University of Trento
Department of Information Engineering and Science

Exploring student behavior in Trento by constructing a Knowledge Graph with spatiotemporal data

Final Report

Studies on Human Behavior

by
Nina Verbeeke
Student nr: 246964
MSC Data Science

Submission Date: 15th January 2024
Course: Studies on human behaviour [145899]

Contents

1 Introduction

The rapid advancement of geospatial and information technology contributes significantly to human behavior research. Smartphones, equipped with built-in GPS capabilities and various sensors, enable the collection of massive amounts of high-resolution spatial-temporal data, tracking the location of individuals over specific time periods. This data provides a dynamic perspective for analyzing human dynamics across space and time, providing an unprecedented opportunity to understand how individuals move, interact, and influence their environments.

Movements are often recorded as trajectories, which consist of sequences of geo-located and time-stamped points outlining the path followed by an object moving through space. In this research, our emphasis lies on the locations where individuals spend time, rather than the entire trajectory or movement between places. To further refine the research scope, we will specifically concentrate on students in the Trento region. Geographic information for this research will be sourced from OpenStreetMapData, a freely accessible and continually updated geographic database, maintained by a community of volunteers through open collaboration. This research will also integrate data from the Smart UniTn 2 (SU2) project, which employed questionnaires, time diaries, and smartphone sensors to comprehensively collect and analyze data with the aim of understanding various aspects of student behavior.

Existing algorithms can be employed to extract stays from a trajectory. These stays represent instances where students spend time at particular locations. Conceptually, a stay can be seen as a cluster of GPS locations that are closely grouped in both time and space, with the latitude and longitude at the cluster's center. To conduct behavior analysis, a crucial step in this research involves associating each stay with its corresponding place. This entails mapping the recorded coordinates onto a dataset to pinpoint the real-world places corresponding to each stay. In addition to this, we augment our dataset with further information about these locations, extending beyond mere coordinates. This additional data might include details about the type of location (e.g., library, restaurant), its characteristics, or other relevant contextual information.

While prior research has extensively explored spatial-temporal analysis, this study introduces a novel approach that leverages the power of knowledge graphs to offer insights into student behavior in Trento. The use of a knowledge graph enables efficient organization and representation of intricate relationships among various entities. This innovative methodology allows for the integration of multiple datasets into a single knowledge graph, providing a robust framework for analysis. The significance of this study is underscored by its contribution to our understanding of student behavior and preferences in Trento. It provides crucial insights that can guide urban planning, campus management, and the enhancement of the overall student experience. As we aspire to build smarter cities, a comprehensive understanding of the activity patterns of populations within urban environments becomes increasingly important. This understanding serves as the motivation for our research into constructing a knowledge graph.

The report unfolds in the following structure: In Chapter 2, the specifics of the research problem are delved into, breaking down its components for a clearer understanding. Chapter 3 takes a closer look at the methodology, explaining the steps taken to address the research problem. Moving on to Chapter 4, the Knowledge Graph Construction is presented. Chapter 5 is dedicated to discussing the evaluation. Finally, Chapter 6 summarizes the key discoveries and mentions directions for future work.

2 Research Problem

The central aim of this research is to delve into the behavior of students in Trento. To achieve this objective, diverse datasets will be integrated, encompassing geographic information crucial for understanding spatial characteristics, GPS trajectory data offering insights into individuals' movement patterns, and personal data about the students from whom GPS data is collected. While each of these datasets contributes valuable perspectives independently, the challenge lies in their seamless integration and analysis.

To address this challenge, the research will focus on constructing a knowledge graph. The primary goal of this project is to create a knowledge graph that comprehensively encapsulates the life sequences of students. This graph will serve as a potent tool for gaining insights into various facets of students' daily lives, capturing details such as visiting points of interest and participation in events. The knowledge graph's construction is pivotal for providing a holistic view of student behavior, contributing to a nuanced understanding of their experiences.

In summary, the research problem revolves around exploring student behavior in Trento through the integration of diverse datasets, and the primary challenge lies in the seamless integration and analysis of these datasets within the construction of a knowledge graph. The construction of this graph is crucial for achieving a comprehensive understanding of the interconnected dynamics of student life.

3 Methodology

This section describes the methodology employed in this research. This research adopts a twofold methodology, encompassing both the construction of a KG and its subsequent application to study human behavior. Subsequent sections will provide a more detailed exploration of both phases of the methodology.

3.1 Phase 1: KG Construction

In the initial phase, the construction of the Knowledge Graph (KG) unfolds through distinct steps, with a clear distinction between the development of the *data layer* and the *schema layer*.

Within the *data layer*, we gather and clean data from various sources, beginning with the identification of relevant datasets to ensure a comprehensive and diverse collection. Subsequent data pre-processing addresses issues such as missing or irrelevant information.

In the *schema layer*, a complete scheme will be constructed to construct a knowledge graph. This schema, or ontology, articulates the high-level structure of the KG, defining the types of entities and relations present. The implementation of this well-defined schema culminates in the systematic organization and representation of data within the KG.

The subsequent sections will elaborate on the specific steps undertaken within the Data layer and the Schema layer, providing a detailed understanding of the KG construction process.

Data layer

The data layer encompasses a variety of activities, such as data collection, cleaning, and processing, handling different types of data that can be structured, semi-structured, or unstructured. In this research, our emphasis is on identifying and reusing existing datasets instead of collecting new data. This approach enables us to leverage the wealth of data already available, optimizing our resource utilization. Data could be gathered from a diverse array of sources, including web services, sensors, and other potential sources identified during the research process. The collected data can be categorized into the following three categories:

Spatial Data: This category comprises objects within Trento that are equipped with spatial or location information. These entities, such as Points of Interest (POI), are specified by longitude and latitude coordinates, denoted as $x = (lat, lng)$.

GPS Data: This category consists of latitude and longitude pairs collected at specific time points for individual users. Using existing algorithms, we extract 'stays' from these trajectories, indicating instances where individuals spend time at specific locations. For the purposes of this research, it is essential to collect a dataset containing GPS data from students.

Attribute Data: This category serves to enhance the other data sources by providing supplementary attribute or property information. This includes details such as Points of Interest information, demographic information of the user, and various auxiliary details commonly referred to as features for objects. Demographic information specifically includes parameters such as gender, age, and education level.

Knowledge layer

The Knowledge Layer focuses on the creation of a schema for the KG. This layer involves the reuse of a set of pre-existing reference schemas or ontologies. Moreover, the following steps are part of this layer:

Entity Identification: The first step is to identify and define the types of entities that will be included in the KG. This involves creating a comprehensive list of entity types along with their definitions.

Relation Extraction: Once the entity types are identified, the next step is to extract relations that describe the semantic connections between these entities. This process helps in establishing meaningful relationships within the KG.

Entity Enrichment: The entities are then enriched with additional attributes. For instance, if 'Restaurant' is an entity type, it might be enriched with attributes such as name, opening times, cuisine, website, and so on.

These steps collectively contribute to the creation of a schema, which serves as a blueprint when building the KG.

