HULTIG-C: NLP Corpus and Services in the Cloud

Sebastião Pais (sebastiao@di.ubi.pt)
Universidade da Beira Interior https://orcid.org/0000-0003-2337-0779
João Cordeiro
University of Beira Interior
Muhammad Jamil
University of Beira Interior

Research Article

Keywords: Natural Language Processing, Multilingual Corpus, Annotated Corpora, Corpus Construction

Posted Date: January 10th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-696114/v2

HULTIG-C: NLP Corpus and Services in the Cloud

Sebastião Pais[1,2,3*†], João Cordeiro[1†] and Muhammad Jamil[1†]

[1*]Department of Computer Science, University of Beira Interior, Portugal.
[2]NOVA Laboratory for Computer Science and Informatics (NOVA LINCS, Portugal.
[3]GREYC, Groupe de Recherche en Informatique, Image et Instrumentation de University of Caen Normandie, Caen, France.

*Corresponding author(s). E-mail(s): sebastiao@di.ubi.pt;
Contributing authors: jpaulo@di.ubi.pt; luqman.jamil@ubi.pt;
†These authors contributed equally to this work.

Abstract

Nowadays, the use of language corpora for many purposes has increased significantly. General corpora exist for numerous languages, but research often needs more specialized corpora. The Web's rapid growth has significantly improved access to thousands of online documents, highly specialized texts and comparable texts on the same subject covering several languages in electronic form. However, research has continued to concentrate on corpus annotation instead of corpus creation tools. Consequently, many researchers create their corpora, independently solve problems, and generate project-specific systems. The corpus construction is used for many NLP applications, including machine translation, information retrieval, and question-answering. This paper presents a new NLP Corpus and Services in the Cloud called HULTIG-C. HULTIG-C is characterized by various languages that include unique annotations such as keywords set, sentences set, named entity recognition set, and multiword set. Moreover, a framework incorporates the main components for license detection, language identification, boilerplate removal and document deduplication to process the HULTIG-C. Furthermore, this paper presents some potential issues related to constructing multilingual corpora from the Web.

Keywords: Natural Language Processing, Multilingual Corpus, Annotated Corpora, Corpus Construction

1 Introduction

The Web plays a vital role in the search for information these days. It contains billions of text words that can be used for any linguistic research [1]. The Web has become a great source of documents in various languages and different areas. It is also an endless source of textual, graphical, and sound data. These resources enable the rapid construction of the corpus.
Moreover, the explosion of online information has made it easier to create a corpus for a specific purpose by accessing relevant texts from the internet. Web corpus refers to a collection of texts obtained and processed from the Web into a static corpus [2]. A corpus can comprise written texts, generally spoken language texts, or represent only a specific genre or variety of languages. The corpus building is significant and helpful in advancing different language applications such as machine translation, information retrieval, and question-answering. One of the most exciting and intrinsic features of the Web is that it is multilingual.

Building a text corpus from the Web is a difficult task because it requires an ample amount of resources to be used.
Consequently, there are three different approaches to collecting large web corpora. The first approach is classic using crawlers; this approach crawls every resource that addresses search engines, walks through the pages, and clears the material from irrelevant information. This method does not enable the total amount of metatext data to be saved since all the boilerplate is removed, but it allows a lot to be collected in a short time, for instance, in a standard crawl. The second type is the right approach. In this approach, all of the content is crawled from the thousands of URLs listed. Often a fit function is used to decode URLs that are appropriate or not, while the crawler addresses the search engine for that purpose, similar to the first approach. The third type is the differential approach, where a limited number of significant resources are crawled but downloaded to the fullest extent possible. Ultimately, This download enables the linguistic variation of resources to be covered wholly.
This paper presents a Natural Language Processing (NLP) Corpus and Services in the Cloud called HULTIG-C – a framework for constructing multilingual corpora from the Web that consists of some of the NLP Linguistic annotations. HULTIG-C operates as a platform cloud-based service. It provides researchers and users with scalable state of the art NLP tools. Moreover, users can utilize these available resources in multiple languages and various domains on-demand. Cloud computing is a model that allows flexible on-demand network access to a shared pool of configurable computing resources that can be accessed and delivered quickly with minimal management effort. This paradigm is based on many existing technologies such as the internet, virtualization, grid computing, and web services. Hence, Cloud Computing is the combination of software as service and utility computing. Cloud computing is designed to provide flexible and low-cost on-demand computing infrastructure with good service reliability.
Cloud computing is an enticing paradigm that has many advantages, such as:
Reduced Cost: Cost containment is a clear benefit of cloud computing, both in capital and operational expenses. Reducing capital expenditure is evident because an organization can spend in increments of required capacity and does not need to build infrastructure for maximum or overflow capacity. Enterprises can utilize a cloud provider or adopt cloud paradigms internally to save operational and maintenance expenses;
Improved Automation: Cloud computing is based on the premise that services are provisioned and unprovisioned in a highly automated fashion. This attribute offers significant efficiency to enterprises;
Flexibility: Flexibility benefits derive from rapid provisioning of new capacity and rapid relocation or migration of workloads. For example, in public sector settings, cloud computing provides agility in terms of procurement and acquisition process and timelines;
Sustainability: Through leveraging economies of scale and the capacity to manage assets more efficiently, cloud computing consumes far less energy and other resources than a traditional data centre. The low energy efficiency of most existing data centres due to poor design or poor asset utilization are environmentally and economically unsustainable in the long run.
Cloud services for NLP analysis are becoming very popular among scientists and all users interested in the field. There are few reviews available about the services for NLP. Examples of the most prominent natural language processing APIs and cloud-based services are Amazon Comprehend, Microsoft Azure Cognitive Services, Google Cloud Natural Language.
Cloud computing can provide a big processing capacity for data that require scalability, easiness-to-growth, fault tolerance, and hardware virtualization. Big Data and cloud computing are two complementary concepts: Cloud can make Big Data available, scalable, and fault-tolerant.
Big Data is about storing vast amounts of data and ways of processing and extracting knowledge from it. In practice, a Big Data database can contain structured and unstructured data that can overlap, vary, and have different volumes at different speeds. Big Data attributes vary in five dimensions from other data: volume, velocity, variety, value, and complexity.
Volume: concerns the vast loads that typically Big Data has to deal with in processing, and to store large volumes of data is somewhat tricky as it

involves (among others): scalability (vertical, horizontal or both) to promote the growth of storage and processing power; Flexibility that guarantees access to data and ways of conducting data-related operations; bandwidth and efficiency that ensure access to data at the right time; **Velocity**: concerns the different rates of each source of data. For example, an Enterprise Data Warehouse (EDW)is usually updated once a day, while information is continuously updated from wireless sensor systems. To aggregate data from several data sources, Big Data must handle with which data arrives at different velocities; **variety**: concerns the various data types from different sources that data frameworks have to deal with(typically, different sources output different kinds of data) while digging. Big Data is a way to overcome these differences and unify data. Internet of Things is a Big Data related topic that examines data from individual objects of everyday life that can be varied from Internet traffic, smartphones, wearable technology, and others. For the processing of different types of data, Big Data must provide data-type abstraction frameworks; **Value**: refers to the actual value of the data (i.e., the potential value of the data regarding the information they contain). Enormous amounts of data are worthless unless they give value to who is discovering it; **Veracity**: refers to the trustfulness of the data (i.e., addressing data confidentiality, integrity, and availability). Data is meaningless or even misleading if its source is unreliable. Therefore, organizations need to ensure that the data is accurate and that the analyses carried out on the data are correct.

