FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A Data Analysis Pipeline for the Study and Categorization of User Content in Online Health Communities

Sara Filipa Mendes da Silva

Mestrado em Engenharia Informática e Computação

Supervisor: Prof. Rosaldo J. F. Rossetti, PhD

Co-supervisor: Diogo Santos, MSc

October 31, 2022

A Data Analysis Pipeline for the Study and Categorization of User Content in Online Health Communities

Sara Filipa Mendes da Silva

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. Ana Paula Rocha, PhD
External Examiner: Prof. Brígida Mónica Faria, PhD
Supervisor: Prof. Rosaldo J. F. Rossetti, PhD

October 31, 2022

Abstract

As the amount of health information available on the web increases, more people start to resort to online methods to quickly obtain answers regarding medical conditions. The existence of web-based communities dedicated to health topics attracts increasingly more users with health concerns. These communities provide a space for health consumers to share their experiences, ask questions and get support from other users who might share the same issues.

There has been a growing interest in using the messages exchanged in these online forums to study the dynamics of the communities and to identify potential health patterns. In recent research, these messages have been used to help detect, for example, the appearance of symptoms related to epidemics and new adverse reactions to drugs. This information can be used to benefit health consumers in general and to improve health care services by providing new means of research.

In order to properly analyse these messages, it is necessary to find an adequate source community and study it, then extract and analyse the user-created content and finally classify it, manually or automatically. Ideally, there would be a data pipeline that would guide the community study from message extraction to categorization. This pipeline should describe the process of data extraction, ideally through automatic means, then data categorization or grouping by similarities. The latter process can be done manually, making use of human resources, or automatically, through machine learning algorithms. Manual categorization of a basic training set of data is usually a necessary step and should be considered in this data pipeline. However, doing this for hundreds of thousands of messages is a resource-intensive task that might not be feasible in most situations. For this reason, there has been a lot of attention dedicated to automatic categorization of texts in recent years. By identifying certain characteristics in the messages (features), it is possible to automatically categorize them, and so we also consider this approach in the current work. But there is no solution that fits all problems, opening a lot of opportunities in this research area.

Given the aforementioned context, the overall goal of this dissertation is to develop and evaluate an integrated data pipeline for the analysis of online health communities. Specifically, the aim is to study the applicability of information retrieval and data extraction techniques, as well as manual and machine learning methods to text categorization in online health communities, to analyzing data in an integrated pipeline.

To accomplish this, several modules were developed to deal with the different steps of the proposed pipeline, from dataset extraction, to descriptive statistics and manual and automatic categorization, while evaluating different approaches and algorithms. As a focus of the study, we picked the online community *MedHelp*, where we extracted data for analysis and validation of the developed modules, using a prototype for a data crawler created for this dissertation. We used this data for training and testing classification approaches. The full message content of the online community *MedHelp*, to the best of our knowledge and as of November 2021, was extracted and studied. Then, a subset of it was manually classified by volunteer judges through an application built for that purpose. The categorization was based on an existing classification schema, that contains labels related to emotions and intentions in terms of an online health-related community

context. These manually classified messages were then put through a series of text processing techniques and automatic extraction of linguistic features. With this dataset, it was possible to test different approaches used in similar text categorization problems and compare their performance in terms of success at classifying messages in each category.

The flexibility of the developed systems should allow for extracting data from other online health communities, and for different classification schemas and categorization techniques to be used when studying the data, manually labeling it and automatically classifying it. As such, it is expected that the developed work will contribute to future research on user interaction in different health boards, by providing an integrated approach to message extraction, analysis and classification, in the context of online health communities. Ideally, this will also improve further studies on large-scale annotated health-related datasets, as well as providing a set of baseline tools for this sort of work.

Keywords: natural language processing, information extraction, web mining

Resumo

Com o aumento da informação sobre saúde existente online, também se evidencia um aumento do número de pessoas que recorrem à internet para rapidamente obter respostas sobre condições médicas. A existência de comunidades online dedicadas a tópicos de saúde atrai cada vez mais utilizadores com preocupações médicas. Estas comunidades são um espaço para os utilizadores partilharem as suas experiências, fazerem perguntas e obterem apoio de outras pessoas que possam compartilhar os mesmos problemas.

Devido a esta tendência, tem havido um interesse crescente em usar as mensagens trocadas nestes fóruns para estudar a dinâmica das comunidades e identificar potenciais padrões de saúde. Por exemplo, em estudos recentes, estas mensagens foram utilizadas para ajudar a detetar o aparecimento de sintomas relacionados com epidemias e de novas reações adversas a medicamentos recentemente lançados. Esta informação pode ser usada para beneficiar os consumidores de saúde em geral e para melhorar os serviços de saúde fornecendo novos meios de investigação.

Para analisar estas mensagens, é primeiro necessário encontrar uma comunidade adequada para fonte de dados e estudá-la, pasasndo depois por extrair e analisar o conteúdo criado pelos utilizadores e, finalmente, classificá-lo, manual ou automaticamente. Idealmente, haveria uma *pipeline* de dados que guiaria o estudo das comunidades desde a extração de mensagens, à sua categorização. Esta *pipeline* deveria descrever o processo de extração de dados, idealmente através de meios automatizados, depois categorização destes ou agrupamento por semelhanças. Este processo de categorização poderá ser feito manualmente, através de recursos humanos, ou automaticamente, através de, por exemplo, algoritmos de aprendizagem máquina. A categorização manual de um conjunto base de dados de treino é, normalmente, um passo necessário e deverá ser considerado nesta *pipeline* de dados. No entanto, aplicar este trabalho manual a, potencialmente, centenas de milhares de mensagens, é uma tarefa que requer muitos recursos, o que pode não ser viável na maioria das situações. Por esta razão, nos últimos anos, a categorização automática de textos tem recebido muita atenção. Ao identificar certas características nas mensagens ("features"), é possível classificá-las automaticamente através de variadas abordagens, sendo, então, uma abordagem também considerada e estudada no presente trabalho. Apesar de tudo, não existe uma solução que se encaixe em todos os problemas, criando muitas oportunidades nesta área de estudo.

Dado este contexto, o objetivo geral desta dissertação é desenvolver e avaliar uma *pipeline* integrada de dados, para análise de comunidades online de saúde. Especificamente, o nosso objetivo é estudar a aplicabilidade de técnicas de recuperação de informação e extração de dados, assim como de métodos de categorização de texto por vias manuais ou automáticas, ao estudo e análise de dados, neste domínio particular das comunidades de saúde, usando métricas de avaliação *standard*, sempre que possível.

Para tal, foram desenvolvidos vários módulos para lidar com as diferentes etapas desta *pipeline* proposta, desde a extração de conjuntos de dados (*datasets*), a estatística descritiva e classificação manual e automatizada de conteúdo, com avaliação de diferentes abordagens e algoritmos. Como

foco deste estudo, escolhemos a comunidade online *MedHelp*, de onde foram extraídos dados usando um *data crawler* desenvolvido no âmbito desta dissertação, para posterior análise e validação dos módulos criados. Utilizamos estes dados para treino e teste de abordagens de classificação de mensagens. A totalidade do conteúdo da comunidade, à data de Novembro de 2021, foi extraída, na sua totalidade, e estudada. Depois, um subconjunto destas mensagens foi classificado manualmente por diferentes pessoas, através da uma aplicação criada, por nós, para esse fim. As categorias escolhidas foram baseadas num esquema de classificação já existente, que contém *labels* relacionadas com emoções e intenções de utilizadores no contexto das comunidades online de saúde. Estas mensagens, já classificadas, são depois submetidas a uma série de técnicas de processamento de texto e extração automática de certas características linguísticas. Com este conjunto de dados, é possível testar diferentes abordagens, usadas frequentemente em problemas semelhantes de categorização de texto e comparar o seu desempenho em termos de sucesso na classificação de mensagens dentro de cada categoria.

A flexibilidade do sistema desenvolvido permite a extração de dados de diferentes comunidades-fonte, online, de discussão de saúde e a utilização de diferentes esquemas de classificação e técnicas de categorização para estudar esses dados, etiquetando-os manualmente ou classificando-os automaticamente. Assim, é esperado que o trabalho desenvolvido possa contribuir para estudos futuros com base na interação de utilizadores em diferentes fóruns de saúde, disponibilizando uma abordagem integrada de extração, análise e classificação de mensagens neste domínio. Idealmente, este trabalho irá contribuir também para propicionar trabalhos futuros que envolvam conjuntos de dados de grande escala relacionados com a saúde, assim como disponibilizar um conjunto de ferramentas base para este tipo de atividade.

Palavras-chave: processamento de linguagem natural, extração de informação, web mining

Acknowledgements

I would like to give my most heartfelt thanks to my supervisor, Prof. Dr. Rosaldo Rossetti for believing in me and always supporting me during the development of this work. It was largely due to his knowledge, patience, guidance and, perhaps even more so, his kindness, that I was now able to surpass this challenge of mine.

A very special thank you also to my co-supervisor and dear friend, Diogo Santos, who was always next to me for the good and bad parts of this journey, never stopped believing in me and always kept pushing, pulling, and even shoving me towards this goal.

To my family, a warm thank you for their support and for enabling me to study and achieve this degree. To my friends, I am grateful for all the lovely moments we shared, memories I will cherish forever. Thank you for believing in me and for all the kind words and advice.

Thank you so much,

Sara Silva

"Remember, hate is always foolish and love is always wise.
Always try to be nice, but never fail to be kind."

12th Doctor

Contents

List of Figures

List of Tables

Abbreviations

AUC	Area Under Curve
API	Application Programming Interface
cf	Collection Frequency
df	Document Frequency
idf	Inverse Document Frequency
IG	Information Gain
IR	Information Retrieval
KNN	k-Nearest Neighbors
LDA	Latent Dirichlet Allocation
ML	Machine Learning
NLP	Natural Language Processing
OHC	Online Health Community
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
tf	Term Frequency

Chapter 1

Introduction

1.1 Context

With the appearance of large amounts of information online regarding health-related topics, it is becoming easier, cheaper and quicker to resort to online methods to obtain answers regarding medical conditions. A study performed in 2013 about health care trends by the *Pew Research Center* [11] reveals that 72% of the internet users looked online for health information in that year. The most common search queries were about symptoms, specific diseases and health care professionals. It also shows that 18% of the users went online, at some point, to find others who might share the same issues. With the growth of online communities in the recent years, there has been an increase in the amount of users seeking support online in these types of spaces [4]. Online Health Communities (OHCs), in specific, provide a space for users to share their medical experiences, ask health-related questions and obtain support. In turn, these communities become large repositories of information in the format of messages exchanged between people.

The content of these messages and the interaction between the users of these environments can be a valuable source for studying, for example, what users seek in these environments and how they interact with other patients and physicians. In the recent years, there has been a developing interest in studying these spaces with the intent of understanding the dynamics of the communities or identifying potential medical issues and patterns [8, 30, 12]. Further research in this area can lead to the use of this kind of data in health applications with the potential to benefit not only the users of these spaces, but health consumers in general, and to provide insight towards the improvement of health care services.

Extracting relevant information from these large datasets often involves several, laborious steps. A first approach to extract raw data is necessary, which nowadays is best performed through automated means, given the amount of information even a medium sized online social space can hold. Afterwards, one needs to study this raw data and, usually, perform text categorization, the task of labeling texts based on its content and attributes [48]. With the amounts of existing information and at the speed new data appears on a daily basis, manual categorization is no longer a viable option to label the entirety of available information. Nowadays, researchers often look into

automatic methods of text categorization to help with this complex task, complementing manual means. Due to the multitudes of different domains of data, even inside the health area, there is not a solution that fits all problems [31]. In the context of online health communities, however, there might be relevant characteristics that remain constant across different communities, which can be used to establish patterns for information extraction, analysis and, finally, manual and automated text categorization and can then be integrated into a flexible data processing pipeline.

1.2 Motivation and Goals

With the rising attention towards text analysis and categorization in social media in the recent years, there have been several different approaches to these problems. The process, from extracting the dataset to evaluating results, usually follows a general sequence, but with different techniques, algorithms and tools in each step.

Several studies have been performed in online communities regarding health concerns [13, 6, 8, 4, 47], from studying users' sentiment to detecting new adverse drug reactions, with many different techniques being used and without an unique solution that fits all, as previously mentioned. As highlighted in the work by Zhang et al. [49], the intention or context of the posts is a very important characteristic when trying to extract relevant information for a study. For example, a user showing anger because of a technical problem in the forum is very different from showing anger because of a medical situation. This difference might influence research results. For these reasons, the existence of a framework that extracts and categorizes messages exchanged in OHCs, according to their intentions and emotions, can contribute to large-scale analyses of these online spaces and further research in this area.

Thus, the main goal of this dissertation is to explore methods for message and user data extraction, analysis and their manual labelling, as well as studying different text categorization approaches to automatically classify messages in an *Online Health Community*, and evaluating the performance of these methods and approaches, based on common metrics. We propose, for this, the concept of a sequential data pipeline: broadly, from data extraction, to analysis, to classification. This work will focus the study on a specific online health community, and make use of an existing classification schema [42] as its basis. The aim is to help contribute to larger scale study and analysis of OHCs in terms of users' intents and emotions displayed.

More specifically, the objectives of this work are as follows:

- To develop an approach for collecting datasets of user-created messages and user profiles, from an OHC and generate a large-scale dataset of messages and user data from a specific community;
- To study the collections of data as a step of initial analysis and discovery of a community's characteristics;
- To manually annotate some of the collected messages, devising a simple and domain agnostic approach for this;

- To select algorithms and tools commonly used for text processing and automatic text categorization, that are adequate in the OHC context;
- To apply and evaluate each of the above listed techniques, individually, as well as integrated in a potential data pipeline.

1.3 Dissertation outline

This report is divided in several chapters as follows: Chapter 2 presents a literature review of approaches in automatic text categorization. Unlike other problems we intend on studying in this dissertation, with more straightforward approaches for solution, we believe that automatic text categorization is still an open-ended problem requiring careful review. Given this, we focused the literature review on text analysis, processing and categorization, in general domains but also specifically in the field of online health communities, in hopes of providing the reader with helpful context to the problem and typical solutions. Chapter 3 focuses on stating the problem identified in this dissertation, describing the main research questions to be analysed and proposing a methodological approach to tackle these. We propose an integrated pipeline of tools and approaches for data extraction, analysis and categorization, focusing on a single online health community. Main results and their analysis, as well as relevant implementation notes and considerations resulting from this work, are presented and discussed in Chapter 4. Following this, Chapter 5 brings this document to an end, summarizing what has been done and the main conclusions derived from the work. Complementing this, we also outline the main contributions from this dissertation and describe what, in our opinion, would be future developments worth pursuing to improve and expand the present work.

Chapter 2

Text categorization applied to online health communities - A review on automatic approaches and their usage

As mentioned before, while other important research questions and technical problems we intend to tackle in this work already have, often, multiple well-established possible solutions to handling them, we believe that the issue of automatic text categorization is still a very open-ended research problem with active research being done about it. Given this, we opted to focus our main literature review in studying this problem of text analysis and categorization, in the broader sense, before refining into the specific domain of online health communities. Now follows our main findings.

The problem of *text categorization*, or *classification*, is not new in the area of information science. With the increase in the amount of information that surrounds us, there's also a rising need in exploring solutions to automatically treat that information and extract useful patterns. From automatically separating e-mails into folders, as approached by Bekkerman et al. [5], to analyzing users' sentiment over time, as in the work by Zhang et al. [49], there has been a lot of interest in this area in the recent years due to the large amount of possibilities. This also happens in the medical field, where the messages exchanged between health consumers in online communities can be used in multiple contexts, from tracking patients' behaviors to predicting epidemics through reported symptoms online.

In order to understand what techniques have been used in this field, from the acquisition and processing of the dataset to the training of models, a literature review on the current state of the art is essential. This chapter will present methods found during the review of different works. Section 2.1 starts by describing the meaning of text categorization and the text categorization problem. Section 2.2 presents the text categorization process, followed by a detailed explanation of the techniques used in each step of the process, from Section 2.2.1 to 2.2.4. Section 2.5 highlights to the reader the main points to take from this chapter.

After a review on text categorization techniques, this chapter will then focus on online health communities (OHCs) and text categorization applied in this context. Although the focus of this dissertation is in health communities, all literature pertaining to any type of community with health-related content was considered in this review, for the purpose of studying a broad range of techniques used in text categorization applied to health-related messages.

Section 2.3 explains the different types of OHCs, presents analytics regarding existing OHCs and lists possible research topics using these communities. Section 2.4 highlights text categorization methods applied in studies that target OHCs.

2.1 Text categorization

The general representation of this problem is called *document categorization*, where the concept of *document* can be extended to multiple media types, like images. In the context of this literature review, a document refers to texts, usually the messages shared between users, in more specific cases. Text categorization, or classification, is a domain-dependent task of attributing categories to documents, based on the features that represent them. This can be done by manually annotating messages or through classification systems with classification rules. These rules can be manually set by experts or, more recently, automatically through machine learning algorithms. The need for automatic text categorization is increasing due to the necessity of keeping up with the growth of information available. It is important to realize that there is not an unique solution that can be applied to all text categorization problems, due to the core issue of text categorization: the problem is not about the algorithm, the main issue is the domain itself [31].

2.1.1 Applications

In recent research, text categorization has been applied to different purposes, such as:

- Spam filtering and e-mail foldering [5, 22] - Categorizing incoming e-mails according to their content is a subject of research interest that started with the objective of automatically removing spam e-mails from inboxes. Recently, the goal has become more ambitious, now aiming to automatically move e-mails to user-created folders;
- Document indexing [15] - Indexing documents and properly retrieving them according to user queries, in an attempt to best match the information needs of the user, has been a topic of research since the early days of document categorization;
- Sentiment analysis [9, 30] - More recently, studies have focused on detecting author's sentiments from written text. This goes from predicting a user's position regarding a topic to extracting the emotional state of the writer;
- Document sorting and filtering [15] - Involves sorting documents according to categories, such as organizing news stories by subject, grouping texts by their genre and filtering documents according to their language.

2.1.2 Defining the categorization problem

Depending on the number of available categories, text categorization can be divided into the following types [48]:

- *Binary*, if a document can belong to one of two categories;
- *Multi-class*, if a document can belong to one of multiple categories;
- *Multi-label*, if a document can fit into multiple categories.

Furthermore, the categorization problem can be either a hard categorization or a ranking/soft categorization, as follows [41]:

- *Hard categorization*, if a "hard" decision is taken and a specific category is attributed;
- *Soft/Ranking categorization*, if chosen categories appear in a rank ordered by the appropriateness of the label to the text.

2.1.3 Types of classification systems

The task of classifying messages can be done manually or by using classification systems that are built according to classification rules, manually added by experts or automatically inferred by machine learning algorithms.

Manual annotation is a laborious task that involves having a group of judges attributing labels to texts, according to a given set of rules. Besides being resource-intensive in terms of time, and sometimes money, there is also the risk of human mistakes caused by caused by distraction and boredom [33]. These systems become impossible to maintain when the stream of information to analyze is not static, as the amount of messages to label increase [18]. However, in studies where the dataset is static and small, manual categorization can still be used [29].

A **classification system** can also be used to categorize messages, based on a set of rules. These systems can follow a knowledge engineering approach or a machine learning approach.

In the **knowledge engineering**-based approach, domain experts manually set the classification rules for the inference engine of the system. This also becomes impracticable to maintain over time, with the rapid growth of information.

With the problem of dealing with significant amounts of data and keeping up with the constant stream of new information, researchers have turned to **machine learning** (ML). The ML approach is based on algorithms that can build a model to represent the classification system, by *learning* the characteristics of the categories, and automatically predict the labels of the documents. They can be *supervised*, if there is an existing set of labeled data from which the algorithm can learn the classification patterns from, or *unsupervised*, if the algorithm must find structure on its own from unlabeled sets of data (often through clustering [41]). *Semi-supervised* learning is a combination of the two, where the training dataset has both labeled and unlabeled data.

This ML approach is the core of *automatic text categorization*. However, processing natural language automatically is not easy, due to the ambiguity of the language and the difficulty in conveying domain concepts to algorithms. In an online context, there is also the presence of typos, word shortenings, slangs and *emoticons* [43] that add difficulty to this process.

The remainder of this chapter will focus on *automatic* categorization, explaining the process, the text preprocessing techniques and the ML algorithms.

2.2 Automatic categorization tasks

The process to achieve automatic categorization is composed by multiple steps, where different decisions must be taken at each step, depending on the problem at hand, such as:

- *How to obtain the dataset?* One of the first tasks is to decide the source of the information and how to extract it;
- *What categories to use?* Building a classification schema depending on the objective of the system is an important step;
- *What is considered relevant information?* Not every word in a document is relevant to the objectives of the system. It might be useful to extract vocabularies and remove or keep certain words, depending on the domain of the dataset;
- *How to represent the documents?* Due to the extensive amount of text elements in a document, they are usually represented by a set of its features. Choosing the appropriate features for the document model is an essential step;
- *What algorithms to use for the classifiers?* It is important to decide the type of algorithms to use that fit with the available dataset as well as choose its parameters to guarantee the best results possible;

The process taken by most of the literature studied can be resumed into 6 steps represented in Figure 2.1 and explained below.

1. Dataset extraction - The sources of the dataset can be varied and, for this reason, the ways of obtaining the dataset can also be different. Although there are existing corpus available online [20], in most cases there is a necessity of collecting up-to-date messages. Some communities might provide an API (even with certain limitations), such as *Twitter*, and, for the others that do not, developers have to turn to scraping methods that parse the HTML of the pages.
2. Manual annotation - In order to apply some classification algorithms, it is necessary to have a manually annotated dataset to train the model and also to evaluate its performance. This can be done by gathering a group of volunteers or paid judges to manually attribute labels

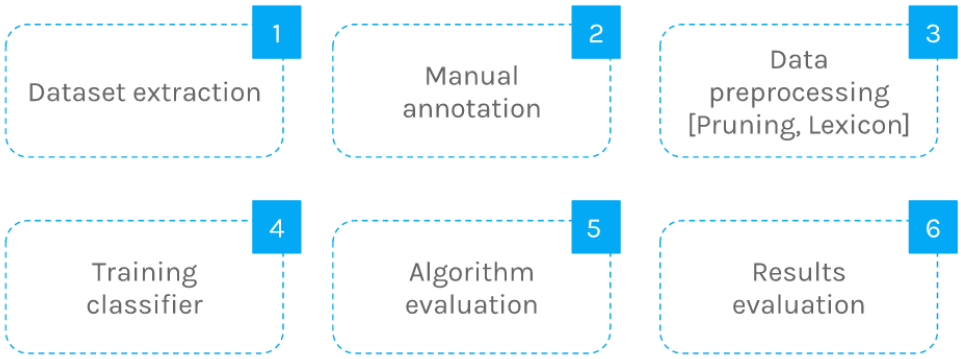

Figure 2.1: Usual tasks in text categorization.

to the messages, according to a set of rules or details for each category. If multiple judges are classifying the same messages, it might be important to use metrics to calculate the agreement between them;

3. Data preprocessing - This phase consists of removing elements in the extracted documents that are not relevant to the classification (pruning) and collecting the elements or vocabulary that are deemed important (feature selection).

