GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN

The present work was submitted to the Text Technology Lab at Goethe-Universität Frankfurt am Main

Geospatial visualization of Wikipedia news reports about armed conflicts

Bachelor's thesis by

Jan Frederik Bausch

Course of study:	Computer Science
Semester:	8
Matrikel-Id:	6588979
E-Mail:	s3398921@stud.uni-frankfurt.de

First examiner:	Prof. Dr. A. Mehler
Second examiner:	Prof. Dr. C. Chiarcos
Supervisor(s):	M.Sc. G. Abrami M.Sc. D. Baumartz

Due date:	September 30th, 2021

Frankfurt, September 24th, 2021

Erklärung

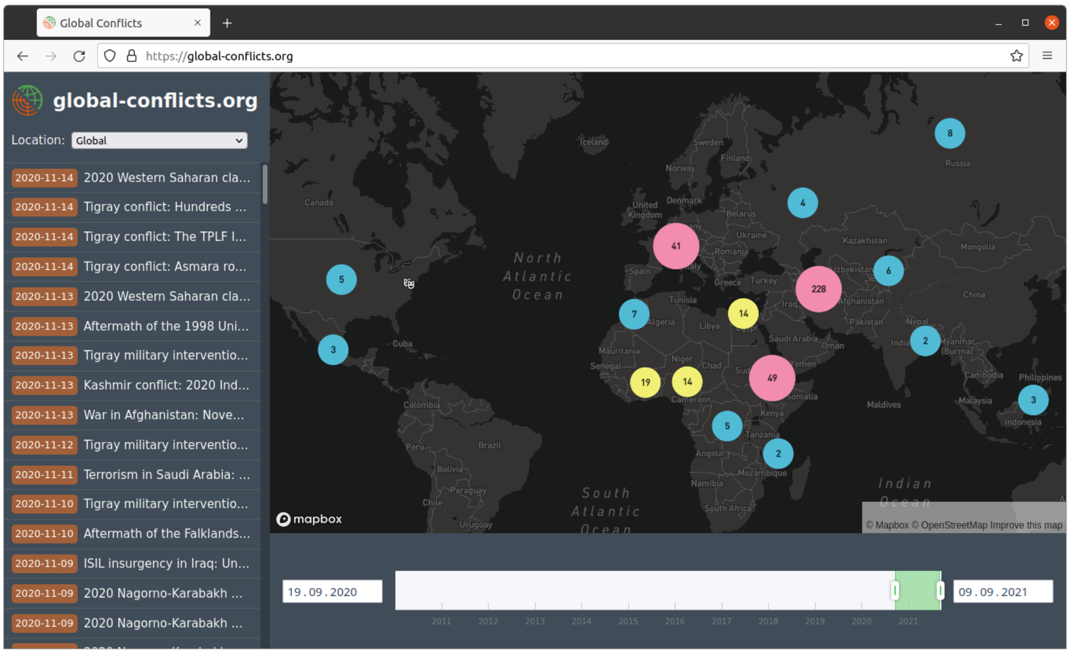
Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel verfasst habe.

Ebenso bestätige ich, dass diese Arbeit nicht, auch nicht auszugsweise, für eine andere Prüfung oder Studienleistung verwendet wurde.

Frankfurt, September 24, 2021 Jan Bausch

Abstract

This thesis presents a way to generate and visualize a geospatial dataset on armed conflicts from the Wikipedia Current Events Portal. We build a frontend web dashboard with React, Redux, and Mapbox - as well as a modular data processing pipeline that extracts additional geographical and casualty metadata with Wikidata, TextImager and OpenIE. An evaluation concludes that the dataset is mostly valid and accurate, but not complete in some edge-cases.



Location: Global

2020-11-14 2020 Western Saharan cla...
2020-11-14 Tigray conflict: Hundreds ...
2020-11-14 Tigray conflict: The TPLF l...
2020-11-14 Tigray conflict: Asmara ro...
2020-11-13 2020 Western Saharan cla...
2020-11-13 Aftermath of the 1998 Uni...
2020-11-13 Tigray military interventio...
2020-11-13 Kashmir conflict: 2020 Ind...
2020-11-13 War in Afghanistan: Nove...
2020-11-12 Tigray military interventio...
2020-11-11 Terrorism in Saudi Arabia: ...
2020-11-10 Tigray military interventio...
2020-11-10 Aftermath of the Falklands...
2020-11-09 ISIL insurgency in Iraq: Un...
2020-11-09 2020 Nagorno-Karabakh ...
2020-11-09 2020 Nagorno-Karabakh ...

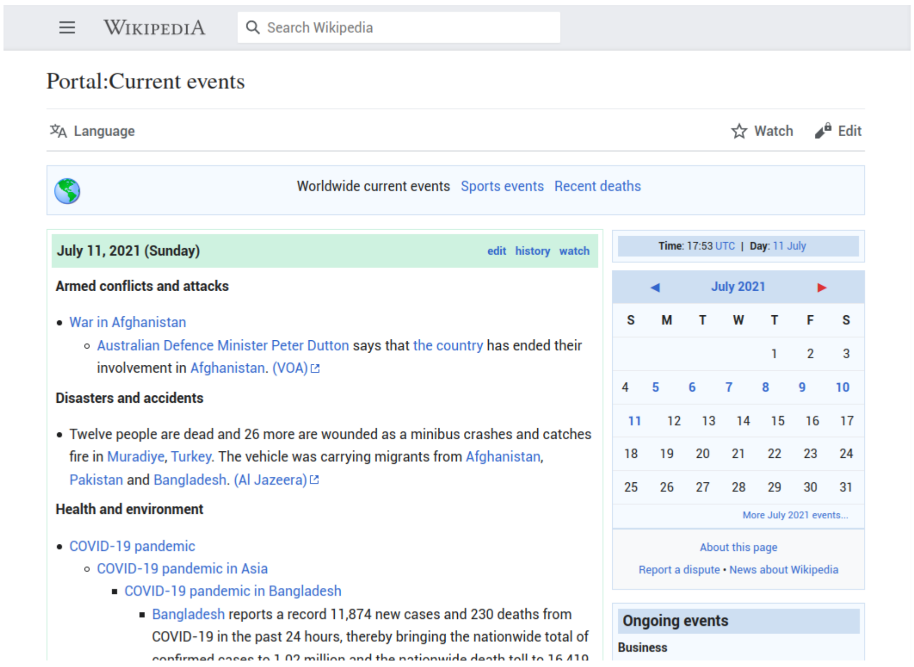
19.09.2020
09.09.2021

Figure 1: A live demo is available on `https://global-conflicts.org`

Contents

1 Introduction

Wikipedia is the world's largest encyclopedia (*Guiness World Records: Largest encyclopedia online*), providing public domain knowledge in 309 languages. Most of it is in form of written articles, currently 55 million and counting (*Wikipedia:Size comparison*). But there is a lesser-known side of Wikipedia, the *Current Events Portal* (*Wikipedia Current Events Portal*), which provides daily summaries of current affairs:



July 11, 2021 (Sunday)

Armed conflicts and attacks

- War in Afghanistan
  - Australian Defence Minister Peter Dutton says that the country has ended their involvement in Afghanistan. (VOA)

Disasters and accidents

- Twelve people are dead and 26 more are wounded as a minibus crashes and catches fire in Muradiye, Turkey. The vehicle was carrying migrants from Afghanistan, Pakistan and Bangladesh. (Al Jazeera)

Health and environment

- COVID-19 pandemic
  - COVID-19 pandemic in Asia
    - COVID-19 pandemic in Bangladesh
      - Bangladesh reports a record 11,874 new cases and 230 deaths from COVID-19 in the past 24 hours, thereby bringing the nationwide total of confirmed cases to 1.02 million and the nationwide death toll to 16,419

Figure 1.1: Screenshot of the Wikipedia Current Events Portal

The WCEP's archive of over ten years of news on politics, science, sports, arts and international conflicts capture a simple and yet informative perspective on recent world events. In plain text, this much information can be overwhelming. But it can be a rich source of information for advanced analyses.

For this research, the most interesting data in the WCEP is the news archive on armed conflicts and attacks. When consuming news it has almost become routine to hear about tragic wars and conflicts. It is common to brush aside the number of fatalities that occurred, parties involved and regions affected. Without context, those facts are not very meaningful. But a smarter representation of this data could be a helpful way to understand complex international relations and developments.

In this thesis, we will explore the news archive on global armed conflicts. Our goal is to provide a geospatial visualization of armed conflicts and attacks in the past ten years. In the end, we will design and implement an application to view and interact with relevant incidents.

1.1 Fundamentals

According to (*IBM: What is geospatial data?*), *geospatial data* is a combination of location information (like GPS coordinates), attribute information (characteristics of an object, event or phenomena) and temporal information (time or life span).

As been seen in the figure below, we, therefore, have to answer three questions: »*Where?*«, »*What?*« and »*When?*«

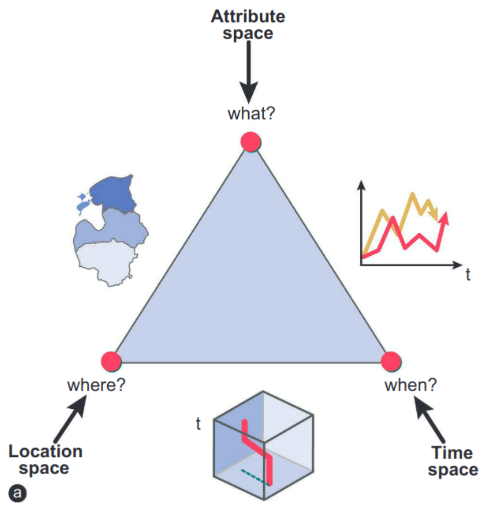


Figure 1.2: The characteristics of geospatial data, adopted from (Ormeling 2020)

In our case, the answer to »What?« is data on *armed conflict incidents*. Wikipedia guidelines (*List of ongoing armed conflicts*) refer to (Red Cross (ICRC) 2008) when defining that armed conflicts »consist in the use of armed force between two or more organized armed groups, governmental or non-governmental«. This includes interstate (between two or more states), intrastate (between governmental and non-governmental groups) and non-state (between non-governmental groups) conflicts.

In the context of our application, the »Where?« refers to the country and - if available - the detailed region of a conflict incident. The »When?« refers to the date when the particular incident occurred.

1.2 Research Questions

We will discuss the following research questions:

First, we want to find out **how to extract relevant data about armed conflicts**. To answer this, we need to understand how this information is structured and how we can access it in an automated way. This step is crucial because it affects all later stages of our project.

Secondly, we want to explore **how we can visualize this data**.

Lastly, we will **evaluate if our data is correct** by measuring the validity, accuracy and completeness. This allows us to verify our results.

2 Overview and related work

First, we have to answer the most consequential question: Where can we get data on armed conflicts in the past and present?
In this chapter, we will discuss different sources of information inside the Wikipedia cosmos and summarize other related scientific articles.

2.1 Gathering news from Wikipedia

Wikipedia is the well-known online encyclopedia used by many people regularly to research, look up facts or educate themselves.
It is part of a larger non-profit organization, the *Wikimedia Foundation*, which has the self-proclaimed goal:

»*Wikimedia is a global movement whose mission is to bring free educational content to the world.*« (*Wikimedia*)

Besides Wikipedia, the Wikimedia Foundation organizes and funds a variety of other public-domain projects:

Figure 2.1: Screenshot of `https://wikimedia.org`

Data collected in projects of the Wikimedia Foundation is public domain and designed to be consumed on the web and by software.
There are several different projects and initiatives which report news or store other data which is relevant to our use case:

2.1.1 Wikipedia

Wikipedia is an online encyclopedia with well-structured, neutral and verifiable articles on all relevant branches of knowledge. (*Wikipedia:Purpose*) Wikipedia articles are not necessarily updated on a daily basis and do not necessarily report on singular events like armed conflict incidents. For this reason, it is not a first-class source of information for this thesis.

