Storing and Processing Personal Narratives in the Context of Cultural Legacy Preservation

Pierrick Bruneau, Olivier Parisot and Thomas Tamisier
Luxembourg Institute of Science of Technology, L-4362 Esch-sur-Alzette, Belvaux, Luxembourg

Keywords: Narrative Data Model, Narrative Knowledge Extraction, RESTful API.

Abstract: An important, yet underestimated, aspect of cultural heritage preservation is the analysis of personal narratives told by citizens. In this paper, we present a data model and implementation towards facilitating narratives storage and sharing. The proposed solution aims at collecting textual narratives in raw form, processing them to extract and store structured content, and then exposing results through a RESTful interface. We apply it to a corpus related to the time of the European construction in Luxembourg. We disclose details about our conceptual model and implementation, as well as evidence supporting the interest of our approach.

1 INTRODUCTION

An important aspect of cultural heritage preservation is the collection and collation of personal views and anecdotal stories of citizens. While the wide availability of social media will facilitate this work for future generations when they analyze our times, such means are not available for e.g. the European construction period (roughly 1945-1975). Adapted tools are needed to collect such testimonies by elderly people, as well as facilitate their collation and dissemination.

In this work, we focus on the time of the European construction in Luxembourg and the surrounding region. This work has been conducted in the context of a funded project in collaboration with elderly people organizations. In this context, witnesses of the time frame of interest (aged between 75 and 85 years old) have been interviewed, and their testimonies have been transcribed. In addition, this corpus has been enriched by extractions from online platforms - more details about this corpus may be found in Section 5.

Eventually, the collected corpus is merely more than a set of short texts. To make this narrative data actionable, e.g. allow effective indexing, browsing, or exploitation by web applications, knowledge extraction techniques are needed to extract relevant keys to these stories, such as people, places and time frames involved. Hence in this paper we focus on the means to store, process and access such narrative information. More precisely, a dedicated data model and a back-end server are needed in order to model and store the collected stories.

The rest of this article is organized as follows. Firstly, related work about narrative structures and knowledge is discussed in Section 2. Then, in Section 3, we define a data model appropriate for storing personal stories and narratives. Next, we describe a software pipeline to map raw text into the proposed data model. After describing the French data corpus alluded to earlier in the introduction, we disclose a RESTful architecture dedicated to exposing and sharing the data stored in machine-readable format in Section 5. After showing evidence of the interest of the consumption of this data by knowledge extraction facilities, we conclude with numerous perspectives.

2 RELATED WORK

The study of the structure of narratives and stories has been applied to a variety of domains, e.g. emergency response (Scherp et al., 2009), situational awareness (Van Hage et al., 2012), or collections of historical documents (Segers et al., 2011). A major concern in this domain is to bridge the gap between raw text (i.e. the form under which testimonial stories are generally acquired) and structured information with semantic value, that would enable story linking and advanced queries.

Associating properties to entities and composing them is the core concern of ontology engineering. Well known ontologies include YAGO (Suchanek

Bruneau, P., Parisot, O. and Tamisier, T.
Storing and Processing Personal Narratives in the Context of Cultural Legacy Preservation.
DOI: 10.5220/0006422800480055
In *Proceedings of the 6th International Conference on Data Science, Technology and Applications (DATA 2017)*, pages 48-55
ISBN: 978-989-758-255-4

et al., 2008), the Google Knowledge Graph (Google, 2012), and DBpedia (Mendes et al., 2011). These ontologies are often used in conjunction with controlled vocabularies such as Dublin Core (Weibel et al., 1998) or FOAF (Brickley and Miller, 2007), that facilitate the interoperability of data sources, most notably via the RDF representation language.

Rather than an ensemble of unrelated facts, narration implies relationships between atomic facts or events. (Scherp et al., 2009) define a taxonomy of such links (compositionality, causality, correlation and documentation). These links are relevant to our context, but they consider events at a coarse level with no controlled vocabulary of predicates. Similarly to (Van der Meij et al., 2010; Segers et al., 2011), they are mostly concerned by interoperability between different ontologies. Likewise, (Van Hage et al., 2012) emphasize link types, with little consideration of specific data structures for events. With close resemblance to the CIDOC CRM (Doerr, 2003), (Segers et al., 2011) define explicitly roles (also known as facets in (Mulholland et al., 2012)) applying to events (e.g. actors, dates and locations), which are appropriate for historical events in a broad sense (e.g. the French Revolution in (Segers et al., 2011)), but not for events as constituents of a first-person narrative. The objective is then to propose a standard metadata description space for historical artifacts, rather than exploring the structure of narration.

