FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Information Extraction from Unstructured Recipe Data

Nuno Gonçalo Neto Silva

U. PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Liliana Ferreira

July 10, 2018

Information Extraction from Unstructured Recipe Data

Nuno Gonçalo Neto Silva

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. João Moreira, PhD.

External Examiner: Prof. Mário Rodrigues, PhD.

Supervisor: Prof. Liliana Ferreira, PhD.

July 10, 2018

Abstract

Food recipes are an essential part of the lives of many individuals, who use these as a source of information for learning how to cook new dishes or as an aid for their food choice. Multiple websites on the Internet offer thousands of food recipes submitted by its users, which contain fields of structured information such as the name, ingredients, and directions, that allow for its users to filter through them according to their personal needs.

However, the structured information available for these recipes is often missing relevant information for its users, which can include the nutritional values of the recipe, cooking utensils required or each ingredient's applied cooking method. This information is often present in the recipe, albeit in an unstructured form. Finding a way to automatically retrieve and structure this information would allow for more fine-tuned searching and to improve the recommendation systems used by websites that offer food recipes. Being able to accurately determine a recipe's nutritional values using the extracted information could also help bring further clarity to the recipe's users on its nutritional content and overall effect on health.

In order to solve this problem, a system was developed that accomplishes the following goals: The extraction of the name, quantity, units, applied cooking method and food preparation techniques of each ingredient in a food recipe; The extraction of the used cooking utensils in a recipe; The calculation of the nutritional values of a recipe, using the aforementioned extracted information in conjunction with a food composition database hosted at Fraunhofer Portugal AICOS[1].

A set of 100 annotated recipes was used for testing. For the extraction of cooking actions and utensils used in a food recipe, the system achieved an average F-measure of 0.89. For the association of ingredients to their applied cooking method and food preparation techniques, the system achieved an average F-measure of 0.84. Additionally, for twenty ingredients with validated extracted information, the system was able to correctly associate eight ingredients to their database entries using the extracted information, an improvement over the three correct associations achieved by the baseline used.

The results suggest the system can reliably extract relevant information and associations in food recipes. They also imply it is possible to determine more accurate nutritional information for each ingredient through the use of additional structured information.

[1] https://www.fraunhofer.pt/en/fraunhofer_aicos/about_us.html

Resumo

Atualmente, receitas de cozinha são uma parte essencial do quotidiano de muitos indivíduos, que as utilizam para aprender a cozinhar novos pratos ou para os ajudar a escolher o que cozinhar no seu dia-a-dia. Existem vários *websites* que disponibilizam milhares de receitas de cozinha submetidas pelos seus utilizadores, receitas estas que contêm campos de informação estruturada como o título, ingredientes, e passos, de forma a permitir aos seus utilizadores filtrar estas receitas de acordo com as suas preferências.

No entanto, são vários os casos em que falta a esta informação estruturada outros campos considerados relevantes pelos utilizadores, campos estes que podem incluir os valores nutricionais de uma receita, os utensílios de cozinha necessários para a sua confeção ou o método de cozinha aplicado a cada ingrediente. Esta informação, geralmente, encontra-se presente na receita, de forma não-estruturada. Como tal, desenvolver um sistema capaz de extrair e estruturar esta informação permitiria refinar os sistemas de pesquisa e informação presentes em *websites* que oferecem receitas de cozinha. A explicitação dos valores nutricionais relativos à receita através do uso desta informação adicional traria, também, uma maior claridade aos utilizadores destes *websites* sobre o conteúdo nutricional e impacto na saúde destas.

De forma a lidar com estes casos, foi desenvolvido um sistema que cumpre os seguintes objetivos: A extração do nome, quantidade, unidades, método de cozinha e técnicas de preparação de alimentos para cada ingrediente da receita de cozinha; A extração dos utensílios de cozinha necessários para a receita; O cálculo dos valores nutricionais da receita através do uso da informação extraída em conjunto com uma base de dados de composição alimentar alojada nos servidores da Fraunhofer Portugal AICOS[2].

Um conjunto de 100 receitas de cozinha anotadas foi utilizado para testar o sistema. Para as componentes de extração de ações e utensílios de cozinha, o sistema alcançou uma *F-measure* média de 0.89. Para as componentes de associação de ingredientes ao método de cozinha aplicado e técnicas de preparação de alimentos utilizadas, o sistema alcançou uma *F-measure* média de 0.84. Para um conjunto de vinte ingredientes cuja informação extraída foi previamente validada, o sistema associou corretamente oito ingredientes às suas respetivas entradas na base de dados, uma melhoria relativamente às três associações corretas alcançadas pela linha de base.

Os resultados sugerem que o sistema é capaz de extrair informação relevante de receitas de cozinha de uma forma fiável. Estes também implicam que é possível determinar informação nutricional mais precisa para cada ingrediente através do uso de informação adicional estruturada.

[2] https://www.fraunhofer.pt/en/fraunhofer_aicos/about_us.html

Acknowledgements

First and foremost, I would like to thank my advisors at Fraunhofer Portugal AICOS, David Ribeiro and Liliana Ferreira, for their help, patience and guidance throughout the development of this dissertation.

I would also like to thank my colleagues and friends Catarina, Cristiana, Inês and João, for all their help and support throughout the year, and for making working at Fraunhofer Portugal AICOS a much more fun experience than I could have ever imagined.

Finally, I would like to express my deep gratitude to all my family and friends, for providing me with the love, support and motivation in what has been a period of intense learning, on both an academic and personal level. I could not have possibly done this without you, not by a long shot. I love you all.

Nuno Gonçalo Neto Silva

"Wisdom's a gift, but you'd trade it for youth"

Ezra Koenig

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

API	Application Programming Interface
CRF	Conditional Random Field
HMM	Hidden Markov Model
MEMM	Maximum Entropy Markov Model
NE	Named Entity
NER	Named Entity Recognition
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
SVM	Support Vector Machine

Chapter 1

Introduction

In this chapter, the **context and the motivation** for the development of this project is presented. The **problems** approached in this dissertation as well the delineation of the **goals** expected to be achieved are also provided, alongside a brief overview of the **dissertation's structure**.

1.1 Context and Motivation

An essential aspect of today's cooking world are food recipes, a set of instructions that describe how to prepare a culinary dish. Food recipes are generally used by individuals who are interested in learning how to cook a certain dish, or merely as an aid for the individual's food choice for a given meal. While traditionally made available in culinary books, websites offering food recipes have been steadily growing in popularity over the past two decades, making available thousands of recipes submitted by its users to the World Wide Web [Thr11].

In order to allow for the users of food recipe websites to search and filter through the recipes available, the recipes submitted to these websites contain fields of structured information such as the name, ingredients and preparation steps fields. However, the amount of structured information contained in these recipes is often not enough to completely fulfill a user's information need, lacking relevant information such as the nutritional values of the recipe, kitchen utensils necessary to prepare the dish or the applied cooking method to each ingredient of the recipe. This information is usually present in the recipe in an unstructured fashion, either in the recipe's ingredient descriptions, instructions or even its title.

The identification and structuring of additional unstructured information in a recipe would allow for food recipe websites to offer more detailed search filters and further improve its recommendation systems through the use of the additional structured fields. This would let, for example, a user filter recipes through the cooking utensils used in these (e.g. search for every recipe on the website that doesn't require the use of a blender). It would also allow for a website's recommendation system to recommend recipes based on the similarity of the recipe's additional structured information, on top of the already present information (e.g. a recommendation system could recommend to a user a recipe that uses similar cooking methods to a recipe the user previously clicked on).

Determining a recipe's nutritional values automatically through the structuring of the information pertaining to the recipe's ingredients alongside the use of a food composition database hosted would allow for a user to ascertain a recipe's nutritional information (e.g. calories, amount of sodium) without having to find this information by him or herself. This would create an additional layer of transparency pertaining a recipe's nutritional content, which could help prevent an individual's nutritional choices from having a negative impact on his or her health. According to the World Health Organization, nutrition is the cornerstone of good health, and an unhealthy diet resulting from poor nutritional choices can have a negative impact on mental health and cause severe chronicle diseases [Org].

These two main problems fall into the nutrition area, since they both pertain to information regarding food and nutrition. The nutrition area is one of several health related areas where Fraunhofer Portugal AICOS, the research center where this dissertation was elaborated, develops applied research projects on. The system developed for this dissertation is part of a larger project being developed, at the time of writing, at the research center, which aims to extract information in food recipes in order to build a personalized recommendation and meal-planning application.

1.2 Problems and Goals

The purpose of this dissertation was the development of a system capable of solving two distinct problems - the identification and structuring of relevant unstructured information in a food recipe, and the calculation of a recipe's nutritional values using the extracted information. Each of these problems required a different approach to reach a viable solution and, as such, had different goals associated to them.

For the problem of identifying and structuring relevant unstructured information in a food recipe, the following goals were delineated in order to achieve a solution for the problem at the end of the development phase:

- Parse and identify segments and expressions in a recipe's title, ingredients list and preparation steps through the use of text segmentation techniques;

- Classify the newly identified segments and expressions according to the information they provide, through the use of segment classification techniques;

- Identify associations between the classified segments in order to extract additional valuable information from the recipe, through the use of techniques relative to the association of segments;

- Structure the newly obtained information in such a way that it's possible to use the information in order to improve search and recommendation systems in food recipe websites, and to aid in the calculation of the nutritional values of the recipe;

For the problem of identifying and calculating the nutritional information in a culinary dish, the following goals were delineated in order to achieve a solution for the problem at the end of the development phase:

- Standardize units pertaining to the ingredients, through the use of techniques of normalization;
- Through cross-referencing the identified ingredients and respective proportions with the food composition database hosted at Fraunhofer Portugal AICOS, taking into consideration as well the information extracted relative to the ingredients, determine the nutritional values associated to each of the ingredients and, consequently, the culinary dish described by the food recipe;

1.3 Dissertation Structure

In addition to this chapter, this dissertation is composed by the following chapters:

- **Chapter 2** presents an overview on the related works to the project, the techniques developed until the time of writing on information extraction, and the information sources relevant to the project;
- **Chapter 3** presents a detailed description of the methodology applied for the development of the system and an account of the system's characteristics and components;
- **Chapter 4** presents the methodology applied for the testing and validation of the system, the results obtained, and a discussion of the results;
- **Chapter 5** presents the conclusions obtained through the system's development as well as potential future improvements applicable to the system;

Chapter 2

Preliminary Considerations

In this chapter, the preliminary considerations for the areas relative to the problems approached in this dissertation are presented. The chapter is divided into three main sections: **information extraction**, **related work** and **information sources**.

For the **information extraction** section, an overview on the main tasks involved in an information extraction activity is provided. These tasks include the segmentation, classification, association, normalization and co-reference resolution over written text. An analysis on the technologies primarily used in information extraction activities is also presented.

For the **related work** section, an analysis on a few of most relevant systems to this dissertation developed in the food recipe domain is presented.

For the **information sources** section, an overview on the information structure of recipes on food recipe websites is provided, alongside an analysis on both the dataset that will serve as the primary information source for the project and food composition databases.

2.1 Information Extraction

Information extraction can be defined as "the automatic extraction of structured information such as entities, relationships between entities, and attributes describing entities from unstructured sources" [Sar08]. From this definition, it's possible to infer that the extraction and structuring of information are two deeply-woven areas, since the extraction of information traditionally implies the need for its structuring.

Information extraction generally involves the processing of human language texts through the use of natural language processing, but other activities such as the automatic annotation and extraction of content from multimedia sources (images/audio/video) can also be looked at as information extraction activities [BC12]. It's possible to denote, then, that information extraction is an activity that can be applied to a multitude of areas, including the nutrition area.

Depending on the problems being tackled and the information extraction activity at hand, there are multiple different approaches that can be followed in order to accomplish the proposed activity. However, since information extraction activities tend to be complex ventures, these are usually divided into tasks. Decomposing an information extraction activity into a set of tasks brings some advantages, such as the possibility to choose and apply the techniques and algorithms that best befit each task, and the simplification of local debugging that comes with the modulation of the activity into tasks, since each module is independent from one another [SGC09].

Information extraction tasks can be decomposed into four main categories. The categories and the pipeline that an information extraction activity generally employs are shown in Figure 2.1.

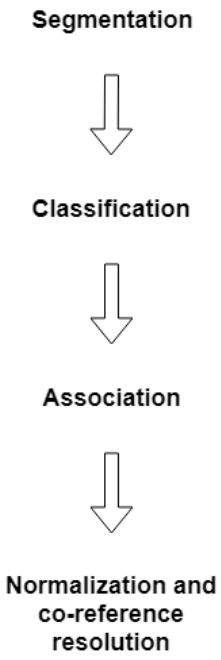


Figure 2.1: The main categories and general pipeline of an information extraction activity.

In this section, the general goals and problems of each information extraction task are presented, alongside the techniques that are most frequently used in each of these tasks.

2.1.1 Segmentation

Goals

The goal of the segmentation task is to divide the text on which the information extraction task is being performed into meaningful units, generally denominated segments or tokens. These segments or tokens can represent words, sentences or topics of information [McC05].

The segmentation task is the first task to be performed when undertaking an information extraction activity, as it is an essential part of each other task in the activity [SGC09].

Problems

Segmentation is, effectively, a non-trivial task, since the problems that can be associated to it vary not only based on the region of origin of the language the text is written on (e.g. while Western languages generally have explicit word boundary markers, such as spaces, these markers are often missing on East Asian languages) but also for the language itself (e.g. the use of apostrophes for contractions in English and French, which lead to ambiguities during the process of word segmentation, is not traditionally present in other Western languages) [MRS08]. As such, segmentation has a wide array of problems that need to be dealt with on a case-by-case basis.

The task of segmentation is usually associated to word segmentation, that is, the division of a string or strings of written language into the words that compose it. However, this task can also be used for other types of segmentation, which include but are not limited to: intent segmentation (the division of written words into key phrases); sentence segmentation (the division of a string or strings of written language into the sentences that compose it); topic segmentation (the division of a string or strings of written language into the different topics that compose it); other types of segmentation, which can include the division of a text into paragraphs or morphemes (the smallest grammatical unit in a language) [KTS13].

Depending on the nature of the segmentation task, and the language of the text used in the information extraction activity, some specific problems can occur. Some of the problems associated to Western languages include [MS99]:

- **Ambiguity of the period mark** (.): Words can have attached to them punctuation in the form of a comma, semicolon or a period. While the period mark often denotes the end of a sentence, this is not always the case. The period mark can also be a part of abbreviations (e.g. "etc.") and, in these cases, should be considered a part of the word when segmenting the text. However, these cases can lead to situations where, when the abbreviation is the last word in a sentence, the period mark denoting the abbreviation presents both the function of marking an abbreviation and a full stop in the text;

- **Use of apostrophes** ('): As previously mentioned in this section, in languages such as English and French, apostrophes are often used to indicate contractions. The use of apostrophes can lead to ambiguities when segmenting the text in situations such as the contraction "I'll", which can be looked at as both a single word or two words (since "I'll" is the contraction of "I will"). As such, some segmentation systems will interpret contractions as a single word while others will interpret these as two words;

- **Hyphenation**: There are cases where compound words represent a single word and should be treated as such (e.g. "e-mail", "mother-in-law"), but there are also some situations where the hyphenation present is purely lexical, and it is possible to separate these words into two independent segments (e.g. "well-known");

- **Whitespace**: While generally considered a word boundary marker, two words separated by a whitespace should not always be treated as a different segment. Segments such as "New York" and "data base", despite being separated by a whitespace, refer to a single concept, and should be treated as a single segment.

Problems specific to East Asian languages can include, for example, the lack of whitespaces between words and compound nouns being written as a single word [MS99].

It's possible to conclude, then, that there are a number of problems associated to the segmentation of written text, some of which are specific to a few languages, while others are nearly universal.

Techniques

There is no generic technique that can universally solve the problems mentioned in the previous section, and different techniques must be applied to solve the problems in the segmentation of text for each specific language.

Generally speaking, each individual problem is solvable by applying language-specific rules that show how each of the cases discussed in the previous section should be handled for a certain language. While not a perfect or universal solution, it nonetheless allows for the precise segmentation of text for specific languages. However, language-specific rules are only often sufficient for the segmentation of text written in Western languages, as East Asian languages face other significant issues.

