Department of Information Engineering and Computer Science

Bachelor's Degree in
Computer Science

Final dissertation

Automatic generation of marketing personas from social media data

Supervisors

Alberto Montresor

Carlo Caprini

Student

Nicola Farina

Academic year 2020/2021

Acknowledgments

I want to thank all the people that have supported me during these three years: my parents, who never failed to provide me with everything I needed and were always there for me; and my closest friends, who lightened my mood in my most stressful periods.

A huge thank you goes also to the people that helped and guided me in writing this thesis: first of all, Daniele Miorandi and Carlo Caprini, who relentlessly offered me the best advice and support whenever I needed it during my internship; all the U-Hopper team, which unfortunately I could not meet in person due to Covid restrictions, but who still made me feel like I was working in such a friendly environment; and Alberto Montresor, who is responsible for introducing me to U-Hopper in the first place.

Contents

Abstract

Nowadays digital marketing heavily relies on the segmentation of an audience at both aggregated and at the single user's level. Personas are profiles of fictional users that "humanize data" by giving an identity to each audience segment; they are used by marketeers to better understand their customers and make marketing decisions for each customer segment in an efficient way.

Personas are traditionally created manually, by collecting data through interviews and surveys and analyzing it in a lengthy and expensive process. For this reason, in the latest years, the concept of automatic persona generation has increased in popularity. Nonetheless, very few services exist that offer such a solution, and those that do exist share limitations in either how or what data is collected to create the personas.

The best and easiest place to look for data to build personas on is social media: they are one of the most used means of communication, and offer a great source of both demographics and behavioral data, which is precious in order to create meaningful personas. This work explores the possibility of using social media data as a source for creating marketing personas.

This thesis was developed at U-Hopper[1], a company that specializes in big data analytics and artificial intelligence, and that is already present in the world of customer knowledge.

The proposed solution not only shows that automatic persona generation from social media data is feasible, but also that a system with high scalability, expandability and accuracy can be designed and implemented to fulfill that task. It uses several components, each with a specific task, from data collection, to data enrichment, all the way to clustering and persona generation. Many classifiers are employed to extract insights from the available data, which are then used to create customers segments. All the components are linked in a stream-processing architecture, which allows to avoid bottlenecks and achieve high separation of concerns. Finally, users are segmented with very good accuracy, precision and recall measures, all above 0.9.

[1] `www.u-hopper.com`

1 Introduction

This thesis has been developed at U-Hopper, during a 4-month period in which I was able to combine a work experience in a company with the writing of my bachelor's thesis. U-Hopper is a company that defines itself as a "Data Intelligence Lab", with an expertise on Big Data Analytics, Business Intelligence and Artificial Intelligence. Headquartered in Trento, it provides big data-enabled solutions and technologies, for which it received numerous awards throughout the years.

The company has brought its presence into the user profiling world by developing Tapoi[1], a service that makes heavy use of artificial intelligence and machine learning algorithms to provide human-centered insights on existing customers. The existence of Tapoi is what inspired the work of this thesis, which proposes to further humanize customer data in order to make marketing decisions that result in a better custom experience. What follows is a brief introduction of the problem at hand.

1.1 Problem statement

Marketing is a crucial aspect for the success of any brand offering a service. Especially in today's globalized market, good advertising and communication strategies are the key to stand out amongst the competitors. Ultimately, marketing boils down to one thing: knowing your customers. With this knowledge, a company can understand the actual needs of their clients and design solutions that will satisfy them, ultimately increasing profit. In fact, customers, now more than ever before, want a personalized experience when interacting with a brand, which spans throughout the whole interaction (from product recommendations, to tailored emails).

A popular way to achieve customer knowledge is through *personas*. A persona is an imaginary person representing a real user segment, that is, a group of people that have similar characteristics. Their power is that they allow to abstract a high number of customers, even millions, into only a few, easy to read profiles. Marketing specialists and sales teams can then use the data contained in these personas to make informed decisions on their marketing campaigns and to design the aforementioned personalized experiences.

Traditionally, these personas are built manually by marketing specialists based on the outcome of interviews, focus groups and surveys. This approach is lengthy and expensive, as it often involves direct contact with customers and manual processing of a lot of data. The purpose of this thesis is to explore the possibility of automating this process.

In particular, we propose the following research question: is it possible to automatically generate marketing personas for a brand or organization, based on public data available on the legal basis? Let us better define some terms:

- **automatically**: the user is not required to make decisions nor possess knowledge on how to create personas in order to use the service;
- **publicly available data**: data that can be collected publicly. Some examples are: social media, like Twitter or Facebook; other websites, for example blogs or sector-specific services like GitHub; third party data, such as national surveys or company data;
- **legal base**: the service needs to be fully compliant with GPDR, the General Data Protection Regulation.

This thesis shows that fulfilling this task is indeed possible, and proposes a system design to carry it out, together with a working prototype for demonstration purposes.

[1] `www.tapoi.cloud`

1.2 Personas

Personas were first introduced by Alan Cooper in 1999 [8] in the context of software development. They are "*fictitious, specific and concrete representations of target users*" [22] to be used throughout the design process. With time, the persona technique has expanded to other fields, such as design, marketing, healthcare [32], and even games [31].

Given the high number of use-cases for personas, the information they contain and the way data is collected cannot be uniquely specified and can vary significantly between each project. Regardless, some information is commonly found in most personas, as can be seen in many general-purpose persona templates found on the Internet. This includes: demographics, such as name, age, place of residence, marital status or living situation, profession, photo; and psychographics, like personality traits, hobbies, interests and opinions, values, lifestyle and habits, frustrations, needs and motivations.

1.3 Outline

This thesis is organized as follows: Chapter 2 presents the state of the art related to the persona creation process, which is needed to understand the current research problems and gaps. Chapter 3 describes the design solution, with particular focus on the logical components of the system and on data modeling. In Chapter 4, the implementation choices are presented and justified, while in Chapter 5 the implemented prototype is evaluated. Chapter 6 concludes this thesis with some final remarks for future work.

2 State of the art

In this chapter, the state of the art regarding the creation of personas is presented. It is divided in four sections which reflect the main steps of the process:

1. **data collection**, which consists in getting data about the users on which personas are built;
2. **data enrichment**, which consists in analysing the previously collected data to extract additional insights.
3. **clustering**, which consists in identifying groups of similar users based on the characteristics obtained in the previous two steps;
4. **persona generation**, which consists in creating a persona for each identified group.

Each section will cover the work that has been done so far, with emphasis on the latest developments in the use of enriched social media data. The current research problem of fully automating the whole process is tackled as well.

2.1 Data collection

In this section, the main data sources and methods used to create personas are presented in chronological order, starting from hands-on qualitative methods, followed by surveys and the latest developments with web data.

2.1.1 Qualitative methods

Traditionally, data was collected using purely qualitative methods. Those can be split into two categories: implicit and explicit. Implicit methods are described by the concept "don't ask, observe" and include observations, field studies and, in part, usability tests [18, 23, 21]. Explicit methods rely instead on directly asking questions to users: the most used ones are interviews and focus groups [18, 23, 21]. All these interactions, whether implicit or explicit, have to be carefully crafted, usually by experts of the given research field, in order to gain useful insights to be later analysed.

Because of their open-ended nature, qualitative methods allow researchers to explore user behaviour in depth, but they also come with some shortcomings:

- **small coverage**: the research is usually done with few users (10-20) [18]. Thus, there is no evidence that personas actually represent the full user base [7];
- **subjectivity**: interview results and related qualitative data are subjective by nature, and can also present biases from both users and researchers [18].

2.1.2 Surveys

With the aforementioned flaws in mind, the concept of data-driven personas was introduced. It consists in grounding personas in real, larger-scale data by using quantitative research methods alongside qualitative ones (or completely replacing the latter) [18]. The most popular data sources to achieve this are surveys: 47% of papers about data-driven personas development report their use [25]. They are easy to share, allow a broad exploration of the user base, and are easier to cluster than open-ended interview results.

R. Sinha was one among the first ones to report the use of surveys in 2003, when developing personas for a restaurant recommendation system [28]. She and her team crafted a survey after having identified 32 relevant dimensions of the restaurant experience, asking users to rate them on a five point Likert scale. The same approach was taken by McGinn and Kotamraju in 2008 [17], who

spent three weeks designing a survey together with the stakeholders and got 1300 responses, much more data than qualitative methods can achieve.

The use of surveys was further explored in an article by Tu et al. in 2010 [29]. They begin by proposing four steps for the persona creation process, with the first one reading: "*gather user data (personal data, users' relationships with the product and users' goals and motivations) through large scale questionnaires and user survey*". They highlighted that the design of the questionnaire is crucial in order to get relevant dimensions for clustering, and that it should focus on capturing user behaviors, goals and motivations. To build theirs, they used a template designed by George Olsen in his article "*Persona creation and usage toolkit*" [21], but noted that it was difficult to fully adapt it to their situation, suggesting that a "universal" template may not exist.

2.1.3 Web data

The next development in data collection came with the popularization of web data. Almost 30% of papers on data-driven personas report its use, and the number is increasing and surpassed that of surveys [25]. Web data includes sources such as social media platforms (e.g. YouTube [4]), online discussion forums [12], and online analytics (e.g. clickstream data [33]). The increasing popularity of the web also resulted in larger datasets and survey samples, with an average size about five times that of the first surveys [25].

An example of the use of online analytics was proposed by Zhang et al. in 2016 [33]. Clickstream data is defined as sequences of users' clicks (or taps) when using a product. Using it as a data source comes with the following benefits: clickstreams directly describe user behavior and workflow in a product; additionally, they are collected automatically without the need of human intervention; on top of that, the constant collection of clickstreams allows personas to be easily updated in case their workflows evolve over time. This approach is suitable when personas need to capture the user experience (e.g. e-commerce websites or smartphone applications).

