NELL2RDF: Reading the Web, and Publishing it as Linked Data

José M. Giménez-García, Maísa Duarte, Antoine Zimmermann, Christophe Gravier, Estevam R. Hruschka Jr., and Pierre Maret

1 Univ Lyon, UJM-Saint-Étienne, CNRS, Laboratoire Hubert Curien UMR 5516, F-42023 Saint-Étienne, France {jose.gimenez.garcia, maisa.duarte, christophe.gravier, pierre.maret}@univ-st-etienne.fr
2 Univ Lyon, MINES Saint-Étienne, CNRS, Laboratoire Hubert Curien UMR 5516, F-42023 Saint-Étienne, France antoine.zimmermann@emse.fr
3 Federal University of Sao Carlos - UFSCar, S são Carlos, Brazil
4 Carnegie Mellon University - CMU, Pittsburgh, United States estevam@cs.cmu.edu

Abstract. NELL is a system that continuously reads the Web to extract knowledge in form of entities and relations between them. It has been running since January 2010 and extracted over 50,000,000 candidate statements. NELL’s generated data comprises all the candidate statements together with detailed information about how it was generated. This information includes how each component of the system contributed to the extraction of the statement, as well as when that happened and how confident the system is in the veracity of the statement. However, the data is only available in an ad hoc CSV format that makes it difficult to exploit out of the context of NELL. In order to make it more usable for other communities, we adopt Linked Data principles to publish a more standardized, self-describing dataset with rich provenance metadata.

Keywords: NELL, RDF, Semantic Web, Linked Data, Metadata, Reification

1 Introduction

Never-Ending Language Learning (NELL) is an autonomous computational system that aims at continually and incrementally learning. NELL has been running for about 7 years in Carnegie Mellon University (US). Currently, NELL has collected over 50 million of candidate beliefs, from with about 3.6 million have been promoted as trustworthy statements. NELL learns from the web and uses an ontology previously created to guide the learning. One of the most significant resource contributions of NELL, in addition to the millions of beliefs learned from the Web, is NELL’s internal representation (or metadata) for categories, relations and concepts. Such internal representation grows in every iteration, and is used by NELL as a set of different (and constantly updated)
feature vectors to continuously retrain NELL’s learning components and build its own way to understand what is read from the Web. Zimmermann et al. published in 2013 a solution to convert NELL’s beliefs and ontology into RDF and OWL. However, NELL’s internal metadata is not modeled in their work. Thus, the main contribution of this work is to extended the approach to include all the provenance metadata (NELL’s internal representation) for each belief. We publish this data using five different representation models: RDF reification [2, Sec. 5.3], N-Ary relations [19], Named Graphs [5], Singleton Properties [18], and NdFluents [10]. In addition, we publish not only the promoted beliefs, but also the candidates. As far as we know, this dataset contains more metadata about the statements than any other available dataset in the linked data cloud. This in itself can also be interesting for researchers that seek to manage and exploit meta-knowledge.

Our intention is to keep this information updated and integrate it on NELL’s web page.

The rest of the paper is organized as follows: Section 2 presents NELL and the components it comprises; in Section 3 describes the transformation of NELL data and metadata to RDF; Section 4 presents the dataset generated in this paper and how it is published; finally, Section 5 provides final remarks and future work.

2 The Never-Ending Language Learning System

NELL [3, 16] was built based on a new Machine Learning (ML) paradigm, the Never-Ending Learning (NEL). NEL paradigm is a semi-supervised learning [1] approach focused on giving the ability to a machine learning system to autonomously use what it has previously learned to continuously become a better learner. NELL is based on a number of coupled components working in parallel. These components read the web and use different approaches to, not only infer new knowledge in the form of beliefs, but also to infer new ways of internally representing the learned beliefs and their properties. Beliefs are divided into candidates and promoted beliefs. In order to be promoted a belief needs to have a confidence score of at least 0.9.

1. **AliasMatcher** finds relations between entities and their Wikipedia URL on Freebase. It was run only once and is currently not active.

2. **CML (Coupled Morphologic Learner)** [4] is responsible for identifying morphological regularities (such as that words finished in burg could be cities). It makes use of orthographic features of noun phrases (e.g., length and number of words, capitalization, prefixes and suffixes). CMC is the previous version of this component.