The contributions by (Zarri, 2009) are the most closely related to our work. In their *Narrative Knowledge Representation Language* (NKRL), they define data structures and controlled vocabularies of predicates and links to support the analysis of non-fictional and factual narratives. They avoid the use of the term *story* as it has led to ambiguities in the literature. Rather, They define a set of events and facts as the *fabula*. The *plot* level adds chronological, logical and coherence links between events. The *presentation* level is about the form in which plots are shown. Some related work in narrative analysis and storytelling is concerned with mapping arbitrary stories to a classical narrative structure (Tilley, 1992; Yeung et al., 2014). In our work, stories are potentially made of anecdotal testimonies, and as such cannot be expected to match these structures. More abstract properties, such as sentiment attached to stories, were also extracted in (Min and Park, 2016) in order to analyze the structure of books.

The way arbitrary text is remapped automatically to taxonomies of entity types, relationships and predicates is seldom considered in the literature. Some authors explicitly assume that this mapping has to be performed manually (Mulholland et al., 2012), or via crowdsourcing (Bollacker et al., 2008). Wikipedia page structure has also been exploited in (Suchanek et al., 2008). Alternatively, a term-based heuristic is used in (Gaeta et al., 2014) to determine links between events, and the use of *Natural Language Processing* (NLP) techniques such as *Named Entity Recognition* (NER) to automatically extract facts and events has been evaluated in (Segers et al., 2011; Van Hooland et al., 2015). Entity types in event models such as SEM (Van Hage et al., 2012) are closely related to types extracted by standard NER methods such as (Favre et al., 2005) (e.g. people, locations, dates).

3 NARRATIVE ENTITY MODEL

To suit the needs of the project described in the introduction, and put people and spatio-temporal coordinates at the center of narratives, we developed a simplified variant of NKRL (Zarri, 2009). In a nutshell, our model can be thought of as a database schema to enable storage and facilitate indexation of narrative data.

The root object type is denoted as *entity*, as a reference to the Drupal terminology, that supports our implementation of the model, described to further extent in Section 5. Except primitive types such as text and numbers, all non-primitive types (e.g. story) are specializations of this root object type. Entity labels in Figure 1 are meant to be unique. Entity references, i.e. references to other entities in the database (e.g. person referred to in a story) are underlined. Arrows denote typed dependencies (i.e. pointers depend on pointees), when other dependencies may refer to several kinds of entities. To emphasize the story-centric aspect of this model, most types, such as *person* and *location*, directly store references to stories that refer to them. This can be thought of as a kind of reverse index.

The data model in Figure 1 is heavily inspired by the model underlying NKRL (Zarri, 2009), but exhibits decisive distinctions. The proposed structure was designed with flexibility in mind. For example, it easily supports partial specification - a typical narrative may occasionally omit spatial and or temporal specifications. Similarly, the proposed custom date format supports loose specification. The *approximate* flag indicates whether the precision of the temporal bounds should not be accounted for, and all fields except year are optional. Both points and intervals in time can be described with the same format, simply by equaling *from* and *to* respective fields.

Most entities in the model (e.g. artifacts, people, places) may have alternative writings. This ambiguity

