KnowMore - Knowledge Base Augmentation with Structured Web Markup

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Ran Yu [a,*], Ujwal Gadiraju [a], Besnik Fetahu [a], Oliver Lehmberg [b], Dominique Ritze [b] and Stefan Dietze [a]
[a] *L3S Research Center, Appelstr. 4, 30167 Hannover, Germany*
E-mail: {yu, gadiraju, fetahu, dietze}@l3s.de
[b] *Data and Web Science Group, University of Mannheim, B6, 26 68159 Mannheim, Germany*
E-mail: {oli,dominique}@informatik.uni-mannheim.de

Abstract. Knowledge bases are in widespread use for aiding tasks such as information extraction and information retrieval, where Web search is a prominent example. However, knowledge bases are inherently incomplete, particularly with respect to tail entities and properties. On the other hand, embedded entity markup based on Microdata, RDFa, and Microformats have become prevalent on the Web and constitute an unprecedented source of data with significant potential to aid the task of knowledge base augmentation (KBA). RDF statements extracted from markup are fundamentally different from traditional knowledge graphs: entity descriptions are flat, facts are highly redundant and of varied quality, and, explicit links are missing despite a vast amount of coreferences. Therefore, data fusion is required in order to facilitate the use of markup data for KBA. We present a novel data fusion approach which addresses these issues through a combination of entity matching and fusion techniques geared towards the specific challenges associated with Web markup. To ensure precise and non-redundant results, we follow a supervised learning approach based on a set of features considering aspects such as quality and relevance of entities, facts and their sources. We perform a thorough evaluation on a subset of the Web Data Commons dataset and show significant potential for augmenting existing knowledge bases. A comparison with existing data fusion baselines demonstrates superior performance of our approach when applied to Web markup data.

Keywords: Knowledge base augmentation, Web markup, microdata, data fusion, entity resolution, structured data

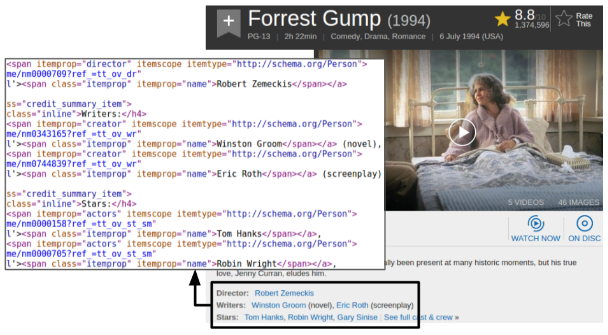
1. Introduction

Knowledge bases (KBs) such as Freebase [3] or YAGO [36] are in widespread use to aid a variety of applications and tasks such as Web search and Named Entity Disambiguation (NED). While KBs capture large amounts of factual knowledge, their coverage and completeness vary heavily across different types or domains. In particular, there is a large percentage of less popular (long-tail) entities and properties that usually are insufficiently represented. For instance, considering a selected set of popular properties used to describe books (Section 3), Freebase is missing corresponding statements in 63.8% of entities, Wikidata in 60.9% and DBpedia in 49.8%. Here, gaps are in particular observable for less popular books or attributes, such as *translator* or *number of pages*.

Recent efforts aim at exploiting data extracted from the Web to aid knowledge base augmentation (KBA). For instance, Knowledge Vault [10] uses triples extracted from Web documents, while recent works exploit semi-structured data from Web tables [30,31]. The approach described in [39] uses Web search templates. On the other hand, data fusion techniques aim at identifying the most suitable value (or fact) from

[*]Corresponding author. E-mail: yu@l3s.de.

a given set of observed values, for instance, the correct director of a movie from a set of candidate facts extracted from the Web [11]. To this end, data fusion techniques are fundamental when attempting to solve the KBA problem from observed Web data.


```
<span itemprop="director" itemscope itemtype="http://schema.org/Person">
me/nm0000709?ref_=tt_ov_dr"
l'><span class="itemprop" itemprop="name">Robert Zemeckis</span></a>

ss="credit_summary_item">
class="inline">Writers:</h4>
<span itemprop="creator" itemscope itemtype="http://schema.org/Person">
me/nm0343165?ref_=tt_ov_wr"
l'><span class="itemprop" itemprop="name">Winston Groom</span></a> (novel),
<span itemprop="creator" itemscope itemtype="http://schema.org/Person">
me/nm0744839?ref_=tt_ov_wr"
l'><span class="itemprop" itemprop="name">Eric Roth</span></a> (screenplay)

ss="credit_summary_item">
class="inline">Stars:</h4>
<span itemprop="actors" itemscope itemtype="http://schema.org/Person">
me/nm0000158?ref_=tt_ov_st_sm"
l'><span class="itemprop" itemprop="name">Tom Hanks</span></a>,
<span itemprop="actors" itemscope itemtype="http://schema.org/Person">
me/nm0000705?ref_=tt_ov_st_sm"
l'><span class="itemprop" itemprop="name">Robin Wright</span></a>,
```

Forrest Gump (1994)
PG-13 | 2h 22min | Comedy, Drama, Romance | 6 July 1994 (USA)
8.8/10 1,374,596

...lly been present at many historic moments, but his true love, Jenny Curran, eludes him.

Director: Robert Zemeckis
Writers: Winston Groom (novel), Eric Roth (screenplay)
Stars: Tom Hanks, Robin Wright, Gary Sinise | See full cast & crew »

Fig. 1. Example of entity annotation with Web markup.

Although the extraction of structured data from Web documents is costly and error-prone, the recent emergence of embedded and structured Web markup has provided an unprecedented source of explicit entity-centric data, describing factual knowledge about entities contained in Web documents. Figure 1 shows an example of a web page containing entity markup, where explicit entity information about actor Tom Hanks is embedded in a machine readable format, using vocabularies provided by *schema.org*. Building on standards such as RDFa[1], Microdata[2] and Microformats[3], and driven by initiatives such as schema.org, a joint effort led by Google, Yahoo!, Bing and Yandex, markup data has become prevalent on the Web. The Web Data Commons (WDC) [22], an initiative investigating a Web crawl of 2.01 billion HTML pages from over 15 million pay-level-domains (PLDs) found that 30% of all pages contain some form of embedded markup already, resulting in a corpus of 20.48 billion RDF quads[4]. Authors also identify an upward trend of adoption, where the proportion of pages containing markup increased from 5.76% to 30% between 2010 and 2014.

Through its wide availability, markup lends itself as a diverse source of input data for the KBA problem. In particular when attempting to complement information about long tail attributes and entities, the diversity and scale of markup provide opportunities for enriching existing knowledge bases and graphs [23].

However, the specific characteristics of facts extracted from embedded markup pose particular challenges [40]. Coreferences are very frequent, but are not linked through explicit statements. For instance, in the WDC2013 corpus, 18,000 disconnected entity descriptions are retrieved when querying the label of instances of type *schema:Product* for *'Iphone 6'*. In contrast to traditional strongly connected RDF graphs, markup statements mostly consist of isolated nodes and small subgraphs. In addition, extracted RDF markup statements are highly redundant and often limited to a small set of highly popular predicates, such as *schema:name*, complemented by a long tail of less frequent statements. Moreover, data extracted from markup contains a wide variety of errors, ranging from typos to the frequent misuse of vocabulary terms [21].

