 / 6 \approx 61\%$.

This means we can find (recall) $\approx 61\%$ of relevant RFT lots by using TendeR-Sims then manually verifying only the five lots that were returned in the answer. For comparison, if we want to find the same amount of relevant lots by hand, we could randomly shuffle and select 21 lots (60% of the dataset), then verify all the selected lots manually. The number of objects to be manually verified in this case was reduced from 21 to 5, a 76.2% cost reduction.

By adjusting this k — the similarity threshold, it is possible to recall more RFT lots by trading off precision. In Table 2 we observe that setting $k = 2$ raises Recall to 90.2% and lowers Precision to 47.4%. In this case, TendeR-Sims will return 11 results, where 5 or 6 (≈ 5.4) of them will be relevant lots for the company. Thus, to find 90% of the relevant RFT lots, the manual effort is reduced from 32 (randomly picking 90% of the dataset) to 11 objects by using TendeR-Sims, a 65.6% cost reduction for achieving 90% recall. It still is a significant cost reduction to find nearly all (90%) of the relevant items in the database.

This **Estimated Cost Reduction** (ECR in Table 2) is then an expected estimate of how many RFT lots will not be needed to be manually processed if the user wants find the same proportion (recall) of relevant RFT lots from the database. Larger databases will benefit even more from our system, for example, a company looking for tendering any RFT lot in the country may need to analyze tens of thousand lots. As the ECR is a ratio, proportinally many hours may be saved by TendeR-Sims. We then conclude that TendeR-Sims can reduce the amount of manual work and the associated cost by a significant amount.

5 CONCLUSION

TendeR-Sims is a system carefully designed to filter requests for tender (RFTs) according to whether or not a company of interest is suited to tender them.

Considering that product descriptions in a request document may differ from the companies product catalogs by little variations, so any comparison by identity (=) will retrieve only exact matches and leave out relevant items, making this approach unsuited for this type of data. On the other hand, comparing complex data is better addressed by how similar (or inversely, how distant) their are. Thus, our system relies on the Similarity-aware Relational Division Operator.

We created a dataset, proposed an ontology and extended a similarity retrieval module so that all data manipulation runs inside a well-known commercial RDBMS (Oracle 12c).

Our results show that TendeR-Sims can significantly reduce the number of manually analyzed RFT lots and their overall associated cost by up to 76% of the original amount. As future work we plan to improve the ontology, the similarity measure and explore other ontology domains.

ACKNOWLEDGEMENTS

Special thanks to the Databases and Images Group of ICMC/USP, to the CAPES, CNPq and FAPESP funding agencies, to the ICMC/USP staff and also to the many individuals that contributed to this study.

REFERENCES

Barioni, M. C. N., Razente, H. L., Traina, A. J. M., and Traina Jr., C. (2006). SIREN: A similarity retrieval engine for complex data. In *Proceedings of the 32nd International Conference on Very Large Data Bases*, pages 1155–1158.

Batista, H. G. and Prestes, C. (2004). *Guia valor econômico de licitações*. Globo Livros.

Bedo, M. V. N., Traina, A. J. M., and Traina Jr., C. (2014). Seamless integration of distance functions and feature vectors for similarity-queries processing. *JIDM*, 5(3):308–320.

Codd, E. F. (1972). Relational completeness of data base sublanguages. *IBM Research Report*, RJ987.

Gonzaga, A. S. and Cordeiro, R. L. F. (2017). The similarity-aware relational division database operator. In *Proceedings of the Symposium on Applied Computing, SAC 2017*, pages 913–914.

Guliato, D., Melo, E. V., Rangayyan, R. M., and Soares, R. C. (2009). POSTGRESQL-IE: an image-handling extension for postgresql. *J. Digital Imaging*, 22(2):149–165.

Kaster, D. S., Bugatti, P. H., Traina, A. J. M., and Traina Jr., C. (2010). Fmi-sir: A flexible and efficient module for similarity searching on oracle database. *Journal of Information and Data Management*, 1(2):229–244.

Mazza, A. (2011). *Manual de Direito Administrativo*. Saraiva.

Ministério do Planejamento Desenvolvimento e Gestão do Governo Federal do Brasil (2007). Comprasnet. https://www.comprasgovernamentais.gov.br/index.php/comprasnet-siasg. Accessed: 2017-10-08.

Ministério do Planejamento Desenvolvimento e Gestão do Governo Federal do Brasil (2017). Painel de Preço. http://paineldeprecos.planejamento.gov.br/perguntas.html. Accessed: 2017-10-06.

Oliveira, P. H., Fraideinberze, A. C., Laverde, N. A., Gualdron, H., Gonzaga, A. S., Ferreira, L. D., Oliveira, W. D., Rodrigues Jr., J. F., Cordeiro, R. L. F., Traina Jr., C., Traina, A. J. M., and Sousa, E. P. M. (2016). On the support of a similarity-enabled relational database management system in civilian crisis situations. In *Proceedings of the 18th International Conference on Enterprise Information Systems, ICEIS 2016*, pages 119–126.

Rajaraman, A. and Ullman, J. D. (2011). *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

Silva, Y. N., Aly, A. M., Aref, W. G., and Larson, P. (2010). Simdb: a similarity-aware database system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010*, pages 1243–1246.

Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2010). *Similarity Search: The Metric Space Approach*. Springer Publishing Company, Incorporated, United States, 1st edition.