FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Automatic conversion of cooking recipes to grocery lists

Marcelo Diocleciano Rodrigues Ferreira

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Liliana Ferreira

Co-Supervisor: David Ribeiro

July 24, 2019

Automatic conversion of cooking recipes to grocery lists

Marcelo Diocleciano Rodrigues Ferreira

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Doctor Carla Alexandra Teixeira Lopes
External Examiner: Doctor Mário Jorge Ferreira Rodrigues
Supervisor: Doctor Liliana da Silva Ferreira

July 24, 2019

Abstract

Usually, supermarkets and online stores have a large offer of products. This large offer gives the consumer a wide range of choice and at the same time attract different consumer segments. On the other hand, it can make harder the task of the consumer in finding all the products he wants to buy. Cooking recipes are a source of information that can be used to ease the purchase of products. The number of shared cooking recipes through the World Wide Web has been increasing. They are not only shared by professional cooking chefs, but also by "home" chefs. These cooking recipes are often written in a text-free form, which raises problems to the computer process the information. The aim of this thesis is to develop a system that transforms a recipe written in Portuguese into a shopping list. With the information extracted from the recipe match the ingredients and quantities with a real-world database and then return that information to the consumer.

This problem has been divided into two steps: the first consists in extracting relevant information from the recipe, the second is to match the necessary ingredients to prepare the recipe with products from a real retailer database. Concerning the extraction of the ingredients from the unstructured text, the goal is to extract the name of the ingredient and the respective quantity. This is done by applying a Conditional Random Fields model. The next step on the system is to match the ingredients obtained with products presented in a database. There are some challenges in this phase. The match between the extracted information from the recipe and the database, sometimes may not produce the expected results. That's because of some noise, the use of synonyms or even misspell words. To pass through this challenge, it is necessary to use external resources, such as ontologies, thesaurus or taxonomies.

To evaluate the developed components, in addition to a previously available dataset, two more were annotated. The information extraction got an F1 of 0.98 in one of the experiments made. The similarity match got an average F1 of 0.354 and an average precision of 0.364 in one of the experiments with a path based measure.

As previously mentioned, it is believed that this system will ease the customers work in creating a shopping list from a recipe.

Resumo

Normalmente, supermercados e lojas online têm uma grande oferta de produtos. Esta grande oferta dá ao consumidor uma ampla gama de opções e, ao mesmo tempo, atrai diferentes segmentos de consumidores. Por outro lado, pode dificultar a tarefa do consumidor em encontrar todos os produtos que deseja comprar. As receitas são uma fonte de informação que pode ser usada para ajudar na compra de produtos. O número de receitas de culinária compartilhadas através da *World Wide Web* tem aumentado. As mesmas são partilhadas não só por cozinheiros profissionais, mas também por cozinheiros convencionais. Estas receitas de culinária são muitas vezes escritas em texto livre, o que levanta alguns problemas no processamento da informação pelo computador. O objetivo desta tese é desenvolver um sistema capaz de transformar uma receita, escrita em português, numa lista de compras. Com a informação extraída da receita são combinados ingredientes com produtos presentes numa base de dados real e, em seguida, essa informação é retornada ao consumidor.

Este problema foi dividido em duas etapas: a primeira consiste em extrair informações relevantes da receita, a segunda é combinar os ingredientes necessários para preparar a receita com produtos de uma base de dados de um retalhista. No que diz respeito à extração dos ingredientes do texto não estruturado, o objetivo é extrair o nome do ingrediente e a respetiva quantidade. Isso é feito aplicando um modelo que usa *Conditional Random Fields*. O próximo passo no sistema é combinar os ingredientes obtidos com produtos apresentados numa base de dados. Existem alguns desafios nesta fase. Fazer corresponder o nome do ingrediente extraído da receita com produtos de uma base de dados, às vezes, pode não produzir os resultados esperados. Isso deve-se a algum ruído, o uso de sinónimos ou mesmo palavras com erros ortográficos. Para ultrapassar este obstáculo, é necessário utilizar recursos externos, como ontologias, tesauros ou taxonomias.

Para avaliar os componentes desenvolvidos, para além de um *dataset* previamente disponível, foram criados mais dois. A extração de informações obteve um *F1* de 0.98 em uma das experiências realizadas. Por sua vez, a correspondência entre ingredientes e produtos obteve um *F1* de 0.354 e uma *average precision* de 0.364 em uma das abordagens, com uma medida baseada no caminho.

Como mencionado anteriormente, acredita-se que este sistema irá facilitar o trabalho das pessoas na criação de uma lista de compras, automaticamente, a partir de uma receita.

Acknowledgements

First of all, I would like to thank my supervisors at Fraunhofer Portugal AICOS, David Ribeiro and Liliana Ferreira and also to Diego Silva and Nuno Silva for their support and help during the development of the project.

I would also like to thank my family for all the support, strength and encouragement they gave me during this long and arduous journey. Their help was paramount for me to achieve what I have achieved.

Finally, I would like to thank my girlfriend Cláudia, for the love, the support, the motivation, everything. Without her, I would not have made it so far!

Marcelo Diocleciano Rodrigues Ferreira

"The computer was born to solve problems that did not exist before."

Bill Gates

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

NLP	Natural Language Processing
CRF	Conditional Random Field
ML	Machine Learning
WWW	World Wide Web
IE	Information Extraction
NER	Named Entity Recognition
IC	Information Content
LCS	Least Common Subsumer
MICA	Most Informative Common Ancestor
OWL	Web Ontology Language
BT	Broader Term
NT	Narrower Term
RDF	Resource Description Framework
TFIDF	Term Frequency - Inverse Document Frequency
CBOW	Continuous Bag-Of-Words
SKOS	Simple Knowledge Organization System
AP	Average Precision

Chapter 1

Introduction

An essential task of every day is cooking. On the day-to-day, people sometimes do not have time or ideas to do so. Moreover, typical supermarkets and online stores have a large offer of products. This large offer gives the consumer a wide range of choice which can attract different consumer segments. On the other hand, this wide range of choice can make harder the task of the consumer in finding all the products he wants to buy. Therefore it is necessary to develop methods to assist in the creation of grocery lists. Cooking recipes are a source of information that can ease the purchase of products.

For decades the recipes were only shared hand to hand or by culinary books. On these days, that still happens but the share of cooking recipes through the World Wide Web(WWW) is increasing [Soc]. Not only professional cooking chefs have been sharing their knowledge, but also "home" cooking chefs have been doing the same.

The objective of this project is to develop a system, that from an online cooking recipe creates a grocery list of products, that are necessary to buy, in order to make the corresponding recipe, with alternatives. The main focus of this theses will be the Portuguese language, which by itself composes a challenge due to the lack of resources and work done on the area.

Ingredient lines on recipes do not have a structure they are in free text. Since there is not a predefined structure like a zipcode or a programming language have. Because of that extract, the ingredients from it is not straightforward. So it is necessary to use Natural Language Processing (NLP) to extract those ingredients and the respective quantities from every ingredient line on an online cooking recipe.

Match the ingredients presented on an ingredient line with products from a real database, sometimes is not straightforward. That's because the ingredient can be a meronym, a hyperonym, a hyponym, synonyms or can even have spelling errors. To tackle this problem is necessary to make use of external resources, like ontologies, thesaurus or taxonomies. With the information presented on these resources can be applied semantic similarity measures such as path-based,

feature based, information content-based and hybrid. These measures will access the level of similarity between an ingredient and a product so that it can be made a filter to the products.

These two main problems fall on the nutrition area that is one of the areas worked on at Fraunhofer Portugal AICOS where this thesis was developed.

1.1 Dissertation Structure

Apart from the introduction, this thesis contains more four chapters.

In chapter 2, it is described the state of the art. Since there are two main problems, extract ingredients from an online cooking recipe and match the ingredients with products, the state of the art is also divided into two parts. Furthermore, it is made a survey of the technologies and frameworks, used when working with machine learning (ML) and natural language processing.

In chapter 3, it is explained in detail all the steps taken to implement the proposed solution. Reaching points such as architecture, technologies and datasets used.

Chapter 4 focuses on the results obtained with the system. Before showing the results it is made a description of the datasets used for a better understanding of the results. It is presented a description of the metrics that were used to evaluate. In the end, it is made a discussion of the results pointing out what had a negative impact on the results.

Finally, in chapter 5, it is summarized the document and made a description of the future work that can be done to improve the implemented solution.

Chapter 2

State of the Art

This chapter details the state of the art in this context, for a better understanding of the strategies that can be used to extract information and measures to calculate the semantic similarity.

At first, section 2.1 explains what is information extraction and the different phases for text processing, the algorithms used and the different usages. More focused on the context of this project, it is explained some of the approaches used to extract ingredients and quantities from an online cooking recipe.

Then, in section 2.2, it is explained why a direct verification of the equality of two terms sometimes is not enough. Giving then a glance at the different approaches already used. Explaining also the need of external structures, such as ontologies, thesaurus or taxonomies, to help in the process of matching ingredients with commercial products.

Afterwards, in section 2.3, is made a description of the type of informational resources existing with some examples for each one.

Finally, in section 2.4, are described some relevant tools and programming languages, that can be used in the context of this project.

2.1 Information Extraction

The idea behind information extraction (IE) is to automatically retrieve information from a natural language text. Russel and Norving [RN10] state that IE goal is to process natural language text in order to retrieve certain class of objects or events and the relation between them. Similarly, Riloff states that IE is a way of natural language processing in which certain information is recognized and extracted from the text [WD10].

In the last years, the amount of information published on social networks, blogs, websites and others have been increasing at a fast pace. Most of this information is shared on free text, without structure. This led to the necessity to effectively analyze text and retrieve valuable information in

a structured way, leading to the rise of IE algorithms [PY13]. One of the challenges faced is that a word depending on the context can have different meanings [IBM].

Depending on the problem being tackled there are multiple different approaches that can be followed. However, since it can be a complex activity, normally it is divided into different tasks: Segmentation, Classification, Association and Normalization and Co-reference Resolution [Sim05] [McC05]. The division of the information extraction activity into smaller tasks brings some advantages such as the use of different techniques and algorithms that best fit the task for the problem being tackled.

Normally, the pipeline presented on an information extraction activity is the one shown in Figure 2.1

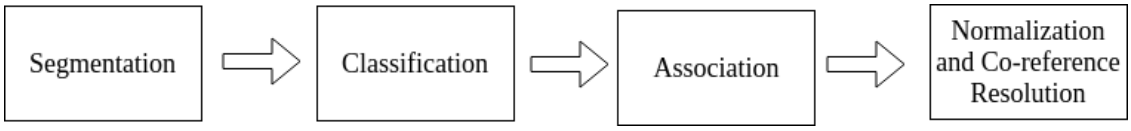


Figure 2.1: General pipeline for information extraction activity

The tasks mentioned above can be described as:

Segmentation

The goal of this task is to divide the text and sentences into atomic elements called segments or tokens. Despite the fact the Western languages have this task simplified, due to the whitespaces however some times that is not enough [Sim]. Depending on the language different problems can be faced [Sim05]:

- **Whitespace**: It is generally considered a word boundary, but sometimes they should not be considered as a different token. Take as an example "Good morning". They are separated by a whitespace, but they refer to only one concept.
- **Hyphens**: Compound words sometimes should be considered as only one token, like "mother-in-law" and other times it should be considered as different tokens "non-layer".
- **Apostrophes**: Can lead into ambiguities when segmenting the text with contraction forms.
- **Full stop**: Normally the period mark means the end of a sentence, but that is not entirely true. It can also be used for abbreviations like "P.S" that stands for "postscript".

Depending on the problem and the language of the text that it will be processed the technique used can change in order to better adapt to the problem. One of those techniques can pass by using external resources, such as grammars, to perform a syntactic or lexical analysis. Besides that, it can also be used Hidden Markov Models, which uses hand-segmented words. This task can also be analyzed statistically using n-grams or Viterbi algorithm. [Sim]

Classification

Occurs after the segmentation task. The main objective is given a set of classes determine to which class the segments or tokens belongs to. At the end of this task, it is expected to have a list of segments with the respective classification. Sometimes this task can also occur at the same time as Segmentation with a finite state machine. [McC05]

Practical examples of the use of classification are for example:

- Spam filtering
- Email routing
- Language identification
- Sentiment analysis

Like in any other problem, the difficulties faced when tackling a problem depending on the algorithm used. In a general way this task as the following problems [Sil18]:

- **Homonyms and homographs:** Humans are used to ambiguity, but computers don't, so they don't deal well with it. One simple example is the word "rock". Depending on the context it can mean a style of music or it can refer to a stone.
- **Specific domain:** When processing a text it is necessary to pay attention to the domain. Depending on the domain different expressions can be used, as so reuse a system used for another domain, may need readjustments for a good performance.

Some of the techniques used to classify text are generally also used for other types of data. Those techniques are [AZ12]:

- Decision Tree
- Rule-based classifiers
- SVM
- Neural Networks
- Bayesian (Generative) classifiers

Association

The association task tries to find which entities previously classified are related, taking cues from the text. Sometimes this task is referred as *relation extraction*. Let's take for example the sentence "Pedro works for company X" then there is an association worksFor(Pedro -> X) [McC05].

To perform this task the methods can be characterized as [Kon18]:

- **Knowledge-based:** It is usually used on domain-specific tasks. In sum, this method since is very domain-specific is not easily portable to another domain. That would involve a lot of human labour.
- **Supervised:** It relies on the training set used to train the model. The principal problem is that train the model to suit the tagged corpus can take a lot of time. On the other hand, it is easily changed the domain as long as there is training data available. Some of these methods are: kernel methods, logistic regression, augmented parsing and conditional random fields.
- **Self-supervised:** It is an autonomous supervised learning. This approach eliminates the necessity of labelling the data by a human. It uses patterns to automatically extract relation extraction rules.

Normalization and Co-reference Resolution

Is the less generic task, because its dependent on the context.

Normalization is used to transform all the data into a standard representation, chosen by the developer. For instance the hours "14:00", "2pm" or "14H", can be transformed in "14H".

Co-reference arises when different expressions are used to refer for example to an ingredient, this can be done by [McC05]:

- Different names
- Classification expressions
- Pronouns

Different approaches were made. Rule-based was used to deal with co-reference considering the semantic relatedness between the entities. With this, it can be made filtering to detect the semantic information associated. Then it is calculated the probability of being a co-referent [Sim05].

Machine learning can also be used as a technique to tackle the problem of a co-referent task. This approach can be done using clustering algorithms. On this approach it is proposed an analysis of the text, calculating the distance between entities. In the end, if two entities were considered to be on the same cluster, it was considered to be co-related [CW99].

2.1.1 Related Work

Over the years different approaches have been made to efficiently extract the ingredients and respective quantities from a cooking recipe.

One of those approaches is made by Erica Greene [Gre]. Is propose intends to convert unstructured data, like a cooking recipe, into structured data. To achieve this it tries to predict the structure of the phrase using a conditional random field (CRF). To remove the data in a structured way the algorithm uses two lists, one containing the phrases and the other containing the tags (NAME,

QUANTITY, UNIT, COMMENT and OTHER). Then the idea is to learn a model to predict the correct tag sequence using conditional probability. It would not be efficient to compute the score of all, but with CRF it reduces the computational time.

