Web Semantics: Science, Services and Agents on the World Wide Web 81 (2024) 100816

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

# Extraction of object-action and object-state associations from Knowledge Graphs

Alexandros Vassiliades [a,b,*,1], Theodore Patkos [b,1], Vasilis Efthymiou [b], Antonis Bikakis [c], Nick Bassiliades [a], Dimitris Plexousakis [b]

[a] Aristotle University of Thessaloniki, School of Informatics, Thessaloniki, 541 24, Central Macedonia, Greece
[b] Institute of Computer Science, Foundation for Research and Technology Hellas, Heraklion, 700 13, Crete, Greece
[c] University College London Department of Information Studies, London, WC1E 6BT, Greater London City of London, United Kingdom

ARTICLE INFO

*Keywords:*
Association extraction
Knowledge Graphs
Semantics-based extraction
Topology-based extraction
Linking entities

ABSTRACT

Infusing autonomous artificial systems with knowledge about the physical world they inhabit is a critical and long-held aim for the Artificial Intelligence community. Training systems with relevant data is a typical approach; however, finding the data required is not always possible, especially when much of this knowledge is commonsense. In this paper, we present a comparison of topology-based and semantics-based methods for extracting information about object-action and object-state association relations from knowledge graphs, such as ConceptNet, WordNet, ATOMIC, YAGO, WebChild and DBpedia. Moreover, we propose a novel method for extracting information about object-action and object-state associations from knowledge graphs. Our method is composed of a set of techniques for locating, enriching, evaluating, cleaning and exposing knowledge from such resources, relying on semantic similarity methods. Some important aspects of our method are the flexibility in deciding how to deal with the noise that exists in the data, and the capability to determine the importance of a path through training, rather than through manual annotation.

## 1. Introduction

Infusing autonomous artificial systems with knowledge about the physical world they inhabit is a critical and long-held aim for the Artificial Intelligence (AI) community. Training systems with relevant data is a typical approach; however, finding the data required is not always possible, especially when much of this knowledge is commonsense. A method that can correctly identify positive and negative associations between entities by exploiting knowledge stored in Knowledge Graphs (KGs) in the presence of noise can increase the quality of data that a machine can utilize. This can improve the performance of autonomous AI systems, such as cognitive robotic systems operating in a household environment, Computer Vision modules, and other AI application domains. Yet, constructing a generic method for extracting positive and negative associations seems a far catch for the time being.

Humans are able to identify meaningful associations, by relying not only on observations, but also on their commonsense knowledge. Machines, on the other hand, require a vast amount of data, in order to be properly trained and learn the various association relationships. KGs, such as ConceptNet [1], WordNet [2], ATOMIC [3], WebChild [4], YAGO [5] and DBpedia [6] contain to some extent knowledge about association relations, which can help data-driven models to train classifiers. The knowledge that exists in such KGs though is typically inserted via crowd-sourced methods and often contains a portion of inaccurate or noisy data. As a result, when extracting or retrieving knowledge from such KGs, evaluation procedures are critical [7].

In this paper, we compare a number of methods of different nature that are commonly used in practice for the extraction of associations from KGs, concentrating our attention on the household application domain. We organize the methods into *topology-based* and *semantics-based* ones, and also introduce a novel semantics-based approach to extract associations from KGs, which can achieve or improve state-of-the-art performance, while offering flexibility in ironing out noise. Its

* Corresponding author at: Aristotle University of Thessaloniki, School of Informatics, Thessaloniki, 541 24, Central Macedonia, Greece.
*E-mail addresses:* valexande@csd.auth.gr (A. Vassiliades), patkos@ics.forth.gr (T. Patkos), vefthym@ics.forth.gr (V. Efthymiou), a.bikakis@ucl.ac.uk (A. Bikakis), nbassili@csd.auth.gr (N. Bassiliades), dp@ics.forth.gr (D. Plexousakis).
*URLs:* https://intelligence.csd.auth.gr/people/vassiliades-alexandros/ (A. Vassiliades), https://www.csd.uoc.gr/~patkos/ (T. Patkos), https://sites.google.com/site/vefthym/ (V. Efthymiou), https://www.ucl.ac.uk/information-studies/antonis-bikakis (A. Bikakis), https://intelligence.csd.auth.gr/people/bassiliades/ (N. Bassiliades), https://www.ics.forth.gr/person/plexousakis/dimitris (D. Plexousakis).

[1] This project has received funding from the Hellenic Foundation for Research and Innovation (HFRI), Greece and the General Secretariat for Research and Technology (GSRT), Greece, under grant agreement No 188.

https://doi.org/10.1016/j.websem.2024.100816
Received 14 January 2022; Received in revised form 20 March 2023; Accepted 9 March 2024
Available online 19 March 2024

main characteristic is the exploitation of patterns of relations, which carry important information as to which associations to trust and which to dismiss. Moreover, we evaluate the aforementioned methods over various KGs, such as ConceptNet, ATOMIC, WebChild, YAGO and DBpedia, which, to the best of our knowledge, constitutes one of the most extensive evaluations of these methods over association relations. Moreover, we compare how data-driven models perform over the same task, i.e., that of link prediction between two elements.

More specifically, the problem we investigate is, given a directed KG, whether two nodes are associated or not, one of which relates to a household *object class* and the other to an *action* or an *object state class*. The methods considered in our study are domain-agnostic and are not confined to the household domain; nonetheless, as we explain in the sequel, their use is very popular for this particular domain, therefore our choice helps obtain a common reference level for a comparative analysis. Furthermore, the plethora of features that this domain introduces makes the problem non-trivial.

The main contributions of this paper are the following:

- A comparative analysis of popular methods for extracting associations from KGs, focusing on the household domain.
- The proposal of a new, enhanced method that better exploits the semantic knowledge that exists in the KG, in order to extract object-action/state association relations.
- An extensive analysis of the object-related information existing in ConceptNet, ATOMIC, WebChild, YAGO and DBpedia for association relations.
- The generation of a dataset of positive and negative object-action and object-state relations, comprising labels that are commonly used for benchmarking both research and practical approaches.

Our method and the dataset are publicly available.[2]

This paper is based on and significantly extends the work presented in [8], by: (a) studying methods for extracting object-state (in addition to object-action) associations and (b) considering a much broader set of KGs for the evaluation of the different methods, whereas [8] concentrates on ConceptNet, exclusively.

The rest of the paper is organized as follows: Section 2 presents the motivation of our study. Section 3 discusses related work. The existing and proposed approaches for extracting object-action and object-state relations are presented in Section 4. The experimental assessment is described in Section 5, the results are discussed in Section 6, and the study concludes in Section 7.

## 2. Motivation

Filtering information to infer associations from problem-agnostic KGs in the presence of noise is a long-lasting goal in many fields related to AI research. In this paper, we focus on an instance of the problem pertaining to the identification of associations among concepts that exists in a KG, that of positive and negative object-action and object-state association. We were motivated mostly by the plurality of methods that are being used in practice for that purpose and to that particular domain; therefore, we decided to compare their performance, aiming to identify which characteristics of each take advantage of the nature of the underlying data, e.g., the structure, semantics, context etc.

The types of associations we are focusing on constitute valuable information for a wide spectrum of application areas, especially in the field of Robotics. Cognitive and social robots need to operate in environments populated by a wide variety of objects; the identification of proper correlations among the objects and the actions that the humans perform on them or the states obtained by these objects can become an important leverage in understanding how to classify or even to operate novel appliances [9]. It can also significantly enhance human–robot interactions and collaboration [10,11]. But even in the fields, such as Computer Vision or Ambient Intelligence and smart spaces, the identification of object-action-state associations can help address traditional problems, such as action or activity recognition [12–14].

Finally, through this comparative analysis, we also wanted to spot differences in the data that exist in some of the most popular KGs, which may affect the performance of the methods. For example, the overuse of certain properties, as noticed for instance in ConceptNet with the *RelatedTo* property, tends to work inversely to the semantic information that it can offer. Our criteria for choosing which KGs to consider were: (a) the KG has a taxonomy linked to WordNet, as the relation with WordNet is needed in our methodology, (b) the KG contains *object-state* relations OR *object-affordance* relations (affordances of one object are the real-world actions that can be performed on/with that object), and (c) the KG is publicly available.

The challenge for the methods we consider mostly lies in the existence of the noise in the KGs. As noise in a KG we consider: (i) conflicting information, (ii) wrongly annotated information (due to the crowdsourced nature of many knowledge graphs), (iii) the heterogeneity of granularity of node population in the KG, meaning that in some areas of the KG there may exist many interconnected nodes about some sub-domain, which leads to an over-fitting of knowledge for these sub-domains, whereas in some other areas nodes describing another sub-domain may be sparse, and (iv) the heterogeneity of granularity in properties, meaning that some properties are used so often that they could be considered as super properties, among others (see Section 5.3 and Section 6). Scalability is another challenge faced, especially since we are contrasting the performance against generic repositories.

## 3. Related work

This section first presents studies about general purpose associations (i.e., the entities which are associated can be of any type). Then, it proceeds to studies with object-action associations, and concludes with studies about object-state associations. This loosely-defined relatedness between two concepts falls under the broader task of link prediction in Knowledge Graphs, which we refer to as associations in our study.

### 3.1. General purpose associations

Retrieving commonsense information from problem-agnostic repositories has been used to tackle challenges in a variety of AI-related disciplines. ConceptNet is used by the authors of [15] to find word similarities, which they subsequently utilize to improve the performance of sentence-based picture retrieval methods. The authors use the labels detected in an image to retrieve information from the ConceptNet triplets that contain the detected labels. In [16], the authors use KGs to solve the problem of zero-shot label learning in photos by building KGs based on labels identified visually and correlations established in external sources. The authors utilize WordNet to populate the graph and Wu Palmer similarity[3] to generate property labels. The authors of [17], use Bayesian logic networks to give labels to the objects in a picture and rely on commonsense knowledge derived from WordNet and ConceptNet. With the help of WordNet hypernyms, seed words are disambiguated. ConceptNet attributes like *LocatedAt* and *UsedFor*, which can help locate an object's location, are also obtained. The system can then construct a compact semantic knowledge base using this method with only a limited number of objects. In [18,19], the authors infer the label of the room through the objects that the cognitive robotic system perceived from its vision module. The authors use the DBpedia comment boxes of the objects in the room in order to infer the label of the room.

[2] https://github.com/valexande/AssociationKG

[3] https://www.nltk.org/howto/wordnet.html

The studies listed above aim to incorporate knowledge from general-purpose Web resources identified in a KG without paying close attention to the veracity of the information retrieved from such resources. Furthermore, they rely on the simplistic premise that if two nodes are connected by an edge, they are semantically related. On the other hand, we are interested in techniques that may filter out the noise or incorrect information that may exist in such Web resources, before adding new knowledge to a KG. Furthermore, we offer a mechanism for associating objects with actions and states which is not covered in these studies. In contrast to the prior studies, we evaluate several approaches across a larger number of KGs.

