Fact Checking on Knowledge Graphs

Weichen Luo and Cheng Long

Abstract Fact checking, which verifies whether a given statement is true, could play a vital role in fake news detection. For example, for a given piece of news, a potential solution could involve a series of steps, including extracting statements from the news via text parsing, checking the validity of the extracted statements (i.e., fact checking), and classifying the news as fake if some statements have been confirmed to be false and performing further fake news detection processes otherwise. Considering that knowledge graphs are a popular way of representing knowledge, which could be used for verifying or counter-verifying statements, several solutions have been proposed that make use of knowledge graphs for fact checking. In this chapter, recent studies on fact checking with the help of knowledge graphs are reviewed, and three representative solutions, namely, Knowledge Linker, PredPath, and Knowledge Stream, are introduced with some details. Specifically, Knowledge Linker utilizes the semantic proximity metrics for mining knowledge graphs, PredPath employs the link prediction method and introduces a newly defined metric, and Knowledge Stream models the fact-checking problem as an optimization problem and uses flow theory for solving the problem.

Keywords Fact checking · Knowledge graph · Knowledge linker · Predicate path · Knowledge stream

1 Introduction

Rumors, misinformation, and fake news fill the Internet and social media these days, mostly due to the inability to identify fake news in large amounts of data quickly and accurately. These rumors and fake news will not only have negative impacts on public opinion but also affect people's judgment if they cannot be identified and corrected in a timely manner [13, 18, 23]. In order not to be misled, it is important

© Springer Nature Switzerland AG 2021

Deepak P et al., *Data Science for Fake News*, The Information Retrieval Series 42, https://doi.org/10.1007/978-3-030-62696-9_7

to separate true news from a large scale of information mixed with fake news.

Dozens of methods or models have been proposed to detect and prevent the spread of rumors or fake news [47]. Most approaches are based on the contextual indicators of fake news for detecting the veracity of information, such as the abundance of inquiry tweets, the credibility of the information source, and the temporal patterns of news spread. A closely related issue is the evaluation of those statements that are presented in news media. This issue is called *fact checking*.

In order to be able to utilize as much information or fact data as possible, knowledge graphs (KGs) [31] are introduced to structure the existing knowledge and facts. Several models have been proposed for the fact-checking problem, which are based on knowledge graphs, including Knowledge Linker (KL) [11], PredPath [37], Knowledge Stream (KS) [39], PRA [21], Katz [20], TransE [9], Adamic & Adar [2], and Jaccard coefficient [24]. Most of these models rely on the traversal of the knowledge graph. For example, PRA [21] utilizes random walk, Knowledge Linker (KL) [11] employs the shortest path method, and PredPath [37] uses path enumeration.

In this chapter, we first review some preliminary knowledge of knowledge graphs and then introduce the three most recent and representative methods that use knowledge graphs for fact checking.

2 Preliminaries

While the idea of the "knowledge graph" can be traced to 1972 [34], the modern definition of the knowledge graph was first put forward by Google [42] in 2012. There are further developments of knowledge graphs by other companies, such as Facebook [30], LinkedIn [15], and Microsoft [40].

2.1 Knowledge Graph

A knowledge graph represents a data graph, which accumulates and transmits knowledge gathered from a real-world database [8]. The nodes of knowledge graphs denote entities, and each edge denotes the relationship between two entities. Most knowledge graphs are extracted from external knowledge bases containing numerous true statements. These statements can be divided into simple statements,

such as "Sacramento is the capital of California," and qualitative statements, such as "capitals are cities." Simple statements can serve as edges in knowledge graphs.

There are two types of knowledge graphs: open knowledge graph and enterprise knowledge graph. Open knowledge graph refers to one that is published online and is freely accessible. Some open knowledge graphs may accumulate data directly from Wikipedia (such as DBpedia [22] and YAGO2 [16]), while others use crowdsourcing methods to gather knowledge from volunteers collaboratively (such as Freebase [7] and Wikidata [46]). There are also some open knowledge graphs on specific topics, such as government [35], news [33], tourism [25], and geography [43]. The enterprise knowledge graph is mostly for internal use and/or commercial purposes. Based on applications, enterprise knowledge graphs can be classified into commerce (such as Uber[1] and eBay [2]), finance (such as Bloomberg[3] and Accenture[4]), social network (such as LinkedIn[5] and Facebook [30]), etc.

2.2 RDF

To allow the computer to better understand the information contained in statements, resource description framework (RDF) triples in the form of ⟨subject, predicate, object⟩ have been proposed [27]. Predicate illustrates the binary relationship between subject and object. For example, the statement "Sacramento is the capital of California" could be represented by an RDF triple ⟨Sacramento, CapitalOf, California⟩. RDF can build a labeled directed graph, where nodes denote entities (i.e., subject and object) and directed edges denote predicates. Different edge labels denote various predicates.

A formal definition of a knowledge graph constructed with RDF triples is as follows.

Definition 1 (Knowledge Graph) A knowledge graph is a directed graph $G = (V, E, \mathcal{R}, \mathcal{O}, g, h)$, where V denotes a node set, E denotes an edge set, $\mathcal{R}$ denotes the relation set, and $\mathcal{O}$ denotes the ontology set. g: $E \rightarrow \mathcal{R}$ is the labeling function, which maps edges to predicates, and h: $V \rightarrow \mathcal{O}$ is the function, which maps nodes to ontologies.

[1] https://eng.uber.com/uber-eats-query-understanding/.

[2] https://www.ebayinc.com/stories/news/cracking-the-code-on-conversationalcommerce/.

[3] https://speakerdeck.com/emeij/understanding-news-using-thebloomberg-knowledge-graph.

[4] https://www.accenture.com/us-en/insights/digital/data-to-knowledge.

[5] https://engineering.linkedin.com/blog/2016/10/building-the-linkedin-knowledge-graph.

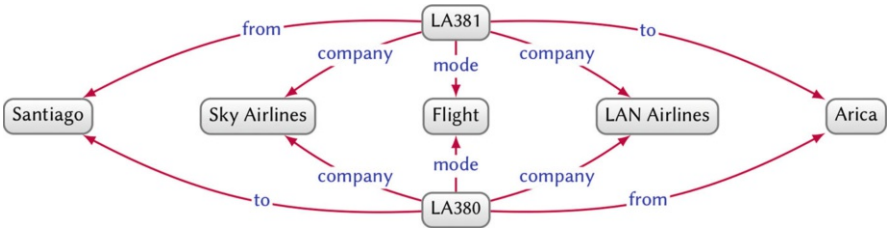

Fig. 1 A directed edge-labeled graph of companies that offer flights between Santiago and Arica [17]

Figure 1 shows an example of a knowledge graph constructed with triples.

