

Learning OWL 2 Property Characteristics as an Explanation for an RNN

J. POTONIEC^{1,2*}¹Institute of Computing, Poznan University of Technology, ul. Piotrowo 2, 60-965 Poznan, Poland²Center for Artificial Intelligence and Machine Learning, Poznan University of Technology, ul. Piotrowo 2, 60-965 Poznan, Poland

Abstract. We propose an approach to indirectly learn the Web Ontology Language OWL 2 property characteristics as an explanation for a deep recurrent neural network (RNN). The input is a knowledge graph represented in Resource Description Framework (RDF) and the output are scored axioms representing the characteristics. The proposed method is capable of learning all the characteristics included in OWL 2: functional, inverse functional, reflexive and irreflexive, symmetric and asymmetric, transitive. We report and discuss experimental evaluation on DBpedia 2016-10, showing that the proposed approach has advantages over a simple counting baseline.

Key words: recurrent neural networks, ontology learning, property characteristics, knowledge graphs, semantic web, deep learning.

1. Introduction

Machine learning models are constructed inductively representations of knowledge. So-called white-box models, such as decision trees or rules are interpretable, in this sense that the user can understand the reasons behind the decision. Conversely, black-box models, such as neural networks, are opaque to the user. They provide a result for a given set of inputs, but an explanation of why such a result was provided, is not available to the user. On the other hand, recent years showed that for some tasks deep neural networks can offer huge improvements over white-box models, reaching or even surpassing human-level performance.

Ontologies are also representations of knowledge, but they are by definition white-box: all the represented knowledge is explicit. Their applications are widespread and they are used, e.g., to convey domain-specific information in multi-agent systems [1], in competence management systems for education [2, 3], or to represent the domain of interest for a robotic system [4]. Unfortunately, the process of constructing an ontology is labor-intensive and means of supporting the responsible person, i.e., the ontology engineer, are needed. This problem is addressed by an area called ontology learning. The approaches used so far to ontology learning are similar to those used for learning white-box machine learning models and concerned mostly with the efficiency of traversing the search space and selecting measures with desirable properties.

In this paper, we propose to bridge the gap between efficient black-box learning and transparent white-box ontology learning. We hypothesize that the latent semantics of graph embed-

dings along with memory-like properties of recurrent neural networks (RNNs) can be exploited in order to evaluate property characteristics axioms that could extend the considered ontology. We believe that the sequential nature of the considered learning examples requires the usage of a recurrent architecture.

The contribution of this work is as follows: we propose a method explaining a deep RNN in order to score new axioms representing property characteristics that could be added to the ontology. The method is capable of scoring axioms corresponding to all the property characteristics defined by the RDF-based semantics of the Web Ontology Language OWL 2 [5]: functional, inverse functional, reflexive, irreflexive, symmetric, asymmetric, transitive. The obtained ontology is a surrogate model, i.e., an interpretable representation of some parts of the knowledge gathered in the neural network [6].

The rest of the paper is organized as follows: Section 2 is devoted to the basics of RDF and OWL, and in Section 3 we review the related literature. Section 4 is devoted to the details of the proposed approach, from preparing the data, to training the neural network, to extracting the explanations. We report the results of an experimental evaluation in Section 5, offer extended discussion of the results in Section 6 and conclude in Section 7.

2. Preliminaries

Resource Description Framework (RDF) is a popular, universal knowledge representation formalism equipped with formal semantics and standardized by the W3C [7]. The basic notion of RDF is an *RDF triple* consisting of a *subject*, a *predicate* and an *object* and stating that the subject is related to the object with the relation denoted by the predicate. A set of RDF triples forms an *RDF graph*, and the set of subjects and objects forms a set of *RDF nodes* of the graph, while each triple represents a labeled edge.

*e-mail: jpotoniec@cs.put.poznan.pl

Manuscript submitted 2019-08-26, revised 2020-06-22, initially accepted for publication 2020-07-17, published in December 2020

In general, an RDF node can be either an *Internationalized Resource Identifier* (IRI), a blank node, or a literal. An IRI is a globally-unique identifier for an entity, while a blank node represents an identifier, which is meaningful only in the scope of the considered graph. Usually, it is assumed that a single entity may be represented by multiple IRIs and blank nodes. A literal is a concrete value, such as a string of character or a number.

In the paper, we represent RDF graphs with a subset of the Turtle notation [8]: each triple is written in the order subject-predicate-object and ended with a dot. Throughout the paper we are interested only in triples consisting only of IRIs, we thus do not introduce the notation for other kinds of nodes. Frequently, an IRI is a long string, so for sake of clarity, we replace common namespaces with prefixes, and so `dbr:` stands for `http://dbpedia.org/resource/`, `dbo:` stands for `http://dbpedia.org/ontology/` and `dul:` stands for `http://www.ontologydesignpatterns.org/ont/penaltyz@{ }dul/DUL.owl#`, i.e., a foundational ontology DOLCE+DnS Ultralite. A sample RDF graph consisting of 10 triples, an excerpt from DBpedia [9], is presented in Table 2.

To represent taxonomical knowledge related to an RDF graph one may use an ontology expressed in Web Ontology Language OWL 2 [5]. An ontology consists of a set of *axioms*, equipped with a formal semantics underpinned by the Description Logics, and enabling deductive reasoning. An axiom consists of entities, corresponding to RDF nodes, and constructors, that provide the necessary expressiveness. Entities in an ontology can be divided into three separate categories: *individuals*, representing objects of the considered domain; *classes*, representing sets of individuals; *properties*, which denote relations between pairs of individuals (*object properties*) or between individuals and literals (*datatype properties*). Properties are an ontological view on RDF predicates.

In this paper, we are interested in axioms representing characteristics of object properties. OWL defines 7 such characteristics, which we present briefly using RDF triples notation:

functional For a functional property p , if (x, p, y_1) and (x, p, y_2) are true, then $y_1 = y_2$.

inverse functional For an inverse functional property p , if (x_1, p, y) and (x_2, p, y) are true, then $x_1 = x_2$.

reflexive For a reflexive property p , (x, p, x) is true for any individual x .

irreflexive For an irreflexive property p , (x, p, x) is false for any individual x .

symmetric For a symmetric property p , if (x, p, y) is true, then (y, p, x) is true.

asymmetric For an asymmetric property p , if (x, p, y) is true, then (y, p, x) is false.

transitive For a transitive property p , if (x, p, y) and (y, p, z) are true, then (x, p, z) is true.