The study of Zhou et al. [20] is more comparable to ours. To anticipate a path between two nodes in the ConceptNet graph, the authors train a Long Short-Term Memory (LSTM) model. The authors collect the most qualitative pathways for a set of node pairs, defining quality as the most natural set of edges connecting two nodes. For instance, the path $Lead \xrightarrow{\text{HasProperty}} Toxic \xleftarrow{\text{RelatedTo}} Lethal \xleftarrow{\text{RelatedTo}} Poison$ is considered the most natural among those connecting *Lead* and *Poison*. The quality of paths is annotated manually by a group of volunteers. A data-driven model predicts a path between two ConceptNet nodes in [21,22]. The quality of a path is hand-coded by the authors. Our method, on the other hand, uses training rather than manual annotation to identify the relevance of a path. This has two advantages: (*i*) it considers the structural and semantic properties of the underlying KG to a greater extent, and (*ii*) it is more adaptable to changes in the KG or application domain.

Interesting studies in the area of explainable recommendation over KGs are [23–25], where the notion of patterns of relations is also introduced. Similarly to our study a relation pattern is a specific sequence of relations that contains semantically rich information for two entities. But as it is easily understood the problem we are addressing is different than the one in the aforementioned studies, as in our work we concentrate more on information retrieval for entity linking. Also, even though the authors use various datasets to prove the scalability of their method, they use the same underlying KG, which is Amazon's KG.[4]

### 3.2. Object-action associations

Many studies in the field of cognitive robotics have focused on the representation and recognition of object-action relations. The semantic correlation of physical entities is captured in the KnowRob [26] and RoboSherlock [27] projects, but object-action relations are either learnt entirely through observed data or captured in a problem-specific method. The authors integrate ConceptNet knowledge into a KG in [28]. They build ConceptNet subgraphs with only two properties given an object or action label in order to train a data-driven model that can predict if an object is associated to an activity. RoboCSE [29], which employs embeddings to encode object and action labels and infer object-action links based on the similarity of their vectors, follows a similar method. Our proposed method combines both semantically relevant and commonsense information stored in general-purpose repositories, which can be used to supplement and enhance the findings of the previous studies.

The studies [30,31] propose a method where Markov Logic Networks are used in order to relate real-world objects with their affordances, in a zero-shot learning problem that tackles the need of training classifiers. Even though their method seems more scalable than a data-driven model, the information that the method can utilize exists solely in the Markov Logic Network, which cannot access external knowledge (e.g., a Semantic Web KG). On the other hand, our method is not restricted to a specific KG, as it can retrieve information from any given KG, which has different types of relations.

[4] https://aws.amazon.com/neptune/knowledge-graphs-on-aws/

### 3.3. Object-state associations

The problem of object-state association is referred in the literature as *state detection*, and is mostly encountered in the field of computer vision. The problem of state detection usually serves as a stepping stone to achieve action recognition.

In [32], state detection is studied in the context of videos containing manipulation actions performed upon seven classes of objects. The authors formulate state detection as a discriminative clustering problem and attempt to address it by optimization methods. [33] represents state-altering actions as concurrent and sequential object fluents (states) and utilize a beam search algorithm for fluent detection and action recognition. Similarly, [34] explores state detection in tandem with action recognition. The method is based on the learning of appearance models of objects and their states from video frames which are used in conjunction with a state transition matrix which maps action labels into a pre-state and a post-state. In [35], the states and transformations of objects/scenes on image collections are studied and the learned state representations are extended to different object classes. [36,37] examine the causal relations between human actions and object fluent changes. [38] develops a weakly supervised method to recognize actions and states of manipulated objects before and after the action, proposing a weakly supervised method for learning the object and material state that are needed for recognizing daily actions. [39] designs a Siamese network to model precondition states, effect states and their associate actions. Jiang in [40] defines a Multi-Agent System framework, where each agent maintains an incomplete and noisy perspective of the world. Jiang on top of the different perspectives that each agent has, develops a graph neural network that exploits the information in the various KGs to learn how to predict a post-state for the objects when an action is performed on them.

In most cases, state detection is a problem encountered in computer vision and is used as a means for action recognition. Therefore, the amount of object-state relations is restricted and classifiers are trained for each individual object-state relation. We treat the identification of object-state associations as a standalone problem and develop a method to address it that relies only on information available in a KG, which is more scalable than a classifier. Moreover, our method can identify a greater amount of object-state relations.

## 4. Methodology

In order to evaluate the performance of each method, we follow a number of steps for preparing the data, shown in Fig. 1. The first two pre-processing steps in this pipeline are described in Section 4.2, but first we start with the formulation of the problem in Section 4.1. Then, we analyze the methods that utilize the topological features of the underlying KG (Section 4.3), and continue with the methods that rely on the semantics of the nodes and their connections (Section 4.4).

Our proposed **Relation Pattern Method** is included in the latter group. Finally, the decision problem regarding a given association, i.e., whether the association is positive or negative (the 'Conclusion' step in Fig. 1), can be answered by comparing the confidence value of each method to a threshold (Section 4.5). This threshold can be learned from the training data, as we explain in our experimental evaluation.

### 4.1. Problem formulation

The problem we aim to solve is: given a directed knowledge graph $G = (E, R)$, where $E$ denotes the set of nodes that correspond to entities, $R$ denotes the set of edges that correspond to relations (i.e., $R$ contains triples of the form $(t_1, r, t_2)$ where $t_1, t_2 \in E$ and $r$ denotes a relation between $t_1$ and $t_2$), and a pair of nodes $(e_1, e_2)$ with $e_1, e_2 \in E$, where $e_1$ represents an action or a state and $e_2$ an object ($E$ may contain other types of nodes as well), find whether $e_1$ and $e_2$ are *related*. If $e_1$ is an action and $e_2$ is an object, we consider these two nodes related

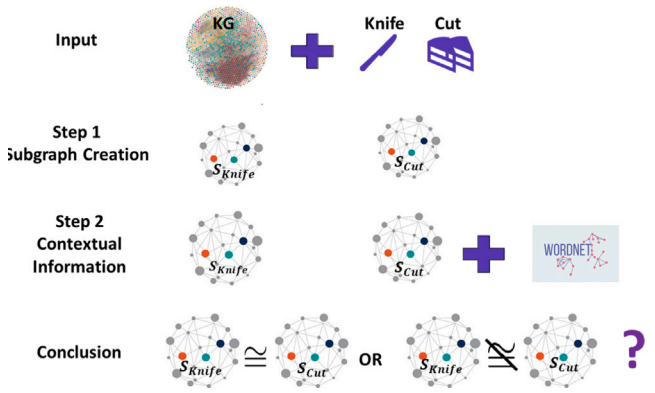


**Fig. 1.** Pipeline of the Problem.

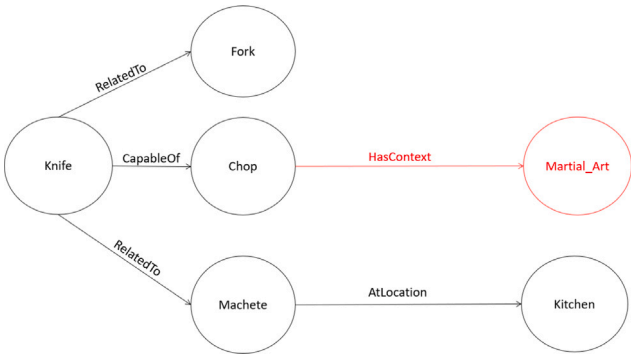


**Fig. 2.** Part of the subgraph for the label *Knife*. The red node is pruned in **Step 2**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

if the following question yields a positive answer: *"Can the action* $e_1$ *be performed by/on the object* $e_2$*?"*. For instance, the question *"Can the action Fold be performed by the object Knife?"* should yield a negative answer. Similarly, if $e_1$ is a state and $e_2$ an object, we consider the two nodes $e_1$ and $e_2$ related if the following question yields a positive answer: *"Can object* $e_2$ *be in the state* $e_1$*?"*. For example, the question *"Can the object Knife obtain the state Dirty?"* should yield a positive answer.

### 4.2. Pre-processing of subgraphs

We first describe how we can generate a graph $G'$ that helps us answer the aforementioned questions, from a given knowledge graph $G$ and a given collection of labels $L$ that relate to real-world objects, actions, and states, and then, we show the methods we assessed to tackle the aforementioned problem. We extract the object, action and state labels from the Something-Something Dataset,[5] a dataset that is commonly used by the Computer Vision community (see Section 5.1 for more details); yet, any set of object, action, state labels can be used to create $G'$. Moreover, notice that for each KG (i.e., ConceptNet, ATOMIC, WebChild, YAGO and DBpedia), we get a different $G'$ and a different set of labels $L$ (see Section 5.1 for more details). Now, for every $G$ and $L$, we take every label $l_i \in L$ and generate a graph $S_i$, by appending information relevant to $l_i$ from each KG $G$ that we have at hand. We construct $G'$ by unifying all $|L|$ graphs $S_1, \ldots, S_{|L|}$, i.e., every graph $S_i$ is a subgraph of $G'$. Notice that for constructing $G'$, we do not omit any noise (see Section 2).

**Step 1:** For each object, action or state label, we search for a node with the same lemmatized label in the KG at hand and extract a subgraph containing a set of the properties found that are considered relevant to the domain of interest. More specifically, for ConceptNet we hand-picked the relations shown below. ConceptNet, due to its very good documentation,[6] enabled us to comprehend what each relation represents and we omitted only 2 relations: *Desires*, which, while seemingly relevant, is human centric and explains the emotions that are elicited in humans as a result of an event, and *ExternalURL*, to avoid appending information from external sites other than WordNet.

The edge types for ConceptNet that we consider are:

$$\{RelatedTo, UsedFor, CapableOf, LocatedAt, HasA,$$
$$IsA, Synonym, Antonym, DefinedAs, SimilarTo, PartOf,$$
$$EtymologicallyRelatedTo, EtymologicallyDerivedFrom,$$
$$HasLastSubevent, HasPrerequisite, CreatedBy, Causes,$$
$$ReceivesAction, DistinctFrom, DerivedFrom, MadeOf,$$
$$MannerOf, FormOf, HasContext, HasFirstSubevent,$$
$$AtLocation, HasProperty, HasSubevent, LocatedNear,$$
$$SymbolOf\}$$

For DBpedia, we also omitted some properties in order for the subgraphs which are created to reflect the nature of our problem. We give some examples here, but notice that more properties than the ones mentioned here were omitted (a complete list existing in our documentation). The property *wikiPageWikiLink* was omitted because we did not want information that does not belong into the DBpedia ontology. The property *wikiPageRedirects* was omitted because in most cases Wikipedia redirection lists contain noise, as they relate entities with other contextually irrelevant entities. For instance, the object *pan* is related, among others, with *Pediatric acute-onset neuropsychiatric syndrome (PANS), Peter Pan,* and *Pan the God*.

The other three KGs we considered, i.e., ATOMIC, YAGO, and WebChild, do not provide a detailed documentation for the relations they contain and we therefore decided to use all the relations.

The subgraphs contain either 1-hop or 2-hop paths from the object, action, or state label (see Section 5 for more details).

