Development of Web Crawler to Build Indonesian Text Corpus

Janson Hendryli* and Viny C Mawardi
Faculty of Information Technology, Universitas Tarumanagara, Jakarta, Indonesia

jansonh@fti.untar.ac.id, viny@untar.ac.id

Abstract. Recent improvement in natural language understanding research can be attributed to the availability of large scale datasets. Those datasets are mainly in English. In this work, we develop a web crawler with the purpose of extracting Indonesian news content from the DetikNews website and building a large dataset of texts. The web crawler is developed by following the waterfall model using Python, Scrapy, and BeautifulSoup4. It collects more than 790k news from DetikNews, spanning from 2011 to 2020, which consists of a total number of more than 190 million words, almost 2 million unique words, and more than 14 million sentences.

1. Introduction
Nowadays, there has been a growing amount of interest in natural language understanding (NLU) research. Some examples of NLU research topics are for question answering system [1], sentiment analysis [2], and language modeling [3]. The improvement in NLU model performance is largely due to not only recent advancements in deep learning models such as [4] but also the availability of large scale datasets. For example, the BooksCorpus [5] is a document-level corpus containing more than 900 million English words.

The purpose of this work is to construct a large text dataset or corpus in the Indonesian language. Past work of [6] introduced the IndoSum dataset which contains around 20k news articles from several online news sources. We argue that to build a good NLU model, a larger dataset is needed. Hence, this work describes the development of a web crawler that specifically extracts contents from the DetikNews1 web pages. The main reason is that the corpus must comply with the standardized Ejaan Bahasa Indonesia or Indonesian Spelling System. It should also be noted that the dataset does not include any tagging as in [7].

2. Software Development Life Cycle
The process of building computer software commonly follows a system or methodology, which is known as the software development life cycle (SDLC). Several SDLC models have been studied and analyzed in many literatures, such as the waterfall [8], agile software development [9], unified process [10], and so on.

The development of the crawler in this paper follows the waterfall model as described in [11]. It is one of the earliest SDLC models which consists of five consecutive phases: business analysis, design,
implementation, testing, and maintenance. The waterfall model is a sequential approach to software development, starts from customer specification of requirements, and ends with completed product deployment, testing, and support [12]. In [13], the authors argue that the model is predictable and focus on architecture and software system structure planning in detail which is important when dealing especially with a large system. The illustration of the waterfall model can be seen as in figure 1 where the phases are linear with feedback loops.

![Figure 1. The waterfall model [11].](image)

2.1. Analysis

The first phase is the analysis phase where the complete and comprehensive requirements of the software are described according to the needs of the customers [13]. The exact functional and non-functional requirements can be varied based on the users but, generally, it covers the purpose, scope, functions, software attributes, functionalities specifications, and limitations of the software [11].

The main goal is to develop a web crawler with the purpose of collecting web pages in Bahasa Indonesia and build a text corpus. The use case diagram in figure 2 describes the interactions of the user with the web crawler. The user can run the crawler and collect the result. The crawler saves the crawling result in a database or other storage system which can be accessed by the user. The storage system must ensure the accessibility, integrity, durability, and maintainability of the data.

The functional requirements of the web crawler are thus as follows:
1. It follows the hyperlinks from DetikNews index pages.
2. It crawls all news pages from the current timestamp until terminated or the last news in the web pages is reached.
3. It can handle paginated news contents
4. The data collected on each news page are the title, author names, publication date and time, news content, and the URL. The HTML, CSS, and JavaScript in the crawled contents should also be cleaned and discarded from the data.
5. The crawler should ignore videos or images on the web pages.

![Figure 2. The use case diagram of the web crawler to be developed.](image)
2.2. Design
The requirements from the analysis phase are used in the process of planning and designing the software in the second phase. The design phase is where the software developers define the design of the software solution, including the algorithm design, software architecture, user interface, and database schema [11].

The web crawler starts from https://news.detik.com/indeks/?date=target_date where target_date can be set to a particular date. The crawler is set to crawl from the date when it initially runs and retrieve all news on the website or until it is shut down. The index page lists all the news from the target date and it may span to several pages. The crawler should take this into account when parsing the index page. In the news page, the author’s name, title, publication date are stated at the top of the page. To clean the HTML, CSS, and JavaScript from the news content, the BeautifulSoup4 library in Python can be used. The crawler is developed using the Python programming language and utilized Scrapy for a fast high-level web crawling and scraping framework.