3 Models

Quite a few models have been proposed to mine the knowledge graph for fact checking, including Knowledge Linker (KL) [11], PredPath [37], Knowledge Stream (KS) [39], PRA [21], Katz [20], TransE [9], Adamic & Adar [2], and Jaccard coefficient [24]. In this chapter, three models, namely, KL, PredPath, and KS, are introduced. KL utilizes the semantic proximity metrics for mining knowledge graphs. While it uncovers relationships among some nodes, it neglects the predicate between each pair of nodes. Sometimes, the results are difficult to interpret. PredPath employs the link prediction method and introduces a newly defined metric. KS models the fact-checking problem as an optimization problem and uses flow theory for solving the problem.

4 Knowledge Linker

The model Knowledge Linker (KL) [11] is based on the simple idea that fact checking on a knowledge graph aims to check whether the statement serves as an edge of the knowledge graph or if there exists a path to connect the target's subject and its object in the knowledge graph.

When checking a statement, it is seldom the case that a corresponding edge exists in the knowledge graph. Therefore, it is important to deduce the relation between the subject node and the object node by effectively mining the connectivity of the knowledge graph. KL adopts the epistemic closure theory [26]. The epistemic closure theory refers to a set of entities closed under logical implication, which means that a given statement could be deduced to be true through the entailment from what is already known. Generally, it can be regarded as a specific example of link prediction in knowledge graphs [29].

Semantic Proximity After establishing the link prediction method for fact checking in knowledge graphs, the next question is how to define the "path length" of different paths that connect the subject node and the object node. A path containing a lot of generic entities may sometimes provide weak information or even the wrong information. To illustrate, consider the following example.

The paths connecting entities "Sacramento" and "California" can be as follows:

. $\{\text{Sacramento}\} \xrightarrow{CityOf} \{\text{the United States}\} \xleftarrow{StateOf} \{\text{California}\}$

. $\{\text{Sacramento}\} \xleftarrow{Headquarter} \{\text{California State Police Department}\} \xrightarrow{Jurisdiction} \{\text{California}\}$

The entity "the United States" is a generic one, which means it can be related to many entities, thus providing little information. For any city in California or even in the United States, the first path could connect two entities, such as "Los Angeles" and "California" or "Chicago" and "California." Subsequently, the paths made in this way are of little value for checking the statement "Chicago is a city of California."

In the second path, however, two entities are connected to the middle entity, "California State Police Department." In addition, the entity "California State Police Department" has much fewer entities associated with it than the entity "the United States." Therefore, the second path depicts the special correlation information between these two entities. In fact, the statement "Chicago is a city of California" would be confirmed as a false statement with the second path.

From the example above, the length of a path can be defined by the generality of the nodes that comprise it. When a node is related to many nodes, such as "the United States," it has a higher generality score. There are three possible ways to illustrate whether two nodes are related:

1. If they are connected with the specific edge in the knowledge graph
2. If there exists a path connecting the two nodes in the knowledge graph
3. If the shortest path connecting the two nodes in the knowledge graph has a shorter length than the preset threshold

For the first way, the relation established contains less information and is inconsistent with the epistemic closure principle. The second and third ways both use intermediate nodes to establish relations. In addition, the third way takes into account the fact that the relevance decreases as the number of intermediate nodes increases, which seems to be more rational than the first way, but it is too computationally intensive to be practical and the threshold may be difficult to preset. Therefore, the second way is adopted in [41].

Since KL is based on the epistemic closure theory, when KL considers the relations, it does not care much about the predicate, and it models the knowledge graph as an undirected graph $G = (V, E)$, where V and E are the same as in Definition 1.

Definition 2 (Transitive Closure) $G = (V, E)$ is an undirected knowledge graph. Two nodes $a,\ b \in V$ are regarded as adjacent if there exists an edge $e = (a, b) \in E$. Two nodes $a,\ b \in V$ are regarded as connected if there is a sequence of nodes $(a = v_1, v_2, \ldots, v_n = b)$ connecting a and b $(n \geq 2)$. $G^* = (V, E^*)$ is the transitive closure of G. The node sets of G and G^* are the same. Two nodes in G^* are determined to be adjacent iff the two nodes are connected in G.

A statement in the form of RDF triple $c =< s, p, o >$, where s denotes a subject, o denotes an object, and p denotes a predicate, is extracted from the transitive closure G^* of an undirected knowledge graph G. A path connecting subject s and object o is denoted as $P_{s,o} = (s = v_1, v_2, \ldots, v_n = o)$. The length of the path $P_{s,o}$ is defined as follows:

$$\mathcal{L}\left(P_{s,o}\right) = \mathcal{L}\left(v_1 \ldots v_n\right) = \left[1 + \sum_{i=2}^{n-1} \log k\left(v_i\right)\right]^{-1},$$

where $k(v_i)$ represents the entity v_i's degree, which means the number of appearances of the statement in the knowledge graph. With the help of the degree, the generality of an entity in the knowledge graph is defined. If c truly exists as an edge connecting entity s and entity o in the knowledge graph, then the corresponding value surely should be assigned the maximum value, i.e., $\mathcal{L}(P_{s,o}) = 1$. The semantic proximity $\mathcal{L}$ would be assigned the value 1 *iff* $n = 2$ since there are no nodes between the subject and the object.

When considering an alternative principle *the widest bottleneck* of the optimization problem, the length of the path $P_{s,o}$ could be measured with a new method:

$$\mathcal{L}^{'}\left(P_{s,o}\right) = \mathcal{L}^{'}\left(v_1 \ldots v_n\right) = \begin{cases} 1 & n = 2 \\ \left[1 + \max_{i=2}^{n-1}\left\{\log k\left(v_i\right)\right\}\right]^{-1} & n > 2, \end{cases}$$

where the function $k(v_i)$ has the same definition as above.

Since there could be several paths between the subject and the object, the truth value of a statement $c =< s, p, o >$ could be measured by finding the shortest path between the subject s and the object o [5, 28]. Formally, it is defined as follows:

$$\tau(c) = \max \mathcal{L}\left(P_{s,o}\right) \left(or \max \mathcal{L}^{'}\left(P_{s,o}\right)\right).$$

Figure 2 shows an example of a path on the knowledge graph for a statement which has a low truth value.