A hybrid approach is made by Hamon and Grabar [HG13] based on the French language. It combines rube-base and machine learning system. The first one was used to recognize terms and associated information. It is divided into three main steps:

- *Term extraction* where all the entries are recognized and extracted, as well as, the information associated;
- *Ingredient name weighting* to identify the most important ingredients on the recipe by is weight. The weight is given according to his position on the recipe, the frequency of his canonical form and the association with the respective quantities;
- *Ingredient name filtering*, performed after the respective weighting, scoring and filtering. If the case of the same weight to different ingredient names happens, the score is given according to the canonical form frequency within the recipe.

The other one used is machine learning with CRF. As input uses the output that comes from applying the rule-based method. The sentences are considered as sequences, where each element is linguistically annotated. In the end, the output is post-processed selecting the correct form of the ingredient name.

Mori et al. [MSYY12] on the other side approach this problem with machine learning. He divides the problem into four different tasks for disambiguation: word segmentation, natural entity recognition, syntactic analysis and predicate-argument structure analysis. After that, the input sentence is transformed into a dependency tree.

2.2 Similarity Matching

Through all the internet concepts are written in different forms. Two terms can mean the same thing, for example, the terms "residence" and "home". Some of those terms can be written with errors. Because of that sometimes to verify if two terms mean the same, a simple check if they are written the same way is not enough.

Humans learn how to deal with the ambiguity of the language, in this case, the Portuguese language, which in the case can be very ambiguous. "Computers can be programmed to do the same thing by consulting data structures called "ontologies" that represent terms and their relationships" [SEGJ08].

Ontology is a way to represent knowledge on a specific domain. They represent the properties of terms and relationships between terms [CJB99]. In this context, there is, for example, FoodOn [1] an ontology that represents the knowledge about food in a structured and unambiguous way. In addition to an ontology, there also thesaurus and taxonomies that also represent knowledge.

[1] https://foodon.org/

The relatedness of two terms can be calculated using ontologies can be done using: path-based methods, feature-based methods, measures based on information content and hybrid measures [TAH14].

- **Structured based** - Also know as edge counting or path based, measures the similarity between two terms based on the number of taxonomic links between them on an ontology hierarchy structure(is-a, part-of) [Sli13]. It assumes that the link distance between concepts is uniform.

  A simple measure, proposed by Rada et al. (1989) uses distance as a conceptual measure to calculate the similarity between concepts. Conceptual distance is a metric because it follows some properties: zero property, symmetric property, positive property and triangular inequality. Distance is the inverse of similarity. Which means the shorter the distance, the more semantic close the terms are. [RMBB89].

$$Sim(C1,C2) = shortestPath(C1,C2) \tag{2.1}$$

  An extension of the approach above is done by giving different weights to the links between concepts based on the depth of the taxonomy and density of the taxonomy and connotation between a concept and is parent. [Sli13]

  Leacock & Chodorow (1998) introduced a new variant. They proposed to normalize the distance between two concepts is calculated using the shortest path and normalized with the maximum depth(D) of the taxonomy.[Sli13]

$$Sim(C1,C2) = -\log\left(\frac{shortestPath(C1,C2)}{2*D}\right) \tag{2.2}$$

  Wu & Palmer (1994), made a different approach. Considering C1 and C2 the concepts on which it is pretended to calculate the similarity. Assuming that they are related there is at least one common concept between them and if there are multiple they have different specificity. So the idea is to use the distance between c1 and C2 to the most specific common concept(N1 and N2) and distance from there to root the (N3).[WP94]

$$Sim(C1,C2) = \frac{2*N3}{N1+N2+2*N3} \tag{2.3}$$

  Hirst and St-Onge(HSO) proposed a measure to calculate the relatedness between concepts with the path distance, the number of changes in direction of the path and the allowableness of the path. An allowable path is a path that does not deviate from the meaning of the source concept. Let d be the number of changes of direction, k and S are constants that are derived from experiments. [Sli13] [HSO95]

$$Sim(C1,C2) = S - shortestPath(C1,C2) - k*d \tag{2.4}$$

T. Slimani, B. Ben Yaghlane, and K. Melloulil [SBYM04], adopted as a base to their work the Wu and Palmer semantic similarity, since a comparison between methods carried out by Dekang Lin [Lin98], reveals that besides being simple to calculate it is as expressive as others. However, Wu and Palmer's measure has a problem that comes from the fact that the edges on an ontology have uniform distances. This can lead to a problem, that two concepts on the same hierarchy can be less similar than two neighbours. Taking the ontology represented on figure 2.2, according to Wu and Palmer the semantic similarity between the concepts C3 and C4 is greater than C3 and C1. Which can be a problem in semantic information retrieval.

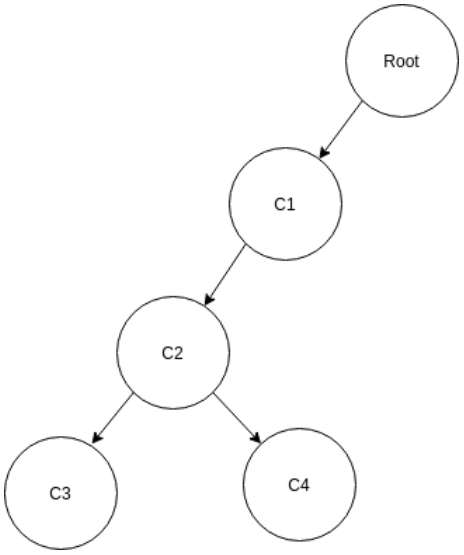


Figure 2.2: Ontology representation

So they propose an extension of Wu and Palmer approach, by penalizing the value of similarity between two concepts if not on the same hierarchy. Formally represented by:

$$Sim(C1,C2) = \frac{2*N3}{N1+N2+2*N3} * PF(C1,C2) \tag{2.5}$$

$$PF(C1,C2) = (1-\lambda)*(Min(N1,N2)-N)+\lambda*(\mid N1-N2 \mid +1)^{-1} \tag{2.6}$$

where N1 and N2 is the distance from the concepts to the root and N3 is the distance from the lowest common subsumer to the root. λ is a boolean variable, that indicates 0 or 1. If 0 indicates two concepts in the same hierarchy and if 1 indicates two concepts in the neighbourhood.

Yuhua Li, Zuhair A. Bandar, and David McLean [LBM03] consider that a major problem of other path-based metrics, such as the ones stated above, is that they use a particular information source without considering the contribution of others. So they propose a semantic

similarity between two concepts as a function of the attributes path length, depth and local density. Formally represented by:

$$Sim(W1,W2) = f(l,d,h) \tag{2.7}$$

where l stands for the shortest path length between C1 and C2. h represents the depth of the lowest common subsumer on the hierarchy and d the local density of W1 and W2. They made a few experiments combining depth, path and local density. Concluding that the one that comes closest to the individual human judgment(with a correlation of 0.9015) is given by the equation below (with a correlation coefficient of 0.8914):

$$Sim(W1,W2) = e^{-\alpha * l} * \frac{e^{\beta * h} - e^{-\beta * h}}{e^{\beta * h} + e^{-\beta * h}} \tag{2.8}$$

where $\alpha \geq 0$ and $\beta > 0$ are weight constants for scaling the contribution of shortest path length and depth, respectively. Using the a benchmark data set, they conclude that the best value for the parameters are: $\alpha = 0.2$ and $\beta = 0.6$.

- **Information content(IC)** - Calculates the semantic similarity between two concepts, as a function of the information content that both concepts share [HTBAB15]. IC is calculated, by first calculating the frequency of each term on the corpus. After that, the value of IC is given by the negative log of the probability of its occurrence $IC = -log(p(A))$. With this calculation, it is given more impact to the ones that have a low frequency. Which make sense because the higher the frequency the less informative is. Then the similarity between two terms is obtained by the IC of the Least Common Subsumer (LCS) [Res95] [SBIV12].

$$Sim(C1,C2) = IC(LCS(C1,C2)) \tag{2.9}$$

Nuno Seco, Tony Veale and Jer Hayes [SVH04], proposed a new formula to calculate the IC of a concept. They believe that the WordNet taxonomy can be used to calculate IC value alone. For that, they assume that the taxonomy is organized in a meaningful and principled way. Arguing that the more hyponyms a concept has the less information it brings and so the leaf nodes are the concepts that express the maximal information. Then, the IC value can be calculated with:

$$IC(c) = 1 - \frac{\log(hypo(C) + 1)}{\log(n_{concepts})} \tag{2.10}$$

In the formula function *hypo* returns the number of hyponyms of the given concept, *n_concepts* is a constant that represents the maximum number of concepts on the taxonomy and k is a weight constant.

Zili Zhou, Yanna Wang and Junzhong Gu [ZWG08a], proposed a new approach to calculate the IC value. This approach includes the depth of the concept and the maximum depth of the taxonomy, in order to try to emphasize the discrete nature of the concepts. So they came up with the following formula to calculate the IC value:

$$IC(C) = k * (1 - \frac{\log(hypo(C)+1)}{\log(n_{concepts})}) + (1-k) * (\frac{\log(deep(c))}{\log(max_{deep})}) \tag{2.11}$$

Where, as stated above, the function *hypo* gives the number of hyponyms of the concept. The function *deep(c)* gives the depth of the concept in the given taxonomy and *max_deep* is a constant that represents the maximum depth of the taxonomy Lin and Jiang & Conrath measures try to improve Resnik approach, by increasing the individual IC of each concept [Ped10].

Lin measure is the ratio between the amount of information needed to assume commonality between two concepts and the amount of information needed to completely describe the concepts. [Lin98].

$$Sim(C1,C2) = \frac{2 * IC(MICA(C1,C2))}{IC(C1) + IC(C2)} \tag{2.12}$$

Jiang & Conrath proposed a combined approach between edge-based notion deriving the semantic distance from it and IC as a decision factor. In order to tackle the problem of edge counting the edges are weighted according to the associated probability based on corpus statistics and also the strength of the link between concepts. [JJWC97]

$$Sim(C1,C2) = IC(C1) + IC(C2) - 2 * IC(MICA(C1,C2)) \tag{2.13}$$

- **Feature based** - Calculates the semantic similarity between concepts as a function of their properties.

  Tversky [Tve77] came up with an approach where he ignores the position of the concepts in the taxonomy as well as the information content of them.[Sli13] The similarity between concepts is calculated as a function of common and noncommon properties. Common characteristics tend to increase the similarity while the noncommon tend to diminish. [SBIV12] To get the semantic similarity between two concepts, Tversky proposed the following formula:

$$Sim(C1,C2) = \frac{|X \cap Y|}{|X \cap Y| + \alpha * (|X - Y|) + \beta(|Y - X|)} \tag{2.14}$$

where X and Y are the sets of features of the concepts (C1 and C2) and $\alpha, \beta > 0$, which are index parameters.

M. Andrea Rodríguez and Max J. Egenhofer [RE03], based the semantic similarity between two concepts as a weighted sum of the similarity between synsets, features and neighbour concepts.

$$Sim(C1,C2) = w * S_{synsets}(C1,C2) + u * S_{features}(C1,C2) + v * S_{neigborhoods}(C1,C2) \quad (2.15)$$

where w, u and v are constants that represent the weight given to each component. The function S expresses the overlapping between features and can be formally represented as:

$$Sim(a,b) = \frac{|\, A \cap B \,|}{|\, A \cap B \,| + \gamma(a,b) * (|\, A \backslash B \,|) + (1 - \gamma(a,b)) * |\, B \backslash A \,|)} \quad (2.16)$$

$\gamma(a,b)$ expresses the relative importance that non-common characteristics have and can be calculated with the following:

$$\gamma(a^p, b^q) = \begin{cases} \frac{depth(a^p)}{depth(a^p)+depth(b^q)}, & \text{if } depth(a^p) \leq depth(b^q) \\ 1 - \frac{depth(a^p)}{depth(a^p)+depth(b^q)}, & \text{if } depth(a^p) > depth(b^q) \end{cases} \quad (2.17)$$

- **Hybrid** - A hybrid approach combines structural characteristics (path length and depth) with some of the above approaches, trying this way to use the best points of each measure. [TAH14]

Zili Zhou, Yanna Wang, Junzhong Gu [ZWG08b], proposed a hybrid semantic similarity measure combining the path length with the IC value. Path length uses the path, which is a highly discrete factor, as a parameter to calculate the semantic similarity between concepts, making difficult to improve the performance. On the other side, information-based approaches use as parameter the IC value, that is a continuous factor. This can lead to a more accurate calculation of the semantic similarity. Sometimes, it can happen the case that two pairs of nodes have the same semantic similarity even though the distance is different and the pairs are at different levels because they share the same lowest super-ordinate. To distinguish the semantic similarity in these cases it is introduced the path distance. So the formula proposed to calculate how similar two concepts are the one bellow. In the calculation, the variable k is a weight factor. If k=0 it is only used IC value if k=1 it is only used

the path length. So if k=0.5, it is given the same weight to both.

$$Sim(C1,C2) = 1 - k * \frac{\log(shortestPath(C1,C2)+1)}{\log(2*max_{deep}-1)} - (1-k)*(\frac{IC(C1)+IC(C2)-2*IC(LSC(C1,C2))}{2}) \tag{2.18}$$

A sum up of all semantic similarities aborded above is presented in table 2.1.

Structure Based	Rada et al.
	Leacock & Chodorow
	Wu & Palmer
	Hirst and St-Onge
	T.Slimani et al.
	Yuhua Li et al.
Information Content	Resnik
	Lin
	Jian & Conrath
Feature Based	Tversky
	M. Andrea Rodríguez and Max J. Egenhofer
Hybrid	Zili Zhou et al.

Table 2.1: Semantic similarities

2.3 Information Sources

Information sources can be represented in different ways, such as ontology, taxonomy or thesaurus. The structure of those representations is described below with an example for a better understanding.

Ontology

Ontology has becoming more and more important due to the lack of standards. It is built to conceptualize and represent knowledge in order to tackle various challenges. Extract accurate information from the web is arduous because current search engines use keyword-based search techniques. One of the challenges is the incapability to correctly use the abundant information on the web. The integration of information from various sources is another challenge due to the existence of synonyms and homonyms.[SA17]

The DBpedia [2] is one example of an ontology. It brings a huge amount of information by extracting structured information from Wikipedia [3]. It is currently available in 125 languages. The English version has 4.22 million things described in the ontology, where the most common are on table 2.2[Abo].

[2] https://wiki.dbpedia.org/
[3] https://www.wikipedia.org/

Table 2.2: Distribution of the most frequency terms on DBpedia

Class	Entities
Person	1.445.000
Place	735.000
Organization	241.000
Specie	251.000
Disease	6.000
Creative work	411.000

Over other knowledge bases, DBpedia has several advantages, such as:

- covers main domains
- automatically update with Wikipedia changes
- it is multilingual
- it is accessible on the web

Due to the wide range of covered domains and the amount of information that it has, many publishers have started making RDF links between their data source and DBpedia. As a result, DBpedia is one of the central interlinking hubs of Web of Data. [BLK+09]

Another example of an ontology is FoodOn. FoodOn [4] is an open-source ontology, built to interoperate with the OBO Library. OBO Foundry [5] has the goal to develop a family of interoperable ontologies that are scientifically accurate as well as logically well-formed [SA+07]. FoodOn contains plant and animal food sources, food categories and products, contact surfaces, packaging and preservation process. It is provided in the Web Ontology Language(OWL) in English.[DGG+18]

Taxonomy

Initially, taxonomy was used to the hierarchical classification of life forms in the 18th century by Carl Von Linné. Nowadays not only continues to be used for that purpose but is also used as a controlled vocabulary with hierarchy structure used for the classification of things or terms. This hierarchy is based on one type of relationship narrower/broader.[Gar04]

The most well-known example of a taxonomy is WordNet[6]. It is a lexical database of English available online. Nouns, verbs, adjectives and adverbs are organized into sets of synonyms, where each one represents a different concept. A synonym set is also called a synset. WordNet includes the following semantic relations that can be between word forms or synsets[Mil95]:

- **Synonymy** - Is a symmetric relation between word forms.

