VÍCTOR MACÊDO ALEXANDRINO

UM WEB CRAWLER FOCADO NO RASTREAMENTO E CATALOGAÇÃO DE WEB FEATURE SERVICES (WFS) E WEB MAP SERVICES (WMS)

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

Orientador: Jugurta Lisboa Filho

Coorientadores: Altigran Soares da Silva

Giovanni Ventorim Comarela

VIÇOSA - MINAS GERAIS

2020

Ficha catalográfica elaborada pela Biblioteca Central da Universidade Federal de Viçosa - Campus Viçosa

T
A382w
2020

Alexandrino, Víctor Macêdo, 1994-
Um web crawler focado no rastreamento e catalogação de web feature services (WFS) e web map services (WMS) / Víctor Macêdo Alexandrino. – Viçosa, MG, 2020.
58 f. : il. (algumas color.) ; 29 cm.

Texto em português e inglês.
Inclui anexo.
Inclui apêndices.
Orientador: Jugurta Lisboa Filho.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Inclui bibliografia.

1. Sistema de recuperação de informação - Geografia. 2. Web crawler (Programa de computador). 3. Serviços da Web. 4. Open Geospatial Consortium. I. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-Graduação em Ciência da Computação. II. Título.

CDD 22. ed. 004.678

VÍCTOR MACÊDO ALEXANDRINO

UM WEB CRAWLER FOCADO NO RASTREAMENTO E CATALOGAÇÃO DE WEB FEATURE SERVICES (WFS) E WEB MAP SERVICES (WMS)

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 26 de maio de 2020.

Assentimento:

Víctor Macêdo Alexandrino
Autor

Jugurta Lisboa Filho
Orientador

A Deus, minha família, orientadores e professores

AGRADECIMENTOS

Agradeço a Deus pela oportunidade de viver e permanecer vivo, e pela ciência das coisas das quais Ele mesmo me permite saber. Exalto o Seu Nome pois, nada do que foi feito até o momento se faria se, por Ele e para Ele não fosse. Glória à Deus para sempre.

À minha esposa por não só me suportar no sentido “aquele que aguenta”, mas principalmente no sentido “aquele que ergue”, e me dar muito mais do que o amor eros promete. Aos meus sogros, cunhada e cunhado pela amizade e por rirem das minhas piadas.

Agradeço aos meus pais por terem me ajudado conforme a necessidade e disponibilidade. Ao meu pai pela inspiração na descoberta do novo, à minha mãe pela segurança e prontidão para mim, ao meu irmão por ter sido bom amigo e à minha irmã por ter me influenciado a estudar.

Obrigado à UFV por tudo, lá pude aprender mais do que assuntos acadêmicos, aprendi a viver. Obrigado também à Divisão de Assistência Estudantil que me proporcionou alojamento e alimentação durante a graduação. Sem dúvida, sem tais assistências eu não teria chegado até aqui no mestrado. Agradeço muito aos idealizadores de programas do Governo que me proporcionaram isto. Agradeço também o financiamento da CAPES por ter me concedido bolsa de mestrado e à FAPEMIG por ter me cedido um bom computador para uso na pesquisa.

Agradeço aos meus professores, em especial aos professores Altigran Soares da Silva e Giovanni Ventorim Comarela pela coorientação, e aos professores Lucas Francisco da Matta Vegi, Mauro Nacif Rocha e Ricardo dos Santos Ferreira por terem me ensinado tanto. E agradeço, em especial, ao Jugurta Lisboa Filho por ter me orientado durante tanto tempo, desde a graduação até aqui, foi uma jornada muito boa pra mim. Muito obrigado, Jugurta!

Enfim, obrigado a todos os amigos e colegas que não mencionei aqui, seja por esquecimento, ou pra não me prolongar demais. Obrigado pessoal!

RESUMO

ALEXANDRINO, Víctor Macêdo, M.Sc., Universidade Federal de Viçosa, maio de 2020. **Um Web Crawler focado no rastreamento e catalogação de Web Feature Services (WFS) e Web Map Services (WMS)**. Orientador: Jugurta Lisboa Filho. Coorientadores: Altigran Soares da Silva e Giovanni Ventorim Comarela.

O volume de páginas disponíveis na Internet vem crescendo significativamente, e isto se deve à democratização do acesso à tecnologia e ao próprio desenvolvimento da tecnologia que, por si só tem evoluído muito. Este fato se deve especialmente ao surgimento de novos *frameworks*, sejam eles proprietários ou em sua maioria *open source*, linguagens de programação mais modernas e metodologias de desenvolvimento e gestão mais ágeis. Com isto surge a questão de como se localizar páginas na *Web* mais facilmente. Esta é uma questão já parcialmente resolvida, principalmente quando se trata da busca por informações de maneira genérica. Existem máquinas de buscas, inicialmente desenvolvidas como pesquisa, que hoje já estão em estado consolidado e comercial, cujo propósito é realizar a busca de informações de maneira genérica, isto é, buscar informações de qualquer tópico de interesse do usuário. O problema vem à tona e, se mostra ainda em pauta, quando o assunto é a recuperação de informações mais específicas, ou que são de interesse de um público restrito. Esta pesquisa visa o desenvolvimento de uma máquina de busca cujo objetivo é encontrar *Web Feature Services* e *Web Map Services, ou seja, Web Services* que seguem os padrões do Open Geospatial Consortium. A metodologia empregada nesta pesquisa buscou resolver de maneira mais eficiente, em comparação com o estado da arte, a questão da recuperação de *Web Feature Services* e *Web Map Services* na *Web.* Para a apresentação dos resultados obtidos com o *core* da pesquisa foi implementado um portal *Web* que permite visualizar os resultados da busca por esse tipo de *Web service*.

Palavras-Chave: Recuperação de Informação Geográfica. Rastreamento na Web. Serviços Web. *Open Geospatial Consortium*.

ABSTRACT

ALEXANDRINO, Víctor Macêdo, M.Sc., Universidade Federal de Viçosa, May, 2020. **A Web Crawler focused on tracking and cataloging Web Resource Services (WFS) and Web Map Services (WMS)**. Advisor: Jugurta Lisboa Filho. Co-advisors: Altigran Soares da Silva and Giovanni Ventorim Comarela.

The volume of pages available on the Internet has been growing significantly, and this is mostly due to the democratization of access to technology, and to the own development of technology, which in itself has evolved a lot, a fact that is due especially to the appearance of new frameworks, being them proprietary or mostly Open Source, more modern programming languages, and more agile development and management methodologies. This raises the question of how to find such pages on the Web more easily. This is a problem that has already been partially resolved, especially when it comes to the search for information in a generic way because, there are search engines initially developed as research, and are now in a consolidated and commercial state, whose purpose is to search for information in a general, that is, to search for information on any topic of interest to the user. The problem comes to the fore, and it is still on the agenda, when the subject is the recovery of very specific information, or that are of interest to a very restricted audience, as is the case of the subject addressed by this research, Web Services that follow Open Geospatial Consortium standards: Web Feature Service and Web Map Service. This research addresses the subject with the presentation of a methodology that, proposes to solve more efficiently, in comparison to the state of the art, the issue of retrieving Web Feature Services and Web Map Services on the Web and, more than that is, the presentation of a web portal to view the results obtained with the research core.

Keywords: Geographic Information Retrieval. Web Crawling. Web Services. Open Geospatial Consortium.

LISTA DE ILUSTRAÇÕES

Figura 1- Graph components connected by irrelevant vertices on the left, BFC selecting seeds among the vertices of two of components on the right. 20
Figura 2 - SpatiHarvest's architecture. 23
Figura 3 - SpatiHarvest class diagram. 25
Figura 4 - Comparison between SpatiHarvest and PolarHub with regard to the number of identified WFSes and WMSes. 29
Figura 5 - Tela inicial do SpatiHarvest Portal. 34
Figura 6 - Tela com a lista de resultados retornada pelo SpatiHarvest Portal para a consulta "australia". 34
Figura 7 - Arquitetura na perspectiva de infraestrutura do SpatiHarvest Portal. 36
Figura 8 - Trecho do código onde são configurados os parâmetros do BFC. 41
Figura 9 - Execução do script de geração de conjuntos de treinamento. 42
Figura 10 - Inserção dos conjuntos de treinamento no banco de dados. 43
Figura 11 - Listagem de URLs irrelevantes no MongoDB. 43
Figura 12 - O número de *threads* é configurado na linha 30. 43
Figura 13 - Start do SpatiHarvest. 44
Figura 14 - Dockerfile do SpatiHarvest Portal - Back-end. 45
Figura 15 - Dockerfile do SpatiHarvest Portal - Front-end. 45
Figura 16 - *Build* do container do SpatiHarvest Portal - Back-end. 46
Figura 17 - *Build* do container do SpatiHarvest Portal - Front-end. 47
Figura 18 - *Start* do container do SpatiHarvest Portal - Front-end. 47
Figura 19 - *Start* do container do SpatiHarvest Portal - Back-end. 47
Figura 20 - Fazendo download e executando a imagem Docker do MongoDB. 48
Figura 21 - Recurso *GetCapabilities* de um *Web Map Service* recuperado pelo SpatiHarvest. 53
Figura 22 - Recurso *GetCapabilities* de um *Web Feature Service* recuperado pelo SpatiHarvest. 56

LISTA DE TABELAS

Tabela 1 - Results for query “brazil wfs services” and 89 seeds. SpatiHarvest (Spati.) and PolarHub (Polar.). 28
Tabela 2 - Results for query “ogc wms sources” and 96 seeds. 28
Tabela 3 - Results for query “ogc web services” and 96 seeds. 29
Tabela 4 - URLs de páginas consideradas relevantes considerando-se o tópico de interesse. 49
Tabela 5 - URLs de páginas consideradas irrelevantes considerando-se o tópico de interesse. 51

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
OGC	*Open Geospatial Consortium*
WFS	*Web Feature Service*
WMS	*Web Map Service*
WWW	*World Wide Web*
BFC	*Bootstrapping Focused Crawling*
SVM	*Support Vector Machine*
ETL	*Extract, Transform, Load*
JSON	*JavaScript Object Notation*
W2GIS	*Web and Wireless Geographical Information Systems*

SUMÁRIO

1 INTRODUÇÃO 11

1.1 Objetivos 12

1.2 Organização da Dissertação 12

2 ARTIGO CIENTÍFICO 14

2.1 A Focused Crawler for Web Feature Service and Web Map Service Discovering 15

2.1.1 Introduction 15

2.1.2 Background 17

2.1.3 Related Work 20

2.1.4 The SpatiHarvest Focused Web Crawler 21

2.1.5 Results and Discussion 27

2.1.6 Conclusion and Future Work 29

3 SPATIHARVEST PORTAL 33

3.1 Workflow 33

3.2 Tecnologias e Arquitetura 35

4 CONCLUSÃO 37

APÊNDICE A – Configuração do BFC no SpatiHarvest 41

APÊNDICE B – Setup para execução do SpatiHarvest 42

APÊNDICE C – Setup para execução do SpatiHarvest Portal 45

APÊNDICE D – Lista de URLs utilizadas para treinamento do SVM 49

ANEXO A – Exemplos de resultados encontrados pelo SpatiHarvest 53

1 INTRODUÇÃO

Nos últimos anos, com o avanço na difusão dos sistemas de informação pelo mundo, e o surgimento de novas tecnologias que agilizam e tornam mais acessível a criação de *Websites*, APIs e softwares de um modo geral, o tamanho e a complexidade da *World Wide Web* (WWW) vêm crescendo dia após dia, assim como um grande volume de informações espaciais tem sido produzido (LI; WANG; BHATIA, 2016), sendo tais fatores responsáveis pelo surgimento de questões relativas à localização e indexação de páginas *Web*.

A área de estudo de recuperação de informação lida com a representação, armazenamento, organização e acesso à informação (BAEZA-YATES; RIBEIRO-NETO, 2011) e, devido ao avanço das tecnologias provenientes deste ramo, o problema em questão já está parcialmente resolvido por buscadores como o Google (GHEMAWAT; GOBIOFF; LEUNG, 2003), Bing (MICROSOFT, 2020), Yahoo! (YAHOO!, 2020) entre outros. No entanto, este tema ainda está em pauta, pois grande parte da Internet ainda não foi indexada e a localização de assuntos muito específicos pode ser difícil em máquinas de busca genéricas.

O tópico de serviços geográficos, especificamente *Web Feature Services* (WFS) e *Web Map Services* (WMS), que são padrões de *Web Services* para o fornecimento de informações geográficas (OPEN GEOSPATIAL CONSORTIUM, 2020a) (OPEN GEOSPATIAL CONSORTIUM, 2020b), são exemplos de informações bastante específicas que são de interesse de um grupo relativamente pequeno de pessoas e, por isso, juntamente com os fatores citados, são assuntos difíceis de serem encontrados em máquinas de busca genéricas, tornando assim pertinente o investimento de esforços na busca de um método mais apropriado para tal tarefa.

Existem na literatura soluções já apresentadas para se resolver a busca por serviços Web geográficos, como o Polarhub (LI; WANG; BHATIA, 2016) e o GeoWeb Crawler (HUANG; CHANG, 2016). Este trabalho utiliza o Polarhub como *baseline* para fins de comparação de resultados. Tal solução foi escolhida por se tratar de algo computacionalmente mais avançado e com melhores resultados de acordo com

análises realizadas. Através de avanços que se acredita poderem ser obtidos nesta pesquisa, será possível indexar em menor tempo, maior número de fontes de informações geográficas, tornando assim mais acessível este tipo de serviço que tem potencial de dar suporte a diversas atividades de engenharia.

1.1 Objetivos

O objetivo geral desta pesquisa é desenvolver um método capaz de recuperar *Web Feature Services* e *Web Map Services* na Internet com menor esforço, isto é, em comparação com a melhor metodologia encontrada na literatura, Polarhub (LI; WANG; BHATIA, 2016). Uma melhor solução para tal tarefa pode auxiliar na catalogação de tais serviços, possibilitando assim a geração de uma maior base de conhecimento de dados geográficos que seja capaz de apoiar diversos tipos outras aplicações que tenham representações computacionais de informações geográficas como base.

Os objetivos específicos são:

1. Entender a estrutura dos *Web Feature Services* e *Web Map Services* e criar a automação da leitura de tais informações;
2. Definir e propor uma solução capaz de superar o estado da arte na recuperação de WMS e WFS;
3. Aplicar um método eficiente de recuperação de *seeds* como parte do diferencial na metodologia definida;
4. Desenvolver uma aplicação para fins de validação experimental da metodologia proposta;
5. Planejar e aplicar um experimento para analisar a eficiência do método proposto neste trabalho, sendo tal análise realizada de maneira comparativa com outro método que represente o estado da arte;
6. Desenvolver um portal Web para fornecer acesso aos resultados obtidos com o método proposto e implementado.

1.2 Organização da Dissertação

Esta dissertação foi elaborada tendo como base o artigo produzido e aceito para publicação na conferência *W2GIS 2020: Web and Wireless Geographical*

Information Systems – 2020, bem como o portal *Web* denominado SpatiHarvest Portal que, foi construído a fim de prover acesso facilitado às informações recuperadas pelo *core* da pesquisa.

O Capítulo 2 é composto integralmente pelo artigo produzido, onde apresenta-se a fundamentação, a metodologia e os resultados obtidos. O título do artigo é: *A Focused Crawler for Web Feature Service and Web Map Service Discovering*.

No Capítulo 3 é apresentado o SpatiHarvest Portal. O sistema é descrito detalhando-se o *workflow* da aplicação, assim como as tecnologias utilizadas na implementação do mesmo. Por último apresenta a arquitetura com ênfase na infraestrutura e serviços de operações utilizados.

O Capítulo 4 apresenta o fechamento do trabalho e possibilidades de extensões e aprimoramentos futuros tanto do núcleo da pesquisa como do SpatiHarvest Portal.

O Apêndice A descreve como é feita a configuração do BFC, algoritmo que recupera *seeds* para o SpatiHarvest.

Os Apêndices B e C descrevem como deve ser feito o *setup* para a execução do SpatiHarvest e SpatiHarvest Portal, isto é, a configuração do ambiente de maneira que esses sistemas possam ser executados.

O Apêndice D contém um exemplo de *Web Feature Service* e um exemplo de *Web Map Service*, ambos recuperados pelo SpatiHarvest.

Por fim, o Anexo A lista o conjunto de URLs que foi utilizado para o treinamento do SVM dentro do SpatiHarvest.

2 ARTIGO CIENTÍFICO

Este capítulo contém o artigo resultante da pesquisa. O artigo, cujo título é "*A Focused Crawler for Web Feature Service and Web Map Service Discovering*", foi aceito para publicação no *W2GIS 2020: Web and Wireless Geographical Information Systems - 2020* que aconteceria nos dias 14 e 15 de maio em Wuhan, China. No entanto, devido à pandemia Covid-19 se tornou um evento a ser realizado por videoconferência na mesma data. O artigo apresenta a fundamentação teórica sobre a qual a solução de construção do SpatiHarvest foi concebida. Ao final, detalha os resultados obtidos em comparação com outro algoritmo que se propõe a fazer trabalho similar e sugere trabalhos futuros que podem ser pertinentes à implementação.

