Finding Streams in Knowledge Graphs to Support Fact Checking

Prashant Shiralkar*, Alessandro Flammini*†, Filippo Menczer*†, Giovanni Luca Ciampaglia†
*Center for Complex Networks and Systems Research, School of Informatics and Computing
†Network Science Institute
Indiana University, Bloomington (USA)

***Abstract*—The volume of information generated online makes it impossible to manually fact-check all claims. Computational approaches for fact checking may be the key to help mitigate the risks of massive misinformation spread. Such approaches can be designed to not only be scalable and effective at assessing veracity of dubious claims, but also to boost a human fact checker's productivity by surfacing relevant facts and patterns to aid their analysis. We present a novel, unsupervised network-flow based approach to determine the truthfulness of a statement of fact expressed in the form of a triple. We view a knowledge graph of background information about real-world entities as a flow network, and show that computational fact checking then amounts to finding a "knowledge stream" connecting the subject and object of the triple. Evaluation on a range of real-world and hand-crafted datasets of facts reveals that this network-flow model can be very effective in discerning true statements from false ones, outperforming existing algorithms on many test cases. Moreover, the model is expressive in its ability to automatically discover several useful patterns and surface relevant facts that may help a human fact checker.**

***Index Terms*—Knowledge Stream, Fact Checking, Knowledge Graph, Network Flow, Minimum Cost Maximum Flow**

I. Introduction

Misinformation, rumors, hoaxes, and lies have become rampant primarily due to the ability to quickly disseminate information through the Web and social media. This phenomenon poses a severe threat to society at large [1]. Numerous approaches have been designed to study and mitigate the effects of misinformation spread (see Zubiaga *et al.* [2]). Most strategies rely on contextual indicators of rumors (e.g., temporal patterns or source credibility) for detection of false claims. Ideally, a fact-checking system would assess the truthfulness of claims by reasoning about their content and related facts, and operate in near real time to match the rate at which misleading claims are made.

With advances in information extraction and semantic web standards, structured knowledge has become available in the form of knowledge graphs (KGs). Nodes in a KG represent entities, and edges correspond to facts about them, as specified by semantic predicates, or relations. A wide class of empirical facts can be thus represented by a triple (s, p, o), where the subject entity s is related to the object entity o by the predicate relation p. For example, `(Joe, spouse, Jane)` indicates that Jane is the spouse of Joe. DBpedia [3] and Wikidata [4] are examples of publicly available KGs that contain vast amounts of high-quality knowledge, and thus could be harnessed by fact-checking agents.

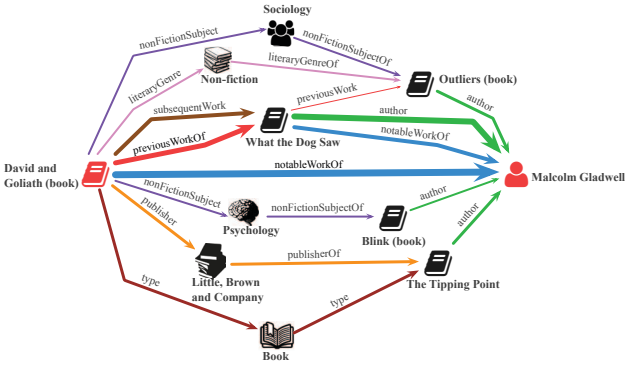

Fig. 1. Best paths identified by Knowledge Stream for the triple `(David and Goliath (book), author, Malcolm Gladwell)`. Edge width is roughly proportional to knowledge flow.

How can we automatically assess the truthfulness of a triple, given prior knowledge in a KG? We find some related problems and approaches in the literature [5]. These include statistical learning models (e.g., RESCAL [5], TransE [6]), which can be hard to interpret and suffer from scalability. AMIE [7]), PRA [8], and PredPath [9] mine first-order Horn clauses and are thus easy to interpret. They mine relation paths in the KG, which are used as features in a supervised learning framework. These approaches spend significant computational resources on feature generation and selection. Nevertheless, they have been shown to be very effective on fact-checking test cases [9]. They also offer some interpretability due to the features they learn. Link prediction algorithms can also be applied to fact checking [10]–[12].