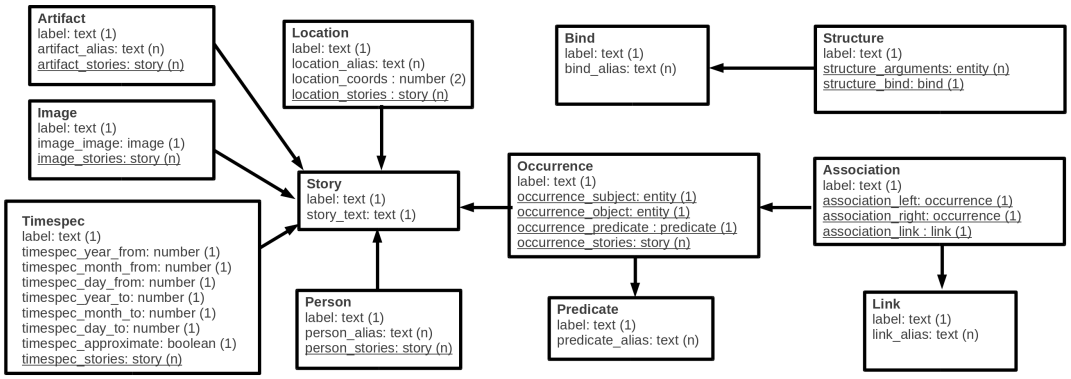

Figure 1: Simplified Narrative Knowledge Model. Entity types are denoted by boxes. The multiplicity of typed properties is parenthesized. References to entities are underlined, while arrows denote the dependencies implied by references whenever they are explicitly typed.

is handled by the *alias* properties in Figure 1, that hold all alternative writings for a given entity. Doing so will notably facilitate checking for duplicates when processing new stories.

Following our simplified schema, complex narrative structures are stored using deeply nested structures. This way of proceeding reminds of RDF-based ontologies, that rely on binary relations, and reification to represent more complex semantics. In the context of general ontologies, (Hoffart et al., 2013) indeed show that using reification, complex facts can be unfolded as several atomic facts. In brief, with reification, an instance of a binary relation $a\mathcal{R}_1 b$ can act as the argument of another relation, effectively allowing e.g. $(a\mathcal{R}_1 b)\mathcal{R}_2 c$. This design has been subject to debate in the literature. For example, (Zarri, 2009) advocates the usage of pure n-ary relations. Our eventual choice has been motivated by its greater flexibility, and better compliance with the technologies chosen for its implementation (see Section 5).

An importance terminological nuance lies between our schema proposition and those based on reified facts: in the latter, properties are linked to entities by *predicates*, when in our proposition the *predicate* is a full-fledged entity type, the purpose of which is to link two entities. This choice is justified by the syntactic and semantic considerations developed in Section 4, where *predicate* is generally understood as a synonym for *verb*.

With distinction to many works in ontology engineering which are mainly focused towards reasoning, i.e. inference of novel facts that can be deduced from the current fact base, in narration, the focus is not so much on deduction than on facilitating the access and the presentation of the data. This is linked to the consumption mode of the data, that is more contemplative for stories, if compared to other more economically involved domains, where actionable data is sought (Scherp et al., 2009; Van Hage et al., 2012).

The layout in Figure 1 has been chosen as to emphasize the nesting level of the entities. On the left the entities are close to the *physical* world (e.g. people, places), with primitive-typed properties mostly. The right of the schema displays higher-order entities, such as *associations* and *structures*, that tend to bind entities from the left-end side of the schema.

Figures 2 and 3 illustrate typical narrative data structures following our model. For the sake of clarity, merely entity labels are reported in figures, other fields are left implicit, and entity references are explicited by links. Figure legends report the original French text, while for convenience terms indicated in the diagrams are translated to English. The resulting data structures were constructed manually. The nested construction of complex structures from simpler entities discussed in this section is highlighted by shadings.

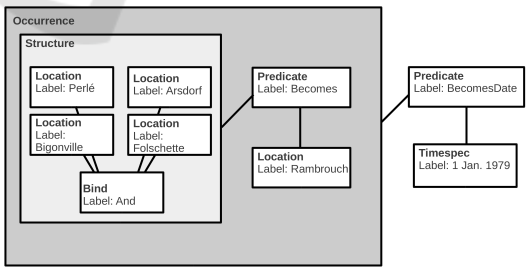

Figure 2: Data structure that can be associated to the following text: *Perlé fut une commune jusqu'au 1er janvier 1979, date à laquelle elle a fusionné avec les communes d'Arsdorf, Bigonville et Folschette pour former la nouvelle commune de Rambrouch.*

Inspired by (Zarri, 2009), in our data model entity types *predicate*, *bind* and *link* take values in closed vocabularies. These vocabularies are reported in Table 1.