2.1 A Focused Crawler for Web Feature Service and Web Map Service Discovering

Víctor Macêdo Alexandrino, Giovanni Comarela,
Altigran Soares da Silva, Jugurta Lisboa-Filho

ABSTRACT

Due to globalization and the technological advances of the last decades, a large amount of data is created every day, especially on the Web. Web Services are one of the main artifacts created on the Web; they can provide access to sources of data of many sizes and types. In this work, we approach the challenge of designing and evaluating a Web Crawler to find two OGC (Open Geospatial Consortium) web services standards: WFS (Web Feature Service) and WMS (Web Map Service). Commercial search engines, e.g., Google and Bing, are indubitably doing a useful job as general-purpose search engines. However, some applications require domain-specific search engines and Web crawlers to find detailed information on the Web. Therefore, this work presents SpatiHarvest, a Web crawler that focuses on the task of discovering WFSes and WMSes. SpatiHarvest is a combination of some of the most advanced techniques found in the literature. Through experiments, we demonstrate that it is capable of finding more WFSes and WMSes, with less effort, when compared to the state of the art.

Keywords: Geographic Information Retrieval · Web Crawling · Web Services · Open Geospatial Consortium.

2.1.1 Introduction

Spatial information has been widely used in different types of systems and applications, mainly due to the increasing facility of obtaining this kind of data. Spatial information is useful to answer questions and support the decision making in different areas, such as: environment, climate, urban traffic, geo-marketing, a fast reaction in natural disaster cases, among others. Many private and governmental institutions get large amounts of data of such nature, which little by little are made available through products and services, such as web services [1].

Studies show that a large volume of non-indexed pages, even by large search engines, is present on the Web [2]. Therefore, due to the variety of content and the high volume of information on the Web, it is worth studying and developing Web crawlers (also known as spiders or robots), especially those that are domain-specific. In this context, it is possible to assist the robot to explore the domain more extensively by applying techniques such as BFC (Bootstrap Focused Crawler) [3], filters, and supervised learning. This paper considers the domain of Web Services under the WFS (Web Feature Service) and WMS (Web Map Service) standards, which are both for providing spatial data interoperability and maintained by the OGC (Open Geospatial Consortium). More specifically, our goal is to build a Web crawler focused on discovering Web pages that can be or contain URLs (Uniform Resource Locators) related to WFSes or WMSes. In this paper, these Web services are also referred to as spatial information sources.

Due to search engines, the problem of discovering WFSes and WMSes is partially solved, once the surface Web is constantly scanned and indexed. However, the surface layer of the Web is estimated to be considerably smaller than its hidden layer [4]. Such a difference is mainly due to two reasons: sometimes, a crawler is not capable of reaching certain pages, for instance, due to the pruning applied in favor of the search space reduction; and many pages are dynamically generated via queries and transactions.

Given that the non-indexed layer of the Web is considerably larger than its surface and that until 2020 the volume of information on the network may reach 44 zettabytes [5], it is a reasonable hypothesis that there may be a significant amount of spatial data sources yet to discover and to index. Search engines, such as, Bing, Google, and Yahoo!, are not specifically designed to find URLs of OGC Services when using basic queries [26]. Furthermore, even resources that are already known may not be so easily located via queries made to those search engines. This is because such resources may be amid a large volume of related, but not relevant search results, once commercial search engines are designed to be general-purpose.

Motivated by the such problematic, this paper presents a Web crawler called SpatiHarvest, which was designed by combining techniques and concepts from the Web crawling literature such as bootstrap focused crawling, parallel crawling, and

focused Web crawling. SpatiHarvest's goal is to improve the discovery of WMSes and WFSes. We conducted a series of experiments to compare SpatiHarvest to the state of the art PolarHub [6]. Our results show that SpatiHarvest is more efficient and that it is capable of finding more WMSes and WFSes for the same query.

The remaining of this paper is organized as follows: Section 2 presents the theoretical foundation behind SpatiHarvest. Section 3 presents related work. In Section 4, design and implementation are described. In Section 5, we present a comparison between SpatiHarvest and PolarHub. Finally, Section 6 presents conclusions, open questions, and directions to future work.

2.1.2 Background

In this section, we present the basic concepts, which are necessary to understand SpatiHarvest's architecture. More specifically, we discuss general web crawling, focused web crawling, parallel web crawling, and bootstrap focused crawlers.

2.1.2.1 Web Crawler

Web Crawler is a software, also known as robot or spider, built with the purpose of indexing Web pages. The first crawlers appeared almost at the same time as the Internet itself [24]. Such crawlers can be constructed using algorithms such as Depth-First Search and Breadth-First Search. In any case, the search starts from a set of URLs (seeds). Such a set can be chosen arbitrarily or using some method of seed selection. Once a page is discovered and retrieved, its source is parsed and analyzed. At this stage, any URL present in the page's source is also processed in a similar fashion [7].

There are several challenges for implementing a crawler. One of these challenges is choosing which URLs to use during the search. More specifically, given the URLs (links) on a page, which ones should the crawler also explore? In this context, the set of URLs that will be explored and the order of such exploration are relevant. The decision of which URLs should be used in the search can be guided by different methods. Some examples of criteria are: current depth of the search, page's language, domain's country, and, in some cases, the page's topic (or subject). In the latter,

supervised learning (e.g. SVM - Support Vector Machines) techniques can be used in order to classify if the URLs contained in a certain page will be added to the frontier of exploration or not [8].

2.1.2.2 Focused Web Crawler

A Focused Web Crawler is a regular Web crawler, which aims at indexing only Web pages related to a specific topic of interest. For instance, one can create a Web crawler that focuses on health care pages or academic papers available on the Web. The idea of using crawlers on certain topics is not new. Since 1999, there have been many proposals on reducing the domain of pages to be indexed by a robot. In this context, the robot may not be able to answer any question queried by a user (such as queries outside the scope of interest). However, queries made within the topic explored by the robot can be answered with more up-to date and relevant information. Moreover, the focused version can considerably save computational resources (e.g., network and disk) since it does not have to explore nor index the whole Web [9].

As for regular Web crawlers, Focused Web Crawling algorithms can be the result of several combinations of techniques, such as Breadth-First Search and Best-First Search [22]. However, there are specific challenges that need to be addressed to achieve efficiency. For instance, a general crawler aims at indexing the whole Web, while a focused crawler has to remain within the topic of interest, otherwise, the results of the query might not be meaningful and computational resources may be unnecessarily used. In addition, the work of a focused crawler starts even before any query is made. Initially, a set of seed pages [10] has to be formed. These pages are necessary to inform the robot where to start exploring. Ideally, the seeds have to enable the discovery of any page on the Web related to the topic of interest. Hence, given a topic, building the right set of seeds is an important challenge by itself.

In particular, a crawler that focuses on WFS and WMS tracking significantly differs from a general one. For instance, the focused crawler should prioritize WFSes and WMSes over regular Web pages, which not necessarily is true in the general case. Furthermore, queries can be performed differently in these two scenarios, i.e., generally, queries are textual, composed by keywords, but in the focused case, they may be based on a set of geospatial attributes.

2.1.2.3 Parallel Web Crawler

From an implementation perspective, a crawler may have multiple instances running in parallel, and there are different ways, with different challenges, to do so. Two of the possible architectures for a robot are the distributed and intrasite. In the first, the robot instances are not in the same computer network, i.e., they can be far away, at different facilities, communicating over the Internet. In the second, the robots are in the same local network, i.e., they are usually closer and might communicate over the Intranet [23]. There is also the possibility of running different robot's instances in the same computer, but in different processors, cores, or threads, or even using a Graphic Processing Unit (GPU).

When using a parallel crawler, it is necessary to define some architectural details besides those mentioned above. The robots can function in an independent, static, or dynamic fashion. In the independent mode, each robot's instance works from the seed set itself, without consulting other instances for any purpose, i.e, it is a simpler architecture, but it may suffer from serious problems, such as duplicate page indexing. In the dynamic operation, there is a central controller regulating the work of all other robots, constantly distributing link partitions to each instance of the Crawler. In the static operation, initially each instance of the robot receives a seed partition and then only operates from that partition. Hence, a static architecture does not demand a central controller [23].

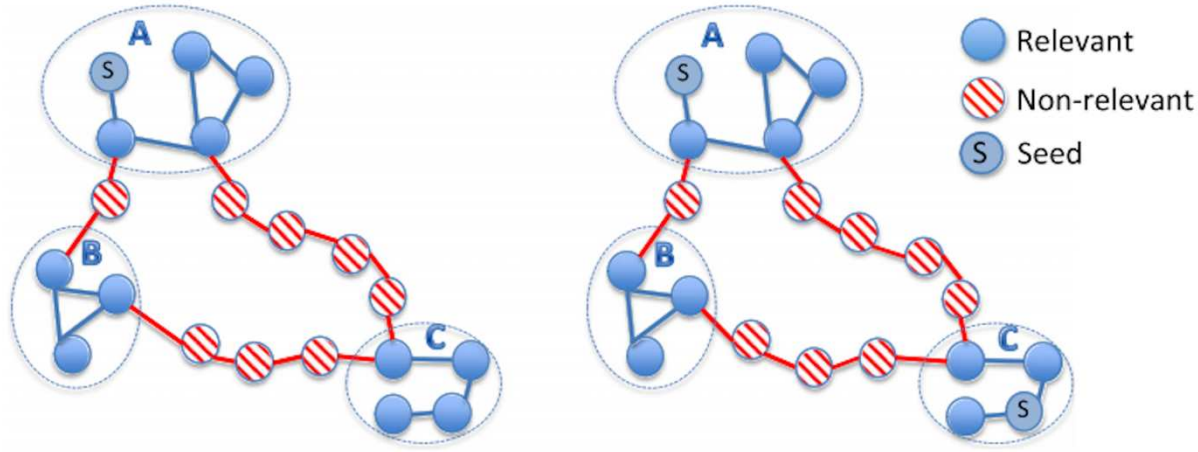
2.1.2.4 Bootstrapping Focused Crawling

Bootstrapping Focused Crawling (BFC), a framework created with the purpose of supporting focused Web crawlers, is a method to improve the process of automatically finding a good set of seeds, given a query, hence, increasing the reach and harvest rate within a topic [3].

The motivation for using BFC is presented in Figure 1, where an important challenge that needs to be overcome can be seen. In the figure, the nodes (i.e., pages) of interest (solid blue) are in different connected components. Those components are isolated from each other by non-relevant nodes (dashed red). A crawler that places all the seeds in a single component (e.g., Figure 1(left)) will not be able to discover all

relevant pages via standard methods, such as Breadth-First Search or Depth-First Search. The idea of using BFC is to automatically spread the seeds across several connected components of relevant pages (e.g., Figure 1(right)).

Figura 1- Graph components connected by irrelevant vertices on the left, BFC selecting seeds among the vertices of two of components on the right.



Fonte: (VIEIRA *et al.*, 2016)

In order to spread seeds in different components, BFC works as follows: given an input query, (e.g., “Web Feature Services from Australia”), N new queries are built using different keywords related to the same subject of the input. Those keywords are extracted from pages already retrieved on the first query. Then, for each new query, searches on general-purpose search engines are performed in order to obtain a more varied, but still within the scope, set of results. Finally, by joining all the retrieved results, the algorithm has as output a set of seeds that more likely will spread across the Web.

2.1.3 Related Work

There is a significant amount of literature related to general-purpose Web crawlers and focused Web crawlers. Specifically in the context of this work, a geospatial focused Web crawler, significantly fewer initiatives were found. Some of those are worth taking note of. GeoWeb Crawler [11] is a search engine designed with the primary purpose of “providing access to geospatial data or services through the worldwide coverage of the Internet”. The algorithm presented in [11] has some characteristics that stand out, such as the use of the map-reduce paradigm, where a

master performs the mapping of a set of URLs to workers. Then each worker plays the role of retrieving the page from its URLs. In the end, when each worker has finished its task, a reduce step is performed and the recovered pages are returned to the master. Another important feature is the method used to detect duplicate pages.

A bloom filter [12], which allows storing a URL via hash, thus saving memory and processing time. However, such a method does not solve the problem of Duplicate URLs with Similar Text (DUST) [13], since the duplication analysis is solely based on the comparison of the hashes corresponding to the URLs.

Polarhub is a search engine built to, “discover geospatial data distributed on the Web in an efficient and effective way” [6]. The algorithm developed has, among others, the following characteristics: it uses seeds retrieved from Google, which is a common practice in the literature [14–16], considering that Google Search is able to return relevant generic results from a given input query, thus providing subsidy for a good start of search and indexing; the crawler uses parallel processing to “speed up” the page retrieval process; and it performs a URL rebuild process, in an attempt to create a URL that may correspond to an OGC Web service standard.

A disadvantage of both algorithms mentioned above is that they do not perform any type of pruning when exploring the Web graph. Thus, even though a retrieved page is completely irrelevant (e.g., out of topic), the crawler will still use the URLs contained in this page to continue the search process, as long as the maximum depth allowed is not reached. Hence, processing time and storage may be wasted by analyzing irrelevant pages. Another problem is that no kind of refined technique is applied to the process of building an appropriate set of seeds, i.e., the presented algorithms only retrieve URLs via Google queries and by using the results from those queries, they perform all the search and indexing. As described in the remainder of this manuscript, SpatiHarvest is designed to overcome these two main issues.

2.1.4 The SpatiHarvest Focused Web Crawler

In this section, we present the SpatiHarvest architecture and the principles that guided its design. We also discuss the technologies that were used in our proof of

concept. We emphasize that, despite our implementation decisions, there are many other viable choices.

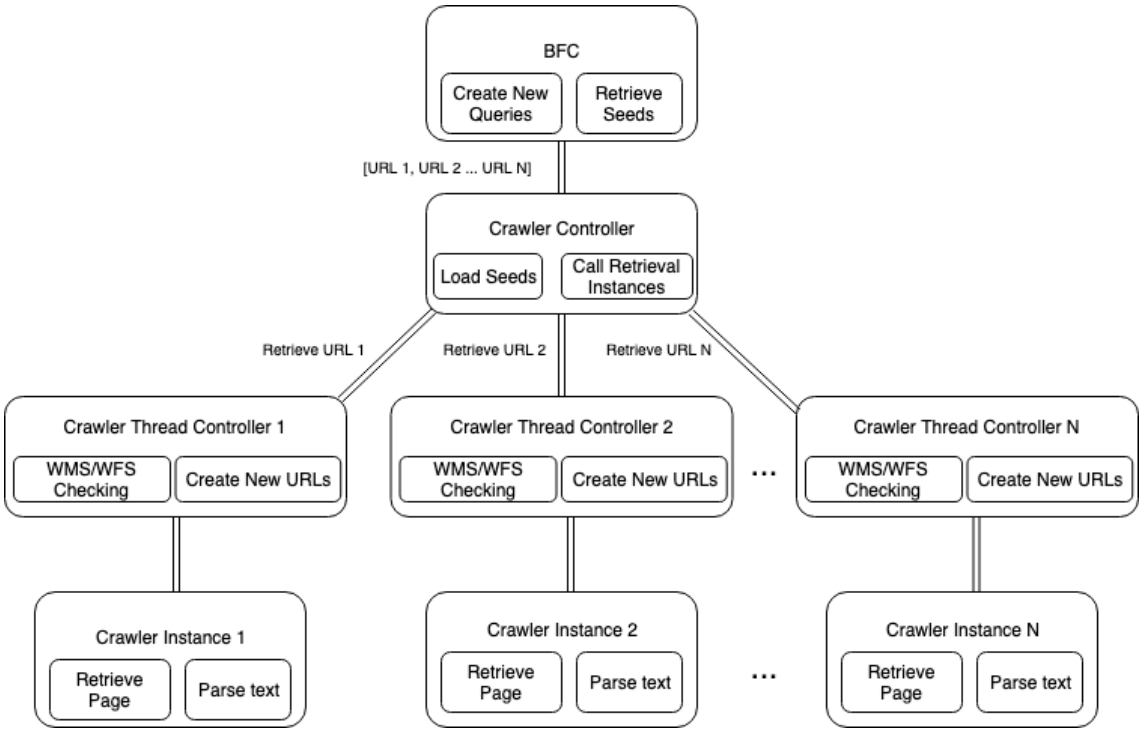
2.1.4.1 System Architecture

SpatiHarvest's architecture is illustrated in Figure 2. The system has three main components. The first one is the BFC (Bootstrap Focused Crawler), which is responsible for the seed selection. The second, Crawler Controller, has the role of a central controller, being responsible for distributing URLs to the workers. The third component, Crawler Controller Thread, may have many instances, each one with the same role as a worker node, where most of the crawler's work is effectively performed. More specifically, given a URL sent by the Crawler Controller, each Crawler Controller Thread is responsible for retrieving the page, parsing the page's source, checking the page's type (i.e., whether the page is a Web Feature Service, Web Map Service, or none of them), and persisting the data into a database.

Figure 2 also reveals the SpatiHarvest's parallel nature. There is a central controller that is in charge of managing and delegating URLs to several workers, which run in parallel. SpatiHarvest's multi-thread architecture can be scaled vertically or horizontally [25], depending on specific implementation decisions and application requirements.