To represent OWL axioms we use the Manchester syntax [10]. As we are interested only in the property characteristics axioms, all the axioms considered in the paper are of the form: ObjectProperty: p Characteristics: c , where p denotes the property and c denotes the characteristic.

3. Related work

Research on inducing new ontological axioms from RDF data concentrated mostly on various ways to extend a class hierarchy. Potoniec et al. considered mining subclass axioms with a fixed superclass and arbitrary OWL 2 EL class expression as a subclass [11]. Völker et al. proposed approaches to automatically discover class disjointness axioms by using associative rule mining and by posing the problem as a classification task [12]. Potoniec and Ławrynowicz presented an approach to split an existing class into a set of subclasses, each equipped with a formal definition through EquivalentTo axiom. The approach is based on frequent pattern mining [13] and poses the problem of splitting the class as an integer mathematical programming task [14]. Lehmann et al. developed DL-Learner, a tool for inductive learning in the Semantic Web, based on refinement operators [15]. Their goal was to learn complete definitions for the existing classes in an ontology.

Conversely, interest in mining property axioms was much more limited. Fleischhacker et al. presented a method for mining domain and range restrictions and property characteristics in the context of OWL 2 RL. The approach uses association rule mining on top of a propositionalized RDF graph [16].

More broadly, logical rules, OWL class expressions, and similar constructs were considered as building blocks of the hypothesis spaces for learning algorithms. For example, algorithms were proposed to solve classification tasks [17], to complete knowledge bases [18], to discover frequent patterns [13]. A comprehensive overview of approaches to ontology learning is presented in [19].

In recent years, the neural networks are on the rise and there are applications to many areas of computing. One of the popular approaches is to create embeddings, i.e., dense, low-dimensional representations of sparse and high-dimensional vectors of features, used in order to reduce dimensionality while keeping all the necessary information [20]. One of the particular applications is computing embeddings to preserve semantic information of a knowledge graph [21–23]. In the context of knowledge graphs, neural networks are also employed to translating from natural language to ontological axioms [24] and providing knowledge graph-based recommendations [25] have been proposed. In particular, Ghiasnezhad Omran et al. described a method for link prediction based on rules mined on top of matrix-based embeddings [26]. Yang et al. considered various embeddings methods and showed that they can be used to extract explicitly represented rules [27].

4. Proposed approach

An overview of the proposed approach is presented in Fig. 1 and the following sections describe its details.

4.1. Data preparation. Given is an OWL 2 ontology \mathbf{O} , an RDF graph \mathbf{G} such that $\mathbf{O} \cup \mathbf{G}$ is consistent and a set of properties P , whose characteristics are to be discovered. First, we split \mathbf{G} into a training set \mathbf{G}_{train} and a scoring set \mathbf{G}_{score} in such a way

Learning OWL 2 Property Characteristics as an Explanation for an RNN

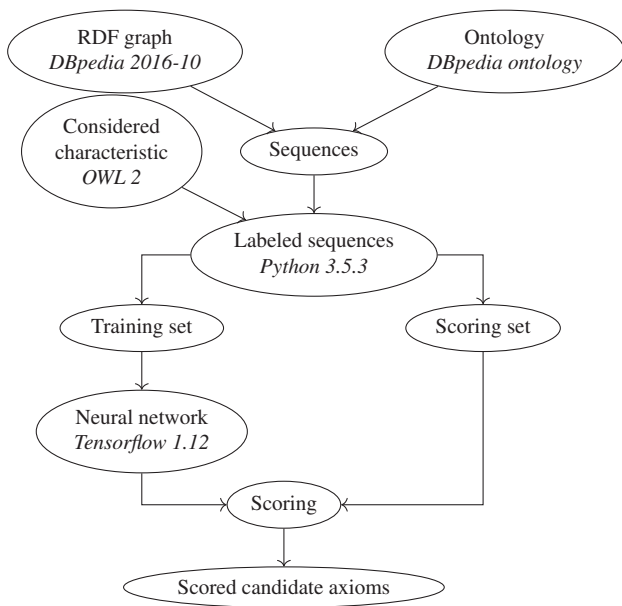


Fig. 1. An overview of the presented approach. First, an RDF graph, optionally enhanced with an ontology is used to generate a set of sequences, which are then labeled according to the considered characteristic. They are then split into a training and scoring set (Subsection 4.1). The training set is used to train an RNN (Subsection 4.2). Finally, the RNN is used to make classification on the scoring set, and the results are used to score the candidate axioms (Subsection 4.3). We used italics to indicate datasets and tools we used in the experiment described in Section 5

that the relative numbers of triples with each of the properties are approximately the same in \mathbf{G} , \mathbf{G}_{train} and \mathbf{G}_{score} . Then, we generate the sequences of triples using patterns presented in Table 1. For a fixed property p , they correspond to seven possible candidate axioms describing characteristics of p . We consider a single characteristic at a time, i.e., while generating sequences, one selects only the patterns corresponding to this characteristics. Moreover, we represent reflexive/irreflexive and symmetric/asymmetric as single patterns as they are opposites: a piece of evidence against one (e.g., reflexive) is a piece of evidence in favor of the other (e.g., irreflexive).

Table 1

Patterns used to construct sequences of triples for the neural network, based on [5]. Premises must be present in the graph to construct a sequence and the presence of conclusions in the graph is used as a label for the sequence. The variables are denoted by x, y, z , and different variables represent different IRIs. There are two conclusions corresponding to (ir)reflexivity, as each triple yields two conclusions, thus a single triple generates two sequences.

characteristics	premises	conclusions
functional	$(x, p, y), (x, p, z)$	$(y, owl:sameAs, z)$
inverse functional	$(y, p, x), (z, p, x)$	$(y, owl:sameAs, z)$
(ir)reflexive	(s, p, o)	$(s, p, s), (o, p, o)$
(a)symmetric	(s, p, o)	(o, p, s)
transitive	$(x, p, y), (y, p, z)$	(x, p, z)

For each set of triples from \mathbf{G} matching the premises of a pattern, we generate a sequence consisting of these triples and the conclusions prescribed by the pattern. The set of sequences is then split into a training set \mathbf{S}_{train} and a scoring set \mathbf{S}_{score} : a sequence is assigned to \mathbf{S}_{score} if any of the triples in its premises is in the set \mathbf{G}_{score} and to \mathbf{S}_{train} otherwise. This ensures that no triple from \mathbf{G}_{score} is present in \mathbf{S}_{train} and thus available for the network during training. All the sequences in \mathbf{S}_{train} are labeled: if the conclusions follow from $\mathbf{O} \cup \mathbf{G}_{train}$, the sequence is labeled with 1, otherwise with 0.

