



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 870811



Social cohesion, Participation, and Inclusion
through Cultural Engagement

D3.7: Final Clustering Techniques

Deliverable information	
WP	WP3
Document dissemination level	PU Public
Deliverable type	R Document, report
Lead beneficiary	UCM
Contributors	UCM
Date	26/04/2023
Document status	Final
Document version	V1.0

Disclaimer: The communication reflects only the author's view and the Research Executive Agency is not responsible for any use that may be made of the information it contains

INTENTIONALLY BLANK PAGE

Project information

Project start date: 1st of May 2020

Project Duration: 36 months

Project website: <https://spice-h2020.eu>

Project contacts

Project Coordinator

Silvio Peroni

ALMA MATER STUDIORUM -
UNIVERSITÀ DI BOLOGNA

Department of Classical
Philology and Italian Studies –
FICLIT

E-mail: silvio.peroni@unibo.it

Project Scientific coordinator

Aldo Gangemi

Institute for Cognitive Sciences
and Technologies of the Italian
National Research Council

E-mail: aldo.gangemi@cnr.it

Project Manager

Adriana Dascultu

ALMA MATER STUDIORUM -
UNIVERSITÀ DI BOLOGNA

Executive Support Services

E-mail:
adriana.dascultu@unibo.it

SPICE consortium

No.	Short name	Institution name	Country
1	UNIBO	ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA	Italy
2	AALTO	AALTO KORKEAKOULUSAATIO SR	Finland
3	DMH	DESIGNMUSEON SAATIO - STIFTELSEN FOR DESIGNMUSEET SR	Finland
4	AAU	AALBORG UNIVERSITET	Denmark
5	OU	THE OPEN UNIVERSITY	United Kingdom
6	IMMA	IRISH MUSEUM OF MODERN ART COMPANY	Ireland
7	GVAM	GVAM GUIAS INTERACTIVAS SL	Spain
8	PG	PADAONE GAMES SL	Spain
9	UCM	UNIVERSIDAD COMPLUTENSE DE MADRID	Spain
10	UNITO	UNIVERSITA DEGLI STUDI DI TORINO	Italy
11	FTM	FONDAZIONE TORINO MUSEI	Italy
12	CELI	MAIZE SRL	Italy
13	UH	UNIVERSITY OF HAIFA	Israel
14	CNR	CONSIGLIO NAZIONALE DELLE RICERCHE	Italy

Executive summary

SPICE community model provides the technological infrastructure that enables reasoning about citizens from their characteristics, opinions, and preferences. By defining different perspectives, the community model identifies groups of citizens that are related to certain similarity criteria but sparsely connected to other users. Community creation is based on clustering user-generated content for creating implicit communities.

In this report, we described the deployed software for cluster analysis of community content, the VISIR Tool for defining and visualizing perspectives, and examples of community detection, visualization, and reflection processes with the different application scenarios, namely GAM (Torino), MNCN (Madrid), DMH (Helsinki), and HECHT (Haifa).

Document History

Version	Release date	Summary of changes	Author(s) -Institution
V0.1	02/04/2023	A first internal draft released	UCM
V0.8	17/04/2023	Internal review of WP3 members	UCM
V0.9	18/04/2023	Internal review other WP members	Manuel Striani, Gautam Vishwanath
V1.0	26/04/2023	Final Revision	UCM
...			

Project information.....	3
Project contacts	3
SPICE consortium	3
Executive summary.....	4
Document History.....	5
Introduction	8
Development and deploy.....	8
Workflow	9
Metadata configuration.....	10
Creating perspectives	10
Perspective maintenance	12
Interaction with other SPICE systems	12
Community Model architecture	13
Community Model API.....	13
CM Core	16
DAO.....	16
CM Core API.....	16
Community detection	17
Clustering algorithms.....	17
Similarity Measures	19
Equal similarity.....	19
Similarity between Plutchik emotions	19
Similarity between taxonomies	20
Expert-based similarity	22
Range similarity.....	24
Year similarity	24
Color similarity.....	24
Similarity between lists.....	24
Community explanations.....	24
Community Similarity	28
Application Scenarios.....	29
Case Study Analysis using VISIR	29
GAM (Torino)	32
Perspectives	32
Visualization and insights.....	37
MNCN (Madrid)	41
Perspectives	41
Visualization and insights.....	42

DMH (Helsinki)	44
Perspectives	44
Visualization and insights.....	46
HECHT (Haifa).....	47
Perspectives	48
Visualization and insights.....	49
Conclusions	51
References	52
Appendix 1: Community Model Deployment	54
Appendix 2: Data samples	55
GAM (Torino)	55
MNCN (Madrid).....	56
DMH (Helsinki)	57
HECHT (Haifa).....	59

Introduction

SPICE community model provides the technological infrastructure that enables reasoning about citizens from their characteristics, opinions, and preferences. Each community identifies a group of citizens that are related according to certain similarity criteria.

Citizens belong to certain **explicit communities**, also called interest or target groups –the static *a priori* categories—, according to their demographic attributes. Besides, citizens may be *inferred* to belong to the so-called **implicit communities** –the emerging communities— created by using *citizen contribution attributes* that relate users with artefacts through different types of interactions: stories, opinions, selections, and others.

Explicit communities line up with the museum's interests, identify personas or profiles that have been formalized in the project ontology, and represent user archetypes. As previously described in “D3.3 Prototype User and Community Model”, each SPICE case study focuses on specific groups of interest, e.g., students from a certain school, teachers, members of an association, elder people, asylum seekers, children with serious illnesses, deaf people, and people from different religious and secular communities. Membership of citizens with respect to explicit communities would be asserted directly (and not inferred) and this is useful information in the analysis of preconceptions of homogeneity of opinions inside explicit communities.

Implicit communities are computed by community detection algorithms. Most of these algorithms rely on similarity functions that use *Artefact Attributes* and *Citizen Contribution Attributes* (pre-processed by semantic textual analysis, and stored in the user models of the citizens, see details in deliverable “D3.2: Semantic annotation of social curatorial products”). In the previous deliverable “D3.5: Prototype clustering techniques” we described algorithms to detect, visualize and explain implicit communities. Consequently, different communities may be formed using different configurations, also called **perspectives**. The so-called **reflection processes** of the SPICE project (described in the Interpretation Reflection Loop, IRL) are based on the fact that the same citizen can be classified in different communities using different perspectives.

Note that there is no clustering algorithm that can be universally used for every type of dataset and there is no similarity measure that can be used by every clustering algorithm on every dataset. Parameter settings are crucial in the performance of a clustering algorithm and similarity configuration (user-user, item-item) affects the results. The community model described in this final deliverable uses a selection of the best-performing clustering algorithms and similarity functions to create implicit communities based on perspectives. It stores a list of implicit and explicit communities and enables visualization and explanation processes. It is used in the visualization tool VISIR (described in Section “Case Study Analysis using VISIR” and deliverable “D5.3: Integrated interfaces for citizen curation”) that helps in the discovery and the exploration of emerging communities and their narrative identities. Good data visualizations help users analyze, validate, and explain the clusters generated by the algorithms. VISIR allows museum curators to configure, visualize and compare different perspectives at the same time, allowing reflection processes, and different viewpoints that can arise from analysis of their opinions expressed through the contributions, highlighting their similarities, differences, and relations. VISIR is also employed by the curators to define the perspectives that will be employed by the community model (and, hence, the recommendation system) to promote the interaction-reflection loop in the museum visitors.

Development and deploy

Community Model source code is publicly available at <https://github.com/spice-h2020/spice-community-model> Github repository. To deploy a Community Model, we need an environment file with configuration details like the user and password for the REST API and the databases, the seed file for the case study and the artefact data in a JSON file (if the Community Model uses this cache file instead of connecting to the

artwork datasets in the Linked Data Hub). “Appendix 1: Community Model Deployment” provides detailed instructions to deploy a Community Model instance using Docker¹.

Workflow

The Community Model is a subsystem inside the whole ecosystem of the SPICE project, shown in Figure 1. Citizen contributions stored in the Linked Data Hub are enhanced by a catalogue of reasoners that extracts the emotions, sentiments, values, polarity and crucial concepts, among others, from them. User Model aggregates and adapts the content extracted by the reasoners and feed the Community Model with these enhanced citizen contributions. The Community Model infer implicit communities based on these contributions and provides visualizations to the VISIR tool, which supports the visualization of discovered implicit communities for curators. Community model is also used by the recommender system to provide recommendations to the users in terms of the communities that a user belongs to and the similarities and differences among these communities.

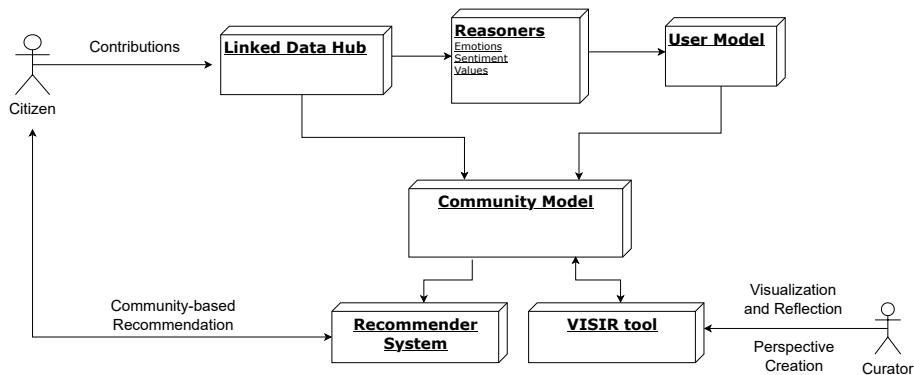


Figure 1. Community Model inside the SPICE ecosystem

Community Model must be configured beforehand to support this process. Figure 2 sketches the configuration workflow and next subsections describe this workflow until its deployment and use.

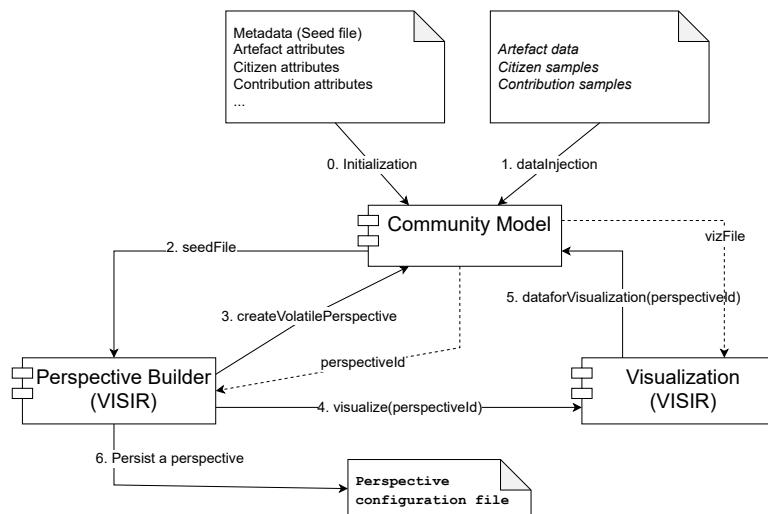


Figure 2. Community Model configuration

¹ <https://www.docker.com/>

Metadata configuration

First, the Community Model needs metadata information about the case study. This metadata is stored in a *seed file*² that contains information about the Artefact attribute types, the Citizen attributes, the Citizen Contribution attributes and the similarity measures that can be applied for each data type.

For the next step (perspective creation), the seed file can include a list with the artefacts involved in the case study, to define perspectives related with an explicit artefact. Additionally, the Community Model can be fed with citizen sample contributions, sample citizen data and artefact data to create the perspectives in an interactive way, described in the following section.

Creating perspectives

The term ***perspective*** refers to a specific configuration of the community model. A perspective specifies the configuration that allows to calculate the implicit communities in terms of the citizen contributions and how similar are the artworks that citizens interact with. Instead of defining the perspectives of a case study only in a programmatic way (based on the creation of configuration files by the developers), curators are involved in the creation of perspectives in an interactive way that shows in advance the kind of communities and insights that can be extracted from a perspective. VISIR tool (described in Section “Case Study Analysis using VISIR”) supports this process, integrating a perspective configuration tool (the Perspective Builder) to create volatile perspectives and communities, that can be visualized in the same tool. A perspective is created using a minimal and non-technical interface, specifically for curators without any technical knowledge about clustering algorithms and similarity functions, based on the creation of a sentence that explains (in plain English) how the clustering algorithm will behave, which citizen attributes will be employed, and which artefact attributes will be employed for deciding the similarity between two artefacts. Figure 3 shows the aspect of the Perspective Builder.

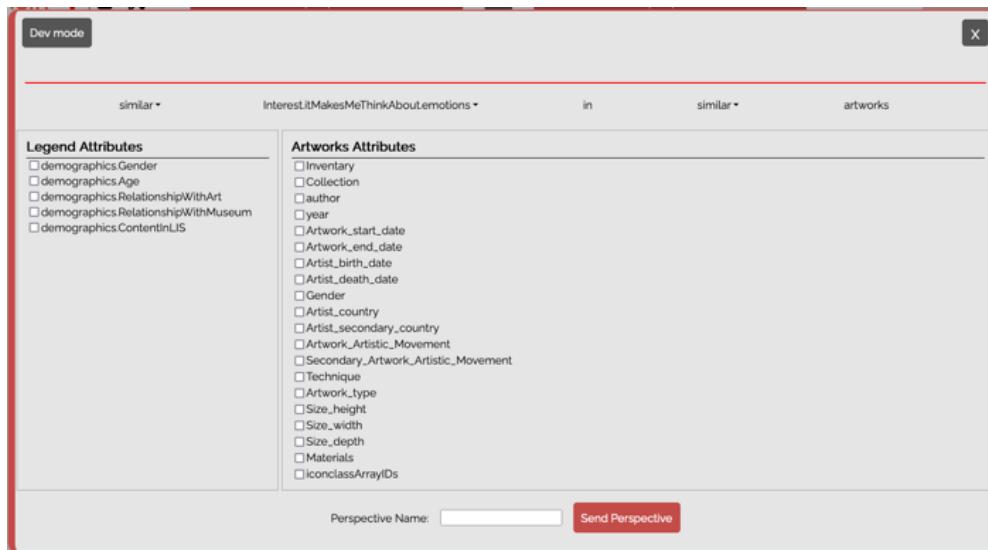


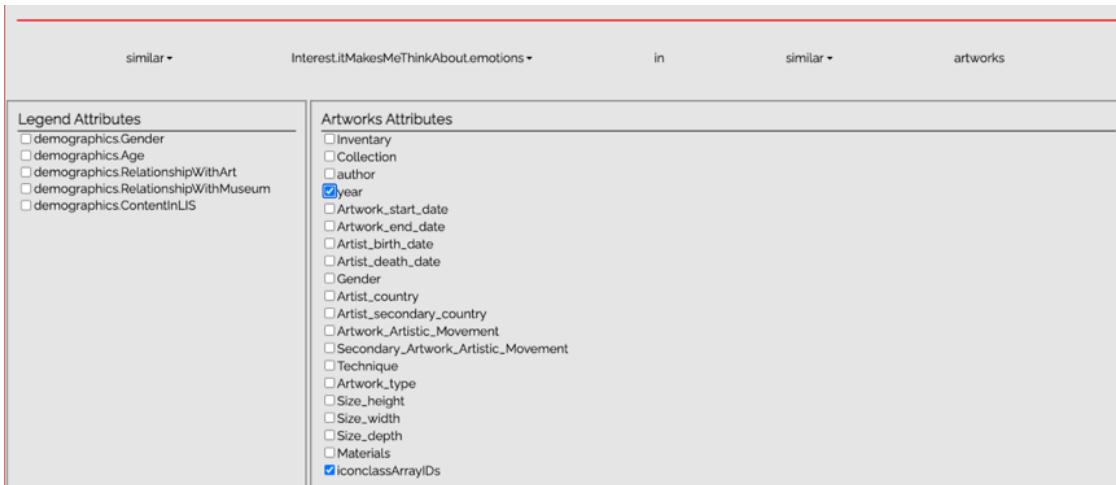
Figure 3: Interface to build Configuration Perspectives in the VISIR tool

Some examples about this configuration based on sentences are the following:

- A sentence like “Similar emotions in similar artworks” (Figure 4) represents a perspective that will perform the clustering using the Plutchik similarity function (see Section “Similarity between Plutchik emotions”) for the Citizen Contributions, grouping the citizens who has interacted with similar artefacts. Artefact similarity is computed based on the attributes selected by the curator (in the example, year and

² Sample seed files for SPICE case studies are available in “/demo/data/{museum}/seedFile.json” folder in the Community Model repository.

Iconclass³ [19] concepts). The clustering algorithm and the similarity functions employed in the perspective are defined beforehand and stored in the seed file by the developer.

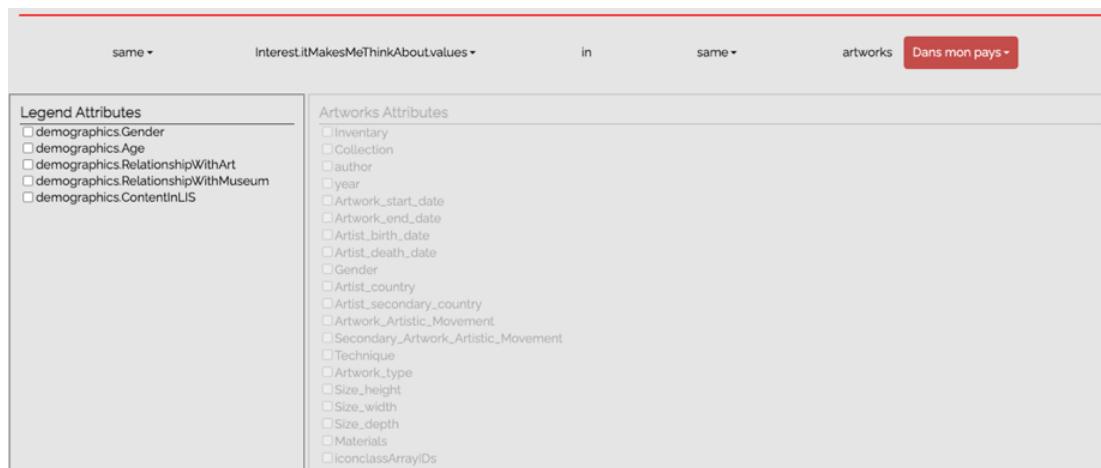


The screenshot shows the Perspective configuration interface with the following settings:

- Legend Attributes:** A list of demographic attributes: demographics.Gender, demographics.Age, demographics.RelationshipWithArt, demographics.RelationshipWithMuseum, demographics.ContentInLIS.
- Artworks Attributes:** A list of artwork attributes: Inventory, Collection, author, year, Artwork_start_date, Artwork_end_date, Artist_birth_date, Artist_death_date, Gender, Artist_country, Artist_secondary_country, Artwork_Artistic_Movement, Secondary_Artwork_Artistic_Movement, Technique, Artwork_type, Size_height, Size_width, Size_depth, Materials, iconclassArrayIDs.
- Criteria:** Similarity criteria: **similar** (dropdown), **Interest.itMakesMeThinkAbout.emotions** (dropdown), **in** (operator), **similar** (dropdown), **artworks** (entity).

Figure 4: Perspective configuration example. Selection of attributes and criteria that will participate in the community detection process “Similar emotions in similar artworks”

- A sentence like “Same values in same artworks” (Figure 5) represents a perspective that will perform the clustering using value equality on an artwork that can be chosen by the curator (in the example, *Dans mon Pays*, from the GAM museum).



