Bachelor's Thesis

June 17, 2020

Visualizing Business Landscapes Using Maps

Acquiring, maintaining, processing, clustering and visualizing business data from public sources

Timothée Wildhaber

of Zürich, Schweiz (13-751-532)

supervised by

Prof. Dr. Harald C. Gall

Dr. Carol V. Alexandru

University of Zurich[UZH]

s.e.a.l.

software evolution & architecture lab

Bachelor's Thesis

Visualizing Business Landscapes Using Maps

Acquiring, maintaining, processing, clustering and visualizing business data from public sources

Timothée Wildhaber

University of Zurich[UZH]

s.e.a.l.
software evolution & architecture lab

Bachelor's Thesis

Author: Timothée Wildhaber, timothee.wildhaber@uzh.ch

Project period: 01.12.2019 - 18.06.2020

Software Evolution & Architecture Lab
Department of Informatics, University of Zurich

Acknowledgements

I would like to thank the team at the Software Evolution and Architecture Lab (S.E.A.L.), Prof. Harald Gall, and especially Carol V. Alexandru for the opportunity to write my thesis at their research group as well as their valuable inputs. Additionally, I wish to thank my friends who supported me during this time of writing and programming.

Abstract

The Swiss business landscape is vast and diverse, making it difficult to quickly gain a high-level overview of companies and industries in Switzerland. This thesis investigates how publicly available resources can be utilized to facilitate such a bird's-eye view. Different sources were considered and multiple continuous data-scraping applications were created. To cluster the data, for example according to legal forms and business sectors, different methods such as machine learning and keyword mapping were applied with varying degrees of success. In working with the data, numerous deficiencies were identified in the sources publicly available, such as incomplete, outdated, and inaccurate records. Nonetheless, via extensive data cleaning, insights have been obtained and visualized to create an overview of the Swiss business landscape, also giving the possibility to find similar businesses given a search query. It is concluded that, while the developed visualizations provide a broad overview of Swiss businesses, a cleaner data set would have given more space for differentiated clustering as well as allowing for more creative visualizations.

Zusammenfassung

Die Schweizer Firmenlandschaft ist breit und divers. Dies macht es schwierig, einen abstrahierten Überblick der Betriebe und Industrien in der Schweiz zu gewinnen. Diese Arbeit untersucht, wie frei erhältliche, öffentlich einsehbare Quellen genutzt werden können, um eine Ansicht aus der Vogelperspektive zu ermöglichen. Verschiedene Quellen wurden in Betracht gezogen und mehrere Datensammelapplikationen wurden geschrieben. Um die Daten nach Merkmalen wie Geschäftsform und Geschäftsfeld zu gruppieren, wurden mehrere Methoden wie maschinelles Lernen und Schlüsselwortzuordnung mit unterschiedlichem Erfolg angewendet. Während der Arbeit mit den Daten aus öffentlichen Quellen wurden verschiedene Mängel identifiziert, wie zum Beispiel unvollständige, veraltete oder inkorrekte Aufzeichnungen. Trotzdem konnten mittels extensiver Datenbereinigung Einblicke gewonnen und eine Visualisierung der Schweizer Firmenlandschaft geschaffen werden, welche ebenfalls die Möglichkeit bietet, ähnliche Firmen durch eine Suchabfrage zu finden. Abschliessend kann gesagt werden, dass die entwickelte Visualisierung einen breiten Überblick verschafft, wobei jedoch ein saubereres Datenset mehr Möglichkeiten für verschiedene Gruppierungen und kreativere Visualisierungen geboten hätte.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Introduction

The latest data available from the Swiss Federal Statistical Office indicates that in 2017 the business landscape of Switzerland consisted of 590'253 companies. 99.7% of them are considered SMEs, small and medium-sized enterprises with under 250 employees. [str19] This means that per 100 people living in Switzerland, there are about seven companies. With so many different businesses, it is hard to keep track of the ongoing developments and changes, as companies go out of business and new ones are created. While data on Swiss companies is available, its readability and accessibility are severely limited. Information of the Swiss business registries is mostly shown in a way that only visualizes a single business, and does not take into account how they compare against others or how they are related to one another. While some commercially available comparisons exist, there is no existing platform that creates a bigger picture, visualizing a huge number of companies in a way that would be insightful for a consumer to use. Additionally, a new approach would create the possibility of gaining knowledge from analyzing user behavior during navigation through the visualization, which could later be used for further enhancement of the platform or even create the opportunity for financial gain.
The objectives pursued by this thesis can be split into multiple parts. The first goal includes finding and building a process to create and maintain a data set of businesses in Switzerland. Maintaining in this context means the ability to add new information about companies from the same source without rebuilding the whole database from scratch. The second goal is to build a repeatable way of analyzing the existing structures found in the fetched companies and enrich the information in the data set. Moreover, the third goal is to create a web application that can visualize the enriched data from the previous objectives in a new way. The visualization should deliver information about the usage of the web application. It is important to note that all those steps should be easily reproducible, and the data set maintainable through a trained user, allowing new data to be categorized.

1.1 Research Questions

With these goals in mind, it is possible to formulate the research questions for this thesis:

1. How can publicly available data on Swiss companies be obtained initially and continuously maintained afterward?

2. How and along which dimensions can Swiss companies be clustered with the openly available data?

3. How can the available data be visualized and user usage information gathered?

1.2 Related Work

While no directly comparable papers could be found at the time this thesis was written, some papers related to either one or two of the research questions are available. We can divide the found papers into three categories:

Visualizing Information About Businesses

Nowadays, big data is a commonly used term in the business world. The proceedings on how to handle a large set of information in a human-understandable and straightforward way often poses a challenge. In his article, Keahey explains different approaches to solve this problem, like radar charts, heatmaps and chord charts, and shows them in actual use cases. [Kea13] Qian shows in his paper how he clustered companies from the Zhejiang province in China by financial attributes with a K-Means clustering approach. This technique should help to identify creditworthy companies in this region. [Qia06] Yunzhao and Ye try to show some visualization techniques on the basis of the development of micro-credit companies in china. They use previously existing data from between 2011 and 2014 and deliver mostly pie and radar charts in their paper. [YY14] Sharawi and Sammour want to show how knowledge can be won by visualizing company data. They focus on a single company and the data load its inner workings produce. [SS17] Vliegen, Wijk, and van der Linden advocate the usage of modified treemaps in business-related data sets. They argue that with their adaptations, they can mimic known business graphics for substantial data sets. [Vvv06]

Gathering and Clustering Metadata

On the data gathering side, Yamamoto and Miyamura show how company relationships can be extracted from news articles found on the web. By using a Markov logic network, they built company pairs and clustered them in a bigger picture with the other found companies. In the end, they empirically confirm those findings. [YMNO17] Zhao et al. compare short texts from the social media platform Twitter[1], and traditional media by clustering them with topic modeling, using the `LDA` (Latent Dirichlet Allocation) algorithm to cluster the tweets. They also show the difference in tweets by topic on the basis of how opinionated they were. [ZJW+11] Hong and Davidson researched topic modeling on Twitter, present their pitfalls, and provide a way to improve the quality of the result. They also explain why author-topic modeling cannot be used to model hierarchical relationships between found entities in social media platforms. [HD10] Unrelated to business clustering, Xiaoping Sun used topic models to cluster documents and compared his performance against existing document clustering methods. He concluded that simple clustering methods based on topic modeling can be equally as effective as more sophisticated ones. [Sun14] On the topic of `TFIDF`, also known as term frequency–inverse document frequency, Qaiser and Ali examine the relevance of words to documents using this vectorization method. They discuss and summarize the weaknesses and strengths of this approach. [QA18] Jin applies another procedure in his paper; he introduces an algorithm that combines semantic analysis with a suffix tree clustering. He concluded that this kind of combination can lead to useful results when clustering news articles. [Jin12] Finished topic modeling libraries often deliver a proposition while not taking untrained users' feedback into account. Hu et al. present a solution in which user feedback can influence the result of the topic modeling. They call this method "interactive topic modeling". [HBGSS14] To handle a massive amount of unlabeled data, Gertures et al. developed their framework HDBSCAN, which can be used to work through vast data sets with just a small amount of labeled data. This approach is called semi-supervised clustering or semi-supervised

[1] `https://twitter.com`

classification. [GZSC19] On the other side of the spectrum, there is a qualitative example on how to divide businesses into different categories by qualitative factors without the help of machine learning. Hollenstein applied this method in his article focused on innovation indicators. He clustered companies into five classes; the sample size was around n=2731. [Hol03]

Map Visualization

The visualization of large data sets comes with caveats. To combat the loss of information caused by traditional visualization techniques, Gansner, Yu, and Kobourov introduced the "GMap" algorithm, which helps with the presentation of data in a more informative way. This visualization approach does work within different domains as they proved with their examples. [GHK10] On the geospatial site, Panse, Sips, Keim, and North show a framework that handles the pixel placement of geospatial data with respect to the global shape. Their goal was to avoid data loss by overplotting and to keep geospatial constraints. [PSKN06] A more abstract version of maps is built by Yen and Wu, with their centroid projection built upon self-organized maps. They apply a new projection method to produce 2D maps that can be interacted with. [YW08] High information density is a problem today, especially on the internet. Yang, Chen, and Hong present their solution, which helps by visualizing complex information in a 2-dimensional map. They find that their deployed approximation technique, called "fractal views", can improve the user performance in their tests, meaning a more helpful visualization to simplify information. [YCH03]

1.3 Thesis Overview

In the following chapters, we will go into detail about the thought processes behind company data acquisition, such as the origin of the data, a transparent explanation of how it was acquired, which selection of data was chosen to be further used, and why. We will also shed light on the implementation process and the problems encountered during the conception while giving an overview of the format and schematics. The same goes for the section data analysis, where the thesis demonstrates implementation details and explains why specific patterns and approaches worked or did not work as intended. In the following section, the thesis will explain the visualization process by going into detail about implementation. The results are discussed in chapter 3 while the implications of the results are presented in the discussion as part of chapter 4. Furthermore, we show the relation of the results to the goals of the thesis. In addition to this, possible future works around this bachelor thesis will be presented. The appendix contains abbreviations, code usage explanations, and insights into a user study created around the built web application.

Chapter 2

Approach

To get an overview over what exactly this thesis wants to solve, we need to split the problem into smaller parts. Therefore follows the well-known strategy of divide and conquer. The first level of division relates to the research questions asked in the introduction of this thesis. Those tasks, derived from the research questions, include:

1. Data acquisition
2. Data processing
3. Data visualization

These main tasks can be split further into smaller tasks. These subtasks define the structure of the thesis and also set the order of work.

2.1 Architectural Overview

To give a preceding overview of the entities and technologies involved, figure 2.1 shows an architectural diagram with the final revision of our approach. It shows the ecosystem, data sources such as Zefix and SHAB, as well as the written applications, including the data scraper, data transformation, and visualization/data delivery.

① The relevant IDs, the EHRA (Swiss Federal Office for the Commercial Registry) ID's, are fetched in `JSON` (JavaScript Object Notation). ② All data available from the SHAB is fetched to avoid making unnecessary calls to obtain a company for every ID ③. The steps related to data fetching can be found in section 2.2. ④ Data that has been downloaded or transformed is saved in the database for later retrieval The database works with `BSON` (Binary JSON), a binary data interchange format related to `JSON`. [incnda] ⑤ An array of scripts is used during the data analysis to generate new information from the existing data set. This is described in section 2.3. In the next step, section 2.4, it is described how the data is transferred to the web app as `JSON` ⑥ and visualized using `JavaScript`. Some of the used libraries include `D3`[1], `React`[2], and `Redux`[3]. ⑦ In section 2.5, it is described how user behavior is tracked when the client requests data via URL (Uniform Resource Locator) parameters. All data gathering and data delivery services are written in the `Golang`[4] programming language and communicate with the database over an abstraction layer for more accessible communication. The abstraction layer delivers the data from

[1] `https://d3js.org/`
[2] `https://reactjs.org/`
[3] `https://redux.js.org/`
[4] `https://golang.org/`

the database back to the application in `Golang` structs, which follow a similar principle as `C` structures. Data that comes from outside but is not fetched from an API (Application Programming Interface) is presented in ⑧. This includes a `CSV` (Comma-separated values) file mapping postal