4. Training classifier - Taking into account the chosen labels and features, the manually categorized set of documents is used to train a classification model. The duration of this phase is dependent on the used algorithm and the size of the dataset. The trained classifier is expected to correctly predict the categories of new inputs.

5. Algorithm evaluation - This stage involves measuring the performance of the classification systems in terms of the correct assignment of categories, by calculating common metrics, such as precision and recall. Sometimes, the obtained model is also compared to other baseline models, such as one that classifies everything into the same category [49]. In these cases, the trained model is expected to outperform this basic model.

6. Results evaluation - This final stage consists of making conclusions and considerations from the obtained results, in relation to the research being done.

Steps 3 through 5 are further detailed in Sections 2.2.1 through 2.2.4, regarding the commonly used techniques.

2.2.1 Text preprocessing

For the purpose of feature selection, it is important to parse the raw text, making use of natural language processing (NLP) techniques and extracting domain-specific vocabulary, in an attempt to give an uniform structure to the text. This process usually entails the following techniques [32].

Tokenization

Lexical tokens are individual characters or character sequences that collectively give meaning to a text. A single *token* can be defined as a "piece", element or instance of an input string. The process of splitting up messages into tokens is called *tokenization*. A *type* is a collection of tokens that have the same sequence of characters. A *term* is a normalized type that is included in the system's dictionary.

This is a process that involves removing certain punctuation in order to separate the inputs in tokens. Finding the best way to split up the sequences is essential, as it can influence the meaning of the tokens. As an example, Manning et al. [32] provide some options to split the word "*aren't*" shown in Figure 2.2.

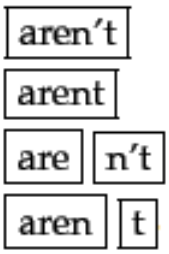

Figure 2.2: Possible tokenization options of the word "aren't".

Stop words

As mentioned above, *terms* are relevant words included in the system's vocabulary. In contrast, *stop words* are excluded from it for not being relevant in the representation of a document. In a language, there are many tokens that are common in practically every text [43]. Thus, they do not add value to the document and are often removed. The stop word list can be obtained by ordering the tokens by their *collection frequency* (how many times they appear in the dataset) and adding the top most common words to the stop list.

Frequency pruning

In contrast to stop words removal, frequency pruning is the task of removing words that are an infrequent occurrence in the dataset.

Normalization

Normalization is the process of applying different transformation rules to the tokens, in order to match terms that should be equivalent despite having different character sequences. For example, "Forum", "forum", "Forums" and "forums" can be represented by the same term "forum".

This task includes dealing with capitalization by transforming everything to lowercase, removing accents, equating different spellings of the same words, like color and colour, dealing with acronyms and so on.

Stemming and lemmatization

Following the logic of normalization, both techniques focus on removing word endings with the intent of normalizing words that are in different forms (plurals and verb conjugations, for example).

Stemming is the removal of word endings arbitrarily, without taking into account morphological rules. *Lemmatization* takes into account the morphology of words with the intent of transforming them into their base or dictionary forms (lemma).

It is relevant to note that, when the length of the messages is usually between small and medium, excessive pruning might have a negative impact on the classification [30]. Depending on the necessity of the system, it might be interesting to use different rules for each of the above techniques, such as different tokenizers [30], and compare the performance of the systems with each.

2.2.2 Document representation and feature selection

In text categorization, due to the complexity of the documents, each document must be represented in the form of a vector of relevant *features*. These vectors are a simplified version of the documents that identify them and are easier to work with when training classifiers.

These features can be elements of the content or associated metadata, such as authorship. Usually, these features are accompanied by their respective *weights* (relevancy metric) in these vectors.

Calculating weights

As said previously, the vector of features that model a document also come with *weights* associated with the features, in order to represent their importance in the document.

Calculating the weight of the features can be done through different methods, as explained by Manning et al. [32]. The *binary* approach consists of giving a score of zero or one depending on if the feature is present in the text. Weight calculation can also make use of the following frequency metrics:

- *term frequency* (tf), which is how many times the term appears in a document. It is frequently used as a weight measure;
- *document frequency* (df), which is how many documents contain a certain term;
- *collection frequency* (cf), or how many times a term appears in an entire collection;
- *inverse document frequency* (idf) is used to represent the rarity of a term: the higher the *idf*, the rarer the term. The idf can be calculated as shown in Equation 2.1, where N represents the size of the collection and df is the document frequency of a term [32];

- *tf-idf*, or term frequency-inverse document frequency, is a metric commonly used for the weight of the terms. It is calculated with the frequencies mentioned above and its objective is to give a higher value to the terms that occur more times within a smaller number of documents. Equation 2.2 shows how it can be calculated.

$$idf_t = \log \frac{N}{df_t} \tag{2.1}$$

$$tf - idf_{t,d} = tf_{t,d} \times idf_t \tag{2.2}$$

N-gram models

N-grams are ordered sequences of *N* characters or words [34, 10]. The features in the vector might not necessarily be just *words* or *characters*. They can be terms composed by sequences of *N* items, where the items might be words or characters. Depending on the length of the sequence, these models can be called *unigram* (N = 1), *bigram* (N = 2), and so on.

A common document representation model is the **Bag-of-Words** approach. In this model, each document is represented by a *bag* of its words, ignoring word order, sentence grammar and structure, but preserving the multiplicity. This model is considered to be a N-gram model with N = 1 (*unigram*), because it stores *words*.

Models with N > 1 maintain some order of the items, unlike the bag-of-words approach, by keeping together items that are normally more relevant when appearing together. As an example, a *trigram* model would keep "Online health community" as a whole, whereas the bag-of-words/unigram approach would have "online", "health" and "community" separate.

Feature selection

Decreasing the amount of *noise*, by removing words that do not influence categorization, is known to increase the effectiveness of the classification systems [41].

There are different ways of selecting what features are used to represent the documents. These features are usually related to the content of the text itself, but can also take advantage of any metadata associated, like the text's authorship, for example [43].

Manually selecting features according to the domain of the dataset is a possible option. Creating our own vocabulary allows for a more flexible control of what we want to highlight in the messages, such as emotions or *emoticons* [43]. In the health context, it is also possible to make use of existing medical vocabularies online [49].

Another option is to **automatically rank features** according to their relevance, by using different metrics or algorithms. One such metric can be the aforementioned *document frequency* and simply select the top most frequent words as features, after pruning. However, there are other

options, such as the *Information Gain* (IG) and the *Chi-square*, which are metrics that take into account the relationship between the categories and the features [15]. IG measures how common a term is inside a certain category, compared to how common it is overall, identifying terms that have greater impact or better represent a category. Chi-square measures the strength of the dependence between features and categories [15].

Like previously noted, features might be *phrases* as well. However, Lewis [24] considers these to be poor content indicators not suitable to be features.

2.2.3 Classifiers

In order to assign labels to texts, it is important to have a set of classification rules that deem a label appropriate for a document. In the case of automatic classification, these rules are learned automatically from a training dataset, using machine learning algorithms. These algorithms might learn from a set of pre-classified data (supervised learning) or try to group documents in a meaningful way, from a set of unlabeled data, without prior information (unsupervised learning) [15]. These algorithms are described below.

Probabilistic classifiers

Probabilistic classifiers calculate the probability of a document belonging to a category. The *Naive Bayes* approach uses the Bayes' theorem to calculate this probability with a naive assumption of the independence between features.

Logistic regression

Logistic regression classifiers try to obtain a model of the approximation of a set of continuous or binary independent features to a set of binary categories. The *Maximum Entropy* classifiers fall into this category.

Decision trees

Decision tree algorithms, as the name indicates, build a tree structure of decisions taken at each node. A node represents a feature, an edge is a decision based on the feature's weight and a leaf is a category. The tree structure is built based on the information gain of the features and, in the end, prediction of a document category is made according to the occurrence of features. Among decision tree algorithms, we can find ID3, C4.5 and Random Forests. Adaptations of the C4.5 algorithm allow multi-label classification.

Support vector machines

The Support Vector Machine (SVM) algorithm maps the feature vectors in a feature space, where the points representing positive instances are separated from the others by a hyperplane with the

maximal margin (maximal distance between the hyperplane and the nearest points from the positive and negative spaces). This algorithm is used in a lot of categorization problems, has no risk of *overfitting* and feature selection is not relevant either. Although originally intended for solving binary categorization problems, the SVM algorithm can be applied to multi-class problems as well. This can be achieved through two possible variations: One-against-all and One-against-one. The former involves having a classifier for each category (so the positive points are part of that category and the negative points are the ones that do not fall in that category). The latter involves having a classifier for each pair of categories and the resulting category is a majority vote between all classifiers. The one-against-one approach has a faster training time and is most commonly used because of this [16].

k-Nearest Neighbors

K-Nearest Neighbors (k-NN) is a similarity-based algorithm, where the classifier calculates the similarity between documents. k-NN attributes a category to a document if a good portion of the k most similar documents from the training set (the nearest neighbors) have that category. This algorithm can be extended to support multi-label classification. Despite its generally poor performance, k-NN is used frequently as a base model for evaluating other models.

Rocchio algorithms

The Rocchio algorithm comes from information retrieval and is also based on the vector space model. It categorizes a document depending on the distance between the document's feature vector to the representative vector of the category. A vector of the category is obtained using the feature vectors of the documents that belong and do not belong to it, using positive and negative feature weights as appropriate. Although easy to implement, it usually does not have good classification performance.

Bagging and Boosting

The concept for Bagging and Boosting is based on the assumption that receiving classifications from multiple judges has better results than receiving input from only one expert. These techniques use multiple classifiers, all differing from each other. Bagging involves training the classifiers in parallel and Boosting in a sequence. The final classification is a result of a majority vote of all the results from the classifiers.

AdaBoost is a boosting algorithm that can be used in multi-label classification problems.

Neural Networks

Neural Networks are based on clusters of artificial neurons, simulating the way a brain would work. Each neuron is connected to other neurons and signals traverse from front to back. They usually have multiple layers: an input layer, an output layer and one or more hidden layers. The

input nodes receive the feature values and the output nodes give the category values [15]. Neural Networks can be trained with backpropagation, where the input vector traverses through the layers until it reaches the output layer. There, the obtained result is compared to the expected result and the error is calculated and propagated backwards until the input layer.

Depending on the exact implementation of the neural net, it can be used with supervised or unsupervised learning.

Back-propagation multi-label learning is an implementation of Neural Networks that can be used to solve multi-label classification problems.

Clustering

Clustering is an unsupervised learning task and its objective is to group documents into representative clusters, in such a way that members of a cluster are considered similar, without prior information. There are several clustering algorithms, such as k-means and fuzzy clustering, each one with its own approach of building the clusters.

2.2.4 Evaluation

As seen in the previous sections, there are multiple options that one can take when building a classification system. The multiple possible algorithms, configuration options for the parameters and preprocessing decisions influence the performance of the classifiers. Measuring and comparing the performance of different classifiers applied to the same dataset, or the same classifiers with different setups, is a critical step in modern classification systems [3].

When evaluating a system, it is common to compare it to baseline systems. Such base system can be as simple as a model that classifies everything into the same class, a model that gives random classifications or even a Naive Bayes model.

When building and evaluating a classifier, it is important to have a separation between *training* set and *testing* set. The former is used to train the classifier and the latter to test its performance. Training and testing on the same set can lead to *overfitting*, a situation where the model is biased towards the training set which decreases its predictive capacity when testing with different data. The training set can be further divided into a *validation* set, to be used when performing experimental evaluation to optimize algorithm parameters.

However, so many splits on the dataset originate smaller and smaller sets. A common technique to avoid this and assure the generalization of the model is the *k-fold cross-validation*. In this process, the whole dataset is divided into k smaller sets of the same size. The model is trained using $k - 1$ sets and tested on the remaining set. This process is run k times, each time alternating the test set. The final accuracy of the model is calculated using averaging methods.

Metrics

Besides metrics like *training time* and *testing time*, it is also common to compare the system's classifications with the manually assigned categories, calculating its classification capability. In the end, there are 4 kind of classified documents:

- True positive (tp) - Correctly classified documents in a category;
- True negative (tn) - Correctly rejected documents from a category;
- False positive (fp) - Incorrectly classified documents, the documents that do not belong in that assigned category;
- False negative (fn) - Incorrectly rejected documents, the documents that should have been classified as part of a certain category but were not.

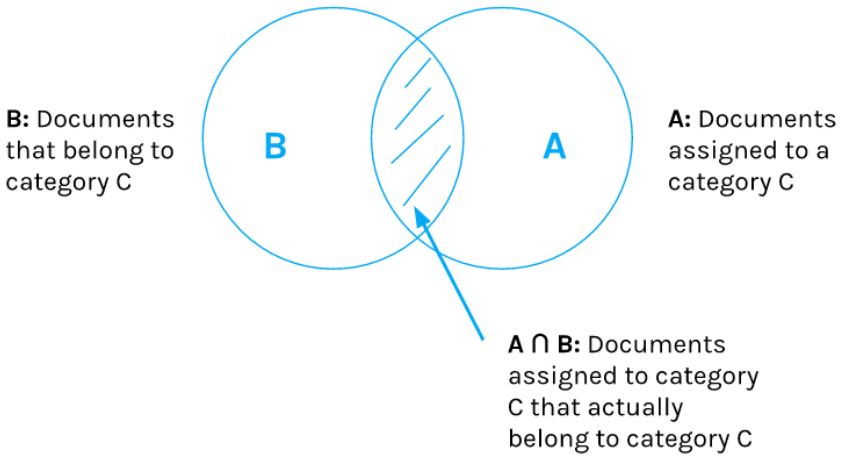

Figure 2.3: Representation of correctly categorized documents in a document collection space, adapted from Baeza-Yates et al. [3].

In information retrieval, using the concepts above, it is possible to calculate metrics to measure a system's classification performance, such as *precision* and *recall* more commonly, among others such as *fallout*, *accuracy*, *error rate* and *F-measure* [3, 25].

- *Precision* is related to the fraction of documents assigned to a category that really belong to that category, represented in Equation 2.3;
- *Recall* is the fraction of documents that belong to a category that the system actually assigned to that category, represented in Equation 2.4;
- *Fallout*, or false positive rate, is the proportion of documents assigned to a category that do not belong to that category, represented in Equation 2.5;

- *Accuracy* can be defined as how well the system categorizes the documents. It is the sum of the number of true positives and true negatives divided by the total number of documents, represented in Equation 2.6;
- *Error rate* represents both the incorrect assignments and errors of omission, represented in Equation 2.7.
- *F-measure* is the weighted harmonic mean of precision and recall, represented in Equation 2.8 where R is recall and P is precision.

$$Precision = \frac{tp}{tp+fp} \tag{2.3}$$

$$Recall = \frac{tp}{tp+fn} \tag{2.4}$$

$$Fallout = \frac{fp}{fp+tn} \tag{2.5}$$

$$Acc = \frac{tp+tn}{tp+tn+fp+fn} \tag{2.6}$$

$$Error = \frac{fp+fn}{tp+tn+fp+fn} \tag{2.7}$$

$$F = \frac{2}{\frac{1}{R}+\frac{1}{P}} \tag{2.8}$$

Figure 2.3 presents the space of the collection and the intersection between the manually categorized sets and the sets categorized by the system, for further understanding of the aforementioned effectiveness measures.

Visualization

To obtain a graphical view, the effectiveness can be represented by a precision-recall curve. ROC (Receiver Operating Characteristic) curves are often used to represent the true positive rate versus the false positive rate.

Table 2.1: Different sources used in different articles.

Sources	Articles
General	
Twitter	[12], [13]
Q&A websites	[6]
Health-specific	
MedHelp	[47], [50]
PatientsLikeMe	[1]
WebMD	[1]
Breast Cancer Forum	[49]
Cancer Survivors Network	[9], [8]
Other forums	[38], [17]

Averaging of metrics

If the classification is applied to different sets of data, such as when using cross-validation, calculating the *Micro* and *Macro averages* is also relevant. *Macro averaging* of precision and recall is a simple average of all the precision values or recall values from the sets. *Micro averaging of precision* involves dividing the sum of the true positives of all the sets by the sum of both true positives and false positives. In contrast, *Micro averaging of recall* is similar, but instead of false positives it uses false negative rates.

2.3 Online Health Communities

The communities that are usually the target of research can be divided into the following groups, according to their purpose:

- General - Communities that are not health-related, but where users might comment on health issues. For example, *Twitter* and generic question-and-answer websites with a broad range of discussion groups, such as *Yahoo! Answers*.
- Health-specific - Communities that are health-related. These types of spaces can be generic and include multiple sub-communities dedicated to different health topics, such as *MedHelp* and *PatientsLikeMe*. There are also forums that are dedicated to specific diseases, such as the *Cancer Survivors Network*.

Health-specific communities have a focus on keeping the patients correctly informed and also on providing emotional support to concerned users. Patients with certain medical issues are also more inclined to join health-related forums seeking information and social support from others who are experiencing the same issues [4].

Furthermore, Ae Chun and McKellar [1] present the following types of health communities online:

- *Moderated question-and-answer websites* - Where users can have their questions answered by a medical specialist;
- *Healthcare blogs* - Can be written by anyone, patients or experts, can be about anything, from providing health-related information to sharing medical experiences, and can be viewed and commented by anyone;
- *Social networking sites* - A structured place where users can ask questions, share experiences and obtain support from others;

Some health communities gather all of those three aspects in one space, offering a discussion board for the users, a place to talk with specialists and also informative blogs.

Table 2.1 briefly presents the communities that were analyzed in the literature reviewed, not all of them specifically health-related.

2.3.1 Community comparisons

The following list contains a brief overview of existing health communities.

- *MedHelp* offers its users a space for discussion as well as information on several health topics and multiple health tools to keep track of health-related conditions. It reaches almost 500 thousand unique monthly visits from the U.S. alone [39] and more than 98,000 unique views a day [44] which makes the discussion boards very active. It is divided into more than 300 official sub-communities focused on different topics, with the option of accessing more than 1000 user groups as well. Across its multiple services, it has more than 12 million users. It is freely accessible and the communities are visible without registration needed.
- *PatientsLikeMe* provide discussion boards on several different health topics as well as trackers for medical conditions, with the option of donating tracked data for research. It has a focus on providing structured quantitative data for health research. This website covers more than 2,500 health conditions with more than 500,000 registered users [36] and more than 12,000 unique views per day [44]. The discussion boards are visible upon registration.
- *WebMD* is a well-known website for health information, news and services, but it also has community forums where users can participate and post their questions. Almost 2 million people visit this website daily [44] with 8 million unique monthly views from the U.S. [39]. The boards are visible without registration.
- *Cancer Survivors Network* is a community dedicated to survivors and caregivers. Users can participate in discussion boards and chatrooms and create personal blogs to give and receive support and share feelings and experiences with other members. It has more than 992,000 cancer-related posts and around 37,000 posts in other discussion boards. The boards are visible without registration.

- *DailyStrength* is a support community composed by more than 400 sub-communities in 34 categories of medical conditions and general well-being topics. It has more than 300,000 registered users and around 10,000 unique daily views [44]. The boards are visible without registration.

2.3.2 The use of text categorization in OHCs

Text categorization has been applied to the OHC context to perform different studies, often based on tracking the intentions of the users or the content being talked about in order to draw explanatory or predictive patterns. The following points present examples of such research.

Sentiment analysis

In an effort to try to explain why people seek these spaces, a lot of research has been conducted in order to study the impact or the influence of participating in these communities, by analyzing the messages exchanged to understand the dynamics of the communities [49, 17, 8, 38]. For example, a study performed by Zhang et al. [49] involved gathering posts from a breast cancer support community, from the moment the user joined. A sentiment analysis of the posts overtime was performed, in terms of positivity and negativity. They came to the conclusion that users were more positive with a sustained participation and more inclined to provide positive replies to their peers' concerns as well.

Besides studying patients' sentiment, there has also been research aimed at establishing patterns in posted content in order to help detect health-related problems.

Outbreak detection

The automatic categorization of health-related messages in social media, according to the appearance of discussions or complaints about certain symptoms, has the potential of contributing to early detections of epidemics. Previous works [12, 13, 6] have shown it is possible to find correlation between the increase of posts mentioning certain symptoms and reported disease and weather conditions statistics from official governmental sources.

Drug adverse reactions detection

Since it is not possible to find all potential adverse reactions before marketing a drug to the public, further reaction detection depends heavily on reporting systems. However, such systems are not ideal due to the lower reporting rate. Instead, the attention has turned to tracking social media posts mentioning the new drugs for adverse reactions detection. Works in this area have shown that it is relevant to further continue research on this topic due to promising results [47].

Overall, these repositories of exchanged messages between users have many applications.

Table 2.2: Intersection of algorithms and datasets.

Algorithms	**Sources**		
	General Communities	**Health-related**	**Others**
Naive Bayes	[30], [43]	[9]	[5], [19], [20]
Support Vector Machines		[49], [9], [50]	[5], [20], [46], [22]
Logistic Regression	[30], [12], [13]	[49], [9]	[5]
Decision Trees	[30]		[20], [18]
k-Nearest Neighbors			[20], [46]
Rocchio			[20]
Bagging, Boosting		[49], [9]	
Winnow			[5]
Other, Own implementation	[6]	[47]	[5], [46]

2.4 Text categorization in online communities

The following subsections will present an overview of common methods and approaches, related to text categorization, used in research applied to this area.

2.4.1 Algorithms used

Table 2.2 shows which algorithms are used in different types of sources.

Several works make use of the *Naive Bayes* classifier [5, 30, 20, 43, 19, 9] and the *Rocchio* algorithm [20] as baseline classifiers for comparison against other systems, due to their simplicity, achieving mixed results.

The *SVM* classifier is often used and usually performs well, achieving good classification results [49, 5, 20, 46, 9, 22]. Zhang et al. [50], for example, apply the SVM classifier to different sub-communities of *MedHelp* with a high percentage of correct classifications.

