University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

Digital Multilingual News Surveillance and Classification System

Adam Fairlie

March 06, 2023

Abstract

Current digital disease surveillance systems lack diversity in their data collection systems, and require results to be translated into English before processing. We developed a more robust data collection system which integrates various types of online data into one system, as well as a web interface to manage the system and view results. Various multilingual machine learning techniques were explored and a classifier to assign categories to incoming multilingual articles was trained and integrated into the collection system. The classifier achieves 89.4% accuracy on collected data

Education Use Consent

Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Project Aims 1
1.3 Chapter outline 2

2 Background **3**
2.1 Digital news surveillance systems 3
2.1.1 BioCaster 3
2.1.2 HealthMap 5
2.1.3 Other Systems 5
2.2 Multiclass News Article Classification 5
2.2.1 Datasets 5
2.2.2 Models 5

3 Design **6**
3.1 Design Principles 6
3.2 System Overview 6
3.2.1 Modules 6
3.3 Data collection 7
3.3.1 Crawler 7
3.4 Data Processing 8
3.4.1 Parser 8
3.4.2 Classifier 9
3.5 Data Management 9
3.5.1 Database 10
3.5.2 Database Connector 11
3.6 Data Presentation 11
3.6.1 Visualisation 11
3.6.2 Web Interface 12
3.7 System Coordination 12
3.7.1 Web Connector 12
3.7.2 Controller 13

4 Implementation **15**
4.1 Web Scraping System 15
4.2 Database 15
4.3 Visualisations and interface 15
4.4 News article classification 16
4.4.1 Research Overview 16
4.4.2 Datasets 16
4.4.3 Model and training choices 18
4.4.4 Data transformations 19
4.4.5 Methodology 20
4.5 Web Interface 20

5 Evaluation **21**
5.1 Automatic web scraping system 21
5.2 Web Interface and Visualisations 21
5.2.1 Usability Testing 21
5.2.2 Conclusions 21
5.3 Multilingual news article classification 21
5.3.1 Evaluation metrics 21
5.3.2 Classifying public data 22
5.3.3 Classifying real-world data 23
5.3.4 Conclusions 23
5.4 Project Limitations 24
5.4.1 News article classification 24

6 Conclusion **25**
6.1 Project Summary 25
6.2 Future Work 25

Appendices **26**

Bibliography **27**

1 Introduction

1.1 Motivation

Diseases can be very quick to spread, travelling quickly between areas and causing harm to many people. It is important to avoid as much harm as possible to have up-to-date information on local disease outbreak risks freely available to members of the public. Traditional methods of disease surveillance can be slow, relying on human experts conducting research and scientific experiments, and often miss coverage of certain diseases or geographic regions. Several digital surveillance systems have been developed to collect real-time information from online sources, such as news articles, which can be used to quickly find and assess the risk of disease outbreaks across the world. This information can allow governments to respond rapidly to any new outbreaks by providing quick risk information to the public and planning for diagnosis, testing and prevention. It is for this reason that many governments are creating systems which incorporate online sources to monitor disease outbreaks, such as the Global Public Health Intelligence Network (GPHIN)[1] from the Canadian government.

Many of these digital surveillance systems are limited in the information they collect. They rely on published news feeds, typically from large established providers in high-resource countries, which retains some of the coverage gap in low-resource areas and minority languages. Other methods of data collection are available which can collect from more sources and are less reliant on the publisher, such as directly scraping and formatting the HTML webpages of news articles. This allows for any source which publishes online news to have its information used in disease risk notifications. Research by Seo and Shin (2017) also suggests that other forms of real-time online data such as search trends and social media posts can be effective in detecting disease outbreaks early.

-The pipeline of disease classification often relies on machine translation, which can slow down the overall system and relies on the accuracy of translation systems which may not capture the nuances of multilingual data. Multilingual models have emerged but are not as widely used due to some limitations. Through the task of multiclass topic classification, I will determine the efficacy of multilingual models under the current limitations of labelled annotated data, and use the system to collect my own data to verify real-world effectiveness.

1.2 Project Aims

This project develops a new digital surveillance system for multilingual news data, which aims to improve aspects of an existing system, BioCaster[2]. The new system developed should have the following aspects:

News collection system The data collection system of BioCaster and many other digital surveillance systems is limited. This project will develop a robust and extensible data collection system which can synthesise data from multiple different sources and source types, allowing for

[1] https://gphin.canada.ca/cepr/aboutgphin-rmispenbref.jsp?language=en$_CA$

[2] http://biocaster.org/

a more extensive selection of online news data to be collected, representing a wider range of public information available including lower-resource countries and minority languages. The project will implement two distinct methods of data collection: RSS[3] feeds and web scraping. It will redevelop an improved version of the current BioCaster data collection system based on RSS feeds so that the current system can integrate into the new system without loss of data sources. A web scraping system will also be developed to extract important information such as article headlines, main bodies and publish dates from a given webpage.

Multilingual news classification The project will explore multilingual classifier models, through the task of multiclass classification. Experiments will be conducted on techniques in multilingual classification and a dataset will be gathered using the data collection system, which will be used to train a multilingual classifier model. This classifier will integrate into the data collection system, allowing for classification of incoming news data without the need for machine translation. The classifier should achieve high accuracy on a dataset which represents the news the data collection system will retrieve.

Data storage and visualisation The collected news data will be stored in structured form in a database. The information will then be used to produce real-time visualisations of collected data, aggregated into useful statistics, graphs and charts, so that data collected can be observed and analysed. These visualisations should include information on the source countries and languages of collected data, the methods of collection used to retrieve the data and the category distributions of news articles. The system should frequently update and be easy to understand.

Web interface A web interface will be developed so that the data collection system can be easily managed, including enabling/disabling data sources, and adding and removing sources. The system should display the last time sources have collected new data, allowing for the highlighting of stale sources which can be removed to save resources. The interface should be responsive and easy to use and understand.

1.3 Chapter outline

This chapter outlines the motivation behind the project and some of the key goals. Future chapters will provide more detail on the exact system design and implementation, and evaluate how effectively the system meets these proposed goals. The paper structure is as follows:

- **Chapter 2** covers research into existing digital surveillance systems, real-time data visualisation systems and the currently available news datasets and approaches to news article classification.
- **Chapter 3** covers the high-level system design of the overall system and its components, and describe the plan for experiments related to multilingual multiclass news article classification.
- **Chapter 4** covers the exact implementation details of the system, and the exact process for data collection and training of multilingual classifier models.
- **Chapter 5** presents the results of the evaluations to consider whether the system achieves its goals. These evaluations are the usability tests for the visualisation and web interface, and the results of training the multilingual classifiers.
- **Chapter 6** summarises and reflects on the project and considers future work which could be carried out to extend or improve certain aspects.

[3]Really Simple Syndication

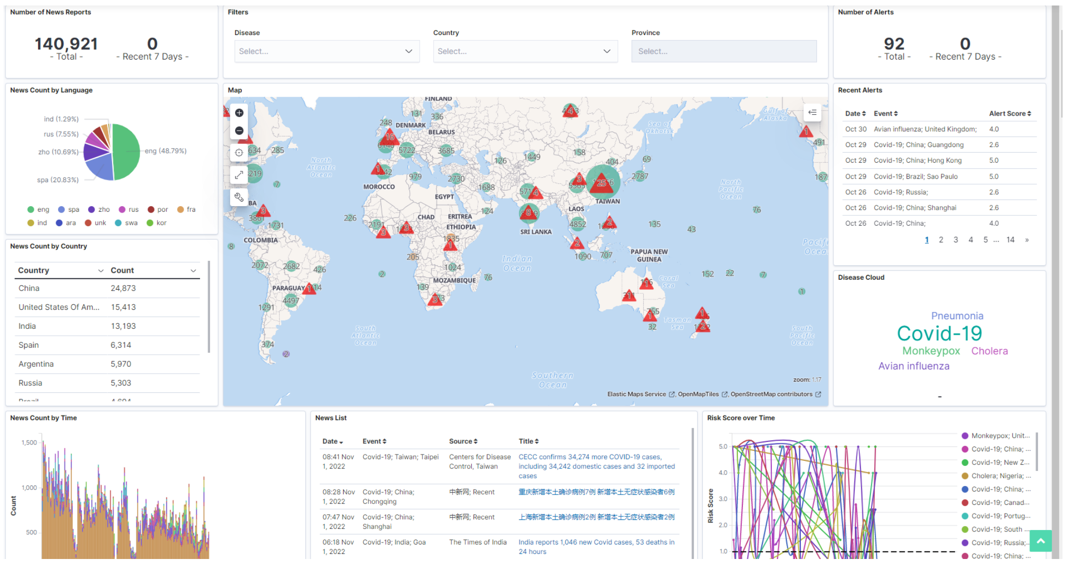
2 Background

2.1 Digital news surveillance systems

Several digital news surveillance systems exist for different purposes, including research systems for public health monitoring and commercial systems designed for businesses to track information related to their brand. This section considers some of these systems and how they operate.