Building the KG

The concluding phase of this methodology involves the construction of the KG. This process uses the schema derived from the knowledge layer and the cleaned datasets from the data layer. The objective is to integrate these resources to form the KG, effectively merging the schema and data layers. To facilitate this integration, a specialized data mapping tool named KarmaLinker is employed. KarmaLinker, an extension of the Karma data integration tool, is enhanced with the capability to perform Natural Language Processing on concise sentences, often referred to as the language of data. Within KarmaLinker, we map the data to the corresponding entity types and properties defined in the schema. This is an iterative process applied to the list of previously selected datasets, processed sequentially. The conclusion of this process results in the generation of an RDF file, signifying the creation of the KG.

3.2 Phase 2: KG Application

In the second phase of this research, we transition to the practical application of the constructed KG. This phase is dedicated to leveraging the KG to derive meaningful insights. The KG, containing spatial, GPS, and attribute data, will serve as a tool to study human behavior. Specifically for this research, the main objective is to study the behavior of students in the Trento region.

GraphDB

This phase begins with importing the KG into Ontotext GraphDB, a graph database, and knowledge discovery tool compliant with RDF and SPARQL. GraphDB provides a platform for the efficient storage and retrieval of graph-structured data. Moreover, GraphDB offers visualization tools that enable a clear and intuitive representation of the interconnected entities and relationships within the KG.

SPARQL

To extract valuable insights from the KG, SPARQL queries will be employed. SPARQL, a query language for RDF databases, allows for precise and targeted retrieval of information. We can run SPARQL queries to analyze specific aspects of human behavior, such as identifying frequently visited locations, uncovering temporal patterns, and exploring interactions embedded in the KG. GraphDB provides tools for recording query results, visualizations, and analytical outcomes. This is important for presenting research findings.

Behavioral analysis using constructed KG

One primary objective in this phase is to conduct a comprehensive analysis of human behavior, particularly focusing on students. Leveraging the KG, we aim to explore patterns, trends, and correlations within the data. The spatial and GPS data contribute to understanding the mobility patterns of students, identifying frequented locations, and analyzing temporal behaviors.

By harnessing the temporal information embedded in the KG, we delve into temporal patterns exhibited by individuals. This includes studying daily routines, identifying peak activity periods, and discerning any recurring behavioral trends over time. Temporal analysis provides valuable insights into the dynamic aspects of human behavior.
The KG's structure facilitates the exploration of semantic interactions among different entities. By analyzing the relationships and connections within the KG, we aim to uncover meaningful insights into how individuals interact with their surroundings, institutions, and each other. Semantic analysis enriches the understanding of the contextual significance of behaviors.

Tailoring the analysis to the objectives of this research, we delve into application-specific insights. For instance, understanding student preferences for certain locations, identifying factors influencing their mobility, and gaining insights into the impact of specific attributes on behavioral patterns. This phase encapsulates the practical application of the KG in studying human behavior.

4 Knowledge Graph Construction

4.1 Data Layer

The data layer encompasses a variety of activities, such as data collection. In this research, our emphasis is on identifying and reusing existing datasets instead of collecting new data. This approach enables us to leverage the wealth of data already available. This section will provide an overview of each identified dataset that will be used.

4.1.1 Data Collection

OpenStreetMap

OpenStreetMap (OSM) is a free, open geographic database updated and maintained by a community of volunteers through open collaboration. To retrieve the OSM data, Geofabrik, a free download server that regularly updates its data extracts from the OSM, will be used. Specifically, we obtain the file `nord-est-latest.osm.pbf`[1], containing OSM data for the northeastern region of Italy, encompassing Trento, our region of interest, from Geofabrik. Later in the project, this data will be filtered, and in the KG, we will only display OSM objects located in the Trento region.

Smart UniTn 2 Dataset

The second dataset used in this research is the dataset from the Smart UniTn 2 (SU2) project, a four-week initiative that began on May 7th and ended on June 7th, 2018, for a total of 32 days [1]. Participants in the study were exclusively University of Trento students, and data collection was facilitated through the installation of the i-Log app on their smartphones. This application collects data from a variety of sensors, including both hardware (e.g., GPS, accelerometer, gyroscope) and software (e.g., applications running on the device).

The dataset contains data from three different sources. First, data is gathered from questionnaires filled out by participants, encompassing gender, age, enrolled departments, and psycho-social characteristics. Secondly, time diaries include responses detailing their activities, movement patterns, locations, social companionship, and mood. Finally, data is collected using sensors on the user's smartphone. In this study, particular attention is directed toward the GPS data. The GPS data includes the rounded-down location collected at one-minute intervals for a comprehensive understanding of spatial information. For comprehensive details on the dataset, including tools used, links to questionnaires, iLog documentation, label descriptions, variable values, and summary statistics, , please refer to the documentation available on the DataScientia website [2]

4.1.2 SU2 Data Preprocessing

We load the parquet files, specifically `timediariesanswers-0.parquet`, containing time diary answers of students, and all parquet files within the folder `locationeventpertime_rd` containing GPS locations of students at specific times. Additionally, as we aim to find the closest Points of Interest (POIs) later on to a StayPoint, we also load the OSM Dataset. We use the Python package OSMnx to facilitate the easy download of data from OSM.

[1] https://download.geofabrik.de/europe/italy/nord-est.html
[2] http://livepeople.datascientia.eu/workspace/smartunitntwo

Listing 1 Use OSMnx to download data from OSM

```
import osmnx as ox
poi = ox.features_from_place("Trento", tags={key: True})
```

For the time diaries data, we translate the A2 answers, which represent responses to the question 'Where are you?', into English. For the GPS data, there are latitude, longitude, and altitude columns. However, a data inconsistency exists where certain columns have values swapped. For example, for the user with ID 25, we noticed that the latitude column contains the longitude values, the longitude column contains the altitude values, and the altitude column contains the latitude values. We used code to identify a specific pattern, which is shown in the list below:

- For most users the order is correct => Order: LAT, LNG, ALT
- For the users with ids `[25,32,41,43,56,57,58,59,60,61,62]` => Order: LNG, ALT, LAT
- For users with ids `[28,36]` => Order: LAT, ALT, LNG
- For users with ids `[30,34,37,38,42]` => Order: ALT, LNG, LAT

We implemented a function, identified as `def swap_columns(df)`, to correct the swapped values for all users in the dataset. This code ensures the proper alignment of columns.

In the subsequent steps, we leverage the sci-kit mobility package[3] to create and process trajectory data. Our goal is to generate a trajectory dataframe, and for this, we need to introduce a new column called 'traj_id.' The addition of this column involves assigning a unique trajectory ID to each row, primarily based on either a time gap exceeding one hour or a change in user_id compared to the preceding row. After implementing this step, we construct a TrajDataFrame object, featuring columns for latitude, longitude, and datetime. Subsequently, we apply a trajectory compression technique, which involves reducing the number of points in a trajectory for each individual in the TrajDataFrame. This compression consolidates all points within a specified radius of 'spatial_radius_km' from an initial point into a single point with the median coordinates and the time of the initial point. Then our focus shifts to stay point detection. We employ the function `stay_locations` from the sci-kit mobility package to identify stay locations (or stops) for each individual in the TrajDataFrame. A stop is identified when an individual spends at least 'minutes_for_a_stop' minutes within a distance of 'stop_radius_factor * spatial_radius' kilometers from a given trajectory point. The coordinates of the stop are determined as the median latitude and longitude values of the points found within the specified distance.

