Further with Knowledge Graphs. M. Alam et al. (Eds.)
AKA Verlag and IOS Press, 2021

doi:10.3233/SSW210048

Object-Action Association Extraction from Knowledge Graphs

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

[a] *Aristotle University of Thessaloniki, School of Informatics*
[b] *Institute of Computer Science, Foundation for Research and Technology Hellas*
[c] *University College London Department of Information Studies*

Abstract. Infusing autonomous artificial systems with knowledge about the physical world they inhabit is of utmost importance and a long-lasting goal in Artificial Intelligence (AI) research. Training systems with relevant data is a common approach; yet, it is not always feasible to find the data needed, especially since a big portion of this knowledge is commonsense. In this paper, we propose a novel method for extracting and evaluating relations between objects and actions from knowledge graphs, such as ConceptNet and WordNet. We present a complete methodology of locating, enriching, evaluating, cleaning and exposing knowledge from such resources, taking into consideration semantic similarity methods. One important aspect of our method is the flexibility in deciding how to deal with the noise that exists in the data. We compare our method with typical approaches found in the relevant literature, such as methods that exploit the topology or the semantic information in a knowledge graph, and embeddings. We test the performance of these methods on the Something-Something Dataset.

Keywords. Relation Pattern Method, Knowledge Graph, Semantics-Based Method

1. Introduction

Humans are able to understand relations between *real-world objects and actions* relying not only on observations, but also on their commonsense knowledge. Machines, on the other hand, need a large quantity of data, in order to learn and reason about object-action relations, as for instance to correlate the object *Knife* with the action *Cut*. Yet, recognizing such types of associations is crucial for a wide spectrum of applications involving autonomous entities. Commonsense Knowledge Graphs (KGs), such as ConceptNet [1] and WordNet [2], contain to some extent knowledge about object-action relations, and can help construct knowledge bases, which subsequently can be used by machines. However, inserting knowledge into a knowledge base from crowd-built KGs hides risks, as these may contain information that is noisy or false. Therefore, evaluation methods are crucial when exploiting knowledge from such KGs.

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

A method that correctly identifies positive and negative object-action associations in the presence of noise can increase the quality of data that a machine can utilize, which in turn can improve the performance of autonomous, artificial intelligence (AI) systems. Driven by this need, in this paper we compare a number of methods of different nature that are commonly used in practice to extract associations from KGs, organizing them into *topology-based*, *semantics-based* and *embeddings-based*. We also introduce a novel semantics-based approach to identify such object-action correlations that can achieve or improve state-of-the-art performance, while offering flexibility in ironing out noise. Its main characteristic is the exploitation of patterns of relations, which carry important information as to which associations to trust and which to dismiss.

Informally, the problem we aim to solve is the following: given a directed KG and a pair of nodes, one of which refers to a real-world object class and the other to a real-world action class, we try to infer whether these two nodes are associated or not, and to what degree, i.e., if the action can be performed by/on that object. This demand is amplified by the volume of current research in fields, such as robotic manipulation, where object affordances play a key role in enabling a robot to accomplish tasks (see for instance [3] for a recent survey of the relevant literature). Of course, the methods described in our study are not limited to the household domain, but can be applied for the detection of a broader class of associations; yet, framing our analysis on the given domain helps us compare more accurately the behaviour of diverse methods.

The main contributions of this paper are (a) a comparative analysis of popular methods for extracting associations from KGs focusing on a specific domain, that of household objects, (b) the proposal of a new, enhanced method that lays more emphasis on the semantic knowledge that exists in the KG, and (c) the generation of a dataset of positive and negative object-action relations, comprising labels that are commonly used for benchmarking both research and practical approaches. Our method and dataset are publicly available[2].

The rest of this paper is structured as follows. Section 2 discusses related work. Section 3 presents the existing and proposed methods for identifying object-action relations. Section 4 describes our experimental evaluation, and Section 5 concludes the paper.