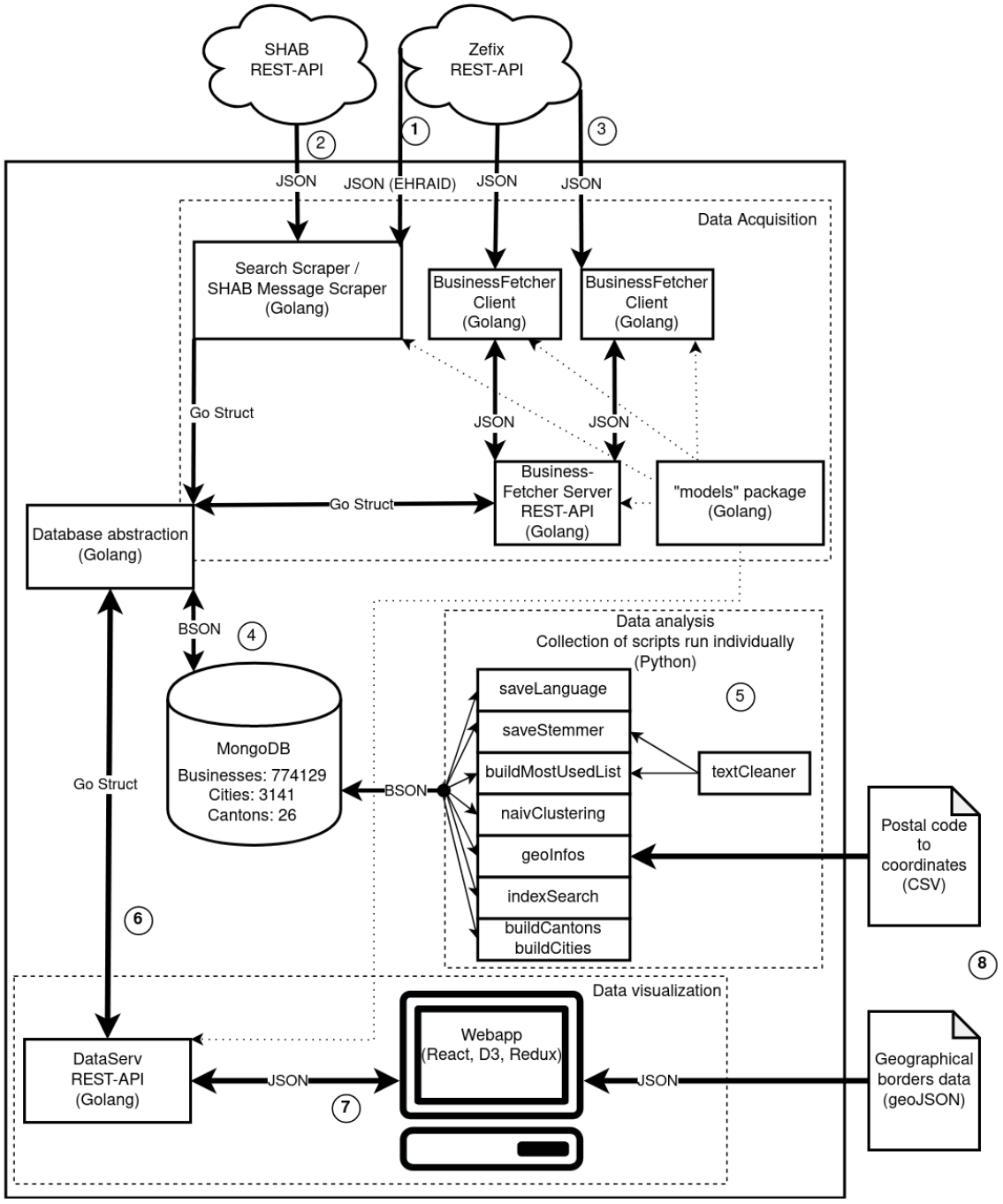

Figure 2.1: Architectural overview of the built tool

codes to geographic locations and a `JSON` file with geographical data for the map of Switzerland in our web app.
The mentioned elements of the diagram are explained in more details in their corresponding sections linked in the text.

2.2 Data Acquisition

Before any computation was done, data sources needed to be identified. After the selection, the available data was scraped from the identified sources. This section gives a detailed account of the sources that were found, the problems each source caused, and the prerequisites which must be met to scrape the data. A server-client architecture is introduced which allowed balancing the data fetching load on multiple clients. Finally, a way of maintaining this data set is presented, which should be easily usable by any trained user. Additional implementation details are given to help the reader understand what has been done to be able to answer the first of the three research questions.

2.2.1 Data Selection

To get a general overview of the information that can be obtained, there needed to be a way to identify companies. As there is no such thing as a 'list of all companies' in Switzerland, the focus was turned to the commercial register. This register is called "Zefix"[5] in Switzerland. It is a database of company identification values like the `ehraid`, an identification value consistent over all commerce registries in Switzerland, or the company name. Furthermore, it holds relevant data about businesses such as their legal form. [oJP20] It is important to note that while the database is organized centrally by the Swiss Federal Department of Justice and Police, the local registries themselfs are not: they work on a cantonal level, meaning that every canton in Switzerland saves, updates, and coordinates these records by themselves. The offices operate on a communal basis, and only the most necessary data is delivered to the national level where it enters the central database. [oJP20] Entries into the registry are published by the SHAB[6] (Swiss Official Gazette of Commerce), which is another source for the same information format: JSON encoded business structures with the same fields as the Zefix database. [SEC20] This data set appeared to be an excellent basis for this thesis as most companies have a purpose specified in this register, holding information on what their core business-activities include. However, this purpose string has no specified minimum number of characters as it seems. Due to the length of this data string being variable, some entities found while analyzing the purpose, had to be disqualified early on in the process. We found other reasons too that lead to the exclusion of businesses. Such disqualifying factors include an empty string for the purpose field, strings that only consist of numbers and no real workable text, and strings only containing specific sentences, which are unwanted due to their frequency of appearance in our data set. For the latter, a prime example would be a company with a business purpose only containing the following sentences:

> "Die Gesellschaft kann im In- und Ausland Zweigniederlassungen errichten. Kann sich bei anderen Unternehmen des In- und Auslandes beteiligen. Ebenfalls gleichartige oder verwandte Unternehmen erwerben oder errichten. Sowie alle Geschäfte eingehen, in denen Synergien mit dem Hauptzweck zu erzielen sind. Sie kann weiter Liegenschaften und Wertschriften erwerben, verwalten und verkaufen".

[5] `https://www.zefix.ch/en/search/entity/welcome`
[6] `https://www.shab.ch/#!/gazette`

which freely translates to

> "This company can open branches in this and other countries, participate in other companies on an equity basis in this other countries, acquire similar or related companies or establish them, as well as enter all deals that can score synergies with the main purpose of this company. It furthermore can acquire, administrate, and sell properties and shares."

These are common-purpose sentences widely used in many German texts analyzed during this thesis. These sentences seem to often be directly copied from each other or a common source.

At our inquiry for the data set, the official contact point of the Zefix answered promptly. They told us that it was not possible to get the database as a whole at once. They were not able to hand out the database in its entirety due to some unexplained constraints. However, there was the possibility to give out an `API` key to gain access to their official, well-documented `SOAP` (Simple Object Access Protocol) `API`, which allows the user to search for company names and other company specific values. The results are returned as IDs, which can then be queried one by one. This did prove to be a challenge due to the fact that the API is rate and entry limited, meaning the provided API key would be voided if too many requests were sent. Moreover, the search terms could not be too general as the search server would return at most 200 businesses from the register without the possibility of a cursor to get more results. However, due to a non-documented web API that the Zefix website itself uses, those limits were avoidable. It was possible to actually get the API URL by observing the Zefix website requests to its backend. In the same way, it was also possible to get the necessary parameters and JSON body for the `HTTP` call to their API. The endpoint for a single business would return a `JSON` response, as expected from a `REST API`. An example response can be seen in listing 2.1, additional explanations are found in section 2.3.1. The fields of the JSON object could then be mapped into `Golang` structs, by the `Golang` standard library for `JSON` encoding.

During the collection phase, we also contacted some firms specialized in company data gathering to acquire some of their data sets providing better information that we could later use to analyze and cluster companies. The companies in question include Moneyhouse[7], a Swiss-based data sampling company that is a subsidiary of the NZZ media group [AG20b] and Kompass,[8] a multinational data sampler with about 500 employees distributed around the world [AG20a], both of them are used by businesses in Switzerland. In our email exchanges, both companies made it abundantly clear that their data set is only for purchase, and their API is solely meant to be used to make specific requests and not to fetch bigger data sets at once. Exporting accumulations of data would cost a hefty sum.
Also worth mentioning is BurWeb[9], a database which can be queried using XML (Extensible Markup Language) to get similar, if not more extensive company data. However, for the purpose of this thesis, there was no possibility of getting access; the usage is limited to official government agencies like the RAV (Regional Employment Centre). [dB20] In the end, it was decided to go ahead with sourcing the Zefix data set due to its accessibility and cost. Moneyhouse and Kompass were left out for monetary reasons, as well as the prudence that getting enough data from them would prove too difficult due to their stance of not allowing the export of their whole data set, which was exactly what was needed. Furthermore, Moneyhouse has large parts of their data overlapping with Zefix, stemming from the fact that SHAB/Zefix is the only national, centralized source for information of this kind. [AG20c]

[7] `https://www.moneyhouse.ch/en/`
[8] `https://ch.kompass.com/`
[9] `https://www.burweb2.admin.ch/BurWeb/Login.aspx`

```
{
   ehraid: 0,
   purpose: "A short description of what a company does"
   name: "Company name",
   chid: "CH00000",
   uid: "CHE0000",
   legalSeatId: 3000,
   legalSeat: "Bern",
   registerOfficeId: 4,
   legalFormId: 1,
   status: "EXISTIEREND",
   rabId: 0000,
   shabDate: Date,
   address: {},
   translation: null,
   shabPub: [],
   mainOffices: [],
   furtherMainOffices: [],
   branchOffices: [],
   hasTakenOver: [],
   wasTakenOverBy: [],
   auditFirms: [],
   auditFirmFor: [],
   oldNames: [],
}
```

Listing 2.1: Sample company data from the Zefix API in JSON, enriched with dummy data

Additionally, a source for mapping postal codes to geographical coordinates was found [Rü15] and would enrich the data set accordingly.

2.2.2 Crawling / Scraping

The work to obtain the data was split into multiple sections. First, we needed to get all available company IDs, as this is the way the Zefix API works. All data the scripts could get was saved into an instance of `MongoDB`[10], a database that belongs to the family of non-relational databases. It does not require a scheme and is easy to set up with `Docker`.[11] For the development of the project, it was decided to use `MongoDB` due to the way it handles data. Quick development cycles can mean various changes to the database, the data mapping, changing field types, and other inconveniences a strict schema can conjure up. We can circumvent a high-maintenance problematic during this time of quick development, by using a database not concerned with the values of its fields and their existence. Another huge plus is that this database allows indexing of text and number fields, which can later be queried directly without the need for additional software. It is necessary to add that this project could benefit greatly from an SQL (Structured Query Language) database like `PostgreSQL`[12] once the development is over, as it is important to have a higher degree of security when it comes to data integrity.

Sampling Swiss Business IDs

Getting Swiss business IDs into our database was the first step to build a data set. This was done by querying the `REST API` of the Zefix register with wild card searches. Since the API needs at least three letters, including the wild card character "*", it was decided to go through the alphabet with changing characters in position two and three. This meant that an application formed a string that would include the wildcard character and two characters, which would change with the sent requests, going through all possible permutations of two letter from the alphabet with an asterisk as prefix. Eg. "*aa". A wildcard search with more than one asterisk would have returned more results per request but also meant that more time was needed to fetch the cursor while getting more duplicate IDs overall. A request with three wildcard characters alone, for example "***", was not possible, and a search request with this specific parameter got declined by the server. The script would iterate over the alphabet in a nested loop for the characters. Searching for "*aa" to "*az" and then continue the in first loop from "*ba" to "*bz", continuing like this until the end of the alphabet was reached. Every search would return a maximum of 200 business IDs with a cursor to continue fetching the IDs for this query. If the cursor was exhausted, the script would move onto the next search query. This method of searching would result in duplication as well, but was chosen because of the fact that more open wildcard searches like *a* would end up with more duplicate IDs and an approach without wildcard would require vast amounts of requests for the search itself, due to the fact that there needs to be a third loop to go through the alphabet. A single wildcard in the query allowed us to have a sure safeguard in case we could not match a substring correctly. There had to be a limitation to the requests that were sent by the program to avoid being blocked. In the case of Zefix, the script should not make more than 200 requests per 10 minutes. This limitation is based on an email exchange with an entity close to Zefix and the limitation of the `SOAP-API`, which is known to be monitored. It allows the user 200 requests per 10 minutes without losing the provided access.

[10] https://www.mongodb.com/

[11] `Docker` is an operating-system-level virtualization software which allows the user to run applications in so-called "containers". Creating a virtualization that delivers an environment, which allows for easy deployments on every system that has `Docker` installed. [Mer14] The container just needs to be fetched via script and ran.

[12] `https://www.postgresql.org/`

This part of the crawling ended up using around twelve hours of fetch time. Duplicate IDs were discarded automatically using database constraints.

Getting Data for an ID

After getting the IDs of all active Swiss businesses, the data needed to be enriched with more information about the companies. The way a whole business structure looks is depicted in listing 2.1. Up until this point, only the ID, `ehraid` as it is called by the Zefix API, and some other fields were present in the database. There were two main approaches with which the missing fields were populated:

1. Pulling data from the SHAB
2. Querying every ID on Zefix

SHAB data is published daily and in almost the same format as the Zefix data. This is the reason why staying updated with our built data set should be much easier in comparison to getting the whole data set. Every time a company has to change information in the register of commerce, the SHAB will reflect the changes by publishing the new updated entry in the same structure as seen in listing 2.1 with the previous SHAB publications missing from the `shabPub` field. The amount of data requested was restricted in the sent parameters to 2 days to avoid bigger data loads. The API returned a cursor if there was more data for these two days. The cursor needed to be fetched until it was empty, meaning no more data was found for the given dates. With this knowledge, we reduced the load of businesses that needed to be fetched directly with single queries to about 250,000 entries. Because the API structure was similar to the previously queried endpoints, this enabled us to reuse parts of the same codebase that was written before. However, the consolidation of new data into the existing database proved challenging; an algorithm had to be devised. This algorithm decided which part of the newly fetched data to keep and which one to update, depending on what part of the two business data structures was newer. For every ID in our database, a real company exists or did exist. With the SHAB data, it was possible to eliminate some of the necessary single queries, but there were still around 250,000 requests to be made. These requests would fill up all the missing information for the companies that had only the ID in our database up until that point. Fetching them all from one computer would take a little under nine days of non-stop continuous requesting data from their server to complete. As calculated in equation (2.1).

$$\frac{(\frac{aor}{rlph})}{24} = x \tag{2.1}$$

Where:

aor = amount of requests
$rlph$ = request limit per hour
x = number of days needed to fetch all requests

These numbers were not a good sign for reproducibility in the context of rebuilding this dataset. Because of this problem, a small server-client application was designed to fetch all the missing data. The client would request IDs from the server, the server would return the IDs of businesses with fields that have not been populated. The client could then use those IDs from a different location or IP address to get the corresponding businesses and return them together as soon as all fetches were done. The clients would be given 200 business IDs, as this is also the 10-minute limit. The server would handle the returned values and update the database. It would keep track of which client is fetching which IDs and free them for other clients if the client did not answer within a given time limit.