Case Study Results *Wikipedia Knowledge Graph* (WKG) is built upon three datasets, namely, the DBpedia ontology dataset, the properties dataset, and the types dataset. The triples in the DBpedia ontology dataset all have the predicate "SubClassOf." The triples in the properties dataset are extracted from the Wikipedia

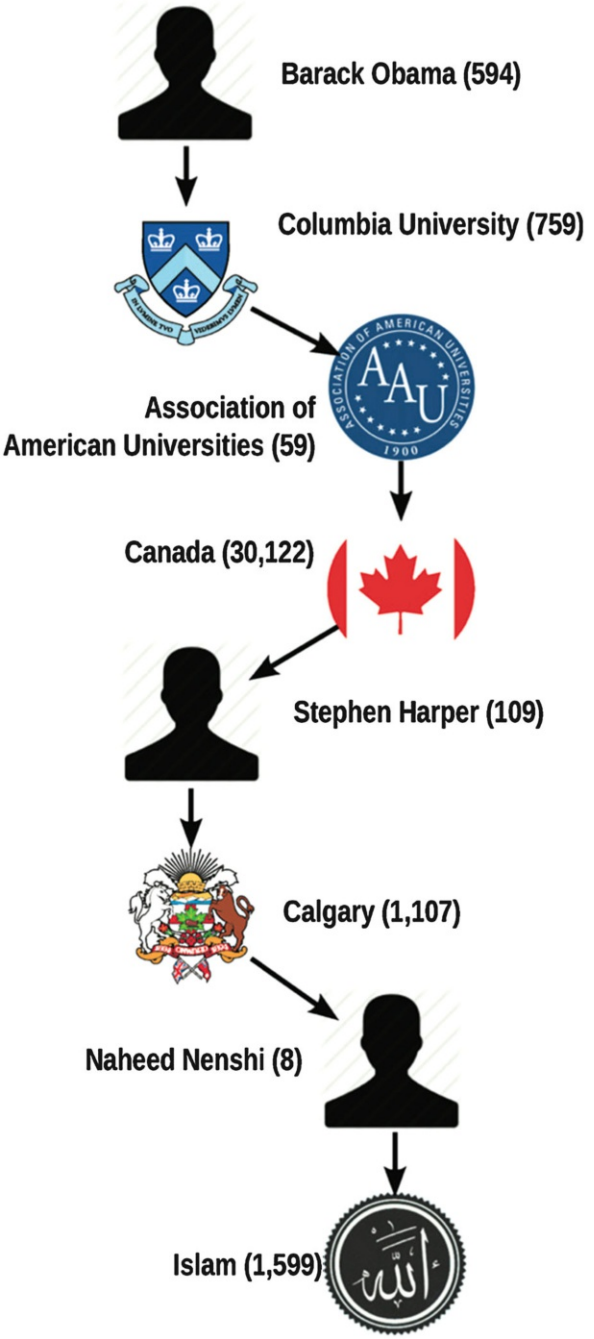

Fig. 2 The shortest path of statement "Barack Obama is a Muslim." Numbers beside nodes represent their degrees. This path traverses nodes with high degrees, i.e., generic entities, such as "Canada," and thus it is assigned a low value [11]

infoboxes.[6] The triples in the types dataset are all in the form of ⟨subject, is-a, Class⟩, and *Class* is derived from the DBpedia[7] ontology.

The experiment is to compute the truth values of different statements such as "a belongs to b," where a is a US Congress member and b is an ideology. A matrix $\mathcal{M} = \{v_{i,j}\}_{n \times m}$ is defined, where n rows represent n members of Congress and m columns represent m ideology nodes in the WKG.

The matrix is computed by the definition of $\mathcal{L}(.)$ with the help of a force-directed layout [19]. The paths connecting blue or red nodes with gray nodes shown in Fig. 3 are all ranked in the top 1% of the truth value. The results shown in Fig. 3 are very much consistent with the results derived from blogs [3] and Twitter [12].

5 PredPath

As discussed in Sect. 3.1, the fact-checking problem based on the knowledge graph can be translated into a link prediction problem. The model Predicate Path (PredPath) [37] (KL) takes the connectivity (i.e., the degree of correlation between the nodes in a knowledge graph) and type information (i.e., the ontologies of each node) into consideration. Specifically, KL mines the knowledge graph based upon not only the connectivity and type information but also the interactions of predicates. The model aims to extract a set of discriminative paths that could illustrate the correlation between two entities uniquely in the knowledge graph.

Note that there exist some association mining methods [1, 14] and link prediction methods [6] on the knowledge graph, but when applied in fact checking, these methods would have drawbacks where in the derived results are general and lack specificity. For example, consider the predicate *CapitalOf* between two entities. Both the link prediction methods and the association mining methods would return the result that the predicate *LargestCityOf* is most related to the predicate *CapitalOf*. To some extent, the predicate *LargestCityOf* can be an alternative to the predicate *CapitalOf*. For example, given the statement "Columbus is the capital of Ohio," Columbus is truly the largest city and capital of Ohio. However, because the statement "Los Angeles is the largest city of California" is true does not mean that the statement "Los Angeles is the capital of California" is true. In fact, California's capital is Sacramento. The PredPath model could derive a discriminative path for statements where cities are capitals of states. If the intermediate nodes in a path have the city's headquarters and own jurisdiction in the located state, we can say that it is equal to the predicate *CapitalOf*.

$$\text{. } \{\text{Columbus}\} \xrightarrow{LargestCityOf} \{\text{Ohio}\} \Longrightarrow \{\text{Columbus}\} \xrightarrow{CapitalOf} \{\text{Ohio}\}$$

[6] https://en.wikipedia.beta.wmflabs.org/wiki/Infobox.

[7] https://wiki.dbpedia.org/.