None of these approaches offers all the qualities of a desirable fact-checking system: accuracy, interpretability, simplicity, scalability. Ciampaglia *et al.* [13] proposed Knowledge Linker (KL), which relies on finding short, specific paths in the KG. Here, we propose Knowledge Stream (KS), an unsupervised approach for fact-checking triples that extends this algorithm in two ways: it accounts for the semantics of target predicates, and uses a flow network to consider multiple paths. Our approach not only delivers performance comparable to state-of-the-art methods; it also produces more meaningful

2374-8486/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDM.2017.105

explanations of its predictions in the form of relation paths. The model is conceptually simple, intuitive, and uses the broader structural and semantic context of the triple.

As an example, Fig. 1 shows the paths computed for a true fact `(David and Goliath (book), author, Malcolm Gladwell)`. We call this set of paths a "stream" of knowledge that can be seen as the best form of evidence in support of the triple that the KG is able to offer. One can note from Fig. 1 that some paths give more evidence than others (wider edges in the figure). For example, the fact that Malcolm Gladwell is the author of the book *What the Dog Saw*, which followed *David and Goliath*, provides stronger evidence than the fact that another book authored by Gladwell, *The Tipping Point*, was published by the same company (Little, Brown and Company) as *David and Goliath*. KS correctly assigns a larger flow to the former path than the latter.

For a given triple (s, p, o), we view knowledge as an abstract commodity to be moved from the subject entity s to the object entity o across the network. Each edge is associated with a *capacity* to carry knowledge related to (s, p, o) across its two endpoints, and a *cost* of usage. We want to identify the set of paths responsible for the maximum flow of knowledge between s and o at the minimum cost. We will define the capacity of an edge based on the similarity between the predicate of the edge and the predicate of the triple under consideration. We use a data-driven approach, mining the structure of the KG itself, to define the similarity between predicates (Section II-A). The cost of an edge will be constructed to penalize paths that go through very generic concept entities [13] and paths that are too long [8], [9].

The method to compute similarity between relations can also be applied to shortest-path approaches such as Knowledge Linker. We propose such an algorithm, "Relational Knowledge Linker" (KL-REL), which verifies a claim based on the single shortest, semantically related path in the KG.

After we present KS and KL-REL in the next section, we show that they offer high interpretability and performance comparable to the state of the art.

II. Methods

In this section we outline Knowledge Stream and Relational Knowledge Linker. Further details on the methods can be found in an extended technical report [14]. Formally, a KG is a directed graph $G = (V, E, \mathcal{R}, g)$, where V, E, and $\mathcal{R}$ denote the node, edge, and relation sets, respectively, and $g : E \to \mathcal{R}$ is a function labeling each edge with a predicate. Although G is a directed network, we follow the convention of treating G as undirected by discarding the directionality of edges.

Both methods presented here rely on the ability to gauge the similarity of any pair of elements of $\mathcal{R}$. Next we explain our data-driven approach to relational similarity.

A. Relational Similarity via the Line Graph of a KG

The *line graph* (a.k.a *dual graph*) $L(G) = (V', E')$ of an undirected graph $G = (V, E)$ is the graph whose node set is $V' = E$ and in which two nodes are adjacent *iff* the corresponding edges of G are incident on the same node in G, that is, $E' = \{(e_1, e_2) : e_1, e_2 \in E \wedge e_1 \cap e_2 \neq \emptyset\}$. The edge labels of G become node labels of $L(G)$. We contract duplicate nodes until there is exactly one node for each element of $\mathcal{R}$. A graph can be *contracted* by replacing two nodes with a new node whose set of neighbors is the union of their neighbors. The contracted graph is edge-weighted; the weight of a new edge reflects the number of old edges that are merged in the contraction. We thus start from G, then build $L(G)$ setting all edge weights to 1, and finally iteratively contract pairs of nodes labeled with the same relation, until there are no duplicate labels. We call the resulting graph the *contracted line graph*, denoted by $L^*(G)$. See Fig. 2 for an example of a small KG with four relations and five nodes.

