Knowledge-Based Systems 104 (2016) 123–133

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Discriminative predicate path mining for fact checking in knowledge graphs

Baoxu Shi[a], Tim Weninger[b,*]

[a] Department of Computer Science and Engineering, University of Notre Dame, 384E Nieuwland Science Hall, Notre Dame, IN 46556, USA
[b] Department of Computer Science and Engineering, University of Notre Dame, 353 Fitzpatrick Hall, Notre Dame, IN 46556, USA

ARTICLE INFO

Article history:
Received 24 February 2016
Revised 15 April 2016
Accepted 16 April 2016
Available online 19 April 2016

Keywords:
Fact checking
Path finding
Knowledge graph

ABSTRACT

Traditional fact checking by experts and analysts cannot keep pace with the volume of newly created information. It is important and necessary, therefore, to enhance our ability to computationally determine whether some statement of fact is true or false. We view this problem as a link-prediction task in a knowledge graph, and present a *discriminative path*-based method for fact checking in knowledge graphs that incorporates connectivity, type information, and predicate interactions. Given a statement $\mathcal{S}$ of the form (subject, predicate, object), for example, (Chicago, capitalOf, Illinois), our approach mines discriminative paths that alternatively define the generalized statement (U.S. city, predicate, U.S. state) and uses the mined rules to evaluate the veracity of statement $\mathcal{S}$. We evaluate our approach by examining thousands of claims related to history, geography, biology, and politics using a public, million node knowledge graph extracted from Wikipedia and PubMedDB. Not only does our approach significantly outperform related models, we also find that the discriminative predicate path model is easily interpretable and provides sensible reasons for the final determination.

1. Introduction

> *If a Lie be believ'd only for an Hour, it has done its Work, and there is no farther occasion for it. Falsehood flies, and the Truth comes limping after it.*
>
> – Jonathan Swift (1710) [1]

Misinformation in media and communication creates a situation in which opposing assertions of fact compete for attention. This problem is exacerbated in modern, digital society, where people increasingly rely on the aggregate ratings from their social circles for news and information. Although much of the information presented on the Web is a good resource, its accuracy certainly cannot be guaranteed. In order to avoid being fooled by false assertions, it is necessary to separate fact from fiction and to assess the credibility of an information source.

Knowledge graphs. We represent a *statement of fact* in the form of (subject, predicate, object) triples, where the subject and the object are entities that have some relationship between them as indicated by the predicate. For example, the *"Springfield is the capital of Illinois"* assertion is represented by the triple (Springfield, capitalOf, Illinois). A set of such triples is known as a knowledge base, but can be combined to produce a multi-graph where nodes represent the entities and directed edges represent the predicates. Different predicates can be represented by edge types, resulting in a heterogeneous information network that is often referred to as a *knowledge graph*. Given a knowledge base that is extracted from a large repository of statements, like Wikipedia or the Web at large, the resulting knowledge graph represents *some* of the factual relationships among the entities mentioned in the statements. If there existed an ultimate knowledge graph which knew everything, then fact checking would be as easy as checking for the presence of an edge in the knowledge graph. In reality, knowledge graphs have limited information and are often plagued with missing or incorrect relations making validation difficult.

Although a knowledge graph may be incomplete, we assume that most of the edges in the graph represent true statements of fact. With this assumption, existing fact checking [2] and link prediction methods [3–7] would rate a given statement to be true if it exists as an edge in the knowledge graph or if there is a short path linking its subject to its object, and false otherwise. Statistical relational learning models [8–11] can measure the truthfulness by calculating the distance between the entities and predicate in a given statement. However, the limitation of existing models make them unsuitable for fact checking. Link prediction methods, Adamic/Adar [4] and personalized PageRank [7] for example,

* Corresponding author. Tel.: +5746316770.
E-mail addresses: bshi@nd.edu (B. Shi), tweninge@nd.edu (T. Weninger).

http://dx.doi.org/10.1016/j.knosys.2016.04.015
0950-7051/

work on untyped graphs and are incapable of capturing the heterogeneity of knowledge graphs; other heterogeneous link prediction algorithms, *e.g.*, PathSim [12] and PCRW [14], not only need human annotated meta paths but also have strict constraints on the input meta paths. Statistical relational learning models such as RESCAL [8], NTN [9], TransE [10], and other variants [11,17] utilize the type information in knowledge graphs but can not work with unseen predicate types and do not model the complicated interactions among relations explicitly.

In the present work, we present a discriminative path-based method for fact checking in knowledge graphs that incorporates connectivity, type information, and predicate interactions. Given a statement $\mathcal{S}$ of the form (subject, predicate, object), for example, (Chicago, capitalOf, Illinois), our approach mines discriminative paths that alternatively define the generalized statement (U.S. city, predicate, U.S. state) and uses the mined rules to evaluate the veracity of statement $\mathcal{S}$.

Unlike existing models, the proposed method simulates how experienced human fact-checkers examine a statement: fact-checkers will first attempt to *understand* the generalized notion of the statement using prior knowledge, and then validate the specific statement by applying their knowledge. The statement usually can be generalized by replacing the specific entities by their type-labels. In the present work, we show how to understand a statement by inspecting the related discriminative paths retrieved from the knowledge graph. Returning to the "*Chicago is the capital of Illinois*" example, a fact checker, as well as our model, will learn to understand what it means for a U.S. city to be the capitalOf a U.S. state. In this trivial example, a fact checker may come to understand that a city is the capital of a state if the state agencies, governor and legislature are located in the city; from this understanding the fact checker ought to rule that Chicago is not the capital of Illinois because this statement does not satisfy the fact checker's understanding of what capitalOf means.

The advantages of this fact checking procedure is in its *generality* and *context-dependency*. Just as humans learn unknown words, model generality means the predicate of a statement can be arbitrary and is not required to be presented in the knowledge base. Moreover, once a prior knowledge is learned, it is associated with a certain type of entity pair relation and can be used for different tasks including general question answering or knowledge base completion. The notion of context-dependency allows the fact checker to discern different definitions of a predicate in different situations. For example, capitalOf could define the capitals of US states, colloquialisms such as "Kansas City is the soccer capital of America", or historical or time-sensitive predicates such as "Calcutta *was* the capital of India" depending on the context.

When performed computationally, the task of discovering interesting relationships between or among entities is known generally as association rule mining. Although there has been some effort to adapt association mining for knowledge graph completion, these methods are not well suited for fact-finding and often resort to finding global rules and synonyms [18,19] rather than generating a robust understanding of the given context dependent predicate [20].

Fig. 1 illustrates three graph fragments from the DBpedia knowledge base [21] containing cities and states. This example demonstrates, via actual results, how the proposed automatic fact checker is able to determine relationships that uniquely define what it means for an entity to be the capitalOf another entity. Association rule miners [19] and link prediction models [5,6] incorrectly indicate that the largestCity is most associated with the capitalOf predicate. In contrast, our framework, indicated by solid edges, finds the rules that most uniquely define what it means to be the capitalOf a state. In this example, our top result indicates that a US state capital is the city in which the headquarters of entities that have jurisdiction in the state are located. In other words, we find that a US state capital is indeed the city where the state agencies, like the Dept. of Transportation, or the Dept. of Health, have their headquarters.

To summarize, we show that we can leverage a collection of factual statements for automatic fact checking. Based on the principles underlying link prediction, similarity search and network closure, we computationally gauge the truthfulness of an assertion by mining connectivity patterns within a network of factual statements. Our current work focuses on determining the validity of factual assertions from simple, well-formed statements; the related problems of information extraction [22], claim identification [23], answering compound assertions [24], and others [25] are generally built in-support-of or on-top-of this central task.

Recent work in general heterogeneous information networks, of which knowledge graphs are an example, has led to the development of meta path similarity metrics that show excellent results in clustering, classification and recommendation [12,14,26,27]. The state of the art in meta path mining works by counting the path-instances or randomly walking over a constrained set of hand-annotated typed-edges [12]. Unfortunately, this means that a human has to understand the problem domain and write down relevant meta paths before analysis can begin. In this work, our focus is on methods that automatically determine the set of path-descriptions called **discriminative paths** that uniquely encapsulate the relationship between two entities in a knowledge graph.

The specific contributions of this paper are as follows:

1. We developed a fast discriminative path mining algorithm that can discover "definitions" of an RDF-style triple, *i.e.*, a statement of fact. The algorithm is able to handle large scale knowledge graphs with millions of nodes and edges.
2. We designed a human interpretable fact checking framework that utilizes discriminative paths to predict the truthfulness of a statement.
3. We modeled fact checking as a link prediction problem and validated our approach on two real world, large scale knowledge graphs, DBpedia [21] and SemMedDB [28]. The experiments showed that the proposed framework outperforms alternative approaches and has a similar execution time.

In this paper, we incorporate lessons learned from association rule mining and from heterogeneous information network analysis in order to understand the meanings of various relationships, and we use this new framework for fact-checking in knowledge graphs. To describe our approach we first formalize the problem in Section 2 and define our solution in Section 3. Section 4 presents extensive experiments on two large, real world knowledge graphs. We present related work in Section 5 before drawing conclusions and discussing future work in Section 6.