Social media is currently being explored as a source of both demographical and behavioral data. It is a good candidate because of the ever increasing number of people who have social media accounts and share content online. The latter often includes insights into users' interests, opinions and demographics, drawing many similarities to what can be found in personas [13]. Furthermore, most social media sites provide APIs that allow to collect data programmatically on a large scale. At the same time, researchers who draw on social media to create personas need to be aware that the data contained in user profiles does not always represent the truth and that a large number of accounts are either fake or bots (Twitter shut down up to 70 million suspicious accounts in 2018 [20]). Moreover, the concern for privacy is nowadays more relevant than ever, and researchers need to fully comply with GDPR when dealing with user data.

Research on the field is mainly done by a team led by Dr. Jim Jansen at the Qatar Computing Research Institute, through the development of *APG (Automatic Persona Generation)*[1]. It is a service that focuses on automatically generating personas for a YouTube channel using YouTube analytics as the main source, with data such as comments, likes and view counts grouped by type of content and demographics [4]. Other tools available online propose the same kind of services, focusing on online and social media analytics[2]. This means that the user of the service needs to already have an established presence on the web (be it their website or their social media accounts) and an analytics tool to keep track of the related data (e.g. Twitter Analytics, Facebook Analytics, Google Analytics).

The use of more traditional social networks (over YouTube, which specialises almost uniquely in video content) has been explored to a lesser extent, with not much documentation available. An et al., when developing personas for Al Jazeera[3], reported the use of Facebook and Twitter data, but with some caveats [2]. The former was limited only to URLs shared by users who followed or talked about Al Jazeera's Facebook page, since Facebook API requires explicit user consent in order to access any personal data. The latter was also limited to users' biographies, with the purpose of extracting non-behavioral aspects such as occupation and hobbies.

[1] `persona.qcri.org`

[2] `www.delve.ai`, `www.mnemonic.ai`

[3] `www.aljazeera.com`

2.2 Data enrichment

Data enrichment has little documentation in persona literature: it became a necessary step only when social media was introduced as a data source, since information like demographics and behavior were not directly available but had to be extracted from user profiles and posts [25].

There are many studies that aim at inferring user demographics from social media, especially in the field of academia research (e.g. for studying gender disparity in publications). The best results are achieved with gender and age predictions [6]. For these purposes, many tools and APIs are available, which mainly use names or images to predict a user's most likely demographics. It should be noted though that these are only predictions, and should be treated with care in order to not introduce fake data into the final personas. Whenever possible, such predictions should be cross-validated between multiple data sources.

To infer user behavior, researchers' main approach is to determine a user's personality. Dos Santos et al. were the first to use this method, when they included the Big Five personality framework in their surveys to create a behavioral persona for a pet robot [9]. Since then, a lot of research has been conducted to extract personality from social media user profiles. The first approaches only used public metrics such as the number of friends, but they turned out to be inaccurate since users typically have a large number of friends and followers regardless of their personality. Later studies focused on the way users interact with their friends, including frequency and intensity of such interactions [1, 10].

Another great source of information is the content of social media posts. By analyzing the text or multimedia content of a user's posts, one can not only understand the author's interests, values and opinions towards a particular topic or product, but also their preferred language, the time they are most active and their tone of voice, which are all good candidates for the definition of a persona. Several NLP (Natural Language Processing) techniques are available to extract such insights from text: semantic analysis, to identify entities (such as nouns and corresponding adjectives) [16]; topic analysis, to either extract or assign topics and find users' topical interests [11]; sentiment scoring, to understand a user's attitude towards a particular topic [19, 11]. Deep learning solutions can be used as well to achieve similar results on images and videos [24]. It is important to consider this, as multimedia content is rapidly growing in popularity over text: according to Mark Zuckerberg, CEO and founder of Facebook: "*Most of the content 10 years ago was text, and then photos, and now it's quickly becoming videos*"[4].

2.3 Clustering

Clustering, also known as segmentation, is a crucial step in the process of persona creation, since its output determines the number of personas and the characteristics which tell them apart.

Traditional segmentation was done by hand, and consisted in analyzing the results of data collection in order to find patterns and common themes. This approach is highly subjective and often involves researchers "listening to their guts" [18], especially if they do not have much expertise in the field they are studying.

With the advent of quantitative research for persona creation, more objective and algorithmical methods started being used. At first, dimensionality reduction techniques were employed to uncover latent patterns in survey responses. Subsequent works adopted a variety of clustering algorithms, depending on the scenario for which personas were being created. Both of the approaches are presented in the following subsections.

2.3.1 Dimensionality reduction

As surveys can have a large number of questions (thus answers), oftentimes with latent correlations between them, it can be useful to reduce their amount while still preserving most of the insights they offer. One method to achieve this is exploratory factor analysis, a statistical technique whose goal is to identify the underlying relationships between measured variables. This method was only documented once in persona literature, by McGinn and Kotamraju in 2008 [17].

More popular and documented is the use of PCA (Principal Component Analysis) [25, 28, 29]. It is a dimension-reducing algorithm used to extract information by removing non-essential elements

[4] `www.fastcompany.com/3057024/mark-zuckerberg-soon-the-majority-of-content-we-consume-will-be-video`

with relatively fewer variations. While it was originally used on its own [28, 29], most of the recent research that employs PCA does so in conjunction with the clustering algorithms presented in the following section [25] in order to avoid the so-called curse of dimensionality[5].

2.3.2 Clustering algorithms

Clustering algorithms take a series of data points as input, and assign a cluster index to each one of them as output. What follows are the most popular ones employed in persona creation.

K-Means is an unsupervised machine learning algorithm which partitions data into k clusters, where k is a parameter that needs to be chosen by the user. Its goal is to find clusters in a way that data points in the same cluster are similar and data points in the different clusters are farther apart. Two things should be noted: firstly, similarity is defined by a distance metric. A common one is euclidean distance, but custom ones may have to be designed for non-trivial problems or when there is a mix of categorical and numerical data. Secondly, since the initialisation is random, results are not reproducible (this can be mitigated with an initialisation setting). In persona creation, K-Means has the flaw that the optimal number of personas k is not known, requiring the use of other methods to determine it. Nonetheless, it is the most employed clustering algorithm in persona literature [25].

Non-negative matrix factorization (NMF) is a matrix factorization method, in which a matrix V (n x m) is approximately factored into two matrices W (n x r) and H (r x m), with the property that all three matrices have no negative elements. Intuitively, the rows of V represent some objects (e.g. n different users), while the columns represent some properties (e.g. m user attributes). The number r is a parameter and is often chosen smaller than m to also achieve dimensionality reduction. NMF is mainly used when dealing with interactions of users with content: a practical example is presented by An et al. for extracting behavioral patterns from interactions between customer segments and YouTube videos [3]. Their matrix V had customer segments as rows and videos as columns, with each cell V_{ij} containing the view count of customer group i on video j. After factorization, they were able to extract video consumption patterns from matrix H and associate customer segments to each pattern through matrix W.

Hierarchical clustering (HC) is an unsupervised machine learning algorithm which produces a hierarchical order of clusters (arranged as a tree). It is an alternative to K-Means that grants reproducible results and does not need prior knowledge of the k parameter, but tends to be inefficient on large datasets both in terms of speed and memory usage. It only works well when there is an underlying hierarchical structure in the data.

Other clustering algorithms used in persona creation are Q-SIM and LDA. Q-SIM (Quality Similarity Clustering) was proposed by Masiero et al. in 2013 as an alternative to K-Means clustering [15]. Its main feature is that it replaces the hyperparameter k with a number representing the desired intra-cluster similarity degree. While they evaluated Q-SIM to perform better than K-Means in their case study, the algorithm has been adopted only once in the following years [9] and its implementation has not been supported further. LDA (Latent Dirichlet Allocation) is a NLP technique that can be adopted to find latent topics in text [5], and therefore applies only when clustering needs to be performed on textual documents.

2.4 Persona generation

This last step involves building the final persona profiles, one for each cluster. The approach has remained consistent through time, regardless of the methods being employed in the previous steps (qualitative, quantitative or mixed). Since each cluster comes with characteristics that tell it apart from the others, this step boils down to "making clusters come to life" [18] by giving them an identity. This is usually achieved by supplying a name, photo, demographics (if not already present in the clusters), biography, quotes and stories [18]. It should be noted that, since personas are meant to be fictional, such characteristics should not be taken directly from the attributes of one or more specific users, but should be abstracted from the data itself or generated from scratch. In doing so, one needs to pay attention to coming up with coherent attributes in order for personas to be believable (e.g. not assigning a male name to a female persona [14]).

[5] en.wikipedia.org/wiki/Curse_of_dimensionality

This step turned out to be problematic when dealing with the full automation of the persona creation process, a problem tackled by the APG research team referenced in Section 2.1.3. They mentioned some challenges: firstly, when populating their personas with user quotes, they noticed that inappropriate comments were sometimes being chosen, lowering the overall persona quality when evaluated by marketing experts [26]. Secondly, personas need to be assigned demographically appropriate names in order to be believable. To tackle this, they proposed a tool that takes age, country of origin and gender as input and returns an appropriate name [14]. The same applies to photos: their first approach was to buy a set of stock photos to cover most demographics, but their latest works study the possibility of using AI generated pictures [27]. Lastly, if personas are provided a textual biography, the latter should be coherent with the rest of the data. As research in the field of automatic persona creation is relatively new, they refer to the challenge of fully automating this last step as an open research problem [26, 25].

3 System design

In this chapter, the design process is presented, starting from the requirements and a high-level view of the system architecture. After that, the main components of the system are described in better detail, together with the data modeling and the flow of the entire process.

The system works in the following way: it takes a list of users and their corresponding data as input, and outputs a collection of personas that represent the input users. It is designed to work with social media data: in this sense, a list of users could be the followers of a brand on a given social network, or a list of users who post about a certain topic.

3.1 Requirements

The system must meet the following main requirements:

- personas should be created automatically without the need of any input other than users' data;
- collection from multiple data sources should be supported;
- users of the service should be able to add new user data (or modify the already present one) for their personas, and those personas should be subsequently updated to reflect the new data;
- the system should be fully GDPR compliant.

As for other non-functional requirements, generated personas should accurately represent the users on which they are based. Additionally, the computation time should not be long, although data comes from an online stream of contents. Moreover, the system should be easily scalable, in case new data sources need to be supported.