Due to time restraints, we did not put safeguards in place, which would have verified the data returned by the client. A verification would have been hard to implement because it would be hard to check the client-delivered data without fetching the company from the Zefix API itself. With this solution, the time used to fetch the data could be divided by the number of clients running. In this specific case, three instances of the client were running, which shortened the download duration to approximately three days. During this time, some of the clients received errors from the API and needed to shut down. The server reallocated the IDs later on. A flow chart with the client's actions can be seen in figure 2.2. The figure presents how the client got the IDs, the way of handling them, and shows how the businesses got returned to the server where they got saved.

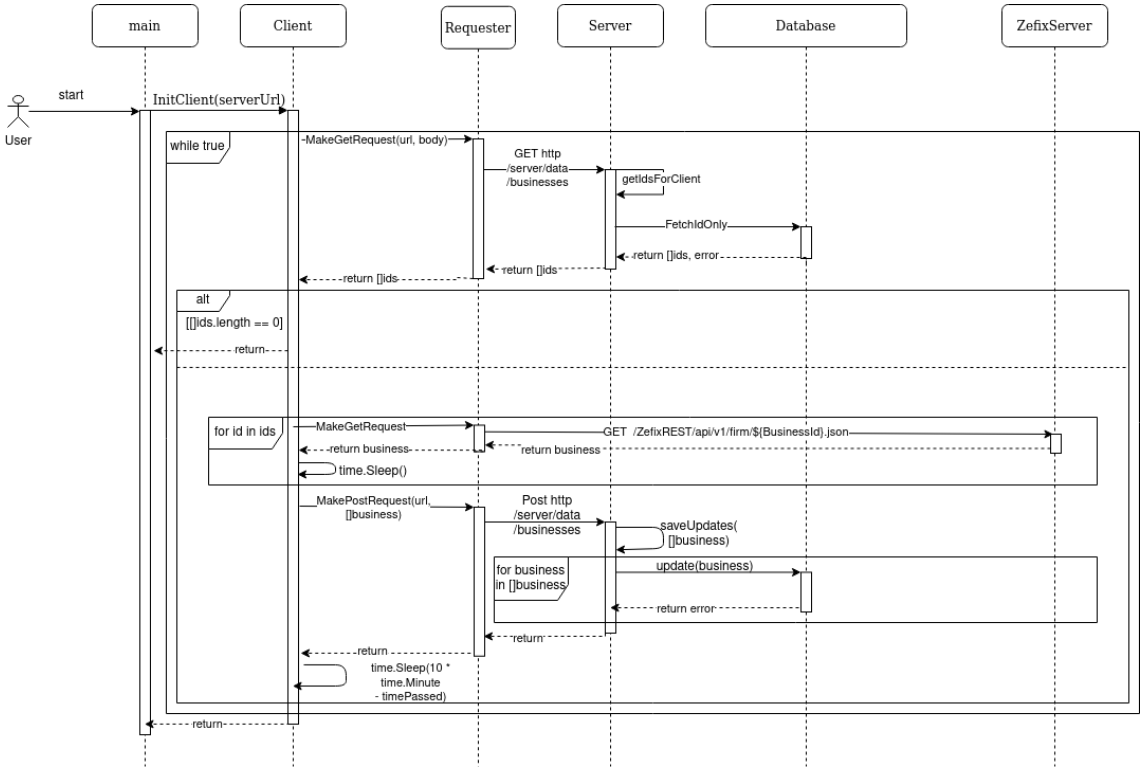

Figure 2.2: Flow chart with the client actions

Make Data Maintainable

The basis for the analysis, clustering, and visualization part of this thesis is the database built by scraping with the previously explained applications. It was mentioned in section 2.2.2, that the official gazette of commerce is closely linked to the Zefix API and delivers changes to the data set on hourly to daily basis. This means that the existing data set is expandable by monitoring the published changes instead of fetching the whole data set again every time the business register should be updated on our database. The application can query the changes from the SHAB-API by specifying the last time the data set was updated and fetching every change since then, consolidating the new data into the previously compiled business database. One of the drawbacks of

this approach is that this data is only queryable for about fours years, as the data returned from the requests seems to suggest. This means that if the local database is older than four years, the application will not be able to get all changes and some modified information or even businesses might be lost. To prevent this situation, it is recommended to keep the database updated automatically.
Additional packages were programmed in the making of the main applications. They were imported by the source code of the scraper and fetcher. These packages include an abstracted database package in `Golang`, which can and should be treated as a layer between the application code and the database. This layer allows the swapping of databases on later revisions of this program and makes it easier to migrate to SQL if the need ever arises. What this exactly means is that no direct database call is made within the main application code. A database package is called, which abstracts the default database API calls. If the data has to be migrated to another database type, the programmer can add a new package that handles the driver, and build the same abstractions that then again can be used in the main application code without changing this code itself. The `Python` codebase has no such abstractions due to the more script-like nature of the code written for the data analysis.

2.3 Data Analysis

The goal of the data analysis is to create a set of data that can be visualized by the client. For this, the structure and informational content of the data needed to be dissected and analyzed properly.

2.3.1 Finding Usable Fields in the Data Set

As shown in listing 2.1, the Zefix business data has a significant number of fields. Many of them were not usable for clustering or building connections between businesses, while others have been used as a basis for further analysis. Fields like `ehraid`, `uid`, `chid`, `rabid` are mainly there to identify the company. As we already disclosed during the data gathering section the `ehraid` was returned for multiple companies during the search on the Zefix register. This ID was used to find more information about a company via shab messages and the direct fetches on the commerce registry API. The `legalSeat`, `legalSeatId`, `address`, and `registerOfficeId` are geolocation indicators. They show where in Switzerland a company is based. `Shabpub` and `shabDate` are both connected to the shab, the official gazette of commerce, showing the date of the last publication for this company as well as holding the messages in an array. These fields were potentially interesting for the development of a company, but the shab data only goes back to 2016. Many companies do not have additional shab information, about 32.99%, to be exact, as can be seen in table 2.1. This table also shows why fields like `hasTakenOver`, `auditFirm`, `branchOffices` and their related fields were of no use for this analysis: none of these fields have more than 2% fill rate, meaning in at least 98% of the cases, these fields are empty. While these 2% could be interesting when looking for outliers, they are not as useful for a general overview of the Swiss business landscape. In comparison, we see that the `purpose` is only missing in 2.01% of all businesses. The `translation` field would have been interesting for sector clustering reasons but is also missing in almost every case. We had to rely on `purpose` to yield results when analyzing it. The field `legalFormId` is given for every company and can be a good basis to group companies. The field `status` is important too to create visualizations based on whether a company exists or was liquidated. At last, the company name has the main task of being the human-readable identifier of a business. Further use of the fields was not foreseeable during the work on this thesis.

field name	shabPub	mainOffices	branchOffices	hasTakenOver
empty fields (percent)	32.99%	99.4%	99.71%	99.11%
field name	auditFirms	purpose	translation	oldNames
empty fields (percent)	98.42%	2.01%	98.67%	95.12%

Table 2.1: Frequency of empty fields in the dataset

2.3.2 Data Preprocessing

There are different ways companies can be clustered to each other. However, before the clustering can be applied, the data needs to be prepared, cleaned, and transformed as necessary. Word clustering works best on texts with multiple words. The field in our business model that came closest to that was `purpose`. Due to the multiple national languages, we needed to know what language we are dealing with. After that, depending on the language, the text needed to be cleaned and prepared differently.

Language Detection

Language detection itself is already a clustering method, and does not require textual preprocessing in this case. The problem of language detection was solved multiple times, as presented in different research papers. Lui and Baldwin compared various libraries against each other as well as their own. [LB14] After some more information gathering, it was decided to use `langdetect`[13], a `Python` port of the `Java` language detection library by Shuyo Nakatani with a self-proclaimed 99% accuracy over 53 languages. [Shu10] The caveat with this library was a bug that does not allow users to specify a subset of languages that should be used for language detection. To avoid this problem, one could edit the code and install the library or edit the library files itself, so the basic language files are just the languages needed. During the development of the application, the bug in the library proved to be problematic and other means of language classification were considered. One that stood out was `Fasttext`. It is a representation learning and text classification library built for customizable use cases. They deliver a pre-built model ready to be downloaded and used with their library, which enables langauge detection; more information about `Fasttext` can be found in their corresponding research papers. [JGBM16] [JGB+16]. To not just overwrite the existing language classification, a second language field was introduced into a `langInfo` struct containing both language categorizations. From the data scraping, it was already known that in most cases, Italian, French, or German were used. Therefore, those were the languages that have been detected. Going forward, the categorization of the `purpose` field by language was important, as not knowing the language would have resulted in clusters of languages, and a proper cleaning would not have been possible.

Cleaning

In modern natural language processing, it is crucial to have a good basis. This includes text preparation. [Bro19] The text to work on later was tokenized, i.e. tokens were created from words, punctuation was stripped from those tokens, and all non-letters were removed. Stop words needed to be removed as well, for they would have created an imbalance since those words show up in most business purposes. The last step was to create the stem and the lemmata of the tokens. This

[13] `https://github.com/Mimino666/langdetect`

was crucial to roughly group similar words together. For the stemmer, it was important to know the language of the text because it needed to remove the pre- and suffixes of words. These are particularly language-specific. For this specific use case, the code contains two extra steps: the removal of stems which should not be used later on, as seen in algorithm 1, and the removal of sentences that are repeated in a big part of the business purposes. The former happens on stem basis, the latter is done on the whole sentence, where the script primitively compares strings of unwanted sentences with each sentence of the text.

Algorithm 1 Removal of unwanted words from Python dictionary

$i \leftarrow 0$
$stems \leftarrow [...stemmedWords]$
$words \leftarrow [...completeWords]$
$mostUsed \leftarrow [...mostUsedWords]$
while $i < length(mostUsed)$ **do**
  if $stems[i]$ **in** $mostUsed$ **then**
    $delete\ words[i]$
    $delete\ stems[i]$
    continue
  end if
end while

In the end, the pipeline to clean the data had a clear order, visualized in figure 2.3, and needed to be finished before the use of clustering algorithms on the text basis. [14]

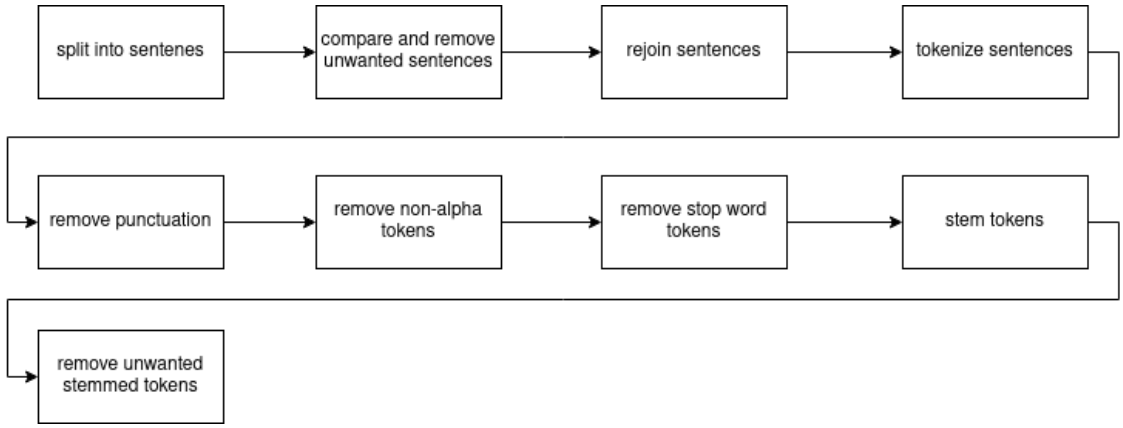

Figure 2.3: Representation of code cleaning pipeline

2.3.3 Clustering and Categorization

As clustering is explained, it is necessary to add that language detection is already a form of categorization needed to continue into this section. This thesis will not go into more detail about language detection, but will focus on the general clustering.

[14] This excludes the language detection

```
A Landwirtschaft, Forstwirtschaft und Fischerei
B Bergbau und Gewinnung von Steinen und Erden
C Verarbeitendes Gewerbe/Herstellung von Waren
D Energieversorgung
E Wasserversorgung, Beseitigung von Umweltverschmutzungen
F Baugewerbe/Bau
G Handel; Instandhaltung und Reparatur von Kraftfahrzeugen
H Verkehr und Lagerei
I Gastgewerbe und Beherbergung
J Information und Kommunikation
K Finanz- und Versicherungsdienstleistungen
L Grundstuecks- und Wohnungswesen
M Erbringung von freiberuflichen, wissenschaftlichen und technischen
    Dienstleistungen
N Erbringung von sonstigen wirtschaftlichen Dienstleistungen
P Erziehung und Unterricht
Q Gesundheits- und Sozialwesen
R Kunst, Unterhaltung und Erholung
S Sonstige Dienstleistungen
```

Listing 2.2: Company business sector taxonomy, source: Federal Statistics Office [str19]

The goal is to be able to cluster the fetched companies in different ways, depending on the data obtained about them. One of the possible options the thesis should be oriented towards was the taxonomy used by "Struktur der Schweizer KMU 2017". [str19] This system is based on the economic sectors in Switzerland with respect to their size in the context of the distribution of small and medium-sized businesses due to the fact that they make up most of the Swiss business landscape, as already mentioned in chapter 1, the introduction of this thesis. [str19] Due to the overlapping nature of these business sections, we were not be able to get a clear distinction. We wanted to create a categorization that we would able to used in different visualizations later in the thesis.
An additional, different angle to work with was the location where a company is based. Based on the address, we found the corresponding geospatial data and useed it for clustering or categorization. For further visualizations, this could prove to be a basis for presentation on which another clustering could be mounted.