2.1.1 BioCaster

This project reworks and extends some of the functionality currently used in BioCaster[1]. Bio-Caster is a digital news surveillance system currently developed by a research team from the University of Cambridge and McGill University, which uses real-time online news data to rapidly create and display risk alerts for disease outbreaks across the world.



Number of News Reports

140,921 - Total -

0 - Recent 7 Days -

Filters

Disease: Select...

Country: Select...

Province: Select...

Number of Alerts

92 - Total -

0 - Recent 7 Days -

News Count by Country

Country	Count
China	24,873
United States Of Am...	15,413
India	13,193
Spain	6,314
Argentina	5,970
Russia	5,303

Recent Alerts

Date	Event	Alert Score
Oct 30	Avian influenza; United Kingdom;	4.0
Oct 29	Covid-19; China; Guangdong	2.6
Oct 29	Covid-19; China; Hong Kong	5.0
Oct 29	Covid-19; Brazil; Sao Paulo	5.0
Oct 26	Covid-19; Russia;	2.6
Oct 26	Covid-19; China; Shanghai	2.6
Oct 26	Covid-19; China;	4.0

1 2 3 4 5 ... 14 »

News List

Date	Event	Source	Title
08:41 Nov 1, 2022	Covid-19; Taiwan; Taipei	Centers for Disease Control, Taiwan	CECC confirms 34,274 more COVID-19 cases, including 34,242 domestic cases and 32 imported cases
08:28 Nov 1, 2022	Covid-19; China; Chongqing	中新网; Recent	重庆新增本土确诊病例7例 新增本土无症状感染者6例
07:47 Nov 1, 2022	Covid-19; China; Shanghai	中新网; Recent	上海新增本土确诊病例2例 新增本土无症状感染者2例
06:18 Nov 1, 2022	Covid-19; India; Goa	The Times of India	India reports 1,046 new Covid cases, 53 deaths in 24 hours

Figure 2.1: The BioCaster visualisation (from http://biocaster.org)

The BioCaster system collects news articles through a variety of RSS (Really Simple Syndication) feeds across different languages including English, French and Mandarin (Collier et al. 2008). It uses a machine translation server to translate the headlines and article descriptions into English, and various machine learning and rule-based models to filter the English-language data to remove all irrelevant (not disease-related) articles and collect specific details which are used to create a risk alert. These alerts are displayed in a visualisation on the website. A full diagram of the BioCaster system architecture is shown in Figure 2.2.

[1] http://biocaster.org/

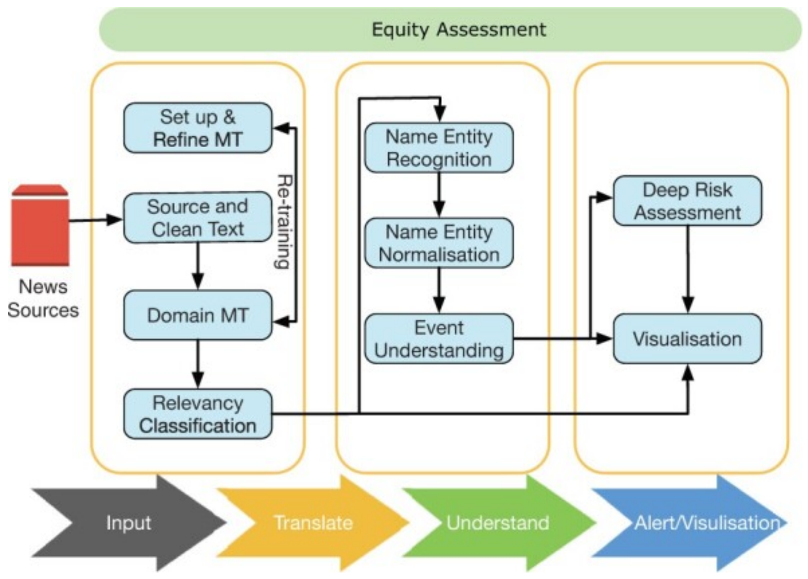


Figure 2.2: *The full architecture of the current BioCaster app. Image obtained from Meng et al. (2022)*

Data collection

The current BioCaster data collection method is dated: it has not been updated in many years and only collects data through RSS feeds, by running a Perl script once per hour to receive updates (Collier et al. 2008). This requires a news source to publish and maintain a feed for its data to be used in the system. These feeds are provided by a limited number of news sources, which are typically larger and written in high-resource languages and countries, where data is not as underrepresented. As This greatly limits the amount of data which can be extracted, particularly from smaller sources in underrepresented countries and minority languages, and as RSS technology is becoming more outdated and falling out of use, the data available for use in surveillance is continuing to diminish in the current system. The data collected from RSS feeds is also limited to a headline and in some cases a short or medium-length description of article content. This provides limited information to models in later stages of the process for performing relevance classification and risk alert creation tasks.

This project will update the BioCaster RSS collection system, as well as add a module for using web scraping to collect news articles from webpage HTML. The system will be designed so that it is easily extensible to other sources of data which may be more prominent in different countries than traditional news, such as social media posts. This also allows for entire article text to be collected instead of just the headline and a short description, which may allow for a richer understanding of the news content and more effective performance in ML tasks.

Visualisation

The BioCaster visualisation is made of many components. Its central component is a diagram of the world populated with red and green icons which show the area, range and severity of disease outbreak events. There are many other visualisation components such as information on the number of reports and alerts and a pie chart representing the language distribution of reports. The widget also contains a filter bar, where each of these visualisations can be filtered by disease, country or province, where the data will update in real-time to reflect the filtering.

2.1.2 HealthMap

HealthMap[2] is another digital disease surveillance system, created by PhHD's John Brownstein and Clark Freifeld. It provides a world map with location-based disease markers for each country, using RSS data and sources gathered by health experts.

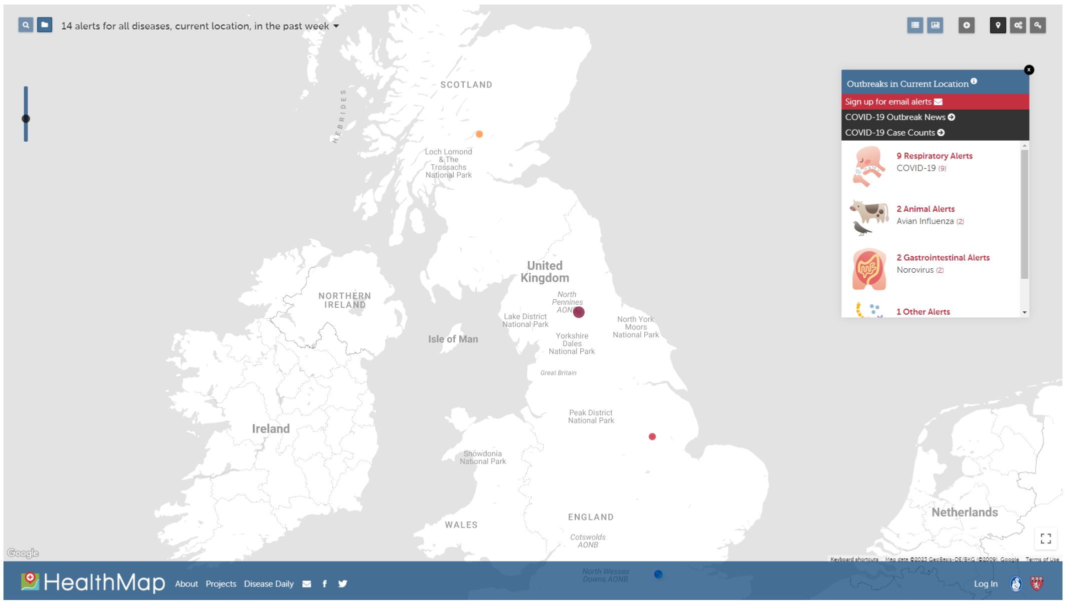


Figure 2.3: HealthMap web interface (from https://www.healthmap.org/en/)

(TO BE COMPLETED)

2.1.3 Other Systems

(TO BE COMPLETED, e.g. Meltwater)

2.2 Multiclass News Article Classification

2.2.1 Datasets

(TO BE COMPLETED) -Valurank news classification huggingface. Chosen because the categories and article distribution is good.

