Representing temporal vagueness on the semantic web for historical datasets

Fabian Witeczek[2749025]

Vrije Universiteit, Amsterdam, The Netherlands

Abstract. Currently, time ontologies on the semantic web do not support concepts of vague temporal data adequately. In the context of this research, an ontology has been created that can represent various forms of vague dates. We present said ontology in this paper. Furthermore, we present the ontology's development process and the related design decisions. The development process started with collecting data records containing temporally vague dates. The records were obtained from the LOD datasets within the domain of digital humanities. The occurrences of vagueness were evaluated, and categories of vagueness were defined. A survey was conducted with domain experts in the digital humanities domain. The experts were questioned about their problems when working with temporally vague dates. The survey results confirmed the meaningfulness of the ontology requirements and the categories of vagueness which were: *Unknown deviation*, *within a time span*, *before or after a specific date*, *date options*, and *complete vagueness*. After that, the ontology was designed and implemented. Lastly, the ontology was tested and evaluated by linking its instances to instances of a historical dataset. This research concludes that the presented vague date ontology offers a clear and easy-to-use way to specify how vague dates are and in which regard. However, the ontology requires much effort to make it work in practice for researchers in digital humanities. This is due to precision and deviation values that need to be set for every record within the datasets. Another limitation of the practicality of the ontology is the design decision to have years as the finest time granularity.

Keywords: Temporal Vagueness · Ontology Engineering · Time Ontology.

1 Introduction

Time or, more specifically, dates are a crucial part of Linked Open Data(LOD) datasets that deal with storing historical data. They indicate when an event took place and are essential for historical classification. However, historians are not always in agreement as to when *exactly* something happened, typically because some events are difficult to reconstruct. This lies in the nature of history: it lies in the past, and insufficient documentation inevitably leads to speculation and uncertainty. This, in turn, leads to temporal vagueness. In the field of digital humanities, this phenomenon manifests itself by the fact that entries in datasets containing historical data partly contain vague dates like "ca. 1880", "1681?" or "na 1854"(meaning "after 1854" in Dutch)[1]. Each of those statements indicates vagueness differently, meaning that the precision level varies and has a different meaning semantically. Therefore, a knowledge gap exists since historians creating new records may know how precise the respective vague dates are without conserving this information.

This research aims at answering the question of what temporal vagueness is and how the handling of temporal vagueness can improve on the semantic web. To understand that, occurrences of temporal vagueness within the two digital humanities LOD datasets, Europeana[2] and BiographyNet[3], will be collected. Based on this data, it will be determined what the occurring statements of vagueness mean and which categories of vague dates can be derived from that. Subsequently, a survey will be conducted with experts in the digital humanities domain to identify the main problems when working with temporally vague data and whether the previously defined categories are able to solve them. Then, a prototype for an ontology will be presented, which is based on the insights gained in the data analysis and the feedback from the conducted survey. This ontology is designed to enable associations of events to vague date entities on the semantic web along the conventional date entities. In the end, the ontology is evaluated with competency questions specified in the requirement gathering phase of the research.

2 Related work

2.1 Temporal information extraction

How to extract temporal information about events and how to represent them semantically is discussed by de Boer et al. Documents containing temporal information(in this case, year numbers) of a specific historical event are retrieved from the web. In the next step, the year numbers get extracted from the documents. After normalizing the year distribution, it is converted into a semantic representation[3]. The methodology presented in this paper was thought to serve as an alternative for gathering the data points needed to find out how vagueness occurs in the LOD datasets. That approach was not necessary but could be relevant for future extensions of this research.

2.2 Vagueness and uncertainty

The related work about temporal vagueness can be split into two categories: Work about temporal vagueness and uncertainty in general(mainly from the

[1] taken from `http://data.biographynet.nl/yasgui/index.html`, used query:

```
SELECT * WHERE {
?event bgn:date ?date
} LIMIT 100
```

[2] `https://www.europeana.eu/en/about-us`

[3] `http://www.biographynet.nl/`, `http://data.biographynet.nl/home`

domain of logic and mathematics) and work concerning vagueness and uncertainty within historical data, also touching upon LOD. The following two papers presented count to the former category:

What uncertainty and vagueness are as well as their meaning semantically and how it occurs within points in time, time intervals, and durations is discussed by Billiet et al [2]. Different types and definitions of fuzzy time intervals are presented, which helped identify how the different occurrences of vagueness differ and where the boundaries for the defined vagueness categories are. Furthermore, it made clear that a limitation of granularity regarding the dates is useful for this research. Specifically, this means that only years are considered, and finer components like months, days, and time are ignored.

An interval-based temporal logic is introduced and presented by Allen as well as how to express temporal intervals effectively and especially for imprecise temporal representations [1]. The work shows how the *when* of happenings is typically described and how the precision of the date in that context is affected. The importance of distinguishing between time instants and time intervals is emphasized as well. Those aspects were important in the analysis phase when defining the vagueness categories and making the decision to ignore high granularities of dates.

Vagueness and uncertainty within historical data are discussed in the following two papers:

Sugimoto discusses how Linked Open Data(LOD) time entities are created as part of the LODE(Linked Open Date Entities) project. Temporal vagueness is touched upon, and the structure of LODE is explained [9]. LODE is a great reference point for understanding how dates are represented and used on the semantic web. The design decisions are partially adopted since vague dates still are dates and can be structured accordingly.

An interval-based temporal model that is capable of representing imprecise temporal knowledge is presented by [7]. Furthermore, important aspects like temporal uncertainty, subjectivity, and vagueness within historical data were discussed, which helped in the process of forming the categories of vagueness. Beyond that, problems and design decisions for an ontology model with such data have been identified and helped this research in the ontology design phase.

2.3 Ontology design

Contreras et al and Fernández-López et al present methodologies and processes for creating ontology designs. The processes and methods presented in those articles were used for designing the ontology of this research[4, 5]. Furthermore, [5] goes into detail about how the evaluation of an ontology is supposed to be. After designing the ontology using those resources, the next step is to implement the ontology. [8] presents best practices such implementations and serves as a more technical guide whereas [4] and [5] are conceptual guides.