Some works also use the *Maximum Entropy* classifier [49, 30, 5], which is a robust probabilistic model, similarly to the Naive Bayes classifier. However, it tends to have better performance due to a lesser amount of assumptions made. Different implementations of the *Decision Tree* algorithms [30, 20, 18] and k-Nearest Neighbors [20, 46] are also applied, with the former being more common for datasets with small to medium-sized messages.

Represented on the last line of Table 2.2, some authors also prefer to extend an algorithm with their own implementations, tuning it to the domain and greatly increasing the performance when compared to other state-of-the-art algorithms. This is the case, for example, of the implementation of the *Winnow* algorithm by Bekkerman et al. [5]. Xiaofei et al. [46] present a *Theme Word Subspace* learning algorithm where the classification is done according to the distances of the documents to subspaces of features.

Table 2.3: Feature selection in OHCs.

Features	Articles
Linguistic features	
Number of words	[49]
Number of sentences	[49]
Number of exclamation/question marks	[9], [49]
Ending in exclamation/question mark	[9], [8]
Use of question-related words	[8]
Average word length	[49]
Part-of-speech tags	[8]
Sentence patterns	[50], [8]
Presence of Domain-specific words	
Manually made vocabularies	[49], [50], [8], [8], [13]
Word frequency	[9], [8], [13]
Entities extraction (people, locations, etc.)	[49]
Existing medical vocabularies (symptoms, drugs, etc.)	[49], [50], [8]
Other	
Emoticon lexicon	[9], [49]
External sentiment classifier results	[9], [49]
Language-related vocabularies (nouns, verbs, subjectivity, etc.)	[9], [8], [8]
Positive and negative words	[9], [49]

Some approaches also use a voting technique, combining the decision of multiple classifiers to increase the probability of a good classification. Zhang et al. [49] use the boosting algorithm *AdaBoost*, obtaining high performance, similar to the performance of the SVM algorithm in the same study. Biyani et al. [9] make use of both boosting and bagging techniques.

2.4.2 Feature selection and vocabulary usage

Several characteristics of the document can be used as features, from statistics such as the length of the message to the content itself through the presence of certain words. Table 2.3 presents features and feature selection techniques used by research in the area of OHCs.

Some works use the **linguistic characteristics** of the documents as features, such as number of words and the presence of question and exclamation marks [49, 9]. Biyani et al. [9] use the presence of exclamation and question marks to denote the intensity of the posts.

The words present in the documents themselves are also often used as features. Many authors make their own vocabularies according to the words they expect to identify in the collection [13, 8]. These are mostly used in detecting words related with emotions, intent and medical terms (such as symptoms). Using an *emoticon* lexicon to help measure the positivity and negativity rate of a post is also a possibility [49, 9]. Regarding medical terms, most authors prefer to use the available

medical vocabularies and ontologies for their systems [50, 49, 8]. Using the words with the highest frequency as the features is also a common possibility [9, 8, 13].

Some studies also look at **sentence patterns** to extract information regarding the intent of the post. Biyani et al. [8] make use of simple patterns, such as < *I/We $opinion* > and < *You $advice* > to identify posts giving advice and opinions. In those patterns, *$advice* and *$opinion* represent a word from the vocabularies obtained by the authors when looking at the dataset. The *$advice* vocabulary contained words such as "must", "need" and "should" and the *$opinion* vocabulary contained, for example, "recommend" and "suggest".

Other possibilities for features include metrics calculated by running external sentiment-related algorithms [49, 9] and entities present in the texts, such as people mentioned [50].

Besides the aforementioned techniques, some other text categorization works also calculate the *Information Gain* of the terms, rank the words from the highest gain to the lowest and select an arbitrary amount of terms from the top words [20, 18]. Joachims [20] breaks up the entire ranking into several groups (from 1 to 200, 201 to 500, and so on) and runs the classifier on each, finding that even the lowest ranking words might still be relevant and impact the classification system.

It is also possible to use implementations of the LDA (*Latent Dirichlet Allocation*) algorithm to uncover a ranking of topics in large datasets [6].

2.5 Summary

This chapter's objective was to explain the concepts of text categorization and go through all the steps where some domain-specific decisions are made, in order to improve the classification system's performance. Alongside this, the chapter included an explanation on the current state of online health communities, from types to statistics, and descriptions on approaches being used in research where health-related messages are used, in terms of algorithms and feature vectors.

Although the focus of this dissertation is in *online health communities*, all types of message categorization were taken into consideration during the first part of this literature review on text categorization methods. The intent was to explore more approaches and different perspectives, since the literature focusing on health-related forums was found to be limited during this review.

In the end, it was possible to compile a list of techniques used in text preprocessing and in feature representation and selection, such as using the term frequency as the weight for the features and calculating the *Information Gain* of the terms to select which ones to use as features. This review also resulted in a collection of possible classifiers and evaluation metrics for classification systems, such as precision and recall.

For the specific application in the study of online health communities, a table compiling the targeted communities and the classifiers used was created for an easier understanding of which algorithms are applied to what situations. Another table resuming the features used in health-related research was also presented.

Chapter 3

Methodological Approach

Taking the objectives detailed previously in Chapter 1 as a guideline, this chapter aims to describe the general strategy followed in this dissertation. The following sections state the problem(s) we are trying to solve, the proposed solution as a pipeline towards automatic text categorization in online health communities (OHCs), and what elements compose this pipeline.

3.1 Problem Statement

In Chapter 1, the issue of the abundance of healthcare-related, user-created, data online was brought up. This identified a necessity for ways of studying healthcare discussion forums to characterize their content and possible value, and some means for automated filtering of user-created content, in order to give its readers tools to more reliably identify content of interest. This comes with a particular set of problems, though, from the gathering of the content itself, to the automatic filtering and even data presentation back to the interested party. Given this, a possible solution to these issues would have to come, at least in part, from answering the following questions:

- How can we study and characterize an online health community?
  - How to identify the range of topics discussed in the community?
  - How to characterize the user-base of the community? How do they interact with each other, how do they gather in sub-communities within the OHC?
- How can we reliably extract, process and analyze user-created content in online health communities?
  - What precautions should we take when choosing an online health community to extract content from?
  - What automated tools or approaches can we use for content extraction?

- How can we automatically categorize these messages, to more easily find helpful information, in the midst of possibly abundant content?
  - What categories to consider when categorizing the user-created messages?
  - What methods can we use for message categorization?
  - How can we evaluate the reliability of the obtained results?
- **Mainly:** Can we build a general, repeatable approach for characterizing an online health community and its user-base, and to extract, analyze and categorize its content?

Although there are several different techniques in the fields of web crawling, text processing or text categorization, due to the different domains these can apply to, there is a lack of complete sequences of tools that apply these techniques from start to finish, that is, from text extraction to automatic categorization and presentation, in the particular field of Online Health Communities (OHC). There are several programming libraries and tools for web crawling [21, 23, 35, 51], text processing [7] and text classification [45, 7, 37]; what we propose in this dissertation is an integrated *pipeline* of selected and/or developed tools and approaches to target the domain of online health communities: choosing a single OHC, studying it, and then extracting messages, processing them, and lastly categorizing them, according to a previously researched classification taxonomy 3.5.1.

3.2 A Pipeline Towards Automatic Text Categorization in Online Health Communities

The proposed methodology to answer the aforementioned questions, our integrated pipeline, can be divided into a conceptual sequence of steps, identified and studied in the context of this dissertation:

1. Identifying a suitable online health community - To list a set of rules that, on first analysis, tells us if the community under analysis gathers characteristics compatible with the methods of the proposed pipeline.
2. Extracting and studying information from the online health community - Methods for automatic extraction of content and user data from the communities (e.g. via data *crawling*) and to further characterize it according to proposed criteria. This includes:
   (a) Extracting topics, categories, sub-communities and messages from the community.
   (b) Extracting user profiles and user relationships from the community.
   (c) Characterizing the community based on the extracted information (descriptive statistics and analysis).

3. Processing and categorizing the extracted content - This step involves computational processing of the extracted content, specifically the user-created messages regarding health topics, in order to build training and classification datasets to be used in machine learning approaches. In complement to this, a step for manual text categorization is also proposed, to provide a way to create and expand sets of labelled data, for algorithm training and validation purposes. This brings us to the following sub-steps:

   (a) Manual text categorization.

   (b) Text processing for automatic categorization.

   (c) Automatic text categorization.

4. Presenting data to the end-user - Lastly, this step brings forward a brief analysis of how to show the results of the previous steps to the end-user, presenting a characterization of the online health community, its user-base, and their content.

The following sections offer the reader a more detailed description regarding each of the previously listed methodology steps.

3.3 The Dissertation Testbed Community: MedHelp

MedHelp [26] is an Online Health Community founded in 1994 where individuals can ask questions, share experiences and offer support in various ways regarding different health-related topics. Their online health discussion boards are split into different topics and sub-topics, such as General Health, Diabetes and Pregnancy.

MedHelp partners with doctors and researchers, so users of the community can get answers from professionals as well.

3.4 A Data Crawler for Online Health Communities

After identifying a suitable community to analyse, one must start extracting information from it. The data crawler is a piece of software that is able to automatically connect to the address of the online community and extract information from its *html*, *css* and eventually *javascript* code. This component is able to follow a predefined path of URL addresses associated to the community, that are considered to have information that is relevant to be extracted. That is, the component *crawls* through the pages, through a predefined path, interpreting the page code and extracting information as it does.

In the context of this dissertation, this crawler component is a tool that, for a community, explores each one of its sub-communities and their topics, and extracts each message, be it a question, a reply or a comment, if existent. In order to also build a dataset of contextualizing information regarding the community's users, the crawler component is also able to extract information from the profile of the user responsible for each of the above-mentioned content.

In a technical sense, for most scenarios, this is the actual first step of the data pipeline, as it is here that the data to be analysed will be gathered.

The following subsections provide more details regarding the concepts of the proposed community crawler.

3.4.1 Message Gathering

For the context of this document, any user-created textual content will be referred to as text or *message*. Online health communities often follow a typical forum structure:

- **Categories**, divided into:
  - **Sub-forums**, each divided into:
    - Further **sub-forums**, or
    - **Topics** (often also called ***threads***), which group a discussion through:
      - User **messages**

To further describe the work flow of the crawler component, we will use the specific example of the OHC used as a testbed in this project, *Medhelp*. With this particular OHC, we consider a different terminology to the terms mentioned above: *Communities* instead of categories, *sub-communities* and/or *user communities* instead of sub-forums, and *Questions* instead of threads. Questions and the discussions within them are the main user content of interest to be extracted from the *MedHelp* community. Each of these questions has a description text, written by the question author. Other users can give *answers* to the question, as well as *comment* on already existing answers.

When executed, the crawler component visits each community of the *MedHelp* web page and crawls through every question created so far in that community. For each of these questions, the crawler extracts all *message*-related information, that is, description text from the question, the content of each answer and comment, and all relevant metadata to each piece of content, such as author, creation date and its unique identifier (ID). The main source of potentially valuable or interesting information comes from the text in the user-created content. In general, the metadata complements this value, by enabling further descriptive and statistical analysis of the community, as well as providing context to whoever analyses the gathered dataset. In particular, the unique identifiers also enable us to maintain the relationship between questions and their answers and comments.

The extracted text and metadata are then stored into a database for further processing and analysis, as well as text files, which may provide a more useful format for other processing tools.

3.4.2 User Data Gathering

As mentioned previously, user data extraction was also made part of the data pipeline, in order to build a dataset of contextualizing information regarding a community's user base. This information might help with providing further insights about the content of the community.

The user data crawler works in a different execution flow from the content crawler that was described before. For this methodology, it was decided that the only user data that would be extracted would be from the profiles of users involved in content generation in the community. That is, only the profiles of those users that posted a question, an answer, or a comment. Users that have not created any data of this kind would not be as useful to characterize the community and its content.

For each piece of the aforementioned content stored in the pipeline's database, the crawler accesses the author's user profile and attempts to extract available information from it. Users may have opted to not fill out all possible information fields in their profiles, so it is possible that some users in the dataset are better characterized than others. When available, the fields we are able to extract include the following:

- Brief description ("about me").
- Location (city, state and/or country).
- Age.
- Date of registration.
- Interests (in health topics).
- Posting activity, including:
  - Date of latest question written by the user.
  - Number of posts made.
  - Communities and user communities they follow.
  - Communities where the user holds one or more "best answers" at (voted by other users).
- In-community friendships with other users.

The association between users and content they create is also stored.

We decided to create a dataset of user data for mainly two reasons. The first is that, as mentioned, knowing the characteristics of the userbase helps with characterizing any online community, and, in turn, evaluating the utility of the content this userbase produces. For example, answers from users with more "best answers" at a given topic will most likely be particularly helpful when seeking support or guidance relating to it. Another case of this would be that users from a given country might be able to provide more adequate answers to the specific healthcare context in that country. The second reason is that such a dataset can be useful for future machine learning developments in user and community analysis, either for training or testing steps, something to consider in future work after this project.

3.4.3 Notes and Considerations on Data Crawling

The development of the aforementioned methodology step, the data crawler, had a number of concerns taken into consideration.

User data privacy As the reader may have noticed, user profiles, depending on the degree of completion, can hold significant personal details that may make a user liable to be tracked within the online community or, in a worst case scenario, identifiable in real life. We propose a set of obfuscation actions in order to minimize these risks, when extracting and storing the user data, such as:

- Transforming each health community user ID into a different, but still unique, ID, that makes future users of the dataset unable to find the profile in the original online platform, without compromising user-post and user-user relationships.
- Perform a similar action as above to usernames too or, alternatively, remove them entirely, since they do not seem to add any relevant elements to community or user analysis.
- Removing or partially filtering the "About Me" section of the user profile, since, given it being an open text field, users can use it to share personally identifiable information.
- Transforming the user's age into an interval instead of a specific number (e.g. Age: 25 - 30, instead of Age: 27).

Handling large amounts of data *Medhelp* has a large number of communities, users and posts, as the reader may have noticed after reading the sections regarding statistical analysis. This means that the amount of data extracted from this web community is very large, even if not, necessarily, *"big data"* large. Each single piece of information needs to be processed and stored somewhere and, as data points tend towards the thousands, this generates significant overhead and resource consumption, at least at the level of typical consumer hardware, as was used during development of this dissertation project (e.g., a common laptop). Table 3.1 details some of the discovered issues and a brief analysis of possible solutions to these, which were taken into account during development of a prototype for this data crawling tool.

3.5 Approaches for Text Processing and Classification

This section now identifies and describes the approaches studied in this research work for implementing the text processing and classification step of the proposed data pipeline.

Besides the manual classification approach, the methods described here were chosen from an analysis of recent practical applications of natural language processing for intent and topic classification, sentiment analysis and dialog act classification to text corpora of different themes and contexts.

Table 3.1: Common problems associated with data crawling and proposed solutions

Problem	Possible solution	Pros	Cons	Implemented (Y/N)
Time to gather all data	Multithreaded / Multiprocess crawling	• Execution parallelism, considerably improves total extraction time. • Better failure tolerance (thread failing typically does not lead to whole crawler failing).	• Tougher to implement right. • Optimal number of threads/processes depends on system. • More network connections, usually, leading to throttling.	Y
	Server based crawling (cloud or on-premises)	• Allows better resource allocation. • Crawler possibly available in permanence. • Can implement incremental and automatic data updates.	• Needs additional development of APIs. • Network configuration necessary. • Extra server equipment for on-premises solution. • Cloud solutions usually require payment.	N
Storage required for data	Cloud-based storage server	• Allows better resource allocation and scalability. • No need for on-premises hardware concerns. • Remote availability of data, if needed.	• Time and resource considerations for data upload. • Cloud solutions usually require payment. • Needs additional development of server connections.	N
	Data compression strategies	• Textual data is usually compressed with good results. • Simple to implement. • No extra expenses (depending on process used).	• Takes time to compress large amounts of data (and later decompress). • Compressed data cannot be queried without decompression.	N
	Semi-manual data filtering (manual analysis and identification of automated filtering rules)	• Can obtain good filters when data is analyzed by experts. • Does not require much extra computational resources. • Promotes further understanding of data.	• Requires extensive analysis (visual inspection and statistical analysis) to find applicable rules. • Need to repeat manual process for adding new data. • Need to implement the automated filters. • Time consuming for large amounts of data.	Y
Website server overload during crawling	Increase time between consecutive server connections	• Proper delay settings usually eliminates this issue. • Simple to implement.	• Increases time of data extraction. • Requires testing to find adequate parameters.	Y
	Optimization of parallel server connections	• Preventing too many parallel connections may avoid overload. • Saves local system resources and bandwidth as well.	• Requires testing to find adequate limits. • Those limits may change with fluctuations in server load. • May increase time of data extraction.	Y
	Source connection obfuscation (e.g. connection via proxy)	• Dynamic obfuscation prevents blocking of crawler connections.	• Usually complex to implement. • Tends to slow down connections. • Usually against websites' terms of service. • May trigger detection of suspicious actions on remote server.	N
Slow data querying in database	"Big data" focused solutions	• Usually, scalable and future-proofed tools. • Brings further efficiency to data processing.	• Increases complexity of implementation. • Some solutions are paid. • Requires studying and learning new approaches and tools.	N
	Improvements to database design (e.g. data indexing)	• Can significantly improve performance of database operations. • Generally simple to implement. • "Lessons learned" from improvements can be applied to future databases.	• All changes must be tested, some may actually be detrimental to performance. • Tougher to apply when database already contains data. • Particularly for SQL databases, some changes can be complex to apply.	Y

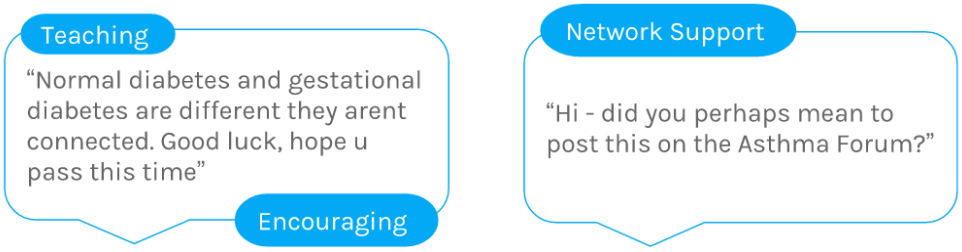

Figure 3.1: Example of two different messages of the existing dataset and their respective categories.

An important criterion when making this choice was the applicability of the method to smaller data sets, given the small size of our sets when compared to some of the most commonly used sets of text data in recent applications. This tends to exclude the usage of deep learning approaches, unless applying pre-trained models or attempting more time-consuming transfer learning approaches to complement those. Other criteria taken into account for this selection included: described classification success and error rate, apparent ease of usage through commonly available software tools and speed of implementation and testing.

3.5.1 The classification schema

The categories used for the message classification in this dissertation are based on the work by Bárbara Silva [42] done in 2016. In this schema, the author highlights four main categories of possible intents conveyed by messages in health forums:

- **Seeking support** - Covers all messages asking for help, which includes seeking information and asking for reassurance;
- **Offering support** - Refers to all messages that have the purpose of helping someone, which can be reflected in the form of giving health-related information (based on medical knowledge or personal experience), offering emotional support and validation, providing help regarding technical issues related to the forum, and so on;
- **Group interaction** - Includes all messages related to general user interactions, such as thanking someone, congratulating someone or sharing a personal experience;
- **Emotions** - Contains a range of basic emotions displayed, such as sadness and happiness;

The schema has two additional layers of categories, each one offering further details on the specific intent conveyed by the message. The classification schema is detailed in Table A.1.

In the case of this classification schema, each message may belong to more than one category, as shown as an example in Figure 3.1. Thus, classifying messages using this schema can be considered a multi-label classification problem.

Table 3.2: The classification schema as described in Bárbara Silva's work [42]

Categories		
Seeking Support		Specific question
		Reassurance
Offering Support	*Information Support*	Advice
		Recommendation
		Teaching
	Emotional Support	Affection
		Sympathy
		Encouragement
		Prayer
		Relief of blame
	Esteem Support	Compliment
		Validation
	Network Support	Access
		Presence
	Tangible Assistance	Perform direct task
		Express willingness
Group Interaction		Gratitude
		Congratulations
		Sharing personal experiences
Emotions	*Negative*	Anger
		Fear
		Sadness
	Positive	Happiness

3.5.2 Message datasets

For the purpose of this dissertation, in particular when exploring the steps of message processing and classification, we used two sets of messages.

The first dataset originates from the work of Bárbara Silva [42], already mentioned before, which was manually labelled as part of the author's research. In the following sections of this document, we shall refer to this dataset as the **preliminary dataset**. The preliminary dataset was used as an exploratory dataset for study of possible database architectures and initial testing of classification approaches.

The second dataset is comprised of content extracted from the online health community described in subsection 3.3, using the data crawler developed for this dissertation and following the methodology described in previous sections of this document. A subset of these extracted messages were manually classified through the manual categorization web interface described in subsection 3.5.4. This dataset will be referred to as the **extended dataset**.

It is worth mentioning that both datasets originate from the same online health community, with the first one having gone through a more selective and manual process of extraction. A brief description of each of these datasets now follows.

3.5.2.1 Preliminary dataset

The first phase of categorization method development and testing made use of manually annotated messages which resulted from Bárbara Silva's master's thesis [42]. This preliminary dataset is a combination of the annotated messages from different phases of schema evaluation in the author's work.

It totals 245 messages extracted from the online health forum *MedHelp*, which we previously described in subsection 3.3 of this document. All the messages were categorized following the schema presented in 3.5.1 by a group of different judges. The agreement between the different judges was measured using the *Cohen's kappa* coefficient, as described in Silva's work.

Due to the small size of this dataset, especially when compared with the total amount of messages the *MedHelp* currently holds, we opted to generate a second dataset, manually annotated, and to be used in conjunction with this existing set, believing this would ease and improve the application of natural language processing tools. To be consistent, these new messages were extracted from the same source community and categorized by us and volunteers. This *extended dataset* is described next.

3.5.3 Extended dataset

For the characterization of an OHC during the second phase, it is important to extract another set of messages, ideally larger than the previous one. This second set can be used to validate pipeline approaches and to be manually annotated and labeled for further text classification usage. We chose to extract an additional number of messages from the community of the original dataset, *MedHelp*, using the previously detailed data crawler. In order to expand the applicability of this

dataset and do some more thorough testing of the pipeline approaches, we made sure to feature messages from sub-communities not originally contemplated in the preliminary dataset. This also gives us more confidence that our methods can be applied to a wide spectrum of medical topics. The extended dataset totals 434 manually labelled messages.