In the example in Figure 2, the *coordination* bind

Table 1: Controlled vocabularies.

Predicates	Binds	Links
Behave (e.g. *attitude*), Exist (e.g. *birth*), Experience (e.g. *positive social interaction*), Move (e.g. *give*), Own, Produce (e.g. *refuse*)	Coordination (i.e. *and*), Disjunction (i.e. *or*)	Cause, Reference (i.e. *weak causality*), Goal, Motivation, Condition, Documentation

is used to model the fact that a set of villages collectively become a *structure*. The *becomes* predicate instance can be mapped to *exist* from Table 1. Our design choice to rely on entity composition leads to derivated predicates such as *becomesDate*, that allow associating a *timespec* entity to a predicative *occurrence*.

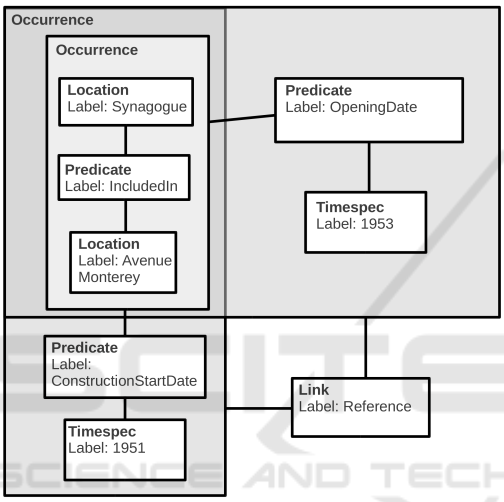

Figure 3: Data structure that can be associated to the following text: *Dans l'avenue Monterey se trouve également la synagogue, dont la première pierre fut posée en 1951 et qui fut inaugurée en 1953.*

Figure 3 shows an inclusion relation quite natural to locations - here the place *synagogue* is included in the street *avenue Monterey*. Such inclusion can be mapped to the predicate *own* in Table 1. *ConstructionStart* could be associated to existence, while *opening* can be seen as some kind of *production*. The overlap in Figure 3 emphasizes a nested structure with a shared argument. The two upper-level occurrences in the diagram are *reference* linked, reflecting the weak causality between them.

4 FROM TEXT TO STRUCTURED DATA

Following the terminology defined by (Zarri, 2009) and recalled in Section 2, a *narrative* is seen as a form of presentation. From this perspective, raw text can be seen as a form of presentation. The *fabula* is made by individual *occurrences* as shown in Figures 2 and 3. *Plots* emerge when such occurrences are *linked* as in Figure 3.

As exposed in Section 2, the most frequent setting in the literature is to consider that the mapping of a presentation to a plot structure is performed manually. In this section, we describe means to extract, at least approximately, the fabula and plot from this initial representation.

Performing this automatic mapping operation can have various utilities in the context of the project described in the introduction. First, as further described in Section 5, testimonies in our data corpus are recorded and transcribed manually, but exhibit no structure that facilitate their presentation in context, and exploration. Offline extraction of the narrative structure would avoid tedious manual efforts.

An interactive variant of the latter application would be the semi-automatic input of a story. When a user types a story in an interface, text would be sent on the fly to processing services. Based upon extracted indexes, related stories can then be displayed live to the client as a contextual help.

As mentioned in the introduction, our research context copes with testimonies collected in French. This constrained the technologies discussed later on in this section.

Named Entity Recognition consists in detecting entities such as people and places automatically in text. For example, the LIA tools (Favre et al., 2005) recognize people, locations, organizations, sociopolitical groups, quantities, dates and products in French text. Such facilities can then be a crucial initial step towards feeding the model described in Figure 1, of which people, places and time specifications are the core. The most renowned general purpose natural language processing system, the Stanford CoreNLP suite (Manning et al., 2014), also provides such NER functionalities for several languages including French. They can also be found in distant APIs such as Watson Natural Language Understanding (formerly AlchemyAPI) [1].