As previously mentioned, text in East Asian languages is written without any spaces between the words, in which case solving the aforementioned problems is often not enough to segment text written in these languages. One of the solutions that can be applied to segment the text involves using external resources such as large lexicons or grammars to perform a syntactic or lexical analysis. This use usually involves taking the longest (and best) vocabulary match for each word in the text with heuristics in order to deal with unknown words [MRS08]. Other methods include the use of Hidden Markov Models or Conditional Random Fields, which are trained over hand-segmented words [KhCL^{+}05], or the use of techniques based on the statistics area, such as n-grams [KH06] and the Viterbi algorithm [For73].

Other methods that can be applied in order to segment a text can also be found in other areas, in particular, the information retrieval area. While information extraction and information retrieval are fundamentally different areas (information extraction is concerned with the extraction and structuring of unstructured information in a written text, information retrieval revolves around finding information that is able to fulfill an information need provided by a query in a collection of documents), both areas involve tasks that perform some segmentation over written text.

The methods that can be applied to the segmentation task in the information extraction activity from the information retrieval area are stemming and lemmatization. Both methods have the goal of reducing the amount of inflectional forms and derivative forms of a word in the text by reducing them to a common base form (e.g. "am", "are" and "is" would be reduced to the base form "be"). Stemming refers to a simple heuristic process that chops off the ends of words in order to try and achieve its goal, while lemmatization refers to a more complex heuristic process which includes the use of both a vocabulary and the morphological analysis of words in order to try and reduce each word to its base or dictionary form, also known as the *lemma* [MRS08]. Both these methods can be particularly useful for the classification task, when the classification of a segment revolves around words that have multiple different inflectional and derivative forms.

2.1.2 Classification

Goals

The main goal of the classification task is to determine the type of each of the segments obtained through the segmentation task, that is, the field of the output data structure that the segment belongs too. The outcome of this task is expected to be the classification of each set of segments as an entity, that is, elements of a given class which could be potentially relevant for the extraction domain [SGC09].

Problems

Problems related to the classification task often vary depending on the techniques used and, as such, there are only a few general issues associated to the task. Some of the general issues related to the classification task include:

- **Homonyms and homographs**: One of the general issues faced in the classification task relates to the ambiguity present in written text through the existence of homonyms and homographs. Situations where a single word can refer to multiple different entities depending on the text's context ("Ford" can refer to a person, a company or even to the ford structure) can be found in written text, which complicates the classification task [Web73];

- **Domain specific performance**: Classification tasks involve the analysis and classification of written text over a certain domain (e.g. the nutrition area). Each domain has specific expressions and vocabulary that are more commonly used within the texts of the domain. This specificity often results in techniques created for a classification task having significantly better performance for the domain they were developed for [PK01]. As such, reusing systems developed for other classification tasks often involves complex readjustments, in order to be properly adapted for the classification task at hand.

Other issues faced in the classification task are usually specific to the use of a certain technique in conjunction with the domain the classification task is being applied on.

Techniques

The techniques used for the classification task in an information extraction activity can be divided into two main types: rule-based and machine learning.

Rule-based techniques are normally based on linguistic resources, making use of lexicons and grammars. These techniques usually revolve around simplistic matching between the different segments obtained through the segmentation of the written text and the elements present in the lexicon. The grammar supports the recognition of some terms that may not be in the lexicon, and is able to provide a solution to the ambiguities that may exist when categorizing a segment through the lexicon (e.g. when there are homonym and homograph relationships in the text). Some techniques of morphological analysis may also be used to complement rule-based techniques [FKK⁺00].

Machine learning techniques used in the classification task are generally supervised, that is, a collection of annotated test documents is needed in order to apply these techniques. Some of the most common supervised learning techniques are Hidden Markov Models (HMM), Maximum Entropy Markov Models (MEMM) [MFP00], Conditional Random Fields (CRF) [LMP01], Support Vector Machines [IK02] and Decision Trees [SGS98].

Hidden Markov Models are extensively used in classification tasks such as grammatical tagging, which consists in the classification of words according to the morphological class they belong too. These models are based on: 1) a set of hidden states that are normally attached to a physical meaning (e.g. in the classification of segments, each state can be associated to an entity type); 2) a set of observable symbols which correspond to observable events (e.g. segments that occur at each time); 3) a function that returns the probability distribution of the state transition; 4) a function that returns the probability distribution of detecting a given symbol in a given state; 5) a vector that corresponds to the distribution of probabilities in the initial state. The probabilistic model is constructed during the training phase of the technique. The classification is generated through the use of the generative model, in which the most likely classification for each segment is found. The most likely classification is normally computed through the Viterbi algorithm, which serves to find the most likely sequence of hidden states (the Viterbi path) that results in a sequence of observed events [SGC09]. While the HMM is an adequate technique for the classification task in an information extraction activity, it is also a limited one. HMM relies on two types of probabilities: $P(Tag|Tag)$ and $P(Word|Tag)$. As such, introducing other knowledge in the classification process (e.g. information about capitalization) is not a simple task, as modeling the knowledge requires coding it only with the two kinds of probabilities that the HMM relies on [MFP00].

Maximum Entropy Markov Models attempt to solve the aforementioned limitation in HMMs via the use of a discriminative model. While HMMs use a generative model, which constructs a probability density model over all the variables involved in a system, MEMMs don't make a direct attempt to model the underlying distribution of the variables. As such, instead of having separate models for $P(Tag|Tag)$ and $P(Word|Tag)$, the MEMM trains a single probabilistic model to estimate $P(Tag_{\mathrm{i}}|Word_{\mathrm{i}}, Tag_{\mathrm{i}} - 1)$. The probabilistic model uses a maximum entropy classifier which estimates, for each given local tag to an observed word, a tag for the prior word and a group of features that correspond to the modeling of additional information being introduced in the process [SGC09]. Very much like the HMM, the MEMM uses the Viterbi algorithm to calculate the most likely classification given to a segment. While MEMMs are a good solution when it is necessary to introduce additional information in the statistical model, their performance can pale in comparison to HMMs if no additional information is used in the model. This is due to the *label bias problem*, as the MEMMs tend to favor states with a lower number of transitions to other states, giving an unfair advantage to those states [SGC09].

In an attempt to solve both the label bias problem and offer all the advantages provided by Maximum Entropy Markov Models, Conditional Random Fields were developed as a potential solution. The major difference between the CRF model and the MEMM is that CRFs use a single exponential model for the joint probability of the entire sequence of labels when given the observation sequence, while MEMMs use a conditional probabilities exponential model for each state. Via this difference, it is possible to normalize the probabilities at a global level, which prevents the label bias problem. However, while CRF models are generally superior to both HMMs and MEMMs, training these models is an expensive process, which makes them difficult to use if, as new data surfaces, the model needs to be continually trained [LMP01].

Support Vector Machines assume it is possible to map the segments that compose the written text in a vector space according to either linguistic or graphical properties of the segments, alongside the words in its neighboring regions. The text segments are mapped into a vector space, followed by the separation of positive elements (the text segments that belong to the class) from negative elements (text segments that do no belong to the class) via the use of an hyperplane for each individual class [IK02]. In the training phase of the algorithm, the goal is to find the hyperplanes which achieve the best separation between positive and negative elements. In the testing phase, the classification is generally performed by looking at which side of the hyperplane the input is located [IK02].

In Decision Trees, during the training phase, a probabilistic decision tree is built based on the morphologic classification, type of characters used (for East Asian languages) and the information found in the dictionary over the neighborhood of the word (consisting of the words preceding and following the word, as well as the word itself) of each word in the written text. During the testing phase, the properties of the neighborhood of each word are analyzed and then compared to the decision tree in order to associate a probability of belonging to a certain class [SGC09]. Since the probabilities of every segment of the written text are computed, the goal of the task consists in discovering the most consistent sequence of probabilities, for which the Viterbi algorithm is used [SGS98].

2.1.3 Association

Goals

The goal of the association task is to identify and characterize how the different entities ascertained in the classification task relate to one another through linguistic cues present in the written text [McC05].

Problems

There are two major general problems with the association task, which are present independently of the technique used for the task. These problems are as follows [McC05]:

- **Reliance on previous tasks**: Success in the association task is not exclusive to the performance of the technique applied in the task. Since the association task is generally performed after both the segmentation and classification tasks, it is necessary that both of these tasks are performed correctly in order to successfully identify associations between segments;

- **Language subtleties**: There are multiple domains, such as news or scientific articles, where the evidence necessary to identify and characterize certain relations or associations requires understanding complex subtleties for the usage and meaning of the language the text is written on. These subtleties are not simple to properly compute.

Other issues in the association task are usually specific to the use of a certain technique in conjunction with the domain the task is being applied on.

Techniques

The techniques used for the association task in an information extraction activity can be divided into two main types: rule-based and machine learning.

Rule-based techniques are usually the favored type of technique for the association task. The simplest technique involves the use of a set of patterns in order to extract a limited set of relationships (e.g. a simple rule could involve the extraction of a relationship between a person and company, on which, upon detection of the segment "<Person> works for <Company>", the values corresponding to <Person> and <Company> would be inserted into the relation worksFor(Person, Company)). However, this technique is only a serviceable solution in simple cases, where a limited variety of relationships is expected. A wide variety of relationships would imply the need to develop a new different rule in order to extract each different relationship present in the written text, which is a time-consuming, often unfeasible task [SGC09].

For more complex cases, the approach can be based on *syntactic analysis*, in order to develop more generic, fitting rules. It is normally the case that the relationships to be extracted are grammatical relationships (e.g. a verb can indicate a relationship between two different entities in the text). However, the execution of a complete syntactic analysis of the text, where the text is analyzed through a single syntactic text, is an expensive task that usually results in a considerable number of errors [Gri97]. These mistakes still occur at the time of writing. As such, a partial syntactic analysis, where the written text is divided into parts where each part is associated to its syntactic tree, is able to return better results while not being as expensive. This analysis has, however, the disadvantage of ignoring some linguistic patterns, such as conjunctions or modifiers [Gri97].

Machine learning techniques are not as commonly used as rule-based approaches in the association task. One of the first machine learning approaches employed for the task was based on probabilistic context-free grammars, which differ from regular context-free grammars due to the use of a probability value associated to each rule in the grammar [MCF$^+$98]. Through the use of a probabilistic context-free grammar, when the syntactic analysis is initiated, it is possible to identify multiple syntactic trees. Probabilistic rules are then used to compute the probability of each tree, and the most probable tree is picked [MCF+98].

It's also possible to adapt some of the machine learning approaches referred to in Section 2.1.2, such as Maximum Entropy Markov Models and Conditional Random Fields, to the association task. In order to do so, it is necessary to add information about the neighborhood of a word through the form of features, that is, instead of tagging each word with a given class, each word would be tagged with relationships to other words in the written text [SGC09].

2.1.4 Normalization and Co-reference Resolution

Goals

The normalization and co-reference resolution tasks each have distinct but related goals.

The normalization task consists in the transformation of information into a standard format, which is usually defined by the user. It is a necessary task since some information types do not always conform to a standard format. An example of this is the representation of hours, which can be done through the formats <Hour>pm, <Hour>h or <Hour>:<Minutes>. The different representations of certain concepts can pose difficulties when it's necessary to perform comparisons between entities [McC05].

The co-reference task consists in handling cases of co-reference, that is, when multiple expressions in a text refer to the same real world entity (e.g. the entity "Kendrick Lamar" can be referred to, in the same text, as "Kung Fu Kenny"). Co-reference occurs not only due to the use of different names to describe the same entity, but also due to the use of classification expressions (e.g. "Kendrick Lamar" being referred to as "the greatest rapper of his generation") and pronouns (e.g. in the text "Kendrick Lamar is the greatest rapper of his generation. He just released a new album", the pronoun "He" refers to "Kendrick Lamar") [SGC09].

Problems

Problems relating to the normalization and co-reference resolution tasks are specific to the technique and the context of the tasks. These techniques and the problems associated to them are described in the following section.

Techniques

The normalization task is traditionally done through the use of conversion rules, which produce the standard formats previously defined for certain information by a user [SGC09]. The only considerable downside to this technique is, over a considerable quantity of information, the amount of necessary conversion rules defined tends to grow significantly, and the definition of these rules can be a time-consuming task.

The techniques used for the co-reference task can be divided into two main types: rule-based and machine learning.

Rule-based techniques for dealing with co-reference normally take into consideration any semantic information about entities. Through the use of this information, it's possible to detect any entities whose semantic information coincides through filtering. The filtering can be done manually or using an independent source, in order to determine the necessary semantic information associated to each word in the text. At the end of the filtering phase, it is necessary to determine the entities that have the highest probability of being co-referent [SGC09].

Machine-learning techniques can also be used for the co-reference resolution task. One such approach was based on clustering algorithms for grouping similar entities [CW99]. The approach involved the analysis of the entities of a document from the ending to the beginning of the document, with the distance between each of the entities being computed using an incompatibility function and a set of weighting constants (using attributes such as the name, position, number, etc.). If the distance between the two entities was less than the pre-defined cluster radius, it was concluded that the entities belonged to the same cluster. If the two entities were contained in the same cluster in the final result, they were considered co-referent [CW99].

Another machine-learning technique employed in the co-reference resolution task involves the use of decision trees [SNL01]. In a manner similar to the previously described clustering approach, the construction of the decision tree in the training phase makes use of a set of attributes such as the name, position, number, etc. Past the training phase, the text is processed from left to right and each entity is compared with every preceding entity. For each pair, the tree is used to verify whether the pair's elements are co-referent [SNL01].

2.1.5 Technologies

In general, the most popular programming languages for natural language processing and, consequently, information extraction activities, are Java and Python, since there is a broad range of NLP toolkits available for these languages. The four most well-known and widely used toolkits by the NLP community are NLTK[1], Apache OpenNLP[2], Stanford CoreNLP[3] and Pattern[4] [POA16].

NLTK is a Python library divided into independent modules that can be used for multiple different tasks in an information extraction activity. These include but are not limited to: word and sentence-level segmentation, stemming and lemming, tagging, parsing, named entity recognition, etc. The library is also bundled with popular corpus samples (e.g. PENN Treebank Corpus, Reuters Corpus) that can be used to train different machine learning algorithms relative to the different tasks involved in natural language processing [LB02].

Apache OpenNLP is a Java library which applies machine learning techniques to natural language processing and, consequently, information extraction tasks. The library has built-in support for word and sentence-level segmentation, stemming and lemming, tagging, parsing, named entity recognition, etc. Users of the library can choose to rely on pre-trained models that are bundled with

[1] http://www.nltk.org/

[2] https://opennlp.apache.org/

[3] https://stanfordnlp.github.io/CoreNLP/

[4] https://www.clips.uantwerpen.be/pattern

the library for their natural language processing needs or train their own models with a Perceptron or a Maximum Entropy Markov Model [POA16].

The Stanford CoreNLP toolkit is a Java pipeline that offers a wide array of language processing techniques. The pipeline has built-in support for word and sentence-level segmentation, stemming and lemming, tagging, parsing, named entity recognition, etc. Additionally, CoreNLP is a relatively simple and straightforward pipeline to set up and run comparatively to other similar frameworks [MSB+14].

Pattern is a Python library with functionality for web mining, natural language processing, machine learning and network analysis. The library has built-in support for tokenization, tagging and chunking [DSD12].

2.2 Related Work

Several articles have been published regarding the extraction and structuring of information in a food recipe, the majority of which focus exclusively on the analysis of the ingredients field. While none attempt to calculate accurate nutritional values through the use of extensive additional extracted information on each ingredient, some attempt to calculate the nutritional information of a recipe through the structuring of the ingredient's name, quantities, used cooking method, etc., or extract and structure a field of relevance to the user.

Four of the most relevant articles related to this project are analyzed in the following sections. A summary of the articles' most relevant characteristics is provided in Table 2.1.

	Characteristics		
Articles	Dataset Language	System Type	Information Extracted
MSYY12	Japanese	Machine-Learning	General ingredient information, actions, utensils, durations, quantities
UII11	Japanese	Rule-Based	Ingredient information, nutritional values
HG13	French	Hybrid	Ingredient names and quantities
SSL+14	English	Machine-Learning	Cuisine types

Table 2.1: Description of the most relevant characteristics of the related work analyzed. These characteristics include the language of the dataset, the type of system developed and the information extracted.

A Machine Learning Approach to Recipe Text Processing

The system developed by Mori et al. [MSYY12] approaches four different tasks related to the extraction and structuring of information in a food recipe through a machine-learning approach. The tasks focused on were: Word segmentation (which is inherently more problematic in the Japanese language, due to the lack of explicit word boundaries), Named Entity Recognition, Syntactic Analysis and Predicate-Argument Structure Analysis.