The screenshot shows the Perspective configuration interface with the following settings:

- Legend Attributes:** A list of demographic attributes: demographics.Gender, demographics.Age, demographics.RelationshipWithArt, demographics.RelationshipWithMuseum, demographics.ContentInLIS.
- Artworks Attributes:** A list of artwork attributes: Inventory, Collection, author, year, Artwork_start_date, Artwork_end_date, Artist_birth_date, Artist_death_date, Gender, Artist_country, Artist_secondary_country, Artwork_Artistic_Movement, Secondary_Artwork_Artistic_Movement, Technique, Artwork_type, Size_height, Size_width, Size_depth, Materials, iconclassArrayIDs.
- Criteria:** Equality criteria: **same** (dropdown), **Interest.itMakesMeThinkAbout.values** (dropdown), **in** (operator), **same** (dropdown), **artworks** (entity), **Dans mon pays** (selected entity dropdown).

Figure 5 Perspective configuration example: “Same values in same artworks (*Dans mon Pays*)”

For more technical users, the Perspective Builder has a Dev mode (active when clicking on the Dev Mode Button, on the top right of Figure 6), where the perspectives can be fine-tuned. In this interface, users can choose the clustering algorithm, the weight employed for obtaining explainable clusters (see Section “Community explanations” for more details), the similarity threshold between two artefacts for computing the similarity between two contributions that involves these artifacts or the similarity measures that can be used for each artefact attribute.

³ See Section “Similarity between taxonomies” - Ikonclass

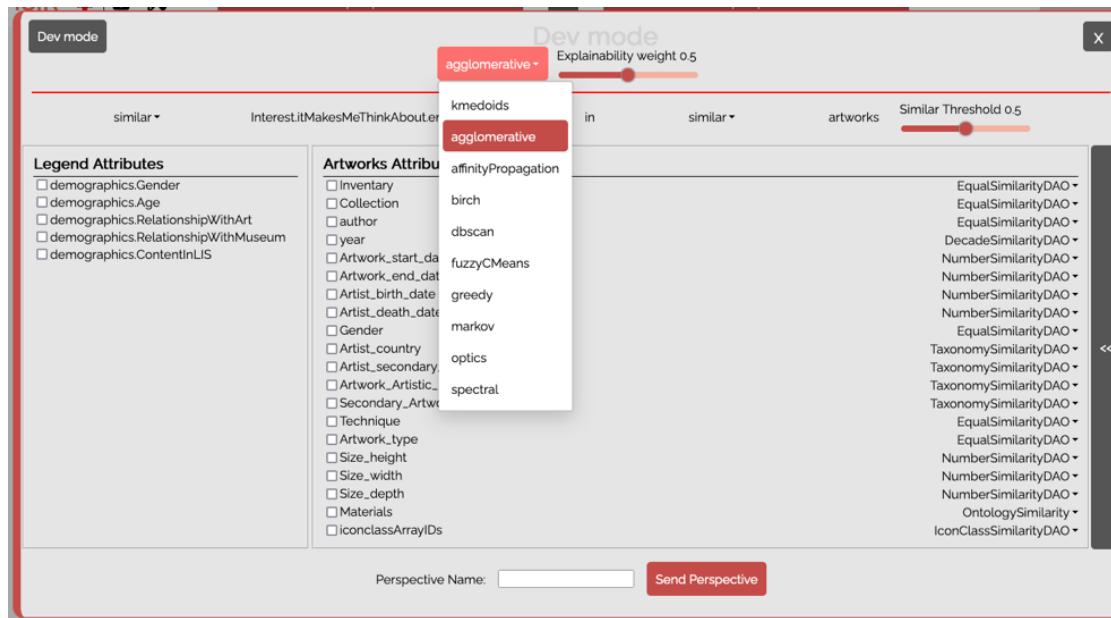


Figure 6. Perspective builder in Dev Mode.

Perspective maintenance

Perspectives are volatile unless they are persisted. When curators find interesting perspectives, they can be downloaded and persisted using VISIR (see Figure 7). The tool generates a perspective configuration file that contains the parameters for the clustering: citizen attributes, artefact attributes, contribution attributes, similarity functions used, the clustering algorithm and, in general, any other parameter that the Community Model may need.

The perspective configuration files are used later for configuring a Community Model that will be employed by the recommender system (described in Deliverable 3.8) to provide recommendations to the citizens during a visit. Perspective configuration files can be uploaded in a new Community Model using its API, as described in the next section.

Finally, while experimenting with different similarity measures and clustering algorithms, some perspectives cannot be considered useful for curator's needs. VISIR uses the same interface to select one or more perspectives and confirm their deletion from the Community Model (see Figure 7).



Figure 7. VISIR interface for persisting and removing perspectives

Interaction with other SPICE systems

Other systems can interact with the community model through the REST API, which is described Section "Community Model API". For example:

- The perspectives created and persisted using VISIR can now be uploaded in the Community Model server using a POST request.
- The User Model will perform POST requests to insert citizen's demographic and contribution data.
- The Recommender System will perform different GET requests to access the data about citizens and their belonging to different communities needed to perform a recommendation.

Depending on the internal state of the Community Model, some requests will start the clustering process for one or more perspectives. Depending on the number of contributions, users and artefacts, the clustering process can take some time to return the expected results, so a batch job system has been implemented. The CM-API can return a job to other system, which can be used to confirm the request status and retrieve the desired information when the community detection is completed.

Community Model architecture

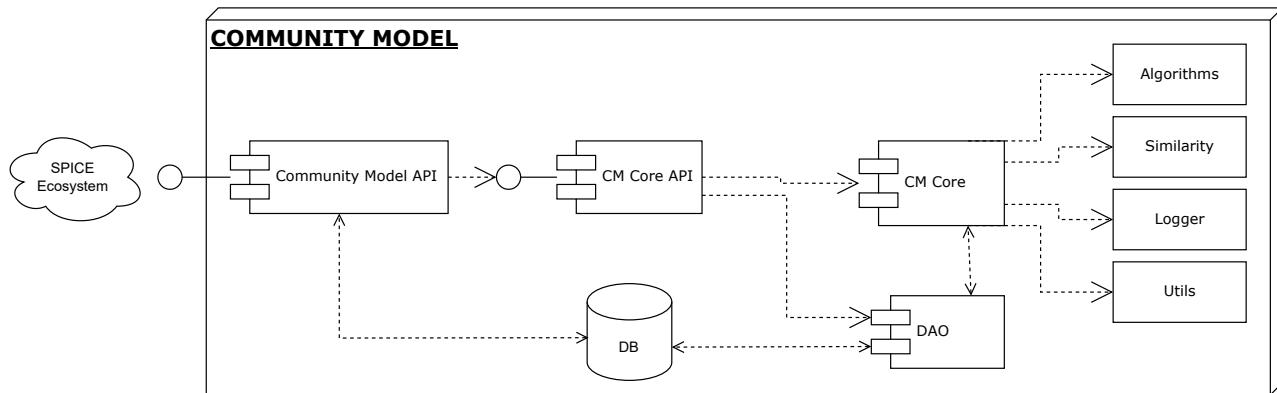


Figure 8. Community Model architecture

The Community Model has been developed as a service that provides a REST API for communication with the different systems that are part of the SPICE application ecosystem. It is fully implemented in Python and the source code is available at the Github SPICE repository: <https://github.com/spice-h2020/spice-community-model>. As illustrated in Figure 8, Community Model is composed of the following components.

Community Model API

The Community Model API (CM-API) has been described in Deliverable 6.8: “APIs Specification and Deployment”. It exposes a set of REST-based operations for accessing information about implicit and explicit communities, the creation and access to perspectives, as well as endpoints for operations related to similar and dissimilar communities. CM-API is also employed by the User Model to notify changes in user attributes and the creation of new user generated content. It also includes endpoints for VISIR, as well as some endpoints for development purposes.

The entry points in CM-API are the following:

- Perspective entry points: Provide functionality for retrieving information about perspectives, as well as for creating and removing perspectives.

GET	/perspectives	Get all perspectives in the model	✓
GET	/perspectives/{perspectiveId}	perspective description	✓
GET	/perspectives/{perspectiveId}/communities	Communities with the same perspective	✓
POST	/perspective	Add perspective	✓
GET	/perspectives	Get all perspectives in the model	✓
GET	/perspectives/{perspectiveId}	perspective description	✓
GET	/perspectives/{perspectiveId}/communities	Communities with the same perspective	✓
POST	/perspective	Add perspective	✓
DELETE	/perspectives/{perspectiveId}	Delete a perspective by ID	✓ 🔒

- Community entry points: To query information about communities

GET	/communities	Get all communities in the model	✓ 🔒
GET	/communities/{communityId}	Get community description and explanation	✓ 🔒
GET	/communities/{communityId}/users	Users who belong to a community	✓ 🔒
GET	/communities	Get all communities in the model	✓ 🔒
GET	/communities/{communityId}	Get community description and explanation	✓ 🔒
GET	/communities/{communityId}/users	Users who belong to a community	✓ 🔒

- Users' entry points: Provide access to the users' (citizen) data and to add it to the Community Model.

GET	/users/{userId}/communities	Communities that a user belongs	✓ 🔒
POST	/users/{userId}/update-generated-content	Update community model with new user generated content	✓ 🔒
GET	/users/{userId}/communities	Communities that a user belongs	✓ 🔒
POST	/users/{userId}/update-generated-content	Update community model with new user generated content	✓ 🔒

- Similarities' entry points: To provide services about similarity and dissimilarity between communities.

<code>GET</code>	<code>/communities/{communityId}/similarity</code> K-most similar communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/similarity/{otherCommunityId}</code> Similarity between two communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/dissimilarity</code> K-most dissimilar communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/dissimilarity/{otherCommunityId}</code> Dissimilarity between two communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/similarity</code> K-most similar communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/similarity/{otherCommunityId}</code> Similarity between two communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/dissimilarity</code> K-most dissimilar communities	✓	🔒
<code>GET</code>	<code>/communities/{communityId}/dissimilarity/{otherCommunityId}</code> Dissimilarity between two communities	✓	🔒

- Job Manager entry points: for knowing a job's state or for checking the pending jobs in the system.

<code>GET</code>	<code>/jobs-manager/jobs/{jobId}</code> Job status	✓	🔒
<code>GET</code>	<code>/jobs-manager/jobs</code> Status of all jobs	✓	🔒
<code>GET</code>	<code>/jobs-manager/jobs/{jobId}</code> Job status	✓	🔒
<code>GET</code>	<code>/jobs-manager/jobs</code> Status of all jobs	✓	🔒

- VISIR entry points: The endpoints for communication between the Community Model and VISIR.

<code>GET</code>	<code>/visir/seed</code> Get seed file	✓	🔒
<code>GET</code>	<code>/visir/files</code> List of all available visualization files	✓	🔒
<code>GET</code>	<code>/visir/files/{fileId}</code> Return the specified visualization file	✓	🔒
<code>GET</code>	<code>/visir/seed</code> Get seed file	✓	🔒
<code>GET</code>	<code>/visir/files</code> List of all available visualization files	✓	🔒
<code>GET</code>	<code>/visir/files/{fileId}</code> Return the specified visualization file	✓	🔒

- Development entry points: for managing database dumps and logs during development.

GET	/database-controller/dump	Get current database state	▼	🔒
POST	/database-controller/dump	Load data into database state	▼	🔒
GET	/database-controller/dump	Get current database state	▼	🔒
POST	/database-controller/dump	Load data into database state	▼	🔒
GET	/logs	Get n latest logs	▼	🔒
GET	/logs/dateRange	Get logs between date range	▼	🔒
GET	/logs	Get n latest logs	▼	🔒
GET	/logs/dateRange	Get logs between date range	▼	🔒

CM Core

The CM Core is the main component of the Community Model, manipulating the citizen data and arranging them in communities. Its logic can be categorized into five main activities depending on their goal:

- Initialization of the clustering configuration using the citizen and perspective data provided by the API server.
- Community detection using clustering algorithms, described in Section “Clustering algorithms”.
- Computation of similarity using the similarity measures described in Section “Similarity Measures”.
- Generation of explanation for the discovered communities, described in Section “Community explanations”.
- Generation of JSON objects encoding the clustering output and storage in the database through the DAO component.

This component delegates these activities in different modules, as shown in Figure 8. It also uses a logger system to store information about these processes for debugging purposes.

DAO

DAO component (Data Access Object) provides CRUD functionality required by other components to access and store information in the database. The database uses MongoDB as a NoSQL database management system and the information about perspectives, communities, citizens, cache similarity matrices and data visualization for VISIR are stored in JSON collections.

CM Core API

This module manages the communication between the Community Model API and the Community Model core: it receives the data required for the Community Model to perform the clustering and it provides updates to the Community Model API about the request status. It isolates the CM Core from the public Community Model API available for other systems in the SPICE ecosystem.

Community detection

The main objective of the community model is to group museum visitors or citizens into communities using clustering techniques and similarity measures, with a focus towards explaining their unique properties and understanding why each citizen belongs to their community.

Once the Community Model starts the community detection process, it retrieves the information about the involved perspectives from the database and follows the procedure below for each of them:

1. Initialization of similarity measures associated to the artefact attributes encoded in the perspective.
2. Computation of the distance matrixes between the artefacts using the previously described similarity measures. These matrixes are combined accordingly to generate a general similarity (or distance) matrix to compare any pair of artefacts.
3. Initialization of the similarity measure associated to the citizen contribution attribute encoded in the perspective.
4. Computation of the distance matrix between citizens using similarity measures on their contributions.
The steps to calculate the distance between two citizens are described below:
 - a. Given a pair of citizens, we have two corresponding lists with the artworks they have contributed to (with comments, stories, likes,...). The community model designates the biggest list as list A and the smallest list as list B.
 - b. The distance between the citizens is initialized to 0.
 - c. For each artefact in list A (artefact A), it retrieves the most similar artefact in list B (artefact B). The similarity between them must be equal or greater than the similarity threshold between two artefacts specified in the perspective. If this condition is satisfied, it computes the distance between the two contribution attributes and adds the value to the distance between the citizens.
 - d. It divides the final distance by the number of valid pairs of artefacts in c). If this number is 0, it sets the final distance to 1 because the citizens did not interact with similar artworks.
5. Clustering using the perspective algorithm (more details in Section “Clustering algorithms”), using the precomputed distance matrix between citizens as a parameter. This process is repeated, starting from 2 clusters up to a maximum of clusters equal to the number of citizens, until all the obtained clusters are explainable, which means they conform to the restrictions imposed upon the attributes used for citizen similarity (described in Section “Community explanations”). If the number of clusters cannot be directly provided to the clustering algorithm, an approximation based on the available parameters is implemented to simulate it.
6. Generation of JSON objects encoding the clustering output and the similarity between communities. Then, they are stored in the database.

Next subsections will explain the specific clustering algorithms and similarity functions employed in the community detection process.

Clustering algorithms

State of the art in clustering techniques is documented in Deliverable “D3.5: Prototype clustering techniques”. In this section, we provide a short introduction of the algorithms compatible with the latest version of the community model as well as a summary of the advantages of disadvantages of the categories they belong to, which may be used as guidelines for choosing an algorithm at the perspective definition stage described in Section “Creating perspectives”. The selected clustering algorithms, described below, are implemented in the Algorithm module that appear in the Community Model architecture (Figure 8).

K-medoids [2, 3] is a variation of the k-means algorithm [17] where the center of a cluster is represented by the nearest data points to the real center, which makes it more robust to outliers. Initially, given the number of clusters k as a parameter, it randomly selects k points as the center of the clusters and assigns each point to the nearest cluster. This process is repeated iteratively until a convergence criterion is satisfied. The algorithm heavily depends on the number of clusters and the initial centers randomly chosen.

It is one of the most used among the partition-based clustering algorithms, a category with the advantage of being simple and able to efficiently find a solution in a relatively low time if the dataset is not too large and there are not many outliers.

AgglomerativeClustering [1,4] is a hierarchical clustering algorithm using an agglomerative strategy. It starts with one cluster per data point and iteratively merges pair of clusters of sample data in such a way that a linkage criterion is minimized. The linkage criterion determines which distance to use between sets of observation or clusters.

- ward: minimizes the variance of the clusters being merged.
- average: uses the average of the distances of each observation of the two sets.
- Complete or maximum: uses the maximum distances between all observations of the two sets.
- Single: uses the minimum of the distances between all observations of the two sets.

Birch (Balanced Iterative Reducing and Clustering using Hierarchies) [5, 6] is another agglomerative algorithm that creates a clustering feature tree in an incremental and dynamic way, so it performs well with large datasets. However, it is sensitive to the order of the data points and does not work well if clusters are not spherical.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [1, 7, 8] is one of the most popular density-based clustering algorithms. It was designed to discover clusters of arbitrary shapes. The main idea is the neighborhood of a datapoint that belongs to a cluster containing a minimum number of points or density. The algorithm distinguishes between core points and border points because the minimum number of points will be different in both cases.

We chose to include a density-based algorithm to assist museum curators in their discovery of clusters of arbitrary shapes, which is one of the weakest points of partition-based and agglomerative clustering. On the other hand, it is important to note that this algorithm also suffers from drawbacks like being heavily dependent on the input parameters or losing quality when the density space is not even.

OPTICS: (Ordering Points to Identify the Clustering Structure) [1, 9, 10] produces a density-based clustering ordering, that can be represented by a reachability plot. Then, a hierarchical cluster structure can be generated using this ordering. It requires more memory and computational power than DBSCAN, but it is less sensitive to the input parameters which makes it more suitable to handle different museums.

Markov Cluster Algorithm [13, 14] belongs to the category that detects communities through graph analysis, based on the graph theory that communities or clusters are groups of nodes having similar properties, affiliations or roles that are different from other nodes in the network [11, 12]. In this algorithm, each graph node represents a data point while the edges between them encode the distance between them. This algorithm follows the random walk principle which considers that if you start walking randomly from a node, you are more likely to move around in the same cluster than to cross clusters.

SpectralClustering [1, 15] is an algorithm from the scikit-learn library which applies clustering to a projection of the normalized Laplacian matrix. It belongs to a category of algorithms using the eigenvectors of a similarity matrix extracted from a graph to reveal implicit properties hidden in the former matrix.

The main advantage of this algorithm is its lack of assumptions about cluster shape, which makes it suitable for museum curators handling datasets with irregular data unsuited for density-based clustering. However, there are some negative characteristics, such as its slowness, its lack of popularity which translates in a smaller community with which discuss issues, its sensitivity to initial parameters and its complexity, which makes it difficult to explain to people without a strong math background.

Affinity propagation [1, 16] detects communities using communications between data points. Each data point notifies the other points about their relative closeness to it. Next, each target answers this input with its disposition to associate with the sender, considering the messages sent by other senders. Once again, after considering their replies, senders update their stances regarding the targets. This process continues until a consensus is reached. Once a relationship between a data point and one of its targets is set, that target becomes the point's exemplar. All points with the same exemplar are placed in the same cluster.

Similarity Measures

Similarity measures are employed during the Community Detection algorithm described before for two purposes:

- Computing the similarity among artefacts using the artefact attributes.
- Computing the similarity among citizens based on their contributions using the citizen contribution attributes.