2. Related Work

Extracting commonsense knowledge from problem-agnostic repositories has been applied to a diversity of AI-related domains to solve various problems. The authors of [4] rely upon ConceptNet to identify word similarities, which they then use in order to improve the performance of sentence-based image retrieval algorithms. A more elaborate use of KGs is presented in [5], where the authors approach the problem of zero-shot label learning in images by creating KGs based on labels detected visually and on correlations found in external sources. The authors rely on WordNet to populate the graph and use Wu Palmer similarity[3] to specify the properties. In [6], the authors assign labels to a visual scene using Bayesian logic networks and relying on commonsense knowledge extracted from WordNet, ConceptNet, and Wikipedia. WordNet is utilized in order to disambiguate seed words with the aid of their hypernym. ConceptNet properties, such as

[2] https://github.com/valexande/Semantics-2021

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

LocatedAt or *UsedFor*, which may pinpoint the location of an object, are also retrieved. With this method, the system can generate a compact semantic knowledge base given only a small number of objects. Similar methods are used in [7,8,9,10].

The aforementioned studies attempt to integrate knowledge from general-purpose Web resources in a KG without, however, paying much attention to the validity of the information extracted from such resources. Moreover, they rely on the rather simplistic assumption that if two nodes are connected via any edge, then the two nodes are semantically related. We, on the other hand, try to iron out the noise or erroneous information that might exist in such Web resources before adding new knowledge to a KG.

The representation, as well as identification, of object-action relations has been the focus of interest of many studies in the field of cognitive robotics. In the projects KnowRob [11] and RoboSherlock [12], semantic correlation of physical entities is indeed captured, yet object-action relations are either learned exclusively through observed data, or captured in a problem-specific way. In [13], the authors integrate knowledge from ConceptNet in a KG. Given an object or action label, the authors construct subgraphs of ConceptNet with only two properties, in order to train a data-driven model, which can predict if an object is related with an action. Similar approaches are also used in RoboCSE [14], which uses embeddings to represent object and action labels and infer object-action relations based on the similarity of their vectors. Our proposed method exploits both semantically relevant and commonsense information captured in general-purpose repositories, which can complement and enrich the outcomes of the aforementioned studies.

The study of Zhou et al. [15] is more closely related to ours. The authors train a Long Short-Term Memory (LSTM) to predict the path between two nodes of the ConceptNet graph. They collect, for a set of node pairs, the most quality paths, defining quality as the most natural set of edges that connects two given nodes. For instance, the path *Lead* $\xrightarrow{\text{HasProperty}}$ *Toxic* $\xleftrightarrow{\text{RelatedTo}}$ *Lethal* $\xleftrightarrow{\text{RelatedTo}}$ *Poison* is considered the most natural among those connecting *Lead* and *Poison*. The quality of paths is annotated manually by a group of volunteers. Similarly, in [16,17] a data-driven model predicts a path between two nodes of ConceptNet; the quality of a path is hand-coded by the authors. Our method, on the other hand, aims to determine the importance of a path through training, rather than through manual annotation. This has two benefits: it takes into account, to a larger extent, the structural and semantic characteristics of the underlying KG and it is more adaptive to changes in the KG or the application domain.

3. Methodology

In this section, we formulate the problem and describe the different methods we evaluated. We classify the methods based on the information they utilize into *Topology-based*, *Semantics-based* and *Embeddings-based*. Topology-based methods exploit the structure of the graph, while semantics-based methods also take into account the types of relations connecting the two nodes. Embeddings-based methods use vector representations of graphs, potentially taking into account the structure of the graph, as well as the semantics of the node labels. Our novel **Relation Pattern Method** is part of the semantics-based methods.

3.1. Problem Formulation

The problem we aim to solve is the following. Given a directed knowledge graph $G = (E, R)$, where E is the set of nodes, corresponding to entities, and R is set of edges, corresponding to relations, and a pair of nodes (e_1, e_2) with $e_1, e_2 \in E$, where e_1 represents an action and e_2 an object (E may contain other types of nodes as well), find whether e_1 and e_2 are related. We consider the two nodes, e_1 and e_2, as related 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.

Before presenting the methods we evaluated to solve the aforementioned problem, we first describe how we can create the graph G from a given set of labels L that refer to real-world objects or actions. We extract the object and action labels from the Something-Something Dataset[4], a dataset that is commonly used by the Machine Vision community (see Section 4.1 for more details). Note, however, that any set of object and action labels can be used to create G. For every label $l_i \in L$, we generate a graph S_i, by appending information relevant to l_i from ConceptNet [1] and WordNet [2] in two steps. Then, we construct G by unifying all $|L|$ graphs $S_1, \ldots, S_{|L|}$, i.e., every graph S_i is a subgraph of G.