For the word segmentation task, Mori et al. adopted a pointwise approach that allows for model training in machine learning by referring to partially annotated sentences, allowing the focus of the annotation resources to center on particularly complicated parts of the text, namely those specific to the domain. The task was formulated as a binary classification problem, which was solved through the use of support vector machines. Information regarding the surrounding characters as well as the presence or absence of words in the domain-specific dictionary were used as features.

For the named entity recognition task, a pointwise approach was also adopted. The system estimates the parameters of the NE classifier based on logistic regression from fully and partially annotated data. Then, given a word sequence, the classifier enumerates all the possible tags for each word, alongside their probabilities. As the final step, a search is done for the tag sequence with the highest probability that satisfies the constraints imposed on the classifier.

For the syntactic analysis task, Mori et al. approached the task similarly to recent projects involving dependency parsing, altering the parameters of a state-of-the-art syntactic analyzer (which are estimated from an annotated corpus in the general domain) through a pointwise approach, in order to adapt the analyzer's parameters to the recipe text domain.

For the predicate-argument structure analysis, the input sentence of the food recipe text is transformed into a dependency tree where the nodes are words, and some subtrees are annotated with a named entity tag. The following steps are then executed on the dependency tree [MSYY12]:

1. The next named entity tagged with *Ac* (action by the chef) or *Af* (action by the food) is found;

2. The named entity is set as the predicate with unknown arguments;

3. All the named entity sequences that depend on the predicate are enumerated;

4. A predicate-argument structure is constructed using the predicate set and sequences enumerated.

The relevance of the work developed by Mori et al. is relevant to the project approached in this dissertation pertains to the tasks of named entity recognition, syntactic analysis and predicate-argument analysis, since these tasks are core subproblems of the larger problems approached in the project.

A Recipe Recommendation System Based on Automatic Nutrition Information Extraction

The system developed by Ueta et al. [UII11] focuses on the recommendation of food recipes to users with health-related conditions. The system's flow is divided into four distinct steps:

1. The user inputs a natural language query relative to his or her health-related condition (e.g. "I want to prevent diabetes");

2. The system segments the query into its morpheme units, extracting the noun;
3. The system searches its co-occurrence dictionary for the most relevant nutrient pertaining to the noun extracted in the previous step;
4. The dishes with the most significant presence of the nutrient that was identified in the previous step are searched for in the food database built for the project. These dishes are then retrieved and shown to the user;

The co-occurrence dictionary contains data for forty five nutrients, having been built through the use of 500 ranked web pages, which were obtained from Google search results for each nutrient. Each of these results was analyzed in order to identify the nouns in the document, and to record which nouns appeared more frequently in the same documents as the nutrient.

The ingredient nutrient database developed for the project consists in nutritional information collected for a total of 1861 ingredients through a crawler. The database is utilized in the project in order to identify which ingredients contain which amounts of each nutrient (e.g. raw chicken liver contains a fairly significant amount of Vitamin B5 and, as such, dishes with this ingredient will be given a higher priority when this nutrient is relevant to the user query inputted).

A nutritional information database was also developed, which contains 800,000 recipes associated to different types and amounts of nutrients. In order to populate the database, two information extraction activities were performed - one to identify the ingredients and respective amounts of each recipe, the other to determine the applied cooking methods to each ingredient. The database was used in order to retrieve the recipes that fulfilled the information need of the user transmitted through the query.

The system developed by Ueta et al. attempts to solve a similar problem to the one approached in this dissertation and, as such, it is fairly informative, especially pertaining to the system's flow. However, the article lacks detailed information regarding the methods applied in the system.

Extraction of Ingredient Names from Recipes by Combining Linguistic Annotations and CRF Selection

The system developed by Hamon and Grabar [HG13] performs an information extraction activity over the title and instructions fields of the recipe in order to extract information relative to the ingredients (name, quantities) of a recipe. The system is a hybrid approach composed by two distinct parts: a rule-based and a machine learning system.

The rule-based system is used to perform the recognition of terms (e.g. ingredients, food, kitchen utensils) and associated information (e.g. quantities and durations) in a recipe. The system's flow is divided into three steps:

1. Term extraction;
2. Ingredient name weighting;
3. Ingredient name selection;

The machine learning system uses a Conditional Random Field implementation with the following settings [HG13]:

- Sentences are considered sequences;
- Each segment of the sequences is annotated with its inflected and lemmatized forms, grammatical category, semantic tag, number of words and co-occurrence with quantities;
- The system predicts if the annotated elements are relevant ingredients, and whether the correct form of the segments corresponds to its inflected or lemmatized formm, through analyzing the order of the element's co-occurrence (the form of the first occurrence of the ingredient is considered the correct one).

The system operates over recipes with an unique structure (since the ingredients field is not structured in the recipes analyzed by the system), and attempts to extract individual ingredient names and associations to each ingredient of relevant information (e.g. quantity) in a food recipe. This is one of the subproblems relative to the extraction tasks approached in the project developed for this dissertation.

Automatic Recipe Cuisine Classification by Ingredients

The system developed by Su et al. [SLL+14] attempts to classify individual recipes in their respective cuisine styles through the analysis and classification of the recipe's ingredients.

The system employs both associative classification and support vector machine techniques in order to achieve its goal. For the associative classification, the recipe is considered an item set of ingredients on which to build the classification rules. The system's flow is divided into three steps:

1. Rule generation to find the frequent patterns containing classification rules, where a rule is a combination of an ingredient set and cuisine label;
2. Classifier building through filtering out redundant rules and organizing useful rules;
3. Input of unlabeled recipes, on which the classifier will select the rules based on how they match up with the different recipes;

For the support vector machine, each ingredient in a recipe is treated as a feature. Through the use of a Boolean model, a recipe-ingredient matrix is calculated and applied into the SVM classification method. To this matrix, single-value decomposition is used in order to discover any latent concepts of cuisines in the matrix. Five-fold cross validation is utilized as the evaluation setting.

The analysis and classification of the recipe's ingredients is a fairly relevant part of this dissertation, as it is a necessary step in order to both calculate each ingredient's nutritional values as well as to develop associations for each ingredient with other structured information in the recipe (e.g. applied cooking methods).

2.3 Information Sources

2.3.1 Food Recipe Websites

The analysis of some of the most popular food recipe websites is an essential step of the project, since it allows one to understand what information is usually structured and unstructured in online food recipes. It is through this analysis that it is possible to better define what information is present in the recipe that is generally unstructured and of value to a user and, thus, worth extracting. As such, the three most popular websites that provide food recipes, according to the eBizMBA rank (which estimates a website's traffic through the Alexa Global Traffic Rank[5], Compete U.S. Traffic Rank[6] and Quantcast U.S. Traffic Rank[7]), were examined in order to obtain a more concrete idea of the general information structure of the recipes available through food recipe websites. The websites chosen were AllRecipes[8], FoodNetwork[9] and Genius Kitchen[10], which ranked as the three most popular food recipe websites worldwide as of May 2018 [eBi18]. A summary of the information structure of these websites is presented in Table 2.2.

	Characteristics				
Websites	Title	Ingredients list	Instructions	Nutritional values	Structured ingredient information
AllRecipes	Yes	Yes	Yes, structured by steps	Yes	No
FoodNetwork	Yes	Yes	Yes	No	No
Genius Kitchen	Yes	Yes	Yes, structured by steps	Yes	Yes

Table 2.2: Summary of the information structure relative to online food recipes on the three most popular food recipe websites worldwide.

AllRecipes

AllRecipes is a website that focuses exclusively on food recipes. The recipes available on the website are, for the most part, user-submitted, although some of them are written by editors of the website.

Through the examination of the structured information relevant to the project, as shown in Table 2.2, it's clear there are a number of fields of structured information present in the food recipes of the website. Interestingly, the website offers complete nutritional information for each recipe, which is not always present in food recipe websites. It includes not only the caloric content of the recipe, but additional information as well (e.g. fat, carbohydrates, protein). This nutritional information is, according to the website's documentation, calculated through the use of ESHA

[5] https://www.alexa.com/topsites
[6] https://www.compete.com/
[7] https://www.quantcast.com/top-sites/
[8] http://allrecipes.com/
[9] https://www.foodnetwork.com/
[10] http://www.geniuskitchen.com/

Research[11]'s nutrient databases for recipe nutrition analysis [All]. Since the information regarding each listed ingredient is not structured, it's possible to assume that an information extraction activity is being performed by the website, which extracts and structures some of the information pertaining to each ingredient of the recipe, and then cross-references this information with ESHA Research's food composition databases.

FoodNetwork

FoodNetwork is a website that focuses on multiple different food-related areas (e.g. restaurants, chefs), including recipes. The recipes available on the website are, for the most part, user-submitted, although some of the recipes offered are written by editors of the website and chefs featured on the website.

Through the examination of the structured information, as shown in Table 2.2, the website presents a similar amount of structured fields of information in its food recipes, although it pales in comparison to the structured information available on the other websites analyzed, as it does not structure any nutritional information nor any information about the recipe's ingredients. Additionally, the directions for the recipe are not structured by steps, unlike on the other websites studied in this section.

Genius Kitchen

Genius Kitchen is a website that focuses almost exclusively on food recipes. The recipes available on the website are, for the most part, user-submitted, although some of the recipes offered are written by editors of the website and chefs featured on the shows provided by the website.

As shown in Table 2.2, the website's food recipes contain the largest number of structured fields of information of the three websites analyzed. Similarly to AllRecipes, the website offers nutritional information on each recipe, something which is not commonly present in food recipe websites. This nutritional information includes not only the caloric content of the recipe, but additional information as well (e.g. fat, carbohydrates, protein). Unlike the AllRecipes website, however, Genius Kitchen also offers structured information on each ingredient, including the quantities needed for the recipe and the unit they are listed in, a brief description of the ingredient, its seasonable availability, information on storing it and adequate substitutes for the ingredient. As such, it's unlikely that an information extraction activity is being performed for the calculation of the nutritional values. It's possible to assume that, instead, the calculation of these values is being done directly through the website's structured information on each recipe's ingredients.

2.3.2 Dataset

Past the analysis of the information structuring present in food recipe websites, it's necessary to choose an adequate dataset on which to develop and base the project's results on. Ideally, the

[11] https://www.esha.com/

information structure of the food recipes of the dataset should be as close as possible to that of the recipes available on the websites studied, in order to allow for the system developed to properly function across similarly modeled online food recipes.

An analysis on the different datasets of food recipes available on the Internet, as well as the characterization of the chosen dataset for the project, are presented in the following subsections.

Dataset Choice

	Characteristics				
Dataset Sources	Free plan	Built Dataset	Title	Ingredients list	Instructions
BigOven API	No	No	Yes	Yes	No
Spoonacular - Recipes API	Yes	No	Yes	Yes	Yes
ReciPal API	No	No	Yes	Yes	No
Yummly-28K	-	Yes	Yes	Yes	No
Epicurious - Recipes with Rating and Nutrition	-	Yes	Yes	Yes	Yes
Recipe Ingredients Dataset	-	Yes	Optional	Yes	No

Table 2.3: Comparison of some of the different dataset sources for food recipes available online, which include public APIs and built datasets.

There are multiple different ways of obtaining a dataset for a project, which can range from downloading one available online to building it specifically for a project through the use of varied sources.

For this project, both the possibility of building a dataset through the use of public APIs made available by food recipe websites or using an already built dataset were analyzed. However, the first aforementioned possibility was quickly discarded, as there were already a substantial amount of datasets of food recipes built in a similar manner available online. Additionally, only a few APIs (such as the Spoonacular - Recipes API[12], as shown in Table 2.3) included all the fields of information necessary for the project's development, while the majority of the APIs available relative to food recipes (such as the BigOven API[13] and ReciPal API[14], as shown on Table 2.3) lacked one or more fields (in this case, the instructions field) essential to the project's development.

Of the food recipe datasets available online, there were multiple datasets that didn't include one or more fields essential to the project, which are comprised by the title, ingredients and instructions fields of the recipe. These datasets include the Yummly-28K dataset, made available by Luis Herranz[15], which does not include the instructions field, and the Recipe Ingredients Dataset, made available on the Kaggle website[16], which has an optional title field and does not present

[12] https://spoonacular.com/api/docs/recipes-api

[13] http://api2.bigoven.com/web/documentation

[14] https://www.recipal.com/api-docs

[15] http://lherranz.org/datasets/

[16] https://www.kaggle.com/kaggle/recipe-ingredients-dataset

an instructions field, as shown in Table 2.3. Given that the information extraction activities being performed in this project are reliant on these fields, it was necessary to choose a dataset that contained all three fields, with each containing some written text over which to perform the information extraction activity.

Given the necessary parameters, the chosen dataset for the project was the Epicurious - Recipes with Rating and Nutrition dataset, made available on the Kaggle website[17]. The chosen dataset is analyzed in the following section.

Dataset Characteristics

The dataset used in this project, obtained via the Kaggle website, consists in a collection of food recipes obtained from the Epicurious[18] website, a website almost fully dedicated to food recipes. written in the English language. The dataset presents the following structured information relevant to the project:

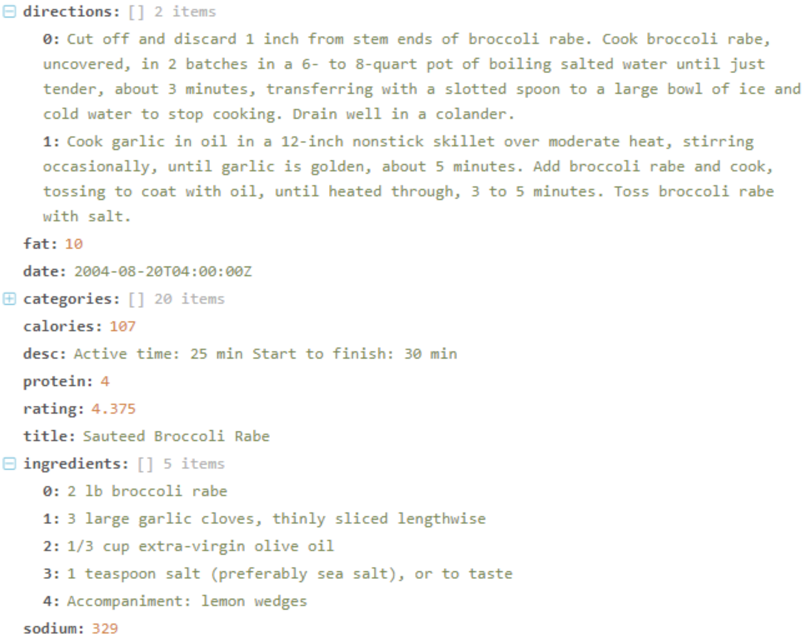
- Title;
- Nutritional information of the recipe (including calories, fat, carbohydrates, etc.);
- List of ingredients for the recipe;
- List of directions for the recipe, structured by steps;

Through the examination of the structured information, it's possible to observe that the dataset provides similar structured information to that of the websites analyzed in Section 2.3.1, including the nutritional information for the recipe. According to the Epicurious website, the nutritional information for a recipe is calculated using the Nutrition Analysis API provided by the Edamam[19] company [Ste13]. Since the information regarding each listed ingredient is not structured, it's possible to assume that an information extraction activity is being done by the website, which extracts and structures some of the information of each ingredient of the recipe, and then makes use of Edamam's Nutrition Analysis API in order to determine the nutritional information of a recipe. An example of the structure for a recipe in the dataset is shown in Figure 2.2.

[17] https://www.kaggle.com/hugodarwood/epirecipes

[18] https://www.epicurious.com/

[19] https://www.edamam.com/



```
⊟ directions: [] 2 items
    0: Cut off and discard 1 inch from stem ends of broccoli rabe. Cook broccoli rabe,
    uncovered, in 2 batches in a 6- to 8-quart pot of boiling salted water until just
    tender, about 3 minutes, transferring with a slotted spoon to a large bowl of ice and
    cold water to stop cooking. Drain well in a colander.
    1: Cook garlic in oil in a 12-inch nonstick skillet over moderate heat, stirring
    occasionally, until garlic is golden, about 5 minutes. Add broccoli rabe and cook,
    tossing to coat with oil, until heated through, 3 to 5 minutes. Toss broccoli rabe
    with salt.
  fat: 10
  date: 2004-08-20T04:00:00Z
⊞ categories: [] 20 items
  calories: 107
  desc: Active time: 25 min Start to finish: 30 min
  protein: 4
  rating: 4.375
  title: Sauteed Broccoli Rabe
⊟ ingredients: [] 5 items
    0: 2 lb broccoli rabe
    1: 3 large garlic cloves, thinly sliced lengthwise
    2: 1/3 cup extra-virgin olive oil
    3: 1 teaspoon salt (preferably sea salt), or to taste
    4: Accompaniment: lemon wedges
  sodium: 329
```

Figure 2.2: An example of the structure of a recipe in the dataset, in this case, for the recipe titled "Sauteed Broccoli Rabe".