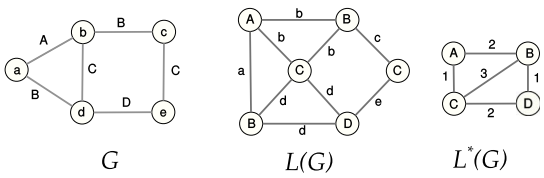

Fig. 2. Example of the line graph $L(G)$ and the contracted line graph $L^*(G)$ of a simple knowledge graph G with four relations (denoted by uppercase letters) and five nodes (lowercase letters). The edge weights in $L^*(G)$ represent how often each relation is co-incident to its neighbors in G.

(a) spouse (b) education

Fig. 3. Top 20 similar relations for two DBpedia predicates (more examples in our technical report [14]). Font size is proportional to similarity.

Let us denote with $C \in \mathbb{N}^{R \times R}$, where $R = |\mathcal{R}|$, the adjacency matrix of the contracted line graph. By construction, C is the co-occurrence matrix of $\mathcal{R}$. The raw co-occurrence counts in C are dominated by the most common relationships, therefore, we apply TF-IDF weighting:

$$C'(r_i, r_j, \mathcal{R}) = \log(1 + C_{ij}) \cdot \log \frac{R}{|\{r_i | C_{ij} > 0\}|} \quad (1)$$

where C_{ij} is the co-occurrence count between $r_i \in \mathcal{R}$ and $r_j \in \mathcal{R}$. We define the *relational similarity* $u(r_i, r_j)$ as the cosine similarity of the i-th and j-th rows of C'. This approach yields meaningful results (Fig. 3).

B. Fact checking as a Minimum Cost Maximum Flow Problem

In Knowledge Stream, fact checking corresponds to finding an optimal flow to transfer knowledge from the source entity to the target entity under a set of constraints. The constraints on the edges dictate that the amount of flow across an edge $e = (v_i, v_j) \in E$ is bounded by its *capacity*:

$$\mathcal{U}_{s,p,o}(e) = \frac{u\,(g(e),\, p)}{1 + \log k(v_j)}, \quad (2)$$

which is the product of the similarity u between the edge label $g(e)$ and the predicate p of the target triple (see Section II-A), and a quantity that represents the *specificity* of the node to which e is incident. The specificity is based on the assumption that the larger the degree k of a node — the more facts in the KG about it — the more *general* the concept [13], [14].

Constraints on nodes ensure conservation of flow: except for the subject s and object o, the amount of flow entering a node must be equal to that leaving the node. We associate with s (resp. o) a fixed *supply* (*demand*) of knowledge, γ, which is the maximum feasible flow through the network.

In network flow problems, costs map to quantities to be minimized, like the distance between two cities. We employ the idea that the degree of a node is a measure of generality to be minimized. We therefore set the cost of an edge $e = (v_i, v_j) \in E$ to $c_e = \log k(v_j)$, where v_j is the incident node in the direction from s to o.

Having defined the constraints and edge costs, we solve a *minimum cost maximum flow* problem [15, Ch. 1, 9, 10]. The flow assignment to the edges of the KG is a non-negative real-valued mapping $f : E \to \mathbb{R}^+$, that maximizes the total flow γ pushed from s to o while minimizing the total cost $\sum_{e \in E} c_e f(e)$ subject to the edge capacity and node conservation constraints.

In practice we solve the problem using an algorithm that finds the maximum flow as well as the set of paths, which we denote as the *stream of knowledge* $\mathcal{P}_{s,p,o}$. The maximum knowledge a path $P_{s,p,o}$ can carry is the minimum of the capacities of its edges, also called its *bottleneck* $\beta(P_{s,p,o})$. It can be shown that the maximum flow is the sum of the bottlenecks of the paths that are part of the stream:

$$\gamma = \sum_{P_{s,p,o} \in \mathcal{P}_{s,p,o}} \beta(P_{s,p,o}). \tag{3}$$

Having determined the maximum flow and the contribution of each individual path in a stream, we need to specify how to use the stream for fact checking. The flow through a path captures the relational similarity and specificity of its bottleneck, as well as the specificity of the intermediate nodes. Nevertheless, long chains of specific relationships could lead us astray. Therefore KS should favor *specific paths* involving few specific entities. We define the specificity $\mathcal{S}(P_{s,p,o})$ of a path $P_{s,p,o}$ with n nodes as:

$$\mathcal{S}(P_{s,p,o}) = \frac{1}{1 + \sum_{i=2}^{n-1} \log k(v_i)}. \tag{4}$$

We say that the net flow $\mathcal{W}(P_{s,p,o})$ in a path $P_{s,p,o}$ is the product of its bottleneck $\beta(P_{s,p,o})$ and specificity $\mathcal{S}(P_{s,p,o})$:

$$\mathcal{W}(P_{s,p,o}) = \beta(P_{s,p,o}) \cdot \mathcal{S}(P_{s,p,o}). \tag{5}$$