**Step 2:** In this step, we include context information. We collect information from WordNet by examining the super-classes of each node in the subgraph formed in **Step 1**; if any super-class of a node fits into a domain-specific category of super-classes, the node is kept in the graph; otherwise, it is deleted. The super-classes we consider are:

$$\{abstraction, physical\_entity, thing, attribute, group,$$
$$measure, set, causal\_agent, matter, object, process,$$
$$change, otherworld, substance, communication\}$$

We have chosen this set of classes based on the findings of [41] that practically every node in the WordNet directed acyclic graph that refers to a real-world object, action, or state has at least one of these as a super-class. When interested in household appliances, for example, this enrichment based on WordNet super-classes can provide domain-specific notions. Fig. 2 is a portion of the subgraph produced from the ConceptNet KG for the label *Knife*. The node that was pruned in **Step 2** is highlighted in red, which was omitted because it was considered out of context.

After creating a subgraph for each object, action and state label, as described in **Steps 1** and **2**, we end up with a set of graphs $\{S_1, \ldots, S_n\}$, such that $S_i = (E_i, R_i)$ for $i = 1, \ldots, n$, where $E_i$ is the set of nodes and $R_i$ the set of edges in $S_i$. Thus, the final graph is defined as $G' = (E', R')$, where $E' = \bigcup_{i=1}^{n} E_i$ and $R' = \bigcup_{i=1}^{n} R_i$.

[5] https://paperswithcode.com/dataset/something-something-v1
[6] https://github.com/commonsense/conceptnet5/wiki/Relations

### 4.3. Topology-based relevance

In order to determine whether two nodes are related, we consider two of the most popular methods found in relevant literature [42,43] that exploit the topology of a graph.

**Connecting Paths Method**: This method considers each sequence of edges that starts at the object node and ends at the action or state node after a finite number of steps, or vice versa . The method omits paths that contain loops, but does not take into account the type of edges a path contains. Given two subgraphs $S_1$ and $S_2$, which correspond to an object node and an action (or state) node, respectively, as stated in Section 4.1, the *connectPath* metric for $S_1$ and $S_2$ is defined as:

$$connectPath(S_1, S_2) = \frac{|C_1 \cup C_2|}{|P_1 \cup P_2|} \tag{1}$$

where $C_1$ is the set of paths that start from the object node and reach the action (or state) node, $C_2$ is the set of paths that start from the action (or state) node and reach the object node, $P_1$ is the set of all paths that start from the object node and $P_2$ the set of all paths that start from the action (or state) node. Since $(C_1 \cup C_2) \subseteq (P_1 \cup P_2)$, it follows that $0 \leq connectPath \leq 1$.

**Example 1.** Let $S_{knife}$ be the subgraph for the object node *knife* and $S_{fold}$ be the subgraph for the action node *fold*, created from the ConceptNet KG and let $S_{knife}$ have two paths that start from the node knife, namely Knife $\xrightarrow{\text{CapableOf}}$ Cut and Knife $\xrightarrow{\text{LocatedAt}}$ Pocket $\xrightarrow{\text{RelatedTo}}$ Wallet $\xrightarrow{\text{RelatedTo}}$ Fold and $S_{fold}$ have only one path, Fold $\xrightarrow{\text{HasContext}}$ Cooking $\xrightarrow{\text{RelatedTo}}$ Spatula $\xrightarrow{\text{RelatedTo}}$ Knife. Then, the connectPath metric will return

$$connectPath(S_{knife}, S_{fold}) = \frac{|C_{knife} \cup C_{fold}|}{|P_{knife} \cup P_{fold}|}$$

$$= \frac{1+1}{2+1} = 0.667$$

Recent studies using this method, with minor adjustments, have focused on inferring object-action relations [16,21] as well as object identification [15,20].

**Common Nodes Method**: Given two subgraphs, the *Common Nodes Method* divides the number of common nodes by the total number of nodes. When two nodes refer to the same entity in the KG at hand (ConceptNet, ATOMIC, YAGO, DBpedia and WebChild), i.e., the nodes have the same label, they are called common. Duplicate nodes are removed, leaving each node with only one instance. The commonNodes metric between two subgraphs $S_1$ and $S_2$ is defined as

$$commonNodes(S_1, S_2) = \frac{|E_1 \cap E_2|}{|E_1 \cup E_2|} \tag{2}$$

where $E_i$ is the set of nodes in $S_i$. Essentially, the commonNodes metric between two graphs is the Jaccard similarity of the sets of nodes in these graphs. Example 2 shows how the commonNodes metric works.

**Example 2.** Let $S_{knife}$ and $S_{fold}$ be the subgraphs from Example 1, for the nodes *knife* and *fold*, respectively. These two subgraphs have no common node, and 7 distinct nodes in total.

$$commonNodes(S_{knife}, S_{fold}) = \frac{|E_{knife} \cap E_{fold}|}{|E_{knife} \cup E_{fold}|} =$$

$$\frac{|\{Knife, Cut, Pocket, Wallet\} \cap \{Fold, Cooking, Spatula\}|}{|\{Knife, Cut, Pocket, Wallet\} \cup \{Fold, Cooking, Spatula\}|}$$

$$= \frac{0}{7} = 0$$

where $E_{knife}$ is the set of nodes in the $S_{knife}$ subgraph and $E_{fold}$ is the set of nodes in the $S_{fold}$ subgraph.

Recent studies using this method, with minor adjustments, have focused on object identification and on finding the similarity of two nodes in a knowledge graph [15,44,45].

### 4.4. Semantics-based relevance

We first describe the very popular *Wu-Palmer similarity measure* (WUP), which was introduced in [46,47]. Then, we introduce our **Related Pattern Method**, which uses a KG's path pattern to determine whether two nodes are semantically related.

**Wu-Palmer Similarity Measure**: WUP calculates relatedness using WordNet's acyclic graph, which takes into account the depth of two nodes in WordNet taxonomies, as well as the depth of their Least Common Subsumer (LCS). The LCS of two nodes in the WordNet acyclic graph is the most specific common ancestor of these nodes. This metric calculates similarity based on how near nodes in the WordNet acyclic network are to one another. The WUP similarity between an object node ($n_o$) and an action (or state) node ($n_a$) is defined as

$$WUP(n_o, n_a) = 2 * \frac{depth(LCS(n_o, n_a))}{depth(n_o) + depth(n_a)}, \tag{3}$$

where $depth(\cdot)$ is the depth of an entity in the WordNet graph. Moreover, the depth of the LCS is never 0, thus the score can never be zero (the depth of the root of the taxonomy is one).

**Example 3.** The WUP similarity for the object *knife* and the action *fold* is

$$\text{WUP}(knife, fold) = 2 * \frac{depth(LCS(knife, fold))}{depth(knife) + depth(fold)}$$

$$= 2 * \frac{depth(physical\_entity)}{depth(knife) + depth(fold)}$$

$$= 2 * \frac{2}{12+6} = 0.221$$

Many studies use the WUP similarity in a wide spectrum of domains. Recent studies, such as [16,41], use WUP scores to infer object-action relations and object identification.

**Relation Pattern Method**: We next propose a new method, which is based on the idea that some paths connecting two nodes carry semantically more meaningful information than others. The Example 4 presents such a case.

**Example 4.** The path *knife* $\xleftrightarrow{\text{Synonym}}$ *node0* $\xleftrightarrow{\text{ReceivesAction}}$ *cut* may appear more often in the knife-cut object-action pair compared to paths composed of other relations, in the ConceptNet KG. Because the relation *Synonym* may relate the *knife* with a similar object which may receive the same set of actions that *knife* receives.

On the other hand, the path *knife* $\xleftrightarrow{\text{RelatedTo}}$ *node0* $\xleftrightarrow{\text{RelatedTo}}$ *cut* may not contain the same semantically meaningful information, as it may connect through the property *RelatedTo* the *knife* with a completely irrelevant entity in the KG, on which the action *cut* cannot be performed.

Similar examples can be given for ATOMIC, YAGO, WebChild and DBpedia. An important aspect of our proposed method is the flexibility in deciding how to deal with the noise that exists in the data, and the capability to determine the importance of a path through training, rather than through manual annotation.

Notice, that although in the KGs we can find both bi-directional relations (e.g., RelatedTo, Synonym) and uni-directional ones (e.g., UsedFor), we decided not to take into account the directionality of the edges, in order to keep the pattern generation method generic. Of course, more fine-grained patterns can also be considered.

A *relation pattern* is any connecting path that is composed of at least one of the relations that one can consider as important for the problem at hand. We regard the presence of a relation pattern between an object-action or an object-state pair in the KG to be an indication that the two nodes are related. If $\mathcal{P} = \{pattern_1, \ldots, pattern_n\}$ is the set of all relation patterns, the goal is to assign a weight of importance $W_{pattern_i}$ to each $pattern_i \in \mathcal{P}$, indicating how certain we are that the pattern yields proper associations.

For the Relation Pattern method, we consider path patterns based on their frequency of appearances in a set of positive and negative object-action/state relations. For instance, given the relation pattern $\overset{\text{RelatedTo}}{\longleftrightarrow}\ \overset{\text{RelatedTo}}{\longleftrightarrow}$ we would count its appearances in the positive object-action/state ($A$) relations and in the negative object-action/state relations ($B$). Next, we would compute the score $C = A - B$. We would do this for each path pattern that was found at least once in an object-action/state relations, and we would short based on the number $C$ of each path pattern. Here, notice that the procedure is separate for object-action and for object-state. Moreover, we have to comment that we usually would cut at the first 100 path patterns, because after the 100 first the $C$ numbers would start to have negative values.

Next, we describe the process of assigning weights to the relation patterns. For a relation pattern $p$, we characterize the results as True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) according to the following definitions:

- TP is when a pair of object-action/state nodes is both present in the ground truth and connected through $p$.
- FP is when a pair of object-action/state nodes is not present in the ground truth, but connected through $p$.
- TN is when a pair of object-action/state nodes is neither present in the ground truth nor connected through $p$.
- FN is when a pair of object-action/state nodes is present in the ground truth, but not connected through $p$.

Then, we define the weight of importance $W_{pattern_i}$ for $pattern_i$ as the F1-score

$$W_{pattern_i} = 2\frac{P \cdot R}{P + R}, \tag{4}$$

where $P = TP/(TP + FP)$ and $R = TP/(TP + FN)$.

Example 5 shows how the weight of importance $W_{pattern_i}$ works in practice for a relation pattern. The example uses ConceptNet as an underlying KG.

**Example 5.** Consider the relation pattern $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)$ and the set of subgraphs $\{S_{knife}, S_{cut}, S_{stab}, S_{fold}\}$, which represent the nodes *knife*, *cut*, *stab* and *fold*, respectively. For this example, let the knowledge graph $G'$ be composed only from the subgraphs $\{S_{knife}, S_{cut}, S_{stab}, S_{fold}\}$. The *knife* is related with *cut* and *stab*, but not with *fold*, according to the ground truth. For each such pair of object-action nodes, we search for a relation path $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)$ connecting the two nodes (see Section 4.3).