When the crawler is ready to be deployed, it is hosted in the cloud using Scrapy Cloud by the ScrapingHub. The platform ensures the crawler is run continuously and provides scalability. It also provides easy crawler monitoring and integration with web proxies and cloud storage, such as Google Cloud Storage and Amazon S3.

2.3. Implementation
The implementation phase involves writing software code and realizing the design of the previous phase. The result of the implementation phase is the application. Furthermore, this phase also involves the deployment of the application to the target platform.

The crawler is implemented using Python and some libraries, such as Scrapy and BeautifulSoup4. With Scrapy, building and managing web crawlers is simpler. After installing the libraries and creating a Scrapy project by running the scrapy startproject news_crawler command from the terminal, the scrapy genspider detik_news news.detik.com command creates the DetikNewsSpider class in the news_crawler/spiders directory.

To continue crawling the index pages from today’s date, the spider obtains the requests to perform from the start_requests() method where it sequentially loops over the target date and generates the Request for the index page URLs. In the parse() method as the callback function, the spider extracts the URL for all news in the index page by using the CSS selector as follows: h3.media__title > a::attr(href). To handle the index page pagination, following CSS selector can be used to get the next page URL: div.pagination > a::attr(href). Table 1 lists all the CSS selectors for extracting the news content. Furthermore, the pagination of the news page can be extracted using div.detail__body-text > div.detail_long-nav > a::attr(href) as the CSS selector. The news content in the next pages can then be concatenated from the previous pages. The meta parameter in the Scrapy Request class is used to pass values from the previous page to the next page requests.

The news in DetikNews may include advertisements, videos, and images in the contents. Furthermore, there are some news pages which only contain a video. Because of the corpus concern with the text documents only, the crawler needs to ignore it all by deleting all script and style tags. Moreover, the BeautifulSoup4 library is employed to find and decompose all <div> and <a> tags, specifically with lihatjg, embed video20detik, detail_body-tag mgt-16, and ratiobox ratio_16_9 sisip_video_ds class attributes.

There are two options to deploy the code to Scrapy Cloud. The first one is by connecting a GitHub account and push the code to a repository, and the second one uses shub from the command line. The shub tools can be installed using pip. The commands to deploy the code are shub login where it asks for an API key and shub deploy <project id>. Running the crawler via the Scrapy Cloud is as easy as clicking the run button and select the detik_news spider.
Table 1. CSS selectors for the extraction of the news’ author, title, publication date, and content.

| CSS Selector          |                      |
|-----------------------|----------------------|
| Author’s name         | div.detail__author::text |
| Title                 | h1.detail__title::text |
| Date                  | div.detail__date::text |
| News content          | div.detail__body-text |

2.4. Testing
The application is then evaluated and validated in the testing phase. The testing includes checking whether the application satisfies the user requirements, specifications, and accomplishes its intended purpose [11]. The verification and validation processes can include several types of testing, such as unit testing, integration testing, system testing, user acceptance testing, and so on, to detect any defects, glitches, and software bugs in the application. The result of the testing phase is described in section 4 below.

2.5. Maintenance
The last phase is the maintenance phase where the developer can refine and improve the performance of the application after deployment to the user’s system. The maintenance phase can also include new user requirements and increasing software reliability [14].

3. Results
This section describes the testing results of the web crawler. A short exploratory data analysis of the text corpus is also explained here, followed by the discussions of the results.

3.1. Testing Results
After running the crawler to extract news from the DetikNews website, the JSON results are downloaded and checked. From the system testing, it is found that the crawler can follow the news URL from the index page and return the news content as specified in the software requirements. News that span several web pages are also concatenated.

By manually observing the extracted texts, some news is found to include the pagination info and non-word sentence, such as “selanjutnya halaman 1 2 3” and news contributor aliases. Since those sentences are not in the content of the news, they must be removed. These bugs are easily resolved using simple text matching and regex. The regex pattern to find the aliases is `\((\w\d+)/(\w\d+)\)`. Every news content also starts with a city name where the news is reported. Since the city name is
always followed by a hyphen symbol, we simply find the location of the symbol and delete every character before the hyphen.

Figure 3 shows the Scrapy Cloud dashboard where the crawler is run. It has collected more than 700k news by June 11, 2020. Subsequently, the logs show some errors when the crawler extracts URL from an advertisement which can be safely ignored because it is always excluded from the crawling result. An example of the extracted data can be seen in figure 4.