Fact checking a triple (s, p, o) then reduces to computing a *truth score* $\tau^{\text{KS}}(s, p, o)$ as the sum of the net flow across all paths in the stream:

$$\tau^{\text{KS}}(s, p, o) = \sum_{P_{s,p,o} \in \mathcal{P}_{s,p,o}} \mathcal{W}(P_{s,p,o}) \tag{6}$$

C. Computing the Knowledge Stream

To solve our optimization problem, we apply the well-known Successive Shortest Path (SSP) algorithm [15], as detailed in the technical report [14]. Our extended version of SSP to compute the stream of knowledge and the truth score $\tau^{\text{KS}}(s, p, o)$ for a given triple is shown in Algorithm 1.

Algorithm 1 Knowledge Stream Algorithm

procedure KNOWLEDGESTREAM(G, s, p, o)
  $\tau \leftarrow 0, \mathcal{P} \leftarrow \emptyset, f \leftarrow 0$
  $\pi \leftarrow 0$
  $c_{v_i,r_m,v_j} = \log(v_j), \forall (v_i, r_m, v_j) \in E$
  $c^{\pi}_{v_i,r_m,v_j} = c_{v_i,r_m,v_j} - \pi(v_i) + \pi(v_j)$
  $d \leftarrow$ compute shortest path distances from s to all other nodes in $G(f)$ w. r. t. c^{π}
  $P \leftarrow$ a shortest path from s to o in $G(f)$
  while P exists **do**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{P\}$
    $\pi \leftarrow \pi - d$
    $\beta(P) \leftarrow \min\left\{x_{v_i,r_m,v_j} \middle| (v_i, r_m, v_j) \in P\right\}$
    Push $\beta(P)$ units of flow along P
    $\mathcal{S}(P) \leftarrow \frac{1}{1+\sum_{i=2}^{n-1} \log k(v_i)}$ for $v_i \in P$
    $\mathcal{W}(P) \leftarrow \beta(P) \cdot \mathcal{S}(P)$
    $\tau \leftarrow \tau + \mathcal{W}(P)$
    update $f, G(f)$ and reduced edge lengths c^{π}
    $d \leftarrow$ compute shortest path distances from s to all other nodes in $G(f)$ w. r. t. c^{π}
    $P \leftarrow$ a shortest path from s to o in $G(f)$
  end while
  return $\tau, \mathcal{P}$
end procedure

The complexity bounds for the SSP algorithm assume that all edge weights are integral, which does not hold for our capacities ($\mathcal{U}_{s,p,o} \in [0, 1]$). However, since capacities are rational numbers, they can be converted to integers. If the maximum flow γ is an integer, the KS algorithm takes at most γ iterations. Since each shortest path computation can be performed in $O(|E| \log |V|)$ time using Dijkstra's algorithm [16] with a binary heap implementation, the overall complexity of the algorithm is $O(\gamma |E| \log |V|)$. As γ is not an integer, it is computed by the algorithm; this makes KS a pseudo-polynomial time algorithm. In practice, for large-scale KGs such as DBpedia, our implementation takes an average of 356 seconds per triple on a laptop.