There is excellent progress in delivering technologies in NLP, such as extracting information from big unstructured data on the Web, sentiment analysis in social networks or grammatical analysis for essay grading. Big Data tools such as Hadoop or High-Performance Cloud Computing (HPCC) are used to gather relevant data. A MapReduce program such as Hadoop is used to map and sort the information into categories and 'reduce' it by conducting a summary function. In NLP, programs are used to look for specific pieces of information that are stored in map-reduce.

Hadoop is an open-source implementation of MapReduce. One of the essential characteristics of Hadoop is that of being oriented to perform batch processing, but it leads to severe problems when using it for real-time streaming processing systems. Apache SPARK overcomes this problem by extending Hadoop with new workloads like streaming, interactive queries and learning algorithms. In any case, using Hadoop or SPARK frameworks require re-implementing the NLP algorithms using a programming language from the MapReduce family.

This paper is structured as follows: Section II describes the types of corpora, Section III discusses the challenge of constructing Multilingual corpora from the Web, Section IV presents related work on collecting a corpus from the Web, Section V discusses the outline of the framework for building multilingual web corpora (HULTIG-C); and finally, the conclusion for future work is drawn in Section VI.

2 Types of Corpora

While it is true that every corpus has its unique purpose and identity, it is also true that corpora tend to be classified in diverse categories depending on their purpose and characteristics. The categorization of corpora can be divided into three different benchmarks based on the data they contain: general and specialized, synchronic and diachronic, and monolingual and multilingual.

General and Specialized: A general corpus is a corpus that contains texts from various fields, genres, and registers. This type of corpora produces reference materials for language learning and translation, such as grammar books or dictionaries. As they contain widely distinct data. Examples of general corpora are the first computer corpus, the Brown Corpus and the British National Corpus (BNC). On the other hand, unlike a general corpus, a specific corpus will forfeit its broadness and only represent a particular field, genre, time, or variety of language. Due to their specificity being shorter than general corpora, but they are also less ambiguous and easier to use and study. Examples of specific corpora are the Corpus of Early English Medical Writing (CEEM) or the Air Traffic Control Speech Corpus (ATCOSIM).

Synchronic and Diachronic: Corpora are also distinguished between synchronic and diachronic corpora. A synchronic corpus contains data from

a certain limited period, trying to portray a language characteristic in a given period. Some examples of synchronic corpora are the Helsinki Corpus of English Texts or the Corpus of Contemporary American English (COCA). Diachronic corpora try to seek the evolution of a language; therefore, they include texts from all ages and periods inside this categorization of corpora. An Example of the diachronic corpora is monitor corpora (MC). The objective of this kind of corpora is to monitor the changes in the language. New texts are continuously being added to update it, so it is constantly growing. The Bank of English (BoE) is an example of monitor corpora, constantly updated with contemporary texts.

Monolingual and Multilingual: The last type of corpus depends on the included languages; they can either be monolingual or multilingual. Multilingual corpora include texts in different languages, and the most common types of multilingual corpora are parallel corpora and comparable corpora. A parallel corpus contains the exact text in two or more languages (at least the original text and a translation). An example is the open-source parallel corpus, an online corpus containing aligned corpora from diverse fields and institutions. On the other hand, a comparable corpus contains two or more collections of texts sharing traits such as genre, topic, and period that can be compared, such as the International Corpus of English (ICE) and the International Corpus of Learner English (ICLE).

Numerous corpora are accessible from the Web in different formats, styles, genres and languages. An example of such corpora is iWeb: The Intelligent Web-based Corpus, British National Corpus (BNC), Corpus of Contemporary American English, Hansard Corpus, and NOW Corpus (News On the Web).

3 Challenging of constructing multilingual corpora from the Web

The WebWeb contains massive quantities of text, thus providing the possibility of producing broad and potentially high-quality collections of text data obtained from the WebWeb consisting of various types of documents that often need to be refined or cleaned before being used. Many subtasks are part of this data refinement with varying complexity and scalability, such as removing markup and text extraction, removing redundancy, and removing the so-called boilerplate.

The latter task is an example of the difficulty of corpora building based on web data. In addition to the key textual content of a web page, elements with varying degrees of importance are often appearing, challenging to determine, elements such as navigation menus, timestamps, advertisements, labels for buttons. These will naturally introduce more noise is added to a corpus. Too much of this noisy data may disturb the models built from such a corpus. Identifying which content is relevant or irrelevant is challenging enough for humans and even more difficult for computers.

Another challenge of building large corpora from the Web is the scale of the data. For instance, most projects take web content snapshots distributed by any of the Popular Crawl as their point of departure, and just reading through all files of one such snapshot sequentially, without any actual data processing, takes around 17 hours, which is not a time-consuming assignment compared to the ones needed to refine the data. Therefore, to build large corpora on such a scale, parallel computing is required.

4 Related work

This section discusses the most relevant existing research in terms of similar requirements for corpora creating corpora from the World Wide Web.

4.1 Large scale multilingual corpora collections

In recent years, as the Internet proliferates is already a significant data source and has been increasingly used as a source of linguistic data [3]. Several extensive and multilingual corpus collections gather their texts from the Web, such as [4]-[8].