3. **CPL (Coupled Pattern Learner)** [4] is the component that learns Named Entities (NE) and Textual Patterns (TP) from text in the web pages. Internally, a different implementation was used between 2010 and 2013 that could learn

http://rtw.ml.cmu.edu/
categories and relations together. After that, CPL was splitted in CPL1 and CPL2, the former learning categories and the latter relations, but the distinction is not made in the knowledge base. All the knowledge from CPL1 is promoted only if CPL2 agrees. \(i.e.,\) CPL will extract TPs for categories (\(\_\) is a city, city such as \(\_,\) etc.) and for relations (arg1 is a city located in arg2, arg1 is the capital of arg2, etc.). Then, using those TPs, CPL will extract NEs for categories (e.g. city(Paris), city(Annecy), etc.) and NE pairs for relations (locatedIn(Paris, France), locatedIn(Annecy, France), etc.).

4. **KbManipulation** is used to correct some old bugs from NELL’s internal indexing knowledge. Several of these bugs should be removed automatically, but NELL has not one automated process for this task yet.

5. **LatLong** matches the literal string of Named Entities against a fixed geolocation database.

6. **LE (Learned Embeddings)** \([23]\) predicts new categories or relations of entities based on Event and Named Entity extraction. It creates a feature space where each dimension is a single NELL predicate, and NELL’s learned NE (or NE pairs for relations) is used as training examples. LE’s process predicts category or relation for NE (or NE pairs) that were not related in the training set.

7. **MBL**, also known as **ErrorBased Integrator** and **Knowledge Integrator**, is the component responsible for taking the decision of promotion based on the contributions of the other components. **EntityResolverCleanup** is the name used for the same MBL process applied during a big alteration in NELL’s knowledge base. In 2010 a big change was made in the NELL’s KB structure to make possible for two words to have different meanings (e.g apple the fruit and Apple the company) and, conversely, for a concept to use different words (e.g Google and Google Inc.).

8. **OE (Open Eval)** \([21]\) queries the web and extract small text using predicate instances. OE calculates the score based on the text distance between the instances in a relation.

9. **OntologyModifier** is used for any ontology alteration. This component appears in the Knowledge base when a new seed or and ontology extension is manually introduced.

10. **PRA (Path Ranking Algorithm)** \([9]\) is based on Random Walk Inference. PRA analyzes the connections between two categories instances which are the arguments for a relation. This component replaced the old **Ruler Learner** component.

11. **RL (Rule Learner)** \([13]\) extracts new knowledge using Horn Clauses based on the ontology. Its implementation was based on FOIL \([20]\). It can be found in NELL’s KB, but its execution stopped when NELL started to deal with polysemy resolution.

12. **SEAL (Coupled Set Expander for Any Language)** \([22]\) is the component responsible for extracting knowledge from HTML patterns. It works in a similar way to CPL, but using HTML patterns instead of textual patterns.
In the past it was called CSEAL, but after some improvements in its performance it changed the name for SEAL.

13. **Semparse**[12] combines syntactic parsing from CCGbank (a conversion of the corpus of trees Penn Treebank[15]) and distant supervision.

14. **SpreadsheetEdits** provides modifications in the NELL’s Knowledge base using human feedback.

Each of these components, with the exception of LE, output provenance information regarding theirs execution. In the next sections we present how this metadata is modeled in RDF.

### 3 Converting NELL to RDF

In this section we describe how NELL data and metadata are transformed into RDF. The first subsection presents how NELL’s ontology and beliefs are converted, following the work by Zimmermann et al.[24]; the second subsection describes how we convert the provenance metadata associated with each belief. NELL’s Knowledge bases used in this paper for the promoted and candidates beliefs are respectively corresponding to the iterations 1075[6] and 1070[7]. The code is publicly available in GitHub[8].

#### 3.1 Converting NELL’s beliefs to RDF

NELL’s ontology is published as a file with three tab separated values per line, where each line expresses a relationship between categories and other categories, relations, or values used by NELL processes. In order to convert NELL’s ontology to RDF each line is transformed into a triple as per Zimmermann et al.[24]. In short, the first and the third values are a pair of categories or relations, or either a category or relation in the first field and a value in the third. The second field is a predicate that indicates the relationship between the two elements. The transformations can be seen in Table 1.