A data model with uncertain historical time as well as different types of date entities are presented by [6]. The model is structured using a Unified Modeling

Language(UML) class diagram. Relevant classes/entities for this research have been identified and adopted for the ontology design.

3 Data collection

This section describes how records of historical data in which the dates are formulated vaguely were collected for this research. Those records will serve as the basis for the following analysis phase, where the categories of vagueness will be determined. The LOD datasets Europeana[4] and BiographyNet[5] have been selected for this purpose because they provide a sufficient amount of records in which temporal vagueness occurs.

In total, I collected 97,722 records from BiographyNet and Europeana using their respective SPARQL endpoints and processed the results with Python. 15,418 records were obtained from BiographyNet and 82,304 records were obtained from Europeana.

3.1 BiographyNet

Listing 3.1 shows the SPARQL query used for obtaining the records from the BiographyNet endpoint. The `<STATEMENT>` placeholder in line 7 of the Listing stands for the respective statement filtered for with the query. The `filter` function interprets the separate statement as a regular expression. This means that records containing the statement within its `date` object are collected.

For Biographynet, the statements ”ca”, ”-”, ”na”, ”voor”, and ”?” have been used, whereas for Europeana just ”ca” and ”?”. The reason for that is a lack of results when querying for the statements ”na”, ”voor”, and ”-”. The statements ”na” and ”voor” mean ”after” and ”before” in Dutch, respectively. The same statements translated into other languages didn’t return any results, therefore they are not part of this research and are not present in the data collected. English, French and German variants of the statements were used to ensure sufficient coverage of European languages.

The RDF predicate `bgn:date` has been used, as can be seen in line 6 of Listing 3.1. A total of 55,126 triples use this predicate[6] from which 15,418 contain the mentioned statements of vagueness. Other time-related predicates are: `bgn:when`, `bgn:from`, `bgn:to`, `bgn:notAfter`, and `bgn:notBefore`. Although those predicates represent dates just like `bgn:date` does, triples using them did not contain any statements of vagueness whatsoever. Therefore, the data collected from BiographyNet for this research only takes the `bgn:date` predicate into account.

[4] `http://sparql.europeana.eu/`

[5] `http://data.biographynet.nl/yasgui/index.html`

[6] according to `http://data.biographynet.nl/browse/list\_predicates?graph=bioned`

```
1  # other prefixes
2  PREFIX bgn: <http://data.biographynet.nl/rdf/>
3
4
5  SELECT * WHERE {
6    ?event bgn:date ?date
7    filter(regex(?date , "<STATEMENT>", "i" ))
8  }
9  LIMIT 1000
```

Listing 3.1. SPARQL query used for the BiographyNet endpoint

3.2 Europeana

The query used to collect the data from the Europeana endpoint can be seen in Listing 3.2. It is structured differently than the query used for the BiographyNet endpoint, mainly due to the constraint of only being able to fetch 10,000 results at most from the endpoint. To get all results rather than just the first 10,000, the nesting of another `SELECT` statement is needed for the `OFFSET` to work. However, the filtering mechanism in line 15 is the same as in line 7 of 3.1. Therefore both queries filter the results in the same way.

```
1  # prefixes
2
3  SELECT ?title ?creator ?mediaURL ?date
4  WHERE {{
5
6  SELECT DISTINCT ?title ?creator ?mediaURL ?date
7  WHERE {
8    ?CHO edm:type ?type;
9        ore:proxyIn ?proxy;
10       dc:title ?title ;
11       dc:creator ?creator;
12       dc:date ?date .
13   ?proxy edm:isShownBy ?mediaURL .
14
15   filter(regex(?date , "<STATEMENT>", "i" ))
16 }
17 ORDER BY ?date
18 }}
19 OFFSET 50000
20 LIMIT 10000
```

Listing 3.2. SPARQL query used for the Europeana endpoint

The results of both queries were exported as a CSV file. For BiographyNet, the selected values `event` and `date` were exported, for Europeana the values `title`, `creator`, `mediaURL`, and `date`. Those files are available in the `data-`

`collection-and-analysis-phase` directory of my public GitHub repository [10]. How they get processed is discussed in the next section.

4 Data analysis

Section 3 addressed how and according to which criteria the vague temporal data has been collected from the datasets. The results of the queries were exported as CSV files. The content of the CSV files is structured differently across the two datasets. To unify the two structures and to be able to process the data further, a jupyter notebook is used. The `pandas` library[7] offers all the necessary tools, and the vague temporal dates end up in one main python list. The code for achieving this can be found in the `data-collection-and-analysis-phase` directory of my public GitHub repository [10].

4.1 Categorization

But even in this state, the semantic meaning of the occurrences of temporal vagueness within the data cannot be analyzed. This is because the date notations vary not only across the two datasets but also within the respective dataset. For example, those four date notations and many more occur within the data collected from the Europeana endpoint, while implying the same statement/thing: "`circa 1970`", "`1967 ca`", "`1968, circa`", "`1970 [ca]`".

For this reason, five categories have been defined. First, the data at hand was examined manually. The semantic meaning of vague dates has been derived from their record's context. Those meanings were then grouped as categories of vagueness. The vague dates only occurred as an instance of one and only one of the five defined categories within the records. Each category represents a different type of statement that the notation of vagueness implies. The main difference between those categories is the time range that the vague statements imply. For example: the statement `before 1870` implies "occurence in any year, considering all years before 1870" and therefore has a much wider time range than `ca. 1870 - 1875`, assuming that both statements aim to represent a point in time.
According to those categories, each date represents a point in time that either

1. has an unknown deviation,
2. is within a time span,
3. is before or after a specific date,
4. is one of two date options, or
5. is completely vague

These categories will be examined more closely in the following subsections.

