STAIRS 2014
U. Endriss and J. Leite (Eds.)

doi:10.3233/978-1-61499-421-3-111

Semantifying Triples from Open Information Extraction Systems

Arnab DUTTA [1], Christian MEILICKE and Heiner STUCKENSCHMIDT
Data and Web Science Research Group, University of Mannheim, Germany

Abstract. The last few years have witnessed some remarkable success of the state-of-the art unsupervised knowledge extraction systems like NELL and REVERB. These systems are gifted with typically web-scale coverage but are often plagued with ambiguity due to lack of proper schema or unique identifiers for the instances. This classifies them apart from extraction systems like DBPEDIA, YAGO or FREEBASE which have precise information content but have smaller coverage. In this work we bring together the former to enrich the later with high precision novel facts and present a statistical approach to discover new knowledge. In particular, we semantify NELL triples using DBPEDIA.

Keywords. open information extraction, information integration, knowledge generation, statistical modeling.

1. Introduction

With the growing popularity of unsupervised techniques of knowledge extraction from text, web corpora, there is an advent of a new genre of extraction systems commonly know as *open* information extraction (OIE) systems. "Open" in the sense, they are not limited to Wikipedia or any specific resource but the whole of web. Systems like NELL [5], REVERB [11] have gained a quick prominence marked by their web scale coverage and huge fact base. However, such systems often lack a schema and hence, is difficult to correctly disambiguate entities from the OIE fact base. On the other hand, knowledge bases (KBs) like DBPEDIA [1], YAGO [19], FREEBASE [2] mark another class of extraction systems which are of higher quality, precise and maintain a well-structured schema but at the expense of a poor coverage (often restricted only to Wikipedia).

There is considerable potential in exploiting the data maintained by OIE for analyzing, reasoning about, and discovering novel facts and generation of web search engines [9]. In this work we integrate these two broad domains in a symbiotic fashion; OIE systems exploit the clean ontological structure of closed IE systems, thereby imparting uniqueness to its entities; and the closed IE systems enrich themselves from the broader coverage of the OIE. We present a methodology to automatically find new DBPEDIA triples by exploiting the information content in NELL. To achieve this, we need a synergistic integration of solving the three major sub tasks:

[1]Corresponding Author: Arnab Dutta, University of Mannheim, 68159 Mannheim, Germany; E-mail: arnab@informatik.uni-mannheim.de

1. how to precisely find the references of the instances within an OIE triple, to a closed KB instance (DBPEDIA in our case). This resembles with the task of entity linking where a term in a text is linked to a KB entry. But the lack of any context for OIE triples sets the scenario different for us and deters us to use any of-the-shelf entity linking tool.
2. assuming we have correctly deciphered the references, how can we map the relationship within an OIE triple to an analogous closed domain property.
3. how can the instance and property matching interplay to enrich a closed KB.

Let us consider a NELL triple of the form *stadiumlocatedincity*(*riverfest*, *little_rock*), where two entities are in a semantic relationship defined by *stadiumlocatedincity*. Even though we might have an intuitive understanding of the property, it is difficult to interpret the exact real world entities the terms are referring to; *little_rock* can refer to a range of cities in USA or a person or even a US naval ship. The problem gets more complicated since, unlike the well defined properties in DBPEDIA or YAGO, the OIE properties often lack strict domain and range definitions. This makes it difficult to determine what fits in as a subject and object. Note, NELL has its own schema but in an attempt to propose a general solution, we keep our approach agnostic to this information.

We propose a *learn-and-fill* approach to use the ambiguous triples in order to generate new facts. First, we map the NELL instances to DBPEDIA instances (Section 2.1) using a probabilistic approach. This is not the core contribution of this paper and has been already addressed before [7]. Second, we use the mappings to look for a semantic relationship (clean infobox properties with "/ontology" namespace) in DBPEDIA (Section 2.2) and use it as a likely predictor for the NELL property. For instance, we *learn* from other NELL triples with same property, like *stadiumlocatedincity*(*meyerson_symphony_center*, *dallas*) and many more that `location` can be a likely mapping. The final piece of the solution lies in integrating these two mapping solutions to generate new facts (Section 2.3). We can now *fill in* the original NELL triple with the learnt property `location` to generate a new facts like `location(Cincinnati_Bell/WEBN_Riverfest, Little_Rock,_Arkansas)`. We apply statistical techniques for the purpose and show its impact in generating high quality facts (Section 3).

