ATONTE: towards a new methodology for seed ontology development from texts and experts

Helen Mair Rawsthorne[1] Nathalie Abadie[1] Eric Kergosien[2] Cécile Duchêne[1] Eric Saux[3]

[1] LASTIG, Univ Gustave Eiffel, IGN-ENSG [2] GERiiCO, Université de Lille [3] IRENav, École navale

This work is co-financed by the Shom and the IGN and is being carried out at the LASTIG, a research unit at Université Gustave Eiffel.

Introduction

- ATONTE: ATlantis methodology for ONtology development from Texts and Experts
- Methodology for manual development of low-level seed ontologies
- Combines knowledge from non-fiction text corpora (manuals, information guides, sets of instructions) and the knowledge of domain experts
- Seed ontologies created with ATONTE can be used to develop and populate knowledge graphs for use in specific applications within given technical domains
- ATONTE is being refined empirically via the creation of the ATLANTIS (coAsTaL mAritime NavigaTion InstructionS) ontology [1]

State of the art

- Primarily automatic and semi-automatic approaches for ontology construction from text
- Still require manual checking, especially for legally-binding or security-oriented content

Proposal

- Reverse methodology integrating domain experts' knowledge and manual modelling techniques
- Depends on automation only for completing, instantiating and verifying the seed ontology
- Reuses elements from SAMOD [2], MOMo [3] and NeOn [4]

1. Choosing ATONTE

Requirements for ATONTE to be a good choice of seed ontology development methodology:

- The aim of the ontology is to model some or all of the knowledge contained in a non-fiction text corpus, and combine it with the knowledge of domain experts, with a given application in mind
- The end users of the ontology application are known and can be consulted, along with domain experts (can be the same people), on an ad-hoc basis during the development process
- No other ontologies that could be used for the final application, or part of it, already exist and have been published on the Web

2. Groundwork

- Define the ontology application
- Interview end users of the application to find out what knowledge from the corpus needs to be modelled in the ontology, what knowledge needs to be added (if any) and how it all needs to be structured in order to be useful to them
- Unless the knowledge spans only a very small domain, divide it into coherent subdomains in order to facilitate the documentation and modelling phases

3. Documentation

For each subdomain, write a motivating scenario, a list of competency questions and a glossary. Use the results of the interviews to guide the writing of this documentation.

- Motivating scenario
  - Give a name to the subdomain
  - Explain the motivation behind modelling the subdomain for the application in question and with the end users' needs in mind
  - List the principal characteristics of the concepts within the subdomain
  - Give a set of excerpts from the corpus that demonstrate all the ways in which the subdomain knowledge is represented in the text
- List of competency questions
  - Make a list of natural language competency questions that encapsulate all the uses for the knowledge mentioned by the end users during the interviews
  - The answers to the questions should figure in the excerpts in the motivating scenario
- Glossary
  - Define all the technical terms used in the subdomain

Enrich and validate the documentation with the help of domain experts.

4. Modelling and testing

- For each subdomain, semi-formalise the knowledge contained within each excerpt from the motivating scenario
  - Break down the relevant knowledge in the excerpt into finer-grained chunks, favouring a subject-predicate-object structure where possible
- Group together the subjects/objects and the predicates that serve the same purpose to create the first set of classes and properties for the subdomain
- Rewrite the semi-formal expressions formally as triples
- Try to formalise another set of excerpts by creating triples using the newly-created classes and properties
  - If this task cannot be performed satisfactorily, modify the model accordingly whilst ensuring that it still fits all the other excerpts
- Continue iteratively with unseen sets of excerpts until the model has stabilised and you have a solid set of triples specific to the subdomain
- Submit each subdomain model to a series of tests:
  - Model test: use a reasoner to verify the consistency of the model and then manually check that the model corresponds to the motivating scenario written for it
  - Data test: verify the validity of the model by populating it with the triples created from the corpus
  - Query test: translate the natural language competency questions into SPARQL queries and run them on the manually-created set of triples for that subdomain to check that the results match the answers specified in the documentation
- Move on to the next test only once the previous test has been passed
- If the subdomain model fails a test, return to the modelling phase to fix the issue before testing again

5. Merging and refactoring

- Merge the second largest subdomain model into the largest and then perform the series of tests on this intermediate model
- Repeat this process of merging the next-largest subdomain model into the intermediate model and then testing until all subdomain models have been integrated and the final full model has been created and tested
- The refactoring process involves reusing existing knowledge in semantic resources, annotating the model and enriching it using the capabilities of the OWL language
  - The elements of the refactoring process are given in SAMOD and a detailed description of how to reuse existing semantic knowledge resources is given in NeOn
- After the refactoring process has been carried out, the model should undergo a final testing cycle

Seed ontology

Excerpt 1 "The current in the bay flows eastward, but the wind is northerly."
Excerpt 2 "The port features a lighthouse."
Semi-formalisation current 1 - is a - current / bay 1 - is a - bay / current 1 - is in - bay 1 / current 1 - flows - eastward / wind 1 - is a - wind / bay 1 - has - wind 1 / wind 1 - is - northerly / port 1 - is a - port / lighthouse 1 - is a - lighthouse / port 1 - features - lighthouse 1
Formalisation id:current_1 :hasType :Current / id:bay_1 :hasType :Bay / id:current_1 :isLocatedIn id:bay_1 / id:current_1 :hasDirection :east / id:wind_1 :hasType :Wind / id:bay_1 :contains id:wind_1 / id:wind_1 :hasDirection :north / id:port_1 :hasType :Port / id:lighthouse_1 :hasType :Lighthouse / id:port_1 :contains id:lighthouse_1

Conclusion and perspectives

- ATONTE allows the creation of application seed ontologies from non-fiction text corpora and domain experts' knowledge
- Ontology modelling based upon manually-curated dataset
- To come: improving ATONTE by adding automatic extraction from corpus to populate knowledge graph and enrich seed ontology, and final ontology evaluation phase

References

[1] H. M. Rawsthorne, N. Abadie, E. Kergosien, E. Saux, ATLANTIS : Une ontologie pour représenter les Instructions nautiques, in: Journées Francophones d'Ingénierie des Connaissances (IC) Plate-Forme Intelligence Artificielle (PFIA 2022), Saint-Étienne, France, 2022, pp. 154–163.
[2] S. Peroni, A Simplified Agile Methodology for Ontology Development, in: OWL: Experiences and Directions – Reasoner Evaluation, Bologna, Italy, 2016.
[3] C. Shimizu, K. Hammar, P. Hitzler, Modular Ontology Modeling, Semantic Web (2022) 1–31.
[4] M. C. Suárez-Figueroa, A. Gómez-Pérez, M. Fernández-López, The NeOn Methodology for Ontology Engineering, in: M. C. Suárez-Figueroa, A. Gómez-Pérez, E. Motta, A. Gangemi (Eds.), Ontology Engineering in a Networked World, Springer, Berlin, Heidelberg, 2012, pp. 9–34.