[7] `https://pandas.pydata.org/`

Unknown deviation This category represents dates that are uncertain according to [7]. This means, that a concrete date is provided together with an indication of uncertainty. In this case, this is either a variation of the `ca` statement or a question mark. This indication signalizes that the actual date may deviate from the specified date. Examples for dates belonging to this category: `ca. <YEAR>` , `<YEAR>?`. It is assumed that the two example cases imply the same degree of uncertainty, regarding the occurence time range. To make sure that different constellations of the elements `ca`, `?`, and year are considered valid, I created the regular expression as can be seen in Listing 4.1.

```
1 HAS_CA_AND_YEAR = ".*(ca|\d{3,4}).*(ca|\d{3,4}).*"
2 NO_DOUBLE_CA_OR_YEAR =
3 "^(?!.*(ca.*?(ca)))(?!.*(\d{4}.*?(\d{4}))).*$"
4 MATCH_YMD_FORMAT= ".*(\d{4}-\d{2}-\d{2}).*(Ca|ca).*"
5 YEAR_QM = ".*\d{4}\?.*"
```

Listing 4.1. Regular expressions for the unknown deviation category

The first expression in line 1 of Listing 4.1 matches dates that include at least one `ca` and at least one 3 or 4-digit number, in this case a year. The next expression in line 2-3 discards dates that have two `ca` statements or two years. A date having two years and one `ca` statement belongs to the fourth "date option category" since the `ca` statement is related to either one of the two years, hence depicting an option. An example for such a case is a record such as `ca. 1880/1881`.

The expression in line 4 matches dates that are formatted as `YYYY-MM-DD` with a subsequent `ca` statement. A significant amount of records had this specific structure. The last expression in line 5 matches all years with a subsequent question mark.

Within a time span Dates belonging to this category indicate a time span. The point in time at which the **event** took place lies within this time span. Examples for dates belonging to this category: `ca. <YEAR> - <YEAR>` , `<YEAR>-?-?` , `<CENTURY>th century`. Each of the three example cases indicates a time span. Similar to the unknown deviation category described in the previous Section 3.1, different notation and constellations of such time spans exist. Likewise, I created three regular expressions to define how the dates are supposed to be structured in order to be valid for this category. The regular expressions can be seen in Listing 4.2.

```
1 CA_YEAR_DASH_YEAR = ".*(Ca|ca).*\d{4}.*-.*\d{4}.*"
2 YEAR_QM_QM = ".*\d{4}-\?-\?.*"
3 CENTURY =
4 ".*(de eeuw|century|Century|th|jahrhundert|Jahrhundert).*"
```

Listing 4.2. Regular expressions for the time span category

The first expression in line 1 matches all date records that are structured as a `ca` statement followed by a year, a dash, and another year. The second expression in line 2 matches dates that are formatted as YYYY-MM-DD where the month and day is represented with a questionmark. For example: 1880-?-?. In this case, the time span would be from 1880-01-01 to 1880-12-31. The last expression in line 3-4 matches all keywords that mean century in Dutch, English and German. There were no occurrences of other languages indicating a century.

One might argue that the dates qualifying for this category do not mean a point in time within the time span but rather represent the date as a time span where the boundary dates are uncertain. For example: the notation ca.1880-1885 might represent the *time span* from circa 1880 to circa 1885, as opposed to a *point in time* between 1880 and 1885.
Time spans and uncertainty associated with them are not taken into account for this research. This is because the context of the data collected is the domain of digital humanities. Artifacts like paintings, pieces of music etc. typically have a point in time that indicates their creation. Therefore, it is assumed that time spans displaying uncertainty stand for points in time where the deviation is narrowed down to the specified time span. This also makes the relationship to the first category clearer.

Before or after a specific date Dates having a before or after statement followed by a year belong to this category. For example: before 1910, after 1861. Listing 4.3 shows the regular expression for matching those dates.

```
BEFORE_AFTER = ".*(before|after).*\d{4}.*"
```

Listing 4.3. Regular expression for the before or after category

Date options Dates that consist of two years separated by a slash symbol belong to this category. An example for such a date is: 1855/1858. The / between the years indicates that the exact date is either the first or the second option, in this case either 1855 or 1858. Listing 4.4 shows the regular expression for matching those dates.

```
YEAR_OR_YEAR = ".*\d{4}.*\/.*\d{4}.*"
```

Listing 4.4. Regular expression for the date option category

Complete vagueness Dates that are completely vague belong to this category. The only observed date form belonging to this category is a question mark representing the date. With this form, no meaning is transported since no context or time span can be derived from a question mark alone. Listing 4.5 shows the regular expression for matching those dates. The part [^a-z0-9] of the regular expression serves the purpose to eliminate dates that have characters other than white space before and after the question mark. This way, dates like 1880-?-? get excluded from this category.

```
QM = "^[^a-z0-9]*\?[^a-z0-9]*$"
```

Listing 4.5. Regular expression for the complete vagueness category

All of the regular expressions shown above deliver satisfactory results. Every categorized record was assigned the proper category representing its semantic meaning correctly. Manual checks of the categorizations confirmed that. However, this is only true for the data at hand. For example, date records from different datasets with different syntactic structures may be mismatched or not matched at all. In such a case, further regular expressions need to be constructed.

4.2 Analysis results

Each record collected in the data collection phase of this research was filtered with the regular expressions presented in the previous subsections. Each match of a regular expression puts the record in its respective category. Table 1 shows at what proportion the collected records were categorized and how many absolute occurrences there were for each category.

No record belongs to more than one category. It is notable that with 34.77% the *date options* category makes up a large portion of the occurrences within the Europeana dataset. In contrast, the BiographyNet dataset only has 5 total occurrences of date options which translates to a percentage of 0.03%. The situation is similar with the *complete vagueness* category: only 1.26% of records within the Europeana dataset belong to this category whereas 34.76% of BiographyNet's records belong to this category. Records that cannot be assigned to one of the mentioned categories belong to the *miscellaneous* category. 37.93% of Europeana's records are in this category, whereas the BiographyNet dataset has a similar amount of records in the *complete vagueness* category.