2. Methodology

2.1. Mapping Instances

We map the individual subject and object occurring within a NELL triple to DBPEDIA instances. In this regard, we explore two different methods. The first and a naive approach is to look for the most frequent sense of the terms occurring in NELL triples by exploring intra-Wikipedia page links connecting anchor texts to their respective pages. A detailed analysis of this approach can be found in [8]. It is a simple approach, without exploiting any contextual information to improve the mappings.

As an improvement, we incorporate the type information of the mapped DBPEDIA instances. We initiate with the most frequent instance mappings from NELL to DBPEDIA and let them to determine the possible types of DBPEDIA instances *allowable* in the context of the given NELL property. Subsequently, the type information guides the mapping selection process again. These bootstrapped approach of selecting-and-refining is solved

using Markov Logic Networks [17] and leads to better results [7] than the naive way. We employ this technique to generate a refined set of hypotheses where every NELL instance eventually has atmost one mapping to a DBPEDIA instance.

2.2. Mapping Properties

This section presents our approach for mapping an OIE property to an analogous DBPEDIA property. For every NELL triple of the form $n_p(n_s, n_o)$ we map the subject (n_s) and object (n_o) individually to DBPEDIA instances d_s and d_o respectively (refined instance mappings from the probabilistic framework [7]). Using a DBPEDIA SPARQL endpoint, we query[2] for a possible property d_p involved in some triple of the form $d_p(d_s, d_o)$. If such a triple exists, then d_p can be considered as a likely DBPEDIA mapping of n_p. We apply this technique over all the NELL properties. Furthermore, we also consider inverse property mappings. We denote d_p^{inv} to be an inverse mapping for n_p, if the triple d_p^{inv} (d_o, d_s) exists in DBPEDIA. The methodology proposed in this work is applicable for both the two cases and we use d_p as a general notation for DBPEDIA property.

Likelihood Estimate: We want to estimate the likelihood of every possible mapping of n_p to one or more d_p. A naive frequency count of the mappings can give us a likelihood estimate. For instance, if NELL property *bookwriter* is mapped to `author` in k out of n cases and to `writer` in (n-k) out of n cases, then the likelihood of the mapping *bookwriter* to `author` is $\frac{k}{n}$, and to `writer` is $\frac{(n-k)}{n}$. Finally, selecting candidates above a threshold score, could be a simple solution. However, this approach suffers from two major drawbacks: first, any conceptually similar property (as in this case) might be eliminated out due to lack of sufficient evidence (low likelihood score); second, finding a correct threshold. An improved approach could be to incorporate the type information of the mapped DBPEDIA instances as well.

Formally, we define a set T_{n_p} consisting of NELL triples with property n_p. For each such triple, we collect the type of the mapped DBPEDIA subject and object, denoted by $dom(d_s)$ and $ran(d_o)$ respectively. Now, querying for triples like $d_p(d_s, d_o)$ can have the following possibilities:

1. returns an empty set, indicating absence of any d_p. This can happen if there is no such triple in DBPEDIA or the mapped instances are wrong at the first place.
2. returns a single possible value for d_p (e.g. *airportincity*(*helsinki_vantaa_airport*, *helsinki*) maps to `city`(`Helsinki_Airport`, `Helsinki`))
3. returns multiple values for d_p. (e.g. *airportincity*(*vnukovo*, *moscow*) maps to `city` (`Vnukovo_International_Airport`, `Moscow`) and `location`(`Vnukovo_International_Airport`, `Moscow`))

Case (2) and Case (3) are given an unified representation by framing discrete associations as, $\{n_p, d_p, dom(d_s)\}$ and $ran(d_s)$ (refer to Table 1). Hence, the example in Case (3) translates to {*airportincity*, `city`, `Airport`, `Place`} and {*airportincity*, `location`, `Airport`, `Place`}. Case (1) is not translated. All the associations for n_p thus formed is denoted as A_{n_p}. It is important to note that, $|T_{n_p}| \leq |A_{n_p}|; \forall n_p \in T_{n_p}$. A blank value ('-') is attributed to a missing instance type in DBPEDIA or non-mappabiliy of n_p due to reasons mentioned in Case (1) above.