3.2 Main components

The system consists of five main components, which are shown in Figure 3.1: four logical ones that sit on top of a web API. Each logical component carries out a part of the process required to go from raw data - a list of user identifiers from different social networks or other websites - to the final result - personas. The steps are the following:

1. collect user data from social networks/other websites, such as demographics and activities;
2. enrich the data with new insights derived from the results of the previous step;
3. cluster users into similar groups;
4. generate a persona for each cluster.

The first two steps are coupled together, since data is sent to the enrichment module as soon as it is collected. The clustering step needs the enriched data in order to produce useful results, and the persona generator has to wait for the clusters to be created in order to carry out its job. Details on the design of each component are presented in the following sections.

3.2.1 Data collection

This component has the task to fetch user data from the web. Since the system should support the creation of personas for many different contexts, the more data can be collected, the better and richer the final personas will be.

The main data source is social media data. While more traditional social networks like Twitter, Facebook or Instagram are supported for the reasons introduced in Chapter 2, the system is designed

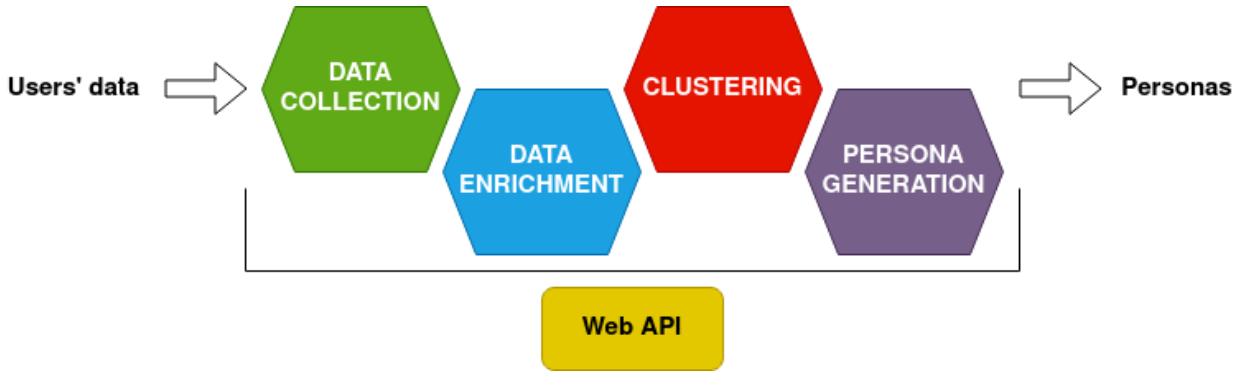

Figure 3.1: High-level view of the main components

to also allow the use of data from less traditional social media (e.g. Strava for sports data, Quora and Medium for user-specific blogging data). To achieve this, a list of all relevant data that can be collected needs to be defined for each data source. Relevant data can include pieces of user information such as name, location, language, number of followers, profile picture and activities (contents the user posted, complete with metadata like the number of likes, comments and shares). It should be noted that different social media usually include different data, which means there could be a lack of data if only one source is used to describe a user. In order to solve this problem, the system allows to associate multiple data sources to a single user.

Another important factor is that user data needs to be periodically collected: in fact, a huge number of activities is posted on social networks every day (around 6000 every second on Twitter[1]). For this reason, the collection component should run periodically (e.g. every 24 hours) in order to keep user data up to date and to retrieve the latest activities.

In general, this component needs as *input* a series of IDs that identify the user among all the data sources associated to them. The expected *output* is, for each data source, the corresponding user information and n user activities. The number n of activities to download could be given as a parameter, but it should not be too large in order to avoid overloading the system. For example, Twitter allows to fetch up to 200 tweets per request.

3.2.2 Data enrichment

This component has the task to derive new insights from the data that was previously collected. These insights are what users will ultimately be clustered on, which means that this phase determines how the final personas will look like. The enrichment component is split in two modules: one enriches user activities, and one enriches user profile information.

Activity enrichment

The goal of this module is to extract, from activity data (such as raw texts and images) and metadata, useful insights, listed in the table below. While some of them are often directly available from the collected data (such as the language or device used), the biggest challenge is to extract what the activity is about. Such information is important because it allows to gain insights on a user's behavior, such as their interests and personality. This task is further complicated when taking different forms of media (text, images) and different languages into account.

Enrichments		
String	language	%Language of the activity (if a text is present)
Float	sentiment	%Value that indicates the tone of the activity
Set[String]	entities	%Strings representing what the user talks about in the activity
String	device	%The device on which the activity was posted

The expected *input* is a user activity, as collected in the previous step. The *output* is the enriched activity.

[1] `https://www.dsayce.com/social-media/tweets-day/`

User profile enrichment

The goal of this module is to make use of whatever data is available from the collected profile information, together with the enriched activities, to extract insights that are relevant for the final personas. For the purpose of designing the system to be as general as possible, we have defined a list of attributes that are present in most persona templates available online: they are shown in the table below. It should be noted that, in many cases, some of the attributes may be impossible to extract: this depends on how much a user is active on a given social media site, and what kind of information they share.

Attributes		
String	gender	%Predicted gender
String	age	%Predicted age range (e.g. 19-29)
String	type	%Whether the user profile is of an individual or brand
String	location	%Where the user lives
String	prefLanguage	%The language most used by the user
String	maritalStatus	%Whether the user is single, married, etc.
Boolean	hasChildren	%Whether the user has children or not
String	job	%The profession of the user
String	personality	%Myers-Briggs personality type, encoded in 4 letters
Set[String]	interests	%List of the top interests of the user
Float	attitude	%Average sentiment found in the user's activities
Dictionary	activityByTime	%How much the user is active during each day
Set[String]	activeChannels	%Social channels where the user is most active

In order to predict attributes such as gender, age or type, machine learning classifiers can be employed. For example, gender can be predicted from a name, or from a profile picture through the use of computer vision algorithms. Life event detectors could predict a user's marital status and/or if they have children by checking if they have posted about marriage or the birth of a child. Personality can be inferred by how the user interacts with other people (as presented in section 2.2), and interests are a result of the entities and topics found in the user's activities.

The expected *input* is the result of the collection of user profile information from a data source and respective enriched activities, while the *output* is the enriched user profile.

3.2.3 Clustering

This component has the task to create clusters of similar users, based on the characteristics that have been extracted in the previous steps. The *input* is a list of users with enriched attributes, and the expected output is composed as follows:

- a mapping of each user to the corresponding cluster index;
- a list of *representative users*, one for each cluster. A representative user is made up by the characteristics that better define a cluster. It should be noted that it can either be a real user or not, depending on the clustering algorithm. For example, in the K-Means algorithm a representative user would be a *centroid*, which is defined as the average of all the users in a given cluster.

The number of clusters that are found could either be specified as a parameter by the user, or could be automatically determined by the system in order to optimize the quality of the clusters.

3.2.4 Persona generation

This component has the task to generate a persona for each cluster defined in the previous step. Since the output of the clustering component includes a list of representative users, this boils down to:

1. assign realistic attributes to the representative users, in case they are not directly mapped to real users (e.g. in case a representative user is defined as the average of all users in a given cluster).

2. give an identity to each representative user: this consists in assigning them a demographically accurate name, photo and textual description.

The *input* is a list of representative users, as output by the clustering component. The *output* is a list of personas, one for each cluster/representative user. Ideally, the user should be able to specify which attributes to include in the personas, in order to tailor them to their specific needs.

3.2.5 Web API

Users of the service interact with this layer to perform their operations. The main resources are the following:

- **tokens**: they are keys used for authorizing access to protected resources;
- **accounts**: in order to use the service, people need to create an account. This allows them to authenticate for future API requests;
- **brands**: an account can create many brands. A brand is essentially a collection of users: an account can create, for example, a brand for Nutella and a brand for Coca Cola, in order to generate personas independently for the two cases;
- **users**: users are the objects subject to collection, enrichment and clustering. Multiple data sources can be associated to one user. Each user is associated to one and only one brand; the reason behind this is that, if users were shared between brands, an account modifying one of their users (e.g. removing a data source) would have side effects on all the other accounts whose brands include that user;
- **clusters** and **personas**.

In order to prevent unauthorized access to important resources like user data and brand's personas, the resources should be protected. The way in which they are protected is explained in Chapter 4.

3.3 Data models

In this section, the data models used by the system are formalized.

3.3.1 Account

The `Account` model is composed as follows; other attributes (e.g. name) could be added if needed.

Account		
String	accountID	%The unique ID of the account inside the system
String	email	%The email of the user, used for authentication
String	hashedPsw	%Hashed password (with salt), used for authentication

3.3.2 Brand

The `Brand` model is composed as follows:

Brand		
String	brandID	%The unique ID of the brand inside the system
String	accountID	%The ID of the account that created the brand
String	name	%The name associated to this brand

3.3.3 User

The `User` model is composed as follows:

User		
String	userID	%The unique ID of the user inside the system
String	brandID	%The ID of the brand the user belongs to
DataSource[]	dataSources	%List of data sources associated to the user
Attributes	attributes	%Enriched attributes of the user

The `Attributes` object is described in Section 3.2.2. User attributes are populated based on the attributes present in the user's data sources (a merging strategy needs to be defined in case of disagreements between different data sources).

3.3.4 Data Source

As data sources can be very heterogeneous, a general `DataSource` object is defined:

DataSource		
String	sourceID	%The unique ID of the data source inside the system
String	sourceName	%The name of the data source (e.g. Twitter)
String	sourceUserID	%ID of the user in this specific data source
String	username	%Username of the user in this specific data source
String	from	%ID of the oldest collected activity of the user
String	to	%ID of the newest collected activity of the user
Attributes	attributes	%Enriched attributes of the user in this specific data source

Then, for each particular data source that is supported, a specific object needs to be defined. For example, a `TwitterDataSource` would contain the general `DataSource` properties, plus additional ones, such as: name, location, profile image, description, website, number of followers and following.