Listing 2 Using functions provided by skmob package to determine stay locations of students

```
import skmob
from skmob.preprocessing import compression, detection

# Create a TrajDataFrame from the DataFrame
tdf = skmob.TrajDataFrame(df, latitude = 'latitude', longitude = '
    longitude',
                              datetime = 'timestamp', user_id = 'user_id',
                                  trajectory_id='traj_id')

# Compress the trajectories
ctdf = compression.compress(tdf, spatial_radius_km = 0.2)

# Detect Stay locations
duration = 5    # value specifying the minimum duration of a stop.
```

[3] https://github.com/scikit-mobility/scikit-mobility

```
radius =  40    # value specifying the maximum radius a stop can have.
stdf = detection.stay_locations(tdf,
                                stop_radius_factor = 0.5,
                                minutes_for_a_stop = duration,
                                spatial_radius_km = radius,
                                leaving_time = True)

stops = pd.DataFrame(stdf)
```

The next step involves identifying the homes of the students. Furthermore, for later analysis, understanding the students' home locations could be relevant. We possess a timediary wherein a student may have indicated being at home. The simplest approach would be to identify the GPS data collected when the student noted being at home, providing us with the latitude and longitude coordinates of their home. However, this approach yielded inaccurate results.
Consequently, we opted for a second approach. We use the stop locations of students, which are retrieved using the code shown in the snippet **??**. We iterate through each row in the 'stops' dataframe, extracts relevant information about stop locations, and matches them with corresponding time diary entries based on user ID and datetime constraints. It then updates a 'homes' dataframe, keeping track of the frequency of each stop location being associated with the specified home category. After processing all stops for a given user, the function identifies the stop location with the highest frequency (occurring at least 10 times) and sets it as the home location for that user. The resulting information is stored in a dictionary named 'homes_users_new', where each user ID is mapped to their determined home location. The function aims to establish a connection between stop locations and home indicators in time diaries, providing a more concrete definition of students' home locations based on their frequently visited stops. The code can be found in the function `connect_stops_and_td` within the provided code.

Now that we have detected the stops, containing latitude and longitude coordinates, our goal is to identify the corresponding locations for these stops using the OSM dataset. However, it is not logical to find OSM places for the homes of the students. Therefore, we initially iterate through each row in the 'stops' dataframe to augment it by adding a 'home' column. This iteration involves checking if the associated 'uid' is present in the 'df_homes' dataframe, which contains home coordinates. If a match is found, the code calculates the distance between the stop's coordinates and the home coordinates. If the distance is less than 'max_distance_home' meters, it sets the 'home' column to True; otherwise, no action is taken.

We filter out the stops of users when they were at home. For the remaining stops, our objective is to identify the exact places that users might have visited in close proximity to the stop locations. To achieve this, we leverage code provided by [2], which includes a function for enriching stops with Points of Interest (POIs) that may be relevant. Specifically, for each stop, the code ranks the POIs based on certain criteria and associates the top k ranked POIs with the stop, considering their distance from the stop. These associated POIs can have various geometric shapes, such as points, lines, and polygons. It's worth noting that the function retrieves POIs from OSM.

Listing 3 Function copied from MAT-Builder repository, which contains code from [2]

```
def stop_enrichment_with_pois(df_stops, df_poi, suffix, max_distance):
    # Prepare the stops for the subsequent spatial join.
    stops = gpd.GeoDataFrame(df_stops,
                                geometry=gpd.points_from_xy(df_stops.lng,
                                     df_stops.lat),
                                crs="EPSG:4326")
    stops.to_crs('epsg:3857', inplace=True)
    stops['geometry_stop'] = stops['geometry']
```

```
    stops['geometry'] = stops['geometry_stop'].buffer(max_distance)


    pois = df_poi.copy()
    pois.to_crs('epsg:3857', inplace=True)

    # Filter out the POIs without a name!
    pois = pois.loc[pois['name'].notna(), :]

    # duplicate geometry column because we loose it during the
        sjoin_nearest
    pois['geometry_' + suffix] = pois['geometry']
    pois['element_type'] = pois['element_type'].astype(str)
    pois['osmid'] = pois['osmid'].astype(str)

    # Execute the spatial left join to associate POIs to the stops.
    enriched_stops = stops.sjoin_nearest(pois, max_distance=0.00001, how=
        'left', rsuffix=suffix)

    # Remove the POIs that have been associated with the same stop
        multiple times.
    enriched_stops.drop_duplicates(subset=['stop_id', 'osmid'], inplace=
        True)

    # compute the distance between the stop point and the POI geometry
    enriched_stops['distance'] = enriched_stops['geometry_stop'].distance
        (enriched_stops['geometry_' + suffix])

    enriched_stops = enriched_stops.sort_values(['stop_id', 'distance'])
    enriched_stops.reset_index(drop = True, inplace = True)
    return enriched_stops
```

This provides us with the OSM ID of the places located close to a student's stop location. The final step is to add a column named `osmid_uri` in the form: `https://www.openstreetmap.org/{element_type}/{osmid}`, where `element_type` is either Node or Way.

4.2 Knowledge Layer

4.2.1 SU2 Schema

In the preliminary stages of this project and documented in [3], an Entity-relationship (ER) model was created that will be used to create a schema of the Smart unitn 2 dataset. Please see the paper [3] for comprehensive background information elucidating the reasoning behind the ER model. The complete ER model is also displayed in Appendix A of this paper.

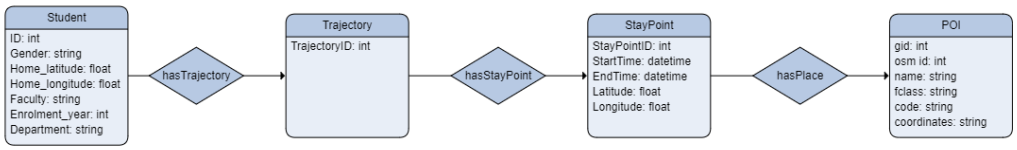


Figure 1 Simplified ER model.