Mapping Addresses to Locations

To get an overview of the towns in which registered businesses are residing, it was decided to map the postal code of the company address to its appropriate geographical location. To be able to do this locally, without the need of an active internet connection, a file from rueegger.me was sourced. [Rü15] The file has 5355 postal code entries of Switzerland and seemed more complete than the official search register of the Swiss Post.[15] The data was read as CSV, and used in a map. To be able to do this more quickly, a `geo`-field was added to the database business struct, which would hold the longitude and latitude of the towns mentioned in the company address. Companies with missing addresses were ignored.
This procedure revealed a bigger problem: the Swiss postal code system has a lot of inconsistency.

[15] `https://www.post.ch/de/kundencenter/onlinedienste/plz-suche/info`

During the data analysis, a list of postal codes that were missing in the database were discovered. Upon further searches, it was noticed that these codes did not even exist on the official Swiss Post website. Multiple reasons were found for this strange behavior:

1. Changing the postal code of cities that were not updated in the Zefix database
2. Typing errors in the postal codes which turned out to be invalid
3. Incomplete postal code lookup from the rueegger.me file

Out of these three cases, the first was most common. Apparently, the merging of different towns and the resulting change in codes are not updated. After updating the database file, there were an additional 200 companies of which the location could not be identified and therefore were ignored for the geographical visualization.

Clustering with TFID, Kmeans and guided LDA

In the following paragraphs, different approaches for clustering by business purpose are listed. All of them are based on machine learning algorithms and they all have in common that they did not yield results good enough to be used in the further course of this thesis. The approaches and their shortcomings are shortly described in the following sections.

TFIDF with K-Means `TFIDF` is used to create a corpus of words, similar to how a bag of words works but with more focus on the weighing of a word. It takes into account how many times a term occurs in documents. [WLWK08] `K-Means` is the clustering algorithm using the `TFIDF` as basic corpus for the grouping; the number of clusters is predefined before running the algorithm. [SB07]
The first try was based on the explanation of Brandon Rose.[16] The idea was to bring the text data into vector space by applying `TFIDF` to it and then running a `K-Means` algorithm on them to create the clusters. [Ros14] For both steps, the `Sklearn` implementation was used due to its popularity in the community. The library itself also includes different approaches to machine learning, but for this thesis, the `TFIDF` vectorization and the `K-Means` algorithm were to be focused on. [BLB+13] `TFIDF` allows the weighing of words in the context of the frequency that they appear in. With weights that were created before, different algorithms, like `K-Means`, can find connections between the terms and create clusters of documents containing them. The vocabulary was built upon the purposes of our data set. Using this approach, no usable data was generated. The noise in the data set was too much for the K-Means algorithm to build viable clusters. There are a lot of companies with one-word descriptions and lots of boilerplate text. Even though it was removed, the different way companies describe their purpose makes it difficult to find a clear pattern for the algorithm. This problem persisted even after having iterated over the text cleaning function multiple times. As for the coming approaches, different numbers of clusters ranging from very few to 100 were tried. In all attempts, no valid cluster could be identified. Another problem with the library manifested itself when trying to calculate the cosine similarity: the application quickly claimed more than 32 gigabytes of RAM, which crashed the application on the available computer. After removing this limit, the application still returned problematic results and this approach was dropped.

LDA Secondly, Latent Dirichlet Allocation, short `LDA`, was considered as an option. `LDA` is a probabilistic model for data collection, as described by Blei et al. [BNJ03]. It is often used in combination with text data to create clusters and seemed like a good option for this thesis. `Gensim`

[16] `http://brandonrose.org/clustering`

is a library developed for topic modeling with big data sets, which also has an implementation of LDA. This is exactly what we needed. [RS10] Cluster numbers were variable, as was the n of the n-gram. This gave room for some experiments in trying to find clusters that could belong to each other. Numbers of clusters between three and fifty were tried, as well as n-grams for the size up to 5. The basis of the vocabulary and its weights were also vectorized by the `TFIDF` vectorization method like in the last approach. The problem also started with the noise in the data set. Different topics got mixed up and put into unrelated groups even with high cluster numbers; there was no way of seeing clear topics that could be combined. Due to these results, we did not pursue `LDA` any further and tried a different method that used `LDA` as basis.

Guided LDA The Federal Statistical Office published in "Struktur der Schweizer KMU 2017" a taxonomy of companies based on the economic sectors. This taxonomy can be found in listing 2.2. [str19] The idea was to create a guided `LDA`, a semi-supervised way to cluster the company data. `GuidedLDA` is built upon this approach and wants to pre-seed the topics which can be generated from the text. [Sin17] With this in mind, we used the mentioned taxonomy and pre-seeded the machine learning algorithm. There are not many libraries that build upon this approach, so it was decided to use one called `GuidedLDA`[17]. Here the numbers of topic clusters were fixed by the numbers of pre-seeded topics. The n-gram was variable and different combinations have been tried, together with the pre-seeded topic number. This caused a set of problems that ultimately lead to the dropping of this approach. Among the most pressing reasons was the fact that the library could not handle the data set too well performance-wise, as well as the quality of the data when smaller slices of the set were given to the library.

Word Embedding In the last try used with machine learning, `word embedding` seemed to promise a solution to the existing noise problem in the data set. `Word2Vec` is a newer `word embedding` concept, which relies on neural networks. Its main usage is to find similarities between words in the context of how they appear. [MCCD13] This meant that we could at least have created a taxonomy of certain words in a document and would have been able to group found words into different sectors. If this had been done, the idea would have been to let new businesses be classified by the distance of the words in the purpose to the words in the dictionary built. At first, this attempt seemed fruitful, only to then be voided by a noisy data set, the same problem as before. Many companies do not have their real purpose focused in their registry entry. Found clusters were also not usable for our purposes for reasons like the clustering of city names due to the sentence structure.

Keywords Mapping

The most straightforward approach that was taken is a simple keyword mapping. This means that keywords for each category were defined by hand. The business purposes were then compared to those defined keywords and a ranking was created. This presents an effective approach to clustering but cannot handle sentences with words that are not in the keyword dictionary, meaning that syntactical patterns are ignored, and non-obvious connections are not represented. This also means that no real connection between words can be made because there is no weight to a word like in the previous approaches. Keywords also had to be defined and modified for every language. Because of the different language structures, this imposed new challenges to the algorithm, such as German composite words. In the first run, the algorithm took the word stems as a whole. If no keyword has been detected in this set of words, the text was cleaned again, but this time we applied a compound word splitter, which tried to separate different words into

[17] `https://github.com/vi3k6i5/GuidedLDA`

sub-words. After this additional cleaning step, the stems were again matched with the keywords. Businesses that could not be classified after the second text cleaning were categorized as -1, meaning that they had no category applied and were filtered out in the next steps if needed.

The basis for each topic again relied on the taxonomy presented by the Federal Statistics Office found in listing 2.2. This time around, the categories were modified to create logical fitting fields. D and E had been combined into a single category. M and N had been left out and a new field was created, which represents companies that are doing administrative and consulting work. Another category was appended, which holds businesses such as locksmiths, funeral homes, and other companies which could not be categorized into one of the previous sectors but were still important enough to keep track of. This category is called "Others".

With the ground work done, the algorithm was implemented in a way that led to a categorization where a business can have multiple categories but still maintain an accuracy which proves to return useful information. In the end, an algorithm similar to the concept of the well known "bag of words" was used. Every word stem from the previously cleaned text was checked for its equal in the existing list. If the word existed, it would award this category a point for this business. After finishing the stem list, the scores for each category would be normalized by dividing these scores by the number of all points that were given for this companies purpose. The scores would then sorted and analyzed: the highest score would be checked if it is over 0.5, the belonging classification would be accepted as the sole category for this business. If the score was lower, the next item in the list would go through the same process. But every time the test was repeated, we would half the number score required to stop adding categories. The resulting logic is represented as pseudocode in algorithm 2.

The categories were then saved as an array to a field of the businesses in the order of the scoring system. This field is called `businessType`. The score for each classification is omitted, but due to the deterministic nature of the algorithm, the results are repeatable if needed and the numbers recoverable.

Building the keyword list As previously mentioned, the keyword list was built by going through company data and picking reappearing words that seemed clearly unambiguous for a sector by hand. Additionally, a list of the most used words had been generated by a self-implemented script. These words were also added to the list if they were specific enough for a category. Words which had already been removed from the most used word list in the analysis step were not taken into consideration due to the noise that they would introduce. The classification algorithm was then applied for each language separately; the results were checked for businesses, which the algorithm was not able to classify. For these companies, a new list with the most used words of their purpose was generated and again checked for eligible candidates for the word list. This approach took multiple iterations to get a good balance between the words and their category. Removing words was also sometimes necessary since too many companies got classified with the wrong sector. The last generated list shows the state of the vocabulary in German in table 2.2. Each word in this list is unspecific and cannot be tied to a category. Words that could be clearly categorized and are not already in the keyword list see less usage in the purpose and are not fit to be used as keywords. This led to the now existing keyword list for German and French. The French word list was derived from the German list and enhanced with words directly from the purpose vocabulary. A direct translation would have proven problematic due to different language structures as for example composite words and the fact that words can be translated differently.

Additional information In the section about keyword mapping, no mention was made about the Italian language. The reason for this is that there is no keyword matching done in Italian,

Algorithm 2 Business data classification algorithm, simplified

```
for all businesses in businesses do
    businessType ← []
    sectorList ← []
    stems ← business.wordStems
    matchedWordCounter, matchedList ← findCategory(stems)
    if matchedWordCounter is 0 then
        newStems ← reStemText(business.purpose)
        if newStems is not false then
            matchedWordCounter, matchedList ← findCategory(stems)
            if matchedWordCounter is 0 then
                businessType ← [−1]
            end if
        else
            businessType ← [−1]
        end if
    end if
    if matchedWordCounter > 0 then
        countValue ← 0
        sortedDict ← sortedByValue(matchedList)
        for all key in sortedDict do
            businessType ← key into businessType
            if sortedDict[key] ≤ (countValue ÷ 2) and countValue ≠ 0 then
                countValue ← countValue ÷ 2
            else
                break
            end if
        end for
    end if
end for
```

meaning that Italian business purposes could not be classified. The main cause for this is the small subset of purposes, which were categorized as Italian, as only around 7% of all businesses were classified with this language. Additionally, no suitable translator was found that would have been able to transform the data set into Italian in the given time window. A simple word-by-word translation from German or French to Italian would not have worked due to different language structures, as already explained in the last paragraph.

2.3.4 Additional Processing

Text indexing To enhance the acquired data and make it queryable more easily, we decided to index all the available companies by their text data. This allowed us to perform full-textual search. Again multiple options were present, for example `Bleve`[18], however it was decided to use `MongoDB`'s default indexing capabilities, one of the reasons why `MongoDB` was chosen as database. It is convinient due to the fact that we already use it and would not need to migrate the query language and data to another system. Due to the idea that the search should mainly focus on the text, the name and the purpose of the businesses were indexed via the built-in `MongoDB`

[18]Bleve is a full-text search and indexing engine written in `Golang`: `https://blevesearch.com/`

Word stems	nr. of occurences	Word stems	nr. of occurences
vollstand	3670	gesellschaftszweck	995
durchfuhr	2102	kommerziell	920
fuhrung	1878	fern	874
kauf	1861	ubernehm	827
fuhr	1658	gebiet	811
zusammenschliess	1620	planung	778
forder	1542	weit	755
vermittl	1448	ubernahm	706
finanziell	1224	gewahr	680
ahnlich	1035	gleich	651
ausub	1028		

Table 2.2: Most used German word stems of companies which could not be categorized, words are shown stemmed

API. The main drawback of this approach was that `MongoDB` only allows full-text search and has no concept of fuzzy search, although it allows us the query of stemmed words. [incndb]

Leveraging liquidated companies Another angle that presented itself during the analysis of the data was the `status`-field of a company. This field holds information on where a business is standing in its lifecycle, meaning if a company is existing, in liquidation or deleted. With this information and the additional knowledge of when the last SHAB message was generated, we can tell when a company was dissolved.[19] This information is easy enough to obtain so we do not need to generate a field in the database because we can query this with little to no effort.
With this, the last field we needed to work with was analyzed and the fetched business structure depleted as far as possible.

2.4 Visualization

In this section, we explore the details of the visualization implementation. The goal is to create a way the user can understand the data and get new insights into the business landscape of Switzerland by browsing through the graphs and maps. It should also be possible to search for similar companies. The knowledge won can be purely anecdotal or used to find patterns that can help with initial business ideas or decisions.