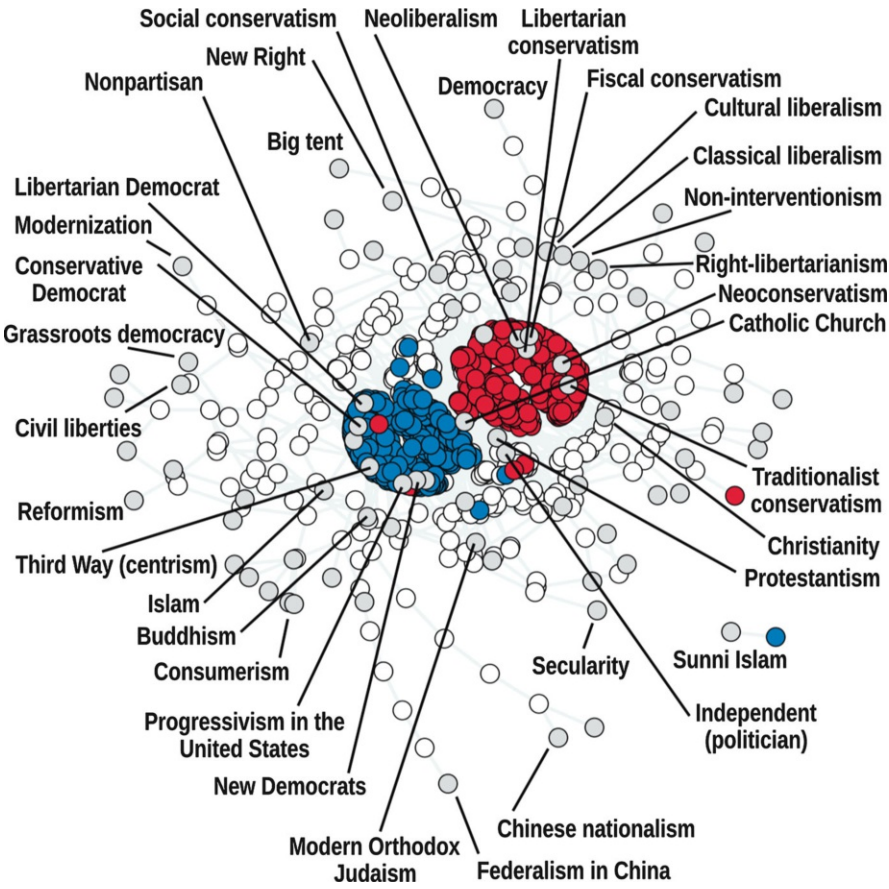

Fig. 3 Ideological map of Congress members [11]. The blue nodes represent the members of the Democratic Party and the red nodes represent the members of the Republican Party. The gray nodes denote the ideologies and the white nodes denote the intermediate nodes. The nodes' positions are calculated by a force-directed layout [10]. Only the most significant paths whose truth values rank in the top 1% are shown

. $\{\text{Los Angeles}\} \xrightarrow{LargestCityOf} \{\text{California}\} \not\Longrightarrow \{\text{Los Angeles}\} \xrightarrow{CapitalOf} \{\text{California}\}$

. $\{\text{Sacramento}\} \xleftarrow{Headquarter} \{\text{California State Police Department}\}$ or $\{\text{California State Department of Transportation}\}$ etc. $\xrightarrow{Jurisdiction} \{\text{California}\} \Longrightarrow \{\text{Sacramento}\} \xrightarrow{CapitalOf} \{\text{California}\}$

To determine a statement's truthfulness, PredPath mines the connectivity characteristics of a knowledge graph by employing the principles of network closure, similarity search, and link prediction. There exist approaches which are based

on meta paths and use the similarity property of paths in a knowledge graph. These approaches show brilliant results for solving problems such as clustering, recommendation, and classification [21, 38, 44, 45]. However, they all require users to know the domain of the problem in advance and the relevant meta paths before conducting the analysis [44]. PredPath can obtain a set of discriminative paths, which describe the relationship between two entities in a path of a knowledge graph uniquely.

Definitions Based on the previous definition of knowledge graph G, an entity in the knowledge graph can be mapped to multiple ontologies. When knowledge bases such as DBpedia[8] are used to build the knowledge graph, entities such as *Columbus*, *Los Angeles*, and *Sacramento* form the node set V; predicates such as *CapitalOf* and *LargestCityOf* form the predicate set $\mathcal{R}$; type labels such as *state* and *city* form the ontology set $\mathcal{O}$; and the edge set E represents the set of links each between two nodes in the knowledge graph G.

With the type information in the knowledge graph, an intensive set of connections named meta path to depict how the type labels can connect entities is defined as follows.

Definition 3 (Meta Path) A meta path $\mathcal{M}_n$ in the knowledge graph G is denoted as a typed, directed sequence of entities and edges: $o_1 \xrightarrow{p_1} o_2 \xrightarrow{p_2} \ldots \xrightarrow{p_{n-1}} o_n$, where o_i represents the ontology of entity e_i, p_i denotes the predicate that links entity e_i to e_{i+1}, and n represents the generalized length of the meta path.

To reduce storage and computational complexity, the intermediate type nodes in a meta path are ignored, but the predicates and endpoints are reserved. An anchored path vividly illustrates the structure of the path, which comprises the start node, the end node, and the predicates linking them.

Definition 4 (Anchored Path) The anchored path $\mathcal{A}_n$ of a meta path $\mathcal{M}_n$ is denoted as a directed path with typed sequences of edges and only the typed endpoints: $\mathcal{A}_n = o_1 \xrightarrow{p_1}\xrightarrow{p_2} \ldots \xrightarrow{p_{n-1}} o_n$.

Discriminative predicate paths are targets of the PredPath model, which are defined as follows.

Definition 5 (Discriminative Predicate Path) We use $D_n(o_u, o_v)$ to represent the set of discriminative predicate paths. It consists of all those anchored paths which could express the given statement $o_u \xrightarrow{p} o_v$ alternatively, and the paths' maximum generalized length is k.

Consider an example for illustration. One meta path that links the two entities "Sacramento" and "California" could be $\mathcal{M}$: {city} $\xleftarrow{Headquarter^{-1}}$ {state

[8] See footnote 7.

agency} $\xrightarrow{Jurisdiction}$ {state}. The anchored path $\mathcal{A}$ anchored by {city} and {state} for this meta path is $\langle$ Headquarter^{-1}, Jurisdiction$\rangle$. The corresponding discriminative predicate path set comprises many other anchored predicate paths connecting {city} and {state}.

The meta paths tend to own more type label information and can thus be more prone to involve labeling error. Therefore, anchored paths are used since they are more tolerant of labeling error.

PredPath solves the fact-checking problem by performing a supervised link prediction task, i.e., it determines a statement triplet $c =< s, p, o >$ to be true or not by first computing the discriminative path set $D_k(o_u, o_v)$, where the subject s's and the object o's ontologies are o_u and o_v, respectively, and then checking whether the edge $s \xrightarrow{p} o$ can be implied in the knowledge graph G. If $p \in \mathcal{R}$, then the positive path set H^+ and the negative path set H^- can be generated as the node pair sets, where $H^+ = \{(u, v)|u \xrightarrow{p} v \in G\}$, $H^- = \{(u, v)|u \xrightarrow{p} v \notin G\}$, $o_u = o_s$, and $o_v = o_o$. When $p \notin \mathcal{R}$, H^+ and H^- ought to be provided by humans.

PredPath considers both the generality and the context dependency of paths for discovering the most discriminative paths. Generality means whether the entities connected by the predicate p are of the same or similar type. The context dependency represents the similarity of different paths, which link the entities of the same or similar type.

Path Extraction Most existing meta path-based models need hand annotation [38] or exhaustive enumeration [21] when extracting the paths from knowledge graphs. In contrast, PredPath can extract the paths automatically, employing a constrained graph traversal algorithm. Though the amount of data in a knowledge graph can be massive, only a small part of the data is truly useful for the given task. Among the extracted meta paths, there are only a few discriminative paths for a certain predicate. When it checks the fact "Sacramento is the capital of California," it only considers those meta paths which start from the ontology city and end at the ontology state. A constrained graph traversal algorithm extracts anchored paths by traversing the graph from the subject entity to the object entity with the length less than k instead of traversing all the possible paths.