2.1.2 Wikipedia Current Events Portal

The *Wikipedia Current Events Portal* provides news reports with short summaries, most of them only a sentence long.
The portal is available in multiple languages, but the English version is by far the most active. The first documented entries go back to as far as 2006. Later, historic events from earlier years have been added retrospectively. (*Current events: Revision history*)

The news items contain a link to the original newspaper articles and also to relevant background articles in Wikipedia. Most items are also categorized in one of the following topics:

- Armed conflicts and attacks,
- Arts and culture,
- Business and economy,
- Disasters and accidents,
- Health and environment,
- International relations,
- Law and crime,
- Politics and elections,
- Science and technology,
- Sports

Sometimes, multiple news items may belong to a larger news story that has unfolded over a period of time. In this case, the news items are categorized in a sub-list. Here is an example for a larger news story:

Politics and elections

- Georgian presidential election, 2018
    - Georgian citizens vote in the last presidential election to be decided by direct election. (RFE/RL)

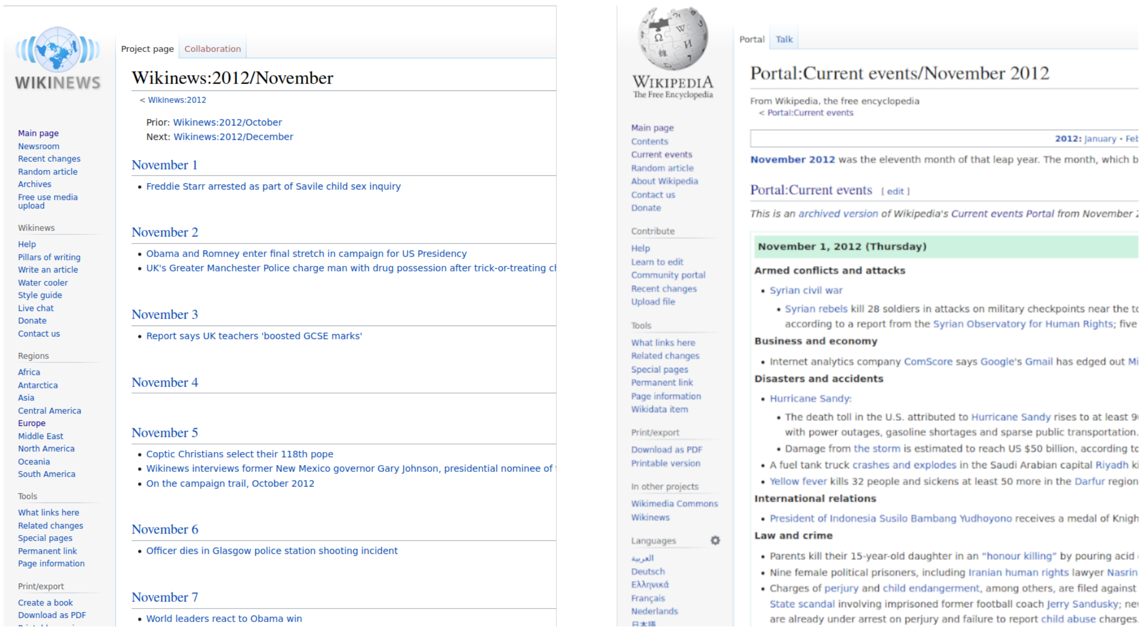
Because of the extensive archive, consistent writing style and rich metadata, the Wikipedia Current Events Portal is a suitable source of information for this project.

2.1.3 Wikinews

(*Wikinews*) is another news-related project in the Wikimedia Foundation. Its goal is to »present up-to-date, relevant, newsworthy and entertaining content without bias«. (*Wikinews:Mission statement*)
Wikinews is unique in the fact that it collects user-generated full-length articles. In contrast to that, the Current Events Portal contains short summaries by users of articles by established media sources.

But the Wikinews archive is missing a lot of news-worthy events. This is most likely due to the fact that it is difficult for voluntary contributors to write original articles with sufficient citations. And because the current user base cannot currently not compete with the global network of established news agencies.

For this reason, we will not use Wikinews in this project.



Wikinews:2012/November

< Wikinews:2012

Prior: Wikinews:2012/October
Next: Wikinews:2012/December

November 1

- Freddie Starr arrested as part of Savile child sex inquiry

November 2

- Obama and Romney enter final stretch in campaign for US Presidency
- UK's Greater Manchester Police charge man with drug possession after trick-or-treating cl

November 3

- Report says UK teachers 'boosted GCSE marks'

November 4

November 5

- Coptic Christians select their 118th pope
- Wikinews interviews former New Mexico governor Gary Johnson, presidential nominee of
- On the campaign trail, October 2012

November 6

- Officer dies in Glasgow police station shooting incident

November 7

- World leaders react to Obama win

Portal:Current events/November 2012

From Wikipedia, the free encyclopedia
< Portal:Current events

2012: January • Feb

November 2012 was the eleventh month of that leap year. The month, which b

Portal:Current events [edit]

This is an archived version of Wikipedia's Current events Portal from November

November 1, 2012 (Thursday)

Armed conflicts and attacks

- Syrian civil war
    - Syrian rebels kill 28 soldiers in attacks on military checkpoints near the to according to a report from the Syrian Observatory for Human Rights; five

Business and economy

- Internet analytics company ComScore says Google's Gmail has edged out Mi

Disasters and accidents

- Hurricane Sandy:
    - The death toll in the U.S. attributed to Hurricane Sandy rises to at least 9 with power outages, gasoline shortages and sparse public transportation.
    - Damage from the storm is estimated to reach US $50 billion, according to
- A fuel tank truck crashes and explodes in the Saudi Arabian capital Riyadh ki
- Yellow fever kills 32 people and sickens at least 50 more in the Darfur region

International relations

- President of Indonesia Susilo Bambang Yudhoyono receives a medal of Knigh

Law and crime

- Parents kill their 15-year-old daughter in an "honour killing" by pouring acid
- Nine female political prisoners, including Iranian human rights lawyer Nasrin
- Charges of perjury and child endangerment, among others, are filed against State scandal involving imprisoned former football coach Jerry Sandusky; ne are already under arrest on perjury and failure to report child abuse charges.

Figure 2.2: Left: Wikinews archive in November 2012. Right: WCEP archive in November 2012.

2.1.4 Wikidata

(*Wikidata*) is also part of the Wikimedia Foundation. It strives to be a knowledge-base for structured data, that can be read and edited by both humans and computers.

All of the information is stored as triples of *subject*, *predicate* and *object*. For example, the entity `Berlin` is linked with the predicate `capital` with another entity `Germany` (*Wikidata:Germany*). The data can be accessed through a query language called *SPARQL*, which is able to describe complex requests to for the Wikidata database.

Data stored in Wikidata is used directly in Wikipedia - most prominently in the *infobox* in many Wikipedia articles. (*Wikidata:Infobox*) It could be a good source of information for metadata related to armed conflicts, such as geographical locations.

2.2 Related work

We have decided that the Current Events Portal is the main source of information for this thesis. It has already been the subject of other scientific papers:

The article, »*Indexing and Analyzing Wikipedia's Current Events Portal, the Daily News Summaries by the Crowd*« (Tran and Alrifai 2014) provides statistics on the portal's news archive. Written in 2014, the authors analyzed that the number of monthly news items has been growing over the past years (see 2.3).

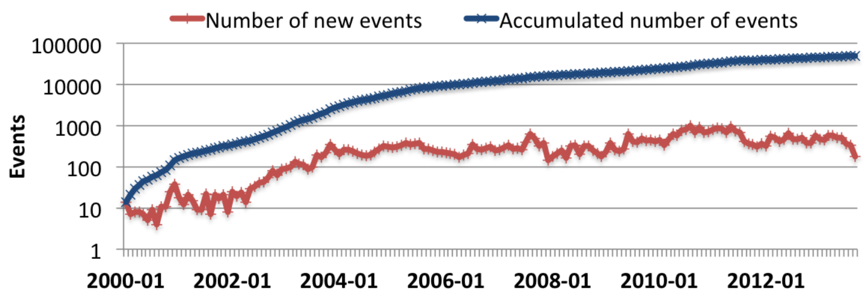


Figure 2.3: Number of events per month, adopted from (Tran and Alrifai 2014)

The authors also wrote that the categorization of news items has begun in 2010. Since then the most frequent categories are armed conflicts, politics, crimes and natural disasters (see 2.4).

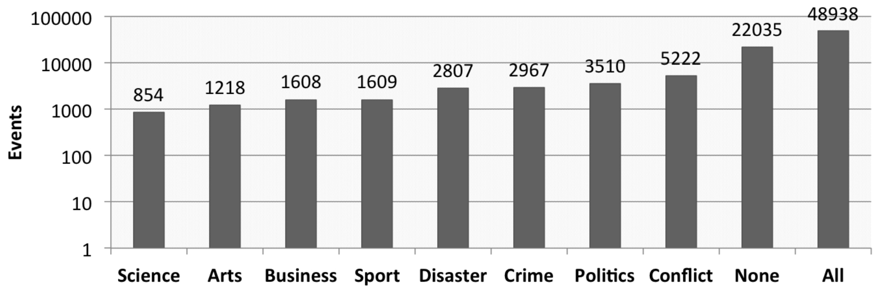


Figure 2.4: Distribution of events over major categories, adopted from (Tran and Alrifai 2014)

The paper also describes that the most-cited news agency is the BBC, followed by Reuters, CNN, Google News and Aljazeera.
Furthermore, it examines the authorship of news categories. It turned out that the armed conflicts category presents an exception, in that nearly half of reports are contributed by anonymous users (see 2.5).

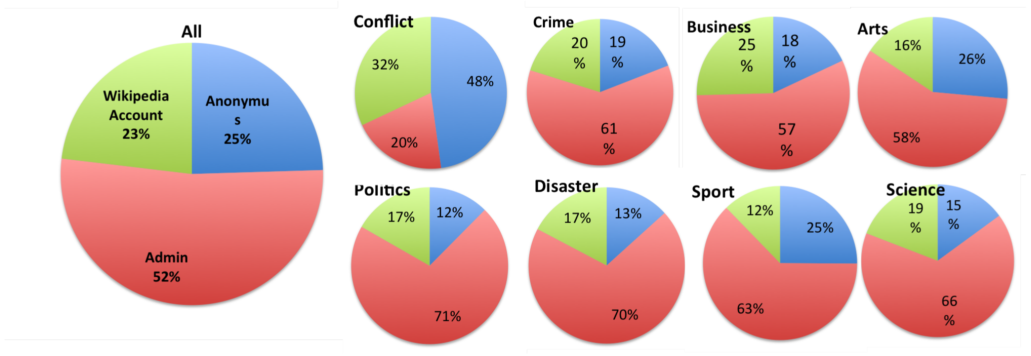


Figure 2.5: News content contribution of editors by account type, adopted from (Tran and Alrifai 2014)

Lastly, the paper presents the software system *WikiTimes*, which was built to extract and index news events from the Current Events Portal. However, the website, (*WikiTimes*), is currently not available anymore. Also, it wasn't possible to find a copy of the source code online.

A second paper, »*A Large-Scale Multi-Document Summarization Dataset from the Wikipedia Current Events Portal*« (Gholipour Ghalandari et al. 2020) has been published more recently in 2020. It describes building a dataset based on the Current Events Portal for *multi-document summarization* (short: MDS).