Although most of the similarity functions are for a specific purpose, some of them can be used for both. Furthermore, the artefact or citizen contribution attributes may be associated with one (e.g., age) or more values (e.g., list of emotions in a citizen contribution). In this last case, the similarity may be calculated by integrating the similarity of pairs of attribute values according to the methodology described in subsection "Similarity between lists"

The following subsections will describe the similarity functions implemented in the Community Model that conform to the **Similarity** module that appears in the Community Model architecture (Figure 8).

Equal similarity

It assigns the maximum distance to any values that are not equal. As special cases, we mention dictionaries and lists. Dictionaries are considered equal if the keys with the highest value are equal. Lists are considered equal if the intersection is not empty. It is implemented in the `EqualSimilarityDAO` class, and it can be used for both artefact and citizen contribution attributes.

Similarity between Plutchik emotions

This similarity measure uses the Plutchik's Wheel of Emotions [18] as the foundation. This model, depicted in Figure 9, has 8 basic emotions (anger, anticipation, joy, trust, fear, surprise, sadness, disgust) with each of them assigned to one of the 8 spokes of the wheel in such a way that opposite emotions are placed in opposite spokes (e.g., joy and sadness, anger, and fear). Each spoke lists 2 additional emotions, which are variations of the 8 basic ones, ranging in intensity. Furthermore, it also includes another 8 emotions which describe what you feel if you combine the emotions of the two spokes it sits between. It is implemented in the `ExtendedPlutchikEmotionSimilarity` class, and it can be used for citizen contribution attributes.

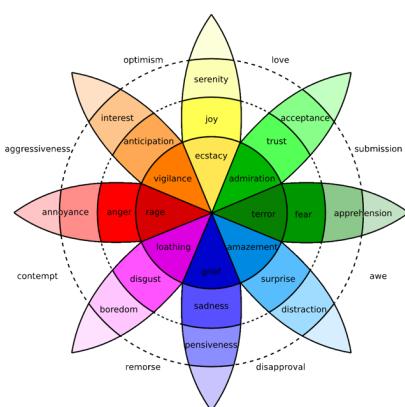


Figure 9. Plutchik's Wheel of Emotions.

Since two opposite emotions are separated by the maximum distance of 1, we can extrapolate that each spoke is separated by 0.25 as there are 4 spokes between them. Analogously, the distance between a spoke emotion and their intermediate one is 0.125. Differences between two emotions belonging to the same spoke are not clear but given that they only differ in intensity we set the distance between spoke levels to 0.01. Following this reasoning, we compute the distance between two emotions belonging to the Plutchik's wheel of emotions according to the following formula:

$$DIST'(A, B) = \min\left(\frac{S(A) - S(B)}{8}, \frac{S(B) - S(A) + 16}{8}\right) + 0.01 * (S_L(A) - S_L(B))$$

where **A** and **B** are the first and second emotion, **S(A)** is the number of the spoke or intermediate space between them where the emotion **A** is placed, starting from a reference spoke (for example, the yellow spoke associated to joy), and **S_L(A)** is the level of intensity of emotion A, quantified from 0 to 2 (e.g., 2 for serenity, 1 for joy and 0 for ecstasy).

If the output is more than 1 or less than 0 (e.g., opposite emotions in different spoke levels), the final distance is set to 1 and 0, respectively.

$$DIST(A, B) = \min(1, DIST'(A, B))$$

This similarity measure may be used to compute the distance between interactions involving more than one emotion. Let two citizen-artwork interactions I_A and I_B be associated to more than one emotion, the distance between them is calculated as a linear combination of the distances between each emotion evoked by I_A and the corresponding emotion evoked by I_B with the lowest distance to it, as described in Section "Similarity between lists". Furthermore, if the interaction emotions are sorted according to their confidence score, we may only consider the k (e.g., $k = 2$) emotions with the highest confidence score to reduce data noise.

Similarity between taxonomies

Similarity between taxonomies and ontologies (implemented in `TaxonomySimilarityDAO` and `OntologySimilarity`) are employed to calculate the similarity between two feature values belonging to a taxonomy using the python package `networkx` [24] and `OwlAlready2` [25], respectively. These similarity functions can be used for both artefact and citizen contribution attributes but they depend on the existence of a taxonomy or ontology for the specific attribute and an alignment between the attribute values and the taxonomy/ontology concepts.

Let K be an inner node of a certain concept hierarchy (Figure 10), then L_K denotes the set of all leaf concepts from the sub-tree starting at K. Further, $K_1 < K_2$ denotes that K_1 is a successor node (subconcept) of K_2 . Moreover, $\langle K_3, K_4 \rangle$ stands for the most specific common object class of K_3 and K_4 , i.e., $\langle K_3, K_4 \rangle > K_3$ and $\langle K_3, K_4 \rangle > K_4$ and it does not exist a node $K' < \langle K_3, K_4 \rangle$ such that $K' > K_3$ and $K' > K_4$ holds.

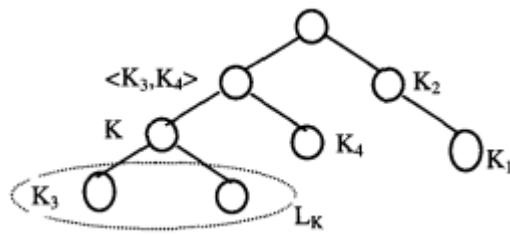


Figure 10. An example of concept hierarchy

The implementation uses the following similarity function:

$$SIM(K_3, K_4) = \frac{\text{depth}(\langle K_3, K_4 \rangle)}{\text{MAX(depth)}}$$

For example, using the materials ontology⁴ in Figure 11, we compute similarity based on the position of concepts (or individuals):

- Glass and ceramics are at depth 3 while its common parent, no organic, is at depth 2. Consequently, SIM (glass, ceramics) = 2/3.
- Silver and glass do not share a common parent. Therefore, SIM (silver, glass) = 0.

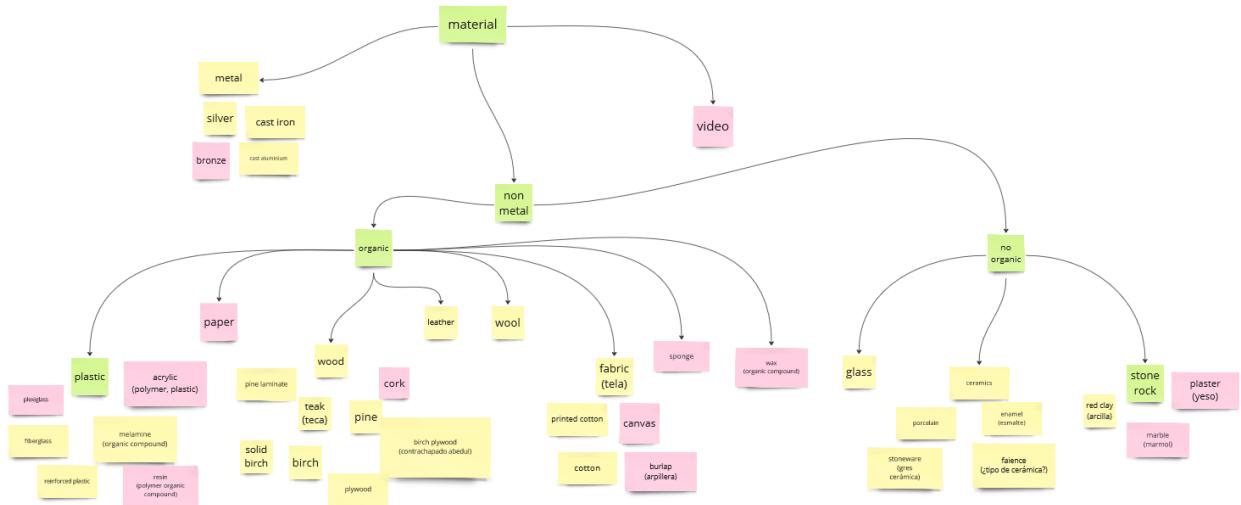


Figure 11. Materials ontology

An extension of this kind of similarity functions is the theme similarity using the Iconclass ontology (and implemented in the `IIconClassSimilarityDAO` class): computes the similarity between two artefacts using entities that belong to the Iconclass ontology. This system started in the 1940s when Henri van de Waal began to develop ideas for a universal classification for the subject matter of works of art [19]. Since then, it has been frequently reviewed and discussed, providing one of the most complete methodologies for classifying subjects, themes, and motifs in art. As illustrated in Figure 12, each Iconclass entity is noted as a combination of letters and numbers encoding the hierarchical path leading to it. In this way, 11G2 means “activities of angels (in heaven)”, 11G3 translates to “angels fighting” and its common prefix 11G is associated to “angels”, their common element. This similarity measure takes advantage of this property to calculate the depth of Iconclass concepts required for the methodology used in the similarity between taxonomies. For example, 11G2 and 11G3 have a depth of 4, while 11G has a depth of 3. Thus, the similarity between the two first concepts is 75%.

⁴ This ontology has been employed in DMH and GAM case studies for computing artefact similarity.

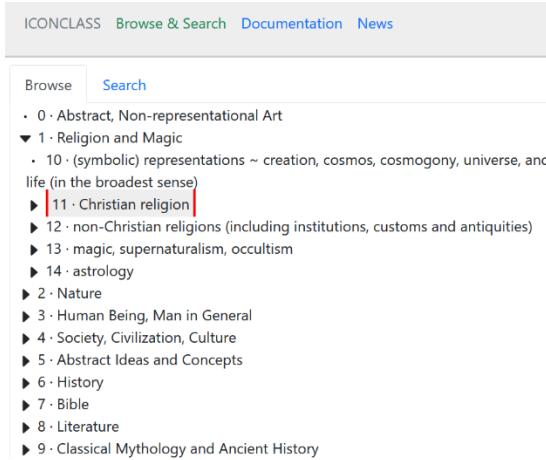


Figure 12. Iconclass hierarchy

Expert-based similarity

These similarity functions are used to compute similarities on special attributes, where the similarity or distance values is directly given by experts or museum curators (e.g., beliefJ in the HECHT museum, Haidt's moral foundations values or sentiments). These similarity functions can be used for both citizen contribution and artefact attributes, and they are implemented as a lookup in a table created by the experts. It is implemented by the `TableSimilarityDAO` class, and the Community Model uses the following expert-based similarity tables.

- Sentiments: this is employed for citizen contribution attributes enriched by the sentiment reasoner (Figure 13)

	positive	neutral	negative
positive	0.0	0.5	1.0
neutral	0.5	0.0	0.5
negative	1.0	0.5	0.0

Figure 13. Distance table for sentiments.

- BeliefR: this is employed for a citizen contribution attribute in HECHT case study about the reasons for Roman Rebellion (Figure 14).
 - **A NatPridePro:** In some cases, one must fight even if there is no realistic chance of winning so as not to come to terms with a reality of humiliation and therefore the Jews had to go in revolt against Rome.
 - **B ReligiousPro:** The history of the Jewish people has undergone many miracles and therefore one should not be afraid to fight against foreign rule - even if it is as powerful as the Roman Empire.
 - **C RealisticPro:** The attack on the part of the Roman government and the non - Jewish inhabitants of the land, threatened the physical, spiritual, and economic existence of the people - and therefore it was necessary to revolt.
 - **D ExtremistNeg:** The great rebellion, defeat, and destruction demonstrate the danger of turning to extremism.
 - **E ReligiousNeg:** Many Jews argued that the decrees of Roman rule should be accepted because it was part of the divine plan and therefore going into rebellion was a religious mistake.
 - **F RealisticNeg:** It is forbidden to go to war if there is no realistic chance of achieving the goals of the war. Therefore, the revolt in Rome was a mistake for the rebels.

Key	ANatPridePro	BReligiousPro	CRealisticPro	DExtremistNeg	EReligiousNeg	FRealisticNeg	NoOpinion	DK
ANatPridePro	0.0	0.2	0.4	1.0	0.9	1.0	0.5	0.5
BReligiousPro	0.2	0.0	0.4	0.9	1.0	1.0	0.5	0.5
CRealisticPro	0.4	0.4	0.0	1.0	1.0	1.0	0.5	0.5
DExtremistNeg	1.0	0.9	1.0	0.0	0.2	0.4	0.5	0.5
EReligiousNeg	0.9	1.0	1.0	0.2	0.0	0.4	0.5	0.5
FRealisticNeg	1.0	1.0	1.0	0.4	0.4	0.0	0.5	0.5
NoOpinion	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0
DK	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0

Figure 14. Distance table for beliefR (opinions about Roman Rebellion)

- BeliefJ: this is employed for a citizen contribution attribute in HECHT case study about whether Joseph Flavius is considered as a traitor (Figure 15)
 - NotTraitor
 - Traitor
 - CantJudge

Key	NotTraitor	CantJudge	Traitor
NotTraitor	0.0	0.5	1.0
CantJudge	0.5	0.0	0.5
Traitor	1.0	0.5	0.0

Figure 15. Distance table for beliefJ (is Joseph Flavius a traitor?)

- BeliefE: this is employed for a citizen contribution attribute in HECHT case study about the citizen opinion about the bias in an exhibition (Figure 16)
 - Oppose
 - NoOpinion
 - Justify
 - Balanced

Key	Justify	Balanced	Oppose	NoOpinion	DK
Justify	0.0	0.5	1.0	0.5	0.5
Balanced	0.5	0.0	0.5	0.5	0.5
Oppose	1.0	0.5	0.0	0.5	0.5
NoOpinion	0.5	0.5	0.5	0.0	0.0
DK	0.5	0.5	0.5	0.0	0.0

Figure 16. Distance table for beliefE (is the HECHT exhibition biased?)

- Haidt moral foundation values: it is based on the Moral Foundations Theory (MFT) [20], by Graham and Haidt, as formalised in [21] and described in Deliverable “D6.6: Knowledge based exploration support”. It is employed for citizen contribution attributes enriched by the value reasoner
 - Care vs harm.
 - Fairness vs Cheating.
 - Loyalty vs Betrayal.
 - Authority vs Subversion.
 - Sanctity vs Degradation.
 - Liberty vs Oppression.

The table shown in Figure 17 indicates the distance between each pair of moral foundation values.

Key	care	harm	fairness	cheating	loyalty	betrayal	authority	subversion	santicty	degradation	liberty	oppression
care	0.0	1.0	0.1	0.9	0.2	0.8	0.5	0.5	0.5	0.5	0.2	0.8
harm	1.0	0.0	0.9	0.1	0.8	0.2	0.5	0.5	0.5	0.5	0.8	0.2
fairness	0.1	0.9	0.0	1.0	0.2	0.8	0.5	0.5	0.5	0.5	0.3	0.7
cheating	0.9	0.1	1.0	0.0	0.8	0.2	0.5	0.5	0.5	0.5	0.7	0.3
loyalty	0.2	0.8	0.2	0.8	0.0	1.0	0.3	0.7	0.5	0.5	0.5	0.5
betrayal	0.8	0.2	0.8	0.2	1.0	0.0	0.7	0.3	0.5	0.5	0.5	0.5
authority	0.5	0.5	0.5	0.5	0.3	0.7	0.0	1.0	0.5	0.5	0.8	0.2
subversion	0.5	0.5	0.5	0.5	0.7	0.3	1.0	0.0	0.5	0.5	0.2	0.8
santicty	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.0	1.0	0.6	0.4
degradation	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	0.0	0.4	0.6
liberty	0.2	0.8	0.3	0.7	0.5	0.5	0.8	0.2	0.6	0.4	0.0	1.0
oppression	0.8	0.2	0.7	0.3	0.5	0.5	0.2	0.8	0.4	0.6	1.0	0.0

Figure 17. Distance table for Haidt's moral foundations

Range similarity

It computes the similarity between two numbers according to the following equation:

$$\frac{(A - B)}{\max(A, B)}$$

This similarity function can be used for both citizen contribution and artefact attributes that are numeric, and it is implemented in the `NumberSimilarityDAO` class.

Year similarity

Although the similarity between the years that commonly appear as artefact attributes can be computed using the previous one, Community model implements functions (`DecadeSimilarityDAO` and `CenturySimilarity` classes, respectively) that interpret the number as a year, obtains the decade and century associated to this number, respectively, and applies the previous similarity between numbers

Color similarity

Color similarity function computes the difference between two colors using the Delta E color difference [26]. Delta E is a standard measurement introduced in 1976 by the International Commission on Illumination (CIE) to aid the fields of colorimetry, photometry, and imaging. It quantifies the difference between two colors as perceived by the human eye on a scale from 0 to 100, where 0 is less color difference, and 100 indicates complete distortion. Unfortunately, the first approaches lacked accuracy because of the nature of our eyes, which are more sensitive to changes in Chroma than lightness. For example, our eyes recognized differently two yellows and two greens with the same dE between them. These circumstances have encouraged the development of newer dE equations. This similarity measure uses the Delta E 2000 color difference, proposed by CIE TC1-47 in CIE Publ.142 in 2001 [22], standardized in 2013 and updated in 2022 [23]. It is currently the most complicated, yet most accurate, CIE color difference algorithm available.

It is implemented in the `ColorSimilarity` class, and it is used for artefact color attributes. This attribute is commonly represented as a name that is converted into RGB and HSVL colors to compute the similarity using the Delta E 2000 color difference.

Similarity between lists

It is a combination similarity function that aggregates the similarity between two lists of attribute values (e.g., color attributes in artefacts or a list of evoked emotions for citizen contribution attributes). It is implemented as part of the `SimilarityDAO` class.

The computation process runs as follows: Let `bigValuesList` be the list with more values and `smallValuesList` the one with less values. The similarity assigned to each value "valueA" of "`bigValuesList`" is the highest similarity value among the values obtained from computing the similarity between "valueA" and each of the values in "`smallValuesList`". The similarity between the two lists is the mean average of the previous similarity values assigned to the values of "`bigValuesList`".

Community explanations

The communities generated by a clustering algorithm should have a meaning for the expert who is analyzing the resulting clusters. If the communities are not meaningful the community detection process might be useless. For this reason, the Community Model described in this deliverable not only generate communities based on clustering algorithms and similarity functions but also uses the later functions and the data attributes involved in the community detection process (citizen attributes, citizen contribution attributes and artefact attributes) to provide an explanation that describes the community.

Community explanations describe the discovered communities using three complementary approaches:

- Community explanation based on explicit attribute values of the citizens who belong to a community. The explanation shows the percentage distribution of values for each explicit attribute.

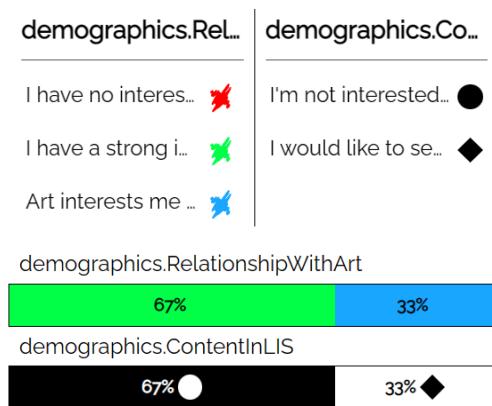


Figure 18. Distribution of the explicit citizen attributes.
Relationship with art: strong interest in art (green), little interest (blue).
ContentInLIS: interested in sign language (circle), not interested (diamond).

For example, Figure 18 shows how an explanation for the community depicted in Figure 19 is displayed in VISIR. According to this explanation 67% of the citizens who belong to this community have a strong interest in art (green) while 33 % are a little interested (blue). On the other hand, 67% are not interested in sign language content (circle) while 33% of them are (diamond).