3.3.5 Activity

The same considerations about data sources heterogeneity apply to activities. A general `Activity` model is composed as follows:

Activity		
String	activityID	%The unique ID of the activity inside the system
String	sourceName	%The name of the data source where the activity is from
String	sourceUserID	%ID of the activity in this specific data source
String	authorID	%ID of the user (in this data source) who posted the activity
Enrichments	enrichments	%Enriched activity properties

where the `Enrichments` model is the one described in Section 3.2.2. A more specific `TwitterActivity` would contain additional properties, like: text, language, media, hashtags, external links, number of likes and shares.

3.3.6 Cluster

The `Cluster` model is composed as follows:

Cluster		
String	clusterID	%The unique ID of the cluster inside the system
String	brandID	%The ID of the brand the cluster belongs to
User[]	users	%List of users who belong to the cluster
Attributes	representative	%Attributes of a user who represents the cluster

3.3.7 Persona

The `Persona` model is composed as follows:

Persona		
String	personaID	%The unique ID of the persona inside the system
String	clusterID	%The ID of the cluster the persona represents
String	name	%Name of the persona
String	photo	%Photo of the persona
String	description	%Brief textual description of the persona
Attributes	attributes	%Attributes of the persona

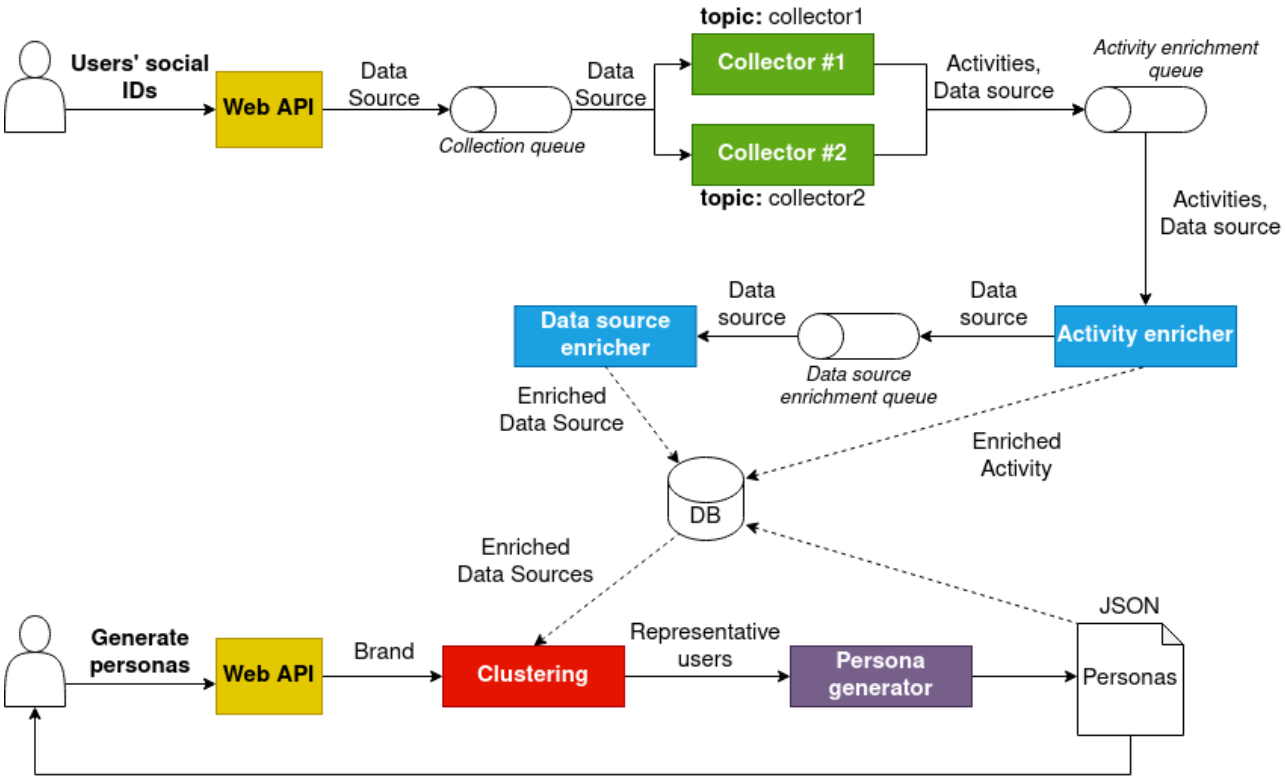

Figure 3.2: System architecture

3.4 System architecture

A diagram of the system architecture is shown in Figure 3.2.

The system can be logically divided in two macro-components: the first contains the collection and enrichment components, the second contains the clustering and persona generation ones. The former is described in Section 3.4.1, due to its complexity .

The latter is less complex, since it mostly works with internal data. Each clustering (and persona generation) request is fulfilled with whatever data is available at the moment. First, clusters are generated; after that, they are passed, together with representative users, to the persona generation module. This macro-component should also be able to run periodically (e.g. every 30 minutes), in order to always keep personas updated when new user data is collected.

3.4.1 Pub/Sub system for Collection and Enrichment

A stream-processing design was chosen for our system. The reasons behind this choice are the following: first of all, social network data naturally falls under the category of *real time data*. As mentioned in Section 3.2.1, a huge number of data is posted on social networks every second, and the system would quickly get overloaded if high quantities of user data is processed in batches. Additionally, the system achieves high separation of concerns: this means that each module does not need to care about the status of other modules, and only needs to carry out its task. This also allows for the implementation of atomic, simple modules and makes the system more easily scalable and expandable, for example, in case support for a new social network needs to be added.

A *pub/sub* (publisher/subscriber) system is one way to achieve the previously mentioned design. It consists in queues that allow communication between each module. A module can publish any kind of message on a queue with a given *topic*, and all the modules that *subscribed* to that topic receive the message. When one module is busy, all the incoming messages are queued and wait until they are taken by the module to be processed.

As an example, let us suppose that a user with two data sources is created through the web API: a Twitter and Facebook account. After saving the user in a database, the user ID on Twitter and Facebook are published on the *collection queue*: the former is published under the topic *twitter*,

the latter under the topic *facebook*. The Twitter and Facebook collector modules are always active, listening on the queue for their respective topics. Once the user IDs are received, the two modules independently download data, save it into the database, and publish the newly collected activities on another queue, the *activity enrichment queue*. Activities are enriched, saved into the database, and once all activities of a given data source have been enriched, the data source is sent to the *data source enrichment queue*. Finally, data sources are enriched and saved into the database.

4 Implementation

In this chapter, the implementation of a prototype of the system is presented. Due to time constraints and the overall complexity of the system, some features that are presented in Chapter 3 are not implemented in the final prototype. They mostly consist of individual classifiers to enrich user data such as marital status or number of children, which ultimately do not impact the larger structure and functionality of the system. In fact, this can be seen as proof that the system design meets the requirement of being highly customizable and expandable.

The whole prototype is developed in Python, one of the most complete programming languages when it comes to the availability of machine learning and other user-made libraries. Every component presented in the design section consists in a separate package. In the sections below, those components are described in more detail one by one: the implementation choices for the database and the pub/sub system; how data is collected only from Twitter due to privacy reasons; how the enrichments are performed and what kind of external tools are used for that purpose; what clustering algorithm is used and why; what tools are used to assign an identity to personas; and how the web API was implemented and what functionalities in particular it provides. Finally, a section is dedicated to showing the typical flow of actions and data through the system, in order to understand how all components interact with each other.

4.1 Database

Given the unstructured nature of the data that the system handles, a non-relational database was chosen. In particular, the choice fell on `Redis`[1], an open source, in-memory data structure store that can be used both as a database, cache and message broker. By working in-memory, it can achieve high performance over traditional databases. Data is stored in key/value pairs, as in a typical dictionary data structure: keys are always strings, while values can be a range of supported data types. For our use-case, since complex, nested data models need to be stored, we installed `RedisJSON`[2], a module that adds support for a JSON data type. This means that all models are converted to and from JSON every time they need to be stored or retrieved from the database. The database runs in a dedicated Docker container.

4.2 Pub/Sub

The Pub/Sub queue system is implemented with the MQTT protocol. The implementation provided in the library `paho-mqtt`[3] is used. All the messages are sent and received with a *Quality of Service* (QoS) level of 1, which guarantees that every message is delivered at least once. This allows for the modules that disconnect for any reason to recover all not-received messages once they go back online.

4.3 Data collection

The data source on which the prototype is based is Twitter. Facebook products (such as Facebook itself and Instagram) were considered, but developing an application that works with them has become difficult due to strict privacy policies. In fact, in order to collect data of a user through their API, it is first needed to get explicit authorization of the user. Finding enough users available to share their data for the development of this system would have been a lengthy process. Also other services were taken into consideration, but were ultimately scrapped due to a lack of an API or the unavailability of a free version of the latter.

[1] `www.redis.io`

[2] `www.redislabs.com/blog/redis-as-a-json-store`

[3] `www.eclipse.org/paho/`

In order to use the Twitter API, it is first needed to submit a request explaining what you are going to use the API and the collected data for. After confirmation, the API provides a free plan, although with some restrictions on the rate at which requests are made (e.g. 1500 requests every 15 minutes to retrieve tweets). The free plan allows us to get all the information that is needed: user profile information and user tweets. The API is accessed using the `tweepy`[4] library, which provides a convenience class that handles requests and parameters. It also handles rate limitations by suspending the process until a new time slot is available.

4.4 Data enrichment

This component is divided in two smaller modules that are run independently: one to enrich activities, and one to enrich data sources. They are presented in the sections below.

4.4.1 Activity enrichment

The main task of this module is to extract *language*, *sentiment* and *entities* from an activity, which can contain text and/or images.

An *entity* is a person, an object or a concept that has an article on Wikipedia; for example, the phrase *"I'm studying computer science at the university"* has two entities - `Computer science` and `University`. Entity extraction is a powerful approach to figure out what a user talks about in an activity: it supports both texts and images, and it is expandable to many languages (it is only needed, for example, to convert entities, which are Wikipedia articles, to their English article counterpart).