In order to structure recognized entities according

[1]https://natural-language-understanding-demo.mybluemix.net

to the schema described in Figure 1, and possibly extract non-named entities (i.e. mostly *artifacts*), syntactic cues are needed. *Part-Of-Speech* (POS) tagging is about estimating the function of words in text (e.g. adjective, verb, determinant). *Semantic Role Labeling* (SRL) builds upon POS-tagging in order to extract higher-order structures (e.g. subject, object, verbal forms), that are very close to the syntactic cues expected in our model. Actually this is not surprising insofar as the same seminal references in language analysis are foundational both for narratology (Zarri, 2009) and SRL (Jurafsky and Martin, 2014). POS-tagging facilities are available in French both in the LIA tools (Favre et al., 2005) and the CoreNLP suite (Manning et al., 2014). The latter also offers facilities in SRL, which are used by examples shown in Section 5.

NLP tools presented above allow extracting predicates and structural information, but the mapping of this information to the controlled vocabularies listed in Table 1 still has to be performed. Classically, this operation is facilitated using explicit taxonomies (see Section 2). Alternatively we propose to achieve this using a word embedding space (Mikolov et al., 2013). Candidates from the controlled vocabulary can be suggested by looking up nearest neighbors in the word embedding space. In other words, instead of explicit taxonomies, we use an implicit structure reflected by the word embedding space. Such mapping functions can be implemented locally using models from libraries such as TensorFlow (Abadi et al., 2016) trained with a corpus in French.

By default *reference* (i.e. weak causality) bindings can be used to link events constructed from a given story text, as inferred from relative positions in text. Time references detected in the text can also be attached by default to all detected events. The diagram in Figure 4 summarizes the proposed knowledge extraction pipeline.

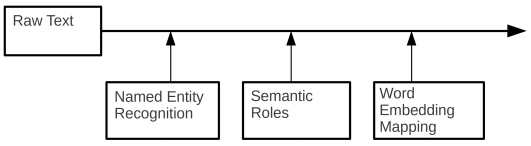

Figure 4: Outline of the proposed pipeline to extract knowledge from raw text.

5 IMPLEMENTATION

The experiments and examples shown in this Section used a corpus in French that aggregates two distinct textual resources:

- 267 short stories related to the period of 1945-1975 in Luxembourg and the surrounding region. The stories were selected and extracted automatically using the the Google Custom Search Engine API [2]. More precisely, the Google API was invoked for a list of heuristical queries about well known Luxemburgish locations or companies (e.g. *Kirchberg, Luxair*) for the targeted time frame. Results originate from several web portals (i.e. Wikipedia, http://www.industrie.lu, http://www.vdl.lu). Each story is associated to a date and a location name. This kind of indexing is easily handled by the model described in Figure 1. Each location name has been mapped with latitude and longitude by using the Google Maps Geocoding API[2]. Figure 5 shows an example of encoding. We refer to this corpus subset as *web* later on.
- Interviews were conducted with elderly people that have lived in Luxembourg in the time frame of interest (1945-1975). We used photograph collections from the spatio-temporal time frame of interest in order to trigger memories, as well as the *web* subset. We obtained approximately 5 hours of audio recordings, by 5 participants, among which 2 hours have been manually transcribed and segmented into approximately 100 short stories. We refer to this subset as *interviews* later on.

```
{
  "date": "29 décembre 1945",
  "locationName": "boulevard du Prince Henri Luxembourg-Ville",
  "text": "Pour rendre cet hommage plus vif encore, le conseil communal
      de la Ville de Luxembourg décida, le 29 décembre 1945, de changer
      la désignation «Boulevard du Prince » en «Boulevard du Prince Henri».",
  "longitude": 6.124179799999999,
  "latitude": 49.6120131,
}
```

Figure 5: Example of story encoding in the *web* subset.

All stories in the corpus do not have named entities or semantic structure extracted a priori. Only stories in the *web* subset are annotated with spatio-temporal metadata, which is allowed by the model described in Figure 1.

Our implementation of the model described in Section 3 has been supported by the Drupal content management system (Drupal, 2011). The choice of Drupal is also motivated by its integrated user credentials system, that will reveal useful when plugging our architecture in interactive views. Even though Drupal is run by a SQL database, the abstraction it uses make it adequate as support to our schema, which is rather akin to NoSQL-like databases (e.g. extensive usage of optional and variable-sized fields). *Entities*, already referred to extensively in the paper, are the abstraction in Drupal for objects. They can be specialized to

[2]https://developers.google.com

our needs, either via the administrative GUI or programatically. All types reported in Figure 1 derive from this root object type. Drupal also support *entity references*, that allow to define fields as references, or even arrays of references, to any other entity specialization in the schema.