![Figure 4](image)

**Figure 4.** An example of the extracted data shown in Scrapy Cloud of a news from March 3, 2020. Each news items saves the information of the author, publication date, news content, title, and the URL.

![Most Common Words](image)

**Figure 5.** Most common words in the corpus.

### 3.2. The DetikNews Dataset

The dataset contains 796,071 news spanning from May 27, 2011, to April 3, 2020. Furthermore, the total size of the JSON files is 1.58 GB in storage. Table 2 shows the summary statistics of the dataset. There are more than 14 million sentences in the dataset which consists of almost 2 million unique
Furthermore, figure 5 shows the 20 most common words in the corpus after employing pre-processing methods such as case folding and stopwords removal (using Sastrawi Python library), with the word jakarta occurs most frequently. The next most frequent words are tahun, warga, indonesia, and korban.

### Table 2. Summary statistics of the dataset

| # of documents | # of sentences | # of words | # of unique words | mean # of sentences per document | median # of sentences per document |
|----------------|----------------|------------|-------------------|----------------------------------|-----------------------------------|
| 796,071        | 14,399,084     | 190,555,047| 1,953,513         | 18.08769                         | 16                                |

4. Conclusions

In this work, we describe the development of a web crawler to extract Indonesian news from the DetikNews website in order to build a large scale Indonesian text corpus. The data collected from roughly 2011 to 2020 contain more than 790k news with roughly 1.9 million unique words.

The crawler extracts the news content and cleans unneeded information such as images, videos, and advertisements by finding specific CSS selectors. In future updates, the website administrator can change the structure of the HTML documents such that the CSS selectors are invalid and the crawler should be updated. For future works, we are also interested in exploring the dataset more and apply natural language tasks to the dataset.

Acknowledgments

The authors would like to gratefully acknowledge DetikNews for providing the news content freely for non-commercial academic research. This research is also supported by faculty research grant from the Institute of Research and Community Engagement of Universitas Tarumanagara.

References

[1] Sultana T and Badugu S 2020 A review on different question answering system approaches Advances in Decision Sciences, Image Processing, Security and Computer Vision. Learning and Analytics in Intelligent Systems vol 4, ed Satapathy S, Raju K, Shyamala K, Krishna D and Favoruskaya M (Springer, Cham)
[2] Gan C, Lu W, Zhang Z and Wang Z 2020 Sparse attention based separable dilated convolutional neural network for targeted sentiment analysis Knowledge-Based Systems 188
[3] Juan, Samson S, Ismail M, Ujir H and Hipiny I 2020 Language modelling for a low-resource language in Sarawak, Malaysia Advances in Electronics Engineering (Singapore: Springer) p 147–158
[4] Devlin J, Chang M, Lee K and Toutanova K 2018 BERT: pre-training of deep bidirectional transformers for language understanding Preprint arXiv:1810.04805
[5] Zhu Y, Kiros R, Zemel R, Salakhutdinov R, Urtasun R, Torralba A and Fidler S 2015 Aligning books and movies: towards story-like visual explanations by watching movies and reading books Proc. IEEE Int. Conf. on Computer Vision p 19–27
[6] Kurniawan K and Louvan S 2018 Indosum: a new benchmark dataset for Indonesian text summarization Int. Conf. on Asian Language Processing p 215-220
[7] Dinakaramani A, Rashel F, Luthfi A and Manurung R 2014 Designing an Indonesian part of speech tagset and manually tagged Indonesian corpus Int. Conf. on Asian Language Processing p 66–69
[8] Royce W 1987 Managing the development of large software systems: concept and techniques Proc. Int. Conf. on Software Engineering p 328–338
[9] Highsmith J A and Highsmith J 2002 Agile software development ecosystems (Addison-Wesley
[10] Jacobson I and Booch G 1999 *The unified software development process* vol 1 (Reading: Addison-Wesley)

[11] Youssef B 2012 A simulation model for the waterfall software development life cycle *Int. J. Engineering & Technology* **2**(5)

[12] Pressman R and Bruce M 2014 *Software engineering: a practitioner’s approach* 8th ed. (New York: McGraw-Hill Education)

[13] Petersen K, Wohlin C and Baca D 2009 The waterfall model in large-scale development *Int. Conf. on Product-Focused Software Process Improvement* p 386–400

[14] Stellman A and Greene J 2005 *Applied Software Project Management* (O'Reilly Media)