[2] select ?d_p where {d_s ?d_p d_o}

Table 1. A snippet of the actual associations presenting a positive example with *airportincity* and a negative example with *agentcreated*. Missing value is marked with "-".

A_{n_p}	n_p	K_{n_p}	i	d_p^i	$dom(d_s^i)$	$ran(d_o^i)$	$\tau_{n_p}^i$	$\hat{\tau}$
$\wedge$			1	location	MilitaryStructure	-	150	1.21
$\ldots A_{airportincity} \ldots$	*airportincity*	55%	2	location	Airport	-	**0.86**	1.21
			3	isPartOf	Settlement	Settlement	150	1.21
			4	isPartOf	Settlement	-	37.5	1.21
			5	city	Airport	-	**0.86**	1.21
$\vee$			...	...	...	...	...	...
$\wedge$			1	notableWork	Writer	Play	496.6	3.12
$\ldots A_{agentcreated} \ldots$	*agentcreated*	9%	2	notableWork	Writer	TelevisionShow	3973	3.12
			3	occupation	Person	Book	12.7	3.12
			4	occupation	Settlement	-	37.5	3.12
			5	knownFor	Scientist	Book	3973	3.12
$\vee$			...	...	...	...	...	...

Now, we introduce K_{n_p}, the mapping factor determining the degree to which a particular NELL property can be mapped, as

$$K_{n_p} = \frac{\sum_{j=1}^{|T_{n_p}|} C(j)}{|T_{n_p}|}; \quad where\ C(j) = \begin{cases} 1; \textit{atleast one property mapping for } n_p \textit{ in } T_{n_p}^j \\ 0; \textit{otherwise} \end{cases}$$

Assuming, we have only ten triples for *airportincity*, eight have been mapped (mixture of Case (2) and (3) above), two have been not (Case (1) above), then $K_{airportincity} = \frac{8}{10}$. Here, under the column K_{n_p} we present the actual value which is 0.55. Then we apply an association rule [14] of the form $\{n_p \Rightarrow dom(d_s^i), ran(d_o^i)\}$, on A_{n_p}. This means, if the NELL property is n_p then the type of the mapped DBPEDIA subject instance is $dom(d_s^i)$ and type of the object instance is $ran(d_o^i)$. We compute the confidence, denoted as *conf*, for each such rule, and which denotes the frequency of co-occurrence of $dom(d_s^i)$ and $ran(d_o^i)$, whenever n_p occurred. Hence, the confidence for the i^{th} association for a property is denoted as $conf_{n_p}^i$, and defined as,

$$conf(n_p \Rightarrow (dom(d_s^i), ran(d_o^i))) = conf_{n_p}^i = count(n_p, dom(d_s^i), ran(d_o^i)) \Big/ |A_{n_p}|$$

Referring to Table 1, $conf_{agentcreated}^3 = count(agentcreated, Person, Book)/|A_{agentcreated}|$. Note the *count* function is not just the frequency count of the joint occurrence of a particular n_p and its associated DBPEDIA domain and range values, but, also the sub-classes of each of the domain and range. The rational is, if we observe an association like *agentcreated* $\Rightarrow$ (Person, Book) then any other association like *agentcreated* $\Rightarrow$ (Scientist, Book) should also be considered as an evidence for the former association. Scientist being a sub-class of Person in the DBPEDIA ontology, is not a different association but a more particular case. Finally, each association, is awarded with a confidence of $conf_{n_p}^i$.

We combine K_{n_p} and $conf_{n_p}^i$ to define the second factor called τ (*tau*) defined as,

$$\tau^i_{n_p} = \frac{(1 - K_{n_p})}{conf^i_{n_p}};\ \forall i \in A_{n_p}$$

This quantifies the badness of a particular association for a particular n_p with mapping factor K_{n_p}. A low confident association with low K_{n_p} will give a high $\tau^i_{n_p}$ ($\tau^3_{airportincity}$ in Table 1) while, a more confident association with high K_{n_p} minimizes the ratio, hence less bad ($\tau^5_{airportincity}$ in Table 1). We are primarily interested in the later ones. We employ, $\tau^i_{n_p}$ as our unit of measurement and define a minimum threshold, $\tau^{min}_{n_p}$ which occurs when $conf^i_{n_p}$ attains the maximum confidence (follows directly from the definition of tau above). Intuitively, $\tau^{min}_{n_p}$ defines the upper bound for the best score possible.