The goal of MDS is to generate short summaries from more than one large text document. Firstly, the dataset is generated by scraping news summaries from the Current Events Portal. Secondly, the media articles cited in the news summaries are also added to the dataset - on average about 1.2 sources per news item. In the last step, additional news articles are gathered from the Common Crawl News dataset (*Common Crawl News dataset*).

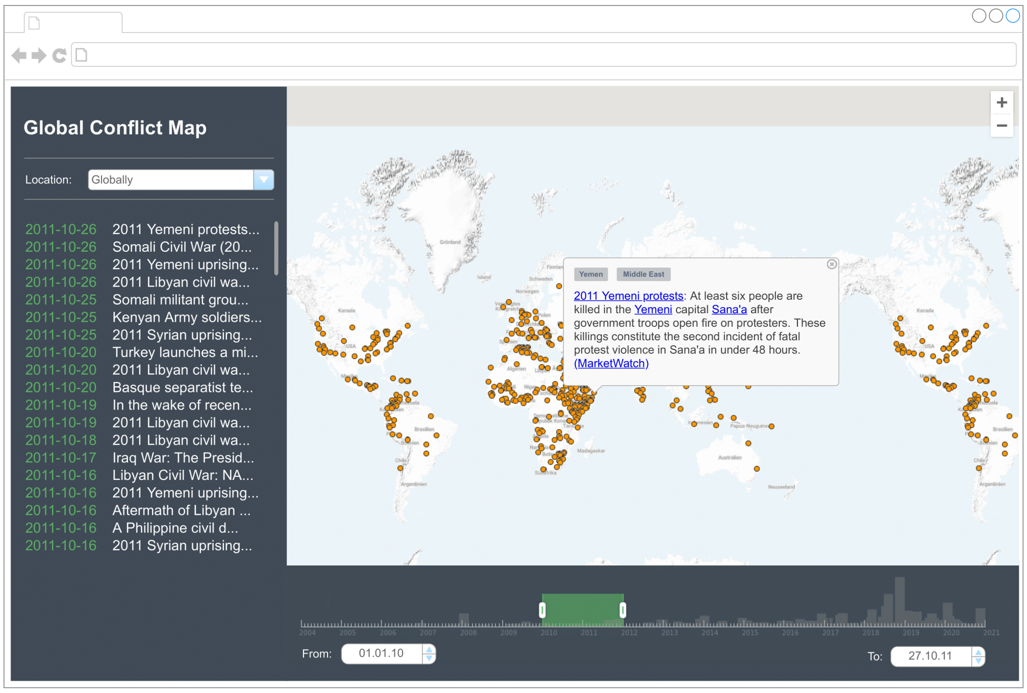
The authors evaluate how the dataset performs with different MDS methods and conclude that it is a realistic benchmark.

The data and software toolkit is available on Github (*WCEP Dataset*). But it is not suitable for this project because the current dataset is from October 2020 and cannot be updated easily. Furthermore, it is missing some news items, because it's not designed to be a complete index of the news archive, but rather a pool of sample data for benchmark tests.

3 Implementation

After having decided on a data source, we'll focus on how to build a data processing pipeline and frontend application. First, we need to specify the scope of our project.

3.1 Specification



Global Conflict Map

Location: Globally

2011-10-26	2011 Yemeni protests...
2011-10-26	Somali Civil War (20...
2011-10-26	2011 Yemeni uprising...
2011-10-26	2011 Libyan civil wa...
2011-10-25	Somali militant grou...
2011-10-25	Kenyan Army soldiers...
2011-10-25	2011 Syrian uprising...
2011-10-20	Turkey launches a mi...
2011-10-20	2011 Libyan civil wa...
2011-10-20	Basque separatist te...
2011-10-19	In the wake of recen...
2011-10-19	2011 Libyan civil wa...
2011-10-18	2011 Libyan civil wa...
2011-10-17	Iraq War: The Presid...
2011-10-16	Libyan Civil War: NA...
2011-10-16	2011 Yemeni uprising...
2011-10-16	Aftermath of Libyan ...
2011-10-16	A Philippine civil d...
2011-10-16	2011 Syrian uprising...

Yemen Middle East

2011 Yemeni protests: At least six people are killed in the Yemeni capital Sana'a after government troops open fire on protesters. These killings constitute the second incident of fatal protest violence in Sana'a in under 48 hours. (MarketWatch)

From: 01.01.10 To: 27.10.11

Figure 3.1: Mock-up of the application

The app should **visualize news reports** about armed conflicts, which are from the Wikipedia Current Events Portal. Each news report is assigned to a **date** and one or more **geographical locations**. Regions are retrieved from Wikipedia pages linked in a news report.

All news reports are listed in a **sidebar**. They can also be filtered for a specific geographical location in a search bar.

Markers are displayed on a map to represent a news item. If the user clicks on a marker on the map, it will be selected as the current region filter.

A **timeline** at the bottom of the page allows selecting a specific time frame. The selected region and time frame should affect the sidebar and map.

If the user clicks on a new report, a **pop-up** opens. It should show the full news report with links and other metadata.

3.2 Major Technologies

3.2.1 Javascript & React

Our goal is to have an interactive application that is easy to use. For this purpose, a web application is suitable, because a website can be accessed easily and runs on nearly every device. *Javascript* is the standard programming language for writing interactive web applications, together with *HTML* for markup and *CSS* for design and layout. In our case, we need to process and display large amounts of data with complicated interdependencies. Therefore the choice fell on (*React*), a web framework built on top of Javascript. React applications are component-based. This makes it easy to split a large application into manageable and reusable units. React is also declarative, which means that we only have defined a simple structure for each state of an application and React takes care of updating and rendering components when data changes. Later we will go into how we use these methods to our advantage.

3.2.2 Python

Besides the user-facing web application - also called *frontend* - we need a piece of software to provide relevant data in the background - generally called a *backend*. In web development, it is common to have a backend server, which can be accessed using an API, e.g. using the *REST* protocol. In our case, a backend server with an API is not necessarily required, because the data that we generate is static and is only updated manually or daily. Instead of a dynamic API, we will therefore develop a data pipeline that downloads and processes data from the Wikipedia Current Events portal and other sources. The final result of this pipeline is a JSON-encoded file that can be fed to the React frontend application. The advantage of this approach in comparison to a dynamic API is that it requires a lot less communication between the browser and the webserver. But it should be noted that it comes with the limitation that the generated file shouldn't exceed the memory and computing capacity of the user's device.

For the pipeline, we will use *Python* as a general-purpose scripting language. It is widely used and therefore has a lot of libraries for various purposes. It is also a commonly used language for data scientists working with large and complex data. This allows to develop new features quickly.

3.2.3 Java & TextImager

One important part of the data processing pipeline is to extract bits of information from plain text news reports. For example, in a news report »*Eight people died in a car bomb attack*« we want to extract the number of casualties (= `8`) in a machine-readable format. This can be achieved using *natural language processing*. For this, we use *TextImager* (Hemati, Uslu, and Mehler 2016) - a distributed system that allows to process large amounts of texts and to visualize the results in a user-friendly GUI. (*TextImager*) TextImager is based on (*Apache UIMA*) and is therefore written in *Java*. It can be extended by writing modules. Already, a variety of plugins for natural language processing are available which can be consumed through a web interface or REST API. (*texttechnologylab/textimager-uima*) Later, we will understand how to write a TextImager module.

3.3 Software Design

When a piece of software becomes larger and more complex, it is more likely that bugs stay unnoticed and it is more difficult to implement new features. This is because one small modification in one part of the codebase could have effects on other parts of the system as well.

Software Design is the process of thinking about the structure and data flow of a software system. It should provide a plan for the actual implementation by a developer. Good software design helps to write code faster, reduces errors due to unwanted side-effects and allows for later enhancements.

When designing the architecture we will focus on the following aspects:

- **Modularity**: For our use-case, we have to utilize a variety of libraries and services. Complexity mainly arises from the integration of these components into our system. Therefore it is important to consider an architecture that allows combining components easily.
- **Flexibility**: For the same reason, having a variety of different libraries and services, the architecture also needs to be flexible to work with different types of data and technical limits. Therefore we have to be able to develop small prototypes to explore the limitations. The software architecture needs to be flexible enough to be compatible with both less and more complex modules.
- **Error tolerance**: Having a system with a large amount of (inter-)dependencies comes at the risk of instability. Consuming data from Wikipedia Current Events Portal is especially prone to failure because all of its content is user generated and the archive of news reports is years old with varying quality and formatting. As a result, the architecture has to be able to handle errors well.

To achieve all of these goals we follow the *Unix philosophy* (*Basics of the Unix Philosophy*). This means that we split the application into small tools which are only designed for a very specific job. For example, one tool could be responsible for scraping Wikipedia articles, another for parsing Wikitext syntax. According to the Unix philosophy, the output of one tool could be the input of another. For example, the input of the parser program could be text from Wikitext, which is consumed from the output of the scraper program.

One benefit of this approach is that the Unix philosophy can easily be implemented using Python. We simply have to write small python scripts that read from the standard input stream and write to the standard output stream. This also makes it very easy to use existing Unix commands to debug scripts and save the output as a file on our computer.

In the following diagram we can see the Unix philosophy in action. It describes the architecture of our pipeline which is composed out of several small tools. In the end, the pipeline produces a JSON file which can be consumed by the frontend:

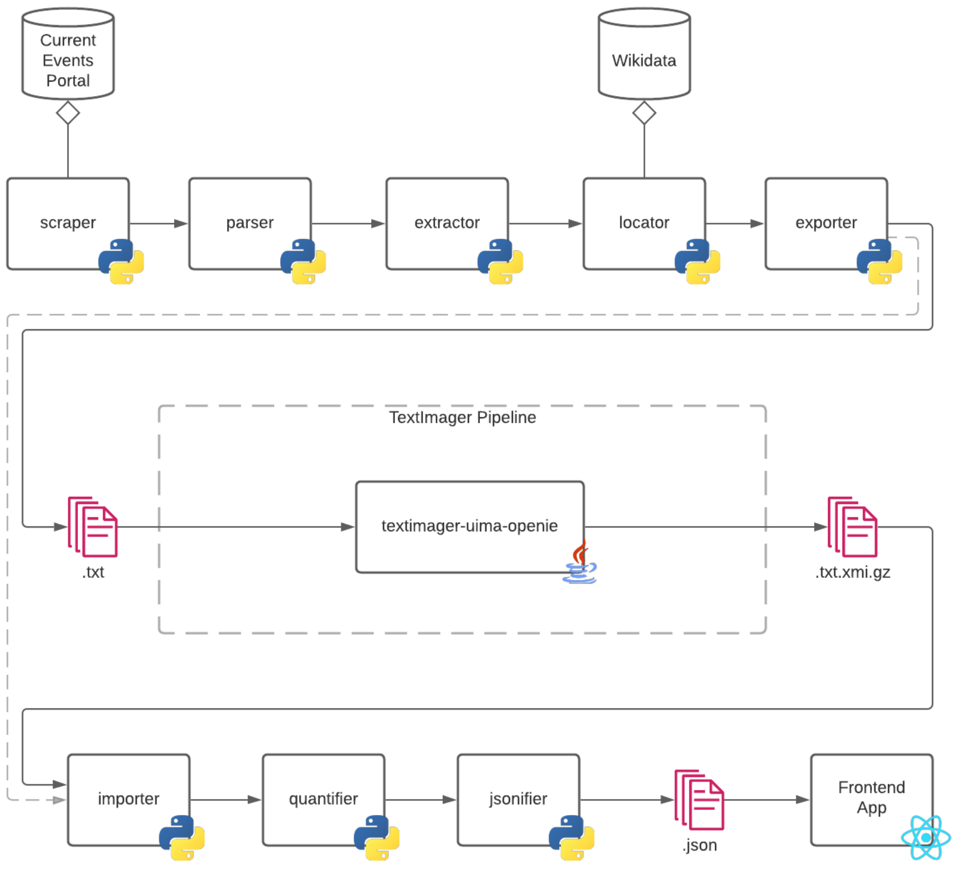


Figure 3.2: System architecture

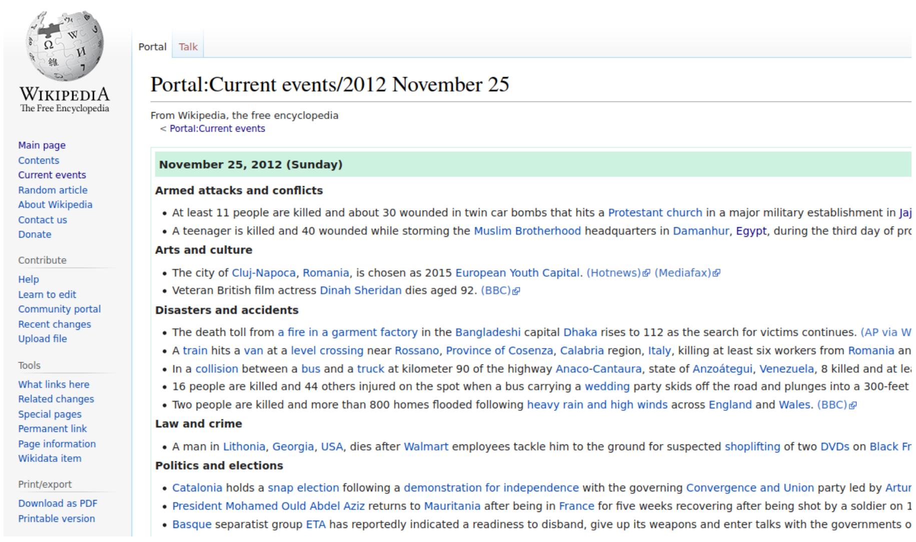
3.4 Pipeline Components

Let us go into detail about how each component in the pipeline works.

3.4.1 Scraper

The `scraper.py` component downloads pages from the Wikipedia Current Events Portal. As an argument, it takes a start and end date, in which time interval it will scrape articles.

In the WCEP, news reports are archived on a daily basis. For example, here is the page for the November 25th, 2012:



Portal:Current events/2012 November 25

From Wikipedia, the free encyclopedia
< Portal:Current events

November 25, 2012 (Sunday)

Armed attacks and conflicts

- At least 11 people are killed and about 30 wounded in twin car bombs that hits a Protestant church in a major military establishment in Jaj
- A teenager is killed and 40 wounded while storming the Muslim Brotherhood headquarters in Damanhur, Egypt, during the third day of prc

Arts and culture

- The city of Cluj-Napoca, Romania, is chosen as 2015 European Youth Capital. (Hotnews) (Mediafax)
- Veteran British film actress Dinah Sheridan dies aged 92. (BBC)

Disasters and accidents

- The death toll from a fire in a garment factory in the Bangladeshi capital Dhaka rises to 112 as the search for victims continues. (AP via W
- A train hits a van at a level crossing near Rossano, Province of Cosenza, Calabria region, Italy, killing at least six workers from Romania an
- In a collision between a bus and a truck at kilometer 90 of the highway Anaco-Cantaura, state of Anzoátegui, Venezuela, 8 killed and at le
- 16 people are killed and 44 others injured on the spot when a bus carrying a wedding party skids off the road and plunges into a 300-feet
- Two people are killed and more than 800 homes flooded following heavy rain and high winds across England and Wales. (BBC)

Law and crime

- A man in Lithonia, Georgia, USA, dies after Walmart employees tackle him to the ground for suspected shoplifting of two DVDs on Black Fr

Politics and elections

- Catalonia holds a snap election following a demonstration for independence with the governing Convergence and Union party led by Artur
- President Mohamed Ould Abdel Aziz returns to Mauritania after being in France for five weeks recovering after being shot by a soldier on 1
- Basque separatist group ETA has reportedly indicated a readiness to disband, give up its weapons and enter talks with the governments o

Figure 3.3: `https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012_November_25`

For later processing, it is important to preserve the document structure of each page, i.e. to extract links and categories. It is quite easy to download the source code of a Wikipedia page, simply by adding the prefix `'?action=raw'` to a given URL. All Wikipedia pages are written in the Wikitext syntax (sometimes also referred to as Wikicode). The scraper only takes care of downloading these source files and printing them to the standard output stream. Here we see the Unix philosophy in full effect: Each component is designed for a specific task.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-27
```

Example output (shortened):

```
{"date": "2012-11-25", "source": "<!-- All news items below this line -->{{
  ↪ Current events header|2012|11|25}}\n\n;Armed attacks and conflicts\n*At
  ↪ least 11 people are killed and about 30 wounded in twin car bombs that
  ↪ hits a [[Protestant church]] in a major military establishment in [[Jaji
  ↪ , Nigeria|Jaji]], [[Kaduna State]] [...]", "url": "https://en.wikipedia.
  ↪ org/wiki/Portal%3aCurrent_events/2012_November_25"}
{"date": "2012-11-26", "source": "<!-- All news items below this line -->{{
  ↪ Current events header|2012|11|26}}\n\n;Armed attacks and conflicts\n*[[
  ↪ Syrian civil war]]:\n**A [[Syrian government]] [[jet aircraft]] drops a
  ↪ [[cluster bomb]] on a [[playground]] in the village of [...]", "url": "
  ↪ https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012_November_26"}
```

```
{"date": "2012-11-27", "source": "<!-- All news items below this line -->{{
    ↪ Current events header|2012|11|27}}\n\n;Armed attacks and conflicts\n*At
    ↪ least 29 people are killed and 126 wounded in [[Iraqi insurgency (post U
    ↪ .S. withdrawal)|eight car bombings]] across [[Iraq]]. [http://www.cnn.
    ↪ com/2012/11/27/world/meast/iraq-violence/ (CNN)]\n*Gunmen attack a pub
    ↪ in central [[Nigeria]]'s [[Plateau State]] and open fire on customers,
    ↪ killing 10 people. [...]", "url": "https://en.wikipedia.org/wiki/Portal
    ↪ %3aCurrent_events/2012_November_27"}
```

Note that the output of the script is separated into one line for each day. Each line is a JSON-encoded object containing the date, Wikicode source and original URL. The next component in the pipeline can simply read from this stream and do further processing on it.

3.4.2 Parser

The `parser.js` script converts the Wikicode source files into a tree structure.

This is how a source file looks (shortened and formatted):

```
{{Current events header|2012|11|25}}
;Armed attacks and conflicts
*At least 11 people are killed and about 30 wounded in twin car bombs that hits
    ↪  a [[Protestant church]] in a major military establishment in [[Jaji,
    ↪ Nigeria|Jaji]], [[Kaduna State]], north central [[Nigeria]]. [https://
    ↪ www.reuters.com/article/2012/11/25/nigeria-bomb-idUSL5E8MP1SE20121125 (
    ↪ Reuters)]
*A teenager is killed and 40 wounded while storming the [[Muslim Brotherhood]]
    ↪ headquarters in [[Damanhur]], [[Egypt]], during the third day of
    ↪ protests as a result from the power grab by [[Mohamed Morsi]]. [http://
    ↪ news.nationalpost.com/2012/11/25/one-killed-40-injured-in-egypt-riot-
    ↪ president-mohammed-morsi-plans-to-meet-judges/ (National Post)] [http://
    ↪ www.cbc.ca/news/world/story/2012/11/25/egypt.html (CBC)]
;Arts and culture
*Veteran British film actress [[Dinah Sheridan]] dies aged 92. [https://www.bbc
    ↪ .co.uk/news/entertainment-arts-20486180 (BBC)]
;Disasters and accidents
*The death toll from [[2012 Dhaka fire|a fire in a garment factory]] in the [[
    ↪ Bangladesh]]i capital [[Dhaka]] rises to 112 as the search for victims
    ↪ continues. [http://hosted.ap.org/dynamic/stories/A/
    ↪ AS_BANGLADESH_FACTORY_FIRE (AP via WVEC)]
*Two people are killed and more than 800 homes flooded following [[2012 Great
    ↪ Britain and Ireland floods|heavy rain and high winds]] across [[England
    ↪ ]] and [[Wales]]. [https://www.bbc.co.uk/news/uk-20488645 (BBC)]
;Law and crime
*A man in [[Lithonia, Georgia|Lithonia]], [[Georgia (U.S. state)|Georgia]], [[
    ↪ United States|USA]], dies after [[Walmart]] employees tackle him to the
    ↪ ground for suspected [[shoplifting]] of two [[DVD]]s on [[Black Friday (
    ↪ shopping)|Black Friday]]. [http://gma.yahoo.com/blogs/abc-blogs/shoplift
    ↪ -suspect-dies-confrontation-walmart-workers-234946540--abc-news-
    ↪ topstories.html (Good Morning America)]
{{Current events footer}}
```

We will look at this example line by line:

```
{{Current events header|2012|11|25}}
```

This is Wikitext syntax for a template with the name `Current events header` with additional parameters. Templates define reusable code snippets - in this case displaying a header menu with a date.

```
;Armed attacks and conflicts
```

This defines a *term* in Wikitext as defined by the leading semicolons. In WCEP news reports, terms usually define categories. We are most interested in the »Armed attacks and conflicts« category, which we will filter out at a later stage.

```
*At least 11 people are killed and about 30 wounded in twin car bombs that hits
↪ a [[Protestant church]] in a major military establishment in [[Jaji,
↪ Nigeria|Jaji]], [[Kaduna State]], north central [[Nigeria]]. [https://
↪ www.reuters.com/article/2012/11/25/nigeria-bomb-idUSL5E8MP1SE20121125 (
↪ Reuters)]
```

A leading asterisk defines an unordered list. In WCEP news reports, list items contain individual news reports. When multiple news reports belong to the same event or topic, they are grouped into sub-items.

Single brackets, i.e. `[https://www.reuters.com/ (Reuters)]` define external links. In WCEP reports, they are used for links to original news sources. Double brackets, like `[[Kaduna State]]` are called free links and denote links to Wikipedia articles.

We do not have to parse this syntax manually. Instead, we use a Python library called (*WikiTextParser*), which offers an easy interface to extract lists, links and other elements from Wikitext files.

In the end, the parser component returns a stream of JSON objects. Instead of the original Wikitext syntax, the news reports are converted into a nested tree-like array.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py
```