- Centroid-based explanations are based on creating a non-existent model citizen that acts as the representative citizen of this community. The data that characterizes this model citizen meets two requirements: it has the highest average similarity with all the other members of the community, and it belongs to the dominant value for each explicit attribute. To define this citizen, we choose the existing community member with the highest average similarity with other community members, also known as the medoid, and assign it the dominant values of the explicit citizen attributes.

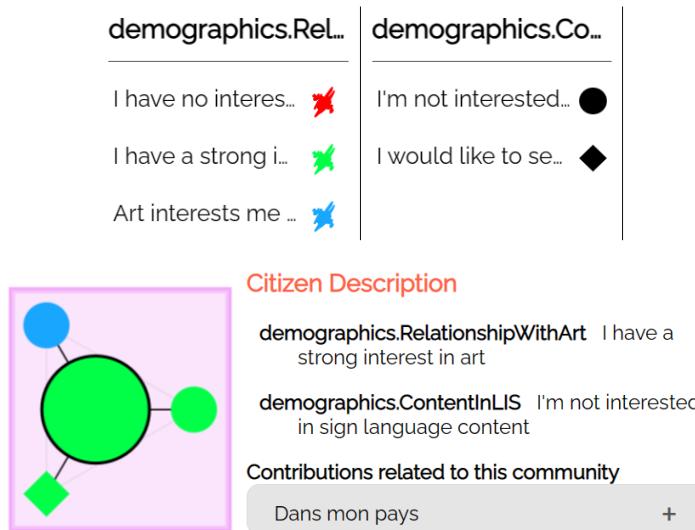


Figure 19. Representative community citizen.
Relationship with art: strong interest in art (green), little interest (blue).
ContentInLIS: interested in sign language (circle), not interested (diamond).

For example, the community in Figure 19 is composed by three original citizens but VISIR displays a fourth citizen. This is the centroid that the Community Model creates as explanation, with the most common explicit citizen attributes (see Figure 18), “I have a strong interest in art” (green) and “I am not interested in sign language content” (circle), and the same contributions as the community member with the highest average similarity with all the other members of the community. The centroid can be differentiated from real citizens because it is represented by a bigger symbol in the community centre.

- Community explanation based on the representative values of the similarity attributes involved in the clustering process (citizen contribution attributes and artefact attributes). The explanation is created in two stages. First, the model stores information describing the relationship between the citizen similarity attributes involved in the computation of the distance matrix used by the clustering algorithm. Second, the model filters the previous information until only the citizen contributions used to compute the similarity between community members remain, reducing data noise and providing representative community interaction data as an explanation. Since these descriptions are usually too long or complex to be properly understood by human beings, a simplification process is applied. Here, we distinguish between similarity attributes that do not follow a hierarchical structure and the ones that do, such as taxonomies and ontologies.

In the first case, for each citizen-citizen comparison, we store in a list the attribute values involved in the computation of its distance with the other user and reduce it to the most frequent value. At the community explanation stage, we perform another simplification: For each citizen, we consider the values encoding its relationships with other community members and extract the most frequent value once again. Then, this information is displayed in the visualization tool as a distribution following the word cloud model.

Percentage distribution of the implicit attribute
(interest.itMakesMeThinkAbout.emotions):

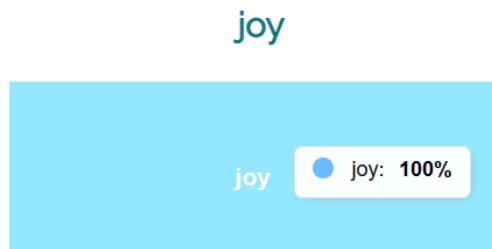


Figure 20. Percentage distribution of the emotions felt by the community citizens.

Let us demonstrate how it works with an example using the emotions (citizen contribution attribute) evoked by artworks. As previously stated in Section “Clustering algorithms”, the similarity between two citizen emotions may involve zero or more contributions. For each contribution and citizen, we store the predominant emotion felt by that citizen according to her contribution. From these emotions, we keep the most frequent emotion among them to represent the emotional relationship with the other citizen. At the community explanation stage, for each citizen, we associate it to the most frequent emotion among the emotions with other community members. This explanation is displayed in VISIR as a word cloud distribution, as shown in Figure 20.

If the attributes used for the explanation are aligned with a hierarchical structure, like a taxonomy or an ontology, for each pair of artefact attribute values used by the similarity functions during the computation of the distance matrix, we store a dictionary where the key is their common parent, and the value is a list of two dictionaries (one for each attribute) in which the key is the attribute and the value is the entity with that attribute (e.g., the artefact).

At the community explanation stage, we simplify the list of all these dictionaries to only include at most five with the largest number of associated entities belonging to the community. This information is displayed as drop-down lists in VISIR (Figure 21).



Figure 21. Common Iconclass category for a subset of the community artworks.

One of the most representative cases to exemplify this type of explanation is the communities discovered using similarity between artworks using the Iconclass taxonomy described in Section “Similarity between taxonomies”. Because it is a taxonomy with a great number of notations, we cannot explain it like we did with emotions. For example, it would not be able to identify close connections like 11G2 (“activities of angels (in heaven)”) and 11G3 (“angels fighting”), leading to poor explanations.

Because of this, it is required to store information about the common elemental themes of each pair of artworks while computing its similarity. In the previous case, this link translates to the following JSON object:

```
{"11G": [{"11G2": [artworkA], {"11G3": [artworkB]}]}]}
```

At the community explanation stage, the dictionaries encoding similarities between community members are simplified. For example, let us suppose there are two new artworks involved in the community: artworkC with 11G11 (“Seraphim”) and artworkD with 11G3 (“angels fighting”). Then, the final JSON object would be:

```
{"11G": [{"11G2": [artworkA], {"11G3": [artworkB, artworkD]}, {"11G11": [artworkC]}]}
```

In VISIR, this information is displayed as a drop-down list (Figure 21). A description is provided with the parent label, the number of artworks and the artwork’s children labels.

Angels /11G/ - Artworks 4. This identifier is the common parent of the following labels: “activities of angels (in heaven)” (artworkA) /11G2/, “angels fighting” (artworkB, artworkD) /11G3/, “Seraphim” (artworkC) /11G11/.

Explanations based on the citizen contribution attributes, such as emotions or sentiments, are especially important because they directly influence the distance matrix and, consequently, the communities discovered by the clustering algorithm. For this reason, we impose a restriction on the citizen contribution attributes to get interesting and understandable results for human users. While artefact attributes are still included in the explanation, they are not subjected to this constraint because the main clustering target are the citizens and the distance matrix provided to the algorithm is built around the citizen contribution attributes.

The explanation process also affects the community detection process described in Section “Community detection”. The results of a community detection process are considered *explainable* if the explanation process can explain its communities according to the citizen attributes used to compute the distance matrix required for clustering. We say that a community C is explainable if, for a given similarity threshold p specified by the perspective and a set of citizen similarity attributes A , at least p % of the community members are described by the same value in one or more of the similarity attributes a in A . For example, in the case of the evoked emotions described above, the community would be considered as explainable only if at least p % of the community members are associated with the same emotion. If a community is not explainable, the clustering output is considered invalid. Next, we apply the clustering algorithm again with an increased number of clusters as described in Section “Community detection”. This process continues until the number of clusters is equal to the number of citizens (in which case, all the citizens are classified as not belonging to any community) or an explainable output is yielded.

Community Similarity

Recommender system process partially depends on finding similar or opposite communities that a citizen belongs to. This way, the recommendation supports the reflection process inducing the citizen to think about her inter and intra community relationships with other citizens and their contributions.

We compute the similarity between two communities C_1 and C_2 as the similarity between their centroids, introduced in Section “Community explanations” and defined as non-existent model citizens with the highest mean similarity to other community members, according to the following formula:

$$SIM(c1, c2) = W_I * SIM_I(c1, c2) + W_E * SIM_E(c1, c2) \mid W_I, W_E \in [0, 1]; W_I + W_E = 1$$

where c_1 and c_2 are the community centroids; SIM_I is the similarity between the community centroids based on similarity (implicit) attributes; SIM_E is the similarity between the community centroids based on citizen (explicit) attributes; W_I and W_E indicate the influence of SIM_I and SIM_E in the final similarity between communities. The CM uses the default values $W_I = 0.7$ and $W_E = 0.3$.

On one hand, the similarity between implicit attributes is calculated as $SIM_I(c_1, c_2) = 1 - DIST_I(c_1, c_2)$, where $DIST_I(c_1, c_2)$ is the distance between c_1 and c_2 from the distance matrix used for clustering. On the other hand, we compute the similarity between explicit attributes using one of the following methods with preference to the first one whenever it is possible:

- Similarity for finite sets of sorted attributes (either by figurative categorical meaning or literal numerical value) –such as the political inclination of HECHT citizens from far left to far right (very left, left, center, right, very right). In this case, given a sorted list X of attribute values, we compute the similarity between them as follows:

$$Sim = 1 - \frac{|index_A - index_B|}{max(|X| - 1, 1)}$$

where $indexA$ and $indexB$ are the position of the two attribute values in the sorted list X starting from 0 and $|X|$ is the cardinality of X .

- Equal similarity: if two explicit attributes have exactly the same value, they have a similarity of 1. Otherwise, the similarity is 0.

Additionally, we implemented alternative versions of community similarity following common inter-cluster similarity measures:

- Single linkage: Cluster similarity is computed as the similarity between the closest pair of citizens in target clusters.
- Complete linkage: Cluster similarity is computed as the similarity between the farthest pair of citizens in target clusters.

- Average linkage: Cluster similarity is computed as the average similarity between the citizens in both clusters.

In these inter-cluster similarity measures, the similarity between a pair of citizens is computed with the formula we used for the centroids, replacing the values of SIM_I and SIM_E with the corresponding similarity values between the two citizens.

Application Scenarios

In this section, we describe how the Community Model has been applied to some SPICE case studies MNCN, HECHT, GAM and DMH. First, a statistical analysis of the data samples is documented in Appendix 2: Data samples and it is used to understand the data provided by the cases studies to decide how to configure promising perspectives. Then, we have created, in collaboration with museum curators from the case studies, several perspectives and analyzed the implicit communities discovered using the VISIR tool, briefly described in the next section. Subsequent sections summarize the most interesting results found during the interpretation of the implicit communities (again, using VISIR tool) for each case study.

Case Study Analysis using VISIR

Reflection processes are based on the fact that the same citizen can be classified in different implicit communities using different perspectives, where a perspective represents how *Artefact Attributes* and *Citizen Contribution Attributes* –using similarity functions and a clustering algorithm– are employed to group citizen into those implicit communities. For example, this information may be used to discover relationships between a citizen contribution attributes (e.g., emotions, sentiments, moral values) towards the same artefact or patterns on the emotions evoked by artefacts belonging to the same category (e.g., author, topic, material). To assist curators in the analysis of these communities and perspectives and, hence, to evaluate how the Community Model is working, we use VISIR (VISualization for Interpretation and Reflection) tool. The main interface is shown in Figure 22. A sample VISIR tool that shows some perspectives for GAM case study is available at <https://gjimenezucm.github.io/SPICE-VISIR/> and <https://spice.fdi.ucm.es/visir/>. The source code is distributed under Apache 2.0 License, and it is available at the repository <https://github.com/spice-h2020/VISIR>. VISIR implementation is described in detail in deliverable “D5.3: Integrated interfaces for citizen curation”.

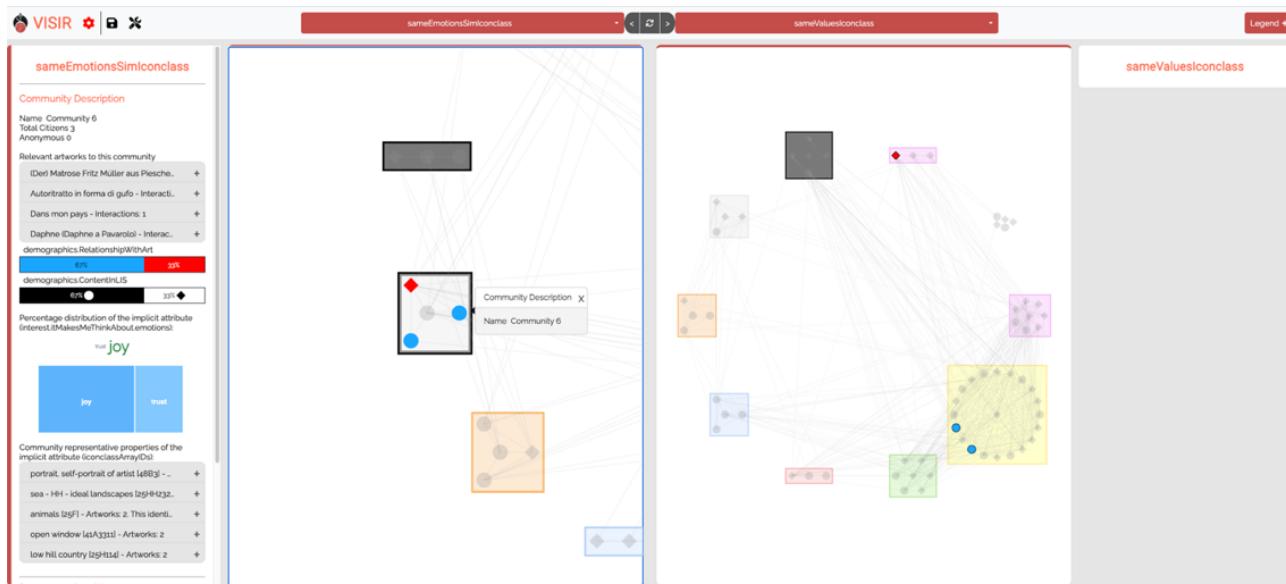


Figure 22. VISIR Interface

VISIR can show simultaneously two different perspectives with common explicit citizen attributes. Each perspective visualizes the communities computed by the Community Model. A community is represented by a rectangle (a bounding box) that encloses a set of nodes, which represent the citizens within the community. A group of nodes without a bounding box represent the citizens that do not belong to any community in this perspective.

Each citizen node is characterized by a colour and shape. Both visual dimensions are employed to identify up to two explicit citizen attributes. Nodes are linked by edges that represent the similarity between citizens according to the similarity functions defined by the perspective.

When the user clicks on a node –a citizen, the data panel displays the information for the selected citizen (see Figure 23), showing the citizen's explicit attributes (legend attributes) and all the citizen's contributions.

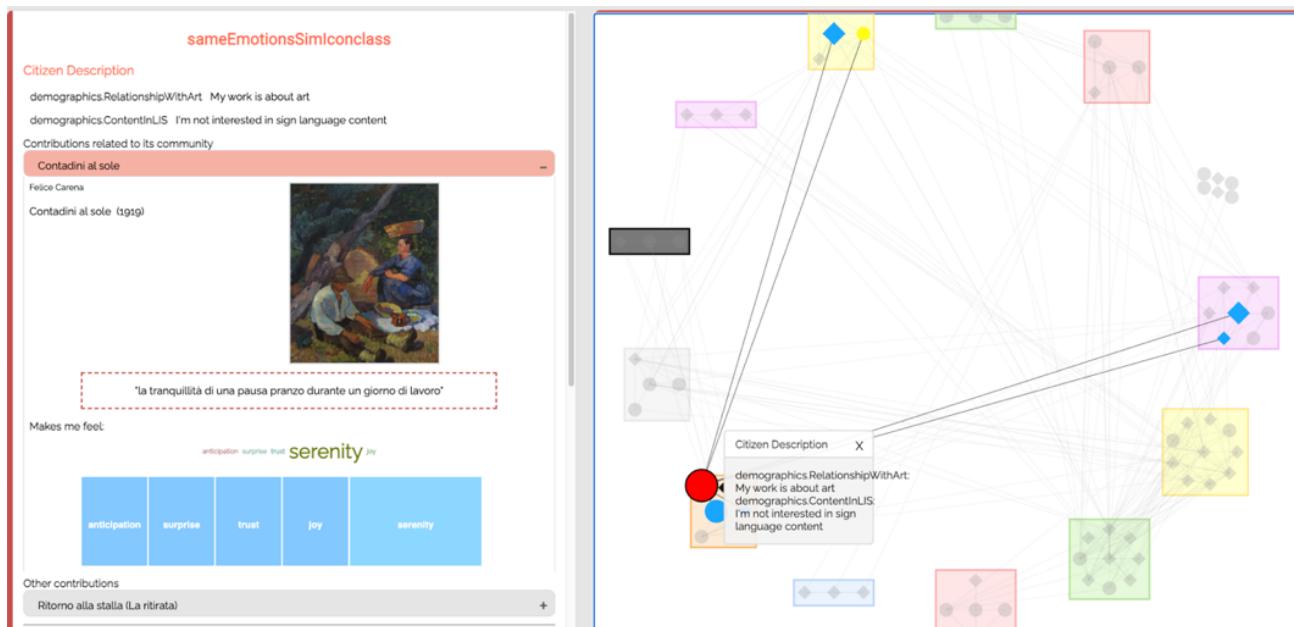


Figure 23. When a user is selected, the Citizen data panel is shown (on the left) and similar citizens are highlighted (on the right)

When the user clicks on a community, the data panel displays information about the community (see Figure 24), like the number of citizens, their distribution according to the explicit citizen attributes and several explanations created by the Community Model to explain the characteristics of the implicit community discovered.

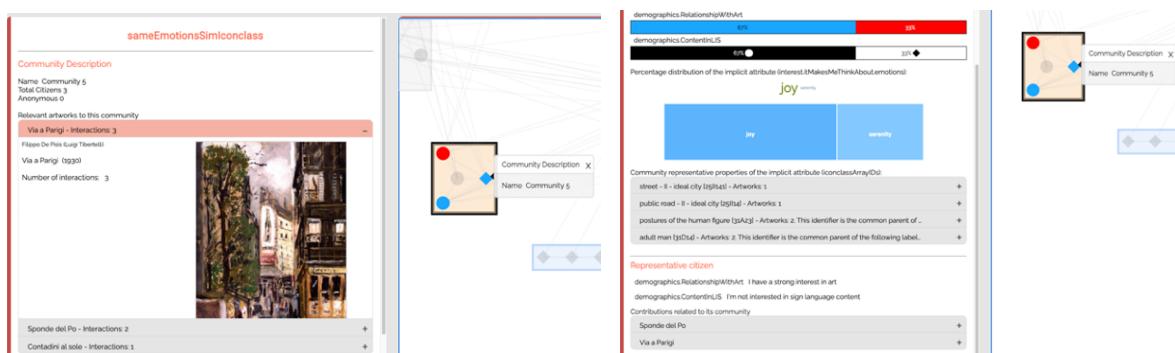


Figure 24. Information about a community: basic information (left) and explanations (right) about a community

VISIR provides filtering options for navigating the communities between perspectives and to focus on how different citizens are clustered according to different perspectives. To assist with the interpretation reflection loop, VISIR helps to find communities sharing a common trait:

- Legend items (Figure 25) are employed to find citizens who share values in explicit attributes.