[4] https://foodon.org/
[5] http://www.obofoundry.org/
[6] https://wordnet.princeton.edu/

- **Antonymy** - Is a symmetric relation, but to mean the opposite between word forms. It is paramount to organize the meaning of adjectives and adverbs.
- **Hyponym** - It is a transitive relation between synsets. This relation organizes nouns into a hierarchical structure.
- **Meronym** - Expresses a part of a hole, distinguishing component parts, substantive parts and member parts.
- **Troponym** - As the same meaning for verbs as hyponym as for nouns, however, the structures are more superficial.
- **Entailment** - Expresses relations between verbs.

Thesaurus

Thesaurus is an extension of a taxonomy. It stills use a hierarchical structure using narrower and broader relation. It is a closed vocabulary and also allows other statements. Moreover, it includes the possibility to have a relationship that refers to another term, which implies they are synonymous. And scope note property, that is a string attached to the term that explains the meaning of the term within the thesaurus. [Gar04]

AGROVOC[7] is a multilingual thesaurus that is available in 19 languages and is managed by FAO[8] of the UN (Food and Agriculture of the United Nations). It covers areas of interest such as agriculture, forestry, fisheries and environment. In the beginning, all the information was on a relational database. In 2004 it was tried to bring AGROVOC to the semantic web, using OWL due to be, at the time, the best option to merging from relational databases to the web, while enabling a rich domain specification. AGROVOC currently uses SKOS-XL, that is an extension of the Simple Knowledge Organization System (SKOS), to handle labels. To manage the relations broader term/narrower term (BT/NT), it uses SKOS. [CRM12]

2.4 Programming Languages and Frameworks

The most popular languages in general, are also the most popular for natural language programming. Those languages are Python and Java. This popularity of them comes from the support of the respective community and the fact that the learning curve is not high. Moreover, is popularity comes also from the toolkits to make the NLP tasks, such as NLTK [9], Pattern [10], OpenNLP [11] and CoreNLP [12] [POO16].

For Python, the packages available are NLTK and Pattern. The first one provides a good interface for external resources, like WordNet. Besides that, it gives support for the different

[7] http://aims.fao.org/standards/agrovoc/concept-scheme
[8] http://www.fao.org/home/en/
[9] https://www.nltk.org/
[10] https://www.clips.uantwerpen.be/pages/pattern
[11] https://opennlp.apache.org/
[12] https://stanfordnlp.github.io/CoreNLP/

phases of text processing and it is open source. The second one supports data mining, machine learning, natural language processing as well as network analysis.

In respect to Java, the packages are OpenNLP and CoreNLP. OpenNLP uses machine learning with NLP. CoreNLP, it was designed to be extensible and flexible. It integrates many of Stanford's NLP tools.

Both languages have support for neo4j [13], which is a graph platform. That was built to successfully store, handle and query highly connected information, such as an ontology or a taxonomy. It has support for data visualization and a friendly query language, Cypher, that is based on SQL but optimized for graphs.

2.5 Conclusion

With the increasing number of shared information over the internet, the necessity of efficiently process the information on it increased. Because of that the number of work done on Natural Language Processing in the last years raised. In this chapter, it is made a review of the state of the art as well as some background relatively to information extraction and similarity matching.

With some search in the text processing, it was possible to verify that in general it is divided into four main tasks: segmentation, classification, association and Normalization and Co-reference Resolution. The analysis of previous works enabled to detect which were the problems that they faced and how did they pass through or not. As well as, identify their strengths and weaknesses.

It is also made an assessment of the existing semantic similarity measures. Those can be path-based, information content-based, feature based measures and also hybrid measures.

All in all, with this chapter it is possible to analyze previous works, taking notes of what went well and what didn't. Based on the annotations taken on this chapter it was designed and created a system that is explained in detail in chapter 3, in order to achieve the desired goal.

[13] https://neo4j.com/

Chapter 3

Development and Methodology

This chapter makes a description of the developed solution. It can be divided into 5 sections: *Problem Statement*, *Technology*, *Architecture*, *Information Extraction* and *Similarity Match*.

At first, on section 3.1 it is made a brief description of the problem that the solution implemented attempts to tackle.

Then, on section 3.2, it is detailed the technologies used and made a brief explanation about why they were chosen to implement the proposed solution.

On section 3.3 it is made an overview of the system architecture, for a better understanding of the system flow and the components that compose it.

Finally, the last two sections (3.4 and 3.5) are dedicated to the main components of the system, where are provided details of the implementation and explained the approaches that were taken.

3.1 Problem Statement

The purpose of this thesis is to retrieve a list of real products for each ingredient, given an online cooking recipe. However, it faces some problems related to the extraction of the ingredient names from recipes and to the match between the extracted ingredients and the name of products present on a real database.

The main problem of the extraction of ingredients from a cooking recipe is that they are in free text. Which means that they do not have a structure like a programming language or a zipcode would have because they have a predefined manner of writing it. The lack of structure makes arduous the task for machines to read it and process it.

Relatively to the match of ingredients with products the problems fall on the existence of meronyms, synonyms, hyponyms and hyperonyms on cooking recipes as well as the presence of misspelling errors. Moreover not always the names presented on the database correspond to what is expected. Considering it is a real database it is expected the existence of some noise. As a result, it is not a straightforward task.

3.2 Technology

The chosen language for this project was Python. This decision was taken based on the fact that it has good community support, a good range of packages that can be used such as NLTK and others for data analysis and machine learning.

For the first part of the project, Information Extraction, it was used Scrapy [1] and brat. Scrapy is a framework built with the purpose to crawl web sites in order to extract structured data [Scr]. It was used with the goal of creating a dataset of cooking recipes. Brat is a web-based tool, designed for text annotation. It is fully configurable and it is ready to support most of the text annotation tasks. Each annotation task is defined by a set of tags: *entities*, *relations*, *events* and *attributes*. As input, the software accepts a plain text file and the output is made for a different file identified with the suffix .ann. Each line in the output file correspond to one annotation, wherein the beginning is a Uniform Resource Identifier(URI), that permits to uniquely identify any annotation [SPT+12]. In figure 3.1 is represented an example of the Brat interface and on figure 3.2 is shown an example of the Brat output.

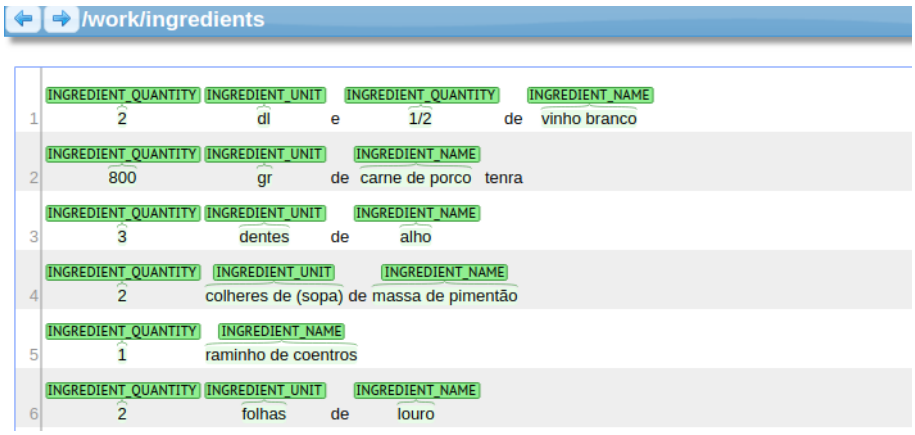


Figure 3.1: Example of Brat interface

```
T1  INGREDIENT_UNIT 2 4 dl
T2  INGREDIENT_QUANTITY 0 1 2
T3  INGREDIENT_NAME 14 26   vinho branco
T4  INGREDIENT_QUANTITY 7 10    1/2
T5  INGREDIENT_QUANTITY 27 30   800
T6  INGREDIENT_UNIT 31 33   gr
T7  INGREDIENT_NAME 37 51   carne de porco
```

Figure 3.2: Example of Brat output file

Neo4j was chosen to save the ontology and also the product database. Neo4j is a graph platform, that was built to store, handle and perform queries in highly connected information, which

[1] https://scrapy.org/

is the case of an ontology, a taxonomy or a thesaurus. The information is represented by nodes and their relationships like it can be shown in figure 3.3. A knowledge structure, like the ones aforementioned, can be easily loaded to a neo4j database using a library from Jesús Barrasa [2].

Figure 3.3: Example of graph visualization on Neo4j

Cypher is the language used to perform queries to the neo4j database. This language is based on SQL but optimized for graphs and it holds a simple and intuitive language. Nodes in cypher are represented with parentheses e.g (NODE) and relationships are represented with brackets. Cypher also allows to indicate the direction of the relationship with "<- -" or "- ->". If the direction of the relationship is anonymous is used double dashes "- -". Below it is possible to see an example of a cypher query, that returns the category of a product, taking into consideration the representation seen on figure 3.4

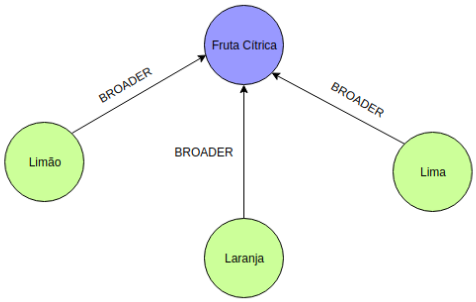


Figure 3.4: Example of graph representation

```
1  MATCH (p:Product)-[:BROADER]-> (c:Category)
2  WHERE p.name = "Laranja"
3  RETURN c
```

Listing 3.1: Example of cypher query

[2]https://jbarrasa.com/2016/06/07/importing-rdf-data-into-neo4j/

Queries can be performed to a neo4j database with Cypher in Python, through an official neo4j driver for python [3].

3.3 Architecture

After analyzing the problems to tackle, picked the technologies and taking into consideration the state of the art, it was created the following pipeline presented in figure 3.5.

The system can be divided into five relevant parts:

- **Information extraction** from recipes, retrieving structured data from unstructured text.
- **Search ingredients on AGROVOC**, to get the respective node for an ingredient.
- **Search products**, in order to get the products that are textually closer to the ingredient.
- **Search products on AGROVOC**, to obtain the respective nodes for the products.
- **Semantic similarity matching**, to get the best match from products with an ingredient.

3.4 Information Extraction

3.4.1 Methodology

Model

The model used to extract structured information from unstructured line of ingredients, was the one used by Nuno Silva [Sil18] that is based on the one developed by Erica Green et al. at the New York Times[4] [Gre].

The structure of an ingredient line on a recipe does not vary much. Therefore the algorithm used was a linear-chain Conditional Random Fields(CRF) to extract structured information[SRF19]. CRF allows to extract structured information even though the model has never seen the ingredient phrase. This is done by modelling the conditional probability of a sequence of tags given a certain input. This can be formally denoted by *p(tag sequence | ingredient phrase)*.[Gre] In sum, the goal of the model used is to correctly predict a sequence of tags, that can be *NAME*, *UNIT*, *QUANTITY*, *COMMENT* or *OTHER* [Sil18].

An example of the model output is shown in figure 3.6.

Annotation

It was annotated a data set from a different source in order to train and evaluate the model described above.

[3] https://neo4j.com/developer/python/
[4] https://www.nytimes.com/

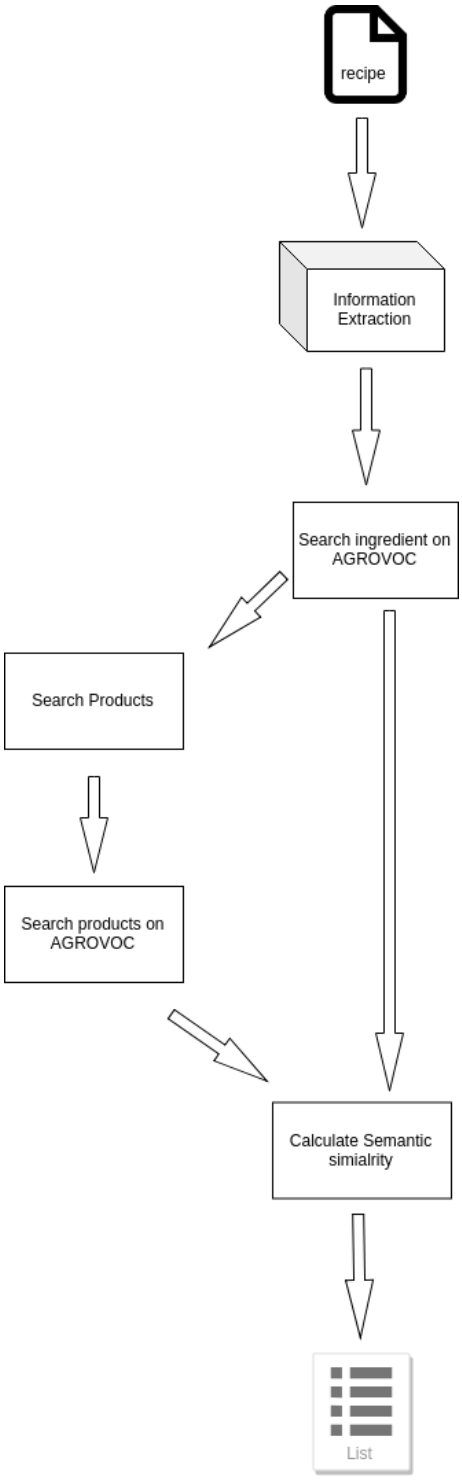


Figure 3.5: System pipeline

Since the scope of this thesis is the Portuguese language, it was paramount that the data set of cooking recipes were written in Portuguese from Portugal. So after some research, the chosen website was http://www.receitas-portuguesas.com/.

Figure 3.6: Example of output from New York Times CRF
[Gre]

From each recipe, only the list of ingredients was extracted like shown on the figure, because the objective is to convert a cooking recipe into a grocery list, so there is no need to get the rest of the recipe such as the cooking processes. In order to collect the structured data from the recipes, it was used the web crawler mentioned above, Scrapy.

The resulting data set is constituted by 137 cooking recipes, which correspond to 1343 lines of ingredients.

After extracting all the information necessary, it was performed text annotation on the ingredient lines using Brat as stated above. Brat allows to annotate *entities*, *relations*, *attributes* and *events*, however for the case it was only necessary to annotate entities. There were used four different entities, according to the ones presented on the model used:

- ***NAME***: The entity is used to identify the name of the ingredient present on the ingredient line.
- ***QUANTITY***: The entity is used to identify the ingredient's quantity needed.
- ***UNIT***: The entity is used to identify the unit of the quantity necessary. For example: "gr", "Kg" or "dl".
- ***OTHER***: The entity is used to identify a comment that sometimes is used to give more information about the ingredient or to express a process that must be applied to the ingredient.

Prior to the begin of the annotation process a few guidelines were made, because there are cases that even for humans its hard to classify:

- The name of the ingredients was annotated taking into consideration is natural form. For example from the following line *"200 gr de queijo fresco para barrar"*, *"queijo fresco"* would be annotated as *NAME*.
- Some cases are a little more complicated, such as "3 dentes de alho" or "2 folhas de louro", some could annotate "dentes de alho" and "folhas de louro" as *NAME* and others would prefer to annotate "dentes" and "folhas" as *UNIT*. In these cases, it was decided to go for the second option.