While there exist tools that allow entity extraction from images, such as the `Cloud Vision API` of Google Cloud[5], they were not used in the prototype due to their heavily restricted free plans. This limits the prototype to only work with texts, but, in a production environment, support for images could greatly improve the accuracy of the final results, since many social media posts don't include any text.

For all three enrichments, we used an external semantic analyzer called `Dandelion`[6], developed by Spazio Dati[7]. It works by using Wikipedia pages as entities, as explained in Section 3.2.2. It provides useful options to optimize entity extraction from short texts coming from social networks, which makes it adequate for our use-case. Additionally, it supports a total of 50 languages, 43 of which are in beta support. The free plan allows the use of 1000 credits per day, and 2.1 credits are used for enriching each activity. This means that only 476 activities can be enriched per day for free. U-Hopper provided a token with 10000 credits, which allows to enrich ten times that amount.

4.4.2 Data source enrichment

This module has the task of enriching the `Attributes` properties detailed in Section 3.2.2.

Gender, **age** and **type** are predicted based on the profile image of a user on a given data source. A customized version of the `M3-Inference`[8] library is used for this purpose. Since predicting an exact age is too difficult, four age brackets are given as possible output: $\leq$18, 19-29, 30-39, $\geq$40. In case a profile image is not available, an external API called `NamSor`[9] is used to predict gender and type from a first or full name. This is not always reliable, since the name field is free form on Twitter, which means that a user can choose a fictional name (as opposed to Facebook, which enforces the use of real names).

The **preferred language** is chosen by keeping a map between all the languages the user has used in their activities, and how many times each language is used. The preferred language is therefore the most used language throughout all activities.

Attitude is simply computed as the average of all sentiment scores found in the user's activities, while **interests** are more complex to extract. First, the system keeps track of an *entity map* for each data source; it consists in a dictionary where the keys are the entities found in the user's activities,

[4] `www.tweepy.org`
[5] `cloud.google.com/vision`
[6] `dandelion.eu`
[7] `spaziodati.eu/it`
[8] `github.com/euagendas/m3inference`
[9] `www.namsor.com`

and the values are how many times that entity was found. This is then given as input to a Tapoi model, proprietary of U-Hopper, which functions as follows:

1. the model is defined by a fixed set of categories of interest, which represent the full set of interests that are expected as output. Those categories are represented by Wikipedia pages, and in our case are the following: `Arts`, `Filmmaking`, `Cooking`, `Culture`, `Economy`, `Entertainment`, `Fashion`, `Geography`, `Health`, `History`, `Literature`, `Music`, `Nature`, `Philosophy`, `Politics`, `Religion`, `Science`, `Sports`, `Technology`;

2. the model traverses the Wikipedia graph upwards, starting from each entity found in the entity map, for a fixed number of steps (usually 5);

3. each time a category of interest is found while traversing the graph, a score is added to that category, proportional to the number of times the respective entity is found in the entity map;

4. the output is a mapping between each category of interest and their respective scores, which are in the $[0, 1]$ range. All scores sum up to 1, and a high score means that the user is interested in the corresponding category.

Two Tapoi models were made available by U-Hopper for this thesis: one that works on English Wikipedia, and one on Italian Wikipedia. While it is possible to use both models for analyzing English and Italian activities, this would mean supporting only two languages. The implemented solution, instead, only uses the English model: if an entity is in another language, it first gets converted to the respective English page on Wikipedia (when available) through the MediaWiki API[10]. The English model was chosen over the Italian one due to English Wikipedia having the largest number of articles than any other language.

As for the rest of the `Attributes`, they are not implemented due to time constraints, but other Tapoi models exist to predict many of them.

4.5 Clustering

The most important implementation choices for this component are: the features on which to perform clustering; the choice of a distance metric, needed to compute how similar (or dissimilar) two users are; and the actual clustering algorithm. The order in which those implementation choices are listed is not casual: the choice of algorithm depends on the distance metric, and the distance metric depends on what types of features are used.

4.5.1 Features

The chosen features are the `Attributes` properties that are implemented, minus the type; that is: gender, age, preferred language, attitude and interests. The type is excluded because we decided to deal only with users who are classified as "humans", since personas are ultimately fictional individuals. The features are a mix of *nominal*, *ordinal* and *numerical* data:

- nominal data is data that takes on discrete values and does not possess a meaningful order. In our case, gender and preferred language fall under this category;

- ordinal data is data that takes on discrete values but possesses a meaningful order. Age is an example of this: it can take on three different values ($\leq$18, 19-29, 30-39, $\geq$40), and it makes sense to say that the bracket $\leq$18 is closer to 19-29 than it is to 30-39;

- numerical data is data that takes on continuous values. Attitude is an example, as it is a decimal number in the range $[0, 1]$.

Interests do not fall directly into any category, but are closer to a mix of nominal data (interest labels) and numerical data (scores).

[10] `www.mediawiki.org/wiki/API:Main_page`

4.5.2 Distance metric

Since we are dealing with mixed data, the most commonly used distance metric, euclidean distance, cannot be used, as it only works with numerical data. This also locks out the choice of K-Means as the clustering algorithm, since it relies on the concept of mean, which is only meaningful in a continuous space. The chosen distance metric is therefore the *Gower distance*, which supports mixed data [30]. It is defined as follows:

$$D_{Gower}(x_1, x_2) = \frac{1}{p} \sum_{j=1}^{p} d_j(x_{1j}, x_{2j})$$

In words, it is computed as the average of partial dissimilarities across objects with p features. A partial dissimilarity d_j is computed between two objects x_1 and x_2 on a particular feature j, and depends on the type of the feature (ordinal, numerical or nominal):

$$d_{j,ord}(x_1, x_2) = \frac{|rank(x_{1j}) - rank(x_{2j})|}{range_j} \qquad d_{j,num}(x_1, x_2) = \frac{|x_{1j} - x_{2j}|}{range_j}$$

$$d_{j,nom}(x_1, x_2) = \begin{cases} 1 & \text{if } x_{1j} \neq x_{2j} \\ 0 & \text{otherwise} \end{cases}$$

where $rank(x_j)$ is the ordinal number assigned to feature x_j, and $range(j)$ is defined as the difference between the maximum and minimum values of feature j.

A modified version of the Gower distance is used in our system: instead of simply averaging all partial dissimilarities, a weighted average is computed. This allows to give more weight to features that can be considered more important, like the interests in our case.

4.5.3 Clustering algorithm

Since we have defined a custom distance metric, only algorithms that support either a distance function as a parameter or a precomputed distance matrix can be chosen. Given n users, we define a distance matrix $M[n \times n]$, where each cell $M[i, j]$ contains the distance between user i and user j.

The chosen algorithm is called *K-Medoids*[11], an alternative version of K-Means that accepts a precomputed distance matrix as an input parameter. Instead of computing a centroid as the mean of all objects in a cluster, the algorithm chooses a medoid; it can be defined as the object of a cluster whose average dissimilarity to all the other objects of the cluster is minimal, that is, the most centrally located point in the cluster. Using K-Medoids also has the advantage that medoids, as opposed to centroids, are real data points (i.e. users), which means that they can be directly given as input to the persona generation component.

The main challenge remains deciding the right number k of clusters, which needs to be specified a priori. The chosen way is trying models with different values of k, ranging from 2 to 10, since we do not want a large number of clusters. For each model, we compute the average *silhuette score*[12]: it is a number between -1 and 1 which measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The optimal number of clusters is therefore the one that maximizes the average silhuette score.

4.6 Persona generation

This component is simpler than the others, especially since the clustering algorithm outputs representative users with meaningful attributes. Given a representative user, it is needed to enrich it with a name, photo and textual description.

The **name** is generated through an external API called `Name Parser`[13]. It allows to optionally specify a gender and a country, in order to get demographically accurate names.

The **photo** is taken from a predefined set of photos; for the purpose of the prototype, the set is composed of five photos for each gender and age bracket, for a total of 40 photos. The photos were

[11] `en.wikipedia.org/wiki/K-medoids`

[12] `en.wikipedia.org/wiki/Silhouette_(clustering)`

[13] `parser.name`

taken manually through a website[14] that uses AI to generate pictures of people that do not exist.

The **description** was not implemented due to time constraints, but a valid approach would be to write one or more templates that can be appropriately filled with the persona attributes.

4.7 Web API

The web API is implemented using the `Flask`[15] framework, extended with the `Flask-RESTful` library. The chosen method for managing authentication and authorization is a JWT Token, a particular implementation of token/bearer authentication. Each token contains the ID number of the account that generated the token, and it is checked in order to grant or deny access to a requested resource. It also contains an expiration date, after which the token is no longer valid. In the prototype, like is often seen in web APIs, the expiration date is not set; this means that it is up to the user of the service to keep the token secret. In any case, it is possible to regenerate a token, a process which invalidates the previous one.