Step 1: For each object or action label, we search for a node with the same lemmatized label in the ConceptNet knowledge graph and extract a subgraph containing a subset of the properties found in ConceptNet that are considered relevant to the domain of interest. The subgraphs contain 2-hop paths from the object or action label. The edge types we consider are:

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

We omit only 3 relations from ConceptNet[5]: *Causes* and *Desires*, which, although seemingly relevant, their use is human centric and they describe the sentiments that are caused to humans after an event, and *ExternalURL*, in order not to append information from other external resources, except WordNet.

Step 2: The next step is to insert context knowledge into the subgraph. We retrieve knowledge from WordNet by looking at the super-classes of each node in the subgraph created in **Step 1**, and if any super-class of a node falls into a domain-specific category of super-classes, then we keep the node in the graph, otherwise we delete it. The super-classes we consider are:

[4] https://20bn.com/datasets/something-something

[5] https://github.com/commonsense/conceptnet5/wiki/Relations

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

We consider this specific set of classes following the findings of [18], which showed that almost all nodes in the WordNet graph that refer to a real-world object or action have at least one of these as a super-class. This pruning of nodes based on WordNet super-classes can give domain-specific concepts, e.g., when interested in household appliances. Figure 1 shows part of the subgraph for the label *Knife*. We highlight in red the node that is pruned in **Step 2**.

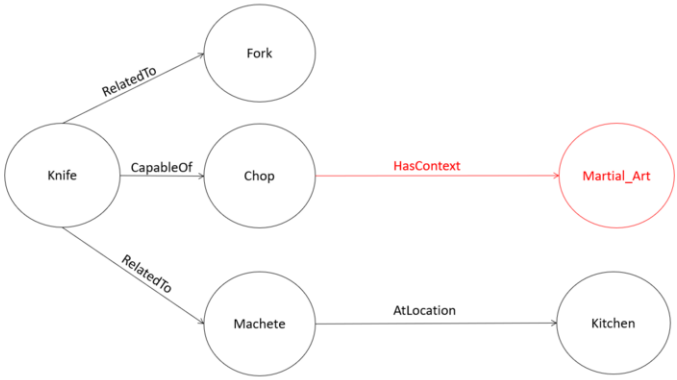

Figure 1. Part of the subgraph for the label *Knife*. The red node is pruned in **Step 2**.

After creating a graph for each object and action 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 $R = \bigcup_{i=1}^{n} R_i$.

3.2. Topology-based Methods

We apply the two most commonly used methods proposed in the relevant literature [10,9] that exploit the topology of a graph, in order to infer the extent to which two nodes are related.

The *Connecting Paths Method* takes into consideration each sequence of edges that begins from the object node and reaches the action node after a finite number of steps, or vice versa. The authors omit paths that contain loops, but do not take into account the type of edges a path contains. Given two subgraphs S_1 and S_2, as described in Section 3.1, corresponding to an object node and an action node respectively, 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 node, C_2 is the set of paths that start from the action 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 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 *and let S_{knife} have two paths that start from the node knife, namely Knife $\xrightarrow{UsedFor}$ Butter and Knife $\xrightarrow{LocatedAt}$ Kitchen, and S_{fold} have only one path, Fold $\xrightarrow{HasContext}$ Cooking $\xleftarrow{UsedFor}$ Butter. Then, the connectPath metric will return*

$$connectPath(S_{knife}, S_{fold}) = \frac{|C_{knife} \cup C_{fold}|}{|P_{knife} \cup P_{fold}|} = \frac{1+1}{2+1} = 0.666$$

Some recent studies that apply this method, as is or with small variations, are [16,4,5,15]. In fact, they also focus on inferring object-action relations ([16,5]) and on object identification ([4,15]).

The *Common Nodes Method* divides the number of common nodes by the number of total nodes in two given graphs. Two nodes are considered common when they refer to the same entity in ConceptNet, i.e., the nodes have the same label. Duplicate nodes are cleared, allowing only one occurrence of each node. 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 one common node,* Butter*, and 5 distinct nodes in total.*

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

$$\frac{|\{Knife, Butter, Kitchen\} \cap \{Fold, Cooking, Butter\}|}{|\{Knife, Butter, Kitchen\} \cup \{Fold, Cooking, Butter\}|} = \frac{1}{5} = 0.2$$

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 that apply this method, as is or with small variations, are [4,19,20]. The focus is on object identification and on finding the similarity of two nodes in a knowledge graph.