In summary, over a third of both datasets' records are not clearly assignable to a date category other than *complete vagueness* or *miscellaneous*. Europeana consists of less records belonging to the *within a time span* and *unknown deviation* categories but over a third of it's records belong to the *date options* category which is represented very small by BiographyNet. All in all, the results between the datasets are quite similar taking into consideration that they differ in size drastically.

However, the two datasets differ in one respect: its vague date occurrences in proportion to the non-vague date occurrences. `bgn:date` is the predicate used by BiographyNet to link various instances with date instances. In the BiographyNet dataset, 55,126[8] unique triples exist that use this predicate. Therefore, about 27.97% of dates present in the BiographyNet dataset are vague. The Europeana

[8] retrieved from `http://data.biographynet.nl/yasgui/index.html` using SPARQL query:
```
 SELECT (COUNT(*) as ?Triples)
FROM <http://data.biographynet.nl/rdf/>
WHERE { ?s bgn:date ?o }
```

Table 1. Category distribution of dates for both datasets

	Occurences BiographyNet		Occurences Europeana	
Date category	Absolute	Percentage	Absolute	Percentage
Unknown deviation	5,007	32.48%	16,047	19.50%
Within a time span	3,421	22.19%	5,353	6.50%
Before or after	0	0%	33	0.04%
Date options	5	0.03%	28,621	34.77%
Complete vagueness	5,359	34.76%	1,033	1.26%
Miscellaneous	1,626	10.55%	31,217	37.93%
Total categorized/records	15,418	100%	82,304	100%

dataset, on the other hand, contains 51,605,349[9] triples using the `dc:date` predicate which is the equivalent to BiographyNet's `bgn:date` predicate. Therefore, only about 0.16% of the dates occurring in Europeana's dataset are vague dates which is an enormous contrast. Considering that BiographyNet has significantly more vague date occurrences compared to Europeana's dataset but also in absolute numbers, this research is an opportunity to add value to BiographyNet's structure and data.

5 Design requirements

In the previous Section 4 we discussed how vague dates occur within the BiographyNet and Europeana datasets and to which semantic categories they belong.

Creating an ontology which enables the allocation of LOD resources to vague date entities of the new ontology addresses the problems mentioned in Section 1. A survey was conducted to be sure that those problems correspond to the problems that researchers in the domain of digital humanities have. 8 digital humanities domain experts participated in the survey. Another aim of the survey is to determine which categories of vagueness are most relevant for the domain experts in practice, based on their feedback.

5.1 Survey structure

The 16 questions in total were related to the categories of temporal vagueness as described in Section 4.

[9] retrieved from `http://sparql.europeana.eu/` using SPARQL query:

```
 SELECT (COUNT(*) as ?Triples)
FROM <http://data.europeana.eu/>
WHERE { ?s dc:date ?o }
```

First, a use-case is described. It states that the respondent wants to find out whether a certain event is associated with a vague date. Those vague dates are example representations of the categories of vagueness. This is one of the use-case descriptions taken from the survey: "*I would like to know whether the date for the creation/happening of a certain artifact/event(which is a point in time) lies between two dates(in other words: within a timespan).*". Second, it is asked how often the respondent is confronted with the described problem in his daily work. A 7-point likert scale presents the answer options for this question ranging from "Strongly disagree" to "Strongly agree". Finally, the respondent is asked which inconveniences he encounters when trying to solve the stated problem. More specifically, it is asked whether finding a solution for the problem is *inefficient*, *time-consuming*, *prone to error*, *resulting in inaccurate results*, *requiring technical work-arounds*, *requiring misuse of one or more systems*, or *straightforward*. Multiple choices are possible.

5.2 Survey results

The categories *vague time span*, *before or after a specific date*, and *unknown deviation* are most present in their daily work according to the domain experts. At the same time, the occurrences of those categories involve a lot of inconvenient work. The domain experts agreed that they are confronted with the mentioned categories often: 87,5% agreement for the *vague time span*, *after a specific date*, and the *unknown deviation* categories. The *before a specific date* category has a slightly lower agreement of 75%. Agreement in this case means all responses where either "Somewhat agree", "Agree", or "Strongly agree" was selected as the answer for the first question.

The second question concerned the inconveniences encountered with each category of vagueness. On average, there were 14.7 inconveniences for the categories *vague time span*, *before or after a specific date*, and *unknown deviation*. The answer possibilities "straightforward" and "I am never/very rarely confronted with this question" were excluded for this average. Having 14.7 inconveniences per category means that every domain expert has at least one and almost two inconveniences per category on average. Considering that those categories occur often in their daily work as well, those categories will be the focus of the ontology design. The categories *Date options*, *Complete vagueness*, and *Other* were significantly less present in the daily work of the domain experts. Nevertheless, they will be considered in the ontology design for the sake of completeness. However, all requirements, queries, and tests for the evaluation will be geared towards the other categories.

The complete survey including its results can be found at [10].

5.3 Ontology requirements

The use-case descriptions presented in the survey were formulated in such a way that they can be used as competency questions for the ontology design. According to [4], competency questions are questions that the ontology system should be able to answer once the implementation is complete. In this case, answers to those questions will be obtained through executing SPARQL queries. The SPARQL queries will deliver results that answer the questions.

A list of all competency question is presented in Table 2. They are used for evaluating the final ontology in Section 7.