[9] created the *terabyte corpus* at the University of Waterloo. The corpus is based on a crawl of the Web seeded with URLs from universities and other educational organizations. Perfect duplicates were discarded. The retrieved pages did not undergo further processing, not even to delete HTML or

ensure that they were in the target language English.
[10] explained an English corpus consisting of 10 billion words crawled from seed URLs chosen from various subjects in the open Directory set. The corpus is split into sentences and tokenized using lexical tools to perform sentence-level filtering. There is no near-duplicate elimination. In two NLP tasks, the corpus is tested, where the performance of algorithms trained on the web corpus is shown to be superior to that of the same algorithms trained on 2 billion journal text words.
[11] presented a Multilingual Web Corpus in over 50 languages with over 10 billion tokens licensed under the Creative Commons license. The texts that made up the corpus were extracted from Common Crawl, with around 2 billion crawled URLs. The scale and diversity of the Web also enabled large specialized corpora to be created.
[12] used the text tools to construct corpora in multiple languages using the Common Crawl corpus while [13] created a corpus in English using the C4Corpus tools to find specific pages in the Common Crawl corpus. [14] presented an accessible subset of a USENET corpus collected between 2005 and 2007, excluding NNTP headers and perfect duplicates. Documents containing less than 500 words or more than 500k words are discarded. Moreover, documents containing less than 90 English dictionary words are discarded as well. They do not perform further processing. The resulting corpus contains about 13 billion words.
[15] showed a method of collecting Web texts for a minor language corpus. In [16] a tool is provided for corpora construction. Both works gather and use Web texts as they are, without any filtering or changes being made (except for HTML tags removal).
This brief review shows that, in web crawling, we are certainly not the first ones. Some approaches cover more languages, more in-depth linguistic annotations of web corpora, and several more considerable web-based resources. However, the HULTIG-C collection provides a compromise between huge size and careful post-processing for linguistic purposes. Thus, HULTIG-C is currently a unique resource, extremely valuable for many different types of research in Natural Language Processing.

4.2 Cloud Computing

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources, for example, networks, servers, storage, applications, and services that can be rapidly provisioned and released with minimal management effort or service provider interaction. Cloud computing is composed of five essential characteristics:
On-demand: Computing and IT resources can be provisioned when needed, released when no longer required, and billed only when used without requiring human intervention.
Broad Network Access: The IT resources and their capabilities can be accessed through a standard network, with a client platform as the endpoint.
Resource Pooling: This implies a Multi-Tenant environment where resources are provided to many customers from a single implementation, utilizing physical and virtual assets.
Rapid Elasticity: This implies that the service provides an illusion of infinite resource availability to meet whatever demands are made of it.
Measured Service: This implies that cloud systems have the metering capability at some level of abstraction appropriate to the type of service. Resource usage can be monitored, controlled, and reported providing transparency for both the provider and consumer of the utilized service.

In this section, we first provide an overview of concepts regarding the deployment models of clouds. Next, we describe the three-layer architectural concept for clouds in detail. Then, we describe the cloud platform for NLP services.

4.2.1 Deployment Models

More recently, six cloud deployment models have been defined in the cloud community, namely: public clouds, private clouds, virtual private clouds, community, hybrid, and intercloud.
Public clouds: It is the dominant form of the current cloud computing deployment model. The general cloud consumers use the public cloud, and the cloud service provider has full ownership of the public cloud with its policy, values, profit, costing, and charging model. Many popular cloud

services are public clouds, such as AmazonEC2, S3, Google AppEngine, and Force.com.

Private clouds: The cloud infrastructure is run solely within a single organization and managed by the organization or a third party regardless of whether it is located on the premise or off-premise. The motivation to set up a private cloud within the company has several aspects: To maximize and optimize the utilization of existing in-house resources; Security concerns, including data privacy and trust, often make private clouds a choice for many businesses; there is still a considerable cost of transferring data from local IT systems to a public cloud; Companies are always in need of complete control of mission-critical operations behind their firewalls; Academic often builds a private cloud for research and teaching purposes.

Virtual private clouds: Virtual private clouds allow service providers to offer unique services to private cloud users. These services allow customers to consume infrastructure services as part of their private clouds. The ability to augment a private cloud with on-demand and at-scale characteristics is typical of virtual private cloud infrastructure. Private cloud customers can smoothly extend the trust boundaries (security, control, service-level management, and compliance) to include virtual private clouds. The virtual private cloud concept introduces the complexities of migrating workloads and related data from a private cloud.

Intercloud: In the long term, the intercloud will emerge as a public, open, and decoupled cloud computing inter-network, much like the Internet. In a sense, the intercloud would be an enhancement and extension of the Internet itself. Just as the Internet decouples clients from content (i.e., you do not have to have a preexisting agreement with a content provider to find and access its website in real-time), the intercloud will decouple resource consumers (enterprises) from cloud resource providers, allowing the enterprises to find resources on-demand and without preexisting agreements with providers. Workload migration will be the dominant use case for the intercloud as an open market establishes trust standards and public subsystems for naming, discovering, and addressing portability and data/workload exchange.

Community Cloud: The cloud infrastructure is shared by organizations that have a common interest. This model supports a specific community with shared concerns (mission, security requirements, policy, and compliance considerations). It may be managed by organizations or a third party and may exist on-premise or off-premise.

Hybrid clouds: The cloud infrastructure combines two or more clouds (private, community, or public) that remain unique entities but are connected by standardized or proprietary technology that allows data and application portability (cloud bursting for load-balancing between clouds). Organizations use the hybrid cloud model to maximize their resources by marginalizing peripheral business functions on the cloud while controlling core on-premise operations via a private cloud.

4.2.2 Architectural layers of cloud computing

Cloud computing generally consists of three varieties of architectures that refer to and provide three types of generic services, namely: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS), respectively. Figure 1 shows the architectural layers of cloud computing.

Software as Service (SaaS): is a model of Software licensing and distribution in which Software is licensed on a subscription basis and is stored centrally on a cloud. Such a type of service is also referred to as on-demand software service, as such a service can be rented whenever desired, and the subscription can also be discontinued if no longer required. Because of this flexible leasing mechanism, individuals do not have to buy expensive Software and substantially reduce business costs. SaaS provides many different advantages such as charging just for what we are using, accessibility of the workforce, access to applications from anywhere, and free use of several client software. SaaS users typically use a web browser with a thin client to access cloud services. It has now become a growing operating mode for many companies.

Platform Service (PaaS): is offering a computing platform as a service. It provides developers with an integrated environment for deploying applications without affecting hardware or Software costs and transactions. PaaS has the

same advantages as IaaS. However, its additional middleware functionality, development tools, and other business resources provide more advantages such as reducing coding time, designing applications for multiple platforms, incorporating staff-free development capabilities, helping geographically distributed development teams, allowing economical use of sophisticated tools and managing the application lifecycle efficiently.

Infrastructure as a Service (IaaS): IaaS is hardware-based resources that are provided as infrastructure by the cloud, such as servers, storage devices, networks, etc., that run and operate in the cloud. Users can use the infrastructure on a pay-per-use basis. The hardware resource pool consists of multiple networks and servers spread through multiple data centres, ensuring reliability and redundancy.

