Nell2RDF: Read the Web, and turn it into RDF

Antoine Zimmermann[‡], Christophe Gravier[†], Julien Subercaze[†], and Quentin Cruzille[‡]

[‡] École Nationale Supérieure des Mines, FAYOL-ENSMSE, LSTI, F-42023 Saint-Étienne, France
[†]Université de Saint-Étienne, Télécom Saint-Étienne, 23 rue du Dr. Rémy Annino, F-42023 Saint-Etienne, France
antoine.zimmermann@emse.fr
{christophe.gravier, julien.subercaze}@univ-st-etienne.fr
quentin.cruzille@gmail.com

Abstract. This paper describes the Nell2RDF platform that provides Linked Data of general knowledge, based on data automatically constructed by a permanent machine learning process called NELL that reads the Web. As opposed to DBpedia, all facts recorded by NELL can be tracked according to its provenance and a degree of confidence. With our platform, we aim at capturing all the data generated by NELL a transform them into state of the art Linked Data, following best practices. We discuss the benefits of the platform in opening new lines of research, while the work is still in progress.

Keywords: NELL, RDF-ising, machine learning

1 Introduction

Heterogeneity and redundancy are often seen as problematic in information system, especially on the Web. However, in many ways, they can be seen as a strength. On the Web, heterogeneity and redundancy allow one to consider the same information from different angles, thus helping build beliefs and trust in some information, while discrediting other. On the Web of Linked Data, most data sources are describing highly specialised information that are not replicated anywhere else. Moreover, Linked Data specialists usually advocate reuse via linking, rather than producing one's own description. Datasets of general knowledge are very scarce, but there is a knowledge base of everything (so to speak) called DBpedia, that is readily available to link to. DBpedia is an excellent project, but however powerful it may be, there is a need to provide alternative general purpose knowledge sources, if only to allow an objective comparison of generic knowledge bases. We propose a new source of general knowledge in the form of Linked Data, by leveraging a machine learning project called NELL. NELL computes facts automatically to form a knowledge base that constantly evolves. We use the freely available dataset to produce an RDF-ised version, that we in turn make available via a public sparql endpoint and data dumps. This paper is a description of the platform and aims at showing the huge potential that it can be for future research on Linked Data. In the following, we shortly describe NELL (Sect. 2); how we map NELL's ontology to OWL (Sect. 3); how NELL's believed facts are turned into RDF (Sect. 4); the platform for querying the dataset (Sect. 5) and finally the research opportunities we foresee (Sect. 6).

2 The Never Ending Language Learner

NELL is the Never-Ending Language Learner [2]. It is a machine learning and text mining programme that reads Web pages, apply NLP analysis to extract formal, logical facts in the form of Entity-Attribute-Value triples annotated with confidence and provenance information, then go on reading other Web pages that are linked from the current page. If the same fact is discovered several times from different places, the confidence in the fact grows. Moreover, NELL learns fact according to rules that it is able to learn, which makes its learning capabilities better and better.

At the same time, NELL embeds a (mostly) hand-made ontology that is used to detect inconsistencies, which prevents it from getting too much confidence in wrong guesses. During its constant reading of Web pages (now several billions), the confidence of certain facts reaches a threshold that is sufficient to consider the fact to be true. Such facts are published online on the Web site `http://rtw.ml.cmu.edu/rtw/` and a Twitter bot publishes those facts[1]. In order to currate data, a voting system is present on the web site. Manual edits are made from time to time to avoid false facts to misguide the learning process.

Both the internal ontology and the facts that are assumed to be true are publicly available and downloadable on the Web site.