Threshold Learning: In this section we devise a technique to learn a correct threshold for $\tau^i_{n_p}$. Our objective is to solve the problem with least number of parameters possible. There can be three broad association patterns possible:

- a single high $conf^i_{n_p}$ association, among many others
- multiple closely spaced possible DBPEDIA properties with almost same $conf^i_{n_p}$
- No clear winner, but multiple candidates with low $conf^i_{n_p}$

We aim at modeling these different scenarios which would select the first two cases but not the third one. The rational is, any association rule with a low confidence is not an apt predictor for n_p. In this regard, we observed that the underlying data set had a distribution pattern over K_{n_p} (detailed figures in Section 3). We use it to manually determine a threshold for K_{n_p}, denoted as K_{thres}. Hence, we select data points having K_{n_p} atleast K_{thres}. This gives us a set of co-ordinates, D given as $\{\ldots, (K_{n_p}, \tau^{min}_{n_p}), \ldots\}$. We fit a linear regression model on set D, motivated by the fact that τ shows a linear dependence on K_{n_p} (our initial analysis had revealed that introducing $conf$ as another variable had minimal effect on τ, hence we use it as a constant). With such a linear predictive analysis method, we can have an estimate of τ, defined as $\hat{\tau}$ for every K_{n_p}. Note that, we trained our model using the data points attained using the maximum confidence (analogously $\tau^{min}_{n_p}$), hence, the linear model is an automatically learnt threshold on τ_{n_p}. We use $\hat{\tau}$ to compare with every $\tau^i_{n_p}, \forall\, i \in A_{n_p}$. Some scores fall below the prediction (the likely associations) and some are way above it (less likely ones) (refer to Table 1, correct association values are marked in bold). The likely associations allow us to select the final DBPEDIA property driven by the rule $(dom(d^i_s), ran(d^i_o)) \Rightarrow d^i_p$. Note that, in determining the property match we exploited the class information and remained un-informed of the actual DBPEDIA property involved. Analyzing the patterns now,

- Multiple associations but a single one with a high $conf^i_{n_p}$. This makes $\tau^i_{n_p} \simeq \hat{\tau}$
- Multiple closely placed associations with almost same $conf^i_{n_p}$, making $\tau^i_{n_p} \simeq \tau^j_{n_p} \simeq \hat{\tau}$; $i \neq j$. (refer Table 1, $\tau^2_{airportincity}$ and $\tau^5_{airportincity}$)
- No clear winner, but multiple candidates with low $conf^i_{n_p}$ making $\tau^i_{n_p} \ggg \hat{\tau}$ (refer Table 1, $\tau^1_{agentcreated}$, $\tau^5_{agentcreated}$)

2.3. Knowledge Generation

As a final module, we combine our two solutions in an attempt to generate new facts missing in DBPEDIA. Reiterating, we have a set of NELL triples, for which we could

Table 2. Precision of knowledge generated, for the properties predicted by our approach. * denotes inverse property mappings. Best results marked in bold

n_p	d_p	p_{inst}	p_{prop}	p_{fact}
headquarteredin	`headquarter`	0.96	1.0	**0.93**
visualartistartform	`movement`	0.99	0.06	0.057
	`field`	0.99	0.95	**0.93**
personhasresidenceincountry	`nationality`	1.0	0.33	**0.33**
airportincity	`city`	0.63	1.0	**0.3**
	`location`	0.50	1.0	0.17
stadiumlocatedincity	`location`	0.95	1.0	**0.91**
televisionstationincity	`locationCity`	0.98	1.0	**0.96**
	`location`	0.38	1.0	0.0
televisionstationaffiliatedwith	`broadcastNetwork`	0.99	1.0	0.98
	`formerBroadcastNetwork`	0.995	1.0	**0.99**
radiostationincity	`broadcastArea`	1.0	1.0	**0.90**
	`city`	0.50	1.0	0.0
personhasethnicity	`deathPlace`	0.70	0.0	0.0
	`birthPlace`	0.70	0.60	**0.20**
haswife	`partner`	0.96	1.0	**0.92**
	`spouse`	0.96	1.0	**0.92**
*musicianinmusicartist**	`bandMember`	0.98	1.0	**0.96**
	`associatedMusicalArtist`	0.50	1.0	0.0
*agentcreated**	`author`	1.0	0.80	**0.80**
*citycapitalofcountry**	`largestCity`	1.0	0.91	0.91
	`capital`	1.0	1.0	**1.0**
*automakerproducesmodel**	`manufacturer`	0.75	1.0	**0.50**
Macro-average	-	0.97	0.96	0.77