Example output (shortened):

```
{"date": "2012-11-25", "url": "https://en.wikipedia.org/wiki/Portal%3
↪ aCurrent_events/2012_November_25", "items": [{"item": "Armed attacks and
↪ conflicts", "subitems": ["At least 11 people are killed and about 30
↪ wounded in twin car bombs that hits a [[Protestant church]] in a major
↪ military establishment in [[Jaji, Nigeria|Jaji]], [[Kaduna State]],
↪ north central [[Nigeria]]. [https://www.reuters.com/article/2012/11/25/
↪ nigeria-bomb-idUSL5E8MP1SE20121125 (Reuters)]", "A teenager is killed
↪ and 40 wounded while storming the [[Muslim Brotherhood]] headquarters in
↪ [[Damanhur]], [[Egypt]], during the third day of protests as a result
↪ from the power grab by [[Mohamed Morsi]]. [http://news.nationalpost.com
```

```
    ↪ /2012/11/25/one-killed-40-injured-in-egypt-riot-president-mohammed-morsi
    ↪ -plans-to-meet-judges/ (National Post)] [http://www.cbc.ca/news/world/
    ↪ story/2012/11/25/egypt.html (CBC)]"]}, {"item": "Politics and elections
    ↪ ", "subitems": ["[[Catalonia]] holds a [[Catalonian parliamentary
    ↪ election, 2012|snap election]] following a [[2012 Catalan independence
    ↪ demonstration|demonstration for independence]] with the governing [[
    ↪ Convergence and Union]] party led by [[Artur Mas]] returning with a
    ↪ reduced number of seats. [http://edition.cnn.com/2012/11/25/world/europe
    ↪ /spain-catalonia-elections/?hpt=hp_c2 (CNN)] [https://www.telegraph.co.
    ↪ uk/news/worldnews/europe/spain/9702100/Catalonia-poll-Spains-unity-put-
    ↪ to-the-test-as-voters-take-step-towards-independence.html (''The
    ↪ Telegraph'')]", "[[President of Mauritania|President]] [[Mohamed Ould
    ↪ Abdel Aziz]] returns to [[Mauritania]] after being in [[France]] for
    ↪ five weeks recovering after being shot by a soldier on 13 October. [http
    ↪ ://www.aljazeera.com/news/africa/2012/11/20121124232662628666.html (Al
    ↪ Jazeera)]", "[[Basque people|Basque]] separatist group [[ETA (separatist
    ↪  group)|ETA]] has reportedly indicated a readiness to disband, give up
    ↪ its weapons and enter talks with the governments of [[France]] and [[
    ↪ Spain]]. [https://www.bbc.co.uk/news/world-europe-20482620 (BBC)]"]}]}
```

3.4.3 Extractor

We don't need to store all news reports, as we are only interested in news about armed conflicts. The `extractor.py` script will filter out a list of given categories and will convert them into single news items. This makes it easier to process these later and also will get rid of unnecessary data.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py | python3
    ↪ extractor.py "Armed attacks and conflicts" "Armed conflicts and attacks"
```

Note that we have provided two parameters, »Armed attacks and conflicts« and »Armed conflicts and attacks«. This is because the notation of the armed conflicts category in WCEP has changed over the years. This will cover both notations.

Example output:

```
{"key": "2012-11-25-49ddd0", "wikitext": "At least 11 people are killed and
    ↪ about 30 wounded in twin car bombs that hits a [[Protestant church]] in
    ↪ a major military establishment in [[Jaji, Nigeria|Jaji]], [[Kaduna State
    ↪ ]], north central [[Nigeria]]. [https://www.reuters.com/article
    ↪ /2012/11/25/nigeria-bomb-idUSL5E8MP1SE20121125 (Reuters)]", "plaintext":
    ↪  "At least 11 people are killed and about 30 wounded in twin car bombs
    ↪ that hits a Protestant church in a major military establishment in Jaji,
    ↪  Kaduna State, north central Nigeria. (Reuters)", "date": "2012-11-25",
    ↪ "url": "https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012
    ↪ _November_25"}
{"key": "2012-11-25-2ade04", "wikitext": "A teenager is killed and 40 wounded
    ↪ while storming the [[Muslim Brotherhood]] headquarters in [[Damanhur]],
    ↪ [[Egypt]], during the third day of protests as a result from the power
    ↪ grab by [[Mohamed Morsi]]. [http://news.nationalpost.com/2012/11/25/one-
    ↪ killed-40-injured-in-egypt-riot-president-mohammed-morsi-plans-to-meet-
    ↪ judges/ (National Post)] [http://www.cbc.ca/news/world/story/2012/11/25/
    ↪ egypt.html (CBC)]", "plaintext": "A teenager is killed and 40 wounded
    ↪ while storming the Muslim Brotherhood headquarters in Damanhur, Egypt,
    ↪ during the third day of protests as a result from the power grab by
    ↪ Mohamed Morsi. (National Post) (CBC)", "date": "2012-11-25", "url": "
    ↪ https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012_November_25"}
```

As the output we now get a multitude of JSON objects, each containing one individual news report - together with additional metadata like date and URL. We also define an individual key and provide the report in both plain text and Wikitext. All of this data is consumed later in the pipeline.

3.4.4 Locator

The `locator.py` script will try to find geographic information for each news report. This is a crucial step for the geospatial visualization in the web app.

In nearly all news reports geographic locations are somewhat specified. For example, a village, like »Jaji, Nigeria« or a country like »Iraq«. Sometimes reports only reference governmental bodies like »Syrian government« or people like »the U.S. President Barack Obama«.

To our advantage, the reports in the WCEP contain links to related Wikipedia pages. In the first step, the `locator.py` script extracts all links in a report, again using the library wikitextparser. Next, one approach could be to scrape all of the linked Wikipedia articles for geographic information. But there is a quicker method instead: Articles in Wikipedia are also stored in Wikidata as so-called *items*. And for many of these items, geographic information is also available.

We can fetch this information from Wikidata using the SPARQL query language:

```
# As the output we only need the gps point and country label name
SELECT ?point ?iso3code WHERE {
  # We define a list of Wikipedia article names to search for
  VALUES ?page {
    "Beijing"@en
    "Jaji, Nigeria"@en
    "Iraq"@en
    "Taliban insurgency"@en
  }
  # Here we search for items with a given Wikipedia url
  ?sitelink schema:about ?item;
            schema:isPartOf <https://en.wikipedia.org/>;
            schema:name ?page.
  # The property P625 defines the coordinate location
  Optional {
    ?item  p:P625 ?coordinate.
    ?coordinate ps:P625 ?point.
  }

  # The property P17 defines an associated country of an item
  Optional {
    ?item p:P17 ?historic_country.
    ?historic_country ps:P17 ?country.
    # Should not be instance of an historic state
    FILTER NOT EXISTS {?country wdt:P31 wd:Q3024240}
    ?country wdt:P298 ?iso3code
  }
}
```

Here is the Wikidata response:

point	iso3code
Point(7.56944444 10.82361111)	NGA
Point(43.0 33.0)	IRQ
Point(116.407526 39.90403)	CHN

Table 3.1: Response of SPARQL query

We store both the list of GPS locations as well as country ISO codes in the JSON object output.

In chapter 4 we will evaluate the validity of these results.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py | python3
    ↪ extractor.py "Armed attacks and conflicts" "Armed conflicts and attacks"
    ↪  | python3 locator.py
```

Example output (shortened):

```
{"key": "2012-11-25-49ddd0", "wikitext": "At least 11 people are killed and
    ↪ about 30 wounded in twin car bombs that hits a [[Protestant church]] in
    ↪ a major military establishment in [[Jaji, Nigeria|Jaji]], [[Kaduna State
    ↪ ]], north central [[Nigeria]]. [https://www.reuters.com/article
    ↪ /2012/11/25/nigeria-bomb-idUSL5E8MP1SE20121125 (Reuters)]", "plaintext":
    ↪  "At least 11 people are killed and about 30 wounded in twin car bombs
    ↪ that hits a Protestant church in a major military establishment in Jaji,
    ↪  Kaduna State, north central Nigeria. (Reuters)", "date": "2012-11-25",
    ↪ "url": "https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012
    ↪ _November_25", "coordinates": [[7.75, 10.333333333], [7.56944444,
    ↪ 10.82361111], [8.0, 9.0]], "regions": ["Nigeria"]}
```

3.4.5 Exporter

The `exporter.py` script will save the plain text reports as individual `.txt` files in a given directory. This will allow us to analyze these text messages with TextImager in the next step.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py | python3
    ↪ extractor.py "Armed attacks and conflicts" "Armed conflicts and attacks"
    ↪  | python3 locator.py | python3 exporter.py -o out/
```

Example output:

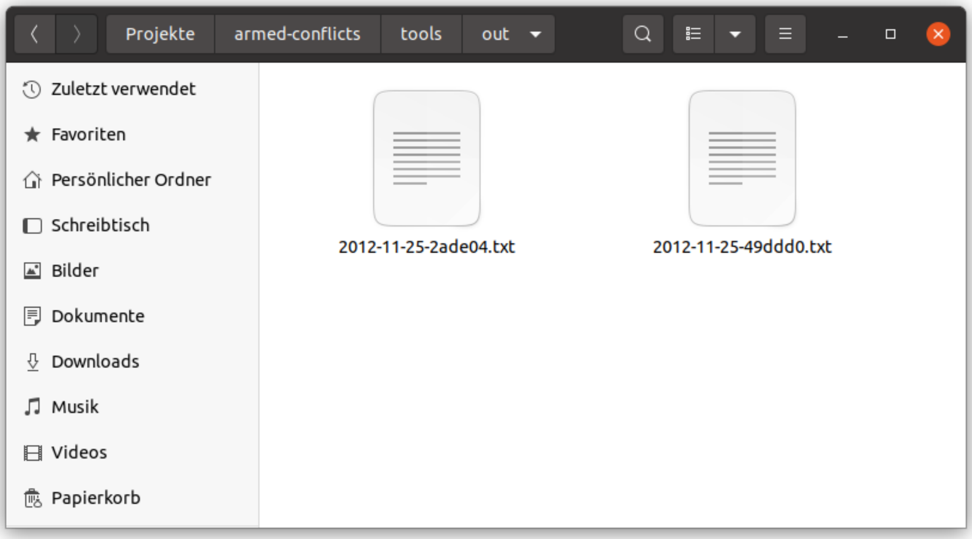


Figure 3.4: Content of the output folder

3.4.6 TextImager OpenIE Module

TextImager is a distributed service to analyze texts with various natural language processing tools. Either by sending individual text snippets to a REST API. Or, like in the case of our large dataset, by uploading files to a remote server which are then fed to TextImager.

At its core, TextImager is based on the Apache UIMA framework. It allows the construction of a pipeline of modules that can parse and annotate data. In the end, the annotated data can be exported to XML files.

We use TextImager to parse certain keywords from news reports which we need later to obtain the number of casualties. To archive this, we integrate the Open Information Extraction system (*OpenIE*) that extracts triplets of related parts of a sentence.

For example, in the the sentence »The U.S. president Barack Obama gave his speech on Tuesday to thousands of people.«, the following triplets are extracted:

- (`Barack Obama`, `is the president of`, `the U.S.`)
- (`Barack Obama`, `gave`, `his speech`)
- (`Barack Obama`, `gave his speech`, `on Tuesday`)
- (`Barack Obama`, `gave his speech`, `to thousands of people`)

To integrate OpenIE, we need to write our own TextImager module in Java. OpenIE 4 was preferred to later versions because it can be easily installed using Maven and does not require external model files.

As a result, our module adds annotations like the following example to the output XML file:

```
<type16:OpenIERelation xmi:id="2681" sofa="17" begin="23" end="74" confidence="
    ↪ 0.957242937750614" beginArg1="23" endArg1="59" valueArg1="Suspected Boko
    ↪  Haram suicide bombers" beginRel="60" endRel="64" valueRel="kill"
    ↪ beginArg2="65" endArg2="74" valueArg2="12 people"/>
```

Some minor tweaks were necessary to run the module successfully. OpenIE 4 requires an outdated version of a Maven package that is already used in other TextImager modules. To avoid version conflicts, the module had to be wrapped inside a Docker container.

Lastly, our module can be started through a REST endpoint, where the parameters `url` and `outputLocation` point to the designated input and output folders:

```
curl -k -X POST "https://textimager.hucompute.org/rest/big-data/analyse?url=%2
    ↪ Fresources%2Fcorpora%2FStudents%2Fba_bausch%2Ftest_in&language=en&
    ↪ inputFormat=TXT&nodepool=default&fileSuffix=txt&sortBySize=false&
    ↪ pipeline=OpenIEParser&process_memory_size=30&process_deployments_max=2&
    ↪ process_per_item_time_max=24000&outputFormat=XMI&outputLocation=%2
    ↪ Fresources%2Fcorpora%2FStudents%2Fba_bausch%2Ftest_out&outputCompression
    ↪ =GZIP&description=Wikipedia%20Tickermeldungen&modificationUser=s3398921&
    ↪ modificationComment=textimager" -H  "accept: application/json"
```

TextImager stores the output as Gzip-compressed XML files in a directory on the server for us to download.

3.4.7 Importer

The `importer.py` script will import the output files from TextImager. The files are stored as separate `.txt.xmi.gz` files in a given directory.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py | python3
    ↪ extractor.py "Armed attacks and conflicts" "Armed conflicts and attacks"
    ↪  | python3 locator.py | python3 exporter.py -o out/ | python3 importer.
    ↪ py -i in/
```

Example output (adjusted):

```
{"key": "2012-11-25-49ddd0", "wikitext": "At least 11 people are killed and
    ↪ about 30 wounded in twin car bombs that hits a [[Protestant church]] in
    ↪ a major military establishment in [[Jaji, Nigeria|Jaji]], [[Kaduna State
    ↪ ]], north central [[Nigeria]]. [https://www.reuters.com/article
    ↪ /2012/11/25/nigeria-bomb-idUSL5E8MP1SE20121125 (Reuters)]", "plaintext":
    ↪  "At least 11 people are killed and about 30 wounded in twin car bombs
    ↪ that hits a Protestant church in a major military establishment in Jaji,
    ↪  Kaduna State, north central Nigeria. (Reuters)", "date": "2012-11-25",
    ↪ "url": "https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012
    ↪ _November_25", "coordinates": [[10.333333333, 7.75], [10.82361111,
    ↪ 7.56944444], [9.0, 8.0]], "regions": ["NGA"], "relations": [{"arg1": "At
    ↪  least 11 people", "rel": "are killed", "arg2": "L:in twin car bombs"},
    ↪ {"arg1": "twin car bombs", "rel": "hits", "arg2": "a Protestant church
    ↪ in a major military establishment in Jaji"}]}
```

3.4.8 Quantifier

The `quantifier.py` script tries to extract the number of casualties from a report. From the previous steps, we have compiled a list of OpenIE triplets.

The script goes through the list of triplets and looks for certain keyword relations, like »killed by« or »kills«. If it finds an occurrence, it will try to extract a number from the argument of the relation. The number can either be in a numerical format like »14« or as string literals like »seven«.

This method exposes some flaws, as discussed more detailed in chapter 4. Still, it provides acceptable results.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py | python3
    ↪ extractor.py "Armed attacks and conflicts" "Armed conflicts and attacks"
    ↪  | python3 locator.py | python3 exporter.py -o out/ | python3 importer.
    ↪ py -i in/ | python3 quantifier.py
```

Example output (adjusted):

```
{"key": "2012-11-25-49ddd0", "wikitext": "At least 11 people are killed and
    ↪ about 30 wounded in twin car bombs that hits a [[Protestant church]] in
    ↪ a major military establishment in [[Jaji, Nigeria|Jaji]], [[Kaduna State
    ↪ ]], north central [[Nigeria]]. [https://www.reuters.com/article
    ↪ /2012/11/25/nigeria-bomb-idUSL5E8MP1SE20121125 (Reuters)]", "plaintext":
    ↪  "At least 11 people are killed and about 30 wounded in twin car bombs
    ↪ that hits a Protestant church in a major military establishment in Jaji,
    ↪  Kaduna State, north central Nigeria. (Reuters)", "date": "2012-11-25",
    ↪ "url": "https://en.wikipedia.org/wiki/Portal%3aCurrent_events/2012
    ↪ _November_25", "coordinates": [[10.333333333, 7.75], [10.82361111,
    ↪ 7.56944444], [9.0, 8.0]], "regions": ["NGA"], "casualty_description": "
    ↪ At least 11 people", "casualty_count": 11}
```

3.4.9 Jsonifier

As the final step of the pipeline, the `jsonifier.py` script will export all individual news reports to one JSON file. This file is consumed in the frontend web application.

Example command:

```
$ python3 scraper.py 2012-11-25 2012-11-25 | python3 parser.py | python3
    ↪ extractor.py "Armed attacks and conflicts" "Armed conflicts and attacks"
    ↪  | python3 locator.py | python3 exporter.py -o out/ | python3 importer.
    ↪ py -i in/ | python3 quantifier.py  | python3 jsonifier.py incidents.json
```

The resulting JSON file for 10 years worth of news is about 10 MB large.

3.5 Frontend

In the next step, we will present the data to the user. We will build a web application using the React framework and look into the challenges that arise when handling and interacting with a large and complex dataset like ours.

3.5.1 Modularization with React Components

When building complex interactive web applications, React is the framework of choice for many developers. One key feature of React is the ability to create user interfaces using reusable components which are only responsible for one designated purpose. A web page can then be composed out of many of these components.
Following the *Divide and conquer* (*OpenLearn*) approach helps reduce complexity as we only have to think about our application one piece at a time.

```
import React, { useCallback } from 'react';
import { setRegion, setIncident } from './redux/actions.js'
import useFilteredIncidents from './redux/useFilteredIncidents.js'

const IncidentListItem = ({ item }) => {

  const onSelectIncident = useCallback(({ target }) => {
    setIncident(item);
  }, [setIncident, item]);

  return (
    <div className="incident-list__item" onClick={onSelectIncident}>
      <span className="incident-list__item-timestamp">{item.date}</span>
      <span className="incident-list__item-preview">{item.plaintext}</span>
    </div>
  );
}

const IncidentList = () => {
  const incidents = useFilteredIncidents();

  return (
    <>
      <div className="incident-list">
        {
          incidents.map(item => <IncidentListItem key={item.key} item={item}
  ↪ />)
        }
      </div>
    </>
  );
}

export default IncidentList;
```

Here we define two components: A `IncidentList` component will render a div element containing a list of incident items. For that, we have defined a `IncidentListItem` component which will display the timestamp and text preview of the item passed as an argument.

Not only do we modularize the visual aspect of a component, we also separate logic. As can be seen in the example above, the `IncidentListItem` is defining a `onSelectIncident` event handler that is subscribed to the `onClick` event of an div element. When a user clicks on the element, the event gets triggered and calls a `setIncident` function.

From this short example we can see that React components make our code easier to understand, test and extend. The web application consists out of many of these components. Below is a diagram of the the component structure:

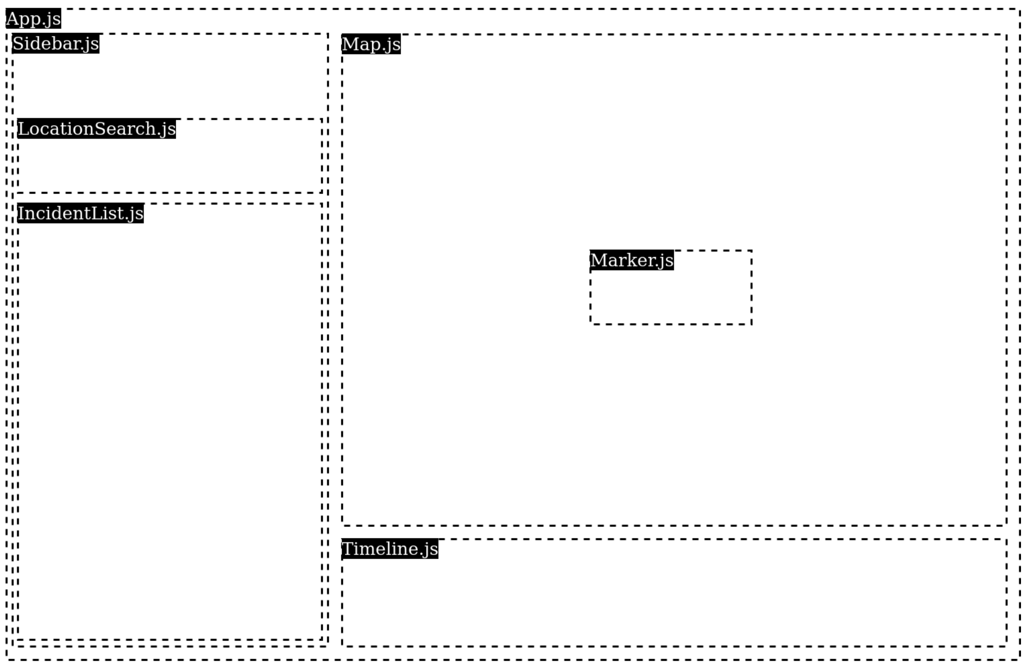


Figure 3.5: Wire-frame of React component structure

3.5.2 Managing state with Redux

Sometimes we need communication and logic between components. For example, when the user selects a specific region, this should be reflected in the sidebar and the markers on the map.

React offers different software patterns to achieve this. One approach is to pass data via component parameters (think of them as parameters of a function). Another approach is to register events, that can be subscribed to and triggered.

Both of these approaches are useful in some contexts. However, this project is characteristic in the fact that a lot of components share the same data - e.g. currently selected incidents. Therefore it would make more sense to store this data globally so that they are available to all components at the same time.

For this exact purpose, there is another framework called (*Redux*) which is built on top of React. The core principle of Redux is to have a global state which stores the information that is communicated across the application. Components then can access this information and render it to the user. If the state is updated, all relevant components are also updated to reflect the new changes. This is also beneficial to the performance of the application because irrelevant components do not need to be updated.

The initial state of the application is defined as follows:

```
import geodata from '../data/geodata.json';
import { incidents } from '../data/incidents.json';

const filtered_incidents = incidents.map((i) => ({...i, timestamp: new Date(i.
    ↪ date)})).reverse();

const timeline = {
  minDate: new Date('2010-01-01'),
  maxDate: new Date(),
  selectedStartDate: new Date(Date.now() - 1000*60*60*24 * 356),
  selectedEndDate: new Date()
};

const initialState = {
  geodata,
  timeline,
  incidents: filtered_incidents,
  selectedIncident: null,
  selectedRegion: 'global'
};