TP is 2 because the pairs knife-cut and knife-stab are related in the ground truth and the relation path $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)$ is a connecting path in both. FP is 1 because the pair knife-fold is not related in the ground truth and the relation path $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)$ is a connecting path. FN is 0 because we do not have a pair that is related in our ground truth and does not have $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)$ as a connecting path. Using this information, we get the following scores.

$$P = \frac{TP}{TP + FP} = \frac{2}{2 + 1} = 0.666 \text{ and}$$
$$R = \frac{TP}{TP + FN} = \frac{2}{2 + 0} = 1$$
$$W_{pattern_{\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)}} = \frac{4}{5} = 0.8$$

The final score of Example 5 shows that the weight of importance $W_{pattern_{\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{UsedFor}}{\longleftrightarrow}\right)}}$ can predict 80% of the positive and negative object action relations. In other words, it shows the proportion of object-action pairs that can be classified correctly (i.e., related or not related), by this relation pattern.

Other heuristics can, of course, be employed instead. Since it is appropriate to examine multiple patterns before deciding on a relationship between two labels, patterns can be grouped together based on their performance, domain-specific importance, or other factors. For example, the weighted sum of the weights of each individual pattern or other heuristics-based metrics can be used to quantify the performance of a cluster $W_C$. As a baseline, we use an even simpler method in our analysis, treating any patterns with weights above a certain threshold as equally important.

For the Relation Pattern method with Clusters based on the procedure that we computed for the Relation Pattern method, we compute all the possible combinations that provide groups of path patterns that are composed of the same type(s) of relations. For example, if we have 5 path patterns that are composed only from the RelatedTo property then that would result in cluster, similarly if we 3 path patterns composed form the relations RelatedTo and UsedFor then this would be another path pattern.

### 4.5. Answering the decision problem

All the methods described in Sections 4.3 and 4.4, except our Relation Pattern Method, result in a relevance score $s_r$. Thus, to answer the decision problem of Section 4.1, we can simply compare this score to a relevance threshold $t$ and yield a positive answer, if $s_r \geq t$, or a negative answer, otherwise. This threshold $t$ can be pre-determined by experts, or learned by using a training set of labeled samples. In this paper, we evaluate those methods assuming that this threshold $t$ is learned.

Our Relation Pattern Method does not yield a relevance score, but instead, it suggests relation patterns that can be exploited to answer the decision problem of Section 4.1. To suggest which patterns, among all possible options, should be exploited, it relies on the weight computation of Eq. (4). Specifically, it computes the average pattern weight $w_{avg}$ from a training set of labeled samples, and suggests those patterns that yield a weight $w \geq w_{avg}$.

## 5. Evaluation

This section describes how we constructed the ground truth from the Something-Something Dataset, followed by an explanation of the experimental setup and the findings obtained by the evaluation of each method.

### 5.1. Data collection

We decided to extract the set of labels for our evaluation from the Something-Something Dataset rather than using a random collection of action, state and object labels in order to get appropriate coverage of entities for the household domain. Something-Something is a big collection of short video clips (about 108k) depicting actions done on and with everyday objects. The activities involve either one type of object (for example, opening a bottle) or two different types of objects (for example, putting coins inside a box). The Something-Something Dataset has become a de-facto benchmark for the evaluation of systems addressing the task of action recognition due to its large number of sample videos. The dataset includes a brief description for each clip that includes action, state, and object(s) labels.

**Ground Truth Creation:** We initially extracted all object-action-state labels. All plural object labels were changed to their singular counterparts, for example *notes* was replaced with *note*. Then, we removed certain object and action labels that were not relevant to the context, as they were not household objects, and ended up with 148 object labels, 25 action labels and 24 state labels. Notice that these labels, apart from being relevant, were not randomly selected. We counted the number of appearances of the object-action and object-state associations that each pair produces in the video descriptions

of Something-Something. Therefore, the 148*25 object-action relations appear in 39 514 different video descriptions, and the 148*24 object-state relations appear in 39 019 different video descriptions. Out of the total 108 499 video descriptions that Something-Something has, we considered these portions a normal coverage of object-action and object-state relations, over (maybe) the largest dataset when it comes to object-action and object-state relations.

Next, for the remaining action, object and state labels, we issued a query to each KG that we evaluated, in order to identify which labels are indeed part of the graph. For the ConceptNet KG, we used the ConceptNet Web API,[7] for DBpedia, we used the Virtuoso SPARQL endpoint[8] (the SPARQL query can be found in *Listing 1*), while for ATOMIC, WebChild and YAGO, we developed our own Python script to search for each label. All the KGs contained 148 object labels, 25 action labels, and 24 state labels, except ATOMIC and DBpedia which contained 24 action labels. Finally, since some actions have the same label with some objects (3 in total), we renamed these labels as follows: (a) pile → *pileO* and pile → *pileV*, (b) stack → *stackO* and stack → *stackV*, and (c) cover → *coverO* and cover → *coverV*, to refer to the object and action label, respectively.

Eventually, the object-action and object-state pairs that existed in the description of at least one video in the Something-Something Dataset were automatically characterized as positive pairs. The remaining were manually annotated, in order to determine if they are negative or if they are positive.

***Listing 1***: SPARQL query for DBpedia

```
PREFIX dbpr: <http://dbpedia.org/resource/>
PREFIX dbpo: <http://dbpedia.org/ontology/>
PREFIX owl: <http://www.w3.org/2002/07/>
SELECT ?property1 ?entity1 ?property2 ?
    entity2
WHERE {
{
dbpr:<Input> ?property1 ?entity1.
?entity1 rdf:type owl:Thing
OPTIONAL {
?entity1 ?property2 ?entity2.
?entity2 rdf:type owl:Thing. }
FILTER (?property1 != dbpo:
    wikiPageRedirects
&& ?property1 != dbpo:wikiPageWikiLink
&& ?property2 != dbpo:wikiPageRedirects
&& ?property2 != dbpo:wikiPageWikiLink)
}
UNION
{
?entity1 ?property1 dbpr:<Input>.
?entity1 rdf:type owl:Thing.
OPTIONAL {?entity2 ?property2 ?entity1.
?entity2 rdf:type owl:Thing.}
FILTER (?property1 != dbpo:
    wikiPageRedirects
&& ?property1 != dbpo:wikiPageWikiLink
&& ?property2 != dbpo:wikiPageRedirects
&& ?property2 != dbpo:wikiPageWikiLink)
}
}
```

[7] https://pypi.org/project/ConceptNet/
[8] https://dbpedia.org/sparql

### 5.2. Experimental setup

The methods described in Section 4 were evaluated for each KG using 10-fold cross validation over the total number of positive and negative relations captured by each KG, as detailed in Section 5.1. We used Sklearn[9] to split our data into 10 folds.

To train the multiple models, each iteration of the 10-fold cross-validation method was used. The training folds specifically helped to identify the appropriate threshold for each method that optimizes the F1 score (see Example 6) for the *Connecting Path Method*, the *WUP similarity* and the *Common Node Method*. The training phase of the Relation Pattern Method assisted in calculating the weights of importance $W_{pattern_i}$ of each relation pattern $pattern_i \in \mathcal{P}$, as explained in Section 4.4. During testing, we evaluated each method's performance using the given thresholds and weights. Patterns that did not perform well throughout training were completely removed.

**Example 6.** Consider the thresholds $thr_{cp} = 0.6$, $thr_{cn} = 0.5$ and $thr_{wup} = 0.47$ for the Connecting Path, Common Node and WUP similarity, respectively, the negative object-action relation *knife-fold* and the corresponding subgraphs $S_{knife}$ and $S_{fold}$. The Connecting Path score for this pair is 0.66 (see Example 1), it is therefore classified as a False Positive pair (if the score was below 0.6 then it would be classified as a True Negative). Similarly, the Common Node and WUP similarity scores are, 0 and 0.22, respectively, (see (2) and Example 3), therefore they classify the pair as a True Negative. Analogously, given the positive object-action relation *knife-cut* and the corresponding subgraphs $S_{knife}$ and $S_{cut}$, and for the sake of the example let the Connecting Path score for this pair be 0.56, while for the Common Node and the WUP similarity the scores be 0.67 and 0.71, respectively. Then, the Connecting Path metric classifies the knife-cut pair as a False Negative (if the score was above the threshold it would classify them as a True Positive). But, the Common Node and WUP similarity metrics classify the pair as a True Positive.

Another variant of this method would have been to limit the anticipated results to those with a confidence score greater than a certain threshold, i.e., restrict the anticipated results only to these that are above/below a threshold. However, we found that this method works best when the minimal confidence criterion is 0 (confidence ratings are extremely low in far too many cases), thus we decided to not report results for this variation.

Of course, when such similar paths are viewed as a group rather than as individuals, they gain practical importance. As a result, for each knowledge graph, we additionally present the performance of two or three (if there exist as many) clusters of patterns. We take a straightforward technique to determine what a cluster is: any pattern that its weight exceeds a certain threshold is considered relevant. As a result, even if a single pattern is detected in the graph, the object-action or object-state pair associated with it, is considered related. We chose a relatively broad threshold for including patterns in the cluster, namely any pattern with a weight greater than 0.1, because we simply want to evaluate a baseline scenario.

More complex methods can be used, such as taking into account the weight of importance among the patterns in each cluster or using domain-specific criteria. Even with this as a starting point, we can see that grouping pathways can enhance F1-scores. However, by neglecting the relative relevance of each individual pattern, we introduce noise, as evidenced by the precision scores when compared to the highest performing patterns, an issue that may be addressed with a more advanced approach.

**KG Embedding-based Baseline.** Recent works in the field of KG embeddings for the task of link prediction [48] represent nodes of

[9] https://scikit-learn.org/stable/modules/cross_validation.html

**Table 1**
ConceptNet knowledge graph and object-action relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.534 | 0.752 | 0.552 | 0.637 |
| WUP | 0.555 | 0.951 | 0.551 | 0.698 |
| Common Node | 0.551 | 0.956 | 0.548 | 0.697 |
| AllenAI-Commonsense (top-1) | 0.502 | 0.191 | 0.596 | 0.289 |
| AllenAI-Commonsense (top-3) | 0.582 | 0.599 | **0.608** | 0.603 |
| AllenAI-Commonsense (top-5) | **0.596** | 0.748 | 0.595 | 0.663 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.551 | 0.964 | 0.548 | 0.699 |
| RelatedTo RelatedTo | 0.539 | **0.985** | 0.536 | 0.695 |
| RelatedTo Synonym | 0.558 | 0.906 | 0.557 | 0.69 |
| RelatedTo RelatedTo RelatedTo | 0.561 | 0.891 | 0.56 | 0.688 |
| RelatedTo RelatedTo RelatedTo Synonym | 0.561 | 0.835 | 0.564 | 0.673 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo-Synonym | 0.539 | **0.985** | 0.546 | **0.703** |
| UsedFor-Synonym | 0.582 | 0.636 | 0.569 | 0.601 |

a KG as vectors in a low-dimensional space, which are generated by considering both textual (e.g., through word/sentence embeddings) and structural (e.g., through graph traversals) features of those nodes.