NELL’s beliefs are also published in tab-separated format, where each line contains a number of fields to express the belief and the associated metadata, such as iteration of promotion, confidence score, or the activity of the components that inferred the belief. All the fields except 4, 5, 6, and 13 are used to convert the beliefs into RDF statements. Table 2 shows the meaning of each field. Fields 1, 2, and 3 are converted into the subject, predicate, and object of an RDF statement; the content of fields 7 and 8 create new statements using `rdf:label` properties; fields 9 and 10 create new triples with the property `skos:prefLabel`; finally, fields 11 and 12 are used to create triples indicating the types of the subject and the object. For a more detailed description of this step, refer to Zimmermann et al.[24].

---

[6] http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.1075.esv.csv.gz
[7] http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.1070.esv.csv.gz
[8] https://github.com/WDAqua/nell2rdf
Table 1: NELL’s ontology predicates and their translation in RDFS / OWL (from [24])

| NELL predicate       | Translation to RDFS / OWL                                                                 |
|----------------------|----------------------------------------------------------------------------------------|
| antireflexive        | rdf:type owl:IrreflexiveProperty                                                        |
| antisymmetric        | antisymmetric Literal(?object,xsd:boolean)                                            |
| description          | rdfs:comment Literal(?object,@en)                                                      |
| domain               | rdfs:domain Class(?object)                                                             |
| domainwithinrange    | domainWithinRange Literal(?object,xsd:boolean)                                        |
| generalizations      | rdfs:subClassOf Class(?object)                                                         |
| humanformat          | humanFormat Literal(?object,xsd:string)                                               |
| instancetype         | 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                                    |
| populate             | populate Literal(?object,xsd:boolean)                                                 |
| range                | rdfs:range ?object                                                                     |
| rangewithindomain    | rangeWithinDomain Literal(?object,xsd:boolean)                                        |
| visible              | visible Literal(?object,xsd:boolean)                                                   |

Table 2: Description of NELL’s beliefs fields

| #  | Field                      | Description                                                                 |
|----|----------------------------|-----------------------------------------------------------------------------|
| 1  | Entity                     | Subject of the belief                                                       |
| 2  | Relation                   | Predicate of the belief                                                     |
| 3  | Value                      | Object of the belief                                                        |
| 4  | Iteration                  | Iteration when the belief was promoted, or a list of iterations when the components generated the belief |
| 5  | Probability                | Confidence score of the belief                                              |
| 6  | Source                     | MBL activity to promote the belief                                          |
| 7  | Entity literalStrings      | Labels of the subject                                                      |
| 8  | Value literalStrings       | Labels of the object                                                        |
| 9  | Best Entity literalString  | Preferred label of the subject                                              |
| 10 | Best Value literalString   | Preferred label of the object                                               |
| 11 | Categories for Entity      | Classes of the subject                                                     |
| 12 | Categories for Value       | Classes of the object                                                       |
| 13 | Candidate Source           | Activity of the components that generated the belief                        |
3.2 Converting NELL metadata to RDF

Fields 4, 5, 6, and 13 of each NELL’s belief are used to extract the metadata. Each belief is represented by a resource, to which we attach the provenance information. In the promoted beliefs process, field 4 is used to extract the iteration when the belief was promoted, while field 5 gives a confidence score about it. On the other hand, in the candidate beliefs process, fields 4 and 5 contain the iterations when each component generated information about the belief, and the confidence score provided by each of them. Field 6 contains a summary information about the activity of MBL when processing the promoted belief. The complete information from field 6 is a summary of field 13. For that reason, we only process field 13. Finally, in field 13 every activity that took part in generating the statement is parsed.

The ontology can be seen in Figure 1. We make use of the PROV-O ontology [14] to describe the provenance. Each Belief can be related with one or more ComponentExecution that, in turn, are performed by a Component. If the belief is a PromotedBelief, it has attached its iterationOfPromotion and probabilityOfBelief. The ComponentIteration is related to information about the process: the iteration, probabilityOfBelief, Token, source and atTime (the date and time it was processed). The Token expresses the concepts that the Component is relating. Those concepts can be a pair of entities for a RelationToken, and entity and a class for a GeneralizationToken (note that LatLong component has a different token GeoToken, further described later). Finally, each component have a source string describing their process for the belief. This string is then further analyzed and translated into a different set if IRIs for each type of component in the subsections below.