In this work, we introduce *KnowMore*, an approach based on data fusion techniques which enables the exploitation of markup crawled from the Web as diverse source of data to aid KBA. Our approach consists of a two-fold process, where first, candidate facts for augmentation of a particular KB entity are retrieved through a combination of blocking and entity matching techniques. In a second step, correct and novel facts are selected through a supervised classification approach and an original set of features. We apply our approach to the WDC2015 dataset and demonstrate superior performance compared to state-of-the-art data fusion baselines. We also demonstrate the capability for augmenting three large-scale knowledge bases, namely Wikidata[5], Freebase and DBpedia[6] through markup data based on our data fusion approach. The main contributions of our work are threefold:

– **Pipeline for data fusion on Web markup.** We propose a pipeline for data fusion (Section 3.3) that is tailored to the specific challenges arising from the characeristics of Web markup (Section 3.1). In particular, given the dynamics and scale of markup data, our approach performs the task of query-centric data fusion, which provides an efficient means to fuse only specific parts of a given markup corpus, obtained through a preliminary blocking step. Relevance, diversity, and correctness of facts is addressed through a combination

[1]RDFa W3C recommendation: http://www.w3.org/TR/xhtml-rdfa-primer/
[2]http://www.w3.org/TR/microdata
[3]http://microformats.org
[4]http://www.webdatacommons.org

[5]https://www.wikidata.org/
[6]http://dbpedia.org

of entity matching, data fusion and deduplication techniques. To the best of our knowledge, this is the first approach addressing the task of data fusion on Web markup data.

- **Model & feature set.** We propose a novel data fusion approach consisting of a supervised classification model (Section 5), utilising an original set of features geared towards validating correctness and relevance of markup facts. Experimental results demonstrate high precison (avg. 91.7%) and recall (avg. 88.2%) of our model, outperforming the state-of-art baselines.
- **Knowledge base augementation from markup data.** As part of our experimental evaluation (Section 7), we demonstrate the use of fused markup data for augmenting three well-established knowledge bases. Our results underline the suitability of markup data for supporting KBA tasks, where *KnowMore* is able to reach 100% coverage gain (Section 6.2.2) for selected properties and types, for instance, book descriptions in Freebase and Wikidata. On average, KnowMore has a coverage gain of 36.49% in Wikidata, 39.42% in Freebase and 34.75% in DBpedia. We also investigate the particular potential for augmenting tail entities and properties in Section 9.1.

The paper is structured as follows: Section 2 discusses related work on knowledge base augmentation and data fusion, while Section 3 introduces the motivation, problem statement and an overview of our approach. Section 4 and 5 describe the detailed steps for entity matching, data fusion and deduplication, while Section 6 describes the experimental setup, followed by the presentation of the results in Section 7. We assess the potential to generalise our supervised approach across distinct types in Section 8 and provide a thorough discussion of the KBA potential of markup as well as limitations of our work in Section 9 and conclude by proposing future work (Section 10).

2. Related Work

In this section we review related literature. We focus on two main lines of work on Linked Data, *knowledge-base augmentation* and *data fusion* as the most closely related fields to our work.

2.1. Knowledge-base Augmentation (KBA)

The main goal of KBA is to discover facts pertaining to entities and augmenting Knowledge Bases (KB) with these facts [38,15].

Some previous works have proposed approaches that suggest augmenting KBs with internal data. Such works typically focus on predicting the type [19] of an entity or finding new relations based on existing data [5,6,34]. Other prior works are more closely relevant to our problem setup; in that they focus on predicting relations with external data. Notable works propose the use of Wikipedia as a text corpus annotated with entities, seek for patterns based on existing KB relations, and further apply the patterns to find additional relations for DBpedia [1] or Freebase [25]. News corpora have also been used for augmenting DBpedia with similar settings [14]. Paulheim et al. [26] proposed to identify common patterns of instances in the Wikipedia list page and apply the patterns to add relations to the remaining entities in the list. Dong et al. proposed '*Knowledge Vault*' [10], a framework for extracting triples from web pages, aimed at constructing a KB from Web data. Dutta et al. [13] focus on the mapping of relational phrases such as facts extracted by '*Nell*' and '*Reverb*' to KB properties. Furthermore, they group the same semantic relationships represented by different surface forms together through Markov clustering. Recent works by Ritze et al. use relational HTML tables available on the Web to fill missing values in DBpedia [30,31]. The authors propose to first match the tables to the DBpedia entities, and then compare several data fusion strategies such as voting and the Knowledge-Base Trust (KBT) score to identify valid facts.

Ristoski et al. proposed an approach to enrich product ads with data extracted from Web Data Commons [29]. The approach extracts attribute-value pairs from plain text and matches them to database entities with supervised classification models. The notable methods described in previous works are tailored to specific data sources, which have different characteristics compared to markup data. Hence, merely adopting the existing methods to cater for markup data is not sufficient. However, we have revised and adopted some of the features in our proposed approach.

Other works suggest using the whole Web as a potential data corpus through search engines [16,39]. QA-based approaches are often designed to facilitate the filling of values of a specific set of properties, and they rely on manually created templates. This limits their application to constrained sets of properties.

Existing works typically assume that there is only one true value for a property when resolving conflicts. In contrast, we aim for more higher recall by catering for multiple (non-redundant) correct values, catering for the fact that multiple-cardinality properties are prevalent. Another limitation of existing KBA works is that the novelty of the discovered facts is ignored; there is an overlap between the result and the facts existing in a KB. On the contrary, our approach aims at providing correct and novel results that can be of immediate value to the KB.

2.2. Data Fusion

Data fusion is defined as "the process of fusing multiple records representing the same real-world object into a single, consistent, and clean representation" [2]. In the context of the Semantic Web, previous works on data fusion can be categorized into two classes – *heuristic-based* and *probability-based.*

Heuristic-based Methods. Schultz et al. introduced '*LDIF*', that uses user provided heuristics to find duplicate real-world entities [33]. Mendes et al. proposed '*Sieve*', which resolves conflicts in Linked Data from different sources by selecting one value for each property based on quality measures such as recency and frequency [20,4]. '*ODCleanStore*' provides heuristic-based mechanisms such as max frequency for resolving conflicts during the fusion of Linked Data [17,24]. One of the limitations of such heuristics-based approaches is that they rely on the observation of a specific dataset, which is often not generalizable for other datasets. Furthermore, the heuristics usually focus on a single aspect of the quality, e.g. recency or frequency, while the quality of a resource is typically influenced by multiple factors to varying degrees.