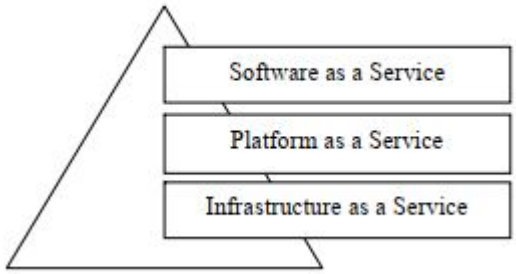

Fig. 1 Architectural layers of cloud computing.

4.3 Cloud platform for NLP services

Cloud services for NLP analysis are becoming very popular among scientists and all users interested in the field. However, there are few reviews of language processing services available. It allows researchers to deploy, share, and use components and resources for language processing, following the data paradigm as a service and software. In this section, we discuss many NLP frameworks built around cloud services, with varying objectives as the following:

Topics Extraction API: this API is offered by MeaningCloud. It provides a solution for tagging locations, people, companies, dates, and many other elements in the text written in Spanish, English, French, Italian, Portuguese, and Catalan. This detection process is implemented by integrating many complex natural language processing techniques that enable the acquisition of morphological, syntactic, and semantic analyses of a text to identify various types of significant elements.

Google Cloud Natural Language: with Cloud Natural Language, Google also emphasizes entity extraction, sentiment analysis, syntax analysis, and categorization. However, this API differs from the others because it is powered by Google's in-depth learning modules – the same ones that drive the query comprehension behind Google Search and the language understanding system behind Google Assistant.

BigML: it is a SaaS approach to Machine Learning. Users can set up data sources, create, visualize, and share prediction models (only decision trees are supported), and use models to generate predictions from a Web interface or programmatically to use REST API.

Myrrix: it is built using Apache MahoutTM. It can be accessed as PaaS using a RESTful interface. It can incrementally update the model once new data is available. It is organized in two layers serving (open source and free) and computation (Hadoop based)

Stanford Core NLP: Stanford CoreNLP provides a set of human language technology tools. It can give the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases and syntactic dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, extract particular open-class relations between entity mentions, get the quotes said by different persons. Stanford CoreNLP integrates many of Stanford's NLP tools, including the part-of-speech (POS) tagger, named entity recognizer (NER), the parser, the coreference resolution system, sentiment analysis, bootstrapped pattern learning, and open information extraction tools. Moreover, an annotator pipeline can include additional custom or third-party annotators. CoreNLP's analysis provides the foundational building blocks for higher-level and domain-specific text understanding applications.

AnnoMarket: it is an open platform for cloud-based text analytics services and language resources acquisition. Providers of text analytics

services and language resources can deploy and monetize their components via the platform.
VMware: Provides several technologies of critical importance to enabling cloud computing and has also started offering its own cloud computing on-demand capability called VMware Cloud. This capability allows enterprises to leverage virtualized clouds inside their own IT infrastructure or hosted with external service providers.
Amazon: As the world's largest online retailer, the core of Amazon's business is e-commerce. While e-commerce can be considered Cloud Computing, Amazon has also provided capabilities that give IT departments direct access to Amazon computation power. Key examples include S3 and EC2. S3 stands for Simple Storage Services. Any internet user can access storage in S3 and access stored objects from anywhere on the Internet. EC2 is the Elastic ComputeCloud, a virtual computing infrastructure able to run diverse applications ranging from web hosts to simulations or anywhere in between. This is all available for a meagre cost per user.Another specific example related to NLP is Amazome comprehend
Salesforce.com: The core mission of Salesforce.com has been in the delivery of Capabilities centred on customer relationship management. However, in pursuit of this core, Salesforce.com has established itself as a thought leader in the SoftwareSoftware as a Service and is delivering an extensive suite of capabilities via the Internet.
AYLIEN API: AYLIEN Text API is a package of Information Retrieval, Machine Learning, and Natural Language APIs that make the analysis of text on an easier scale. It offers eight APIs with functionalities such as article extraction, concept extraction, entity extraction, summarization, classification, semantic labelling, image tagging, sentiment analysis, hash-tag suggestion, language detection and microformat extraction. Most functionalities are available in six languages: English, German, French, Italian, Spanish, and Portuguese. The most exciting feature of Aylien is a text analysis add-on for Google Spreadsheets that allows users to run the API functionality through a familiar interface.

4.3.1 NLP platform as cloud-based service: requirements and methodology

In a cloud computing context, developing an NLP PaaS requires the consideration of the following requirements:
Straightforward deployment and sharing of NLP pipelines – how can we achieve this transparently for the NLP developer, i.e. NLP applications developed on a desktop machine can run without any adaptation on the PaaS. Additionally, developers need to be able to share their NLP pipelines easily as SaaS, with on-demand scalability and robustness ensured by the underlying NLP PaaS;
Efficient upload, storage, and sharing of large corpora – an NLP PaaS needs to provide users with a safe and efficient way to bulk upload, analyze and download large text corpora, i.e., batch processing over large datasets. Furthermore, users need to share their big text corpora between different NLP pipelines running on the PaaS for services bundled within the NLP PaaS and for services generated by the developers; Algorithm-agnostic parallelization. How well it can parallelize the execution of complex NLP pipelines that may contain arbitrary algorithms, not being all implemented / suitable for MapReduce and Hadoop; Load balancing – determine the optimal number of virtual machines for running a given NLP application within the PaaS, taking into account the size of the collection of documents to be processed and the significant overhead of starting up new virtual machines on demand;
Security and fault tolerance – as with any Web application, the NLP PaaS needs to ensure secure data exchange, processing, and storage and be robust in the face of hardware failures and processing errors.
In addition to these technical requirements, an NLP PaaS needs to offer comprehensive methodological support to underpin the life cycle of NLP applications:
(1) Build an initial NLP pipeline prototype, test a small collection of documents, use an NLP application development environment, run on a regular desktop or a local server computer;
(2) Crowd-source, a gold-standard corpus for assessment and/or training, using Web-based collaborative corpus annotation tool, deployed as a service on the PaaS;

(3) Evaluate the performance of the automatic pipeline on the gold standard (either locally within the desktop development environment or through the manual annotation environment on the Cloud). Return to step 1 as required for further development and evaluation cycles;
(4) Upload the large datasets and deploy the NLP pipeline on the PaaS;
(5) Run the large-scale text-processing experiment and download the results as XML, JSON, RDF or schema.org formats. Optionally, an NLP PaaS could also offer scalable semantic indexing and search over the linguistic annotations and the document content;
Lastly, analyze any errors and, if required, iterate again over the critical system development stages, either on a local machine or on the NLP PaaS.

