CLEI electronic journal, Volume 25, Number 3, Paper 3, November 2022

Semantic Similarity of Product and Service Names in Portuguese: An Approach Based on Onto.PT

Eduardo Corrêa Gonçalves
Escola Nacional de Ciências Estatísticas (ENCE-IBGE),
Rio de Janeiro, Brazil, 20231-050
eduardo.correa@ibge.gov.br

Abstract

The problem of conceptual comparison of names plays an important role in the field of natural language processing. In this task, the goal is to choose, among a set of names, which one refers to the same concept or object as a given input name. In this paper, we propose an algorithm for comparing names of products and services in Portuguese based on a hybrid formula that simultaneously takes into account the analysis of alphabetic, lexical and semantic similarity. The semantic similarity between two names is calculated using information from Onto.PT, the largest public lexical ontology for the Portuguese language (which was originally built over European Portuguese). Experiments were conducted on a dataset composed of about 5,000 pairs of names of products and services in Brazilian Portuguese. Our experimental results show that the algorithm based on Onto.PT is more effective than other well-known algorithms for name comparison, producing the highest F1-Score. Moreover, results also provide interesting insights into the advantages and disadvantages of using Onto.PT for assessing the semantic similarity of names and other kinds of short texts.

Keywords: Semantic Similarity, Portuguese, Short Text, Lexical Ontology, Onto.PT

1 Introduction

Several modern applications allow users to search for a particular entity – such as a product, place, person, company, etc. – using alternative names that might be similar but not identical to the entity's actual name. As an example, consider a system in which users could enter the name of a food product to consult its price variation over a period of time. Suppose “white bean” is one of the products in the system’s database. In real-world scenarios, users interested in obtaining information about this product would likely to perform searches using different terms, such as:

- “navy bean” – in this case, a synonym was entered by the user.
- “wh bean” – the word “white” was abbreviated.
- “white bwan” – a typo was made.
- “bean” – since the word “white” was omitted, the system should be capable of presenting the results for all types of beans in the database (“white bean”, “adzuki bean”, “pinto bean”, “black bean” etc.).

The above is an example of a problem of conceptual comparison of names [1-3], in which the goal is to determine whether or not two names correspond to alternative designations of the same semantic entity. Besides product names, other important practical applications are the matching of institution names (“UNAM” × “Universidad Nacional Autónoma de México”), addresses (“Main Avn” x “Main Ave”), toponyms (“Ting Tsi River” × “Tingtze River”), and brand names (“Red Bull” × “Brown Ox”), among others. The common characteristic of all these applications is the fact that the names to be compared tend to be very short, usually composed of less than six words [4].

For certain problems, the use of a technique that solely compares the characters that compose each name is sufficient to guarantee effective results. One of such techniques is the well-known edit distance algorithm [5], which has successfully been used to determine the similarity of addresses, toponyms and personal names [6-8]. This algorithm infers how similar two strings are based on the number of edits (character deletions, insertions, or substitutions) it takes to change one string into the other. The smaller the number of edit operations, the more similar

the strings are. For example, according to this approach the distance between *n1* = "Avn" and *n2* = "Ave" is 1, since we only need to change the character "n" for "e" in order to transform *n1* into *n2*.

Nonetheless, determining the similarity between two names by only comparing their characters is not suitable for problems where semantics plays a relevant role. One example of such problem is the matching of product and service names. As shown before, in this problem different names can be used to represent the same product, such as "white bean" and "navy bean". However, note that although "white bean" and "navy bean" denote the same entity, the edit distance value between these names is equal to 5. Such a large value would lead one to mistakenly believe that they are two different products. Moreover, products might be taxonomically related, such as "black bean", "adzuki bean", "pinto bean", and "white bean" which are all different types of beans. Thus, a similarity algorithm should ideally be able to consider these different types of beans more similar to each other and less similar to other kinds of products.

In this work, we address the problem of performing the conceptual comparison of names of products and services in Portuguese by incorporating an external knowledge source into the name matching process. The knowledge source employed in this study is Onto.PT [9], the most comprehensive lexical ontology for Portuguese (which was originally built over European Portuguese). More specifically, the aim of the present paper is twofold. The first objective is to report the results of an experiment performed on a dataset that stores thousands of names of products and services in Brazilian Portuguese. The experiment compared the performance of a set of well-known algorithms based on alphabetic (character) or lexical (word) similarity against the performance of a new proposed similarity algorithm that simultaneously accounts for the alphabetic, lexical, and semantic levels of similarity. In this algorithm, the semantic similarity between two names is computed using information from Onto.PT. The second and major goal of this work is to identify the advantages and pitfalls of applying Onto.PT to name matching processes. To the best of our knowledge, this is the first time Onto.PT is employed as a tool for enhancing the effectiveness of name matching algorithms.

A preliminary short version of the paper was presented in Ontobras 2021 conference [10]. In the new version, the overall content was extended, we added new results, and improved the discussion on the pros and cons of Onto.PT. The rest of this paper is organized as follows. Section 2 gives an overview of string similarity concepts relevant to this paper. Section 3 revises the related work. Our semantic similarity algorithm based on Onto.PT is described in Section 4. In Section 5, we detail the experimental methodology and report experimental results on a dataset that contains pairs of names of products and services in Portuguese. In the same section we discuss the advantages and problems arising from the use of Onto.PT in the task of performing the conceptual comparison of names of products and services in Portuguese. Finally, we give concluding remarks and discuss some future research directions in Section 6.

2 Background

Given two names (or short strings) n_1 and n_2, a similarity function between these names can be defined as a function S that satisfies three properties [11]:

1. $S(n_1, n_2) = 1$, if n_1 is identical to n_2.
2. $S(n_1, n_2) \approx 1$, if n_1 is very similar to n_2 in some aspect.
3. $S(n_1, n_2) \approx 0$, if n_1 is very dissimilar to n_2 in some aspect.

The function S can be designed to capture different aspects (or levels) of similarity. The simplest is the character (or alphabetic) level, in which S needs only to evaluate whether n_1 and n_2 share many common characters. The second simplest level is the lexical one, where S evaluates whether n_1 and n_2 share many common tokens (words). On the other hand, the most complex is the semantics level, in which S needs to determine whether the two names carry the same meaning. These three distinct approaches are detailed in the following subsections.