– If was mentioned some process to be applied or detail to the ingredient it was considered as *OTHER*. For example: "cortado as rodelas" or "grande picada".

3.5 Semantic Match

3.5.1 Data Sources

In order to assist in the process of obtaining a grocery list from a recipe, it was necessary to use external resources to calculate the semantic similarity between concepts. Some of the possibilities were Foodon, DBpedia, AGROVOC and WordNet.

DBpedia was a good possibility due to the huge amount of information available, moreover, it is constantly being updated with Wikipedia changes. However, it has a scope too wide for this purpose.

WordNet initially was for the English language exclusively, however already exists a Portuguese version [5]. It is hierarchically represented and it has relations between synsets and word forms that express synonymy, which it is necessary to tackle some of the problems. But once again it has a too wide range for the context of the project.

FoodOn is a food-related ontology. Not only has a hierarchical structure (using relation is-a), but also has relations to properties of the food and products. For example it has information like: *apple (caramel-coated) - has part -> caramel*. Meaning that concrete apple has caramel on it. This kind of properties are the ones used in feature-based semantics to calculate the semantic similarity between two concepts. Nevertheless, it does not support the Portuguese language.

AGROVOC is a multilingual thesaurus related to agriculture, forestry, fisheries and environment. It supports the Portuguese language, it has a good variety of terms and the scope is not too wide like in some of the above resources. For that, it was the knowledge resource chosen. The concept scheme of AGROVOC has concepts, relations and terms. Concepts are something that we want to represent or express. Terms are what represent the name for concepts, by means of SKOS-XL (skosxl:prefLabel and skosxl:altLabel). Relations can be hierarchical representing the notion of BT/NT with SKOS (skos:broader and skos:narrower) and non-hierarchical expressing relatedness (skos:related) and more specific vocabulary named Agrontology. It can be formally represented by figure 3.7

Since the idea is to return a list of products that best fit the necessities of a cooking recipe it was necessary to have a real database with real products. A product database was provided by Fraunhofer Portugal AICOS. The database is constituted by 2 different classes *Product* and *Category*. It has 36763 entries for class *Product* and 4980 entries for class *Category*. *Product* has a foreign key pointing to the class *Category*, on the other hand, *Category* has a self-reference because a category is constituted by other categories. This self-reference creates a hierarchy and the root node is obtained when it is referencing *null*. A formal representation of the diagram of classes is shown in figure 3.8

[5] http://wordnet.pt/

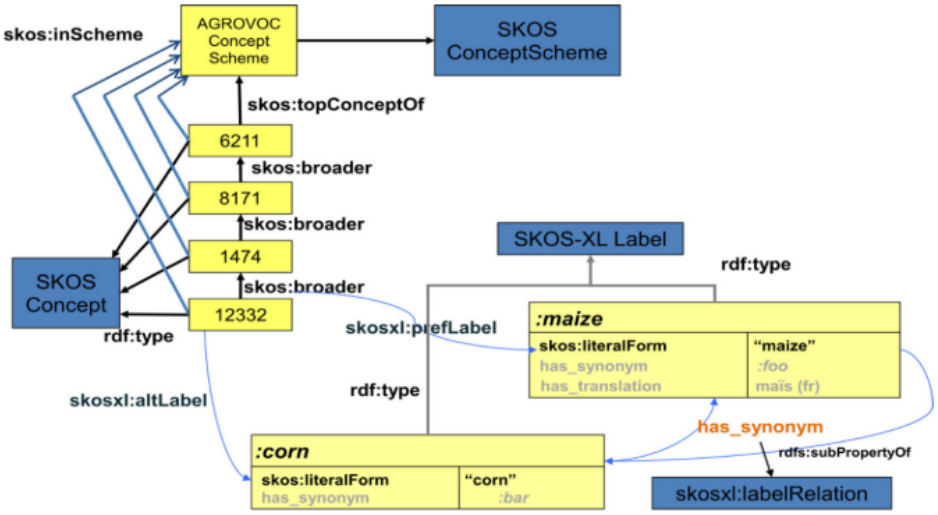


Figure 3.7: AGROVOC structure

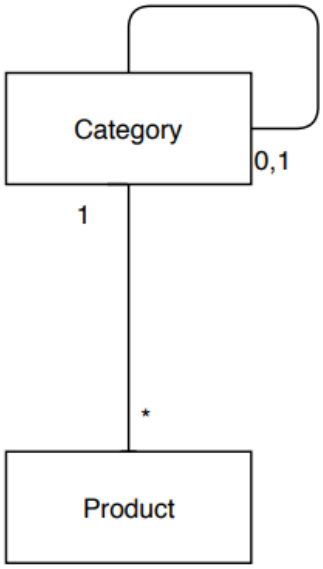


Figure 3.8: Database class diagram

3.5.2 Methodology

Load thesaurus to Neo4j

As mentioned in section 3.2 it was used Neo4j to save the AGROVOC thesaurus. For the purpose, to load the thesaurus presented on a file with extension .rdf was used as an external

library, *neosemantics* [6].

The framework allows loading an RDF file, among other extensions, with a cypher command, where one of the fields can have the following properties:

- **shortenUrls** - full urls are shortened using generated prefixes
- **typesToLabels** - if true, rdf:type is imported as node labels
- **languageFilter** - filter the labels imported by language
- **headerParams** - parameters to pass on a HTTP GET request
- **commitSize** - partial commit every n triples
- **nodeCacheSize** - retain n nodes in cache

Considering that the project is expected to work on online cooking recipes written in Portuguese, when importing the thesaurus it was filtered to exclusively load Portuguese labels. Bellow is the command used to load the RDF:

```
1 CALL semantics.importRDF("file:///.../agrovoc_2019-04-01_core.rdf", "RDF/XML
    ", {languageFilter: "pt", commitSize:5000, shortenUrls:true,
    typeToLabels:true})
```

Listing 3.2: Command used to import AGROVOC thesaurus

In total after performing the command, were loaded 3 829 880 triples. Unfortunately, even though the labels were filtered by the Portuguese language, there were still created nodes for the multiple languages however without literal. Since they were not relevant, every label that did not have a literal form was removed as well as all the relations that they had.

Sadly not all concepts have a Portuguese label, so when filtering by language some concepts got without any label. Every query made on the system always use concepts that have a literal on it. There are 4059 concepts that do not have a Portuguese label, so it would be a time-consuming task to translate all. However after analyzing it was possible to see that some of the top concepts did not have translation and their descendants had. Since it could have repercussions on the calculation of the semantic similarity, and it was a small number of labels, it was translated all the top concepts that did not have a Portuguese label.

Search on AGROVOC

For each ingredient, it was necessary to find the nodes presented on AGROVOC that were more similar to the entry and so could represent them on the thesaurus. The way found to look for the best representation was using full-text search, that at the same time help to tackle some problems, such as pluralized words or misspelling words.

[6] https://github.com/jbarrasa/neosemantics

Neo4j full-text search is performed making an analysis composed of three steps: character filtering, tokenization and token filtering. In character filtering the text is filtered from HTML character filtering and ASCII character collapsing. Then on tokenization, it is divided the text into tokens for on the next section be made filtering of them [Dee].

When creating the full-text index it is possible to add one argument that represents the analyzer chosen. All possible analyzers present can be listed, by performing the following command:

```
1 CALL db.index.fulltext.listAvailableAnalyzers()
```

Listing 3.3: Command used to list all analyzers

In total it has 43 analyzers, where the following are the best suited to the scope of the project:

- **portuguese** - For Portuguese language performs stemming and filters stopwords.
- **simple** - Does not perform stemming, but works well with european languages.
- **classic** - It is a classic Lucene analyzer, similar to *standard* however it has worse unicode support.
- **standard** - It is the default analyzer. Does not perform stemming, but filters punctuation and stopwords.

After analyzing the above analyzers it was possible to verify that since the *simple*, *classic* and *standard* does not perform stemming they do not work well with plural words. On the other hand, the *portuguese* performs stemming and for that handles quite well plural words. The *classic* has another drawback that is a weak unicode support so it does not give good results when the text has accents. For all these reasons the chosen analyzer was *portuguese*.

The target of the search is the name of the concepts or resources and therefore it was created the full-text search index, with:

```
1 CALL db.index.fulltext.createNodeIndex('f_index',['Resource'], ['
    ns0__literalForm'], {analyzer:'portuguese' })
```

Listing 3.4: Command used to create full-text index

The first argument is the name and identifier of the index. The second is the type of nodes analyzed. The third is the property of the node that is being indexed, which in this case represents the name. The last one is optional and it is for configuration, where can be set the pretended analyzer and so on.

After the index is created, it can be performed queries to the database using the full-text index created. The results coming from the query comes in descending scored order. The

score expresses how well the system thinks the result matches the query [51I]. The score is also known as TFIDF, that is a statistic of how well it is believed the word is important for a document on a corpus. It can be formally calculated by [Dee]:

$$tf(t,d) = f_{t,d} \tag{3.1}$$

$$tdf(t,D) = \log(\frac{N}{|d \in D : t \in d|}) \tag{3.2}$$

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{3.3}$$

where N correspond to the total number of documents, t represents the term, d represents a document and finally D correspond to a set of documents. $f_{t,d}$ is the raw count of a term in a document.

One of the problems stated above is the existence of synonyms in ingredient lines on cooking recipes. AGROVOC supports synonyms through a relation *ns2_hasSynonym*. Therefore, when looking for an ingredient it is verified if it has synonyms (with the command below) and if it does they are also used to search for products.

```
1 MATCH (n:Resource)-[:ns2__hasSynonym]->(x:Resource)
2 WHERE n.ns0__literalForm = ingredient
3 AND EXISTS(x.ns0__literalForm)
4 RETURN x.ns0__literalForm
```

Listing 3.5: Command used to look for synonyms

Although, full-text search makes satisfactory filtering of the results is not enough because it does not have context and, as a result, can pass by relevant pieces of information. In order to overcome this problem, was used word2vec. It attempts to produce word embeddings by training a two-layer neural network. For training, it gets a large corpus and produces a large vector with several dimensions. Each unique word present on the corpus is represented by a unique vector.[MCCD13] Word2vec can use either of two architectures, that are represented on figure 3.9 [MCCD13]:

- **Continuous bag-of-words(CBOW)** - On it, the non-linear hidden layer is removed and as results, the projection layer is shared for all words. It is called a bag of words because the order of the words presented does not have an influence on the projection. On calculations, it is also used the words from the future.
- **Skip-gram** - It is similar to CBOW, however instead of predicting the current word taking into consideration the context, attempts to maximize the classification of a word based on another one present on the same sentence.

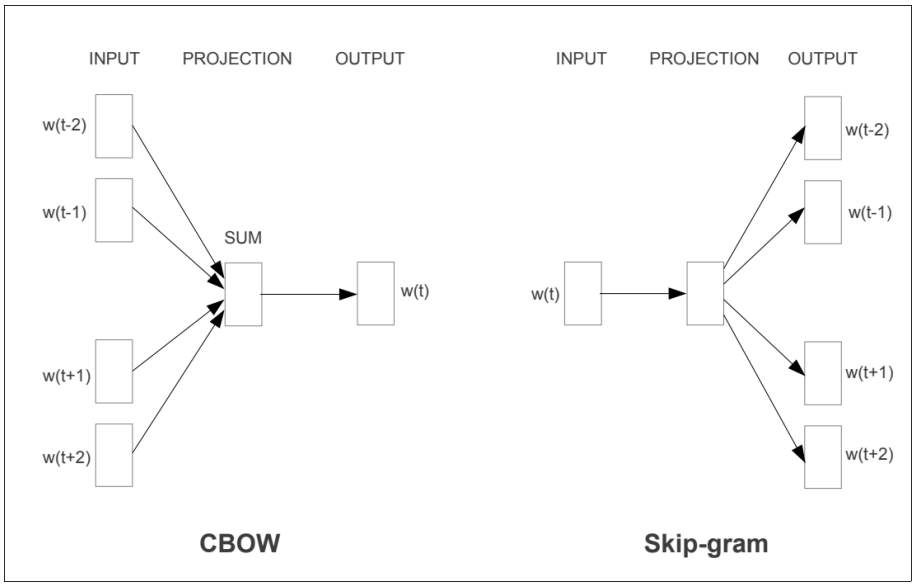


Figure 3.9: Word2vector CBOW and Skip-Ngram architecture [MCCD13]

On the project, it was used a pre-compiled model with a corpus with 1,395,926,282 tokens with the CBOW architecture.[HFS+17]

With word2vec it is calculated the similarity between the entry term and the results obtained from the full-text search performed. The returned node is the one that is more similar to the entry term.

Search on products database

In order to retrieve the appropriate products, it was first necessary to make a selection of the products that were textually more similar to the ingredient. The information about the products provided by a retailer was on a MySQL database and therefore to retrieve the products it was implemented a full-text search. The target of the index was exclusively the name of the product. The score retrieved from MySQL full-text search is different from the one explained above and it is given by:

$$w = \frac{(\log(dtf)+1)}{sumdtf} * \frac{U}{(1+0.0115*U)} * \log(\frac{(N-nf)}{nf}) \tag{3.4}$$

where *dtf* is the number of times the term appears in the document, *sumdtf* is the amount of $(\log(dtf)+1)$ for all terms on the same document, U is the number of unique terms, N represents the total number of documents and finally *nf* is the number of documents that contain the term.[MyS]

However the results obtained represented on table 3.1 were not satisfactory, because by performing a full-text query looking for "limão", the first ten documents have exactly the

same score and furthermore none of the results are the ones expected. The database as the product "Limão" and it has the same score as the ones on the table. So the length of the document is not being taken into consideration, resulting in the same result in almost every document that has the term "Limão" on it.

Table 3.1: Quering "Limão" on a full text search index on MySQL

Product Name	**Score**
Ice Tea Limão	4.576304912567139
Bloco Sanitário Sólido WC Activo Limão	4.576304912567139
Refrigerante sem Gás Limão	4.576304912567139
Continente Tinto Verão Limão	4.576304912567139
Grelhados de Frango Limão e Ervas	4.576304912567139
Ultra Suave Leite Hidratante Limão, Cafeína e Café Verde	4.576304912567139
Bífidus Líquido Magro Lima / Limão Activia	4.576304912567139
Chocolate de Limão e Cardamomo Bio	4.576304912567139
Filetes Primavera Limão	4.576304912567139
Gel de Banho Ultra Suave Limão	4.576304912567139
...	...
Limão	4.576304912567139

Since this is not the expected behaviour, it was decided to migrate the database from MySQL to Neo4j. The structure of the database was maintained as well as all the fields presented on tables *Product* and *ProductCategory*. The entries presented on table *Product* were migrated under the name *ProductFraunhofer* and the ones presented on table *ProductCategory* were inserted with the name *ProductCategoryFraunhofer*. Since Neo4j is not a relational database, but a graph database the relations presented between tables were converted to a new relation named "BROADER". This new relation exists between *ProductFraunhofer* and *ProductCategoryFraunhofer* and also between *ProductCategoryFraunhofer* nodes creating the same hierarchy structure that was presented on MySQL.

After the migration was complete it was created a full-text index, targeting the name of the product and using the *portuguese* analyzer. The full-text index was created using the following command:

```
1 CALL db.index.fulltext.createNodeIndex('product_index',['ProductFraunhofer'
    ], ['name'], {analyzer:'portuguese' })
```

Listing 3.6: Command used to create full-text index on products

The result of the same query that was performed above in MySQL was performed in Neo4j and the result is presented on table 3.2. As it is possible to see the results are more appealing compared to the ones obtained with MySQL full-text search, present on table 3.1. The documents with a higher score are "Lima" and "Limão" as expected.