Consider triples presented in Table 2. Assume that triples T1–T6 represent a training graph \mathbf{G}_{train} , and triples S1–S3 a scoring graph \mathbf{G}_{score} . The triple X is a part of neither and we assume no preexisting ontology.

Table 2

An RDF graph used in the running example. Triples T_1 – T_6 represent a training graph, triples S_1 – S_3 a scoring graph and triple X does not belong to either of the graphs, and is presented here only for convenience.

We use : to denote the dbr: prefix here.

T_1	:Ibaraki_dialect	dbo:languageFamily	:Kanto_dialect.
T_2	:Ibaraki_dialect	dbo:languageFamily	:Japonic_languages.
T_3	:Tokyo_dialect	dbo:languageFamily	:Kanto_dialect.
T_4	:Tokyo_dialect	dbo:languageFamily	:Japonic_languages.
T_5	:Kanto_dialect	dbo:languageFamily	:Japonic_languages.
T_6	:Shikoku_dialect	dbo:languageFamily	:Japonic_languages.
S_1	:Iyo_dialect	dbo:languageFamily	:Shikoku_dialect.
S_2	:Iyo_dialect	dbo:languageFamily	:Japonic_languages.
S_3	:Sanuki_dialect	dbo:languageFamily	:Shikoku_dialect.
X	:Sanuki_dialect	dbo:languageFamily	:Japonic_languages.

For example, consider transitivity of the property $p = \text{dbo:languageFamily}$, represented as the following axiom: `ObjectProperty: dbo:languageFamily Characteristics: Transitive`.

First, we construct the following sequences of triples, denoted ST_x , from the graph using the pattern for transitive characteristics presented in Table 1: $ST_1 = (T_1, T_5, T_2)$, $ST_2 = (T_3, T_5, T_4)$, $ST_3 = (S_1, T_6, S_2)$, $ST_4 = (S_3, T_6, X)$. In each sequence, the first two triples correspond to premises and the last one is a conclusion. Observe that the sequence ST_4 is constructed even though X is not present in the considered graph. The sequence would not be constructed, should either S_3 or T_6 be missing from the graphs.

We now label the sequences. Sequences ST_1 , ST_2 and ST_3 are labeled with 1, as their conclusions (respectively: T_2, T_4, S_2) follow from the graphs. Sequence ST_4 is labeled with 0, as X does not follow from the graphs. Sequences ST_1 and ST_2 will be used as training sequences, as they do not contain any triples from the scoring graph. Sequences ST_3 and ST_4 will be used as scoring sequences.

4.2. Neural network. We use a deep RNN for classifying sequences. The distinguishing property of RNNs takes into account the output from the earlier stages of processing a given sequence. In our case this simulates the reasoning procedure: given the premises, are the given conclusions valid?

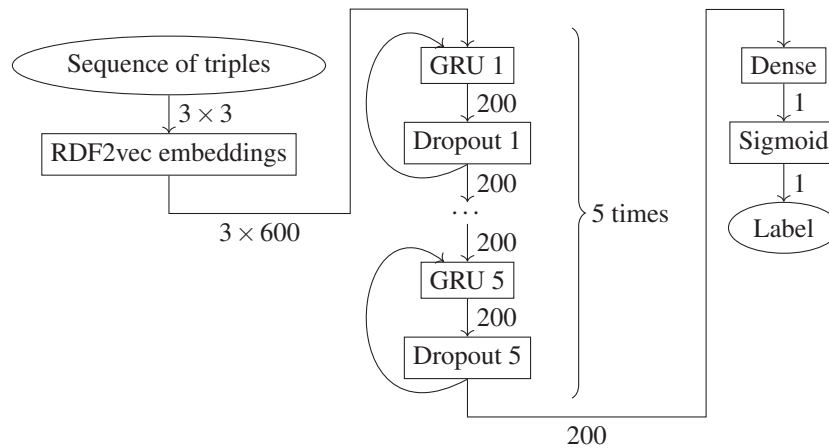


Fig. 2. An architecture of an RNN for sequences of length 3, described in details in Subsection 4.2

The architecture of the network is presented in Fig. 2. The network first encodes a sequence of triples into a sequence of the corresponding dense representations obtained by using pre-trained RDF2Vec embeddings [21]. An embedding for each IRI has a dimension of 200 and, for each triple, they are concatenated to form a single vector of length 600. Then, we stack 5 layers of 200 Gated Recurrent Units (GRU) each [28] and separate them with dropout layers [29] to avoid over-fitting. We preferred GRU over Long-Short Term Memory (LSTM) units, as they are more computationally efficient, but frequently offer a comparable performance [30]. On the last outputs of the last layer, we place a single neuron with a sigmoid activation function, and we threshold its output at 0.5 to obtain a label for the given sequence: 1 if the conclusion is valid (according to the RNN) and 0 otherwise.

4.3. Explanation generation. At this stage, the neural network is trained and their weights represent some knowledge about the presented sequences and thus the underlying RDF graph and ontology. We now aim to extract this black-box knowledge to obtain a white-box representation of ontological axioms.

To achieve this, we split S_{score} into disjoint subsets S_{score}^p , each such that all the triples in the sequences of the subset use the same property p . For each of the sets S_{score}^p , we then use the neural network to perform classification and obtain a set of labels. We average them and get a single score for each of the sets S_{score}^p , i.e., a single score for each property. The score is a measure of confidence of the neural network about the axiom representing the considered characteristics of the property. Such an axiom is a potential explanation for the behavior of the neural network. The axioms that scored high enough are presented to an ontology engineer for validation.

Continuing the example, we split the set S_{score} and obtain a single subset $S_{score}^{dbo:languageFamily}$ consisting of ST_3 and ST_4 . We perform classification with the RNN and obtain labels for the sequences ST_3 and ST_4 . Assume that in both cases the label is 1. Averaging the

labels we obtain the score of 1 for the considered axiom `ObjectProperty: dbo:languageFamily Characteristics: Transitive`.

5. Experimental evaluation

The goal of the presented experiment is to answer the following research question: is the approach presented in Section 4 capable of extracting explanations for the neural network behavior in a form of ontological axioms?