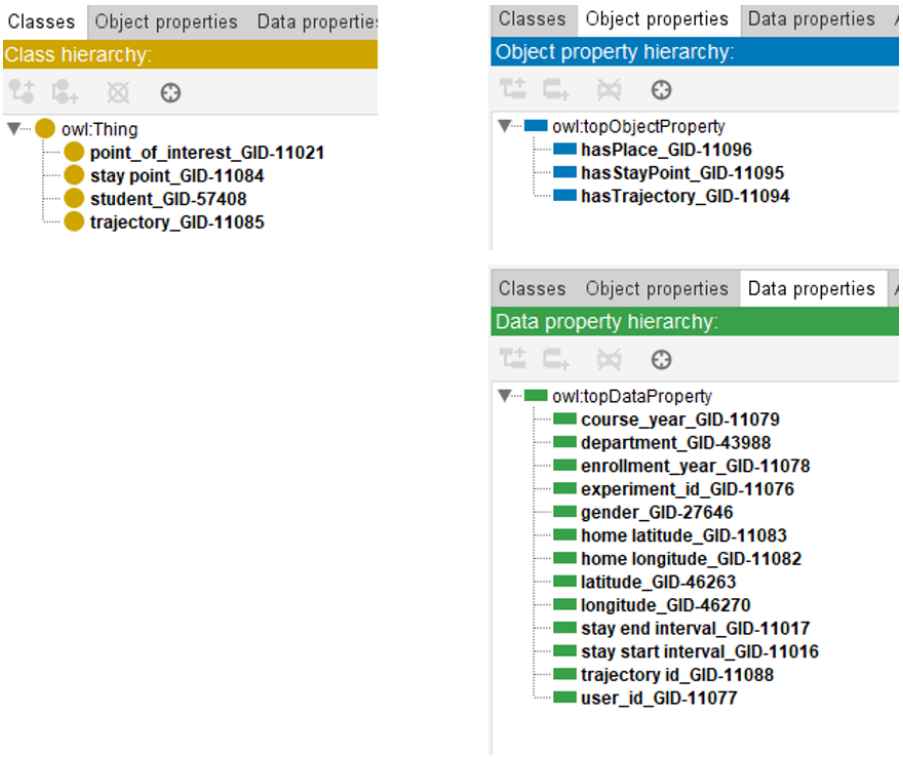


Figure 2 SU2 Schema in Protégé.

The primary objective in constructing this ER model was to align with the research's purpose. Leveraging insights from other research papers aims to align our approach with established related work. Notably the following papers are related: [4] and [5]. The ontology presented in [4], as shown in Appendix B, served as inspiration for the ER model created for this project. The list below provides an overview of entity types along with brief definitions:

- **Trajectory**: the path followed by an object moving through space
- **StayPoint**: a single instance of a person spending some time in one place
- **POI**: a specific and notable place marked on a map due to its distinctive features where one or more objects have stayed

Appendix A delves into the entity types in more detail and provide the motivation behind selecting these particular entity types.

4.2.2 OSM Ontology

OSM serves as a rich source of openly available geographic information. However, the tag-based structure of OSM data does not adhere to a well-defined ontology. OSM does not follow a strict schema. Integrating OSM with knowledge graphs is inherently challenging.

Previous research has developed various ontologies that can be used for integrating OSM data into a KG. One such approach is presented in [6], where WorldKG is introduced as a novel, comprehensive geographic knowledge graph built from the OSM dataset. We will reuse their WorldKG ontology, which is created by converting the flat OSM schema into a hierarchical ontology structure.

Overall, the number of geographic entities in WorldKG is extensive, enabling the provision of semantic annotations to OSM entities. By using this ontology, we can create large-scale semantic

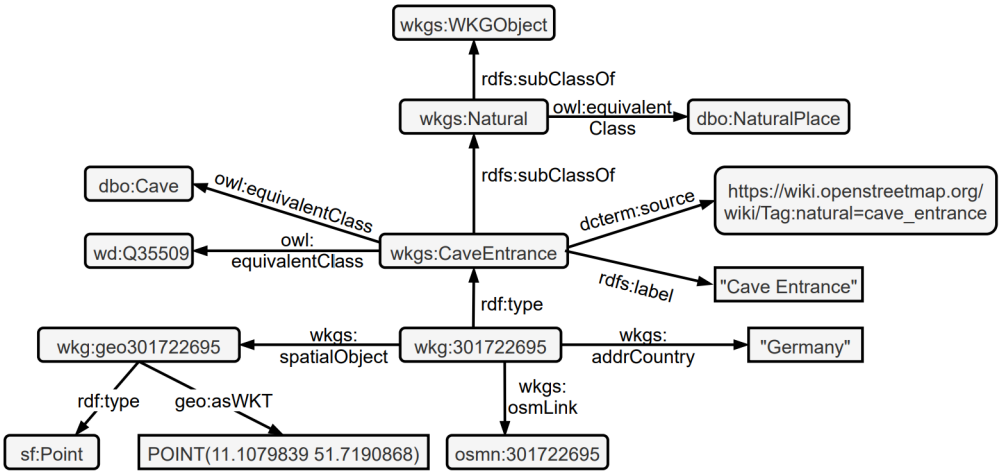


Figure 3 Retrieved from [6]. Example instantiation of the WorldKG ontology for a specific instance of wkgs:CaveEntrance.

geographic data extracted from OSM. The WorldKG ontology semantically describes geographic entities and links them to specific classes in the Wikidata and DBpedia ontologies. As the source code for the entire WorldKG creation pipeline is available on GitHub, we can easily reuse it [4]. The WorldKG ontology, as created by [6], is provided in Appendix C. It is partially displayed in Figure 4.

Each category within the WorldKG ontology functions as a subclass of wkgs:WKGObject, with the namespace wkgs denoting elements of the WorldKG schema. WorldKG properties, identified as wkgs:WKGProperty, furnish information regarding OpenStreetMap (OSM) tags that lack type assignments. To denote the geographical extent of each wkgs:WKGObject entity, a relationship can be established with a geo:SpatialObject using the property wkgs:spatialObject, where a geo:SpatialObject signifies a point. This point represents the entity's geometry type (sf:Point) and the associated coordinates. The OSM community offers a map feature list consisting of established key-value pairs. A class hierarchy will be constructed using this map feature list, treating all keys as top-level classes (e.g., natural) and representing values as their subclasses. For instance, the key-value pair natural=cave_entrance is represented in the WorldKG ontology as follows: The OSM key "natural" transforms into the top-level class wkgs:Natural, encapsulating nature entities, while the OSM value "cave_entrance" becomes a subclass of wkgs:Natural, specifically wkgs:CaveEntrance, signifying cave entrances. Figure 3 provides an illustration of how the key-value pair natural=cave_entrance is represented in the WorldKG ontology. Only categorical values are considered as subclasses, excluding other value types such as boolean or numerical values. Instead, the corresponding key dictates the top-level class, as seen in the example where an entity tagged as building=yes is classified as wkgs:Building. Each entity is linked to the original OSM node through the property wkgs:osmLink.

4.2.3 Complete Schema

In this section, we delve into the complete schema for the KG that is to be constructed in this project. The schema or ontology describes the high-level structure of a KG, including the type of

[4] https://github.com/alishiba14/WorldKG-Knowledge-Graph

entities and relations therein. Figure 4 depicts the overall schema of the KG, wherein the nodes represent types of entities, and the edges describe their relationships within the KG.