The classes of the ontology are described in Table 3 and properties of the ontology are described in Table 4. The classes and properties of each component are described down below.

| Class               | rdfs:subClassOf | Description                                                                 |
|---------------------|-----------------|-----------------------------------------------------------------------------|
| Belief              | prov:Entity     | A belief                                                                     |
| PromotedBelief      | Belief          | A promoted belief                                                           |
| CandidateBelief     | Belief          | A candidate belief                                                          |
| ComponentExecution  | prov:Activity   | The activity of a component in an iteration                                  |
| Component           | prov:SoftwareAgent | A component                                                                |
| Token               | owl:Thing       | The tuple that was inferred by the activity                                 |
| RelationToken       | Token           | The tuple <Entity,Entity> that was inferred for a relation                   |
| GeneralizationToken | Token           | The tuple <Entity,Category> that was inferred for a generalization           |
| GeoToken            | Token           | The tuple <Entity,Longitude,Latitude> that was inferred for a geografical belief |
Table 4: Description of NELL metadata properties

| Property                      | rdfs:subPropertyOf         | rdfs:domain     | rdfs:range     | Description                                                                 |
|-------------------------------|-----------------------------|-----------------|----------------|----------------------------------------------------------------------------|
| generatedBy                   | prov:wasGeneratedBy         | Belief           | ComponentIteration | The Belief was generated by the iteration of the component                  |
| associatedWith                | prov:wasAssociatedWith      | ComponentIteration | Component   | The iteration was performed by the component                               |
| iterationOfPromotion          | owl:DatatypeProperty       | PromotedBelief   | xsd:integer    | Iteration in which the component was promoted                               |
| probabilityOfBelief           | owl:DatatypeProperty       | PromotedBelief   | xsd:decimal    | Confidence score of the Belief                                             |
| iteration                     | owl:DatatypeProperty       | ComponentIteration | xsd:integer  | Iteration in which a component performed the activity                      |
| probability                   | owl:DatatypeProperty       | ComponentIteration | xsd:decimal  | Confidence score given by the component                                    |
| hasToken                      | owl:ObjectProperty          | ComponentIteration | Token        | The concepts that the component is relating                                |
| source                        | owl:DatatypeProperty       | ComponentIteration | xsd:string   | Data that was used by the component in the activity                        |
| atTime                        | owl:DatatypeProperty       | ComponentIteration | xsd:dateTime | Date and time when the component execution was performed                   |
| tokenEntity                   | owl:DatatypeProperty       | Token            | xsd:string    | Entity on which the data was inferred                                      |
| relationValue                 | owl:DatatypeProperty       | RelationToken    | xsd:string    | Entity related the entity appointed by tokenEntity                          |
| generalizationValue           | owl:DatatypeProperty       | GeneralizationToken | xsd:string | Class of the entity appointed by tokenEntity                               |
AliasMatcher execution is denoted by a resource of class AliasMatcherExecution, and includes the date when the data was extracted from Freebase using the property freebaseDate. The added ontology can be seen in Figure 2.

![AliasMatcherExecution metadata ontology](image)

**Fig. 2:** AliasMatcherExecution metadata ontology

CMC execution is denoted by a resource of class CMCExecution. A number of morphological patterns MorphologicalPatternScoreTriple are attached to it, each one containing a name, a value, and a confidence score. The properties used can be seen in Table 5, while the ontology diagram is shown in Figure 3.

| Property                | rdfs:domain       | rdfs:range              | Description                                                                 |
|-------------------------|-------------------|-------------------------|-----------------------------------------------------------------------------|
| morphologicalPattern    | CMCExecution      | MorphologicalPatternScoreTriple | One of the morphological patterns used by CMC                               |
| morphologicalPatternName| MorphologicalPatternScoreTriple xsd:string | name of the morphological pattern (i.e., prefix, suffix, etc.) |
| morphologicalPatternValue| MorphologicalPatternScoreTriple xsd:string | Value of the morphological pattern (i.e., prefix = Saint and suffix = burgh) |
| morphologicalPatternScore| MorphologicalPatternScoreTriple xsd:decimal | Score of the morphological pattern |

CPL execution is denoted by a resource of class CPLExecution. It contains a series of textual patterns patternOccurrences, each one with a literal that describes the pattern, and the number of times it has occurred in the NELL’s data source. The properties used are described in Table 6, and the diagram for the ontology is shown in Figure 4.