Table 3.2: Quering "Limão" on a full text search index on Neo4j

Product Name	**Score**
Lima	5.6016845703125
Limão	5.6016845703125
Refrigerante com Gás Lima Limão	3.960988998413086
Refrigerante Lima-Limão	3.960988998413086
Iogurte Líquido Lima/Limão	3.960988998413086
Gelatina Pronta Lima/Limão	3.960988998413086
Pérola Lima/Limão	3.960988998413086
Concentrado Líquido Lima-limão	3.960988998413086
Sumo de Limão	3.5010528564453125
Rum Limão	3.5010528564453125
Drageias Limão	3.5010528564453125
Lima Mineral	3.5010528564453125
Tarteletes Limão	3.5010528564453125

The order and the improvement on the score were necessary, due to the fact that for a simple ingredient there are a considerable amount of products. Verify if each of those products is similar to a certain ingredient would be unfeasible. Therefore the list of products returned when searching for an ingredient is limited to a maximum of one hundred and fifty products.

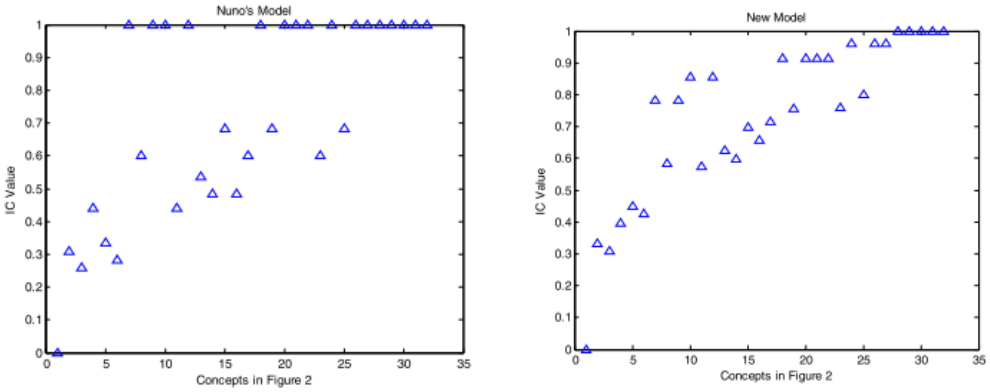
Semantic Similarities

There are four types of semantic similarity metrics: path-based, information content, feature based and hybrid.[Sli13]

Path-based metrics are the simplest ones, use the number of taxonomic links to calculate the similarity between concepts. There were implemented Leacock & Chodorow and Wu & Palmer metrics.

Information content-based metrics are based on the information that the nodes that are being compared have. There are two different ways that can be used to calculate: based on a corpus or based on the thesaurus or taxonomy[ZWG08b]. The first was proposed by Resnik [Res95], the second way was proposed by Nuno Seco et al. [SVH04] and by Zhou et al.[ZWG08a]. It was analyzed the second possibility, making this way use of a thesaurus to calculate IC. Both approaches for determining the information content are similar, in fact, the formula proposed by Nuno Seco (equation 2.10) appears on the approach of Zhou. However, the last one introduced the depth, as it is possible to see on equation 2.11.

By analyzing the figure 3.10 from a study made by Zhou [ZWG08a] . It is possible to see that despite Nuno's model have a higher concentration of concepts with $IC=1$ than Zhou's model, it emphasizes more the discrete nature of the concepts. The variance of the IC on Nuno's model is 0.0916 and Zhou's model is 0.0624. Despite this difference, Zhou's model shown to have a higher correlated coefficient.[ZWG08a]



(a) (b)

Figure 3.10: Distribution of concepts using Nuno's model (a) and Zhou's model (b)[ZWG08a]

For this reason, it was chosen Zhou's model to calculate the value of the information content of a concept with the variable that represents the distribution of the weight of both parts of the equation equal to 0.5. Due to the fact that is computationally heavy to calculate every time the IC of a concept, it was updated the node of each concept in order to insert a property with the IC value. The same was done with *depth* of the concept since it is a calculation that it is done recurrently. The measures based on information content implemented were the ones proposed by Resnik, Lin and Jian & Conrath.

The hybrid measure that was implemented was the one proposed by Zhou. It is an approach that used the information content principles as well as path based. The metric used to assess the information content value is the one that he proposed and as mentioned above is also used in information content measures.

After performing the semantic similarity, the products are filtered accordingly with a constant, that represents the minimum value of similarity for a product be considered similar to an ingredient.

3.6 Conclusion

This chapter details the development of the proposed system, making reference to the technologies, the methodologies, the architecture and the data sources used.

For a better understanding of the goal of the project, it is made an overview of the problems that it seeks to tackle. The first problem faced is how to extract structured information from an unstructured text, like a line of ingredient on a cooking recipe. Then when returning a list of products given an ingredient, other problems arise. Those problems came from the possible existence of synonyms, hyperonyms, hyponyms or even spelling mistakes. Furthermore, they can also come from the existence of noise in the products database and on the ingredient name.

The technologies were chosen based on the documentation that provides, the ease of use, the facility on working with external libraries and more important with the good support for NLP

tasks. Therefore the programming language chosen was Python. When analyzing the problem and based on the review of the state of the art it was chosen the Neo4j to support the information source.

Finally after analyzing the problems faced it was designed the architecture of the system with two components: information extraction and similarity matching. For the first component, it was used a CRF model to extract structured data. To train and to evaluate the model it was created and annotated a dataset extracted from a cooking recipes website. On the second component it is used semantic similarities measures that can be path based, feature based, information content based or hybrid to calculate the semantic relatedness between an ingredient and a list of products. The information source used to calculate those measures is AGROVOC, due to the fact that has Portuguese support and is cover the area of food.

The results of the system that was described in this chapter are shown in chapter 4.

Chapter 4

Testing and Validation

This chapter intends to describe the test and evaluation performed on the system. It is divided into four main categories: *Experiment Design*, *Testing Set Analysis*, *Results* and finally *Discussion*.

On the first section 4.1 it is made an analysis of the metrics used to access the overall performance of the components of the system and what they mean.

Then on section 4.2 it is made an analysis of the datasets used on the system in order to obtain some information about them. In order to be able to perform a better interpretation of the results.

Next section 4.3, presents the results obtained with the system on his two components using the measures mentioned on 4.1.

Finally, on section 4.4 it is made an analysis of the results obtained and discussed what had a positive and a negative influence on them.

4.1 Experiment Design

In order to properly test and validate the system each component was evaluated individually. For this evaluation it was used the *precision*, *recall* and *f1*.

The evaluation of the extraction of information component represents a multiclass classification. That's because the classification as more than 2 classes [Aly05], in this case 4 (*NAME*, *QUANTITY*, *UNIT* and *OTHER*). The results were obtained by macro averaging the values obtained on each class. The results obtained were compared using different datasets on training and tested with the same set of annotated line of ingredients. The test dataset was obtained by assembling the dataset created on this project and the one provided by Fraunhofer. From it, it was randomly selected 30% of the total lines. Since it is a multiclass classification problem it is also used the confusion matrix in order to have a better understanding of the performance of the model.

To evaluate the component where it is performed a match between ingredients and products, not only is used the metrics mentioned above to analyze the performance, but it is also calculated the *average precision*. *Precision* will access the number of correct products that are within the retrieved documents. *Recall* expresses the ratio between the number of correct results that were returned against the total number of correct products. The *average precision* measure was used

because the results are returned in a descendent similarity order and this way it is possible to evaluate taking into consideration also the position in which they are. These calculations are made for each query, which means for each ingredient query. For a better overview of the component it is calculated the *mean average precision*, as well as mean value of *precision*, *recall* and *f1*.

4.2 Testing Set Analysis

Information Extraction

In order to train and test the model above described it was necessary to create and annotate a data set that is above explained, on section 3.4, how it was proceeded. Besides the data set that was created, Fraunhofer Portugal AICOS also provided one with 34673 lines of annotated ingredients. The corresponding cooking recipes come from the website `https://chef.continente.pt/receitas`.

For a better comprehension of the data sets presented it was extracted some statistics. On table 4.1 is presented the average number of words in each ingredient line as well as the standard deviation. Despite the fact that the mean number of words don't have a significant variation between data sets, the standard deviation on the data set annotated on this project is noteworthy taking into consideration that the mean number of words is 4.56.

	Metrics	
Data set	Mean	Standard deviation
New annotated dataset	4.56	2.37
Fraunhofer Portugal AICOS	4.16	1.88

Table 4.1: Mean and standard deviation of the number of words per ingredient line.

It was also analyzed on both data sets the most common ingredient names presented on both. For this analysis, all words were converted to the singular form and with lower case, so that does not happen that for example "Azeite" and "azeite" are considered different ingredients. The results are presented on figure 4.1 and 4.2. From them, it is possible to verify that the main ingredients on them are similar, with some new entrances like "margarina" or "folha de louro".

By executing a more detailed analysis of both datasets it is possible to verify some differences in them. The dataset from Fraunhofer Portugal AICOS frequently abbreviates the unit of the ingredient line. Instead of using "colher" it has "c." or instead of "chávena" it has "cháv.". On the other hand, the new dataset annotated has cases where it tries to express the ingredient and the quantity in a more textual way, for example: "Pode ser também arroz branco", "O peso dos ovos em açúcar" or "A parte branca de um alho francês pequeno cortado em rodelas".

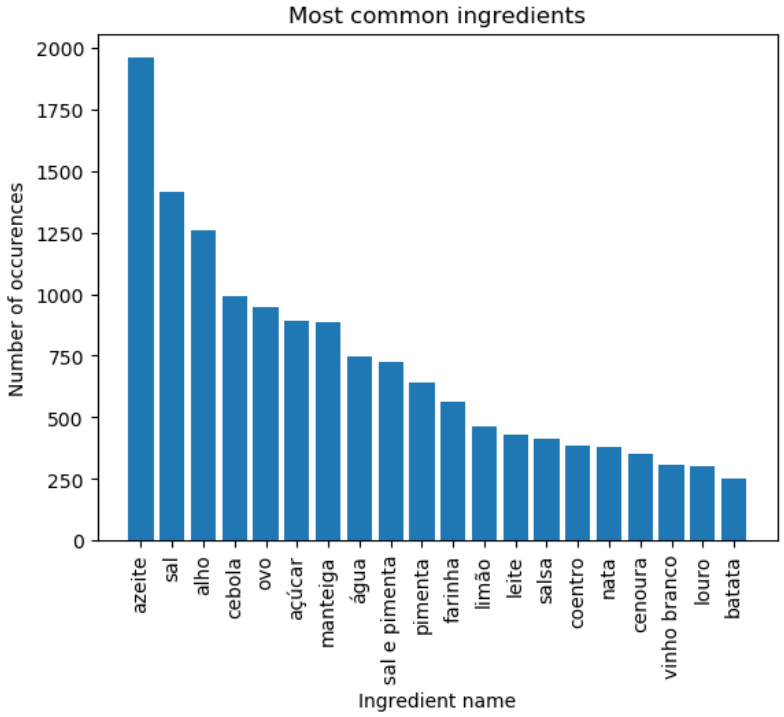


Figure 4.1: Analysis of the most common ingredients of the dataset provided by Fraunhofer Portugal AICOS

Similarity Matching

With the purpose of analyzing the performance of the similarity matching it was annotated the corresponding products from the provided database to the respective ingredient names. The selection of the name of the ingredients was completely random and came from the dataset annotated on this project. In order to try to reach as many different cases as possible, it wasn't performed any transformation to the names obtained.

To choose the products that correspond to a certain ingredient it was selected the ones that are more frequently associated by that name. For example, if the ingredient is "leite" then the respective products are "leite magro", "leite meio gordo" or similar to these ones. It was assumed that for expressing something more specific, the name of the ingredient would indicate so, like "leite de soja" or "leite para crianças".

Since there are multiple products with the same name from different brands, to ease the task it was used the *id* of the product to identify it.

On table 4.2 are presented some statistics relatively to the dataset.

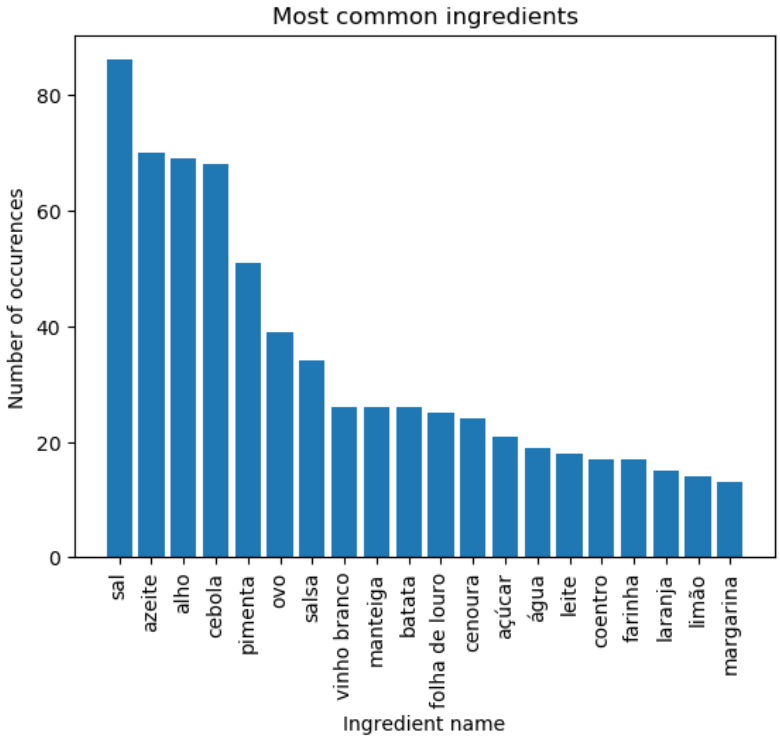


Figure 4.2: Analysis of the most common ingredients of the new annotated dataset

Number of ingredients	45
Coverage (new annotated dataset)	33.058%
Coverage (Fraunhofer Portugal AICOS annotated dataset)	27.143%

Table 4.2: Statistics of the annotated dataset relating products with ingredients

4.3 Results

The results of each component of the system are presented below. The datasets used on both components were analyzed on section 4.2.

Information Extraction

For a better analysis of the relevance of using a different dataset or a combination of them, written by different persons that can probably have a different type of writing and vocabulary, it was created 3 different cases, described below:

- **Case 1**
  - **Train** - The rest of the dataset annotated on this project that is not part of test
  - **Test** - 30% of the join of two datasets
- **Case 2**
  - **Train** - The rest of the Fraunhofer Portugal AICOS dataset that is not part of test

  - **Test** - 30% of the join of two datasets

- **Case 3**
  - **Train** - 70% of the join of two datasets
  - **Test** - 30% of the join of two datasets

Table 4.3, shows the *precision*, *recall* and *F-measure* of each of the cases aforementioned, to have an overview of the evaluation results.

	Metrics		
	Precision	Recall	F-measure
Case 1	0.92	0.93	0.93
Case 2	0.98	0.98	0.98
Case 3	0.98	0.98	0.98

Table 4.3: Evaluation metrics of the CRF model

For a more detailed analysis of the performance of the component on figures 4.3, 4.4 and 4.5 are shown the confusion matrices of each case.