2.1 Character-Based and Lexical-Based Similarity Algorithms

Over the last decades, several character-based similarity algorithms have been proposed in the literature [4, 6-8, 12-15]. In this subsection, we review the algorithms used in this paper. In the definitions throughout the text, we adopted the following notation:

- n_1 and n_2 : two names whose similarity score is to be computed.
- $|n_1|$ and $|n_2|$: the lengths of n_1 and n_2, respectively.

Levenshtein edit distance [5] is an algorithm that computes the smallest number of operations to transform n_1 into n_2. The allowed operations are character deletion, insertion, or substitution, any with cost 1. This distance can

be converted into a similarity score by using the formula defined in Equation (1) [6]. In this equation, $D_L(n_1, n_2)$ denotes the Levenshtein distance between n_1 and n_2.

$$S_{Lev}(n_1, n_2) = 1 - \frac{D_L(n_1, n_2)}{max(|n_1|, |n_2|)} \quad (1)$$

Jaro similarity [12] was developed at the Census Bureau to solve record linkage problems (the task of finding personal names or addresses in common in two datasets). Jaro similarity score is computed according to Equation (2). In this equation, c and t represent the number of character matches and transpositions, respectively. A character from n_1 and a character from n_2 match if they are identical and are located in the same position or within an allowable range defined by the formula $(max(|n_1|, |n_2|) / 2) - 1$. To compute the number of transpositions, the first assigned character on n_1 is compared to the first assigned character on n_2. Next, the second assigned character on n_1 is compared to the second assigned character on n_2, and so on. At the end of the comparison process, the number of transpositions will correspond to the number of mismatched characters divided by 2.

$$S_{Jaro}(n_1, n_2) = \frac{1}{3}\left(\frac{c}{|n_1|} + \frac{c}{|n_2|} + \frac{c-t}{c}\right) \quad (2)$$

A q-gram associated with a string s can be defined as any substring of length q found within s [13, 14]. Given a name n, it is possible to generate a vector containing all its q-grams. For instance, the *2*-gram vector for the product name n = "pepper" can be defined as: v = ['pe', 'ep', 'pp', 'pe', 'er']. Since the substring 'pe' appears twice within n, it might be more appropriate to store each q-gram along with its frequency: v = [('pe',2), ('ep',1), ('pp',1), ('er',1)]. It is possible to measure the similarity between two q-gram vectors v_1 and v_2 using the Cosine measure presented in Equation (3). In this formula, $v_1.v_2$ represents the standard dot product whereas $|v_1||v_2|$ in the denominator corresponds to the product of the vector norms.

$$S_{q-gram}(n_1, n_2) = Cosine(v_1, v_2) = \frac{v_1 \cdot v_2}{|v_1||v_2|} \quad (3)$$

Jaccard [13] is an algorithm that analyzes the similarity between two names in the lexical level. It works in two straightforward steps: first the names are splitted into words – called tokens. For instance, the name n_1 = "white bean" would be transformed into the token set Tok_1 = {"white", "bean"}. Once the token sets from the two names have been generated, Jaccard similarity can be computed according to Equation (4). It consists of the ratio of the size of the intersection of Tok_1 (set of tokens from n_1) and Tok_2 (set of tokens from n_2) to the size of their union.

$$S_{Jac}(n_1, n_2) = \frac{|Tok_1 \cap Tok_2|}{|Tok_1 \cup Tok_2|} \quad (4)$$

Table 1 shows examples of pairs of names that denote the same entity (in this case, equivalent food products) and thus should be assigned a high similarity score. In the first example, n_2 is misspelled, and it is noticeable that S_{Lev} and S_{Jaro} performed more effectively than the q-gram approach. It is also noticeable that S_{Jac} is completely ineffective in this kind of situation. On the other hand, if the words in the names are the same but in different orders, as in the second example, S_{Jac} is the most effective similarity measure and q-gram works better than both S_{Lev} and S_{Jaro}. In the third example, we have two names that are synonyms with completely different spelling. In this case, it is possible to observe that none of the measures is effective – similarity scores are 0 for $S_{2\text{-}gram,}$, $S_{3\text{-}gram,}$, and S_{Jac} and closer to 0 than to 1 for S_{Lev} and S_{Jaro}.

The character-based and token-based similarity algorithms presented in this subsection offer two advantages: they are simple and language independent. However, a considerable disadvantage lies in that they ignore the possible occurrence of semantic relationships between the names under comparison. Section 4 discusses how to extend the character and token-based methods in order to enable them to also exploit semantic issues.

Table 1: Name matching by different character-based algorithms

n_1	n_2	S_{Lev}	S_{Jaro}	$S_{2\text{-}gram}$	$S_{3\text{-}gram}$	S_{Jac}
pepper	pwpper	0,8333	0,8889	0,6761	0,5000	0,0000
peas and corn	corn and peas	0,3846	0,5564	0,8333	0,6363	1,0000
cassava	manioc	0,1429	0,4365	0,0000	0,0000	0,0000

2.2 Semantic-Based Similarity Algorithms

Two names can be considered semantically similar if they carry the same meaning or evoke the same concept [3, 16]. In order to determine the semantic similarity between names (or short texts in general), it is necessary to incorporate an external source of knowledge into the matching process. Nowadays, the two most used types of external sources are word embeddings [17-19] and lexical ontologies [3, 20, 21]. In this work, we opt for a solution based on a lexical ontology due to its inherent ability to produce interpretable results. As will be shown in Section 5, the dataset used in this study contains data from public administration, a field in which the need for comprehensibility (interpretability) tends to be particularly strong [22].

An ontology can be defined as a means to formally model the structure of a system, i.e., the relevant entities and relations that are useful to our purposes [23]. An ontology makes it possible to define a model in terms of a hierarchy of concepts (classes and their subclasses). In this paper, we are interested in a special kind of ontology known as lexical ontology or wordnet.

A lexical ontology is a structure composed of synsets and the semantic relations that connect these synsets [9, 24]. Each synset is a set of synonymous word senses associated with its part of speech and a gloss (a dictionary-style definition). Relations between synsets can include hypernymy (links more general concepts to more specifics ones), antonymy (semantic opposition), meronymy (part-whole relation), and others. Therefore, a lexical ontology can be seen as a graph where nodes are synsets and edges represent their semantic relationships. Fig. 1 presents an example of a hypothetical lexical ontology in which edges represent hypernymy relations. Fig. 2 details one of the synsets (node) of this ontology, by showing its part-of-speech and gloss. In spite of its simplicity, the lexical ontology in Fig. 1 has a similar topology to Onto.PT. The major difference is that Onto.PT represent other relationships besides hypernymy.