2.2.2 Models

(TO BE COMPLETED)

[2]https://www.healthmap.org/en/

3 Design

3.1 Design Principles

The system architecture was designed with consideration towards a few key software design principles and concepts. The main influences and aims of the overall system design are:

- **Separation of concerns:** Each individual component of the system is given its own module which encapsulates its main functionality and interaction with other modules. This also means that components could possibly be re-used in future projects.
- **Configurability:** The system is designed to give the user as much choice as possible. The system can read a config file which allows the user to control many aspects of the system, such as whether a local or cloud database is used and the maximum number of active sources which are loaded. In addition, individual components are designed so that they can easily be switched out if new components are developed.
- **Extensibility:** Modules of the system which can have implementations changed or new versions created are designed so that this process is as easy as possible, abstracting the consistent functionality of the module into a superclass that new components will inherit from. As previously discussed, social media data has been shown to be a powerful indicator of disease outbreaks, so this system could be extended by adding a social media crawler, for example.

3.2 System Overview

Figure 3.1 shows a diagram of the overall system architecture.

3.2.1 Modules

The overall system is designed so that modules can be swapped out and replaced or added into the system in order to cover different sources of online data. In this project, the system and its modules will be designed to gather news articles from RSS feeds and news websites. The purpose of each module in the overall system is as follows:

- **Crawler:** Connect to an internet source (in this case, an RSS feed or website) and obtain a list of URL's to be parsed for the system (in this case of news articles).
- **Article parser:** Parse the given list of URLs, mining the page source content for details (in this case, Headline, publish date etc.)
- **Classifier:** Pass each article through a machine learning model in order to assign it a category (in this case, each news article is given a topic).
- **Database connector:** Provide an interface to allow the system to access the external database.
- **Database:** Permanently store the collected and classified data, and the crawlers, in a structured form.

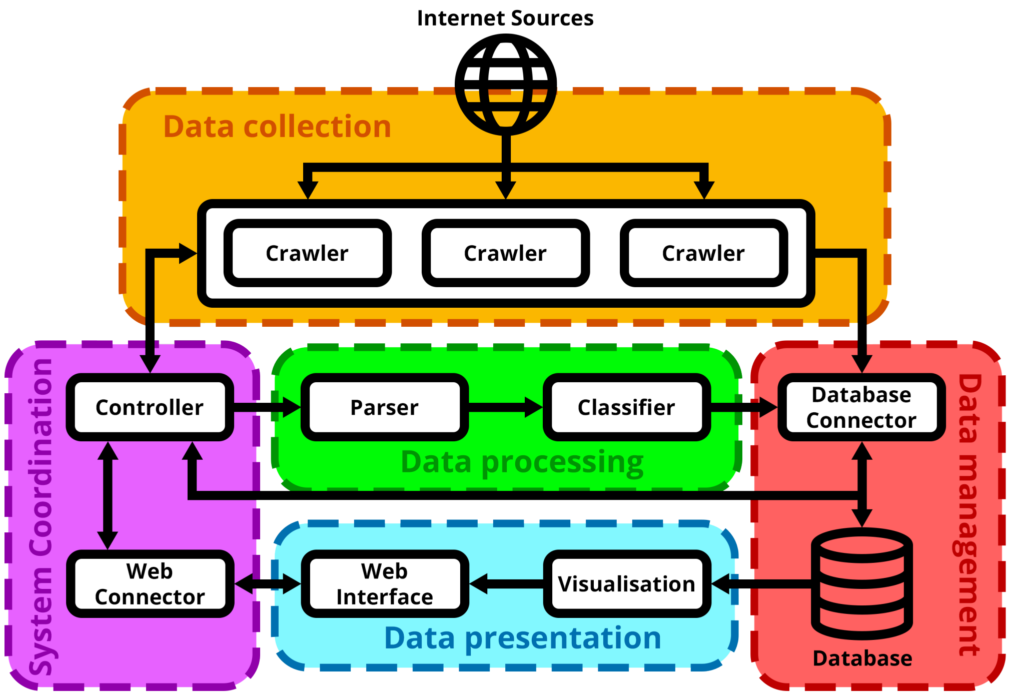


Figure 3.1: The overview of the system architecture for this project, showing the different modules and interactions between them. Each module is contained in a box. Arrows indicate the transfer of data between modules (e.g. A -> B means module B receives data from module A. The larger, coloured boxes indicate different subsystems.

- **Visualisation:** Present the collected and aggregated data in a number of graphs, charts and statistics for monitoring and interpretation
- **Web interface:** Allow administrators to see the visualisation and to manage the active crawlers.
- **Web connector:** Provide an interface for communication between the management website and the scraping system.
- **Controller:** Initialise crawlers from the sources stored in the database. Send the parsed articles to the classifier and communicate with the web interface (send updates and receive instructions).

3.3 Data collection

The data collection system crawls an internet source for all URLs which can possibly contain useful information. In this project, we are concerned with finding the URLs of news articles from a news website or RSS feed. To be appropriate in a disease surveillance system, the data should be updated in near real-time so outbreaks can be quickly understood and responded to.

3.3.1 Crawler

The crawler is the first stage of the data collection process. It represents one individual data source collected in the system, either an RSS feed or a website which is scraped for articles. Crawler objects are stored as a row in a data table of sources. The objects are instantiated from information given by the web interface when a new source is added, or from an existing source in the database. The responsibilities of a crawler are:

- Instantiate itself from information stored in the database or given by the web interface.
- Add itself to the database if it did not already exist.
- Establish a connection with an internet source, in order to receive information.
- Create a list of potential sources of data to be parsed (a list of article URLs).
- Update its state when it has received new data or instructions.
- Update the database with its new state (e.g. delete itself or update its last scrape time).

An abstract crawler class encapsulates most of the functionality of this module, handling all communication between other modules. Any concrete crawler implementations (in this project, there are 2: RSS feed crawler and Web crawler) only have to create a method to crawl specific types of data from the internet and use it to update the last scraped date, as well as override the constructor to add its source type. A full diagram of crawler functionality is shown in Figure 3.2

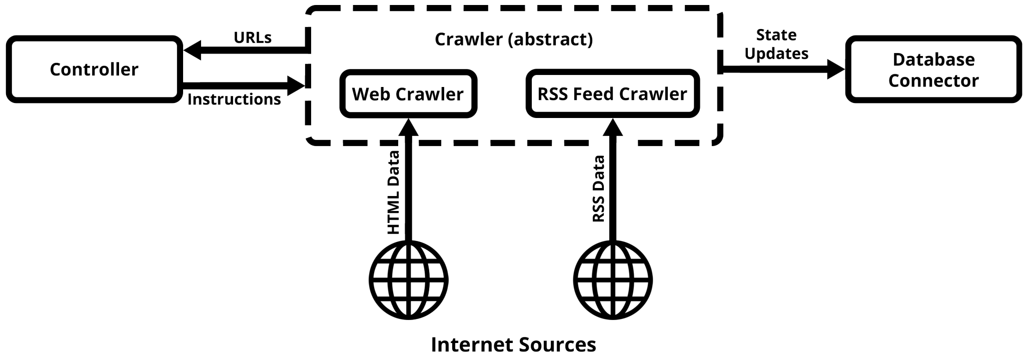


Figure 3.2: *The overview of the inputs and outputs of the crawler system. The diagram shows the concrete Web Crawler and RSS Feed Crawler implementations being encapsulated by the abstract crawler class, which handles interactions with other modules. The concrete classes each collect a different type of data from the internet sources.*

3.4 Data Processing

The data processing system creates structured data from internet sources, converting unclean data such as an RSS feed or the HTML of a webpage into a formatted news article with a headline and article body and associated data. It also attempts to understand the data by assigning a category to the incoming articles. The data processing subsystem should be able to process data at a high volume from sources in 10 different languages: English, French, Spanish, Portuguese, Russian, Indonesian, Mandarin, Korean and Arabic. When data is processed, it will be added to the database so it can be used in visualisation.

3.4.1 Parser

The parser receives as input a combined list of URLs from the crawlers and for each URL, will retrieve the page HTML and use it to extract details about the page (in this case, the pages will be news articles) which can be inserted into the database. The article URLs will be shuffled randomly before parsing to avoid very fast repeated requests to the same host, which smaller sources may not be able to easily handle. The parser assumes that news sources are monolingual to avoid language detection wherever possible, but will attempt to detect the language if parsing is incorrect. After this process is complete, the parser sends the parsed data to the classifier to assign the article a topic. The information required from a news article for this project, which the parser will attempt to scrape, is:

- Article headline
- Article body
- Publish date
- Language

An abstract parser class handles all communication with other modules. Any concrete parser implementations (such as the article parser in this project) only have to implement a specific method for crawling a list of given URLs and retrieving structured data from the URLs and returning a final list. A full diagram of parser functionality is shown in Figure 3.3

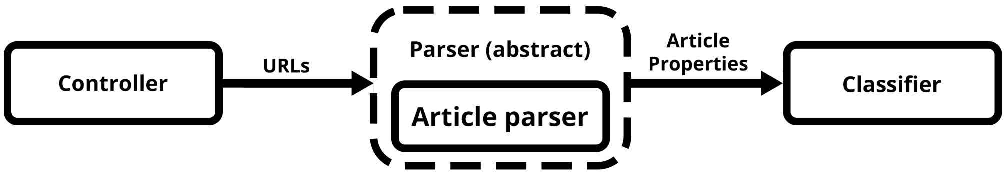


Figure 3.3: *The overview of the inputs and outputs of the parser system. The diagram shows the concrete article parser implementation being encapsulated by the abstract parser class, which handles interactions with the controller and classifier.*

3.4.2 Classifier

The classifier receives a list of parsed articles, with features extracted. It then performs any necessary data preparations and tokenization before feeding the articles through a trained classifier model to assign a category to each news article (e.g. Sports, Entertainment). In this project, the classifier module will use a pre-trained BERT-based classifier model loaded from a file and will tokenize and classify the concatenated article headline and body. The parsed articles with topics will be sent to the database connector and inserted into the database.