2. Problem definition

We view a knowledge graph to be a special case of a heterogeneous information network (HIN) where nodes represent entities and edges represent relationships between entities, and where heterogeneity stems from the fact that nodes and edges have clearly identified type-definitions. The type of an entity is labeled by some ontology, and the type of an edge is labeled by the predicate label. With the above assumptions, we formally define a knowledge graph as follows:

Definition 1 (Knowledge Graph)**.** A knowledge graph is a directed multigraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R}, \mathcal{O}, \psi, \phi)$, where $\mathcal{V}$ is the set of entities, $\mathcal{E}$ is a set of labeled directed edges between two entities, $\mathcal{R}$ represents the predicate label set, and $\mathcal{O}$ is the ontology of the entities in $\mathcal{G}$. The ontology mapping function $\psi(v) = \mathbf{o}$, where $v \in \mathcal{V}$

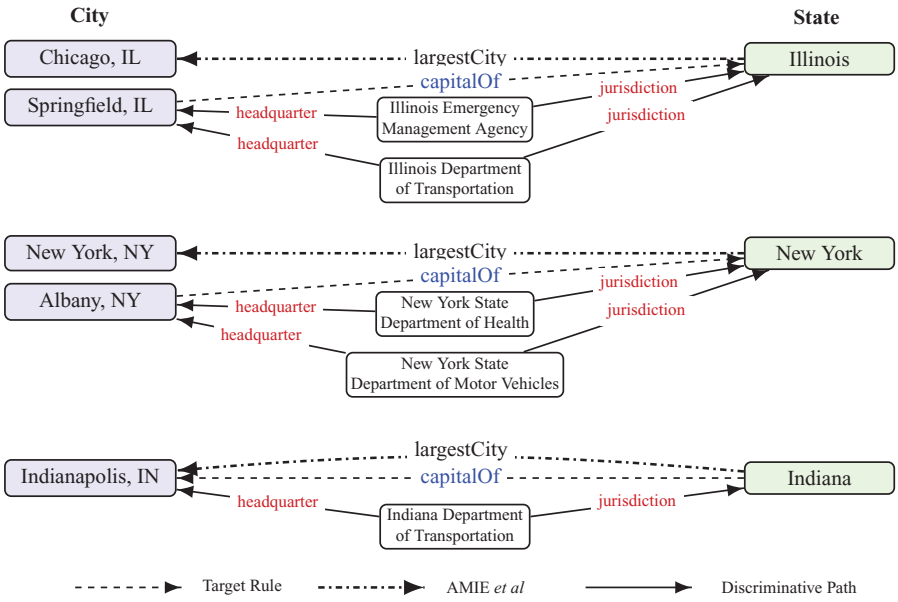

Fig. 1. Knowledge graph of US cities and states from DBpedia. $\{\text{city}\}\xrightarrow{\text{largestCity}^{-1}}\{\text{state}\}$ and $\{\text{city}\}\xrightarrow{\text{headquarter}^{-1}}\{\text{entity}\}\xrightarrow{\text{jurisdiction}}\{\text{state}\}$ are the discriminative paths of $\{\text{city}\}\xrightarrow{\text{capitalOf}}\{\text{state}\}$ mined by AMIE [19] and the proposed method respectively.

and $\mathbf{o} \subset \mathcal{O}$, links an entity vertex to its label set in the ontology. The predicate mapping function $\phi(e) = p$, where $e \in \mathcal{E}$ and $p \in \mathcal{R}$, maps an edge to its predicate type.

The knowledge graph defined here differs from the standard definition of an HIN; Definition 1 dictates that a node may be mapped to multiple types, which is unlike the traditional HIN definition in which each node can be mapped to only one type-label [12]. When $\psi(v) = \mathbf{o}$ satisfies $|\mathbf{o}| = 1$ for all $v \in \mathcal{V}$, then Defn. 1 degenerates to the standard HIN definition.

For example, the DBpedia knowledge base can be represented as a knowledge graph in which $\mathcal{V}$ represents entities like Springfield, Chicago, or Illinois; $\mathcal{E}$ represents some link between two entities; $\mathcal{O}$ represents a classification scheme like the Wikipedia Category graph with type-labels like city and state categories for Chicago and Illinois respectively; and $\mathcal{R}$ represents the predicate labels like capitalOf and largestCity for edges.

Typed nodes and edges given in the knowledge graph naturally result in an enhanced set of connections called *meta paths* that describe how two entities are connected by their type-labels.

Definition 2 (Meta Path)**.** Given a knowledge graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R}, \mathcal{O}, \psi, \phi)$, a meta path Π^k is defined as a directed, typed sequence of vertices and edges $\mathbf{o}_1 \xrightarrow{p_1} \mathbf{o}_2 \xrightarrow{p_2} \dots \xrightarrow{p_{k-1}} \mathbf{o}_k$ in $\mathcal{G}$, where $\mathbf{o}_i$ denotes the set of ontology labels of vertex i, p_i represents the predicate of the directed edge that connects vertex i to $i+1$, and k denotes the length of the meta path.

If we relax the definition of a meta path in Definition 2 in such a way that the edges still carry type-information, but the non-endpoint nodes do not, then the meta path degenerates into an *anchored predicate path* anchored by their starting and ending entity-types:

Definition 3 (Anchored Predicate Path)**.** Given a k-length meta path Π^k, the anchored predicate path P is defined as the corresponding directed, typed sequence of edges with typed-endpoints $P^k = \mathbf{o}_1 \xrightarrow{p_1} \xrightarrow{p_2} \dots \xrightarrow{p_{k-1}} \mathbf{o}_k$.

With the definitions of knowledge graph and anchored predicate path, here we define the *discriminative path* of a statement as:

Definition 4 (Discriminative Paths)**.** The set of discriminative paths $\mathbf{D}^k_{(\mathbf{o}_u, \mathbf{o}_v)}$ are those anchored predicate paths that alternatively describe the given statement of fact $\mathbf{o}_u \xrightarrow{p} \mathbf{o}_v$, where the maximum path length is k.

For example, a meta path Π between two entities Illinois and Springfield is represented by the following sequence {city, settlement} $\xrightarrow{\text{headquarter}^{-1}}$ {state agency} $\xrightarrow{\text{jurisdiction}}$ {state}. The corresponding anchored predicate path P is $\langle\ \text{headquarter}^{-1}, \text{jurisdiction}\ \rangle$ anchored by {city, settlement} and {state}. If $P \in \mathbf{D}$ holds, that means capitalOf can be at least partially defined by P. We discuss how to discover these discriminative paths in the next section.

Note that in our generalization of HIN, the first entity in the meta path is mapped to two type-labels, and could have many more. Entities with many type labels tend to be more prone to label error when using the meta path representation. With this in mind, we choose to use the anchored predicate path representation which we find to have better tolerance on errant type labels. A detailed comparison is given in Section 4.

With the definitions above, the goal of this paper can be formally stated as:

Definition 5 (Fact Checking)**.** Given a knowledge graph $\mathcal{G}$ and a statement of fact $\mathcal{S} = (s, p, t)$, which may be true or untrue, where subject $s \in \mathcal{V}$, object $t \in \mathcal{V}$. Fact checking is the process of using a learned understanding of the relationship $\mathbf{D}^k_{(\mathbf{o}_u, \mathbf{o}_v)}$ to determine whether the edge $s \xrightarrow{p} t$ is missing in $\mathcal{G}$ such that $\mathbf{o}_s = \mathbf{o}_u$ and $\mathbf{o}_t = \mathbf{o}_v$.

Simply put, we view the fact checking problem as a supervised link prediction task and validate a proposed fact statement (s, p, t) by determining if that the proposed fact is implied by the data within the knowledge graph. When $p \in \mathcal{R}$ holds, the positive paths $\mathbf{T}^+$ of the predicate p can be automatically generated by $\mathbf{T}^+ = \{(u, v) | u \xrightarrow{p} v \in \mathcal{G}\}$, and negative descriptions $\mathbf{T}^-$ of the predicate p can be automatically generated by $\mathbf{T}^- = \{(u, v) | u \xrightarrow{p} v \notin \mathcal{G}\}$ such that $\mathbf{o}_u = \mathbf{o}_s$ and $\mathbf{o}_v = \mathbf{o}_t$.

$\mathbf{T}^+$ and $\mathbf{T}^-$ can also be human provided if $p \notin \mathcal{R}$.