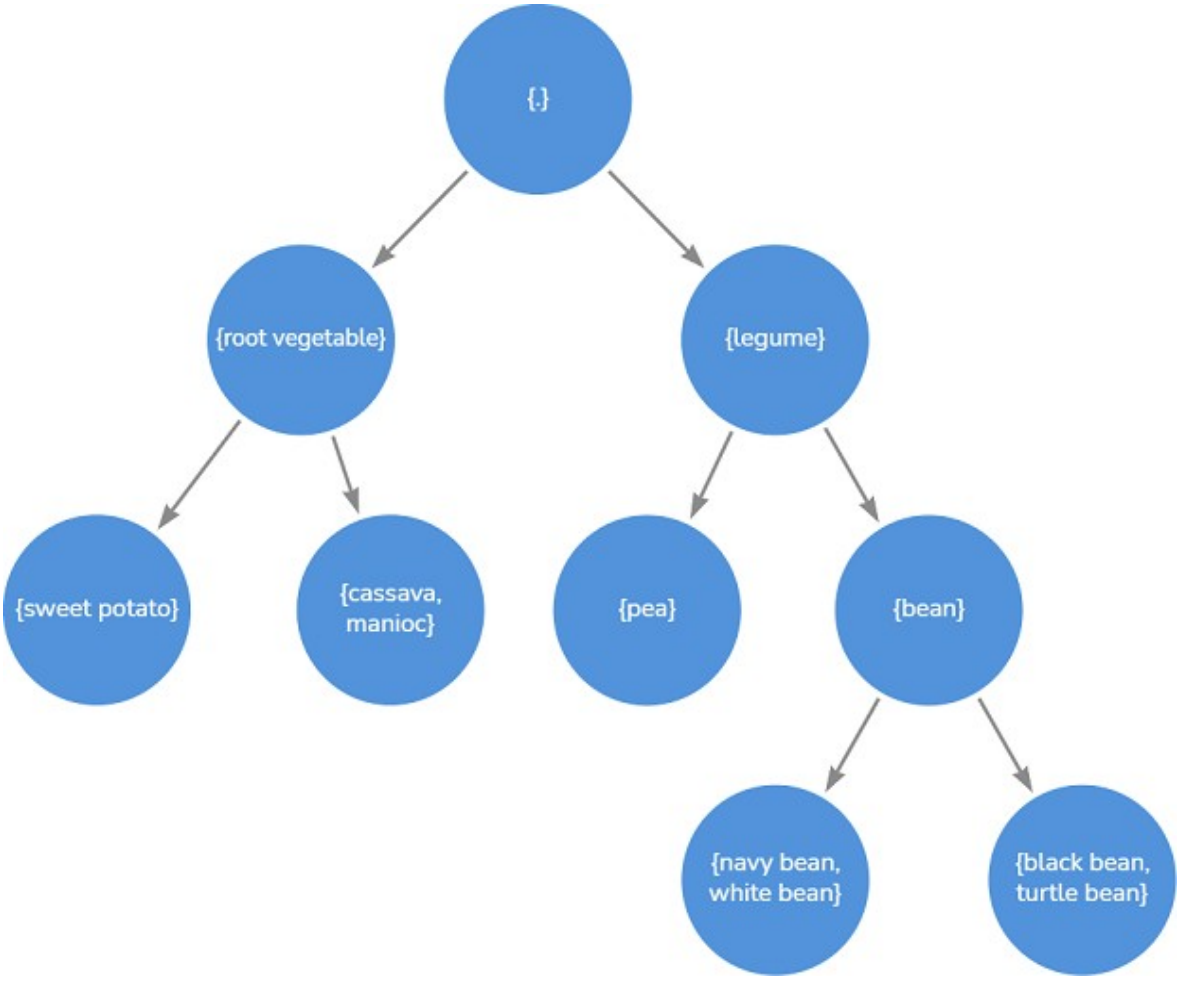

Figure 1: Lexical ontology in which edges represent hypernymy relations

{**navy bean, white bean**} (noun): *a white kidney bean native to the Americas.*

Figure 2: A synset and its properties (part-of-speech and gloss)

A few different approaches can be used for measuring the similarity between concepts in an ontology. For instance, in Equation (5) we show that it is possible to define a semantic version of the Jaccard similarity measure (denoted as S_{SemJac}) in a straightforward manner by including synonyms and hypernyms present in the lexical ontology.

$$S_{SemJac}(n_1, n_2) = \frac{|ExtTok_1 \cap Tok_2|}{max(|Tok_1|, |Tok_2|)} \quad (5)$$

Equation (5) was originally proposed in [3]. In this equation, Tok_1 and Tok_2 correspond, respectively, to the set of tokens obtained from two names n_1 and n_2. $ExtTok_1$ is the extended set derived from Tok_1 which corresponds to Tok_1 augmented with the set of tokens that are directly related to each token in Tok_1. In this work, these are the synonyms and hypernyms obtained from a lexical ontology. For example, for Tok_1 = {"cassava"} and considering the lexical ontology presented in Fig. 1, we have $ExtTok_1$ = {"cassava", "manioc", "root vegetable"} as "manioc" and "root vegetable" correspond, respectively, to the synonym and hypernym of "cassava" in this ontology. Thus, according to this approach and considering the ontology of Fig. 1, the Semantic Jaccard similarity between n_1 = "cassava" and n_2 ="manioc" can be computed as:

$S_{SemJac}(n_1, n_2)$ = |{"cassava", "manioc", "root vegetable"} ∩ {"manioc"} | / max(|{"cassava"}|,|{"manioc"}|) = 1.00.