An abstract classifier class handles communication with other modules and batching of input data. To create a concrete classifier implementation, we would have to implement:

- A constructor which properly initialises the model and any tokenizers.
- A method which preprocesses a list of input data.
- A method which receives an input batch and outputs a list of corresponding classification labels.

A full diagram of classifier functionality is shown in Figure 3.4

3.5 Data Management

The data management subsystem stores the data (data sources and collected articles) in a permanent form, where it can be shown in the visualisations. The database system should be able to handle many reads (for article duplicate checking) and writes (for inserting new articles) at a high volume.

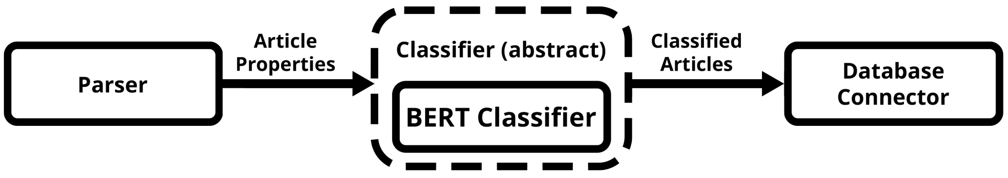


Figure 3.4: The overview of the inputs and outputs of the classifier system. The diagram shows the concrete BERT classifier implementation being encapsulated by the abstract classifier, which handles interactions with the parser and database connector.

3.5.1 Database

For this project, the database will consist of two tables: **Articles** and **Sources**. An ER Diagram for the database system is shown in Figure ??

	Source	
Attribute	**Type**	**Description**
ID	Primary Key	A unique auto-generated source ID.
URL	Text	The data source URL (e.g. "https://www.bbc.co.uk/news")
Name	Text	A name describing the source (e.g. "BBC News")
Country	Text (2)	The ISO 3166-1 alpha-2[1] code of the country the source originates from (e.g. "GB")
Language	Text (2)	The ISO 639-1[2] code of the sources main language (e.g. "EN")
Data Source	Text	The type of source this is (e.g. "RSS/Atom feed")
Last retrieved	Date/Time	The time this source last retrieved new data
Active	Boolean	Whether this source is active (if False, when instantiated the crawler will be disabled)
	Article	
Attribute	**Type**	**Description**
ID	Primary Key	A unique auto-generated article ID.
URL	Text	The URL of the article
Headline	Text	The article headline
Body	Text	The main body of the article
Country	Text (2)	The ISO 3166-1 code of the country of the original source
Language	Text (2)	The ISO 639-1 code of the language of the article
Published	Date/Time	The time this article was published
Retrieved	Date/Time	The time this article was retrieved by the scraping system
Source name	Text	The name of the source this article is collected from
Source type	Text	The type of source this article came from (e.g. "Web scraper")
Category	Text	The assigned category of this article (e.g. "Entertainment")

Table 3.1: List of attributes in the Source and Article tables, with data type and description. Primary key attributes are highlighted in bold. "Text (2)" denotes that this field is a text field which is exactly 2 characters long.

[1]https://en.wikipedia.org/wiki/ISO_3166-1_alpha-2
[2]https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes

3.5.2 Database Connector

The database connector provides a connection between the system and the external database, allowing the other modules to send and retrieve the data. In this system, the connector must have functionality for:

- Creating the Article and Source tables
- Adding new sources and articles
- Retrieving stored sources
- Deleting sources
- Updating sources (enabling/disabling, updating last scrape time)
- Searching articles by URL and headline (for duplicate checking)

A full diagram of database functionality is shown in Figure 3.5

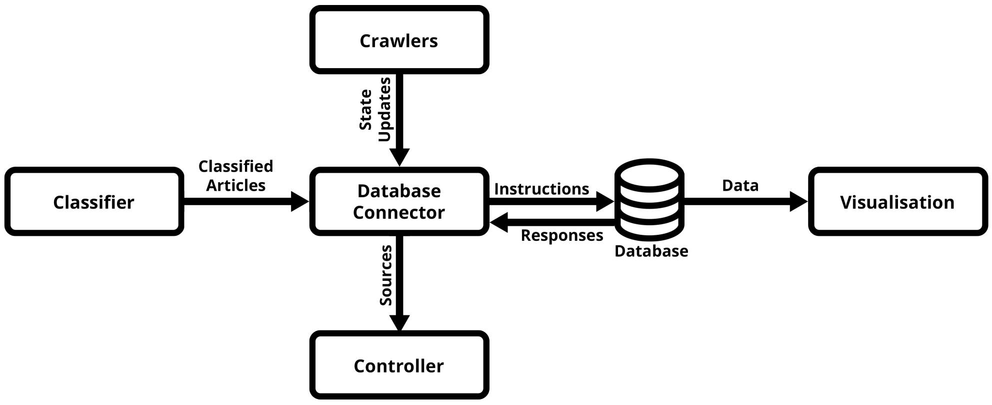


Figure 3.5: The overview of the inputs and outputs of the database and connector.

3.6 Data Presentation

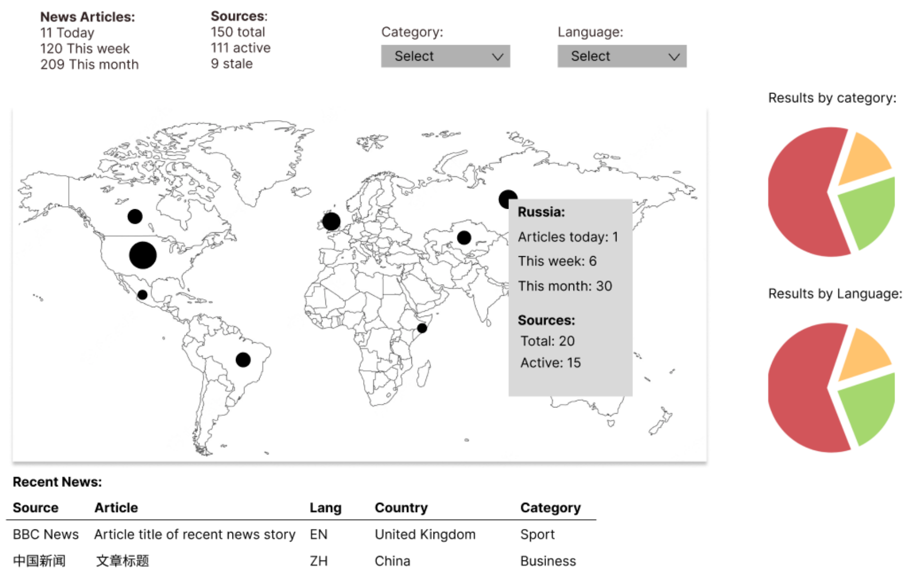
The data presentation subsystem is concerned with creating an easy interface for a user to view, analyse and understand the data collected, as well as easily interact with the system to modify its behaviour. The system should have interfaces which are easy to use and understand and provide useful insight and control on the data collection process.

3.6.1 Visualisation

The visualisation is mainly based on the current BioCaster interface, as shown in Figure 2.1. This design has been updated to reflect the changes from my project to the current system, including the addition of news article topics and different source types of information retrieved. I also did not have the specific region or disease information to create the alerts seen on the BioCaster interface. Shown in Figure 3.6 is an initial wireframe of the new visualisation:

After supervisor feedback, more emphasis was placed on filtering by source type. The final visualisation will have the following modules:

- A bar for filtering by Category, Language and Source type
- A section showing the number of total and active sources
- A section showing the number of articles scraped today, this week and this month



News Articles:
11 Today
120 This week
209 This month

Sources:
150 total
111 active
9 stale

Category: Select

Language: Select

Results by category:

Results by Language:

Russia:
Articles today: 1
This week: 6
This month: 30

Sources:
Total: 20
Active: 15

Recent News:

Source	Article	Lang	Country	Category
BBC News	Article title of recent news story	EN	United Kingdom	Sport
中国新闻	文章标题	ZH	China	Business

Figure 3.6: The wireframe for the new visualisation.