Drupal also features the possibility to easily deploy a RESTful API (RESTful, 2014). A RESTful API facilitates the interaction and consumption of the managed data by any kind of application, as those envisioned in Section 4. In the remainder of the section, we describe our RESTful API implementation, and illustrate its usage by tier applications for either consumption or enrichment of the available data.

The routes displayed in Figure 6 are a limited implementation of the model described in Figure 1. They all support the GET (i.e. retrieving an entity of a set of entities) verb, as well as PATCH (i.e. updating an entity) or POST (i.e. creating a new entity) whenever appropriate. The route subset has been selected so as to address the most immediate needs. For example, the corpus data was provided under the JSON format, with only a *story_text* field, augmented by *location* and *timespec* fields for the *web* subset. The available REST routes allowed to come up quickly with a client for loading the files into the Drupal database. Also, the Drupal RESTful modules allows the fairly straightforward customization of the input and outputs of the routes defined. This facility enables the conversion of the *pretty* date format such as provided in JSON files to the flexible data format required by our model (see Figure 7). Unique labels are simple to initialize for *location* and *timespec* entities (the name itself and the pretty date format, respectively). As distinct stories might have the same *n* initial words, in our implementation, story labels are generated as MD5 hashes (Rivest, 1992) from the respective text.

```
http://[SERVER_URI]/api/stories
http://[SERVER_URI]/api/stories/[ID]
http://[SERVER_URI]/api/locations
http://[SERVER_URI]/api/locations/[ID]
http://[SERVER_URI]/api/timespecs
http://[SERVER_URI]/api/timespecs/[ID]
```

Figure 6: Routes available in the current RESTful API implementation. Routes with the *[ID]* marker retrieve only the entity holding the respective identifier, and all entities of the relevant type else.

The current REST API allows to easily retrieve the list of stories available in the database, or for example the stories associated to a given *timespec* entity (see e.g. Figure 7). Depending on needs formulated by applications consuming the data, straightforward extensions to this API would be to implement a route, or a filter to an existing route, that allows to search stories in the vicinity of given GPS coordinates, or


```
{
  "type": "timespecs",
  "id": "9",
  "attributes": {
    "id": 9,
    "label": "14 mai 1946",
    "self": "http://localhost:8080/api/v1.0/timespecs/9",
    "timespec_year_from": "1946",
    "timespec_month_from": "5",
    "timespec_day_from": "14",
    "timespec_year_to": "1946",
    "timespec_month_to": "5",
    "timespec_day_to": "14",
    "timespec_stories": [
      "7",
      "10",
      "11"
    ],
    "timespec_approximate": null,
    "timespec_pretty_from": "14 Mai 1946",
    "timespec_pretty_to": "14 Mai 1946"
```


Figure 7: Example of *timespec* entity returned by the respective route. Both the native database and *pretty* date formats are supported. The *timespec_stories* field provides the IDs of all stories related to this entity, enabling their programmatic retrieval.

stories that overlap with a specified time frame.

Even if they have still not been converted to full-fledged components connected to the REST API, experiments have also been carried out with knowledge extraction technologies such as described in Section 4.

In Figure 8 and 9, we see the NER results obtained with LIA and CoreNLP tools for the stories already used as examples in Figures 2 and 3, respectively.

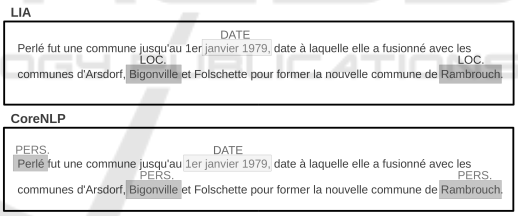

Figure 8: NER results for the example from Figure 2.