An alternative approach is to compute semantic similarity as a function of the path length between concepts in the ontology. For instance, Wu and Palmer [25] propose an algorithm that infers the similarity between two concepts n_1 and n_2 taking into consideration the depth of these concepts in the ontology and the distance between n_1 and n_2 and their least common subsumer (the closest ancestor in common, abbreviated as *LCS*). Wu and Palmer similarity score, denoted as S_{WP}, is computed according to Equation (6). In this formula, d_1 and d_2 correspond, respectively, to the number of edges on the path from n_1 to *LCS* and n_2 to *LCS* whereas d_3 is the number of edges on the path from *LCS* to root. According to this approach and considering the ontology of Fig. 1, the similarity between "white bean" and "black bean" – two concepts that have "bean" as LCS – can be computed as: (2 × 2) / (1 + 1 + 2 × 2) = 0.67.

$$S_{WP}(n_1, n_2) = \frac{2 \times d_3}{d_1 + d_2 + 2 \times d_3} \quad (6)$$

3 Related Work

Over the last decades, character-based and token-based similarity algorithms have been often employed to resolve problems that are related to the matching of addresses, toponyms and personal names [1, 2, 6-8]. In these problems, issues like misspellings, abbreviations, word omissions, and word permutations tend to be more relevant than semantics to infer similarity [2]. A remarkable exception is the work of [26], which covers the similarity of photo titles. This work compared the performance of different character-based similarity algorithms in a dataset composed of 1,000 photo titles in English and Finnish. The number of words in the titles range from 1 to 11, having an average of 2. In the reported experiment, Levenshtein similarity obtained the best overall result, exhibiting correlation of 0.59 with human ratings.

The use of lexical ontologies for determining the semantic similarity of short texts is addressed in [3, 20, 21]. In all these studies, Princeton WordNet [24] was used as external knowledge source. Originally created for English, the WordNet model has become the major standard for representing lexical ontologies. As reported in the review paper of [16], Princeton WordNet has been by far the most adopted knowledge source in semantic-concerned studies from the fields of text mining and machine learning.

The method proposed in [20] is focused on measuring the similarity of sentences (another kind of short text) rather than names. For a pair of sentences to be compared, the first step is to derive two semantic vectors using WordNet, one for each sentence. These vectors incorporate words that belong to the same synsets of the words that actually take part in the sentences. The final similarity score between the sentences is computed comparing their

associated semantic vectors, using a formula that takes into consideration both the semantic similarity of the words (using a path length function) and word order similarity.

In [21], a semantic algorithm for comparing photo titles is presented. In the proposed approach, the first step is to transform the titles of two photos under comparison into two binary word vectors *u* and *w*. Next, WordNet is employed to determine new weight values for each word in the vectors. These weights are computed based on the shortest distance between each term in *u* and all terms in *w*. Lastly, Cosine is used to determine the similarity between the vectors.

The work of [3] addresses the conceptual similarity between trademark names. In this paper, the authors propose a novel similarity function based on Tversky's contrast theory, which defines the similarity between two entities as a function of unique and shared information about these entities. Following this principle, they propose a similarity function that considers the proportion of words and synonyms shared by two trademark names n_1 and n_2, while at the same time takes into account the proximity (in the WordNet) between each word in n_1 and each word in n_2. The results of the experiments showed that their proposed algorithm was able to exceed the performance of character-level algorithms by about 20% in terms of match effectiveness.

It is important to remark that all the techniques proposed in [3, 20, 21] employ Princeton WordNet as external source of knowledge and address the conceptual similarity of either short sentences or trademarks in English. In the next section, we propose an algorithm to compute the conceptual similarity between names in Portuguese – more specifically product and service names in the Portuguese language. This approach rely on the use Onto.PT to determine the similarity between two names in the semantics level.

4 The Proposed Algorithm

We believe that evaluating different levels of similarity at once can lead to a measure that might be more suitable for comparing names of products and services. Based on this assumption, we proposed the hybrid similarity function shown in Equation (7). This function simultaneously takes into account the analysis of character-based similarity (first term of the function), lexical similarity (second term) and semantic similarity (last term).

$$S_H(n_1,n_2)=\frac{1}{3}\left(S_{char}(n_1,n_2)+S_{Jac}(n_1,n_2)+S_{SemJac}(n_1,n_2)\right) \quad (7)$$

In Equation (7):

- The first term, $S_{char}(n_1, n_2)$, can be any chosen character-based similarity function, such as Levenshtein, Jaro, Cosine, a combination of these functions, or any other.

- The second term, $S_{Jac}(n_1, n_2)$ is the Jaccard similarity between the two names. This score reflects similarity at the lexical level [16]. However, as we make use of Onto.PT, we are able to produce tokens that correspond to compound nouns, such as "sweet potato" (which can be regarded as a relevant advantage of our proposal).

- The third term, $S_{SemJac}(n_1, n_2)$ is the Semantic Jaccard similarity between the two names, introduced in Subsection 2.2. Thus, the score obtained in the third part of the equation reflects similarity at the semantic level. Its calculation depends on a lexical ontology.

Next, we give an example on how to compute the similarity between the product names n_1 = "sweet potato and manioc" and n_2 = "cassava + sweet potato" using the proposed function and employing the lexical ontology presented in Fig. 1 as external knowledge source. Also consider that $S_{3\text{-}gram}$ was chosen to assess character similarity (first part of Equation 7). In this example, we have Tok_1 = {"manioc", "sweet potato"} and Tok_2 = {"cassava", "sweet potato"}. Note that the tokens "and" (stop word) and "+" (symbol) are discarded. The set of words directly related to the words in Tok_1 is defined as $ExtTok_1$ = {"manioc", "sweet potato", "cassava", "root vegetable"}. Hence, the value of $S_H(n_1, n_2)$ is computed as follows:

- $S_{3\text{-}gram}(n_1, n_2) = 0.6048$
- $S_{Jac}(n_1, n_2) = 1 / 3 = 0.3333$
- $S_{SemJac}(n_1, n_2) = = 2 / 2 = 1.000$

The final similarity score is obtained taking the average of the above scores:

- $S_H(n_1, n_2) = 1/3 \times (0.6048 + 0.3333 + 1.000) = 0.6460$

It is important to mention that our hybrid function S_H is an adaptation of the similarity function originally proposed in [3]. Nonetheless, there are two important differences. First, we proposed a similarity function that computes a score based on the combination of character, lexical, and semantic similarity. On the other hand, the method proposed in [3] disregards character-based closeness, an aspect that might have some relevance for the task of matching names of products and services. The second difference is that the lexical ontology employed in this study is Onto.PT [9] instead of Princeton WordNet as our goal is to evaluate names in Portuguese rather than in English. It is also important to state that although we are aware that there are other public lexical ontologies currently available for Portuguese [27-29], we decide to choose Onto.PT due to the fact it is the largest Portuguese wordnet [27, 30]. The latest version, Onto.PT 0.6, is available as a standard RDF/OWL file in [31] and includes 67,873 nouns and 20,760 adjectives. This feature of Onto.PT was important for our choice, as names of products and services are mostly composed by nouns and adjectives (conversely, they rarely contain verbs).

