The OwlExporter: Flexible Ontology Population from Text

This page describes the OwlExporter, a component that facilitates populating an OWL Ontology from annotations created by an existing GATE application.

1. Overview

The OwlExporter is an application-independent Processing Resource (PR) that can be included as part of any GATE pipeline. It allows to export document annotations created by the pipeline to individuals in a Web Ontology Language (OWL) model. The OwlExporter manages two ontologies, a domain-specific and a domain-independent one. This allows to link domain-specific entities detected in a text (e.g., an organism in a biomedical text) to their lexical representation, like paragraphs, sentences, or noun phrases. Note that the OwlExporter provides ontology population from text using existing ontologies only: if you need to create ontologies from text (so-called ontology learning), this is not the right component for you. For more background information on the OwlExporter, please read our LREC 2010 paper: Witte, R., N. Khamis, and J. Rilling, "Flexible Ontology Population from Text: The OwlExporter", International Conference on Language Resources and Evaluation (LREC), Valletta, Malta: ELRA, pp. 3845--3850, May 19--21, 2010.

2. Ontology Population Background

Semantics for the Masses
Ontology population from text is becoming increasingly important for NLP applications. Ontologies in OWL format provide for a standardized means of modeling, querying, and reasoning over large knowledge bases. Populated from natural language texts, they offer significant advantages over traditional export formats, such as plain XML. Ontologies have also become a major tool for developing semantically rich applications. Ontology models are capable of representing a large amount of information using a small number of axioms (individuals and relationships).

3. Example: Ontology Population using ANNIE

ANNIE is a "vanilla information extraction system" comprised of a set of core PRs such as a Tokeniser, Sentence Splitter, Part of Speech Tagger, Gazetteers, and JAPE grammars. The OwlExporter is capable of exporting the entities and relations detected by an ANNIE pipeline as individuals and relationships of an ontology. The populated ontology can then be used within any ontology-enabled tool for further querying, reasoning or visualization. The OwlExporter distribution includes an "Onto-ANNIE" demo pipeline that populates a domain- and NLP-ontology from the results of the ANNIE pipeline. The individual steps are explained in detail below; for more information, please refer to the user's guide included with the OwlExporter distribution.

3.1. Exporting Domain Individuals

The core approach of the OwlExporter is to define mappings, stored in document annotations, which define how an annotation is exported. Using simple mapping grammars, the OwlExporter is capable of exporting domain-specific entities from a document as individuals of the related concept in the domain ontology. On the left you can see annotations created by ANNIE (e.g., Location) and the temporary annotations "OwlExportClassDomain" used by the OwlExporter to map the entities of a document to the concepts in the domain ontology. On the right you can see the individuals exported to the corresponding concepts in the ontology.

3.2. Exporting Domain Datatype Relationships

The OwlExporter is also capable of exporting domain-specific relations from a document as datatype relationships in a domain ontology. In the example below, you can see the "hasGender" feature in the table on the left, which belongs to the temporary annotations created by our demo ANNIE pipeline, as well as the "OwlExportClassDomain" annotations used by the OwlExporter to map the entity relations of a document to the datatype property relationship in the ontology. On the right, you can see the resulting datatype relationship created for the individual "Nick" in the ontology.
3.3. Exporting Domain Object Property Relationships

Domain-specific relationships between two entities in a document can be easily exported as object property relationships using the OwlExporter. On the left, you can see the temporary annotation created by our demo ANNIE pipeline "OwlExportRelationDomain", which the OwlExporter uses to identify the relations of a document that need to be exported as object property relationships in the ontology. On the right, you can see the "livesIn" relationship created for the two individuals “Nick” and “Toronto” in the ontology.