3.3. Semantics-based Methods

As a semantics-based method, we consider the very popular WUP similarity, and we also present a novel **Related Pattern Method**, which exploits the pattern of connections in a KG to infer whether two nodes are semantically related.

The *Wu Palmer Similarity* (WUP) uses the acyclic graph of WordNet to calculate relatedness by considering the depth of two nodes in the WordNet taxonomies, along with the depth of their LCS (Least Common Subsumer). Given two nodes from the WordNet acyclic graph, the LCS of these nodes is their most specific common ancestor. The score can never be zero because the depth of the LCS is never zero (the depth of the root of the taxonomy is one). This metric calculates the similarity based on how close the nodes are to each other in the WordNet acyclic graph. The WUP similarity between an object node (n_o) and an action 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.

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

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

$$= 2 * \frac{depth(physical\_entity)}{depth(knife) + depth(fold)} = 2 * \frac{2}{12+6} = 0.222$$

Many studies use the WUP similarity in a wide spectrum of domains. Recent studies, such as [5,18], use WUP scores to infer object-action relations and object identification.

Our proposed method, the **Relation Pattern Method**, is based on the assumption that some of the paths connecting two nodes carry more semantically relevant information than others. For instance, the path *object node* $\xleftrightarrow{\text{Synonym}}$ *node0* $\xleftrightarrow{\text{ReceivesAction}}$ *action node* may appear more often in object-action pairs compared to paths composed of other relations. To verify this assumption, we selected a domain-specific subset

$$[RelatedTo,\ UsedFor,\ ReceivesAction,\ CapableOf,\ Synonym]$$

of the ConceptNet relations that we considered more relevant to the problem at hand; yet, this subset can change according to the context of the problem. An important aspect of our proposed method is the flexibility in deciding how to deal with the noise that exists in the data.

A *relation pattern* is any connecting path that is composed of at least one of the aforementioned relations, except from paths that only contain the relation *Synonym*. The latter are omitted, to avoid connecting an object and an action node having similar labels. In the end, 155 different relation patterns were produced; whenever a relation pattern is found between an object and an action node in the KG, we consider it as an indication that the two nodes are associated. If $\mathscr{P} = \{pattern_1, \ldots, pattern_{155}\}$ is the set of all relation patterns, then, for each $pattern_i \in \mathscr{P}$, the goal is to assign a weight of importance $W_{pattern_i}$, in order to specify how confident we are that the given pattern produces correct associations. For instance, in the next section, we assign the weights based on how well each pattern performs in our training data. Of course, other heuristics can be used instead.

Since it is reasonable to consider more than one patterns before reaching a conclusion about the relation between two labels, one can group together patterns, based on

their performance, their domain-specific relevance, or other criteria. For quantifying the performance of a cluster W_C, one can consider, for instance, the weighted sum of the weights of each individual pattern, the max or min of these weights, or other heuristics-based metrics. In our evaluation, we adopt an even simpler approach as the baseline case, namely to treat all patterns with weight above a given threshold as equally relevant.

3.4. Embeddings-based Methods

Recently, there has been a surge of interest in the field of KG embeddings for the task of link prediction [21]. Studies following this methodology represent nodes of a KG as vectors in a latent space, which are generated by taking into account both the textual and structural features of those nodes. The textual features are considered by acquiring the word and sentence embeddings of node labels, from word embeddings that have been pre-trained on large document corpora, such as Wikipedia. The structural features consider, typically in an iterative way, the node embeddings of each node's neighbors in a KG.

For example, *AllenAI-CommonSense* [22], which constitutes the state of the art for link prediction in ConceptNet, employs a pre-trained BERT [23] model that is fine-tuned on ConceptNet, using Graph Convolutional Networks (GCN) [24] 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).

We can use the results of this system in two different ways for our purposes: a) we can either consider the confidence score returned by AllenAI-CommonSense for a specific relation (e.g., ReceivesAction), given two query nodes from our graph G, or b) we can consider that the answer to the problem formulated in Section 3.1 is positive for two query nodes, when the relation "ReceivesAction" is within the top answers for those query nodes.