Table 2. Complete list of competency questions

Competency question identifier	Competency question
CQ1	Which events are associated with a *vague time span* where the boundary years are 100% correct?
CQ2	Which event are associated with a *vague time span* where the span is temporally before the year 1900?
CQ3	Is a specific event associated with a *vague time span*?
CQ4	Which events are associated with a *circa date* where the deviation is ≤ 5 years?
CQ5	Which events are associated with a *circa date* where the precision is ≥ 0.7 ?
CQ6	For a specific *circa date* with a precision of ≥ 0.9, which events are associated with it?
CQ7	Which events are associated with a *before date* where the date lies before 1900?
CQ8	For a specific *before date*, which events are associated with it?
CQ9	Which events are associated with an *after date* where the date lies after 1900?
CQ10	For a specific *after date*, which events are associated with it?

6 Ontology design

This section describes and justifies the structure of the ontology which has been created in the context of this research. It will be referred to as the *vague date ontology* from now on. The vague date ontology has been created using a software tool called Protégé[10]. Protégé enables the specification of classes, object and data properties, constraints, and many more. The result is exported as a `.owl` file where those specifications are defined as a set of RDF triples. The ontology consists of 198 triples in total and can be found in the `ontology-design-phase` directory of my GitHub repository[10]. Another tool called WebVOWL[11] was

[10] `https://protege.stanford.edu/`

[11] `http://vowl.visualdataweb.org/webvowl.html`

used to create a visualization of the ontology which can be seen in Figure 1. The basis for the visualization is the set of RDF triples exported using Protégé.

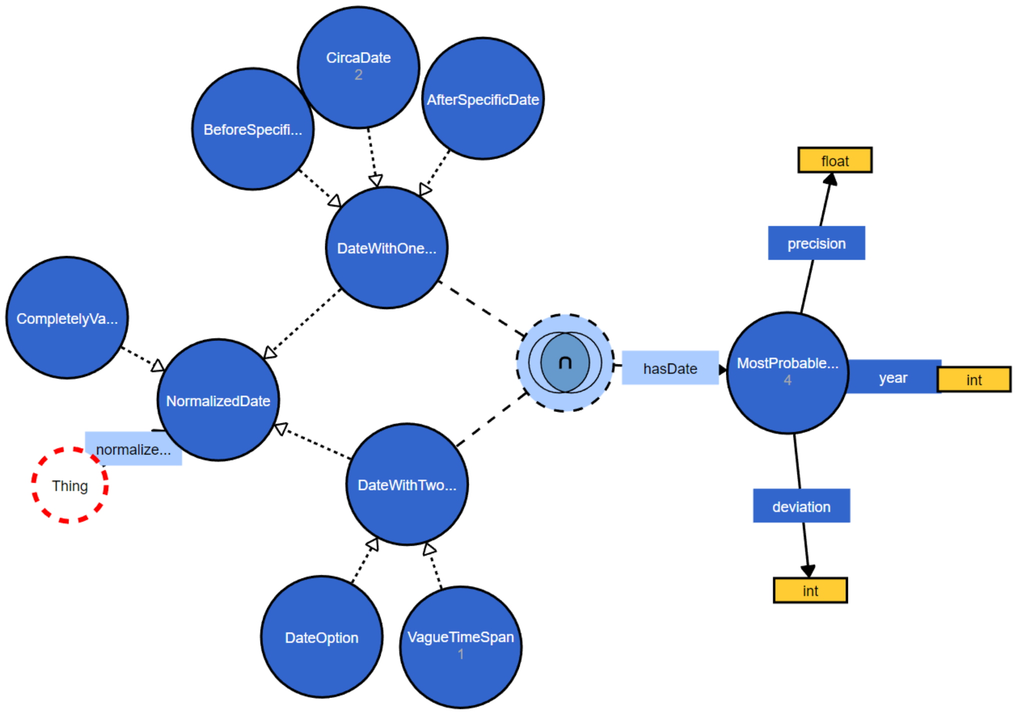


Fig. 1. Visualization of the vague date ontology

The class `owl:Thing` represents any class or instance and can be seen in the bottom left of Figure 1. It can have the object property `normalizedDate` which is the external entry point to the vague date ontology. Each `normalizedDate` object property is linked to the `NormalizedDate` class, which is an abstract superclass. The instances that a `Thing` will actually be linked to are of type `CircaDate`, `BeforeSpecificDate`, `AfterSpecificDate`, `DateOption`, or `VagueTimeSpan`. Those concrete classes in turn have the `hasDate` object property which links them to an instance of a `MostProbableDate`. This class is the carrier of the `NormalizedDate`'s date. For example: The statement "ca. 1900" which could be represented as a `NormalizedDate` and more specifically, as a `CircaDate`, would have the `hasDate` object property pointing to a `MostProbableDate` instance where its `year` data property equals `1900`. The `CircaDate` instance can then be linked to any `Thing` and the connection between an instance of an external ontology and an instance of the vague date ontology is made. Furthermore, the `MostProbableDate` class offers the data properties `precision` and `deviation`. `precision` means the probability of the specified `year` data property to be the historically correct year and is stored as a `float`. `deviation` means the maximum of years that the specified `year` might deviate from the historically correct date, in both the past and the future. Lastly, there are two distinctions concerning `NormalizedDates`. `NormalizedDates` either have one or two year specifications. For example: "ca. 1900" indicates one year, whereas "ca. 1880 - 1900" indicates two. Therefore, the classes `CircaDate`, `BeforeSpecificDate`, and `AfterSpecificDate` are a subclass of `DateWithOneYearSpecification` and the classes `DateOption` and `VagueTimeSpan` are subclasses of `DateWithTwoYear Specifications`. The background is that a `DateWithTwoYearSpecifications` like a `VagueTimeSpan` necessarily needs two `hasDate` object properties pointing to two `MostProbableDate` instances. On the other hand, a `DateWithTwoYear Specifications` only needs to be linked to one `MostProbableDate`. Therefore this distinction has to be made and a constraint is restricting subclasses of `DateWithOneYearSpecification` to have more than one link to a `MostProbable Date`. Conversely, `DateWithTwoYearSpecifications` instances cannot be linked to just one `MostProbableDate`, but exactly two.