Probability-based Methods. Zhao et al. [43] proposed an unsupervised probabilistic graphical model to infer true records and source quality based on the false-positives and false-negatives of the data source. Dong et al. [11] introduced data fusion techniques which identify true subject-predicate-object triples, that are extracted by multiple extractors and originate from multiple sources [10]. In this work, authors selected and adapted three existing data fusion techniques and improved them with a series of refinements. Furthermore, Pochampally et al. proposed to use joint precision and joint recall to indicate correlation between sources in order to penalize the copying between sources [28]. In later work, the authors proposed a probabilistic model to compute the Knowledge-Based Trust (KBT), i.e., a score for measuring the trustworthiness of the resources [12]. KBT focuses on the general quality of a resource, and is computed based on the relation between a resource and Freebase.

The major difference between our work and the aforementioned works is that previous works focus only on the correctness of the source and assign equal weights to all the facts from the same source. In contrast, we not only consider the source quality but also consider features of the predicates, facts and entities. Hence, distinct facts from the same source are classified differently, depending on multiple feature dimensions. Thus, through a more fine-grained classification, our data fusion approach is able to improve both precision and recall. Another difference is that our query-centric data fusion approach does not require the fusion of of the entire dataset after partial changes to the corpus, but can be applied iteratively over specific subsets.

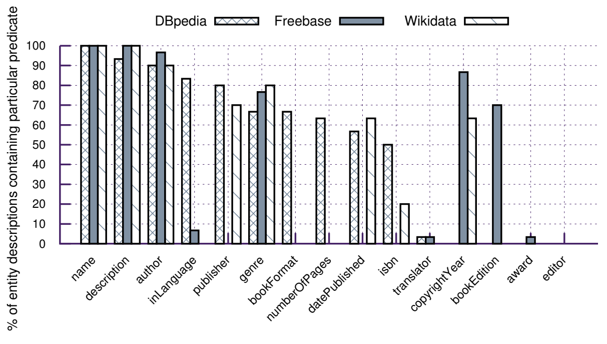
Our recently published work presents an entity summarization approach that retrieves entities from WDC and selects distinct facts to build entity descriptions based on clustering [42]. Additional recent work [41] proposes a data fusion approach focused on ensuring correctness of facts obtained from noisy entity descriptions in Web markup. While the main focus of the former work was on deduplication, the main focus of the latter was correctness. In contrast, this paper proposes a complete two-step pipeline aiming at obtaining correct and non-redundant facts which augment existing KBs.

3. Motivation & Approach

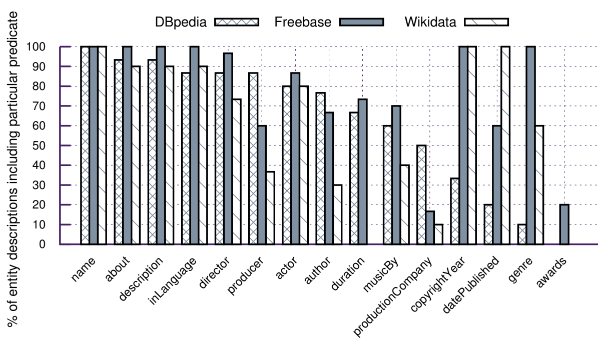
3.1. Motivation

Previous works [32,9,40] have investigated the nature of several type-specific subsets of Web markup, namely, data bibliographic data, metadata about learning content, books and movies. These works assess markup data on several dimensions, such as data quality, the source distribution and the schema usage. Results show the complementary nature of markup data, when compared to traditional knowledge bases, where the extent of additional information varies strongly between types.

For a preliminary analysis of *DBpedia*, *Freebase* and *Wikidata*, we randomly selected 30 Wikipedia entities of type *Movie* and *Book* and retrieved the corresponding entity descriptions from all three KBs. We select the 15 most frequently populated properties for each type and provide equivalence mappings across all KB schemas as well as the *schema.org* vocabulary manually[7]. Since all vocabulary terms and types in the following refer to *schema.org*, prefixes are omitted. Figure 2 shows the proportion of instances for which the respective property is populated for the movie case. Indicated properties refer to the corresponding *schema.org* term. There is a large amount of empty slots across all KBs for most of the properties, with an average proportion of empty statements for books (movies) of 49.8% (37.1%) for DBpedia, 63.8% (23.3%) for Freebase and 60.9 % (40%) for Wikidata.



(a) Book



(b) Movie

Fig. 2. Proportion of book and movie instances per KB that include selected popular predicates.

In addition, coverage varies heavily across different properties, with properties such as *editor* or *translator* being hardly present in any of the KBs.

On the other hand, tail entities/types as well as dynamic properties which require frequent updates, such as the *award* of a book, are prevalent in markup data [23], yet tend to be underrepresented in structured KBs. Hence, markup data lends itself as data source for the KBA task. However, given the specific characteristics of markup data [40], namely the large amount of coreferences and near-duplicates, the lack of links and the variety of errors, data fusion techniques are required which are tailored to the specific task of KBA from Web markup.

3.2. Problem Definition

For the purpose of this work, an *entity description* is considered a semi-structured representation of an actual *entity*, where the latter is either a physical (e.g. a person) or an abstract notion (e.g. a category or theory). Entity descriptions which represent the same entity are considered to be *coreferences*.

In particular, we consider entity descriptions accumulated by extracting structured Web markup from Web documents. We refer to such a dataset as M, where the WDC dataset is an example. Data in M consists of entity descriptions, each consisting of a set of RDF quads, i.e. a set of $\langle s, p, o, u \rangle$ quadruples and referring to entities, either physical or abstract. The elements $\langle s, p, o, u \rangle$ of the quadruple represent subject, predicate, object and the URL of the document from which the triple $\langle s, p, o \rangle$ has been extracted, respectively. There exist $n \geq 0$ subjects $\{s_1, s_2, ..., s_n\}$, and consequently, n entity descriptions $e_i = \langle s_i, p_i, o_i \rangle \in E$ which represent a particular entity q. We define a property-value pair $\langle p, o \rangle$ describing the entity q as a fact of q. Note that we explicitly consider multi-valued properties, i.e. a particular predicate p might be involved in more than one fact for a particular entity q.

As input for the KBA task we consider an entity description e_q from a given KB representing an entity q. We define the KBA task from a given markup corpus M given a knowledge base entity description e_q as follows:

Definition 1 *KBA: For an entity q that is represented through an entity description e_q in a KB, we aim at selecting a subset F' from M, where each fact $f_i \in F'$ represents a valid fact which augments the KB description e_q for q.*

We consider a fact valid for augmentation, if it meets the following criteria:

- A fact is *correct*, i.e. consistent with the real world regarding query entity q according to some ground truth (Section 6).

[7]The mappings are online at: http://l3s.de/ yu/knowmore/

– A fact represents *novel*, i.e. not duplicate or near-duplicate, information with regard to the entity description e_q of q in a given KB.
– The predicate p_i of fact $\langle p_i, o_i \rangle$ should already be reflected in a KBs given schema.

As defined in the last statement, we limit ourselves to the augmentation of existing predicates for the sake of this work. However, we include a detailed discussion of the augmentation of statements involving new properties relative to a given KB in Section 9.