- Pie charts showing data distribution by source type, language and category
- A table showing the most recent articles collected
- Time-series data on articles collected over time
- A world map showing information per country (number of articles, most common category etc.)

3.6.2 Web Interface

The web interface is designed to show the visualisation on a webpage, as well as provide functionality for managing the scraping system. The web interface should provide utility for the following tasks:

- Enabling/disabling the entire scraping system
- Enabling/disabling individual sources
- Deleting individual sources
- Adding new sources
- Viewing stale sources (this was requested by the BioCaster team)

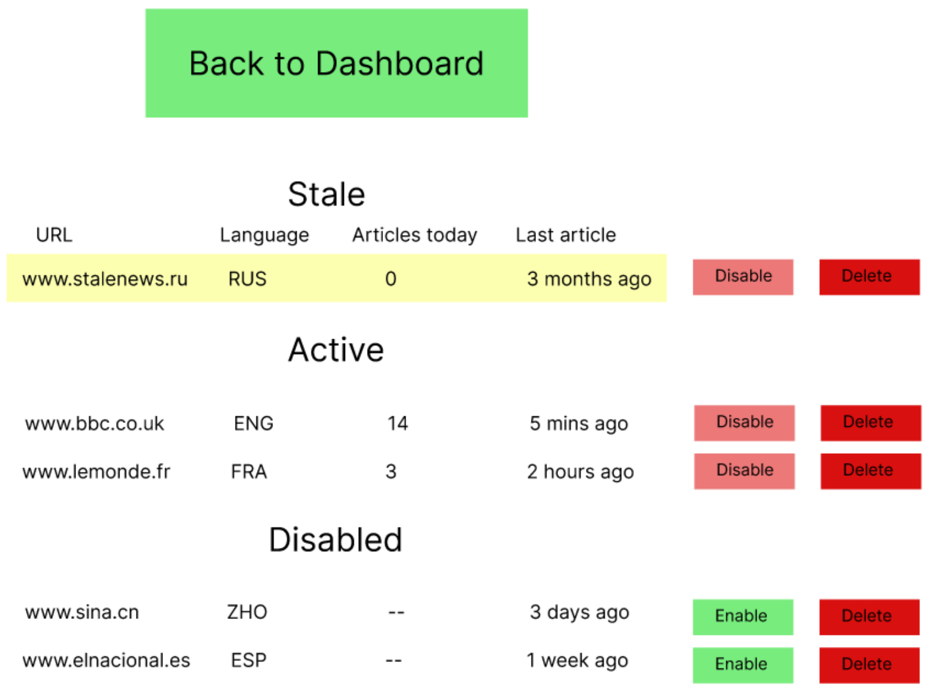
The wireframe for the manage sources page of the web interface is shown in Figure 3.7

3.7 System Coordination

The system coordination subsystem is responsible for orchestrating the data collection process and connecting subsystems, as well as allowing the interface to communicate with the scraping system.

3.7.1 Web Connector

The web connector provides the interface between the web server and the scraping system. It ensures that the commands performed on the interface are translated into instructions for



Back to Dashboard

Stale

URL	Language	Articles today	Last article		
www.stalenews.ru	RUS	0	3 months ago	Disable	Delete

Active

www.bbc.co.uk	ENG	14	5 mins ago	Disable	Delete
www.lemonde.fr	FRA	3	2 hours ago	Disable	Delete

Disabled

www.sina.cn	ZHO	--	3 days ago	Enable	Delete
www.elnacional.es	ESP	--	1 week ago	Enable	Delete

Figure 3.7: The wireframe for the manage sources page of the web interface.

the controller, which will perform the requested action. These instructions include enabling, disabling, deleting and adding sources, as well as enabling or disabling the entire scraping system. The web interface can also request updates through the web connector, to show the user the most recent information on the most recent scrape times of each source. Figure ?? shows how the web connector integrates with all other modules.

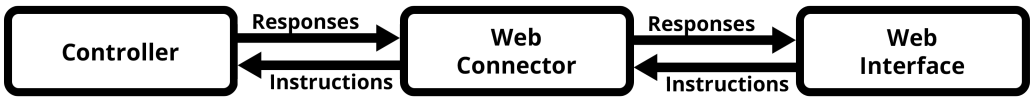


Figure 3.8: The overview of the inputs and outputs of the web connector.

3.7.2 Controller

The controller is the main orchestrator of the system, and is the module called when running the system. It performs the following roles:

- Ensures the tables exist in the database.
- Receives all sources from the database and instantiates Crawlers.
- Instantiates parser and classifier object(s).
- Starts the web server for the web interface.
- Co-ordinates when the crawlers look for new data.
- Sends new crawler data to the parser(s).

- Communicates with the web interface through the web connector, sending updates and performing instructions.

A full diagram of the controller functionality is shown in Figure 3.9

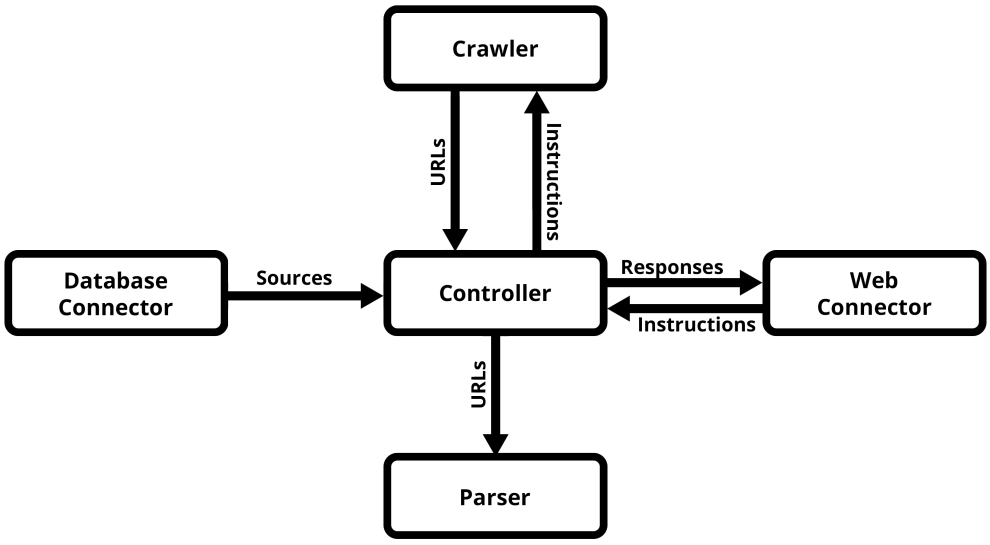


Figure 3.9: *The overview of the inputs and outputs of the controller module.*

4 Implementation

4.1 Web Scraping System

(TO BE COMPLETED) The main free and open-source technologies for scraping news articles from websites in Python were *Newspaper3K*[1] and *news-please*[2], which is built on top of Newspaper3K and adds some extra features. Another option considered was *Newscatcher*[3], but the free API is limited in how many calls can be made and this made it unsuitable for this project. Finally, I considered *pygooglenews*[4] which provided some promise in using google news to find articles under certain subjects, keywords, languages and regions. For this project, I found it desirable to have better control of the exact sources collected, instead of filtering through keywords and relying on Google's source selection, but a scraper using this library could easily be added to extend the capabilities of the current system. I decided to move forward with the former two libraries and conducted an experiment to compare their capabilities.

I compared the features present in each of the two libraries. Notably, Newspaper3K can perform full website scraping in Python, whereas news-please can only do this using its Command Line Interface (CLI). I attempted to scrape 3 articles from each of the 109 previously selected websites, across 10 languages, and compared the number of successful scrapes (without error) and the average speed. The results are shown below:

(Results table to be converted)

Newspaper3K scraped 103 of the 109 websites (94.5%) without error, whereas news-please scraped 102/109 (93.58%). The average scraping times are similar in both libraries, but Newspaper3K was faster at scraping in 8 of the 10 languages, and average scraping time per article was 14.82% lower, Based on these factors, I chose to use Newspaper3K for the scraping system.

4.2 Database

(TO BE COMPLETED)

4.3 Visualisations and interface

(TO BE COMPLETED) The BioCaster visualisation is created using the Elastic software stack[5], where visualisations are created in Elastic Kibana, which automatically generates and updates visualisations from an underlying NoSQL database created using Elasticsearch.

In addition, I chose to change the article density to cover the whole country instead of being a dot, to make the source of the data more visible (so that it does not obstruct the view of other

[1]https://github.com/codelucas/newspaper
[2]https://github.com/fhamborg/news-please
[3]https://newscatcherapi.com/news-api
[4]https://github.com/kotartemiy/pygooglenews
[5]https://www.elastic.co/elastic-stack/

countries) and created a monochromatic green colour map, where darker colours represented more articles retrieved as is common practice in world density maps (Our World in Data 2022; Office for National Statistics 2022).