An important problem that SpatiHarvest has to deal with is whether a URL is relevant or not. More specifically, whether a Web page is related to geospatial services (WMSes or WFSes) or not. This decision is critically important both to the BFC and Crawler Controller components. To deal with this problem, SpatiHarvest relies on supervised machine learning. To that end, prior to SpatiHarvest execution, a set of pages (relevant and not relevant) has to be collected and properly labeled. The labeled dataset can be used to train a binary classifier and, finally, the trained model can be used by SpatiHarvest during its execution. The more accurate the model, the higher precision and recall SpatiHarvest can achieve.

Figura 2 - SpatiHarvest's architecture.



Fonte: O autor.

2.1.4.2 SpatiHarvest Prototype

In order to put SpatiHarvest's architecture to test, we implemented a prototype. Next, we describe our implementation decisions and the prototype's workflow.

2.1.4.2.1 Implementation Description

Our prototype was developed using Python 3.6.5 programming language. The choice of such language was due to the following factors: availability of a wide variety of free and open-source libraries, a very active developer community, and a gentle learning curve. Figure 3 presents the UML class diagram of our prototype. The diagram shows the main classes, their methods, and the relationships among them. We discuss the role of the most important classes next.

The BFC class is responsible for running the BFC algorithm. In our case, the prototype relies on the Google Search Engine to create an initial set of seeds. This is done via the ExternalSources class, which deals with issuing the requests to google.com and parsing the responses.

The CrawlerController class essentially orchestrates the whole process. It loads seeds via the BFC algorithm, it manages the workers by delegating URLs to instances of the CrawlerThreadController class, and it updates the state of the collection process via an instance of the Frontier class.

The CrawlerThreadController and Crawler classes implement the functionalities related to the workers that run in parallel. Our prototype's multi-thread architecture scales vertically, but if computational resources are available, it can be easily extended to scale horizontally. The instances CrawlerThreadController are directly managed by the central CrawlerControler and they are responsible for checking if a page contains WMSes or WFSes and to extract URLs from such a page. The results are then returned to the instance of CrawlerController. Each CrawlerThreadController is associated with one instance of the Crawler class. The latter is responsible for issuing the HTTP request related to a URL (i.e., retrieving the page) and for parsing the content of the response, if any. In terms of libraries, all the text parsing and processing is handled by the NLTK library [17], while for the HTTP requests, we relied on urllib [18].

The Frontier class stores all URLs, whether they belong to pages already visited, pages not visited, or WFS/WMS. Basically, this class is responsible for keeping the consistency of the work that has been already done and the URLs that need to be explored. Also, once a URL has its page collected and parsed, this class is responsible for communicating with an instance of DAO class to save the result.

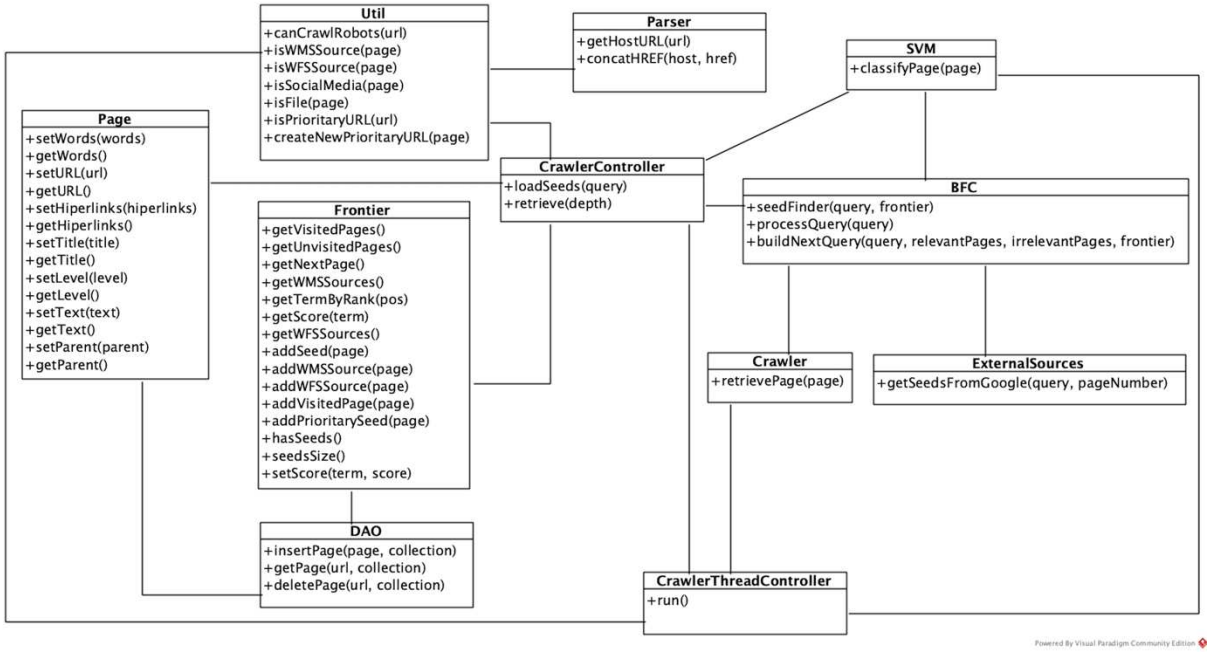
The DAO (Data Access Object) class is responsible for providing an interface for database operations. It is useful to isolate all the database logic into a single component, thus reducing the complexity of the code. Underneath the DAO class, for the persistence of the retrieved data, the Database Management System (DBMS) NoSQL, MongoDB [20] was used. This choice was due to the demand for a large volume data storage, the flexibility and scalability of MongoDB, and the fact that it is a document-oriented DBMS, which fits perfectly the domain of our problem.

The SVM class implements the SpatiHarvest's machine learning capabilities. It has an off-line component for training a SVM (Support Vector Machine) binary classifier prior to any SpatiHarvest query and an on-line component, which can be used to decide whether a page is relevant (i.e., related to the WFS and WMS domains) or not during a query processing. In our prototype, the SVM classifier from the scikit-learn

[19] library, which was trained using the textual content of 100 pages manually labeled. From those, 50 pages were labeled as relevant and 50 as irrelevant. The relevant pages were chosen considering pages containing URLs to WMSes, WFSes, or content closely related to the subject. The irrelevant pages were chosen considering pages with content as different as possible from the relevant pages.

The Util class provides a series of useful resources for the proper functioning of the crawler. For instance, the canCrawlRobots queries the robots.txt of a domain to know if a URL from that domain can, in fact, be crawled by a robot.

Figura 3 - SpatiHarvest class diagram.



Fonte: O autor.

2.1.4.2.2 Workflow

Given an input query, the crawler sends it to the BFC, which performs the seed retrieval for the beginning of the crawl. Once retrieved and properly stored in the Frontier, the CrawlerController creates N parallel instances of the CrawlerThreadController and assigns to each one the task of retrieving the page associated to each seed or URL not yet visited. This process is repeated successively as long as there are URLs not yet visited in Frontier. For each retrieved page, the robot checks if it is a WFS or WMS and if so, the page is added to Frontier's WFS/WMS list. If the page is not a WMS nor a WFS, the SVM is used to determine if the page content

is or not relevant. If it is relevant, the URLs contained in this page are added to the unseen page queue, if it is irrelevant, those URLs are discarded.

The WFS and WMS identification occurs by combining regular expression matching and keywords search in the page's text. For the WFS verification, first, the crawler looks for the string service=wfs in the URL. If there is no occurrence, it is assumed that the page is not a WFS. Otherwise, the HTML tag is searched within the page's text. If such a tag is found, it is assumed that the page is not a WFS. Otherwise, one last check is performed by looking for the regular expression (wfs(:|_)(capabilities|featurecollection|getfeature|valuecollection)). If such a pattern is found in the page's text, the page is finally assumed to be a WFS.

A similar procedure is used for WMS identification. However, during the first step, a search for the string service=wms is performed and during the third step, a regular expression is not used. Instead, a search is performed by looking for an occurrence of the strings wms_capabilities and wmt_ms_capabilities. If any of those are found, there is a confirmation that the page is a WMS.

Algorithm 1 summarizes the SpatiHarvest's workflow. In line 1 the robot checks if there are seeds or unvisited pages loaded in the Frontier. If there are none, a new set of seeds is constructed from URLs returned by the BFC. Otherwise, those seeds are used for crawling. This verification of unvisited URLs in the Frontier becomes useful in the scenario in which the robot, for some reason, has its execution interrupted during a search and then, after some time, it is resumed. In lines 11 through 16, commands are executed in a thread, thus enabling the crawler to retrieve and parse N web pages in parallel. The definition of a priority URL, term used in lines 8 and 22, refers to URLs that have service=wfs or service=wms as substring.

Algorithm 1: SpatiHarvest

```
Input: user defined query
Output: list of WMSes and WFSes
if Frontier has no seed then
    Retrieve seeds using BFC
    Add seeds returned to the Frontier's unvisited page list
while Frontier has seeds do
    Get Frontier's next page
    if page's level is larger than the maximum allowed search depht then
        continue
    if URL is priority then
        Create a new priority page
        Add the priority page to the beginning of the unvisited pages list
    Retrieve Web page
    Tokenize all the words contained in the page
    Remove stopwords
    Search for hyperlinks in the page
    Search for hyperlinks in element tags <a>
    Add the retrieved page to the visited pages list and to the database
    if Retrieved page is a WMS or WFS source then
        Add page to the WMS/WFS list of pages and to the database
    if Retrived page is classified as relevant by SVM then
        for each hyperlink in the page do
            if hyperlink is not a file or a social media link then
                if link is a prioritary link then
                    Add the page to the beginning of the unvisited pages list
                else
                    Add the page to the end of the unvisited pages list
```

2.1.5 Results and Discussion

In order to evaluate our SpatiHarvest prototype and compare it to the state of the art, i.e., PolarHub, we conducted several experiments. First, we had also to implement PolarHub, since its code it is not publicly available. During the implementation, we had to make some assumptions, once some details about PolarHub were omitted in their manuscript. Those assumptions were mainly related to the process of deciding whether a given Web page was a WMS or WFS. Both crawlers were run on an instance of the Google Compute Engine, with 26 GB of RAM, 4 vCPUs, and a Premium network level. For each input query, both crawlers were executed simultaneously and the same number of seeds was used in each one.

The experiments were performed with both crawlers and the results are presented in Tables 1 to 3. Table 1 shows results related to the query brazil wfs services, Table 2 to query ogc wms sources, and Table 3 to the query ogc web services. The “URLs Range” column refers to the number of visited URLs. For example, in Table 1, the first row shows the number of WMSes and WFSes identified in the first 10,000 analyzed URLs.

From the numbers, it is possible to observe that SpatiHarverst has performed better than Polarhub with regard to WMS and WFS identification. For SpatiHarvest, most of the Web services were identified early during the crawling execution and after that, the number dropped significantly. This fact points to the high efficiency of the seed recovery mechanism employed by SpatiHarvest, i.e., BFC. Also, during the experiments, it was detected the inability to identify a WMS with the regular expression used by Polarhub, namely wms(?:t_ms|s)_capabilities. Such expression did not match with WMS URLs nor with the content of the page returned by the getcapabilities functionality of a WMS.

Tabela 1 - Results for query “brazil wfs services” and 89 seeds. SpatiHarvest (Spati.) and PolarHub (Polar.).

URLs Range	WMS – *Spati.*	WFS – *Spati.*	WMS – Polar.	WFS – Polar.
1 – 10000	113	45	0	39
10001 – 20000	0	2	0	2
20001 – 30000	0	0	0	5
30001 – 40000	0	0	0	0
40001 – 50000	0	0	0	0

Fonte: O autor.

Tabela 2 - Results for query “ogc wms sources” and 96 seeds.

URLs Range	WMS – *Spati.*	WFS – *Spati.*	WMS – Polar.	WFS – Polar.
1 – 10000	135	48	0	6
10001 – 20000	1	0	0	0
20001 – 30000	5	0	0	0
30001 – 40000	9	0	0	0
40001 – 50000	6	0	0	0

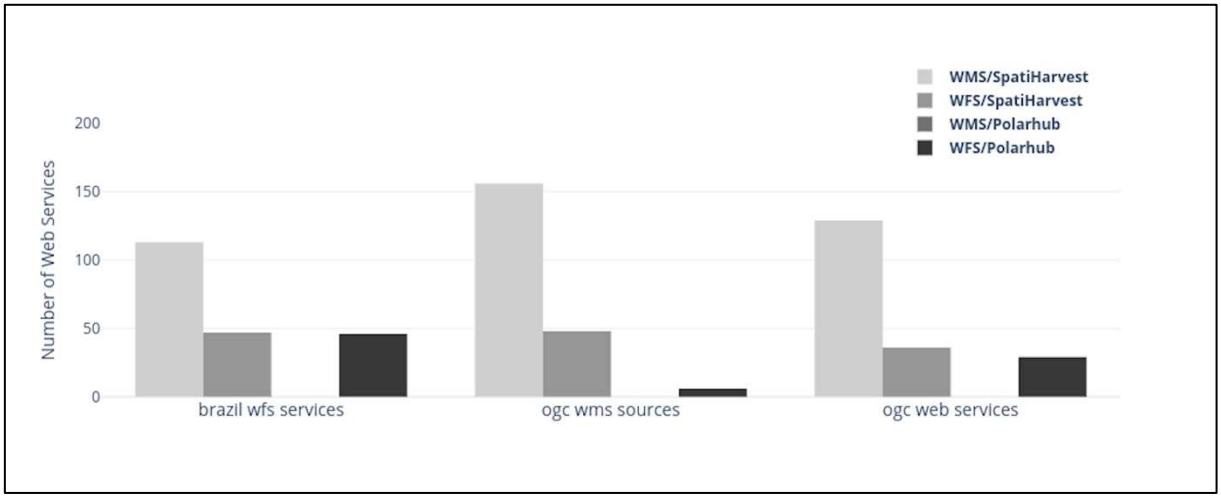
Fonte: O autor.

Tabela 3 - Results for query “ogc web services” and 96 seeds.

URLs Range	WMS – *Spati.*	WFS – *Spati.*	WMS – Polar.	WFS – Polar.
1 – 10000	118	33	0	11
10001 – 20000	1	3	0	2
20001 – 30000	3	0	0	9
30001 – 40000	3	0	0	3
40001 – 50000	4	0	0	4

Fonte: O autor.

Figura 4 - Comparison between SpatiHarvest and PolarHub with regard to the number of identified WFSes and WMSes.



Fonte: O autor.

Figure 4 shows the consolidated results comparing the two crawlers. It was verified that in terms of the number of retrieved Web Map Services, SpatiHarvest performed significantly better, even when the query was not directly related to such type of Web Service.

In terms the number of recovered Web Feature Services, SpatiHarvest also had results significantly better. However, as previously discussed, our implementation of Polarhub was unable to find Web Map Services due to the problem with its regular expression, which did not match any of the web services found.

2.1.6 Conclusion and Future Work

This paper presented a combination of Web crawling techniques used to conceive SpatiHarvest, a contribution to the area of geographic information retrieval. From the experiments that we conducted, it was possible to conclude that our

SpatiHarvest prototype significantly outperformed the state of the art, PolarHub, with regard to the recovery of Web Map Services and Web Feature Services on the Web. We highlight that our PolarHub implementation was unable to identify WMSes and, to the best of our knowledge, we followed the details presented in the PolarHub's manuscript.

There are three main reasons that explain SpatiHarvest's good performance. First, Bootstrap Focused Crawling enables the early discovery of many WMSes and WFSes scattered around the Web. Second, we used fine tuned techniques to decide whether a page was a WFSes or WMSes. Finally, the use of machine learning was a useful pruning technique to filter out pages that were not in the context of geospatial Web services.

Overall, the results show a promising path to SpatiHarvest. Despite the fact that our initial work focused on two types of OGC Web services, WFS and WMS, there are a few other types, for which the crawler could be extended to retrieve. There are also other information retrieval techniques that can further improve crawling efficiency, such as hidden web crawling techniques [2, 21], where in addition to hyperlinks contained in the pages, forms would also be explored in an attempt to find relevant content. We intend to pursue this research direction in future work.

Acknowledgements

Project partially sponsored by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and Fundação de Amparo à Pesquisa do Estado de Minas Gerais (Fapemig).

References

1. The australian government information management office (2019). URL https://www.finance.gov.au/agimo-archive/better-practice-checklists/docs/BPC17.pdf.

2. S. Raghavan, H. Garcia-Molina, in Proceedings of the 27th International Conference on Very Large Data Bases (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001), VLDB '01, pp. 129–138. URL http://dl.acm.org/citation.cfm?id= 645927.672025.

3. K. Vieira, L. Barbosa, A.S. Silva, J. Freire, E. Moura, World Wide Web 19(3), 449 (2016). https://doi.org/10.1007/s11280-015-0331-7.

4. Bergman, M. K. White paper: The deep web: Surfacing hidden value (2001). https://doi.org/10.3998/3336451.0007.104.

5. I. Cyclone Interactive Multimedia Group, I. Cyclone Interactive Multimedia Group. The digital universe of opportunities: Rich data and the increasing value of the internet of things (2019). URL https://www.emc.com/leadership/ digital-universe/2014iview/executive-summary.htm.