map its instances to DBPEDIA and its properties to analogous DBPEDIA properties. This provides strong evidence for the portion of NELL triples for which the property could not be mapped. This fraction is given by $1 - K_{n_p}$. Having n_p successfully mapped to d_p, we can use d_p to *fill-in* the missing relationships between the DBPEDIA instances for these non-mapped triples. Hence, the fraction of non-mapped triples for a NELL property defines an upper bound on the scope for new facts generation. This is a strict upper bound, since, a NELL triple can be non-mapped due to either:

1. a missing semantic relation between two correctly mapped DBPEDIA instances
2. a missing semantic relation between incorrectly mapped DBPEDIA instances
3. there is no mapping of instance possible at the first place

Clearly, points (1) and (2) are the ones which can lead to knowledge generation. As a matter of fact, Case (2) will generate wrong facts (further details in Section 3.1) and Case (3) cannot be dealt with our approach. In this respect, it is interesting to note a dilemma: if a property is nearly 100% mappable, we are more confident with its evidences but it gives us lesser scope of knowledge generation and vice versa.

3. Empirical Results

We use the NELL data set, having approximately 1.9Mi triples, for our experiments, but without the triples with property *generalizations* since it is analogous to `rdf:type` which expresses class instantiation. In this paper, we are focusing on the more complex task of finding the correct mapping for a potentially ambiguous property from NELL instead of generating facts like *isA(london, city)*. Note that mapping the instances, simultaneously solves the problem of determining the class of those instances since most of them are already typed in DBPEDIA.

3.1. Performance

Next we compute the precision of the newly generated triples. Since, a gold standard is not available, we resort to manual annotation scheme. Three annotators were provided samples of 300 NELL triples each. Annotators marked every mapping of the subject, property and object as "Correct" or "Incorrect" and also marked the original NELL triple to be "Correct", "Incorrect" or "Ambiguous". The later annotation was important since, even if the mapping of instances and properties are accurate, a wrong NELL triple in the first place will still lead to a wrong fact generated. Based on this agreement, only the triples with correct instance and property matches were considered as *true positives*. Even if one of the instances or the property match was incorrect, the triple was marked as a *false positive*. In Table 2 we present the precision scores for instance mappings (p_{inst}), property mappings (p_{prop}) and generated facts (p_{fact}). Note that, inaccurate instance mappings often lead to lower fact precision even if property mapping was precise (exactly the Case (2) mentioned in Section 2.3). This happens with {*airportincity*, `location`}. Also, the other way round, lower p_{prop} minimises p_{fact} inspite of a high p_{inst} as seen with {*visualartistform*, `movement`}. Hence, a highly accurate p_{inst} and p_{prop} together contributes to a high p_{fact}. The NELL properties with an asterisk (*) denote the inverse properties learnt and likewise present the precision of new triples. Observe that for some properties, we have dual choices of d_p, *visualartistartform* for instance. When mapped to `field` the precision of new triples were better than when mapped to `movement`, even though the later fitted the domain/range restrictions but when used in fact generation, led to senseless triples. Hence annotators marked it as a false positive. The line of reasoning is similar for *personhasethnicity* which had `birthplace` as the mapped property. On the other hand, `largestCity` was accepted as a mapped property for *citycapitalofcountry* since the triples generated as a whole were correct. Overall, we had a precision of 0.97 for instance mappings, 0.96 for property mappings, giving us 0.77 as macro-averaged precision for the generated facts.