Preliminary Dataset Analysis

In order to better determine the project's requirements and the specific characteristics of the dataset, a preliminary analysis of some of its most relevant aspects was made, through an initial segmentation task and classification task (part-of-speech tagging).

The dataset originally contained 20130 food recipe entries, of which 19 did not contain any information. These invalid entries were removed from the dataset, leaving it with 20111 entries.

The average number of words and, particularly, of names and verbs, was calculated in order to better understand the composition of the written text of each recipe. The information obtained is shown in Table 2.4.

	Metrics	
Dataset	Mean	σ
Words	203.03	132.90
Nouns	66.05	40.64
Verbs	24.52	17.51

Table 2.4: Mean and standard deviation of the number of words and, specifically, nouns and verbs, for the food recipe dataset.

After performing the segmentation task on the dataset, the twenty most common nouns and verbs were calculated to better understand the proportion of relevant nouns and verbs to the information extraction activity (cooking methods, food preparation techniques, cooking utensils, ingredient names, etc.) in each recipe. This information is displayed, respectively, in Figures 2.3 and 2.4.

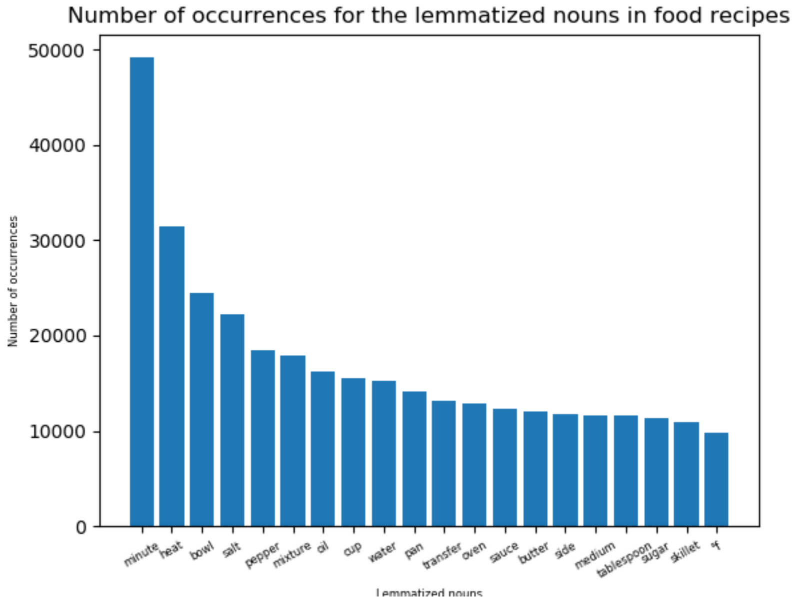


Figure 2.3: The twenty most common nouns present in the written text of the dataset's food recipes.

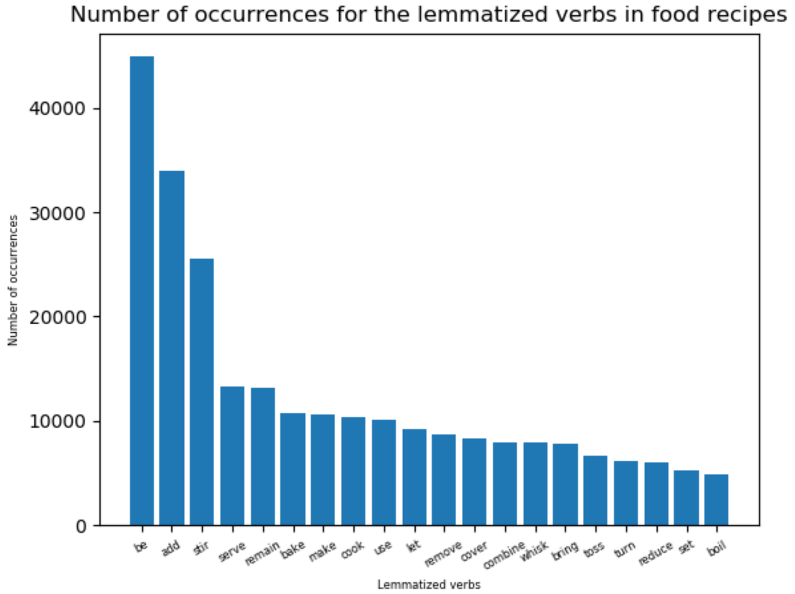


Figure 2.4: The twenty most common verbs present in the written text of the dataset's food recipes.

The figures show that, for the most part, the most common nouns and verbs in a recipe are not directly related to the information being extracted. Of the twenty most common verbs in the dataset, only two relate to cooking methods ("bake" and "boil") and four to food preparation techniques ("stir", "whisk", "combine" and "toss"). Of the twenty most common nouns, again, only four of the most common nouns are related to the information being extracted, in this case, cooking utensils ("bowl", "pan", "oven" and "skillet"). It's important to note, however, that it is not possible to draw any definitive evidence from the figures above in relation to the abundance of certain nouns or verbs in the recipe, as errors in the part-of-speech tagging process can cause these values to be warped in relation to the actual values. Instead, these figures serve merely as an indication of the presence of the most common nouns and verbs in the recipe.

2.3.3 Food Composition Databases

One of the problems dealt with in this project concerns the automatic calculation of the nutritional values of a food recipe. In order to obtain the necessary information for these calculations, it is important to look at food composition databases. These databases contain nutritional information about a wide array of different food items which can be used to solve the problem at hand by cross-referencing the information from a food recipe pertaining to an ingredient with the information available in the database.

Publicly accessible food composition databases can vary significantly in terms of content (which often varies by country) and quality [AD06]. For this project, a food composition database hosted at Fraunhofer Portugal AICOS was used.

Fraunhofer Portugal AICOS Food Composition Database

The food composition database hosted at Fraunhofer Portugal AICOS was built for a system that recommends meals to older adults named CordonGris [RRV+18]. The database contains 962 unique food items. The information contained in the database includes but is not limited to:

- Name of the food item in its native language;
- Name of the food item in English;
- Description of the food item;
- Food group of the food item;
- Quantities of different nutritional values (e.g. energy, water, protein) for a portion of the ingredient;
- Mass corresponding to a single portion of an ingredient, in grams.

The MySQL[20] database was built over LanguaL, a framework that allows for the description, characterization and retrieval of data about food items in an automated manner. The framework

[20] https://www.mysql.com/

is, at its core, a multilingual thesaurus that employs a faceted classification scheme. As such, each food item described by LanguaL is characterized by a set of standard terms chosen from facets related to the nutritional or hygienic quality of a food item. These terms can describe a food item's biological origin, methods of cooking and conservation, etc. The LanguaL framework also allows for a facilitated sharing of data between databases that make use of the framework, since the food items on the different databases have a standardized structured defined by LanguaL [MI15].

2.4 Summary and Conclusions

The review on the information extraction field showed that the extraction and structuring of information can be divided into four main tasks, each of which has its own goals, problems and techniques associated to it. It was also possible to note through this review that there were a considerable number of natural language toolkits available, with each offering multiple different techniques capable of providing adequate solutions to the different information extraction tasks.

The analysis of the related work to this project showed that, in general, there has been extensive work in the food recipe area, the majority of which has focused on the identification and classification of named entities in food recipes, with accentuated emphasis being placed on the ingredients. This analysis also provided important insight into the most common problems faced when executing an information extraction activity over the food recipe domain. Unfortunately, the results relative to the related work are not directly comparable to those of the system developed for this dissertation, as the specific extraction and association tasks performed by the system are not identical to those performed by the systems studied.

The study done on the three most popular food recipe websites as of May 2018 according to the eBizMBA rank [eBi18] showed that, in general, these websites presented a similar information structure on the available food recipes, with some small variations on the fields of the recipe that were structured, most notably the nutritional information field. The dataset chosen presents a similar information structure to that of the websites studied and, as such, should prove to be a relevant foundation for the project. A preliminary analysis of the dataset involved in the project was particularly useful to shed some light on the composition of the food recipes of the dataset, in terms of the quantity and standard deviation for the words (particularly nouns and verbs), as well as the most common nouns and verbs.

The analysis on the food composition databases publicly available showed that, for the most part, these databases don't contain either detailed or accurate information on the food items stored. As such, a food composition database hosted at Fraunhofer Portugal AICOS was used, built over LanguaL. The use of the LanguaL framework, in particular, allows for the facilitated retrieval, storage and use of the information contained in these databases, as the data on the food items stored in these is standardized under the framework.

Chapter 3

Information Extraction System Development

In this chapter, a detailed description of the development of the system to solve the problems outlined in Section 1.2 is presented. The chapter is divided into three main sections: **methodology**, **technologies**, **system architecture** and **annotation**.

For the **methodology** section, a brief analysis on the different methods implementable on the information extraction activity being approached is presented, alongside a description of the chosen approach for each individual information extraction task.

For the **technologies** section, the technologies throughout the system's development are described.

For the **system architecture** section, the system's flow and components are outlined, with each being characterized in its own individual subsection.

For the **annotation** section, an overview of the annotation process performed on a set of 100 recipes, in order to allow for the proper testing and validation of the system, is provided.

3.1 Methodology

The problems being approached in this project relate to an information extraction activity (composed by four individual tasks), with the goal of extracting, structuring and associating information pertaining to each ingredient in a food recipe (e.g. name, quantity, units, applied cooking method), and the calculation of the recipe's nutritional values, with the aid of the results from the aforementioned activity alongside the use of a food composition database.

It's possible to approach each of the tasks involved in the information extraction activity differently, as each has its own independent set of applicable techniques. As such, the methods chosen per task for the activity at hand were analyzed separately, in order to find the most adequate techniques for each.

The best method through which to use the information obtained on each ingredient and find the best match for it in the food composition database was also individually analyzed, as the task is separate from the information extraction activity.

Segmentation

The segmentation task in the project is divided into two distinct parts: The segmentation of the recipe at a sentence and word level, and the lemmatization of each word in the recipe (which is particularly useful for the classification task, since it allows for simplified comparisons between verbs in different tenses, singular and plural forms of a noun, etc.).

Segmentation at a sentence and word level of food recipes is not a trivial exercise. Food recipes are composed almost entirely by imperative sentences, which poses some difficulties in parsing that are not present in texts composed by general domain sentences [HMMT11]. The ideal method applicable to the task would be the use of state-of-the-art machine learning segmentation algorithms for each of the parsing tasks, trained on an extensive dataset composed almost exclusively by annotated imperative sentences of the food recipe domain. However, given that such a dataset is not available, the use of state-of-the-art tokenizers built for general domain sentences should suffice, and provide reasonable, even if not ideal, performance.

Relatively to the lemmatization portion of the task, a lemmatizer specifically built for the English language (since every recipe in the dataset is written in English) should be sufficient to properly lemmatize the recipe's text.

Classification

The classification task for the project is twofold, as it is necessary to not only classify different segments relatively to the domain of the information extraction activity (e.g. identify words as cooking utensils) but also to their grammatical properties (part-of-speech tagging), since these properties provide important pointers for the lemmatization of the recipe's text as well as the association portion of the activity.

Similarly to the segmentation task, state-of-the-art part-of-speech taggers currently face some difficulties when tagging on imperative sentences [HMMT11]. The ideal approach, as such, would be the use of state-of-the-art machine learning solutions for each of the classification subtasks, trained with an annotated dataset from the food recipe domain. Since such a dataset does not exist, and the dataset chosen for the project is not annotated, a rule-based approach to the extraction of information pertaining to the domain alongside the use of a part-of-speech tagger trained on annotated general domain sentences should provide reasonable performance, as well as be less time consuming to implement.

Association

The association task for the project relates to the identification of associations between ingredients and a used cooking method and food preparation techniques.

In order to derive these associations, the ideal methodology is that of a rule-based approach. Through this methodology, the rules are identified and developed (either manually or through the use of a rule-based machine learning algorithm) to allow for the correct extraction of the different associations in the domain. Given that, as was the case with the previously analyzed tasks, the dataset used for this project is not annotated, the rules will have to be manually developed in order to extract the associations relative to the task.

Normalization and Co-Reference Resolution

The normalization and co-reference resolution task in this project is composed by two different parts: The conversion of the units of each ingredient's quantity into an unique, standard unit and the resolution of co-reference cases where ingredients are referred to by different expressions, generally after a cooking method or food preparation technique is applied to these (e.g. "cheese" and, after cutting the cheese, "cheese slices").

The methodology to be adopted for the normalization of the ingredient's units, as previously mentioned in Section 2.1.4, is the use of conversion rules. Generic conversion rules for ingredient quantities can be found online [Sta] as well as in published books [BCc14]. These conversion rules can be easily adapted to the dataset's characteristics.

The ideal methodology for the resolution of co-reference cases in food recipes would be the use of either a rule-based or machine learning algorithm built specifically for the food recipe domain. However, since building such an algorithm would be very time-consuming, and need an extensive annotated dataset from the domain (which is currently not available), the adaptation of a machine learning co-reference resolution system trained on general domain sentences to the domain should provide both adequate and fast results.

Recipe Nutritional Values

The calculation of a recipe's nutritional values can be done through the use of the information extracted from the previously analyzed tasks in this section in conjunction with the use of a food composition database.

In order to calculate the nutritional values, conversion rules must first be applied to the extracted ingredients' units so that the calculations are all done in a single, standard unit, as described in the previous subsection.

The selection of the best match from the food composition database entries with the extracted ingredient should be done through the use of the LanguaL information of the entries in conjunction with a full-text search. LanguaL information contains important pointers for this selection, namely, each entry's associated cooking method (if applicable) and food preparation techniques (if applicable). This information can be compared to the information obtained from the previously described tasks in this section for the extracted ingredient.

As such, an heuristic approach is the most adequate technique applicable to the calculation of a recipe's nutritional values, taking into consideration all of the ingredient's extracted characteristics in order to find the best match on the food composition database.

3.2 Technologies

Past the analysis on the methodology applied in the system's development, the technologies used throughout the development phase were picked. Both the technologies applied in the system as well and the annotation portion of the system's development are described in the following subsections.

System

The Python programming language was picked to develop this project. The language's ease of use, gentle learning curve and extensive support from the NLP community make it ideal for use in information extraction projects that are not overly complex.

From the toolkits available in Python, the Natural Language Toolkit (NLTK) was picked. As previously discussed in Section 2.1.5, the toolkit offers a plethora of corpora, models and techniques widely used in NLP, which makes it an excellent candidate for the project at hand, given the extensive NLP tasks approached in the system. Additionally, the NLTK supports the use of other NLP toolkits and pipelines that allow for a simplified integration of these with any project developed with the toolkit.

In addition to the NLTK, the Stanford CoreNLP pipeline was used, in particular, its part-of-speech tagger and dependency parser. Given that the default part-of-speech tagger used by the NLTK struggled with properly tagging the majority of the imperative sentences that compose a food recipe (e.g. the verb starting the sentence would often be identified as a noun, adjectives would be identified as nouns), the part-of-speech tagger was changed to Stanford CoreNLP's one, which was trained with a model that has a more significant proportion of imperative sentences in its corpora [TKMS03]. While the precision of each part-of-speech tagger was not compared for this project (due to time and resource constraints), a brief analysis of the tagged sentences indicated that CoreNLP's part-of-speech tagger performed generally better than NLTK's solution. The CoreNLP's dependency parser was used since the NLTK does not offer any dependency parser and, given that the pipeline's part-of-speech tagger was already picked for this project's development, the parser was used for the sake of simplifying the system's structure and technological dependencies.