6. W. Li, S. Wang, V. Bhatia, Computers, Environment and Urban Systems 59, 195 (2016). https://doi.org/10.1016/j.compenvurbsys.2016.07.004.

7. R.A. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval (AddisonWesley Longman Publishing Co., Inc., Boston, MA, USA, 1999).

8. M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, B. Scholkopf, IEEE Intelligent Systems and their Applications 13(4), 18 (1998). https://doi.org/10.1109/5254.708428.

9. S. Chakrabarti, M. van den Berg, B. Dom, Computer Networks 31, 1623 (2000). https://doi.org/10.1016/S1389-1286(99)00052-3.

10. S. Batsakis, E.G. Petrakis, E. Milios, Data & Knowledge Engineering 68(10), 1001 (2009). https://doi.org/https://doi.org/10.1016/j.datak.2009.04.002.

11. C.Y. Huang, H. Chang, ISPRS International Journal of Geo-Information 5(8) (2016). https://doi.org/10.3390/ijgi5080136.

12. A. Broder, M. Mitzenmacher, Internet Mathematics 1 (2003). https://doi.org/10.1080/15427951.2004.10129096.

13. K. Rodrigues, M. Cristo, E.S. de Moura, A. da Silva, IEEE Transactions on Knowledge & Data Engineering 27(08), 2261 (2015). https://doi.org/10.1109/TKDE.2015.2407354.

14. M. Jamali, H. Sayyadi, B. Bagheri Hariri, H. Abolhassani, in Web Intelligence (2006), pp. 753–756. https://doi.org/10.1109/WI.2006.19.

15. H. Debashis, K. Amritesh, M. Lizashree, International Journal of Computer Applications 3 (2010). https://doi.org/10.5120/767-1074.

16. A. Pal, D. Singh Tomar, S. C. Shrivastava, ArXiv (2009).

17. Natural language toolkit. https://www.nltk.org/ (2019).

18. urllib - url handling modules. https://docs.python.org/3/library/urllib.html (2019).

19. scikit-learn: machine learning in python - scikit-learn 0.16.1 documentation. https://scikit-learn.org/ (2019).

20. The most popular database for modern apps. https://www.mongodb.com/ (2019).

21. L. Barbosa, J. Freire, in Proceedings of the 16th International Conference on World Wide Web (ACM, New York, NY, USA, 2007), WWW '07, pp. 441–450. https://doi.org/10.1145/1242572.1242632.

22. A. Gupta, P. Anand, in 2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE) (2015), pp. 619– 622. https://doi.org/10.1109/ABLAZE.2015.7154936.

23. J. Cho, H. Garcia-Molina, in Proceedings of the 11th International Conference on World Wide Web (ACM, New York, NY, USA, 2002), WWW '02, pp. 124–135. https://doi.org/10.1145/511446.511464.

24. A. Heydon, M. Najork, World Wide Web 2(4), 219 (1999). https://doi.org/10.1023/A:1019213109274.

25. R. Anandhi, K. Chitra, International Journal of Computer Applications 52, 12 (2012). https://doi.org/10.5120/8172-1485.

26. F. Lopez-Pellicer, A. Florczyk, R. Béjar, P. Muro-Medrano, F. Zarazaga, Online Information Review 35 (2011). https://doi.org/10.1108/14684521111193193.

3 SPATIHARVEST PORTAL

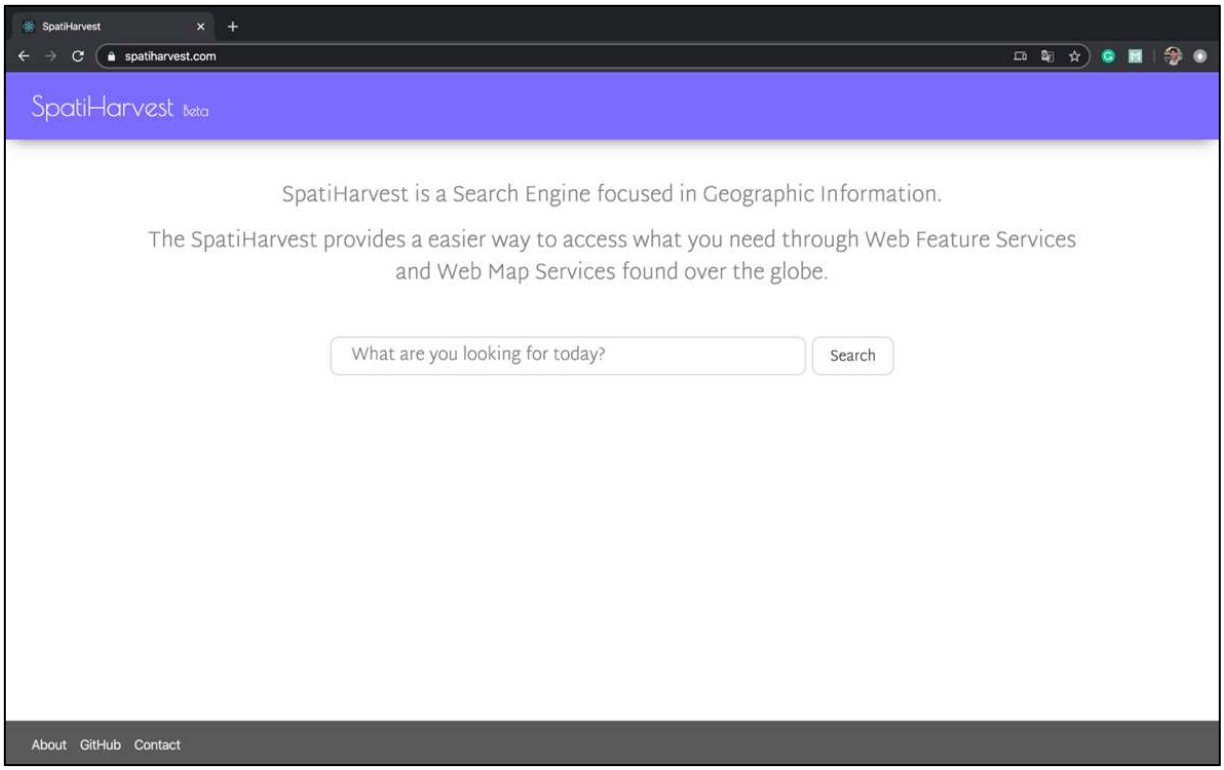
Enquanto o foco principal da pesquisa foi o desenvolvimento do motor de busca apresentado no Capítulo 2, ao final da pesquisa identificou-se a possibilidade de se criar um portal para acesso dos WMSes e WFSes previamente recuperados, isto é, os resultados encontrados pelo core da máquina, o SpatiHarvest. Nas seções seguintes são explanados detalhes como funcionalidades, arquitetura e tecnologias utilizadas para a implementação do mesmo.

3.1 Workflow

Ao iniciar o sistema SpatiHarvest Portal são feitos alguns processamentos, especialmente com a base de dados, a fim de prover performance nas atividades de busca que o portal irá servir. A primeira atividade realizada é o carregamento do banco de dados para a memória. Em seguida tal base de dados é indexada através de uma *hash table*, onde os índices são as palavras chaves presentes na descrição dos Web Services. A motivação para a escolha desta implementação se deu pelo fato de que o volume de dados não é grande o suficiente de maneira que, o custo de memória RAM fosse um problema e é trivial a vantagem, em termos de performance de se acessar a informação diretamente da memória em uma *hash table*, onde os índices são os próprios termos de busca, eliminando assim também o *overhead* de acesso a banco de dados.

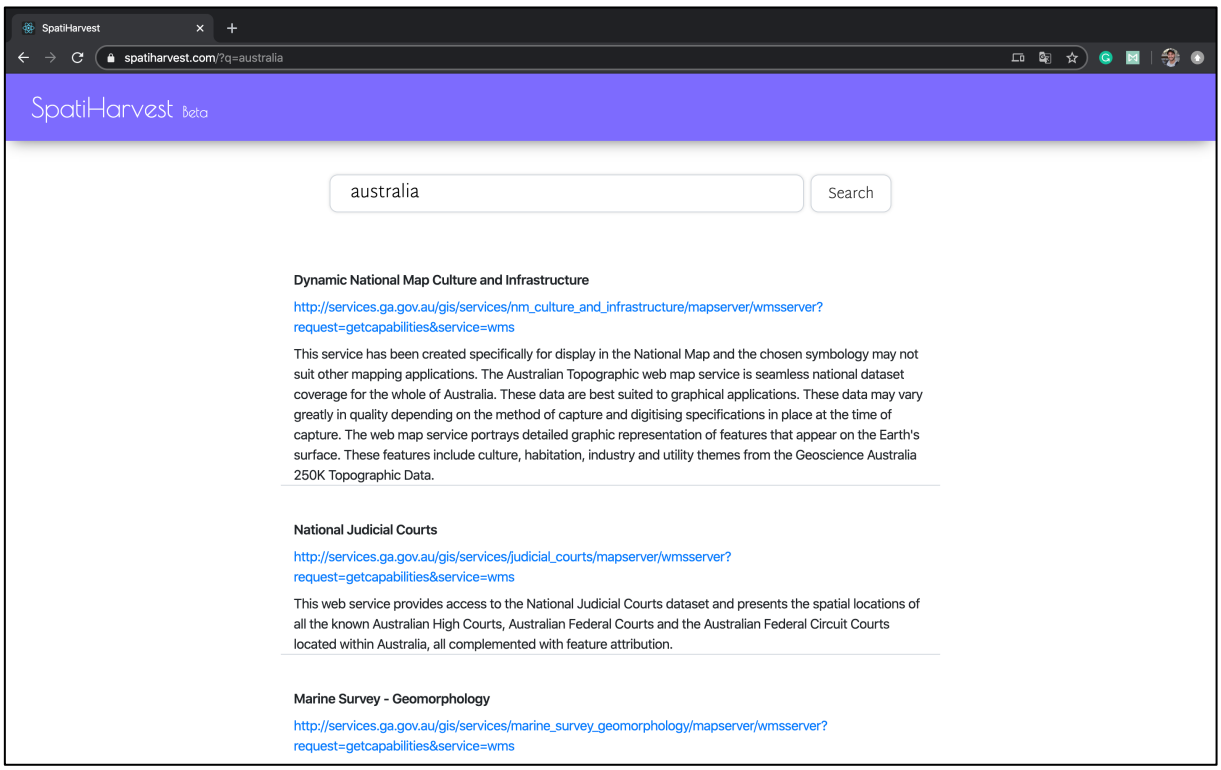
O SpatiHarvest Portal, conforme mostrado na figura 5, fornece acesso à sua base de dados através de um formulário de busca, onde o usuário pode inserir termos pelos quais gostaria de buscar *Web Services*. De posse de tais termos de busca, o SpatiHarvest Portal realiza uma busca em sua base dados a fim de encontrar resultados que possuam match com os termos de busca em sua descrição. A ordem dos resultados retornados é definida de maneira simples, para cada resultado encontrado, soma-se a frequência com que cada termo apareceu na descrição dos Web Services e, por fim, ordena-se a lista de resultados de maneira decrescente pela soma das frequências obtidas. Ao final é retornado via interface uma lista de WFSes e WMSes ao usuário (figura 6).

Figura 5 - Tela inicial do SpatiHarvest Portal.



Fonte: O autor.

Figura 6 - Tela com a lista de resultados retornada pelo SpatiHarvest Portal para a consulta “australia”.



Fonte: O autor.

3.2 Tecnologias e Arquitetura

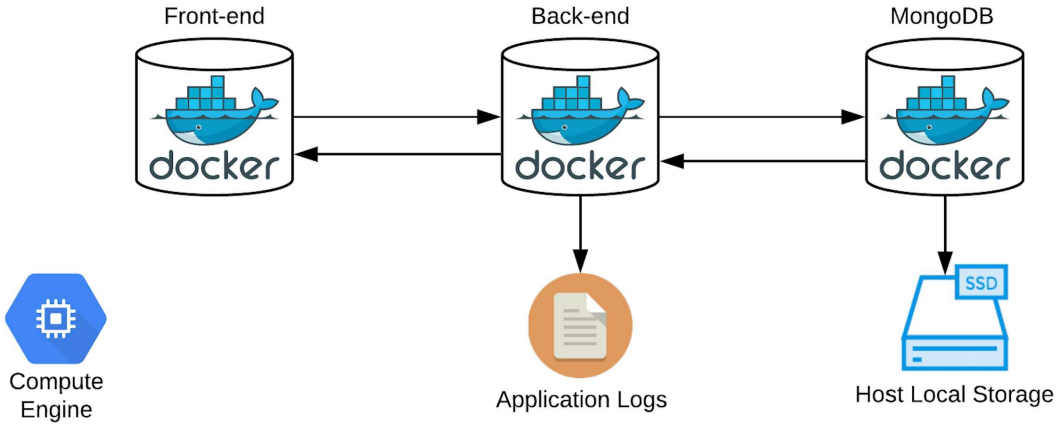
A arquitetura do SpatiHarvest Portal se divide da seguinte maneira: o sistema possui uma camada Front-end, responsável por receber consultas e apresentar os resultados ao usuário, tal camada, como pode ser vista nas figuras 5 e 6, foi construída utilizando-se JavaScript (MOZILLA , 2020c) e React.js (FACEBOOK, 2020), HTML5 (MOZILLA, 2020b) e CSS (MOZILLA, 2020a), e é servida através de containers Docker (DOCKER, 2020) numa infraestrutura Google Compute Engine.

A segunda camada do SpatiHarvest Portal é constituída de um sistema back-end, o qual é responsável por, através de APIs, receber consultas do front-end e retornar Web Services cuja descrição tenha *match* com os termos buscados. A aplicação foi construída utilizando-se JavaScript, Node.js (OPENJS FOUNDATION, 2020) e MongoDB (MONGODB, 2020). Como o front-end, é servida por meio de containers Docker numa infraestrutura Google Compute Engine.

O banco de dados utilizado pelo sistema, MongoDB, foi hospedado na mesma infraestrutura que o front-end e o back-end, também por meio de um container Docker. Para não se perder os dados dentro do container foi criado um mapeamento de volume no container para o *host*. Muitos consideram como sendo uma prática ruim a utilização de containers para bancos de dados, mas dada a natureza do sistema, que utiliza o banco apenas para leitura, sendo tal leitura somente a cada inicialização do sistema, julgou-se pertinente dadas as facilidades que um ambiente com containers proporciona.

Por se tratar de aplicações concebidas para serem isoladas por meio de containers (Figura 7), trata-se de sistemas plenamente escaláveis, podendo ser facilmente implantados em infraestruturas mais modernas como, Kubernetes (THE KUBERNETES AUTHORS, 2020). Durante o desenvolvimento foi utilizado o Google Cloud Build (GOOGLE, 2020a) para o *build* dos containers e o Google Container Registry (GOOGLE, 2020b) para o armazenamento dos mesmos.

Figura 7 - Arquitetura na perspectiva de infraestrutura do SpatiHarvest Portal.



Fonte: O autor.

4 CONCLUSÃO

Este trabalho apresentou o SpatiHarvest, uma pesquisa desenvolvida dentro do assunto de Recuperação de Informação Geográfica, subárea de Recuperação de Informação. A principal contribuição desta pesquisa foi a combinação de técnicas, gerando uma nova solução que experimentalmente se mostrou mais eficiente que o estado da arte em recuperação de *Web Feature Services* e *Web Map Services* na *Web*. Tal resultado é bastante significativo pois abre caminho para o desenvolvimento de melhores máquinas de busca focadas neste tópico, o que por fim representa um maior suporte a áreas como engenharia cartográfica, engenharia ambiental, engenharia florestal e demais áreas que fazem uso de representações computacionais de informações geográficas.

Para o desenvolvimento deste trabalho foram utilizados tanto uma fundamentação teórica de Recuperação de Informação e de Sistemas de Informação Geográfica, como tecnologias modernas que facilitaram a experimentação da metodologia proposta. A contribuição do BFC na recuperação de bons conjuntos de *seeds* para o *crawling*, a técnica de poda na árvore busca baseada na relevância de páginas e as expressões regulares construídas com base nas estruturas dos arquivos e URLs de WFSes e WMSes, foram os fatores chave nos resultados obtidos com a solução proposta, isto é, em termos de *hit rate*, o paralelismo implementado foi também fator decisivo na performance obtida durante os experimentos, isto é, em termos de tempo.

O maior desafio encontrado durante o desenvolvimento da pesquisa foi o tamanho do espaço de busca que o algoritmo encontrava pela frente nos experimentos. Apesar do fato de que o BFC provia ao *crawler* um bom conjunto de seeds, à medida que o *crawler* começava a adicionar URLs ainda não visitadas ao *frontier* muitas delas não eram necessariamente relevantes, ou sequer conduziriam o *crawler* imediatamente a outras páginas relevantes. Foi neste momento que surgiu a ideia de podar estas páginas e o desafio deste processo foi a necessidade de fazer o *tuning* do algoritmo para não podar demais, de maneira que pudesse vir a eliminar páginas relevantes ou que pudessem conter URLs de páginas relevantes. Ou podar de menos, de maneira que o espaço de busca crescesse demais, aumentando assim

o *hit rate* e o tempo de busca de modo geral. Mas à medida que foram sendo feitos os experimentos este problema foi sendo superado e chegou-se no que foi considerada uma boa poda.