In order to answer this question, we use DBpedia 2016-10 as the dataset of choice and compute the axioms out of a neural network trained on the dataset. To be able to perform a comparison, we introduce a simple baseline and establish a gold standard. We perform the comparison twofold: casting it as a ranking comparison problem and as a classification problem, and we leverage appropriate measures to summarize the results and provide evidence related to the research question.

5.1. Setup. We implemented the approach using *TensorFlow 1.12* and *Python 3.5.3*. We used Adam optimizer with the default settings and trained the network for 10 epochs. We used DBpedia 2016-10 as the graph and pre-trained RDF2Vec embeddings [21]. We removed sequences using IRIs for which embeddings were not available. We considered all 653 properties from the DBpedia ontology and dropped those that occurred as the predicate of fewer than 100 triples in the graph, resulting in 496 properties. The limit of 100 triples was posed to ensure that the neural network is capable of learning the characteristics of the considered property.

We used at most 5,000,000 training sequences and required that $|S_{score}^p| \geq 10$ in order to ensure that the final scoring of an axiom is not based on only a very small number of sequences so that any single change does not fundamentally change the final score. In the example considered in the previous section, there were only two sequences in S_{score} , and thus any single change in the labeling caused the score of the axiom to leap from 1 to 0.5.

5.2. Results. Considering transitivity, we report the scores over 0.1 in Table 3, and for symmetry and asymmetry we report scores over 0.01 in Table 4. There were only 2 properties with a non-zero score for reflexivity: `dbo:state` (score

0.004) and `dbo:isPartOf` (score 0.001). It was not possible to meaningfully evaluate functional and inverse functional characteristics, as the used graph does not contain the relevant owl:sameAs statements. The full results, along with the trained model and the source code are published at <http://doi.org/10.5281/zenodo.3952397>.

Table 3

Properties with score above 0.1 (column neural), according to the approach presented in Section 4, when considering transitivity axioms. For comparison, the column baseline reports the scores of the baseline approach, presented in Subsection 5.3. The last column denotes the gold standard described in Subsection 6.2: 1 for a transitive property and 0 for a non-necessarily transitive property

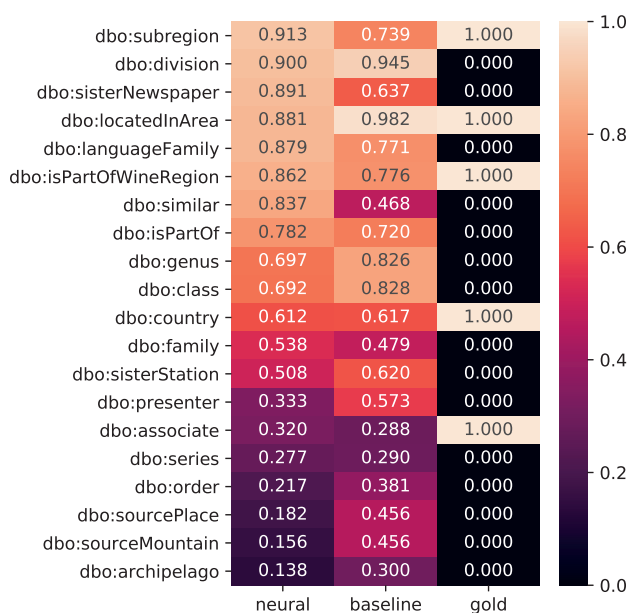
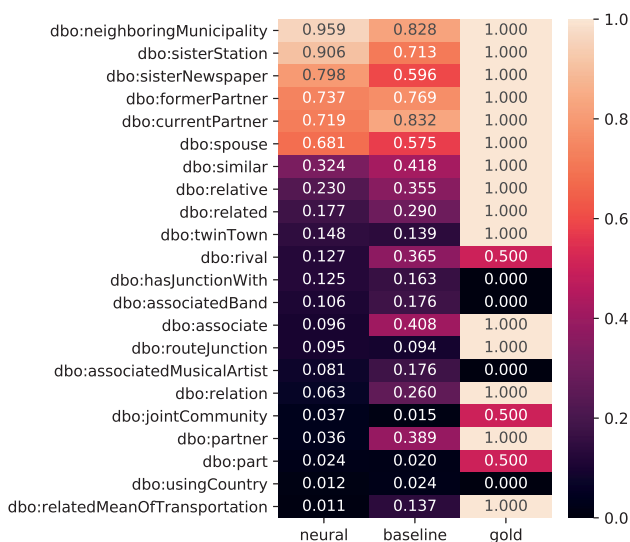


Table 4

Properties with score above 0.01 (column neural), according to the approach presented in Section 4, when considering symmetry and asymmetry axioms. For comparison, the column baseline reports the scores of the baseline approach, presented in Subsection 5.3. The last column denotes the gold standard described in Subsection 6.1: 1 for symmetric, 0 for asymmetric, and .5 for neither



5.3. Baseline. To provide a baseline for our method, we propose a simple counting approach. For a given property p , a given characteristic, and an RDF graph G , we generate all triples that are direct logical conclusions from the graph assuming that p has the considered characteristic, arriving at a graph C . The expected conclusions are given by Table 1. Then, we count the number of triples of C that are present in G and divide by the cardinality of C , obtaining the final score ξ in the range $[0, 1]$, where 0 means that none of the expected logical conclusions was present in the graph, while 1 means that all of them were present. If C is empty, we assume the score to be 0, as there is no evidence to support the considered axiom:

$$\xi = \begin{cases} \frac{|C \cap G|}{|C|} & C \neq \emptyset, \\ 0 & C = \emptyset. \end{cases}$$

For transitivity, there are 25 properties with $\xi = 1$, reported in Table 5, but it seems they all represent coincidences due to very

Table 5

Properties with the highest baseline score $\xi = 1.0$ when considering transitivity. The column $|C|$ reports the size of the corresponding set C , while the column # triples reports the number of triples with the considered property in the graph G

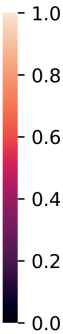
property	C	# triples
dbo:twinCountry	13	212
dbo:usedInWar	11	4002
dbo:guest	10	31740
dbo:showJudge	9	2123
dbo:vehicle	6	173
dbo:narrator	5	4809
dbo:militaryUnit	4	12305
dbo:musicalArtist	3	51667
dbo:musicalBand	3	51667
dbo:portrayer	3	5439
dbo:wineRegion	3	309
dbo:sport	2	5201
dbo:editor	2	4914
dbo:lastAppearance	2	1664
dbo:island	2	461
dbo:manager	1	14721
dbo:assembly	1	12167
dbo:chairman	1	6214
dbo:illustrator	1	2777
dbo:lyrics	1	1825
dbo:maintainedBy	1	1336
dbo:translator	1	1053
dbo:promotion	1	976
dbo:anthem	1	683
dbo:alumni	1	199

small sizes of the set C (at most 13) compared to the number of triples in G with the considered property. We thus introduced additional requirement $|C| \geq 100$ and report the properties with top 10 scores in Table 6. Similarly to the results reported in Subsection 5.2, considering reflexivity, the highest score assigned was 0.02 and it was not possible to evaluate functional and inverse functional characteristics.