Annotation

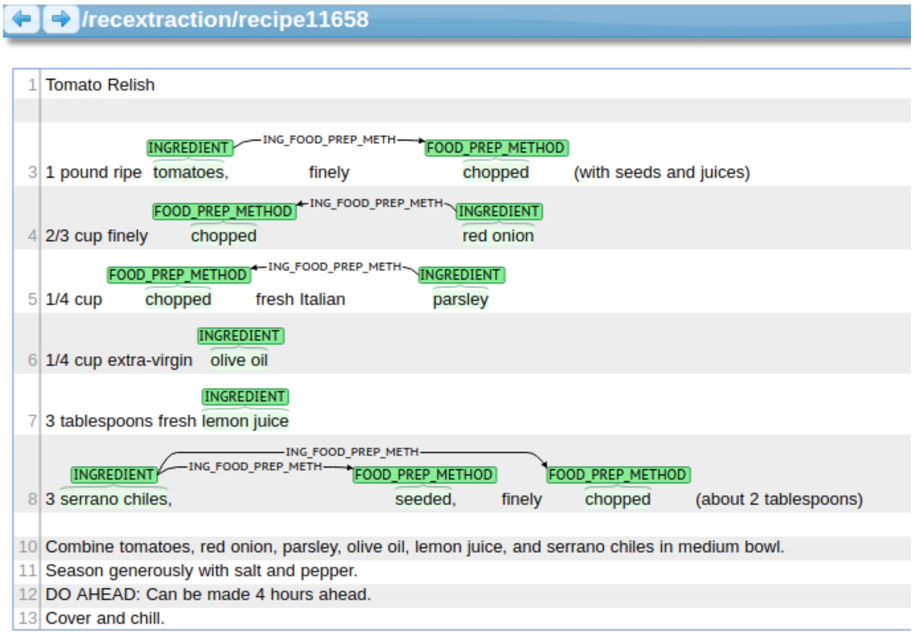
The software used for the annotation process was the brat annotation tool [1], a web-based tool for general-purpose text annotation.

[1] http://brat.nlplab.org/

In order to start the annotation process, it was first necessary to configure the server in accordance with the annotation framework defined by the user. Each annotation task is defined by a set of *entities*, *relations*, *events* and *attributes* tags, which can be individually adjusted by the user to fit its annotation needs.

The system accepts plain text files as input and outputs the document annotations in a standoff format, that is, in a separate file from the original file, identified by the .ann suffix. Each line in the file contains one annotation, and each annotation is associated to an ID that appears at the beginning of the line, separated from the rest of the annotation by a TAB character [SPT+12]. The rest of the structure varies by annotation type.

An example of the brat annotation interface is shown in Figure 3.1, and a sample of its output is shown in Figure 3.2.

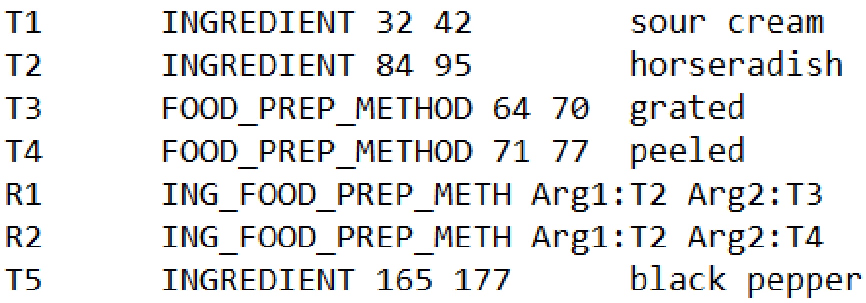


```
1  Tomato Relish

3  1 pound ripe tomatoes, finely chopped (with seeds and juices)
4  2/3 cup finely chopped red onion
5  1/4 cup chopped fresh Italian parsley
6  1/4 cup extra-virgin olive oil
7  3 tablespoons fresh lemon juice
8  3 serrano chiles, seeded, finely chopped (about 2 tablespoons)

10 Combine tomatoes, red onion, parsley, olive oil, lemon juice, and serrano chiles in medium bowl.
11 Season generously with salt and pepper.
12 DO AHEAD: Can be made 4 hours ahead.
13 Cover and chill.
```

Figure 3.1: The brat annotation tool interface.



```
T1      INGREDIENT 32 42        sour cream
T2      INGREDIENT 84 95        horseradish
T3      FOOD_PREP_METHOD 64 70  grated
T4      FOOD_PREP_METHOD 71 77  peeled
R1      ING_FOOD_PREP_METH Arg1:T2 Arg2:T3
R2      ING_FOOD_PREP_METH Arg1:T2 Arg2:T4
T5      INGREDIENT 165 177      black pepper
```

Figure 3.2: A sample of a food recipe's annotations output through by the brat annotation system.

3.3 System Architecture

After both the methodology and technologies relative to the project's development were picked, a design for the system's architecture was developed, taking into account as well the findings of the preliminary analysis performed for the used dataset, as described in Section 2.3.2. The technologies used in the system, its pipeline and the components that compose the system are presented in the following subsections.

3.3.1 Pipeline

The system's pipeline is shown in Figure 3.3. The system can be divided into seven major components, each with its own purpose:

- **Sentence and word tokenization** of each recipe's title and instructions fields;
- **Part-of-speech tagging and lemmatization** of each word in the title and instructions fields of the recipe;
- **Extraction and structuring of the name, quantity and units of each ingredient** in the recipe's ingredient field through the use of a Conditional Random Field model;
- **Extraction and structuring of the used cooking methods, food preparation techniques and cooking utensils** of each recipe in the dataset through a rule-based system;
- **Dependency parsing** of each recipe's title and instructions field;
- **Extraction of associations between ingredients and an applied cooking method and food preparation techniques** of each recipe in the dataset through a rule-based system;
- **Calculation of the recipe's nutritional values** through the use of the aforementioned extracted information in conjunction with a food composition database;

Each of the components of the system is detailed in the following subsection.

3.3.2 Components

Sentence and Word Tokenization

The first component of the system relates to the sentence and word parsing process of the segmentation task relating to the information extraction activity.

In order to properly tokenize each recipe in the dataset, the recipe's title and instructions fields were each stored in a Python list. Then, for the list containing the title field, the NLTK's default word segmentation algorithm was applied, while for the list containing the instructions field, both the NLTK's default sentence and word segmentation algorithms were used (since the instructions field is composed by multiple different sentences, while the title field is composed by a single one). As previously mentioned in Section 3.1, these are not the ideal sentence and word segmentation

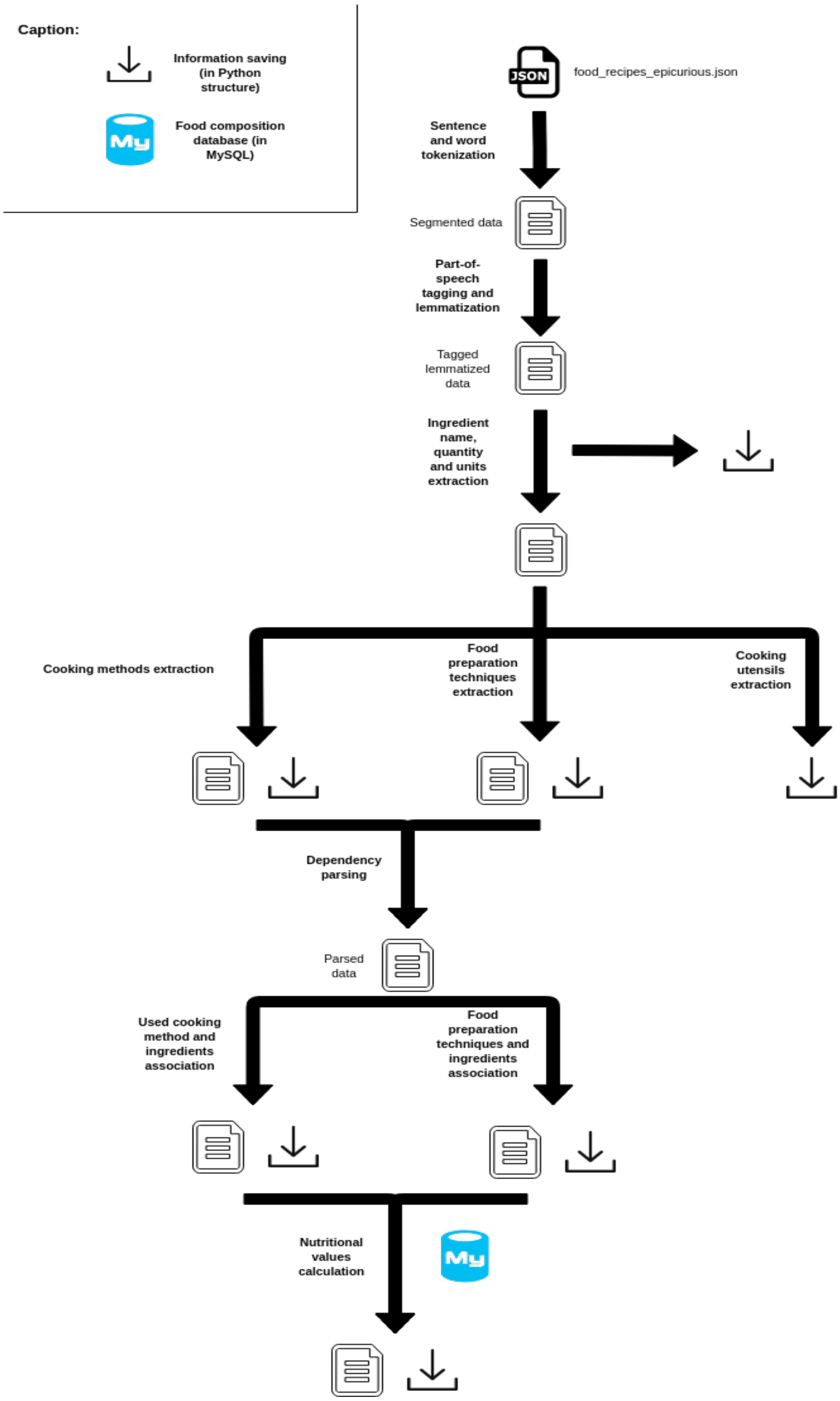

Caption:
Information saving (in Python structure)
My
Food composition database (in MySQL)
JSON
food_recipes_epicurious.json
Sentence and word tokenization
Segmented data
Part-of-speech tagging and lemmatization
Tagged lemmatized data
Ingredient name, quantity and units extraction
Cooking methods extraction
Food preparation techniques extraction
Cooking utensils extraction
Dependency parsing
Parsed data
Used cooking method and ingredients association
Food preparation techniques and ingredients association
Nutritional values calculation
My

Figure 3.3: The pipeline of the system developed, supported by the technologies referred to in Section 3.2.

algorithms that could be applied to a food recipe's text, but the lack of annotated data of the domain makes it impossible to apply better algorithms.

The default sentence segmentation algorithm used in the NTLK is an adaptation of the Punkt system, an unsupervised multilingual sentence boundary detection algorithm that uses information such as the detection of abbreviations, initials, ordinal numbers, etc. in order to identify and separate written text into different sentences [KS06]. A diagram of the algorithm's architecture is presented in Figure 3.4.

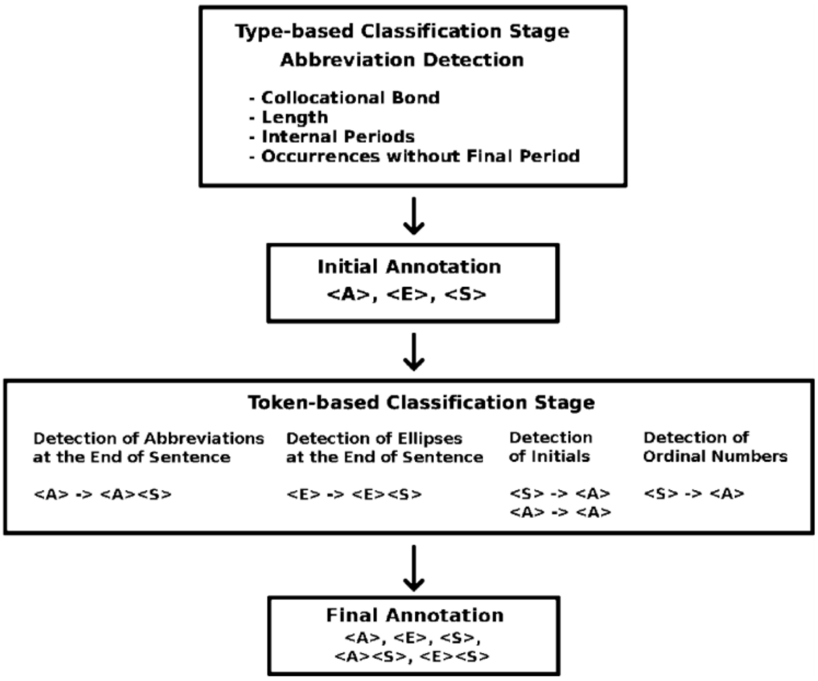


Figure 3.4: System architecture of the Punkt sentence segmentation algorithm [KS06].

The default word segmentation algorithm used in the NLTK is an adaptation of the Penn Treebank tokenizer, which uses regular expressions in order to tokenize text. The tokenizer presents the following characteristics [McI97]:

- Each word is considered an individual token if separated by whitespaces;
- Standard contractions are split;
- The majority of punctuation characters is separated into individual tokens;
- Commas and single quotes are split off, if these are followed by whitespaces;
- Periods are separated into an individual token if they appear at the end of a line.

Part-of-Speech Tagging and Lemmatization

The second component of the system relates to the part-of-speech tagging and lemmatization of the written text in a food recipe.

While, in general, every part of the segmentation task is performed before the beginning of a classification task, this is not the case for this information extraction activity. In order to lemmatize a word, it is necessary to know the context in which it is inserted to ensure its proper lemmatization. One way of obtaining information about this context is through the use of a part-of-speech tagger, which tags a word according to its grammatical role in a sentence. This allows the lemmatizer to derive the correct lemma for homograph words through their context (e.g. in the sentences "Vince Staples is a loving person." and "Vince Staples is loving this!", in the first sentence, the lemma for the word "loving" would be "loving", since it's an adjective, while in the second sentence it would be "love", since it's the present continuous conjugation of the verb "love").

The part-of-speech tagger used was the Stanford log-linear part-of-speech tagger implemented in the CoreNLP pipeline. The tagger is a machine learning maximum entropy-based system that was trained on the Wall Street Journal section of the Penn Treebank [TKMS03]. The model was later updated to cover cases not included in the original corpus (e.g. new "tech" words, imperatives) [Gro]. As previously mentioned in Section 3.2, while the part-of-speech tagger will not perform optimally (since the majority of its corpus is not composed by imperative sentences), it should still nonetheless perform better than similar taggers, since its model was specifically updated to include a significant amount of imperative sentences.

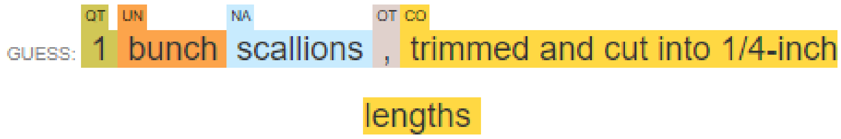
The lemmatizer used was the WordNet lemmatizer, implemented in the NLTK. The lemmatizer uses WordNet, a lexical database in the English language which includes nouns, verbs, adjectives and adverbs, interlinked by means of conceptual-semantic and lexical relations [Kil00]. The NLTK makes use of WordNet's morphological function which, when given a part-of-speech tag and word, returns the lemma of the word in accordance with the tag.

3.3.2.1 Ingredient Name, Quantity and Unit Extraction

The third component of the system relates to the information extraction relative to the name, quantity and units of each ingredient in the ingredients field of a food recipe.

The extraction of these three different characteristics of an ingredient was performed using a linear-chain Conditional Random Field model developed by Green et al. at The New York Times[2] [Gre15]. The model was developed with the goal of predicting a correct sequence of tags (each tag is either a *NAME*, *UNIT*, *QUANTITY*, *COMMENT* or *OTHER*) for an ingredient phrase (e.g. "A pinch of salt", "2 1/2 cups of sugar") in a food recipe , even if the model has never seen the ingredient phrase before [Gre15]. As such, the model allows for the extraction and structuring of an ingredient's name, quantity, unit and comments written by the author from unstructured ingredient lines. An example of the output of the CRF model for an ingredient phrase is shown in Figure 3.5.

[2] https://www.nytimes.com/



GUESS: 1 bunch scallions , trimmed and cut into 1/4-inch lengths

Figure 3.5: An example of the output of the New York Times CRF model for an ingredient line [Gre15].

For this project, the model was trained on 130,000 examples of ingredient lines, manually annotated by overnight contractors at the New York Times. Given that the structure of the ingredient phrases in the project's dataset and the ones annotated by the New York Times are deeply similar, the examples used for training should allow for optimal performance of the model for the dataset used.

3.3.2.2 Recipe Cooking Actions and Utensils Extraction

The fourth component of the system relates to the extraction and structuring of the cooking actions, which include the used cooking methods and food preparation techniques, and utensils used in a food recipe.