4.4 ClueWeb09 Dataset

The ClueWeb09[1] is a dataset created to support research on information retrieval and related human language technologies. It contains archives of linguistically unprocessed web pages. The dataset construction consists of crawling the Web for about 733 million pages, web page filtering, and organizing into a research-ready dataset. The distribution of ClueWeb09 began in January 2013. A comparison between ClueWeb09 and HULTIG-C is presented in Table 4.4:

	ClueWeb09	*HULTIG-C*
Online	Widely-available	Widely-available
Service	Research/Business	Research/Education
Collection	Feb 2012 : My 2012	Recently
Distribution	Jan 2013	In the near future
Source	Tw, WB, WT	WB
Size	1 million pages	Continuously
Availability	Premium	Free

Table 1 The comparison between ClueWeb09 and Hultig-Corpus

As shown in Table I, the main differences between ClueWeb09 and HULTIG-C include:
Recency – The most recent ClueWeb09 page is from May 2012. Whereas the most recent HULTIG-C page is from 2021;
Language – ClueWeb09 supports only ten languages, while HULTIG-C has applied multilanguage filtering;
Continuity – ClueWeb09 was an one time crawl. HULTIG-C crawl runs continuously;
Availability - HULTIG-C is freely available to everyone immediately. ClueWeb09 requires both organizational and individual agreements to be signed then waiting for physical hard drives to be shipped;
Size - HULTIG-C crawls approximately 1 million pages a month. ClueWeb09 crawled 0.7 billion pages once;
Derived content- The ClueWeb09 dataset contains data from three distinct sources: Web crawl, Twitter links crawl, and WikiTravel crawl. HULTIG-C is not limited to WARC files.

5 HULTIG-C

In this section, we present the architecture for building the multilingual corpus from the web. The framework is shown in Figure 2, consisting of significant phases: Collection Building, Preprocessing, Linguistic annotation, and corpus cloud access.

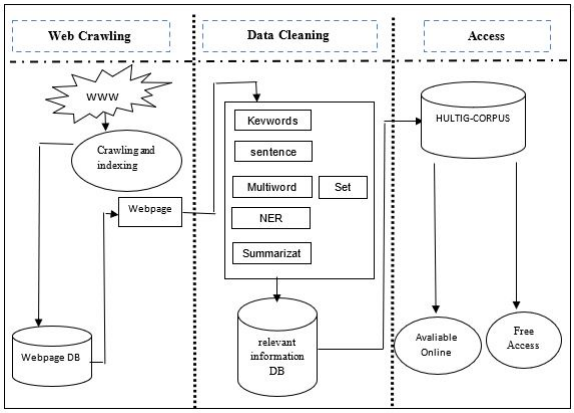

Fig. 2 HULTIG-C.

There are total of 30091927 web pages indexed as of now. This record figure is merely indicative as the size of HULTIG-C is continual rather than fixed. ISO 639-1[2] is used to refer languages in

[1] http://lemurproject.org/ClueWeb09.php/

[2] https://www.loc.gov/standards/iso639-2/php/code_list.php

HULTIC-C. The current record for each language already identified is given in Table 5.

Language	*ISO 639-1 Code*	*# of Indexes*
English	en	4025284
Portuguese	pt	1472009
Welsh	cy	1100927
German	de	541280
Russian	ru	520704
Arabic	ar	202201
Greek	el	192111
Italian	it	185888
Romanian	ro	182659
Lithuanian	lt	166087
Dutch	nl	137345
Finnish	fi	431709
Polish	pl	87410
Albanian	sq	80623
Catalan	ca	60361
Japanese	ja	58395
Ukrainian	uk	17908
Afrikaans	af	1892

Table 2 Number of Indexes for each language in HULTIG-C

In the following subsections, we describe each of these components in detail.

5.1 Multilingual collection building

To construct a web corpus, one needs data. A common way for this data to be collected is by crawling. Web crawling is the process of finding large numbers of web pages by extracting and following hyperlinks from documents that have already been downloaded [17].

The Web Spider component builds our document collections since we use the web as a text source for the Hultig-Corpus. Therefore, it is essential to have an effective tool to download web pages. This is why the web crawler is a critical component of every web-based corpus development project. Many freely accessible and robust crawlers can be used for similar projects such as Apache Nutch, crawler4j, Scrapy, YaCy, WebSPHINX and JSpider. However, many of them are complex, needing extra tools to function effectively, including unnecessary features, while missing some essential features regarding web corpus construction. As a result, our research group has built a high-performance web crawler, the *hultigcrawler*[3], and used it in the HULTIG-C construction process. HultigCrawler is an open-source program developed by the same team, and it is available publicly that can be used to build web page corpus and similar purposes through the crawler. Hultig crawler continuously crawls websites, and it has been indexing since January 2020 in our case.

Hultigcrawler is a python-based web crawler that crawls given websites and extracts specified data from their pages. The python library called "Scrapy" [18] is used for website crawling purposes, while another library known as "Beautiful Soap"[4] is employed to parse the HTML web pages. In the first step, we give the base URL of the website of interest as an input to the hultigcrawler. It starts the crawl process by making requests to the URL defined, obtaining the response object, which is then looped through the elements yielding a Python dict with the extracted items, and finally looking for a link to the next page and scheduling another request using the same process. The crawler runs the process until there are no more web pages left to explore. The crawled data falls under the categories of URL, Tags, Title and Text, which are then saved into the MYSQL database using configured pipelines.

5.2 Preprocessing

After the web pages are collected, the next step consists of preprocessing, which is an integral part of the web corpus building process. License detection, boilerplate elimination, language recognition, and near-duplicate material removal are the four critical components of preprocessing. The following sub-sections go through each of these components in detail:

Boilerplate removal: The elimination of boilerplate is an important phase in building web corpora. In this phase, a web page is cleaned by eliminating uninformative material with no use of in-text comprehensions, such as navigation bars, advertisements, headers, and footers. We use the *jusTool*[5]. It is designed to retain primarily text containing complete sentences, and it is therefore

[3] http://hultigcrawler.di.ubi.pt/
[4] https://www.crummy.com/software/BeautifulSoup/
[5] https://github.com/miso-belica/jusText

well suited for creating linguistic resources such as web corpora;

Language identification: The next step in the processing pipeline is to determine the language of the web pages in the corpus. We depend on an existing python library called *langdetect*[6], which can detect over 55 different languages;

License detection: Copyright is one of the most important considerations when creating a web corpus. Copyrighted content hinders researchers from using or redistributing complete texts within a broad corpus, which stymies the development of many NLP applications. We use Creative Commons (CC) tools. It enables users to grant their work online copyright permissions;

Near-duplicate detection and removal: Deduplication is one of the essential cleaning steps while building a web corpus. Furthermore, crawling data often contains close duplicates or documents that, while not being identical, have considerable overlap. We use the DKPro C4Corpus [33] tool in addition to the specific tools in each subsection. The HULTIG-C is processed with the DKPro C4Corpus tool. It is hosted on Amazon S3, a distributed cloud-based file system. The tool performs cleaning of the corpus in four key phases License detection, boilerplate elimination, language recognition, and near-duplicate material removal.