3.5.4 A Web Interface for Manual Text Categorization

Subsection 3.5.3 raised the necessity of manual annotation and labeling of datasets. Manual annotation of data is usually done by volunteers, often with some kind of compensation (monetary or otherwise) to reward the annotation efforts. With the objective of making this task as effortless and engaging as possible, we opted to design an interface for manual classification of the messages extracted from the online health communities. The following are the characteristics that guided the development of this tool:

- **Web based**, using modern web technologies, in order to make the interface easily accessible (via browser) and pleasant to use.
- **Simple** and **intuitive**, making sure that the user would not need to go through a learning period before being able to fully make use of the annotation tool.
- **Engaging**, trying to derive satisfaction from using the tool, and thus motivate the user to classify a larger number of messages.
- **Complete** and **self-contained**, making sure all annotation and classification capabilities, as well as usage instructions and explanations, are contained in a single tool and easily accessible from the main screen.

A first time user of this tool would be brought to a welcome screen, detailing its objectives and explaining a few first steps towards its usage. After this screen would come the main classification interface. This interface would show a single message, with any immediately identifying elements (such as its ID in the community and the username of the original message creator) removed. Next to the message would be a set of buttons representing each of the possible classification categories for the message. The user would click one, several or none of these buttons in order to classify the message with multiple possible tags. In case no tags suit the message, or the user simply wants to classify a different one, there should be a way for the user to skip the current message and be presented with a new one.

As part of this dissertation, a prototype of this web interface for manual text categorization was developed. The reader can consult further details of this prototype in Section 4.3 and consult screen captures of the main screens of the tool in Appendix B, where it is possible to see how the aforementioned usage flow was implemented.

Another addition we believe would be useful for implementing a manual categorization tool would be elements of *gamification*. *Gamification* promotes engagement by applying typical game

concepts to the usage of the tool. Such concepts could be keeping a score of total messages classified, leaderboards of multiple user scores, to promote friendly competition, or possibly digital awards for usage achievements (e.g. a congratulations message when the user classifies 100 different messages). As of the time of writing, we took into account the development of a user score element to the tool prototype.

3.5.5 Vector Cluster Similarity

This is a supervised learning approach, depending on a pre-classified set of training data. This approach should work for both single label but also multi-label classification problems. The main idea for this method is to represent each target classification label as a cluster of documents from the training set (in this case, a cluster of messages) that were classified with the corresponding label. We call these clusters *vector clusters* because we transform each belonging document into a numeric vector using BERT embeddings. The classification of unlabeled data is then made by analysing the similarity of each document, represented as a vector of BERT embeddings, to the vector clusters, and picking the most similar clusters as its labels.

The following describes the basic set of steps in this approach:

1. The training data is processed in order to transform the text of each message in the set into a numerical vector, using a method such as a BERT tokenizer, which was used in this dissertation.
2. A set of numerical vectors will be generated for each of the classification categories, corresponding to each message with the given classification.
3. This set of vectors is averaged out into a single numerical vector, representing the category.
4. A similar numerical vector transformation is applied to a message we want to classify.
5. The resulting numerical vector from the previous step is compared to each of the average numerical vectors that represent each classification, using a similarity function to calculate its similarity to every possible classification label.
6. The classification with the highest similarity score is chosen for the new word.

3.5.5.1 Input data description and processing

The following describes common steps taken to prepare data both for training the classifier and for classification.

- Sentence segmentation: The messages in the dataset are segmented into individual sentences, following common sentence start words and/or punctuation.
- Word tokenization: Each sentence from the above mentioned step is turned into a set of words, removing duplicate words.

- Stop word, punctuation, lemmatization processing: Common stop words are removed from the previous word sets, as well as punctuation. Lemmatization takes care of transforming different forms of a word ("inflected forms") into a single, base form. Stop word removal and lemmatization are optional processing steps as they may consequently remove some important meaning and context from the messages.

- Generate BERT embeddings for each set of words. Embeddings are numerical vectors that encapsulate the meaning of words. They are useful for representing text as similar sized numerical features, making it easier to compare these features and use them in machine learning algorithms. BERT (Bidirectional Encoder Representations from Transformers), in particular, is a pre-trained model published by Google that is, nowadays, frequently used to create such vector representations.

  BERT is considered a more accurate language representation model than other common ones, such as Word2Vec, since it is able to use the context of other words in a sentence to derive a better numerical representation of a single word. In here we see the importance of keeping the original word order during the previous text processing steps, and perhaps skip stop word and lemmatization steps to allow BERT to derive more meaningful embeddings.

  This process will generate a numeric vector for each sentence in the message.

- Lastly, we do an average of all numeric vectors in the set generated in the previous step. This allows us to represent a message through a single numeric vector.

3.5.5.2 Generating the label clusters

We start by applying the aforementioned pre-processing steps to a labelled dataset, after which we will be left with a group of numerical vectors from the BERT embeddings of each message. For each possible label, we can then generate a group of vectors corresponding to the messages with that classification. Keep in mind that, if you consider that each message can have several labels (a multi-label classification problem, then), it is possible for the same message to be present in several groups.

After this segmentation of messages and their numerical vectors, we proceed again with averaging all the vectors within each group, resulting in a single numerical vector (the cluster mean vector) for each possible label, acting as the center for a label cluster.

3.5.5.3 Classifying the data

With our cluster vectors created, we can start classifying unlabeled data. Each document to be labelled goes through the aforementioned pre-processing steps, until we achieve a single vector representing each document. The classification is now made by calculating the similarity between the document's vector and each label's cluster mean vector, using a measure of similarity. One such measure is *cosine similarity*, which was used in the present work. This particular measure of similarity uses the cosine of the angle between two, non-zero, vectors, which should give us

a number between -1 and 1. One vector is more similar to another one the closer this measure's value is to 1.

Thus, with this approach, we can calculate the similarity of a document to each label cluster. The document is then classified by the label of the most similar label cluster. In order to allow for multi-label classification, a similarity threshold can be defined, and any label whose cluster's similarity is above that threshold, can be considered as a classification label.

3.5.5.4 Related Word Lists, an alternative for label cluster generation

In previous sections of this document we mentioned the small size of the original labelled dataset we were working with. Recognizing not only the need for larger datasets when utilizing supervised classification methods, but also the usual difficulty of obtaining, or creating, these sets, we propose here an alternative training set of data generated through *Related Word Lists*.

As the name implies, related word lists are lists of words that relate with a given topic. In this case, we consider the labels, for which we want to generate training data for, as the *seed* to generate these word lists. There are several ways to obtain related words, including by manual analysis, using natural language processing models to find similar words or combinations of approaches. These word lists are our "document" clusters that we use to calculate each labor cluster's mean vector.

Having obtained our cluster mean vector, the rest of the process is similar as before: calculate the similarity of a non-labelled document's vector to the cluster mean vector. Then choose labels according to the defined similarity thresholds.

3.5.6 Dialog Act Classification

Dialog acts are individual utterances by speakers, serving a specific function, in the context of conversation. These acts can be classified in several types, such as questions, statements, expressions of gratitude or apologies, among many others.

If we consider that a question thread, in an online health community, is a sort of conversation among multiple speakers, the kind of labels we are trying to classify their messages with can be seen as a specification of more general dialog acts: a speaker is asking for help, another is giving advice, a third one is offering their affection, and so on. As such, we propose this as another possible method for message classification, through dialog act classification.

As the reader may recall, Bárbara Silva included an "Emotions" category of classification in the classifications table the author proposed, as listed in Table A.1. In the context of this dissertation work, we considered that emotions would not be suitable to be classified as dialog acts, since they can be seen as more of an aspect imprinted on the dialog act by the speaker, instead of its objective function. For example, one can offer advice (the dialog act) compassionately (the emotion associated to the act), but also angrily, out of exasperation, or authoritatively. We point

out, however, that these considerations are subjective and should be open for discussion, but this is outside the scope of the current document.

3.5.6.1 Considerations on the dialog acts used for classification

Ideally, the classification model used for our dialog act classification would be trained on the specific dialog acts were interested in, the ones listed on Table A.1. However, given the small size of the available labelled dataset, as well as time constraints, we opted to use a pre-trained classifier to perform the classification. This particular classifier was chosen given its availability as a ready-to-use Python package, reported accuracy and ease of use. According to the author, the classifier has two versions, depending on the pre-trained model used as its base: one being the *BERT base model (uncased)* [14], by Jacob Devlin et al., and, the other, the *DistilBERT base model (uncased)* [40], by Victor Sanh et al. (using a distilled version of the BERT base model). This base pre-trained model was then fine-tuned for the dialog act, as a multi-class classification problem. Unfortunately, even after contact attempts, the author has not yet further detailed the exact parameters of that fine-tuning.

Table 3.3 lists the 38 possible dialog acts used by the classifier, as extracted from the classifier author's Github repository information. The dialog acts are called dialog "tags" by the classifier's author.

After achieving these "general" dialog act classifications for each message, it is necessary to make a correspondence to the specific taxonomy we would like to use. Since each message, when classified, is associated with multiple dialog acts (at the very least, one for each sentence), we decided to create a correspondence table that matches combinations of generic dialog acts to specific labels in our desired taxonomy. Table 3.4 illustrates the correspondences used. Therefore, after performing the initial dialog act classification of each message, our classifier then proceeds to use the correspondence table to apply the correct label to the message, from within Bárbara Silva's proposed taxonomy.

3.5.6.2 Input data description and processing

Since, at the moment, there's no classifier training step in this method, the input data is essentially the textual content of the messages we want to classify. These messages were already described in previous sections of this document. For each message, we apply pre-processing steps as described next.

The following is a list of steps we propose performing on unlabeled data, to prepare it for classification. The reader will verify that these steps were also present in the vector similarity approach we described earlier in the document.

- Sentence segmentation: The messages in the dataset are segmented into individual sentences, following common sentence start words and/or punctuation.

Table 3.3: The dialog acts as used by the classifier

Dialog act (or tag)	**Example**
Statement-non-opinion	"Me, I'm in the legal department."
Acknowledge (Backchannel)	"Uh-huh."
Statement-opinion	"I think it's great"
Agree/Accept	"That's exactly it."
Appreciation	"I can imagine."
Yes-No-Question	"Do you have to have any special training?"
Yes answers	"Yes."
Conventional-closing	"Well, it's been nice talking to you."
Uninterpretable	"But, uh, yeah"
Wh-Question	"Well, how old are you?"
No answers	"No."
Response Acknowledgement	"Oh, okay."
Hedge	"I don't know if I'm making any sense or not."
Declarative Yes-No-Question	"So you can afford to get a house?"
Other	"Well give me a break, you know."
Backchannel in question form	"Is that right?"
Quotation	"You can't be pregnant and have cats"
Summarize/reformulate	"Oh, you mean you switched schools for the kids."
Affirmative non-yes answers	"It is."
Action-directive	"Why don't you go first"
Collaborative Completion	"Who aren't contributing."
Repeat-phrase	"Oh, fajitas"
Open-Question	"How about you?"
Rhetorical-Questions	"Who would steal a newspaper?"
Hold before answer/agreement	"I'm drawing a blank."
Negative non-no answers	"Uh, not a whole lot."
Signal-non-understanding	"Excuse me?"
Conventional-opening	"How are you?"
Or-Clause	"or is it more of a company?"
Dispreferred answers	"Well, not so much that."
3rd-party-talk	"My goodness, Diane, get down from there."
Offers, Options Commits	"I'll have to check that out"
Self-talk	"What's the word I'm looking for"
Downplayer	"That's all right."
Maybe/Accept-part	"Something like that"
Tag-Question	"Right?"
Declarative Wh-Question	"You are what kind of buff?"
Apology	"I'm sorry."
Thanking	"Hey thanks a lot"

- Stop word, punctuation, lemmatization processing: Common stop words are removed from the previous word sets, as well as punctuation. Similarly as with the vector cluster similarity method, stop word removal and lemmatization are optional processing steps, as they may consequently remove some important meaning and context from the messages.

3.5.6.3 Applying the classifier

After executing the data processing steps, we are left with a set of sentences representing each message in the unlabeled dataset. We then execute the classifier once for each sentence in a message. We store these classifications into a label set representing the classification of the whole message, ignoring any duplicate label that may come from two sentences being classified with the same dialog act. By the end of this step, each message will have been labeled with multiple, different, labels.

3.5.6.4 Finding correspondences to our taxonomy

As mentioned earlier in this section, ideally, we need to make correspondences from the result set labels to the taxonomy being used in this dissertation. We opted to do this by matching either individual labels, or combination of labels, in the result set, to labels on our taxonomy. Table 3.4 shows how the correspondences are made, at the moment of writing.

3.5.7 Emotion Analysis

For the purpose of classifying messages with the labels in the "Emotions" category of our taxonomy, we opted to use tools for emotion analysis on text. While sentiment analysis concerns itself, mainly, with saying if the overall sentiment of a piece of text is negative or positive, emotion analysis tries to determine what emotions transpire from it, in more detail than just "positive" or "negative". These emotions can be happiness, anger, surprise, sadness, among others.

The process of emotion analysis is to pre-process the textual content we want to classify, a step that is described next in this document. After this, we apply a classification algorithm to ascertain the different emotions present in the text.

3.5.7.1 Input data description and processing

Likewise with previous classification steps, the input data is the textual content of the messages we want to classify. This data is processed, by transforming the text to make it simpler and removing unnecessary textual elements.

The processing being done at this stage is as follows:

- Tokenize the textual content into individual sentences, as described previously for other classification approaches.
- Remove stopwords, as previously described as well.
- Transform word abbreviations and shortcuts into full words (like "idc" to "I don’t care", "ty" to "thank you", among others).
- Apply lemmatization, following processes already described to the reader.
- Transform some commonly used *emojis* to emotions.

Table 3.4: Correspondence of general dialog acts to specific OHC taxonomy

Categories			**Dialog Acts**
Seeking Support		Specific question	Yes-No-Question
			Wh-Question
			Declarative Yes-No-Question
			Open-Question
			Tag-Question
			Declarative Wh-Question
		Reassurance	Yes-No-Question + Fear/Sadness/Anger
			Wh-Question + Fear/Sadness/Anger
			Declarative Yes-No-Question + Fear/Sadness/Anger
			Open-Question + Fear/Sadness/Anger
			Tag-Question + Fear/Sadness/Anger
			Declarative Wh-Question + Fear/Sadness/Anger
Offering Support	*Information Support*	Advice	Statement-opinion
			Statement-opinion + Yes answers / No answers
			Statement-opinion + Affirmative non-yes answers / Negative non-no answers
			Statement-opinion + Dispreferred answers
			Statement-opinion + Action directive
		Recommendation	N/A
		Teaching	Statement-non-opinion + Agree/Accept
			Statement-non-opinion + Yes answers / No answers
			Statement-non-opinion + Affirmative non-yes answers / Negative non-no answers
			Quotation
	Emotional Support	Affection	N/A
		Sympathy	N/A
		Encouragement	N/A
		Prayer	N/A
		Relief of blame	N/A
	Esteem Support	Compliment	N/A
		Validation	Appreciation
	Network Support	Access	N/A
		Presence	N/A
	Tangible Assistance	Perform direct task	N/A
		Express willingness	N/A
Group Interaction		Gratitude	Thanking, Thanking + Appreciation
		Congratulations	N/A
		Sharing personal experiences	N/A

3.5.7.2 Emotion classification

While possibly better emotion classification models may exist in the current state of the art, for the purposes of this project a dictionary based algorithm, developed by Aman Gupta , was chosen to perform this classification. This algorithm uses a dictionary that corresponds each possible emotion to a list of words that commonly convey that emotion, for the moment considering the emotions of *Happiness*, *Anger*, *Sadness*, *Surprise* and *Fear*.

Table 3.5 lists the emotions in the previously mentioned dictionary, and a short example of words associated with them.

Table 3.5: Example of emotions dictionary

Emotion	**Words**
Happiness	*adore, affectionate, congratulations, fondly, happily*
Anger	*angry, abhorrent, coldhearted, enraged, infuriating*
Sadness	*bereaved, brokenhearted, cheerless, demoralizing, sad*
Surprise	*amazed, astonished, incredible, mystified, surprising*
Fear	*anxiously, apprehensive, boding, dread, scary*

The algorithm goes through each word in the sentence being classified and attributes to it the corresponding emotion. Doing this, it finds the distribution of emotions throughout the sentence, by calculating the number of times that emotion appears, divided by the total number of words in the sentence (after pre-processing), and outputs this as a result. Table 3.6 shows the results of the algorithm for a few example sentences.

3.6 Summary

This chapter presented in detail the proposed approach for constructing a pipeline towards automatic text categorization, in online health communities.

We started by raising the questions about how to properly study and characterize an online health community, in order to evaluate the possible interest of its contents to a researcher or a user looking for health. From this followed also the questions about how to reliable extract, process and classify this kind of content. As a possible methodology to answer this, we proposed a pipeline to

Table 3.6: Example of emotion classification

Sentence	**Happiness**	**Anger**	**Sadness**	**Surprise**	**Fear**
*I am so **happy** that you managed to finally find a cure for your **disease**, you have a really **good** doctor!*	0.67	0.0	0.33	0.0	0.0
*I can't continue to see my local physician he's an **ignorant bastard** and I always get so **mad** when I see him!*	0.0	0.67	0.33	0.0	0.0
*It's **incredible how** fast things progressed, I'm **heartbroken** and just want to **cry**...*	0.0	0.0	0.5	0.5	0.0

help with automatic text categorization, in online health communities, that would tackle each of these concerns.

Afterwards, we proceeded with providing some general pointers on how to find an online health community that could be a suitable target for the aforementioned pipeline, both in structure and content. Having found that community, we proposed a data crawler for message content and user profile gathering, listing also some concerns with this approach.

Once the content has been gathered, we can finally move to classifying it, according to a classification schema that we detailed, for the reader. For situations where we need further labelled content, for model training or cross-validation purposes, we proposed a manual classification web tool for researchers and volunteers to be able to annotate datasets. Following this, we then listed the automatic classification methods we considered for the pipeline, along with the content pre-processing steps necessary to apply those methods. In sum, the automatic classification methods we propose are the Vector Cluster Similarity approach, the Dialog Act Classification approach and, complementing these, the Emotion Analysis algorithm.

The next chapter will provide the reader with the main results obtained during development and testing of this pipeline, along with their analysis and some relevant notes for further consideration.

Chapter 4

Results and Analysis

As a way to evaluate the applicability and the usefulness of the pipeline proposed in Chapter 3, we developed prototypes of several of the steps the pipeline integrates. This chapter identifies the main results of these prototypes, along with their analysis after preliminary testing.

We will start by further detailing to the reader the statistical analysis of the preliminary dataset, as introduced in section 3.5.2.1, as well as the unlabeled data obtained by running the data crawler on the MedHelp online health community. Following that, we will analyse our prototype for the Manual Text Classifier web tool. Lastly, we will share with the reader the results of our implementations of the automatic text classification tools that were detailed in the methodological approach.

4.1 Dataset analysis

4.1.1 Statistical analysis of the preliminary dataset

In total, this dataset contains 245 messages taken from MedHelp and manually categorized by a group of judges. In Table 4.4, we can see the number of messages per category. The category with the biggest amount of messages is "Specific question" with 68 messages being classified as such. Some categories, such as "Compliment" and "Validation", contain no messages. This distribution can also be seen in Figure 4.1.

4.1.2 Statistical analysis of the crawled data - *MedHelp* dataset

In order to get an overview on the activity of the different forums and subforums, a statistical analysis was done on the entire dataset of messages taken from MedHelp. This assessment was done in December 2021 and took into account all the messages exchanged since its opening. In total, 398 939 threads were opened and 1 812 204 replies and 93 064 comments were exchanged, the oldest message being from 1999.

The community is split in different topics and sub-communities. In Table 4.1, we have an analysis of the ten topics with the highest number of messages exchanged. Neurology is the most

active topic, containing 17% of all messages in the forum. This topic covers many different sub-communities, such as Migraines and Headaches and Multiple Sclerosis. The activity of each of these Neurology sub-communities is in Table 4.3. The complete list of all topics and total number of messages exchanged is in Table C.1.

In Table 4.2, we can see how the activity in the community changed throughout the years. In 2017, we observe a ratio of 1.2 answers per thread, the lowest of all years. The highest ratio of 10.84 replies per thread happens in 2000, when the community was just starting, despite the low number of threads and replies overall in that year.

Another interesting event happens between 2015 and 2016, where we see a steep decline in the number of answers (from 120 275 replies to 46 305), despite the number of threads not decreasing significantly (from 28 110 to 27 701 threads, respectively). The reason for this decline is not clear. It could have stemmed from the increased adoption of larger scale social media platforms by a wider audience, as it coincided, for example, with the launch of new developments in the *Groups* feature of *Facebook*, a direct competitor of traditional webforums. It is also possible that the MedHelp forum went through significant management changes and user interface updates during that period, common causes for user base exodus events, as seemingly evidenced by the forum's 2015 [27] and 2016 [28] archives provided by the Internet Archive's *Wayback Machine* [2].

Table 4.1: The ten topics in MedHelp with the highest number of messages exchanged.

#	Topic	Total threads	Total answers	Total comments	Combined	% of all messages
1	Neurology	57147	328852	6275	392274	17.02%
2	Hepatitis	19142	171025	3980	194147	8.43%
3	Pregnancy	24335	99943	2710	126988	5.51%
4	Thyroid	17020	101076	6452	124548	5.41%
5	Anxiety	18778	85436	4417	108631	4.71%
6	General Health	20446	70763	13197	104406	4.53%
7	Digestive	22111	76884	2289	101284	4.40%
8	Addiction	10110	79501	4179	93790	4.07%
9	Dermatology	23115	64065	3185	90365	3.92%
10	Heart Disease	18302	66123	1700	86125	3.74%

4.1.2.1 User profile data

An analysis was also performed on the dataset of user profiles extracted from MedHelp. This assessment, as with the message data, was performed in December 2021, taking into consideration all accounts created since the opening of the webforum in 1999. In total, there are 436 403 registered users, gathering in 456 communities and 755 custom user groups. Users are able to create friend relationships between them, currently amounting to a total of 261 792 relationships in the webforum. Table 4.5 shows the distribution of users per year of their registration, as well as the average number of relationships per user and year. Note that users with no registered friend relationships weren't included in this evaluation. 87 users had no year of registration available on their profiles, probably due to being maintenance or test user profiles. We can see that the year

Table 4.2: Number of threads and answers in the MedHelp community per year, as well as answers per thread ratio.