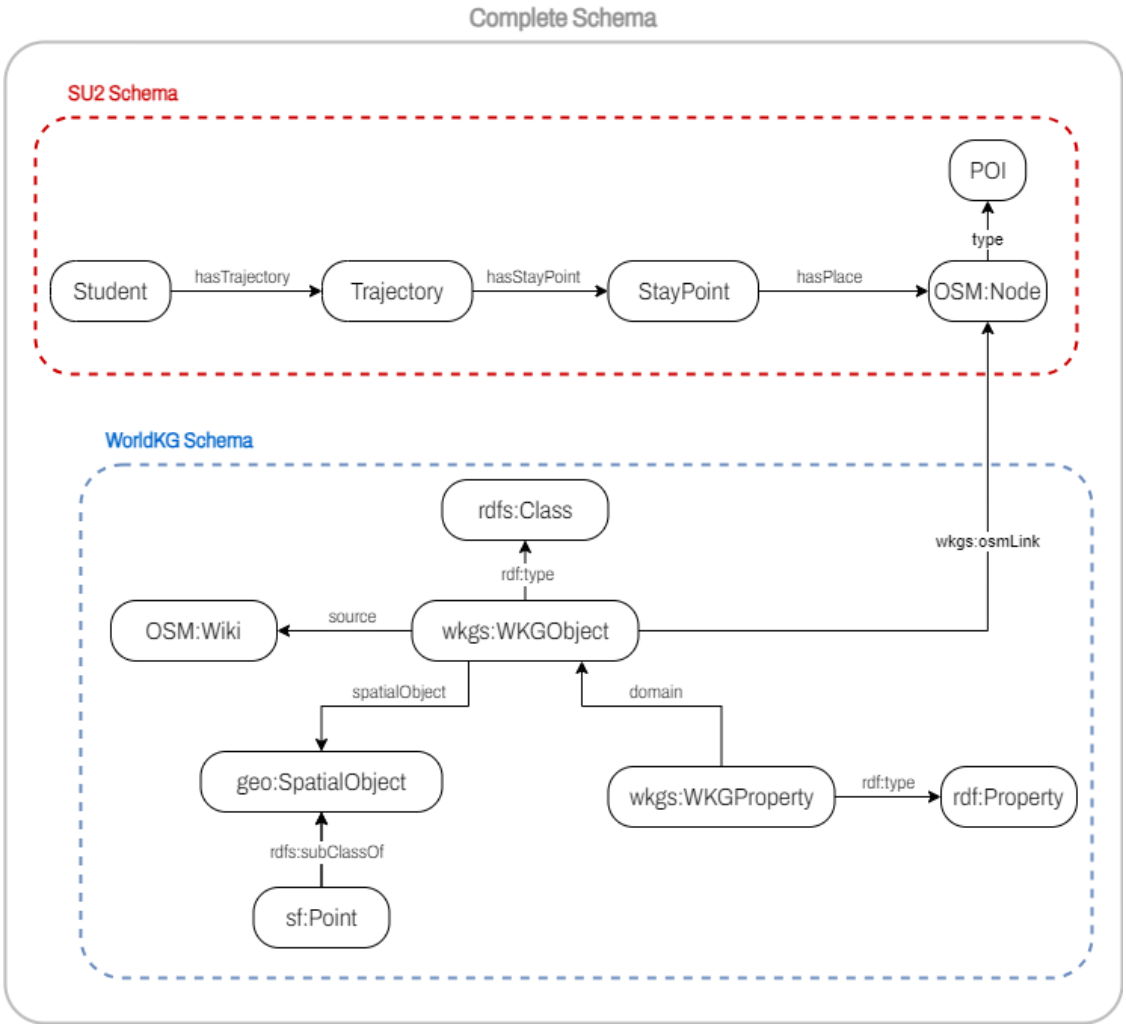


Figure 4 Complete Schema of KG

Please note that it is a combination of the SU2 ontology, created specifically for this research and depicted in red in Figure 4, along with a reuse of the WorldKG ontology, represented in blue in Figure 4. These two ontologies can be easily combined into a single schema since they both have a connection to an OSM node. We will now provide an overview of the complete schema.

The entity types used in the red part, i.e., the SU2 ontology part, of the schema are already defined in Section 4.2.1. We have the entity types Student, Trajectory, StayPoint, and POI, along with relations including hasTrajectory, hasStayPoint, and hasPOI. By employing an existing algorithm, Points of Interest (POIs) that are within x meters of the StayPoint will linked to the StayPoint, where x depends on the configuration of the algorithm being used. As OSM is used, we represent the entity type OSM:Node, which has a relation 'type' to POI. This ontology facilitates the inclusion of trajectory data in the KG. The part in blue showcases the entities of the WorldKG ontology, as defined in Section 4.2.2. Each class in the WorldKG ontology is a subclass of wkgs:WKGObject. WorldKG properties are modeled as wkgs:WKGProperty and

provide information on OSM tags that do not indicate a type assignment. To provide information about its geographic extent, each wkgs:WKGObject entity can be related to a geo: SpatialObject via the property wkgs:spatialObject. The geo:SpatialObject represents the type of geometry of the entity (sf:Point) and the coordinates of the geometry. For each entity, we also provide the property wkgs:osmLink that links the entity to the original OSM node.

4.3 Data Integration

The final step in constructing the KG involves Data Integration. Its input comprises the complete schema (refer to 4) and the cleaned datasets. The goal is to build the KG by integrating the schema and data resources. To achieve this, we follow two steps. The first step involves aligning the WorldKG ontology with OSM data, resulting in an .ttl file. The code provided on GitHub[5] by the authors of [6] was adapted to create triples resulting in an TTL file. The following changes have been made to the code:

- The provided code uses `osmium.SimpleHandler`, which, as seen in the code, focuses only on handling OSM Nodes. However, our interest extends to OSM Ways as well. As an example, consider the building named 'Povo 1', where the Department of Information Engineering and Computer Science is situated. In OSM, this building is represented as a way with the ID 275318445. We aim to ensure that this location is not filtered out, as it is not a node but a way. To accommodate OSM Ways in our code, we need to address the fact that OSM Ways do not have (lat,lon) point coordinates like nodes. Instead, they are represented as polygons. For simplification, we have chosen to use the center point (lat,lon) of the OSM Ways. This modification allows us to include both OSM Nodes and OSM Ways in the code's functionality.
- To reduce the size of the constructed TTL file, we implemented specific filters. For both OSM Nodes and OSM Ways, we require the presence of the 'amenity' tag in the OSM Tags. This filtering criterion aligns with the research focus on point-of-interest visits. Additionally, we check whether the (lat,lon) coordinates, representing either the point of a node or the center point of a polygon, fall within the borders of the Trentino region defined by a polygon. This additional check ensures that only data relevant to the geographical scope of interest, Trentino, is retained in the TTL file.

After implementing the changes to the code, we execute it using the OSM dataset, as specified in Section 4.1.1: `nord-est-latest.osm.pbf`. The .ttl file generated by executing this code is available on GitHub. Please note that since this file already contains triples; the schema and data have already been aligned. There is no need to input this file into KarmaLinker, so it can be inputted directly into GraphDB.

For the SU2 dataset, we import both the schema and the cleaned dataset as inputs into KarmaLinker. The cleaned dataset is stored in the file `stops_with_closest_poi.csv`; however, for privacy reasons, this file has not been published. The schema for SU2 has been created using Protégé, as described in [3] and illustrated in Figure 2. The schema is stored in the `schema_su2.rdf` file, which is available in the GitHub repository. Subsequently, we map the data to the entity types and properties of the schema. The next step involves generating the entities, which are then matched. A screenshot of Karma during the data mapping process can be seen in Figure 5. The process concludes with the export of the integrated data into a .ttl file.