D. Relational Knowledge Linker

Our measure of relational similarity defined in Section II-A can also be used to extend existing KG-based fact-checking methods. One such method is Knowledge Linker (KL) [13]. The approach used by KL for fact checking a triple (s, p, o) is to find the path between entities s and o that maximizes specificity (Eq. (4)). This approach ignores the semantics of the target predicate p. We hypothesize that biasing the search for specific paths to favor edges that are semantically related

to p should improve KL. We therefore replace the definition of path specificity in Eq. (4) by

$$\mathcal{S}'(P_{s,p,o}) = \left[\sum_{i=2}^{n-1} \frac{\log k(v_i)}{u(r_{i-1}, p)} + \frac{1}{u(r_{n-1}, p)}\right]^{-1}. \quad (7)$$

This formulation maximizes the relational similarity between each edge and the target predicate, in addition to the specificity of the intermediate nodes. The last term allows to consider the relation of the last edge without penalizing the generality of the object o. The truth score of triple (s, p, o) is just $\tau^{\text{KL-REL}}(s, p, o) = \max_{P_{s,p,o} \in \mathcal{P}_{s,p,o}} \mathcal{S}'(P_{s,p,o})$.

The truth score and the associated path can be computed efficiently using Dijkstra's algorithm [16]. We call this extended approach the Relational Knowledge Linker (KL-REL).

III. Evaluation

Let us evaluate Knowledge Stream and Relational Knowledge Linker against state-of-the-art approaches from the literature. Next we briefly outline the experimental setup; refer to the extended paper for further details [14].

We select DBpedia as the KG for all evaluations. We use its ontology, instance-types, and mapping-based properties information. We apply some filtering steps, yielding an undirected graph with $|V| = 6$M nodes, $|E| = 24$M triples, and $|\mathcal{R}| = 663$ relations.

We evaluate all methods on two classes of datasets (Table I). *Synthetic* corpora, created for evaluation purposes by our team and others, mix known true and false facts drawn from the domains of entertainment, business, geography, literature, sports, etc. *Real-world* datasets are derived from the Google Relation Extraction Corpora (GREC, research.googleblog.com/2013/04/50000-lessons-on-how-to-read-relation.html) and the WSDM Cup 2017 Triple Scoring challenge (www.wsdm-cup-2017.org/triple-scoring.html). These contain information about birth and death places, education, professions and nationalities of notable people. Real-world ground truth data was obtained through crowd sourcing. In all datasets, false facts are randomly drawn according to a *local closed-world assumption*.

We compare our approaches to three state-of-the-art algorithms designed for fact checking (Knowledge Linker [13], PredPath [9], and PRA [8]), one algorithm for knowledge graph completion (TransE [6]), and six link prediction algorithms (Katz [17], PathEnt [18], SimRank [19], Adamic & Adar [20], Jaccard coefficient [11], and Degree Product [9]). We use the area under the Receiver Operating Characteristic curve (AUROC) as an evaluation metric. Each method emits a list of probabilistic scores, one for each triple, and the AUROC expresses the probability that a true triple receives a higher score than a false one.

The source code for our methods can be found at github.com/shiralkarprashant/knowledgestream. For Katz, PathEnt, PRA and PredPath, we use up to 200 paths for every value of path length $l = 2, 3$. To scale SimRank to the size of DBpedia, we implemented the approximation based on random walks [21], and use decay factor 0.8, 50 terms, and 1000 random walks. In the case of TransE, we create 100-dimensional embeddings using a margin of one and a learning rate of 0.01 for 1000 epochs.

TABLE I
Summary of evaluation datasets. Asterisks mark those used in prior work [9]. Example triples and number of facts per subject can be found in the extended paper [14].

	Dataset	True / Total
Synthetic	NYT-Bestseller*	93/558
	NBA-Team	41/164
	Oscars	78/4680
	CEO*	201/1208
	US War*	126/710
	US-V. President*	47/274
	FLOTUS	16/256
	US-Capital #2*	50/300
Real-World	GREC-Birthplace	273/1092
	GREC-Deathplace	126/504
	GREC-Education	466/1861
	GREC-Institution	1546/6184
	WSDM-Nationality	50/200
	WSDM-Profession	110/440

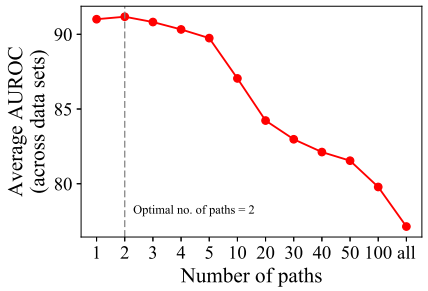

Fig. 4. Average performance of Knowledge Stream across datasets as a function of the number of paths used in the stream.