2.4.1 Visualization Details

The visualization part of this application is written in `JavaScript` with the help of different frameworks like `React` and `D3` for the visualization part and `Redux` on the data handling side. Together with the newly created state reducers and an applied middleware, the application can handle side effects, like the fetching of new data from the server, with ease while abstracting remote calls from the user interface. The web app is bundled via `Webpack`[20] and can be served by

[19] This will only work for the last three years, since the SHAB does not allow to get older SHAB messages.
[20] https://webpack.js.org/

any web server with the capability to serve HTML and JavaScript files. This stack of application was used because of good synergies of libraries connecting them and previous experience with these technologies. A prime example would be the handling of JSON data in JavaScript and Golang. Both programming languages support this data format with their standard library.
From here on, we discuss the different parts of the web app. There are three subcategories which should visualize the same or similar data in a different way to get a new perspective on the Swiss business landscape while delivering the functionality of searching through the data via connections between the companies. We use the connections that were found while analyzing the business in the last section. Every visualization category is built on the same principle: a control bar that controls the flow of the data shown and a visualization part that actually represents the value it gets.

Map The map of Switzerland should give a feeling of how Swiss businesses are distributed geographically. Due to limitations of the data rendering, it was decided to group the companies by their postal code on the map. This led to better performance for the client and interesting groupings. To implement the map, a geographical data source was needed. Multiple sources were present to pick from. The ones used in the process of writing the programming part of this thesis were SwissTopo and EuroGeographics. SwissTopo delivers data over their API; this was then transformed into topoJSON data and broken down to geoJSON, both are extensions to normal JSON with the help of a library[21] and a conversion web app[22]. As previously mentioned, the company data was grouped by the postal code corresponding to longitude and latitude. This opened up another possibility: the implementation of canton groupings. If query values change, the server gets a request with the new parameters and returns the cantons or cities for these values. Internally these different views share the same D3 code for drawing the map and only differ in the points of drawing the color of the cantons or the points which represent the cities on the map. Adjustments to the query parameters in the overview component will lead to automatic fetches of the data which represent these changes; this happens using React's useEffect hook. With this approach, the application decides when it is necessary to get the data, without any additional user inputs, like pressing a search button.
Calculating the colors of the city circles is done via a short calculation. When no language is selected from the dropdown menu, the color of the dot will be determined based on the language distribution in this area. Blue is German, red is French, and green represents Italian. The color of the dot shown on the map is defined by the languages of a city and can be mixed. A place that has the same number of companies with French and German purposes will have a purple colored dot. If a language selection was made, all places with no businesses corresponding to the color would be missing from the map; all others will have an alpha value (transparency) that relates to the percentual representation of the language in this city. It is similar for the cantonal map. There the alpha value of the blue color changes based on the representational strength of what the user searched for on a national level. All available businesses for the current query are summed up, and cantons with the highest percentual share will have the lowest alpha value. In contrast, cantons with smaller percentages will have higher alpha values.
To navigate the map, zooming and panning elements were included on the left side of the web page. This feature was removed again in favor of the integrated zoom/pan functionality of D3. Changing to this approach increased the performance of the map manipulation by avoiding unnecessary re-renders caused by React.

Search The search view is another part of the application. The main part of search does not need any graphical libraries due to the fact that the idea of this view is to allow users the search

[21] https://github.com/interactivethings/swiss-maps
[22] https://mygeodata.cloud/converter/topojson-to-geojson

of similar companies without needing to rely on map data. The data layer exposes the businesses that were searched for via the parameters, and React renders the available businesses in a grid view to HTML. Found companies can also be displayed in a map if this is wished for. A caveat of the search view is that only a limited number of businesses will be returned when queried. This is to avoid the web app failing to handle this rather large amount of data. Loading too many businesses at the same time can crash the browser or take too long to render. In both cases, the user experience is ruined.

Statistics Statistics was also built upon the basis of the D3 library. Bar charts are a more traditional way of displaying information and were implemented to give another insight into the data set. The components are arranged in the same way as in the general map. Leaving the left side of the screen to the graphical representation of the data while keeping the rightmost part to the possible filters to set. In this case, the canton data is fetched from the server and transformed into the nested arrays necessary for D3 bar charts. As with the map, controlling elements were built into the side of the window and were connected to the chart via the React view. This allows a swift manipulation of the represented data as well as giving an overview of what is displayed in the main area of the website. The bar charts only re-render when parameters of the control elements are adjusted or when the data from the server has changed values.

2.4.2 Data Delivery

The data needed for the visualization is delivered by a queryable application written in Golang, which is used as our REST API endpoint. This is another application than the one used to fetch our data but has the same dependencies to the self-implemented universal packages such as models and the database abstraction, as the previous tools written. The server knows three relevant endpoints: businesses, cities, cantons. Each one is corresponding to a resource saved in the Redux store of the client. Depending on the query string of the request, different filters are gathered and applied while querying the database for the requested results. As mentioned before, cantons and cities are already saved on the database but are dependent on the data of the business collection. But the canton and city endpoints can deliver additional business numbers more dynamically. Due to aggregation functions of the MongoDB, and strong filtering, business values for the cantons can be shown more accurately and will return the actual numbers of businesses found, unlike the cities endpoint, which will just filter cities out that don't have the right values. The business endpoint can, additional to the normal queries, return limited answers. This leads to a slimmed-down data response and can avoid crashing the API client, due to a huge amount of businesses returned. All three endpoints return the necessary documents as JSON objects. The Golang part can be compiled into binaries for the most common operating system. Data is, as mentioned in the data gathering section, stored and delivered by MongoDB run in an instance of Docker for easier deployment.

2.5 User Data Collection

In this section, we will give a short overview of the user data collection on the web application and explain why this method was chosen.

2.5.1 Collection Implementation

One of the goals of this thesis was to be able to see what a user does on the visualization. The primary information, in this case, is what exactly the user is looking for. An ID is created to identify the user during his time on the webpage and saved into a cookie. This cookie is then sent to the client with every API query by allowing credentials in the fetch request. The `Golang` server has a custom middleware which checks every request for cookies. If one is found the `UUID` (Universally Unique Identifier) of the cookie is extracted and saved to the database together with the path of the request, the actual request queries, and the time when the request is saved. The request information can then be queried on a separate database collection, meaning timelines can be established while looking at the data.
Another viable option would have been to use Google Analytics[23] with events triggered by user actions on the page. However, due to testing and privacy reasons, the events were tracked over the request to the server itself. It also integrated nicely with the existing preexisting server architecture and allowes for expansions or changed behavior in the future quiet easily. Google Analytics was considered, but ultimately, it was decided to keep the solution in the stack written for this thesis.

2.5.2 User Anonymity

User data should not be directly traceable to the user. This could otherwise lead to various problems with data security. To solve this problem, we first need a way to keep a user anonymously identified via an ID that gives no information about the user. This ID is in the format of a `UUID` and stored in a cookie for the web app. The only thing being able to identify a user is the access time, which is very ambiguous and inaccurate. This should avoid most problems with the user identity.

2.6 Evaluation

After the creation of the user data collection feature, we needed to evaluate our approach. A good way of doing so was by performing a real-world study with the company Skippr. With this, testing the capabilities of the tracking system was possible, and it would allow for some direct and indirect user feedback when watching the user behavior on the web app.

2.6.1 User Study - Skippr ltd.

Skippr ltd. is looking for its competition on the web app. The head of Skippr ltd, a company in German-speaking Fribourg specialized in online presence, social media, and visual design, wanted to know which other companies work in the same field as them. He was given access to the web app under supervision. His goal was to find out about the structure of the canton of Fribourg in the context of similar companies to his own and also find businesses within their field of work, which could be classified as directly competing.
We purposefully choose only one user to observe due to time constraints. But on the other hand, this allowed us to focus on a qualitative observation.

[23] `https://analytics.google.com/`

2.6.2 Procedure

At first, the user was without instructions on how to use the page. We gradually increased the instruction flow whenever necessary until the user was able to complete the task. Given hints were noted down for the analysis of the behavior and helped to update certain aspects of the application. This particular user has no background in computer science and is not familiar with the programming concepts used. The only information he had beforehand was that he should be able to find similar companies to his own. At the beginning of the test, the user was guided onto the "Home" section of the web app. The user then took multiple steps to reach his goal, being interrupted only once to give him a hint on where to find the businesses. The mentioned steps can be found after the main matter of this thesis in appendix C.1. With his steps, the user created a trail of data in our database, which can then be used to compare the hand-noted steps with our implemented user tracking system. The user tracking data was queried from the database by timestamp and user ID. The resulting saved requests can be seen in appendix C.2. As a last step, the user was asked to give feedback about the usage of the application, the answers are included in the appendix C.3 of this thesis and were used to improve the web app.
The results of the user study will be presented in the next chapter under section 3.3.4.

Chapter 3

Results

This chapter gives an overview of the results found during this thesis before coming to a conclusion in the following chapter. We will revisit the different steps taken in this thesis' approach and how they affected the results.

3.1 Data Acquisition and Maintenance

The data from the Zefix register was saved into a `MongoDB` instance. In the end, 774,129 companies were found, around 146,965 of them already dissolved or in liquidation. The data set is easily expandable by the provided scripts. As previously mentioned, only the changes of the Swiss business registry have to be fetched with the Zefix API. Every step that needs to update the data is represented as a script, which is part of the results of this thesis. The concept to keep the dataset updated includes:

1. Downloading new/changed company data
2. Applying transformations to the data set such as language detection, stemming and categorization
3. Rebuilding cantonal and city data, which depends on the business data

With the provided scripts and these predefined steps, a trained user can update the data sets regularly if needed. The instructions are found in appendix A.

3.2 Company Data Analysis

During the analysis, it was discovered that we could categorize about 90% of the data for purposes which were identified as German and approximately 80% for purposes identified as French. For German, 1.8% are identified as part of the primary sector, 22% as part of the secondary sector, and 66% as part of the tertiary sector. For French, we have a distribution of 1.2%, 17%, and 62% respectively. These numbers are represented figure 3.1.[1] They are not too far from the official SME business distribution over the sectors (9%, 15% and 76%), as shown in "Struktur der Schweizer KMU 2017". [str19] The differences can be explained by the fact that not all businesses in Switzerland need to be registered in the registry of commerce. There is also a large portion of businesses which were not categorized due to missing purposes, or purposes that did not contain a keyword.

[1] Number rounded to whole percentages except for the primary sector, where numbers are rounded to a tenth of a percent

It also needs to be mentioned that this statistic includes data from businesses over a multitude of years as well as liquidated businesses, while the numbers referenced from the source are for the year 2017.
It seems fitting to note that the data set used seemed problematic due to its noisy nature. Many

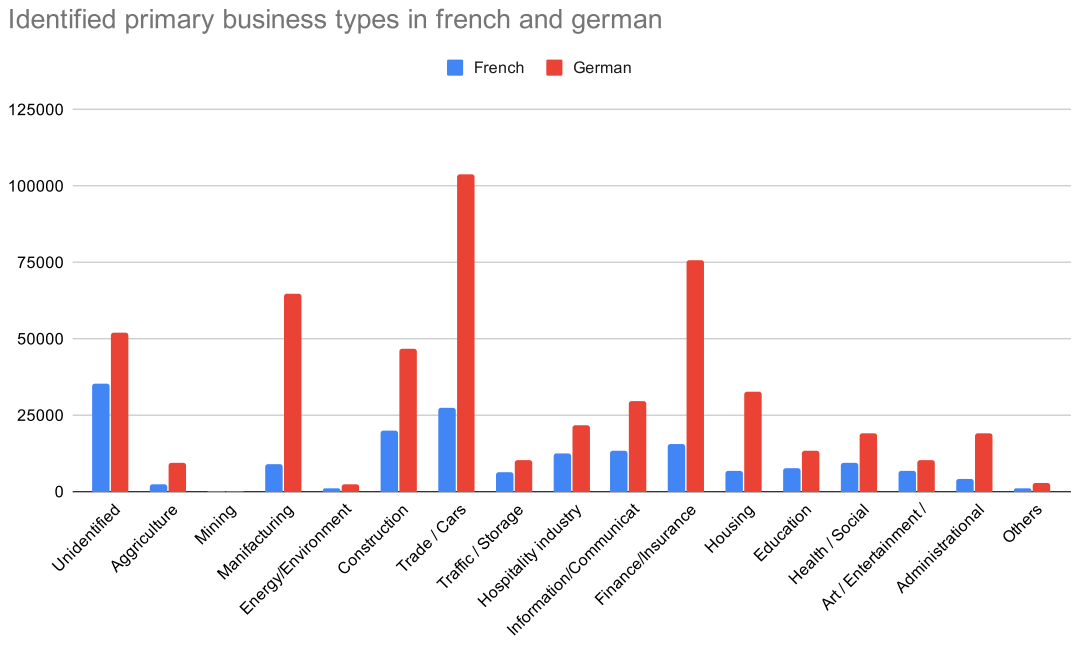

Figure 3.1: Business distribution on business sectors for French and German purposes

company purposes in German, for example, contain just a short description of what they do. The majority of the descriptions contain templated strings to avoid legal problems. It is not helpful for our cause that many companies have missing purposes or just template sentences.

3.3 Visualization

Just as in the previous chapter, it seemed fitting to split the achievements of this thesis in multiple subcategories. In each subcategory, we give an overview of what was accomplished while visualizing the data transformed by us. All three views work with similar information but each of them highlights a different viewpoint on the data set with a juxtaposed focus.