3 Converting NELL's ontology into OWL

NELL's ontology is described in a tab separated value with three columns. The first column contains a term of the ontology that is being described (*i.e.*, the subject). The second column is an attribute or a relation on the term (*i.e.*, the predicate). The third element corresponds to the value of the attribute or another term to which the subject is related (*i.e.*, the object). The detailed meaning of the predicates is the following:

- `antireflexive` is used with a boolean and indicates that the term is an irrefexive property;
- `antisymmetric` indicates whether the term is an antisymmetric property, but there is no equivalent in OWL (although there is the class `owl:AsymmetricProperty`), so we record the information in a custom annotation property;
- `description` is always used with an English prose describing the term. We chose to record this in `rdfs:comment`, but could also use `dc:description`;
- `domain` has the exact same semantics as `rdfs:domain`;
- `domainwithinrange` indicates that anything that belongs to the domain of the property must also belong to its range, *i.e.*, in mathematical terms, `domainwithinrange` is true for predicate p iff $\forall x, \exists y.\mathsf{triple}(x,p,y) \Rightarrow \exists z.\mathsf{triple}(z,p,x)$;
- `generalizations` is translating the `rdfs:subClassOf` relation;

[1] NELL's twitter account. `https://twitter.com/cmunell`

Table 1. List of NELL's ontology predicates and their translation in RDF(S) and OWL

antireflexive	`rdf:type owl:IrreflexiveProperty`
antisymmetric	`nell:antisymmetric Literal(?object,xsd:boolean)`
description	`rdfs:comment Literal(?object,@en)`
domain	`rdfs:domain Class(?object)`
domainwithinrange	`nell:domainWithinRange Literal(?object,xsd:boolean)`
generalizations	`rdfs:subClassOf Class(?object)`
humanformat	`nell:humanFormat Literal(?object,xsd:string)`
instancetype	`nell:instanceType IRI(?object)`
inverse	`owl:inverseOf ?object`
memberofsets	if `?object` is `rtwcategory` then `rdf:type rdfs:Class`, else `?object` is `rtwrelation` then `rdf:type rdf:Property`
mutexpredicates	if `?subject` is a class then `owl:disjointWith ?object`, else `?subject` is a property then `owl:propertyDisjointWith ?object`
nrofvalues	if `?object` is `1` then `rdf:type owl:FunctionalProperty`, nothing otherwise
populate	`nell:populate Literal(?object,xsd:boolean)`
range	`rdfs:range ?object`
rangewithindomain	`nell:rangeWithinDomain Literal(?object,xsd:boolean)`
visible	`nell:visible Literal(?object,xsd:boolean)`

- `humanformat` indicates a format code that is used to display facts using the term on the website;
- `instancetype` has only 6 possible values (`all`, `common`, `proper`, `regexp`, `javaDateFormat`, `url`) and applies to classes to indicate that the instances are of a type that must be treated specially (such as dates). We do not use this information at the moment, we simply record the information in a custom annotation property;
- `inverse` has the same meaning as `owl:inverseOf`;
- `memberofsets` indicates whether the term is a class (`rtwcategory`) or a property (`rtwrelation`);
- `mutexpredicates` expresses disjointness of terms, and may apply to classes and properties alike;
- `nrofvalues` is either number `1` or `any`. The value `1` asserts that the term is a functional property;
- `populate` indicates that the term is not used explicitly in a class membership assertion or relation, but serves to group several subclasses or properties;
- `range` has the same semantics as `rdfs:range`;
- `rangewithindomain` is similar to `domainwithinrange` and mean, mathematically, that `rangewithindomain` is true for a property p iff $\forall x, \exists y.\mathsf{triple}(y,p,x) \Rightarrow \exists z.\mathsf{triple}(x,p,z)$;
- `visible` indicates that the term is not presented on NELL's homepage or in NELL's tweets.

Table 2. Correspondences between columns and RDF constructs.