4.4 News article classification

4.4.1 Research Overview

This experiment aims to consider multiclass classification of news articles in different languages, using multilingual models. In particular, this experiment uses articles in six languages: English, French, Spanish, Portuguese, Mandarin and Indonesian. By researching the topics used by the news sources previously collected, as well as the topics used in publicly available news classification datasets, six news topic categories were chosen: Entertainment/Arts, Sports, Politics, Science/Technology, Business/Finance and Health/Welfare.

In many languages, particularly minority languages, little to no labelled public datasets are available for topic classification of news, meaning models cannot be trained by traditional means. This creates difficulty when trying to perform real-time classification tasks on news written in minority languages, for purposes such as creating disease alerts. We will investigate the efficacy of some techniques and approaches for multilingual document classification, by investigating the following research questions:

- How effective is machine translation of English articles into other languages, as a method of upsampling, in increasing the effectiveness of multilingual models?
- How effective are models trained on public data (both in English and translated into all 6 languages) when used to classify collected news data directly from sources across the six languages?
- What level of performance can be achieved by multilingual models when trained and evaluated on collected articles from sources in each of the six languages?

4.4.2 Datasets

Public data

The dataset used is a dataset made available on huggingface by Villanova and Abdulmatin (2022). The data originally consists of 3,722 classified English-language news articles scraped from various online news sources with 7 topic labels: World, Politics, Tech, Entertainment, Sport, Business, Health, and Science. To match the categories being considered in this experiment, articles belonging to the "World" category were dropped, and those belonging to the "Tech" and "Science" categories were merged into a new "Science/Technology" category. An augmented dataset was also created by adding copies of each article translated into each of the other five languages (French, Spanish, Portuguese, Mandarin and Indonesian). Translations were obtained using the Google translate API through the python library *googletrans*[6]. The distribution of topics in the original and augmented datasets is shown in Table 4.1 and Figure 4.1.

Real-world data collection

We collected news articles from some of the most popular online news sources across the six languages. The list of six topics used was created after considering the most common category sections in news sources across different languages, and combining similar categories which were often combined by the original sources (such as business and finance). A set of keywords relevant to each category was also generated, shown in Figure 4.2. Each collected article was self-labelled by the source, and the articles were collected and categorised using the previously developed

[6]https://pypi.org/project/googletrans/

	Number of Documents						
	Entertainment	**Sports**	**Politics**	**Science/ Technology**	**Business**	**Health**	**Total**
Original	485	454	442	838	461	467	**3147**
Translated	2910	2724	2652	5028	2766	2802	**18882**

Table 4.1: *Category distribution of the Valurank news categorization dataset. This table includes the original dataset and the augmented version obtained from adding the translations of each article in each of the other 5 languages.*

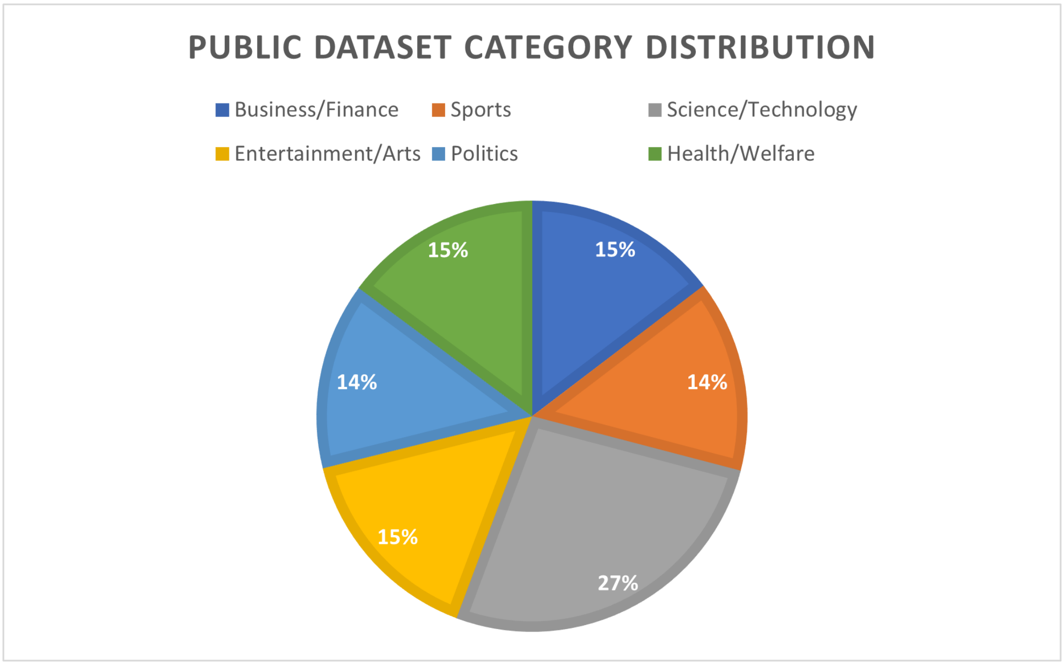


Figure 4.1: *The distribution of categories in the public Valurank dataset*

scraping system, with each article URL obtained and classified either through topic-specific RSS feeds or by matching the URL of article webpages with topics which match category keywords (e.g. abcnews.go.com/Sports). A complete list of data sources used is available in the appendix (link specific appendix).

Each collected article URL was parsed using the *Newspaper3k*[7] library. Any article with a main body under 100 characters in length was discarded, and the article headline and body were concatenated into one text field, used for classification. Originally data was collected for 10 different languages, but only the six considered returned a sufficient number of results for model training. In the final dataset, "Business/Finance" and "Science/Technology" articles in Mandarin were downsampled in order to balance the categories, by randomly dropping articles of these categories and languages at probabilities of $p = 0.75$ and $p = 0.5$ respectively.

The distribution of articles by category and language is shown below in Table 4.3 and in Figures 4.2 and 4.3.

[7]https://newspaper.readthedocs.io/en/latest/

Category	Keywords
Entertainment/Arts	Entertainment, Art, Arts, Culture, Movies, Cinema, Music, Books, Theater, Television, Dance, Celebrity
Sports	Sport, Sports, Soccer, Football, Basketball, Tennis, Golf, Rugby, Motorsport, Formula 1, Physical Education
Politics	Politics, Election, Parliament
Science/Technology	Science, Technology, Tech, SciTech, Space, Physics
Business/Finance	Business, Finance, Economy, Stock exchange, Stock market, Market, Money, Investment
Health/Welfare	Health, Welfare, Coronavirus, Pharma, Pharmacy

Table 4.2: *List of category keywords used for collecting news data. Sources which had RSS feeds or specific URL patterns with these keywords were selected for the corresponding category.*

Language	Category						
	Entertainment/ Arts	**Sports**	**Politics**	**Science/ Technology**	**Business/ Finance**	**Health/ Welfare**	**Total**
Mandarin	511	706	329	961	802	217	3,526
English	885	682	435	607	493	390	3,492
Indonesian	478	545	71	295	598	255	2,242
French	342	369	138	88	235	114	1,286
Portuguese	136	197	196	240	232	167	1,168
Spanish	203	400	159	48	238	84	1,132
Total	2,555	2,899	1,328	2,239	2,598	1,227	**12,846**

Table 4.3: *The distribution of collected news articles by language and category.*

4.4.3 Model and training choices

Multilingual models

To explore the possibility of classification in real-time without the need for machine translation, only multilingual transformer models were considered. One of the most popular multilingual models is mBERT, or multilingual BERT (Devlin et al. 2018). It was trained on the Wikipedias of the 104 languages with the most content, including the six considered here. BERT and its variations are widely used and often produce strong results in classification tasks, and the large variety of languages trained provides flexibility for potential classification of minority language sources. Research by Pires et al. (2019) suggests mBERT may learn an effective language-agnostic latent space which yields impressive results when the model is evaluated on different languages than it is trained on, but this effect decreases as the languages become less structurally similar.

The second model being considered is XLM-RoBERTa, developed by the Facebook AI team (Conneau et al. 2019). It is trained on 2.5TB of CommonCrawl data across 100 languages, including the six considered here. It again provides a lot of coverage for minority languages and has been shown to outperform mBERT in many multilingual tasks such as named entity recognition (NER) and relation extraction (Li et al. 2021; Lan et al. 2020), as well as in news classification (Alam et al. 2020).