The anchored path sets $\mathbf{A}^+{}_{(o_u,o_v)}$ and $\mathbf{A}^-{}_{(o_u,o_v)}$ are extracted separately by using the depth-first traversal algorithm. This algorithm is implemented with the help of a closure function $\mathbb{C}$:

$$\mathbb{C}_p(v) = \left\{v' | \left(v, p, v'\right) \in G\right\} \cup \left\{v' | \left(v', p, v\right) \in G\right\},$$

where v denotes an entity in a knowledge graph and p represents the predicate related to the closure. The function $\mathbb{C}$ returns all the entities that can be reached by v from predicate p or p^{-1}. With the definition of closure function, a transition function $\mathbb{T}(v_i)$ could be defined, which returns all the next nodes v_{i+1} of entity v_i.

$\mathbb{T}(v)$ returns all the entities that can be reached from $\mathbb{C}_p(v)$ without those that have already been visited:

$$\mathbb{T}\left(v_i\right) = \left\{\cup_{p\in\mathcal{R}}\mathbb{C}_p\left(v_i\right) \setminus \cup_{j=1}^{i}\left\{v_j\right\}\right\}.$$

With the definitions of functions $\mathbb{C}_p(v)$ and $\mathbb{T}(v_i)$, the path set $\mathbb{P}$ could be derived with all the paths whose lengths are less than n: $\mathbb{P} = \cup_{i=1}^{n}\mathbb{P}^n$, where

$$\begin{aligned}\mathbb{P}^n = \{&s, \mathbb{T}\left(v_1\right), \mathbb{T}\left(v_2\right), \ldots, \mathbb{T}\left(v_{n-2}\right), o| \\ &(s,o) \in \mathbf{T}, v_1 = s, v_i \in \mathbb{T}\left(v_{i-1}\right), o \in \mathbb{T}\left(v_{n-1}\right)\}.\end{aligned}$$

The next issue is how to measure the importance/helpfulness of a path. This problem is tackled with a regression model.

Path Selection Given the predicate path sets P^+ and P^-, the aim is to select the most discriminative predicate path set $\mathbf{D}$. The training matrix is defined as X, where the i-th row of matrix $X_{n\times m}$ denotes an instance anchored by u and v such that $o_u = o_s$ and $o_v = o_o$. Every member $X_{i,j}$ of the matrix X represents the number of anchored paths P_j anchored by u and v.

The goal for the path selection lies in deriving a new matrix $X'_{n\times m'}$, where the columns for the new matrix X' only contain the most discriminative paths' power:

$$\mathbf{X}' = f(\mathbf{X}, \mathbf{w}, \delta) = \mathbf{X}_{1:n,\{j|j\in 1:m, w_j\geq\delta\}},$$

where $\mathbf{w}$ is a feature importance vector with m dimensions and δ represents a threshold that controls importance.

The element $w_j \in \mathbf{w}$ is the important vector of an anchored predicate path $P_j \in \mathbf{P}$, which is defined by the information gain of $X_{:,j}$ and y:

$$I\left(\mathbf{X}_{:j} : \mathbf{y}\right) = \sum_{x_{i,j}\in\mathbf{X}_{i,j}}\sum_{y_i\in\mathbf{y}} p\left(x_{i,j}\right) p\left(y_i\right) \log\left(\frac{p\left(x_{i,j}, y_i\right)}{p\left(x_{i,j}\right) p\left(y_i\right)}\right),$$

where y is the label vector for the feature vector $X_{:,j}$, $x_{i,j}$ denotes the value of element $X_{i,j}$ [32], and the threshold δ is set empirically.

With the definition of matrix X', the validation of the statement of a fact can be solved by a logistic regression model [36].

Fact Interpretation Not all paths are intuitive enough to describe important information. For example, the statement of fact ⟨Sacramento, CapitalOf, California⟩ may generate some meaningless predicate paths such as ⟨location^{-1},location⟩ and ⟨deathPlace^{-1},deathPlace⟩, which represent the statements that "a capital' s location is in the state" and that "City is place of death for a person who died in the state." In this example, the paths generated do not provide much information related to

"CapitalOf." Therefore, it is necessary to select those vital discriminative predicate paths that only depict the predicate in question.

This could be done by sorting out the predicate paths with the importance vector $\mathbf{w}$: $P_i \prec P_j$ if and only if $w_i \geq w_j$. After ranking the predicate paths, we can remove those unimportant and off-topic predicate paths:

$$\mathbf{D}^* = \left\{ P \,\middle|\, P \in \mathbf{D} \backslash \left\{ P_j | P_j \in \mathbf{P}^-, \sum_{i=0}^{i=n} \mathbf{X}_{i,j} \geq \theta \right\} \right\},$$

where θ represents the threshold, which is chosen empirically between 10 and 20. The function introduced above is able to select a discriminative path set $\mathbf{D}^*$, which contains the paths that can specifically define the predicate provided. The discriminative predicate paths in the top 5 for the predicate "CapitalOf" are listed in Table 1.

Comparison Between Meta Path and Predicate Path Different from those existing studies which use meta paths on heterogeneous networks, PredPath uses the anchored predicate paths. As a knowledge graph can be much more complicated, an entity in the knowledge graph can own multiple labels. For example, Boston's type label is {city, settlement, populated place}, and Sacramento's type label is {settlement, populated place}, though they are, respectively, the capital of Massachusetts and California. Because the type labels do not match exactly, the

Table 1 Top discriminative paths for "CapitalOf"

Rank	**Meta Path** $\mathcal{M}$
1	{city, settlement} $\xrightarrow{location^{-1}}$ {state agency} $\xrightarrow{location}$ {state}
2	{city, settlement} $\xrightarrow{deathPlace^{-1}}$ {person} $\xrightarrow{deathPlace}$ {state}
3	{city, settlement} $\xrightarrow{headquarter^{-1}}$ {state agency} $\xrightarrow{jurisdiction}$ {state}
4	{city, settlement} $\xrightarrow{location^{-1}}$ {state agency} $\xrightarrow{jurisdiction}$ {state}
5	{settlement} $\xrightarrow{location^{-1}}$ {state agency} $\xrightarrow{jurisdiction}$ {state}
	Anchored Path D
1	⟨headquarter^{-1}, jurisdiction⟩
2	⟨location^{-1}, jurisdiction⟩
3	⟨headquarter^{-1}, regionServed⟩
4	⟨garrison^{-1}, country⟩
5	⟨deathPlace^{-1}, deathPlace⟩
	Discriminative Anchored Path D[8]
1	⟨headquarter^{-1}, jurisdiction⟩
2	⟨location^{-1}, jurisdiction⟩
3	⟨garrison^{-1}, country⟩
4	⟨headquarter^{-1}, parentOrganisation⟩
5	⟨location^{-1}, parentOrganisation⟩