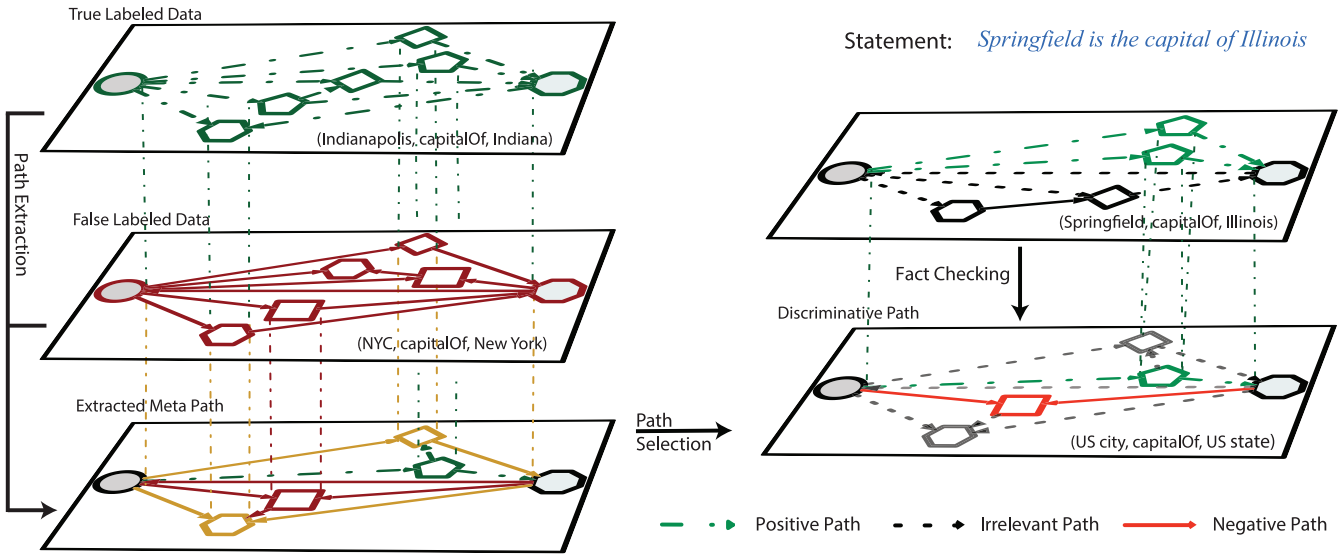

Fig. 2. Overview of the proposed fact checking framework. We first extract meta/predicate paths from labeled data set. Then we preform feature selection on the meta/predicate paths to determine the importance of each meta/predicate path and construct the prediction model. Finally we compare the given statement of fact with the learned model and output the judgement. This figure is best viewed in color.

Unlike traditional link prediction problems which simply identify true edges from all possible edges, the fact checking problem is harder because it needs to distinguish true edges $s \xrightarrow{p} t$ from a smaller edge set $\mathbf{\Pi} = \{s' \xrightarrow{p} t'\}$ where s' and t' are have the same type, connectivity, etc. as s and t respectively. Traditional algorithms, which use purely topological features, do not work well in this task because the topology of the network does not sufficiently distinguish between true and false edges.

In this work we propose a model that automatically discovers discriminative paths in order to perform fact checking. The resulting discriminative paths define the proposed predicate p in terms of its subject s and its object t by asking two questions: 1) does the predicate p connect entities that are of the same or similar type as s and t (generality), and 2) do the paths that connect s to t differ from the paths that connect similarly-typed entities but which are *not* connected by p (context-dependent)?

The paths that maximize the above questions 1 and 2 are those paths that uniquely define the predicate in terms of its subject and object. These paths are critical to modelling statements of fact and determining their veracity.

Next we will demonstrate how to test the veracity of statements of fact using discriminative path analysis.

3. Discriminative path analysis

As defined above, *discriminative* paths $\mathbf{D}^k_{(\mathbf{o}_s,\mathbf{o}_t)}$ are those anchored predicate paths that alternatively represent the predicate p from some proposed fact triple (s, p, t) between subjects of the same type as s (denoted $\psi(s)$) and objects of the same type as t (denoted $\psi(t)$). For example, the proposed fact triple at the top of Fig. 1 is (Springfield, capitalOf, Illinois). Other subjects of the same {city, settlement}-type include Indianapolis, Chicago, New York City, Albany, etc., and other objects of the same {state}-type include New York, Indiana, etc. In this example, one of the predicate paths P that alternatively and uniquely captures this relationship is $\langle \text{headquarter}^{-1}, \text{jurisdiction} \rangle$ which is anchored by {city, settlement} and {state}.

In addition, $\mathbf{D}^k_{(\mathbf{o}_s,\mathbf{o}_t)}$ includes many other discriminative anchored predicate paths of length $\leq k$ that connect $\psi(u) = \mathbf{o}_s$ to $\psi(v) = \mathbf{o}_t$, *i.e.*, {city, settlement} to {state}.

In Fig. 2, we illustrate our proposed fact checking system that contains three phases: 1) extraction, 2) selection, and 3) validation. The extraction phase collects anchored predicate paths that alternatively connect the subject and object of a proposed statement of fact. Using the extracted paths $\mathbf{P}^k_{(\mathbf{o}_s,\mathbf{o}_t)}$, the selection phase defines the fact with the most discriminating anchored predicate paths $\mathbf{D}^k_{(\mathbf{o}_s,\mathbf{o}_t)}$. The validation phase compares the actual statement of fact, *i.e.*, how the subject, predicate and object are actually connected, with a statistical model constructed from the discriminative paths.

3.1. Path extraction

Unlike existing meta path based models, which require hand annotation [12,27] or exhaustive enumeration [14] and are impractical on large-scale systems, we learn the best descriptions automatically.

We propose a fast discriminative path discovery algorithm using a constrained graph traversal process. The key idea is the assumption that although the number of paths in a knowledge graph is huge, only a small portion are actually helpful for a given task. Furthermore, among the reduced set of helpful meta paths, only a few may be discriminative in the presence of some predicate.

For example, if (Springfield, capitalOf, Illinois) is the proposed statement of fact in need of checking, the meta paths we are interested in are only those that start at a city and end at a state. So instead of enumerating all possible paths, we collect anchored predicate paths by traversing the graph starting from the given subject-entity and ending at the given object-entity up to a length of k.

To find which paths between $\mathbf{o}_u$ and $\mathbf{o}_v$ are most discriminating we further create positive and negative node-pair sets $\mathbf{T}^+$ and $\mathbf{T}^-$ and retrieve two anchored predicate path sets $\mathbf{P}^+_{(\mathbf{o}_s,\mathbf{o}_t)}$ and $\mathbf{P}^-_{(\mathbf{o}_s,\mathbf{o}_t)}$ respectively using a depth-first multi-graph traversal. Specifically, our DFS-like graph traversal is based on a closure function $\mathbb{C}$, which we define as:

$$\mathbb{C}_p(v) = \left\{(p, v') | (v, p, v') \in \mathcal{G}\right\} \cup \left\{(p^{-1}, v') | (v', p, v) \in \mathcal{G}\right\}, \quad (1)$$

where v is some entity-node and p denotes the predicate associated with the closure. Simply put, $\mathbb{C}_p(v)$ finds all nodes that can be reached from v via predicate p or p^{-1}.

Then we define a transition function $\mathcal{T}(v_i)$ which returns all v_{i+1} candidates, *i.e.*, all of the next entity-nodes, for a path $v_1 \xrightarrow{p_1} v_2 \xrightarrow{p_2} \dots \xrightarrow{p_{i-1}} v_i$ as

$$\mathcal{T}(v_i) = \left\{ \cup_{p \in \mathcal{R}} \mathbb{C}_p(v_i) \setminus \cup_{j=1}^{i} \{v_j\} \right\}, \tag{2}$$

which contains all of the possible next-nodes that can be visited from $\mathbb{C}_p(v)$ except those that have already been visited. Using the closure function $\mathbb{C}_p(v)$ and transition function $\mathcal{T}(v)$, we retrieve the path set $\mathcal{P}$ with path length $\leq k$ by $\mathcal{P} = \cup_{i=1}^{k} \mathcal{P}^k$, s.t.

$$\mathcal{P}^k = \{s, \mathcal{T}(v_1), \mathcal{T}(v_2), \dots, \mathcal{T}(v_{k-2}), t | \\ (s,t) \in \mathbf{T}, v_1 = s, v_i \in \mathcal{T}(v_{i-1}), t \in \mathcal{T}(v_{k-1})\}. \tag{3}$$

Unlike traditional graph traversal algorithms that follow the edge direction, our implementation records and follows both directions of each visited edge if possible. In this way, the algorithm actually discovers paths such as {city} $\xrightarrow{\text{headquarter}^{-1}}$ {state agency} $\xrightarrow{\text{jurisdiction}}$ {state}, which is technically a three node subgraph rather than a path by traditional definitions.

At this point, our framework will have gathered several anchored predicate paths, some of which may be helpful while others may be unhelpful or spurious. Next, we calculate the importance of each extracted anchored predicate path for inclusion into a final regression model.

3.2. Meta path versus predicate path

Although existing heterogeneous information network analysis methods usually utilize meta paths, in this work we use anchored predicate paths as the feature set. As introduced before, the presence of entity types in meta paths may sometimes be redundant and can typically be inferred by the predicate if there is no ambiguity. For example in the DBLP network [29], writtenBy always connects a paper with an author, and cite always connects two papers. In such cases, meta paths can be converted into predicate path without ambiguity.