Este trabalho pode ser estendido de diversas maneiras, como extensão foi produzido o SpatiHarvest Portal que, por si só pode ser melhorado, por exemplo com a adição da funcionalidade de sugestão automática de *queries* baseadas em termos já inseridos no campo de busca. Também o SpatiHarvest pode ser estendido para indexar outros tipos de *Web Services* como: *Web Coverage Service* e *Web Map Tile Service*. Também existem outras técnicas mencionadas no Capítulo 2, como *Hidden Web Crawling* que podem fazer diferença. Por fim, este é um trabalho que foi iniciado e já apresentou bons resultados experimentalmente, e além de representar um avanço teórico, tem também o potencial de impactar positivamente de uma maneira aplicada, seja para suporte em outras pesquisas ou para suporte ao desenvolvimento de produtos no mercado.

REFERÊNCIAS

AUSTRALIAN GOVERNMENT. Borehole Samples. Disponível em: <http://services.ga.gov.au/geoserver/borehole_samples/wfs?service=wfs&version=1.0.0&request=getcapabilities>. Acesso em: 24 abr. 2020b.

AUSTRALIAN GOVERNMENT. National Judicial Courts. Disponível em: <http://services.ga.gov.au/gis/services/judicial_courts/mapserver/wmsserver?request=getcapabilities&service=wms>. Acesso em: 24 abr. 2020a.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval: The Concepts and Technology behind Search**. 2. ed. USA: Addison-Wesley Publishing Company, 2011.

FACEBOOK. React – A JavaScript library for building user interfaces. Disponível em: <https://reactjs.org/>. Acesso em: 24 abr. 2020.

DOCKER. Empowering App Development for Developers | Docker. Disponível em: <https://www.docker.com/>. Acesso em: 24 abr. 2020.

GHEMAWAT, S.; GOBIOFF, H.; LEUNG, S. The Google file system. *In*: ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES (SOSP '03), 19., 2003, Bolton Landing, USA. **Proceedings…** New York, USA: Association for Computing Machinery, 2003. p. 29-43.

GOOGLE. Cloud Build | Google Cloud. Disponível em: <https://cloud.google.com/cloud-build>. Acesso em: 24 abr. 2020a.

GOOGLE. Container Registry | Google Cloud. Disponível em: < https://cloud.google.com/container-registry>. Acesso em: 24 abr. 2020b.

HUANG, C.Y.; CHANG, H. GeoWeb Crawler: An Extensible and Scalable Web Crawling Framework for Discovering Geospatial Web Resources. **ISPRS International Journal of Geo-Information**, v. 5, n. 8, ago. 2016.

LI, W.; WANG, S.; BHATIA, V. PolarHub: A large-scale web crawling engine for OGC service discovery in cyberinfrastructure. **Computers, Environment and Urban Systems**, v. 59, p. 195-207, set. 2016.

MICROSOFT. Bing. Disponível em: <https://www.bing.com>. Acesso em: 24 abr. 2017.

MONGODB. The most popular database for modern apps | MongoDB. Disponível em: <https://www.mongodb.com/>. Acesso em: 24 abr. 2020.

MOZILLA. CSS | MDN. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/CSS>. Acesso em: 24 abr. 2020a.

MOZILLA. HTML5 - HTML: Linguagem de Marcação de Hipertexto | MDN. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5>. Acesso em: 24 abr. 2020b.

MOZILLA. JavaScript | MDN. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 24 abr. 2020c.

OPEN GEOSPATIAL CONSORTIUM. Web Feature Service | OGC. Disponível em: <https://www.ogc.org/standards/wfs>. Acesso em: 24 abr. 2020a.

OPEN GEOSPATIAL CONSORTIUM. Web Map Service | OGC. Disponível em: <https://www.ogc.org/standards/wms>. Acesso em: 24 abr. 2020b.

OPENJS FOUNDATION. Node.js. Disponível em: <https://nodejs.org/en/>. Acesso em: 24 abr. 2020.

THE KUBERNETES AUTHORS. Production-Grade Container Orchestration - Kubernetes. Disponível em: <https://kubernetes.io>. Acesso em: 24 abr. 2020.

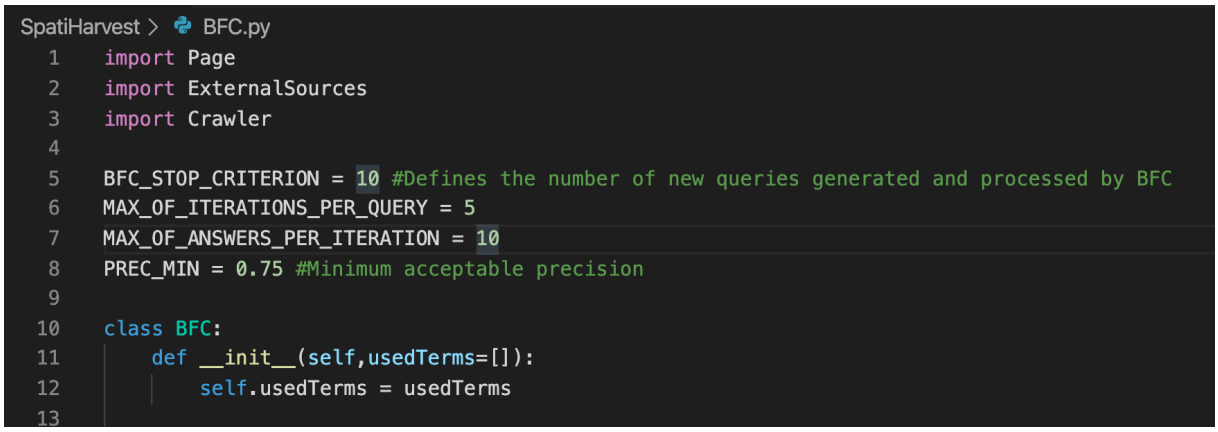
VIEIRA, K.; BARBOSA, L.; SILVA, A. S.; FREIRE, J.; MOURA, E. Finding seeds to bootstrap focused crawlers. **World Wide Web**, v. 19, p. 449–474, mai. 2016.

YAHOO!. Yahoo Brasil: Email, notícias, finanças, esportes, entretenimento. Disponível em: <https://br.yahoo.com>. Acesso em: 24 abr. 2020.

APÊNDICE A – Configuração do BFC no SpatiHarvest

A implementação do BFC possui quatro parâmetros a ser definidos, sendo eles: o critério de parada do BFC; o número máximo de iterações que o algoritmo deve fazer para cada *query* gerada; o número máximo de respostas a se recuperar para cada iteração; e a precisão mínima na classificação de uma página dada pelo SVM. O experimento obteve maior sucesso com os valores do algoritmo conforme mostra a Figura 8.

Figura 8 - Trecho do código onde são configurados os parâmetros do BFC.



Fonte: O autor.

APÊNDICE B – Setup para execução do SpatiHarvest

O SpatiHarvest, conforme descrito neste trabalho, foi construído utilizando-se Python 3.6 e o Sistema de Gerenciamento de Banco de Dados MongoDB. Abaixo são mostrados os passos para o setup e execução do sistema.

No início da pesquisa foi selecionado manualmente dois conjuntos com 50 URLs cada, sendo um conjunto composto por páginas relevantes e outro por páginas relevantes. De posse de tais conjuntos, é necessário realizar o pré-processamento das páginas. Durante o processo é feita a remoção de *stopwords* e contagem de frequência de palavras contidas nas páginas. O resultado final é a exportação de arquivos *JSON*, sendo os mesmos preenchidos com as URLs e as palavras contidas nas mesmas, juntamente com o número de repetições (Figura 9).

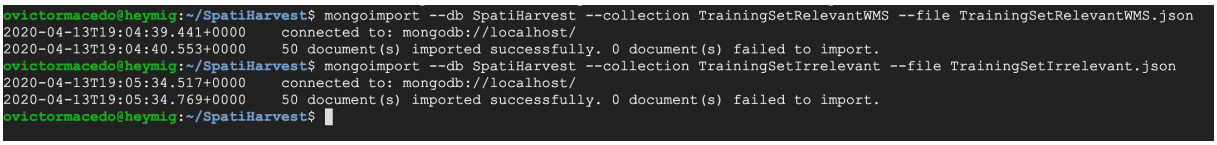
Figura 9 - Execução do script de geração de conjuntos de treinamento.

```
ovictormacedo@heymig:~/SpatiHarvest$ ll
total 2276
drwxrwxr-x  7 ovictormacedo ovictormacedo    4096 Apr 13 18:55 ./
drwxr-xr-x 13 ovictormacedo ovictormacedo    4096 Apr 13 19:39 ../
drwxrwxr-x  8 ovictormacedo ovictormacedo    4096 Apr 13 18:55 .git/
-rw-rw-r--  1 ovictormacedo ovictormacedo    3262 Apr 13 18:55 BFC.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    3849 Apr 13 18:55 Crawler.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    1784 Apr 13 18:55 CrawlerController.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    2891 Apr 13 18:55 CrawlerThreadController.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    1625 Apr 13 18:55 DAO.py
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 Docs/
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 EmptyUnvisitedPages/
-rw-rw-r--  1 ovictormacedo ovictormacedo    1156 Apr 13 18:55 ExternalSources.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    4390 Apr 13 18:55 Frontier.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     885 Apr 13 18:55 Page.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    1067 Apr 13 18:55 Parser.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     553 Apr 13 18:55 README.md
-rw-rw-r--  1 ovictormacedo ovictormacedo    6744 Apr 13 18:55 SVM.py
-rw-rw-r--  1 ovictormacedo ovictormacedo   21059 Apr 13 18:55 SVM_CreateTrainingSet.py
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 Tests/
-rw-rw-r--  1 ovictormacedo ovictormacedo 1635649 Apr 13 18:55 TrainingSetIrrelevant.json
-rw-rw-r--  1 ovictormacedo ovictormacedo  554401 Apr 13 18:55 TrainingSetRelevantWMS.json
-rw-rw-r--  1 ovictormacedo ovictormacedo    2687 Apr 13 18:55 Util.py
-rw-rw-r--  1 ovictormacedo ovictormacedo       0 Apr 13 18:55 __init__.py
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 __pycache__/
-rw-rw-r--  1 ovictormacedo ovictormacedo     674 Apr 13 18:55 main.py
-rw-rw-r--  1 ovictormacedo ovictormacedo   10690 Apr 13 18:55 mainEmail.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     746 Apr 13 18:55 monitor.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     218 Apr 13 18:55 resetDatabase.py
ovictormacedo@heymig:~/SpatiHarvest$ python SVM_CreateTrainingSet.py
```

Fonte: O autor.

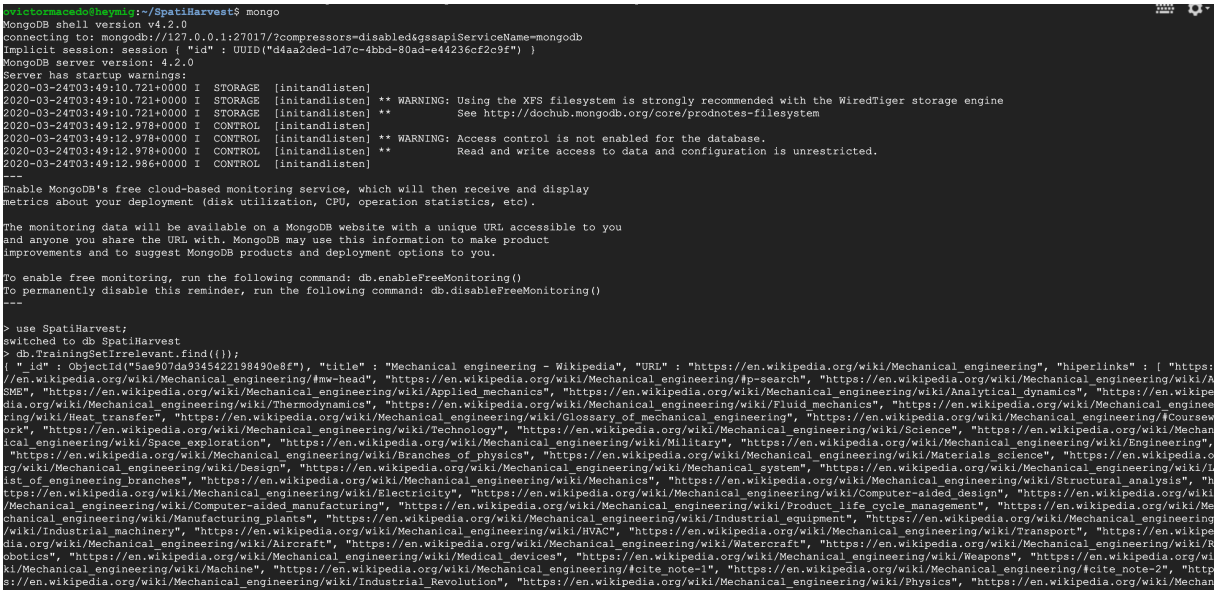
Uma vez que o MongoDB esteja instalado, basta importar os conjuntos de URLs gerados acima (Figuras 10 e 11).

Figura 10 - Inserção dos conjuntos de treinamento no banco de dados.



Fonte: O autor.

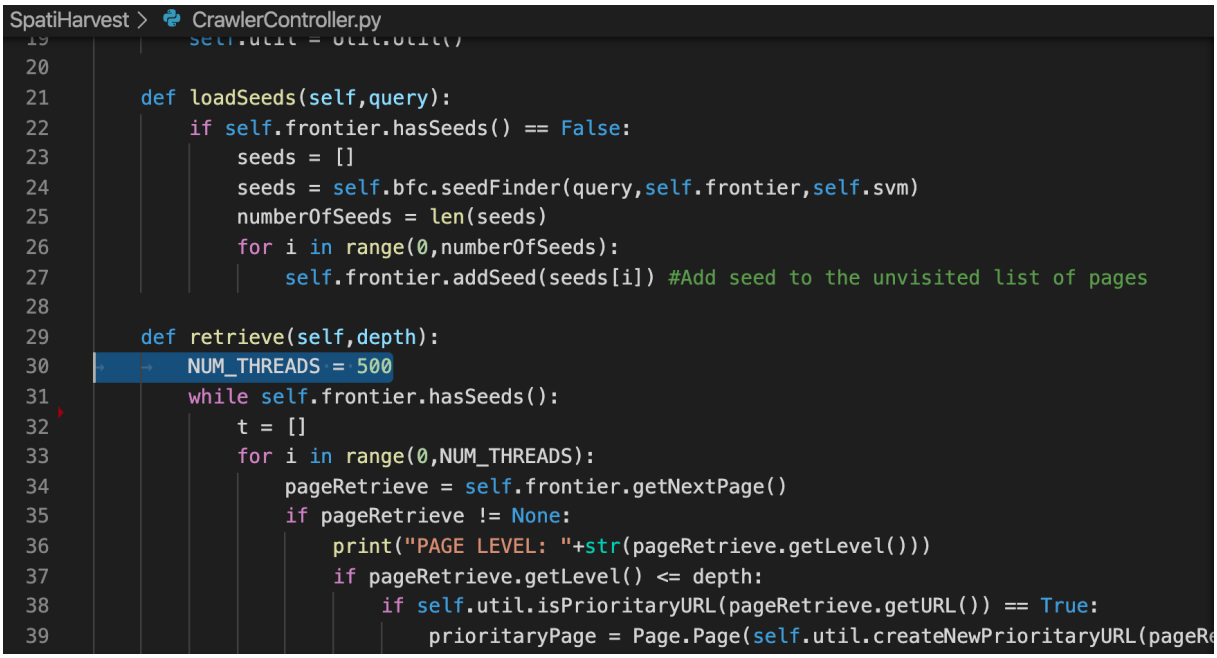
Figura 11 - Listagem de URLs irrelevantes no MongoDB.



Fonte: O autor.

Após isto é feito o *setup* do número de *threads* que o *crawler* irá utilizar. Isto é feito conforme mostra a figura 12.

Figura 12 - O número de *threads* é configurado na linha 30.



Fonte: O autor.

Com o BFC e o número de *threads* configurado, e as bases de dados carregadas, inicia-se o processo de *crawling* (Figura 13).