Year	Total threads	Total answers	Answer per thread
1999	14	52	3.71
2000	50	542	10.84
2001	79	702	8.89
2002	138	738	5.35
2003	107	1033	9.65
2004	158	1356	8.58
2005	412	2730	6.63
2006	2204	11390	5.17
2007	20808	92737	4.46
2008	54210	228701	4.22
2009	55767	218015	3.91
2010	63311	249992	3.95
2011	55263	222115	4.02
2012	51414	225525	4.39
2013	42056	171553	4.08
2014	33656	141953	4.22
2015	28110	120275	4.28
2016	27701	46305	1.67
2017	19256	23139	1.20
2018	17780	21468	1.21
2019	8682	13367	1.54
2020	7281	10965	1.51
2021	5475	7551	1.38
Total	**493932**	**1812204**	**0.27**

with most newly registered users was 2010, with the number of new registrations dwindling from that year forward, having a sharp decrease in 2019. This data seems to follow the trends identified previously, when evaluating the message dataset. We can also see that users that have been on the forum for longer maintain, on average, a higher number of relationships with other MedHelp users.

4.2 Analysis of the Data Crawler

In this section we will detail to the reader the main results obtained with the developed prototype of an OHC Data Crawler. This section will describe some characteristics of the dataset extracted with the tool, details and metrics of its execution, configurations tested and the reasoning behind them. We will also share some issues found during development and how they were mitigated.

Table 4.3: Number of messages (threads, answers and comments) exchanged in the Neurology topic per sub-community.

Neurology sub-communities	**Threads**	**Answers**	**Comments**	**Total messages**
Multiple Sclerosis	27818	198810	1615	228243
Chiari Malformation	9981	65772	1944	77697
Neurology	13932	45764	2063	61759
Stroke	1300	4893	182	6375
Migraines and Headaches	1104	4490	282	5876
Traumatic Brain Injury	1032	2627	47	3706
Epilepsy	402	1239	28	1669
Spinal Cord Conditions/Disorders	330	1083	29	1442
Amyotrophic Lateral Sclerosis (ALS)	308	882	11	1201
Cerebral Palsy	237	851	6	1094
Trigeminal Neuralgia	231	825	23	1079
Tourette Syndrome	114	423	8	545
Brain (Cerebral) Aneurysm	84	314	17	415
Peripheral Nerve Hyperexcitability (PNH)	106	281	6	393
Restless Leg Syndrome	57	211	10	278
Sensory Integration Disorder (SID)	35	123	0	158
Muscular Dystrophy	42	108	2	152
Ataxia	22	87	1	110
Pediatric Tourette Syndrome	12	69	1	82

4.2.1 Extracted data and execution metrics

4.2.1.1 Message data

Section 3.4.1 described our objective of gathering message content from online health communities, in order to create useful datasets for future work. This content is, for the most part, health-related questions, answers and discussions between users in online forums. We are mostly interested in the textual content of the messages exchanged during discussion, along with useful metadata such as user evaluations of content (e.g. number of *upvotes*), time of creation and modification and what community (or sub-forum) the message comes from. Details regarding the attributes of this sort of data were already shared in the section of the methodological approach that was mentioned before.

A total of 2 304 207 messages were extracted, distributed among over 234 communities, after a total run time of approximately 15 hours and 30 minutes. Table 4.6 lists metrics from the execution of the data crawling tool.

4.2.1.2 User profile data

As described in Section 3.4.2, we attempted to create a dataset of OHC user data. Given the lack of a complete user directory on the *MedHelp* website, and to avoid having to crawl the whole forums all over again, we opted to gather the profiles of only the users that posted the messages extracted

Table 4.4: Number of messages in both labeled datasets

Category	**No. messages (Preliminary)**	**No. messages (Extended)**	**Additional messages**
Access	1	4	3
Advice	43	58	15
Affection	1	8	7
Anger	1	4	3
Compliment	0	2	2
Congratulations	0	2	2
Encouragement	3	7	4
Express willingness	3	7	4
Fear	12	29	17
Gratitude	11	20	9
Hapiness	9	14	5
Perform direct task	0	2	2
Prayer	1	4	3
Presence	2	8	6
Reassurance	16	30	14
Recomendation	2	6	4
Relief of blame	0	11	11
Sadness	6	9	3
Sharing personal experiences	46	64	18
Specific question	68	92	24
Sympathy	0	9	9
Teaching	20	40	20
Validation	0	4	4

beforehand by the crawler. This approach also ensures that the users taken into consideration were somewhat active in the website some time in the past, as active message posters, which means they probably have more useful profiles than totally passive (i.e. non participating) forum users.

A total of 436 403 user profiles were extracted, distributed by 456 communities and 755 user groups, after a total execution time of approximately 10 hours and 50 minutes. The main profile attributes we were able to gather from the users were already described in the above mentioned section of the methodological approach. Besides the profile information, we were also able to map the relationship between users, as described in their "Friends" section, and store these relationships in the database. Alongside the user profile data, which explicitly characterizes the user, some information about user communities and groups was also extracted, which may allow future work to infer other characteristics about the participating users. In Table 4.7 the reader can observe metrics from the execution of the user profile data crawling tool.

It is possible to verify that the rate of user profiles processed per second is lower than the number of messages processed per second, when comparing with Table 4.2.1.1. While for message extracted it is only necessary for the crawler to traverse a single web page (the thread, or question, page), for each user profile the crawler needs to access and crawl through multiple pages, implying

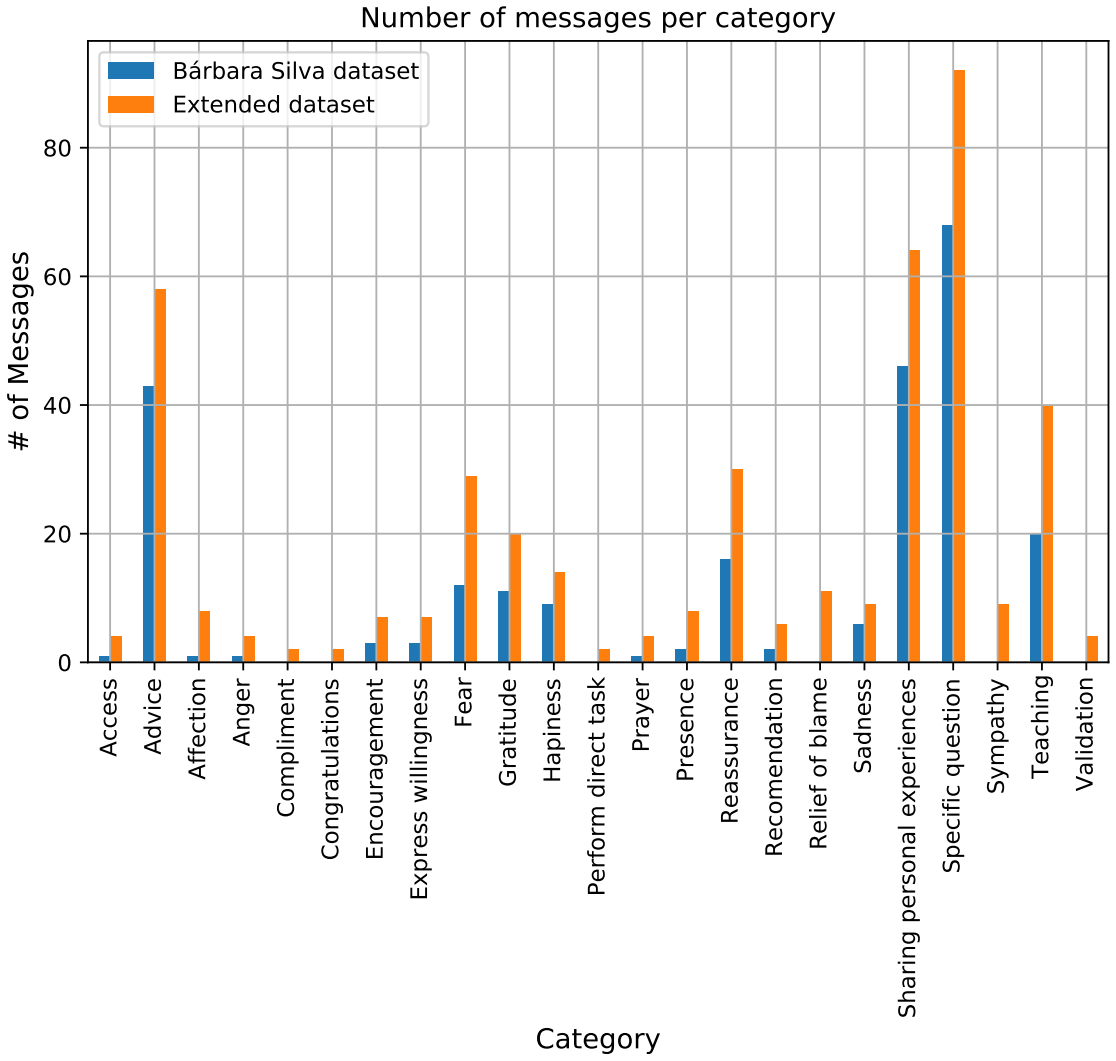

Figure 4.1: Comparison of number of messages per category, between the two datasets

further server connections for content retrieval, and therefore slower speeds. This happens because the information is spread over different hyperlinks instead of concentrated in the user profile page. For example, the friends list for a user is displayed in a different page, linked in the profile (which requires authentication to access). In addition, while for each message we are mostly interested in its content and related metadata, for a user profile there are a multitude of useful attributes, which need higher amounts of text and *HTML* processing to retrieve, as well as traversing paginated lists of data, which contributes to the slower processing speed.

4.2.2 Technical details and notes

In order to simplify testing of the tools and allow their adaption to different data extraction situations and system characteristics, we included the possibility of configuring some of its execution

Table 4.5: Users and their average number of relationships, per year of registration

Year	New users	Average relationships per user
N/A	87	N/A
1999	224	12
2000	292	14
2001	228	7
2002	452	33
2003	529	8
2004	614	7
2005	1561	14
2006	4915	17
2007	24574	15
2008	52471	9
2009	49402	8
2010	52957	8
2011	42842	7
2012	40847	6
2013	34297	5
2014	30850	4
2015	24678	3
2016	25951	2
2017	18600	1
2018	15533	1
2019	5525	1
2020	5108	1
2021	3866	1
Total	436403	

parameters. Table 4.8 lists the configuration parameters that influence the execution performance the most.

The following list further details each parameter's role in the configuration.

- **HTTP connection pool size:** Size of the HTTP connection pool. When the crawler needs to connect to a webpage, it looks for an HTTP connector not currently in use from within the pool, and creates a new HTTP connector, adding it to the pool. if none available. No further new connections are created if the pool size is exhausted. The values chosen for this configuration were obtained after empirical observation of the crawler performance with different parameters. We verified that, in the message extraction phase, values much higher than 200 would make us reach a state of connection throttling after some time. During user extraction, given the lower amount of total connections being made to the website, we verified that a higher number of simultaneous connections could be attempted.

- **Thread pool size:** Max size for the thread pool. When dispatching work to a new thread, the system verifies first if there is an available thread in the thread pool to be reused. It creates a new one, otherwise, if the pool is not full yet. If the pool is full and a new connection is

Table 4.6: Metrics from complete message dataset extraction, as of November 2021

Total execution time	15 hours and 30 minutes
Total messages extracted	2304207
Average messages extracted per second	41
Average messages extracted per minute	2477

needed, the crawler will just wait until a connection becomes available. We verified that, for message extraction, a pool size of 8 provided us with the best performance, without impacting the thread resource management much. For the user profile extraction, we were able to increase the number of threads used simultaneously for the data crawler, without noticeable performance decreases.

- **Starting message/user ID:** A message or user ID that is used as a starting point, if the user wants to start the crawler from a specific point, instead of starting all over each time. Not used when extracting the complete dataset.
- **Forbidden communities:** Forum communities (or sub-forums) for the crawler to avoid processing. Used mostly to filter out non-English speaking forums. Used only when message crawling.
- **User batch size:** How many user IDs are extracted from the messages, at once, for parallel processing. Used only when user profile crawling.

When it comes to time performance of the crawler, we believe that the most important parameters are the HTTP connection pool size and the thread pool size. The connection pool allows us to reuse existing HTTP connections and ports to connect to the website, in order to extract messages. Without a reuse pool for connections, given the speed at which the data crawler attempts to connect to the website to read messages, we verified that we ran out of available network ports from the ones supplied by the operating system. This happens because, after closing a connection, there is a delay in clearing the used port and making it available for a new connection. With a connection pool, we do not need to constantly use different ports for each new connection, as we can just reuse a connection from the pool, if available. In addition, controlling the maximum number of connections available in the connection pool also allows us to manage the maximum number of simultaneous connections to the website being crawled, which enables us to try to avoid connection throttling from the server, something that some times happens when too many connection attempts are detected.

Applying a multi-threaded approach with the data crawler allows us to process multiple messages and multiple user profiles at once. However, there is a limit, dependent on the system hardware, after which new threads will just have to compete for resources against others, ending up clogging resources and stalling the execution, negating the benefits of the parallelism. Typically, the recommended maximum amount for parallel running threads on a given application is twice the number of CPU cores. On the other hand, as with the HTTP connections, the system does not

Table 4.7: Metrics from complete user profile dataset extraction, as of January 2022

Total execution time	10 hours and 50 minutes
Total user profiles extracted	436403
Average profiles extracted per second	11
Average profiles extracted per minute	674

instantly clear the resources allocated to a running thread. By applying a thread pool, we allow for thread reuse and, therefore, a better management of system resources. In addition, this parameter also allows us to better control the maximum number of threads being executed at any given time, in order to optimize task parallelism performance against the hardware limit mentioned before.

To provide the reader with further context for the data crawler execution metrics, Table 4.9 lists the main characteristics of the system the data crawler was executed on, to extract the message and user profile data.

Issues and considerations on running the data crawler In complement of the information in Table 3.1, which identifies expected problems stemming from data crawling, from a more theoretical point of view, the following list describes some issues found during the actual execution of the developed data crawler, and some notes on how to mitigate problems, in no particular order:

- **Slowness:** While processing information for storage is relatively fast, connecting to the remote webserver to transfer the content is usually a slow process and highly dependent on the resources of the host. The first versions of the data crawler were predicting over 3 days to extract all the content from the website. This issue was mitigated by applying multi-threading to the tool, as previously mentioned. Other solutions could be multi-processing or even distributed computing to maximize speed. Alternatively, offloading the work to cloud resources could be a good alternative, especially for a passive, "always running" version of the crawler.
- **Data updating:** Online health communities, like other internet discussion groups, get many messages every day. Therefore it would be useful to have an always running version of the data crawler that is able to update the dataset over time, without having to restart the full

Table 4.8: Possible data crawler configurations, and used values

Parameter	**Description**	**Value (messages)**	**Value (users)**
HTTP connection pool size	Size of the HTTP connection pool.	200	300
Thread pool size	Max size for the thread pool.	8	16
Starting message/user ID	Message or user ID used as a starting point.	N/A	N/A
Forbidden communities	Communities avoided by the crawler.	1855, 259, 1076	N/A
User batch size	User IDs considered for parallel processing.	N/A	100

Table 4.9: Main characteristics of the system the data crawler was executed on

CPU model	AMD Ryzen 9 5900HS
CPU cores	8
CPU base clock speed	3.0 Ghz
RAM	32 GB
Operative System	Windows 11 Pro
Network speed	Approx. 400 Mbps

crawling process. A possible solution for this would be to keep track of the time of last update of the last message extracted by the crawler. Then, if available on the crawled website, filter for content produced or updated *after* this time of last update. This functionality is not implemented at the time of writing, mainly due to the unavailability of such filters in the studied online health community.

- **Storing the data:** The best way to store the extracted content must be studied. At the time of writing, the crawler saves the content in JSON format, for ease of readability and sharing, but also in a MySQL database, to facilitate data querying and integration with other tools, such as the manual message classifier we developed during this dissertation.

- **Anti-crawling measures in the target website:** Some websites have measures to avoid automated crawling by software tools. The ones detected in MedHelp were: **connection throttling**, and **disabling**, after a high number of connection attempts were made in a short period of time; browser **user-agent analysis**, blocking content when the user-agent is not from a credible browser; some content viewable only after **authentication**. As solutions, we implemented the reusable connection pool mentioned beforehand, to attempt to counter throttling, and changed the data crawler user-agent to emulate a typical browser's identification when making connection requests. Lastly, we created an account for the OHC under analysis, and used its authentication token in the header of connections made by the data crawler, in order to crawl through connection-only content.

- **Data anonymization:** To protect user privacy, it is important to hide personally and uniquely identifiable content from the data extracted from the crawler. We approached this by generating a unique ID to be associated with each OHC user profile, as a substitute to the ID extracted from the website. We also change usernames in order to avoid future users of the data crawling tool to be able to find the exact profile of an extracted user back in the website. Despite this, it would be important to devise an algorithm that analyses the content of each message and profile and detects possible personally identifiable data and removes or obfuscates.

- **Target agnosticism:** It would be important for the data crawler tool to be easily executed against different websites, in the future. At the moment, the tool is not ready for that usage, despite being implemented with modularity and adaptability in mind. A possible solution

would be allowing the definition of the target structure in a configuration schema of some sorts. The tool would then use this schema to identify where and how to fetch content. It is worth considering, though, that some websites have very particular structures that would be hard, if not impossible, to model in a schema of this sort.

4.3 Analysis of the Manual Message Classifier

A prototype for a manual message classifier was developed during this dissertation. This prototype was implemented as a web application, hosted on a remote server with access to a database of messages for volunteers to classify. We will start this section by detailing some of the main characteristics of our solution.

Welcome page: When accessing the page for the first time, the user of the manual classifier is presented with a welcome page, explaining the purpose of the website, providing usage instructions, guaranteeing the anonymity of the answers given and sharing a contact e-mail address for any questions that might come up. Figure 4.2 shows a cutout of the content of this welcome page.

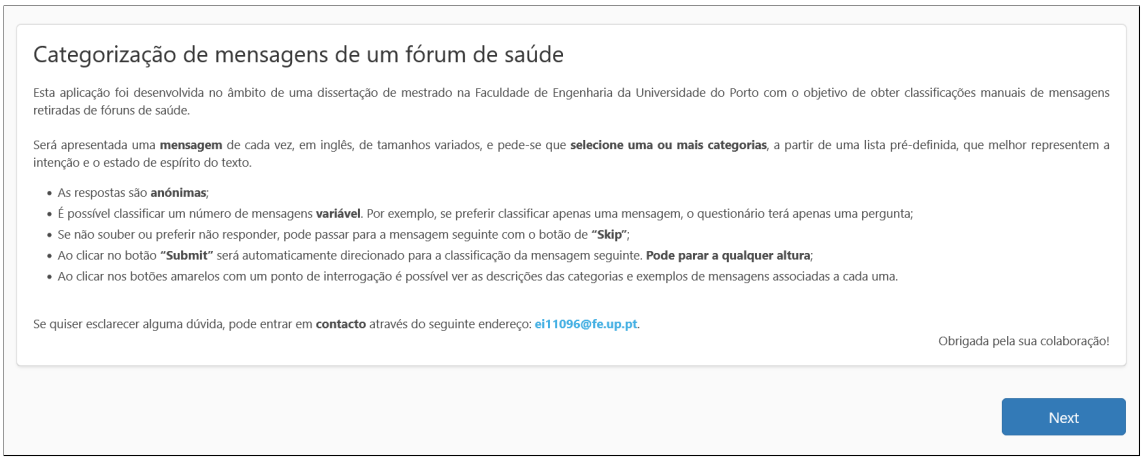

Figure 4.2: Manual Message Classifier - Welcome page content

Message classification page: After the welcome page, the user can start classifying messages. The classification page allows the user to read the content of the message they need to classify, with a sidebar on the right with each possible classification for the message. The user can select one or more categories from the sidebar, then press the "Submit & Continue" button to submit the classification and proceed to the next message. Alternatively, the user can skip the current message and proceed to classify a different one. Figure 4.3 exemplifies this message classification page. The reader can refer to Appendix B to see additional screen captures of the message classification page, as well as other images from the manual classification tool.

The message classification page also shows the user a current count of how many messages they have classified thus far, as shown in Figure 4.4. This is a simple attempt of *gamification*, to

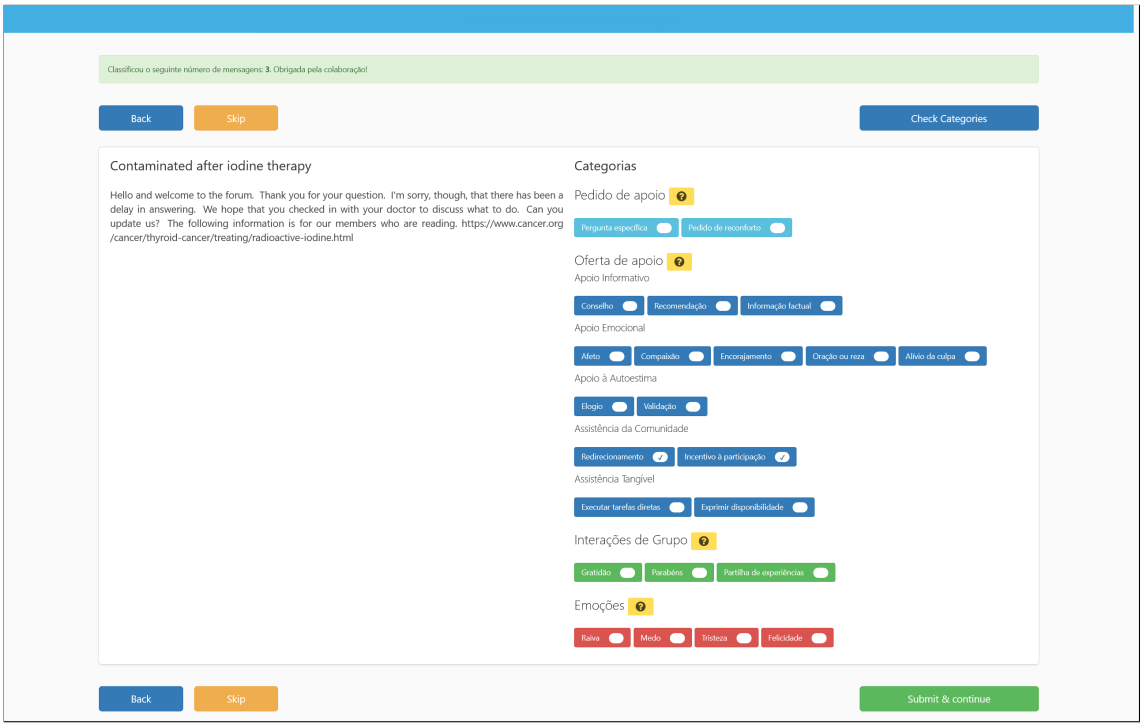

Figure 4.3: Manual Message Classifier - Classification page

keep the user motivated to continue the classification. A future development would be to present all the users a current ranking of top volunteers, in order to provide some friendly competition.