In more complex knowledge graphs, such as DBpedia [21] and SemMedDB [28], an entity can have multiple type-labels. The labels of the same types of entities may not always be consistent due to mislabelling or different interpretations. An example would be the type labels of Boston and Sacramento, which are the capitals of Massachusetts and California respectively. The type-labels of Boston is {city, settlement, populated place}, whereas the type-labels of Sacramento are {settlement, populated place}. Because the type labels of Sacramento do not exactly match the type-labels of Boston, then a meta path model would treat these paths differently, resulting in many highly overlapping, but not exactly matching, paths.

We use the Jaccard coefficient [30] to measure the similarity of entity-label sets and use this score to reduce redundant meta paths when possible:

$$J(\psi(u), \psi(v)) = \frac{|\psi(u) \cap \psi(v)|}{|\psi(u) \cup \psi(v)|}. \tag{4}$$

We combine two meta paths $\{s\} \xrightarrow{p} \{t\}$ and $\{s'\} \xrightarrow{p} \{t'\}$ if $J(\psi(s), \psi(s'))$ and $J(\psi(t), \psi(t'))$ are larger or equal to a threshold. In practice, we set the threshold to 0 for all non-endpoint entities, which means we ignore the entity-type information in meta paths; for meta path endpoints we set the threshold as $\frac{1}{|\psi(u) \cup \psi(v)|}$. Ultimately, this method converts the meta paths into anchored predicate paths.

Intuitively, the use of more tightly constrained meta paths instead of predicate paths ought to increase the information richness leading to better results, but our initial trials showed that: 1) the use of meta paths significantly increased the number of variables in our fact checking model without any noticeable improvement in performance, and 2) the inclusion of noisy entity types from meta paths actually lowered the occurrence-rate of important meta paths resulting in lower discriminative power in the set of paths. A detailed discussion of these counter-intuitive results are provided in the experiment section.

To recap, at this point we have selected positive anchor-entities $\mathbf{T}^+$ and negative anchor-entities $\mathbf{T}^-$. Using those anchors we find many predicate paths $\mathbf{P}^+$ and $\mathbf{P}^-$ that connect the positive and negative anchors; these paths are essentially alternate descriptions of the original predicate that provide evidence that can be used to uniquely define the original predicate.

3.3. Path selection

In this section we describe the procedure used to find the most discriminative predicate paths $\mathbf{D}$ from predicate path sets $\mathbf{P}^+$ and $\mathbf{P}^-$. For this we define $\mathbf{X}$ to be an $n \times m$ training instance matrix, wherein the ith row in $\mathbf{X}$ represents a training instance from a pair of anchors u and v such that $\mathbf{o}_u = \mathbf{o}_s$ and $\mathbf{o}_v = \mathbf{o}_t$. The cell $\mathbf{X}_{i,j}$ is the number of anchored predicate paths P_j that are anchored by u and v. Class labels indicate if the training instance is connected by the predicate of interest or not. The goal of path selection is to create a new $n \times m'$ matrix $\mathbf{X}'$, where m' contains only the paths/features with the most discriminative power. This is achieved by a feature selection function:

$$\mathbf{X}' = f(\mathbf{X}, \mathbf{w}, \delta) = \mathbf{X}_{1:n, \{j | j \in 1:m, w_j \geq \delta\}}, \tag{5}$$

where $\mathbf{w}$ is an m-dimensional feature importance vector, and δ is an importance threshold.

The importance $w_j \in \mathbf{w}$ of a predicate path $P_j \in \mathbf{P}$ is measured using the information gain of $\mathbf{X}_{:,j}$ and $\mathbf{y}$:

$$I(\mathbf{X}_{:,j} : \mathbf{y}) = \sum_{x_{i,j} \in \mathbf{X}_{:,j}} \sum_{y_i \in \mathbf{y}} p(x_{i,j}) p(y_i) \log \left(\frac{p(x_{i,j}, y_i)}{p(x_{i,j}) p(y_i)} \right), \tag{6}$$

where $\mathbf{X}_{:,j}$ denotes the column vector of feature j, $\mathbf{y}$ represents the corresponding label vector, and $x_{i,j}$ is the data value of the cell at $\mathbf{X}_{i,j}$ [31]. In order to reduce the rank of $\mathbf{X}$ we set δ empirically.

With the discriminative predicate paths extracted, pruned and represented in $\mathbf{X}'$, we train a standard logistic regression model and use it to validate the original statement of fact.

3.4. Fact interpretation

Although the discriminative paths are used for fact checking, there is no guarantee that all of the predicate paths actually describe intuitive or important attributes of the given statement of fact. For example, the fact checking model trained with the statement of fact (Springfield, capitalOf, Illinois) contains many spurious predicate paths like $\langle \text{location}^{-1}, \text{location} \rangle$ and $\langle \text{deathPlace}^{-1}, \text{deathPlace} \rangle$. These predicate paths indicate that "the capital is located in the state" and that "a city is the death place of a person who died in that state", which are indeed accurate in their respective instances, but not very descriptive of what capitalOf actually means. Although these supportive predicate paths seem superfluous they may actually be used to help identify false statements, according on their learned regression weights. However, these spurious statements should probably not be included in any "definition" of a given fact; instead, we want *human-interpretable* definitions to consist of only the most important predicates. In other words, we need to find those important discriminative predicate paths that define *only* the predicate in question.

To do this, we sort all of the extracted predicates by their importance $\mathbf{w}$ and construct an ordered list $P_x \prec P_y \prec \dots$, where $w_x \geq w_y$.

Table 1
Top discriminative paths defining capitalOf, ordered by **w**.

Rank	Meta path Π
1	$\{\text{city, settlement}\} \xrightarrow{\text{location}^{-1}} \{\text{state agency}\} \xrightarrow{\text{location}} \{\text{state}\}$
2	$\{\text{city, settlement}\} \xrightarrow{\text{deathPlace}^{-1}} \{\text{person}\} \xrightarrow{\text{deathPlace}} \{\text{state}\}$
3	$\{\text{city, settlement}\} \xrightarrow{\text{headquarter}^{-1}} \{\text{state agency}\} \xrightarrow{\text{jurisdiction}} \{\text{state}\}$
4	$\{\text{city, settlement}\} \xrightarrow{\text{location}^{-1}} \{\text{state agency}\} \xrightarrow{\text{jurisdiction}} \{\text{state}\}$
5	$\{\text{settlement}\} \xrightarrow{\text{location}^{-1}} \{\text{state agency}\} \xrightarrow{\text{jurisdiction}} \{\text{state}\}$
	Anchored predicate path **D**
1	$\langle \text{headquarter}^{-1},\ \text{jurisdiction}\rangle$
2	$\langle \text{location}^{-1},\ \text{jurisdiction}\rangle$
3	$\langle \text{headquarter}^{-1},\ \text{regionServed}\rangle$
4	$\langle \text{garrison}^{-1},\ \text{country}\rangle$
5	$\langle \text{deathPlace}^{-1},\ \text{deathPlace}\rangle$
	Discriminative anchored predicate path $\mathbf{D}^*$
1	$\langle \text{headquarter}^{-1},\ \text{jurisdiction}\rangle$
2	$\langle \text{location}^{-1},\ \text{jurisdiction}\rangle$
3	$\langle \text{garrison}^{-1},\ \text{country}\rangle$
4	$\langle \text{headquarter}^{-1},\ \text{parentOrganisation}\rangle$
5	$\langle \text{location}^{-1},\ \text{parentOrganisation}\rangle$

After ordering the predicate paths, we remove unnecessary and verbose predicate paths by

$$\mathbf{D}^* = \left\{ P | P \in \mathbf{D} \setminus \left\{ P_j | P_j \in \mathbf{P}^-, \sum_{i=0}^{i=n} \mathbf{X}_{i,j} \geq \theta \right\} \right\}, \tag{7}$$

where θ is an importance threshold chosen empirically and varies between 10 and 20. As a result of this function, the set of important discriminative predicate paths $\mathbf{D}^*$ contains the specific definers of the provided predicate. The top five discriminative predicate paths for capitalOf are illustrated in Table 1.

4. Experiments

In this section we report the results of two tasks: 1) fact checking and 2) definition interpretation using thousands of fact statements from eight different test cases on two large, real world knowledge graphs: DBpedia [21] and SemMedDB [28]. Before we present the results, we describe the datasets, alternative approaches and the experimental setup.

4.1. Data set

The fact checking model requires a knowledge graph as input. For these experiments we generated two large heterogeneous information networks from two widely used knowledge bases. In order to construct a heterogeneous multigraph from each knowledge base, we converted each RDF triple (subject, predicate, object) into a directed edge subject $\xrightarrow{\text{predicate}}$ object in $\mathcal{G}$, we further combined entities with same name or identifier into a single entity node in the final knowledge graph. The statistics of two resulting knowledge graphs are shown in Table 2.