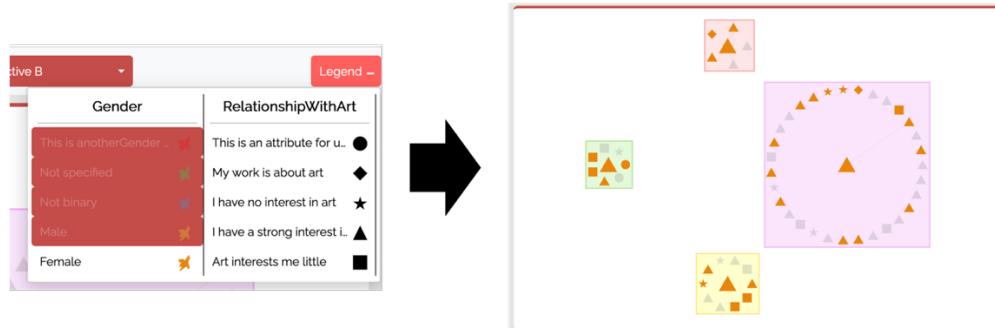


Figure 25. Legend item filtering features: If we click on legend items and leave active only female values (left), then only the citizens that belong to this explicit community (females) are highlighted (right).

- Clicking on word clouds (Figure 26) and label lists (Figure 27) employed to represent explanations based on implicit (similarity) attributes helps to find communities that share the selected attributes.

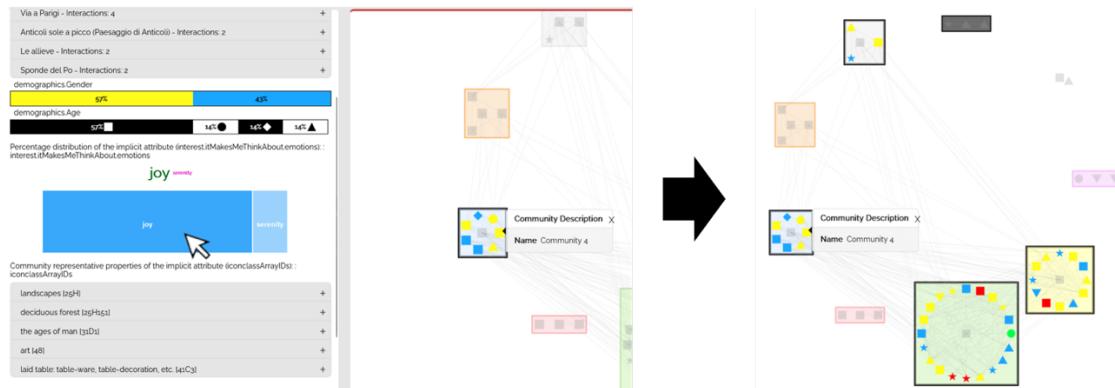


Figure 26. Word cloud filtering features: If we click on the “joy” word in the emotional explanation (left), VISIR identifies all the other communities whose predominant emotion is “joy” (right)

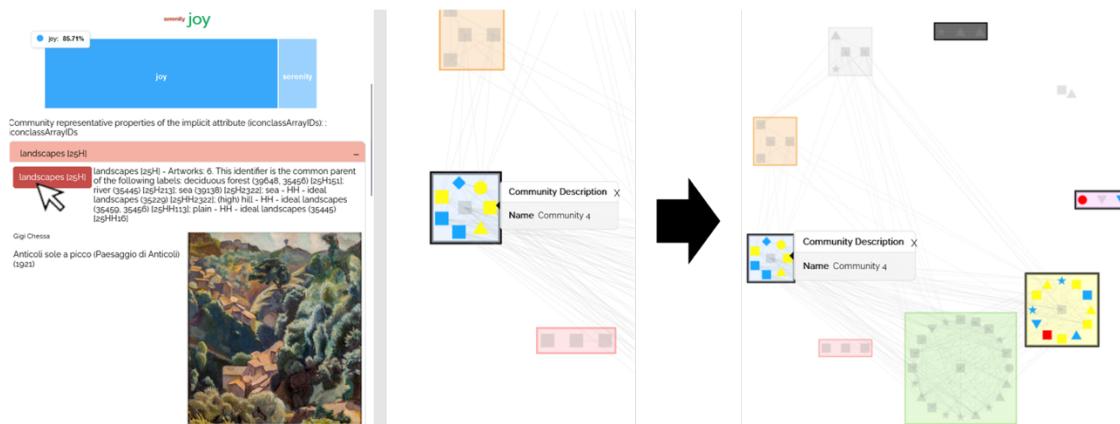
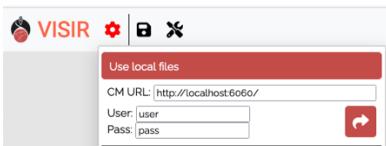


Figure 27. List item filtering features: If we click on the “landscapes” theme in the explanation based on Ikonclass concepts (left), VISIR identifies all the other communities with artworks falling under this category. (right)

VISIR has been used to exemplify the analysis of the case studies, described in the next sections. All visualization files are available at <https://gimenezucm.github.io/SPICE-VISIR/>. Additionally, VISIR can be configured to live interact with the community model for each case study:

1. Visit <https://gjimenezucm.github.io/SPICE-VISIR/> or <https://spice.fdi.ucm.es/visir/>
2. Click the cogwheel icon on the interface



3. Fill in the inputs with the data that precedes each case study in the next sections.

GAM (Torino)

CM URL	https://spice.fdi.ucm.es/gam/
User	userGAM
Password	passGAM

Perspectives

Although the community model could find similar artworks based on several properties, such as author, year, artist country... we found that the best results are obtained with the Iconclass ontology. We also include the materials as a second example since most citizens do not have the required art knowledge to be influenced by other properties. While it could be meaningful to compare the reactions of citizens with a strong interest in art vs the ones that do not, the provided sample is not suitable for it because most artworks have a different value for some properties (e.g., author) while they are dominated by one attribute value in the others (e.g., most artworks originate from Italy).

On the other hand, because the contribution attributes (emotions, sentiments, and moral values) are heavily dominated by one value, the insights offered about them by the discovered communities are limited.

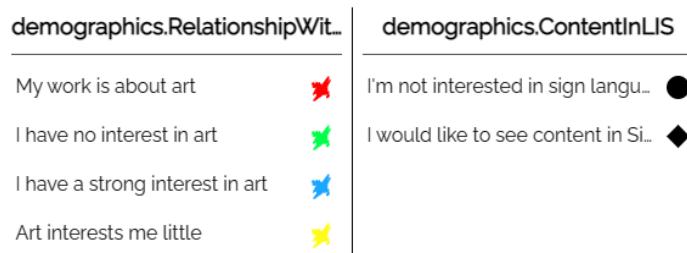


Figure 28. Colors and shapes employed in VISIR for GAM example

The symbols and colours used in the subsequent visualizations are illustrated in Figure 29.

Similar emotions in similar artworks based on their Iconclass theme (Figure 29).

It is reasonable to assume that our emotional perception is heavily influenced by an artwork's theme, which is represented by the Iconclass ontology. With this perspective, we aim to identify what reactions each theme is likely to evoke in most of the citizens. For example, do citizens feel positive emotions while interacting with romantic artworks? Or is it possible that citizens are reminded of past heartbreak experiences? What about artworks focused on humanity? Another possible viewpoint could be how much the theme influences the citizen vs their personal life perspective. Is the citizen mainly feeling positive emotions because of the theme or does that person hold positive emotions for most themes in general? How much does a citizen's knowledge about the artwork (or art in general) prejudice their opinion about it?

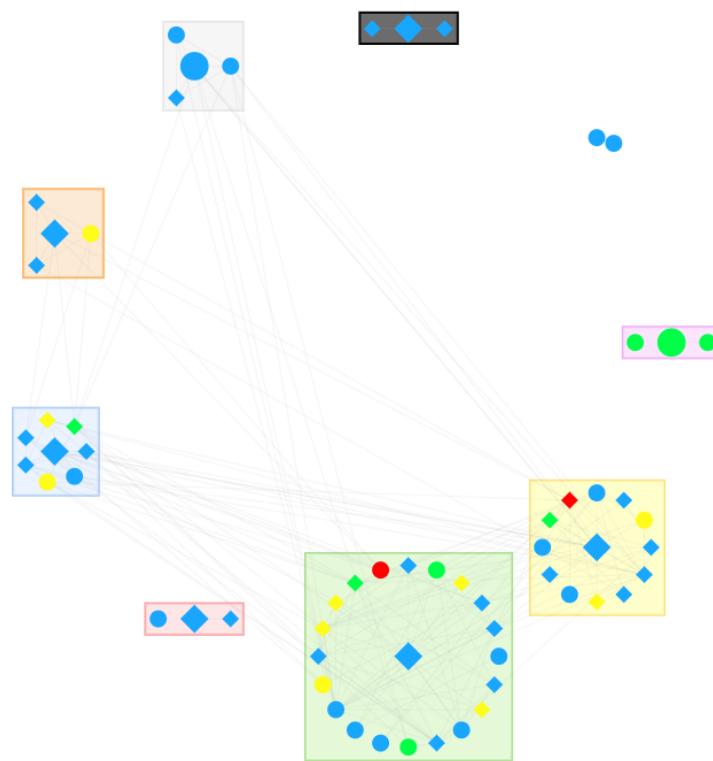


Figure 29. Similar emotions in similar artworks based on their Iconclass theme.

Name Community 0

Total Citizens 2

Anonymous 0

Relevant artworks to this community

Daphne (Daphne a Pavarolo) - Inter... +

Dans mon pays - Interactions: 1 +

demographics.RelationshipWithArt

100%

demographics.ContentInLIS

100% ●

Percentage distribution of the implicit attribute
(interest.itMakesMeThinkAbout.emotions):

fear sadness

sadness fear

Figure 30. Community 0 description.

Percentage distribution of the implicit attribute
(interest.itMakesMeThinkAbout.emotions):

anticipation anger



Community representative properties of the implicit attribute (iconclassArrayIDs):

colours, pigments, and paints: blue... -

colours, pigments, and paints: blue
[22C4(BLUE)] - Artworks: 2

Hans Hartung

Composition T. 50 - 5
(1950)



Figure 31. Community 3 description.

Percentage distribution of the implicit attribute
(interest.itMakesMeThinkAbout.emotions):

sadness



Community representative properties of the implicit attribute (iconclassArrayIDs):

the ages of man [31D1] - Artworks: ... +

adult woman [31D15] - Artworks: 2 +

the (nude) human figure; 'Corpo hu... +

Figure 32. Community 5 description.

As most contributions are represented by “joy”, this example focuses on other emotions to get more relevant results. From community 0 (purple), we can see that “Dans mon pays” and “Daphne” are interpreted as negative by citizens. Since a negative reaction is rare in this sample, it could be interesting to study these artworks thoroughly. We describe “Daphne” as an example below. From the community 3 (red), we notice that abstract techniques evoke more negative emotions, such as anger. From community 5 (orange), we may infer that artworks about the “ages of man” relate to sadness. One possible theory could be that it reminds us of our mortality. From community

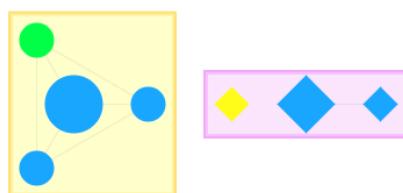


Figure 33. Similar emotions evoked by Daphne.

Name Community 0
Total Citizens 2
Anonymous 0

Relevant artworks to this community

Daphne (Daphne a Pavarolo) - Inter... +

demographics.RelationshipWithArt



Percentage distribution of the implicit attribute (interest.itMakesMeThinkAbout.emotions):

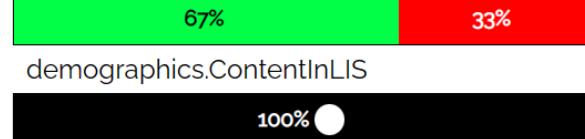
fear joy

Name Community 1
Total Citizens 3
Anonymous 0

Relevant artworks to this community

Daphne (Daphne a Pavarolo) - Inter... +

demographics.RelationshipWithArt



Percentage distribution of the implicit attribute (interest.itMakesMeThinkAbout.emotions):

sadness

Figure 34. Similar emotions evoked by Daphne – community descriptions.

In Figure 34 the left community is represented by “joy” and “fear” while the right community is represented by “sadness”. Consequently, although there is an outlier citizen which felt joy, we can predict most citizens will find this picture to be gloomy.

Same emotions in similar artworks based on their Iconclass theme (Figure 35).

While the previous perspective may offer a good overview of the emotion-theme relationship, a museum curator may be interested in exploring the relationship between a specific emotion and art topics. This perspective imposes a stronger constraint in the clustering, separating the citizens into communities almost dominated by one emotion.

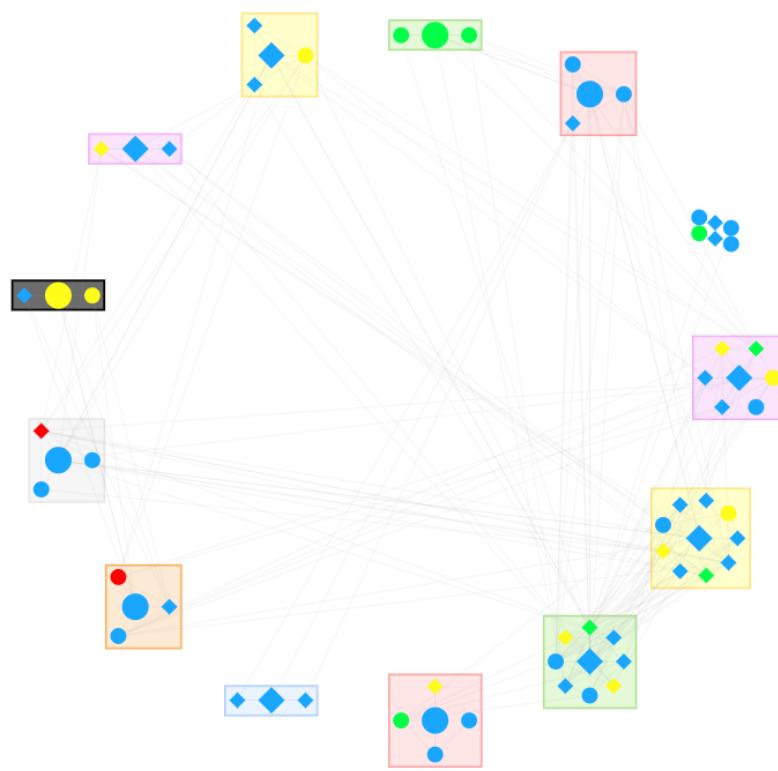


Figure 35. Same emotions in similar artworks (Iconclass)

Similar emotions in similar artworks based on their materials (Figure 36).

Another attribute that is easily identified by citizens is the material. As such, a museum curator may be interested in finding out how much it affects emotional perceptions. Figure 36 illustrates the communities for citizens with similar emotions in artworks of similar materials, according to the ontology introduced in Figure 11.

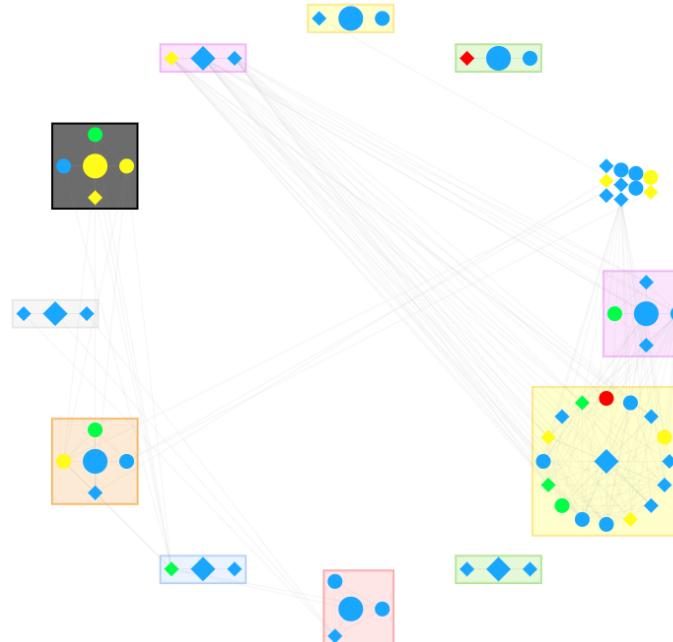


Figure 36. Similar emotions in similar artworks (materials)

Same emotions in similar artworks based on their materials (Figure 37).

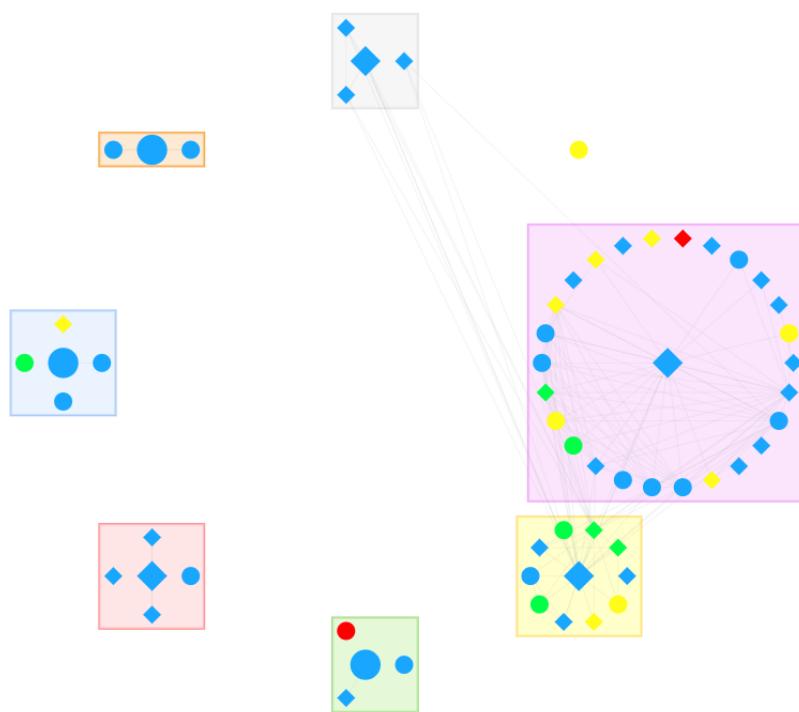


Figure 37. Same emotions in similar artworks (materials)

Visualization and insights

As we have described in previous sections, museum curators will define perspectives and visualizations in VISIR to help the interpretation reflection loop. Visualization helps finding interesting insights through the communities detected based on interactions and similarity. We have defined Scripts to help case studies to replicate the same interpretation reflection processes with other perspectives. We use the idea of the Script theory which posits that human behaviour largely falls into patterns called "scripts" because they function analogously to the way a written script does, by providing a program for action.

SCRIPT 1: GAM example

Objectives:

- Find out about emotions evoked by a certain artwork.
- Studying similar artworks, search for a correlation between the theme and the emotions it evokes. Is there a correlation between the communities and the citizen explicit attributes?
- Is the emotion influenced by the personal experiences of the citizen? Choose a citizen and study their emotions in other artworks, especially ones with opposite themes.