As an illustrative example, let q be the book *Brideshead Revisited*. In a given KB, such as DBpedia, there is an entity description[8] e_q which represents q. From the Web markup corpus (M), we can extract a set of coreferring entity descriptions E representing the query entity q, that is, the book *Brideshead Revisited*. An example entity description $e_i \in E$ consists of 3 triples $\{$*<_:node1, author, Evelyn Waugh>, <_:node1, datePublished, 1940>, <_:node1, isbn, 9781904605577>*$\}$. From all the facts (e.g. *isbn, 9781904605577*) in E, we aim at selecting the ones that are valid, according to the previous definition, for augmenting the KB at hand.

3.3. Approach Overview

Our approach (*KnowMore*) for addressing the KBA problem defined above, consists of two steps, namely (i) entity matching, and (ii) data fusion. We introduce the intuition behind each step below and describe the actual method in the following sections.

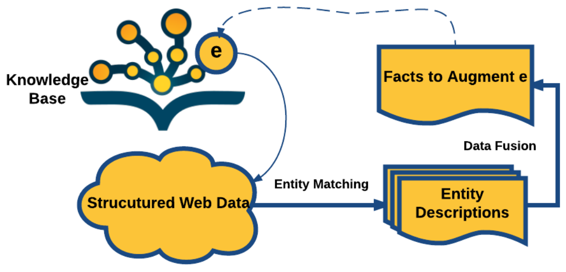


Fig. 3. Overview of pipeline.

Entity matching. The first step aims $KnowMore_{match}$ at obtaining candidate facts by collecting the set E of co-referring entity descriptions $e_i \in E$ from M which describe q and corefer to the entity description e_q in a given KB. We use a three step approach in order to efficiently achieve high accuracy results.

– Data cleansing to improve general data quality.
– Blocking with standard BM25 entity retrieval on the value of property *name* of all indexed entity descriptions to reduce the search space.
– Validation based on a weighted similarity between e_q and the entity descriptions in E.

Hence, we retrieve the set E containing candidate entity descriptions represented through facts $f \in F$ that potentially describe q.

Data fusion. During the data fusion step $KnowMore_{class}$, we aim at selecting a subset $F' \subset F$ that fulfills the criteria as listed in Section 3.2. More specifically, we introduce *data fusion* techniques based on supervised classification to ensure the correctness (Section 5.1) and two deduplication steps to ensure novelty (Section 5.2), namely deduplication with respect to M ($KnowMore_{ded}$) and deduplication with respect to KB ($KnowMore_{nov}$). The latter is a prerequisite for the KBA task.

For clarity, we summarize the notations used for identifying each step of the *KnowMore* pipeline and the corresponding in- and outputs in Table 1.

Table 1
Summarize of notions.

Step	Notion	Input	Output
Entity matching	$KnowMore_{match}$	q, M	F
Data fusion - correctness	$KnowMore_{class}$	F	F_{class}
Deduplication with respect to M	$KnowMore_{ded}$	F_{class}	F_{ded}
Deduplication with respect to KB	$KnowMore_{nov}$	F_{ded}	F'

We describe each step of the approach in the following two sections in detail.

4. Entity Matching

The entity matching step ($KnowMore_{match}$) aims at detecting a set of candidate entity descriptions $e_i \in E$ with $E \subset M$ which are likely to be coreferences of a given KB entity description e_q. We apply three steps in $KnowMore_{match}$, namely cleansing, blocking and matching. These steps are applied over all entities which are to be augmented.

[8]http://dbpedia.org/resource/Brideshead_Revisited

4.1. Data Cleansing

This initial data cleansing step aims at (i) resolving object references and (b) fixing common errors [21] to improve overall usability of the data.

While schema.org property range definitions are not bound in a strict way but constitute mere recommendations, previous studies [9] observe a strong tendency towards statements which refer to literals rather than objects, i.e. URIs. For instance, within a markup corpus of 44 million quads, 97% of transversal properties referred to literals rather than URIs/objects, despite the fact that only 64% of quads involved properties where schema.org recommends literals as property range [9]. Given this prevalence of literals in Web markup and the need to homogenise entity descriptions for further processing, we resolve object references into literals by replacing object URIs with the labels of the corresponding entity.

In addition, based on earlier work [21] which studied common errors in Web markup, we implement heuristics and apply these to E as a cleansing step thereby improving the quality of the data. In particular, we implemented the heuristics as proposed in [21] to:

- **Fix wrong namespaces.** Most namespace issues seem due to typing errors, e.g. lacking a slash or using *https://* instead of *http://*. Another reason is the misuse of the upper/lower-cases in a case-sensitive context, e.g. use of *Schema.org* instead of the valid term *schema.org*.
- **Handle undefined types and properties.** The use of undefined types and properties are frequent in Web markup data. Some of the undefined types exist due to typos and the misuse of the upper/lower-cases, e.g. the use of *creativework* for the intended type *CreativeWork*, where simple heuristics can be applied to resolve these issues.

Applying these heuristics improves the performance of the subsequent step by providing a wider and improved pool of candidates.

4.2. Blocking

Blocking is typically used as a pre-processing step for entity resolution to reduce the number of required comparisons. The goal of blocking is to place the potentially relevant entity descriptions into the same block, thus the entity resolution algorithm is applied to entity descriptions within the same block only [7].

We implement the blocking step through the entity retrieval using the BM25 retrieval model, i.e. a probabilistic ranking function used to rank matching documents according to their relevance to a given search query, to reduce the search space. We created an index for each type-specific subset using Lucene, and then use the *label* of e_q to query the field *name*[9] within a type-specific index. Hence, queries for a specific type/label-combination which represents e_q result in a set of candidate entity descriptions $e_i^0 \in E_0$ that potentially describe the same entity as e_q. [37] shows that string comparison between labels of markup entities is an efficient way for obtaining potential co-references.

For instance, considering the query "*Brideshead Revisited*" (of type *Book*), as part of the the blocking step we query the Lucene index and obtain 1,657 entity descriptions consisting of 15,940 quads. An excerpt of the result set is shown in Table 2.

Table 2
Excerpt from the result set (1,657 entity descriptions in total) for the query "*Brideshead Revisited*" (of type *Book*) after blocking.

Node ID	**Property**	**Value**
_:node1	author	Evelyn Waugh
_:node1	datePublished	1940
_:node1	isbn	9781904605577
_:node2	author	Waugh, Evelyn
_:node2	publisher	Back Bay Books.
_:node3	author	Roger Parsley
_:node3	publisher	Samuel French Ltd

4.3. Entity Matching

When attempting to match entities, one can build on the observation that particular property-value pairs can be considered near-unique identifiers for a specific entity, so-called pseudo-key properties. For instance, *taxID* can be considered one of the pseudo-key property for instances of type *Person*, and *isbn* for instances of type *Book*. However, as studied by Meusel et al. [23], resolving coreferences simply through pseudo-key properties does not produce sufficient results when applied on sparsely described and heterogeneous entity descriptions obtained from Web markup.