5 Experiments

5.1 Dataset

The dataset studied in this work consists of 4,956 pairs of matched names in the Portuguese language. All names correspond to descriptions of products and services that can be acquired by families that live in the metropolitan areas of the major Brazilian cities. For each pair (p, s) in the dataset, p represents a name used in the questionnaire of the Consumer Expenditure Survey (POF-IBGE) [32] whilst s corresponds to a name used by the National System of Consumer Price Indexes (SNIPC-IBGE) [33]. The dataset was built manually by a team of researchers and technicians at the Brazilian Institute of Geography and Statistics (IBGE) and can be obtained at [34].

An excerpt from the dataset is shown in Table 2. It is important to state that in this dataset, the relationship between SNIPC names and POF names is 1 to N, which means that one name from SNIPC can be matched with one or more names from POF. Conversely, each POF name matches one and only one SNIPC name. For example, in Table 2 it is possible to observe that the POF names “arroz polido” (“polished rice”) and “arroz com casca” (“paddy rice”) are both matched with the SNIPC name “arroz” (“rice”). Table 3 summarizes the characteristics of the dataset. To conduct the experiments reported in this paper, the only preprocessing tasks we carried out in the dataset were the following: converting names to lowercase, removing symbols and punctuations and correcting POF names that were not accented in the original dataset.

Table 2: An excerpt from the studied dataset

POF name	***SNIPC name***
ARROZ POLIDO	Arroz
ARROZ COM CASCA	Arroz
COCO BURITI	Buriti (coco)
GELADEIRA	Refrigerador
LIVRO ESCOLAR DE PRIMEIRO OU SEGUNDO GRAU IMPRESSO	Livro didático
CAVAQUINHO	Instrumento musical
AUTO ESCOLA	Autoescola
ACADEMIA	Atividades físicas
CONTA DE TELEFONIA CELULAR (INTERNET)	Plano de telefonia móvel

Table 3: Summary of the POF x SNIPC dataset

Source	***Distinct Names***	***Length of the names (in number of words)***
POF	4,956	Min=1, Max=14, Avg=3.51 (88% composed of up to 5 words)
SNPIC	2,570	Min=1, Max=11, Avg=2.59 (95% composed of up to 5 words)

5.2 Evaluation Measures

To compare the algorithms presented in Sections 2 and 4, we decided to treat the name matching problem as an information retrieval (IR) problem [35, 36] where the goal was to find the name *s* from SNIPC that best matches a POF name *p*. Although there is only and exactly one correct SNIPC match for each POF name, the evaluated algorithms often return two or more names as the best match (i.e., they may return different names with the same highest similarity score).

Due to this fact, we decided to assess the performance of the similarity algorithms using three popular IR evaluation metrics capable of taking into consideration results that are partially correct. These are Precision (Pre), Recall (Rec), and F1-Score (F1) [35, 36], respectively shown in Equations (8), (9), and (10). In these formulas, the set with the single relevant SNIPC name for a POF name is denoted as *Relevant* whilst the set of SNIPC names that were identified as the most similar according to the similarity algorithm is denoted as *Retrieved*. In our experiments, the IR task was performed separately for each POF name present in the dataset and the results were averaged.

$$Pre = \frac{|Relevant| \cap |Retrieved|}{|Retrieved|} \tag{8}$$

$$Rec = \frac{|Relevant| \cap |Retrieved|}{|Relevant|} \tag{9}$$

$$F1 = 2 \times \frac{Rec \times Pre}{Rec + Pre} \tag{10}$$

5.3 Results

The experiments conducted in this work aim to investigate three research questions:

1. What is most appropriate character-based algorithm for matching names of products and services in Portuguese?
2. Does the hybrid function that uses Onto.PT lead to an increase in effectiveness?
3. What are the pros and cons of using Onto.PT for performing conceptual name matching?

5.3.1 RQ1: What is the most appropriate character-based algorithm for matching names of products and services in Portuguese?

The first experiment compared the performance of the character-based and token-based similarity algorithms discussed in Subsection 2.1. We used the implementations available at the strsimpy package [37], an open-source Python library that implements different string similarity and distance algorithms. Levenshtein, Cosine *q*-gram, and Jaccard similarity functions are implemented exactly in the same way presented in Subsection 2.1. However, Jaro implementation in strsimpy is a variation called Jaro-Winkler [15], which slightly modified the original Jaro function to provide higher weight to prefix matches.

Results are shown in Table 4. The first column indicates the name of the algorithm whilst columns 2, 3, and 4, respectively, show the obtained values for Precision, Recall, and F1-Score (computed according to the approach described in the previous subsection). Best scores are highlighted in bold.

The results presented in Table 4 show that the Cosine *q*-gram performance is superior to Jaro and Levenshtein with respect to the three evaluation metrics and superior to Jaccard in two out of the three metrics (Precision and F1-Score). Cosine 3-gram achieved the overall best results (Precision of 61.53% and F1-Score of 62.02%). Although Jaccard obtained the best Recall score, it is important to state that this was due to the fact that it tends to produce very large *Retrieved* sets (i.e., several different names with the same highest similarity score). On the other hand, Jaccard performance is the worst in terms of Precision and F1, two measures that penalize false positives.

Table 4: Performance of the character-based and token-based similarity algorithms

algorithm	***Pre***	***Rec***	***F1***
Levenshtein (S_{Lev})	0.4480	0.4702	0.4547
Jaro (S_{Jaro})	0.5373	0.5556	0.5432
Cosine *2*-gram ($S_{2\text{-}gram}$)	0.6061	0.6204	0.6106
Cosine *3*-gram ($S_{3\text{-}gram}$)	**0.6153**	0.6306	**0.6202**
Jaccard (S_{Jac})	0.2963	**0.6937**	0.4153

The superior results of the *q*-gram approach can be explained by two characteristics of the studied dataset: (i) the presence of name pairs that contain the same words but differently ordered (e.g.: "coco buriti" x "buriti coco"); (ii) the existence of many pairs of the type {hyponym, hypernym} in which the hypernym is a short string contained within a longer hyponym string (e.g.: "macarrão sem ovos" x "macarrão"). In both situations, similarity algorithms based on character edit distance like Levenshtein and Jaro tend to perform rather poorly. On the other hand, there are very few cases of misspelled names and abbreviations, which are the most suitable use cases for Jaro and Levenshtein.