| Property                | rdfs:domain       | rdfs:range              | Description                                                                 |
|-------------------------|-------------------|-------------------------|-----------------------------------------------------------------------------|
| patternOccurrences      |                  |                         |                                                                             |

KbManipulation execution is denoted by a resource of class KbManipulationExecution. It contains the bug oldBug that was manually fixed. Its shown in Figure 5.
Table 6: Description of CPL metadata properties

| Property          | rdfs:domain         | rdfs:range                      | Description                                                                 |
|-------------------|---------------------|--------------------------------|----------------------------------------------------------------------------|
| patternOccurrences| CPLExecution        | PatternNbOccurrencesPair        | One of the textual patterns used by CPL                                     |
| textualPattern    | PatternNbOccurrencesPair | xsd:string                   | Textual pattern in the form of a sentence                                     |
| nbOfOccurrences   | PatternNbOccurrencesPair | xsd:nonNegativeInteger       | Number of times it has occurred in the NELL’s source data                   |

Fig. 3: CMC metadata ontology

Fig. 4: CPL metadata ontology
**Fig. 5:** KbManipulation metadata ontology

*LatLong* execution is denoted by a resource of class *LatLongExecution*. It contains a list of locations *NameLatLongTriple* that were used to infer the belief. Each one containing the *name* and the latitude and longitude values. This execution has also its own token *GeoToken* with the latitude and longitude values reusing the same properties. The properties are detailed in Table 7, and the ontology diagram is shown in Figure 6.

**Table 7:** Description of LatLong metadata properties

| Property | rdfs:domain   | rdfs:range    | Description                        |
|----------|---------------|---------------|-----------------------------------|
| location | LatLongExecution NameLatLongTriple | One of the locations used by *Latlong* |
| name     | NameLatLongTriple rdf:langString   | Name of the location                  |
| latitudeValue | NameLatLongTriple xsd:decimal     | Latitude of the location               |
| longitudeValue | NameLatLongTriple xsd:decimal | Longitude of the location               |

*LE* execution is denoted by a resource of class *LEExecution*. It does not contain any additional triples.

*MBL* execution is denoted by a resource of class *MBLExecution*. It contains the entities and the categories of the other belief that was used to promote this one. The properties used are described in Table 8, and the ontology diagram is shown in Figure 7.

*OE* execution is denoted by a resource of class *OEExecution*. It contains a set of pairs *TextUrlPair*, each one including the sentence that was used to infer the belief, and the URL from where it was extracted. The properties used can be found in Table 9, and the ontology diagram in Figure 8.
Table 8: Description of MBL metadata properties

| Property             | rdfs:domain             | rdfs:range       | Description                                                                 |
|----------------------|-------------------------|-----------------|----------------------------------------------------------------------------|
| promotedEntity       | MBLExecution            | xsd:string      | Entity of a belief previously promoted                                      |
| promotedEntityCategory | MBLExecution            | xsd:string      | Category of the entity of the promoted belief                               |
| promotedRelation     | MBLExecution            | xsd:string      | Relation of the promoted belief                                             |
| promotedValue        | MBLExecution            | xsd:string      | Value of the promoted belief                                                |
| promotedValueCategory | MBLExecution            | xsd:string      | Category of the promoted belief, if applicable                             |

Fig. 6: LatLong metadata ontology

Fig. 7: MBL metadata ontology
Table 9: Description of OE metadata properties

| Property | rdf:domain | rdf:range | Description |
|----------|------------|-----------|-------------|
| textUrl  | OEExecution TextUrlPair |  | One of the pairs <text, url> used by OE |
| text     | TextUrlPair rdf:langString |  | Text extracted from the web |
| url      | xsd:anyURI |  | Web page where the text was extracted |

Fig. 8: OE metadata ontology
OntologyModifier execution is denoted by a resource of class OntologyModifierExecution. It contains the ontologyModification, which can be either a modification of a category or a modification of a relation. The ontology diagram can be seen in Figure 9.