Figure 4.1 shows the main functionalities provided by the web API (the lock icons show what resources need authorization). The complete specification and documentation is available at the link below:

`app.swaggerhub.com/apis-docs/nicola-farina/personas/1.0.0`

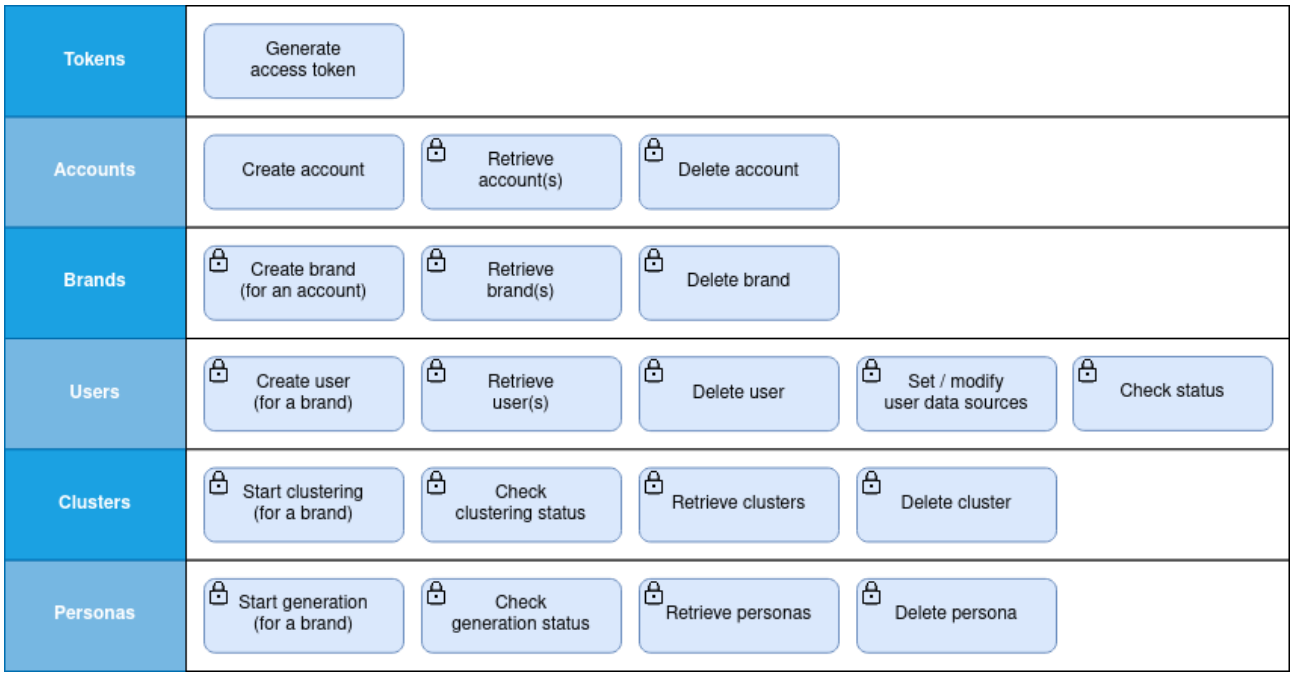

Figure 4.1: Functionalities provided by the API

4.8 Flow of the system

Figure 4.2 shows a typical flow of the data collection and enrichment components in detail, while Figure 4.3 shows the flow of the whole system.

The implementation of the collection and enrichment components takes the advantages of real time data processing while still allowing to keep track of progress. In order to do so, a `Computation` object is created whenever a process of data collection and enrichment begins: it is simply composed by a unique ID and a *status* field. Those objects are stored in the database, and every 24 hours all computation objects whose value equals *done* are deleted, since they are no longer needed. Another advantage of keeping track of the progress is that data sources can be sent for enrichment only when

[14] `thispersondoesnotexist.com`
[15] `flask.palletsprojects.com/en/2.0.x/`

all related activities are enriched. This is preferable because, if there was no way to keep track of how many activities have been enriched, data sources would have to be enriched after every single activity. This would result in a lot of external API calls, which may have severe rate limitations.

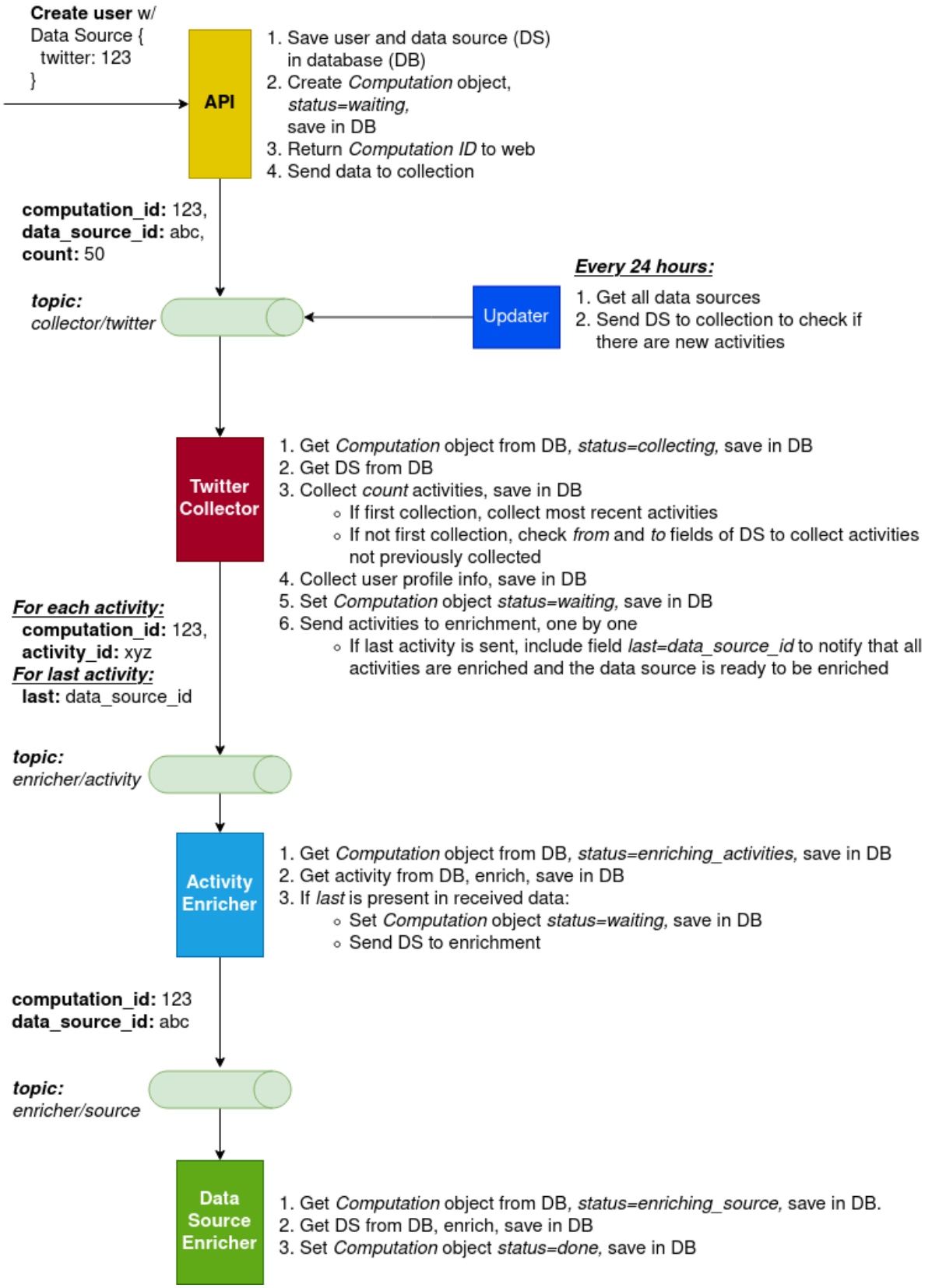

Figure 4.2: Detailed flow of collection and enrichment components

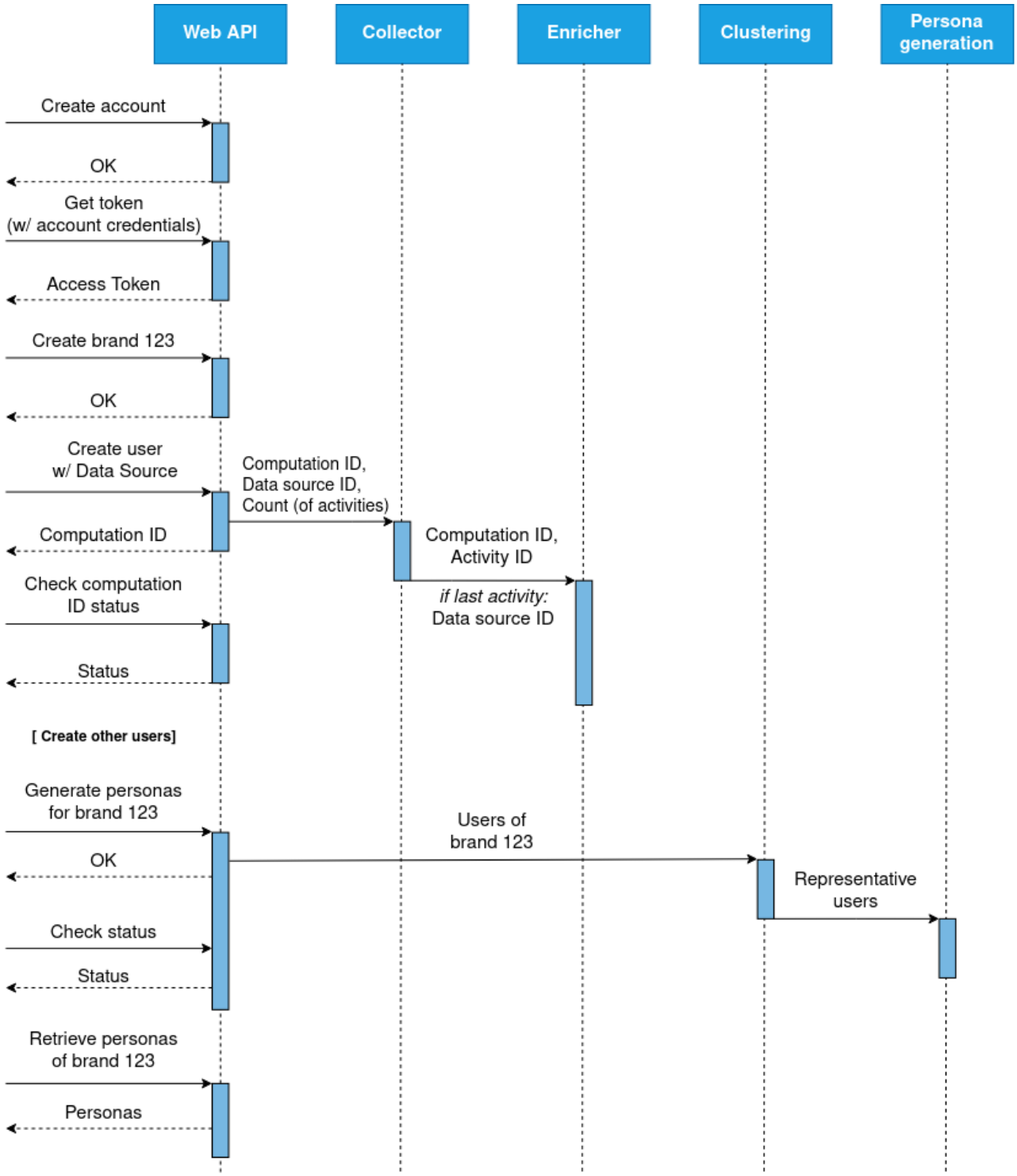

Figure 4.3: Typical flow of actions and data in the system

5 Evaluation

In this chapter, the evaluation of the implemented prototype is presented. First, the different evaluation metrics and experiments are discussed. Then, the test dataset is described. Finally, the experiments and their results are analyzed.