5.3.2 RQ2: Does the hybrid function with Onto.PT lead to an increase in effectiveness?

In the second experiment, we investigated whether the use of Onto.PT increases the effectiveness of the name matching process. We compared the hybrid similarity algorithm S_H proposed in Section 4 – which simultaneously accounts for character, lexical, and semantic similarities (the last two with the use of Onto.PT) – against Cosine *3*-gram, the best performing character-based algorithm according to the previous experiment. Since the S_H function does not impose any specific function for the evaluation of similarity at the character level (the first part of the function) we decide to use a combination of Levenshtein, Jaro, and Cosine *3*-gram: $max(S_L, S_J, S_{3\text{-}gram})$.

Table 5 presents the results, which are structured in the following manner:

- In the first line, we reproduce the results obtained by Cosine *3*-gram (the same as previously shown in Table 4).
- The second line presents the results obtained by $max(S_L, S_J, S_{3\text{-}gram})$ used alone. Hence, these results correspond to using only the first term of S_H.
- The third line shows the results obtained by using only the first two terms of S_H. Thus, a measure that combines the evaluation of character and lexical similarity.
- Finally, in the last line we present the results obtained by the complete S_H function, as shown in Equation (7).

Results shown in Table 5 indicate that the complete hybrid similarity achieved the best performance in the three evaluation metrics (with Precision, Recall and F1 superior to 66%). Thus, in the studied dataset, the use of Onto.PT in tandem with the proposed S_H function provided a gain of 5.0% in terms of F1-Score in comparison with Cosine *3*-gram, the best character-based algorithm. Results also show that the use of the complete function yield superior results compared to using only the first two terms (character-based + lexical similarity). In this case, the gain in terms of F1-Score was superior to 4.0%.

In order to study the difference in the behavior of the algorithms S_H and Cosine *3*-gram we decide to analyze the cases in which the two algorithms returned different results (i.e., situations where the *Retrieved* sets generated by S_H and Cosine *3*-gram are composed of different elements). This occurred in 1,634 of the 4,960 comparisons (32.94%). In Table 6, we present a contingency table containing a summary of these divergent results. In this table, "S_H correct" and "S_H incorrect" show, respectively, the number of times that the correct SNIPC match for a POF name was either present or not present in the *Retrieved* set returned by the S_H algorithm. Analogously, the columns "$S_{3\text{-}gram}$ correct" and "$S_{3\text{-}gram}$ incorrect" show the same information for $S_{3\text{-}gram}$. According to the values in this table, the correlation strength between the two algorithms is low (-0.1184), which indicates that S_H and Cosine *3*-gram actually behave differently in this subset of the results. Moreover, it is noticeable that S_H achieved 484 correct matches whereas Cosine *3*-gram obtained only 249 correct matches. Therefore, the experiments suggest that Onto.PT was capable of enhancing the effectiveness of the conceptual comparison of names.

Table 5: Performance of the S_H algorithm

algorithm	***Pre***	***Rec***	***F1***
Cosine *3*-gram ($S_{3\text{-}gram}$)	0.6153	0.6306	0.6202
S_H - only the first term	0.5384	0.5567	0.5443
S_H - average of the first two terms	0.6201	0.6357	0.6278
S_H - complete (with Onto.PT)	**0.6674**	**0.6756**	**0.6700**

Table 6: Contingency table: cases where S_H and Cosine *3*-gram returned different results

	$S_{3\text{-}gram}$ correct	$S_{3\text{-}gram}$ incorrect	**TOTAL**
S_H correct	42	442	484
S_H incorrect	207	943	1,150
TOTAL	249	1,385	1,634

5.3.3 RQ3: What are the advantages and disadvantages of using Onto.PT for conceptual name matching?

To end this section, we discuss the pros and cons of using Onto.PT as the chosen external source of knowledge to perform conceptual name matching. Onto.PT has two appealing characteristics. First, it is freely available as a standard RDF/OWL file that can be easily integrated to any system. Second and more importantly, Onto.PT covers a comprehensive number of lexical items. We found that 75.87% and 77.11% of the individual words that appear in SNIPC and POF names, respectively, are also present in Onto.PT. This conforms to the study of [27] which observed that Onto.PT includes about three times more lexical items than the second largest wordnet-like ontology for Portuguese. Interestingly, we identified that a large part of the absent words consist of number and gender inflections of other words that do actually belong to the ontology. These cases could thus be easily treated.

Nonetheless, differently from the WordNet and even to some of the other lexical ontologies for Portuguese [27-29], Onto.PT was not hand-crafted by experts. Instead, it was built by a fully automated process based on the exploitation of European Portuguese dictionaries and thesauri. Consequently, it is prone to limitations and errors, as previously pointed out in [9, 27, 30]. We consider that two of OntoPT's limitations might have been responsible for negative impacts on the effectiveness of the hybrid similarity function. First, we identified that although Onto.PT covers most of the individual words in the database, the same is not true for the *open compounds*, i.e., names formed by two separated words. This is a relevant disadvantage in the studied problem, since product names are often composed of two nouns or a noun and an adjective. There exist about 350 compound nouns in the dataset (which may appear in the middle of several other longer names), however less than 12% can be found in Onto.PT. For instance, product names like “feijão azuki” (“adzuki bean”), “milho-verde” (“green corn”) and “arroz branco” (“white rice”) are absent from Onto.PT, although they do exist as lexical items in Princeton WordNet.

The second limitation is stated by the authors of Onto.PT themselves in [9]: (i) the fact that most paths from the more specific synsets to the root of the ontology are not more than three edges long; (ii) the existence of cycles – when node A is hypernym of node B and, at the same time, node B is hypernym of node A. We found a total of 1,568 cycles like the example shown in Fig. 3. These aspects of the graph topology have hindered us from evaluating path-based similarity algorithms, such as the Wu and Palmer approach presented in Subsection 2.2. This is a relevant issue to the name matching application, since a number of methods for short text comparison proposed in the literature rely on path-based similarity functions [3, 20, 21]. Moreover, path-based functions are also essential to remove ambiguities that occur when a word belongs to more than one synset [38].