PRA execution is denoted by a resource of class PRAExecution. It includes a series of Path resources describing the path followed in NELL dataset to infer the belief. Each Path includes its direction and a confidence score, along with a list of relations followed. The properties used can be seen in Table 10 while the ontology diagram is shown in Figure 10.

| Table 10: Description of PRA metadata properties |
|-----------------------------------------------|
| Property   | rdfs:domain   | rdfs:range | Description                                      |
| relationPath | PRAExecution Path |           | Relation path that entails the belief            |
| direction   | Path          | DirectionOfPath | Direction of the path                           |
| score       | Path          | xsd:decimal | Score assigned to the entailment                 |
| listOfRelations | Path   | rdf:List   | Ordered list of relations in the path            |

RL execution is denoted by a resource of class RLExecution. It contains a resource RuleScoresTuple that contains the Rule and a set of scores indicating the confidence, and the number of beliefs that are estimated to be correctly and incorrectly inferred (and the number of inferred beliefs for which it is not known if they are correct or not) with that rule. The rule itself contains the variables.
Technical Report. November 2017.

Fig. 10: PRA metadata ontology

and their values, and the predicates that are part of it. Each Predicate includes the name of the predicate and the two variables it uses. The complete list of properties can be found in table 11. The ontology diagram is presented in Figure 11.

SEAL execution is denoted by a resource of class SEALExecution. It includes the URL it used with the property url. The ontology diagram can be seen in Figure 12.

Semparse execution is denoted by a resource of class SemparseExecution. It includes a literal with the sentence used during it, using the property sentence. The ontology diagram can be seen in Figure 13.

SpreadsheetEdits execution is denoted by a resource of class SpreadsheetEditsExecution. It contains a set of literals describing the user who made the modification, the file used as input, the action made, and the modified entity, relation, and value. The list of properties can be seen in Table 12, while the ontology diagram is shown in Figure 14.

4 The NELL2RDF Dataset

The current version of NELL2RDF updates the promoted beliefs to the last version, adding the provenance triples about them. It also adds the candidate beliefs and their corresponding provenance triples. We provide the dumps for the promoted beliefs ⁹ and the candidate beliefs ¹⁰. The ontologies for the beliefs ¹¹

⁹ https://w3id.org/nellrdf/nellrdf.promoted.n3.gz
¹⁰ https://w3id.org/nellrdf/nellrdf.candidates.n3.gz
¹¹ https://w3id.org/nellrdf/ontology/nellrdf.ontology.n3
Table 11: Description of RL metadata properties

| Property        | rdfs:domain      | rdfs:range                  | Description                                                                 |
|-----------------|------------------|-----------------------------|-----------------------------------------------------------------------------|
| ruleScores      | RLExecution      | RuleScoresTuple             | The rule and set of scores used by RL                                         |
| rule            | RuleScoresTuple  | Rule                        | The rule RL used to infer the belief, in the form of horn clauses            |
| accuracy        | RuleScoresTuple  | xsd:decimal                 | Estimated accuracy of the rule in NELL                                        |
| nbCorrect       | RuleScoresTuple  | xsd:nonNegativeInteger      | Estimated number of correct beliefs created by the rule                     |
| nbIncorrect     | RuleScoresTuple  | xsd:nonNegativeInteger      | Estimated number of incorrect beliefs created by the rule                   |
| nbUnknown       | RuleScoresTuple  | xsd:nonNegativeInteger      | Number of rules created by the rules with no known correctness               |
| variable        | Rule             | xsd:string                  | One of the variables that appear in the rule                                |
| valueOfVariable | Rule             | xsd:string                  | Value of the variable inferred by the rule                                  |
| predicate       | Rule             | Predicate                   | One of the predicates that appear in the rule                                |
| predicateName   | Predicate        | xsd:string                  | Name of the predicate                                                       |
| firstVariable   | Predicate        | xsd:string                  | First variable of the predicate                                             |
| secondVariable  | Predicate        | xsd:string                  | Second variable of the predicate                                            |

Table 12: Description of SpreadsheetEdits metadata properties

| Property        | rdfs:domain      | rdfs:range                  | Description                                                                 |
|-----------------|------------------|-----------------------------|-----------------------------------------------------------------------------|
| user            | SpreadsheetEditsExecution | xsd:string       | User that made the modification                                               |
| entity          | SpreadsheetEditsExecution | xsd:string       | Entity of the belief affected by the modification                             |
| relation        | SpreadsheetEditsExecution | xsd:string       | Relation of the belief affected by the modification                           |
| value           | SpreadsheetEditsExecution | xsd:string       | Value of the belief affected by the modification                             |
| action          | SpreadsheetEditsExecution | xsd:string       | Action made in the modification                                               |
| file            | SpreadsheetEditsExecution | xsd:string       | File where the modification was saved and then read by SpreadsheetEdits       |
Fig. 11: RL metadata ontology

Fig. 12: SEAL metadata ontology
Fig. 13: Semparse metadata ontology

Fig. 14: SpreadsheetEdits metadata ontology
and the provenance metadata is common for both dumps. Metadata about the dataset is modeled using VoID and DCAT vocabularies.