Table 6

Top 10 properties with the highest baseline score when considering transitivity, requiring $|C| \geq 100$. The column neural represent the score assigned by the approach proposed in Section 4. The last column denotes the gold standard described in Subsections 6.2 and 6.3: 1 for a transitive property and 0 for a non-necessarily transitive property

	neural	baseline	gold
dbo:locatedInArea	0.881	0.982	1.000
dbo:division	0.900	0.945	0.000
dbo:state	0.000	0.903	0.000
dbo:class	0.692	0.828	0.000
dbo:genus	0.697	0.826	0.000
dbo:isPartOfWineRegion	0.862	0.776	1.000
dbo:languageFamily	0.879	0.771	0.000
dbo:subregion	0.913	0.739	1.000
dbo:isPartOf	0.782	0.720	0.000
dbo:starring	0.082	0.686	0.000



6. Discussion

In this section, we establish a gold standard for a subset of the considered properties, based mostly on annotations (labels and comments) in the DBpedia ontology, the DBpedia mappings wiki¹ and literature concerning the subject of particular properties, as well as more general literature related to ontological properties of some popular constructs.

6.1. Symmetry. We discuss on the case-by-case basis all properties with a score over 0.01 and report the aggregate results in Table 4.

First, we argue that there are multiple properties that, on the basis of their labels and the dictionary meaning of the words used in them, should be deemed symmetric: (a) family-related: `dbo:spouse`, `dbo:currentPartner`, `dbo:formerPartner`, `dbo:partner` (DBpedia is vastly incomplete, possibly because of the following guidelines in Wikipedia: *If particularly relevant, or if the partner is notable*; [31]), `dbo:relation` (always mapped from `relations` property of an infobox, described as *Names of siblings or other relatives*. [31]); (b) neighbourhood-related: `dbo:neighboringMunicipality`, `dbo:twinTown` [32, 33], `dbo:routeJunction`; (c) similarity: `dbo:similar`, `dbo:relative`, `dbo:related` and

`dbo:relatedMeanOfTranspion` (the corresponding infobox property is described as: *The related field lists similar vehicles that share considerable componentry with the subject vehicle*. [34]).

Wikipedia defines a sister paper as *one of two or more newspapers which share a common owner* [35], and gives a similar definition for a sister station [36], we thus mark both `dbo:sisterStation` and `dbo:sisterNewspaper` as symmetric, assuming that in such cases the definitions in Wikipedia is what guides Wikipedia editors.

For `dbo:associate` analysis of the mappings reveals that it is always mapped from the infobox property *alongside*, which is described as *For two or more people serving in the same position from the same district. (e.g. United-States Senators.)* [37]. We find this to sufficiently justify the symmetry of this property.

We also observe that there are numerous properties that are asymmetric. We stipulate that `dbo:associatedBand` (a subproperty of `dbo:memberOf`), and `dbo:associatedMusicalArtist` (a subproperty of `dbo:hasMember`) are asymmetric under a reasonable assumption that an object cannot be a member of itself

The domain of `dbo:usingCountry` is `dbo:currency` and its range is `dbo:country`. While the DBpedia ontology does not enable inference that currency and country are disjoint concepts, it seems to be a reasonable assumption, yielding the property asymmetric.

`dbo:hasJunctionWith` is a property with the domain and range `dbo:canal`. In the mappings, it is used only for for the property `join` in `Infobox_canal`. After the release of DBpedia 2016-10, `join` was renamed to `connects_to` [38], and from the presence of *to* we infer this is an asymmetric property.

`dbo:rival` is a very specific property with both domain and range being `dbo:School`. However, a more broad look at the concept of rivalry reveals that rivalry may be defined (...) *as a perceptual categorizing process in which actors identify which states are sufficiently threatening competitors to qualify as enemies*. [39]. This, in turn, strongly hints that it may be possible that rivalry is not symmetric (but is also not asymmetric): while one side sees the other as a rival, the other side may or may not sees the first one as a rival.

`dbo:jointCommunity` is a rarely used property derived from `Samtgemeinde` property of infoboxes related to German and Israel municipalities. The name strongly hints that the property is not meant to be symmetric, as one side should (typically) represent a community consisting of multiple members. This is also supported by the fact that this is a sub-property of `dul:isPartOf`. Without detailed knowledge of the respective law systems in this regard, it is hard to decide whether this is asymmetric property.

For `dbo:part`, mappings and range (`dbo:Location`) show that it is geography-related subproperty of `dul:hasPart`. Part-of relations are not symmetric, but also are not necessarily asymmetric, unless they are meant as proper-part-of [40].

6.2. Transitivity. In Table 3 we report the properties with a transitivity score of at least 0.1 and compare them with the gold

¹<http://mappings.dbpedia.org/>, a dump is available in a git repository <https://github.com/dbpedia/mappings-tracker.git>

standard. Below, we give justifications for the labels in the gold standard.

We observe that transitivity is frequently considered in the context of various forms of *part of* relation. At the first sight, one may be tempted to assume that *part of* is a transitive relation, but this is not necessarily so. Keet and Artale [40] introduce a classification of *part of* relations exactly for the purpose of distinguishing between transitive and non-transitive (in the sense of not being transitive, not being intransitive) variants. The classification is related to DOLCE [41] and its notion of *endurant* and *pendurant*, the first one roughly corresponding to entity types and the other one to processes and relations. We find this rough distinction sufficient for our purposes, we thus leave it at that, referring the interested reader to the literature. For our purposes, we use the first two levels of their taxonomy:

- Properties that are transitive:
 - structural part of** between *endurant* and *endurant*;
 - spatial part of** between *endurant* and *endurant*, but such that region of the first is a part of of the region of the second;
 - involved in** between *pendurant* and *pendurant*;
- Properties that are non-transitive:
 - member of** between physical or social object and social object;
 - constitutes** between matter and physical object;
 - subquantity of** between matter and matter;
 - participates in** between *endurant* and *pendurant*

First, `dbo:isPartOf` is a non-transitive property. For the rest, we begin by discussing properties that can be assigned to one of the presented classes.