Nevertheless, it is important to state that the limitations of Onto.PT discussed in this section are relevant for name matching processes, but they might not be as important to other applications. Moreover, it is also worth reinforcing that in spite of these limitations, our experiments indicated that the proposed hybrid similarity function that works with Onto.PT obtained results superior to character-based algorithms according to the F1 accuracy score.

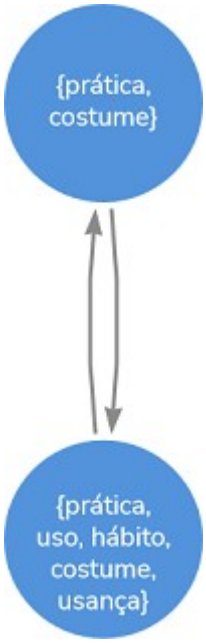

Figure 3: An example of cycle in Onto.PT

6 Conclusions

This paper contributed to the conceptual name matching problem by evaluating the effectiveness of using Onto.PT (the largest public Portuguese wordnet) to allow the assessment of semantic similarity between names of products and services in the Portuguese language. We proposed a hybrid function that employs Onto.PT as external knowledge source and simultaneously takes into account the analysis of three similarity aspects: character, lexical, and semantic. Our experiments on a dataset that stores product and service names in Portuguese show that the proposed function is more effective than other well-know algorithms for name matching. To the best of our knowledge, this is the first paper to propose the use of Onto.PT for the problem of conceptual comparison of names.

Moreover, we also discussed the main advantages and disadvantages of using Onto.PT as tool for conceptual name matching. We identified that this lexical ontology suffers from two drawbacks: (i) differently from Princeton WordNet, Onto.PT contains few of the product names that are compound nouns; (ii) there are cycles in the ontology and most of the more specific synsets (leaf nodes in the graph topology) have a depth of three edges or less, which prevents the effective use of path-based similarity functions. Nonetheless, it is important to consider that while these disadvantages are clearly relevant to the conceptual name matching problem, they may not be as important to other problems in the field of natural language processing. Despite its limitations, we strongly believe that Onto.PT still represents a valuable tool since it includes a comprehensive number of lexical items, apart from being simple to use and free.

As future research, we first plan to construct a domain ontology of products and services to be used by the hybrid similarity function. We consider that the construction of this ontology will be facilitated if we inherit several of the lexical items that are already included in Onto.PT. Also as future work, we intend to evaluate other character and token-based measures that have shown good performance in recent work on similarity of short texts [4, 39]. Finally, following the approach of [40], we intend to modify our hybrid function S_H to also incorporate the evaluation of the syntactic similarity between names. This corresponds to an important similarity level of linguistic knowledge [16] that has not yet been addressed in this work.

References

[1] A. E. Monge and C. P. Elkan, "The field matching problem: Algorithms and applications," in *Proc. of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, Portland, USA, Aug. 1996, pp. 267–270.

[2] K. L. A. Branting, "A Comparative evaluation of name matching algorithms," in *Proc. of the 9th International Conference on Artificial Intelligence and Law (ICAIL'03)*, Edinburgh, Scotland, UK, Jun. 2003, pp. 224–232. [Online]. Available: https://doi.org/10.1145/1047788.1047837

[3] F. M. Anuar, R. Setchi, and Y-K Lai, "Semantic Retrieval of Trademarks based on Conceptual Similarity," *IEEE Trans. on System, Man, and Cybernetics: Systems,* vol 46, no. 2, pp. 220-233, Feb. 2016. [Online]. Available: https://doi.org/10.1109/TSMC.2015.2421878

[4] N. Gali, R. Mariescu-Istodor, D. Hostettler, and P. Fränti, "Framework for Syntactic String Similarity Measures," *Expert Systems with Applications,* vol 129, pp. 169-185, Sep. 2019. [Online]. Available: https://doi.org/10.1016/j.eswa.2019.03.048

[5] I. V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Cybernetics and Control Theory,* vol 10, no. 8, pp. 707-710, 1966.

[6] P. Christen, "A comparison of personal name matching: Techniques and practical issues," in *Proc. of the IEEE 6th Data Mining Workshop (ICDMW'06)*, Hong Kong, China, Dec. 2006, pp. 290–294. [Online]. Available: https://doi.org/10.1109/ICDMW.2006.2

[7] C. A. Davis Jr. and E. Salles, "Approximate string matching for geographic names and personal names," in *Proc. of the IX Brazilian Symposium on Geoinformatics*, Campos do Jordão, Brazil, Nov. 2007, pp. 49–60.

[8] G. Recchia and M. Louwerse, "A comparison of string similarity measures for toponym matching," in *Proc. of the 1st ACM SIGSPATIAL International Workshop on Computational Models of Place*, Orlando, USA, Nov. 2013, pp. 54–61. [Online]. Available: https://doi.org/10.1145/2534848.2534850

[9] H. G. Oliveira and P. Gomes, "ECO and Onto.PT: A Flexible Approach for Creating a Portuguese Wordnet Automatically," *Language Resources and Evaluation,* vol 48, no. 2, pp. 373-393, Jun. 2014. [Online]. Available: https://doi.org/10.1007/s10579-013-9249-9

[10] E. C. Gonçalves and T. P. Meirelles, "Improving the effectiveness of name matching algorithms with a portuguese wordnet," in *Proc of the XIV Seminar on Ontology Research in Brazil (ONTOBRAS 2021)*, Online, Brazil, Nov. 2021, pp. 244–251.

[11] D. Lin, "An information-theoretic definition of similarity," in *Proc. of the 15th International Conference on Machine Learning (ICML'98)*, San Francisco, USA, Jul. 1998, pp. 296–304.

[12] M. A. Jaro, "Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida," *Journal of the American Statistical Association,* vol 84, no. 406, pp. 414-420, 1989. [Online]. Available: https://doi.org/10.1080/01621459.1989.10478785

[13] J. Leskovec, A. Rajaraman, and J. Ullman, *Mining of Massive Datasets*. 3rd ed., New York: Cambridge University Press, 2020.

[14] L. Gravano, et al., "Using q-grams in a DBMS for Approximate String Processing," *IEEE Data Engineering Bulletin,* vol 24, no. 4, pp. 28-34, Dec. 2001.

[15] W. E. Winkler, "Advanced methods for record linkage", in *Proc. of the Survey Research Methods Section, American Statistical Association*, p. 467-472, 1994.