[...]
```

The initial state object contains various attributes: The `geodata` attribute imports a file containing geographical data about countries, which we display on the map. `timeline` defines a selected time span between January 2010 and the current date. `incidents` is the list of conflicts imported from our backend pipeline.

`selectedIncident` and `selectedRegion` reflect the current selected incident and geographical region. By default, »global« is selected.

Sometimes we also need to modify this state - e.g. when the user selects a different geographical region or time. To do this in Redux, we define *actions* that behave like events that can be triggered in components. All of this comes together in a so-called *reducer* function, which takes the initial state and currently triggered action and calculates a new state which is returned by the reducer function. (*Redux Reduer*)
Here is how the reducer looks in our application

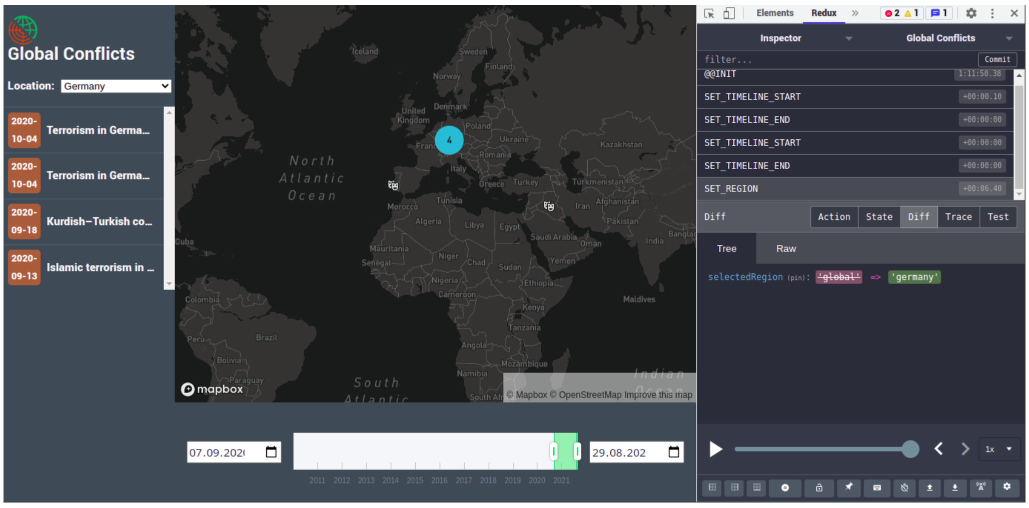
```
const reducer = (state = initialState, action) => {
  switch (action.type) {
    case 'SET_REGION':
      return { ...state, selectedRegion: action.region };
    case 'SET_INCIDENT':
      return { ...state, selectedIncident: action.incident };
    case 'SET_TIMELINE_START':
      return { ...state, timeline: { ...state.timeline, selectedStartDate:
    ↪ action.start } };
    case 'SET_TIMELINE_END':
      return { ...state, timeline: { ...state.timeline, selectedEndDate: action
    ↪ .end } };
    default:
      return state;
  }
}

export { reducer, initialState };
```

As an example, in line three we define an action called `SET_REGION`. If this action is triggered, we calculate a new state which is the same as the old state except that the selected region has been updated.

The Redux reducer is adopted from functional programming concepts. The *reduce* pattern - or sometimes also called *fold* - takes an array and executes a custom reducer function for every item of the array. The function returns a calculated value which is passed to the next item call. As a result, it returns a value that is modified by all items of the array. (*Mozilla: Reduce*)
In the context of Redux, this resulting value is the modified version of the initial state of the application that has been modified by all actions up to this point.

This is useful because it helps with debugging the application. Redux offers a browser extension (*Redux DevTools*) that visualizes the actions that have been called. Step by step, we can understand, replay and analyze the state of the application.



2020-10-04 Terrorism in Germa...
2020-10-04 Terrorism in Germa...
2020-09-18 Kurdish–Turkish co...
2020-09-13 Islamic terrorism in ...

```
@@INIT                  1:11:50.38
SET_TIMELINE_START      +00:00.10
SET_TIMELINE_END        +00:00:00
SET_TIMELINE_START      +00:00:00
SET_TIMELINE_END        +00:00:00
SET_REGION              +00:06.40
```

```
selectedRegion (pin): ~~'global'~~ => 'germany'
```

Figure 3.6: "Redux DevTools" browser extension

3.5.3 Geo-visualization with Mapbox

There are several providers of online maps: The best-known provider is (*Google Maps*), another public-domain option is (*OpenStreetMap*). Both offer an API, which allows embedding maps into an application or website. However, these APIs have limited features and don't allow for much customization. Especially third-party data can not be integrated easily.

As an alternative, the company (*Mapbox*) offers services and tools to create heavily customized maps. Mapbox also provides a Javascript library (*mapbox-gl*) that allows the integration of maps into web applications, makes them interactive and enriches them with data. It is a paid service but free for smaller amounts of traffic, and therefore suitable for this project.

To use Mapbox in our React application, we import the `mapbox-gl` library, define a map instance and add it to the React DOM.

Mapbox allows importing custom data sources. A `countries` source contains geo-data for all countries in the world, including their border. And a `incidents` source contains the list of armed conflict incidents together with their geographical location.

To add these sources to the map, we need to define `layers`. Layers also allow setting custom render rules to draw icons, circles and polygons on the map. For the `countries` source we define a new `countries` layer. For the `incidents` source we add three new layers `clusters`, `cluster-count` and `unclustered-point`. Mapbox allows defining clusters that are useful for datasets with large quantities of points. Points near each other are automatically merged into a cluster. In our case, we use this to render bubble-like circles that combine multiple points in the same geographical region. Frequently, points are located at an identical GPS coordinate. To be able to click on individual points, we add a small random delta to each value.

Mapbox allows registering events for each layer. For example, we register a `mouseenter` and `mouseleave` event for the unclustered-point layer to show a different mouse icon when we hover over a marker. We listen to double-click events on the country layer, to select a new country. And we listen to events when clicking on an incident marker, to open a pop-up with more detailed information.

In the last step, it requires a bit of additional work to make Mapbox compatible with React and Redux. Most notably we need to open a pop-up if a new incident is selected and update the incident geo-data source if the user changes the selected region or time span.

3.5.4 User-interface design

Another important part of an application is the user-face. Good design means, that an application shouldn't only look good, it should also be intuitive to use and behave transparently and coherently.
We will go through the top-level ideas that have shaped the design of this application.

At the core, this application is a dashboard for geographical and temporal data. Therefore the map and the sidebar with a list of armed conflict incidents should be in the user's focus.

We also handle a lot of information, but it would be overwhelming if we display all of it at once. Instead, we allow the user to filter a specific region and time span. If a user selects an incident, a pop-up will open to display more information. Also, there are multiple ways to achieve the same result. The region can be selected from a drop-down list or by clicking on the map. The time span can be adjusted in a date picker or by sliding on a timeline. The incident can be selected from the sidebar or the map. All of this allows the user to interact with the application in different contexts.

Colors are also important. In general, we go with a dark color scheme that is fitting to the content of armed conflict and attacks. Signal colors, like orange and green denote important data points. Less important information like locations labels on the map is grayed out.

We use modern CSS standards to build the user-interface, such as *flexbox* (*w3schools: Flexbox*) to create a layout of containers. We follow the *BEM* convention (*BEM*) to name CSS classes in a consistent way. Also, we optimize the user-interface to be compatible with smaller mobile device screens (»*Responsive Webdesign*«) (*Mozilla: Responsive Webdesign*).

It may be noted that websites should optimally be usable by people with visual or physical disabilities (»*Accessibility*«) (*Mozilla: Accessibility*). However, this was not implemented in the limited scope of this project.

3.5.5 Source code & Live Demo

The source code is managed with *Git* and is available on *Github* (*jfbausch/thesis-armed-conflicts*). Follow the *README* instructions on how to build the application yourself.

A live demo can be found on `https://global-conflicts.org`

4 Evaluation

In this section, we will evaluate the results of the implementation presented in the last chapter. We are interested in three metrics:

- Is the data **valid**?
- Is the data **accurate**?
- Is the data **complete**?

Data validity signifies if data conforms to a format or rule. For example, a region of a news report could be stored as »España« whereas it should be »Spain«. Accuracy refers to the degree to which information reflects reality. For example, if a news report is located in »Europe« whereas it should be »Belgium«. Completeness is the level to how much data is provided. For example, if a news report is assigned to »Germany« although, »Austria« and »Switzerland« would be more complete. (*Collibra*)

4.1 Generating a test dataset

To evaluate these metrics, we generate a test sample that we can analyze manually. We will select a number of random news reports from our dataset and export them to an Excel sheet.

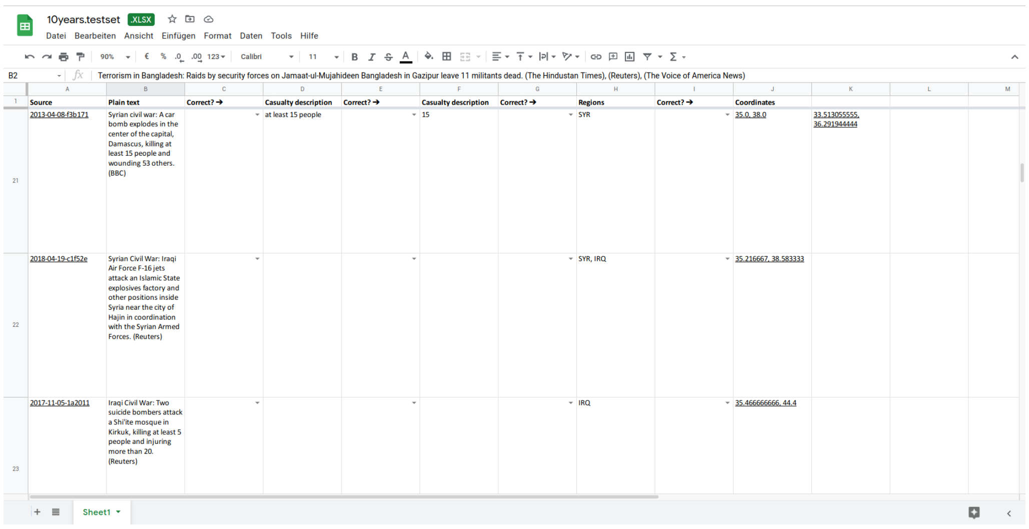
For that, we'll write a `tester.py` script which will automate this process. As an argument, it takes the number of news reports to be picked randomly. The following attributes are exported for each report:

- Source URL in the Wikipedia Current Events Portal
- Plain text news report
- Casualty description
- Casualty count
- Regions
- GPS-Coordinates (with a link to open the coordinates in OpenStreetMap quickly)

Here is how we run the script:

```
cat ../data/10years.quantifier.txt | python3 tester.py -n 50 ../data/10years.
    ↪ testset.xlsx
```

This is how the exported Excel sheet looks like:


10years.testset .XLSX
Datei Bearbeiten Ansicht Einfügen Format Daten Tools Hilfe
90% € % .0 .00 123 Calibri 11
Sheet1

B2 Terrorism in Bangladesh: Raids by security forces on Jamaat-ul-Mujahideen Bangladesh in Gazipur leave 11 militants dead. (The Hindustan Times), (Reuters), (The Voice of America News)

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Source	Plain text	Correct? →	Casualty description	Correct? →	Casualty description	Correct? →	Regions	Correct? →	Coordinates			
21	2013-04-08-f3b171	Syrian civil war: A car bomb explodes in the center of the capital, Damascus, killing at least 15 people and wounding 53 others. (BBC)		at least 15 people		15		SYR		35.0, 38.0	33.513055555, 36.291944444		
22	2018-04-19-c1f52e	Syrian Civil War: Iraqi Air Force F-16 jets attack an Islamic State explosives factory and other positions inside Syria near the city of Hajin in coordination with the Syrian Armed Forces. (Reuters)						SYR, IRQ		35.216667, 38.583333			
23	2017-11-05-1a2011	Iraqi Civil War: Two suicide bombers attack a Shi'ite mosque in Kirkuk, killing at least 5 people and injuring more than 20. (Reuters)						IRQ		35.466666666, 44.4			

Figure 4.1: Excel sheet of test set

4.2 Analyzing the test dataset

Next, we can analyze the generated Excel sheet step by step. It is formatted in a way so that we can mark each attribute as (in-)valid, (in-)accurate and (in-)complete. Our test dataset contains 50 entries with four generated attributes each - meaning that we have to manually score 200 data points.

The complete test set can be found as an attachment. We will go through some valid and invalid examples:

> »Syrian civil war: A car bomb explodes in the center of the capital, Damascus, killing at least 15 people and wounding 53 others. (BBC)«
> - Casualty description: »at least 15 people«
> - Casualty count: 15
> - Regions: SYR
> - Coordinates: (35.0, 38.0) (33.513055555, 36.291944444)

For the example above, all attributes are valid, accurate and complete.

»Syrian uprising (2011–present): 31 people are reportedly killed by the Syrian army, primarily in Homs. (Yahoo! News)«
- Casualty description: None
- Casualty count: None
- Regions: None
- Coordinates: None

In the upper example, all attributes are incomplete because the information is missing. This is because neither »Syrian army« nor »Homs« are linked to Wikipedia articles as they should. And because the phrase »are reportedly killed« is not in the list of filter keywords for casualties.

»Syrian Civil War: A roadside bomb in Syria kills one British and one U.S. soldier and injures five more coalition personnel. It is the first death of a British soldier fighting ISIL. (The Telegraph)«
- Casualty description: »one British and one U.S. soldier«
- Casualty count: 1
- Regions: SYR, USA, GBR
- Coordinates: (35.2166, 38.5833) (39.8281, -98.5795) (54.6, -2.0)

Here, the casualty description is extracted correctly, but the casualty count is inaccurate. This is because our algorithm returns the first number that it finds (»one«) and ignores everything else.

In the same way, we assess the rest of our data. After looking at the complete dataset, the analysis yields the following statistic:

	Casualty description	**Casualty count**	**Regions**	**Coordinates**
Valid	49	49	49	48
Accurate	50	49	48	48
Complete	35	34	34	33
All above	34	33	32	31

Table 4.1: Compiled statistics of test set sheet

The validity and accuracy of the data are quite high, with only one or two exceptions in a total of 50 entries. This means, that if a value is defined, we can assume that it is correct most of the time.

However, only about 66% to 70% of the entries are complete. Meaning that in the other third, some information is missing. This is due to the fact, that our pipeline applies rule-of-thumb filters and algorithms which are bad at detecting certain edge cases.

Interestingly, the metrics are very similar in comparison between the four attributes - although the casualties and geographical locations are extracted in completely different ways. One explanation for this could be that there are very simple entries on one side and very complex entries on the other, that are difficult to process for all kinds of algorithms.

5 Discussion

In the chapter, I will reflect on the previous work and provide my opinion on the tools, methodology and process used in this thesis.

I am satisfied with the technologies I've chosen for this project. Python turned out to be a very productive programming language for the backend pipeline. It was helpful to have extensive documentation, a large developer community and plenty of software packages on data-science-related fields. Javascript together with the React framework also was a good fit for an interactive web application, because it allowed me to write modular and declarative code. The combination with Redux to manage the application's state centrally was especially convenient for my use case.

The integration of TextImager helped me to leverage some performance-heavy tasks in a distributed cloud environment. Looking back, it also could have been an option to do more, if not all data processing on the TextImager servers. However this project involved a lot of exploratory prototyping, therefore it was critical to have a lightweight software architecture that could be easily refactored or extended.

As for data sources, the Wikipedia Current Events Portal was a great foundation for this thesis. Its news archive turned out to be extensive, neutral and well-maintained. The references to related Wikipedia articles also were a great starting point for further (automated) research.

Generating other metadata from the plain text news reports turned out to be very tedious and somewhat fault-prone. Querying geospatial data from Wikidata was easy, but more than once, the underlying references to entities were inaccurate or misleading. To improve this, more time should be spent on fine-tuning the Wikidata query. Extracting casualties worked less than optimal. OpenIE does a good job detecting relations in sentences. But the unnatural writing style of highly condensed news reports puts this to the test. The results could be better if the filter keyword list would be extended.

To me, the general topic of armed conflicts remained to be very interesting. During the development and writing of this thesis, two major international crises occurred between Israel and Palestine as well as in Afghanistan. It was captivating to see these events unfolding and to see the data in my application at the same time. This has motivated me to work on this thesis and similar projects in the future.

6 Outlook

6.1 Conclusion

This thesis presents a way to build a geospatial dataset on armed conflicts that are visualized in an interactive web application.

The Wikipedia Current Events Portal was chosen as a primary source, with a decades-long archive of user-generated news report summaries which are short, neutral, categorized and annotated.

We built a modular data processing pipeline to scrape, parse, structure and enrich individual conflict incidents. Additionally, we derived geographical information from Wikidata. Also, we integrated the natural language processing frameworks TextImager and OpenIE to extract the number of casualties from plain text reports.

The new dataset is used in a frontend web dashboard built with React. With the help of the Redux framework, we can efficiently and elegantly manage a large, interdependent application state. Mapbox provided us with the tools to visualize this data on a custom world map.

Lastly, we evaluated if the dataset is valid, accurate and complete. We found out that the validity and accuracy are quite high, while a significant portion of data is incomplete. This is because we designed our algorithms to work well for typical cases but not for certain edge cases.

In conclusion, we learned how we can present seemingly boring data interestingly and interactively. With more features, this proof-of-concept has the potential to turn into an instrument to understand intertwined international conflicts.

6.2 Future Work

As low-hanging fruits, more focus could be spent on fine-tuning the extraction of metadata, like geographical locations and casualties. This could be done by taking the edge-cases presented in chapter 4 into account and refactoring the Wikidata query and post-processing logic.

From there, the project could take different directions. Either, one could concentrate on the visualization. Optimally users should be intrigued to explore the information presented on the map. For example, by adding statistics, animations and links to further resources or by chronologically visualizing the timeline of a chain of related conflicts.

Or, one could concentrate on generating a reliable dataset on armed conflicts for further analysis. For example, one could train a machine learning model to predict how long a conflict might take or how many casualties it could cause.

Alternatively, it would be possible to move away from the limitation on armed conflicts. Instead one could geo-visualize the complete news archive of the Current Events Portal, including sports, arts, politics and other topics. Presenting news on a map would be a new way to follow current affairs in a globalized world.

Glossary

WCEP	Wikipedia Current Events Portal
GPS	Global Positioning System
SPARQL	SPARQL Protocol And RDF Query Language
MDS	Multi-document summarization
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
API	Application Programming Interface
REST	Representational State Transfer
JSON	JavaScript Object Notation
UIMA	Unstructured Information Management
URL	Uniform Resource Locator
ISO3	ISO 3166-1 alpha-3
OpenIE	Open Information Extraction
BEM	Block Element Modifier

List of Figures

List of Tables

Bibliography

Apache UIMA. https://uima.apache.org/. Accessed: 2021-07-24.

Basics of the Unix Philosophy. https://homepage.cs.uri.edu/~thenry/resources/unix_art/ch01s06.html. Accessed: 2021-07-24.

BEM. http://getbem.com/naming/. Accessed: 2021-07-24.

Collibra. https://www.collibra.com/blog/the-6-dimensions-of-data-quality. Accessed: 2021-07-24.

Common Crawl News dataset. https://commoncrawl.org/2016/10/news-dataset-available/. Accessed: 2021-09-05.

Current events: Revision history. https://en.wikipedia.org/w/index.php?title=Portal:Current_events&dir=prev&action=history. Accessed: 2021-07-21.

Gholipour Ghalandari, Demian et al. (July 2020). »A Large-Scale Multi-Document Summarization Dataset from the Wikipedia Current Events Portal«. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 1302–1308. DOI: 10.18653/v1/2020.acl-main.120. URL: https://aclanthology.org/2020.acl-main.120.

Google Maps. https://www.google.com/maps. Accessed: 2021-07-24.

Guiness World Records: Largest encyclopedia online. https://www.guinnessworldrecords.com/world-records/85651-largest-encyclopedia-online. Accessed: 2021-07-11.

Hemati, Wahed, Tolga Uslu, and Alexander Mehler (2016). »TextImager: a Distributed UIMA-based System for NLP«. In: *Proceedings of the COLING 2016 System Demonstrations*. Federated Conference on Computer Science and Information Systems. Osaka, Japan.

IBM: What is geospatial data? https://www.ibm.com/topics/geospatial-data. Accessed: 2021-09-04.

jfbausch/thesis-armed-conflicts. https://github.com/jfbausch/thesis-armed-conflicts. Accessed: 2021-07-24.

List of ongoing armed conflicts. https://en.wikipedia.org/wiki/List_of_ongoing_armed_conflicts. Accessed: 2021-09-04.

Mapbox. https://www.mapbox.com/. Accessed: 2021-07-24.

mapbox-gl. https://www.npmjs.com/package/mapbox-gl. Accessed: 2021-07-24.

Mozilla: Accessibility. https://developer.mozilla.org/en-US/docs/Web/Accessibility. Accessed: 2021-07-24.

Mozilla: Reduce. https://developer.mozilla.org/de/docs/Web/JavaScript/Reference/Global_Objects/Array/Reduce. Accessed: 2021-07-24.

Mozilla: Responsive Webdesign. https://developer.mozilla.org/en-US/docs/Glossary/Responsive_web_design. Accessed: 2021-07-24.

OpenIE. https://github.com/allenai/openie-standalone. Accessed: 2021-09-16.

OpenLearn. https://www.open.edu/openlearn/science-maths-technology/approaches-software-development/content-section-1.5. Accessed: 2021-07-24.

OpenStreetMap. https://www.openstreetmap.org/. Accessed: 2021-07-24.

Ormeling, Kraak (2020). »Cartography: Visualization of Geospatial Data (4th ed.)« In: *CRC Press*. DOI: 10.1201/9780429464195.

React. https://reactjs.org/. Accessed: 2021-07-24.

Red Cross (ICRC), International Committee of the (2008). »How is the Term "Armed Conflict" Defined in International Humanitarian Law?« In:

Redux. https://redux.js.org/. Accessed: 2021-07-24.

Redux DevTools. https://chrome.google.com/webstore/detail/redux-devtools/lmhkpmbekcpmknklioeibfkpmmfibljd. Accessed: 2021-07-24.

Redux Reduer. https://redux.js.org/usage/structuring-reducers/basic-reducer-structure. Accessed: 2021-07-24.

TextImager. https://www.texttechnologylab.org/textimager/. Accessed: 2021-07-24.

texttechnologylab/textimager-uima. `https://github.com/texttechnologylab/textimager-uima`. Accessed: 2021-07-24.

Tran, Giang and Mohammad Alrifai (Apr. 2014). »Indexing and Analyzing Wikipedia's Current Events Portal: The Daily News Summaries by the Crowd«. In: DOI: `10.1145/2567948.2576942`.

w3schools: Flexbox. `https://www.w3schools.com/css/css3_flexbox.asp`. Accessed: 2021-07-24.

WCEP Dataset. `https://github.com/complementizer/wcep-mds-dataset`. Accessed: 2021-09-05.

Wikidata. `https://www.wikidata.org/wiki/Wikidata:Main_Page`. Accessed: 2021-07-23.

Wikidata:Germany. `https://www.wikidata.org/wiki/Q183`. Accessed: 2021-07-23.

Wikidata:Infobox. `https://www.wikidata.org/wiki/Wikidata:Infobox_Tutorial`. Accessed: 2021-07-23.

Wikimedia. `https://www.wikimedia.org/`. Accessed: 2021-07-21.

Wikinews. `https://en.wikinews.org/wiki/Main_Page`. Accessed: 2021-07-21.

Wikinews:Mission statement. `https://en.wikinews.org/wiki/Wikinews:Mission_statement`. Accessed: 2021-07-21.

Wikipedia Current Events Portal. `https://en.wikipedia.org/wiki/Portal:Current_events`. Accessed: 2021-07-11.

Wikipedia:Purpose. `https://en.wikipedia.org/wiki/Wikipedia:Purpose`. Accessed: 2021-07-21.

Wikipedia:Size comparison. `https://en.wikipedia.org/wiki/Wikipedia:Size_comparisons#Wikipedia`. Accessed: 2021-07-11.

WikiTextParser. `https://pypi.org/project/wikitextparser/`. Accessed: 2021-07-24.

WikiTimes. `http://wikitimes.l3s.de/`. Accessed: 2021-09-05.