In order to accomplish these goals, a rule-based approach was chosen for this task. Given that, as previously mentioned in Section 3.1, there was no available annotated data of the domain that could be used for this extraction, a statistical approach was not possible. As such, rules were manually created which, in conjunction with a dictionary built for the domain, allow for the extraction of the cooking actions and utensils of a recipe.

The dictionary created for the task makes use of multiple different sources [MAB10, Kip12, CCH01] and includes the most generally used cooking methods, food preparation techniques and cooking utensils in Western cuisine.

Relatively to the manually developed rules, for the cooking actions portion of the extraction, two lexical rules were created. The rules are as follows:

- If a word in the recipe's text is tagged as a **verb**, the word's **lemma is in the lexicon**, and the word **is not** already associated to the recipe as a cooking action, the word is classified as a new cooking action, and it's associated to the recipe;

- If a word's **non-lemmatized form is in the lexicon**, and the word **is not** already associated to the recipe as a cooking action, the word is classified as a new cooking action, and it's associated to the recipe.

The extraction of the cooking utensils was more complex, since it was not only necessary to search for the presence of the utensil in the recipe, but also for any cooking actions that imply the use of an utensil (e.g. the verb "cut" implies the use of a knife). As such, several rules were manually created for these cooking actions, in which, if the cooking action is present in the recipe, the cooking utensil implied by the action is associated to the recipe. Some of the cooking actions and the cooking utensils associated to these are shown in Table 3.1.

Rules

Cooking actions	Cooking utensil extracted
Cut, halve, quarter, slice	knife
Grill, barbecue	grill
Char, broil	broiler

Table 3.1: Some of the cooking actions and the cooking utensils these actions are associated to, for the extraction of cooking utensils component of the system.

In addition to these rules, two lexical rules were created. The rules are as follows:

- If the word in the recipe's text is tagged as a **noun**, the word's **lemma is in the lexicon**, and the word **is not** already associated to the recipe as a cooking utensil, the word is classified as a cooking utensil and associated to the recipe;

- If the word's **non-lemmatized form** is **in the lexicon**, and the word **is not** already associated to the recipe as a cooking utensil, the word is classified as cooking utensil and associated to the recipe.

The presence of the second lexical rule in both parts of the extraction task is justified by the fact that, as pointed out in Section 3.1, the part-of-speech tagging is not always reliable, and it would often associate the incorrect tag to a word. As such, this rule helps improve the overall recall of the system by immediately associating the cooking action or utensil to a recipe if it appears on the text as it does on the lexicon, at the cost of a small amount of precision.

Dependency Parsing

The fifth component of the system relates to the dependency parsing of the sentences in the written text of a food recipe.

The dependency parsing of written text is an activity where a dependency grammar is applied to the sentences of the text. In this grammar, "the syntactic structure of a sentence is described solely in terms of the words (or lemmas) in a sentence and an associated set of directed binary grammatical relations that hold among the words" [Jur00]. Syntactic information is essential to the extraction of associations in written text, since these are generally described through the language syntax. As such, dependency parsing is particularly useful for this project since, through it, it's possible to develop rules for the extraction of associations in food recipes that make use of the syntactic information available.

The dependency parser used was the Stanford dependency parser implemented in the CoreNLP pipeline. The parser is a greedy, transition-based implementation using an underlying neural network classifier [SM16]. In order to extract the different syntactic relationships between the different words of a sentence, every word needs to be tagged for its part-of-speech information, which can be done through the use of any part-of-speech tagger. For this project, the part-of-speech tagger described in Section 3.3.2 is used. An example of the parser's expected output for a given sentence is shown in Figure 3.6.

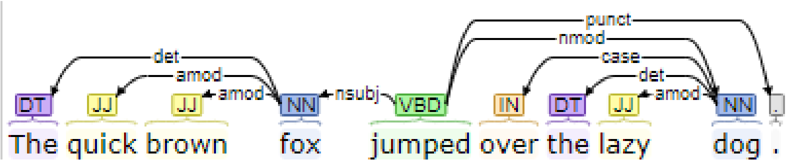


Figure 3.6: An example of the output of the Stanford dependency parser for the sentence "The quick brown fox jumped over the lazy dog.", showing the syntactic analysis made by the parser.

3.3.2.3 Ingredients Associations Extraction

The sixth component of the system relates to the extraction of the associations between the ingredients of a food recipe and the used cooking method and food preparation techniques relative to each.

As previously mentioned in Section 3.1, a rule-based system was the chosen approach for the extraction of the associations in a food recipe. A brief analysis of the syntactic structure of the instructions field of the recipes from the dataset showed that it was possible, theoretically, to achieve a fairly strong performance in the association task through the use of a few handwritten rules.

For the association of an ingredient to a used cooking method in a food recipe, it was possible to note that, for the large majority of cases, the used cooking method for an ingredient was the first cooking method mentioned in the instructions after the ingredient is initially mentioned. An example of this case is as follows:

1. Blend the chicken, cauliflower and broccoli in a bowl.

2. Transfer the mixture onto a saucepan, and sauté it from 5 to 10 minutes.

For this particular case, the *chicken*, *cauliflower* and *broccoli* ingredients would be associated to the cooking method *sauté*, which would be the correct association. Taking into consideration these cases, a rule was put in place to handle these cases, which is complemented by two other handwritten rules to extract other associations which are not covered by this rule. These rules are as follows:

- If a word in the recipe's title corresponds to an extracted **cooking method**, and the word has either a **compound** or **adjectival modifier** relationship with a word that matches an extracted **ingredient** (e.g. "braised beef", "fried chicken"), the cooking method is associated to the ingredient;

- If a word in the recipe's instructions corresponds to an extracted **cooking method**, and the word has either a **direct object** or **nominal modifier** relationship with a word that matches an extracted **ingredient** (e.g. "Cook the chicken", "Fry the scallops and eggs"), the cooking method is associated to the ingredient;

- If a word in the recipe's instructions corresponds to an extracted **cooking method**, associate the cooking method to any **ingredients** that have been previously mentioned in the instructions and **do not have** a cooking method already associated to them.

It's worth pointing out as well that, for each ingredient, only a single cooking method is associated to it. While an ingredient can have multiple cooking methods applied, generally, the most nutritionally relevant cooking method is the first one applied.

Certain ingredients were excluded from being associated to a cooking method, which include herbs (e.g. cilantro, thyme, basil), seasonings (e.g. salt, pepper, sugar) and certain condiments (e.g. mustard, ketchup), since the nutritional information of these ingredients does not alter significantly with the used cooking method. A dictionary of excluded ingredients was created in order for the system to not associate a cooking method or food preparation techniques to these.

For the association of an ingredient to food preparation techniques used in a food recipe, the majority of the associations were extracted directly from the recipe's ingredients list. An ingredient phrase is usually composed by the ingredient's quantity, units, name and food preparation techniques applied to it. Additionally, similarly to the associations present between an ingredient and cooking method, it was possible to note that a food preparation technique was usually applied to an ingredient if the ingredient was mentioned in the recipe's text closely after the food preparation technique. Referring back to the aforementioned example for the association between an ingredient and used cooking method, the food preparation technique *blend* would be associated to the ingredients *chicken*, *cauliflower* and *broccoli*, which would be a correct association.

Taking into consideration these cases, three rules were manually developed for the extraction of associations between ingredients food preparation techniques. These rules are as follows:

- If a word in a recipe's ingredient phrase corresponds to an extracted **food preparation technique**, the technique is associated to the **ingredient** extracted from the phrase;

- If a word in the recipe's instructions corresponds to an extracted **food preparation technique**, and the word has either a **direct object** or **nominal modifier** relationship with a word that matches an extracted **ingredient** (e.g. "Cut the chicken breast", "Blend the scallops and eggs"), the technique is associated to the ingredient;

- If a word in the recipe's instructions corresponds to an extracted **food preparation technique**, associate the technique to any **ingredients** that are mentioned after the technique for in the same sentence or until another **food preparation technique** is mentioned.

3.3.2.4 Recipe Nutritional Values

The seventh and final component of the system relates to the calculation and structuring of a recipe's nutritional values.

This component makes use of a food composition database hosted at Fraunhofer Portugal AICOS, described in Section 2.3.3. The calculation of each recipe's nutritional information is done at an ingredient level, that is, each ingredient's individual nutritional values are first calculated, and then added to make up the recipe's nutritional values.

In order to do these calculations, the following steps are executed:

1. The ingredient's units are converted into a standard format (milligrams) through the use of handwritten conversion rules, based on different sources [MAB10, Kip12, CCH01];

2. A full-text search using the ElasticSearch[3] search engine connected to the database is done, with the query corresponding to the ingredient's extracted name;

3. The LanguaL descriptors of the five best matches for the query are then iterated on. The descriptors contain information relative to the database entry's cooking method and food preparation techniques, which is compared to the extracted cooking method and food preparation techniques associated to the ingredient, in order to find the best database entry match for the ingredient;

4. The nutritional values of the ingredient are then obtained, taking into consideration the ingredient's quantity and units.

Unfortunately, this calculation proved impossible for some ingredients due to a lack of information in the database. The units for ingredients in the liquid form in food recipes are almost always volumetric, while the units for the entries in the database were strictly weight based. Given that the conversion of volume to weight units is entirely dependent on the ingredient (e.g. 500ml of water have a different weight than 500ml of olive oil), and creating handwritten conversion rules for each of the different ingredients in liquid form would be significantly time consuming, the portion of the component relative to the calculation of the nutritional values for liquid ingredients was not completed.

3.4 Annotation

The annotation of a set of food recipes was necessary for the proper testing and validation of the system, as otherwise there would be no ground truth to which one could compare and validate the system's results.

The annotation framework and a description of the annotation process are presented in the following subsections.

[3] https://www.elastic.co/

Framework

For the annotation's framework, four different types of entity tags were defined, which are as follows:

- *INGREDIENT*: The entity is used to denote sequences of a recipe where food ingredients are involved, namely in the ingredients section of the recipe. Food items that are referred to exclusively in the instructions section are not assigned as *INGREDIENT* entities;
- *COOKING_METHOD*: The entity is used to denote sequences of a recipe where a cooking method is involved. Cooking methods are assigned as *COOKING_METHOD* entities in every section of the recipe (title, ingredients list and instructions);
- *FOOD_PREP_METHOD*: The entity is used to denote sequences of a recipe where a food preparation method is involved. Food preparation methods are assigned as *FOOD_PREP_METHOD* entities in every section of the recipe (title, ingredients list and instructions);
- *COOKING_UTENSIL*: The entity is used to denote sequences of a recipe where a cooking utensil is involved. Cooking utensils are assigned as *COOKING_UTENSIL* entities in every section of the recipe (title, ingredients list and instructions).

In addition to the entities defined, two relation tags were created, which are characterized as follows:

- *ING_COOK_METH*: The relation is used to associate an *INGREDIENT* entity to a *COOKING_METHOD* entity, and represents the relationship between the ingredient and the cooking method applied to it;
- *ING_FOOD_PREP_METH*: The relation is used to associate an *INGREDIENT* entity to a *FOOD_PREP_METHOD* entity, and represents the relationship between the ingredient and a food preparation method applied to it.

No events or attributes tags were defined for this framework.

Process

An initial selection of 100 recipes from the dataset was done for the annotation task. Due to a lack of relevant categories associated to each recipe (the categories associated to the recipes in the dataset are user-inputted, and offer little information on the type of dish described by the recipe) with which an heuristic could be developed so that the test set covered an amount of types of food recipes proportional to that of the dataset, the selection of the recipes to annotate was done through simple random sampling.

Prior to the annotation process, a few guidelines were developed to help streamline the process and help its cohesion. These guidelines are as follows:

- Seasonings and sauces are not considered to have a cooking method associated to them (e.g. salt, pepper, sugar, honey, vinegar, olive oil, lemon juice), as this information does not alter significantly the nutritional composition of these types of ingredients;
- An action is only considered a food preparation technique if it is applied prior to the cooking method (e.g. when boiling a chicken breast and cutting it afterward, the cutting is not considered a food preparation technique, unless another cooking method is applied to it afterwards);
- Ingredients should only be associated to a single cooking method. In the event that more than one cooking method is applied to the ingredient, the first used cooking method is associated to it, as it is often the most nutritionally relevant cooking method;
- For a cooking method, should it appear more than once in a food recipe, only its first mention should be annotated;
- Ingredients can have multiple food preparation techniques associated to them;
- If a specific cooking method is not explicitly present in the recipe and it is, instead, implied (e.g. "Cook the bacon at medium heat in the skillet" implies the cooking method "fry"), no cooking method should be annotated or associated to an ingredient;
- Variants of an ingredient (e.g. "extra-virgin olive oil", "kosher salt") must be annotated as the same ingredient, in its base form, unless its variants have a substantially different nutritional composition;
- Variations of a food preparation technique (e.g. "thinly sliced", "coarsely sliced") must be annotated as the same food preparation technique, in its base form (e.g. for the previous example, "sliced" would be the base form);

The annotation process consisted of six steps, which were as follows:

1. The *INGREDIENT* entities were identified and tagged;
2. The *FOOD_PREP_METHOD* entities were identified and tagged;
3. The *ING_FOOD_PREP_METH* associations were created between the appropriate *INGREDIENT* and *FOOD_PREP_METHOD* entities;
4. The *COOKING_METHOD* entities were identified and tagged;
5. The *ING_COOK_METH* associations were created between the appropriate *INGREDIENT* and *COOKING_METHOD* entities;
6. The *COOKING_UTENSILS* entities were identified and tagged;

Given that the annotation of the dataset was done solely by myself, I did not perform any sort of inter annotator agreement score calculation. However, my annotations were informally revised by a colleague at Fraunhofer Portugal AICOS, as to try to guarantee a higher level of robustness to my annotations.

3.5 Summary

In this chapter, the development of the information extraction system is detailed, focusing on the methodology applied, the technologies used, the system's architecture and the annotation process applied.

Given that the the solution to the problems approached in the system involves an information extraction activity, which can be divided into different tasks, and the calculation of nutritional values, different approaches were required and selected for each of the tasks, taking into consideration the characteristics of the dataset and the problems approached.

The technologies used for the project were analyzed and chosen in accordance with their ease of use, documentation available and the quality of the technology itself.

An analysis of the system's architecture, which involves the system's pipeline and the different components of the system (each of which is further characterized in its own section), brings additional clarity on the system's functionality.

Finally, the annotation process, necessary for the testing and validation of the system, is detailed in this chapter, which includes the framework designed for the process, and the guidelines and steps taken for the annotation of the different entities and relations for 100 randomly sampled food recipes.

Chapter 4

Testing and Validation

4.1 Introduction

In this chapter, a description on the testing and validation of the system is provided. The chapter is divided into four main sections: **experiments design**, **testing set analysis**, **results** and **discussion**.

For the **experiments design** section, an analysis on the experiments conducted and the metrics selected to get the most complete understanding of the system's overall performance is presented.

For the **testing set analysis** section, the set's characteristics are outlined and compared to the those of the full dataset, previously presented in Section 2.3.2.

For the **results** section, the results of the components specifically developed for this project are presented, in accordance with the metrics chosen.

For the **discussion** section, an analysis on the system's results and the major influences in its performance are presented.

4.2 Experiments Design

In order to properly test and validate the system developed, two experiments were designed, taking into consideration the characteristics of each of the components tested.

The first experiment consisted in the calculation of the *precision*, *recall* and *F-measure* achieved for each of the parts of the system developed from the ground up relating to the classification and association tasks. The performance of each of these tasks is compared to a baseline specific to the task. The formulas used for the calculation of these metrics are, however, different for some of the components of the system. This happens because the extraction of the cooking actions and utensils in a food recipe, as well as the association of each ingredient to different food preparation techniques, were framed for this experiment as *multilabel classification* problems, where a sample (in this case, a food recipe for the extraction tasks, and an ingredient for the classification task) can be assigned multiple different labels (e.g. different cooking methods for a food recipe or food preparation techniques). However, the association of an ingredient to multiple food preparation techniques was framed as a *multiclass classification* problem, where each sample (in this case, each ingredient) of a recipe is assigned to a single class (i.e. an ingredient can't be associated to more than a single cooking method). As such, the different metrics involved in the experiment have unique formulas associated to them, depending on the type of problem being approached. A brief description of the metrics and their respective calculations is as follows:

- For *multiclass classification* problems, the **precision of the system** is obtained by averaging the **precision of each individual class**, weighted in accordance with the number of elements of each class. For each class, precision is calculated as the number of true positives (i.e. number of samples correctly labeled as belonging to the class) divided by the total number of samples labeled as belonging to the class (i.e. the sum of true and false positives, the latter of which refers to samples incorrectly labeled as belonging to the class). For *multilabel classification* problems, the **precision of the system** is calculated as "the proportion of predicted correct labels to the total number of actual labels, averaged over all instances" [Sor];

- For *multiclass classification* problems, the **recall of the system** is obtained by averaging the **recall of each individual class**, weighted in accordance with the number of elements of each class. For each class, recall is calculated as the number of true positives divided by the total number of samples that actually belong to the positive class (i.e. the sum of true positives and false negatives, the latter of which refers to samples which were not labeled as belonging to the class when they should have been). For *multilabel classification* problems, the **recall of the system** is calculated as "the proportion of predicted correct labels to the total number of actual labels, averaged over all instances" [Sor];

- For both the *multiclass classification* and *multilabel classification* problems, the **F-measure of the system** is calculated identically. It is the harmonic mean of the precision and recall and serves as a measure of a test's accuracy.