5.1 Evaluation metrics

Two main factors are taken into consideration for evaluating the prototype: the quality of the clusters, which directly translates into the quality of the personas; and the performance.

5.1.1 Quality of clusters

With quality of a cluster we mean how coherent a cluster, both within itself (*are the users of a cluster similar?*) and with all the users that were clustered (*if there are football players among them, a cluster of sports enthusiast should be expected*). This depends on the number of clusters chosen a priori, on the distance metric and on the minimum number of activities that are collected for each user. The number of activities is relevant because, if it is too low, the chance of users' interests being misclassified in the enrichment part gets higher, since the impact of outlying activities is stronger.

Evaluating the quality of a cluster is not an easy task, since clustering falls under the category of *unsupervised learning*. This means that, usually, there is no ground truth that allows to check whether the clustering result is good or not. In fact, oftentimes the results are analyzed manually. This approach is not feasible for our fully deployed system, since the requirement of full automation would not be met. For this reason, and for the purpose of evaluation, we manually labeled our testing dataset with ground truths. This allows us to use four classic machine learning metrics, which range from 0 to 1, with 1 being the best score:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad Precision = \frac{TP}{TP + FP} \quad Recall = \frac{TP}{TP + FN}$$

where TP means *true positives*, TN *true negatives*, FP *false positives* and FN *false negatives*.

5.1.2 Performance

For performance, relevant metrics are the average time needed to enrich a single source or a single activity, as well as the total time needed for the entire process of generating personas from a collection of users.

5.2 Test dataset

The test dataset was built manually in order to have accurate ground truths. For this reason, and due to time constraints (further enforced by the APIs rate limitations), it is only composed of 90 users.

Those 90 users are all public figures who have a profile on Twitter, and were chosen in the following way: 30 football players, 30 politicians and 30 musicians/singers. These categories can be mapped to the interests classified by our system: respectively, *sports*, *politics* and *music*. Of course, not every tweet of a politician is about politic. This was taken into account during the creation of the dataset, and only users who mainly tweeted about their respective categories were chosen.

As mentioned previously, each user is assigned a ground truth: their gender, their age bracket and their main field of interest (sports, politics, music).

5.3 Experiments setup

We have identified a set of tests, in order to evaluate the key features of the prototype. They are shown in Table 5.1.

In `Test1` and `Test2`, the time it takes to enrich activities and data sources is computed. `Test3` is the most important test, in which the quality of the clusters is computed and the effects of parameters like the distance metric, the number of clusters and the number of activities per user are analyzed.

	Operation	**# of activities per user**
Test1	Activities enrichment	[20, 50, 100]
Test2	Data sources enrichment	[20, 50, 100]
Test3	Clustering	[20, 50, 100]

Table 5.1: Experiments

Other tests, for example, to evaluate the quality of the individual classifiers used in the enrichment process, are left out due to time constraints.

5.4 Evaluation results

5.4.1 Activities enrichment

The results are shown in Table 5.2. As expected, the time to enrich a single activity is more or less constant. It is computed as the time difference between the reception of the activity on the enrichment queue and the moment the enriched activity is saved in the database. This average is, as expected, not influenced by the number of activities collected per user, since each enrichment is independent from the others. It is only influenced by the external API call and the time it takes to save the enrichments in the database.

The total time to enrich a given number of activities for each of the 90 users is not simply the number of total activities multiplied by the time it takes to enrich a single activity. This is caused by the quality of service setting of the pub/sub queue: by having it set to one, more time is required since the messages needs to be acknowledged before they can be processed. We have not tried setting it to zero, since some queue clients seem to disconnect at times, and that would mean losing all the messages that are sent while a client is offline. It is worth noting how, in order to collect 100 activities, the system needs to wait an entire day; this is because of the activity enrichment API, which allows at most 4500 activities to be fully enriched per day.

Operation	**Elapsed time**
Single activity	0.52 seconds (average)
20 activities per 90 users	16 minutes
50 activities per 90 users	36 minutes
100 activities per 90 users	75 minutes + 24h wait

Table 5.2: Performance results for activity enrichment

5.4.2 Data sources enrichment

The time to enrich a single data source sits at an average of 13,5 seconds, and is dominated by the interests classifier. As expected, the total time to enrich a data source is practically not affected by the number of activities per user, since the only thing that changes is the number of activities that need to be retrieved from the database. The total time to enrich 90 users is around 20 minutes.

5.4.3 Clustering

Distance metric

The distance metric used for clustering is specified in Section 4.5.2. The only parameters that needed tuning are the weights assigned to each feature. The weights were tuned in order to maximize the quality of the clusters that are presented in the following sections. Furthermore, a lot of importance was given mostly to interests, followed by gender, based on the fact that interests are the most relevant

behavioral attributes available in our case for defining personas. The weights are the following:

$$Gender = 0.5 \quad Age = 0.5 \quad Language = 0.3 \quad Interests = 13 \quad Attitude = 0.3$$

While the weight for interests may seem high, based on tuning and experiments we conducted, it only begins to suppress the other weights around the value 100. If lowered, the gender feature tends to dominate, resulting most of the times in only two clusters, one for males and one for females.

Number of clusters and activities per user

Figure 5.1 shows the silhuette score for various number of clusters and for the three different values of activities per user. It is clear that going from 20 to 50 activities per user helps narrow down the main interests of a user, thus reducing the optimal number of clusters from ten to four. Going from 50 to 100 activities does not change much, so we chose 50 as the optimal number of activities per user, which also avoids having to wait one day due to API rate limitations.

Why this happens is that, if only a small number of activities are analyzed, the risk of misclassifying a user's interests is high. For example, during the period when our dataset was created, many Italian users tweeted about the UEFA European Football Championship, even politicians and singers. One way to avoid classifying users based on outlying activities is to simply increase the number of activities per user, thus decreasing the impact of outliers. A more sophisticated approach would consist in keeping track of what the users talk about over time, for example, month by month. This way, interests that are present only for a limited amount of time could be classified as outliers, and left out of the overall interests of the user.

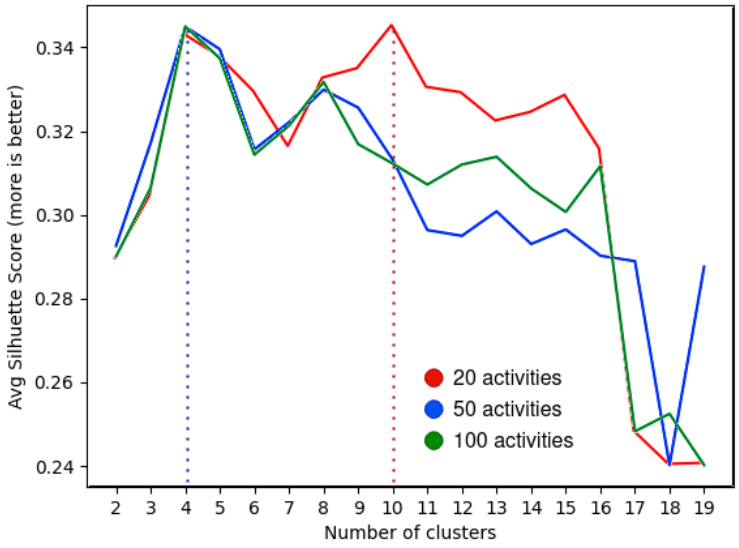

Figure 5.1: Silhuette score for each number of clusters, for different values of activities per user

Composition of each cluster

At this point, we analyzed how each cluster is composed. Table 5.3 shows the representative users, and we assigned a name to every cluster based on them. Table 5.4 shows data that can be used for computing precision, recall and accuracy.

At first glance, the results seem to be coherent with the ground truth. Two things should be noted: the first is that only musicians were split in two clusters based on gender (while the other categories are left with mixed genders); the second is that all of the representative users have *culture* as their second main interest. This last observation probably happens due to the very general scope of a category such as culture. This issue could be solved by using more specialized interests classifiers,

Cluster	Gender	Age	Pref. lang.	Top 2 interests	Attitude
1, Politicians	male	≥ 40	it	Politics (0.35), Culture (0.16)	0.02
2, Musicians (F)	female	19-29	en	Music (0.26), Culture (0.10)	0.36
3, Musicians (M)	male	≥ 40	en	Music (0.49), Culture (0.16)	0.13
4, Footballers	male	30-39	it	Sports (0.62), Culture (0.08)	0.12

Table 5.3: Representative users of each cluster

Cluster	# of users	# of expected users	TP	FP	FN
1, Politicians	31	30	29	2	1
2, Musicians (F)	17	15	15	2	0
3, Musicians (M)	13	15	13	0	2
4, Footballers	29	30	29	0	1

Table 5.4: Summary of the contents of each cluster

which could be chosen as a parameter by the user in order to obtain more accurate results for a specific sector (e.g. there could be one classifier specialized for the music industry, one for sports, one for food etc.). Another approach to solve it would be to ignore a specific interest if it shows up in the majority of the clusters.

With this data, we can compute the metrics introduced earlier in this chapter to assess the quality of the clusters: they are shown in Table 5.5.

Cluster	Accuracy	Precision	Recall
1, Politicians	0.96	0.93	0.96
2, Musicians (F)	0.97	0.88	1.0
3, Musicians (M)	0.97	1.0	0.86
4, Footballers	0.98	1.0	0.96
Global (average)	0.97	0.95	0.94

Table 5.5: Results of clustering

While the results are very positive, it should be noted that there is the possibility of *overfitting*: that means that the system performs well on this specific test dataset, but does not generalize enough to perform well on other datasets. The only way to find out if this is the case is to try to evaluate the system with other datasets, for example by adding other categories of users. This is left out of this thesis due to time constraints.