DBpedia. DBpedia is a community project which converts facts extracted from Wikipedia into knowledge base triples that follow Semantic Web and Linked Data standards. The resultant knowledge base is split into several components such as infobox-facts (*i.e.*, (*s, p, t*)), and entity type mappings (*i.e.*, $\psi(u) = \mathbf{o}_u$).

From this knowledge base, we use the infobox-facts and article ontology from the April 2014 DBpedia knowledge base to create nodes and edges. We did not include article content, such as text and hyperlinks, in the knowledge graph because it was not the focus of this work.

SemMedDB. The Semantic MEDLINE Database contains 82 million triples extracted from biomedical text using the SemRep extractor [32]. Unlike DBpedia, which does not have duplicate triples, SemMedDB contains a large number of duplicate records and uses the amount of duplication as a measure of credibility. For example, an incorrect statement (Chicago, isA, country) appears only once in the SemMedDB knowledge base, while the correct statement (Chicago, isA, city) appears 10 times. Interestingly, although there are 82 million edges in SemMedDB, only 20.9% of the edges are unique.

We use the June 2012 version of SemMedDB and translate it to a knowledge graph in the same way as with DBpedia. We do not remove any duplicate edges because comparable algorithms often work better on multigraphs; Adamic/Adar, for example, may leverage duplicate edge information to improve accuracy.

4.2. Experiment setting

We view the fact checking task as a type of link prediction problem because a fact statement (*s, p, t*) can be naturally considered as an edge $s \xrightarrow{p} t$ in a given knowledge graph $\mathcal{G}$. The probability that an unknown statement of fact $s \xrightarrow{p} t$ is true is equivalent to the probability that the edge $s \xrightarrow{p} t$ is missing in $\mathcal{G}$. To test the ability of our method to validate missing facts and unseen relations, we remove all edges labelled by the given predicate p and perform fact checking on the modified multigraph $\mathcal{G}' = \mathcal{G} - p$.

All experiments are performed using 10-fold cross validation. The source code of our method and the comparison algorithms, including data preprocessing tools, can be found at https://github.com/nddsg/KGMiner.

We compared our fact checking algorithm with nine alternative approaches including Adamic/Adar (AA) [4], Preferential Attachment (PA) [5], Katz [6] with $k = 3$ and $\beta = 0.05$ as recommended by Kleinberg and Liben-Nowell [3], Semantic Proximity (SP) [2] with $k = 3$, personalized PageRank (PPR) [7] with damping factor $d = 0.15$, SimRank [33], Path-Constrained Random Walk (PCRW) [14], AMIE [19], and TransE [10].

In order to run SimRank on the large knowledge graphs, we implemented Kusumoto and Maehara's SimRank approximation [34] with $c = 0.8$, $T = 100$ and $R = 10^4$ set according to the values in their original work.

We use Lin et al.'s TransE implementation [11] with $\lambda = 0.01$, $\gamma = 1$, and $d = L_2$ according to Bordes et al. [10]. The feature dimension is set to 100 and the training phrase stops after 1000 epochs.

Although AMIE is designed for association rule mining, in this work we employ it as a link prediction method by assuming that the given statement of fact (*s, p, t*) is true if and only if at least one association rule found by AMIE connects s and t in the graph $\mathcal{G}'$.

As for meta path based methods, such as PCRW, we use the association rule mined by AMIE as the input rather than hand-labeled meta paths. We chose to use AMIE instead of the discovered meta paths from Meng et al. [20] because the implementation of AMIE is publicly available. Unfortunately, PathSim [12] requires input meta paths to be symmetric, *i.e.*, $a \to b \to a$ or $a \to b \to c \to b \to a$, but the rules mined by AMIE are very rarely symmetric in our test cases because the (*s, p, t*) endpoints typically have different entity-type labels; therefore we cannot compare our method with PathSim and other symmetric-only algorithms.

Due to the large size of the knowledge graphs, it is impractical to run AMIE to completion. In these experiments, we executed AMIE for 2690 CPU hours on DBpedia and 1190 CPU hours on SemMedDB. The number of AMIE-mined rules on the knowledge graphs is 1326 and 5188 respectively.

We also tried other statistical relational learning models including RESCAL [8] and NTN [9] but the publicly available

Table 2
Statistics of knowledge graph datasets.

$\mathcal{G}$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{O}\rvert$	$\lvert\mathbf{o}\rvert = 0$	$\lvert\mathbf{o}\rvert > 1$
DBpedia [21]	4,743,012	27,313,477	671	451	524,889	3,313,257
SemMedDB [28]	282,934	82,239,653	58	132	137	73,936

implementations were either incapable of dealing with the huge data sets we use in this work or returned incomprehensible results. We do not compare search engine based models [35] because, unlike the original authors, we do not have access to search engine APIs to the extent necessary to carry out a proper comparison.

4.3. Test cases

Here we briefly describe the test cases we use for fact checking. Each test case is constructed to be as difficult as possible.

CapitalOf #1. Check the capital of a US state. In this task we check $\{\text{city}\} \xrightarrow{\text{capitalOf}} \{\text{state}\}$ for the top five most populous cities in all 50 US states. In nine instances the capital city is not in the set of top five most populous cities of a state, in these cases we further include the capital city in the test set thereby checking a total of $5 \times 50 + 9 = 259$ statements of fact with 50 true instances, and 209 false instances.

CapitalOf #2. Check the capital of a US state. In this task we check $\{\text{city}\} \xrightarrow{\text{capitalOf}} \{\text{state}\}$ by creating 200 incorrect random matchings of capitals to states. For example, we check if Springfield, the actual capital of Illinois, is the capital of 4 other states. This random assignment results 250 statements of fact with 50 true instances and 200 false instances.

US civil war. Check the commander of a US Civil War battle. In this task we check $\{\text{person}\} \xrightarrow{\text{commanderOf}} \{\text{battle}\}$ by creating 584 incorrect random matchings of civil war commanders to civil war battles, as well as 126 true statements about the Union and Confederate commanders of Class A (*i.e.*, decisive) US Civil War battles.

Company CEO. Check CEO of a company. In this task we check $\{\text{person}\} \xrightarrow{\text{keyPerson}} \{\text{company}\}$ by creating 1025 incorrect random matchings of CEOs to companies, as well as true statements about the CEOs of the 205 notable companies in the Wikipedia List of chief executive officers.

NYT Bestseller. Check author of a book. In this task we check $\{\text{person}\} \xrightarrow{\text{author}} \{\text{book}\}$ by creating 465 incorrect random pairs of authors to books, as well as true statements about the 63 authors who wrote 93 books that appeared on the New York Times bestseller list between 2010-2015.

US Vice-President. Check vice president of a president. In this task we check $\{\text{person}\} \xrightarrow{\text{vicePresidentOf}} \{\text{person}\}$ by creating 227 incorrect pairs of vice-presidents to presidents, as well as true statements about the 47 vice-presidents of the United States.

Disease. Check if amino acid, peptide, or protein causes a disease or syndrome. In this task we check $\{\text{aapp}\} \xrightarrow{\text{causes}} \{\text{dsyn}\}$ where aapp and dsyn are types in SemMedDB corresponding to amino acid, peptide, or protein and disease or syndrome respectively. We do this by creating 457 incorrect statements, as well as 100 true statements.

Cell. Check if a gene causes a certain cell function. In this task we check $\{\text{gngm}\} \xrightarrow{\text{causes}} \{\text{celf}\}$ where gngm and celf are types in SemMedDB corresponding to gene or genome and cell function respectively. We do this by creating 435 incorrect statements, as well as 99 true statements.

These eight test cases listed above represent a 20/80 true to false label split of instances. We will experiment with different label ratios in later experiments.

4.4. Predicate path analysis

Earlier we argued in favor of anchored predicate paths over the use of meta paths. Recall that the reasoning behind this choice is that the anchored predicate paths are less restrictive than meta paths, which require one or more type-labels for every entity in the path, whereas anchored predicate paths only require type-labels for entities on the endpoints of the path.

Fig. 3 shows the results of a comparison between meta paths $\mathbf{\Pi}$ and discriminative anchored predicate paths $\mathbf{D}$ on the six DBpedia tasks. We find that the performance of anchored predicate paths (solid circle) are comparable to meta paths (solid square) despite having a much smaller feature set.

We also constructed a subset of meta paths and anchored predicate paths by computing the information gain of each path, sorting by the information gain and chosing the top k that maximizes the area under the receiver operator characteristic (AUROC) score. The empirical result of the meta path subset (hollow square) and anchored predicate path subset (hollow circle) is also illustrated in Fig. 3.

We find that, even if we select the most informative paths, the feature set generated by meta paths is typically bigger than the set of predicate paths, but results in similar performance. Moreover, the 165, 331 unique meta paths extracted from SemMedDB that match (gngm,causes,celf) was too large to work with effectively. On the other hand, the number of unique anchored predicate paths totalled only 1066.