Based on names, domain and ranges we consider the following as *spatial part of*, and thus claim them transitive: `dbo:subregion`, a subproperty of `dul:hasPart`, with both its domain and range `dbo:Place`; `dbo:locatedInArea`, a subproperty of `dul:hasLocation` with both domain and range `dbo:Place`; `dbo:isPartOfWineRegion`, a subproperty of `dul:hasPart`, with both its domain and range `dbo:WineRegion`, a subclass of `dbo:Place`; `dbo:country`, a subproperty of `dul:hasLocation`, with range `dbo:Country` and an unspecified domain. This last case may seem counter-intuitive, so consider the following example: *Abess End* is a hamlet in *England* (country), which is also a part of *The United Kingdom* (also a country), and – by transitivity – *Abess End* is a hamlet in *The United Kingdom*.

There are also few candidates for transitivity on the grounds of *spatial part of*, which must be rejected due to the incompatibility of domain and range. In particular, `dbo:sourcePlace` and `dbo:sourceMountain` are always used together to map infobox property *source_location* in `Infobox_river` and `Geobox` (the latter was removed from Wikipedia since²). To complicate matters, the range of

the first is `dbo:PopulatedPlace`, whereas for the other it is `dbo:Mountain`. While the notion of a source is not surprising for flows of water, it is rather unexpected for a populated place or a mountain (at least in spatial, and not temporal, sense). Similarly, `dbo:archipelago` is a sub-property of `dbo:isPartOf` with the domain of `dbo:Island` and an unspecified range. However, we expect the range of this property to consist of archipelagos, i.e., *groups of islands*. Thus, we consider these properties as a non-transitive.

`dbo:series` is used in contexts such as games, books, television series, etc. We believe this is a *member of*, and thus not transitive.

Further, there are numerous properties representing similarity and relatedness: `dbo:sisterNewspaper`, `dbo:sisterStation`, `dbo:similar`. We believe they are not transitive, as one can easily imagine context such as three half-siblings A, B, C such that A and B share a parent and B and C share the other parent of B, but A and C have no common parents.

For `dbo:associate`, based on the discussion in 6.1, we stipulate that this property is transitive, as it requires *same position* and *same district* to hold.

`dbo:division` is a subproperty of `dul:isPartOf` and is used in context of biological taxonomies (to map infobox properties *subdivisio*, *divisio* and *superdivisio* of `Taxobox`), companies (e.g., *divisions* of `Infobox_company`), sports teams (*division* of `Infobox_sports_team`) and German settlements (*divisions* and *Gliederung* of `Infobox_German_location`). We observe that the usage in the context of sports teams is a clear example of the *member of* usage, making the property non-transitive.

The domain of `dbo:languageFamily` is `dbo:Language` and we observe that Ethnologue [42], an established source of information about languages, introduces a distinction between languages and language families, even in the cases when there is exactly one language in the family³. We thus conclude that the domain and range of the property are disjoint and thus the notion of transitivity is irrelevant.

`dbo:genus`, `dbo:family` and `dbo:order` is used strictly in the context of biological taxonomies, where each level is separate and named differently (genus, family, order, etc.) [43], so effectively (if not in the DBpedia ontology) the domain and range of the property are disjoint, making the considered property non-transitive.

The domain of `dbo:class` is `dbo:MeanOfTransportation`, but in the mappings, it is used both in the context of means of transportation, as well as in the context of biological taxonomies. For the second usage, the discussion for `dbo:genus` applies, and for the first, there is a lack of definitive evidence in favor of transitivity, we thus stipulate that this is not a transitive property.

`dbo:presenter` has domain of `dbo:TelevisionShow` and range of `dbo:Person`,

²<https://en.wikipedia.org/w/index.php?title=Special:Log/delete&page=Template:Geobox>

³E.g. <https://www.ethnologue.com/subgroups/coosan>

which commonly are considered disjoint. There is thus no reason for making it transitive.

6.3. Comparison with baseline. Comparing the obtained results, we observe the following. First, for reflexivity, on the considered dataset, the approach does not matter. One can easily suppose that this is a general observation, as reflexivity is an extremely potent characteristic, forcing all individuals in an ontology to occur in the considered property, and thus unlikely to be used. Irreflexivity is, on the other hand, probably a rather common characteristic. For example, if domain and range of a property are disjoint, that alone is enough to deduce irreflexivity of the property, as there cannot be an individual s , such that (s, p, s) is true and s belongs to both domain and range.

For transitivity, the baseline without additional requirement on the size of the set C failed, as presented in Table 5. With the requirement, in the top 10, there still are highly suspicious results, e.g., properties `dbo:genus` and `dbo:class` are non-transitive, as discussed in the previous section.

There is a glaring difference in scores between the baseline and the neural network for `dbo:state`. Analysis of the dataset reveals that this is due to two spurious assertions in DBpedia: `dbr:Azores dbo:state dbr:Azores` and `dbr:Madeira dbo:state dbr:Madeira`. As there are numerous individuals that are in the subject position for triples of shape `o dbo:state dbr:Azores` and `o dbo:state dbr:Madeira`, these two spurious triples are enough to produce this disparity.

Transitivity of `dbo:starring` is a byproduct of numerous triples with this property and the subject equal to the object. These triples by themselves make no sense, but they also produce nonsensical chains of reasoning, e.g., because `dbr:Panaah dbo:starring dbr:Panaah` and `dbr:Panaah dbo:starring dbr:Panaah` (the same triple), it follows that `dbr:Panaah dbo:starring dbr:Panaah`.

6.4. Comparison with the gold standard. In order to answer the research question, we must summarize the results presented in Table 3 and Table 4, which can be seen either as ranking comparison or as measuring classification accuracy. The rationale for treating the reported values as rankings is that one might expect a score to represent confidence in the presence of considered characteristics, and thus expect that values ranked higher according to the score are more likely to have the characteristic. The approach of comparing rankings requires from the method the ability to handle ties in a ranking, we thus used Spearman's rank correlation coefficient ρ (higher is better). The correlation for the transitivity scores reported in Table 3 is $\rho(\text{neural}, \text{gold}) = 0.310$ and $\rho(\text{baseline}, \text{gold}) = 0.170$, and for the symmetry scores reported in Table 4 $\rho(\text{neural}, \text{gold}) = 0.454$ and $\rho(\text{baseline}, \text{gold}) = 0.555$.