Table II and Table III compare fact-checking performance across approaches on all datasets. Although statistical significance tests do not reveal a clear overall winner, we can make a few observations. KL-REL performs better than the original KL, TransE, and all link prediction algorithms. It outperforms all other algorithms on real-world datasets and has comparable performance to PredPath on synthetic data.

KS lags behind KL, suggesting that not all additional paths may yield useful signals. To investigate, we analyzed the average performance as a function of the number of paths in the stream. Fig. 4 shows that the overall optimum is attained when two paths are considered. This confirms the signal of multiple paths must be traded off against their noise. Based on this insight, we include two variants of Knowledge Stream: KS-AVG uses two paths, and KS-CV uses cross-validation to tune the optimal number of paths for each dataset; this makes KS-CV a supervised approach. As we see from both tables, KS-AVG and KS-CV have better performance on average than KS, and even better than KL-REL on synthetic datasets.

We observe that KL-REL, KS, KS-AVG and KS-CV outper-

TABLE II
FACT-CHECKING PERFORMANCE (AUROC) ON SYNTHETIC DATA. BEST SCORES FOR EACH DATASET ARE SHOWN IN BOLD.

Method	NYT-Bestseller	NBA-Team	Oscars	CEO	US-War	US-V. President	FLOTUS	Capital #2	Avg. (S.E.)
KL-REL	96.32	99.94	97.67	**89.88**	86.34	87.29	98.32	**100.00**	94.47 (2.0)
KS	89.72	**99.96**	95.00	81.19	72.11	77.80	98.05	**100.00**	89.23 (3.9)
KS-AVG	91.95	99.01	98.13	80.96	**99.98**	**99.53**	99.09	99.76	96.05 (2.3)
KS-CV	93.63	99.29	97.72	80.52	**99.98**	99.47	99.27	99.28	96.14 (2.3)
PredPath [9]	**99.80**	92.31	**99.97**	88.67	99.51	94.40	**100.00**	99.68	**96.79** (1.6)
KL [13]	94.99	99.94	97.56	89.77	63.55	74.62	98.59	99.42	89.80 (4.8)
PRA [8]	96.24	91.26	99.54	87.73	99.96	50.00	60.48	98.88	85.51 (6.8)
TransE [6]	80.99	56.71	82.66	82.68	53.22	72.50	84.82	85.31	74.86 (4.6)
Katz [17]	96.52	98.50	98.98	87.53	57.80	72.92	97.42	99.97	88.70 (5.5)
PathEnt [18]	97.53	97.00	98.99	92.96	93.65	90.60	**100.00**	**100.00**	96.34 (1.2)
SimRank [19]	88.36	97.74	51.47	80.83	50.03	65.78	91.85	99.50	78.19 (7.1)
Adamic & Adar [20]	95.84	99.73	56.54	84.97	54.98	81.06	99.40	**100.00**	84.06 (6.7)
Jaccard [11]	92.64	99.42	53.35	78.74	49.68	70.79	97.89	**100.00**	80.31 (7.3)
Degree Product [9]	56.52	53.21	54.42	49.17	64.08	49.55	50.00	52.10	53.63 (1.7)

form existing fact-checking methods (PredPath, KL and PRA), even though the latter are supervised while KL-REL and KS-AVG are not.

Finally, link prediction algorithms (PathEnt, SimRank, Adamic & Adar, Jaccard coefficient, and Degree Product) tend to perform poorly. Katz is the exception; its performance is comparable to KL-REL on real-world datasets. However, both KS or KL-REL are computationally efficient compared to Katz; KL-REL focuses on a single path, whereas KS is better able to penalize long paths thanks to the capacity constraints.

For each triple, the paths discovered by algorithms like KS, KL-REL, PRA, and PredPath can be seen as the evidence used by the algorithm in deciding whether a fact is true. By pooling together evidence from many triples, we can discover data-driven patterns that define a relation in the KG. It is natural to ask whether the patterns discovered by our methods conform to common-sense understanding of these relations. To do so, we perform the following exercise. For each relation, we define the two sets A and B of all paths discovered from either true or false triples, respectively. We then rank the paths in decreasing order of their frequency of occurrence in the set difference $A - B$. Table IV shows the top patterns discovered by KS for a few example relations. The patterns are highly relevant. Many other interesting examples are omitted. With only a few true and false examples, the patterns discovered by KS can also be applied to information extraction concept learning.