[16] R. Sinoara, J. Antunes, and S. O. Rezende, "Text Mining and Semantics: A Systematic Mapping Study," *Journal of the Brazilian Computer Society,* vol 23, no. 9, pp. 1-20, Jun. 2017. [Online]. Available: https://doi.org/10.1186/s13173-017-0058-7

[17] D. S. M. Gomes, F. C. Cordeiro, B. S. Consoli, N. L. Santos, V. P. Moreira, R. Vieira, S. Moraes, A. G. Evsukoff, "Portuguese Word Embeddings for the Oil and Gas Industry: Development and Evaluation," *Computers in Industry,* vol 1024, pp. 1-14, Jan. 2021. [Online]. Available: https://doi.org/10.1016/j.compind.2020.103347

[18] T. P. Meirelles, E. C. Gonçalves, and D. T. Gomes, "Pareamento de Nomes de Produtos e Serviços Utilizando Medidas de Similaridade Textual nos Níveis Alfabético, Léxico e Semântico," *Cadernos do IME – Série Informática,* vol 129, pp. 104-117, Dec. 2021. [Online]. Available: http://dx.doi.org/10.12957/cadinf

[19] A. S. Romualdo, L. Real, and H. M. Caseli, "Measuring brazilian portuguese product titles similarity using embeddings," in *Proc. of the XIII Symposium in Information and Human Language Technology (STIL'21)*, Porto Alegre, Brazil, Nov. 2021, pp. 121–132. [Online]. Available: https://doi.org/10.5753/stil.2021.17791

[20] Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea, and K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," *IEEE Transactions on Knowledge and Data Engineering,* vol 18, no. 8, pp. 1138-1150, Jun. 2006. [Online]. Available: https://doi.org/10.1109/TKDE.2006.130

[21] D. Croft, S. Coupland, J. Shell, and S. C. Brown, "A fast and efficient semantic short text measure," in *Proc. of the 13rd UK Workshop on Computational Intelligence (UKCI'13)*, Guildford, UK, Sep. 2013, pp. 221–227. [Online]. Available: https://doi.org/10.1109/UKCI.2013.6651309

[22] K. R. Varshney, "Data Science of the people, for the people, by the people: A viewpoint on an emerging dichotomy," in *Proc. of the Bloomberg Data for Good Exchange 2015 (D4GX'2015)*, New York, USA, Sep. 2015, pp. 1–6.

[23] N. Guarino, D. Oberle, and S., Staab. "What Is an Ontology?", *Handbook on Ontologies*, International Handbooks on Information Systems, pp. 1-17, May. 2009. [Online]. Available: https://doi.org/10.1007/978-3-540-92673-3_0

[24] C. Fellbaum, *WordNet: an electronic lexical database*. 3rd ed., New York: Cambridge University Press, 1998.

[25] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proc. of the 32nd Annual Meeting of the Association for Computational Linguistics*, New Mexico, USA, Jun. 1994, pp. 133–138. [Online]. Available: https://doi.org/10.3115/981732.981751

[26] N. Gali, R. Mariescu-Istodor, and P. Fränti, "Similarity measures for title matching," in *Proc. of the 23rd International Conference on Pattern Recognition (ICPR'16)*, Cancún, Mexico, Dec. 2016, pp. 1549–1554. [Online]. Available: https://doi.org/10.1109/ICPR.2016.7899857

[27] V. de Paiva, L. Real, H. G. Oliveira, A. Rademaker, C. Freitas, and A. Simões, "An overview of portuguese wordnets", in *Proc. of the 8th Global WordNet Conference (GWC'16)*, Bucharest, Romania, Jan. 2016, pp. 74-81.

[28] A. Branco, S. Grilo, M. Bolrinha, C. Saedi, R. Branco, J. Silva, A. Querido, R. de Carvalho, R. Gaudio, M. Avelãs, and C. Pinto, "The MWN.PT wordnet for portuguese: projection, validation, cross-lingual alignment and distribution," in *Proc. of the 12th Conference on Language Resources and Evaluation (LREC'20)*, Marselle, France, Dec. 2020, pp. 4859-4866.

[29] V. de Paiva, A. Rademaker, and G. Melo "OpenWordnet-pt: An open brazilian wordnet for reasoning", in *Proc. of the Cooling 2012: Demonstration Papers*, Mumbai, India, Dec. 2012, pp. 353-360.

[30] H. G. Oliveira, "CONTO.PT: Groundwork for the automatic creation of a fuzzy portuguese wordnet," in *Proc. of the 12th International Conference on the Computational Processing of Portuguese (PROPOR'16)*, Tomar, Portugal, Jul. 2016, pp. 283-295. [Online]. Available: https://doi.org/10.1007/978-3-319-41552-9_29

[31] Onto.PT. *Ontologia Lexical para o Português.* 2022. http://ontopt.dei.uc.pt/

[32] IBGE. *POF – Consumer Expenditure Survey.* 2022. https://www.ibge.gov.br/en/statistics/social/health/25610-pof-2017-2018-pof-en.html?=&t=o-que-e

[33] IBGE. *IPCA - Extended National Consumer Price Index.* 2022. https://www.ibge.gov.br/en/statistics/economic/prices-and-costs/17129-extended-national-consumer-price-index.html?=&t=o-que-e

[34] IBGE. *IPCA - Downloads.* 2022. https://www.ibge.gov.br/estatisticas/economicas/precos-e-custos/9256-indice-nacional-de-precos-ao-consumidor-amplo.html?=&t=downloads

[35] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. New York: Cambridge University Press, 2008.

[36] J. Baeza-Yates and B Ribeiro-Neto, *Modern Information Retrieval: The Concepts and Technology Behind Search*. 2nd edition, New York: Addison-Wesley Professional, 2011.

[37] strsimpy 0.2.1. *python-string-similarity.* 2022. https://pypi.org/project/strsimpy/

[38] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd edition (draft), 2021. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/

[39] N. S. Hartmann, "Solo Queue at ASSIN: Combinando Abordagens Tradicionais e Emergentes," *Linguamática,* vol 8, no. 2, pp. 59-64, Dec. 2016.

[40] J.Yang, Y. Li, G. Gao, and Y. Zhang. "Measuring the Short Text Similarity Based on Semantic and Syntactic Information," *Future Generation Computer Systems*, vol 114, pp. 169–180, Jan 2021. [Online]. Available: https://doi.org/10.1016/j.future.2020.07.043