[9]http://schema.org/name

Thus, we adapt the entity matching approach described in [29] to filter out noise in E_0, for instance, entity descriptions which are not relevant to q but fetched through the initial blocking step due to ambiguous labels. Our matching approach builds on the assumption that the importance of distinct properties differs when computing entity similarity, with pseudo-key properties being the most decisive ones. For example, instances of type *Book* which share the same *author* have a higher probability to be equivalent than books which share the same *bookFormat*.

In order to compute the similarity for each property, we consider all properties as attributes of the feature space $\overrightarrow{A} = \{a_1, a_2, ..., a_n\}$, so that each entity description e can be represented as a vector of values $\overrightarrow{v} = \{o_{a1}, o_{a_2}, ..., o_{a_n}\}$ which represent the objects of the considered $\langle p, o \rangle$ tuples. We construct a similarity vector $\overrightarrow{sim}(\overrightarrow{v^{KB}}, \overrightarrow{v})$ between e_q and each entity description $e_i^0 \in E_0$ as in Equation 1.

$$\overrightarrow{sim}(\overrightarrow{v^{KB}}, \overrightarrow{v}) = \{\lambda_{a_1}, \lambda_{a_2}, ..., \lambda_{a_n}\} \quad (1)$$

$$\lambda_{a_i} = sim(o_{a_i}^{KB}, o_{a_i}) \quad (2)$$

In order to compute $sim(o_{a_i}^{KB}, o_{a_i})$, we employ datatype-specific similarity metrics, i.e., we implemented one similarity measure for each *schema.org* datatype[10], and automatically select the appropriate metric. Specifically, for 1) *Text*, we employ cosine similarity, for 2) *Number* or *Boolean* attributes, *sim*-$(o_{a_i}^{KB}, o_{a_i})$ equals to 1 if $o_{a_i}^{KB}$ is the same as o_{a_i}, and 0 otherwise. For 3) *Time* or *DateTime*, we first unify different formatting styles with the *java DateTimeFormatter*[11] class, and then split values into separate units (e.g. year, month, date), where $sim(o_{a_i}^{KB}, o_{a_i})$ is 0 if there is a conflict in the observed values. For instance, 1990 April and May 10 would constitute a conflict. Otherwise, we compute the Jaccard similarity between both triples $\langle year, month, date \rangle$ as a metric of the overlapping semantics.

The source code of this step can be found online[12].

We then train a supervised classification model, to make the decision whether or not e_i^0 is a match for e_q. We experimented with several state-of-the-art classifiers (SVM, Logistic Regression and Naive Bayes). Since Naive Bayes achieves a $F1$ score that is 7.95% higher than the best SVM (linear kernel), and 12.25% higher than the Logistic Regression (LR), throughout the remaining paper we rely on a trained Naive Bayes classifier unless otherwise stated. More details about classifier performance are provided in Section 7.1. The classification and clustering implementation in our approach is built on top of the Java-ML toolkit[13]. The training data is described in Section 6.2.

The final result of the entity matching step is the set of co-referring entity descriptions $e_i \in E$ which constitute candidate facts $f_i \in F$ for the following steps.

Referring to our example, after removing the unmatched entity descriptions through the entity matching step from the blocking result, there are 44 matched entity descriptions remaining in the result set. Some examples are shown in Table 3, where, for instance, *_:node3* had been removed since it refers to the stage play rather than the book and does not match entity q.

Table 3
Excerpt from result set (44 entity descriptions in total) for the query, "*Brideshead Revisited*" (of type *Book*) after entity matching.

Node ID	**Property**	**Value**
_:node1	author	Evelyn Waugh
_:node1	datePublished	1940
_:node1	isbn	9781904605577
_:node2	author	Waugh, Evelyn
_:node2	publisher	Back Bay Books.

5. Data Fusion

This step aims at fusing candidate entity descriptions in E by detecting the correct and novel facts $f' \in F'$ with $F' \subset F$ to augment e_q.

5.1. Correctness - Supervised Classification

The first step ($KnowMore_{class}$) aims at detecting correct facts by learning a supervised model that produces a binary classification for a given fact $f \in F$ into one of the labels {*'correct'*, *'incorrect'*}. For the classification model, we have experimented with sev-

[10]http://schema.org/DataType
[11]java.time.format.DateTimeFormatter
[12]http://l3s.de/ yu/knowmore/
[13]http://java-ml.sourceforge.net/

eral different approaches, namely Naive Bayes classification, SVM with different kernel functions and Logistic regression. We rely on a Naive Bayes classification since our experiments have shown superior performance over the second best performing approach Logistic Regression with an increase in $F1$ score of 1.6%, and over SVM (linear kernel) with an increase in $F1$ score of 4.4%. More details are provided in Section 7.2. We introduce the features used for our supervised learning approach in Table 4 and describe them in detail below. Through an initial data analysis step, all features have been identified as potential indicators of fact correctness.

While we aim to detect the correctness of a fact, we consider characteristics of the *source*, that is the Pay-Level-Domain (PLD, i.e. the sub-domain of a public top-level domain, for which Website providers usually pay for), from which a fact originates, the *entity description*, the *predicate* term as well as the *fact* itself. The four different categories are described below.

Source level. As has been widely studied in previous works, source quality is an important indicator for data fusion [43,28,12]. Features $t_i^r, i \in [1,3]$ consider the PageRank score as an authority indicator of the PLD from which a fact is extracted, assuming a higher PageRank indicates higher authority and hence quality. With respect to $t_i^r, i \in [4,9]$, we follow the intuition that, if more errors have been detected across markup from a respective PLD, there is a higher potential of a PLD to provide incorrect facts. We consider the rate of common errors detected based on previously identified heuristics [21] to compute $t_i^r, i \in [4,6]$ and the precision of a PLD computed based on our ground truth (Section 6.2) as quality indicators and hence extract feature $t_i^r, i \in [7,9]$.

Entity level. Based on the data analysis, entity descriptions containing a large number of facts are usually of higher quality. Thus, we use the size of entity descriptions, reflected through features $t_i^e, i \in [1,3]$, as additional indicator of quality.

Property level. The quality of facts strongly varies across predicates, as identified in previous studies [40, 23], with some properties being more likely to be part of a correct fact than others. One example of a predicate often included in incorrect statements is *datePublished* of a movie, that is often mistakenly used to describe the publishing time of the Web document. Following this observation, we extract features $t_i^p, i \in [1,5]$ to consider characteristics of the involved predicate terms, such as their frequency.