3.4. Exporting NLP Instances

Similar to domain-specific entities, the OwlExporter is also capable of exporting NLP entities of a document as individuals in a separate NLP ontology (which is linked to the domain ontology as shown below). In the table, you can see the annotations created by ANNIE (e.g., Sentence), and the temporary annotations "OwlExportClassNLP" used by the OwlExporter to map the entities of a document to the concepts in the NLP ontology. The figure shows the resulting individuals exported to the corresponding concepts in the ontology.
<table>
<thead>
<tr>
<th>Type</th>
<th>Start</th>
<th>End</th>
<th>Id</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document</td>
<td>0</td>
<td>73</td>
<td>81</td>
<td>{source=/home/minus/Repository/durm/Applications/OwlExporterDemo/corpora/demo.txt, title=demo.txt}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>0</td>
<td>73</td>
<td>87</td>
<td>{className=Document, corefChain=[null], instanceName=demo.txt, kind=Class, representationId=81}</td>
</tr>
<tr>
<td>Sentence</td>
<td>0</td>
<td>29</td>
<td>39</td>
<td>{}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>0</td>
<td>29</td>
<td>86</td>
<td>{beginLocation=0, endLocation=29, className=Sentence, kind=Class, corefChain=[null], instanceName=Nick Khamis lives in Toronto, representationId=39}</td>
</tr>
<tr>
<td>Sentence</td>
<td>30</td>
<td>72</td>
<td>40</td>
<td>{}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>30</td>
<td>72</td>
<td>90</td>
<td>{beginLocation=30, endLocation=72, className=Sentence, kind=Class, corefChain=[null], instanceName=Toronto is a six hour drive from Montreal., representationId=40}</td>
</tr>
<tr>
<td>NP</td>
<td>0</td>
<td>17</td>
<td>76</td>
<td>{HEAD=lives, HEAD_END=17, HEAD_START=12, MOD=Nick Khamis, MOD_END=11, MOD_START=0}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>0</td>
<td>17</td>
<td>88</td>
<td>{HEAD=lives, HEAD_END=17, HEAD_START=12, className=NP, kind=Class, corefChain=[null], instanceName=Nick Khamis lives, representationId=76}</td>
</tr>
<tr>
<td>NP</td>
<td>21</td>
<td>28</td>
<td>77</td>
<td>{HEAD=Toronto, HEAD_END=28, HEAD_START=21}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>21</td>
<td>28</td>
<td>89</td>
<td>{HEAD=Toronto, HEAD_END=28, HEAD_START=21, className=NP, kind=Class, corefChain=[null], instanceName=Toronto, representationId=77}</td>
</tr>
<tr>
<td>NP</td>
<td>30</td>
<td>37</td>
<td>78</td>
<td>{HEAD=Toronto, HEAD_END=37, HEAD_START=30}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>30</td>
<td>37</td>
<td>91</td>
<td>{HEAD=Toronto, HEAD_END=37, HEAD_START=30, className=NP, kind=Class, corefChain=[null], instanceName=Toronto, representationId=78}</td>
</tr>
<tr>
<td>NP</td>
<td>41</td>
<td>57</td>
<td>79</td>
<td>{DET=a, DET_END=42, DET_START=41, HEAD=drive, HEAD_END=57, HEAD_START=52, MOD=six hour, MOD_END=51, MOD_START=43}</td>
</tr>
<tr>
<td>OwlExportClassNLP</td>
<td>41</td>
<td>57</td>
<td>92</td>
<td>{HEAD=drive, HEAD_END=57, HEAD_START=52, className=NP, kind=Class, corefChain=[null], instanceName=a six hour drive, representationId=79}</td>
</tr>
</tbody>
</table>
3.5. Exporting NLP Datatype Property Relationships

Exporting NLP related relations to an NLP ontology as datatype property relationships is also possible using the OwlExporter. On the left, you can see the "HEAD" feature (originating from the MuNPEX NP chunker) in the temporary annotation "OwlExportClassNLP", mapping the entity's relationship in the document to the corresponding datatype property relationship in the NLP ontology. On the right, you can see the same datatype relationship created for the individual "Toronto" in the output ontology.

3.6. Exporting NLP Object Property Relationships

The OwlExporter also supports the exporting of NLP object property relations between entities in a document. NLP relationships
between two entities in a document can be easily exported. On the left, you can see the temporary annotation created by our demo ANNIE pipeline "OwlExportRelationNLP", and on the right, you can see the "contains" relationship exported for the two individuals in the ontology.

### 3.7. Exporting Domain/NLP Relationships

Using the OwlExporter, relationships between the domain-specific and NLP entities in a document can be exported as object property relationships, linking domain and NLP individuals with each other. On the left, you can see the temporary annotation created by our demo ANNIE pipeline "OwlExportRelationDomainNLP", which the OwlExporter uses to identify the relations of a document that need to be created between two individuals from the domain and NLP ontologies. On the right, you can see the "appearsIn" relationship created for the two individuals "Nick" and "Toronto" from the domain ontology, with an instance of the "Sentence" concept from the NLP ontology. This facilitates advanced queries on the populated ontology – for example, retrieving all sentences that contain a particular domain concept.

### 3.8. Exporting Coreferences

The OwlExporter also supports creating relationships for entities that re-occur within a document, if they have been linked through a co-reference engine. On the left, you can see the temporary annotations "OwlExporterClassDomain", linking two entities of the same annotation type using the "corefChain" feature. The equivalent entities in the corpus are linked together using the "owl:sameAs" construct in the ontology, as shown on the left. (Note: in the current version, the expected coreference input format is different from the standard ANNIE coreferences.)
4. Installation

**OwlExporter Demo Pipeline** To install the OwlExporter PR and the demo pipeline:

1. Download, install and configure **GATE** (v7 or better).
2. Start GATE, open the Plugin Manager.
3. Go to the "Configuration" tab and enable the "Semantic Software Lab" repository. Click "Apply All". (You might also need to specify your "User Plugin Directory", where the downloads will be stored.)
4. Go to the "Available to install" tab and select the OwlExporter. Click "Apply All".
5. Go to the "Installed Plugins" tab and either check "Load Now" or "Load Always". Click "Apply All".
6. Load the example pipeline from "File" ? "Ready Made Applications" as shown in the screenshot.
7. Run the example pipeline (see included README and Documentation).

When the pipeline completed processing, you can find the two populated ontologies, "domain_out.owl" and "nlp_out.owl", in the gate/application-resources/ontologies/ directory of your OwlExporter installation.
5. Downloads

Additional downloads:

- The OwlExporter documentation (this is the same file as included with the distribution in the doc/ directory)
- Our research paper about the OwlExporter
- The GNU AGPL3 license under which you can use this tool.
- You can also download the install package creole.zip manually (but the recommended method of installation is to use the GATE Plugin Manager as described above).


6. Feedback

For questions, comments, etc., please use the Forum.

7. Version history

- 2.1: 18.05.2010. Added doc info features.
- 2.0: 01.03.2010. Initial public release.

Except where otherwise noted, all original content on this site is copyright by its author and licensed under a Creative Commons Attribution-Share Alike 2.5 Canada License.

Source URL (retrieved on 2023-11-21 04:32): https://www.semanticsoftware.info/owlexporter