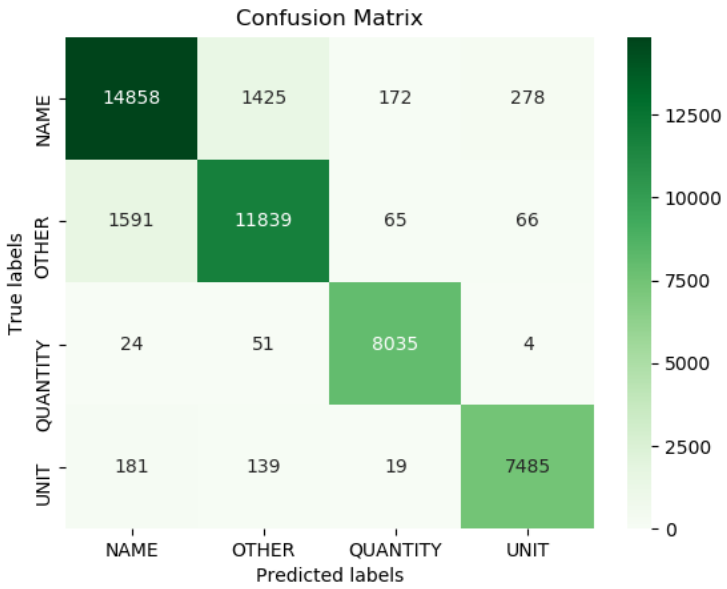


Figure 4.3: Confusion matrix referent to Case 1

Similarity Matching

The following results are obtaining by testing the system with the dataset aforementioned. The table 4.4 expresses the results without word embeddings and table 4.5 with word embedding . The values of the measures shown are the arithmetic mean of the measures on each query. The evaluation metrics for each ingredient query can be seen on appendix A.

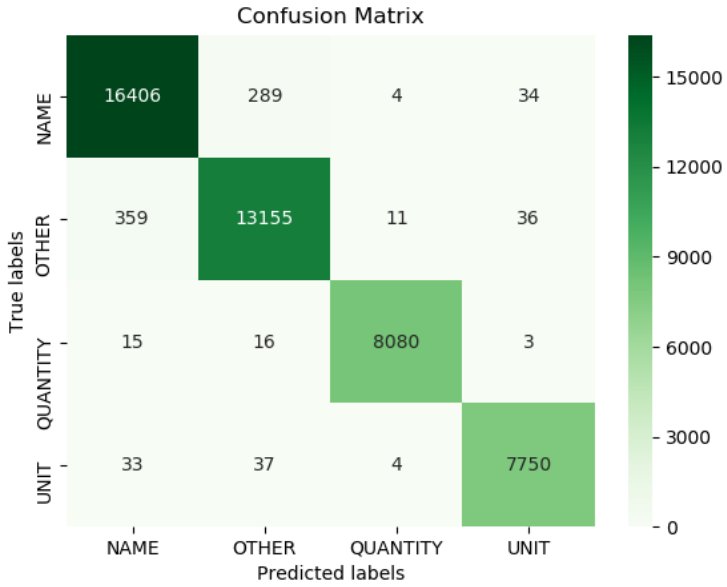


Figure 4.4: Confusion matrix referent to Case 2

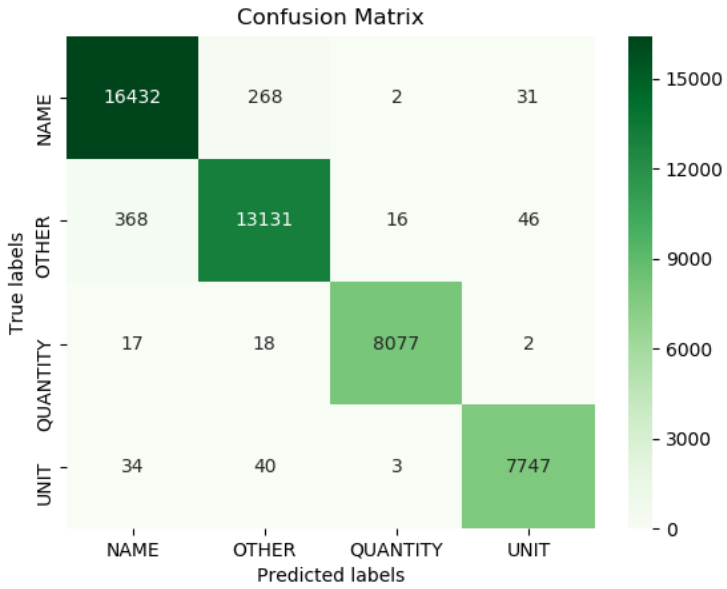


Figure 4.5: Confusion matrix referent to Case 3

		Metrics			
		Precision	Recall	Average Precision	F1
Path Based	Wu & Palmer	0.365	0.672	0.356	0.418
	Leacock & Chodorow	0.304	0.762	0.306	0.375
Information Content	Resnik	0.211	0.525	0.213	0.261
	Lin	0.239	0.567	0.248	0.293
	Jian & Conrath	0.199	0.468	0.217	0.241
Hybrid	Zhou	0.220	0.513	0.241	0.267

Table 4.4: Result obtained searching for products for an ingredient with semantic similarity measures.

		Metrics			
		Precision	Recall	Average Precision	F1
Path Based	Wu & Palmer	0.364	0.501	0.364	0.354
	Leacock & Chodorow	0.303	0.655	0.304	0.334
Information Content	Resnik	0.214	0.461	0.214	0.236
	Lin	0.243	0.485	0.252	0.263
	Jian & Conrath	0.216	0.410	0.238	0.228
Hybrid	Zhou	0.234	0.443	0.261	0.247

Table 4.5: Result obtained searching for products for an ingredient with semantic similarity measures and word embeddings.

4.4 Discussion

4.4.1 Information Extraction

The results obtained on this component is in line with what was expected. The combination of two datasets extracted from different sources, written by different persons with different vocabulary brings a higher performance to the CRF. This cannot be seen clearly through the metrics show on 4.3, however by analyzing figure 4.4 and 4.5 it is possible to see an increase of correct classifications on class *NAME*. The difference is not so noticeable because there is a huge difference in the size of both datasets. The one annotated on this thesis has 1343 entries and the one from Fraunhofer has 34673 entries, which means that the last one is 25.8 times bigger. Moreover, the component already has a *F1* of 0.98, making it harder to improve.

Looking to the confusion matrices it is possible to conclude that the model gets more confused when classifying *NAME* and *OTHER* class. This is explained because normally the *QUANTITY* class is an integer and next class is *UNIT*. After only rests *NAME* and *OTHER*, which can have different sizes. Taking into consideration that the normal sequence is *NAME* followed by *OTHER*, can be concluded that the CRF has difficulties determining the end of the ingredient name and the begin of the commentary. However, as aforementioned, it is normal since even for humans some cases are not easy to classify.

Even though the model got a good mark, it is considered that there is one aspect that is lowering the evaluation mark of the component, which is the annotation of the dataset. The two datasets

used for the extraction of information were not annotated by the same person. Despite there was some agreement on what was considered a *NAME*, a *UNIT*, a *QUANTITY* and *OTHER*, there was not a validation so that the annotations could be verified. Therefore it is not possible to assure that both datasets are annotated equally, which can result in lower accuracy. Furthermore, both datasets can have some classification mistakes.

In addition to the improvement mentioned above, it is believed that a change in the classes of the CRF would bring advantages to the next component. This change would be made by adding another class that would represent a variation of the product. For example "pão torrado" is still bread, however, has a variation that is toasted. Therefore, the idea would be classify "torrado" with a different class. In order to on the next step could be made better filtering of the products.

4.4.2 Similarity Matching

The results obtained on this component were reasonable. All the measures used obtained more or less the same results. However with some highlight to the measure Wu & Palmer, which is a path based measure. Obtain good results on *recall* it is easier when comparing to *precision* because it can be simply returned all the products that are analyzed. Therefore, even though it is important to have a good recall it is more important to have good precision and average precision. Since the goal is to return a list of products that are the most similar to that ingredient.

Comparing both experiments it is possible to see that it got better performance when using word2vec even though the difference is not significant when considering *precision* and *average precision*. However, without word embeddings, it obtains a higher *recall*.

As it is possible to see on appendix A, there are queries that obtain a good result in all the evaluation measures, others that do not do as good and some show that the system was not capable to return any correct result.

Moreover comparing the results obtained with and without word embeddings it is possible to see that in some cases it improved significantly and worst in others. Nevertheless, it can be noticed that in some cases where the correspondent node on AGROVOC was not so clear, it was able to correctly match. Take as example "camarão" that on AGROVOC is expressed with literal "Lagostim de água doce", which by performing a full-text search is not the best scored. Therefore, when looking for appendix A it can be seen that without word embeddings it got a *F1* equal to 0.0 and with got a *F1* of 0.358 and a *Precision* of 0.444.

Some of the noted problems that decrease the evaluation of the component are described below:

Database

The database used on the system containing the products is a real database provided by a retailer.

The database has 36 763 products, therefore when looking for a product many results appear. For instance, when looking for waters, using a full-text search it returns 307 results.

As a result, for each ingredient line, there will be a big set of calculations that are time-consuming. One of the calculations that are performed in almost all measures evaluated is calculating the shortest path between two nodes, using the Dijkstra algorithm. In order to the system returns an answer on a shorter period of time it was limited the number of products that are analyzed. This limitation obviously can decrease the recall, since it is possible to lose some products that were the correct.

Furthermore, after analyzing the database it is possible to verify that it as some unwanted noise. For a better understanding, the following examples can be taken into account:

- The database contains a product named "Limão", however after some inspection, it is possible to verify that it does not refer to the fruit but to a drink. The name of the product that refers to the lemon fruit is "Limão cal.2/3"
- When looking for "leite", multiple products appear with higher similarity to the query that does not refer to the common cow milk, but for example to special milk for kids on its different ages ("Leite 1", "Leite 2").

This leads to that when it performs the search for the best match of a node on the thesaurus to a product, it gets the wrong node and possibly the same node of the ingredient, even when using word embeddings. As it is possible to understand if that is the case, the level of similarity is great and not correspondent to reality. As a result, this leads to a lower performance of the component.

Lack of specification and synonyms on AGROVOC

AGROVOC is a multilingual thesaurus with 38637 Portuguese labels. Despite this huge amount of literals, it does not have a good variety of synonyms and it stills does not cover all ingredients. From the annotated dataset it cannot find "safio" and "bacon".

The thesaurus does not have enough specification in some cases. This lack of specification can result in a wrong result when performing the search for a node, giving a label that is the most similar but possibly the wrong one. For example, the thesaurus does not have the label "chocolate de leite", "chocolate branco" or "chocolate culinário". It only has the concept with the label "chocolate". Hence if looking for one of those literals it will try to take the best match textually, that will not be the correct one. It can be "chocolate", but it can also be another label related to milk for example. Inevitably, will result in a wrong calculation of the semantic similarity between nodes.

Another problem inherent to the knowledge resource used is that it has a low amount of synonyms that would enrich the vocabulary. For example, the literal "Ananás" does not exist. The literal used to refer to a pineapple is "Abacaxi".

This lack of specification of some concepts and synonyms contributes negatively to the accuracy of the similarity matching component.

Annotation

The dataset was annotated always with the thought of what would someone buy if wanted to buy a certain ingredient. Furthermore, the annotation was made by only take into consideration the name of the ingredient. They were not selected taking into consideration the recipe in which the ingredient was. Consequently, some cases were not clear what would be considered a relevant match.

Moreover, the annotation was not verified by someone else and therefore, it can exist products that are wrongly annotated.

Word embeddings

Even though the model was trained with 1 395 926 282 tokens, which is a meaningful amount of tokens, only Wikipedia is somehow related to food and nutrition. A training dataset with a corpus more related to food would probably increase the *precision* and *average precision* of the component.

It can be concluded that the similarity measures by their own are not sufficient to return as precise as possible the pretended products. It is necessary to add something like ranking algorithms on top of the similarity metrics in order to improve the performance.

4.5 Conclusion

Since the system was divided in two steps, it was made an evaluation to both separately and for the purpose were annotated datasets.

The experiments performed on the first step, information extraction, prove that a train with a combination of data sources improves the correct classification of classes. Even though the difference is not significant, it is possible to see on the confusion matrices some improvements. The reasons pointed out for the fact the difference is not more significant is the size of the datasets and the fact that they were annotated by different persons. The results show that the model is able to correctly extract information from an ingredient line.

The evaluation performed on the second step shows that the similarity measures do not return a list of products with high precision. Some of the problems that lower the values of the evaluation metrics are: the name of the products on the database have noise, the test dataset did not have a proper validation. Moreover, the AGROVOC has a lack of specification and the training dataset of the word2vec model were not specific to food or nutrition.

Overall, it is believed that the system is capable of retrieving a reasonable set of products given an ingredient line. However, some improvements can be made in order to improve an enrich the system, that will be addressed in section 5.1.

Chapter 5

Conclusion

This dissertation describes a system that has the goal of giving an online cooking recipe retrieve a list of products that are best suited for each ingredient line.

Alongside the document, it is possible to understand the methodology used to build the system. As well as, the reasons behind the choices made, regarding data sources and technologies.

The system created is divided into two steps: the first consists of extracting relevant information from the recipe, while the second is to match the necessary ingredients to prepare the recipe with products from a real retailer database.

Regarding the first part of the system, it can be concluded that a more wide vocabulary, more reach with synonyms and expressions improve the performance of the model. In respect to the second step, similarity matching, from the results, it can be concluded that the similarity metrics alone are not sufficient to retrieve results with high precision. Although, there are some reasons not related to the similarity measures that lower the evaluation values.

All in all, it is believed that the system can build a satisfactory shopping list. However, below are listed some enhancements that can be executed to improve the system.

5.1 Future Work

Analyzing some of the problems that can decrease the evaluation of the system and features that can improve the system, it was created the list below with improvements that can be done:

- **Validation system for annotations**

  As mentioned above, both annotated datasets were not validated by someone else. Therefore in order to create a model with more trustworthy datasets, it is paramount that the datasets have some validation. This validation can be done by another person by analyzing the annotated dataset and verifying if there is any error or by different persons annotate the same dataset and check the differences that they have.

- **Addition of synonyms and more specification**

  Like it was referenced on section 4.4.2, the thesaurus does not have enough specification. Since some ingredients do not exist on the thesaurus, when performing full-text search on it, it returns wrong nodes. Obviously, this represents a huge problem, since it can decrease the precision, recall of the system.

  In order to tackle this problem, the idea is to use the Portuguese wordnet to complement the AGROVOC. This would be done by performing a search of an AGROVOC literal on wordnet and include possible synonyms that it may have. And also by checking if it has literals that have the same "father" that is relevant to include.

  The idea is to bring more specification to AGROVOC in order to increase the performance of the system.

- **Learn to rank**

  The use of learning to rank is the application of machine learning to relevance ranking. Usually, this is done on top of a ranking layer. Which means after ranking a set of documents they are re-ranked before be returned. This is usually done on a different layer because it would be expensive, use the algorithm on thousands of results.

  Some of the algorithms used are PageRank, FastAP or CRR. The idea behind the use of these algorithms is to increase the values obtained for the evaluation metrics.

- **Product quantities**

  For a more complete system and also more interesting as a product, it would be paramount to include the quantities of the products on the grocery lists.