[5] https://github.com/alishiba14/WorldKG-Knowledge-Graph/blob/main/CreateTriples.py

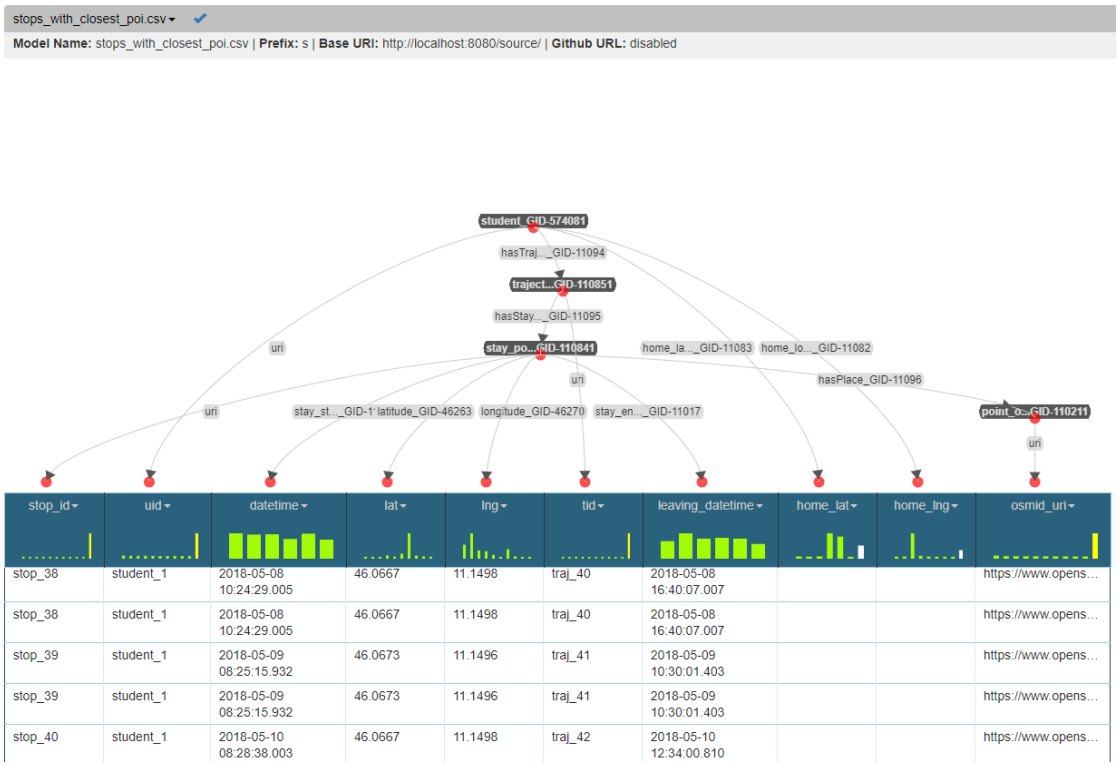


Figure 5 Screenshot of KarmaLinker used as data mapping tool

4.4 Final KG

As detailed in the preceding section, two TTL files are created. Both files are imported into GraphDB, and collectively, they constitute a unified KG.
The figure below shows a screenshot of a small part of the KG.

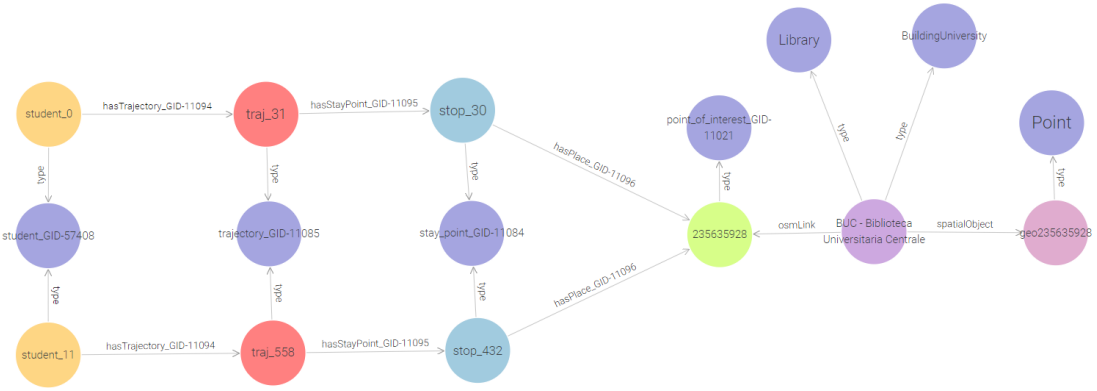


Figure 6 Screenshot of small part of KG

5 Evaluation

In the subsequent stage of this study, we shift towards the practical implementation of the developed Knowledge Graph (KG). This phase focuses on harnessing the KG to extract valuable insights, with a specific emphasis on investigating the behavior of students in the Trento region. The potential insights that can be obtained are vast, as the flexibility of writing SPARQL queries provides freedom to explore various aspects.

Example 1

In the first example, we write a SPARQL query with the goal of figuring out which libraries in Trento are the most visited by students. The query and its results can be seen below.

Listing 4 SPARQL query 1

```
PREFIX wkgs: <http://www.worldkg.org/schema/>
PREFIX etype: <http://knowdive.disi.unitn.it/etype#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?poi  ?osmLinkLabel (COUNT(?stayPoint) AS ?totalStayPoints) ?
    osmLink
WHERE {
  ?poi rdf:type etype:point_of_interest_GID-11021 .
  ?stayPoint rdf:type etype:stay_point_GID-11084 ;
             etype:hasPlace_GID-11096 ?poi .
  ?osmLink wkgs:osmLink ?poi .
  ?osmLink rdf:type wkgs:Library .
  ?osmLink rdfs:label ?osmLinkLabel .
}
GROUP BY ?poi ?osmLink ?osmLinkLabel
ORDER BY DESC(?totalStayPoints)
```

	poi	osmLinkLabel	totalStayPoints
1	https://www.openstreetmap.org/way/275318462	"BUP - Biblioteca Universitaria Povo"	"130"^^xsd:integer
2	https://www.openstreetmap.org/way/235635928	"BUC - Biblioteca Universitaria Centrale"	"97"^^xsd:integer
3	osmn:9630961821	"Sala studio Cavazzani"	"57"^^xsd:integer
4	https://www.openstreetmap.org/way/1181361605	"Biblioteca Comunale di Trento"	"34"^^xsd:integer
5	osmn:10033381157	"Facoltà di Giurisprudenza Entrata"	"26"^^xsd:integer
6	osmn:1559090545	"BUM - Biblioteca Universitaria Mesiano"	"1"^^xsd:integer

Figure 7 Results after executing SPARQL Query 1

Example 2

In the second example, we are interested in selecting the libraries that are closest to Jane, who is living at the location with coordinates latitude 46.065088834088634 and longitude 11.123805065193205. The query and its results can be seen below.