In summary, the vague date ontology consists of three layers: The external `Thing` that wants to be linked to a `MostProbableDate` with the help of a `NormalizedDate`. The `NormalizedDate` communicates how the `Thing` relates to the `MostProbableDate` by defining the vague date category and its associated properties and constraints. With the help of SPARQL queries, it will then be possible to query for specific vague date types like `CircaDate` or `VagueTimeSpan` as well as for properties of the `MostProbableDate` class like `precision`. Those queries will be presented in Section 7.

Having the two data properties `precision` and `deviation` within the `Most ProbableDate` class solves the problem of historians being not sure what certain representations of vague dates truly mean. A vague date having a `precision` of `0.95` is a clear indication that the stated year is 95% precise. This leads to better assessments if the data is the only reference point. It also allows queries for certain degrees of precision and deviation, which are the parameters that ultimately make up vagueness. However, there are two main problems with using this ontology system in the context of historical datasets. First, each `precision` and `deviation` value has to be defined for each historical record. Second, the vague date ontology instances holding the respective values have to be created and linked to the desired historical records. Both tasks are a lot of work for the digital humanities researchers to carry out in order to make this solution effective in practice.

7 Ontology testing with historical data

Now that the ontology is set up and exported in the form of RDF triples, the next step is to test its application with actual historical data. This section describes how the historical data collection was carried out and how the data and the vague date ontology were linked. Finally, the resulting ontology will be evaluated using the competency questions set up previously in Section 5. A SPARQL query has been constructed for each question, designed to answer the particular question.

Data from the Europeana dataset is used as test data that will be linked to instances of the vague date ontology. For this purpose, the software tool GraphDB[12] will be used. GraphDB is a RDF database which allows to import different datasets and to manipulate them locally. This is a simple solution to link instances of the Europeana dataset to instances of the vague date ontology. The complete data and the final GraphDB repository can be found in the `ontology-testing-phase` directory of my public GitHub repository[10].

7.1 Data collection

First, data has to be collected from the Europeana SPARQL endpoint a second time. Section 3 presented how the first set of data was collected for the initial data collection phase of this research. The problem with this collected data is that it only consists of individual objects and is in a `.csv` format. The relationships between those objects are not apparent from the data since those relationships have not been queried for. In essence, the first data collection focused on retrieving entries from the Europeana dataset whereas now we need to retrieve the structure of the dataset. Hence, the properties connecting the objects need to be collected. Listing 7.1 shows how that was done.

```
CONSTRUCT {?CHO dc:title ?title . ?CHO dc:creator ?creator.
?CHO ore:proxyIn ?proxy . ?proxy edm:isShownBy ?mediaURL .
?CHO dc:date ?date}

WHERE {
  ?CHO edm:type ?type;
      ore:proxyIn ?proxy;
      dc:title ?title ;
      dc:creator ?creator;
      dc:date ?date .
  ?proxy edm:isShownBy ?mediaURL .

  filter(regex(?date , "<STATEMENT>", "i" ))
}
ORDER BY ?date
LIMIT 1000
```

Listing 7.1. SPARQL query for constructing Europeana's cultural heritage objects

In contrast to the first data collection as presented in Listing 3.2, Listing 7.1 shows that a `CONSTRUCT` instead of a `SELECT` statement is used. The difference between those two is that `SELECT` simply selects the values of the variables that are represented by the `?`. `CONSTRUCT`, on the other hand, takes two variables and connects them with a predicate. For example: `?CHO dc:date ?date`. In this case, the cultural heritage object(short: CHO) gets connected to the `?date`

[12] `https://www.ontotext.com/products/graphdb/`

variable that has been retrieved in the `WHERE` clause. `dc:date` is the predicate that connects the two instances. `dc:date` can be seen as the Dublin Core[13] equivalent of the vague date ontology's `normalizedDate` predicate.

The result of the SPARQL query presented in figure 7.1 is a turtle `.ttl` file where all triples created through the `CONSTRUCT` clause are stored. This `.ttl` file is imported through GraphDB. Likewise, the `.ttl` file of the vague date ontology is imported and GraphDB is set up as the test environment.

7.2 Creating ontology instances

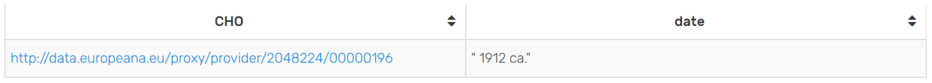
Now, instances of the Europeana dataset and the structure of the vague date ontology are imported in GraphDB. What is missing now are concrete instances of the vague date ontology classes.

Instances like `CircaDate`s and `MostProbableDate`s need to exist in order to link them to CHO objects. To find out which year and vague date type is needed, the SPARQL query shown in Listing 7.2 is executed. It selects all `CHO` instances together with their date instances.

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select * where {
        ?CHO dc:date ?date.
}
```

Listing 7.2. SPARQL query for retrieving *CHO* instances together with their *dc:date*

One result of the query presented in Listing 7.2 can be seen in figure 2. The CHO is a unique URI that represents an exhibit or event of the Europeana dataset. The `date` indicates the year `1912` and `ca.`. This means that the date has to be represented as a `CircaDate` within the vague date ontology.



CHO	date
http://data.europeana.eu/proxy/provider/2048224/00000196	" 1912 ca."

Fig. 2. Result of SPARQL query from Listing 7.2

Because a `CircaDate` has to be linked to a `MostProbableDate`, the first step for creating a `CircaDate` is creating a `MostProbableDate`. The SPARQL query for doing so can be seen in Listing 7.3. The `:` symbol stands for the URI of the vague date ontology which is local currently. `:MostProbableDate1912` is the name of the new instance and `rdf:type :MostProbableDate` specifies that the instance is of type `MostProbableDate`. Furthermore, the statements in lines 6-8 set the data properties of the instances. The `deviation` and `precision` properties are set exemplary for the sake of demonstration.

[13] see `https://www.dublincore.org/specifications/dublin-core/dcmi-terms/`

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

INSERT
{
    :MostProbableDate1912 rdf:type :MostProbableDate;
                          :year 1912;
                          :deviation 5;
                          :precision 0.7 .
}
WHERE {}
```