4. Evaluation

In this section, we first describe how we created the ground truth from the Something-Something Dataset[6] and then we discuss the experimental setup and the results that each method achieved.

4.1. Data Collection

Rather than using a random set of action and object labels, aiming to achieve an adequate coverage of entities for the household domain, we decided to extract the set of labels for our evaluation from the Something-Something Dataset. Something-Something consists of a large collection of short video clips (more than 220k) containing actions performed on and with common household objects. The actions involve either one type of object (e.g., opening a bottle) or two distinct types of objects (e.g., putting coins inside a box). Due to its vast number of sample videos, the Something-Something Dataset has become a de-facto benchmark for the assessment of systems addressing the task of action recog-

[6] `https://20bn.com/datasets/something-something`

nition. The dataset provides for each clip a small description that contains action and object(s) labels.

Ground Truth Creation: From Something-Something we initially extracted 247 object labels and 35 action labels, which produced 8,645 object-action pairs. We replaced all object labels in plural form with their singular form, for example *notes* was replaced with *note*. Then, we removed certain object and action labels that we did not consider context related[7]. Next, for the remaining action and object labels, we issued a query to the ConceptNet KG using the ConceptNet Web API[8], in order to identify which labels are indeed part of the graph. We ended up with 148 object labels and 25 action labels. Since some actions have the same label with some objects (3 in total), we renamed these labels as follows: (a) pile $\rightarrow$ *pileO* and pile $\rightarrow$ *vpile*, (b) stack $\rightarrow$ *stackO* and stack $\rightarrow$ *vstack*, and (c) cover $\rightarrow$ *coverO* and cover $\rightarrow$ *vcover*, to refer to the object and action label, respectively.

Eventually, 3,700 object-action relations were kept in total. Those 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 but it so happens that no clip in the dataset refered to them. At the end, 1,965 positive and 1,735 negative object-action relations were produced, forming our ground truth[9].

4.2. Experimental Setup

The evaluation of the methods described in Section 3 was performed using 10-fold cross validation over the 3,700 positive and negative relations described in Section 4.1. We used Sklearn[10] to split our data into 10 folds. Each fold contained 370 relations, 52% of which were positive and 48% negative, which reflects the distribution of the relations in the original dataset.

Each iteration of the 10-fold cross-validation process was used, in order to train the different models. Specifically, for the *Connecting Path Method*, the *WUP*, and the *Common Node Method*, the training folds helped specify the optimal threshold for each method that maximizes the F1 score. For the *Relation Pattern Method*, the training phase helped compute the weights of importance $W_{pattern_i}$ of each relation pattern $pattern_i \in \mathscr{P}$, as described in Section 3.3. During testing, we measured the performance of each method with the given thresholds and weights. Patterns that performed poorly during training were omitted completely.

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 nodes (n_o, n_a) is related in the ground truth and also achieves a score above the threshold (for the threshold-based methods) or the pattern under consideration connects node n_o with n_a (for the pattern-based methods)

[7]The reader can find all the labels that were removed or replaced in our documentation: `https://github.com/valexande/Semantics-2021`

[8]`https://pypi.org/project/ConceptNet/`

[9]The dataset of positive and negative object action relations can be found in our documentation: `https://github.com/valexande/Semantics-2021`

[10]`https://scikit-learn.org/stable/modules/cross\_validation.html`

- FP is when a pair of object-action nodes (n_o, n_a) is not related in the ground truth, but achieves a score above the threshold (for the threshold-based methods) or the pattern under consideration connects node n_o with n_a (for the pattern-based methods)
- TN is when a pair of object-action nodes (n_o, n_a) is not related in the ground truth and also achieves a score below the threshold (for the threshold-based methods) or the pattern under consideration does not connect node n_o with n_a (for the pattern-based methods)
- FN is when a pair of object-action nodes (n_o, n_a) is related in the ground truth, but achieves a score below the threshold (for the threshold-based methods) or the pattern under consideration does not connect node n_o with n_a (for the pattern-based methods)

Finally, note that we define the weight of importance $W_{pattern_i}$ for $pattern_i$ as the harmonic mean between precision P and recall R (Equation 4).