Listing 5 SPARQL query 2

```
PREFIX geo: <http://www.opengis.net/ont/geosparql#>
PREFIX sf: <http://www.opengis.net/ont/sf#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX wkgs: <http://www.worldkg.org/schema/>

SELECT ?entity ?type  ?osmLinkLabel ?coordinates ?lat ?lon  ?
    distanceSquared ?pointEntity
WHERE {
  ?entity a sf:Point;
          geo:asWKT ?coordinates;
          a ?type.
  ?pointEntity wkgs:spatialObject ?entity.
  ?pointEntity rdf:type wkgs:Library .
    ?pointEntity rdfs:label ?osmLinkLabel .
  BIND (STR(?coordinates) AS ?coordinatesStr)
  BIND (STRAFTER(?coordinatesStr, "Point(") AS ?pointCoords)
  BIND (STRBEFORE (?pointCoords, "␣") AS ?lon)
  BIND (STRAFTER (?pointCoords, "␣") AS ?lat1)
  BIND (STRBEFORE(?lat1, ")") AS ?lat)
  BIND((xsd:double(?lon) - 11.123805065193205)*(xsd:double(?lon) -
      11.123805065193205) + (xsd:double(?lat) - 46.065088834088634) * (
      xsd:double(?lat) - 46.065088834088634) AS ?distanceSquared)
}
ORDER BY DESC(xsd:double(?distanceSquared))
LIMIT 5
```

	entity	type	osmLinkLabel	coordinates	lat	lon	distanceSquared	pointEntity
1	wkg:geo7095449517	sf:Point	"Piccola biblioteca di Cadine"	"Point(11.0628437 46.0857594)"^^<http://www.opengis.net/ont/geosparql#wktLiteral>	"46.0857594"	"11.0628437"	"0.004143560341315383"^^xsd:double	wkg:7095449517
2	wkg:geo7095449514	sf:Point	"Biblioteca Sopramonte"	"Point(11.0599502 46.0716169)"^^<http://www.opengis.net/ont/geosparql#wktLiteral>	"46.0716169"	"11.0599502"	"0.004120059453385572"^^xsd:double	wkg:7095449514
3	wkg:geo4715779117	sf:Point	"Biblioteca Meano"	"Point(11.1161227 46.1265883)"^^<http://www.opengis.net/ont/geosparql#wktLiteral>	"46.1265883"	"11.1161227"	"0.003841203042345235 5"^^xsd:double	wkg:4715779117
4	wkg:geo1607714595	sf:Point	"Biblioteca Mattarello"	"Point(11.1278038 46.0060602)"^^<http://www.opengis.net/ont/geosparql#wktLiteral>	"46.0060602"	"11.1278038"	"0.003500369522424888"^^xsd:double	wkg:1607714595
5	wkg:geo7095449515	sf:Point	"Punto di prestito Romagnano"	"Point(11.1059033 46.0171551)"^^<http://www.opengis.net/ont/geosparql#wktLiteral>	"46.0171551"	"11.1059033"	"0.002618116060712847 7"^^xsd:double	wkg:7095449515

Figure 8 Results after executing SPARQL Query 2

6 Conclusions and Future Work

In conclusion, this research has successfully achieved its objective of constructing a knowledge graph that encapsulates the life sequences of students. This knowledge graph serves as a valuable tool for gaining insights into various aspects of students' daily lives, capturing points-of-interest places they visit, and conducting events. The practical outcomes of this project signify a promising starting point, with room for further enhancement and refinement.

Looking ahead, several potential directions for future work have been identified. These include integrating more data from the student dataset, such as mood or personality, and incorporating more location-specific data into the knowledge graph, possibly from additional sources. There is also the potential to implement the use of a different dataset with GPS locations in the Trentino region. A shift in focus towards the student life sequence, as opposed to merely points-of-interest visits, could provide further depth to the analysis. Additionally, a more detailed examination of events, including the implementation of a method to analyze events taking place at specific locations, could enrich the understanding of student behavior patterns.

Overall, while the project has yielded practical outcomes, it represents just the beginning of a broader exploration into student behavior using KGs. The potential for future work is extensive, promising exciting developments in this field of research.

References

1 Fausto Giunchiglia et al. *A survey on students' daily routines and academic performance at the University of Trento*. Technical Report DISI-2001-DS-04. University of Trento, 2022. URL: https://iris.unitn.it/retrieve/e3835199-ed48-72ef-e053-3705fe0ad821/2022_DataScientia_LivePeople_SmartUnitn2.pdf.

2 Chiara Pugliese et al. MAT-Builder: a System to Build Semantically Enriched Trajectories. *2022 23rd IEEE International Conference on Mobile Data Management (MDM)*. 2022, p. 274–277. DOI: 10.1109/MDM55031.2022.00058.

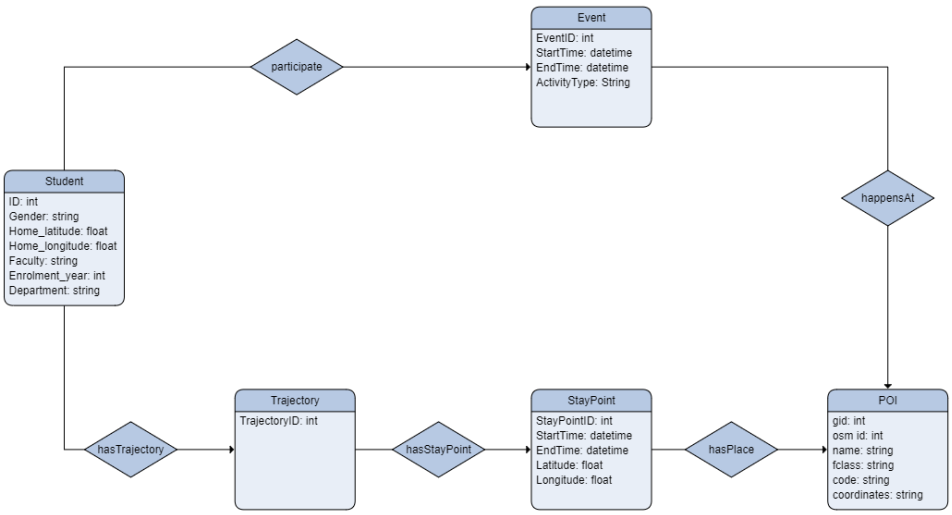
3 Nina Verbeeke and Munkhdelger Bayanjargal. *DISI Student Lives Points of interest in Trentino*. Jan. 2024. URL: https://n-verbeeke.github.io/kge-project-11-webpage/assets/KGE_Project11_Report.pdf.

4 Yingjie Hu et al. A Geo-ontology Design Pattern for Semantic Trajectories. *Spatial Information Theory*. ed. by Thora Tenbrink et al. Cham: Springer International Publishing, 2013, p. 438–456.

5 Zhangqing Shan et al. Extract Human Mobility Patterns Powered by City Semantic Diagram. *IEEE Transactions on Knowledge and Data Engineering*, 2022. 34(8): p. 3765–3778. DOI: 10.1109/TKDE.2020.3026235.

6 Alishiba Dsouza et al. WorldKG: A World-Scale Geographic Knowledge Graph. *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*. ACM, 2021. DOI: 10.1145/3459637.3482023.

A Entity-Relationship Model



Student

On the left hand side of the ER model there is the entity type 'Student'. The student has the data properties ID, Gender, Home_latitude, Home_longitude, Faculty, Enrolment_year, Department. The inclusion of these data properties is purpose-driven, consider for example Persona 5, who is doing a sociology research on student behavior analysis, if she is using the knowledge graph to perform analysis, she might be interested to have these properties because she want to be able make comparisons between students. For example, using the department and faculty to see if there is differences between students dependent on these properties. If we go down from student we can see the object property 'hasTrajectory' that goes to Trajectory.