In order to attach the metadata to each belief, we need to reify the statement into a resource. We follow five different models, described down below. A graphical representation of the models is shown in Figure 15. A summary of the triples and resources of each model can be seen in Table 13.

- **RDF Reification** [2, Sec. 5.3] represents the statement using a resource, and then creates triples to indicate the subject, predicate and object of the statement.
- **N-Ary relations** [19]: This model creates a new resource that identifies the relation and connects subject and object using different design patterns. Wikidata makes use of this model of annotation.
- **Named Graphs** [3]: A forth element is added to each triple, that can be used to identify a triple or set of triples later on. This model is used by Nano-publications.
- **The Singleton Property** [18] creates a unique property for each triple, related to the original one. It defines its own semantics that extend RDF, RDFS.
- **NdFluents** [10] creates a unique version of the subject and the object (in the case it is not a literal) of the triple, and attaches them to the original resources and the context of the statement.

| Model               | Promoted Size Triples | Candidates Size Triples | Total Size Triples |
|---------------------|-----------------------|-------------------------|--------------------|
| W/O metadata        | 2.99GB                | 0.02B                   | 162GB              | 1.45B              | 165GB | 1.48B |
| RDF Reification     | 50.9GB                | 0.24B                   | 776GB              | 4.50B              | 827GB | 4.74B |
| N-Ary Relations      | 50.7GB                | 0.24B                   | 770GB              | 4.50B              | 821GB | 4.74B |
| Named Graphs         | 49.8GB                | 0.24B                   | 727GB              | 4.24B              | 777GB | 4.48B |
| Singleton Property   | 49.8GB                | 0.24B                   | xxxGB              | xxxGB              | xxxGB | xxxGB |
| NdFluents            | 51.3GB                | 0.25B                   | xxxGB              | xxxGB              | xxxGB | xxxGB |

5 Discussion and Future Work

In this work we present the conversion of both data and metadata from NELL into RDF. It presents a thesaurus of entities and binary relations between them, as well as a number of lexicalizations for each entity. It also includes detailed provenance metadata along with confidence scores, encoded using five different reification approaches.

https://w3id.org/nellrdf/provenance/ontology/nellrdf.ontology.n3
https://w3id.org/nellrdf/metadata/nellrdf.metadata.n3
https://www.wikidata.org
Fig. 15: Reification models
Our goals for this dataset are twofold: First, we want to improve WDAqua-core0 [6] query answering system, providing it with more relations and lexicalizations, along with confidence scores that can help to give hints about how trustworthy is the answer. Second, given that it contains a big proportion of metadata statements, we want to use it as a testbed to compare how the different different metadata representations behave in current triplestores.

While currently we only publish the dumps of the datasets, we plan to provide SPARQL endpoint and full dereferenceable URLs. In addition, NELL is starting to be explored in languages different than English, such as Portuguese [7, 11] and French [8]. Our intention is to convert those datasets to RDF as they become available to the public, since the system and knowledge base are exactly the same used in the English one.

Acknowledgements: This work is supported by funding from the EU H2020 research and innovation program under the Marie Skłodowska-Curie grant No 642795. We would like to thank Bryan Kisiel from NELL’s CMU team for the technical support about NELL’s components.