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

Example 4 *Consider the relation pattern* $\left(\xrightarrow{RelatedTo}, \xrightarrow{UsedFor}\right)$ *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* $\left(\xrightarrow{RelatedTo}, \xrightarrow{UsedFor}\right)$ *connecting the two nodes (see Section 3.2). Using this information, we get the following scores.*

$$P = \frac{TP}{TP+FP} = \frac{2}{2+1} = 0.666 \text{ and } R = \frac{TP}{TP+FN} = \frac{2}{2+0} = 1$$

$$W_{pattern\left(\xrightarrow{RelatedTo}, \xrightarrow{UsedFor}\right)} = \frac{4}{5} = 0.8$$

TP is 2 because the pairs knife-cut and knife-stab are related in the ground truth and the relation path $\left(\xrightarrow{\text{RelatedTo}}, \xrightarrow{\text{UsedFor}}\right)$ is a connecting path in both. FP is 1 because the pair knife-fold are not related in the ground truth and the relation path $\left(\xrightarrow{\text{RelatedTo}}, \xrightarrow{\text{UsedFor}}\right)$ is a connecting path. FN is 0 because we do not have a pair that is related in our ground truth and does not not have $\left(\xrightarrow{\text{RelatedTo}}, \xrightarrow{\text{UsedFor}}\right)$ as a connecting path. The final score of Example 4 shows that the weight of importance $W_{pattern\left(\xrightarrow{\text{RelatedTo}}, \xrightarrow{\text{UsedFor}}\right)}$ 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.

Additionally, we evaluated the embeddings-based method AllenAI-Common Sense (top-k) over all ground truth object-action pairs, both positive and negative, by testing

whether the relation "ReceivesAction" was within the top-k results for each pair, for $k \in \{1,3,5\}$. If "ReceivesAction" is within the top-k results, we consider this as a predicted positive pair, otherwise, a predicted negative, and follow the same conventions (TP, FP, TN, FN) as described above.

We note that another variation of this approach would have been to restrict the predicted results to those having a confidence score above a predefined threshold. However, our experiments showed that this method performs best when such minimum confidence threshold is 0 (confidence scores are extremely low in too many cases), so we do not report numbers for this variation.

4.3. Results

Table 1 summarizes the overall performance measures for each method. Although the differences among the first three popular approaches are small, the WUP similarity seems to achieve higher scores both in terms of accuracy (.555) and of F1 score (.696). We also see that there are patterns that achieve similar or better scores in the one or the other measure, but not in both (the weight of importance coincides with the F1 score). Due to the plurality of relation patterns, we display only the Top-20 relation patterns. The AllenAI-CommonSense (top-k) methods, 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).

An investigation of the figures for the Relation Pattern method reveals some interesting insights, not easily detectable with the other methods. First of all, we can see that at least one occurrence of the relation *RelatedTo* exists in almost all relation patterns. This is because, although not explicitly stated in the ConceptNet documentation[11], *RelatedTo* plays the role of a super-property, i.e., it subsumes the other relations. While one would expect that less abstract relations among nodes, such as *UsedFor*, would produce better results, this is not the case. This conclusion reflects, to some extent, the quality of data in ConceptNet and provides hints as to where there exists room for data cleaning.

We also observe that certain longer paths, such as $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{RelatedTo}}{\longleftrightarrow} \right)$, achieve better performance than shorter paths involving the same type of relations, e.g., $\left(\overset{\text{RelatedTo}}{\longleftrightarrow}, \overset{\text{RelatedTo}}{\longleftrightarrow} \right)$. This might seem odd at first, as one would expect that the closer two nodes are in the graph, the more semantically tightly related they would be. This finding is probably owed to the nature of our problem. In contrast to entity resolution for instance, the nodes whose association we try to find are of different type, namely object and action.

Of course, such similar paths obtain practical meaning if considered as a group, rather than as individuals. For this reason, Table 1 also reports indicatively the performance of two clusters of patterns, one that is composed only of *RelatedTo* and *Synonym* relations, and one composed of *UsedFor* and *Synonym* relations. We adopt a simplistic approach in deciding what the answer of a cluster is: any pattern above a threshold is considered relevant. As such, even if a single pattern is found in the graph, the corresponding object-action pair is considered related. As we only wish to measure a baseline

[11] `https://github.com/commonsense/conceptnet5/wiki/Relations`

Table 1. Overall scores.