Baselines

To test the effectiveness of machine translation upsampling, we will use the models trained on the English only public dataset as a benchmark, to measure how much (if any) accuracy increases using translation to augment the dataset. When comparing how models trained on public data transfer to collected real-world multilingual data, we can use static models which do not learn

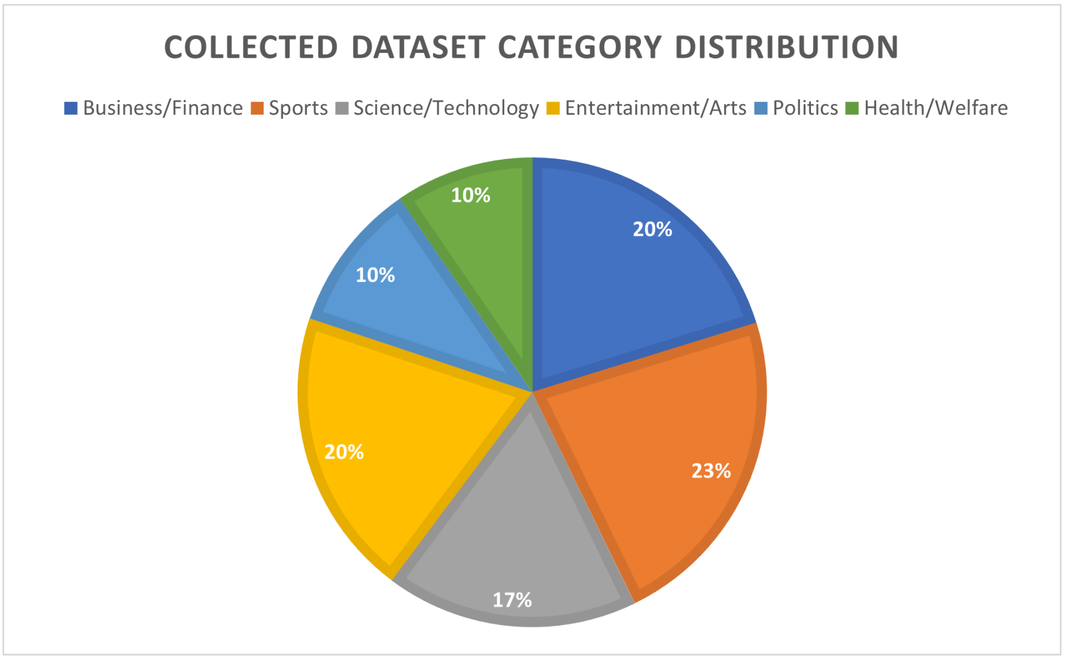


Figure 4.2: *The distribution of categories in the collected dataset*

Parameter	Values
Batch size	16, 32
Number of epochs	2, 3, 4
Learning rate	5×10^{-5}, 4×10^{-5}, 3×10^{-5}, 2×10^{-5}

Table 4.4: *A list of hyperparameters tuned before training each model for classification. In each case, a grid search was performed with these values.*

from the data as a baseline. In this case, we use two static baseline models: a model which uses stratified sampling (it samples a prediction for an input randomly, using the category densities as a probability distribution) and most common sampling (always predicts the category with the most training examples). This allows us to consider how much of what the model learns from the public data can be applied to the real-world data.

Hyperparameters

The hyperparameters investigated in this experiment were batch size (how many examples are used in training per backpropagation cycle), number of epochs (how many times does the model repeat the training data) and learning rate (how quickly does the model update its parameters to improve predictions). The values considered are based on the recommendations and experiments from the original BERT paper, by Devlin et al. (2018), and shown in Table 4.4. In each case, an exhaustive grid search was performed (24 trials) on the hyperparameter values and the highest accuracy model was chosen for the evaluation.

4.4.4 Data transformations

Tokenization

Each of the datasets were tokenized using the tokenizer associated with the multilingual model. multilingual BERT uses WordPiece tokenization, where each sentence is split into words by

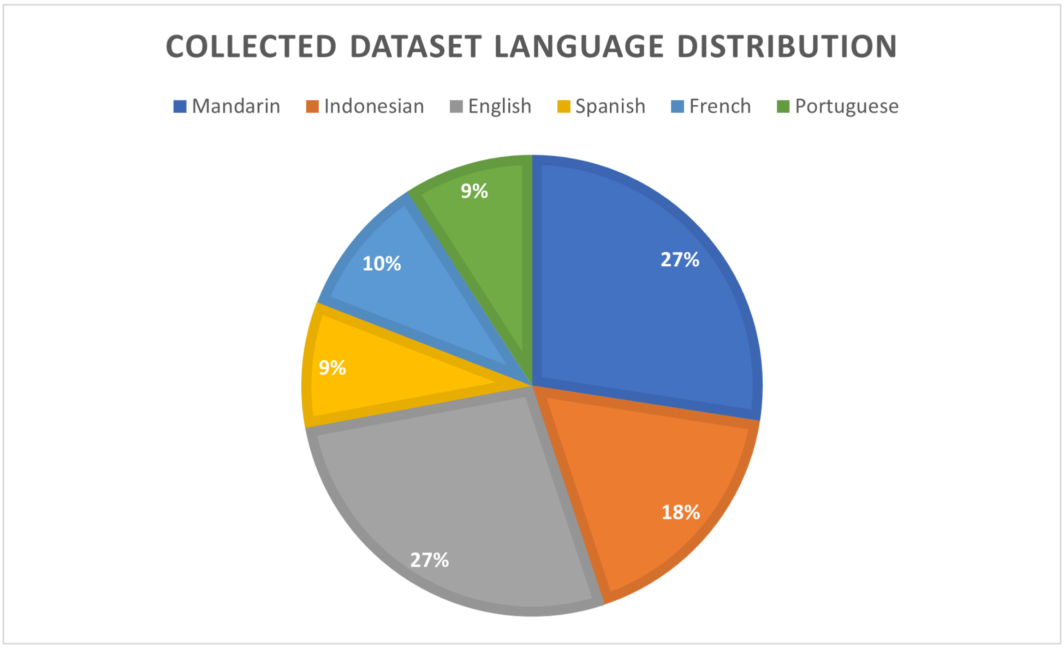


Figure 4.3: The distribution of language in the collected dataset

spaces or punctuation, and each word is broken into one or more subwords. Each unique subword is associated with a token ID, and input data is converted into a sequence of these token ID's, which are input into the mBERT model for classification. XLM-RoBERTa uses SentencePiece tokenization, a similar subword tokenization scheme which is lossless (it can exactly replicate the input sequence as it retains whitespace and punctuation information) and language independent (Kudo and Richardson 2018).

Data splitting

Because the categories in our datasets were not too imbalanced (the largest category disparity is around 2:1) we decided that data can be split randomly into a train and a test split. For each dataset, a train/test split of 80/20 (80% training data) was used.

4.4.5 Methodology

(TO BE COMPLETED)

4.5 Web Interface

(TO BE COMPLETED)

5 Evaluation

5.1 Automatic web scraping system

(TO BE CONDUCTED)

5.2 Web Interface and Visualisations

5.2.1 Usability Testing

(Results to be collected/analysed)

5.2.2 Conclusions

(Results to be collected/analysed)

5.3 Multilingual news article classification

5.3.1 Evaluation metrics

We will evaluate each model's performance using the classification accuracy (which percentage of predicted categories are correct), as well as its weighted F1 score. These ratings are widely used in machine learning literature, including classification tasks (Alam et al. 2020; Al-Masni et al. 2020; Ranasinghe and Zampieri 2020). F1 score is a common metric for effectiveness in classification problems, which aims to balance precision and recall. In binary (two label) classification, precision measures the proportion of predicted positive labels (true positives and false positives) are true positives. Recall measures how many of the true positive data items (true positive and false negative predictions) were classified correctly as positive (true positive). F1 score is the harmonic mean of precision and recall, and aims to balance these metrics to favour a model which avoids false positives but also misses as few positive examples.

In a multiclass context, the F1 score for each possible label is calculated separately and combined. There are different methods for combining these scores, including taking a global average (micro-F1) or taking an unweighted sum of per-category averages (macro-F!). In this experiment we will use weighted F1 score, which takes a weighted average of category F1 scores based on the category weighting (the proportion of the data represented by this category). The weighted F1 score for a dataset D with categories $\{c_1, c_2, ..., c_N\}$ can be calculated as:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\ score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Model	English Data		Multilingual Data	
	Accuracy	**Weighted F1**	**Accuracy**	**Weighted F1**
mBERT-base	**96.50±0.22**	**96.52±0.22**	**98.77±0.10**	**98.77±0.10**
XLM-RoBERTa-base	95.66±0.37	95.66±0.34	98.70±0.00	98.70±0.00

Table 5.1: *Accuracy and weighted F1 score (mean ± standard deviation) of models trained and tested on the Valurank public dataset, both in English only and translated into 5 additional languages, over 3 training runs. The most effective model for each dataset is denoted in* ***bold.***

Model	English Data		Multilingual Data	
	Accuracy	Weighted F1	Accuracy	Weighted F1
Stratified Sampling	17.72%±0.80	17.74±0.80	17.72%±0.80	17.74±0.80
Most Common	23.54%±0.00	8.97±0.00	23.54%±0.00	8.97±0.00
mBERT-base	65.78%±0.21	65.96±0.14	66.03%±0.51	66.00±0.48
XLM-RoBERTa-base	**74.78%±1.27**	**74.92±1.24**	**73.38%±0.00**	**73.34±0.00**

Table 5.2: *Accuracy and weighted F1 score (mean ± standard deviation) of stratified sampling and most common static baseline models, as well as multilingual models trained on the Valurank public dataset (English only and translated into 5 additional languages). All models were evaluated on the collected multilingual dataset, over 3 training runs. The most effective model for each dataset is denoted in* ***bold.***

$$Weighted\ F1 = \sum_{i=1}^{N} w_i \cdot F1\ score_i \qquad w_i = \frac{|D_{d \in c_i}|}{|D|}$$

Where TP, FP and FN are the number of true positive, false positive and false negative classifications. Each experiment will be repeated for three training runs, and the mean and standard deviation of the accuracy and weighted F1 scores will be used as metrics for the effectiveness of each model.