Apart from the increase in feature set size, the use of meta paths also reduced the understandability of the results. The top discriminative paths found from the $\{\text{city}\} \xrightarrow{\text{capitalOf}} \{\text{state}\}$ example in Table 1 from earlier show that the re-ranked anchored predicate paths $\mathbf{D}^*$ are more intuitive than the discovered meta paths $\mathbf{\Pi}$. Unfortunately, "intutive"-ness is a difficult concept to test fully, so we leave a complete test of the understandability of predicate paths and meta paths as a matter for future work.

4.5. Fact checking

In Table 3 we compare the proposed fact-checking algorithm with nine other link prediction, knowledge base completion and data mining algorithms on data from DBpedia and SemMedDB.

In the **CapitalOf #1** task, where the true capitalOf statements are mixed with false statements that match the largestCity predicate, the proposed method is shown to significantly outperform other methods. Recall that the set of discriminative predicate paths represent a sort of "definition" of the given predicate that is used as a model for fact checking. The top five most discriminative predicate paths for the **CapitalOf #1** task were originally shown in Table 1, and the top two most discriminative predicate paths are shown in Table 5.

The Adamic/Adar, Preferential Attachment, and Katz models performed very poorly in this example because the features that these purely topographical models rely on most strongly connects the largest city with the state. Unfortunately, only 17 US capital-cities are also the largest city in their state resulting in very poor performance for the topographical models.

Tasks in which the negatively-labeled data is randomly generated, as in **CapitalOf #2** for example, are easier for topological

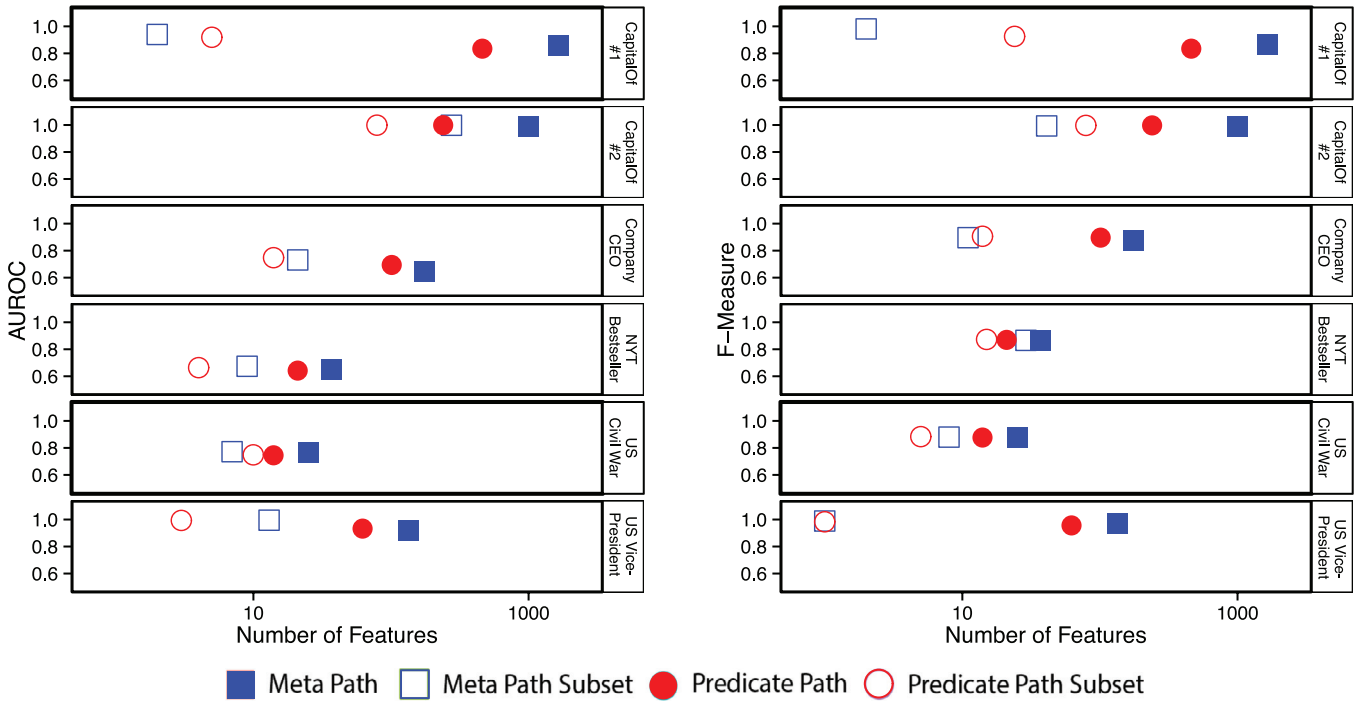

Fig. 3. Fact-checking performance on DBpedia. Solid and hollow symbols denote original and selected best performance subset respectively. This figure demonstrates that anchored predicate paths contain fewer features but have similar performance compared to sets of meta paths. Top left is better.

Table 3
Result of fact checking test cases. The score is the area under ROC curve score computed by logistic regression with 10-fold cross validation. * means the value is missing due to the large size of feature set (Section 3.2). All test cases are explained in Section 4.3.

Algorithm	CapitalOf #1	CapitalOf #2	Company CEO	NYT Bestseller	US Civil War	US Vice-President	Disease	Cell
Adamic/Adar [4]	0.387	0.962	0.665	0.650	0.642	0.795	0.671	0.755
Semantic proximity [2]	0.706	0.978	0.614	0.641	0.582	0.805	0.871	0.840
Preferential attachment [5]	0.396	0.516	0.498	0.526	0.599	0.474	0.563	0.755
Katz [6]	0.370	0.976	0.600	0.623	0.585	0.791	0.763	0.832
SimRank [33]	0.553	0.976	**0.824**	**0.695**	0.685	0.912	0.809	0.749
AMIE [19]	0.550	0.500	0.669	0.520	0.659	0.987	0.889	0.898
Personalized PageRank [7]	0.535	0.535	0.579	0.529	0.488	0.683	0.827	0.885
Path-constrained random walk [14]	0.550	0.500	0.542	0.486	0.488	0.672	0.911	0.765
TransE [10]	0.655	0.775	0.728	0.601	0.612	0.520	0.532	0.620
Discriminative predicate path (D) Count	0.920	**0.999**	0.747	0.664	0.749	0.993	**0.941**	**0.928**
Discriminative meta path (Π) Count	**0.940**	0.998	0.731	0.674	**0.772**	**0.995**	*	*

models because, in many cases, the true-labeled statement is the one that is the best connected (especially when compared to random statements). Interestingly, SimRank performs slightly better than our model on the **Company CEO** and **NYT Bestseller** tasks. This is most probably because of the high connectivity between the path anchors, and because of the lack of meta path variation, *e.g.*, book authors and company CEOs have relatively few alternate paths that are suitable defining the given statement.

Despite being a deep learning, word2vec-like knowledge base completion system, TransE does not perform well in these tasks. This may due to the large knowledge graph we used in this work, but may also be because TransE is not designed to accept duplicated edges, which seems to help identify factual relations especially in the SemMedDB dataset.

Recall that these results use a true/false label ratio of 20/80 to simulate real-world fact checking scenarios where the proportion of false statements are significantly larger than true statements. This is not to say that there are more false statements in real-life, just that there are many more possible false statements than there are true statements. With this in mind, we further test the robustness of our model under various true/false label proportions. Fig. 4 illustrates the results of these tests where the discriminative predicate path performance (solid blue circle) is found to be relatively invariant to the percentage of labeled data as it changes from 10% positively-labeled to 90% positively labeled.

Apart from the accuracy and robustness tests above, we also analyze the amount of time that each algorithm uses while calculating the score for a single statement of fact, *i.e.*, the time it takes to calculate $\mathbf{X}'$. The 8 tasks have similar computational complexity, so we combine the execution times and present the mean average in Table 4. We find that our method (labeled PredPath), although slower than shared neighbor methods and untyped path based model such as Adamic/Adar, Preferential Attachment, SimRank and Katz, has an execution time comparable to heterogeneous path-based method Semantic Proximity, and is faster than stochastic models like TransE, Path Constrained Random Walk (PCRW), personalized PageRank (PPR), and the fast, approximate version of SimRank.