Given that our candidate set contains vast amounts of near-duplicate facts, we approach the problem of identifying semantically equivalent statements through clustering of facts which use varied surface terms for the same or overlapping meanings. We employ the X-Means algorithm [27], as it is able to automatically determine the number of clusters. This clustering step aims at grouping or canonicalizing different literals or surface forms for specific object values. For instance, *Tom Hanks* and *T. Hanks* are equivalent surface forms representing the same entity. To detect duplicates and near-duplicates, we first cluster facts that have the same predicate p into n clusters $(c_1, c_2, \cdots, c_n) \in C$. In this way, considering string similarity, we can canonicalize equivalent surface forms. The performance of the clustering on removing near duplicates is discussed in Section 7.3. Another challenge considered here is the cardinality of predicates. Depending on the predicate, the number of potentially correct statements varies. For example, *actor* is associated with multiple values, whereas *duration* normally has only one valid statement. This is reflected in the cluster amount n for a given predicate (t_3^p). The intuition behind feature t_4^p is that the average size of clusters is a indicator of the frequency of facts in p which usually correlates with the quality. Feature t_5^p is extracted based on the observation that in most cases, wrong facts have lower frequency than average, thus the variance is larger if there are wrong facts among the facts of p.

Fact level. Fact frequency [20] has been used in previous data fusion works and is shown to provide efficient features for determining the correctness of facts. Based on these insights, we extract features $t_i^f, i \in [1,2]$. We consider the size of a cluster as feature t_2^f indicating the frequency of a fact, where the normalized size of cluster c_i is $|c_i|/\sum_{j=1}^{n} |c_j|$.

From the computed features we train the classifier for classifying the facts from F into the binary labels $\{$*'correct'*, *'incorrect'*$\}$. More details about the training and evaluation through 10-fold cross-validation are presented in Section 7.2. The *'correct'* facts form a set F'_{class} that is the input for next steps.

Referring to our running example, after removing wrong facts from the candidate facts, such as *datePublished: 1940* through the classification step, we obtain 37 correct facts in the result set for the query *Brideshead Revisited, type:(Book).* An excerpt of the resulting facts are shown in Table 5.

Table 4
Features for supervised data fusion from markup data.

Category	**Notation**	**Feature description**
Source level	t^r_1, t^r_2, t^r_3	Maximum, minimum, average PageRank score of the PLDs containing fact f
	t^r_4, t^r_5, t^r_6	Maximum, minimum, average percentage of common errors [21] of the PLDs containing fact f
	t^r_7, t^r_8, t^r_9	Maximum, minimum, average precision (based on training data) of the PLDs containing fact f
Entity level	t^e_1, t^e_2, t^e_3	Maximum, minimum, average size (number of facts) of e_i containing f
Property level	t^p_1	Predicate term
	t^p_2	Predicate frequency in F
	$\dagger t^p_3$	Amount of clusters of predicate p
	$\dagger t^p_4$	Average cluster size of predicate p
	$\dagger t^p_5$	Variance of the cluster sizes of predicate p
Fact level	t^f_1	Fact frequency in F
	$\dagger t^f_2$	Normalized cluster size that f belongs to

†-features extracted based on clustering result

Table 5
Excerpt from result set (37 distinct correct facts) for query "*Brideshead Revisited*" (of type *Book*) for *KnowMore$_{class}$*.

Class	**Property**	**Value**
correct	s:author	Evelyn Waugh
correct	s:isbn	9781904605577
correct	s:author	Waugh, Evelyn
incorrect	datePublished	1940
correct	s:publisher	Back Bay Books.

5.2. Novelty

A fact f is considered to be *novel* with respect to the KBA task, if it fulfills the conditions: i) not duplicate with other facts selected from our source markup corpus M, ii) not duplicate with any facts existing in the KB. Each of these two conditions corresponds to a deduplication step.

Deduplication with respect to M (*KnowMore$_{ded}$*). As introduced in Section 5.1, we detect near-duplicates via clustering. For each predicate p, all the facts $f = \langle p, o_i \rangle$ corresponding to p are clustered into n clusters $\{c_1, c_2, \cdots, c_n\}$. Each cluster $c_i, i = 1, ..., n$ contains a set of near-duplicates. To fulfill i), we select only one fact from each cluster by choosing the fact that is closer to the cluster's centroid. This results in the fact set F_{ded} that is the input for next deduplication step.

Deduplication with respect to KB (KnowMore$_{nov}$). We compute the similarity $Sim(f_i, f_{KB})$ between facts f_{KB} in a respective KB for a particular predicate p and facts f_i for the same (mapped) predicate p in F_{ded} with the datatype-specific similarity metrics as introduced in Section 4. If $Sim(f_i, f_{KB})$ is higher than a threshold τ, we remove the fact along with its near-duplicates, i.e. the facts in the same cluster from the candidate set F'. We explain τ and its configuration during the experimental Section 6.3. The facts selected from F' in this step are the final result for augmenting the KB. Referring to the example, the fact *author: Waugh, Evelyn* is removed during the deduplication with regard to M as it is a duplicate of fact *author: Evelyn Waugh*, yet has been selected as more representative.

With respect to the KBA task, consider the augmentation of the example entity "*Brideshead Revisited*" (of type *Book*) as illustrated in Table 6. The example facts #2 and #3, would be valid results of the KBA task for DBpedia since they are **novel**, while only fact #2 is a valid augmentation for Freebase and Wikidata as it is the only fact that is novel.

Note that our deduplication step considers and supports multi-valued properties. By relying on the clustering features, computed during the fusion step, we

Table 6
Novelty of correct, distinct facts with regard to KBs for the query "*Brideshead Revisited*" (of type *Book*).

ID	Fact	DBpedia	Wikidata	Freebase
1	author, Evelyn Waugh	✗	✗	✗
2	isbn, 9781904605577	✔	✔	✔
3	publ., Back Bay Books	✔	✗	✗

select facts from multiple clusters (corresponding to multiple predicates) as long as they are classified as correct. As documented by the evaluation results, (Section 7), this does neither negatively effect precision while improving recall for multi-valued properties.

6. Experimental Setup

6.1. Data

Dataset We use the WDC2015 dataset[14], where we extracted 2 type-specific subsets consisting of entity descriptions of the *schema.org* types *Movie* and *Book*. Initial experiments indicated that these types are well reflected in the WDC2015 datasets, and at the same time, their facts are comparably easy to validate manually when attempting to label a ground truth. The *Movie* subset consists of 116,587,788 quads that correspond to 23,334,680 subjects/nodes, and the *Book* subset consist of 174,459,305 quads and 34,655,078 subjects.

Entities & KBs to Augment As input for the KBA task, we randomly select 30 entities for each type *Book* and *Movie* from Wikipedia. We evaluate the performance of our approach for augmenting entity descriptions of these 60 entities obtained from three different KBs: DBpedia (*DB*), Freebase (*FB*) and Wikidata (*WD*). For DBpedia, we retrieve entity descriptions through the SPARQL endpoint[15] where resource URIs were obtained by replacing the Wikipedia namespace of our selected entities with the DBpedia resource path. URIs of corresponding Freebase and Wikidata entity descriptions are obtained through the *owl:sameAs* links present in DBpedia. Using these URIs, the respective entity descriptions are obtained through the latest available version of Freebase[16] (accessed Sep 30, 2016) and the Wikidata SPARQL endpoint[17].