5.3.2 Classifying public data

Effectiveness of translation upsampling

Shown in Table 5.1 are the results of training and testing the multilingual models in the original English-only and augmented translated Valurank public dataset. The results show that all models achieve very high (> 95%) accuracy and F1 score for the public dataset. We can also see that both models achieve an increased performance of 2-3% on the larger translated dataset, reaching near-perfect (> 98%) accuracy and F1 score in both cases. In both cases, the mBERT model performs slightly better than the XLM-RoBERTa model, but there is very little difference (< 1%) in accuracy.

Transfer of models to real-world data

Table 5.2 shows the results of transferring the models trained on public data to real-world collected data. The trained multilingual models perform significantly better than the baseline models, but achieve significantly lower accuracies and F1 scores than when they are evaluated on the public datasets. The XLM-RoBERTa model seems to generalise more effectively on both training sets, achieving 7-9% better accuracy and F1 score than the corresponding mBERT model. In both models, there is no significant difference in how the models perform on real-world collected data when trained on English or multilingual data.

Model	Accuracy	Weighted F1
mBERT-base	87.81%±0.24	87.75±0.26
XLM-RoBERTa-base	**89.40%±0.15**	**89.34±0.17**

Table 5.3: Accuracy and weighted F1 score (mean ± standard deviation) of models trained and tested on the collected multilingual dataset, over 3 training runs. The most effective model for each dataset is denoted in ***bold.***

Actual label	Predicted label					
	Entertainment/ Arts	Sports	Politics	Science/ Technology	Business/ Finance	Health/ Welfare
Entertainment/Arts	**445**	6	1	19	4	6
Sports	6	**558**	0	3	2	0
Politics	5	1	**196**	11	48	15
Science/Technology	4	3	0	**385**	39	8
Business/Finance	6	3	8	55	**486**	8
Health/Welfare	3	2	0	14	14	**211**

Table 5.4: Confusion matrix for the XLM-RoBERTa-base on real-world multilingual test data. The matrix shows where the prediction errors lay, and is colour coded so that deeper reds indicate more common misclassifications (the green diagonal shows correct classifications)

5.3.3 Classifying real-world data

Table 5.3 shows the results of training and testing the multilingual models on the real-world collected dataset. There is little significant difference between the two multilingual models, with XLM-RoBERTa achieving slightly higher accuracy and F1 scores of 89.5%. Both models achieve a lower accuracy on the collected dataset than on either of the public datasets, but perform significantly better when trained on the same dataset than on the public dataset. Table 5.4 shows the confusion matrix for the best performing model (XLM-RoBERTA-base) on the real-world test data. We can see that the most common misclassifications are predicting Business/Finance as Science/Technology (55 occurrences), predicting Politics as Business/Finance (48 occurrences) and predicting Science/Technology as Business/Finance (39 occurrences).

Confusion matrix

5.3.4 Conclusions

Machine translation of news articles shows some promise as a means of upsampling annotated public data, yielding an improvement in classification accuracy and F1 score over training on the monolingual dataset (as shown in Table 5.1. For smaller datasets, this could be an effective way of synthesising much more training data to allow multilingual models to learn more effectively and achieve increased performance. Future research is required to determine if machine translation is as effective on multilingual data.

The large increase in performance from the baseline models when the multilingual models are trained on public data and evaluated on collected data (as shown in Table 5.2) suggests that the multilingual models have learned some patterns which still apply to real-world data. The decrease in accuracy and F1 score when generalising to real-world data is likely due to the differences in how news articles are categorised and written across languages. Even when translated into multiple other languages, the public dataset only captures the writing and categorisation conventions of English-language news and the exact translations do not generalise effectively to news sources in other languages. This suggests that for effective classification of news in minority languages, effort should be taken to obtain training data which is written in the language instead of relying on machine translation.

While performance on real-world datasets (shown in Table 5.3) remains high (89% accuracy) there is a significant decrease in performance in the models from the public dataset evaluation (shown in table 5.1. This is likely because real-world data is less simple to sort into one category than most publicly annotated data: category boundaries are often not clear and even human annotators may disagree on classifications, many articles could be considered to fit into multiple categories and some are even given multiple categories by the original news source. The model performs very well on Sports articles, where there is less overlap with other categories and hence less ambiguity, but often confuses Business/Finance, Politics and Science/Technology articles. In real-world articles, these subject areas often overlap: many political decisions affect the economy, and technology is often a topic of political debate. Future research could consider a more appropriate task for news article classification, such as multiclass classification with more labels, using human-annotated data.

5.4 Project Limitations

5.4.1 News article classification

Due to the lack of public labelled data for news article topic classification in many languages, in order to consider this task in a multilingual setting it was necessary to collect my own public data using RSS feeds and web scraping. The data collected is not annotated or reviewed by a human, and instead solely relies on the tagging of the news sources being used. This means that training data may be mislabelled or contain errors, and results achieved through experiments using these data may not be as valid as results obtained from manually annotated data.

Research by Wu and Dredze (2020) suggests that multilingual BERT is less effective in transferring its results to low-resource languages, and pre-training on a multilingual task instead of masked language modelling (MLM) would produce a better model for language transfer. Due to limited resources and time, this pre-training was not possible in this evaluation.

6 Conclusion

(TO BE COMPLETED)

6.1 Project Summary

6.2 Future Work

(TO BE ADDED)

6 Bibliography

M. A. Al-Masni, D.-H. Kim, and T.-S. Kim. Multiple skin lesions diagnostics via integrated deep convolutional networks for segmentation and classification. *Computer methods and programs in biomedicine*, 190:105351, 2020.

T. Alam, A. Khan, and F. Alam. Bangla text classification using transformers. *arXiv preprint arXiv:2011.04446*, 2020.

N. Collier, S. Doan, A. Kawazoe, R. M. Goodwin, M. Conway, Y. Tateno, Q.-H. Ngo, D. Dien, A. Kawtrakul, K. Takeuchi, et al. Biocaster: detecting public health rumors with a web-based text mining system. *Bioinformatics*, 24(24):2940–2941, 2008.

A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

T. Kudo and J. Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

W. Lan, Y. Chen, W. Xu, and A. Ritter. An empirical study of pre-trained transformers for arabic information extraction. *arXiv preprint arXiv:2004.14519*, 2020.

B. Li, Y. He, and W. Xu. Cross-lingual named entity recognition using parallel corpus: A new approach using xlm-roberta alignment. *arXiv preprint arXiv:2101.11112*, 2021.

Z. Meng, A. Okhmatovskaia, M. Polleri, Y. Shen, G. Powell, Z. Fu, I. Ganser, M. Zhang, N. B. King, D. Buckeridge, et al. Biocaster in 2021: automatic disease outbreaks detection from global news media. *Bioinformatics*, 38(18):4446–4448, 2022.

Office for National Statistics. Population density - census maps, ons, 2022. URL `https://www.ons.gov.uk/census/maps/choropleth/population/population-density/population-density/persons-per-square-kilometre`.

Our World in Data. Population density, 2022, 2022. URL `https://ourworldindata.org/grapher/population-density`.

T. Pires, E. Schlinger, and D. Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

T. Ranasinghe and M. Zampieri. Multilingual offensive language identification with cross-lingual embeddings. *arXiv preprint arXiv:2010.05324*, 2020.

D.-W. Seo and S.-Y. Shin. Methods using social media and search queries to predict infectious disease outbreaks. *Healthcare informatics research*, 23(4):343–348, 2017.

A. Villanova and O. Abdulmatin. valurank/news$_{a}rticles_{c}ategorization, 2022. URL.$

S. Wu and M. Dredze. Are all languages created equal in multilingual bert? *arXiv preprint arXiv:2005.09093*, 2020.