Trajectory

The decision to incorporate the 'Trajectory' entity type in this research project is underpinned by the study's specific objectives and ambitions. The main purpose of the research is to construct a knowledge graph that delves into the analysis of student behavior. While the focus includes the places students visit, the project extends beyond a mere examination of individual locations. A key facet of the research is the desire to understand and study the *journey* or trajectory of a student. This acknowledges the importance of capturing trajectories, recognizing that each student's journey involves unique patterns and variations. By incorporating the 'Trajectory' entity type, the knowledge graph serves as a tool to perform analysis on student behavior based on trajectories and to identify potential patterns in those trajectories. It should be noted that trajectory data can be analysed in order to obtain interesting mobility patterns. Having a seperate 'Trajectory' entity simplifies the formulation of complex queries related to trajectories. Instead of relying on SPARQL queries to reconstruct trajectories from stay points, you can directly query and analyze trajectory-specific properties, making the querying process more intuitive and efficient.
To further support the use of 'Trajectory', let us look at a few competency questions that would be interesting to answer and that support the inclusion of the 'Trajectory' entity type:

- Provide trajectories of students who visit the university, then optionally stop at the supermarket,

and end at home, without intermediate stops.

- For each trajectory, compute the distance travelled between each pair of consecutive stops.

These competency questions are out of the scope of this research; however, they are still formulated because they are part of a project running in parallel to this one and closely related to it. This related project is conducted for the university course 'Studies on Human Behavior.'
In the knowledge graph, stay points are grouped within trajectories. From the Trajectory, there is an object property to 'StayPoint', indicating that a trajectory consists of one or more StayPoints.

StayPoint

Another important entity type in the ER model is StayPoint. A StayPoint represents a spatial-temporal instance where a person spends time in a specific location. In other words, a stay point is a location where a person spent x minutes within a distance of y meters. The exact values of these parameters depend on the configuration of the algorithm used to determine stay locations. A StayPoint can be viewed as a cluster of GPS locations closely situated in time and space, with the latitude and longitude at the center of this cluster. By grouping together GPS points that are close both temporally and spatially, a StayPoint is defined. Additionally, each StayPoint has a StartTime and an EndTime, signifying the moment when someone arrives at the stay and when they depart from it.

POI

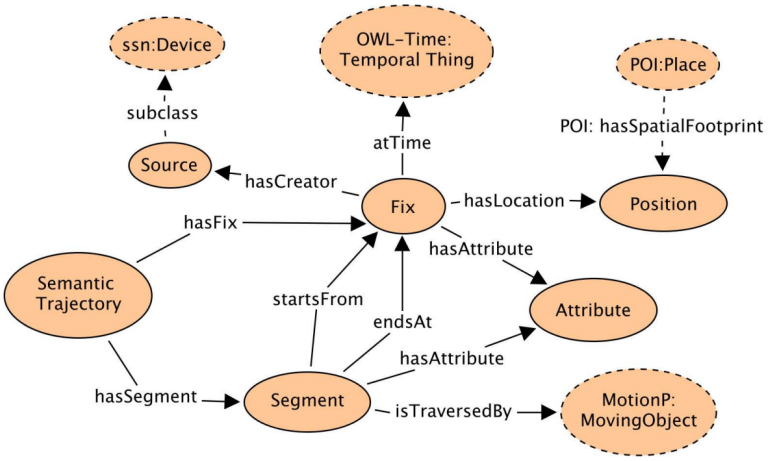
The next entity type in the ER model is POI, an abbreviation for point-of-interest. A POI represents any location where one or more objects have experienced a stay, and it is a specific and notable place marked on a map due to its distinctive features.
The distinction between StayPoint and POI lies in their nature. A StayPoint is spatial-temporal, indicating a single instance of a person spending time in one place, characterized by closely clustered GPS locations. On the other hand, a POI is a spatial point, representing any place where one or more objects have experienced a stay, emphasizing the significance of the location. It may be observed that we have decided to use the term 'POI' and not the more general term 'Place.' As defined earlier, a POI is a subclass of Place, but it specifically refers to a notable location marked on a map due to its distinctive features. The motivation behind including POI is rooted in the purpose of this research, which focuses on studying the visits to POIs by students.

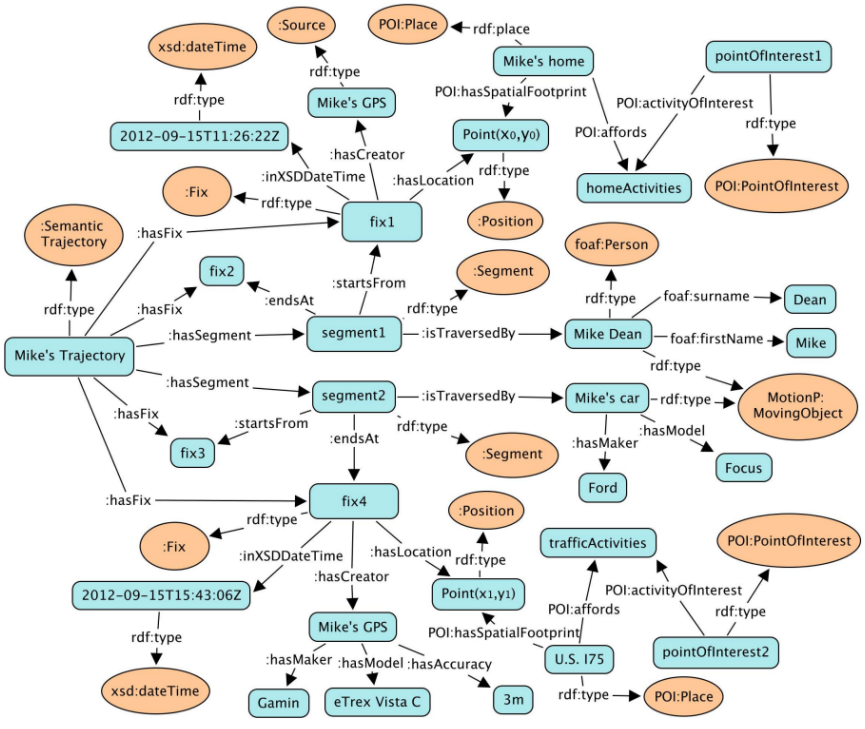
Event

If we move up from Student, we observe that a Student can participate in an event. The motivation for including this entity type is rooted in the project's purpose, which is to analyze student behavior. An important aspect of this analysis could be to observe events occurring at specific locations. We define an event as occurring at a specific location and involving one or more persons. The entity type 'Event' is linked to 'POI' through the object property 'happensAt'. This connection is established because, for the purpose of this research, analyzing events occurring at specific locations is of particular interest.
Although 'Event' and 'StayPoint' share many similar properties, a crucial distinction exists between these entity types due to their distinct roles in understanding and analyzing different aspects of student behavior. An Event is defined as occurring at a specific location and involving one or more persons, focusing on activities at particular locations. A 'StayPoint' represents a spatial-temporal instance where a person spends time in a specific location. 'StayPoints' capture instances of prolonged stays by individuals in specific spatial-temporal locations, helping analyze student behavior in terms of duration and location. While Events observe activities at specific locations, StayPoints emphasize instances of extended stays.

B Geo-ontology Design Pattern

The following two figures, retrieved from [4] served as inspiration for SU2 ontology that was created in this project.





C WorldKG Ontology