3.2. Regression Analysis

In Figure 1(a, c) we present the distribution of $\tau_{n_p}^{min}$ over K_{n_p} (defined in Section 2.2), both for the direct and inverse property mappings respectively. We observe a similar trend in both the figures in the sense that higher values are attained for poorly mapped properties and properties with higher K_{n_p} tend to have lower values for $\tau_{n_p}^{min}$. This allows us to select the points on and beyond a particular threshold (in our case, $K_{thres} = 35\%$). In Figure 1(b,

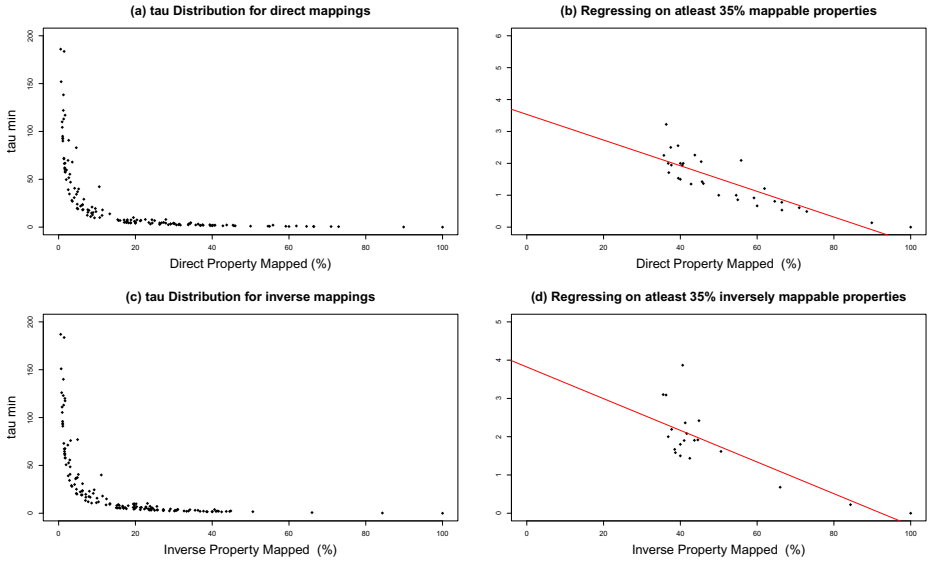

Figure 1. Regression analysis of direct and inverse property mappings.

d), we *zoom in* the data points beyond 35%, where we observe a linear variation. These points comprise of the set D (defined in 2.2). Likewise, we use these as training points to fit a linear predictive model having a single independent variable K_{n_p}. This is shown with the line fitting the points such that the squared error loss is minimized. The linear dependence relation between K_{n_p} and τ allows for this design choice. The regression line sets a self adjusting threshold varying across properties.

Furthermore, in Table 3 we present few examples which shows an interesting aspect of our method. The column labeled n_p denotes the NELL property and d_p the analogous DBPEDIA property learnt. The data is interpreted as follows: when the NELL property *headquarteredin* was mapped to `headquarter`, 39.4% of the mapped subjects were of type `Airline`, 16.6% were `Monarch` and analogously for the range values. The interesting aspect of the approach is that we are able to conserve the fine grained information latent in the facts and not just broadly classify them with some top level concepts.

4. Related Work

Matching Candidates: Seminal work include contributions by Bunescu and Paşca [4] and Cucerzan [6] who focused on the usage of Wikipedia categories and global contexts, respectively. The Silk framework [20] discovers missing links between entities across linked data sources by employing similarity metrics between pairs of instances. In contrast to these approaches, our method employs the most frequent sense of a term from Wikipedia. We combine this information together with the type-information from DBPEDIA in order to automatically refine the entity references [7].

Matching Properties: Much work has been done in the area of aligning ontologies of which PARIS [18] requires special mention which performs probabilistic alignment of relations, instances and schema across ontologies.

Table 3. Domain and Range distribution of mapped NELL properties for the set of new facts generated.

Domain (%)	n_p	d_p	Range (%)
Company(33.33)	*newspaperincity*	headquarter	City(50) Administrative-Region(33.33)
Airline(39.4) Company(36.6) BroadcastNetwork(5.6)	*headquarteredin*	headquarter	City(46.1) Settlement(37.6) Administrative-Region(11.3)
OfficeHolder(50) Person(33.3) Monarch(16.6)	*personhasethnicity*	birthPlace	Country(100)
OfficeHolder(40) SoccerManager(30) Model(10)	*personhas-residenceincountry*	nationality	Country(100)
Television-Station(73.8) RadioStation(26.2)	*televisionstation-affiliatedwith*	broadcastNetwork formerBroadcast-Network	Broadcast-Network(100)
Artist(98.8) Writer(1.2)	*visualartistartform*	field	-

Automated Knowledge Base Creation: The linking and filling approach is the most popular way of knowledge generation [13]. The last few years have witnessed some of the major works in automated information extraction systems and thereby targeting at large scale knowledge base constructions with minimal amount of human supervision. To this end, much work has explored unsupervised bootstrapping for a variety of tasks, including the acquisition of binary relations [3], facts [10], and instances [15]. OIE further focused on approaches that do not need any manually-labeled data [12]. Pujara et. al. [16] have used probabilistic soft logic to detect inconsistencies in knowledge graphs by exploiting dependencies within the graph. Furthermore, some pioneering works have been done by Wang et. al. [21] using statistical inference mechanisms (MCMC). However, our approach is different from these methods since, it exploits the open KBs to discover novel facts on a structured KB.