Steps:

For this example, we chose “relationship with art” and “content in LIS” as the explicit attributes, as depicted in Figure 38. We chose the first to analyse the influence of art knowledge into the emotions we experience while interacting with artefacts and the second because sign language and deaf people are one of the core issues of GAM.

demographics.RelationshipWithArt	demographics.ContentInLIS
My work is about art	●
I have no interest in art	◆
I have a strong interest in art	◆
Art interests me little	◆

Figure 38. Legend of the demographic attributes (GAM)

4. Choose one artwork: “La pittrice”.

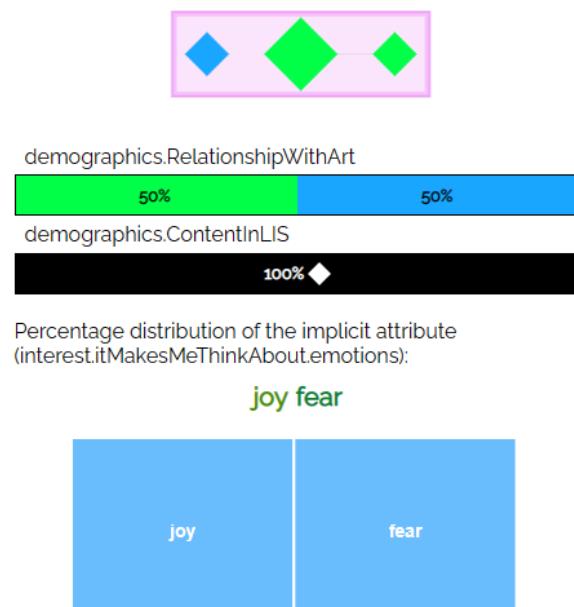


Figure 39. Community visualization and attribute distribution – similar emotions in “La pittrice”

As illustrated in Figure 39 we can see that there is a difference between the emotions depending on their art interest: the citizen with a strong interest in art feels fear while the citizen with no interest feels joy. This may be caused by knowledge about the artwork. Nevertheless, the sample is too small to draw meaningful conclusions. Therefore, we continue with step 2.

5. Compare with one or more similar artworks. For example, “Le allieve” and “Anticoli sole a picco (Paesaggio di Anticoli) (1921)” are similar artworks with a common theme: deciduous forest (25H151).

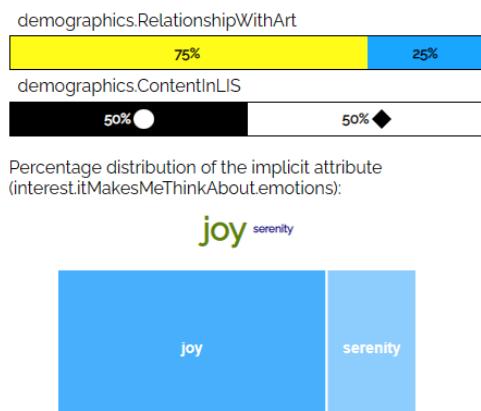


Figure 40. Community attributes distribution – similar emotions in “Le allieve”

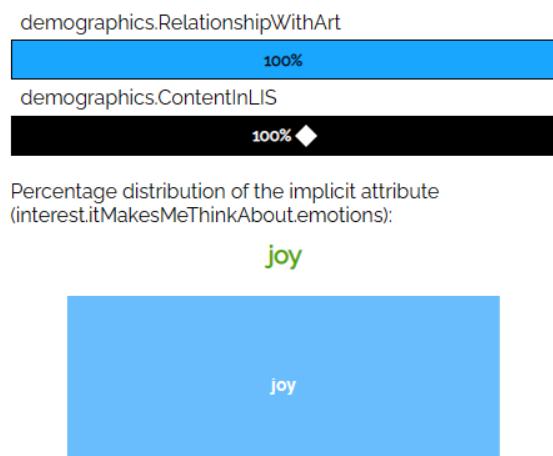


Figure 41. Community attributes distribution – similar emotions in “Anticoli sole a picco (Paesaggio di Anticoli) (1921)”

Although there are no common citizens who interact with the three artworks and the sample is small, we can still draw some conclusions from the demographic groups and the emotion distribution. As shown in Figure 40 and Figure 41, “Le allieve” and “Anticollì” induce joyful reactions. These artworks depict forest themes as illustrated in Figure 44. Furthermore, if we study the only outlier, we can see that, while the dominant emotion is fear, there is still a minor part of interest which is shared by some citizens in the three artworks.

- Choose a citizen from the first perspective and analyse the emotional response to other artworks.

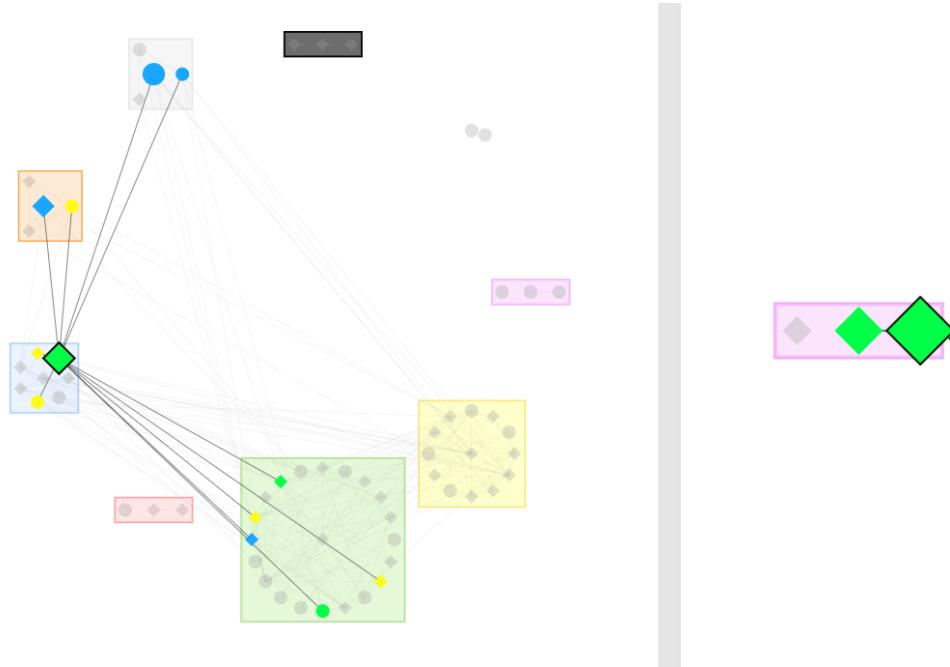


Figure 42. Similar emotions in similar artworks VS similar emotions in “La pittrice”

First, we display the first perspective, similar emotions in “La pittrice”, and a second perspective which considers all the artworks the citizen interacted with. The result is shown in Figure 42. If we select a citizen of the first perspective, we can identify its community in the second perspective. Then, we can start the analysis of the citizen’s emotional responses by examining the emotion distribution of that community.

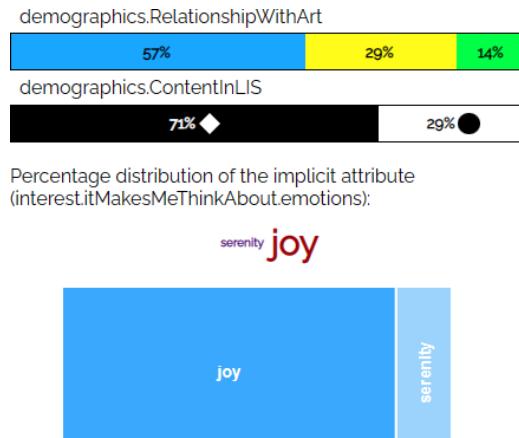


Figure 43. Community description.

As we can see in Figure 43, joy is the representative emotion of the community. This is consistent with the previous hypothesis.

Next, we check the community themes.

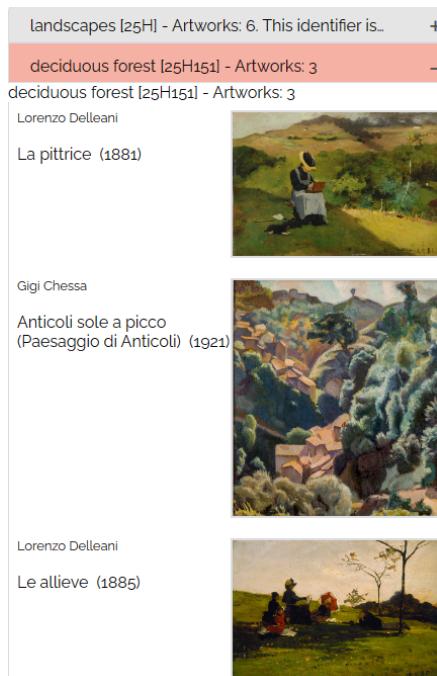


Figure 44. Themes shared by artworks interacted by the community members.

As illustrated in Figure 44, this community is mainly represented by artworks depicting landscapes. These results support a relationship between the artwork and the theme.

On the other hand, is it possible that the citizen feels that way because of personal circumstances and not because of the theme? To check this out, we analyse other citizen contributions.



Figure 45. Two artworks the selected citizen interacted with and the experienced emotions.

As illustrated in Figure 45, the citizen shows different reactions to different topics. Based on this analysis, a museum curator may conclude that the theme has a meaningful impact in the emotions we feel.

MNCN (Madrid)

CM URL	https://spice.fdi.ucm.es/mncn/
User	userMNCN
Password	passMNCN

Perspectives

The three perspectives below are built about the citizen stances regarding the three main points of discussion: water and electricity, food, and consumption. As described in Appendix 2: Data samples, citizens are categorized into five groups: very worried, worried, neutral, unworried, and very worried.

Similar stances about water and electricity (Figure 46).

Displays citizen communities with similar behaviors and opinions on misusing water and electricity. We found that only one citizen is not worried about it.

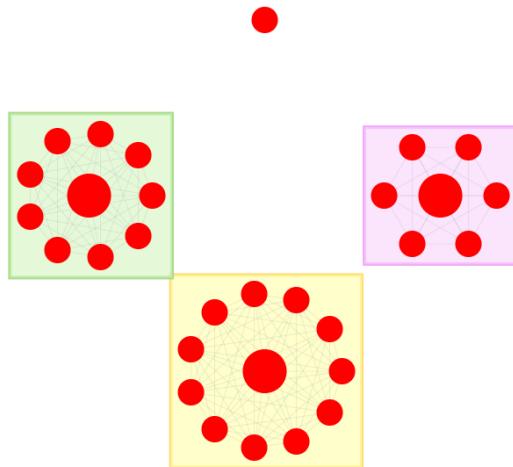


Figure 46. Similar stances about water and electricity

Similar stances about food (Figure 47).

Citizen communities with similar attitudes towards food issues: availability, consumption, waste... The analysis of this perspective reveals that the students are much less concerned about it than they are about water.

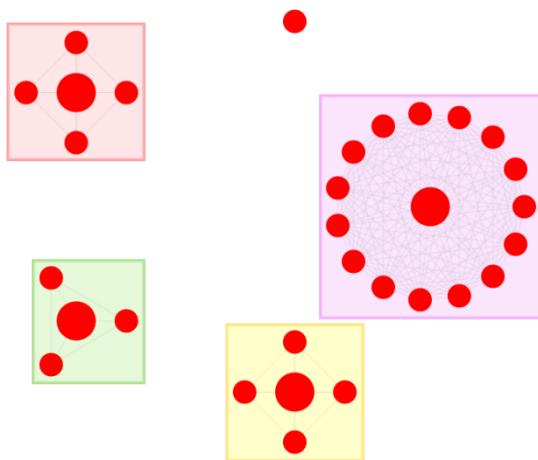


Figure 47. Similar stances about food

Similar stances about consumption (Figure 48).

Groups citizens with similar thoughts about nonspecific modern consumption. While it is not as strong as with water, the results show that the citizens feel negatively about the current consumption problems in society.

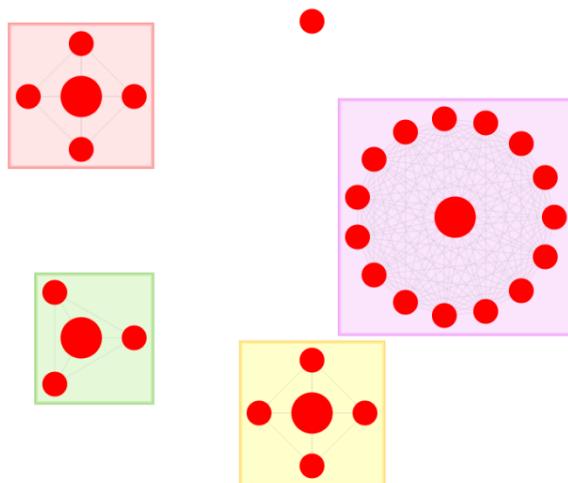


Figure 48. Similar stances about consumption

Visualization and insights

SCRIPT 1: MNCN example

Objectives:

- Find interesting combinations. For example, are there citizens that are worried about water but do not care about wasting food?
- Can we find a correlation between the citizen demographic and the sentiments?

Steps:

1. Choose two perspectives grouping the citizens according to their similar stances towards a societal issue. For example, similar thoughts and actions on the consumption of water and food.
2. Select a community that is worried about water as shown in Figure 49. Explore the citizen counterparts in the food perspective for one or more citizens that are not worried about food.

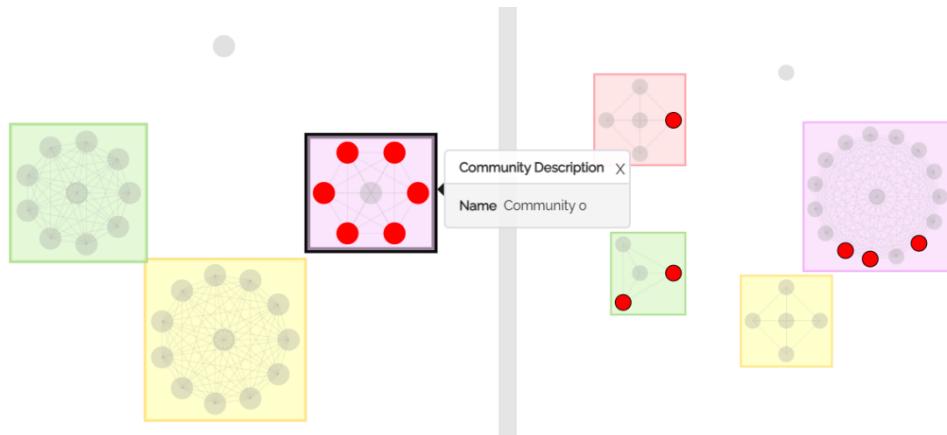


Figure 49. Similar stances about water and electricity & similar stances about food.

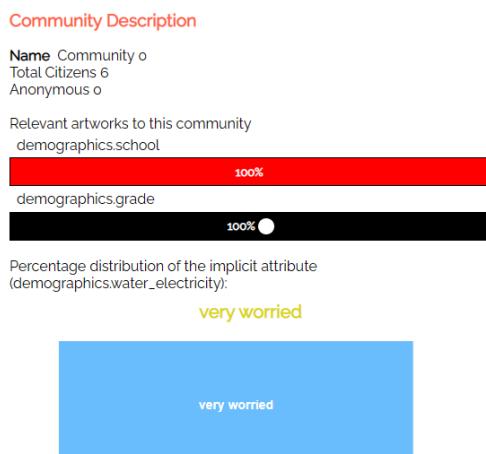


Figure 50 Community description (similar stances about water and electricity)

If we examine the communities in the food perspective (Figure 51), we can see that while most citizens fall under the “worried” or “neutral” category, there is one citizen in the red community that is “unworried” about food problems.



Figure 51 Community description (similar stances about food)

DMH (Helsinki)

CM URL	https://spice.fdi.ucm.es/dmh/
User	userDMH
Password	passDMH

Perspectives

Similar emotions in similar artworks based on object groups (Figure 52)

Groups of users who feel similar emotions while interacting with artefacts of similar categories: furniture, cups, clothes, and others. As there is not a suitable similarity relationship between the categories, this perspective imposes an equal restriction between them. That means, objects belonging to different categories are considered completely different.

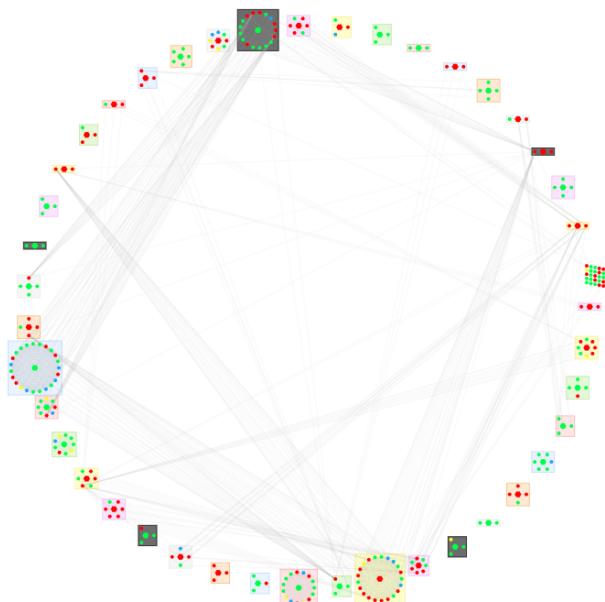


Figure 52. Similar emotions in similar artworks (object group)

Similar emotions in similar artworks based on materials (Figure 53)

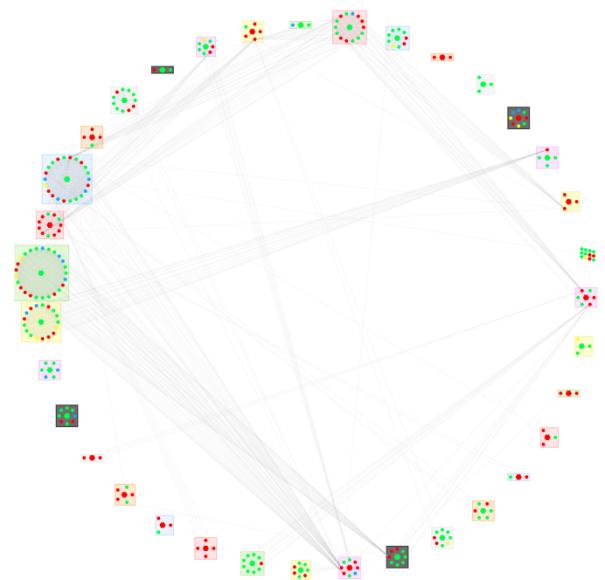


Figure 53. Similar emotions in similar artworks (materials)

Figure 53 shows citizen communities with similar emotions in artworks made with similar materials. In this case, materials are interconnected according to a predefined materials ontology, which allows us to group artworks belonging to derivatives of a material (as depicted in Figure 54) and, consequently, find more citizen relations that with a strict equivalence between artwork materials.

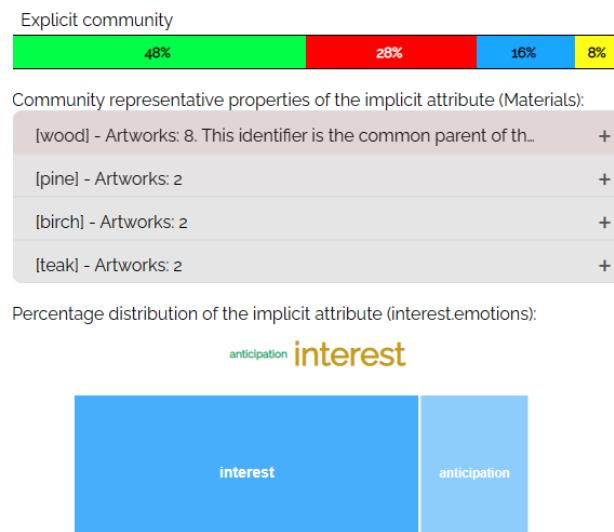


Figure 54. Community explanations. Pine, birch and teak are grouped under the wood category.

At the same time, it facilitates the analysis of a potential connection between the artwork material and the emotion it evokes. For example, glass is a material that could be linked with negative emotions, such as fear, because of its dangerous nature or with positive emotions, such as joy and interest, because of its visual attractiveness. A quick examination of the communities (Figure 55 and Figure 56) confirms the hypothesis.