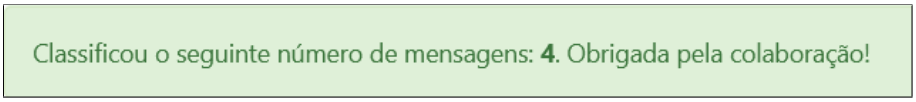

Figure 4.4: Manual Message Classifier - Banner with number of classifications done by current user

Category details page and tooltips: Within the classification page, the different classification categories are grouped by topic and color coded. The user can quickly refer to the meaning of each category by clicking the question mark next to each category topic name, which opens a modal window, as shown in Figure 4.5.

Alternatively, the user can click the "Check Categories" button to be redirected to a standalone page with descriptions and examples for every single classification category. This page can be seen in Figure B.5 of Appendix B.

4.3.1 Usage and performance metrics

One of our objectives for this dissertation was to have a group of volunteers test out the manual classifier prototype to categorize some of the messages in our extracted dataset. Due to time constraints, this group was small, composed of 3 people, from family and friends. Future work

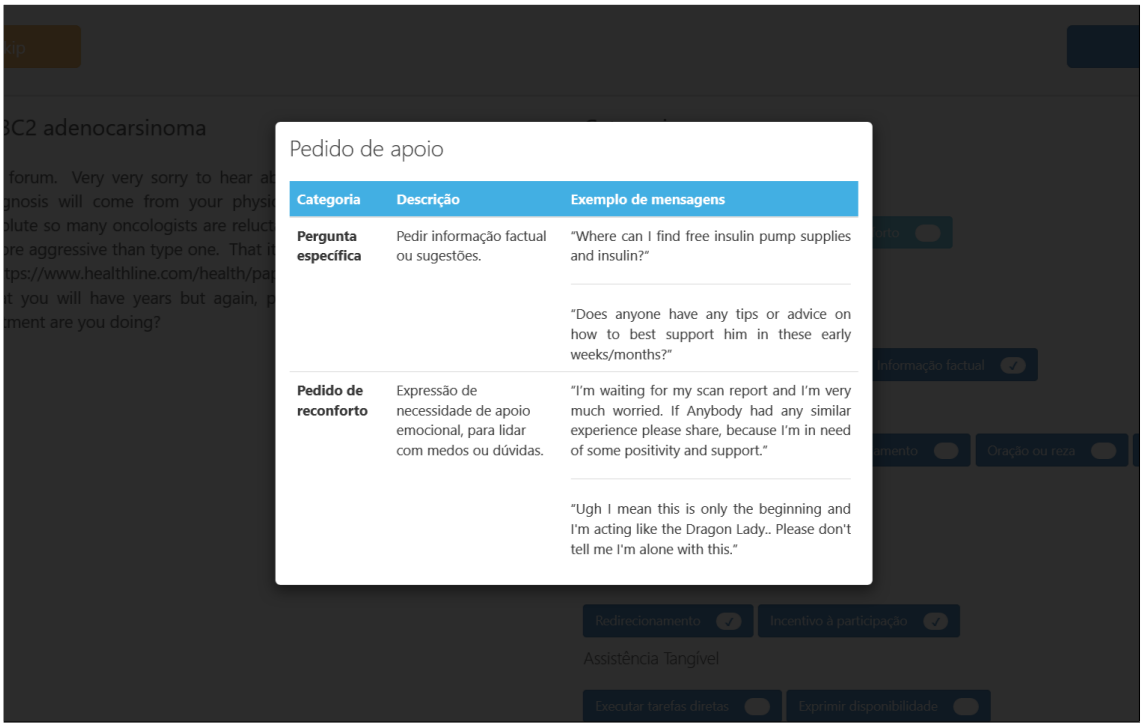

Figure 4.5: Manual Message Classifier - Category details tooltip

would include a larger number of independent volunteers. The group was given a small set of 50 messages to classify in a week. It took around 3 days for all the messages to be classified, as the participants were performing the classifications at their own pace. From a visual inspection of the number of messages classified, in a row, in each session, we could verify that each message took from 30 to 60 seconds to be classified, so we will assume a rough estimate of 45 seconds per message for our predictions. As expected, messages containing less common categories took a bit longer to classify, probably because the volunteer had to read the reference page or the category description tooltip to understand what each meant. Table 4.10 summarizes some results of this small test.

Table 4.10: Manual message classifier test results and predictions

Number of volunteers	3
Total messages classified	50
Average time per message	45 seconds
Predicted time to classify 100 messages	1 hour and 15 minutes
Predicted time to classify 1000 messages	12 hours and 30 minutes
Predicted time to classify entire dataset	Over 3 years
Predicted time to classify entire dataset (assuming it is divided by each volunteer)	1 year, 1 month and 5 days

It is possible to verify that with just a team of 5 volunteers it would take a prohibitively long time to properly classify every message in the dataset extracted with the data crawler, especially if

we consider a "vote" by 3 different volunteers on the same category, for each message, in order to accept a given classification as true (this aspect is discussed in the following subsection). However, to achieve a minimum dataset of 1 000 messages, which would allow us to start exploring more complex classification algorithms, the total predicted time would be more reasonable, of around 12 hours and 30 minutes total, assuming that the volunteers would agree on the classification of each message, to give us the minimum confidence vote of 3. Larger teams of volunteers would allow us to split the dataset among groups of volunteers and further speed up obtaining the 1 000 message mark. It is also possible that the classification time per message would start decreasing as the volunteers got more comfortable with the web interface and more familiar with the meaning of each possible classification category.

4.3.2 Issues and considerations

In this section we will talk about some issues we found while implementing and using our manual classification prototype, as well as some noteworthy considerations that can be of interest to the reader.

- **Classification voting approach:** One of the main issues with resorting to volunteers, as opposed to experts, for manual message classification is that we cannot be sure of how reliable their classifications are. One way we proposed to mitigate this issue was to implement a way to count how many classifications of each label were made for each message. Then, we could implement a variation of a "voting" system, and consider as correct only the labels with as many or more votes than a given threshold. Given the low amount of volunteers we had so far, we considered a number of 3 votes per label to consider it a valid classification for a given message.

- **Gamification:** As described before and shown in Figure B.7, we decided to implement a counter of total classified messages for each user, and display it at the top of the website. Our intention with this is to apply a basic element of *gamification* to the classification tool, and hopefully motivate the volunteers to keep interacting with it. The way we implemented, for now, is still in its very first version, but by talking with the volunteers we verified that the banner has some utility already. Our classifier users often used that counter to try and break some personal records, such as, for example, attempting to classify 10 messages in under 5 minutes.

- **Translation:** Currently, we do not have a full translation of the website to English. Translating to this language, specifically the explanations of each label, would allow us to seek a wider audience of volunteers through online means.

- **Random message to classify:** In order to balance the messages that get selected for classification by a user, we try to do a complete random choice of a message from the database. However, since we already have over two million messages up for classification, the random

operation takes a long time to perform when applied to this full dataset. A possible improvement on this, which is currently at use in the website, is to pick a random, smaller, sample of messages from the database beforehand (e.g. 1 000 messages), without any filtering, and have users classify those messages before moving on to the next sample. Besides speeding up the selection of a message for classification, this also facilitates achieving the necessary votes to accept a label for a message, as user classifications will not be as dispersed as with a larger dataset being picked from at random.

- **Other considerations:** According to the feedback given by the volunteers that used it, the manual classification tool was simple to use, the label explanations were easily consulted and sufficiently clear and it was overall pleasant enough to use to motivate multiple classifications within a session. Users shared some difficulties with classifying messages where it was not clear if the message was posted by the original thread author or by someone replying to the thread, information that we also believe should be added as metadata to the dataset, in the future.

4.4 Automatic Message Classification - results and analysis

In this section we will show and discuss results of the automatic message classifiers we implemented and evaluated for this dissertation. At the time of writing, three approaches were studied, based on *Vector Cluster Similarity*, *Dialog Act Classification* and *Emotion Analysis*.

4.4.1 Vector Cluster Similarity

In this subsection we describe the results of testing done to the Vector Cluster Similarity approach. For time constraint reasons that prevented us from, at the time of writing, finding adequate configuration parameters, the Related Word List alternative to this approach, described in 3.5.5.4, was not fully tested and, therefore, not added to the document.

4.4.1.1 First test - Preliminary dataset

The first test used only messages from the preliminary dataset to train the algorithm (creating the cluster vectors) and validating it. Given the lack of parameter tuning for this classifier, we skipped the cross validation step.

For testing purposes, we picked out a sample of roughly 10% of the number of messages classified with each possible label, making sure that at least 1 message per label was picked out. It is worth reminding the reader that this dataset of messages had a few categories without labeled samples, which impacted the ability of the classifier to successfuly classify these labels. This approach left us with a sample of 29 messages composing our testing set.

After testing, we verified a calculated accuracy of approximately **55%** and an Area Under Curve (AUC) of **53%**. Table 4.11 further details other testing metrics, such as precision, recall and f-score, for each possible label.

Table 4.11: Vector cluster similarity using the preliminary dataset - Testing results

Label	**Precision**	**Recall**	**F1-score**	**Support**
Access	1.00	1.00	1.00	1
Advice	0.50	0.25	0.33	4
Affection	1.00	1.00	1.00	1
Anger	1.00	1.00	1.00	1
Encouragement	0.50	1.00	0.67	1
Express willingness	1.00	1.00	1.00	1
Fear	0.20	1.00	0.33	1
Gratitude	0.00	0.00	0.00	1
Happiness	1.00	1.00	1.00	1
Prayer	1.00	1.00	1.00	1
Presence	0.33	1.00	0.50	1
Reassurance	0.00	0.00	0.00	1
Recommendation	1.00	1.00	1.00	1
Sadness	0.33	1.00	0.50	1
Sharing personal experiences	1.00	0.50	0.67	4
Specific question	0.50	0.17	0.25	6
Teaching	0.50	0.50	0.50	2
Accuracy			0.55	29
Area Under Curve (AUC)			0.53	

It is possible to verify that the labels for "sympathy", "relief of blame", "compliment", "validation", "perform direct task" and "congratulations" are missing from the aforementioned table. This happens because the preliminary dataset did not have examples of messages with these labels and, therefore, we were not able to train the classifier to consider them using this as training data.

The accuracy and AUC values so close to 50% show that, at the moment, this classifier is very close to behaving similarly to a random predictor.

If, given the small size of the training set, we opt not to remove the testing sample from the training set, and use those messages also for the training of the model, we obtain slightly better results. We are able to achieve an accuracy of **66%** with an AUC of **55%**. Table 4.12 details these results.

4.4.1.2 Second test - Extended dataset

For this test, in order to attempt to close some of the gaps identified in the previous test due to the size of the training set of data, we extended the preliminary dataset with messages that were manually classified using the manual classification web tool, developed as a prototype for this project. This dataset features 189 additional messages and now contains messages with labels that were previously not represented in the set. Table 4.4 details how many more messages per label were added, while Figure 4.1 shows the distribution of messages per each category under the two datasets.

Table 4.12: Vector cluster similarity using the preliminary dataset - Testing results without removing testing sample

Label	**Precision**	**Recall**	**F1-score**	**Support**
Access	1.00	1.00	1.00	1
Advice	0.80	1.00	0.89	4
Affection	1.00	1.00	1.00	1
Anger	1.00	1.00	1.00	1
Encouragement	0.50	1.00	0.67	1
Express willingness	0.50	1.00	0.67	1
Fear	1.00	1.00	1.00	1
Gratitude	0.00	0.00	0.00	1
Happiness	1.00	1.00	1.00	1
Prayer	1.00	1.00	1.00	1
Presence	0.33	1.00	0.50	1
Reassurance	0.50	1.00	0.67	1
Recommendation	1.00	1.00	1.00	1
Sadness	0.00	0.00	0.00	1
Sharing personal experiences	0.60	0.75	0.67	4
Specific question	1.00	0.17	0.29	6
Teaching	0.00	0.00	0.00	2
Accuracy			0.66	29
Area Under Curve (AUC)			0.55	

Testing was done under similar conditions as in the first test, with a testing sample of approximately 10%, making sure that at least 1 message per label was chosen for the sample. This resulted in a test sample size of 47 messages. Similarly to the second attempt of the previous test, we opted not to remove the test samples from the training set.

After testing, we obtained a calculated accuracy of **38%** and an AUC of **51%**. Table 4.13 details other testing metrics, the same as with the first test.

We can now attest that the previously missing labels from the first test are present in this table, as we now have these categories represented in the training set, thanks to the dataset extension performed, with manually categorized data. It seems that the extra messages in the testing set, as well as the additional labels added to the possible classifications impacted the general performance negatively, verifying a significant decrease in accuracy and AUC. This brings to light the low performance of this classifier as it was implemented, at the time of writing. Further analysis on configuration parameters is necessary, but we believe that the small size of the training set, even after extension, as well as the uneven distribution of messages per class in both the training set and the testing set are possible and relevant reasons for this lack of performance. It would be necessary to normalize this dataset prior to training and testing to validate if this would lead to better results.

Table 4.13: Vector cluster similarity on the extended dataset - Testing results

Label	Precision	Recall	F1-score	Support
Access	0.50	1.00	0.67	1
Advice	0.25	0.20	0.22	5
Affection	1.00	1.00	1.00	1
Anger	1.00	1.00	1.00	1
Compliment	0.00	0.00	0.00	1
Congratulations	0.50	1.00	0.67	1
Encouragement	0.33	1.00	0.50	1
Express willingness	0.00	0.00	0.00	1
Fear	0.00	0.00	0.00	2
Gratitude	0.00	0.00	0.00	2
Happiness	1.00	1.00	1.00	1
Perform direct task	1.00	1.00	1.00	1
Prayer	1.00	1.00	1.00	1
Presence	1.00	1.00	1.00	1
Reassurance	0.00	0.00	0.00	3
Recommendation	1.00	1.00	1.00	1
Relief of blame	0.00	0.00	0.00	1
Sadness	0.14	1.00	0.25	1
Sharing personal experiences	0.40	0.33	0.36	6
Specific question	0.50	0.11	0.18	9
Sympathy	0.00	0.00	0.00	1
Teaching	0.33	0.75	0.46	4
Validation	0.00	0.00	0.00	1
Accuracy			0.38	47
Area Under Curve (AUC)			0.51	

4.4.2 Dialog Act Classification

This subsection summarizes the results of the Dialog Act Classification approach, as described in 3.5.6. The previously described extended dataset was used for testing this approach. Unlike in the Vector Cluster Similarity approach, we do not need to train the algorithm first, so only the manually extended set of messages was used to validate the results of this step. Table 4.14 shows some metrics for analysis. However, it does not list every single possible label because, as explained before and shown in Table 3.4, we were only able to apply this method to classify a subset of the labels.

As the reader can see, while the accuracy values are generally high, precision and recall are often not as much. This can be explained mainly by the small size of the training dataset being used which, additionally, is unbalanced in the number of messages per classification label (there are labels with fewer than 2 samples, or even none). Given that, for some labels, only 2 messages out of the complete dataset are classified with it. Since, in one of those situations, most of the test set does not possess that particular label, even a classifier that fails to classify the label 100% of

Table 4.14: Testing metrics for Dialog Act Classification based method

Label	Accuracy	Precision	Recall
Advice	0.76	0.44	0.56
Gratitude	0.94	0	0
Reassurance	0.76	0.14	0.36
Specific question	0.89	0.85	0.82
Teaching	0.21	0.10	0.85
Validation	0.92	0	0

times will have good accuracy, because it will "accurately" classify that label as missing.

4.4.3 Emotion Analysis

Similarly to the previous one, this subsection summarizes the results of the Emotion Analysis phase of the automatic text classification step of the pipeline. As before, with the dialog act classification method, we did not train classification model from scratch, so only the manually extended set of messages was used to validate the results of this step. Table 4.15 displays the results of the classifier on our testing set.

Table 4.15: Testing metrics for Emotions classification method

Label	Accuracy	Precision	Recall
Anger	0.91	0	0
Fear	0.49	0.05	0.42
Happiness	0.73	0.02	0.11
Sadness	0.64	0.04	0.5

We can see here that the problem identified in the dialog act classification method also seems to be present here. By analysis of the testing set, we verified that most of the messages do not have any emotion label applied. Therefore, a classifier that simply does not apply emotion labels to most messages will still result in having an apparently high accuracy.

4.5 Summary

In this chapter we focused on describing and analysing the most notable results from the work developed for this dissertation. This work consisted in prototyping the most important steps of the pipeline proposed in Chapter 3 and studying the data we committed to work with.

An overview of the preliminary dataset allowed us to validate most of the categories present in Bárbara Silva's proposed taxonomy and understand the sort of classification task that step of the pipeline would have to work with. More generally, studying over 2 million extracted messages from the *MedHelp* community taught us the general structure of an online health community, how posting habits evolved over the years and what sort of content topics tend to be more popular in these kinds of online social spaces. In addition, studying the extracted user profile data also

provided us with insights into the community users themselves, clearly showing us that the social aspect of these communities is highly sought after, even if seeking and providing health-related information is presented as being the main objective of participation.

Regarding the prototype for an OHC data crawler, we saw that this tool was able to extract large amounts of apparently useful data from a single community, though it takes over 15 hours to do so, regarding message content, in a consumer-grade computer. We were able to identify important aspects to take into account when developing these kinds of tools, both in a technical sense (e.g. applying multi-threading techniques) and a more procedural one, with data anonymization concerns and responsible website usage ones.

We also described our prototype for a manual message classifier and shared the findings that came from having a small group of volunteers using it. We identified the need for further volunteers in order to tackle such a big unlabeled set of data. However, we were able to ascertain that keeping the tool simple to use, while applying concepts of gamification, appears to be important in order to have a classification experience that is positive and motivating of longer usage sessions.

Lastly, we talked about the automatic message classification approaches applied to this step of the prototype. We analysed approaches using Vector Cluster Similarity, Dialog Act Classification and Emotion Analysis, including the input data used and testing parameters. We identified the need for a larger amount of labeled data, as all the tested methods of classification had accuracy problems, stemming most likely from the small and unbalanced representation of each category in the available training dataset. Despite this issue, we were able to implement automatic classification tools that still had better results than a completely random classification approach, which shows potential for improvement with further labeled data and parameter tuning.

Chapter 5

Conclusions

It is undeniable the growing importance of online sources of information to tackle a myriad of daily-life problems, from casual to serious, in the recent years. Long gone are the times when we had to submit to our ignorance or blindly see as true people's opinions on subjects we were less knowledgeable about, until we could make our way to a set of encyclopedias, a library or, ideally, an expert. Nowadays, we have a world of knowledge in our pockets, as phones with internet access became ubiquitous. But access to factual knowledge was not the only thing brought to us by the newly connected world. Now, we can also benefit from the experience of millions of people, the opinions of countless experts and the support of many empathetic human beings, through participation in the many online social spaces that exist. Both these sources of information and support are especially important in times where medical care is not always easy to access, for time or financial reasons. However, the Internet is a big place and sifting through all the content produced daily is a hard task, even within the specific field of medical information, the focus of the present work.

Given the above context, this dissertation focuses on the issue of studying those online social spaces for the discussion of medical issues and healthcare. We believe that there is a massive amount of important content regarding those topics in online communities for health discussion, which, on the one hand, should be easily accessible by interested users and, on the other hand, should be further studied by researchers of these fields. With that in mind, we identified the problem of studying these communities, extracting useful information from them and categorizing it for ease of access and further study. As a possible solution, we proposed in this dissertation an integrated pipeline of methods for extracting information and user-created content from online health communities, studying that information and automatically classifying it.

We began by studying and reviewing approaches for the general task of text processing and automatic classification, attempting to describe the several steps that compose these tasks, and then moving on to the specific domain of online health communities and their study and content classification. While we believe that this literature review should have been expanded to study

also the topics of online community and social media analysis, automated data extraction and manual dataset labeling, given time constraints we focused on the research topic of automated text classification, as we consider it to be the most technically complex challenge for the practical implementation of our proposed pipeline. We think that this review supplied us with sufficient knowledge and tools to tackle the problems of text processing and classification that we found, but also serves as helpful context of the overall problem for the reader of this work.

Following that, we identified the main research problem we would tackle with this dissertation - can we build a general and reproducible approach for online health community characterization and analysis? As an answer, we proposed the development of an integrated pipeline, the stages of which would allow us to choose a community, automatically extract message content and user profiles from it, study that data, manually classify the data, if needed and, lastly, automatically classify the data into a domain specific classification taxonomy. We proceeded to describe each of these steps of the pipeline, prototype them and study their performance, using the *MedHelp* online health community as target for this work.

We verified that it was possible to develop a comprehensive data crawling and extraction tool for the *Medhelp* community. We were successful with extracting message content and helpful metadata that we believe would be important to the development of a health-related text dataset for training and testing automatic text processing and classification models, as well as study these specific social spaces. A total of 2 304 207 messages were successfully extracted, dating from 1999 to 2021. In addition, we also succeeded in extracting user profile information of the authors of these messages, currently having 436 403 user profiles in our database. With the caveat that this user data must be anonymized, we verified that this would also be a very useful set of data for the overall analysis of a health-related social space, through automated means or otherwise. It was possible to see that the general structure of these forum-like communities (e.g. forum - sub-forums - threads - replies) allowed for an easier implementation of the crawling behaviour of the tool and for the support of a future agnosticism of the specific implementation of the online health community being crawled.

As an example of the importance of metadata for the analysis of these communities, we were able to extract some interesting insights from what we retrieved with the data crawler. We verified that the activity in the *MedHelp* community changed drastically over the years, we assume due to the fall in usage of forum-like online communities as more general social media platforms rose and took preference. Even with the appearance of the *COVID-19* global pandemic, in late 2019, we did not verify a significant increase in activity. A possible conclusion of this is that the activity of these forum-like communities is supported by an established userbase of individuals that have participated in the webforum for a long time now, and nurtured personal connections with other users in the same situation. This seems to be further validated by the fact that most of the more recently active users in the community have, generally, older registration dates and a larger number of friends, as verified by studying the extracted user profile data.