References

[1] Blum, A., Mitchell, T.: Combining Labeled and Unlabeled Data with Co-Training. Proceedings of the Eleventh Annual Conference on Computational Learning Theory (1998)
[2] Brickley, D., Guha, R.: RDF Schema 1.1 - W3c Recommendation. Tech. Rep. 9780123735560 (2014)
[3] Carlson, A., Betteridge, J., Hruschka, Jr., E.R., Mitchell, T.M.: Coupling Semi-Supervised Learning of Categories and Relations. Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing (2009)
[4] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, Jr., E.R., Mitchell, T.M.: Toward an Architecture for Never-Ending Language Learning. Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI) (2010)
[5] Carroll, J.J., Bizer, C., Hayes, P.J., Stickler, P., Ellis, A., Hagino, T.: Named Graphs, Provenance and Trust. Tech. rep., ACM (2005)
[6] Diefenbach, D., Singh, K., Maret, P.: WDAqua-Core0: A Question Answering Component for the Research Community. ESWC, 7th Open Challenge on Question Answering over Linked Data (QALD-7) (2017)
[7] Duarte, M.C., Hruschka, Jr., E.R.: How to Read The Web In Portuguese Using the Never-Ending Language Learner’s Principles. Proceedings of the 14th International Conference on Intelligent Systems Design and Applications (2014)
[8] Duarte, M.C., Maret, P.: Vers une instance fran¸caise de NELL : chaˆıne TLN multilingue et mod´elisation d’ontologie. Revue des Nouvelles Technologies de l’Information Extraction et Gestion des Connaissances, RNTI-E-33, 469–472 (2017)
[9] Gardner, M., Talukdar, P.P., Krishnamurthy, J., Mitchell, T.M.: Incorporating Vector Space Similarity in Random Walk Inference over Knowledge Bases. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014)

[10] Giménez-García, J.M., Zimmermann, A., Maret, P.: NdFluents: An Ontology for Annotated Statements with Inference Preservation. Proceedings of the 14th Extended Semantic Web Conference (ESWC) (2017)

[11] Hruschka, Jr., E.R., Duarte, M.C., Nicoletti, M.C.: Coupling as Strategy for Reducing Concept-Drift in Never-Ending Learning Environments. Fundamenta Informaticae (1) (2013)

[12] Krishnamurthy, J., Mitchell, T.M.: Joint Syntactic and Semantic Parsing with Combinatory Categorial Grammar. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL) (2014)

[13] Lao, N., Mitchell, T., Cohen, W.W.: Random Walk Inference and Learning in A Large Scale Knowledge Base. Proceedings of the Conference on Empirical Methods in Natural Language Processing (2011)

[14] Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., Zhao, J.: PROV-O: The Prov Ontology. W3C Recommendation (2013)

[15] Marcus, M., Kim, G., Marcinkiewicz, M.A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., Schasberger, B.: The Penn Treebank: Annotating Predicate Argument Structure. Proceedings of the Workshop on Human Language Technology (1994)

[16] Mitchell, T.M., Cohen, W.W., Hruschka, Jr., E.R., Talukdar, P.P., Betteridge, J., Carlson, A., Mishra, B.D., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., Mohamed, T., Nakashole, N., Platanios, E.A., Ritter, A., Samadi, M., Settles, B., Wang, R.C., Wijaya, D.T., Gupta, A., Chen, X., Saparov, A., Greaves, M., Welling, J.: Never-Ending Learning. Proceedings of the 29th AAAI Conference on Artificial Intelligence (2015)

[17] Mons, B., Velterop, J.: Nano-Publication in the e-science era. Workshop on Semantic Web Applications in Scientific Discourse (SWASD) (2009)

[18] Nguyen, V., Bodenreider, O., Sheth, A.: Don’t like RDF Reification?: Making Statements about Statements Using Singleton Property. Proceedings of the 23rd International Conference on the World Wide Web (WWW) (2014)

[19] Noy, N., Rector, A., Hayes, P., Welty, C.: Defining N-Ary Relations on the Semantic Web. Tech. rep. (2006)

[20] Quinlan, J.R., Cameron-Jones, R.M.: FOIL: A Midterm Report. Proceedings of the European Conference on Machine Learning (1993)

[21] Samadi, M., Veloso, M.M., Blum, M.: OpenEval: Web Information Query Evaluation. Proceedings of the 27th AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA. (2013)

[22] Wang, R.C., Cohen, W.W.: Language-Independent Set Expansion of Named Entities Using the Web. Proceedings of the 7th IEEE International Conference on Data Mining (2007)

[23] Yang, B., Mitchell, T.M.: Joint Extraction of Events and Entities within a Document Context. Proceedings of the 2016 Conference of the North
[24] Zimmermann, A., Gravier, C., Subercaze, J., Cruzille, Q.: Nell2rdf: Read the Web, and Turn it into RDF. CEUR Workshop Proceedings (2013)