Listing 7.3. SPARQL query for creating a new *MostProbableDate* instance

Now that the exemplary `MostProbableDate` instance `MostProbableDate1912` exists, a corresponding `CircaDate` instance has to be created and linked to it. Listing 7.4 shows a SPARQL query which does that. The new instance has the identifier `CircaDate1912` and is of type `CircaDate`. The `:hasDate` object property specifies that the `CircaDate1912` is linked to the `MostProbableDate1912` instance.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

INSERT DATA
{
    :CircaDate1912 rdf:type :CircaDate;
                   :hasDate :MostProbableDate1912 .
}
```

Listing 7.4. SPARQL query for creating a new *CircaDate* instance

7.3 Connecting ontology instances to the dataset

Now that a `CircaDate` instance exists which is linked to a `MostProbableDate` instance holding the proper year property, it is time to link the `CHO` instance to the `CircaDate1912` instance. This happens with the help of the `normalizedDate` object property as can be seen in Listing 7.5. The `CHO` URI, which is the reference to the unique identifier of the Europeana record, is connected to the `CircaDate1912` instance in this way.

```
INSERT DATA
{
 <http://data.europeana.eu/proxy/provider/2048224/00000196>
     :normalizedDate :CircaDate1912 .
}
```

Listing 7.5. SPARQL query for linking Europeana's *CHO* to a *CircaDate*

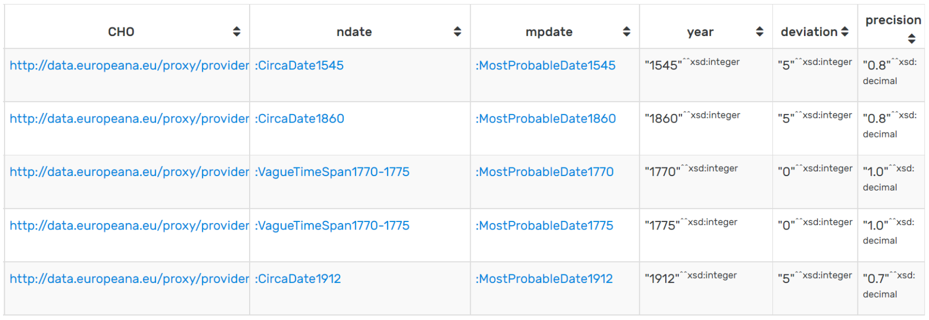
Listing 7.6 shows a SPARQL query that selects all CHOs having a `Normalized Date` connected through the `normalizedDate` predicate. Furthermore, their `Most`

`ProbableDate` instance get selected as well as its properties. This way, we can check whether all the previous steps were successful.

```
SELECT * WHERE {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :NormalizedDate .
    ?ndate :hasDate ?mpdate.
    ?mpdate :year ?year;
       :deviation ?deviation;
       :precision ?precision.
}
```

Listing 7.6. SPARQL query for selecting all linked events along with all date details

The result from this query can be seen in figure 3 and in fact, the last row shows our new instances and properties bound to the right `CHO`.



CHO	ndate	mpdate	year	deviation	precision
http://data.europeana.eu/proxy/provider	:CircaDate1545	:MostProbableDate1545	"1545"^^xsd:integer	"5"^^xsd:integer	"0.8"^^xsd:decimal
http://data.europeana.eu/proxy/provider	:CircaDate1860	:MostProbableDate1860	"1860"^^xsd:integer	"5"^^xsd:integer	"0.8"^^xsd:decimal
http://data.europeana.eu/proxy/provider	:VagueTimeSpan1770-1775	:MostProbableDate1770	"1770"^^xsd:integer	"0"^^xsd:integer	"1.0"^^xsd:decimal
http://data.europeana.eu/proxy/provider	:VagueTimeSpan1770-1775	:MostProbableDate1775	"1775"^^xsd:integer	"0"^^xsd:integer	"1.0"^^xsd:decimal
http://data.europeana.eu/proxy/provider	:CircaDate1912	:MostProbableDate1912	"1912"^^xsd:integer	"5"^^xsd:integer	"0.7"^^xsd:decimal

Fig. 3. Result of SPARQL query

7.4 Evaluation

The list below shows all competency questions together with the SPARQL queries that answer them.

CQ1 Which events are associated with a *vague time span* where the boundary years are 100% correct?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :VagueTimeSpan.
    ?ndate :hasDate ?mpdate.
    ?mpdate :precision ?precision.
    filter(?precision = 1.0)
}
```

CQ2 Which events are associated with a *vague time span* where the span is temporally before the year 1900?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
?ndate rdf:type :VagueTimeSpan.
    ?ndate :hasDate ?mpdate.
    ?mpdate :year ?year.
    filter(?year < 1900)
}
```

CQ3 Is a specific event associated with a *vague time span*?

```
ASK {
    <http://data.europeana.eu/proxy/provider/2048224/00004515>
        :normalizedDate ?ndate.
    ?ndate rdf:type :VagueTimeSpan.
}
```

CQ4 Which events are associated with a *circa date* where the deviation is ≤ 5 years?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :CircaDate.
    ?ndate :hasDate ?mpdate.
    ?mpdate :deviation ?deviation.
    filter(?deviation <= 5)
}
```

CQ5 Which events are associated with a *circa date* where the precision is ≥ 0.7 ?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :CircaDate.
    ?ndate :hasDate ?mpdate.
    ?mpdate :precision ?precision.
    filter(?precision >= 0.7)
}
```

CQ6 For a specific *circa date* with a precision of ≥ 0.9, which events are associated with it?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :CircaDate.
    ?ndate :hasDate ?mpdate.
    ?mpdate :precision ?precision;
```

```
              :year ?year.
    filter(?precision >= 0.9)
    filter(?year = 1912)
}
```

CQ7 Which events are associated with a *before date* where the date lies before 1900?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :BeforeSpecificDate.
    ?ndate :hasDate ?mpdate.
    ?mpdate :year ?year.
    filter(?year < 1900)
}
```

CQ8 For a specific *before date*, which events are associated with it?