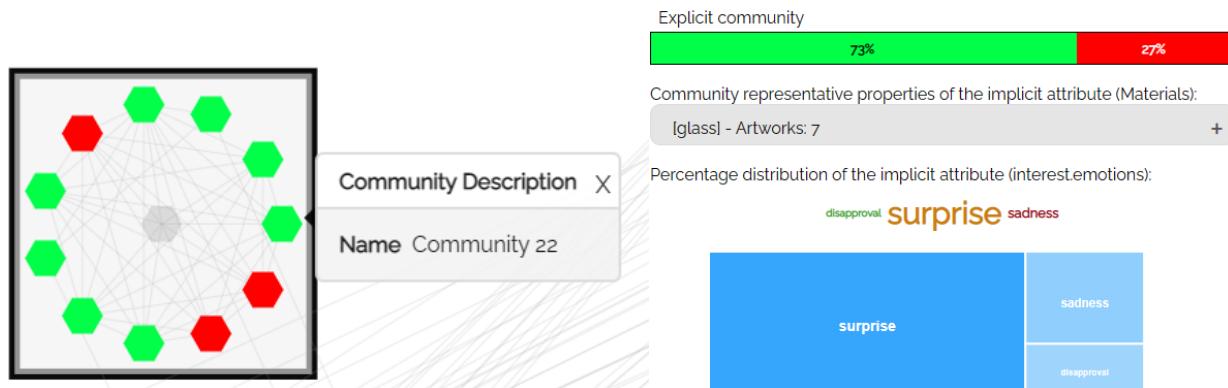


Figure 55. Community represented by surprise in glass artworks.

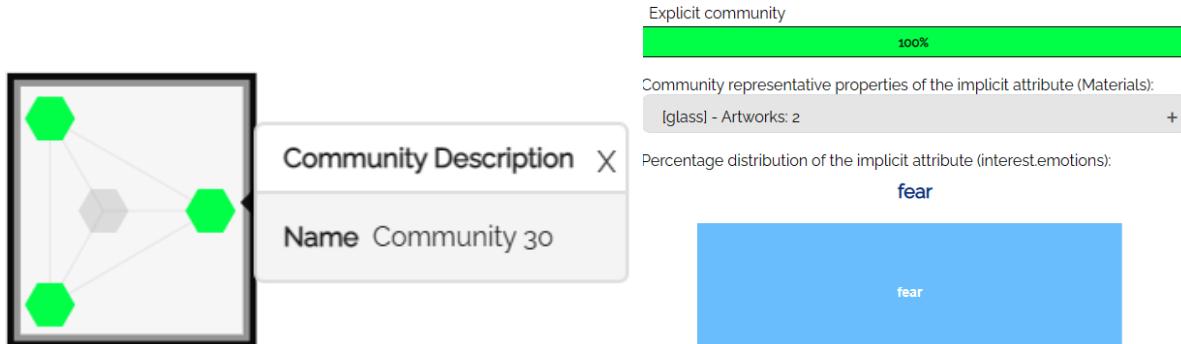


Figure 56. Community represented by fear in glass artworks.

Visualization and insights

Below we show an example of Script using the DMH case study.

SCRIPT 1: DMH example

Objectives:

- Find out a controversial artwork evoking opposite sentiments.
- Can we find a correlation between the citizen demographic and the sentiments?

Steps:

1. Choose one artefact: Fiskars scissors.
2. Find communities with similar sentiments about the artefact (Figure 57).

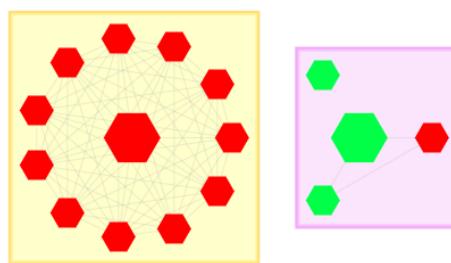


Figure 57. Similar sentiments about Fiskars scissors.

3. Can we find a correlation between the citizen demographic and the sentiments?

Although the sample is a little small and there is an outlier, it seems like senior citizens and the general audience have different views about this artefact: positive and neutral, respectively.

SCRIPT 2: DMH example

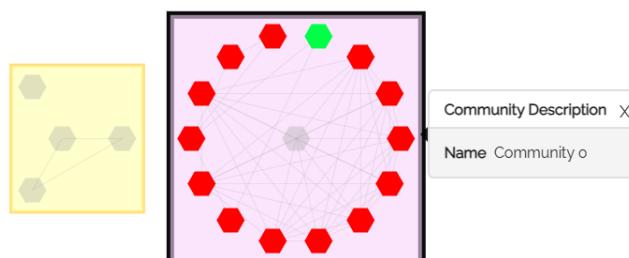
Objectives:

- Find out about emotions evoked by a certain design object
- Can we find homogeneity in the community of people sharing the same (similar) emotions?
- Are these emotions due to the specific object or do other similar objects share similar emotions?

Steps:

1. Choose one artwork: [Pastille-Chair](#)
2. Find the community of people with similar emotions when interacting with this object.

2.1 Let us focus on one emotion. As shown in Figure 58, we can explore the different groups (click one by one in the bounding boxes), observe the explanation panel, and choose one community of our interest.



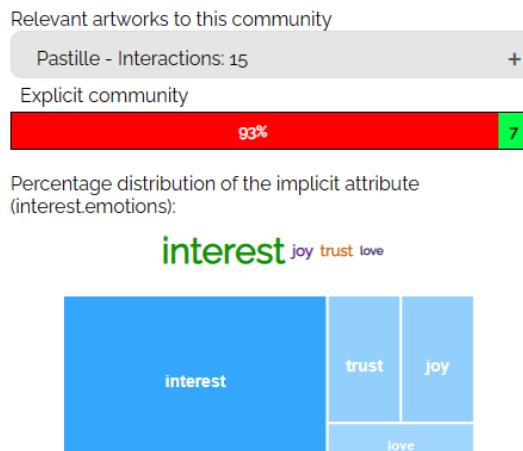


Figure 58. Community explanations - Similar emotions towards Pastille.

2.2. Let us observe how people from different explicit communities distribute in this group. Use the legend to learn about colours and shapes. In this example 93% are senior people. Looks like a very homogeneous community.

Finding: "Senior people feel interest about pastille-chair"

But maybe they also feel other emotions. Let us find out.

3. Apply a filter (using the legend) clicking the color for the explicit group. Yes.

Finding: "We have found that senior people also feel **joy** and **trust** regarding this object"

4. Is this something about this specific object? or are these emotions common for other similar objects?
For example, all the chairs, all the furnitures..

4.1. We need to open two perspectives, one for the object, and another one for the category and compare. [Furniture is the category of pastille](#).

4.2. Click the citizens in one side and search them in the other perspective. You can click a group and find out where all these individuals are in the other side.

1 Interest → 2 goes to Joy

Finding: "Senior people feel interested and joy for furniture in general, not only pastille chair"

5. We can repeat step 4 with other types of similarity between objects:

Is this something about the object or is it common about the colour? Is this something about the object or is it common about the designer? Is this something about the object or is it common about the manufacturer?

HECHT (Haifa)

CM URL	https://spice.fdi.ucm.es/hecht/
User	userHECHT
Password	passHECHT

In this case study, citizens are grouped according to their beliefs and feelings about three Roman Rebellion themes: if they support the Roman Rebellion and why, if Joseph Flavius was a traitor or if they think the museum exhibition is biased towards supporting the Roman Rebellion. Citizens provide demographic data such as gender, religiosity, politics, and their thoughts about historical relevance before and after the exhibition. Some interesting perspectives could be the relationship between Roman Rebellion views and religiosity or politics level as well as the differences between their thoughts about historical relevance before and after the exhibition.

Perspectives

Similar emotions and beliefs about the Roman Rebellion (Figure 59).

Citizens are grouped according to their emotions and beliefs regarding the Roman Rebellion. Citizens without a community are identified by not being contained into a square. Most of these citizens didn't provide a valid emotion or belief regarding Roman Rebellion, which cause them to not fall under any group.

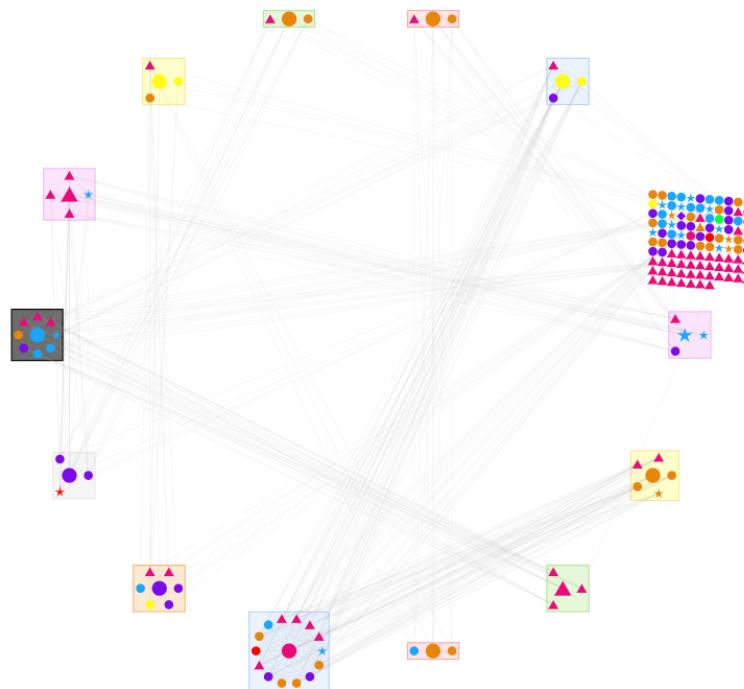


Figure 59. Similar emotions and beliefs about the Roman Rebellion.

Similar sentiments and beliefs about Joseph Flavius (Figure 60).

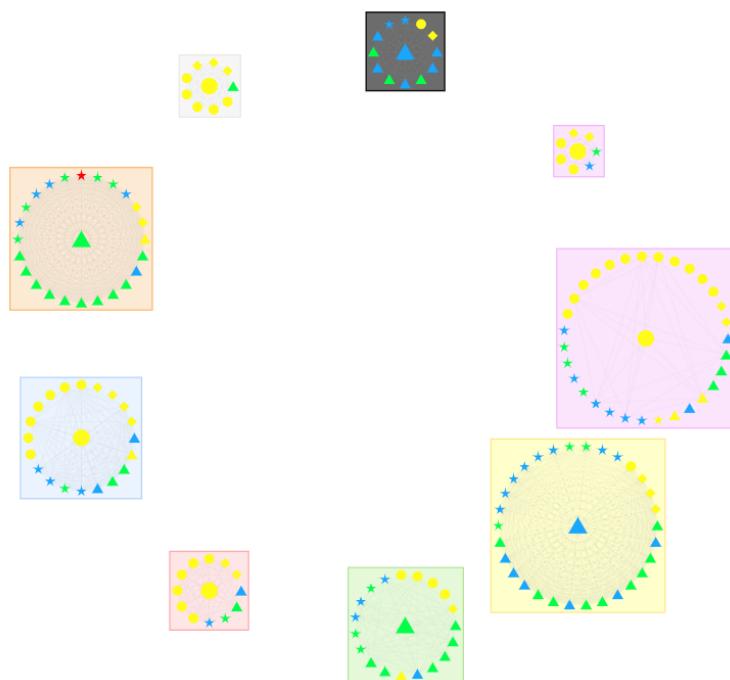


Figure 60. Similar sentiments and beliefs about Joseph Flavius.

Visualization and insights

Since HECHT collects several explicit information from the citizens, museum curators could infer interesting reflections from the study of explicit-implicit relationships. Below we describe a possible guideline to draw conclusions about the relationships between the citizen beliefs and the explicit groups they belong to.

SCRIPT 1: HECHT example

Objectives:

- Discover correlations between a citizen's religious and political views and that person's beliefs regarding the Roman Rebellion.

Steps:

For this example, we chose "politics" and "religious group" as the explicit attributes since they are two of the most important features that shape our opinions. The attribute values are shown in Figure 61.

Political demography	Religious demography
Very right	✖
Very left	✖
Right	✖
Left	✖
Dont know	✖
Center	✖
(empty)	✖

Figure 61. Legend – Demographic attributes used in this script.

1. Choose one belief and the closest emotion regarding the topic. For example, Roman Rebellion and the emotion it evokes as shown in Figure 59.
2. Examine the distribution of explicit attributes for each community.

1. Filter by politics

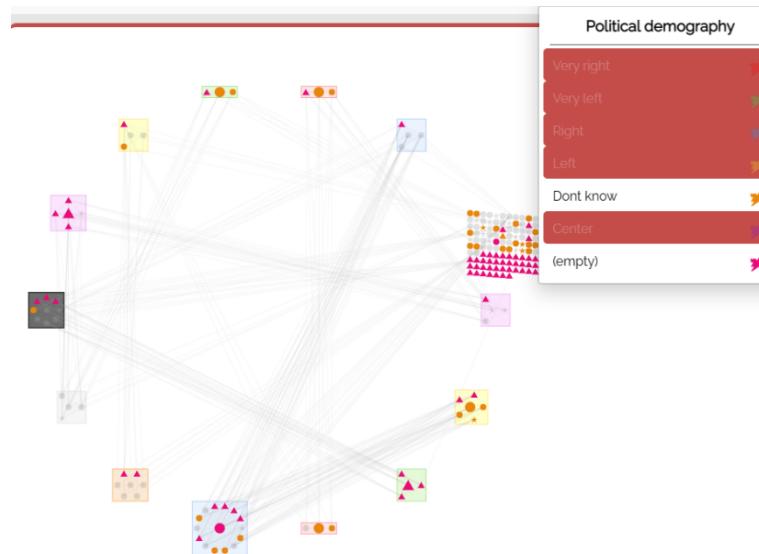


Figure 62. Similar emotions and beliefs about the Roman Rebellion. Filter citizens who didn't provide information.

2. Filter by religiosity

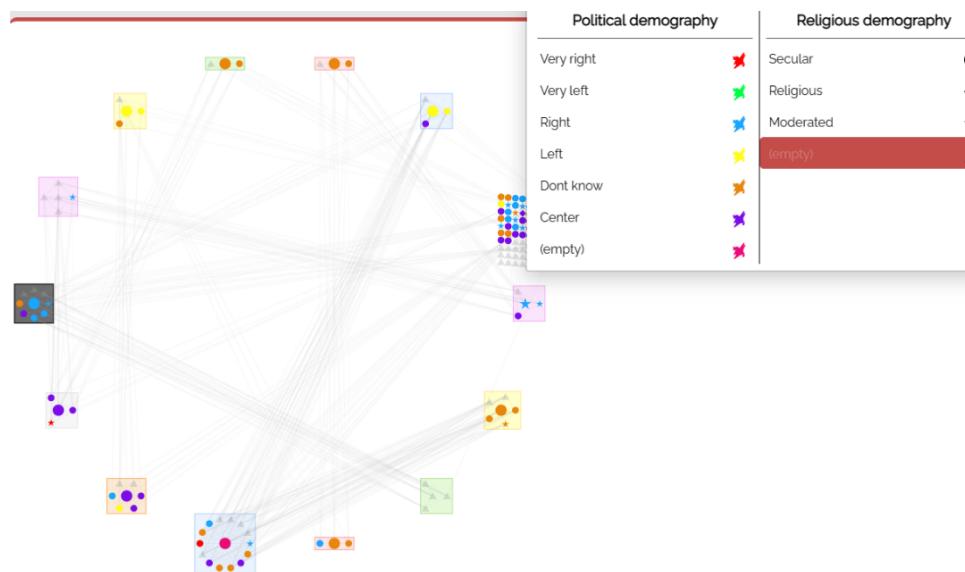


Figure 63. Similar emotions and beliefs about the Roman Rebellion. Filter citizens who indicated their religious group.

In the first case (Figure 62), we cannot draw meaningful conclusions due to the fact that most people did not provide their political affiliation. In the second case (Figure 63), there is a little information about the religiosity level. However, after examining the community description, it is learnt that the emotions and beliefs about the Roman Rebellion are not influenced by it in this sample. In conclusion, this sample does not provide evidence of any correlation between citizen political and religious groups and his or her stances regarding the Roman Rebellion.

Since the chosen explicit attributes (politics and religious groups) did not provide any relevant conclusions, we repeated the process with another perspective and different explicit attributes: gender and school (Figure 64).

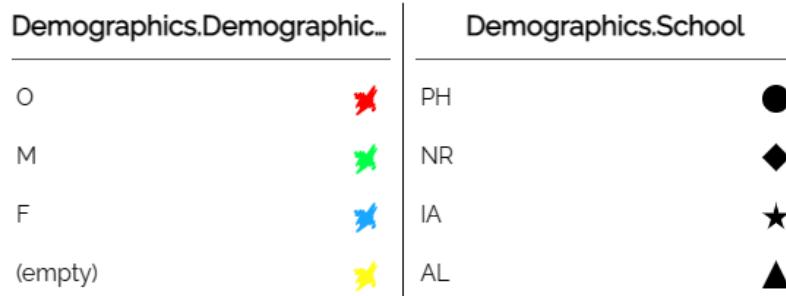


Figure 64. Legend – Demographic attributes used in this script.

1. Choose one belief and the closest sentiment regarding the topic: sentiments about Joseph Flavius.
2. Examine the distribution of explicit attributes for each community.
 1. Filter by gender.

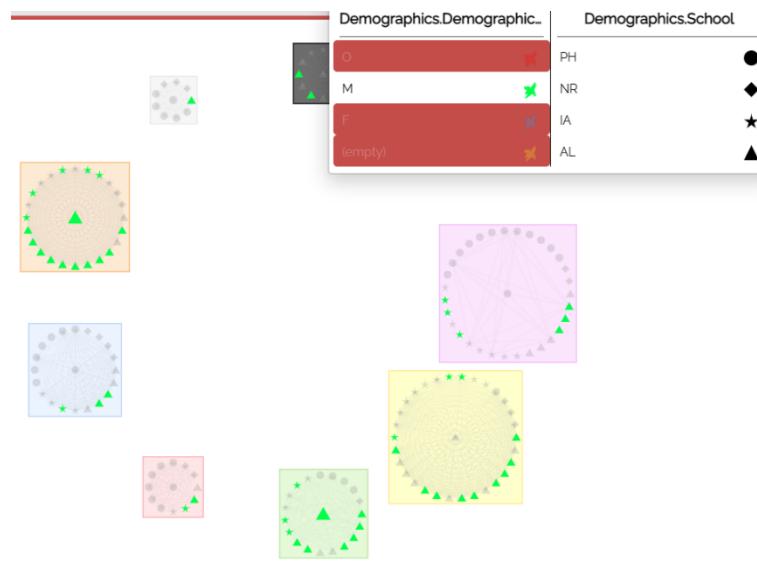


Figure 65. Similar sentiments and beliefs about Joseph Flavius. Filter male citizens.