3.3.1 Map

This view contains two sub-views: a city view and a cantonal view. The city map shows all cities which meet the criteria defined in the control panel of the map view. Changing one of the input values will result in a change of the display, meaning cities might vanish or popup depending on the queried values. It also enables the user to show the language distribution of the company purposes across Switzerland. A checkbox at the bottom of the input overview enables the user

to visualize the number of companies a city has in comparison to other cities by enlarging city circles accordingly. Clicking on a place will display all companies that meet the criteria defined and opens up an information box. From there, it is possible to switch to the search view and continue to examine the shown companies with other tools. Additional information for a single company can be viewed when clicking on the corresponding button in the list. The resulting overlay presents more detailed information from the analysis and previously existing data.
When a user clicks on the "cantons" tab, a geographical parted cantonal view is presented with the cantons colored in different shades of blue. The default query result for the cantons can be seen in figure 3.2. The lower the alpha value of the blue, the more businesses adhere to the search query in this canton. As previously mentioned, the cantonal data can be queried using the selectors in the overview. Here, the full-text search of businesses is enabled, so businesses matching the search query are represented in the percentual values. Additionally, the values cannot just be compared nationally but also locally in the canton. By checking the corresponding checkbox, the user will see what percentage of all businesses in this canton fall under the searched values.

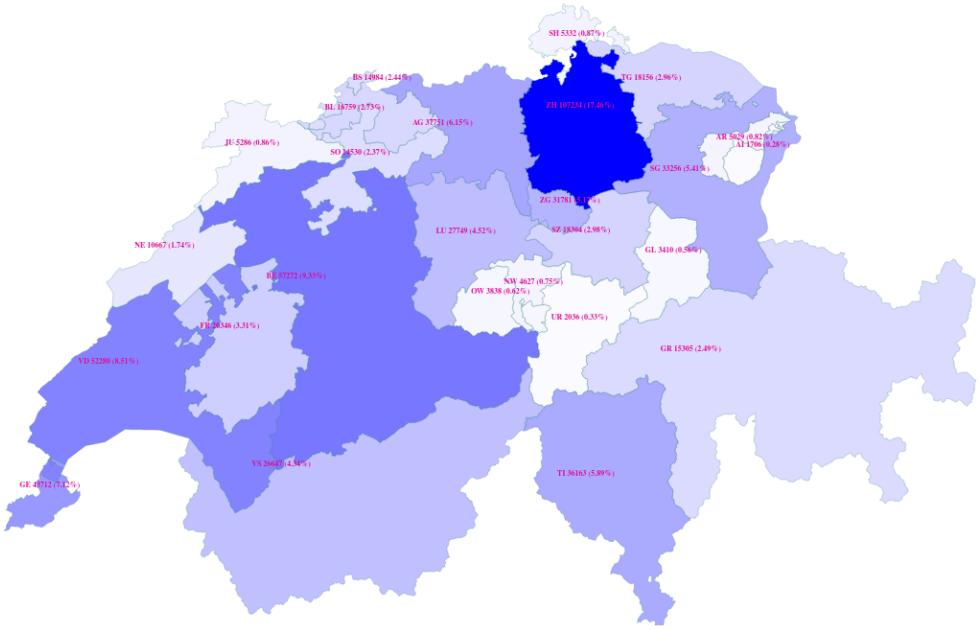

Figure 3.2: Cantonal map with default search values, representing the distribution of all companies over the cantons of Switzerland. Source of geographical data: Federal Office of Topography swisstopo [fLs20]

3.3.2 Search

The main functionality of the search view is less graphical and more based on finding similarities between companies, but the queried results can still be visualized. When first visiting the site, legal, sectors, cantonal, and textual limitations can be applied to the company query. By clicking a labeled checkbox, the user can choose between searching existing and liquidated companies. The number of returned companies is limited to 500 at a time. The reason for this is to avoid handling too much data at once on the client-side. When a company is found that seems interesting, it can

be selected and the filters are changed to match its properties accordingly. From here on, the user can adjust the settings so similar companies can be found. With this feature, it is possible to identify competition or market niches locally. When a selection is made, an additional geographic and heatmap can be displayed. The geographical map visualizes the cities of the shown companies similar to the display in the map view. The heatmap juxtaposes the legal forms and the sectors of the queried data in opposition. Correlations between these two fields can be seen with the help of this map. An example search can be found in figure 3.3. Clicking on a field of the heatmap will close the modal in which the data is displayed and fill out the query parameters of the search with the sector and legal form corresponding to the pressed field.

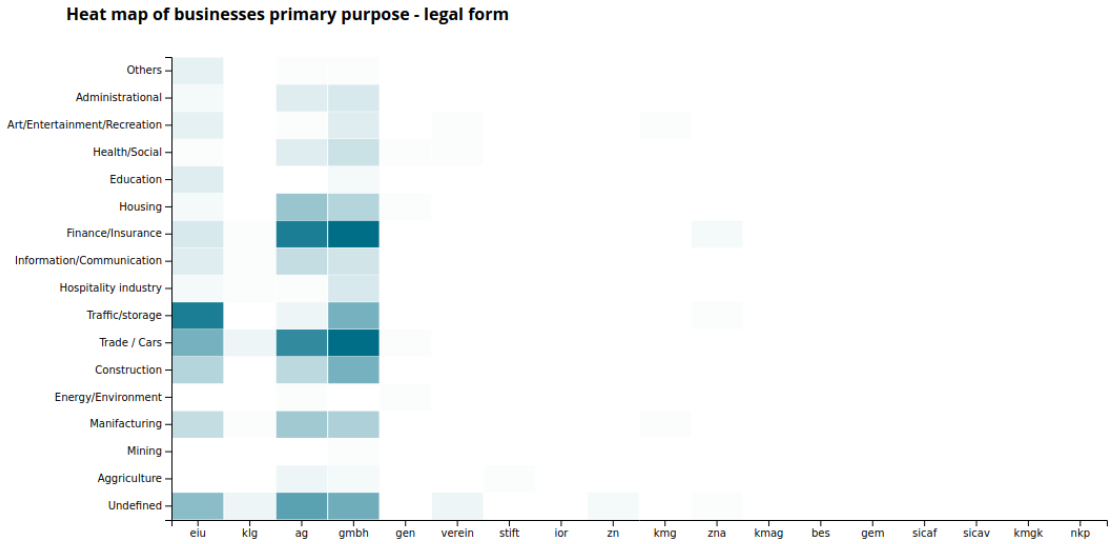

Figure 3.3: Heatmap showing the legal form and business sector combination frequency

3.3.3 Statistics

In order to present a more traditional look on the data set, another view was added which represents specific company data with bar charts. By selecting the right menu option, legal form- and branch-specific distributions are displayed, and each subcategory of the two main categories can be selected. The resulting bar charts represent the distribution of companies with the selected subcategory over the cantons in a more detailed perspective.
Additionally, clicking a checkbox allows the user to see how many companies were liquidated per canton, legal form, and sector. The numbers are split into the years 2016-2020, while 2016 and 2020 are not complete due to data not being available for 2016 and 2020 being the current year. A selection element on the page can change which year is displayed.

3.3.4 User Study and Data Tracking

Data gained in the study shows that the application can deliver information to find similar businesses. Furthermore, the user gave some valid feedback, which led to updates to the user interface, such as additional styling and the implementation of cantons as a search variable in the

"Search" view. The user feedback can be found in appendix C.3. The user's behavior also indicated some weaknesses in the way the web app is presented. Features were not used and the study participant took a different approach than expected. The user seemed to have the biggest issue with understanding the range of possibilities the web app delivers. An overflow of UI elements with no corresponding explanation led to the user trying to find his information on the wrong view in the web app. This insight calls for better labeling of the features and some instructions on the page itself.
The user study has proven itself as a useful tool to show how much information the user tracking delivers. Every search query can be connected to the actions of the user in our study. With a more detailed script, all kinds of usage information could be collected, and the data be brought into a more specific form. In the action transcript, found in appendix C.1, steps 1, 7, 8, 9, 10, 11, 15, can be matched directly to the saved requests. Step 14 is not entirely unambiguous, but in the context of previous requests, it is evident that a company must have been selected. Even though name of the selected company is not in the data, the study shows that this approach is effective to track the user behavior and highlights what an anonymized user is interested in business branch-wise.

Chapter 4

Discussion

4.1 Summary

During the work on this thesis, a method to download and maintain a dataset consisting of Zefix business entries had to be created. Other data sources were also considered but had to be left out due to monetary constraints. We concluded that it is possible to create a dataset of Swiss business information, as seen in listing 2.1, that can be updated with newly generated values by the for this thesis written data analysis scripts. This data was then analyzed and transformed to gain a deeper knowledge of the sampled structures, and to craft new insights for visualizations further down the line. During the analysis, different approaches to clustering businesses were taken. However, most of them were fruitless due to an extremely noisy dataset, and the unstructured approach companies have when describing their company for the local registry. In the end a simple self-designed keyword mapping algorithm was used to assign companies to different business sectors. These sectors were derived from an official publication of the government and modified for our purposes. The data and its new additional information were then visualized in three different approaches: geographical map, business search, and statistics. All three views use the same data but were developed for different purposes. The frontend `JavaScript` can fetch the necessary data via a `REST` endpoint provided by a `Golang` application connected to the database. This backend provides routes for every resource needed by the client. User tracking was implemented by storing a cookie with a userId and sending it to the `Golang` server where search requests were then saved to the database. In the end, a user study was done to see how users were able to handle the application. Smaller changes were made based on this user study.

4.2 State of Open Data in Switzerland

In Switzerland, companies have to submit a record of their business information to the official registry of commerce due to Art. 934 Abs. 1 OR. Generally speaking, not all companies or single-person businesses are obliged to do that, as long as their gross profit lies under 100,000 CHF. The registry data is openly accessible to the public, and all cantonal registries notify the central register of their changes, so the central database can be updated. In some cases found in the data set, the data was corrupt, and errors occurred, which led to wrong information being stored by the central database. For example, certain companies lost their business purposes during the transfer, and others had transposed digits in their postal code. Another problem arises when a company is based in a city that merged with another city. The change in postal code is not automatically reflected in the national registry. It is small problems like these which make it harder to create a clean data set for the usage in computer science applications.

Another obstacle is the accessibility of the data sets, originating from the commerce registry. Even though the registry is accessible to the public, Zefix did not want to hand out the complete data set as a whole and highly limited their official API. It does not seem to be in the interest of a publicly available registry to limit the access that strongly.
The problem continues with other kinds of data, like postal codes. There is no official set of postal codes to city names and geographic location mappings, and the search function the Swiss Post offers a confusing combination of results when used.[1] While private individuals like rueegger.me supply similar solutions, it gives off the image that the official administration responsible does not seem to prioritize open data. The picture drawn by these experiences is confirmed when looking at opendata.swiss[2]. This website presents open data about Switzerland, many sources are from cities like Zurich, Bern and Geneva. The Swiss federal government is also represented with topology data, but the contribution seems rather small in comparison to other sources.

4.3 Future Work

This thesis lays the barebone groundwork of Swiss company data visualization and clustering. There are different areas in which the thesis could be expanded. This includes, but is not limited to, the following subsections.

4.3.1 Data Acquisition

To improve the data set, one could acquire the company data directly from the cantonal commercial registry offices (e.g., Commercial Registry Office Zürich).
To enrich the already existing company data, one could also rely on buying more information from sites like Moneyhouse or Kompass. Another approach could be the sampling of smaller data sources and enriching the existing database entries by smaller portions of data, which might not be as complete as the central registry. One source that comes to present itself prominently would be the list of companies that SKV, Swiss SME association, maintains.[3] If this data is usable remains to be examined. In addition to the already mentioned approaches, simply searching the phone book could lead to new data for the database. With a lot more time, writing a web crawler might prove useful in gathering more information but this would require a significant time effort to get usable data.

4.3.2 Data Processing

With the publication of new papers on the topic of data processing and machine learning, as well as the implementation of new frameworks, novel ways to work through the data set and create better clusters with those technologies will arise. Also, it would be interesting to apply used techniques to other data sets on a similar topic with less noise. The result would give hints to what could have been done differently in the case of existing data sets.
A different approach would be to work with pre-classified data; building training sets upon them might lead to better results than pure topic clustering.

[1] https://www.post.ch/de/pages/plz-suche
[2] https://opendata.swiss/en/
[3] https://www.kmuverband.ch/firmenliste.html

4.3.3 Data Visualization

On the topic of data visualization, a continuation could mean more traditional data plotting, like pie charts and other static displays, and a more modern approach. An example of this could be a more detailed connection map between companies if the business owner can be identified. Like this, a network of people could be created. Sadly, this was not possible because signing parties were not listed in the central registry.
Expanding on the data gathering, additional tools to process the usage data could be written and be presented for easier understanding of what users look for on the web app. Additional quantitative user studies could be conducted to find other possibilities on how to improve the web app on the user experience level and to gather information about the usage of the website itself.
As mentioned in the approach, Google Analytics would have been a tool that could have been valuable to implement instead of the cookie-based self-implemented user tracking. In future iterations, one could add a Google Analytics integration to track other user behavior too on the site.

4.3.4 Conclusion

The Swiss business landscape kept being a field of great interest during this thesis. The applications and scripts created during the writing of this thesis build a basis for automated categorization and transformation while delivering visualizations on the companies with the necessary information. Different obstacles were encountered while fetching the data and categorizing them correctly. The problems identified were mostly caused by the data, which was difficult to obtain and of suboptimal quality. During the business clustering, different categorization methods were tested, but in the end, a simple keyword mapping was the most successful option.
The tool saw its first success when conducting the user/case study, and helped identify companies with similar purposes around a relatively small business. Nevertheless, the web app received constructive feedback regarding its aesthetics and a lack of usability hints. Some of these problems were fixed in an update to the web application. In general, the underlying data set forms a useful basis for continued research. It is easily updatable while being expandable for new data with the limitations being that the upgradeability via script is dependent on the unofficial Zefix API.
As a closing statement, one could say that it was possible to build a company database from publicly available information and use the data to create simple business clusters that then could be visualized and show connections between companies to a certain degree. There have been significant problems during this thesis with the dataset, but in the end, it was possible to circumnavigate those and create a working application which provides value to its users.