5.4.4 Final considerations

The results of the clustering evaluation directly reflect the quality of the personas. In fact, almost all users in our dataset fall under a cluster that represents them, and the clusters themselves are meaningful and can be given an identity.

As for evaluating the personas themselves, this is still an open issue that is currently being researched. This is the case because there currently are no ways to algorithmically (thus, automatically) evaluate the quality of a persona profile; the current, most employed solution simply consists in handing out the personas to a marketing specialist, who is able to judge whether they could be useful for defining marketing strategies and choices. Again, this part is not covered in this thesis due to time constraints.

6 Conclusions

This thesis tries to answer the research question about the feasibility of automatic persona generation from social media data, proposing a solution that is scalable and expandable to multiple social media, and presenting a prototype that works with Twitter data.

It develops on what is the current state of the art, especially because it is one of the only works that discusses in depth about all the phases of the process, and does not focus only on clustering. The enrichment component makes use of innovative techniques, such as using Wikipedia contents to understand what a user talks about, which allows to support a large number of languages and to use all components of an activity (text, images, but also external links etc.) for classification purposes. The proposed system architecture and data models are for general purpose personas, but they are designed to allow the creation of sector-specific personas by introducing new classifiers. Furthermore, the stream processing design allows to avoid bottlenecks as much as possible and achieves a high separation of concerns.

The system shows great results, with clustering accuracy, precision and recall measures well above 0.9, as well as high accuracy enrichments.

While many features were not implemented in the final prototype, either due to time constraints or privacy reasons, this work offers a solid base for automatically generating marketing personas from social media data, which can be deepened in some aspects based on the specific scope in which it will be used.

6.1 Future work

There are several aspects that can be deepened or added. First of all, the system was designed to work with several social networks and similar sites in mind, but due to the reasons explained in Section 4.3 only Twitter was taken into consideration for the prototype. The first logical addition would be to add support for other social networks: first of all, Facebook would be a good choice, since it is a great source of personal information and demographics that are hard to come by on Twitter.

Another important improvement would be adding enrichment modules to predict all the attributes that were not considered in the prototype due to time constraints. Information such as location, marital status and personality would add great value to the final personas. The same applies for personas themselves: implementing a textual description for each of them would add a lot of value to their use.

More time could be spent on the evaluation of the clusters and on fine-tuning parameters such as the distance metric weights. Testing the system on a dataset with a lot more than 90 users would provide more accurate results, and possibly confirm or rule out the possibility of overfitting.

Making use of sector-specific classifiers would also greatly improve the quality of personas for non-general use cases.

Finally, providing a web application with a graphical interface to use the service would be an important addition to provide a better user experience and simplify the interactions. It would also allow to visualize personas with nice graphics, which is how they are usually presented to marketing specialists.

Bibliography

[1] Sibel Adali and Jennifer Golbeck. Predicting personality with social behavior. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 302–309. IEEE, 2012.

[2] Jisun An, Hoyoun Cho, Haewoon Kwak, Mohammed Ziyaad Hassen, and Bernard J Jansen. Towards automatic persona generation using social media. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pages 206–211. IEEE, 2016.

[3] Jisun An, Haewoon Kwak, Soon-gyo Jung, Joni Salminen, and Bernard J Jansen. Customer segmentation using online platforms: isolating behavioral and demographic segments for persona creation via aggregated user data. *Social Network Analysis and Mining*, 8(1):1–19, 2018.

[4] Jisun An, Haewoon Kwak, Soongyo Jung, Joni Salminen, M Admad, and B Jansen. Imaginary people representing real numbers: Generating personas from online social media data. *ACM Transactions on the Web (TWEB)*, 12(4):1–26, 2018.

[5] David Bamman, Brendan O'Connor, and Noah A Smith. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 352–361, 2013.

[6] Nina Cesare, Christan Grant, Quynh Nguyen, Hedwig Lee, and Elaine O Nsoesie. How well can machine learning predict demographics of social media users? *arXiv preprint arXiv:1702.01807*, 2017.

[7] Christopher N Chapman and Russell P Milham. The personas' new clothes: methodological and practical arguments against a popular method. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 50, pages 634–636. SAGE Publications Sage CA: Los Angeles, CA, 2006.

[8] Alan Cooper et al. *The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity*, volume 2. Sams Indianapolis, 2004.

[9] Thiago Freitas dos Santos, Danilo Gouveia de Castro, Andrey Araujo Masiero, and Plinio Thomaz Aquino Junior. Behavioral persona for human-robot interaction: a study based on pet robot. In *International Conference on Human-Computer Interaction*, pages 687–696. Springer, 2014.

[10] Jennifer Golbeck, Cristina Robles, Michon Edmondson, and Karen Turner. Predicting personality from twitter. In *2011 IEEE third international conference on privacy, security, risk and trust and 2011 IEEE third international conference on social computing*, pages 149–156. IEEE, 2011.

[11] Shu Huang, Wei Peng, Jingxuan Li, and Dongwon Lee. Sentiment and topic analysis on social media: a multi-task multi-label classification approach. In *Proceedings of the 5th annual ACM web science conference*, pages 172–181, 2013.

[12] Jina Huh, Bum Chul Kwon, Sung-Hee Kim, Sukwon Lee, Jaegul Choo, Jihoon Kim, Min-Je Choi, and Ji Soo Yi. Personas in online health communities. *Journal of biomedical informatics*, 63:212–225, 2016.

[13] Aaron Humphrey. User personas and social media profile. *Persona Studies*, 3(2):13–20, 2017.

[14] Soon-Gyo Jung, Joni Salminen, and Bernard J Jansen. All about the name: Assigning demographically appropriate names to data-driven entities. In *Proceedings of the 54th Hawaii International Conference on System Sciences*, page 4034, 2021.

[15] Andrey Araujo Masiero, Ricardo de Carvalho Destro, Otavio Alberto Curioni, and Plinio Thomaz Aquino Junior. Automa-persona: A process to extract knowledge automatic for improving personas. In *International Conference on Human-Computer Interaction*, pages 61–64. Springer, 2013.

[16] Dastan Hussen Maulud, Subhi RM Zeebaree, Karwan Jacksi, Mohammed A Mohammed Sadeeq, and Karzan Hussein Sharif. State of art for semantic analysis of natural language processing. *Qubahan Academic Journal*, 1(2):21–28, 2021.

[17] Jennifer McGinn and Nalini Kotamraju. Data-driven persona development. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1521–1524, 2008.

[18] Steve Mulder and Z Yaar. Approaches to creating personas. *The user is always right: A practical guide to creating and using personas for the web. Berkeley, CA: New Riders*, pages 33–54, 2007.

[19] Federico Neri, Carlo Aliprandi, Federico Capeci, Montserrat Cuadros, and Tomas By. Sentiment analysis on social media. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 919–926. IEEE, 2012.

[20] BBC News. Twitter 'shuts down millions of fake accounts'. `https://www.bbc.com/news/technology-44682354`, July 2018. Accessed: 14 May 2021.

[21] George Olsen. Persona creation and usage toolkit. *Retrieved March*, 25:2014, 2004.

[22] John Pruitt and Tamara Adlin. *The persona lifecycle: keeping people in mind throughout product design*. Elsevier, 2010.

[23] John Pruitt and Jonathan Grudin. Personas: practice and theory. In *Proceedings of the 2003 conference on Designing for user experiences*, pages 1–15, 2003.

[24] Pau Rodríguez, Diego Velazquez, Guillem Cucurull, Josep M Gonfaus, F Xavier Roca, Seiichi Ozawa, and Jordi Gonzàlez. Personality trait analysis in social networks based on weakly supervised learning of shared images. *Applied Sciences*, 10(22):8170, 2020.

[25] Joni Salminen, Kathleen Guan, Soon-Gyo Jung, and Bernard J Jansen. A survey of 15 years of data-driven persona development. *International Journal of Human–Computer Interaction*, pages 1–24, 2021.

[26] Joni Salminen, Soon-gyo Jung, and Bernard J Jansen. The future of data-driven personas: A marriage of online analytics numbers and human attributes. In *ICEIS (1)*, pages 608–615, 2019.

[27] Joni Salminen, Soon-gyo Jung, Ahmed Mohamed Sayed Kamel, João M Santos, and Bernard J Jansen. Using artificially generated pictures in customer-facing systems: an evaluation study with data-driven personas. *Behaviour & Information Technology*, pages 1–17, 2020.

[28] Rashmi Sinha. Persona development for information-rich domains. In *CHI'03 extended abstracts on Human factors in computing systems*, pages 830–831, 2003.

[29] Nan Tu, Qiuyun He, Tian Zhang, Haofeng Zhang, Yahui Li, Han Xu, and Yang Xiang. Combine qualitative and quantitative methods to create persona. In *2010 3rd International Conference on Information Management, Innovation Management and Industrial Engineering*, volume 3, pages 597–603. IEEE, 2010.

[30] Gulanbaier Tuerhong and Seoung Bum Kim. Gower distance-based multivariate control charts for a mixture of continuous and categorical variables. *Expert systems with applications*, 41(4):1701–1707, 2014.

[31] Anders Tychsen and Alessandro Canossa. Defining personas in games using metrics. In *Proceedings of the 2008 conference on future play: Research, play, share*, pages 73–80, 2008.

[32] Sandra Vosbergen, JMR Mulder-Wiggers, JP Lacroix, HMC Kemps, Roderik A Kraaijenhagen, Monique WM Jaspers, and Niels Peek. Using personas to tailor educational messages to the preferences of coronary heart disease patients. *Journal of biomedical informatics*, 53:100–112, 2015.

[33] Xiang Zhang, Hans-Frederick Brown, and Anil Shankar. Data-driven personas: Constructing archetypal users with clickstreams and user telemetry. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 5350–5359, 2016.