The full list of entities can be found online[12]. A preliminary analysis of the completeness of these obtained entity descriptions is shown in Figure 2 in Section 3.1.

Properties to Augment To simplify the schema mapping problem between WDC data and the respective KBs while at the same time, taking advantage of the large-scale data available in our corpus, we limit the task to entities annotated with the *http://schema.org* ontology for this experiment. Previous works have shown that schema.org is the only vocabulary which is consistently used at scale [23]. We manually create a set of schema mappings that maps the *schema.org* vocabularies to the *DB, FB, WD* vocabularies. For this, we first select all the *schema.org* predicates appearing in F. We identify the ones that have equivalent properties within all involved vocabularies and create equivalence mappings (*owl:equivalentProperty*). The list of predicates and the mapping statements can be found online[12].

6.2. Ground Truth & Metrics

This section describes the ground truth used for training and testing together with the evaluation metrics used for assessing performance in different tasks.

6.2.1. Ground Truth via Crowdsourcing

Entity Matching. We used crowdsourcing to build a ground truth by acquiring labels for each $e_i \in E$. In each case, crowd workers were presented with the entity description e_q, i.e. the Wikipedia page, and entity description $e_i \in E$, and were asked to validate e_i as either *valid*, *invalid* or *insufficient information to judge* with respect to e_q. We deployed the task on CrowdFlower[18], and gathered 5 judgments from distinct workers on each $(q, e_i \in E)$ pair. To ensure high quality, we restricted the participation to Level 3 workers alone[19]. In addition to this, we used test questions to flag and reject untrustworthy workers. Workers were compensated at the rate of 6 USD cents per judgment. On average, workers performed with an accuracy of 92% on the test questions, indicating high reliability. The inter-rater agreement between workers was 75%

[14]http://webdatacommons.org/structureddata/index.html#toc3
[15]http://dbpedia.org/sparql
[16]http://commondatastorage.googleapis.com/freebase-public/rdf/freebase-rdf-latest.gz

[17]https://query.wikidata.org/sparql/
[18]http:www.crowdflower.com
[19]Level 3 workers on CrowdFlower have the best reputation and near perfect accuracy in hundreds of previous tasks.

using Krippendorff's Alpha [18], and 89% using pairwise percent agreement (PPA). By applying this process on E_0, we obtain 89 (180) *valid* and 128 (118) *invalid* entity descriptions for *Movie* (*Book*) entities respectively.

Data Fusion - Correctness. Similarly, we used crowdsourcing to build a ground truth for the correctness of facts $f_i \in F$. For the valid entity descriptions in E, we acquire labels for all distinct facts, as either *correct* or *incorrect* with respect to q. We acquired 5 judgments from distinct workers for each entity and corresponding facts through Crowdflower. We used similar quality control mechanisms as in the entity matching task. Workers were compensated at the rate of 6 USD cents per judgment. Workers performed with an accuracy of 95% on the test questions. The inter-rater agreement between workers was 71.1% using Krippendorff's Alpha, and 86.9% using pairwise percent agreement (PPA). This indicates a high reliability of the ground truth. This process results in 371 (out of 456) and 298 (out of 341) *correct* facts for *Movie* and *Book* dataset respectively. Distinct facts were obtained by removing duplicate literals, null values, URLs and the unresolved objects (e.g. *node3* that could not be resolved in the dataset). The ground truth is publicly available [12].

Data Fusion - Novelty. We built corresponding ground truths for validating (i) deduplication performance within M, as well as (ii) novelty with respect to the different *KBs*. Three authors of this paper acted as experts and designed a coding frame to decide whether or not a fact is novel. After resolving disagreements on the coding frame on a subset of the data, every fact was associated with one expert label through manual deliberation. We followed the guidelines laid out by Strauss [35] during the coding process, which provide guidelines for designing reliable coding frames and carrying out manual coding and are frequently used to design coding frames and conduct qualitative analysis. Distinct facts were obtained by removing duplicate literals, null values, URLs and unresolved objects.

6.2.2. Metrics

We consider distinct metrics for evaluating each step of our approach.

- *Entity matching.* Precision P - the percentage of entity descriptions $e_i \in E$ that were correctly matched to e_q, R - the percentage of $e_i \in E_0$ that were correctly matched to KB, and the $F1$ score.
- *Correctness.* We evaluate the performance of the approaches through standard precision P, recall R and $F1$ scores, based on our ground truth.
- *Novelty.* We evaluate the performance of both substeps: i) deduplication with respect to M ($KnowMore_{ded}$), ii) deduplication with respect to a given KB ($KnowMore_{nov}$). For i), we compute $Dist\%$ - the percentage of distinct facts within the respective result set. We compare between $Dist\%$ (F_{ded}) and $Dist\%$ (F_{class}), that is, before and after the deduplication within M. For (ii), we measure the novelty as Nov - the percentage of novel facts - and compare between Nov (F_{ded}) and Nov (F'), that is, the novelty before and after this step. We also measure the recall R - the percentage of distinct and accurate facts in F_{ded} that have been selected by $KnowMore_{nov}$ into F'.
- *Coverage Gain.* We demonstrate the potential of our approach for augmenting a given KB by measuring the coverage gain. The coverage gain of predicate p is computed as the percentage of entity descriptions having p populated through the *KnowMore* approach with at least one fact $\langle p, o \rangle$, out of the ones that did not have at least one statement involving property p within the respective KB before augmentation. Note that according to this metric a coverage gain of less than 100% might not necessarily indicate non-optimal recall but might be caused by the non-applicability of attributes to a particular entity. For instance, not all entities of type *Movie* have a value for the property *award*. The result is reported in Section 7.4.

6.3. Configuration & Baselines

Configuration. We deploy our approach as described in Section 4.3 and 5. For the entity matching step, we use Lucene for indexing and BM25 retrieval with the Lucene default configuration where $k_1 = 1.2$, $b = 0.75$. For the deduplication with respect to KBs, we report the evaluation result of $KnowMore_{nov}$ using different $\tau = \{0.3, 0.5, 0.7\}$ in Section 7.3.

Baselines. We compare ($KnowMore_{class}$) with *PrecRecCorr* that is proposed by Pochampally et al. [28] and $CBFS$ [42]. To the best of our knowledge, the $CBFS$ approach is the only available method so far directly geared towards the challenges of markup data, while *PrecRecCorr* represents a recent and highly related data fusion baseline. We also present the results of $KnowMore_{match}$ and $KnowMore_{class}$ using different classifiers.