5.3 Linguistic annotation

HULTIG-C annotated with NLP information consists of four main components, Keywords set, Sentence set, NER set, Multiword set, and Summarization set. Each of these components is described in the following sub-sections:

Keywords set: A subset of words or phrases from a document that can determine the document's context [19]. Since keyword is the smallest unit that can express the meaning of the document, many applications can take advantage of it, such as automatic indexing, automatic Summarization, automatic classification, automatic clustering, topic detection and tracking, information visualization [20]. Therefore, keywords extraction can be considered the core technology of all automatic document processing. Existing keyword set extraction techniques can be classified into four groups: statistics, linguistics, machine learning, and other approaches [21].To identify and extract keywords set by using the statistics approach. The statistics information of the words can be used to identify the keywords in the document. [22] used N-Gram statistical information to index the document automatically. N-Gram is language and domain-independent. Other statistics methods include word frequency [23], IF-IDF [24], word co-occurrences [25], and PAT-tree [26]. Despite numerous methods for keyword generation such as Rake, YAKE, TF-IDF. We used "Gensim(Generate Similar)" [34] software for keyword extraction. Text Summarization module of Gensim supports keyword extraction, which is similar in working as summary generation (i.e. sentence extraction). The algorithm attempts to search significant or seem representative of the entire text. The keywords are not always single words; in the case of multiword keywords, they are typically all nouns.

Sentence set: Sentence extraction is the recovery of a given set of sentences from some document. It is useful for tasks such as document summarization or question answering. Sentence extraction has many approaches that have been applied successfully. From a machine learning perspective, sentence extraction is interesting because, typically, the number of sentences is a tiny fraction of the total number of sentences in the document. Furthermore, the clues that decide whether a sentence should be extracted are extremely precise or weak and interact unexpectedly. From a linguistic perspective, the success of this approach hinges upon the ability to integrate diverse levels of linguistic description. This approach uses the linguistics features of words, sentences, and documents. The linguistics approach includes lexical analysis [27], syntactic analysis[28], discourse analysis [29]. To identify sentence sets, We use LingPipe [35]. It extracts sentences heuristically by identifying tokens in a context that end the SentenceSentence. Furthermore, TextRank Algorithm in Python is used for Sentence Extraction. The crux of this algorithm is to fetch the most relevant Sentences from the piece of the text.

Summarization set: Automatic text summarization is the task of producing a concise and fluent summary while preserving key information content and overall meaning. There are two

[6] https://github.com/Mimino666/langdetect

different groups of text summarization: indicative and informative. Indicative Summarization gives the main idea of the text to the user. The length of this Summarization is around 5% of the given text. The informative Summarization gives brief information about the main text. The length of the informative summary is around 20% of the original text. Due to the significant amount of information and Internet technologies' development, text summarization has become essential for interpreting text information. Text summarization methods can also be classified into extractive and abstractive Summarization. An extractive summarization method involves selecting sentences of high rank from the document based on word and sentence features and putting them together to generate a summary. The importance of the sentences is decided based on the statistical and linguistic features of sentences. Abstractive Summarization is used to understand the main concepts in a given document and then expresses those concepts in clear natural language. We rely on an existing Python's "Gensim" library for summarizing text. It provides functions for summarizing texts based on ranks of text sentences using a variation of the TextRank algorithm [36]. It gives an output summary consisting of the most representative sentences, and it is returned as a string, divided by newlines.

Named Entity Recognition set: NER aims to label the text portions that refer to specific entities, such as persons, locations, organizations. It is a subtask of information extraction and the core of the natural language processing system [30]. Existing NER approaches can be classified into three classes[31]. Namely, Dictionary-based NER, Rule-based NER, and machine learning-based NER approach. To identify NER by Machine learning technique. The ML-based NERs use statistical models to identify unique entity names by using a feature-based representation of the data observed, depending on the annotated documents. The ML algorithms used for NER are Supervised learning (CRFs, SVMs, HMM), Unsupervised learning (clustering), Semi-supervised algorithms (bootstrapping). To identify NER, we use polyglot[7]. It identifies the sequence of words, objects, such as people's and companies' names in a text. The main task of Named entity extraction aims to extract phrases from the plain text that correspond to entities, which can be categorized under three entities of locations, organizations and persons.

Multiword set: Multiword is a sequence of words that acts as a single unit at some level of linguistic analysis" [32]. Many Natural Language Processing tasks are needed to use multiword such as Machine Translation, Natural Language Generation, Automatic Simplification of Text, Enhancing natural language lexical resources. Multiword expression extraction methods from a corpus can be divided into statistical, rule-based, and hybrid. Statistical approaches based on frequency and co-occurrence affinity. Rule-based approaches depend heavily on POS taggers, chunkers, shallow or deep parsers and sometimes even semantic taggers. A hybrid approach is combining different methods. For the identification of multiword, the Multi-Word Units(MWU) toolkit called *sentaweb*[8] is used. It is a tool that aids to identify multiword units in large textbases. It is characterized by applying virtually any text collection, language and multiword expression. It is also helpful in any corpus-based study in computational linguistics.

5.4 HULTIG-C in the C4 Cloud

The HULTIG-C is hosted in the C4 Computing Cloud Platform[9]. The service can be browsed using API called HULTIG Corpus API. This API is constructed using *node.js* package called "swaggerql[10]". It provides the basic structure of API including interface and middleware to connect with database for querying the data. HULTIG Corpus API[11] can be accessed in two different types of full-text searching modes[12] provided by MySQL namely Natural Language Full-Text

[7] https://polyglot.readthedocs.io/en/latest/NamedEntityRecognition.html

[8] http://sentaweb.di.ubi.pt

[9] http://c4.ubi.pt/

[10] https://github.com/swaggerql/swaggerql

[11] http://hultigcorpus-api.di.ubi.pt/

[12] https://dev.mysql.com/doc/refman/8.0/en/fulltext-search.html

Searches[13] and Boolean Full-Text Searches[14]. The interface of API can be viewed in Figure 3.