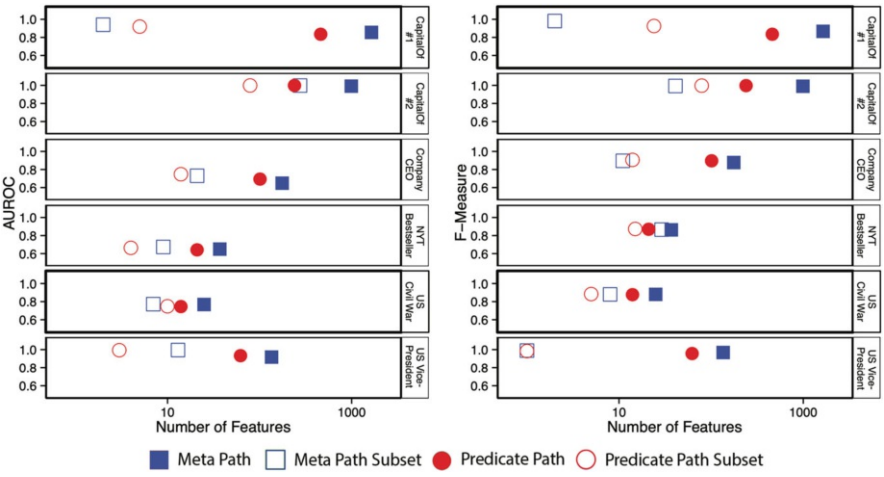

Fig. 4 Comparison experiment between Meta Path and Predicate Path [37]

PredPath model will treat the two paths differently, which could result in high overlap.

As shown in Fig. 4, the performance of four models for fact checking on DBpedia,[9] namely, Meta Path, Meta Path Subset, Predicate Path, and Predicate Path Subset, is shown. Meta Path Subset and Predicate Path Subset both represent the paths selected by the function mentioned above.

According to these results, the predicate path performs almost as accurately or even better than the meta path, though it has fewer features with entities removed. The subset selected by the importance selection function performs better than the original set.

6 Knowledge Stream

Knowledge Stream (KS) [39] is based on a novel and unsupervised network flow framework for fact checking. The model measures the trustworthiness of a statement in the form of a RDF triple. For the problem of fact checking on knowledge graphs, many approaches involve some traversal on knowledge graphs. Knowledge Linker utilizes the shortest path algorithm, and PredPath utilizes the path enumeration algorithm. KS is based on the fact that the information carried by multiple paths can provide more semantic context information than a single path as the non-disjoint paths may send additional flow on the knowledge graph. The KS model can

[9] See footnote 7.

automatically extract the meaningful patterns and contextual facts with a broader structure.

As shown in Fig. 5, the paths drawn in different colors form a stream of knowledge for the RDF triple ⟨David and Goliath, WrittenBy, Malcolm Gladwell⟩. To visually represent the flow of information on the knowledge graph, each path has been assigned a different width based on the amount of evidence it can offer for the RDF triple. KS would assign larger flows to those paths that provide more and discriminative information.

KS can be vividly interpreted as a network flow model. Given an RDF triple (s, p, o), it could be regarded as the knowledge flow starting from the subject entity s through the network and ending up at the object entity o. The remaining issue is to quantify the capacity and cost for each edge in the network. The capacity quantifies the amount of knowledge or information carried related to statement (s, p, o). The cost can be regarded as a constraint for the knowledge to pass a certain edge, which ensures that the paths extracted by KS are short. KS aims to extract the set of paths which can provide maximum flow of knowledge and minimize the cost.

The capacity of each edge $e' \in E$ in a knowledge graph can be intrinsic. With the definition above, the edge e' will be mapped to a certain predicate p'. For the statement (s, p, o) to be checked, the capacity of each edge in the knowledge graph is quantified as the relevance or similarity between the target predicate p and p' in the knowledge graph. The more relevant or similar p is p', and a higher capacity can be assigned to edge e'. It then measures the capacity of each path by the minimum capacity of all the edges on the path, i.e., the bottleneck [4]. The bottleneck can be interpreted as the least relevant or similar triple to the target statement in the path. Since there could be many paths connecting subject entity s and object entity o, the sum of their bottlenecks corresponds to the upper bound of knowledge flow through these paths. For KS, the path length is defined by not only the number of entities in the path but also the degrees of the connections from entities to other entities in the graph [11].

Relational Similarity Different from the models Knowledge Linker and PredPath, Knowledge Stream treats the knowledge graph as an undirected graph, only considering whether two entities are connected. The line graph $L(G) = (V', E')$ is defined on the undirected graph $G = (V, E)$. The node set V' of $L(G)$ is defined as $V' = E$, in which two new nodes are adjacent if and only if the corresponding edges in set E are connected by the same node in G, i.e., $E' = \{(e_1, e_2) | e_1, e_2 \in E, e_1 \cap e_2 \neq \phi\}$. With the definition of line graph, the edge-labeled graph G could be transformed into a node-labeled graph $L(G)$. In addition, a contracted line graph $L^*(G)$ could be defined, which is an edge-weighted graph that substitutes two nodes with a new node if the new nodes' set of neighbors corresponds to the union of the sets of the two nodes' neighbors. For illustration, an example is shown in Fig. 6.

An adjacency matrix $C \in \mathbb{N}^{R \times R}$ is defined for the contracted line graph $L^*(G)$, where $R = |\mathcal{R}|$. Matrix C is defined as the co-occurrence matrix of $\mathcal{R}$. The similarity between two relationships is measured by computing the cosine value between two corresponding rows of vectors in C. Similar to information retrieval,