Recognizing that the amount of labeled data we currently had, stemming from the previous work of classification taxonomy that preceded this dissertation, was possibly too small to properly

train and evaluate automatic classification models, we decided to integrate a step of manual message classification into the pipeline. This manual message classifier was developed as a web-based tool in order to facilitate sharing it with volunteers, requiring no installation and making it accessible on mobile devices. Due to time constraints, a full release to a large number of volunteers was not possible, and instead we relied on the participation of three people, chosen from friends and family of the dissertation author, who kindly accepted to test the tool and provided us with very useful feedback. We concluded that it could take, on average, up to a minute to classify each message, the actual time dependant on the size of the text content of the message and the possible classification with less common categories that the labeling volunteer was not as comfortable with. To mitigate the issue of possibly wrong classifications by the volunteers, we considered a "voting" approach where each individual label classification was only considered as valid after several volunteers assigned the same label. The threshold of equal classifications to be considered valid is configurable, but for testing purposes, and given the size of our group of testers, we considered labels as valid after 3 repeated classifications. We posited that labeling a dataset of 1000 messages, still small for most deep learning classification approaches other than for transfer learning and model fine-tuning, but workable for more classical classification algorithms, would require an average of 12 hours and 30 minutes of work by volunteers. To make this classification work more appealing, we proposed the addition of *gamification* elements to the classification tool. A first implementation of such an approach was made for the prototype tool (a simple count of successive classifications made by each volunteer, and displayed to them only) and already proved useful, according to tester feedback, as it pushed them to beat their own personal records of the number of successively classified messages in a single classification session. However, an expansion on these gamification elements was deemed necessary, as well as solving some implementation issues stemming from the large number of unlabeled messages in the database, something to be expanded upon future releases of the tool. Despite this, we believe to have been successful with the implementation of the manual message classifier, having proven to be a useful and easy to use tool to help expand these kinds of datasets.

The final step we tested, for the proposed integrated pipeline, concerned itself with automatic classification of the textual content of messages in online health communities. This classification problem was interpreted as a multi-label classification problem, as each message could be assigned to more than one category, from the taxonomy developed in previous works, and detailed several times along the present document. Given readily accessible tools and speed of implementation, as well as the size of the existing labeled dataset, we tested three different approaches for this classification: Vector Cluster Similarity, Dialog Act Classification and Emotion Analysis. To apply these automatic classification methods, similar steps of text pre-processing were considered, such as sentence and word segmentation and *tokenization*, *lemmatization*, word *stemming* and vectorial representation of the textual content, specifically making use of BERT embeddings. The first classification approach was able to perform classification on the whole taxonomy, though the overall accuracy was rather low (a best result of 66% as of the last testing round done) and the precision and recall of each label was quite heterogeneous, resulting in skewed F1-scores. We concluded

that it is likely that this issue comes from the lack of balance in distribution of messages per each of the labels, in an already small training set, given that the most common categories ("Specific question" or "Advice", for example) had some times over 20 times the amount of messages in the least common categories (e.g. "Validation" or "Perform direct task"). In fact, the original dataset we had to work with, which motivated the development of the manual message classifier, had zero samples for some of these least common categories, something we somewhat managed to mitigate with the help of manual classification. Besides the unbalanced training set, we also identified other issues with this classification approach. The most relevant of these is the problem that the resulting category clusters, generated from the BERT embeddings of each message in the training set, were unfortunately not very independent of each other, being easy to verify that the clusters intersected each other on several occasions. This context of non well-separated clusters makes it tougher for the similarity function to reliably attribute a sample to one specific cluster. The second classification approach based itself on the concept of Dialog Act Classification - classifying an utterance, by a speaker, by its specific intention within the dialog. Given time concerns, instead of attempting to train our own model or fine-tune an existing one, we opted for using a pre-trained classification algorithm that was able to classify messages with a set of generic dialog acts. We empirically devised a classification key where a set of specific dialog acts, as well as emotions, when present in a given message, would map to categories of our domain-specific taxonomy. This approach had the unfortunate consequence of many of our categories having to be skipped from classification, as we did not find an adequate mapping of generic dialog acts to those specific categories. This method showed a relatively high degree of accuracy, of over 70% for all the considered labels, other than "Teaching". However, despite the accuracy, the precision and recall numbers were generally low, something that we believe stems, once more, from the unbalanced characteristics of the training and testing sets. The testing set had low number of messages for some of the categories, meaning that, for example, true-negatives are most likely over-valued when calculating accuracy. Lastly, the Emotion Analysis approach concerned itself only with the *Emotions* category of our taxonomy - Anger, Fear, Sadness and Happiness. This method is meant to be used as a complement to the dialog act classification approach. We opted to use a pre-trained model for this phase, as well. Once again we verified high accuracy but with low precision and recall rates. Upon further analysis of the dataset, we verify that there is a significantly low number of messages classified with emotion categories, when compared with other categories. While we were successful in prototyping and integrating the automatic classification step into our proposed pipeline, we conclude that we achieved generally weak results and recognize the need for additional work in this step of the pipeline. The priority should be to extend the available training and testing sets with additional labelled data, in efforts to achieve a balanced set of data over the multiple categories.

In sum, despite the issues and setbacks identified throughout this dissertation, we believe we were successful in devising a data extraction, processing and analysis pipeline for studying online health communities and answering the research questions we initially proposed. Even with being an open-ended field of research, it was possible to suggest a general set of steps, rules and considerations to take into account, during each of the multiple phases that are usually present

when approaching an online health community to study its userbase and user-generated content, manually or automatically. While we focused on studying one particular community - *MedHelp* - we think the results obtained from our research and development are generalizable enough to be applied to other similar social spaces, specifically if they also present a structure close to an internet forum and have a userbase consisting mostly of non-experts trying to give support through their shared experiences and knowledge. We believe this work to be a helpful addition to online health community study and hope it proves a useful baseline for future work on the fields touched by each step of the proposed pipeline.

5.1 Main contributions

We can divide the main contributions of this work into three domains, as follows.

Technical contributions: The main technical contributions of this work stem from the developed prototypes for each of the steps of the proposed data pipeline, which should be made available to the general public shortly after conclusion of this project.

The data crawler is a configurable tool that should be executable on any system that will allow the user to extract messages and user profiles in the exact same way we did for this work, allowing them to achieve an unlabeled set of data, ready for use in whatever application necessary.

The manual message classifier can be adapted to any type of data with minimal changes, or used as is with the *MedHelp* content structure and our classification taxonomy.

Additionally, the automatic classification models and algorithms should also be made publicly available for users interested in applying them as-is, or attempting to improve their performance.

The code for these tools should be made open-source after the conclusion of this project, alongside documentation and pre-compiled binary versions, where applicable.

Scientific contributions: As a main scientific contribution, rather than focusing on the very specific issue of automatic text classification, we proposed a full pipeline that integrated the research focuses of content source choice and evaluation, data extraction and study and manual, as well as automatic, data classification. We believe this contributes to bridging a research gap where previous work mostly deals with each of these issues on a separate basis, while not considering how each of them may influence and inform the others.

As another, more specific, contribution, we identified the online health community text content classification problem as a possible subset of the broader Dialog Act Classification problem, which should open new avenues of research when seeing this automatic classification task in that specific context.

Furthermore, we gathered two datasets, one of user-created messages, and another of user profiles, in the context of medical issues and healthcare discussion, which should prove useful for research work and other projects in the field, given their size, characteristics and available metadata. We intend to publish these datasets after a more extensive curation is performed and

personally identifiable data is properly anonymized, provided permission by the *MedHelp* platform owners.

Application domain contributions: Regarding contributions to the wider domain of online health community study, we believe this work identified a set of steps, rules, common issues and considerations that should be taken into account when studying these sorts of communities, especially when that study relates to the user-created content and ways to extract and classify it. Additionally, this dissertation emphasized the need for extensive and balanced datasets when applying machine learning methodologies to this study and clarified some of the main issues to look out for with this kind of data.

We also provided a detailed analysis of the *MedHelp* online health community, one of the largest of its kind, bringing to light the characteristics of its content and its users, as well as the evolution over the years of the activity of its userbase. Given that this community is a reference for health-related social spaces, the insights shared in this document should prove transferable and useful to the general study of any of these spaces.

5.2 Future work

We now share a list of issues and developments we believe would be of great interest to be approached in the future, which would expand on the work done in this dissertation and bring added value to the research developed so far on this theme.

- **Develop an extra data visualization step in the pipeline:** Data visualization is an important field of study and it is particularly hard to properly display metrics and statistics when studying complex data. We believe a data visualization step would be a great addition to our proposed data pipeline, displaying to the user, automatically, any insights obtained from the extracted data and its classification, whether manual or automatic.
- **Integrate the whole pipeline into a single tool:** One of our objectives with this pipeline is that it is one day packaged into a single executable tool that integrates each of the execution steps, from data extraction, to statistical analysis, automatic classification and, in the future, result visualization. We exclude from here the manual classification web application, as it ideally would be deployed into a remote server.
- **Make the pipeline completely independent of the *MedHelp* community structure:** For the purpose of this dissertation, we based the whole of our work in the *MedHelp* community. As such, our statistical analysis steps, our data crawler implementation and automatic classifiers rely on the specific structure of that community and its content. While we tried to keep development as agnostic as possible, some currently *hard-coded* parameters would need to be abstracted into configuration steps of each tool. In our opinion this would allow the steps of the pipeline to be configured to work with any other similarly structured community as a data source.

- **Improve the data crawler:** Besides the general work of turning all the steps of the pipeline data source agnostic, including the crawler, there are also other improvements we think should be done. The overall performance of the crawler should be improved, including finding an approach that allows the crawler to continuously update the message database as new content comes up on the tracked online community. It would also be very useful to keep track of speech order in the metadata of each message, as many models in the field of dialog act study and classification make use of this information. For example, it would be important to keep track if any given message is standalone or a direct answer to any other particular message.

- **Expand and enhance the labelled dataset:** We made obvious that we need larger amounts of labelled data in order to create proper training and testing sets for the classification algorithms. It stands to reason that we should work on improving and expanding the labelled data we currently have, ideally by finding more volunteers to perform manual classification of the messages extracted from *MedHelp*.

- **Improve the automatic classification methods:** It is very important that we improve on the automatic classification step of the proposed pipeline. The achieved results, so far, were not entirely satisfying to us, and further research should be done so as to improve this aspect. Besides finding larger datasets for training and testing, as mentioned previously, it would be important to do some further sensitivity analysis on the parameters of the methods we used so far, to see if we obtain improved results. It would also be useful to study how changing, adding or disabling some text pre-processing steps impacts the overall classification performance. It would be very helpful to establish baseline classification methods by using more common and, generally, simpler approaches based on Support-vector Machines (SVM) and Naive Bayes classifiers, for example, in order to compare results with other, more novel, approaches. Lastly, once a larger set of training data is achieved, it would be interesting to try and train deep learning classification models for our tasks, or try and apply transfer learning to fine-tune an existing model.

References

[1] Soon Ae Chun and Bonnie MacKellar. Social health data integration using semantic web. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 392–397, New York, NY, USA, 2012. ACM.

[2] Internet Archive. Wayback machine. https://web.archive.org/. Last accessed September, 2022.

[3] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[4] Christopher E. Beaudoin and Chen-Chao Tao. Benefiting from social capital in online support groups: An empirical study of cancer patients. *CyberPsychology & Behavior*, 10(4):587–590, August 2007.

[5] R. Bekkerman, A. McCallum, and G. Huang. Automatic categorization of email into folders: Benchmark experiments on enron and sri corpora. *Center for Intelligent Information Retrieval, Technical Report IR*, 418, 2004.

[6] Alexander Beloborodov, Artem Kuznetsov, and Pavel Braslavski. *Characterizing Health-Related Community Question Answering*, pages 680–683. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[7] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, 2009.

[8] Prakhar Biyani, Cornelia Caragea, Prasenjit Mitra, and John Yen. Identifying emotional and informational support in online health communities. In *25th International Conference on Computational Linguistics: Technical Papers*, COLING 2014, pages 827—-836, 2014.

[9] Prakhar Biyani, Cornelia Caragea, Prasenjit Mitra, Chong Zhou, John Yen, Greta E. Greer, and Kenneth Portier. Co-training over domain-independent and domain-dependent features for sentiment analysis of an online cancer support community. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, pages 413–417, New York, NY, USA, 2013. ACM.

[10] William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[11] Pew Research Center. Health fact sheet. http://www.pewinternet.org/fact-sheets/health-fact-sheet/, 2013. Last accessed October, 2016.

[12] Aron Culotta. Detecting influenza outbreaks by analyzing twitter messages. *CoRR*, abs/1007.4748, 2010.

[13] Aron Culotta. Towards detecting influenza epidemics by analyzing twitter messages. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, pages 115–122, New York, NY, USA, 2010. ACM.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[15] Ronen Feldman and James Sanger. *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, New York, NY, USA, 2006.

[16] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, Mar 2002.

[17] Mette Terp Høybye, Christoffer Johansen, and Tine Tjørnhøj-Thomsen. Online interaction. effects of storytelling in an internet breast cancer support group. *Psycho-Oncology*, 14(3):211—-220, 2005.

[18] Muhammad Imran, Patrick Meier, Carlos Castillo, Andre Lesa, and Manuel Garcia Herranz. Enabling digital health by automatic classification of short messages. In *Proceedings of the 6th International Conference on Digital Health Conference*, DH '16, pages 61–65, New York, NY, USA, 2016. ACM.

[19] Tommi Jaakkola, Jason D. M. Rennie, and Jason D. M. Rennie. Improving multi-class text classification with naive bayes. Technical report, 2001.

[20] Thorsten Joachims. *Text categorization with Support Vector Machines: Learning with many relevant features*, pages 137–142. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[21] Jonathan Hedley. Jsoup: Java HTML parser, built for HTML editing, cleaning, scraping, and XSS safety. https://jsoup.org/. Last accessed September, 2022.

[22] Bryan Klimt and Yiming Yang. *The Enron Corpus: A New Dataset for Email Classification Research*, pages 217–226. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[23] Leonard Richardson. Beautiful Soup: We called him Tortoise because he taught us. https://www.crummy.com/software/BeautifulSoup/. Last accessed September, 2022.

[24] David D. Lewis. Feature selection and feature extraction for text categorization. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '91, pages 212–217, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics.

[25] David D. Lewis. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, pages 246–254, New York, NY, USA, 1995. ACM.

[26] Vitals Consumer Services LLC. Medhelp - medical information, forums and communities. https://www.medhelp.org. Last accessed September, 2022.

[27] Vitals Consumer Services LLC. Medhelp - medical information, forums and communities [archive 2015]. https://web.archive.org/web/20151230075151/http://www.medhelp.org/, December 2015. Last accessed September, 2022.

[28] Vitals Consumer Services LLC. Medhelp - medical information, forums and communities [archive 2016]. https://web.archive.org/web/20161229045955/medhelp.org, December 2016. Last accessed September, 2022.

[29] Diane Maloney-Krichmar and Jenny Preece. A multilevel analysis of sociability, usability, and community dynamics in an online health community. *ACM Trans. Comput.-Hum. Interact.*, 12(2):201–232, June 2005.

[30] Benjamin Mandel, Aron Culotta, John Boulahanis, Danielle Stark, Bonnie Lewis, and Jeremy Rodrigue. A demographic analysis of online sentiment during hurricane irene. In *Proceedings of the Second Workshop on Language in Social Media*, LSM '12, pages 27–36, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[31] Christopher D. Manning. Computational linguistics and deep learning. *Comput. Linguist.*, 41(4):701–707, December 2015.

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

[33] D.G. Maynard and K. Bontcheva. Challenges of evaluating sentiment analysis tools on social media. In *10th edition of the Language Resources and Evaluation Conference*. LREC, May 2016.

[34] Peter Náther. N-gram based text categorization. *Lomonosov Moscow State Univ*, 2005.

[35] Octopus Data Inc. Web Scraping Tool & Free Web Crawlers | Octoparse. https://www.octoparse.com/. Last accessed September, 2022.

[36] PatientsLikeMe. Milestones|patientslikeme. http://news.patientslikeme.com/milestones, 2017. Last accessed February, 2017.

[37] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[38] Nambisan Priya. Information seeking and social support in online health communities: impact on patients' perceived empathy. *J Am Med Inform Assoc*, abs/1007.4748, May 2011.

[39] Quantcast. Traffic and demographic statistics by quantcast. https://www.quantcast.co, 2017. Last accessed February, 2017.

[40] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

[41] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, March 2002.

[42] Bárbara Silva. Análise de conteúdo em fóruns de saúde na Web: proposta de um esquema de classificação. Master's thesis, Faculdade de Engenharia da Universidade do Porto, Portugal, 2016.

[43] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 841–842, New York, NY, USA, 2010. ACM.

[44] Webmaster. Website stats. http://http://stats.webmaster.net/, 2017. Last accessed February, 2017.

[45] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[46] Zhou Xiaofei, Guo Li, Tan Jianlong, and Jiang Wenhan. Theme word subspace method for text document categorization. In *Proceedings of the Data Mining and Intelligent Knowledge Management Workshop*, DM-IKM '12, pages 6:1–6:7, New York, NY, USA, 2012. ACM.

[47] Haodong Yang and Christopher C. Yang. Using health-consumer-contributed data to detect adverse drug reactions by association mining with temporal analysis. *ACM Trans. Intell. Syst. Technol.*, 6(4):55:1–55:27, July 2015.

[48] Y. Yang and T. Joachims. Text categorization. 3(5):4242, 2008.

[49] Shaodian Zhang, Erin Bantum, Jason Owen, and Noémie Elhadad. Does sustained participation in an online health community affect sentiment? In *AMIA Annual Symposium Proceedings*. American Medical Informatics Association, 2014.

[50] Thomas Zhang, Jason H. D. Cho, and Chengxiang Zhai. Understanding user intents in online health forums. In *Proceedings of the 5th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '14, pages 220–229, New York, NY, USA, 2014. ACM.

[51] Zyte Limited. World's Leading Web Scraping Service | Zyte. https://www.zyte.com/. Last accessed September, 2022.

Appendix A

Classification Schema

Table A.1: The classification schema as described in Bárbara Silva's work [42]

Categories		
Seeking Support		Specific question
		Reassurance
Offering Support	*Information Support*	Advice
		Recommendation
		Teaching
	Emotional Support	Affection
		Sympathy
		Encouragement
		Prayer
		Relief of blame
	Esteem Support	Compliment
		Validation
	Network Support	Access
		Presence
	Tangible Assistance	Perform direct task
		Express willingness
Group Interaction		Gratitude
		Congratulations
		Sharing personal experiences
Emotions	*Negative*	Anger
		Fear
		Sadness
	Positive	Happiness

Appendix B

Manual Message Classifier - Screenshots

This appendix contains screen captures of the main pages and elements of the manual message classification tool, as implemented at the time of writing of this document. These screens focus on the initial welcome page, three example of the classifications page, and a demonstration of how we provide the user with an explanation of each classification category.