As an indicative method for this family of algorithms, in this work we employ *AllenAI-CommonSense* [49] as a baseline, which constitutes the state of the art for link prediction in ConceptNet. This method employs a pre-trained BERT [50] model that is fine-tuned on ConceptNet, using Graph Convolutional Networks (GCN) [51] for embedding the ConceptNet graph. This model returns a list of possible relations between a given pair of ConceptNet nodes, ranked in descending order of likelihood. We consider that the answer to the object-action problem formulated in Section 4.1 is positive for two query nodes, when the relation "ReceivesAction" is within the top-$k$ answers for those query nodes (for $k \in \{1, 3, 5\}$).

### 5.3. Results

Next, we summarize the overall performance measurements for each method, over the knowledge graphs of ConceptNet, YAGO, WebChild, ATOMIC and DBpedia. Moreover, we compare the performance of ConceptNet using only the object, action and state labels which exist in YAGO, WebChild, ATOMIC and DBpedia, respectively.

In the following tables, we display the accuracy, precision, recall and F1-score for the Connecting Path, Common Node and WUP similarity metrics, as well as for the Relation Pattern Method and Relation Pattern Method with clusters. Notice that due to the plurality of relation patterns we display only the Top-5 relation patterns with respect to F1-score performance (i.e., the Top-5 relation patterns that achieved the best F1-score). Analogously, we display only the Top-3 clusters of relation patterns with respect to F1-score, if there exist as many (see Tables 5–20).

In general, the differences among the Connecting Path, Common Node and WUP similarity metrics, with respect to their F1-score are small for each KG, when evaluating object-action relations. The same does not hold when evaluating object-state relations, as the Connecting Path method achieves worse scores with respect to F1-score than the Common Node and WUP metrics. On the other hand, in almost all cases either the Relation Pattern Method or the Relation Pattern Method with clusters outperform the three aforementioned methods in regard to F1-score. Moreover, when evaluating the DBpedia KG, the nodes in DBpedia (i.e., the URIs) are not single word labels, but instead they are small descriptions or phrases. In this case, we can see that the WUP similarity metric achieves its worst performance. The reason for that is because the WUP similarity metric needs single word labels in order to provide a similarity measure, instead of small phrases. We elaborate more on our results in Section 6.

One may notice that we evaluate ConceptNet two times. Tables 1 and 3 for object-action relations, and Tables 2 and 4 for object-state relations. The reason was that, although not explicitly stated in the ConceptNet documentation, *RelatedTo* plays the role of a super-property, i.e., it subsumes the other relations. Thus, we wanted to see the performance of ConceptNet with and without the RelatedTo property.

The AllenAI-CommonSense (top-$k$) methods (Table 1), despite their high accuracy, underperform in F1 scores, compared to the other methods. This is due to a considerable difference noticed in the accuracy for positive pairs (.19) with respect to that for negative pairs (.854).

As mentioned, not all graphs have the same number of object, action and state labels. For this reason, we did a second round of experiments where we used the labels that each KG has and evaluated the ConceptNet KG only on the object-action and object-state relations formed from these labels. The reason for this was that ConceptNet is our baseline KG, and we wanted to see how ConceptNet performs on the same batch of labels that each KG has in order to compare performances.

Notice that WebChild and YAGO have the same number of object and action labels with ConceptNet. Therefore, when we evaluate ConceptNet with the labels existing in WebChild and YAGO, the results will be the same as if we evaluated ConceptNet with its own labels. The same holds for each KG when considering object state relations. Either way, we display a table for these cases even though the results are the same with ConceptNet when using its own labels.

Also, notice that the subgraphs of WebChild and YAGO which are described in Section 4 have depth 1, and the subgraphs of ConceptNet, ATOMIC and DBpedia have depth 2. The reason for that was two-fold: (a) we wanted the subgraphs of each KG to have almost the same number of nodes, approximately 1000 (as 1000 nodes gave us adequate information) and (b) the subgraphs of WebChild and YAGO contained approximately 1000 nodes having only depth 1 paths.

Finally, it is true that the relation patterns which achieved high scores in YAGO are quite obscure, in how they could help in inferring object-action and object-state association relations. For this reason, we present 3 examples for each case. Example 7 shows 3 examples of object-action association relations, and Example 8 shows 3 examples of object-state association relations.

**Example 7.** Object-action association relations for the relation patterns which achieved high F1-scores in YAGO.

$$lift \xleftrightarrow{\text{inlanguage}} english\_language, french\_language, korean\_language, japanese\_language \xleftrightarrow{\text{inlanguage}} bucket$$

**Table 2**
ConceptNet knowledge graph and object-state relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.526 | 0.384 | 0.453 | 0.415 |
| WUP | 0.536 | 0.856 | 0.588 | 0.697 |
| Common Node | 0.556 | 0.892 | 0.577 | 0.701 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo IsA | **0.654** | 0.699 | **0.724** | 0.711 |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.63 | 0.693 | 0.715 | 0.704 |
| RelatedTo AtLocation RelatedTo RelatedTo | 0.62 | 0.69 | 0.709 | 0.699 |
| RelatedTo RelatedTo IsA RelatedTo | 0.615 | 0.674 | 0.69 | 0.681 |
| DerivedFrom HasContext RelatedTo HasContext | 0.645 | 0.663 | 0.674 | 0.668 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo | 0.573 | **0.938** | 0.582 | **0.718** |
| RelatedTo-Synonym | 0.567 | **0.938** | 0.568 | 0.708 |
| RelatedTo-AtLocation | 0.557 | 0.865 | 0.565 | 0.684 |

**Table 3**
ConceptNet knowledge graph and object-action relations from something-something (without RelatedTo).

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.554 | 0.879 | 0.552 | 0.678 |
| WUP | 0.549 | **0.942** | 0.539 | **0.685** |
| Common Node | 0.543 | 0.905 | 0.535 | 0.672 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| HasContext HasContext AtLocation AtLocation | 0.532 | 0.398 | 0.58 | 0.472 |
| IsA IsA | 0.521 | 0.303 | 0.565 | 0.394 |
| HasContext HasContext HasContext HasContext | 0.503 | 0.243 | 0.558 | 0.339 |
| DerivedFrom HasContext AtLocation HasContext | 0.52 | 0.245 | 0.515 | 0.332 |
| DerivedFrom DerivedFrom AtLocation AtLocation | 0.518 | 0.233 | 0.482 | 0.314 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| HasContext-AtLocation-DerivedFrom | **0.588** | 0.791 | 0.577 | 0.667 |
| IsA-HasContext-Antonym | 0.57 | 0.672 | 0.575 | 0.62 |
| IsA-HasContext-Antonym | 0.564 | 0.38 | **0.595** | 0.464 |

**Table 4**
ConceptNet knowledge graph and object-state relations from something-something (without RelatedTo).

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.524 | 0.324 | 0.347 | 0.335 |
| WUP | 0.478 | 0.519 | 0.426 | 0.467 |
| Common Node | 0.542 | 0.564 | 0.519 | 0.540 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| HasProperty ReceivesAction AtLocation ReceivesAction | 0.478 | 0.413 | 0.598 | 0.489 |
| Antonym HasProperty AtLocation AtLocation | 0.481 | 0.506 | 0.467 | 0.486 |
| DistinctFrom HasProperty AtLocation AtLocation | 0.481 | 0.506 | 0.467 | 0.486 |
| HasProperty AtLocation AtLocation | 0.465 | 0.513 | 0.391 | 0.444 |
| HasProperty IsA AtLocation AtLocation | 0.479 | 0.505 | 0.38 | 0.434 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| AtLocation-IsA-HasContext | **0.643** | **0.672** | 0.716 | **0.693** |
| AtLocation-UsedFor-ReceivesAction | 0.64 | 0.49 | **0.756** | 0.595 |
| AtLocation-HasProperty | 0.555 | 0.467 | 0.714 | 0.565 |

**Table 5**
Atomic knowledge graph and object-action relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.558 | 0.923 | 0.557 | 0.695 |
| WUP | 0.523 | 0.992 | 0.521 | 0.683 |
| Common Node | 0.519 | 0.987 | 0.52 | 0.681 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| prefix prefix prefix prefix | 0.568 | 0.923 | **0.587** | 0.718 |
| xAttr prefix | **0.588** | 0.943 | 0.579 | 0.717 |
| prefix prefix | 0.558 | 0.91 | 0.577 | 0.706 |
| xAttr xAttr | 0.528 | 0.899 | 0.523 | 0.661 |
| xAttr prefix xAttr prefix | 0.519 | 0.889 | 0.513 | 0.653 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| xAttr-prefix | 0.531 | **0.995** | 0.586 | **0.738** |
| prefix | 0.536 | 0.966 | 0.549 | 0.701 |
| xAttr | 0.531 | 0.938 | 0.528 | 0.676 |

**Table 6**
Atomic knowledge graph and object-state relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.45 | 0.461 | 0.6 | 0.521 |
| WUP | 0.526 | 0.863 | 0.527 | 0.654 |
| Common Node | 0.546 | 0.891 | 0.55 | 0.68 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| xAttr prefix xAttr prefix | 0.58 | **0.92** | **0.74** | **0.821** |
| prefix prefix xAttr prefix | 0.58 | **0.92** | **0.74** | **0.821** |
| prefix xAttr xAttr | 0.56 | 0.89 | 0.69 | 0.78 |
| prefix xAttr prefix xAttr | 0.56 | 0.85 | 0.67 | 0.75 |
| prefix xAttr xAttr xAttr | 0.56 | 0.84 | 0.65 | 0.732 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| xAttr-prefix | 0.6 | 0.6 | 0.6 | 0.6 |
| prefix | **0.615** | 0.592 | 0.6 | 0.596 |
| xAttr | 0.53 | 0.584 | 0.6 | 0.592 |

**Table 7**
ConceptNet knowledge graph and object-action relations only existing in ATOMIC.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.561 | 0.938 | 0.563 | 0.704 |
| WUP | 0.522 | 0.934 | 0.524 | 0.671 |
| Common Node | 0.528 | 0.941 | 0.53 | 0.678 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo RelatedTo | **0.681** | 0.71 | 0.685 | 0.697 |
| RelatedTo RelatedTo | **0.681** | 0.695 | 0.685 | 0.69 |
| RelatedTo RelatedTo RelatedTo | 0.653 | 0.669 | **0.687** | 0.678 |
| RelatedTo Synonym RelatedTo RelatedTo | 0.667 | 0.702 | 0.588 | 0.64 |
| Synonym RelatedTo RelatedTo RelatedTo | 0.624 | 0.711 | 0.521 | 0.601 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo-Synonym | 0.545 | 0.955 | 0.571 | **0.715** |
| UsedFor-Synonym | 0.531 | **0.991** | 0.527 | 0.688 |