E	Subject of a triple (IRI)
R	Predicate from the ontology, or `rdf:type`
V	Object of a triple (IRI or literal)
EL	`rdfs:label` of the subject
VL	`rdfs:label` of the object, if not literal
BEL	`skos:prefLabel` of the subject
BVL	`skos:prefLabel` of the object, if not literal
CE	`rdf:type` of the subject
CV	`rdf:type` of the object, if not literal

4 Converting facts to RDF

In this part, we describe how we turn NELL data to RDF. For this, we use the file that contains all beliefs of NELL[2]. Each line contains tab-separated values that we can treat separately to produce triples by extracting the content of the columns `Entity`, `Relations`, `Value`, `Entity literalStrings`, `Value literalStrings`, `Best Entity literalString`, `Best Value literalString`, `Categories for Entity` and `Categories for Value`. In the algorithm below, these columns will be respectively identified by the short names: E, R, V, EL, VL, BEL, BVL, CE and CV. There are a few extra columns that we do not consider for the current work and will omit in this discussion.

The first three columns directly map to subject, predicate and object respectively, but they do not form RDF triples as they are because the terms are neither IRIs nor literals. When V is not starting with `concept:`, it is converted into a literal, otherwise all terms are IRIs. The columns EL and VL provide English phrases that name the entity E and value V (*i.e.*, subject and object) respectively, while CE and CV define classes for E and V. Finally, BEL and BVL provide the most appropriate English phrase that name E and V. Table 2 defines how the columns are translated in their RDF counterpart, and Alg. 1 provides a detailed algorithm. Since EL, VL, CE and CV are lists rather than single values, we use them as sets in the algorithm.

In order to simplify the algotihm, we use the following functions: `add(s,p,o)` adds an RDF triple to an in-memory representation of the RDF graph, `geniri(v)` (generate an IRI from `v`), `genpropiri(v)` (generate a property IRI from `v`), `genclassiri(v)` (generate a class IRI from `v`), `genstr(v)` (generate a string from `v`), `genlit(v)` (generate a literal from `v`) and `isliteral(v)` (return true if `v` is a literal). Function `genlit(v)` can generate literals of type `xsd:integer`, `xsd:string` or `xsd:anyURI` that can correspond to sport match scores, geolocation, ages or wikipedia URL. The algorithm is applied to each line of the CSV file provided by the NELL web site. IRIs are generated by replacing the prefix `concept:` of NELL's terms by our IRI prefix `http://nell-ld.telecom-st-etienne.fr/`. Properties and classes have their own IRI scheme, with prefix `http://nell-ld.telecom-st-etienne.fr/ontology/`.

Example 1. We give an example of what can be found in NELL's belief set, followed by the RDF we generate. One line is presented in column for readability:

[2] Retrieved January, 2013. `http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.690.esv.csv.gz`

```
foreach c in CE do
    add(geniri(E), rdf:type, genclassiri(c));
end
foreach l in EL do
    add(geniri(E), rdfs:label, genstr(l));
end
add(geniri(E), skos:prefLabel, genstr(BEL));
if isliteral(V) then
    add(geniri(E), genpropiri(R), genlit(V));
end
else
    if E = generalizations then
        add(geniri(E), rdf:type, genclassiri(V));
        foreach l in VL do
            add(genclassiri(V), rdfs:label, genstr(l));
        end
        add(genclassiri(V), skos:prefLabel, genstr(BVL));
    end
    else
        add(geniri(E), genpropiri(R), geniri(V));
        foreach l in VL do
            add(geniri(V), rdfs:label, genstr(l));
        end
        add(geniri(V), skos:prefLabel, genstr(BEL));
        foreach c in CV do
            add(geniri(V), rdf:type, genclassiri(c));
        end
    end
end
```

Algorithm 1: Converting a line of NELL's belief set to RDF

E	concept:company:prudential001
R	concept:hasofficeincity
V	concept:city:boston
EL	"prudential" "Prudential"
VL	"boston" "Boston" "BOSTON"
BEL	prudential
BVL	boston
CE	
CV	concept:city concept:island