Acronyms

API application programming interface. ix, 6, 8, 9, 10, 11, 12, 13, 21, 22, 23, 24, 27, 34, 35

BSON Binary JSON. 5

CSV Comma-separated Values. 6

EHRA Eidgenössisches Amt für das Handelsregister (Swiss Federal Office for the Commercial Registry). 5

HTML Hypertext Markup Language. 22, 23

HTTP Hypertext Transfer Protocol. 8

IP Internet Protocol. 11

JSON JavaScript object notation. ix, 5, 7, 8, 9, 22, 23

LDA Latent Dirichlet Allocation. 2, 17, 18

RAV Regionalen Arbeitsvermittlungszentren (Regional Employment Centre). 8

REST Representational State Transfer. 8, 10, 23, 33

SHAB Schweizerisches Handelsblatt (Swiss Official gazette of commerce). 5, 8, 11, 12, 21

SKV Schweizer KMU Verband. 34

SME Small and Medium-Sized Enterprises. 1, 27

SOAP Simple Object Access Protocol. 8, 10

SQL Structured Query Language. 10, 13

TFIDF term frequency–inverse document frequency. 2, 17, 18

UI user interface. 31

URL Unique Resource Locator. 5, 8

UUID universally unique identifier. 24

XML Extensible Markup Language. 8

Appendix A

Program Usage

A.1 Requirements

There are some applications which need to be installed. The whole software was written on an Ubuntu 20.04 but the way it was designed this application should run under the most common modern operating systems at the time of writing.

Docker

Docker should be installed from docker directly or a trusted source. Community version 19.03.08 (Server/Client engine) was used during the making of this thesis.

Python

Python 3.8 was used to create the data script, this means it's not compatible with Python 2. As seen in the requirements.txt file for the python scripts, we need to install multiple dependencies. Everything needed to run all scripts is listed there as well as below.

- cwsplit, v0.4.1
- fasttext, v0.9.2
- gensim, v3.8.3
- joblib, v0.14.1
- matplotlib, v3.2.1
- nltk, v3.5
- numpy, v1.17.4
- pandas, v1.0.3
- pyenchant, v3.0.1
- pymongo, v3.10.1
- scikit-learn, v0.22.2.post1
- scipy, v1.4.1

- sklearn, v0.0
- wordcloud, v1.7.0

Golang

Golang version 1.14 was used to build the data scraping program (initial fetcher, change fetcher, server client system to get additional data). Any lower versions should be used with caution.

Node.js

Node.js is a JavaScript engine used to develop sever-side and front-end application. Version 14 was used to develop this application.

Yarn

Yarn is a package manager used to download npm packages for the visualization part. Yarn 1.22.4 was used to fetch the packages for this build. There are multiple other dependencies used during the build of this project, they can all be seen in the lock file of the source code.

MongoDB

MongoDB was the database used during this project. Version 4.2.5 was used during development.

A.2 Running The Code

To run the application code, all prerequisite must be satisfied. Every subsection of this section needs to have a MongoDB instance running to save data to and or read the data. In the scripts folder a script with the name runDb.sh is found. It can be started using ./scripts/runDB.sh. It is important to note that the MongoDB folder with the saved database information needs to be in the same directory as the scripts folder.

A.2.1 Data Scraper

There are multiple scraping applications that need to be run in the right order. First we need to get the IDs when scraping the whole Zefix database. Then the api of the gazette of commerce is scraped and as last step the missing businesses get fetched separately by ID with our server-client system.

Core Scraper

Use `go run server/CoreDataScrapper/*.go` to run the search scraper, which will return the api entries. This is destructive and will delete existing entries out of the database if the ehraid match.
To use the shab scraper use: `go run server/CoreDataScrapper/*.go -shab` This will delete the already existing transformation field on the database like geo and langInfo. They need to be newly generated.
Both of the above applications do have additional flags which can be set in case of a crash or

power loss. Those flags help to pick up where the application left of. They are nicely documented in the server/CoreDataScrapper/main.go file in the main method and can be found there.

To fetch the IDs that do not have a corresponding business we need to fetch them with our client-server system.
Start the server by running `go run server/DataEnricher/*.go -server` and the client with `go run server/DataEnricher/*.go -address=localhost` and change the "localhost" string to the ip of your server. Make sure to start the server first.
The server and client can also be compiled with `Golang` into binaries so host and client OS do not need to have go installed. It can be compiled by replacing `run` with `build` in the command below. To build it for other operating systems then your own, search the Golang documentation.

A.2.2 Data Transformation

Here a bundle of script is used. The order is important for a few scripts so try to keep it like the list below.

1. Set the language with `python3 NLPWorker/setLanguage2.py`
2. Save stemmed words with `python3 NLPWorker/saveStemmer.py {langName}` where {langName} is the language short (de/fr/it)
3. Save geographical information with `python3 NLPWorker/geoInfos.py`
4. Save business sector with `python3 NLPWorker/naivClusterin.py {langName}` where {langName} is the language short (de/fr/it)
5. Index purpose and title for search with `python3 NLPWorker/indexSearch.py`
6. Build city and canton structures with `python3 NLPWorker/buildCities.py` and `python3 NLPWorker/buildCantons.py` respectively.

For step 1, the fasttext language detector binary has to be downloaded on:
`https://fasttext.cc/blog/2017/10/02/blog-post.html`

A.2.3 Data visualization

To run the visualization locally we need to have the `Golang` backend and the JavaScript frontend up. For local development you can start the server with `ENV_SOURCE="http://localhost:8080" go run server/DataServ/*.go` and the client with `cd countryviz && yarn && yarn start` To deploy the application we need to build the JavaScript web app and serve it with a web-server like nginx or apache. Run `yarn build` for that in the countryviz folder. To compile the `Golang` server do the same was in the run command but replace `run` with `build` and leave out env variable `ENV_SOURCE="http://localhost:8080"`. To build it for other operating systems then your own, search the Golang documentation.

Appendix B

Additional Data

B.1 Keyword Map List

Table B.1: Keyword list - primary sector

Category	DE	FR
Agriculture	landwirtschaft, forstwirtschaft, Förster, Fischerei, Forst, Bauer, Landwirtschaftsunternehmen, Fischfang, Jagen, bauernhof, hof, landwirtschaftlichen, Viehzucht, Braunviehzucht, gnadenhof, zucht, aufzucht, ei, schweinemästerei, mästerei, schweinehaltung, tierhaltung, milch	agriculture, sylviculture, forestier, pêche, paysan, fermier, chasser, ferme, agricole, bétail, élévage, engraissement, lait, oeuf

Table B.2: Keyword list - secondary sector

Category	DE	FR
Mining	bergbau, Steinbruch, Erde, Erd, Abbau, abbau	minier, carrière, terre, minerai, extraction
Manifacturing	Herstellung, verwertung, metzgerei, spinnen, flechten, schneider, schneiderei, sägerei, bäckerei, bäckereibetrieb, schneideratelier, Brauerei, Bierbrauerei, Design, Produktion, Warenherstellung, Verarbeitung, Waren, stoffe, metall, metallisch, metallbau, veredelung, bearbeitung, erstellung	frabrication, production, manufacturer, boucherie, filage, torsader, couturier, tailleur, couture, scierie, boulangerie, brasserie, Design, transformation, tissu, métal, métallurgique, métallique, façonnage
Energy / Environment	Energieherstellung, Energie, Kraftwerk, Kernkraftwerk, Wasserkraft, Energiezulieferung, Energieinfrastruktur, Energietechnik, Wasserversorgung, Wasserwerk, Kläranlage, wasserreinigung, Umweltverschmutzung	énergie, éléctricité, hydroéléctrique, approvisionnement, pollution
Construction	Bau, Fräsarbeiten, maurer, akustikbau, Bauarbeiten, gartenbau, Holzarbeiten, mörtel, maler, malerarbeit, montage, Bauabdichtungen, betonsanierung, Gartenbauunternehmung, isolierungsarbeit, Verlegen, bedachung, fassaden, flachdach,gartenpflege, Baugewerbe, Häuserbau, rückbau, Installationen, Restaurationsbetrieb, beton, sanierung, umbauten, renovationen, Hochbau, Tiefbau, Schreinerei, Möbelschreinerei, Schreiner, heizung, anlagen, klima, sanitär, kälte, innenausbau, untergründen, Fertigung, Konstruktion, Elektroinstallation, Innenausbauarbeiten, Fertigung, möbel	construction, fraisage, maçon, acoustique, horticole, horticulture, menuiserie, mortier, peintre, peinture, raccordement, installations, assainissement, isoler, installer, poser, toiture, façade, toit,jardin, installations, béton, rénovations, génie, ébénisterie, menuiserie, menuisier, chauffage, installation, climat, sanitaire, froid, consturction, éléctrique, meubles

Table B.3: Keyword list - tertiary sector

Category	DE	FR
Trade / Cars	tabak, export, import, autohandel, veräusserung, lebensmitteln, karosseriewerkstatt, detailhandel, autos, occasion, carrosserie, geschenk, getränkehandel, einfuhr, onlinehandel, buchhandlung, Marktstandes, Optikergeschäften, produkte, wein, blumen, webshop, Floristik, vertrieb, Apotheke, Drogerie, Kleidern, möbel, kioskbetrieb, esswaren, handel, verkauf, konsumgüter, kiosk, Autogewerbes, lebensmittel, verbrauchsgüter, verkauf, Güter, Ankauf, Werkstatt,Reparatur, Instandsetzung, Instandhaltung, Kraftfahrzeuge, Automobil, Auto, PKWs, PKW, Carosserie, Neuwagen, Fahrzeug, LKW, Lastwagen, Fahrzeuge, garage, ersatzteile, fahrzeugzubehör	tabac, export, import, exportation, importation, cession, alimentaires, aliments, carrosserie, vente, auto, voiture, occasion, carrosserie, cadeau, commerce, librairie, étal, opticien, produits, vin, fleurs, webshop, fleuriste, distribution, pharmacie, droguerie, vêtements, vestimentaire, meubles, kiosque, comestibles, denrées, achat, atelier,réparation, réfection, maintenance, automobile, carosserie, véhicule, camion, véhicules, garage
Traffic / storage	Logistik, transportunternehmen, kleintransporte, lieferungen, kurierdienste, carreise, taxiunternehmen, gütertransport, onlineshops, umzüge, umzug, Transport, Güter, verschiebung, lastwagen, Bahn, Personentransport, taxi, taxibetrieb, Regiefahrern, Beförderung, taxigeschäft, Kurierunternehmen, Räumung	logistique, transport, livraisons, fournitures, messagerie, autocar, taxi, déménagement, déménagements, train, manutention
Hospitality industry	Hotel, gasthof, Restaurant, pizzeria, essen, trinken, tearoom, imbiss, pizzaria, kebap, takeaway, service, Catering, Caterings, Tourismus, Reka, bar, rezeption, zimmmer, übernachtungen, herberge, Massenschlag, Bed, Breakfast, BNB, Stern, frühstück, Vollpension, Gast, away, Cafés, Café, Spezialitäten, Gäste, Verpflegungsständen, verpflegung, Gastgewerbebetriebes,restaurantbetrieb, gastronomie, gastronomiebetrieb	hôtel, auberge, restaurant, pizzeria, manger, boire, café, snack, snack-bar, bar-snack, kebab, kebap, takeaway, service, traiteur, restauration, tourisme, Reka, bar, réception, chambre, nuitée, Bed, Breakfast, BNB, déjeuners, pension, away, Cafés, spécialité, alimentation, gastronomie

Information / Communication	presse, druck, buchverlag, Zeitschriftenverlag, print, printmedien, zeitung, drucklegung, gratiszeitung, news, information, kommunikation, , geräte, telefon, internet, SMS, telefonieren, anschlüsse, Druckerzeugnissen, verlagsgeschäft, zeitungsverlag,Informatik, IT, EDV, IT-consulting, Telekommunikation, Computer, Informatiker, Programmierung, Internet, Internetanbieter, telekommunikation, kabelfernsehen, Netzwerk, Anschluss, science, ITBeratung, EDVSysteme, soft, hardware, webdesign, elektronik, softwareentwicklung, software, datenschutz, hard, informationstechnologie, webseiten	presse, impression, êdition, magazine, revue, print, journal, news, informations, communication, appareils, téléphone, internet, message, téléphoner, correspondance, informatique, IT, EDV, IT-consulting, télécommunications, ordinateur, computer, informaticien, programmer, Internet, télécommunication, télé, réseau, connexion, science, consultation, systèmes, soft, hardware, webdesign, élctronique, logiciel, données, web
Finance / Insurance	Darlehen, vermögen, vorsorge, grundeigentum, schuldschein, Vermögensverwaltung, anlageberatung, Geld, treuhand, wertpapier, treuhandgesellschaft, buchhaltung, finanzbuchhaltung, Salärbuchhaltung, Finanzen, Finanz, Beratung, gesetz, rechtsdienstleist, Notariatsdienstleistungen, finanzdienstleistung, Versicherung, treuhandbereich, Wertschiften, versicherungsbereich, bank, transfer, kredit, Finanzanlagen, lebensversicherung, Beteiligungen, konzessionen, Haftpflichtversicherung, beteiligungen, rechte, patente, anlagen, betriebswirtschaft, wertschrift, finanzierungsgeschäfte, Effektenhandel	prêt, fortune, prévoyance, foncier, cédule, argent, fiduciaire, titre, comptabilité, finances, financier, consultation, loi, notaire, assurance, banque, transferet, credit, participations, concessions, droits, patent, gestion
Housing	Grundstück, immobili, Liegenschaft, verwaltung, vermietung, verkauf, immobilien, Architekturbüros, Architektur, projektierung, architektur,Überbauung, immobilenberatung, wohnung, Unterhalt, Verwaltung, bauten, Innenarchitektur, Architekturbüro	Immeuble, foncier, administration, louer, vente, propriétê, architecte, architecture, appartement, entretien, subsistance , intérieur