  In order to accomplish that, the idea is to convert every unit to a single one. For example, every unit related to weight would be converted to kilograms. For the units that do not own so obvious measures, it would be created rules for the conversion. That product capacity would also be converted to the same unit. These conversions would ease the task of finding the number of units of that product the person would need to do the cooking recipe.

References

[51I] 5.1. indexes - chapter 5. schema. https://neo4j.com/docs/cypher-manual/3.5/schema/index/#schema-index-fulltext-search. (Accessed on 06/12/2019).

[Abo] About | dbpedia. https://wiki.dbpedia.org/about. (Accessed on 06/04/2019).

[Aly05] Mohamed Aly. Survey on Multiclass Classification Methods. (November):1–9, 2005.

[AZ12] Charu C. Aggarwal and ChengXiang Zhai. A survey of text classification algorithms. In *Mining Text Data*, 2012.

[BLK+09] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia - A crystallization point for the Web of Data. *Journal of Web Semantics*, 7(3):154–165, 2009.

[CJB99] B Chandrasekaran, T R Johnson, and V Richard Benjamins. Ontologies: what are they? why do we need them? *IEEE Intelligent Systems and Their Applications*, 1999.

[CRM12] Caterina Caracciolo, Sachit Rajbahndari, and Ahsan Morshed. Thesaurus maintenance , alignment and publication as linked data : the AGROVOC use case Armando Stellato Gudrun Johannsen , Yves Jaques and Johannes Keizer. *Int. J. Metadata, Semantics and Ontologies*, 7(1):489–499, 2012.

[CW99] Claire Cardie and Kiri Wagstaff. Noun phrase coreference as clustering, 1999.

[Dee] Deep dive into neo4j 3.5 full text search. https://graphaware.com/neo4j/2019/01/11/neo4j-full-text-search-deep-dive.html. (Accessed on 06/12/2019).

[DGG+18] Damion M. Dooley, Emma J. Griffiths, Gurinder S. Gosal, Pier L. Buttigieg, Robert Hoehndorf, Matthew C. Lange, Lynn M. Schriml, Fiona S. L. Brinkman, and William W. L. Hsiao. FoodOn: a harmonized food ontology to increase global food traceability, quality control and data integration. *npj Science of Food*, 2(1), 2018.

[Gar04] Lars Marius Garshol. Metadata? Thesauri? Taxonomies? Topic maps! Making sense of it all. *Journal of Information Science*, 30(4):378–391, 2004.

[Gre] Erica Greene. Extracting structured data from recipes using conditional random fields - the new york times. https://open.blogs.nytimes.com/2015/04/09/

extracting-structured-data-from-recipes-using-conditional-random-fields/?_r=0&module=ArrowsNav&contentCollection=General&action=keypress&region=FixedLeft&pgtype=Blogs. (Accessed on 02/01/2019).

[HFS+17] Nathan Hartmann, Erick Fonseca, Christopher Shulby, Marcos Treviso, Jessica Rodrigues, and Sandra Aluisio. Portuguese Word Embeddings: Evaluating on Word Analogies and Natural Language Tasks. (Section 3), 2017.

[HG13] Thierry Hamon and Natalia Grabar. Extraction of ingredient names from recipes by combining linguistic annotations and crf selection. In *Proceedings of the 5th International Workshop on Multimedia for Cooking & Eating Activities*, CEA '13, pages 63–68, New York, NY, USA, 2013. ACM.

[HSO95] Graeme Hirst and David St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. 1995.

[HTBAB15] Mohamed Ali Hadj Taieb, Mohamed Ben Aouicha, and Yosra Bourouis. Fm3s: Features-based measure of sentences semantic similarity. volume 9121, pages 515–529, 06 2015.

[IBM] Ibm knowledge center - about text mining. https://www.ibm.com/support/knowledgecenter/en/SS3RA7_18.0.0/ta_guide_ddita/textmining/shared_entities/tm_intro_tm_defined.html. (Accessed on 30/01/2019).

[JJWC97] Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *Proceedings of the International Conference on Research in Computational Linguistics*, 10, 10 1997.

[Kon18] Natalia Konstantinova. Analysis of Images, Social Networks and Texts. 10716:15–28, 2018.

[LBM03] Yuhua Li, Z.A. Bandar, and D. McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–882, jul 2003.

[Lin98] Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[McC05] Andrew McCallum. Information extraction: Distilling Structured Data from Unstructured Text. *Queue - Social Computing*, 3(9):48–57, 2005.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. pages 1–12, 2013.

[Mil95] George A Miller. WordNet1.pdf. 38(11):39–41, 1995.

[MSYY12] Shinsuke Mori, Tetsuro Sasada, Yoko Yamakata, and Koichiro Yoshino. A Machine Learning Approach to Recipe Text Processing. *1st Cooking with Computer Workshop*, pages 29–34, 2012.

REFERENCES

[MyS] Mysql :: Mysql internals manual :: 10.7 full-text search. https://dev.mysql.com/doc/internals/en/full-text-search.html. (Accessed on 06/12/2019).

[Ped10] Ted Pedersen. Information content measures of semantic similarity perform better without sense-tagged text. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 329–332, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[POO16] Alexandre Pinto, Hugo Gonçalo Oliveira, and Ana Oliveira Alves. Comparing the Performance of Different NLP Toolkits in Formal and Social Media Text. *Slate*, (3):1–16, 2016.

[PY13] Jakub Piskorski and Roman Yangarber. Multi-source, Multilingual Information Extraction and Summarization. pages 23–50, 2013.

[RE03] M. Andrea Rodríguez and Max J. Egenhofer. Determining semantic similarity among entity classes from different ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 15(2):442–456, 2003.

[Res95] Philip Resnik. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. 1, 1995.

[RMBB89] Roy Rada, Hafedh Mili, Ellen Bicknell, and Maria Blettner. Development and application of a metric on semantic nets. *IEEE Trans. Systems, Man, and Cybernetics*, 19:17–30, 1989.

[RN10] Stuart Russell and Peter Norvig. *Artificial Intelligence A Modern Approach Third Edition*. 2010.

[SA+07] Barry Smith, , Michael Ashburner, Cornelius Rosse, Jonathan Bard, William Bug, Werner Ceusters, Louis J Goldberg, Karen Eilbeck, Amelia Ireland, Christopher J Mungall, Neocles Leontis, Philippe Rocca-Serra, Alan Ruttenberg, Susanna-Assunta Sansone, Richard H Scheuermann, Nigam Shah, Patricia L Whetzel, and Suzanna Lewis. The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature Biotechnology*, 25(11):1251–1255, November 2007.

[SA17] Awny Sayed and Amal Al Muqrishi. IBRI-CASONTO: Ontology-based semantic search engine. *Egyptian Informatics Journal*, 18(3):181–192, 2017.

[SBIV12] David Sánchez, Montserrat Batet, David Isern, and Aida Valls. Ontology-based semantic similarity: A new feature-based approach. *Expert Systems with Applications*, 39(9):7718–7728, 2012.

[SBYM04] Thabet Slimani, B Ben Yaghlane, and K Mellouli. A new similarity measure based on edge counting. 01 2004.

[Scr] Scrapy at a glance — scrapy 1.6.0 documentation. https://docs.scrapy.org/en/latest/intro/overview.html. (Accessed on 06/08/2019).

[SEGJ08] Souripriya Das, Eugene Inseok Chong, George Eadon, and Jagannathan Srinivasan. United States Patent USOO7328209B2, 2008.

REFERENCES

[Sil18] Nuno Gonçalo Neto Silva. Information extraction from unstructured recipe data. 2018.

[Sim] Gonçalo Simões. E-txt2db: Giving Structure to Unstructured Data (Extended Abstract). Technical report.

[Sim05] Luisa Simoes, Goncalo; Galhardas, Helena; Coheur. Information Extraction tasks : a survey. page 13, 2005.

[Sli13] Thabet Slimani. Description and evaluation of semantic similarity measures approaches. *International Journal of Computer Applications*, 80(10):25–33, oct 2013.

[Soc] Social media and the great recipe explosion: does more mean better? | food | the guardian. https://www.theguardian.com/lifeandstyle/2017/jun/18/great-recipe-explosion-social-media-does-more-mean-better-instagram-pinter (Accessed on 02/06/2019).

[SPT+12] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. Brat: A web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 102–107, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[SRF19] Nuno Silva, David Ribeiro, and Liliana Ferreira. Information extraction from unstructured recipe data. In *Proceedings of the 2019 5th International Conference on Computer and Technology Applications*, ICCTA 2019, pages 165–168, New York, NY, USA, 2019. ACM.

[SVH04] Nuno Seco, Tony Veale, and Jer Hayes. An intrinsic information content metric for semantic similarity in wordnet. In *Proceedings of the 16th European Conference on Artificial Intelligence*, ECAI'04, pages 1089–1090, Amsterdam, The Netherlands, The Netherlands, 2004. IOS Press.

[TAH14] Mohamed Ali Hadj Taieb, Mohamed Ben Aouicha, and Abdelmajid Ben Hamadou. Ontology-based approach for measuring semantic similarity. *Engineering Applications of Artificial Intelligence*, 36:238 – 261, 2014.

[Tve77] Amos Tversky. Features of similarity. - 1977 - Tversky.pdf. *Psychological Review*, 84(4):327–352, 1977.

[WD10] Daya C. Wimalasuriya and Dejing Dou. Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science*, 36(3):306–323, 2010.

[WP94] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

[ZWG08a] Zili Zhou, Yanna Wang, and Junzhong Gu. A new model of information content for semantic similarity in wordnet. *Proceedings of the 2008 2nd International Conference on Future Generation Communication and Networking, FGCN 2008*, 3:85–89, 2008.

[ZWG08b] Zili Zhou, Yanna Wang, and Junzhong Gu. New model of semantic similarity measuring in WordNet. In *Proceedings of 2008 3rd International Conference on Intelligent System and Knowledge Engineering, ISKE 2008*, pages 256–261, 2008.

REFERENCES

Appendix A

Similarity Matching Result

A.1 Wu & Palmer

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
leite	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
laranjas	0.027	0.25	0.027	0.049	0.02	0.25	0.02	0.037
camarão	0.444	0.3	0.444	0.358	0.0	0.0	0.0	0.0
cebola	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
alho francês	0.036	0.75	0.036	0.069	0.045	1.0	0.045	0.086
alho	0.145	1.0	0.145	0.253	0.136	1.0	0.136	0.239
presunto	0.595	0.647	0.595	0.62	0.579	0.971	0.579	0.725
fermento em pó	0.8	1.0	0.8	0.889	1.0	1.0	1.0	1.0
Molho de coentros	0.009	0.5	0.009	0.018	0.069	1.0	0.069	0.129
Azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
ovos	0.421	0.444	0.421	0.432	0.269	1.0	0.269	0.424
feijão manteiga	0.75	0.5	0.75	0.6	0.15	0.75	0.15	0.25
polvo	1.0	0.462	1.0	0.632	0.929	1.0	0.929	0.963
gelatina	0.859	0.509	0.859	0.639	0.835	0.889	0.835	0.861
louro	0.6	0.857	0.6	0.706	0.583	1.0	0.583	0.737
cerveja	0.831	0.397	0.831	0.537	0.901	0.801	0.901	0.848
Salsa	0.471	0.8	0.471	0.593	0.345	1.0	0.345	0.513
molho inglês	0.026	1.0	0.026	0.051	0.028	1.0	0.028	0.054

Similarity Matching Result

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
nata	0.308	0.148	0.308	0.2	0.606	0.741	0.606	0.667
caldo de galinha	0.31	1.0	0.31	0.473	0.26	1.0	0.26	0.413
lombo de porco	0.257	0.6	0.257	0.36	0.177	0.733	0.177	0.285
água	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Farinha	0.182	0.091	0.182	0.121	0.222	0.091	0.222	0.129
gemas de ovos	0.421	0.444	0.421	0.432	0.0	0.0	0.0	0.0
manteiga	0.125	0.02	0.125	0.034	0.5	0.612	0.5	0.55
tomate	0.2	0.667	0.2	0.308	0.228	1.0	0.228	0.371
mexilhão	1.0	0.857	1.0	0.923	1.0	1.0	1.0	1.0
chouriço de vinho	1.0	0.333	1.0	0.5	0.286	0.667	0.286	0.4
sal	0.27	0.667	0.27	0.384	0.155	0.6	0.155	0.246
cebolas	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
Pimenta	0.25	0.045	0.25	0.076	0.156	0.227	0.156	0.185
carne de vitela	0.167	0.667	0.167	0.267	0.364	0.444	0.364	0.4
azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
Chocolate	0.114	0.519	0.114	0.187	0.168	0.741	0.168	0.274
chocolate para culinária	0.0	0.0	0.0	0.0	0.017	0.105	0.017	0.029
fiambre de peru	0.036	0.286	0.036	0.064	0.077	0.143	0.077	0.1
Noz-moscada	0.857	0.857	0.857	0.857	1.0	0.857	1.0	0.923
açúcar baunilhado	0.333	0.667	0.333	0.444	0.087	0.667	0.087	0.154
Colorau	0.25	1.0	0.25	0.4	0.778	1.0	0.778	0.875
batata	0.444	0.286	0.444	0.348	0.222	0.714	0.222	0.339
broa de milho	0.111	0.167	0.111	0.133	0.15	0.5	0.15	0.231
Gelado de frutos dos bosques	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
nabos	0.0	0.0	0.0	0.0	0.625	1.0	0.625	0.769

A.2 Leacock & Chodorow

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
leite	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
laranjas	0.027	0.25	0.027	0.049	0.02	0.25	0.02	0.037
camarão	0.444	0.3	0.444	0.358	0.0	0.0	0.0	0.0
cebola	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
alho francês	0.036	0.75	0.036	0.069	0.045	1.0	0.045	0.086
alho	0.145	1.0	0.145	0.253	0.136	1.0	0.136	0.239
presunto	0.595	0.647	0.595	0.62	0.579	0.971	0.579	0.725
fermento em pó	0.8	1.0	0.8	0.889	1.0	1.0	1.0	1.0
Molho de coentros	0.009	0.5	0.009	0.018	0.069	1.0	0.069	0.129
Azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
ovos	0.421	0.444	0.421	0.432	0.269	1.0	0.269	0.424
feijão manteiga	0.75	0.5	0.75	0.6	0.15	0.75	0.15	0.25
polvo	1.0	0.462	1.0	0.632	0.929	1.0	0.929	0.963
gelatina	0.859	0.509	0.859	0.639	0.835	0.889	0.835	0.861
louro	0.6	0.857	0.6	0.706	0.583	1.0	0.583	0.737
cerveja	0.831	0.397	0.831	0.537	0.901	0.801	0.901	0.848
Salsa	0.471	0.8	0.471	0.593	0.345	1.0	0.345	0.513
molho inglês	0.026	1.0	0.026	0.051	0.028	1.0	0.028	0.054
nata	0.308	0.148	0.308	0.2	0.606	0.741	0.606	0.667
caldo de galinha	0.31	1.0	0.31	0.473	0.26	1.0	0.26	0.413
lombo de porco	0.257	0.6	0.257	0.36	0.177	0.733	0.177	0.285
água	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Farinha	0.182	0.091	0.182	0.121	0.222	0.091	0.222	0.129
gemas de ovos	0.421	0.444	0.421	0.432	0.0	0.0	0.0	0.0
manteiga	0.125	0.02	0.125	0.034	0.5	0.612	0.5	0.55
tomate	0.2	0.667	0.2	0.308	0.228	1.0	0.228	0.371
mexilhão	1.0	0.857	1.0	0.923	1.0	1.0	1.0	1.0
chouriço de vinho	1.0	0.333	1.0	0.5	0.286	0.667	0.286	0.4
sal	0.27	0.667	0.27	0.384	0.155	0.6	0.155	0.246
cebolas	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
Pimenta	0.25	0.045	0.25	0.076	0.156	0.227	0.156	0.185
carne de vitela	0.167	0.667	0.167	0.267	0.364	0.444	0.364	0.4
azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
Chocolate	0.114	0.519	0.114	0.187	0.168	0.741	0.168	0.274
chocolate para culinária	0.0	0.0	0.0	0.0	0.017	0.105	0.017	0.029
fiambre de peru	0.036	0.286	0.036	0.064	0.077	0.143	0.077	0.1
Noz-moscada	0.857	0.857	0.857	0.857	1.0	0.857	1.0	0.923
açúcar baunilhado	0.333	0.667	0.333	0.444	0.087	0.667	0.087	0.154
Colorau	0.25	1.0	0.25	0.4	0.778	1.0	0.778	0.875
batata	0.444	0.286	0.444	0.348	0.222	0.714	0.222	0.339
broa de milho	0.111	0.167	0.111	0.133	0.15	0.5	0.15	0.231
Gelado de frutos dos bosques	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
nabos	0.0	0.0	0.0	0.0	0.625	1.0	0.625	0.769