Note that CE is empty, while EL, VL and CV have multiple values. This line is translated into the following triples, in Turtle syntax:

```
@prefix  :  <http://nell-ld.telecom-st-etienne.fr/> .
@prefix no: <http://nell-ld.telecom-st-etienne.fr/ontology> .
:company/prudential001  no:hasofficeincity  :city/boston ;
    rdfs:label  "prudential", "Prudential" ;
    skos:prefLabel  "prudential" .
:city/boston  rdfs:label  "boston", "Boston", "BOSTON" ;
    skos:prefLabel  "boston" ;
    rdf:type  no:city, no:island .
```

5 The Nell2RDF dataset and platform

We set up a triple store running Sesame with a SPARQL endpoint at `http://nell-ld.telecom-st-etienne.fr/sparql`. This store contains the ontology and the translated belief set. The ontology has 541 properties and 310 classes, which is relatively few, but has 118,796 triples, mainly because NELL systematically asserts mutual disjointness of classes or properties. The existence of property disjointness, and other features like antireflexive and antisymmetric properties, put NELL in a fragment of OWL for which no efficient reasoners are known. Consequently, we only apply the simple RDFS materialisation that Sesame is able to perform.

The belief set has 5,805,102 triples concerning 1,475,390 distinct subjects and having 2,820,751 distinct objects, among which 2,799,591 are literals. The vast majority of triples concern the `rdfs:label` (2,479,546), `skos:prefLabel` (1,475,396) and `rdf:type` (1,244,493). Interestingly, there is a large amount of wikipedia URL (314,525) which could be used to help map NELL's entities to DBpedia.

We publish a dump of the data set, in gzipped ntriples as well as in gzipped HDT [3] available from the website `http://nell-ld.telecom-st-etienne.fr/`. Metadata about the dataset is available too, containing a SPARQL service description [4], VoiD metadata [1] and more.

6 Opportunities and Future Directions

Nell2RDF is currently a work in progress for which many improvements can be made. Especially, our conversion tool is currently ignoring the annotations regarding confidence and provenance of NELL's beliefs. We expect to develop a new version where named graph will be used for such metadata on triples. The annotation will be used as a real life test bed for annotated RDF reasoning and querying [5]. We also would like to offer an interface for browsing the data in function of the metadata (*e.g.*, filter facts by provenance or confidence). Moreover, the current implementation computes the dataset from the complete data dump, while the NELL website provides iterations in smaller files on a regular basis. We want to provide an automated update mechanism that keeps Nell2RDF in sync with NELL. In the meantime, we will work on comparing NELL to DBpedia in several ways: first, we will map our ontology to DBpedia's; second, we will experiment with (semi-)automatic instance matching techniques to link to DBpedia; third, we will compare coverage and quality of the two knowledge bases. From our preliminary inspection of NELL's belief, we recognise that NELL is still very far from DBpedia in terms of completeness and reliability.

Finally, we hope that this work will also serve other people improve the state of the art in Linked Data production, publishing and evaluation.

References

1. and Richard Cyganiak, Michael Hausenblas, and Yuting Zhao. Describing Linked Datasets with the VoID Vocabulary, W3C Interest Group Note 03 March 2011. W3C IG Note, World Wide Web Consortium (W3C), March 3 2011.
2. Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an Architecture for Never-Ending Language Learning. In Ronan Fox and David Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press, July 2010.
3. Javier D. Fernández, Miguel A. Martínez-Prieto, Claudio Gutiérrez, Axel Polleres, and Mario Arias. Binary RDF Representation for Publication and Exchange (HDT). *Journal of Web Semantics*, 2013.
4. Gregory Todd Williams. SPARQL 1.1 Service Description, W3C Recommendation 21 March 2013. W3C Recommendation, World Wide Web Consortium (W3C), March 21 2013.
5. Antoine Zimmermann, Nuno Lopes, Axel Polleres, and Umberto Straccia. A general framework for representing, reasoning and querying with annotated Semantic Web data. *Journal of Web Semantics*, 11:72–95, 2012.