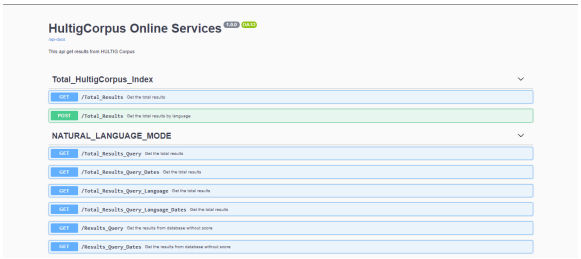

Fig. 3 API interface of HultigCorpus Online Services

Natural language search mode interprets the search string as a literal expression. It is enabled by default if no modifier is provided or when the *IN NATURAL LANGUAGE MODE* modifier is specified. This mode performs a natural language search against a given text collection (one or more columns). When MATCH() is used together with a WHERE clause, the rows are automatically sorted by the highest relevance.

Boolean full-text search mode enables searching for complex queries that include boolean operators such as less than (“<”). More than (“>”) operators which are for lower or higher value, subexpressions (“(” and “)”) which can be in nested form, the plus (+) sign which implies that the word must be present, the minus (-) sign implies that the word must not be present, double quotes (“”) which matches the phrase, literally as typed, an operator that lowers the value’s contribution to the results ( ) and the wildcard operator (*) - the wildcard operator allows searching with fuzzy matching (for example, “intern*” would also match “international”). This mode is enabled using *IN BOOLEAN MODE modifier*.

Besides full-text searching mode, HULTIG Corpus API has a section of ”Total_HultigCorpus_Index”, which returns total results and results based on specific language. As of now, several total results are “30091927”. The API call for total results is without parameters, and the preview can be seen in Figure 4.

[13] https://dev.mysql.com/doc/refman/8.0/en/fulltext-natural-language.html

[14] https://dev.mysql.com/doc/refman/8.0/en/fulltext-boolean.html

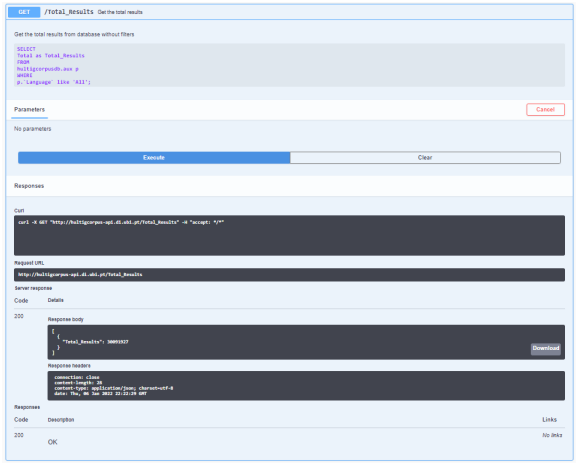

Fig. 4 API call for total results

The API call of total results for a specific language takes a parameter which is ISO 639-1 based language code. Hultig-C support 55 languages that can be queried using the mentioned ISO code. In our case, the results belong to English languages whose parameter value is ’en’. The query returns a total of 4025284 results for English which can be viewed in figure 5.

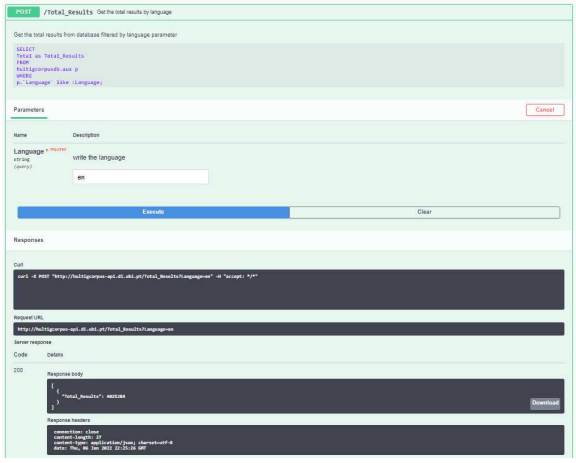

Fig. 5 API call for total results based on language

6 Conclusion

Over the past few years, the use of corpora for various linguistic and non-linguistic purposes has increased rapidly, leading to a great demand for general and specific corpora. This paper has shed light on the NLP Corpus and Services in the Cloud

– HULTIG-C, a new multilingual corpus from the web. The corpus currently includes a variety of domains and subjects. However, it is characterized by language diversity with specific NLP features annotated and stored in a uniform XML format. Hence, as a research goal, the HULTIG-C will be released freely via our team website, including more future updates as the recent corpus becomes available; this will make a valuable corpus for many applications in computational linguistics. Moreover, this paper presents and discusses the challenge of building multilingual corpora from the web. Finally, we believe that making the HULTIG-C available for public use can significantly contribute to NLP and corpus linguistics research, especially for under-resourced languages.

Acknowledgments. This work was supported by National Founding from the FCT Fundação para a Ciência e a Tecnologia, through the MOVES Project - PTDC/EEI-AUT/28918/2017, and by Operação Centro-01-0145-FEDER-000019–C4–Centro de Competências em Cloud Computing, co-financed by the Programa Operacional Regional do Centro (CENTRO 2020), through the Sistema de Apoio à Investigação Científica e Tecnológica – Programas Integrados de IC&DT.

Declarations

- Funding
  This work was supported by National Founding from the FCT Fundacão para a Ciência e a Tecnologia, through the MOVES Project- PTDC/EEI-AUT/28918/2017, and by Operação Centro-01-0145-FEDER-000019 – C4 – Centro de Competências em Cloud Computing, co-financed by the Programa Operacional Regional do Centro (CENTRO 2020), through the Sistema de Apoio à Investigação Científica e Tecnológica – Programas Integrados de IC&DT.
- Conflict of interest/Competing interests
  The authors declare that they have no known competing financial interests or personal relationships or conflicts of interest that could have appeared to influence the work reported in this paper.
- Ethics approval
  The authors declare that they have no known ethics issue that could have appeared to influence the work reported in this paper.
- Availability of data and materials
  Not applicable.
- Code availability
  Not applicable.
- Authors' contributions
  All authors had the same contribution.

References

[1] A. Kilgarriff, and G. Grefenstette, "Web as corpus." In Proceedings of Corpus Linguistics," vol. 2001, pp. 342-344. 2001.

[2] W. H. Fletcher, Corpus Analysis of the World Wide Web. The Encyclopedia of Applied Linguistics. Chapelle:Wiley-Blackwell, 2012.