For the validation of the calculation of a recipe's nutritional values, given the limitations of the component discussed in Section 3.3.2.4, and that annotating the nutritional information for each ingredient of a recipe would be very time-consuming, an experiment was designed where, for twenty random extracted ingredients whose extracted information was manually validated, the best database entry picked for the ingredient by the system was compared to one manually annotated as the most appropriate corresponding entry to the ingredient in the database. This experiment provides a solid testing and validation foundation for the component's results, while being significantly simpler and less time-consuming to implement than other alternatives.

4.3 Testing Set Analysis

The testing set was analyzed in order to better determine its most relevant characteristics, specifically when compared to those of the primary dataset.

The test set is composed by 100 annotated food recipes, in the format described in Section 3.4. Its average number of words and, particularly, names and and verbs per recipe, was calculated and compared to the averages obtained for the dataset, shown in Section 2.3.2, as to better understand the composition of the written text of each recipe in the test set comparatively to the dataset as a whole. The information calculated is displayed in Table 4.1.

	Dataset		Test set	
	Mean	σ	Mean	σ
Words	203.03	132.90	183.62	100.80
Nouns	66.05	40.64	60.56	34.16
Verbs	24.52	17.51	22.37	13.45

Table 4.1: Mean and standard deviation of the number of words and, specifically, nouns and verbs for the food recipe dataset and the test set.

It's possible to denote, from a brief analysis of the table's content, that the dataset and test set only differ significantly in the number of words in each recipe, with each recipe in the dataset being, on average, twenty words longer than the recipes in the test set. The standard deviation is lower for the test set, which implies that the recipes in the set have a more similar total number of words.

The total, mean and standard deviation for each of the entities and relation annotated in the test set was also calculate. This information is shown in Table 4.2.

	Metrics		
Entities and relations	Total	Mean	σ
Ingredients	960	9.60	3.88
Cooking methods	146	1.46	1.21
Food preparation techniques	510	5.10	2.92
Cooking utensils	404	4.04	1.93
Ingredient and cooking methods associations	330	3.30	2.73
Ingredient and food preparation techniques associatons	577	5.77	3.40

Table 4.2: Totals, mean and standard deviation for the entities and relations annotated in the test set.

The calculations are about what was to be expected from the annotation. The majority of food recipes in the test set have around one to two cooking methods associated to these and, given that each of these is only annotated once (in its first appearance), the fairly low quantity of annotated cooking methods and associations between ingredients and the applied cooking methods is to be expected. The annotations referring to associations between ingredients and food preparation techniques and food preparation techniques having the highest total is also to be expected, as unique food preparation techniques appear more frequently and more than one technique can be associated to each ingredient.

Additionally, the twenty most common nouns and verbs were calculated for each recipe of the test set. This information is displayed, respectively, in Figures 2.3 and 4.2.

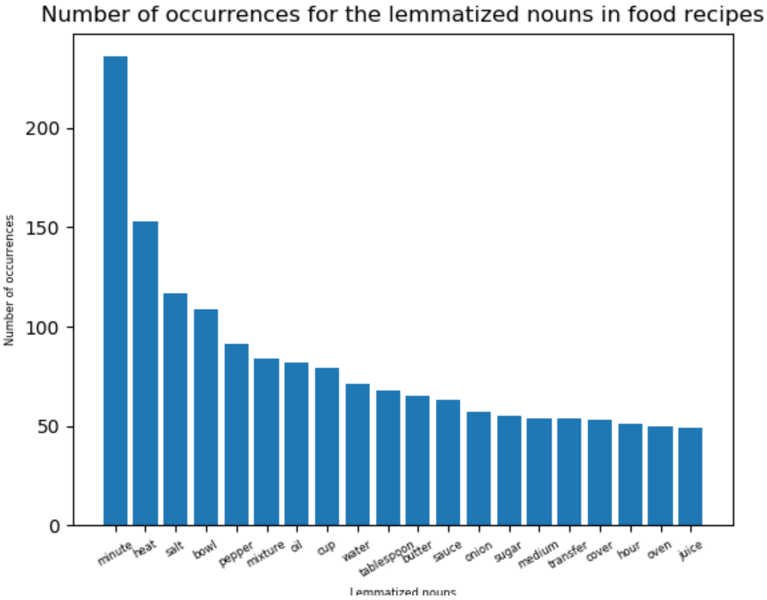


Figure 4.1: The twenty most common nouns present in the written text of the test set's food recipes.

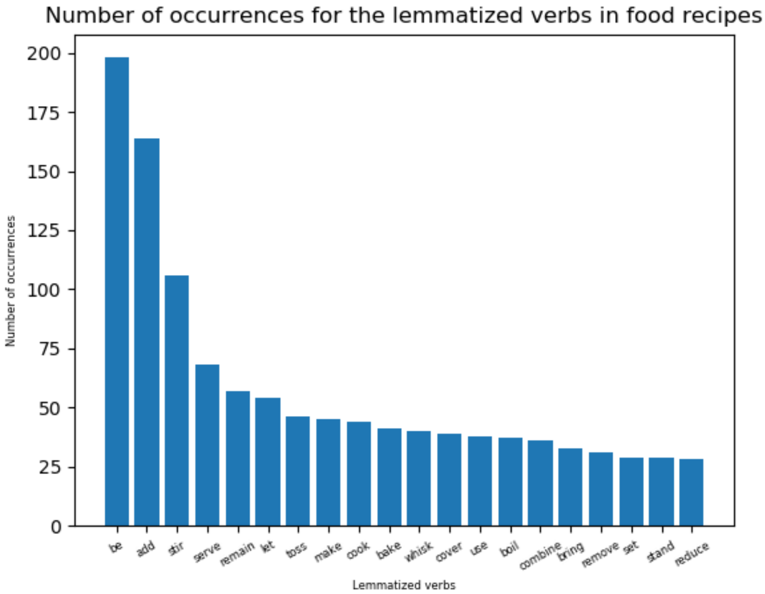


Figure 4.2: The twenty most common verbs present in the written text of the test set's food recipes.

Comparatively to the twenty most common nouns and verbs of the full dataset (shown in Figures 2.3 and 2.4, these are very similar, with a few differences in the order of the ranking and in the lower ranked nouns and verbs. This is expected due to the use of a random simple sampling technique, as well as the small sample of the test set in comparison with the full dataset. It's important to note, however, that it is not possible to draw any definitive evidence from the figures above in relation to the abundance of certain nouns or verbs in the recipe, as errors in the part-of-speech tagging process can cause these values to be warped in relation to the actual values. Instead, these figures serve merely as an indication of the presence of the most common nouns and verbs in the recipe.

4.4 Results

As to provide better context for the project's results, these are compared to a baseline, which is unique for each of the components developed. Utilizing a baseline for each of the components is particularly useful to compare how the different aspects of the component influence the results obtained, so as to have a more concrete idea on which aspect is more influential.

A description of the baseline used for the components and the results obtained is presented in the following subsections.

Cooking Actions and Utensils Extraction

The baseline used for the components of the experiments relating to the extraction of the cooking actions associated to a food recipe was the implementation of each of the components as described in Section 3.3.2.2, but without the use of any rules that attempt to correct mistakes made by the part-of-speech tagger used.

The baseline used for the component of the experiment relating to the extraction of the cooking utensils associated to a food recipe was the implementation of the component as described in Section 3.3.2.2, but without the use of any rules developed to extract components when a certain cooking action is used (e.g. the cooking utensil "knife" is associated to the recipe when the cooking action "cut" is present in the recipe).

The baseline's results and the results for the final implementation for the cooking methods, food preparation techniques and cooking utensils extraction tasks are shown, respectively, in Tables 4.3, 4.4 and 4.5.

	Metrics		
Implementation	Precision	Recall	F-Measure
Baseline	0.96	0.79	0.87
Final	0.90	0.95	0.92

Table 4.3: Results for the baseline and the final component, which uses a rule that attempts to lessen the impact of incorrect part-of-speech tags, for the cooking methods extraction component.

	Metrics		
Implementation	Precision	Recall	F-Measure
Baseline	0.79	0.87	0.83
Final	0.77	0.90	0.83

Table 4.4: Results for the baseline and the final component, which uses a rule that attempts to lessen the impact of incorrect part-of-speech tags, for the food preparation techniques extraction component.

	Metrics		
Implementation	Precision	Recall	F-Measure
Baseline	0.94	0.65	0.78
Final	0.94	0.92	0.93

Table 4.5: Results for the baseline and the final component, which uses rules to extract cooking utensils from cooking actions, for the cooking utensils extraction component.

Ingredient Associations Extraction

The baseline used for the experiments of the components regarding the extraction of associations between ingredients and a used cooking method as well as food preparation techniques was the implementation of the components as described in Section 3.3.2.3, but without the use of any dependency parsing information.

The baseline's results and the results for the final implementation of the association of each ingredient of a recipe to its applied cooking method and food preparation techniques are shown, respectively, in Tables 4.6 and 4.7.

It's important to note that, for the multiclass problem relating to the association of ingredients to their applied cooking method, the classes taking into considerations in this problem relate to the cooking methods in the dictionary built for the domain. These include "bake", "fry", "pan-fry", "stir-fry", etc.

	Metrics		
Implementation	Precision	Recall	F-Measure
Baseline	0.74	0.70	0.72
Final	0.77	0.73	0.74

Table 4.6: Results for the baseline and the final component, which uses information from dependency parsing, for the association of ingredients to its applied cooking method.

	Metrics		
Implementation	Precision	Recall	F-Measure
Baseline	0.97	0.89	0.94
Final	0.97	0.90	0.94

Table 4.7: Results for the baseline and the final component, which uses information from dependency parsing, for the association of ingredients to its applied food preparation techniques.

Recipe Nutritional Values

As previously mentioned in Section 4.2, the methodology used for the testing and validation of the component regarding the extraction of the nutritional values only takes into consideration if, from twenty ingredients with previously validated extracted information, the system is able to pick the best database entry pertaining to each ingredient. As such, the component's ability to correctly calculate a recipe's nutritional values is not what is being evaluated in this situation, but rather the component's capability to correctly pick the database entries. As a result, the component's normalization task is not tested nor validated.

The baseline used for this experiment was the best match resulting from the full-text search query corresponding to an extracted ingredient's name, without the use of any additional information.

The baseline's results and the results for the final implementation of the association of an ingredient with previously validated extracted information to a database entry are shown in Table 4.8.

Implementation	Number of correctly matched entries for 20 ingredients
Baseline	3
Final	8

Table 4.8: Number of correctly matched entries for twenty random ingredients with previously validated information relatively to the food composition database, for the component's cross-matching task.

4.5 Discussion

The results obtained for each of the components tested, are, for the most part, in line with what was to be expected. While the related work analyzed in Section 2.2 in the food recipe domain mostly focuses on the extraction of the ingredient's name, quantities and units (these extraction tasks were not tested for this dissertation, given that the system used for the extraction of these was not developed by myself), some of the systems analyzed which extracted actions and utensils achieved F-measures for the extraction of these around the 0.90 mark. However, these results are not directly comparable to those obtained by the system developed for this dissertation, as the actions extracted are much broader and not just specific to cooking actions. Additionally, it was not possible to compare the results obtained for the ingredients associations extraction with related works, as these either did not associate ingredients with a used cooking method or food preparation techniques, or only tried to extract more general associations (e.g. associating ingredients with any action in a food recipe, not separating these by types).

To properly understand the results obtained for each of the components tested, it's important to analyze the issues faced by the system both by each component and as a whole. The components of the system are intimately related and, as such, have problems that are general to some groups of components, but each also has its own set of specific issues. These issues are described in the following subsection.

4.5.1 Extraction Tasks Problems

The main problems that the extraction tasks face can be divided into three categories:

- Part-of-speech tagging errors;
- Annotation mistakes;

- Spelling mistakes.

Part-of-speech tagging errors

As previously mentioned in Section 3.1, part-of-speech taggers struggle to perform at an adequate level when tagging imperative sentences. Even though the tagger used had been trained on a model that, according to its documentation, contains a significant amount of imperative sentences [Gro], the tagger nonetheless struggled, and was often the source of mistakes in the extraction tasks. This is verifiable by the significant difference in the F-measure calculated between the baseline and the final implementation for the extraction of cooking methods component, which successfully employs a rule with the goal of reducing the impact of erroneous tagging by the part-of-speech tagger. The rule consists in the association of a cooking method to the food recipe if a word in the recipe's text is present in the dictionary created for the extraction tasks (which is composed by the most common cooking actions and utensils in their base forms), bypassing the verification of the word's tag. The presence of this rule improves the F-measure of of the component by 0.16, due to a significant increase in recall at the cost of slightly lower precision. Given that, for the extraction tasks, recall is a more important metric than precision (a higher recall allows for better results in the association tasks), this trade-off is worth it.

The mistakes made by the part-of-speech tagger are also at the core of the significantly lower precision achieved in the extraction of the food preparation techniques associated to the recipe when compared to the other extraction tasks. While the presence of the non-lemmatized words corresponding to cooking methods and utensils in the dictionary almost always implies the presence of these in a food recipe, the same is not true for food preparation techniques. An example of this would be the presence of an ingredient line in a recipe consisting of "1 slice of bacon". In this situation, the food preparation method "slice" should not be associated to the recipe. However, with the presence of the aforementioned rule, it would be erroneously associated.

This results in the fairly small difference in the F-measure calculated for the baseline and final implementation of the food preparation techniques extraction component. While the recall is slightly increased by 0.03, it leads to a small decrease in precision of 0.02, resulting in an equal F-measure.

Annotation mistakes

While the annotations pertaining to the test set of 100 food recipes were reviewed both by myself and a colleague at Fraunhofer Portugal AICOS, through an analysis of the results obtained for the system, it's possible to conclude that mistakes were made nonetheless during the annotation process. These range from the same cooking actions and utensils being annotated more than once in a food recipe (which is against the defined guidelines) to certain cooking actions and utensils not being properly annotated.

A superficial analysis of the individual results for the extraction components shows that, while the amount of annotation mistakes isn't considerable, it is nonetheless significant enough that it influences the results shown in Section 4.4.

Spelling mistakes

Spelling mistakes account for a marginal proportion of the errors present in extraction tasks. These include misspellings such as missing accents (e.g. "saute" instead of "sauté"), letters in the wrong order ("Bkae" instead of "Bake"), among others, and, as such, end up having an influence on the extraction tasks' results, as misspelled words will not be correctly identified as cooking actions or utensils.

4.5.2 Associations Extraction Problems

The main problems relative to the extraction of associations between ingredients and its applied cooking methods and food preparation techniques can be divided into six main categories:

- Accumulated error from extractions tasks;
- Co-reference related errors;
- Dependency parsing errors;
- Linguistic interpretation problems;
- Excluded ingredients dictionary limitations;
- Annotation mistakes.

Accumulated error from extraction tasks

Given that the association tasks performed in this project rely on the extraction of the entities to be associated in the food recipe, any errors in the extraction tasks are carried over to the association tasks. As such, these errors have an impact on the results obtained for the extraction of associations, even though they are not a direct result of flaws present in the components relative to the association task.