Figura 13 - Start do SpatiHarvest.

```
ovictormacedo@heymig:~/SpatiHarvest$ ll
total 2276
drwxrwxr-x  7 ovictormacedo ovictormacedo    4096 Apr 13 18:55 ./
drwxr-xr-x 13 ovictormacedo ovictormacedo    4096 Apr 13 19:39 ../
drwxrwxr-x  8 ovictormacedo ovictormacedo    4096 Apr 13 18:55 .git/
-rw-rw-r--  1 ovictormacedo ovictormacedo    3262 Apr 13 18:55 BFC.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    3849 Apr 13 18:55 Crawler.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    1784 Apr 13 18:55 CrawlerController.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    2891 Apr 13 18:55 CrawlerThreadController.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    1625 Apr 13 18:55 DAO.py
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 Docs/
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 EmptyUnvisitedPages/
-rw-rw-r--  1 ovictormacedo ovictormacedo    1156 Apr 13 18:55 ExternalSources.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    4390 Apr 13 18:55 Frontier.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     885 Apr 13 18:55 Page.py
-rw-rw-r--  1 ovictormacedo ovictormacedo    1067 Apr 13 18:55 Parser.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     553 Apr 13 18:55 README.md
-rw-rw-r--  1 ovictormacedo ovictormacedo    6744 Apr 13 18:55 SVM.py
-rw-rw-r--  1 ovictormacedo ovictormacedo   21059 Apr 13 18:55 SVM_CreateTrainingSet.py
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 Tests/
-rw-rw-r--  1 ovictormacedo ovictormacedo 1635649 Apr 13 18:55 TrainingSetIrrelevant.json
-rw-rw-r--  1 ovictormacedo ovictormacedo  554401 Apr 13 18:55 TrainingSetRelevantWMS.json
-rw-rw-r--  1 ovictormacedo ovictormacedo    2687 Apr 13 18:55 Util.py
-rw-rw-r--  1 ovictormacedo ovictormacedo       0 Apr 13 18:55 __init__.py
drwxrwxr-x  2 ovictormacedo ovictormacedo    4096 Apr 13 18:55 __pycache__/
-rw-rw-r--  1 ovictormacedo ovictormacedo     674 Apr 13 18:55 main.py
-rw-rw-r--  1 ovictormacedo ovictormacedo   10690 Apr 13 18:55 mainEmail.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     746 Apr 13 18:55 monitor.py
-rw-rw-r--  1 ovictormacedo ovictormacedo     218 Apr 13 18:55 resetDatabase.py
ovictormacedo@heymig:~/SpatiHarvest$ python main.py
```

Fonte: O autor.

APÊNDICE C – Setup para execução do SpatiHarvest Portal

O SpatiHarvest Portal é um sistema dividido entre Front-end e Back-end. Ambos os módulos do sistema são executados utilizando Docker. As Figuras 14 e 15 mostram exemplos de *Docker Images* utilizadas e as Figuras 16 a 19 ilustram como é feito o *build* e implantação manuais.

Figura 14 - Dockerfile do SpatiHarvest Portal - Back-end.

SpatiHarvestBackend > Dockerfile

```
1   FROM node:13.7.0-alpine3.10
2
3   WORKDIR /var/www/SpatiHarvestBackend/src
4
5   COPY . /var/SpatiHarvestBackend
6
7   RUN cp -R /var/SpatiHarvestBackend/src /var/www/SpatiHarvestBackend
8   RUN mv .env-production .env
9   RUN cp -R /var/SpatiHarvestBackend/docker_assets/ssl /etc/ssl/SpatiHarvestBackend
10  RUN npm install
11
12  EXPOSE 3000
13  EXPOSE 27017
14
15  CMD ["node", "app.js"]
```

Fonte: O autor.

Figura 15 - Dockerfile do SpatiHarvest Portal - Front-end.

SpatiHarvestUI > Dockerfile

```
1   FROM node:13.7.0-alpine3.10 as build-deps
2
3   WORKDIR /var/www/SpatiHarvestUI
4   COPY . /var/www/SpatiHarvestUI
5
6   RUN yarn
7   RUN yarn build
8
9   FROM nginx:1.12-alpine
10
11  COPY --from=build-deps /var/www/SpatiHarvestUI /var/SpatiHarvestUI
12
13  RUN cp -R /var/SpatiHarvestUI/build /usr/share/nginx/html/SpatiHarvestUI
14  RUN cp /var/SpatiHarvestUI/docker_assets/default.conf /etc/nginx/conf.d
15  RUN cp -R /var/SpatiHarvestUI/docker_assets/ssl /etc/ssl/SpatiHarvestUI
16
17  EXPOSE 443
18
19  CMD ["nginx", "-g", "daemon off;"]
20
```

Fonte: O autor.

Figura 16 - *Build* do container do SpatiHarvest Portal - Back-end.

```
spatiharvest@instance-1:~/SpatiHarvestBackend$ docker build . -t spati_backend:13042020
Sending build context to Docker daemon  265.2kB
Step 1/10 : FROM node:13.7.0-alpine3.10
 ---> 107d3c7a4fb8
Step 2/10 : WORKDIR /var/www/SpatiHarvestBackend/src
 ---> Using cache
 ---> b159ff877572
Step 3/10 : COPY . /var/SpatiHarvestBackend
 ---> 5a083cb3353e
Step 4/10 : RUN cp -R /var/SpatiHarvestBackend/src /var/www/SpatiHarvestBackend
 ---> Running in 5d46ca745ee6
Removing intermediate container 5d46ca745ee6
 ---> 112ac1b4c864
Step 5/10 : RUN mv .env-production .env
 ---> Running in cbab79b2ac4d
Removing intermediate container cbab79b2ac4d
 ---> a14790b8e295
Step 6/10 : RUN cp -R /var/SpatiHarvestBackend/docker_assets/ssl /etc/ssl/SpatiHarvestBackend
 ---> Running in 6406d22b0cbc
Removing intermediate container 6406d22b0cbc
 ---> 04e6a7e6e296
Step 7/10 : RUN npm install
 ---> Running in 7ea9cd7ea427
npm WARN spatiharvest@1.0.0 No description

added 125 packages from 91 contributors and audited 254 packages in 5.846s

1 package is looking for funding
  run `npm fund` for details

found 0 vulnerabilities

Removing intermediate container 7ea9cd7ea427
 ---> 70850cb36061
Step 8/10 : EXPOSE 3000
 ---> Running in 8d0a35edf3e0
Removing intermediate container 8d0a35edf3e0
 ---> 1cf2bb028e8d
Step 9/10 : EXPOSE 27017
 ---> Running in fda82cf3a07f
Removing intermediate container fda82cf3a07f
 ---> c3dc263b0418
Step 10/10 : CMD ["node", "app.js"]
 ---> Running in e28c40a89602
Removing intermediate container e28c40a89602
 ---> 472721e192aa
Successfully built 472721e192aa
Successfully tagged spati_backend:13042020
spatiharvest@instance-1:~/SpatiHarvestBackend$
```

Fonte: O autor.

Figura 17 - *Build* do container do SpatiHarvest Portal - Front-end.

```
spatiharvest@instance-1:~/SpatiHarvestUI$ docker build . -t spati_ui:13042020
Sending build context to Docker daemon   3.25MB
Step 1/12 : FROM node:13.7.0-alpine3.10 as build-deps
 ---> 107d3c7a4fb8
Step 2/12 : WORKDIR /var/www/SpatiHarvestUI
 ---> Using cache
 ---> 8d44bfd1c311
Step 3/12 : COPY . /var/www/SpatiHarvestUI
 ---> Using cache
 ---> f9d44c956cd6
Step 4/12 : RUN yarn
 ---> Using cache
 ---> 7407a5088798
Step 5/12 : RUN yarn build
 ---> Using cache
 ---> 8cb34d8f1e6f
Step 6/12 : FROM nginx:1.12-alpine
 ---> 24ed1c575f81
Step 7/12 : COPY --from=build-deps /var/www/SpatiHarvestUI /var/SpatiHarvestUI
 ---> Using cache
 ---> 7e50d03209b6
Step 8/12 : RUN cp -R /var/SpatiHarvestUI/build /usr/share/nginx/html/SpatiHarvestUI
 ---> Using cache
 ---> 7858db3fef78
Step 9/12 : RUN cp /var/SpatiHarvestUI/docker_assets/default.conf /etc/nginx/conf.d
 ---> Using cache
 ---> 0c6c31aadb2a
Step 10/12 : RUN cp -R /var/SpatiHarvestUI/docker_assets/ssl /etc/ssl/SpatiHarvestUI
 ---> Using cache
 ---> a8f54edaea60
Step 11/12 : EXPOSE 443
 ---> Using cache
 ---> 51eb32e1d6be
Step 12/12 : CMD ["nginx", "-g", "daemon off;"]
 ---> Using cache
 ---> f5d18d2008ee
Successfully built f5d18d2008ee
Successfully tagged spati_ui:13042020
spatiharvest@instance-1:~/SpatiHarvestUI$
```

Fonte: O autor.

Figura 18 - *Start* do container do SpatiHarvest Portal - Front-end.

```
spatiharvest@instance-1:~/SpatiHarvestUI$ docker run -d -P -p 443:443 spati_ui:13042020
701364e6ab2465696a126526161cebc6c5a7c816398386d44fa27681d68fc0f3
spatiharvest@instance-1:~/SpatiHarvestUI$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                                                  NAMES
701364e6ab24        spati_ui:13042020   "nginx -g 'daemon of…"   14 seconds ago      Up 12 seconds       0.0.0.0:443->443/tcp, 0.0.0.0:32781->80/tcp            wizardly_colden
e843f9e8f891        tutum/mongodb       "/run.sh"                11 days ago         Up 11 days          0.0.0.0:27017->27017/tcp, 0.0.0.0:32772->28017/tcp     intelligent_bell
spatiharvest@instance-1:~/SpatiHarvestUI$
```

Fonte: O autor.

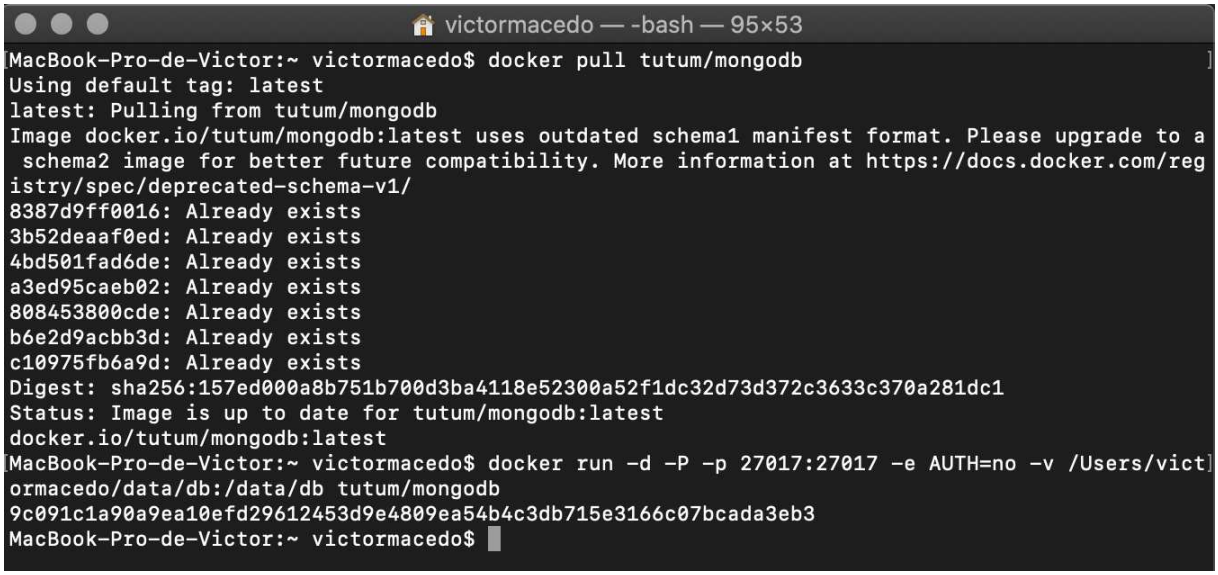
Figura 19 - *Start* do container do SpatiHarvest Portal - Back-end.

```
spatiharvest@instance-1:~/SpatiHarvestBackend$ docker run -d -P -p 3000:3000 -v /var/log:/var/log spati_backend:13042020
50d896b876752ca33fc1ff0f25daf368539546715e4d15affa3790031a2063aa
spatiharvest@instance-1:~/SpatiHarvestBackend$ docker ps
CONTAINER ID        IMAGE                    COMMAND                  CREATED             STATUS              PORTS                                                  NAMES
50d896b87675        spati_backend:13042020   "docker-entrypoint.s…"   4 seconds ago       Up 3 seconds        0.0.0.0:3000->3000/tcp, 0.0.0.0:32782->27017/tcp       quirky_bhask
701364e6ab24        spati_ui:13042020        "nginx -g 'daemon of…"   2 minutes ago       Up 2 minutes        0.0.0.0:443->443/tcp, 0.0.0.0:32781->80/tcp            wizardly_col
e843f9e8f891        tutum/mongodb            "/run.sh"                11 days ago         Up 11 days          0.0.0.0:27017->27017/tcp, 0.0.0.0:32772->28017/tcp     intelligent_
spatiharvest@instance-1:~/SpatiHarvestBackend$
```

Fonte: O autor.

Conforme mencionado anteriormente, o Sistema MongoDB também é isolado através de um *container*. Idealmente deveria existir uma rotina de ETL para automatizar o processo de carregamento das informações no banco, no entanto, dado o caráter experimental do projeto, foi feito manualmente (Figura 20).

Figura 20 - Fazendo download e executando a imagem Docker do MongoDB.



Fonte: O autor.

APÊNDICE D – Lista de URLs utilizadas para treinamento do SVM

As tabelas 4 e 5 listam os conjuntos de URLs relevantes e irrelevantes utilizados no treinamento do SVM.

Tabela 4 - URLs de páginas consideradas relevantes considerando-se o tópico de interesse.

http://docs.geoserver.org/stable/en/user/services/wms/reference.html
http://enterprise.arcgis.com/en/server/latest/publish-services/linux/communicating-with-a-wms-service-in-a-web-browser.htm
http://webhelp.esri.com/arcims/9.2/general/mergedProjects/wms_connect/wms_connector/get_capabilities.htm
http://reference1.mapinfo.com/software/spectrum/lim/8_0/developerguide/Spatial/source/Development/devguide/howdoI/wms/example_capabilities.html
http://cite.opengeospatial.org/OGCTestData/wms/1.1.1/spec/wms1.1.1.html
https://gis.stackexchange.com/questions/137282/arcgis-desktop-from-getcapabilities-to-getmap
http://ferret.pmel.noaa.gov/LAS/documentation/end-user-documentation/google-earth-products/basic-wms-requests
https://www.supermap.com/EN/online/iServer%20Java%206R/API/WMS/WMS111/GetCapabilities/getcapabilities_Request.htm
https://geoserver.geo-solutions.it/edu/en/multidim/accessing_multidim/basic_requests.html
https://www.sentinel-hub.com/develop/documentation/api/ogc_api/wms-parameters
http://www.geoportal.lt/geoportal/
http://www.onegeology.org/howto/1_4.html
http://www.neracoos.org/erddap/wms/documentation.html
http://www.unm.edu/~lijuan/GEO585L/lab05.html
http://training.gismentors.eu/isprs-summer-school-2016/lesson5/wms.html
https://trac.osgeo.org/mapserver/ticket/1262
https://nl.wikipedia.org/wiki/Web_Map_Service
http://www.opengee.org/geedocs/answer/4441137.html
http://geoserver.itc.nl/mapserver/
http://www.e-cartouche.ch/content_reg/cartouche/webservice/en/html/wms_learningObject3.html

https://dgcsportal.readme.io/v2017.2.1/docs/3-web-map-service
http://catalogue.aodn.org.au/geonetwork/srv/eng/metadata.show?uuid=1bccec28-3db2-4fb4-9dd6-ddcdafd23297
https://wiki.deegree.org/deegreeWiki/HowToUseWMSGetMapRequestsWithSLD
https://nsidc.org/data/atlas/ogc_services.html
https://www.gebco.net/data_and_products/gebco_web_services/web_map_service/
https://ca.nfis.org/cubewerx/cubeserv?SERVICE=WMS&REQUEST=GetDescription
https://www.tankonyvtar.hu/en/tartalom/tamop425/0027_ADW3/ch01s04.html
https://developer.tomtom.com/online-maps/online-maps-documentation-raster/wms
http://udig.refractions.net/files/docs/latest/user/Tracing%20WMS%20Calls.html
http://www.neracoos.org/erddap/wms/documentation.html
https://georezo.net/wiki/main/standards/wms
https://publicwiki.deltares.nl/display/OET/WMS+primer
http://oos.soest.hawaii.edu/erddap/wms/documentation.html
http://climserv.ipsl.polytechnique.fr/documentation/idl_help/Web_Map_Service_(WMS)_Overview.html
https://docs.oracle.com/cd/E29542_01/web.1111/e10145/vis_wms.htm#JIMPV9400
http://www.geog.leeds.ac.uk/courses/postgrad/web/lectures/server-side-technologies/map-servers/services.html
https://okmaps.org/ogi/WMS.aspx
https://docs.qgis.org/2.18/en/docs/user_manual/working_with_ogc/server/services.html
http://fuzzytolerance.info/blog/2012/03/06/2012-03-06-parsing-wms-getcapabilities-with-jquery/
http://lab.usgin.org/groups/best-practices-aasg-web-service-hosting/validating-your-wfs-and-wms-services
http://geo.ices.dk/message.php?title=WMS%20information&content=wms_desc.html
https://support.nearmap.com/hc/en-us/articles/226679027-GeoMedia-WMS-Integration
http://geowebcache.org/docs/1.11.0/services/wms.html
http://freegisdata.org/

https://geonetwork-opensource.org/manuals/2.10.4/eng/users/managing_metadata/harvesting/ogcwxs/index.html
https://mygeoconcept.com/doc/gcweb/docs/en/gcweb-ws-book/web-services-ogc.html
http://www.fao.org/nr/gfims/active-fire-data/web-map-services-wms/en/
http://www.marineregions.org/webservices.php
http://sccoos.org/erddap/wms/documentation.html
https://mapproxy.org/docs/nightly/services.html

Fonte: URLs retiradas da Internet e encontradas através do Google.