Education	Schulung, Schule, Ausbildung, Weiterbildung, Erziehung, kinder, kind, tagesstätte, kinderkrippe, krippe, lernen, bildung, bildungstätte,kinderkrippe, universität, kleinkindererziehung, gefängniss, kita, kindertagesstätte, Kinderbetreuung, kurse, lehranstalt, erteilung, unterrichtung, stunden, nachhilfeunterricht, tanzschule, sprachschule, seminar, schulungen	formation, école, apprentissage, éducation, enfants, enfant, garderie, crèche, apprendre, université, prison, garder, course, enseigner, instruire, professer, cours, soutien, danse, langue, seminaire
Health / Social	coiffeursalon, coiffeur, haare, kosmetikstudios, haareschneiden, stress, hairstyling, haar, Schönheitssalons, Gesundheit, physio, Physiotherapie, fitness, sozial, pflege, wohlfahrt, medizin, chirurgischen, veterinarmedizin, medizinaltechnik, betreuung, altersheim, spital, arzt, zahnarzt, ärzt, gesundheitswesen, krankenpflege, operation, krankheit, heilung, therapie, praxis ,Arztpraxis, innere, medizinischen, pflegheim, demenz, massagen, neurologie, Coiffeurprodukten, Coiffeurgeschäftes, Färben, Makeup, Kosmetik, Pflegeeinrichtungen, patienten	salon, coiffeur, cheveux, cosmetique, couper, ongle, stress, hairstyling, cheveux, beauté, santé, physio, Physiothérapie, fitness, social, soins, aide, médicine, chirurgical, vétérinaire, solicitude, retraite, hôpital, médecin, dentiste, docteur, santé, opération, maladie, guéèrison, cicatrisation, thérapie, interne, médical, médico-social, démence, massages, neurologie, teinter, maquillage, cosmétiques, patients
Art / Entertainment / Recreation	kunst, bücher, freizeit, sport, freizeitanlage, sportanlage, festival, openair, filme, welt, theater, künstler, tanz, ausstellung, reisevermittl, kultur, literatur, kino, lichtspielhaus, film, anlässen, grafik, künstler, jass, dart, Künstlervermittlung, aufführungen, erholung, unterhaltung, malerei, Reiseveranstalter, Auslandsreisen, event, atelier, Festwirtschaften, Sportclubs, sportschule, fitnesscenter, fitness, Fitnessstudios, sportveranstaltungen, Kampfsportarten	art, livres, loisirs, sports, festival, openair, films, monde, theatre, artiste, décors danse, exposition, culture, literature, cinéma, film, occasion, graphique, artist, dart, représentation, spectacle, repos, délassement, divertissement, animation, peinture, voyagiste, voyage, event, atelier, sport, club, fitness, manifestations, martiaux

Administrational	personal, werbung, management, hr, humanresources, personalvermittlung, marketing, marketingdienste, consulting, administration, administrationsleistung, kommunikationsberatung, beratungsdienstleist, administrativ, managementdienstleistungen, coaching, Unternehmensberatung	personnel, publicité, direction, rh, ressources, marketing, consulting, administration, administratif, coaching, conseil
Others	putzen, geputzt, reinigung, reinigungsarbeit, reinigungsunternehm, entsorgung, schloss, schlüssel, bestattungsinstitut	nettoyer, nettoyé, épuration, nettoyage, élimination, serrure, clé, funèbres

Appendix C

User Study

C.1 User Behavior Transcript Skippr ltd

1. User starts on "home" section of the web app.
2. User navigates to "statistics"
3. User switches from legal forms to sectors in statistics
4. User selects 'Information/Communication' from additional selection menu
5. User is tries to find a way to get to the canton of fribourg by trying to click on it
6. User changes to search view after being told that statistics works as source of information for cantons and is not usable for business searches
7. User types "web" and postal code of skipper ltd. main office in search bar and 'Information/Communication' as sector field then proceeds to clicks on search button
8. User tries again with only the postal code as search term this time
9. User tries "web fribourg" as search terms
10. Uer tries "web" alone as search term
11. User searches for "skippr" as keyword which delivers the right firm to his screen
12. User clicks on find similar companies which fills out the possbile search fields
13. User selects use location checkbox
14. User searches for businesses again with the pre-filled values
15. User seaches again with legal form "all", goes through the list

C.2 Tracked User Data


```
{
   timeStamp: Date("2020-06-03T08:18:15.324Z"),
   query: {},
   path: "/api/cantons"
},
{
   timeStamp: Date("2020-06-03T08:20:01.891Z"),
   query: {
      type: [
         "8"
      ],
      search: [
         "web, 1700"
      ]
   },
   path: "/api/businesses"
},
{
   timeStamp: Date("2020-06-03T08:20:06.553Z"),
   query: {
      search: [
         "1700"
      ],
      type: [
         "8"
      ]
   },
   path: "/api/businesses"
},
{
   timeStamp: Date("2020-06-03T08:20:16.688Z"),
   query: {
      type: [
         "8"
      ],
      search: [
         "web fribourg"
      ]
   },
   path: "/api/businesses"
},
{
   timeStamp: Date("2020-06-03T08:20:24.600Z"),
   query: {

      type: [
```

```
            "8"
         ],
         search: [
            "web"
         ]
      },
      path: "/api/businesses"
   }
   {
      timeStamp: Date("2020-06-03T08:20:45.959Z"),
      query: {
         type: [
            "8"
         ],
         search: [
            "skippr"
         ]
      },
      path: "/api/businesses"
   },
   {
      timeStamp: Date("2020-06-03T08:21:16.129Z"),
      query: {
         legal: [
            "gmbh"
         ],
         type: [
            "8"
         ],
         search: [
            "onlinemarketing werbung informationstechnologie applikationen
               erstellen webseiten anbieten vorgenannten nebenzweck"
         ],
         radius: [
            "30"
         ],
         long: [
            "7.19101182937966"
         ],
         lat: [
            "46.8447407857849"
         ]
      },
      path: "/api/businesses"
   },
   {

      timeStamp: Date("2020-06-03T08:23:31.838Z"),
```

```
   query: {
      legal: [
         "all"
      ],
      type: [
         "8"
      ],
      search: [
         "onlinemarketing werbung informationstechnologie applikationen
             erstellen webseiten anbieten vorgenannten nebenzweck"
      ],
      radius: [
         "30"
      ],
      long: [
         "7.19101182937966"
      ],
      lat: [
         "46.8447407857849"
      ]
   },
   path: "/api/businesses"
}
```


Listing C.1: Traced user data (unnecessary search fields removed)

C.3 User Feedback

- Application should be explained in clearer way. Generally views should be explained.
- Search application should be able to handle postal code searches or something similar to search for an area before a company is selected.
- Not all companies are sorted to their respective business sectors correctly.

Bibliography

[AG20a] Kompass Schweiz AG. Firmenverzeichnis - search for company information - kompass. https://ch.kompass.com/, 2020. Accessed: 2020-02-28.

[AG20b] Moneyhouse AG. Moneyhouse - commercial register and business information. https://www.moneyhouse.ch/en/, 2020. Accessed: 2020-02-28.

[AG20c] Moneyhouse AG. Moneyhouse - data sources. https://www.moneyhouse.ch/en/datasources, 2020. Accessed: 2020-02-28.

[BLB+13] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[Bro19] Jason Brownlee. How to clean text for machine learning with python. https://machinelearningmastery.com/clean-text-machine-learning-python/, August 2019. Accessed: 2020-06-05.

[dB20] Informatiksteuerungsorgan des Bundes. Burweb. https://www.isb.admin.ch/isb/de/home/e-services-bund/services/buraweb.html, 2020. Accessed: 2020-05-03.

[fLs20] Bundesamt für Landestopografie swisstopo. swissboundaries3d. https://shop.swisstopo.admin.ch/en/products/landscape/boundaries3D, 2020. Accessed: 2020-06-10.

[GHK10] E. R. Gansner, Y. Hu, and S. Kobourov. Gmap: Visualizing graphs and clusters as maps. In *2010 IEEE Pacific Visualization Symposium (PacificVis)*, pages 201–208, 2010.

[GZSC19] Jadson Castro Gertrudes, Arthur Zimek, Jörg Sander, and Ricardo JGB Campello. A unified view of density-based methods for semi-supervised clustering and classification. *Data Mining and Knowledge Discovery*, 33(6):1894–1952, 2019.

[HBGSS14] Yuening Hu, Jordan Boyd-Graber, Brianna Satinoff, and Alison Smith. Interactive topic modeling. *Machine learning*, 95(3):423–469, 2014.

[HD10] Liangjie Hong and Brian D Davison. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88, 2010.

[Hol03] Heinz Hollenstein. Innovation modes in the swiss service sector: a cluster analysis based on firm-level data. *Research policy*, 32(5):845–863, 2003.

[incnda] MongoDB inc. Json and bson. `https://www.mongodb.com/json-and-bson`, n.d. Accessed: 2020-06-14.

[incndb] MongoDB inc. Mongodb docs. `https://docs.mongodb.com/manual/tutorial/specify-language-for-text-index/`, n.d. Accessed: 2020-06-10.

[JGB+16] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. *arXiv*, abs:1612.03651, 2016.

[JGBM16] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv*, abs:1607.01759, 2016.

[Jin12] Y. Jin. A topic detection and tracking method combining nlp with suffix tree clustering. In *2012 International Conference on Computer Science and Electronics Engineering*, volume 3, pages 227–230, 2012.

[Kea13] T Alan Keahey. Using visualization to understand big data. *IBM Business Analytics Advanced Visualisation*, 2013.

[LB14] Marco Lui and Timothy Baldwin. Accurate language identification of twitter messages. In *Proceedings of the 5th workshop on language analysis for social media (LASM)*, pages 17–25. Association for Computational Linguistics, 2014.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv*, abs:1301.3781, 2013.

[Mer14] Dirk Merkel. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239), March 2014.

[oJP20] Federal Departement of Justice and Police. Zefix – central business name index. `https://www.zefix.ch/en/search/entity/welcome`, 2020. Accessed: 2020-02-28.

[PSKN06] C. Panse, M. Sips, D. Keim, and S. North. Visualization of geo-spatial point sets via global shape transformation and local pixel placement. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):749–756, 2006.

[QA18] Shahzad Qaiser and Ramsha Ali. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181:25–29, 07 2018.

[Qia06] Y. Qian. An application based on k-means algorithm for clustering companies listed. In *2006 IEEE International Conference on Service Operations and Logistics, and Informatics*, pages 723–727, 2006.

[Ros14] Brandon Rose. Document clustering with python. `http://brandonrose.org/clustering`, December 2014. Accessed: 2020-06-05.

[ŘS10] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[Rü15] Samuel Rüegger. Schweizer postleitzahlen mit koordinaten. https://rueegger.me/blog/schweizer-postleitzahlen-mit-koordinaten/, June 2015. Accessed: 2020-05-03.

[SB07] Douglas Steinley and Michael J Brusco. Initializing k-means batch clustering: A critical evaluation of several techniques. *Journal of Classification*, 24(1):99–121, 2007.

[SEC20] SECO. Swiss offivial gazette of commerce sogc. https://www.shab.ch/, 2020. Accessed: 2020-02-28.

[Shu10] Nakatani Shuyo. Language detection library for java. https://github.com/shuyo/language-detection, 2010. Accessed: 2020-06-10.

[Sin17] Vikash Singh. How we changed unsupervised lda to semi-supervised guidedlda. https://www.freecodecamp.org/news/how-we-changed-unsupervised-lda-to-semi-supervised-guidedlda-e36a95f3a1 October 2017. Accessed: 2020-06-03.

[SS17] L. I. Sharawi and G. Sammour. Utilization of data visualization for knowledge discovery in modern logistic service companies. In *2017 Sensors Networks Smart and Emerging Technologies (SENSET)*, pages 1–4, 2017.

[str19] *Struktur der Schweizer KMU 2017*. The Swiss Federal Statistical Office, Oct 2019. Accessed: 2020-04-02.

[Sun14] X. Sun. Textual document clustering using topic models. In *2014 10th International Conference on Semantics, Knowledge and Grids*, pages 1–4, 2014.

[Vvv06] R. Vliegen, J. J. van Wijk, and E. van der Linden. Visualizing business data with generalized treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):789–796, 2006.

[WLWK08] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3):1–37, 2008.

[YCH03] Christopher C Yang, Hsinchun Chen, and Kay Hong. Visualization of large category map for internet browsing. *Decision support systems*, 35(1):89–102, 2003.

[YMNO17] A. Yamamoto, Y. Miyamura, K. Nakata, and M. Okamoto. Company relation extraction from web news articles for analyzing industry structure. In *2017 IEEE 11th International Conference on Semantic Computing (ICSC)*, pages 89–92, 2017.

[YW08] G. G. Yen and Z. Wu. Ranked centroid projection: A data visualization approach with self-organizing maps. *IEEE Transactions on Neural Networks*, 19(2):245–259, 2008.

[YY14] D. Yunzhao and Y. Ye. Visualization of micro-credit company development in china. In *2014 International Conference on Management of e-Commerce and e-Government*, pages 379–382, 2014.

[ZJW+11] Wayne Zhao, Jing Jiang, Js Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing twitter and traditional media using topic models. volume 6611/2011, pages 338–349, 04 2011.