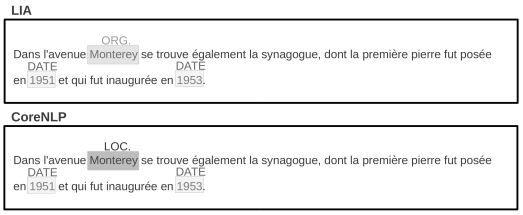

Figure 9: NER results for the example from Figure 3.

The recognition in Figure 8 suffers from discrepancies with both systems. The date is only partially recognized by LIA. The village names are only partially recognized by both systems, and even confused with people names by CoreNLP. Results for the example in Figure 9 are more satisfactory as dates and street names are identified - LIA confuses the street name with an organization name though.

Excerpts from graphical representations of syntactic trees extracted by CoreNLP are also shown in Figure 10. Higher level syntactic structures, such as subject-predicate-object or coordination (resp. l.h.s. and r.h.s. in Figure 10) can hence be extracted. NER performed in the previous step could then label extracted tree branches, yielding tentative data structures close to the format expected by our data model.

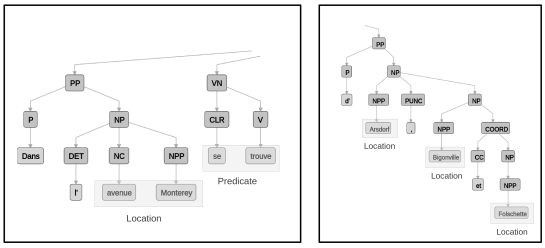

Figure 10: Excerpts from syntactic trees produced by CoreNLP tools. Relevant entities and predicates are highlighted.

We used the word embedding implementation provided as part of the Tensorflow software distribution (Abadi et al., 2016), and trained it with the complete French Wikipedia archive available at (Wikimedia, 2016). It contains approximately 2.3M articles in XML format. We converted them to a plain text format as expected by the training algorithm using the tool proposed by (Attardi, 2016). Punctuation and other non-textual characters were then removed using regular expressions. No stemming is required as all forms of a given word are embedded separately, and generally end up in close vicinity to each other.

Figure 11 shows an example of nearest neighbor search for the predicate extracted previously using a simple lookup script. Restricting the sets of nearest neighbors to the controlled vocabularies displayed in Table 1 effectively implements the needed mapping. We note that the uncertainty associated to language polysemy is naturally handled with this technique, as nearest neighbors returned belong to both possible meanings of the verb *trouver* (i.e. *finding*, and *situated* when associated with the reflexive prefix *se*). Information extracted using the sequence of operations described in this section could then be written in the database using adequate REST routes.

```
In [1]: nearby(['trouve'], 10)
b'trouve':
retrouve
trouvent
trouvait
situe
présente
près
situé
trouverait
ouest
trouvant
```

Figure 11: Nearest neighbors found for the predicate *trouve*. Responses associated to the *finding* meaning are in dark grey, those associated to *situated* are in light grey.

6 CONCLUSION

We described a data model and associated software pipeline to process, store and share personal narratives related to the period of 1945-1975 in Luxembourg. A RESTful interface that facilitates the exposition and the processing of such narrative information has been tested with real data collected from the web and interviews conducted with witnesses of the spatio-temporal time frame of interest.

Extending the field of the implementation to the full data model, and integrating NLP primitives into the Drupal platform workflow beyond the examples shown in Section 5 are the most immediate perspectives to this work.

Drupal features modules that facilitate the exposition of its managed data as a RDF schema, hence facilitating its interoperability with tier knowledge bases (Corlosquet et al., 2009). In this paper we focused on the consistency of the proposed data model, but linking it to external data sources is certainly a relevant perspective.

A key feature would also be to enable the detection of conflicts between stories and plots. Classical use of reasoning is to enable deduction of novel facts if adequate rules are defined (Suchanek et al., 2008), but this range of techniques has also been used to detect contradictions (Paulheim, 2016).

Finally, we will study the issue of subjectivity: personal narratives often contain explicit self-references (e.g. me, my brother). As far as first experiments show, all tested NER systems in Section 5 were not able to detect subjective entities. Simple heuristics could be implemented based on a closed list of keywords, or using the syntactic tree extracted by CoreNLP. *Anaphora Resolution* tools like GUITAR could also be tested (Poesio and Kabadjov, 2004).