Tabela 5 - URLs de páginas consideradas irrelevantes considerando-se o tópico de interesse.

https://en.wikipedia.org/wiki/Mechanical_engineering
https://money.usnews.com/careers/best-jobs/mechanical-engineer
https://www.mtu.edu/mechanical/engineering/
https://www.prospects.ac.uk/job-profiles/mechanical-engineer
http://me.columbia.edu/what-mechanical-engineering
https://www.bls.gov/ooh/architecture-and-engineering/mechanical-engineers.htm
https://www.youtube.com/watch?v=ocqceS7KlzE
https://www.youtube.com/watch?v=ua6D8b5I1c4
https://www.coursera.org/browse/physical-science-and-engineering/mechanical-engineering?languages=pt
https://www.learnhowtobecome.org/mechanical-engineer/
https://en.wikipedia.org/wiki/Genetically_modified_food
https://www.scientificamerican.com/article/the-truth-about-genetically-modified-food/
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3791249/
https://www.nature.com/articles/nbt.2686
https://www.precisionnutrition.com/all-about-gm-foods
http://www.who.int/foodsafety/areas_work/food-technology/faq-genetically-modified-food/en/
https://www.nytimes.com/topic/subject/genetically-modified-food
http://learn.genetics.utah.edu/content/science/gmfoods/
https://www.huffingtonpost.com/builtlean/diet-and-nutrition_b_4323937.html
https://www.livescience.com/40895-gmo-facts.html
http://www.modifiedstreetcars.com/
https://www.motoroids.com/modified-cars/
https://www.digitaltrends.com/cars/best-tuner-cars/

https://www.skyinsurance.co.uk/modified-car-insurance.html
https://www.adrianflux.co.uk/modified/
https://www.finder.com.au/car-insurance/modified-car-insurance
https://www.gumtree.com/cars/uk/modified
https://www.jmcacademy.edu.au/courses/audio-engineering-sound-production
https://www.youtube.com/romeramirez
https://www.careersinmusic.com/music-producer/
https://en.wikipedia.org/wiki/Record_producer
https://www.waves.com/six-stages-of-music-production
http://www.cras.edu/music-production/
https://majoringinmusic.com/music-production/
https://www.youtube.com/watch?v=-gKPagTrLIA
https://audiojungle.net/?gclid=CjwKCAjwoKDXBRAAEiwA4xnqvwDF7magkiT6k-UlNmLCgVEqo2DW-ZLlPUSKy7C3HLp4yt10_oYWQRoCFoQQAvD_BwE
https://www.youtube.com/watch?v=Cv00838NOLQ
https://en.wikipedia.org/wiki/Agronomy
http://www.mdpi.com/journal/agronomy
https://www.agronomy.org/about-agronomy
http://stores.ebay.com/SICHLEY-CYCLE-PARTS
https://www.motosouls.com/
https://en.wikipedia.org/wiki/Motorcycle
https://www.ebay.com/rpp/motorcycles
http://fortune.com/2017/11/17/richest-country-in-the-world/
http://uk.businessinsider.com/the-richest-countries-in-the-world-2017-3
https://www.worldatlas.com/articles/the-richest-countries-in-the-world.html
https://en.wikipedia.org/wiki/List_of_countries_by_GDP_(PPP)_per_capita
https://www.countries-ofthe-world.com/richest-countries.html
http://www.worldsrichestcountries.com/

Fonte: URLs retiradas da Internet e encontradas através do Google.

ANEXO A – Exemplos de resultados encontrados pelo SpatiHarvest

Neste apêndice podem ser vistos dois exemplos de resultados recuperados pelo SpatiHarvest. A figura 21 mostra uma página do recurso *GetCapabilities* de um *Web Map Service*. A figura 22 mostra também uma página do recurso *GetCapabilities* de um *Web Feature Service.*

Figura 21 - Recurso *GetCapabilities* de um *Web Map Service* recuperado pelo SpatiHarvest.

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<WMS_Capabilities xmlns="http://www.opengis.net/wms" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:esri_wms="http://www.esri.com/wms" version="1.3.0"
 xsi:schemaLocation="http://www.opengis.net/wms http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd http://www.esri.com/wms
 http://services.ga.gov.au/gis/services/Judicial_Courts/MapServer/WmsServer?version=1.3.0%26service=WMS%26request=GetSchemaExtension">
  <Service>
    <Name>WMS</Name>
    <Title>National Judicial Courts</Title>
    <Abstract>
      This web service provides access to the National Judicial Courts dataset and presents the spatial locations of all the known Australian High Courts, Australian Federal Courts
      and the Australian Federal Circuit Courts located within Australia, all complemented with feature attribution.
    </Abstract>
    <KeywordList>
      <Keyword>courts</Keyword>
      <Keyword>web service</Keyword>
      <Keyword>Australia</Keyword>
    </KeywordList>
    <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="http://services.ga.gov.au/gis/services/Judicial_Courts/MapServer/WmsServer?"/>
    <ContactInformation>
      <ContactPersonPrimary>
        <ContactPerson/>
        <ContactOrganization>Geoscience Australia</ContactOrganization>
      </ContactPersonPrimary>
      <ContactPosition>Client Services</ContactPosition>
      <ContactAddress>
        <AddressType>Postal</AddressType>
        <Address>GPO Box 378</Address>
        <City>Canberra</City>
        <StateOrProvince>ACT</StateOrProvince>
        <PostCode>2601</PostCode>
        <Country>Australia</Country>
      </ContactAddress>
      <ContactVoiceTelephone>+61 2 6249 9111</ContactVoiceTelephone>
      <ContactFacsimileTelephone/>
      <ContactElectronicMailAddress>clientservices@ga.gov.au</ContactElectronicMailAddress>
    </ContactInformation>
    <Fees>none</Fees>
    <AccessConstraints>
      © Commonwealth of Australia (Geoscience Australia) 2016. This product is released under the Creative Commons Attribution 4.0 International Licence.
      http://creativecommons.org/licenses/by/4.0/legalcode
    </AccessConstraints>
    <MaxWidth>4096</MaxWidth>
    <MaxHeight>4096</MaxHeight>
  </Service>
  <Capability>
    <Request>
      <GetCapabilities>
        <Format>application/vnd.ogc.wms_xml</Format>
        <Format>text/xml</Format>
        <DCPType>
          <HTTP>
            <Get>
              <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="http://services.ga.gov.au/gis/services/Judicial_Courts/MapServer/WmsServer?"/>
```

```
      </Get>
     </HTTP>
    </DCPType>
   </GetCapabilities>
  ▼<GetMap>
     <Format>image/bmp</Format>
     <Format>image/jpeg</Format>
     <Format>image/tiff</Format>
     <Format>image/png</Format>
     <Format>image/png8</Format>
     <Format>image/png24</Format>
     <Format>image/png32</Format>
     <Format>image/gif</Format>
     <Format>image/svg+xml</Format>
    ▼<DCPType>
     ▼<HTTP>
      ▼<Get>
        <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="http://services.ga.gov.au/gis/services/Judicial_Courts/MapServer/WmsServer?"/>
       </Get>
     </HTTP>
    </DCPType>
   </GetMap>
  ▼<GetFeatureInfo>
     <Format>application/vnd.esri.wms_raw_xml</Format>
     <Format>application/vnd.esri.wms_featureinfo_xml</Format>
     <Format>application/vnd.ogc.wms_xml</Format>
     <Format>application/geojson</Format>
     <Format>text/xml</Format>
     <Format>text/html</Format>
     <Format>text/plain</Format>
    ▼<DCPType>
     ▼<HTTP>
      ▼<Get>
        <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="http://services.ga.gov.au/gis/services/Judicial_Courts/MapServer/WmsServer?"/>
       </Get>
     </HTTP>
    </DCPType>
   </GetFeatureInfo>
  ▼<esri_wms:GetStyles>
     <Format>application/vnd.ogc.sld+xml</Format>
    ▼<DCPType>
     ▼<HTTP>
      ▼<Get>
        <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="http://services.ga.gov.au/gis/services/Judicial_Courts/MapServer/WmsServer?"/>
       </Get>
     </HTTP>
    </DCPType>
   </esri_wms:GetStyles>
  </Request>
 ▼<Exception>
    <Format>application/vnd.ogc.se_xml</Format>
    <Format>application/vnd.ogc.se_inimage</Format>
    <Format>application/vnd.ogc.se_blank</Format>
    <Format>text/xml</Format>
    <Format>XML</Format>
  </Exception>
 ▼<Layer>
    <Title>National Judicial Courts</Title>
   ▼<Abstract>
     This web service provides access to the National Judicial Courts dataset and presents the spatial locations of all the known Australian High Courts, Australian Federal Courts
     and the Australian Federal Circuit Courts located within Australia, all complemented with feature attribution.
    </Abstract>
    <CRS>EPSG:4283</CRS>
    <!--  GDA94 lat-long  -->
    <CRS>EPSG:4326</CRS>
    <!--  WGS84 lat-long  -->
    <CRS>CRS:84</CRS>
    <!--  WGS84 with reversed coordinate order  -->
    <CRS>EPSG:3857</CRS>
    <!--  Web Mercator WGS84  -->
    <CRS>EPSG:28348</CRS>
    <!--  GDA94 / MGA  zone 48  -->
    <CRS>EPSG:28349</CRS>
    <!--  GDA94 / MGA  zone 49  -->
    <CRS>EPSG:28350</CRS>
    <!--  GDA94 / MGA  zone 50  -->
    <CRS>EPSG:28351</CRS>
    <!--  GDA94 / MGA  zone 51  -->
    <CRS>EPSG:28352</CRS>
    <!--  GDA94 / MGA  zone 52  -->
    <CRS>EPSG:28353</CRS>
    <!--  GDA94 / MGA  zone 53  -->
    <CRS>EPSG:28354</CRS>
    <!--  GDA94 / MGA  zone 54  -->
    <CRS>EPSG:28355</CRS>
    <!--  GDA94 / MGA  zone 55  -->
    <CRS>EPSG:28356</CRS>
    <!--  GDA94 / MGA  zone 56  -->
    <CRS>EPSG:28357</CRS>
    <!--  GDA94 / MGA zone 57  -->
    <CRS>EPSG:28358</CRS>
    <!--  GDA94 / MGA zone 58  -->
    <CRS>EPSG:32748</CRS>
    <!--  WGS84 / UTM zone 48S  -->
    <CRS>EPSG:32749</CRS>
    <!--  WGS84 / UTM zone 49S  -->
    <CRS>EPSG:32750</CRS>
    <!--  WGS84 / UTM zone 50S  -->
    <CRS>EPSG:32751</CRS>
    <!--  WGS84 / UTM zone 51S  -->
    <CRS>EPSG:32752</CRS>
    <!--  WGS84 / UTM zone 52S  -->
    <CRS>EPSG:32753</CRS>
    <!--  WGS84 / UTM zone 53S  -->
    <CRS>EPSG:32754</CRS>
    <!--  WGS84 / UTM zone 54S  -->
```