To measure classification accuracy two considerations must be made. First, jointly assessing symmetry and asymmetry may be either considered as a ternary classification problem (symmetric/asymmetric/neither) or as two binary problems: symmetric vs. not-symmetric and asymmetric vs. not-asymmetric.

Second, the scores from the proposed solutions are on a continuous scale, whereas the gold standard is on a discrete scale and thus one must either apply a measure capable of dealing with it or introduce additional hyperparameters as cut-off points to discretize the scores.

Because there are established measures capable of dealing with the disparity between continuous and discrete scales and introducing additional hyperparameters would only cloud the results further, we concentrate only on using an established measure of the area under the retriever operating characteristic curve AUC (higher is better). Unfortunately, it does not easily generalize to a higher number of classes, we thus split the ternary classification problem into two binary classification problems, as described above.

For the transitivity scores reported in Table 3 $AUC(\text{neural}, \text{gold}) = 0.707$ and $AUC(\text{baseline}, \text{gold}) = 0.613$. For the symmetry scores reported in Table 4 we introduce two helper vectors, gold^s and gold^a representing the gold labels in the binary classification problems of, respectively, symmetric vs not-symmetric and asymmetric vs not-asymmetric:

$$\text{gold}_i^s = \begin{cases} 1 & \text{gold}_i = 1 \\ 0 & \text{gold}_i \in \{0, 0.5\} \end{cases} \quad \text{gold}_i^a = \begin{cases} 1 & \text{gold}_i \in \{1, 0.5\} \\ 0 & \text{gold}_i = 0 \end{cases}$$

Using these, we were able to compute the AUC scores: $AUC(\text{neural}, \text{gold}^s) = 0.790$, $AUC(\text{baseline}, \text{gold}^s) = 0.857$, $AUC(\text{neural}, \text{gold}^a) = 0.722$, $AUC(\text{baseline}, \text{gold}^a) = 0.764$.

Using both approaches for comparison we arrive at the same conclusion: the approach presented in Section 4 outperforms the baseline approach when it comes to the task of deciding on transitivity, while the simpler baseline outperforms the proposed approach in the task of deciding on symmetry and asymmetry. From this, we infer that the proposed approach is capable of extracting explanations as ontological axioms and that in more complex scenarios it has the potential to outperform baseline solutions.

7. Conclusions and future work

In this paper, we presented an approach to train a neural network and explain its behavior in order to extend an ontology with property characteristic axioms. The neural network is recurrent and multi-layer, with 5 layers of 200 Gated Recurrent Units each. The training is conducted in a classification setup, where the goal of the network is to discriminate between RDF triples present and absent in the considered RDF graph. The explanation generation is based on averaging the responses of the network for the triples that were not presented during training. The explanations are OWL 2 property characteristic axioms and the proposed solutions cover all of them. Comparing the obtained results with the gold standard we observe that the obtained axioms must be treated only as suggestions for an ontology engineer, as DBpedia is at the same time incomplete and noisy, leading to learning knowledge that may be untrue in a general way.

The difference between the generated axioms and a baseline is the most striking for the most complex characteristic, which is transitivity, which requires joint consideration of three RDF triples. The proposed approach is, due to its recurrent nature, very suitable for such a task. The obtained results hints that the proposed recurrent architecture could be employed to obtain axioms in setups requiring considering even more RDF triples at once, which will be considered in the future work.

Acknowledgements. We acknowledge support from the grant 09/91/DSMK/0659 and by the National Centre for Research and Development, Poland, grant no. LIDER/14/0086/L-10/18/NCBR/2019. We thank Aleksandra Walkiewicz of Nicolaus Copernicus University in Toruń and Patrycja Potoniec of The Institute of Literary Research of the Polish Academy of Sciences for invaluable linguistic consultations.