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: a system for large-scale machine learning. In *12th USENIX conference on Operating Systems Design and Implementation*, pages 265–283.

Attardi, G. (2016). A tool for extracting plain text from Wikipedia dumps. https://github.com/attardi/wikiextractor.

Bollacker, K. et al. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250.

Brickley, D. and Miller, L. (2007). Foaf vocabulary specification 0.91. Technical report, ILRT Bristol.

Corlosquet, S., Delbru, R., Clark, T., Polleres, A., and Decker, S. (2009). Produce and Consume Linked Data with Drupal! In *International Semantic Web Conference*, pages 763–778.

Doerr, M. (2003). The CIDOC conceptual reference module: an ontological approach to semantic interoperability of metadata. *AI Magazine*, 24(3):75–92.

Drupal (2011). Drupal modules. https://www.drupal.org/project/.

Favre, B., Béchet, F., and Nocéra, P. (2005). Robust named entity extraction from large spoken archives. In *HLT/EMNLP 2005*, pages 491–498.

Gaeta, A., Gaeta, M., and Guarino, G. (2014). RST-based methodology to enrich the design of digital storytelling. In *IEEE INCOS 2015*, pages 720–725.

Google (2012). Introducing the knowledge graph. http://tinyurl.com/zofw8fb.

Hoffart, J., Suchanek, F. M., Berberich, K., and Weikum, G. (2013). YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61.

Jurafsky, D. and Martin, J. H. (2014). *Speech and Language Processing*, chapter Semantic Role Labeling. Pearson.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., and McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *ACM SEMANTICS*, pages 55–60.

Mendes, P. N., Jakob, M., García-Silva, A., and Bizer, C. (2011). DBpedia spotlight: shedding light on the web of documents. In *ACM SEMANTICS*, pages 1–8.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

Min, S. and Park, J. (2016). Mapping out narrative structures and dynamics using networks and textual information. *arXiv preprint arXiv:1604.03029*.

Mulholland, P., Wolff, A., and Collins, T. (2012). Curate and storyspace: an ontology and web-based environment for describing curatorial narratives. In *ESWC 2012*, pages 748–762.

Paulheim, H. (2016). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, pages 1–20.

Poesio, M. and Kabadjov, M. A. (2004). A general-purpose, off-the-shelf anaphora resolution module: Implementation and preliminary evaluation. In *LREC*.

RESTful (2014). RESTful best practices for Drupal. https://github.com/RESTful-Drupal/restful.

Rivest, R. (1992). The MD5 Message-Digest Algorithm. RFC 1321, MIT and RSA Data Security.

Scherp, A., Franz, T., Saathoff, C., and Staab, S. (2009). F-A Model of Events based on the Foundational Ontology DOLCE+DnSUltralite. In *K-CAP 2009*, pages 137–144.

Segers, R. et al. (2011). Hacking history: Automatic historical event extraction for enriching cultural heritage multimedia collections. In *K-CAP 2011*.

Suchanek, F. M. et al. (2008). Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the WWW*, 6(3):203–217.

Tilley, A. (1992). *Plot snakes and the dynamics of narrative experience*. Univ. Press of Florida.

Van der Meij, L., Isaac, A., and Zinn, C. (2010). A web-based repository service for vocabularies and alignments in the cultural heritage domain. In *ESWC 2010*, pages 394–409.

Van Hage, W. et al. (2012). Abstracting and reasoning over ship trajectories and web data with the Simple Event Model (SEM). *Multimedia Tools and Applications*, 57(1):175–197.

Van Hooland, S. et al. (2015). Exploring entity recognition and disambiguation for cultural heritage collections. *Digital Scholarship in the Humanities*, 30(2):262–279.

Weibel, S., Kunze, J., Lagoze, C., and Wolf, M. (1998). Dublin core metadata for resource discovery (no. rfc 2413).

Wikimedia (2016). Wikipedia dumps archive. https://dumps.wikimedia.org/frwiki/latest/.

Yeung, C. et al. (2014). A knowledge extraction and representation system for narrative analysis in the construction industry. *Expert systems with applications*, 41(13):5710–5722.

Zarri, G. (2009). *Representation and management of narrative information: Theoretical principles and implementation*. Springer Science & Business Media.