Co-reference related errors

The resolution of co-reference cases generally involves the use of an algorithm, applicable to the domain, that is capable of identifying when a single entity is being referred to in different words (e.g. "pasta" and "linguini"). Co-reference resolution is, however, more complex in the food recipe domain. An example of a case where a co-reference algorithm would not be able to properly identify the entity being referred to by different expressions is shown in the following recipe directions:

1. Mix the first and second ingredients in a bowl.

2. Fry the mixture in a heavy large saucepan.

For this set of directions, while it's fairly easy for an individual to interpret that the expression *first and second ingredients* is referring to the first and second ingredients listed in the ingredients section of the recipe, the co-reference resolution algorithm would not be capable of making such an interpretation, as the algorithm operates over a single text and identifies certain linguistic cues in the text that are not present in this situation. The algorithm would, however, be expected to properly associate the expressions *first and second ingredients* and *mixture* as referring to the same entities. This association, however, would be of little importance, given that it would still not possible to associate the ingredients to the used cooking method, due to the entities not being associated to the ingredients they are referring to.

Dependency parsing errors

Dependency parsing information is used in the rules developed for the association tasks of this project. This information is important in order to extract associations between entities in the same sentences through the details relative to their semantic role.

Dependency parsing, however, relies on the tags obtained through the part-of-speech tagger in order to properly parse the written text. As previously mentioned in Section 4.5.1, however, part-of-speech tagging is not reliable for sentences in the food recipe domain, due to their imperative nature. This results in erroneous dependency parsing, which has an important impact in the results obtain for the association tasks. The results displayed in Tables 4.6 and 4.7 show that the use of dependency parsing information increases both the precision and recall of the association of ingredients to their applied cooking method, and the recall of the association of ingredients to different food preparation techniques. While the use of this information only contributes to small improvements in both association tasks, it nonetheless improves the performance of the components, even when taking into consideration the unreliability of the dependency parser. Were the dependency parsing to be significantly more reliable, the improvement in the results obtained would likely be much greater.

Linguistic interpretation problems

Some of the errors present in the association tasks relate to difficulties the system faces when interpreting sentences in the food recipe task which are either poorly structured by the author or contain a misleading expression. An example where the system misinterprets the cooking method applied to the ingredient, due to the use of an ambiguous expression, can be found in the following recipe directions:

1. Mix the chicken thighs with flour;

2. Add water to a large pot and bring it to a boil.

3. Stir in mixture and simmer until the chicken thighs are cooked through.

For this particular example, the system would associate to the ingredients *chicken thighs* and *flour* the cooking method *boil*, as per the rules developed. However, the correct applied cooking method to both of the ingredients would be the cooking method *simmer*. This mistake happens due to the use of the expression *bring to a boil*, which, even though it often implies the use of the *boil* cooking method, it is not always associated to it, and can be associated to other cooking methods (in this case, the cooking method *simmer*).

Excluded ingredients dictionary limitations

One of the dictionaries created for the association tasks was the excluded ingredients dictionary. This dictionary contains ingredients for which the association of applied cooking methods is irrelevant, since cooking methods do not alter significantly the nutritional composition of the ingredients.

However, there is a small portion of ingredients which were excluded from being associated to a cooking method which are not present in the excluded ingredients dictionary. This is generally the case for wine ingredients, that are often referred to by their names instead of their types (e.g. Port wine is usually referred to as "Port" instead of "red wine").

While the ingredients that were being excluded from the association task were written down throughout the annotation process, some ingredients must have been skipped over during this process and, as such, have not been included in the excluded ingredients dictionary. This leads to the presence of additional false positives in the results for the association tasks, and lowers the overall precision of the tasks.

Annotation problems

As was the case with the extraction tasks, mistakes were also identified when analyzing the results obtained for the association tasks for the annotated dataset. These mistakes include an ingredient being associated to more than one cooking method, an ingredient not being associated to the correct applied cooking method, ingredients that should have been excluded from the association tasks being included, etc.

4.5.3 Recipe Nutritional Values

As previously mentioned in Section 4.4, while the actual nutritional values calculated by the sistem for a food recipe were not validated due to a lack of information, twenty ingredients were randomly sampled from a set of ingredients with validated extracted information and the best corresponding database entries were annotated. For each of the selected ingredients, the database entry selected by the component for the ingredient was analyzed to determine whether it was the same as the annotated one.

The results shown in Table 4.8 show that the system was able to more accurately determine the best database entry for each ingredient through the use of the extracted cooking actions. While the system's overall precision was still fairly low (only eight of the twenty ingredients analyzed were matched with the best corresponding entry), it was still a significant improvement over the baseline's results. As such, while it's not possible to make any definitive conclusions from such a small test sample, the results indicate that it's possible to calculate more precise nutritional values for a food recipe with the use of additional extracted information relevant to an ingredient's nutritional composition.

4.6 Summary

In this chapter, the testing and validation process is detailed, focusing on the experiments designed, the analysis of the test set made prior to obtaining the results, the results themselves and their discussion.

Two experiments were designed for the testing and validation process, the first of which involving the calculation of the precision, recall and F-measure values for the extraction and association tasks. The extraction tasks and the association of the ingredients with food preparation techniques were framed as a multilabel problem, while the association of the ingredients with a used cooking method was framed as a multiclass problem. As such, the formulas used for the metrics calculated were different for each of the problem types. Due to limitations in the information available in the food composition database, the recipe's nutritional values were not validated. An alternative experiment was devised, where the best database entries for twenty randomly sampled ingredients with previously validated extracted information were annotated. These annotations were then compared with the database entries selected by the system for the ingredients.

The preliminary analysis of the test set used for this project allowed to compare the composition of the food recipes in the test set with those of the dataset, relatively to the quantity and standard deviation for the words (particularly nouns and verbs) and the most common nouns and verbs. The analysis showed that the test set contained similar characteristics to those of the dataset it was sampled from, and served as a good base for the testing and validation of the system.

For each of the components tested, a baseline was established as a term of comparison for the results obtained for the finalized component.

The results showed that extraction and association tasks both had their own specific set of problems, with the part-of-speech tagger being the most significant cause of errors in both tasks, either directly or indirectly (through dependency parsing errors). The results also showed that it was possible to obtain more accurate nutritional information for a recipe through the use of extracted information relative to each ingredient of the recipe, namely, its applied cooking method and food preparation techniques.

Chapter 5

Conclusions

This dissertation describes the development of a system which extracts and structures information relative to a food recipe, which includes the name, quantity, units, applied cooking methods and food preparation techniques for the ingredients of a recipe as well as the necessary cooking utensils for the recipe. This description includes an introduction of the project's context, motivation, problems and the goals expected to achieve, preliminary considerations prior to the project's development, a detailed overview of the system developed, the results achieved and an analysis of these results for the components developed specifically for the system.

The system developed achieved the goals that were originally established for this dissertation and, as such, it is capable of accurately extracting and structuring information of use to recommendation systems deployed on food recipe websites, and provide the basis of information for additional filters in the search engine used by these websites. This is shown by the F-measures above 0.9 for all but one of the extraction tasks and one of the association tasks. It was also possible to infer, through the system's results, that it's possible to determine more accurate nutritional information through the use of additional structured information relative to the ingredients of a recipe.

The work detailed in this dissertation can be adapted to a variety of nutrition-related projects and food recipe datasets, as well as serve as a foundation on the extraction and structuring of relevant information for users in the food recipe domain.

5.1 Future Work

Throughout the development of the system detailed in this dissertation, several shortcomings were identified. A list of the most important work to pursue in order to address these shortcomings is as follows:

- **Part-of-speech tagger for imperative sentences**: The biggest constraint faced during the system's development was related to the poor performance of the part-of-speech tagger used in the domain. The tagger would often misidentify verbs as nouns, adjectives as nouns, etc., which impacted the performance of the lemmatizer and dependency parsers (since these use part-of-speech information) and forced the development of rules for the extraction and association tasks implemented in the system with the goal of overriding the errors made by the tagger. As previously mentioned in Section 3.1, a part-of-speech tagger specifically built for use in imperative sentences, or a machine-learning tagger trained almost exclusively on a dataset of annotated imperative sentences, would lead to significantly better results for the system developed;

- **A more rigorous and extensive annotation process**: The annotation process relative to this dissertation was done by a single person, who does not have an extensive understanding of linguistics or the nutritional domain. As such, while there was an attempt to be as rigorous as possible with the annotations made, an amount of mistakes were still made throughout this process, which had an impact on the results obtained. A more meticulous approach to the annotation process by a team of individuals with an intimate grasp on linguistics and the nutritional area can lead to the development of a much more extensive annotated dataset, which can also potentially include part-of-speech annotations. This would allow for the implementation of statistical approaches to the problems tackled in this dissertation, and to train a machine-learning part-of-speech tagger for use in imperative sentences;
- **The implementation of a flow graph for food recipes**: The implementation of a flow graph designed for food recipes would be helpful in solving co-reference cases in recipes, while also allowing for a more precise extraction of associations between the actions described in the recipe and the ingredients. Any action and compositional change applied to an ingredient could be detailed through the recipe's flow graph, for each of the ingredients of the recipe. It could also lead to a reduced reliance on part-of-speech information of the recipe for information extraction activities performed in the domain.

References

[AD06] Tanvir Anwar and Mahshid Dehghan. Food composition database development for between country comparison. 5:2, 02 2006.

[All] AllRecipes. Nutritional information. Available at http://dish.allrecipes.com/customer-service/nutrition-information/, last accessed on the 4th of June, 2018.

[BC12] B. Aysha Banu and M. Chitra. A novel ensemble vision based deep web data extraction technique for web mining applications. In *2012 IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pages 110–114, Aug 2012.

[BCc14] BCcampus. *Basic Kitchen and Food Service Management*. BCcampus, BC Open Textbook Project, 2014.

[CCH01] C. Conran, T. Conran, and S. Hopkinson. *The Conran Cookbook*. Conran Octopus, 2001.

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

[DSD12] Tom De Smedt and Walter Daelemans. Pattern for python. *J. Mach. Learn. Res.*, 13(1):2063–2067, June 2012.

[eBi18] eBizMBA. Top 15 most popular recipe websites | may 2018. Available at http://www.ebizmba.com/articles/recipe-websites, May 2018.

[FKK$^+$00] Dimitra Farmakiotou, Vangelis Karkaletsis, John Koutsias, George Sigletos, Constantine D. Spyropoulos, and Panagiotis Stamatopoulos. Rule-based named entity recognition for greek financial texts. In *Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries (COMLEX 2000*, pages 75–78, 2000.

[For73] G. David Forney. The viterbi algorithm. In *Proceedings of the IEEE*, pages 268–278, 1973.

[Gre15] Erica Greene. Extracting structured data from recipes using conditional random fields. Available at https://open.blogs.nytimes.com/2015/04/09/extracting-structured-data-from-recipes-using-conditional-random-field last accessed on the 15th of June, 2018, April 2015.

[Gri97] Ralph Grishman. Information extraction: Techniques and challenges. In *International Summer School on Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology*, SCIE '97, pages 10–27, London, UK, UK, 1997. Springer-Verlag.

[Gro] The Stanford Natural Language Processing Group. Stanford log-linear part-of-speech tagger. Available at https://nlp.stanford.edu/software/tagger.shtml, last accessed on the 18th of June, 2018.

[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 &#38; Eating Activities*, CEA '13, pages 63–68, New York, NY, USA, 2013. ACM.

[HMMT11] Tadayoshi Hara, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. Exploring difficulties in parsing imperatives and questions. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 749–757, 2011.

[IK02] Hideki Isozaki and Hideto Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[Jur00] Dan Jurafsky. *Speech & language processing*. Pearson Education India, 2000.

[KH06] Seung-Shik Kang and Kyu-Baek Hwang. A language independent n-gram model for word segmentation. In *Proceedings of the 19th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*, AI'06, pages 557–565, Berlin, Heidelberg, 2006. Springer-Verlag.

[KhCL$^+$05] Kazuaki Kishida, Kuang hua Chen, Sukhoon Lee, Kazuko Kuriyama, Noriko Kando, Hsin-Hsi Chen, and Sung Hyon Myaeng. Overview of clir task at the fifth ntcir workshop. *NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-Lingual Information Access*, December 2005.

[Kil00] Adam Kilgarriff. Wordnet: An electronic lexical database, 2000.

[Kip12] B.A. Kipfer. *The Culinarian: A Kitchen Desk Reference*. Houghton Mifflin Harcourt, 2012.

[KS06] Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *Comput. Linguist.*, 32(4):485–525, December 2006.

[KTS13] Epaminondas Kapetanios, Doina Tatar, and Christian Sacarea. *Natural Language Processing: Semantic Aspects*. 11 2013.

[LB02] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[LMP01] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[MAB10] E.W. Miller, A. Achilleos, and R. Bayless. *How to Cook Like a Top Chef*. Chronicle Books, 2010.

[McC05] Andrew McCallum. Information extraction: Distilling structured data from unstructured text. *Queue*, 3(9):4:48–4:57, November 2005.

[MCF$^+$98] Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, and The Annotation Group. Algorithms that learn to extract information bbn: Description of the sift system as used for muc-7. In *Proceedings of MUC-7*, 1998.

[McI97] Robert McIntyre. Treebank tokenization. Available at ftp://ftp.cis.upenn.edu/pub/treebank/public_html/tokenization.html, last accessed on the 13th of June, 2018, October 1997.

[MFP00] Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 591–598, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[MI15] Anders Møller and Jayne Ireland. *LanguaL^{TM} 2014 - Thesaurus*. 01 2015.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[MS99] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999.

[MSB$^+$14] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014.

[MSYY12] Shinsuke Mori, Tetsuro Sasada, Yoko Yamakata, and Koichiro Yoshino. A machine learning approach to recipe text processing. In *Proceedings of the 1st Cooking with Computer Workshop*, pages 29–34, 2012.

[Org] World Health Organization. 10 facts on nutrition. Available at http://www.who.int/features/factfiles/nutrition/en/, last accessed on the 4th of February, 2018.

[PK01] Thierry Poibeau and Leila Kosseim. Proper name extraction from non-journalistic texts. In *Computational Linguistics in the Netherlands*, pages 144–157, 2001.

[POA16] Alexandre Miguel Pinto, Hugo Gonçalo Oliveira, and Ana Oliveira Alves. Comparing the performance of different nlp toolkits in formal and social media text. In *SLATE*, 2016.

[RRV$^+$18] Jorge Ribeiro, David Ribeiro, Maria João M. Vasconcelos, Ayla Schwarz, Filomena Gerardo, Ciska van Harten, Riccardo Succu, Robbie Davison, Tiago Oliveira, Marlos Silva, and Tiago Miguel da Silva Ferreira. Cordon gris: Integrated solution for meal recommendations. 2018.

[Sar08] Sunita Sarawagi. Information extraction. *Foundations and Trends® in Databases*, 1(3):261–377, 2008.

[SGC09] Gonçalo Simoes, Helena Galhardas, and Luısa Coheur. Information extraction tasks: a survey. In *Proc. of INForum*, volume 2009, 2009.

[SGS98] Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A decision tree method for finding and classifying names in japanese texts. 08 1998.

[SLL$^+$14] Han Su, Ting-Wei Lin, Cheng-Te Li, Man-Kwan Shan, and Janet Chang. Automatic recipe cuisine classification by ingredients. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, UbiComp '14 Adjunct, pages 565–570, New York, NY, USA, 2014. ACM.

[SM16] Sebastian Schuster and Christopher D Manning. Enhanced english universal dependencies: An improved representation for natural language understanding tasks. In *LREC*, 2016.

[SNL01] Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.*, 27(4):521–544, December 2001.

[Sor] Mohammad S Sorower. A literature survey on algorithms for multi-label learning.

[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*, pages 102–107. Association for Computational Linguistics, 2012.

[Sta] Allrecipes Staff. Cooking conversions. Available at http://allrecipes.co.uk/how-to/44/cooking-conversions.aspx, last accessed on the 7th of June, 2018.

[Ste13] Tanya Steel. Nutrition analysis on epicurious recipes. Available at https://www.epicurious.com/archive/blogs/editor/2013/10/nutrition-analysis-on-epicurious-recipes.html, August 2013.

[Thr11] Oliver Thring. A fruitful search: recipes on the internet. Available at https://www.theguardian.com/lifeandstyle/wordofmouth/2011/oct/19/fruitful-search-recipes-internet, October 2011.

[TKMS03] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

[UII11] Tsuguya Ueta, Masashi Iwakami, and Takayuki Ito. A recipe recommendation system based on automatic nutrition information extraction. In *Proceedings of the 5th International Conference on Knowledge Science, Engineering and Management*, KSEM'11, pages 79–90, Berlin, Heidelberg, 2011. Springer-Verlag.

[Web73] Heinz J. Weber. The automatically built up homograph dictionary a component of a dynamic lexical system. In *Computational And Mathematical Linguistics: Proceedings of the 5th International Conference on Computational Linguistics, COLING 1973, Pisa, Italy, August 27 - September 1, 1973*, pages 457–470, 1973.