Method	Accuracy	Recall	Precision	F1 Score
Connecting Path	0.534	0.752	0.552	0.625
WUP	0.555	0.951	0.551	0.696
Common Node	0.551	0.956	0.548	0.695
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	0.697
RelatedTo RelatedTo	0.539	0.985	0.536	0.695
RelatedTo Synonym	0.558	0.906	0.557	0.688
RelatedTo RelatedTo RelatedTo	0.561	0.891	0.56	0.686
RelatedTo RelatedTo RelatedTo Synonym	0.561	0.835	0.564	0.67
RelatedTo Synonym RelatedTo RelatedTo	0.552	0.84	0.556	0.667
Synonym RelatedTo	0.528	0.611	0.556	0.579
Synonym RelatedTo RelatedTo RelatedTo	0.517	0.642	0.532	0.567
RelatedTo Synonym RelatedTo RelatedTo	0.525	0.543	0.561	0.549
RelatedTo Synonym RelatedTo Synonym	0.537	0.497	0.58	0.531
UsedFor UsedFor	0.528	0.484	0.569	0.521
Synonym Synonym RelatedTo RelatedTo	0.488	0.338	0.606	0.394
RelatedTo Synonym RelatedTo	0.509	0.275	0.593	0.37
Synonym RelatedTo RelatedTo	0.518	0.28	0.612	0.361
RelatedTo RelatedTo Synonym	0.518	0.253	0.584	0.346
Synonym RelatedTo UsedFor RelatedTo	0.507	0.228	0.595	0.315
Synonym RelatedTo RelatedTo Synonym	0.509	0.213	0.588	0.301
RelatedTo Synonym UsedFor RelatedTo	0.488	0.192	0.575	0.283
RelatedTo RelatedTo Synonym Synonym	0.487	0.176	0.567	0.264
Synonym RelatedTo Synonym RelatedTo	0.494	0.162	**0.613**	0.248
Cluster Relation Pattern	Accuracy	Recall	Precision	F1 Score (W_C)
RelatedTo-Synonym	0.539	**0.985**	0.546	**0.702**
UsedFor-Synonym	0.582	0.636	0.569	0.597

case, we set a rather generous threshold for including patterns in the cluster, namely any pattern with weight above 0.1.

More elaborate methods can of course be implemented, e.g., by taking into consideration the weight of importance among the patterns of each cluster or by utilizing domain-specific criteria. Yet, even with this baseline, we notice that clustering paths can produce improved state-of-the-art F1 scores (.702). By ignoring the relative importance of each individual pattern though, we end up introducing noise, as shown in the precision scores if compared to the best performing patterns, an aspect that a more advanced method could eliminate.

Overall, probably the most important advantage of our proposed method, beyond its prominent performance, is the flexibility in deciding how to deal with noise in the data. By carefully choosing which patterns to trust, one can decide where to focus when importing new data. Such an adaptive behavior is not offered by the other methods, such as data-driven models, which are more vulnerable to noisy data, due to the domain-agnostic way of treating the KG.

5. Conclusion

In this paper, we present a novel method for extracting and evaluating relations between objects and actions from KGs, such as ConceptNet and WordNet. We compared our method with popular approaches proposed in relevant literature, such as methods that exploit the topology or the semantic information in a KG, and embeddings. Our method can improve state-of-the art performance in terms of F1 scores. But its most important advantage, beyond its very good performance, is the flexibility in finding and adapting to the noise in the data. In the future, we plan to integrate knowledge from other commonsense knowledge graphs, such as ATOMIC [25,26], and to evaluate our methods on other types of relations, such as those between an object and a state, and causal relations (i.e., in which states can the object be before and after we perform an action on it).