Future Work and Conclusion

In this work, we present a statistical approach to find accurate analogous properties across NELL and DBPEDIA . In the process we combine our probabilistic instance alignment method with this to generate set of facts. Our approach avoids tweaking of multiple parameters. We exploit the data set to train a simplistic model and use the model to learn threshold value across various NELL properties. Our approach generates highly accurate set of new DBPEDIA facts previously not extracted from the Wikipedia info-boxes. This can serve as an additional set of facts to DBPEDIA and thereby proving essential for any LOD based question-answering system.

We were able to generate approximately 1.6K new facts with direct mapping and approximately 0.5K with inverse mapping. These numbers are low given the fact that we started with 96K NELL triples. However, we hope to generate more triples with REVERB since its fact base is approximately 14Mi. Working with REVERB brings on the additional task of clustering similar properties (e.g. *is wife of*, *was married to*, *is spouse of*). Furthermore, our approach suffers from the manual selection of K_{thres}. We want to devise an automated technique to overcome this selection.

References

[1] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. {DBpedia} – {A} Crystallization Point for the Web of Data. *Journal of Web Semantics*, 7(3), 2009.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proc. of the 2008 ACM SIGMOD international conference on Management of data*, 2008.

[3] S. Brin. Extracting patterns and relations from the World Wide Web. In *Proc. of WebDB Workshop at EDBT-98*, 1998.

[4] R. Bunescu and M. Paşca. Using encyclopedic knowledge for named entity disambiguation. In *Proc. of EACL-06*, 2006.

[5] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, and T. M. Mitchell. Toward an architecture for never-ending language learning. In *Proc. of AAAI*, 2010.

[6] S. Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *Proc. of EMNLP-CoNLL-07*, 2007.

[7] A. Dutta, C. Meilicke, and S. P. Ponzetto. A probabilistic approach for integrating heterogeneous knowledge sources. In *ESWC*, volume 8465 of *LNCS*, pages 286–301. Springer, 2014.

[8] A. Dutta, M. Niepert, C. Meilicke, and S. P. Ponzetto. Integrating open and closed information extraction : Challenges and first steps. In *Proc. of the ISWC-13 NLP and DBpedia workshop*, 2013.

[9] O. Etzioni. Search needs a shake-up. *Nature*, 476:25–26, 2011.

[10] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale information extraction in KnowItAll (Preliminary results). In *WWW*, 2004.

[11] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *Proc. of EMNLP-11*, 2011.

[12] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *Proc. of EMNLP-11*, 2011.

[13] H. Ji and R. Grishman. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1148–1158, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[14] K. Lai and N. Cerpa. Support vs confidence in association rule algorithms. In *OPTIMA*, 2001.

[15] M. Paşca and B. Van Durme. Weakly-supervised acquisition of open-domain classes and class attributes from Web documents and query logs. In *Proc. of ACL-08*, 2008.

[16] J. Pujara, H. Miao, L. Getoor, and W. Cohen. Large-scale knowledge graph identification using psl. In *AAAI Fall Symposium on Semantics for Big Data*, 2013.

[17] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2), 2006.

[18] F. M. Suchanek, S. Abiteboul, and P. Senellart. PARIS: Probabilistic Alignment of Relations, Instances, and Schema. *PVLDB*, 5(3):157–168, 2011.

[19] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.

[20] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov. Silk - A Link Discovery Framework for the Web of Data. In *Proc. of LDOW '09*, 2009.

[21] D. Z. Wang, Y. Chen, S. Goldberg, C. Grant, and K. Li. Automatic knowledge base construction using probabilistic extraction, deductive reasoning, and human feedback. In *Proceedings of the Joint Workshop on*, AKBC-WEKEX '12, pages 106–110, 2012.