```
select ?CHO where {
    ?CHO :normalizedDate :BeforeSpecificDate1700.
}
```

CQ9 Which events are associated with an *after date* where the date lies after 1900?

```
select ?CHO where {
    ?CHO :normalizedDate ?ndate.
    ?ndate rdf:type :AfterSpecificDate.
    ?ndate :hasDate ?mpdate.
    ?mpdate :year ?year.
    filter(?year > 1900)
}
```

CQ10 For a specific *after date*, which events are associated with it?

```
select ?CHO where {
    ?CHO :normalizedDate :AfterSpecificDate1860.
}
```

All queries deliver complete and valid results. Therefore, it is concluded that the practical test application of the vague date ontology passes the evaluation. Whether the application is practical in the a real-world scenario will be discussed in the next Section 8. All queries can be found in the `ontology-testing-phase` directory of my public GitHub repository[10].

8 Discussion

This paper presented a semantic web ontology capable of representing dates with various degrees of temporal vagueness.

Having properties that indicate the vagueness of dates is helpful, but there is a drawback. As discussed earlier, much effort has to be put into labeling existing vague dates to make using the vague date ontology viable for researchers. Another problem is that the precision and deviation values are impossible to define for some records. This may be because knowledge about such records is not available anymore, for example, if their creators have passed.

However, the vague date ontology can be extended in several ways to get more out of it: First, instead of having the integer datatype representing the year property of a MostProbableDate instance, references to LODE instances can be made. This makes sense because LODE is an established ontology for time, offering functionalities that would be needed with the vague date ontology sooner or later.

If the vague date ontology is not used in unison with LODE, its structure needs to be adjusted. This is because the vague date ontology works with years as the time unit with the finest granularity. This is not a good condition for being widely used across the semantic web. Essentially, the deviation property in the current vague date ontology needs to be transformed into a class. Then, properties within this class can define a date's granularity or its components, assuming a granularity is agreed on.

Machine learning offers possibilities to support the process of assigning vague dates to vague date categories. For example, a machine learning model can predict which category of vagueness a vague date statement should be assigned. This is a better alternative for categorizing occurrences of vagueness than the procedure presented in this research using regular expressions. However, records of historical data containing vague dates must be acquired on a much larger scale to create a representative machine learning model. On the other hand, the categorization of vague dates then is more universally valid because of the diversity of records.

However, the labeling process cannot be supported by machine learning applications. This is because assigning precision and deviation properties to events requires knowledge about the specific events. A machine learning application cannot provide such knowledge. The strengths of machine learning are learning and recognizing patterns, which is why the categorization of dates can be enhanced with its help.

9 Conclusion

This paper presented how temporal vagueness occurs within the LOD datasets BiographyNet and Europeana. String literals are the format in which vague dates are represented. The dates are structured in many different forms within those literals across the datasets.

We found out that the notations of vagueness have different meanings. Five meanings, to be exact: A vague date can imply an unknown deviation, a vague time span, an option between two dates, complete vagueness, or a date lying either before or after a specific date.

We found a way to represent temporal vagueness as LOD to be usable on the semantic web for the domain of digital humanities. The five meanings of vagueness were used as categories to which a vague date representation belongs. The vague date ontology was designed in which external events can be linked to instanced of each category. Such a category is then, in turn, linked to an instance explaining details about the date, such as its deviation and precision values. This way, those details get associated with the external events and vice versa. Ultimately, those details represent the vagueness of a date.

The vague date ontology in its current state is not a production-ready ontology. Instead, it serves more as a proof of concept but has the potential to become an efficient tool with the proper adjustments discussed in the previous section.

References

1. Allen, J.F.: Maintaining knowledge about temporal intervals. Commun. ACM **26**, 832–843 (1983)
2. Billiet, C., De Tré, G.: Combining uncertainty and vagueness in time intervals. In: Angelov, P., Atanassov, K., Doukovska, L., Hadjiski, M., Jotsov, V., Kacprzyk, J., Kasabov, N., Sotirov, S., Szmidt, E., Zadrożny, S. (eds.) Intelligent Systems'2014. pp. 353–364. Springer International Publishing, Cham (2015)
3. de Boer, V., van Someren, M., Wielinga, B.J.: Extracting historical time periods from the web. Journal of the American Society for Information Science and Technology **61**(9), 1888–1908 (2010). https://doi.org/https://doi-org.vu-nl.idm.oclc.org/10.1002/asi.21378, `https://onlinelibrary-wiley-com.vu-nl.idm.oclc.org/doi/abs/10.1002/asi.21378`
4. Contreras, M.C.B., Reyes, L.F.H., Ortiz, J.A.R.: Methodology for ontology design and construction. Contaduría y Administración (2019)
5. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: Methontology: From ontological art towards ontological engineering. In: AAAI 1997 (1997)
6. Matousek, K., Falc, M., Kouba, Z.: Extending temporal ontology with uncertain historical time. Comput. Informatics **26**, 239–254 (2007)
7. Nagypál, G., Motik, B.: A fuzzy model for representing uncertain, subjective, and vague temporal knowledge in ontologies. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds.) On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. pp. 906–923. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
8. Noy, N.: Ontology development 101: A guide to creating your first ontology (2001)
9. Sugimoto, G.: Building linked open date entities for historical research. In: Garoufallou, E., Ovalle-Perandones, M.A. (eds.) Metadata and Semantic Research. pp. 323–335. Springer International Publishing, Cham (2021)
10. Witeczek, F.: Temporal vagueness analysis on the semantic web (3 2022). https://doi.org/10.5281/zenodo.1234, `https://github.com/witefab/master-thesis-code`