**Table 8**
ConceptNet knowledge graph and object-state relations only existing in ATOMIC.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.526 | 0.384 | 0.453 | 0.415 |
| WUP | 0.536 | 0.856 | 0.588 | 0.697 |
| Common Node | 0.556 | 0.892 | 0.577 | 0.701 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo IsA | **0.654** | 0.699 | **0.724** | 0.711 |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.63 | 0.693 | 0.715 | 0.704 |
| RelatedTo AtLocation RelatedTo RelatedTo | 0.62 | 0.69 | 0.709 | 0.699 |
| RelatedTo RelatedTo IsA RelatedTo | 0.615 | 0.674 | 0.69 | 0.681 |
| DerivedFrom HasContext RelatedTo HasContext | 0.645 | 0.663 | 0.674 | 0.668 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo | 0.573 | **0.938** | 0.582 | **0.718** |
| RelatedTo-Synonym | 0.567 | **0.938** | 0.568 | 0.708 |
| RelatedTo-AtLocation | 0.557 | 0.865 | 0.565 | 0.684 |

**Table 9**
YAGO knowledge graph and object-action relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.574 | 0.901 | 0.562 | 0.692 |
| WUP | 0.534 | **0.911** | 0.539 | 0.677 |
| Common Node | 0.54 | 0.936 | 0.541 | 0.685 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| inlanguage inlanguage | **0.66** | 0.708 | **0.689** | 0.698 |
| knowslanguage inlanguage | 0.656 | 0.684 | 0.679 | 0.681 |
| inlanguage knowslanguage | 0.656 | 0.654 | 0.659 | 0.656 |
| genre genre | 0.619 | 0.627 | 0.623 | 0.625 |
| hasoccupation hasoccupation | 0.577 | 0.575 | 0.58 | 0.577 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| inlanguage-knowslanguage | 0.534 | 0.896 | 0.636 | **0.744** |
| sport-about | 0.518 | 0.601 | 0.327 | 0.424 |
| familyname-givenname-parenttaxon | 0.536 | 0.72 | 0.218 | 0.335 |

**Table 10**
YAGO knowledge graph and object-state relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.48 | 0.334 | 0.432 | 0.377 |
| WUP | 0.529 | 0.752 | **0.556** | 0.639 |
| Common Node | 0.532 | 0.773 | 0.554 | 0.645 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| inlanguage inlanguage | 0.518 | 0.665 | 0.548 | 0.601 |
| genre genre | 0.532 | 0.594 | 0.55 | 0.571 |
| inlanguage knowslanguage | 0.51 | 0.574 | 0.548 | 0.561 |
| knowslanguage inlanguage | 0.51 | 0.574 | 0.548 | 0.561 |
| taxonrank taxonrank | 0.485 | 0.238 | 0.388 | 0.295 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| inlanguage-knowslanguage | 0.548 | **0.9** | 0.541 | **0.677** |
| genre-alumniof-memberof | **0.566** | 0.745 | 0.545 | 0.63 |
| birthplace-deathplace-homelocation | 0.471 | 0.7 | 0.45 | 0.55 |

**Table 11**
ConceptNet knowledge graph and object-action relations only existing in YAGO.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.534 | 0.752 | 0.552 | 0.636 |
| WUP | 0.555 | 0.951 | 0.551 | 0.697 |
| Common Node | 0.551 | 0.956 | 0.548 | 0.696 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.551 | 0.964 | 0.548 | 0.699 |
| RelatedTo RelatedTo | 0.539 | **0.985** | 0.536 | 0.694 |
| RelatedTo Synonym | 0.558 | 0.906 | 0.557 | 0.69 |
| RelatedTo RelatedTo RelatedTo | 0.561 | 0.891 | 0.56 | 0.688 |
| RelatedTo RelatedTo RelatedTo Synonym | 0.561 | 0.835 | 0.564 | 0.673 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo-Synonym | 0.539 | **0.985** | 0.546 | **0.703** |
| UsedFor-Synonym | **0.582** | 0.636 | **0.569** | 0.601 |

**Table 12**
ConceptNet knowledge graph and object-state relations only existing in YAGO.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.526 | 0.384 | 0.453 | 0.415 |
| WUP | 0.536 | 0.856 | 0.588 | 0.697 |
| Common Node | 0.556 | 0.892 | 0.577 | 0.701 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo IsA | **0.654** | 0.699 | **0.724** | 0.711 |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.63 | 0.693 | 0.715 | 0.704 |
| RelatedTo AtLocation RelatedTo RelatedTo | 0.62 | 0.69 | 0.709 | 0.699 |
| RelatedTo RelatedTo IsA RelatedTo | 0.615 | 0.674 | 0.69 | 0.681 |
| DerivedFrom HasContext RelatedTo HasContext | 0.645 | 0.663 | 0.674 | 0.668 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo | 0.573 | **0.938** | 0.582 | **0.718** |
| RelatedTo-Synonym | 0.567 | **0.938** | 0.568 | 0.708 |
| RelatedTo-AtLocation | 0.557 | 0.865 | 0.565 | 0.684 |

**Table 13**
WebChild knowledge graph and object-action relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.437 | 0.777 | 0.434 | 0.557 |
| WUP | 0.435 | 0.841 | 0.458 | 0.593 |
| Common Node | 0.434 | 0.84 | 0.457 | 0.591 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| quality quality | 0.458 | 0.911 | 0.458 | 0.61 |
| haspart haspart | 0.463 | 0.88 | 0.46 | 0.604 |
| size size | 0.461 | 0.807 | 0.455 | 0.582 |
| state state | 0.45 | 0.785 | 0.449 | 0.571 |
| age age | **0.47** | 0.668 | **0.468** | 0.55 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| quality-state | 0.458 | **0.912** | 0.459 | **0.611** |
| haspart-size | 0.458 | 0.895 | 0.458 | 0.606 |
| state-weight-motion | 0.449 | 0.862 | 0.452 | 0.593 |

**Table 14**
WebChild knowledge graph and object-state relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.462 | 0.34 | 0.292 | 0.314 |
| WUP | 0.437 | 0.568 | 0.414 | 0.479 |
| Common Node | 0.429 | 0.578 | 0.404 | 0.475 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| haspart haspart | 0.451 | 0.575 | 0.324 | 0.414 |
| state state | 0.467 | 0.388 | 0.424 | 0.405 |
| quality quality | 0.459 | 0.434 | 0.292 | 0.349 |
| haspart size | **0.488** | 0.366 | 0.356 | 0.361 |
| size haspart | 0.474 | 0.38 | 0.316 | 0.345 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| haspart-quality-state | 0.459 | **0.874** | **0.664** | **0.755** |
| size-weight-age | 0.487 | 0.742 | 0.458 | 0.567 |

**Table 15**
ConceptNet knowledge graph and object-action relations only existing in WebChild.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.534 | 0.752 | 0.552 | 0.636 |
| WUP | 0.555 | 0.951 | 0.551 | 0.697 |
| Common Node | 0.551 | 0.956 | 0.548 | 0.696 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.551 | 0.964 | 0.548 | 0.699 |
| RelatedTo RelatedTo | 0.539 | **0.985** | 0.536 | 0.694 |
| RelatedTo Synonym | 0.558 | 0.906 | 0.557 | 0.69 |
| RelatedTo RelatedTo RelatedTo | 0.561 | 0.891 | 0.56 | 0.688 |
| RelatedTo RelatedTo RelatedTo Synonym | 0.561 | 0.835 | 0.564 | 0.673 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo-Synonym | 0.539 | **0.985** | 0.546 | **0.703** |
| UsedFor-Synonym | **0.582** | 0.636 | **0.569** | 0.601 |

**Table 16**
ConceptNet knowledge graph and object-state relations only existing in WebChild.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.526 | 0.384 | 0.453 | 0.415 |
| WUP | 0.536 | 0.856 | 0.588 | 0.697 |
| Common Node | 0.556 | 0.892 | 0.577 | 0.701 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo RelatedTo RelatedTo IsA | **0.654** | 0.699 | **0.724** | 0.711 |
| RelatedTo RelatedTo RelatedTo RelatedTo | 0.63 | 0.693 | 0.715 | 0.704 |
| RelatedTo AtLocation RelatedTo RelatedTo | 0.62 | 0.69 | 0.709 | 0.699 |
| RelatedTo RelatedTo IsA RelatedTo | 0.615 | 0.674 | 0.69 | 0.681 |
| DerivedFrom HasContext RelatedTo HasContext | 0.645 | 0.663 | 0.674 | 0.668 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo | 0.573 | **0.938** | 0.582 | **0.718** |
| RelatedTo-Synonym | 0.567 | **0.938** | 0.568 | 0.708 |
| RelatedTo-AtLocation | 0.557 | 0.865 | 0.565 | 0.684 |

**Table 17**
DBpedia knowledge graph and object-action relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.553 | **0.727** | 0.45 | 0.556 |
| WUP | 0.492 | 0.252 | 0.534 | 0.342 |
| Common Node | **0.496** | 0.258 | 0.508 | 0.342 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| wikiPageDisambiguates ⟷ genre ⟷ wikiPageDisambiguates ⟷ genre ⟷ | 0.494 | 0.441 | **0.63** | 0.519 |
| wikiPageDisambiguates ⟷ hypernym ⟷ wikiPageDisambiguates ⟷ hypernym ⟶ | 0.489 | 0.495 | 0.508 | 0.501 |
| wikiPageDisambiguates ⟷ hypernym ⟷ wikiPageDisambiguates ⟷ type ⟷ | 0.486 | 0.425 | 0.532 | 0.473 |
| wikiPageDisambiguates ⟷ type ⟷ wikiPageDisambiguates ⟷ hypernym ⟶ | 0.484 | 0.43 | 0.468 | 0.448 |
| wikiPageDisambiguates ⟷ occupation ⟷ wikiPageDisambiguates ⟷ hypernym ⟶ | 0.487 | 0.406 | 0.4 | 0.403 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| wikiPageDisambiguates-other | **0.496** | 0.618 | 0.548 | **0.581** |

**Table 18**
DBpedia knowledge graph and object-state relations from something-something.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | **0.538** | 0.388 | 0.314 | 0.347 |
| WUP | 0.495 | 0.264 | 0.489 | 0.343 |
| Common Node | 0.499 | 0.182 | 0.508 | 0.268 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| wikiPageDisambiguates ⟷ hypernym ⟷ wikiPageDisambiguates ⟷ hypernym ⟶ | 0.49 | **0.442** | 0.533 | 0.483 |
| wikiPageDisambiguates ⟷ hypernym ⟷ wikiPageDisambiguates ⟷ type ⟷ | 0.472 | 0.439 | 0.536 | 0.482 |
| wikiPageDisambiguates ⟷ type ⟷ wikiPageDisambiguates ⟷ hypernym ⟶ | 0.489 | 0.436 | 0.327 | 0.374 |
| wikiPageDisambiguates ⟷ language ⟷ wikiPageDisambiguates ⟷ language ⟷ | 0.487 | 0.425 | 0.275 | 0.334 |
| wikiPageDisambiguates ⟷ type ⟷ wikiPageDisambiguates ⟷ type ⟷ | 0.478 | 0.413 | 0.233 | 0.298 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| wikiPageDisambiguates-other | 0.489 | 0.423 | **0.599** | **0.495** |