4.6. Statement interpretation

So far we have seen that the predicate path model presented in this work is able to accurately and quickly check the validity of statements of fact. Perhaps the most important contribution of

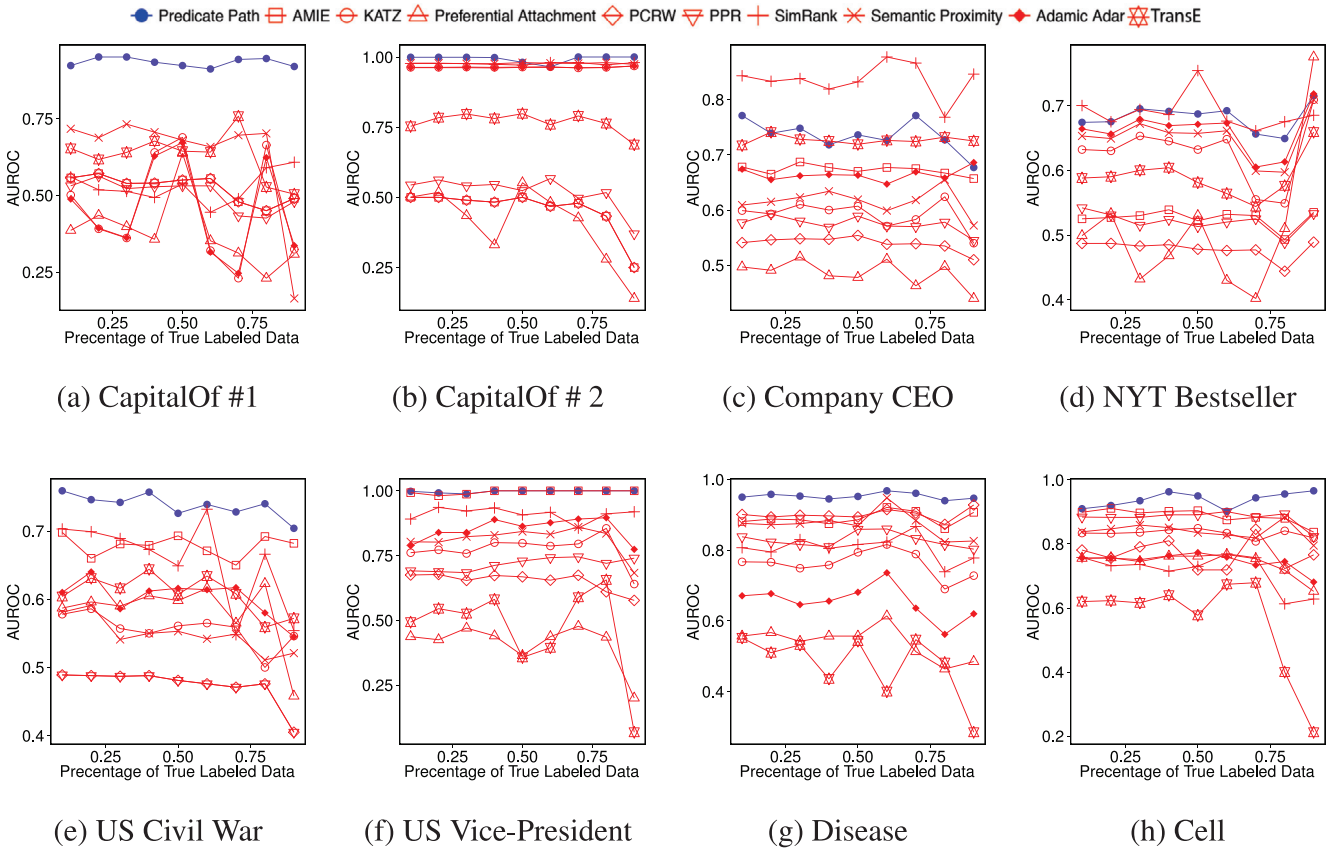

Fig. 4. Fact checking performance with different true to false ratio of labeled data.

Table 4
Seconds each algorithm consumes for feature generation. The value represents the average feature generation time of each statement. The execution time of AMIE and TransE includes the average time spent on association rule mining and embedding learning respectively. PredPath is faster than other typed methods.

Model	Adamic/Adar	AMIE	Katz	PA	PCRW	PPR	PredPath	Semantic Proximity	SimRank	TransE
Time(second)	0.04	3194.88	0.57	0.03	77.16	446.95	1.00	0.94	0.86	368.32

Table 5
Top discriminative paths found by proposed method that are missing in AMIE. Predicate path anchors are for illustrative purposes and do not represent the full entity label set.

Task	Top discriminative path missed by AMIE	
CapitalOf #1	{city}	⟨headquarter^{-1}, jurisdiction⟩⟨location^{-1}, jurisdiction⟩ {state}
CapitalOf #2	{city}	⟨location^{-1}, location⟩⟨isPartOf⟩ {state}
Company CEO	{person}	⟨employer⟩ {company}
US Civil War	{person}	⟨notable commander^{-1}, takePartIn⟩ {battle}
NYT Bestseller	{person}	⟨notable work, previous work⟩⟨notable work, subsequent work⟩ {book}
US President	{vice president}	⟨successor, president^{-1}⟩ {president}
Disease	{aapp}	⟨associatedWith, isA⟩⟨stimulates, affects⟩ {dsyn}
Cell	{gngm}	⟨comparedWith, negativeAssociatedWith⟩ {celf}

this work, is not just in the ability to check facts, but rather in the ability to explain the meaning of some relationship between entities. Current progress in knowledge and reasoning in artificial intelligence is limited by our inability to understand the meaning behind data. For instance, although neural network-based technologies, like TransE, can produce accurate results, their learning mechanism does not provide an easily interpretable explanation for their answers. In contrast, our model explicitly provides a commonsense reason as to why a fact is deemed to be true or false. Table 5 shows some of the top predicate paths that are found by our model; we argue that they are generally intuitive and describe at least one key property about the given statement of fact.

One particularly interesting finding from Table 5 is the predicate path: {vice president} ⟨successor, president^{-1}⟩ {president}, which encodes, for example, that eventual-President Andrew Johnson succeeded Hannibal Hamlin as the second vice president under President Abraham Lincoln. Indeed, eight US presidents have had two or more vice presidents (one succeeding the other) that have gone on to become president, meaning that the US constitution allows for the possibility to replace one vice president with another – a little known, yet valid part of the definition of what it means to be the US vice-president.

5. Related work

Discriminative path generation. Although meta paths have been used in many methods such as similarity search, clustering, semi-supervised learning, and link prediction [12–14,26,27], these algorithms either require human annotated meta paths [15] or enumerate all possible meta paths in the graph. Recently efforts have been made to meta path discovery and association rule mining in concept graphs [16], but most of the approaches have their own limitations. Meng et al., proposed a meta path generation algorithm that prunes the enumeration space by logistic regression, but this approach is prone to premature rejection and may miss important discriminative paths [20]. AMIE [19] is a global association rule mining algorithm which can not mine personalized, *i.e.*, context dependent, association rules as we shown in Sections 1 and 4. Abedjan and Naumann proposed a predicate expansion algorithm [18] which can find predicate synonyms, but cannot find predicate paths that have discriminating power. The proposed discriminative path discovery framework in this work extracts meta paths and predicate paths from the graph directly with given endpoints, therefore our framework will not miss important predicate paths in the graph and is context-sensitive.

Fact checking. With the large volume of data generated every day, the number of unverified statements begets the need for automated fact checking [36,37]. To that end, many researchers have focused on automated fact checking in recent years. Finn et al. introduced a new interactive tool to help human fact checkers determine the quality of a statement by extracting the propagation of facts on Twitter [38]. Ennals et al. created a crowd-sourced platform that highlight disputed claims [39]. Kwok et al. proposed a ensemble method utilizing the result from search engines to check a given statement [40]. Hassan et al. proposed a numerical fact monitor, called FactWatcher [41], that uses an append only database and certain skyline operators [42,43], but FactWatcher is not applicable to knowledge graphs or nonnumerical statements. The True Knowledge System [44] validates a statement of the fact using 1500 predefined and user provided association rules; unfortunately, this means that it is impossible to check a statement that does not already have a predefined association rule within True Knowledge. Ciampaglia et al. published a knowledge graph based fact checking algorithm [2] utilizing node connectivity, but does not take advantage of the type-labels in the heterogeneous information networks. Recently, Guu et al. published a question answering algorithm that converts a given question into a vector space model to find the answer [45], but, like neural network based models [46], the learned model is generally uninterpretable. Li et al. proposed T-verifier, a search engine based fact checker [35], but such approach needs extensive access to search engine APIs which is not easy to gain. Knowledge graph completion methods, such as TransE [10], TransR [11], and NTN [9] is not ideal for fact checking in large knowledge graphs because the slow convergence rate.

Link prediction. Apart from classic homogeneous link prediction methods, such as Adamic/Adar [4], SimRank [33], Katz [6], Preferential Attachment [5], and Personalized PageRank [7] *etc.*, many heterogeneous methods have been developed to leverage the rich information in heterogeneous information networks. Heterogeneous graphlet base methods [47] predict the relation between two endpoints by counting the occurrence of certain heterogeneous motifs, which are not applicable to complex knowledge graphs due to the exponential number of possible graph motifs. Other heterogeneous link prediction methods that adapt from classic homogeneous algorithms, HeteSim [27] and PCRW [14], depend on human annotated meta paths. PathSim [12], a heterogeneous similarity metric, also requires hand crafted and symmetric meta paths as the input. Recently Dong et al., proposed a heterogeneous link prediction algorithm based on coupled networks, but also needs human annotated meta paths as input [48]. In contrast, this work automatically discovers the important meta paths and predicate paths that related to the given statement of fact.