A.3 Resnik

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
leite	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
laranjas	0.027	0.25	0.027	0.049	0.02	0.25	0.02	0.037
camarão	0.444	0.3	0.444	0.358	0.0	0.0	0.0	0.0
cebola	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
alho francês	0.036	0.75	0.036	0.069	0.045	1.0	0.045	0.086
alho	0.145	1.0	0.145	0.253	0.136	1.0	0.136	0.239
presunto	0.595	0.647	0.595	0.62	0.579	0.971	0.579	0.725
fermento em pó	0.8	1.0	0.8	0.889	1.0	1.0	1.0	1.0
Molho de coentros	0.009	0.5	0.009	0.018	0.069	1.0	0.069	0.129
Azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
ovos	0.421	0.444	0.421	0.432	0.269	1.0	0.269	0.424
feijão manteiga	0.75	0.5	0.75	0.6	0.15	0.75	0.15	0.25
polvo	1.0	0.462	1.0	0.632	0.929	1.0	0.929	0.963
gelatina	0.859	0.509	0.859	0.639	0.835	0.889	0.835	0.861
louro	0.6	0.857	0.6	0.706	0.583	1.0	0.583	0.737
cerveja	0.831	0.397	0.831	0.537	0.901	0.801	0.901	0.848

Similarity Matching Result

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
Salsa	0.471	0.8	0.471	0.593	0.345	1.0	0.345	0.513
molho inglês	0.026	1.0	0.026	0.051	0.028	1.0	0.028	0.054
nata	0.308	0.148	0.308	0.2	0.606	0.741	0.606	0.667
caldo de galinha	0.31	1.0	0.31	0.473	0.26	1.0	0.26	0.413
lombo de porco	0.257	0.6	0.257	0.36	0.177	0.733	0.177	0.285
água	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Farinha	0.182	0.091	0.182	0.121	0.222	0.091	0.222	0.129
gemas de ovos	0.421	0.444	0.421	0.432	0.0	0.0	0.0	0.0
manteiga	0.125	0.02	0.125	0.034	0.5	0.612	0.5	0.55
tomate	0.2	0.667	0.2	0.308	0.228	1.0	0.228	0.371
mexilhão	1.0	0.857	1.0	0.923	1.0	1.0	1.0	1.0
chouriço de vinho	1.0	0.333	1.0	0.5	0.286	0.667	0.286	0.4
sal	0.27	0.667	0.27	0.384	0.155	0.6	0.155	0.246
cebolas	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
Pimenta	0.25	0.045	0.25	0.076	0.156	0.227	0.156	0.185
carne de vitela	0.167	0.667	0.167	0.267	0.364	0.444	0.364	0.4
azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
Chocolate	0.114	0.519	0.114	0.187	0.168	0.741	0.168	0.274
chocolate para culinária	0.0	0.0	0.0	0.0	0.017	0.105	0.017	0.029
fiambre de peru	0.036	0.286	0.036	0.064	0.077	0.143	0.077	0.1
Noz-moscada	0.857	0.857	0.857	0.857	1.0	0.857	1.0	0.923
açúcar baunilhado	0.333	0.667	0.333	0.444	0.087	0.667	0.087	0.154
Colorau	0.25	1.0	0.25	0.4	0.778	1.0	0.778	0.875
batata	0.444	0.286	0.444	0.348	0.222	0.714	0.222	0.339
broa de milho	0.111	0.167	0.111	0.133	0.15	0.5	0.15	0.231
Gelado de frutos dos bosques	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
nabos	0.0	0.0	0.0	0.0	0.625	1.0	0.625	0.769

A.4 Lin

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
leite	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
laranjas	0.027	0.25	0.027	0.049	0.02	0.25	0.02	0.037
camarão	0.444	0.3	0.444	0.358	0.0	0.0	0.0	0.0
cebola	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
alho francês	0.036	0.75	0.036	0.069	0.045	1.0	0.045	0.086
alho	0.145	1.0	0.145	0.253	0.136	1.0	0.136	0.239
presunto	0.595	0.647	0.595	0.62	0.579	0.971	0.579	0.725
fermento em pó	0.8	1.0	0.8	0.889	1.0	1.0	1.0	1.0
Molho de coentros	0.009	0.5	0.009	0.018	0.069	1.0	0.069	0.129
Azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
ovos	0.421	0.444	0.421	0.432	0.269	1.0	0.269	0.424
feijão manteiga	0.75	0.5	0.75	0.6	0.15	0.75	0.15	0.25
polvo	1.0	0.462	1.0	0.632	0.929	1.0	0.929	0.963
gelatina	0.859	0.509	0.859	0.639	0.835	0.889	0.835	0.861
louro	0.6	0.857	0.6	0.706	0.583	1.0	0.583	0.737
cerveja	0.831	0.397	0.831	0.537	0.901	0.801	0.901	0.848
Salsa	0.471	0.8	0.471	0.593	0.345	1.0	0.345	0.513
molho inglês	0.026	1.0	0.026	0.051	0.028	1.0	0.028	0.054
nata	0.308	0.148	0.308	0.2	0.606	0.741	0.606	0.667
caldo de galinha	0.31	1.0	0.31	0.473	0.26	1.0	0.26	0.413
lombo de porco	0.257	0.6	0.257	0.36	0.177	0.733	0.177	0.285
água	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Farinha	0.182	0.091	0.182	0.121	0.222	0.091	0.222	0.129
gemas de ovos	0.421	0.444	0.421	0.432	0.0	0.0	0.0	0.0
manteiga	0.125	0.02	0.125	0.034	0.5	0.612	0.5	0.55
tomate	0.2	0.667	0.2	0.308	0.228	1.0	0.228	0.371
mexilhão	1.0	0.857	1.0	0.923	1.0	1.0	1.0	1.0
chouriço de vinho	1.0	0.333	1.0	0.5	0.286	0.667	0.286	0.4
sal	0.27	0.667	0.27	0.384	0.155	0.6	0.155	0.246
cebolas	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
Pimenta	0.25	0.045	0.25	0.076	0.156	0.227	0.156	0.185
carne de vitela	0.167	0.667	0.167	0.267	0.364	0.444	0.364	0.4
azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
Chocolate	0.114	0.519	0.114	0.187	0.168	0.741	0.168	0.274

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
chocolate para culinária	0.0	0.0	0.0	0.0	0.017	0.105	0.017	0.029
fiambre de peru	0.036	0.286	0.036	0.064	0.077	0.143	0.077	0.1
Noz-moscada	0.857	0.857	0.857	0.857	1.0	0.857	1.0	0.923
açúcar baunilhado	0.333	0.667	0.333	0.444	0.087	0.667	0.087	0.154
Colorau	0.25	1.0	0.25	0.4	0.778	1.0	0.778	0.875
batata	0.444	0.286	0.444	0.348	0.222	0.714	0.222	0.339
broa de milho	0.111	0.167	0.111	0.133	0.15	0.5	0.15	0.231
Gelado de frutos dos bosques	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
nabos	0.0	0.0	0.0	0.0	0.625	1.0	0.625	0.769

A.5 Jian & Conrath

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
leite	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
laranjas	0.027	0.25	0.027	0.049	0.02	0.25	0.02	0.037
camarão	0.444	0.3	0.444	0.358	0.0	0.0	0.0	0.0
cebola	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
alho francês	0.036	0.75	0.036	0.069	0.045	1.0	0.045	0.086
alho	0.145	1.0	0.145	0.253	0.136	1.0	0.136	0.239
presunto	0.595	0.647	0.595	0.62	0.579	0.971	0.579	0.725
fermento em pó	0.8	1.0	0.8	0.889	1.0	1.0	1.0	1.0
Molho de coentros	0.009	0.5	0.009	0.018	0.069	1.0	0.069	0.129
Azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
ovos	0.421	0.444	0.421	0.432	0.269	1.0	0.269	0.424
feijão manteiga	0.75	0.5	0.75	0.6	0.15	0.75	0.15	0.25
polvo	1.0	0.462	1.0	0.632	0.929	1.0	0.929	0.963
gelatina	0.859	0.509	0.859	0.639	0.835	0.889	0.835	0.861
louro	0.6	0.857	0.6	0.706	0.583	1.0	0.583	0.737
cerveja	0.831	0.397	0.831	0.537	0.901	0.801	0.901	0.848
Salsa	0.471	0.8	0.471	0.593	0.345	1.0	0.345	0.513
molho inglês	0.026	1.0	0.026	0.051	0.028	1.0	0.028	0.054
nata	0.308	0.148	0.308	0.2	0.606	0.741	0.606	0.667

Similarity Matching Result

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
caldo de galinha	0.31	1.0	0.31	0.473	0.26	1.0	0.26	0.413
lombo de porco	0.257	0.6	0.257	0.36	0.177	0.733	0.177	0.285
água	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Farinha	0.182	0.091	0.182	0.121	0.222	0.091	0.222	0.129
gemas de ovos	0.421	0.444	0.421	0.432	0.0	0.0	0.0	0.0
manteiga	0.125	0.02	0.125	0.034	0.5	0.612	0.5	0.55
tomate	0.2	0.667	0.2	0.308	0.228	1.0	0.228	0.371
mexilhão	1.0	0.857	1.0	0.923	1.0	1.0	1.0	1.0
chouriço de vinho	1.0	0.333	1.0	0.5	0.286	0.667	0.286	0.4
sal	0.27	0.667	0.27	0.384	0.155	0.6	0.155	0.246
cebolas	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
Pimenta	0.25	0.045	0.25	0.076	0.156	0.227	0.156	0.185
carne de vitela	0.167	0.667	0.167	0.267	0.364	0.444	0.364	0.4
azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
Chocolate	0.114	0.519	0.114	0.187	0.168	0.741	0.168	0.274
chocolate para culinária	0.0	0.0	0.0	0.0	0.017	0.105	0.017	0.029
fiambre de peru	0.036	0.286	0.036	0.064	0.077	0.143	0.077	0.1
Noz-moscada	0.857	0.857	0.857	0.857	1.0	0.857	1.0	0.923
açúcar baunilhado	0.333	0.667	0.333	0.444	0.087	0.667	0.087	0.154
Colorau	0.25	1.0	0.25	0.4	0.778	1.0	0.778	0.875
batata	0.444	0.286	0.444	0.348	0.222	0.714	0.222	0.339
broa de milho	0.111	0.167	0.111	0.133	0.15	0.5	0.15	0.231
Gelado de frutos dos bosques	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
nabos	0.0	0.0	0.0	0.0	0.625	1.0	0.625	0.769

A.6 Zhou

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
leite	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
laranjas	0.027	0.25	0.027	0.049	0.02	0.25	0.02	0.037
camarão	0.444	0.3	0.444	0.358	0.0	0.0	0.0	0.0
cebola	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
alho francês	0.036	0.75	0.036	0.069	0.045	1.0	0.045	0.086
alho	0.145	1.0	0.145	0.253	0.136	1.0	0.136	0.239
presunto	0.595	0.647	0.595	0.62	0.579	0.971	0.579	0.725
fermento em pó	0.8	1.0	0.8	0.889	1.0	1.0	1.0	1.0
Molho de coentros	0.009	0.5	0.009	0.018	0.069	1.0	0.069	0.129
Azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
ovos	0.421	0.444	0.421	0.432	0.269	1.0	0.269	0.424
feijão manteiga	0.75	0.5	0.75	0.6	0.15	0.75	0.15	0.25
polvo	1.0	0.462	1.0	0.632	0.929	1.0	0.929	0.963
gelatina	0.859	0.509	0.859	0.639	0.835	0.889	0.835	0.861
louro	0.6	0.857	0.6	0.706	0.583	1.0	0.583	0.737
cerveja	0.831	0.397	0.831	0.537	0.901	0.801	0.901	0.848
Salsa	0.471	0.8	0.471	0.593	0.345	1.0	0.345	0.513
molho inglês	0.026	1.0	0.026	0.051	0.028	1.0	0.028	0.054
nata	0.308	0.148	0.308	0.2	0.606	0.741	0.606	0.667
caldo de galinha	0.31	1.0	0.31	0.473	0.26	1.0	0.26	0.413
lombo de porco	0.257	0.6	0.257	0.36	0.177	0.733	0.177	0.285
água	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Farinha	0.182	0.091	0.182	0.121	0.222	0.091	0.222	0.129
gemas de ovos	0.421	0.444	0.421	0.432	0.0	0.0	0.0	0.0
manteiga	0.125	0.02	0.125	0.034	0.5	0.612	0.5	0.55
tomate	0.2	0.667	0.2	0.308	0.228	1.0	0.228	0.371
mexilhão	1.0	0.857	1.0	0.923	1.0	1.0	1.0	1.0
chouriço de vinho	1.0	0.333	1.0	0.5	0.286	0.667	0.286	0.4
sal	0.27	0.667	0.27	0.384	0.155	0.6	0.155	0.246
cebolas	0.257	0.562	0.257	0.353	0.5	0.875	0.5	0.636
Pimenta	0.25	0.045	0.25	0.076	0.156	0.227	0.156	0.185
carne de vitela	0.167	0.667	0.167	0.267	0.364	0.444	0.364	0.4

Similarity Matching Result

	Metrics							
	With word embeddings				Without word embeddings			
Ingredient Name	Precision	Recall	AP	F1	Precision	Recall	AP	F1
azeite	0.748	0.832	0.748	0.788	0.664	0.832	0.664	0.739
Chocolate	0.114	0.519	0.114	0.187	0.168	0.741	0.168	0.274
chocolate para culinária	0.0	0.0	0.0	0.0	0.017	0.105	0.017	0.029
fiambre de peru	0.036	0.286	0.036	0.064	0.077	0.143	0.077	0.1
Noz-moscada	0.857	0.857	0.857	0.857	1.0	0.857	1.0	0.923
açúcar baunilhado	0.333	0.667	0.333	0.444	0.087	0.667	0.087	0.154
Colorau	0.25	1.0	0.25	0.4	0.778	1.0	0.778	0.875
batata	0.444	0.286	0.444	0.348	0.222	0.714	0.222	0.339
broa de milho	0.111	0.167	0.111	0.133	0.15	0.5	0.15	0.231
Gelado de frutos dos bosques	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
nabos	0.0	0.0	0.0	0.0	0.625	1.0	0.625	0.769