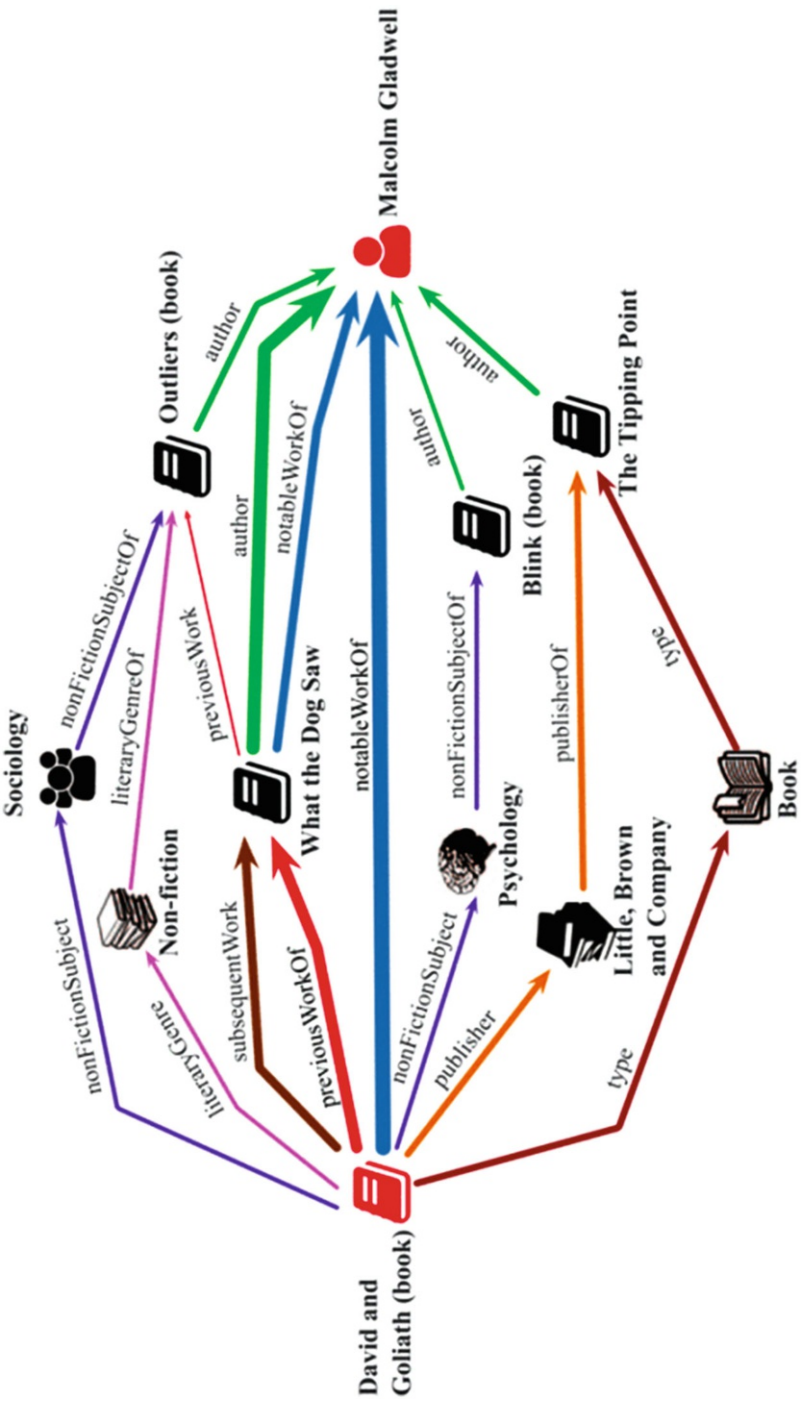

Fig. 5 An example of the paths selected by KS [39]

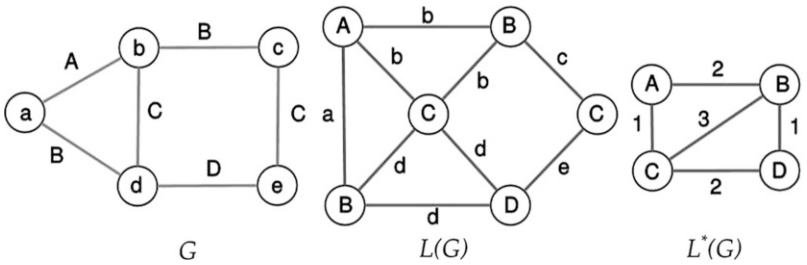

Fig. 6 Example of line graph and contracted line graph [39]

IF and IDF terms could be defined based on matrix C:

$$\begin{aligned}
\mathrm{TF}\left(r_i, r_j\right) &= \log\left(1 + C_{ij}\right), \\
\mathrm{IDF}\left(r_j, \mathcal{R}\right) &= \log \frac{R}{\left|\left\{r_i \mid C_{ij} > 0\right\}\right|}, \\
C'\left(r_i, r_j, \mathcal{R}\right) &= \mathrm{TF}\left(r_i, r_j\right) \cdot \mathrm{IDF}\left(r_j, \mathcal{R}\right),
\end{aligned}$$

where $C_{i,j}$ denotes the count of co-occurrences between $r_i, r_j \in \mathcal{R}$, as discussed before. Then, the relational similarity $u(r_i, r_j)$ is computed as the cosine value between i-th and j-th rows of C'.

Fact Checking as a Network Flow Problem As discussed before, the fact-checking problem can be viewed as a problem of finding an optimal approach to transfering the knowledge across the knowledge graph under certain constraints, which could be modeled as a minimum cost maximum flow problem.

The next question lies in how to specifically utilize the knowledge stream for fact checking. As the long chain path may lead to a general or obvious result, we need to define the specificity of a path $P_{s,p,o}$. The specificity $S(P_{s,p,o})$ is defined proportionally to the inverse of the sum of degrees:

$$\mathcal{S}\left(P_{s,p,o}\right) = \frac{1}{1 + \sum_{i=2}^{n-1} \log k\left(v_i\right)}.$$

Combined with the definitions above, the net flow $\mathcal{W}(P_{s,p,o})$ of a path $P_{s,p,o}$ can be set as the product of its bottleneck and specificity:

$$\mathcal{W}\left(P_{s,p,o}\right) = \beta\left(P_{s,p,o}\right) \cdot \mathcal{S}\left(P_{s,p,o}\right).$$

To check whether the statement triple (s, p, o) is true or not, KS derives the truth score $\tau^{KS}(s, p, o)$ of the triple by summing all the paths' flow in the net together:

$$\begin{aligned}\tau^{\mathrm{KS}}(s, p, o) &= \sum_{P_{s,p,o} \in \mathcal{P}_{s,p,o}} \mathcal{W}\left(P_{s,p,o}\right) \\ &= \sum_{P_{s,p,o} \in \mathcal{P}_{s,p,o}} \beta\left(P_{s,p,o}\right) \cdot \mathcal{S}\left(P_{s,p,o}\right).\end{aligned}$$

7 Conclusion and Future Work

In this chapter, three methods, namely, Knowledge Linker (KL), Predicate Path (PredPath), and Knowledge Stream (KS), which utilize knowledge graphs for fact checking, are introduced. There are quite a few future research directions. One possible future direction is to introduce deep learning models such as GNN into the fact-checking problem on knowledge graphs given that deep learning has been used successfully for many complex problems such as those in computer vision, natural language processing, and control. In addition, it seems necessary to bring the temporal dimension into consideration for fact checking. Take the capital of the Roman Empire, for example; the statement "Roman Empire's capital is Rome" is correct only before 323 CE, because the capital was later changed to Constantinople.