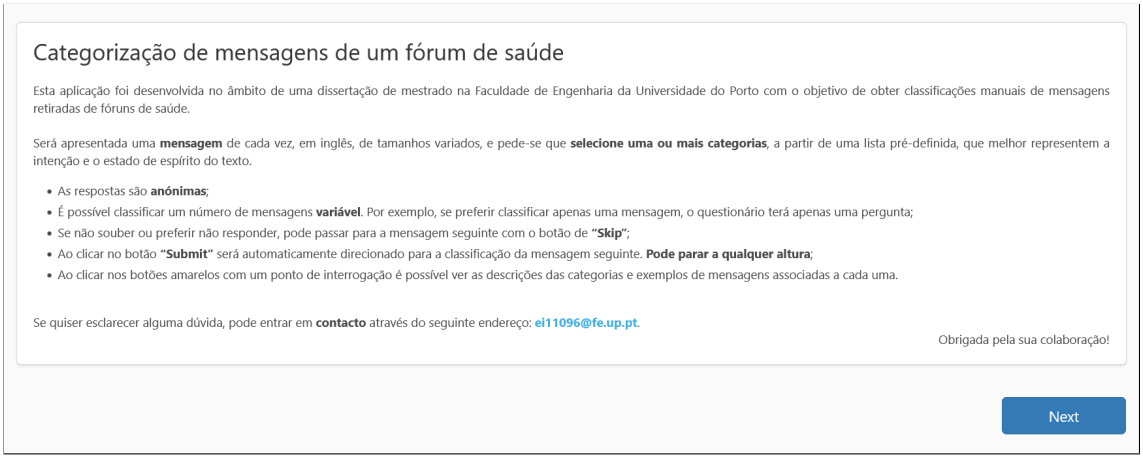

Figure B.1: Manual Message Classifier - Welcome page

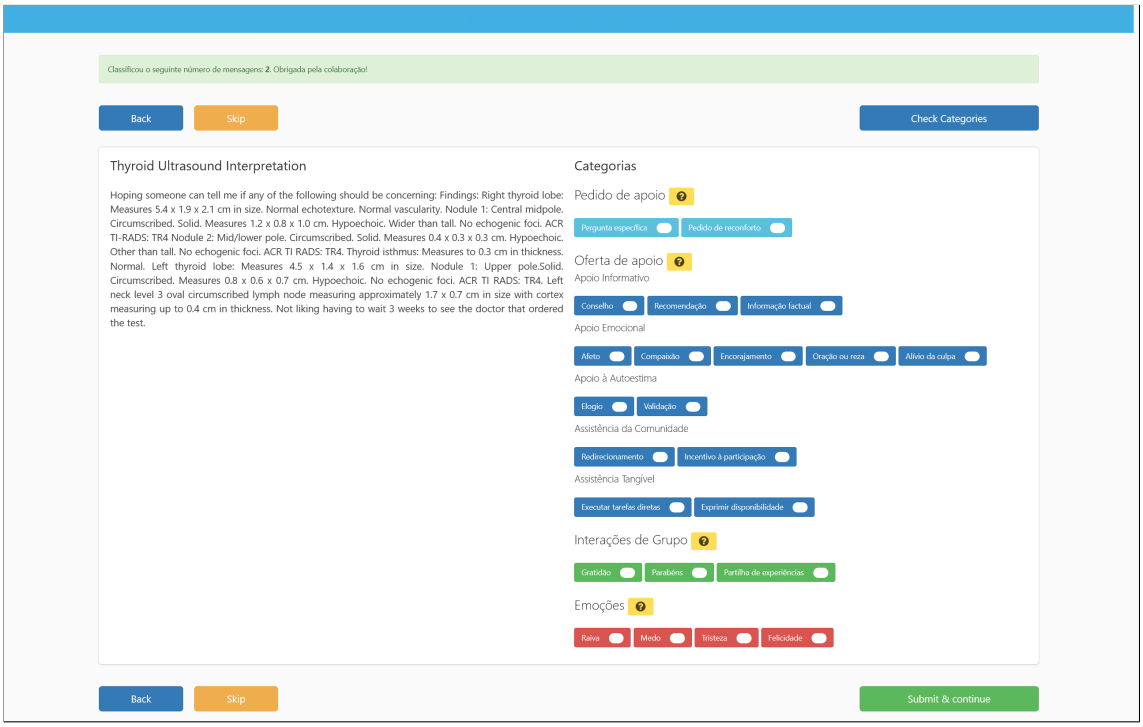

Figure B.2: Manual Message Classifier - Classification page - Example 1

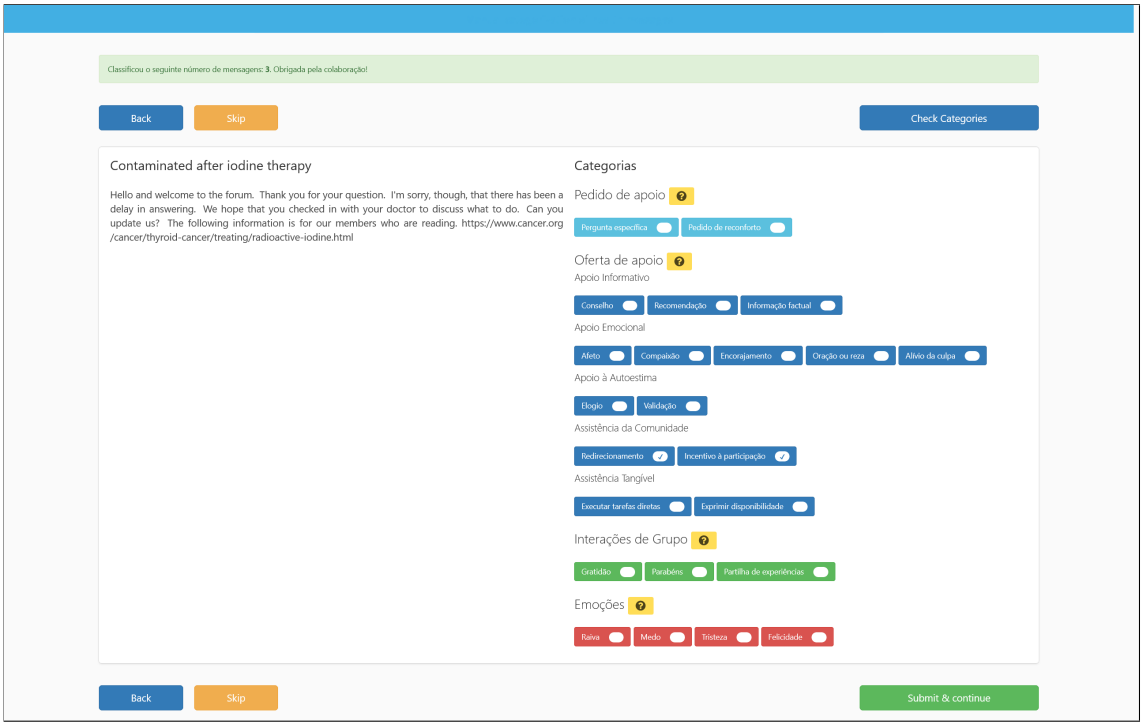

Figure B.3: Manual Message Classifier - Classification page - Example 2

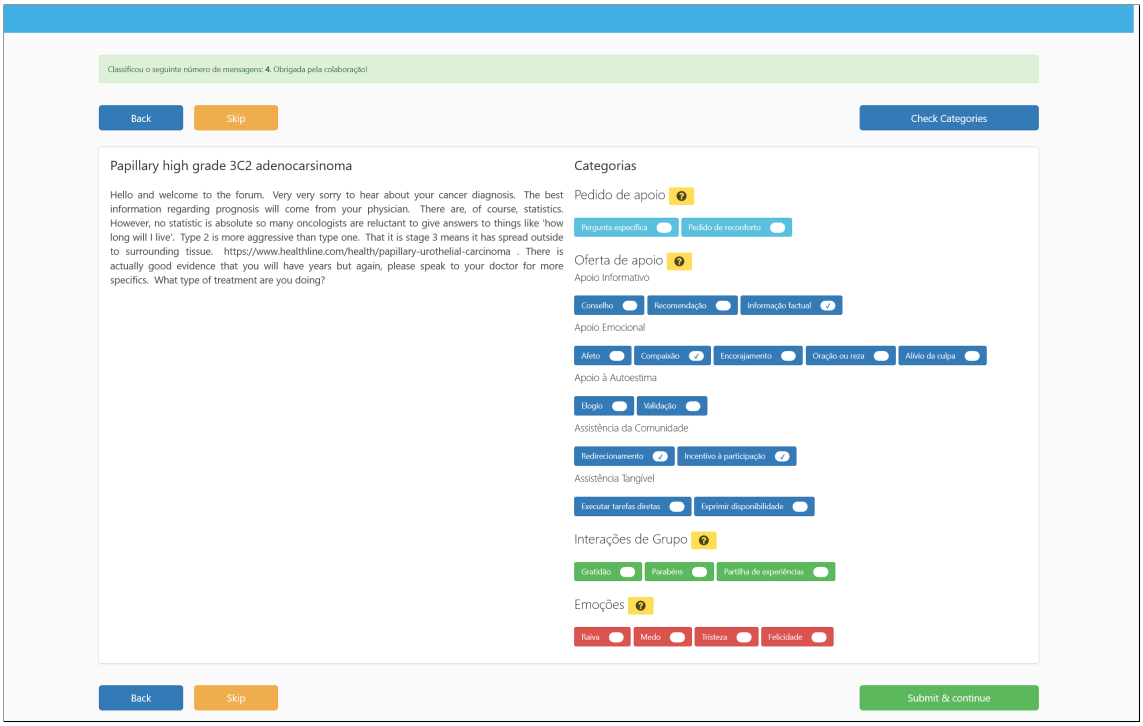

Figure B.4: Manual Message Classifier - Classification page - Example 3

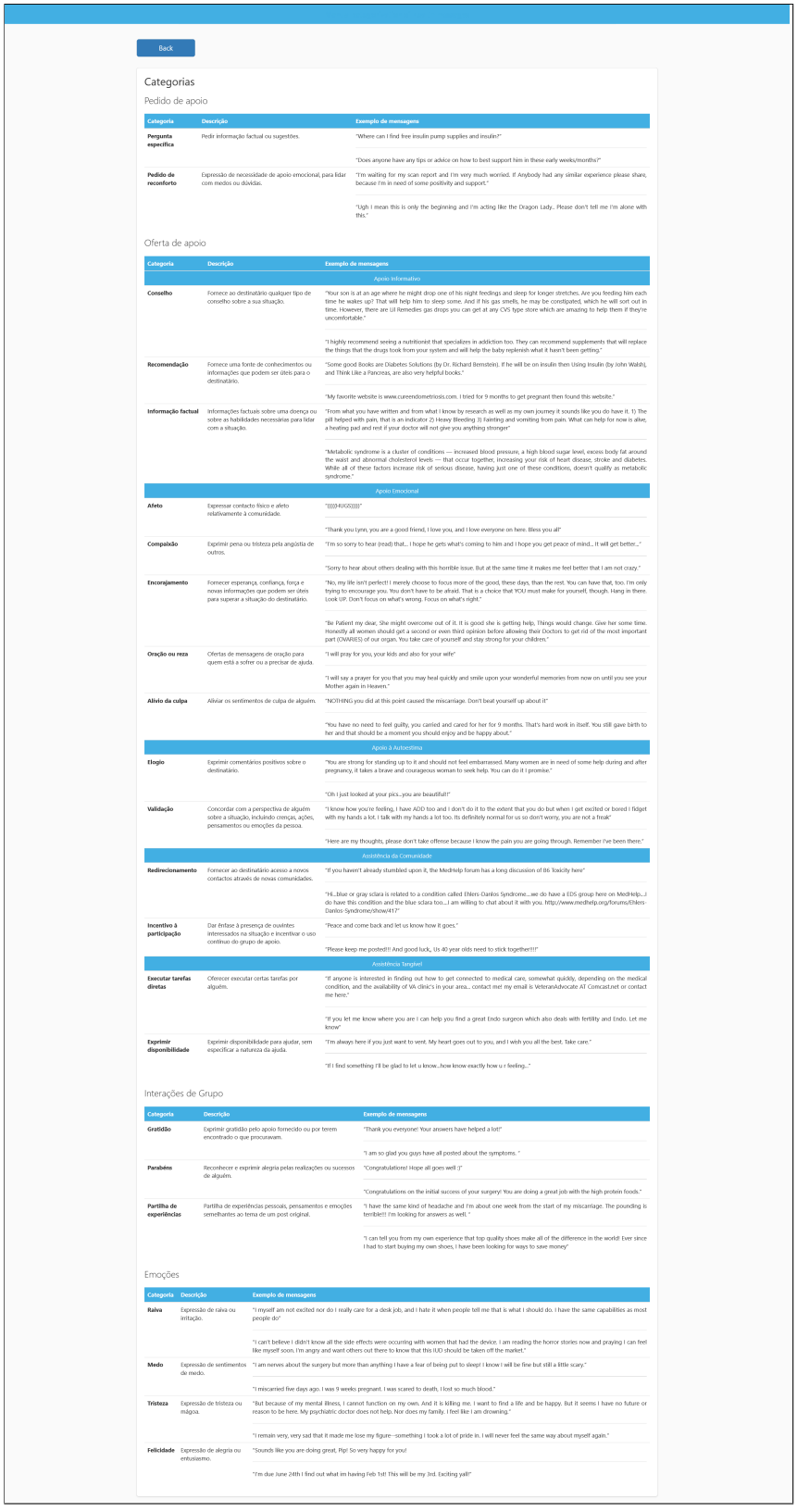

Figure B.5: Manual Message Classifier - Category details page

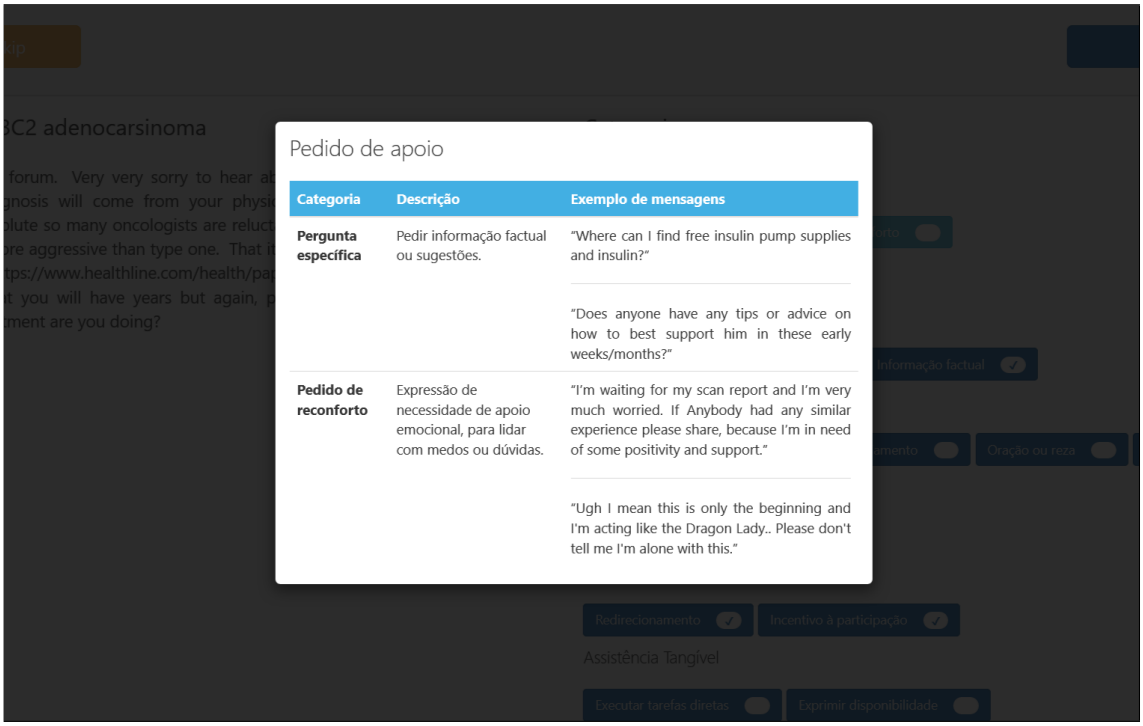

Figure B.6: Manual Message Classifier - Category details tooltip

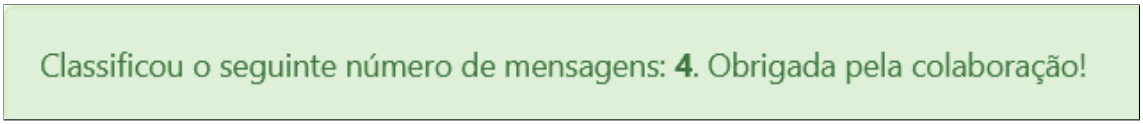

Figure B.7: Manual Message Classifier - Banner with number of classifications done

Appendix C

Statistical analysis of the crawled data

Table C.1: Number of threads and answers in the MedHelp community per forum topic.

Topic	Total threads	Total answers	Total comments	Combined	%
Neurology	57147	328852	6275	392274	17.02%
Hepatitis	19142	171025	3980	194147	8.43%
Pregnancy	24335	99943	2710	126988	5.51%
Thyroid	17020	101076	6452	124548	5.41%
Anxiety	18778	85436	4417	108631	4.71%
General Health	20446	70763	13197	104406	4.53%
Digestive	22111	76884	2289	101284	4.40%
Addiction	10110	79501	4179	93790	4.07%
Dermatology	23115	64065	3185	90365	3.92%
Heart Disease	18302	66123	1700	86125	3.74%
Pain	15804	68963	1078	85845	3.73%
Women's Health	13415	47223	3525	64163	2.78%
Children's Health	11910	48155	1313	61378	2.66%
Mood Disorders	10852	47535	1834	60221	2.61%
Dogs	8361	36206	714	45281	1.97%
Relationships	5292	36925	2564	44781	1.94%
Breast Cancer	8017	29964	264	38245	1.66%
Cats	5184	29670	768	35622	1.55%
Healthy Living	5625	25424	2011	33060	1.43%
Urology	8360	22950	1419	32729	1.42%
Autoimmune Diseases	5104	25614	479	31197	1.35%
Asthma	6018	21668	458	28144	1.22%
Ear, Nose, Throat	6080	20512	918	27510	1.19%
Cancer	4719	17434	317	22470	0.98%

Orthopedics	5553	15557	489	21599	0.94%
HIV	3601	10491	6847	20939	0.91%
Diabetes	3968	14679	661	19308	0.84%
Parenting	3338	15034	383	18755	0.81%
Arrhythmias	2720	13645	1677	18042	0.78%
Infectious Diseases	2403	13245	1103	16751	0.73%
Dental	3993	11696	308	15997	0.69%
Children's Development	2584	12207	271	15062	0.65%
Respiratory Disorders	3473	10735	737	14945	0.65%
STDs	2112	4827	7743	14682	0.64%
Sexual Health	1654	5463	971	8088	0.35%
Alternative Medicine	1179	6437	93	7709	0.33%
Trying to Conceive	1248	5845	471	7564	0.33%
Other Pets	1882	5580	7	7469	0.32%
Leukemia & Lymphoma	1076	5091	861	7028	0.31%
Ovarian Cancer	1260	5039	128	6427	0.28%
Sleep Disorders	1230	4184	388	5802	0.25%
Rare Diseases	1132	3718	42	4892	0.21%
Men's Health	1279	1701	1378	4358	0.19%
Adolescent Health	984	2570	219	3773	0.16%
Senior Health	825	2811	28	3664	0.16%
Eating Disorders	752	2603	41	3396	0.15%
Lung Cancer	696	2143	74	2913	0.13%
Personality Disorder	456	1602	469	2527	0.11%
Genetics	543	1823	17	2383	0.10%
Colorectal Cancer	562	1749	32	2343	0.10%
Prostate Cancer	524	1634	79	2237	0.10%
Coronavirus	212	701	1053	1966	0.09%
Skin Cancer	528	1121	17	1666	0.07%
Mental Health	314	935	329	1578	0.07%
Organ Donors	186	1148	19	1353	0.06%
Birds	282	1034	8	1324	0.06%
Blood & Vascular	302	955	5	1262	0.05%
Cosmetic Surgery	294	866	34	1194	0.05%
Cervical Cancer	183	427	14	624	0.03%
Babies	122	442	1	565	0.02%
Testicular Cancer	108	250	15	373	0.02%
Gastro Cancer	98	170	2	270	0.01%

MedHelp General	36	135	4	175	0.01%

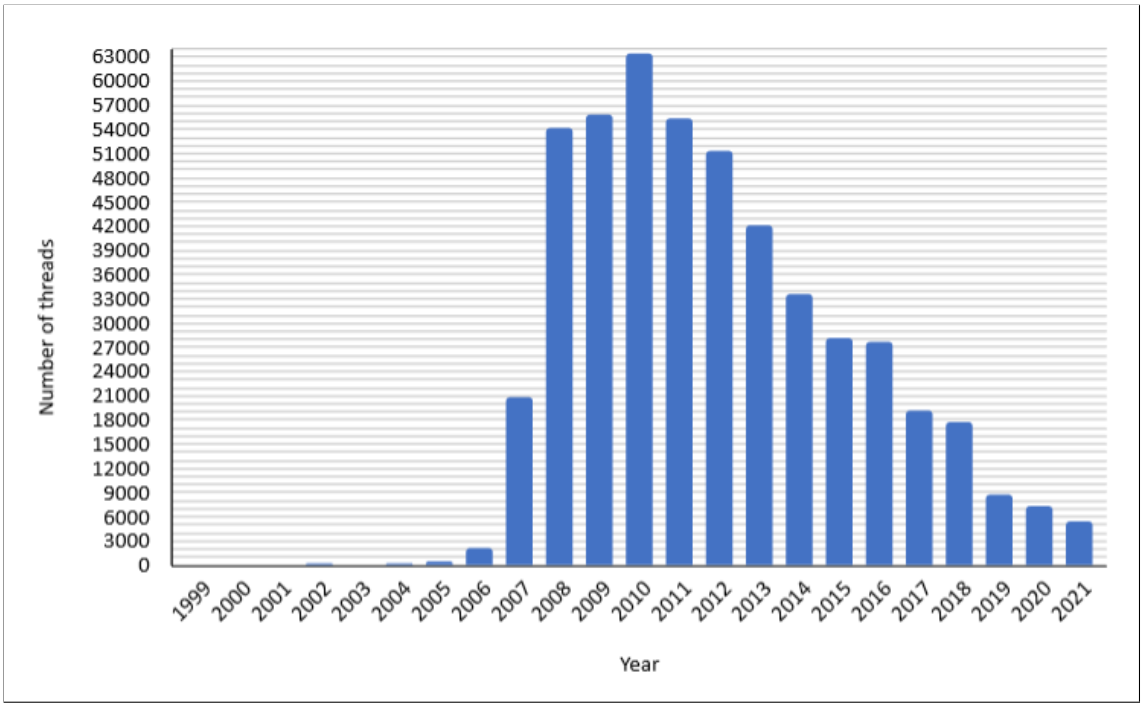

Figure C.1: Number of threads per year in the MedHelp community.

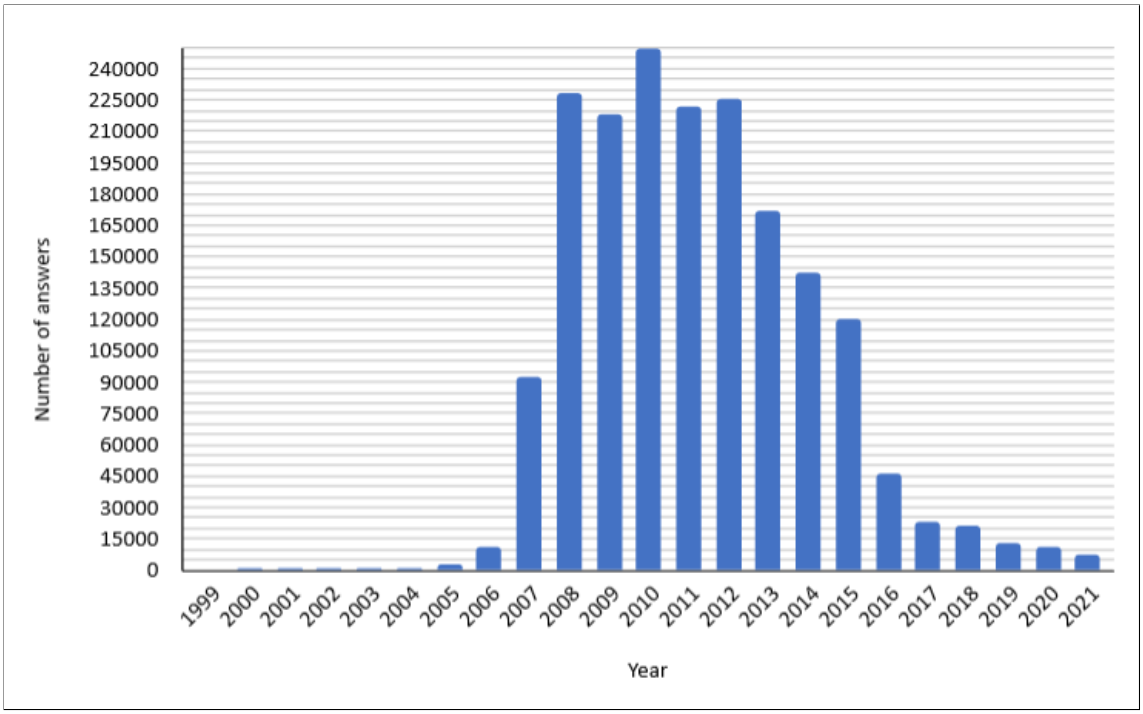

Figure C.2: Number of answers per year in the MedHelp community.