IV. DISCUSSION AND FUTURE WORK

In this paper, we have shown that network flow theory can serve as a useful toolbox for reasoning about facts, and for fact checking in particular. We presented two novel, unsupervised approaches to assess and explain the truthfulness of a statement of fact by leveraging its semantic context in a knowledge graph. We also proposed a method to measure the similarity of any two relations based on their co-occurrence in the KG. We evaluated both approaches on a diverse set of real-world and synthetic test cases, and found that their performance is on par with the state of the art. Moreover, we saw that, in many cases, multiple paths can provide additional evidence to support fact checking. Our Knowledge Stream model offers high expressive power by its ability to automatically surface useful path patterns and relevant facts about a claim. Based on this experience we believe that network flow techniques are particularly promising for fact checking.

In practice, KS can assist a human fact checker by identifying the general context of a triple. As an illustration, Fig. 1 shows the set of most relevant facts (paths) for the triple `(David and Goliath (book), author, Malcolm Gladwell)`, with the width of edges roughly proportional to their net flow. Notice the diversity in the set of facts that support this triple. KS is able to "bubble up" the most relevant facts (wider edges). KS automatically surfaces these relevant facts in an unsupervised way. We believe that it is the first computational fact-checking approach featuring such an expressive power.

Much room remains for improving KS to address complex test cases. For example, we have used relational similarity to design edge capacities in the graph; other alternatives should be explored. The development of effective relational similarity metrics is another important avenue of future work. The capacities could also incorporate metadata from the KG itself, for example confidence scores or spatio-temporal details from the KG. Lastly, our version of KS relies on successive path-finding, which can be slow for triples involving subjects with a large search space. Our implementation takes a few minutes to check each triple with DBpedia. Other approaches could be explored in the future. For example, the network simplex algorithm [15] has better theoretical and run-time behavior.

Acknowledgements: We thank B. Shi and T. Weninger for sharing evaluation data. This work was supported in part by NSF (Award CCF-1101743) and DARPA (grant W911NF-12-1-0037). Funding agencies had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

REFERENCES

[1] L. Howell, "Digital wildfires in a hyperconnected world," World Economic Forum, Tech. Rep., 2013. [Online]. Available: http://reports.weforum.org/global-risks-2013/risk-case-1/digital-wildfires-in-a-hyperconnected-world/

TABLE III
FACT-CHECKING PERFORMANCE (AUROC) ON REAL TEST DATASETS. BEST SCORES FOR EACH DATASET ARE SHOWN IN BOLD.

Method	GREC Birthplace	GREC Deathplace	GREC Education	GREC Institution	WSDM Nationality	WSDM Profession	Avg. (S. E.)
KL-REL	**92.54**	**90.91**	86.44	85.64	96.92	97.32	**91.63** (2.0)
KS	72.92	80.02	89.03	78.62	97.92	98.66	86.20 (4.4)
KS-AVG	81.38	83.58	75.46	81.31	93.37	92.93	84.67 (2.9)
KS-CV	82.28	82.57	75.23	81.33	94.20	95.84	85.24 (3.3)
PredPath [9]	84.64	76.54	83.21	80.14	95.20	92.71	85.41 (2.9)
KL [13]	92.10	90.49	62.32	**87.61**	96.05	91.36	86.65 (5.0)
PRA [8]	74.34	75.58	70.51	63.95	83.87	50.00	69.71 (4.8)
TransE [6]	54.88	56.47	66.32	44.99	77.09	82.91	63.78 (5.9)
Katz [17]	88.46	84.07	89.55	82.99	**99.23**	**98.84**	90.52 (2.9)
PathEnt [18]	84.02	79.2	83.48	82.26	55.31	48.13	72.07 (6.5)
SimRank [19]	86.91	85.48	67.60	72.92	89.39	92.27	82.43 (4.0)
Adamic & Adar [20]	82.79	79.13	50.00	74.58	97.21	95.07	79.80 (7.0)
Jaccard [11]	80.39	75.99	49.95	69.88	95.93	90.01	77.02 (6.6)
Degree Product [9]	52.82	50.86	**91.51**	64.56	84.38	86.36	71.75 (7.3)

TABLE IV
RELATIONAL PATTERNS DISCOVERED BY KNOWLEDGE STREAM. ADDITIONAL PATTERNS IN TECHNICAL REPORT [14].