References

1. Abedjan, Z., Naumann, F.: Synonym analysis for predicate expansion. In: Extended Semantic Web Conference, pp. 140–154. Springer, New York, (2013)
2. Adamic, L.A., Adar, E.: Friends and neighbors on the web. Soc. Netw. **25**(3), 211–230 (2003)
3. Adamic, L.A., Glance, N.: The political blogosphere and the 2004 US election: divided they blog. In: Proceedings of the 3rd International Workshop on Link Discovery, pp. 36–43 (2005)
4. Ahuja, R.K., Magnanti, T.L., Orlin, J.B., Weihe, K.: Network flows: theory, algorithms and applications. ZOR-Methods Models Oper. Res. **41**(3), 252–254 (1995)
5. Aiello, L.M., Barrat, A., Schifanella, R., Cattuto, C., Markines, B., Menczer, F.: Friendship prediction and homophily in social media. ACM Trans. Web **6**(2), 1–33 (2012)
6. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
7. Bollacker, K., Tufts, P., Pierce, T., Cook, R.: A platform for scalable, collaborative, structured information integration. In: International Workshop on Information Integration on the Web (IIWeb'07), pp. 22–27 (2007)
8. Bonatti, P.A., Decker, S., Polleres, A., Presutti, V.: Knowledge graphs: new directions for knowledge representation on the semantic web (dagstuhl seminar 18371). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2019)
9. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, pp. 2787–2795 (2013)

10. Cheong, S.H., Si, Y.W.: Force-directed algorithms for schematic drawings and placement: a survey. Inf. Visual. **19**(1), 65–91 (2020)
11. Ciampaglia, G.L., Shiralkar, P., Rocha, L.M., Bollen, J., Menczer, F., Flammini, A.: Computational fact checking from knowledge networks. PloS ONE **10**(6), e0128193 (2015)
12. Conover, M.D., Ratkiewicz, J., Francisco, M., Gonçalves, B., Menczer, F., Flammini, A.: Political polarization on twitter. In: Fifth International AAAI Conference on Weblogs and Social Media (2011)
13. Flanagin, A.J., Metzger, M.J.: Perceptions of internet information credibility. Journal. Mass Commun. Quart. **77**(3), 515–540 (2000)
14. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 413–422 (2013)
15. He, Q., Chen, B., Argawal, D.: Building the LinkedIn knowledge graph. In: LinkedIn (2016)
16. Hoffart, J., Suchanek, F.M., Berberich, K., Lewis-Kelham, E., De Melo, G., Weikum, G.: Yago2: exploring and querying world knowledge in time, space, context, and many languages. In: Proceedings of the 20th International Conference Companion on World Wide Web, pp. 229–232 (2011)
17. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Gayo, J.E.L., Kirrane, S., Neumaier, S., Polleres, A., et al.: Knowledge graphs (2020). Preprint. arXiv:2003.02320
18. Howell, L., et al.: Digital wildfires in a hyperconnected world. WEF Rep. **3**(2013), 15–94 (2013)
19. Kamada, T., Kawai, S., et al.: An algorithm for drawing general undirected graphs. Inf. Process. Lett. **31**(1), 7–15 (1989)
20. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
21. Lao, N., Cohen, W.W.: Relational retrieval using a combination of path-constrained random walks. Mach. Learn. **81**(1), 53–67 (2010)
22. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. Seman. Web **6**(2), 167–195 (2015)
23. Lewandowsky, S., Ecker, U.K., Seifert, C.M., Schwarz, N., Cook, J.: Misinformation and its correction: continued influence and successful debiasing. Psychol. Sci. Publ. Int. **13**(3), 106–131 (2012)
24. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks. J. Am. Soc. Inf. Sci. Technol. **58**(7), 1019–1031 (2007)
25. Lu, C., Laublet, P., Stankovic, M.: Travel attractions recommendation with knowledge graphs. In: European Knowledge Acquisition Workshop, pp. 416–431. Springer, New York (2016)
26. Luper, S.: The epistemic closure principle (2008)
27. Manola, F., Miller, E., McBride, B., et al.: RDF Primer. W3C Recommend. **10**(1–107), 6 (2004)
28. Markines, B., Menczer, F.: A scalable, collaborative similarity measure for social annotation systems. In: Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, pp. 347–348 (2009)
29. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proceedings of the IEEE **104**(1), 11–33 (2015)
30. Noy, N., Gao, Y., Jain, A., Narayanan, A., Patterson, A., Taylor, J.: Industry-scale knowledge graphs: lessons and challenges. Queue **17**(2), 48–75 (2019)
31. Paulheim, H.: Knowledge graph refinement: a survey of approaches and evaluation methods. Seman. Web **8**(3), 489–508 (2017)
32. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc., Burlington (1992)
33. Raimond, Y., Ferne, T., Smethurst, M., Adams, G.: The BBC world service archive prototype. J. Web Semant. **27**, 2–9 (2014)

34. Schneider, E.W.: Course modularization applied: the interface system and its implications for sequence control and data analysis (1973)
35. Shadbolt, N., O'Hara, K.: Linked data in government. IEEE Intern. Comput. **17**(4), 72–77 (2013)
36. Shewhart, W.A., Wilks, S.S.: Applied Logistic Regression, 2nd edn. Wiley, New York (2005)
37. Shi, B., Weninger, T.: Discriminative predicate path mining for fact checking in knowledge graphs. Knowl.-Based Syst. **104**, 123–133 (2016)
38. Shi, C., Kong, X., Huang, Y., Philip, S.Y., Wu, B.: Hetesim: a general framework for relevance measure in heterogeneous networks. IEEE Trans. Knowl. Data Eng. **26**(10), 2479–2492 (2014)
39. Shiralkar, P., Flammini, A., Menczer, F., Ciampaglia, G.L.: Finding streams in knowledge graphs to support fact checking. In: 2017 IEEE International Conference on Data Mining (ICDM), pp. 859–864. IEEE, New York (2017)
40. Shrivastava, S.: Bring rich knowledge of people, places, things and local businesses to your apps. Bing blogs (2017)
41. Simas, T., Rocha, L.M.: Distance closures on complex networks. Netw. Sci. **3**(2), 227–268 (2015)
42. Singhal, A.: Introducing the knowledge graph: things, not strings. Official google blog **16** (2012)
43. Stadler, C., Lehmann, J., Höffner, K., Auer, S.: Linkedgeodata: a core for a web of spatial open data. Seman. Web **3**(4), 333–354 (2012)
44. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: PathSim: meta path-based top-k similarity search in heterogeneous information networks. Proc. VLDB Endowm. **4**(11), 992–1003 (2011)
45. Sun, Y., Norick, B., Han, J., Yan, X., Yu, P.S., Yu, X.: Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. ACM Trans. Knowl. Discov. Data **7**(3), 1–23 (2013)
46. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
47. Zubiaga, A., Aker, A., Bontcheva, K., Liakata, M., Procter, R.: Detection and resolution of rumours in social media: a survey. ACM Comput. Surv. **51**(2), 1–36 (2018)