**Table 19**
ConceptNet knowledge graph and object-action relations only existing in DBpedia.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.534 | 0.752 | 0.552 | 0.636 |
| WUP | 0.555 | 0.951 | 0.551 | 0.697 |
| Common Node | 0.551 | 0.956 | 0.548 | 0.696 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟶ | 0.551 | 0.964 | 0.548 | 0.699 |
| RelatedTo ⟷ RelatedTo ⟶ | 0.539 | **0.985** | 0.536 | 0.694 |
| RelatedTo ⟷ Synonym ⟶ | 0.558 | 0.906 | 0.557 | 0.69 |
| RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟶ | 0.561 | 0.891 | 0.56 | 0.688 |
| RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟷ Synonym ⟶ | 0.561 | 0.835 | 0.564 | 0.673 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo-Synonym | 0.539 | **0.985** | 0.546 | **0.703** |
| UsedFor-Synonym | **0.582** | 0.636 | **0.569** | 0.601 |

**Table 20**
ConceptNet knowledge graph and object-state relations only existing in DBpedia.

| Method | Accuracy | Recall | Precision | F1 score |
|---|---|---|---|---|
| Connecting Path | 0.526 | 0.384 | 0.453 | 0.415 |
| WUP | 0.536 | 0.856 | 0.588 | 0.697 |
| Common Node | 0.556 | 0.892 | 0.577 | 0.701 |
| Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_{pattern}$) |
| RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟷ IsA ⟷ | **0.654** | 0.699 | **0.724** | 0.711 |
| RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟷ RelatedTo ⟶ | 0.63 | 0.693 | 0.715 | 0.704 |
| RelatedTo ⟷ AtLocation ⟷ RelatedTo ⟷ RelatedTo ⟶ | 0.62 | 0.69 | 0.709 | 0.699 |
| RelatedTo ⟷ RelatedTo ⟷ IsA ⟷ RelatedTo ⟶ | 0.615 | 0.674 | 0.69 | 0.681 |
| DerivedFrom ⟷ HasContext ⟷ RelatedTo ⟷ HasContext ⟶ | 0.645 | 0.663 | 0.674 | 0.668 |
| Cluster Relation Pattern | Accuracy | Recall | Precision | F1 score ($W_C$) |
| RelatedTo | 0.573 | **0.938** | 0.582 | **0.718** |
| RelatedTo-Synonym | 0.567 | **0.938** | 0.568 | 0.708 |
| RelatedTo-AtLocation | 0.557 | 0.865 | 0.565 | 0.684 |

**Table 21**
The difference in performance for the relation pattern without clusters over the KGs.

| KG | Object-Action | Object-State | Subset Object-Action | Subset Object-State |
|---|---|---|---|---|
| ConceptNet | 0.001 | 0.001 | – | – |
| ConceptNet (no RelatedTo) | −0.213 | −0.051 | – | – |
| ATOMIC | **0.011** | **0.141** | **0.03** | **0.01** |
| YAGO | 0.006 | 0.032 | 0.002 | **0.01** |
| WebChild | 0.008 | 0.017 | 0.002 | **0.01** |
| DBpedia | −0.037 | 0.136 | 0.001 | **0.01** |

$$spill \xleftrightarrow{\text{knowslanguage}} english\_language,$$
$$french\_language, irish\_language, german\_language,$$
$$norwegian\_language \xleftrightarrow{\text{inlanguage}} perfume$$

$$close \xleftrightarrow{\text{inlanguage}} english\_language,$$
$$french\_language, irish\_language, german\_language,$$
$$norwegian\_language \xleftrightarrow{\text{knowslanguage}} cap$$

**Example 8.** Object-state association relations for the relation patterns which achieved high F1-scores in YAGO.

$$lift \xleftrightarrow{\text{inlanguage}} english\_language,$$
$$american\_language \xleftrightarrow{\text{inlanguage}} money$$

$$lift \xleftrightarrow{\text{genre}} country\_music,$$
$$dance\_music, rock\_music, punk\_music,$$
$$spoken\_word, satire$$
$$\xleftrightarrow{\text{genre}} coat$$

$$lift \xleftrightarrow{\text{inlanguage}} english\_language,$$
$$american\_language \xleftrightarrow{\text{knowslanguage}} speaker$$

## 6. Discussion

We start this section with a discussion over the results by comparing how the exploitation (and non exploitation) of the semantics of a KG helps in inferring the answer to a specific set of commonsense questions. Table 21 lists the cases that the simple Relation Pattern Method (without clusters) outperforms (w.r.t. F1 score) the other three methods in identifying object-action (column 1) and object-state (column 2) relations. The Table shows the difference in F1-performance in each case, highlighting the cases where this is the largest (see Eq. (5)).

$$\begin{aligned} result = {} & f1\_score\_relation\_pattern\_method \\ & - max\{f1\_score\_common\_node, \\ & f1\_score\_common\_path, f1\_score\_wup\} \end{aligned} \tag{5}$$

Table 22 presents the same results for the Relation Pattern with Clusters method (see Eq. (6)). We also provide the best performance when we use ConceptNet as an underlying KG, but with the subset of labels that exist in each of ATOMIC, YAGO, WebChild and DBpedia, respectively, 3rd column for object-action relations and 4th column for object-state relations.

$$\begin{aligned} result = {} & f1\_score\_relation\_pattern\_method\_with\_clusters \\ & - max\{f1\_score\_common\_node, \\ & f1\_score\_common\_path, f1\_score\_wup\} \end{aligned} \tag{6}$$

The negative scores in Tables 21 and 22 imply that our semantics-based method did not achieve better F1 score than any one of the compared methods. As one can notice our Relation Pattern Method without Clusters has performed better when using the ATOMIC KG. The reason for that is because ATOMIC has fewer types of relations (only 3 relation types in the subgraphs that we created) than the other KGs. Therefore, it is expected that there will not exist too many different relation patterns, and those that exist will appear more frequently in object-action and object-state pairs.

On the other hand, one can see that the Relation Pattern Method with Clusters achieves the best score over YAGO when we evaluate object-action association, and the best score over WebChild when we evaluate object-state associations. This is also expected as YAGO and WebChild have the biggest number of different relation types (both close to 1200 relations) which leads to many relation patterns that appear less oftenly in object-action or object-state relations, and that is what a cluster of relation patterns needs in order to achieve big F1 scores. In other words, a cluster with relation patterns needs every relation pattern that exists in it to appear in an adequate number of object-action, or object-state pairs, instead of having just some relation patterns to appear in almost all object-action or object-state pairs.

A more detailed analysis on the results reveals that the exploitation of the semantics in a KG, can show when a relation in a KG can be considered as super property of other relations. In more detail, if we see a specific relation appearing in almost all relation patterns that achieve the biggest scores, with respect to F1, we can confidently conclude that this property connects too much information, which may lead to noise in the KG. This conclusion is quite important because an appropriate usage of the semantics in a KG can show insights on when a refinement of the properties is needed. This conclusion becomes quite clear with the *RelatedTo* property of ConceptNet. Almost all relation patterns contain the *RelatedTo*. This is because, despite not being stated directly in the ConceptNet specification, *RelatedTo* serves as a super-property, i.e. it encompasses all other relations. While it might seem that less abstract node-to-node relationships, such as *UsedFor*, would yield better results, this is not the case. The main reason is that the Relation Pattern method is based on the frequency that a property appears in paths that connect a set of object-action/state pairs. Therefore, some properties such as UsedFor which we would expect (based on our commonsense), to achieve bigger scores did not, because it was not so common, regarding the frequency of appearance in connecting paths. The reason for properties like this, i.e., these that based on our commonsense we would expect to appear more in connecting paths, not achieving bigger scores can be many. Basically, our understanding was that there was a preference on using more general properties such as RelatedTo. Our method can help to tackle this fact by pointing for which properties there might be a need to define sub properties.

We also observe that certain longer paths, such as $\left(\xleftrightarrow{\text{RelatedTo}}, \xleftrightarrow{\text{RelatedTo}}, \xleftrightarrow{\text{RelatedTo}}, \xleftrightarrow{\text{RelatedTo}}\right)$ in ConceptNet, achieve better performance than shorter paths involving the same type of relations, e.g., $\left(\xleftrightarrow{\text{RelatedTo}}, \xleftrightarrow{\text{RelatedTo}}\right)$. Similar is the case in ATOMIC and DBpedia KGs, and if we did not construct the subgraphs of YAGO and WebChild with depth 1 then most probably the last two KGs would have revealed the same characteristic. This may appear strange at first, because one would anticipate that the closer two nodes in a graph are, the more semantically linked they are. This result is most likely due to the nature of our problem. Unlike entity resolution, for example, the nodes with which we are attempting to find a connection are of a

**Table 22**
The difference in performance for the relation pattern with clusters over the KGs.

| KG | Object-Action | Object-State | Subset Object-Action | Subset Object-State |
|---|---|---|---|---|
| ConceptNet | 0.006 | 0.017 | – | – |
| ConceptNet (no RelatedTo) | −0.018 | 0.152 | – | – |
| ATOMIC | 0.017 | −0.08 | **0.03** | **0.017** |
| YAGO | **0.036** | 0.032 | 0.007 | **0.017** |
| WebChild | 0.008 | **0.28** | 0.007 | **0.017** |
| DBpedia | 0.021 | 0.148 | 0.006 | **0.017** |

different type, namely object and action (or state). But what is even more interesting is that any given KG with this characteristic (i.e., has too many connections among its entities) leads to an "over-fitting" of knowledge to the point that it may contain noisy and conflicting information. Therefore, the exploitation of semantic information from a KG, can give us a hint that the knowledge in the KG needs refinement, or pruning.

Overall, the freedom in deciding how to deal with noise in the data is perhaps the most important benefit of employing the semantics of a KG. When importing new data, one can decide where to focus by carefully selecting which semantics to trust. Due to the domain-agnostic way of addressing the KG, other methods, such as data-driven models, which are more prone to noisy data, do not offer such adaptive behavior. This aspect is supported by the fact that our semantics-based method achieves better F1-scores, than the other commonly used methodologies over the KG of DBpedia and ConceptNet which are known for their noisy data.

Additionally, another important advantage when exploiting the semantics inside a KG is the generality scalability over different knowledge graphs, to find object-action and object-state relations. Tables 21 and 22 show that our Relation Pattern Method with and without clusters, which exploit the semantics in a KG, achieved better F1-scores than the other methods, over six different KGs. A fact that entails that methods which exploit the semantics in a KG are more generic, and they could be used on any KG that has different types of relations.

Notice that in our previous study [8] we compared some baseline deep learning methods with our method. More specifically, we used *AllenAI-CommonSense* [52], which constitutes the state of the art for link prediction in ConceptNet, by employing a pre-trained BERT [53] model that is fine-tuned on ConceptNet, using Graph Convolutional Networks (GCN) [54] for embedding the ConceptNet graph. This model returns a list of possible relations between a given pair of ConceptNet nodes, ranked in descending order of likelihood (aka confidence score).

The evaluation for AllenAI-CommonSense was performed only on ConceptNet, but the results even for one KG were not the expected ones, as the AllenAI-CommonSense (Top-k) methods, for Top-1 returned an F1-score of 0.289 compared to 0.699 of our method.