```
<CRS>EPSG:32755</CRS>
<!--  WGS84 / UTM zone 55S  -->
<CRS>EPSG:32756</CRS>
<!--  WGS84 / UTM zone 56S  -->
<CRS>EPSG:32757</CRS>
<!--  WGS84 / UTM zone 57S  -->
<CRS>EPSG:32758</CRS>
<!--  WGS84 / UTM zone 58S  -->
<CRS>EPSG:3031</CRS>
<!--  WGS84 / Antarctic Polar Stereographic  -->
<CRS>EPSG:3032</CRS>
<!--  WGS84 / Australian Antarctic Polar Stereographic  -->
<CRS>EPSG:102100</CRS>
<!--  alias 3857 (s) -->
<EX_GeographicBoundingBox>
  <westBoundLongitude>95</westBoundLongitude>
  <eastBoundLongitude>155</eastBoundLongitude>
  <southBoundLatitude>-45</southBoundLatitude>
  <northBoundLatitude>-8</northBoundLatitude>
</EX_GeographicBoundingBox>
<BoundingBox CRS="CRS:84" minx="95" miny="-45" maxx="155" maxy="-8"/>
<BoundingBox CRS="EPSG:4326" minx="-45" miny="95" maxx="-8" maxy="155"/>
<BoundingBox CRS="EPSG:4283" minx="-45" miny="95" maxx="-8" maxy="155"/>
<Layer queryable="1">
  <Name>National_Judicial_Courts</Name>
  <Title>National Judicial Courts</Title>
  <Abstract>
    The judicial courts dataset was digitized in 2012 from aerial photography, orthophotos and satellite imagery held within Geoscience Australia. Source Information: The
    primary information source used to identify and locate the judicial courts was the relevant government website: http://www.hcourt.gov.au/ http://www.fedcourt.gov.au/
    http://www.federalcircuitcourt.gov.au/
  </Abstract>
  <CRS>EPSG:4283</CRS>
  <!--  GDA94 lat-long  -->
  <CRS>EPSG:4326</CRS>
  <!--  WGS84 lat-long  -->
  <CRS>CRS:84</CRS>
  <!--  WGS84 with reversed coordinate order  -->
  <CRS>EPSG:3857</CRS>
  <!--  Web Mercator WGS84  -->
  <CRS>EPSG:28348</CRS>
  <!--  GDA94 / MGA  zone 48  -->
  <CRS>EPSG:28349</CRS>
  <!--  GDA94 / MGA  zone 49  -->
  <CRS>EPSG:28350</CRS>
  <!--  GDA94 / MGA  zone 50  -->
  <CRS>EPSG:28351</CRS>
  <!--  GDA94 / MGA  zone 51  -->
  <CRS>EPSG:28352</CRS>
  <!--  GDA94 / MGA  zone 52  -->
  <CRS>EPSG:28353</CRS>
  <!--  GDA94 / MGA  zone 53  -->
  <CRS>EPSG:28354</CRS>
  <!--  GDA94 / MGA  zone 54  -->
  <CRS>EPSG:28355</CRS>
  <!--  GDA94 / MGA  zone 55  -->
  <CRS>EPSG:28356</CRS>
  <!--  GDA94 / MGA  zone 56  -->
  <CRS>EPSG:28357</CRS>
  <!--  GDA94 / MGA zone 57  -->
  <CRS>EPSG:28358</CRS>
  <!--  GDA94 / MGA zone 58  -->
  <CRS>EPSG:32748</CRS>
  <!--  WGS84 / UTM zone 48S  -->
  <CRS>EPSG:32749</CRS>
  <!--  WGS84 / UTM zone 49S  -->
  <CRS>EPSG:32750</CRS>
  <!--  WGS84 / UTM zone 50S  -->
  <CRS>EPSG:32751</CRS>
  <!--  WGS84 / UTM zone 51S  -->
  <CRS>EPSG:32752</CRS>
  <!--  WGS84 / UTM zone 52S  -->
  <CRS>EPSG:32753</CRS>
  <!--  WGS84 / UTM zone 53S  -->
  <CRS>EPSG:32754</CRS>
  <!--  WGS84 / UTM zone 54S  -->
  <CRS>EPSG:32755</CRS>
  <!--  WGS84 / UTM zone 55S  -->
  <CRS>EPSG:32756</CRS>
  <!--  WGS84 / UTM zone 56S  -->
  <CRS>EPSG:32757</CRS>
  <!--  WGS84 / UTM zone 57S  -->
  <CRS>EPSG:32758</CRS>
  <!--  WGS84 / UTM zone 58S  -->
  <CRS>EPSG:3031</CRS>
  <!--  WGS84 / Antarctic Polar Stereographic  -->
  <CRS>EPSG:3032</CRS>
  <!--  WGS84 / Australian Antarctic Polar Stereographic  -->
  <CRS>EPSG:102100</CRS>
  <!--  alias 3857 (s) -->
  <EX_GeographicBoundingBox>
    <westBoundLongitude>95</westBoundLongitude>
    <eastBoundLongitude>155</eastBoundLongitude>
    <southBoundLatitude>-45</southBoundLatitude>
    <northBoundLatitude>-8</northBoundLatitude>
  </EX_GeographicBoundingBox>
  <BoundingBox CRS="CRS:84" minx="95" miny="-45" maxx="155" maxy="-8"/>
  <BoundingBox CRS="EPSG:4326" minx="-45" miny="95" maxx="-8" maxy="155"/>
  <BoundingBox CRS="EPSG:4283" minx="-45" miny="95" maxx="-8" maxy="155"/>
  <MetadataURL type="TC211">
    <Format>text/html</Format>
    <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:href="http://www.ga.gov.au/metadata-gateway/metadata/record/74767" xlink:type="simple"/>
  </MetadataURL>
  <Style>
    <Name>default</Name>
    <Title>National Judicial Courts</Title>
    <LegendURL width="300" height="50">
      <Format>image/png</Format>
      <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:href="http://services.ga.gov.au/Judicial_Courts/capabilities/Judicial_Courts.png" xlink:type="simple"/>
    </LegendURL>
  </Style>
</Layer>
</Layer>
</Capability>
</WMS_Capabilities>
```


Fonte: (AUSTRALIAN GOVERNMENT, 2020a)

Figura 22 - Recurso *GetCapabilities* de um *Web Feature Service* recuperado pelo SpatiHarvest.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<WFS_Capabilities xmlns="http://www.opengis.net/wfs" xmlns:borehole_samples="http://www.ga.gov.au/borehole_samples" xmlns:ogc="http://www.opengis.net/ogc" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0.0" xsi:schemaLocation="http://www.opengis.net/wfs http://services.ga.gov.au/geoserver/schemas/wfs/1.0.0/WFS-capabilities.xsd">
  <Service>
    <Name>WFS</Name>
    <Title>Borehole Samples</Title>
    <Abstract>
      This is an Open Geospatial Consortium (OGC) web feature service (WFS) providing access to a subset of Australian sample data held by Geoscience Australia. The subset currently relates specifically to Australian Boreholes.
    </Abstract>
    <Keywords>Borehole, Sample, web service</Keywords>
    <OnlineResource>
      http://services.ga.gov.au/geoserver/borehole_samples/wfs
    </OnlineResource>
    <Fees>NONE</Fees>
    <AccessConstraints>
      © Commonwealth of Australia (Geoscience Australia) 2016. This product is released under the Creative Commons Attribution 4.0 International Licence.
      http://creativecommons.org/licenses/by/4.0/legalcode
    </AccessConstraints>
  </Service>
  <Capability>
    <Request>
      <GetCapabilities>
        <DCPType>
          <HTTP>
            <Get onlineResource="http://services.ga.gov.au/geoserver/borehole_samples/wfs?request=GetCapabilities"/>
          </HTTP>
        </DCPType>
        <DCPType>
          <HTTP>
            <Post onlineResource="http://services.ga.gov.au/geoserver/borehole_samples/wfs"/>
          </HTTP>
        </DCPType>
      </GetCapabilities>
      <DescribeFeatureType>
        <SchemaDescriptionLanguage>
          <XMLSCHEMA/>
        </SchemaDescriptionLanguage>
        <DCPType>
          <HTTP>
            <Get onlineResource="http://services.ga.gov.au/geoserver/borehole_samples/wfs?request=DescribeFeatureType"/>
          </HTTP>
        </DCPType>
        <DCPType>
          <HTTP>
            <Post onlineResource="http://services.ga.gov.au/geoserver/borehole_samples/wfs"/>
          </HTTP>
        </DCPType>
      </DescribeFeatureType>
      <GetFeature>
        <ResultFormat>
          <KML/>
          <GML2/>
          <GML3/>
          <SHAPE-ZIP/>
          <CSV/>
          <JSON/>
        </ResultFormat>
        <DCPType>
          <HTTP>
            <Get onlineResource="http://services.ga.gov.au/geoserver/borehole_samples/wfs?request=GetFeature"/>
          </HTTP>
        </DCPType>
        <DCPType>
          <HTTP>
            <Post onlineResource="http://services.ga.gov.au/geoserver/borehole_samples/wfs"/>
          </HTTP>
        </DCPType>
      </GetFeature>
    </Request>
  </Capability>
  <FeatureTypeList>
    <Operations>
      <Query/>
    </Operations>
    <FeatureType>
      <Name>borehole_samples:BoreholeSamples</Name>
      <Title>Borehole Samples</Title>
      <Abstract>
        This is an Open Geospatial Consortium (OGC) web service providing access to a subset of Australian sample data held by Geoscience Australia. The subset currently relates specifically to Australian Boreholes.
      </Abstract>
      <Keywords>Sample, Borehole, web service</Keywords>
      <SRS>EPSG:4283</SRS>
      <LatLongBoundingBox minx="93.41" miny="-60.5599999991932" maxx="173.4" maxy="-8.469999999724733"/>
    </FeatureType>
  </FeatureTypeList>
  <ogc:Filter_Capabilities>
    <ogc:Spatial_Capabilities>
      <ogc:Spatial_Operators>
        <ogc:Disjoint/>
        <ogc:Equals/>
        <ogc:DWithin/>
        <ogc:Beyond/>
        <ogc:Intersect/>
        <ogc:Touches/>
        <ogc:Crosses/>
        <ogc:Within/>
        <ogc:Contains/>
        <ogc:Overlaps/>
        <ogc:BBOX/>
      </ogc:Spatial_Operators>
    </ogc:Spatial_Capabilities>
    <ogc:Scalar_Capabilities>
      <ogc:Logical_Operators/>
      <ogc:Comparison_Operators>
        <ogc:Simple_Comparisons/>
        <ogc:Between/>
        <ogc:Like/>
        <ogc:NullCheck/>
      </ogc:Comparison_Operators>
      <ogc:Arithmetic_Operators>
        <ogc:Simple_Arithmetic/>
```

```
<ogc:Simple_Arithmetic/>
<ogc:Functions>
<ogc:Function_Names>
<ogc:Function_Name nArgs="1">abs</ogc:Function_Name>
<ogc:Function_Name nArgs="1">abs_2</ogc:Function_Name>
<ogc:Function_Name nArgs="1">abs_3</ogc:Function_Name>
<ogc:Function_Name nArgs="1">abs_4</ogc:Function_Name>
<ogc:Function_Name nArgs="1">acos</ogc:Function_Name>
<ogc:Function_Name nArgs="2">AddCoverages</ogc:Function_Name>
<ogc:Function_Name nArgs="-1">Affine</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">Aggregate</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Area</ogc:Function_Name>
<ogc:Function_Name nArgs="1">area2</ogc:Function_Name>
<ogc:Function_Name nArgs="3">AreaGrid</ogc:Function_Name>
<ogc:Function_Name nArgs="1">asin</ogc:Function_Name>
<ogc:Function_Name nArgs="1">asMultiGeometry</ogc:Function_Name>
<ogc:Function_Name nArgs="1">atan</ogc:Function_Name>
<ogc:Function_Name nArgs="2">atan2</ogc:Function_Name>
<ogc:Function_Name nArgs="-1">BandMerge</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">BandSelect</ogc:Function_Name>
<ogc:Function_Name nArgs="-6">BarnesSurface</ogc:Function_Name>
<ogc:Function_Name nArgs="3">between</ogc:Function_Name>
<ogc:Function_Name nArgs="1">boundary</ogc:Function_Name>
<ogc:Function_Name nArgs="1">boundaryDimension</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Bounds</ogc:Function_Name>
<ogc:Function_Name nArgs="2">buffer</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">BufferFeatureCollection</ogc:Function_Name>
<ogc:Function_Name nArgs="3">bufferWithSegments</ogc:Function_Name>
<ogc:Function_Name nArgs="7">Categorize</ogc:Function_Name>
<ogc:Function_Name nArgs="1">ceil</ogc:Function_Name>
<ogc:Function_Name nArgs="1">centroid</ogc:Function_Name>
<ogc:Function_Name nArgs="2">classify</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">Clip</ogc:Function_Name>
<ogc:Function_Name nArgs="1">CollectGeometries</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Average</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Bounds</ogc:Function_Name>
<ogc:Function_Name nArgs="0">Collection_Count</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Max</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Median</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Min</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Nearest</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Sum</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Collection_Unique</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">Concatenate</ogc:Function_Name>
<ogc:Function_Name nArgs="2">contains</ogc:Function_Name>
<ogc:Function_Name nArgs="-1">Contour</ogc:Function_Name>
<ogc:Function_Name nArgs="2">convert</ogc:Function_Name>
<ogc:Function_Name nArgs="1">convexHull</ogc:Function_Name>
<ogc:Function_Name nArgs="1">cos</ogc:Function_Name>
<ogc:Function_Name nArgs="1">Count</ogc:Function_Name>
<ogc:Function_Name nArgs="-1">CoverageClassStats</ogc:Function_Name>
<ogc:Function_Name nArgs="2">CropCoverage</ogc:Function_Name>
<ogc:Function_Name nArgs="2">crosses</ogc:Function_Name>
<ogc:Function_Name nArgs="2">dateFormat</ogc:Function_Name>
<ogc:Function_Name nArgs="2">dateParse</ogc:Function_Name>
<ogc:Function_Name nArgs="2">difference</ogc:Function_Name>
<ogc:Function_Name nArgs="1">dimension</ogc:Function_Name>
<ogc:Function_Name nArgs="2">disjoint</ogc:Function_Name>
<ogc:Function_Name nArgs="2">disjoint3D</ogc:Function_Name>
<ogc:Function_Name nArgs="2">distance</ogc:Function_Name>
<ogc:Function_Name nArgs="2">distance3D</ogc:Function_Name>
<ogc:Function_Name nArgs="1">double2bool</ogc:Function_Name>
<ogc:Function_Name nArgs="1">endAngle</ogc:Function_Name>
<ogc:Function_Name nArgs="1">endPoint</ogc:Function_Name>
<ogc:Function_Name nArgs="1">env</ogc:Function_Name>
<ogc:Function_Name nArgs="1">envelope</ogc:Function_Name>
<ogc:Function_Name nArgs="2">EqualInterval</ogc:Function_Name>
<ogc:Function_Name nArgs="2">equalsExact</ogc:Function_Name>
<ogc:Function_Name nArgs="3">equalsExactTolerance</ogc:Function_Name>
<ogc:Function_Name nArgs="2">equalTo</ogc:Function_Name>
<ogc:Function_Name nArgs="1">exp</ogc:Function_Name>
<ogc:Function_Name nArgs="1">exteriorRing</ogc:Function_Name>
<ogc:Function_Name nArgs="3">Feature</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">FeatureClassStats</ogc:Function_Name>
<ogc:Function_Name nArgs="1">floor</ogc:Function_Name>
<ogc:Function_Name nArgs="3">FormatDateTimezone</ogc:Function_Name>
<ogc:Function_Name nArgs="0">geometry</ogc:Function_Name>
<ogc:Function_Name nArgs="1">geometryType</ogc:Function_Name>
<ogc:Function_Name nArgs="1">geomFromWKT</ogc:Function_Name>
<ogc:Function_Name nArgs="1">geomLength</ogc:Function_Name>
<ogc:Function_Name nArgs="2">getGeometryN</ogc:Function_Name>
<ogc:Function_Name nArgs="0">getID</ogc:Function_Name>
<ogc:Function_Name nArgs="1">getX</ogc:Function_Name>
<ogc:Function_Name nArgs="1">getY</ogc:Function_Name>
<ogc:Function_Name nArgs="1">getz</ogc:Function_Name>
<ogc:Function_Name nArgs="2">greaterEqualThan</ogc:Function_Name>
<ogc:Function_Name nArgs="2">greaterThan</ogc:Function_Name>
<ogc:Function_Name nArgs="-3">Grid</ogc:Function_Name>
<ogc:Function_Name nArgs="-5">Heatmap</ogc:Function_Name>
<ogc:Function_Name nArgs="0">id</ogc:Function_Name>
<ogc:Function_Name nArgs="2">IEEEremainder</ogc:Function_Name>
<ogc:Function_Name nArgs="3">if_then_else</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">in</ogc:Function_Name>
<ogc:Function_Name nArgs="11">in10</ogc:Function_Name>
<ogc:Function_Name nArgs="3">in2</ogc:Function_Name>
<ogc:Function_Name nArgs="4">in3</ogc:Function_Name>
<ogc:Function_Name nArgs="5">in4</ogc:Function_Name>
<ogc:Function_Name nArgs="6">in5</ogc:Function_Name>
<ogc:Function_Name nArgs="7">in6</ogc:Function_Name>
<ogc:Function_Name nArgs="8">in7</ogc:Function_Name>
<ogc:Function_Name nArgs="9">in8</ogc:Function_Name>
<ogc:Function_Name nArgs="10">in9</ogc:Function_Name>
<ogc:Function_Name nArgs="2">InclusionFeatureCollection</ogc:Function_Name>
<ogc:Function_Name nArgs="1">int2bbool</ogc:Function_Name>
<ogc:Function_Name nArgs="1">int2ddouble</ogc:Function_Name>
<ogc:Function_Name nArgs="1">interiorPoint</ogc:Function_Name>
<ogc:Function_Name nArgs="2">interiorRingN</ogc:Function_Name>
<ogc:Function_Name nArgs="-5">Interpolate</ogc:Function_Name>
<ogc:Function_Name nArgs="2">intersection</ogc:Function_Name>
<ogc:Function_Name nArgs="-2">IntersectionFeatureCollection</ogc:Function_Name>
<ogc:Function_Name nArgs="2">intersects</ogc:Function_Name>
<ogc:Function_Name nArgs="2">intersects3D</ogc:Function_Name>
<ogc:Function_Name nArgs="1">isClosed</ogc:Function_Name>
<ogc:Function_Name nArgs="0">isCoverage</ogc:Function_Name>
```

```
            <ogc:Function_Name nArgs="1">isEmpty</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">isInstanceOf</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">isLike</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">isNull</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">isometric</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">isRing</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">isSimple</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">isValid</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">isWithinDistance</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">isWithinDistance3D</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">Jenks</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">length</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">lessEqualThan</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">lessThan</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">list</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">log</ogc:Function_Name>
            <ogc:Function_Name nArgs="4">LRSGeocode</ogc:Function_Name>
            <ogc:Function_Name nArgs="-4">LRSMeasure</ogc:Function_Name>
            <ogc:Function_Name nArgs="5">LRSSegment</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">max</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">max_2</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">max_3</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">max_4</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">min</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">min_2</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">min_3</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">min_4</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">mincircle</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">minimumdiameter</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">minrectangle</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">modulo</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">MultiplyCoverages</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">Nearest</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">not</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">notEqualTo</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">numberFormat</ogc:Function_Name>
            <ogc:Function_Name nArgs="5">numberFormat2</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">numGeometries</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">numInteriorRing</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">numPoints</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">octagonalenvelope</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">offset</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">overlaps</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">parameter</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">parseBoolean</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">parseDouble</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">parseInt</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">parseLong</ogc:Function_Name>
            <ogc:Function_Name nArgs="0">pi</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">PointBuffers</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">pointN</ogc:Function_Name>
            <ogc:Function_Name nArgs="-6">PointStacker</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">PolygonExtraction</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">pow</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">property</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">PropertyExists</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">Quantile</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">Query</ogc:Function_Name>
            <ogc:Function_Name nArgs="0">random</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">RangeLookup</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">RasterAsPointCollection</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">RasterZonalStatistics</ogc:Function_Name>
            <ogc:Function_Name nArgs="-6">RasterZonalStatistics2</ogc:Function_Name>
            <ogc:Function_Name nArgs="5">Recode</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">RectangularClip</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">relate</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">relatePattern</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">Reproject</ogc:Function_Name>
            <ogc:Function_Name nArgs="-3">rescaleToPixels</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">rint</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">round</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">round_2</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">roundDouble</ogc:Function_Name>
            <ogc:Function_Name nArgs="-5">ScaleCoverage</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">sdo_nn</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">setCRS</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">simplify</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">sin</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">Snap</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">sqrt</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">StandardDeviation</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">startAngle</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">startPoint</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">strCapitalize</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strConcat</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strEndsWith</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strEqualsIgnoreCase</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strIndexOf</ogc:Function_Name>
            <ogc:Function_Name nArgs="4">stringTemplate</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strLastIndexOf</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">strLength</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strMatches</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">strPosition</ogc:Function_Name>
            <ogc:Function_Name nArgs="4">strReplace</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strStartsWith</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">strSubstring</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">strSubstringStart</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">strToLowerCase</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">strToUpperCase</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">strTrim</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">strTrim2</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">StyleCoverage</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">symDifference</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">tan</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">toDegrees</ogc:Function_Name>
            <ogc:Function_Name nArgs="-1">toDirectPosition</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">ToEnvelope</ogc:Function_Name>
            <ogc:Function_Name nArgs="-3">toLineString</ogc:Function_Name>
            <ogc:Function_Name nArgs="-2">toPoint</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">toRadians</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">touches</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">toWKT</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">toXlinkHref</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">Transform</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">union</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">UnionFeatureCollection</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">Unique</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">UniqueInterval</ogc:Function_Name>
            <ogc:Function_Name nArgs="-4">VectorToRaster</ogc:Function_Name>
            <ogc:Function_Name nArgs="3">VectorZonalStatistics</ogc:Function_Name>
            <ogc:Function_Name nArgs="1">vertices</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">Vocab</ogc:Function_Name>
            <ogc:Function_Name nArgs="2">within</ogc:Function_Name>
          </ogc:Function_Names>
        </ogc:Functions>
      </ogc:Arithmetic_Operators>
    </ogc:Scalar_Capabilities>
  </ogc:Filter_Capabilities>
</WFS_Capabilities>
```

Fonte: (AUSTRALIAN GOVERNMENT, 2020b)