REFERENCES

- [1] D. Choiński and M. Senik, “Distributed control systems integration and management with an ontology-based multi-agent system”, *Bull. Pol. Ac.: Tech.* 66 (5), 613–620 (2018).
- [2] P. Rózewski and O. Zaikin, “Integrated mathematical model of competence-based learning-teaching process”, *Bull. Pol. Ac.: Tech.* 63 (1), 245–259 (2015).
- [3] O. Zaikin, R. Tadeusiewicz, P. Rózewski, L.B. Kofoed, M. Malinowska, and A. Żyławski, “Teachers’ and students’ motivation model as a strategy for open distance learning processes”, *Bull. Pol. Ac.: Tech.* 64 (4), 943–955 (2016).
- [4] S. Ambroszkiewicz, W. Bartyna, M. Faderewski, and G. Terlikowski, “Multirobot system architecture: environment representation and protocols”, *Bull. Pol. Ac.: Tech.* 58 (1), 3–13 (2010).
- [5] M. Schneider, “OWL 2 web ontology language RDF-based semantics (second edition)”, W3C, W3C Recommendation, 2012.
- [6] K. Krawiec, R. Slowinski, and I. Szczesniak, “Pedagogical method for extraction of symbolic knowledge from neural networks”, in *Rough Sets and Current Trends in Computing, First International Conference*, ser. Lecture Notes in Computer Science, vol. 1424, Springer, 1998, pp. 436–443. doi: 10.1007/3-540-69115-4_60.
- [7] D. Wood, M. Lanthaler, and R. Cyganiak, “RDF 1.1 concepts and abstract syntax”, W3C, W3C Recommendation, 2014. <http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/>.
- [8] G. Carothers and E. Prud’hommeaux, “RDF 1.1 turtle”, W3C, W3C Recommendation, 2014. <http://www.w3.org/TR/2014/REC-turtle-20140225/>.
- [9] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P.N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, “Dbpedia – A large-scale, multilingual knowledge base extracted from wikipedia”, *Semantic Web* 6 (2), 167–195 (2015). doi: 10.3233/SW-140134.
- [10] M. Horridge and P. Patel-Schneider, “OWL 2 web ontology language manchester syntax (second edition)”, W3C, W3C Note, 2012. <http://www.w3.org/TR/2012/NOTE-owl2-manchester-syntax-20121211/>.
- [11] J. Potoniec, P. Jakubowski, and A. Lawrynowicz, “Swift linked data miner: Mining OWL 2 EL class expressions directly from online RDF datasets”, *J. Web Semant.* 46–47, 31–50 (2017). doi: 10.1016/j.websem.2017.08.001.
- [12] J. Völker, D. Fleischhacker, and H. Stuckenschmidt, “Automatic acquisition of class disjointness”, *J. Web Semant.* 35, 124–139 (2015). doi: 10.1016/j.websem.2015.07.001.
- [13] A. Lawrynowicz and J. Potoniec, “Pattern based feature construction in semantic data mining”, *Int. J. Semantic Web Inf. Syst.* 10 (1), 27–65 (2014).
- [14] J. Potoniec and A. Lawrynowicz, “Combining ontology class expression generation with mathematical modeling for ontology learning”, in *Proc. of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI Press, 2015, pp. 4198–4199 [Online] Available: <http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9526>.
- [15] J. Lehmann, S. Auer, L. Bühmann, and S. Tramp, “Class expression learning for ontology engineering”, *J. Web Semant.* 9 (1), 71–81 (2011). doi: 10.1016/j.websem.2011.01.001.
- [16] D. Fleischhacker, J. Völker, and H. Stuckenschmidt, “Mining RDF data for property axioms”, in *On the Move to Meaningful Internet Systems: OTM 2012, Confederated International Conferences: CoopIS, DOA-SVI, and ODBASE 2012*, ser. Lecture Notes in Computer Science 7566. Springer, 2012, pp. 718–735. doi: 10.1007/978-3-642-33615-7_18.
- [17] G. Rizzo, C. d’Amato, N. Fanizzi, and F. Esposito, “Tree-based models for inductive classification on the web of data”, *J. Web Semant.* 45, 1–22 (2017). doi: 10.1016/j.websem.2017.05.001.
- [18] L. Galárraga, C. Teflioudi, K. Hose, and F.M. Suchanek, “Fast rule mining in ontological knowledge bases with AMIE+”, *VLDB J.* 24 (6), 707–730 (2015).
- [19] J. Lehmann and J. Völker, *Perspectives on Ontology Learning*, ser. Studies on the Semantic Web. vol. 18. IOS Press, 2014.
- [20] F. Horn and K.-R. Müller, “Predicting pairwise relations with neural similarity encoders”, *Bull. Pol. Ac.: Tech.* 66 (6), 821–830 (2018).
- [21] P. Ristoski and H. Paulheim, “Rdf2vec: RDF graph embeddings for data mining”, in *The Semantic Web – ISWC 2016 – 15th International Semantic Web Conference*, ser. Lecture Notes in Computer Science, vol. 9981, 2016, pp. 498–514. doi: 10.1007/978-3-319-46523-4_30.
- [22] B. Shi and T. Weninger, “Proje: Embedding projection for knowledge graph completion”, in *Proc. of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI Press, 2017, pp. 1236–1242. [Online]. Available: <http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14279>.
- [23] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data”, in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*, 2013, pp. 2787–2795. [Online]. Available: <http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data>.
- [24] G. Petrucci, M. Rospocher, and C. Ghidini, “Expressive ontology learning as neural machine translation”, *J. Web Semant.* 52–53, 66–82 (2018). doi: 10.1016/j.websem.2018.10.002.
- [25] W. Song, Z. Duan, Z. Yang, H. Zhu, M. Zhang, and J. Tang, “Explainable knowledge graph-based recommendation via deep reinforcement learning”, *CoRR*, vol. abs/1906.09506, 2019.
- [26] P.G. Omran, K. Wang, and Z. Wang, “Scalable rule learning via learning representation”, in *Proc. of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pp. 2149–2155. ijcai.org, 2018.
- [27] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, “Embedding entities and relations for learning and inference in knowledge

- bases”, in *3rd International Conference on Learning Representations, ICLR 2015*, 2015.
- [28] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation”, in *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, ACL*, 2014, pp. 1724–1734.
- [29] Y. Gal and Z. Ghahramani, “A theoretically grounded application of dropout in recurrent neural networks”, in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, Barcelona, Spain, 2016, pp. 1019–1027.
- [30] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling”, *CoRR*, vol. abs/1412.3555, 2014.
- [31] Wikipedia contributors, “Template: infobox person – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Template:Infobox_person&oldid=947719598, 2020 [Online; accessed 6-May-2020].
- [32] Wikipedia contributors, “Sister city – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Sister_city&oldid=952706943, 2020 [Online; accessed 6-May-2020].
- [33] N. Clarke, “Globalising care? town twinning in Britain since 1945”, *Geoforum* 42 (1), 115–125, 2011.
- [34] Wikipedia contributors, “Template: infobox automobile – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Template:Infobox_automobile&oldid=948513824, 2020 [Online; accessed 6-May-2020].
- [35] Wikipedia contributors, “Sister paper – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Sister_paper&oldid=918056329, 2019 [Online; accessed 6-May-2020].
- [36] Wikipedia contributors, “Sister station – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Sister_station&oldid=944023428, 2020 [Online; accessed 6-May-2020].
- [37] Wikipedia contributors, “Template: infobox office holder – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Template:Infobox_officeholder&oldid=893739564, 2020 [Online; accessed 6-May-2020].
- [38] Wikipedia contributors, “Template: infobox canal: Difference between revisions – Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Template%3AInfobox_canal&type=revision&diff=951115764&oldid=751761235, 2020 [Online; accessed 6-May-2020].
- [39] W.R. Thompson, “Identifying rivals and rivalries in world politics”, *International Studies Quarterly* 45 (4), 557–586 (2001). doi: 10.1111/0020-8833.00214.
- [40] C.M. Keet and A. Artale, “Representing and reasoning over a taxonomy of part-whole relations”, *Applied Ontology* 3 (1-2), 91–110 (2008). doi: 10.3233/AO-2008-0049.
- [41] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari, “Wonderweb deliverable D18: Ontology library (final)”, Tech. Rep., IST Project 2001-33052 WonderWeb, 2003. <http://www.loa.istc.cnr.it/old/Papers/D18.pdf>.
- [42] D.M. Eberhard, G.F. Simons, and C.D. Fennig, Eds., *Ethnologue: Languages of the World. Twenty-third edition*. Dallas, Texas: SIL International., 2020. Online version: <http://www.ethnologue.com>.
- [43] W. Ride, H. Cogger, C. Dupuis, O. Kraus, A. Minelli, F.C. Thompson, and P. Tubbs, Eds., *International code of zoological nomenclature Fourth Edition*. The International Trust for Zoological Nomenclature, 1999.