6. Conclusions and future work

We presented a fact checking framework for knowledge graphs that discovers the definition of a given statement of fact by extracting discriminative predicate paths from the knowledge graph, and uses the discovered model to validate the truthfulness of the given statement.

To evaluate the proposed method, we checked the veracity of several thousand statements across eight different tasks on DBpedia and SemMedDB. We found that our framework was the all around best in terms of fact-checking performance and has a running time similar to existing models. We further tested the robustness of our algorithm by examining different ratios of true to false information and found that our framework was generally invariant to the class ratio. Finally, we showed that the proposed framework can discover interpretable and informative discriminative paths that are missed by other methods.

As this framework is the first of its kind, we leave much as a matter for future work. The next steps that are immediately obvious to us include extensions to this framework that perform (1) predicate identification, (2) enhanced entity representation, and (3) fact qualification. Predicate identification is the most natural extension to this framework wherein unnamed and unknown relationships can be implied through transitivity; for example, if a set of highly discriminative predicate paths between two sets of entities x and y exists, along with another set of highly discriminative predicate paths between y and a third set of entities z, then we may be able to encode some special, transitive relationship between the entities in x with the respective entities in z. Because we are encoding the meaning behind relationships and between entities, it is likely that we will be able to find natural implications that arise in arithmetic combinations of entities such as the canonical King-man+women=Queen, but with a human-interpretable representation for each operator that is not present in current vector-based models. Finally, we should be able to use mismatches and errors in our model to qualify some statement of fact; for example, the statement “Rome is the capital of the Roman Empire” is only true before 323 CE, after which the capital was changed to Constantinople.

Acknowledgments

This work is sponsored by an AFOSR grant FA9550-15-1-0003, and a John Templeton Foundation grant FP053369-M.

References

[1] J. Swift, The examiner. 15.
[2] G.L. Ciampaglia, P. Shiralkar, L.M. Rocha, J. Bollen, F. Menczer, A. Flammini, Computational fact checking from knowledge networks, PLoS ONE 10 (6) (2015).
[3] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, JASIST 58 (7) (2007) 1019–1031.
[4] L.A. Adamic, E. Adar, Friends and neighbors on the web, Soc. Netw. 25 (3) (2003) 211–230.
[5] A.L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[6] L. Katz, A new status index derived from sociometric analysis, Psychometrika 18 (1) (1953) 39–43.
[7] T.H. Haveliwala, Topic-sensitive pagerank, in: WWW, 2002, pp. 517–526.
[8] M. Nickel, V. Tresp, H.P. Kriegel, A three-way model for collective learning on multi-relational data, in: ICML, 2011, pp. 809–816.
[9] R. Socher, D. Chen, C.D. Manning, A.Y. Ng, Reasoning with neural tensor networks for knowledge base completion, in: NIPS, 2013, pp. 926–934.

[10] A. Bordes, N. Usunier, A. García-Durán, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: NIPS, 2013, pp. 2787–2795.
[11] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: AAAI, 2015, pp. 2181–2187.
[12] Y. Sun, J. Han, X. Yan, P.S. Yu, T. Wu, Pathsim: meta path-based top-k similarity search in heterogeneous information networks, in: VLDB, 2011, pp. 992–1003.
[13] M. Zhao, T.W. Chow, Z. Zhang, B. Li, Automatic image annotation via compact graph based semi-supervised learning, Know. Based Sys. 76 (2015) 148–165.
[14] N. Lao, W.W. Cohen, Relational retrieval using a combination of path-constrained random walks, Mach. Learn. 81 (1) (2010) 53–67.
[15] X. Fu, G. Qi, Y. Zhang, Z. Zhou, Graph-based approaches to debugging and revision of terminologies in DL-lite, Know. Based Sys. 100 (2016) 1–12.
[16] P.P. Ruiz, B.K. Foguem, B. Grabot, Generating knowledge in maintenance from experience feedback, Know. Based Sys. 68 (2014) 4–20.
[17] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: AAAI, 2014, pp. 1112–1119.
[18] Z. Abedjan, F. Naumann, Synonym analysis for predicate expansion, in: ESWC, 2013, pp. 140–154.
[19] L.A. Galárraga, C. Teflioudi, K. Hose, F. Suchanek, AMIE: association rule mining under incomplete evidence in ontological knowledge bases, in: WWW, 2013, pp. 413–422.
[20] C. Meng, R. Cheng, S. Maniu, P. Senellart, W. Zhang, Discovering meta-paths in large heterogeneous information networks, in: WWW, 2015, pp. 754–764.
[21] J. Lehmann, R. Isele, M. Jakob, DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia, Semantic Web 5 (1) (2014) 167–195.
[22] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.M. Popescu, T. Shaked, S. Soderland, D.S. Weld, A. Yates, Web-scale information extraction in knowitall: (preliminary results), in: WWW, 2004, pp. 100–110.
[23] N. Hassan, C. Li, M. Tremayne, Detecting check-worthy factual claims in presidential debates, in: CIKM, 2015, pp. 1835–1838.
[24] Y. Wu, P.K. Agarwal, C. Li, J. Yang, C. Yu, Toward computational fact-checking, in: VLDB, 2014, pp. 589–600.
[25] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs: from multi-relational link prediction to automated knowledge graph construction, Proc. IEEE 104 (1) (2016) 11–33.
[26] Y. Sun, B. Norick, J. Han, X. Yan, P.S. Yu, X. Yu, Integrating meta-path selection with user-guided object clustering in heterogeneous information networks, in: KDD, 2012, p. 11.
[27] C. Shi, X. Kong, Y. Huang, P.S. Yu, B. Wu, Hetesim: a general framework for relevance measure in heterogeneous networks, TKDE 26 (10) (2014) 2479–2492.
[28] H. Kilicoglu, D. Shin, M. Fiszman, G. Rosemblat, T.C. Rindflesch, SemmedDB: a pubmed-scale repository of biomedical semantic predications, Bioinformatics 28 (23) (2012) 3158–3160.
[29] M. Ley, DBLP: some lessons learned, in: VLDB, 2009, pp. 1493–1500.
[30] M. Levandowsky, D. Winter, Distance between sets, Nature 234 (5323) (1971) 34–35.
[31] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
[32] T.C. Rindflesch, M. Fiszman, The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text, Biomed. Inf. 36 (6) (2003) 462–477.
[33] G. Jeh, J. Widom, Simrank: a measure of structural context similarity, in: KDD, 2002, pp. 538–543.
[34] M. Kusumoto, T. Maehara, K.i. Kawarabayashi, Scalable similarity search for simrank, in: SIGMOD, 2014, pp. 325–336.
[35] X. Li, W. Meng, C.T. Yu, T-verifier: Verifying truthfulness of fact statements, in: ICDE,IEEE, 2011, pp. 63–74.
[36] L. Graves, T. Glaisyer, The Fact-Checking Universe in Spring, New America, 2012.
[37] S. Cohen, J.T. Hamilton, F. Turner, Computational journalism, CACM 54 (10) (2011) 66–71.
[38] S. Finn, P. Metaxas, E. Mustafaraj, M. O'Keefe, L. Tang, S. Tang, L. Zeng, TRAILS: a system for monitoring the propagation of rumors on twitter, in: Computational Journalism, 2014, pp. 1–5.
[39] R. Ennals, B. Trushkowsky, J.M. Agosta, Highlighting disputed claims on the web, in: WWW, 2010, pp. 341–350.
[40] C. Kwok, O. Etzioni, D.S. Weld, Scaling question answering to the web, TOIS 19 (3) (2001) 242–262.
[41] N. Hassan, A. Sultana, Y. Wu, G. Zhang, C. Li, J. Yang, C. Yu, Data in, fact out: automated monitoring of facts by factwatcher, in: VLDB, 2014, pp. 1557–1560.
[42] Y. Wu, P.K. Agarwal, C. Li, J. Yang, C. Yu, On "one of the few" objects, in: KDD, 2012, pp. 1487–1495.
[43] X. Jiang, C. Li, P. Luo, M. Wang, Y. Yu, Prominent streak discovery in sequence data, in: KDD, 2011, pp. 1280–1288.
[44] W. Tunstall-Pedoe, True knowledge: open-domain question answering using structured knowledge and inference, AI Mag. 31 (3) (2010) 80–92.
[45] K. Guu, J. Miller, P. Liang, Traversing knowledge graphs in vector space, in: EMNLP, 2015, pp. 318–327.
[46] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: NIPS, 2013, pp. 3111–3119.
[47] R.N. Lichtenwalter, N.V. Chawla, Vertex collocation profiles: subgraph counting for link analysis and prediction, in: WWW, 2012, pp. 1019–1028.
[48] Y. Dong, J. Zhang, J. Tang, N.V. Chawla, B. Wang, Coupledlp: link prediction in coupled networks, in: KDD, 2015, pp. 199–208.