References

[1] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, pages 4444–4451, 2017.
[2] Christiane Fellbaum. Wordnet: An electronic lexical database and some of its applications, 1998.
[3] Paola Ardón, Èric Pairet, Katrin Lohan, Subramanian Ramamoorthy, and Ronald Petrick. Building affordance relations for robotic agents - a survey. In *30th International Joint Conference on Artificial Intelligence (IJCAI-21)*, 2021.
[4] Rodrigo Toro Icarte, Jorge A. Baier, Cristian Ruz, and Alvaro Soto. How a general-purpose commonsense ontology can improve performance of learning-based image retrieval. In *IJCAI*, pages 1283–1289, 2017.
[5] Chung-Wei Lee, Wei Fang, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. Multi-label zero-shot learning with structured knowledge graphs. In *CVPR*, pages 1576–1585, 2018.
[6] Sonia Chernova, Vivian Chu, Angel Daruna, Haley Garrison, Meera Hahn, Priyanka Khante, Weiyu Liu, and Andrea Thomaz. Situated bayesian reasoning framework for robots operating in diverse everyday environments. In *ISRR*, 2017.
[7] Jay Young, Valerio Basile, Lars Kunze, Elena Cabrio, and Nick Hawes. Towards lifelong object learning by integrating situated robot perception and semantic web mining. In *ECAI*, pages 1458–1466, 2016.
[8] Jay Young, Valerio Basile, Markus Suchi, Lars Kunze, Nick Hawes, Markus Vincze, and Barbara Caputo. Making sense of indoor spaces using semantic web mining and situated robot perception. In *ESWC*, pages 299–313, 2017.
[9] Ganggao Zhu and Carlos Angel Iglesias. Computing semantic similarity of concepts in knowledge graphs. *IEEE Trans. Knowl. Data Eng.*, 29(1):72–85, 2017.
[10] Ganggao Zhu and Carlos A Iglesias. Exploiting semantic similarity for named entity disambiguation in knowledge graphs. *Expert Systems with Applications*, 101:8–24, 2018.
[11] Michael Beetz, Daniel Beßler, Andrei Haidu, Mihai Pomarlan, Asil Kaan Bozcuoglu, and Georg Bartels. Know rob 2.0 - A 2nd generation knowledge processing framework for cognition-enabled robotic agents. In *ICRA*, pages 512–519, 2018.
[12] Michael Beetz, Ferenc Bálint-Benczédi, Nico Blodow, Daniel Nyga, Thiemo Wiedemeyer, and Zoltán-Csaba Marton. Robosherlock: Unstructured information processing for robot perception. In *ICRA*, pages 1549–1556, 2015.

[13] Keerthiram Murugesan, Mattia Atzeni, Pushkar Shukla, Mrinmaya Sachan, Pavan Kapanipathi, and Kartik Talamadupula. Enhancing text-based reinforcement learning agents with commonsense knowledge. *CoRR*, abs/2005.00811, 2020.

[14] Angel Andres Daruna, Weiyu Liu, Zsolt Kira, and Sonia Chernova. Robocse: Robot common sense embedding. In *ICRA*, pages 9777–9783, 2019.

[15] Yilun Zhou, Steven Schockaert, and Julie Shah. Predicting conceptnet path quality using crowdsourced assessments of naturalness. In *WWW*, pages 2460–2471, 2019.

[16] Matt Gardner, Partha Pratim Talukdar, Jayant Krishnamurthy, and Tom M. Mitchell. Incorporating vector space similarity in random walk inference over knowledge bases. In *EMNLP*, pages 397–406, 2014.

[17] Yankai Lin, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. In *EMNLP*, pages 705–714, 2015.

[18] Alexandros Vassiliades, Nick Bassiliades, Filippos Gouidis, and Theodore Patkos. A knowledge retrieval framework for household objects and actions with external knowledge. In *SEMANTiCS*, volume 12378 of *Lecture Notes in Computer Science*, pages 36–52, 2020.

[19] Chuanming Yu, Xiaoli Zhao, Lu An, and Xia Lin. Similarity-based link prediction in social networks: A path and node combined approach. *Journal of Information Science*, 43(5):683–695, 2017.

[20] Palash Dey and Sourav Medya. Manipulating node similarity measures in networks. In *AAMAS*, pages 321–329, 2020.

[21] Andrea Rossi, Donatella Firmani, Antonio Matinata, Paolo Merialdo, and Denilson Barbosa. Knowledge graph embedding for link prediction: A comparative analysis. *CoRR*, abs/2002.00819, 2020.

[22] Chaitanya Malaviya, Chandra Bhagavatula, Antoine Bosselut, and Yejin Choi. Commonsense knowledge base completion with structural and semantic context. In *AAAI*, pages 2925–2933, 2020.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019.

[24] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[25] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *AAAI*, pages 3027–3035, 2019.

[26] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. COMET: commonsense transformers for automatic knowledge graph construction. In *ACL*, pages 4762–4779, 2019.