Relation	Pattern	Freq.	Example
Spouse	(child, childOf)	34	J. F. Kennedy $\xrightarrow{\text{child}}$ Patrick Kennedy $\xrightarrow{\text{childOf}}$ Jacqueline Kennedy Onassis
	(parentOf, parent)	20	J. F. Kennedy $\xrightarrow{\text{parentOf}}$ Patrick Kennedy $\xrightarrow{\text{parent}}$ Jacqueline Kennedy Onassis
CEO	(parentCompanyOf, keyPerson)	32	News Corporation $\xrightarrow{\text{parentCompanyOf}}$ Sky TV plc $\xrightarrow{\text{keyPerson}}$ Rupert Murdoch
	(employerOf)	24	Twitter $\xrightarrow{\text{employerOf}}$ Dick Costolo
US Capital	(deathPlaceOf, deathPlace)	491	Delaware $\xrightarrow{\text{deathPlaceOf}}$ Nathaniel B. Smithers $\xrightarrow{\text{deathPlace}}$ Dover, Delaware
	(part, isPartOf)	123	Delaware $\xrightarrow{\text{part}}$ Delaware Valley $\xrightarrow{\text{isPartOf}}$ Dover, Delaware

[2] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter, "Detection and resolution of rumours in social media: A survey," *arXiv preprint arXiv:1704.00656*, 2017.

[3] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "Dbpedia-a crystallization point for the web of data," *Web Semantics*, vol. 7, no. 3, pp. 154–165, 2009.

[4] F. Erxleben, M. Günther, M. Krötzsch, J. Mendez, and D. Vrandečić, "Introducing wikidata to the linked data web," in *Proc. International Semantic Web Conference*, 2014.

[5] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proc. of the IEEE*, vol. 104, no. 1, pp. 11–33, 2016.

[6] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems*, 2013, pp. 2787–2795.

[7] L. A. Galárraga, C. Teflioudi, K. Hose, and F. Suchanek, "AMIE: Association rule mining under incomplete evidence in ontological knowledge bases," in *Proc. 22nd Intl. Conf. on World Wide Web*, 2013, pp. 413–422.

[8] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Machine Learning*, vol. 81, no. 1, pp. 53–67, 2010.

[9] B. Shi and T. Weninger, "Discriminative predicate path mining for fact checking in knowledge graphs," *Knowledge-Based Systems*, vol. 104, pp. 123–133, 2016.

[10] A. G. Maguitman, F. Menczer, F. Erdinc, H. Roinestad, and A. Vespignani, "Algorithmic computation and approximation of semantic similarity," *World Wide Web*, vol. 9, no. 4, pp. 431–456, 2006.

[11] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[12] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A*, vol. 390, no. 6, pp. 1150–1170, 2011.

[13] G. L. Ciampaglia, P. Shiralkar, L. M. Rocha, J. Bollen, F. Menczer, and A. Flammini, "Computational fact checking from knowledge networks," *PLoS ONE*, vol. 10, no. 6, p. e0128193, 2015.

[14] P. Shiralkar, A. Flammini, F. Menczer, and G. L. Ciampaglia, "Finding streams in knowledge graphs to support fact checking," arXiv, Technical Report 1708.07239, 2017.

[15] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice hall, 1993.

[16] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[17] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[18] Z. Xu, C. Pu, and J. Yang, "Link prediction based on path entropy," *Physica A*, vol. 456, pp. 294–301, 2016.

[19] G. Jeh and J. Widom, "Simrank: a measure of structural-context similarity," in *Proc. 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 2002, pp. 538–543.

[20] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social networks*, vol. 25, no. 3, pp. 211–230, 2003.

[21] M. Kusumoto, T. Maehara, and K.-i. Kawarabayashi, "Scalable similarity search for simrank," in *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, 2014, pp. 325–336.