## 7. Conclusion

In this paper, we compared topology- and semantics-based methods for extracting object-action and object-state associations from knowledge graphs such as ConceptNet, WordNet, ATOMIC, YAGO, WebChild and DBpedia. We also presented a novel method for extracting and analyzing relationships between objects-actions and objects-states from knowledge graphs. In terms of F1-score, our method can improve current state-of-the-art performance. The flexibility in deciding how to deal with the noise in the data, as well as the capacity to assess the importance of a path through training rather than manual annotation, are two key features of our method. In the future, we plan to use our method in order to evaluate causal relations (i.e., in which states can the object be before and after we perform an action on it), which is a sensible next step that will build on the results of object-action and object-state associations that we present in this paper. A semantics-based method for identifying causal relations would be a significant contribution to AI systems, as it would be more generic and scalable than data-driven models, which are trained on specific datasets.

## CRediT authorship contribution statement

**Alexandros Vassiliades:** Conceptualization, Methodology, Software, Data creation, Data curation, Writing – original draft, Visualization, Investigation. **Theodore Patkos:** Conceptualization, Methodology. **Vasilis Efthymiou:** Conceptualization, Methodology, Evaluation. **Antonis Bikakis:** Supervision, Contribution in writing. **Nick Bassiliades:** Supervision, Contribution in writing. **Dimitris Plexousakis:** Supervision.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Alexandros Vassiliades reports financial support was provided by Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT).

## Data availability

Data will be made available on request.

## References

[1] R. Speer, J. Chin, C. Havasi, ConceptNet 5.5: An open multilingual graph of general knowledge, in: AAAI, 2017, pp. 4444–4451.
[2] C. Fellbaum, WordNet: An Electronic Lexical Database and Some of Its Applications, MIT Press, Cambridge, 1998.
[3] M. Sap, R. Le Bras, E. Allaway, C. Bhagavatula, N. Lourie, H. Rashkin, B. Roof, N.A. Smith, Y. Choi, Atomic: An atlas of machine commonsense for if-then reasoning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 3027–3035.
[4] N. Tandon, G. De Melo, G. Weikum, Webchild 2.0: Fine-grained commonsense knowledge distillation, in: Proceedings of ACL 2017, System Demonstrations, 2017, pp. 115–120.
[5] T. Rebele, F. Suchanek, J. Hoffart, J. Biega, E. Kuzey, G. Weikum, YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames, in: International Semantic Web Conference, Springer, 2016, pp. 177–185.
[6] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, Dbpedia: A nucleus for a web of open data, in: The Semantic Web, Springer, 2007, pp. 722–735.
[7] D. Zheng, X. Song, C. Ma, Z. Tan, Z. Ye, J. Dong, H. Xiong, Z. Zhang, G. Karypis, Dgl-ke: Training knowledge graph embeddings at scale, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 739–748.
[8] A. Vassiliades, T. Patkos, V. Efthymiou, A. Bikakis, N. Bassiliades, D. Plexousakis, Object-action association extraction from knowledge graphs, in: Further with Knowledge Graphs, IOS Press, 2021, pp. 241–254.
[9] A. Chiatti, E. Motta, E. Daga, Robots with commonsense: Improving object recognition through size and spatial awareness, in: CEUR, 2022.
[10] H. Xu, G. Qi, J. Li, M. Wang, K. Xu, H. Gao, Fine-grained image classification by visual-semantic embedding, in: IJCAI, 2018, pp. 1043–1049.
[11] S. Aditya, Y. Yang, C. Baral, C. Fermuller, Y. Aloimonos, From images to sentences through scene description graphs using commonsense reasoning and knowledge, 2015, arXiv preprint arXiv:1511.03292.
[12] A. Zareian, S. Karaman, S.-F. Chang, Bridging knowledge graphs to generate scene graphs, in: European Conference on Computer Vision, Springer, 2020, pp. 606–623.
[13] E. Amador-Domínguez, E. Serrano, D. Manrique, J.F. De Paz, Prediction and decision-making in intelligent environments supported by knowledge graphs, a systematic review, Sensors 19 (8) (2019) 1774.
[14] X. Zou, A survey on application of knowledge graph, J. Phys.: Conf. Ser. 1487 (2020) 12–16.

[15] R.T. Icarte, J.A. Baier, C. Ruz, A. Soto, How a general-purpose commonsense ontology can improve performance of learning-based image retrieval, in: IJCAI, 2017, pp. 1283–1289.
[16] C. Lee, W. Fang, C. Yeh, Y.F. Wang, Multi-label zero-shot learning with structured knowledge graphs, in: CVPR, 2018, pp. 1576–1585.
[17] S. Chernova, V. Chu, A. Daruna, H. Garrison, M. Hahn, P. Khante, W. Liu, A. Thomaz, Situated Bayesian reasoning framework for robots operating in diverse everyday environments, in: ISRR, vol. 10, 2017, pp. 353–369.
[18] J. Young, V. Basile, L. Kunze, E. Cabrio, N. Hawes, Towards lifelong object learning by integrating situated robot perception and semantic web mining, in: ECAI, 2016, pp. 1458–1466.
[19] J. Young, V. Basile, M. Suchi, L. Kunze, N. Hawes, M. Vincze, B. Caputo, Making sense of indoor spaces using semantic web mining and situated robot perception, in: ESWC, 2017, pp. 299–313.
[20] Y. Zhou, S. Schockaert, J. Shah, Predicting conceptnet path quality using crowdsourced assessments of naturalness, in: WWW, 2019, pp. 2460–2471.
[21] M. Gardner, P.P. Talukdar, J. Krishnamurthy, T.M. Mitchell, Incorporating vector space similarity in random walk inference over knowledge bases, in: EMNLP, 2014, pp. 397–406.
[22] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, S. Liu, Modeling relation paths for representation learning of knowledge bases, in: EMNLP, 2015, pp. 705–714.
[23] Z. Fu, Y. Xian, R. Gao, J. Zhao, Q. Huang, Y. Ge, S. Xu, S. Geng, C. Shah, Y. Zhang, et al., Fairness-aware explainable recommendation over knowledge graphs, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 69–78.
[24] Y. Xian, Z. Fu, S. Muthukrishnan, G. De Melo, Y. Zhang, Reinforcement knowledge graph reasoning for explainable recommendation, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 285–294.
[25] L. Gan, D. Nurbakova, L. Laporte, S. Calabretto, Enhancing recommendation diversity using determinantal point processes on knowledge graphs, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 2001–2004.
[26] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A.K. Bozcuoglu, G. Bartels, Know Rob 2.0 - A 2nd generation knowledge processing framework for cognition-enabled robotic agents, in: ICRA, 2018, pp. 512–519.
[27] M. Beetz, F. Bálint-Benczédi, N. Blodow, D. Nyga, T. Wiedemeyer, Z.-C. Marton, ROBOSHERLOCK: Unstructured information processing for robot perception, in: ICRA, 2015, pp. 1549–1556.
[28] K. Murugesan, M. Atzeni, P. Shukla, M. Sachan, P. Kapanipathi, K. Talamadupula, Enhancing text-based reinforcement learning agents with commonsense knowledge, 2020, CoRR abs/2005.00811.
[29] A.A. Daruna, W. Liu, Z. Kira, S. Chernova, RoboCSE: Robot common sense embedding, in: ICRA, 2019, pp. 9777–9783.
[30] Y. Zhu, A. Fathi, L. Fei-Fei, Reasoning about object affordances in a knowledge base representation, in: European Conference on Computer Vision, Springer, 2014, pp. 408–424.
[31] J. Zhu, Z. Nie, X. Liu, B. Zhang, J.-R. Wen, Statsnowball: A statistical approach to extracting entity relationships, in: Proceedings of the 18th International Conference on World Wide Web, 2009, pp. 101–110.
[32] J.-B. Alayrac, I. Laptev, J. Sivic, S. Lacoste-Julien, Joint discovery of object states and manipulation actions, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2127–2136.
[33] S.P. Tipper, M.A. Paul, A.E. Hayes, Vision-for-action: The effects of object property discrimination and action state on affordance compatibility effects, Psychon. Bull. Rev. 13 (3) (2006) 493–498.
[34] N. Aboubakr, J. Crowley, R. Ronfard, Recognizing Manipulation Actions from State-Transformations (Technical Report) (Ph.D. thesis), Univ. Grenoble Alps, CNRS, Inria, Grenoble INP, LIG, 38000 Grenoble, France, 2019.
[35] P. Isola, J.J. Lim, E.H. Adelson, Discovering states and transformations in image collections, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1383–1391.
[36] A. Fire, S.-C. Zhu, Learning perceptual causality from video, ACM Trans. Intell. Syst. Technol. 7 (2) (2015) 1–22.
[37] A. Fire, S.-C. Zhu, Inferring hidden statuses and actions in video by causal reasoning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 48–56.
[38] A. Fathi, J.M. Rehg, Modeling actions through state changes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 2579–2586.
[39] X. Wang, A. Farhadi, A. Gupta, Actions˜ transformations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2658–2667.
[40] M. Jiang, Improving situational awareness with collective artificial intelligence over knowledge graphs, in: Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II, vol. 11413, International Society for Optics and Photonics, 2020, p. 114130J.
[41] A. Vassiliades, N. Bassiliades, F. Gouidis, T. Patkos, A knowledge retrieval framework for household objects and actions with external knowledge, in: SEMANTiCS, in: Lecture Notes in Computer Science, vol. 12378, 2020, pp. 36–52.
[42] H. Wang, J.M. Hernandez, P. Van Mieghem, Betweenness centrality in a weighted network, Phys. Rev. E 77 (4) (2008) 046105.
[43] J.M. Hernández, P. Van Mieghem, Classification of Graph Metrics, Delft University of Technology, Mekelweg, the Netherlands, 2011, pp. 1–20.
[44] C. Yu, X. Zhao, L. An, X. Lin, Similarity-based link prediction in social networks: A path and node combined approach, J. Inf. Sci. 43 (5) (2017) 683–695.
[45] P. Dey, S. Medya, Manipulating node similarity measures in networks, in: AAMAS, 2020, pp. 321–329.
[46] G. Zhu, C.A. Iglesias, Exploiting semantic similarity for named entity disambiguation in knowledge graphs, Expert Syst. Appl. 101 (2018) 8–24.
[47] G. Zhu, C.A. Iglesias, Computing semantic similarity of concepts in knowledge graphs, IEEE Trans. Knowl. Data Eng. 29 (1) (2017) 72–85.
[48] A. Rossi, D. Firmani, A. Matinata, P. Merialdo, D. Barbosa, Knowledge graph embedding for link prediction: A comparative analysis, 2020, CoRR abs/2002.00819.
[49] C. Malaviya, C. Bhagavatula, A. Bosselut, Y. Choi, Commonsense knowledge base completion with structural and semantic context, in: AAAI, 2020, pp. 2925–2933.
[50] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: NAACL-HLT, 2019, pp. 4171–4186.
[51] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR, 2017.
[52] C. Malaviya, C. Bhagavatula, A. Bosselut, Y. Choi, Commonsense knowledge base completion with structural and semantic context, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 2925–2933.
[53] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint arXiv:1810.04805.
[54] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.