2. Filter by school.

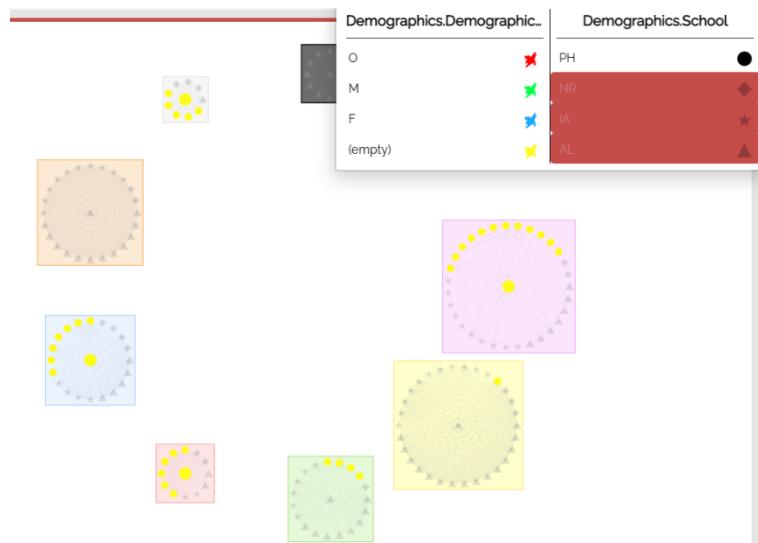


Figure 66. Similar sentiments and beliefs about Joseph Flavius. Filter PH school

In the first case (Figure 65), we found that communities dominated by male citizens think that Joseph Flavius is not a traitor while communities that are mostly female think that Joseph Flavius is a traitor. On the other hand, balanced communities with an equal number of males and females dominated by people that did not provide their gender, remain neutral in sentiment and beliefs. From the point of view of sentiments, it is not possible to draw conclusions because there are communities with different beliefs about Joseph Flavius but the same sentiment.

In the second case (Figure 66), we found that almost all students that did not provide their gender belong to the first two schools: PH and NR. The other two do not have a preference for a concrete belief or sentiment.

Conclusions

In this document we introduced the Community Model, a subsystem of the SPICE project with the aim of detecting explainable citizen communities using their contributions as input and providing this information

to other systems through a REST API. We described the model architecture and explained the workflow: initial configuration, perspective creation and community detection.

We reviewed the clustering techniques and similarity measures compatible with the Community Model and recounted some of their advantages and disadvantages so that museum curators can decide on a suitable algorithm and similarity measure for their needs. We also gave a detailed description of our approach for generating and explaining communities based on citizen similarity features and explicit attributes, as well as how we determine community similarity, which can be used by recommender systems.

We tested the feasibility of our methods with different case studies (GAM, MNCN, HECHT and DMH): we analyzed the data, selected interesting perspectives, visualized the communities in VISIR and interpreted the results. Our objective is to find communities whose description may be used to reflect on the citizens, confirming and denying preconceptions about entities (e.g., citizens and artworks) as well as discovering new relationships between them, such as popular artwork interpretations by citizens that are not contemplated by the author, art experts or museum curators. VISIR assists museum curators in this endeavor: facilitating perspective creation, displaying communities in a simple and approachable layout; providing functionality to visualize and compare two perspectives; and presenting citizen and community descriptions so they can be used by humans to extract information and reflect on them to draw interesting conclusions about citizens and communities. Museum curators can use this information to guide citizens towards artefacts of interest to them or introduce citizens belonging to an explicit group to sympathetic views from other explicit communities, encouraging interactions between the groups.

References

- [1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vander-plas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* 12 (2011) 2825–2830.
- [2] Hae-Sang Park and Chi-Hyuck Jun. 2009. A simple and fast algorithm for K-medoids clustering. *Expert Syst. Appl.* 36, 2 (March 2009), 3336–3341. DOI:<https://doi.org/10.1016/j.eswa.2008.01.039>
- [3] sklearn.extra.cluster.kmedoids, https://scikit-learn-extra.readthedocs.io/en/stable/generated/sklearn_extra.cluster.KMedoids.html, accessed: 2023-03-30.
- [4] sklearn.cluster.agglomerativeclustering, <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html>, accessed: 2023-03-30.
- [5] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1996. BIRCH: an efficient data clustering method for very large databases. *ACM SIGMOD Rec.* 25, 2 (1996), 103–114. DOI:<https://doi.org/10.1145/235968.233324>
- [6] sklearn.cluster.birch, <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.Birch.html>, accessed: 2023-03-30.
- [7] M. Ester, H. P. Kriegel, J. Sander, and Xu Xiaowei. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. (December 1996). Retrieved January 14, 2022 from <https://www.osti.gov/biblio/421283-density-based-algorithm-discovering-clusters-large-spatial-databases-noise>
- [8] sklearn.cluster.dbscan, <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html>, accessed: 2023-03-30.
- [9] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. OPTICS: ordering points to identify the clustering structure. *ACM SIGMOD Rec.* 28, 2 (June 1999), 49–60. DOI:<https://doi.org/10.1145/304181.304187>
- [10] sklearn.cluster.optics, <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.OPTICS.html>, accessed: 2023-03-30.

- [11] Santo Fortunato. 2010. Community detection in graphs. *Phys. Rep.* 486, 3–5 (February 2010), 75–174. DOI:<https://doi.org/10.1016/j.physrep.2009.11.002>
- [12] Muhammad Aqib Javed, Muhammad Shahzad Younis, Siddique Latif, Junaid Qadir, and Adeel Baig. 2018. Community detection in networks: A multidisciplinary review. *J. Netw. Comput. Appl.* 108, (April 2018), 87–111. DOI:<https://doi.org/10.1016/j.jnca.2018.02.011>
- [13] Stijn van Dongen and Cei Abreu-Goodger. 2012. Using MCL to Extract Clusters from Networks. In *Bacterial Molecular Networks*, Jacques van Helden, Ariane Toussaint and Denis Thieffry (eds.). Springer New York, New York, NY, 281–295. DOI:https://doi.org/10.1007/978-1-61779-361-5_15
- [14] van Dongen, Stijn, Graph clustering via a discrete uncoupling process, *Siam Journal on Matrix Analysis and Applications* 30-1, p121-141, 2008. <https://doi.org/10.1137/040608635>.
- [15] sklearn.cluster.spectralclustering, <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html>, accessed: 2023-03-30.
- [16] sklearn.cluster.affinitypropagation, <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html>, accessed: 2023-03-30.
- [17] James MacQueen and others. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 281–297.
- [18] Plutchik R. The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American scientist*, 89(4):344–350, 2001.
- [19] H. van de Waal, De rangschikking en catalogisering van een topografischen atlas. *Oudheidkundig Jaarboek* 9 (1940), pp 14-25. Available: <https://iconclass.org/>
- [20] J. Graham, J. Haidt, S. Koleva, M. Motyl, R. Iyer, S. P. Wojcik, and P. H. Ditto, “Moral foundations theory: The pragmatic validity of moral pluralism,” in *Advances in experimental social psychology*. Elsevier, 2013, vol. 47, pp. 55–130.
- [21] S. De Giorgis, A. Gangemi, and R. Damiano, “Basic human values and moral foundations theory in valuenet ontology,” in *International Conference on Knowledge Engineering and Knowledge Management*. Springer, 2022, pp. 3–18.
- [22] CIE. Improvement to industrial colour-difference evaluation. Vienna: CIE Publication No. 142-2001, Central Bureau of the CIE; 2001. Available: <https://cie.co.at/publications/improvement-industrial-colour-difference-evaluation>
- [23] Colorimetry-Part 6: CIEDE2000 Colour-Difference Formula ISO/CIE 11664-6:2022(E), CIE International Standard, 2022. Available: <https://cie.co.at/publications/colorimetry-part-6-ciede2000-colour-difference-formula-1>
- [24] Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, [“Exploring network structure, dynamics, and function using NetworkX”](#), in [Proceedings of the 7th Python in Science Conference \(SciPy2008\)](#), Gael Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008
- [25] J.-B. Lamy, Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies, *Artificial Intelligence in Medicine* 80 (2017) 11–28. doi:<https://doi.org/10.1016/j.artmed.2017.07.002> URL <https://www.sciencedirect.com/science/article/pii/S0933365717300271>
- [26] Alan, R., Robertson. (1977). The CIE 1976 Color-Difference Formulae. *Color Research and Application*, 2(1):7-11. doi: 10.1002/J.1520-6378.1977.TB00104.X

Appendix 1: Community Model Deployment

Requirements:

- Docker Desktop (<https://docs.docker.com/desktop/install/windows-install/>)
- Community Model Repository (<https://github.com/spice-h2020/spice-community-model>)

Deployment:

1. Install Docker using instructions on Docker Desktop official website.
2. Open **env.template** config file located inside “**/deploy**” folder and configure the settings as needed or use default settings, and later change it format to **.env**.
 - a. API user and password: user credentials to access the community model REST API.
 - b. MongoDB user and password: user credentials for the database where the community model stores the information.
 - c. Host ports for mongoDB, the community model API and the internal core.
3. Include a museum configuration file at “**/apiServer/app/src/seedFile.json**” with the explicit communities a citizen can belong to (user attributes), the artwork features (artwork attributes), the interaction features (interaction_similarity_functions), a list with the name and id of the artworks and the available algorithms.
4. Include the artwork data at “**/cmServer/cmSpice/data/artworks.json**”.
5. Launch Docker Desktop
6. Open windows terminal at community-model-api\deploy folder and execute the following:
 - a. Build docker image:

```
docker compose --env-file .env build
```
 - b. Run docker image:

```
docker compose up
```

Initial configuration and use:

Other systems can interact with the community model through the API REST managed by the CM-API, which is described in Deliverable 6.8. The workflow can be simulated with the steps below:

1. Perform a POST request (“POST /users/{user-id}/update-generated-content”) to insert citizen’s demographic and interaction data.
2. Perform a POST request (“POST /perspectives/”) to insert a perspective with the desired parameters for the clustering: citizen attributes, artwork features, interaction features and clustering algorithm.
3. Perform any GET request provided by the community model, such as “GET /communities”. Since there is a new perspective, the internal core of the community model starts the clustering process for that perspective while the CM-API returns a job to the API user which can be used to confirm the request status and retrieve the desired information when the community detection is completed.

Demo files are provided to facilitate the procedure above. These files are located inside “**/demo**” folder.

- **post.py**: performs POST requests with user demographic data.
- **postContribution.py**: performs POST requests with user contribution data.
- **postPerspective.py**: performs POST requests with perspective data.
- **get.py**: performs GET request to communities and monitors the job status.

These scripts use museum data samples located inside “**/demo/data**” folder.

- **ugcUsers.json**: user demographic data.
- **ugcContributions.json**: user contribution data.
- **samplePerspective.json**: perspective data.

Appendix 2: Data samples

GAM (Torino)

Users

Title: GAMGame - information on registered users (deaf users)

Description: GAMGame - information on GAMGame registered users from the experiment collected from 01/10/2022 to 10/11/2022.

UUID: 495778c1-2509-4a9c-be15-fb0b2e9afc08

URL: <https://spice.kmi.open.ac.uk/dataset/details/122>

Number of items: 63

Distribution of demographic groups – Gender:

Female	31
Male	24
Not specified	6
Not binary	2

Distribution of demographic groups – Age:

20-30 age	31
45-60 age	14
31-45 age	11
Not specified	4
13-19 age	3

Distribution of demographic groups – Relationship with art:

I have a strong interest in art	40
Art interests me little	13
I have no interest in art	7
My work is about art	3

Distribution of demographic groups – Relationship with museums:

I visit museums and exhibitions from time to time	28
I often visit museums	25
I rarely visit museums and exhibitions	10

Distribution of demographic groups – Content in sign language.

I would like to see content in Sign Language (LIS)	35
I'm not interested in sign language content	28

Items

Title: GAM_Catalogue_plus

Description: GAM Museum catalogue (version 24/10/2022) enriched with information on iconclass concepts and curators' notes on materials, artistic techniques, authors and their artistic movements.

UUID: 2a2a5c9a-a8ce-4977-ba09-f4134c95d744

URL: <https://spice.kmi.open.ac.uk/dataset/details/113>

Number of items: 56

Contributions

Number of items: 161

Number of interactions with emotions: 120 interactions with emotions, 41 interactions without them.

Distribution of emotions with the highest confidence score:

joy	52
sadness	19
fear	12
serenity	9
anger	5
surprise	5
interest	4
love	4
anticipation	4
trust	3
disgust	3

Number of interactions with sentiments: 113 interactions with sentiments, 48 interactions without them.

Distribution of sentiments with the highest confidence score:

Positive	84
Negative	30
Neutral	11
Mixed	1

Number of interactions with moral values: 137 interactions with moral values, 24 interactions without them.

Distribution of moral foundation values with the highest confidence score:

fairness	93
care	30
purity	13
authority	1

MNCN (Madrid)

Users

Number of items: 27 users from IES Menedez Pelayo (Getafe) 3º ESO.

Citizens answered five questions about three topics of interest in our current society: our water and electricity prospects, food issues and the problem of excessive and irresponsible consumption. Each question is given a score between 0 and 2, with 2 being a good response for a person trying to avoid wasting resources, 1 being neutral and 0 being an answer that accentuates the problem. We assigned the citizens to one of the following groups depending on their total score: very worried (10-9), worried (8-7), neutral (6-5), unworried (4-3) and very unworried (2-0).

Distribution of citizen stances about water and electricity consumption:

very worried	6
worried	11
neutral	9
unworried	0
very unworried	1

Distribution of citizen stances about food consumption:

very worried	1
worried	3
neutral	15
unworried	4
very unworried	4

Distribution of citizen stances about nonspecific consumption:

very worried	3
worried	14
neutral	9
unworried	1
very unworried	0

DMH (Helsinki)

Items

Title: DMH - Pop_up_VR_Museum - Objects classification - English

Description: Classification of all objects from Design Museum Helsinki's permanent collection that are currently in the Pop-up VR Museum. The information about these objects is stored in this dataset in English.

UUID: 0daa0287-d7f4-4f03-a068-95f43afcc347

URL: <https://spice.kmi.open.ac.uk/dataset/details/111>

Number of items: 64

Contributions

Title: DMH Pop-up VR Museum - DEGARI Classification

Description: The dataset contains written stories of 1-4 sentences of design objects in the Pop-up VR Museum. Translations are available at least in Finnish, English, and Swedish. This dataset has been classified by DEGARI, SenticNet7 and Plutchik's library.

UUID: 86e6f3e5-6647-4107-b98b-bbde5cf9ebf5

URL: <https://spice.kmi.open.ac.uk/dataset/details/127>

Number of items: 122

Title: DMH - Pop-up_VR_Museum - Values and Emotions for Transcribed Audio-recorded Stories

Description: MFT and BE extracted values and emotions for the audio-recorded stories carried out by Stefano De Giorgis.

UUID: 5b50a3c1-fcc4-4ad8-aa32-af4019b7b493

URL: <https://spice.kmi.open.ac.uk/dataset/details/132>

Number of items: 122

Additional information: It complements the dataset 127 with the Haidt moral values.

Title: DMH - Pop-up_VR_Museum - Semantic Annotation for Transcribed Audio-recorded Stories

Description: DMH - Pop-up_VR_Museum - Semantic Annotation for Transcribed Audio-recorded Stories

UUID: 03553943-65a9-43f1-ae00-fd83497fe331

URL: <https://spice.kmi.open.ac.uk/dataset/details/134>

Number of items: 121

Additional information: It complements the dataset 127 with emotions and sentiments.

Title: DMH - Pop-up_VR_Museum - Values and Emotions for Written Stories

Description: MFT and BE extracted values and emotions for the written stories carried out by Stefano De Giorgis.

UUID: 5f9f3f35-f36a-4e99-8180-f33988182bc7

URL: <https://spice.kmi.open.ac.uk/dataset/details/131>

Number of items: 406

Title: DMH - Pop-up_VR_Museum - Semantic Annotation for Written Stories

Description: Semantic annotation for the written stories.

UUID: e62f825e-b9b0-4823-baf1-afb9cb211e0a

URL: <https://spice.kmi.open.ac.uk/dataset/details/133>

Number of items: 403

Additional information: It complements the dataset 131 with emotions and sentiments.

We combined the information recorded in these datasets using the storyID as reference. At the same time, for each user, we combined the emotions, moral values and storyIDs of their stories about the same object.

Number of items: 466 interactions.

Number of interactions with emotions: 294 interactions with emotions, 172 interactions without them.

Distribution of emotions with the highest confidence score:

interest	113
joy	39
anticipation	31
surprise	30
sadness	20
trust	17
anger	13
fear	12
disgust	10
love	5
disapproval	3
serenity	1

Number of interactions with sentiments: 350 interactions with sentiments, 116 interactions without them.

Distribution of sentiments with the highest confidence score:

Positive	270
Negative	47
Neutral	33

Number of interactions with moral values: 105 interactions with moral values, 361 interactions without them. Since the moral values are provided as a list, we don't know the moral values with the highest confidence score. Consequently, we consider all of them for the clustering.

Distribution of moral foundation values (an interaction may be associated to more than one moral value):

care	57
harm	38
loyalty	21
fairness	16
liberty	5
sanctity	3
oppression	2

Users

We retrieved the citizen information from the interaction datasets above, since there is not a dataset with the users involved in the interactions.

Number of items: 413 users.

Distribution of demographic groups

General audience	219
Senior citizens	148
Finnish language students	28
Asylum seekers	18

HECHT (Haifa)

Data was directly provided by the user model pipeline.

Users

Number of items: 164 users

Beliefs

Belief (Roman Rebellion): 133 users provide information about their beliefs while 31 don't.

Distribution of citizen viewpoints regarding the Roman Rebellion:

FRealisticNeg	52
EReligiousNeg	9
DExtremistNeg	8
CRealisticPro	34
BReligiousPro	4
ANatPridePro	26

Belief (Joseph Flavius): 143 users provide information about their beliefs while 21 don't.

Distribution of citizen opinions on Joseph Flavius:

Traitor	41
NotTraitor	50
CantJudge	52

Belief (exhibition): 126 users provide information about their beliefs while 38 don't.

Distribution of citizen stances towards the exhibition:

Oppose	31
NoOpinion	36
Justify	6
Balanced	53

Contributions

This museum also provides information about the emotions and sentiments evoked by the topics above.

Number of interactions with emotions (Roman Rebellion): 86 interactions with emotions, 78 interactions without them.

Distribution of emotions (Roman Rebellion) with the highest confidence score:

Disapproval	26
Serenity	14
Joy	9
Sadness	8
Anger	8
Interest	6
Fear	5
Disgust	4
Trust	3
Love	2
Anticipation	1

Number of interactions with emotions (Joseph Flavius): 95 interactions with emotions, 69 interactions without them.

Distribution of emotions (Joseph Flavius) with the highest confidence score:

Disapproval	34
Disgust	18
Trust	14
Sadness	7
Anger	6
Anticipation	5
Fear	4
Interest	3
Joy	2
Surprise	2

Number of interactions with emotions (exhibition): 31 interactions with emotions, 133 interactions without them.

Distribution of emotions (exhibition) with the highest confidence score:

Serenity	10
Disapproval	7
Anticipation	5
Interest	3
Disgust	2
Sadness	2
Joy	2

Number of interactions with sentiments (Roman Rebellion): 137 interactions with sentiments, 27 interactions without them.

Distribution of sentiments (Roman Rebellion) with the highest confidence score:

Negative	64
Positive	43
Neutral	30

Number of interactions with sentiments (Joseph Flavius): 126 interactions with sentiments, 38 interactions without them.

Distribution of sentiments (Joseph Flavius) with the highest confidence score:

Neutral	53
Negative	45
Positive	28

Number of interactions with sentiments (exhibition): 83 interactions with emotions, 81 interactions without them.

Distribution of sentiments (exhibition) with the highest confidence score:

Negative	42
Neutral	21
Positive	19
Mixed	1