[3] A. Kilgarriff, and G. Grefenstette, "Introduction to the special issue on the web as corpus," Computational linguistics, vol. 29, no. 3, pp. 333–347, 2013

[4] M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta, "The WaCky wide web: A collection of very large linguistically processed web-crawled corpora," Language Resources and Evaluation, vol. 43, no. 3, pp. 209–226, 2009

[5] M. Jakubicek, A. Kilgarriff, V. Kovar, P. Rychly,and V. Suchomel, "The TenTen corpus family," in 7th International Corpus Linguistics Conference CL, 2013, pp. 335–342

[6] R. Schafer, and F. Bildhauer, "Building Large Corpora from the Web Using a New Efficient Tool Chain," In Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), 2012, pp. 486–493

[7] M. Hundt, N. Nesselhauf, and C. Biewer, Corpus linguistics and the web. Amsterdam and New York: Rodopi, 2007.

[8] C. Fairon, H. Naets, A. Kilgarriff,and G.M. Schryver, "Building and exploring web corpora," In Proceedings of the 3rd web as corpus workshop, incorporating Cleaneval. Louvain: Presses Universitaires de Louvain,

2007

[9] C. Clarke, G. Cormack, M. Laszlo, T. Lynam, and E. Terra, "The impact of corpus size on question answering performance," In Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, 2002, pp. 369–370

[10] V. Liu, and J. Curran, "Web text corpus for natural language processing," In Proceedings of the 11th conference of the European chapter of the association for computational linguistics, 2006 pp. 233–240

[11] I. Habernal, O. Zayed, and I. Gurevych, "C4Corpus: Multilingual Web-size Corpus with Free License," In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), 2016, pp. 914–922

[12] R. Schäfer,"CommonCOW: Massively Huge Web Corpora from CommonCrawl Data and a Method to Distribute them Freely under Restrictive EU Copyright Laws," In Proceedings of the 10th International Conference on Language Resources and Evaluation, 2016, pp. 4500-4504

[13] A. Panchenko, E. Ruppert, S. Faralli, S. P. Ponzetto, and C. Biemann, "Building a Web-Scale Dependency-Parsed Corpus from CommonCrawl," In Proceedings of the 11th International Conference on Language Resources and Evaluation, 2018, pp. 1816–1823

[14] C. Habernal, and C. Westbury, "A USENET corpus," accessed January 28, 2028. Online. Available: http://www.psych.ualberta.ca/westburylab/downloads/usenetcorpus.download.html

[15] R. Jones, and R. Ghani, "Automatically building a corpus for minority language from the web," In Proceedings of theStudent Workshop of the 38 th Annual Meeting of the Associationfor Computational Linguistics, 2000, pp. 29–36

[16] C. Orasan, and R. Krishnamurthy, "An open architecture for the construction and administration of corpora," In Proceedings of the Second International Conference on Language Resources and Evaluation, 2000, pp. 793– 799

[17] C. Olston, and M. Najork, "Web crawling," ournal of Foundations and Trends in Information Retrieval, vol. 4, no. 3, pp. 175–246, 2013.

[18] D. Kouzis-Loukas, Learning scrapy, 1rd ed., Birmingham: Packt Publishing, 2016, pp.243.

[19] A. Hulth,"Building a Web-Scale Dependency-Parsed Corpus from CommonCrawl," In Proceedings of the Conference on Emprical Methods in Natural Language Processing, 2003, pp. 216-223.

[20] D. B. Bracewell , F. REN, and S. Kuriowa, "Multilingual Single Document Keyword Extraction For Information Retrieval," In Proceedings of the Natural Language Processing and Knowledge Engineering, 2005, pp. 517-522.

[21] C. Zhang,"Automatic keyword extraction from documents using conditional random fields," Journal of Computational Information Systems, vol. 4, no. 3, pp. 1169-1180, 2008.

[22] J. D. Cohen, "Highlights: Language and Domain-independent Automatic Indexing Terms for Abstracting," Journal of the American Society for Information Science, vol. 46, no. 3, pp. 162-174, 1995.

[23] J. D. Cohen, "A Statistical Approach to Mechanized Encoding and Searching of Literary Information," Journal of Research and Development, vol. 1, no. 4, pp. 309-317, 1957.

[24] G. Salton, C. S. Yang, and C. T. Yu, "A Theory of Term Importance in Automatic Text Analysis," Journal of the American society for Information Science, vol. 26, no. 1, pp. 33-44, 1975.

[25] Y. Matsuo, M. Ishizuka, "Keyword Extraction from a Single Document Using Word Co-ocuurrence Statistical Information," International Journal on Artificial Intelligence Tools, vol. 13, no. 1, pp.157-169, 2004.

[26] L. F. Chien, "PAT-tree-based Keyword Extraction for Chinese Information Retrieval," In Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1997, pp. 50-59

[27] G. Ercan, and I. Cicekli, "Using Lexical Chains for Keyword Extraction," Information Processing and Management, vol. 43, no. 6, pp. 1705-1714, 2007.

[28] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, c. Nunes, and A. Jatowt, "YAKE! Keyword extraction from single documents using multiple local features," Information Sciences, vol. 509, pp. 257-289, 2020.

[29] H. P. Dick, and J. Nightlinger, Discourse analysis. The International Encyclopedia of Linguistic Anthropology, 2000.

[30] A. Mansouri , L.S. Affendey , and A. Mamat , "Named entity recognition approaches," International Journal of Computer Science and Network Security, vol. 8, no. 2, pp. 339-344, 2008.

[31] Y.C. Wu, T.K. Fan, Y.S. Lee, and S.J. Yen, "Extracting Named Entities Using Support Vector Machines," In Proceedings of the the international conference on Knowledge Discovery in Life Science Literature, 2006, pp. 91-103

[32] N. Calzolari, C. J.Fillmore, R. Grishman, N. Ide, A. Lenci, C. MacLeod, and A. Zampolli, "Towards best practice for multiword expressions in computational lexicons," In Proceedings of the 3rd international conferenceon Language Resources and Evaluation, 2002, pp. 1934–1940

[33] Habernal, I., Zayed, O., & Gurevych, I. (2016). C4Corpus: Multilingual Web-size corpus with free license. In N. Calzolari et al. (Eds.), Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016) (pp. 914–922). Portorož, Slovenia: European Language Resources Association (ELRA).

[34] ŘEHŮŘEK, Radim and Petr SOJKA. Software Framework for Topic Modelling with Large Corpora. In Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks. Valletta, Malta: University of Malta, 2010. p. 46–50. ISBN 2-9517408-6-7.

[35] Alias-i. 2008. LingPipe 4.1.0. http://alias-i.com/lingpipe (accessed October 1, 2008)

[36] Federico Barrios, Federico L´opez, Luis Argerich, Rosita Wachenchauzer (2016). Variations of the Similarity Function of TextRank for Automated Summarization, https://arxiv.org/abs/1602.03606