- $KnowMore_{class}$: facts selected based on the $KnowMore_{class}$ pipeline from F, using Naive Bayes as classifier for selecting correct facts.
- $KnowMore_{class}(SVM)$: facts selected based on the $KnowMore_{class}$ pipeline from F, using Support Vector Machine (linear kernel) as the classifier for selecting correct facts.
- $KnowMore_{class}(LR)$: facts selected based on the $KnowMore_{class}$ pipeline from F, using Logistic Regression (LR) as classifier for selecting correct facts.
- *PrecRecCorr*: facts selected based on the approach from candidate set F. We consider each PLD as a source and implemented the *exact solution* as described in the paper. We use the threshold as presented in the paper, i.e. 0.5, to classify facts.
- $CBFS$: facts selected based on the $CBFS$ approach from F. The $CBFS$ approach clusters the associated values at the predicate level into n clusters $(c_1, c_2, \cdots, c_n) \in C$. Facts that are closest to the cluster's centroid of each cluster are select, provided they meet the following criteria:

$$|c_j| > \beta \cdot \max(|c_k|), c_k \in C \tag{3}$$

where $|c_j|$ denotes the size of cluster c_j, and β is a parameter used to adjust the number of facts. In our experiment, β is empirically set to 0.5, which is the same as the best-performing setup as defined in the original paper

7. Evaluation Results

In this section, we present experimental results obtained through the setup described in the previous sections.

7.1. Entity Matching

While the entity matching step ($KnowMore_{match}$) is a precondition for the subsequent fusion step, we provide evaluation results for this step and compare it to entity descriptions obtained through BM25@k as baseline. The BM25 configuration is the same as used in our approach (Section 6.3). Since we obtain the corresponding URIs of Freebase and Wikidata entities through the *sameAs* link in DBpedia, here we present only the result of matching entity descriptions to DBpedia. Table 7 shows the evaluation results of the standard precision P, recall R and $F1$ scores.

As presented in Table 7, our supervised matching approach $KnowMore_{match}$ (using NB as classifier) achieves high $F1$ scores of 0.83 and 0.887 respectively, thereby outperforming the BM25@20 and BM25@50 baselines and providing a sound set of candidates for the subsequent step. Among different configurations of the $KnowMore_{match}$ approach, the Naive Bayes classifier achieves the highest precision and $F1$ score compared to the Logistic Regression (LR) classifier $KnowMore_{match}(LR)$ and SVM $KnowMore_{match}(SVM)$ on both types.

Table 7
Performance of $KnowMore_{match}$ and baselines.

	Movie			Book		
Approach	**P**	**R**	**F1**	**P**	**R**	**F1**
KnowMore$_{match}$	**0.943**	0.742	**0.83**	**0.88**	0.894	**0.887**
KnowMore$_{match}$(SVM)	0.583	**0.87**	0.698	0.824	**0.899**	0.86
KnowMore$_{match}$(LR)	0.627	0.645	0.636	0.821	0.851	0.836
BM25@10	0.659	0.652	0.655	0.325	0.533	0.404
BM25@20	0.592	0.831	0.692	0.219	0.722	0.336
BM25@50	0.406	1.000	0.578	0.124	1.000	0.220

7.2. Correctness - Data Fusion

The result of $KnowMore_{class}$ are shown in Table 8.

Table 8
Performance of $KnowMore_{class}$ and baselines.

	Movie			Book		
Approach	**P**	**R**	**F1**	**P**	**R**	**F1**
KnowMore$_{class}$	**0.954**	0.896	**0.924**	0.880	**0.868**	**0.874**
KnowMore$_{class}$(SVM)	0.845	**0.976**	0.906	0.810	0.799	0.804
KnowMore$_{class}$(LR)	0.894	0.946	0.919	0.889	0.809	0.847
PrecRecCorr	0.924	0.861	0.891	**0.893**	0.48	0.624
CBFS	0.802	0.752	0.776	0.733	0.842	0.784

As shown in Table 8, our chosen configuration, i.e. using a Naive Bayes classifier ($KnowMore_{class}$) achieves highest $F1$ scores among all the different configurations. Compared to the different configurations of our approach deploying different classification models, the precision is 2.5% higher (8.9%), and the $F1$ score is 1.6% higher (4.4%) than using LR (SVM).

The presented F1 score of the *PrecRecCorr* baseline is the best possible configuration for our given task, where we experimented with different thresholds ([0,1], gap 0.1) as discussed in [28] and identified 0.5 experimentally as the best possible configuration. We observe that the $F1$ score of our approach is 14.1% higher than *PrecRecCorr* and 11.9% higher than $CBFS$ on average across datasets. This indicates that our approach provides the most efficient balance between precision and recall across the investigated datasets, when applied to the novel task of data fusion from Web markup. Although, the precision of the baseline approach *PrecRecCorr* is 1.3% higher than the one from $KnowMore_{class}$ on the *Book* dataset, the baseline fails to recall a large amount of correct facts, where the recall of $KnowMore_{class}$ is approximately 38.8% higher. This also is reflected in the average size of entity descriptions obtained through both approaches, where the entity descriptions from *PrecRecCorr* consist of 4.88 statements on average, and the ones from $KnowMore_{class}$ are 8.83, indicating a larger potential for the KBA task. A more detailed discussion of the potential impact on the KBA task is provided in Section 9, investigating the KBA potential beyond the narrow definition of the investigated task of this setup, e.g. by augmenting additional predicates not already foreseen in a given KB schema or to populate KBs with additional entities.

We ran 20 iterations of 10-fold cross validation for different baselines and our approach in order to test the statistical significance of our results, as suggested in [8]. We conducted paired T-tests and employed Bonferroni's correction for Type-I error inflation. We found that all comparisons were statistically significant at the 95% confidence interval, with p<.05, with some comparisons significant at the 99% confidence interval; p<.01. Tables 7 and 8 reflect the average values over 20 runs of the corresponding algorithms.

7.3. Novelty

This section presents the evaluation results for the deduplication steps introduced in Section 5.2.

Diversity. Table 11 presents the evaluation result before ($Dist\%$ (F_{class})) and after ($Dist\%$ (F_{ded})) the step $KnowMore_{ded}$.

Table 9
Diversity $Dist\%$ before and after deduplication.

Dataset	**Dist%($\mathbf{F_{class}}$)**	**Dist%($\mathbf{F_{ded}}$)**
Movie	94.8	**96.1**
Book	82.1	**95.6**

The novelty of facts improves 1.3% for the *Movie* dataset and 13.5% for the *Book* dataset (Table 11). The less improvement gain for the *Movie* dataset presumably is due to the nature of the randomly selected *Movie* entities. As these appear to be mostly tail entities, candidate facts in our markup corpus M are fewer and less redundant. Hence, the amount of duplicates and near-duplicates is smaller, reducing the effect of the deduplication step. Hence, deduplication is of particular importance for popular and well-represented entities.

Novelty with respect to KB. The results before (Nov (F_{ded})) and after (Nov (F')) the deduplication for specific KBs using different similarity thresholds (τ) are presented in Table 10.

Since our approach is not aware of the total number of novel facts for a particular entity description on the Web a priori, in this evaluation, we consider all the novel facts in F_{ded} as the gold standard, and compute the recall of F' after applying the $KnowMore_{nov}$ accordingly. We evaluate the performance of $KnowMore_{nov}$ using τ in $\{0.3, 0.5, 0.7\}$ since (1) 0.5 is widely used as threshold for identifying duplicate text using cosine similarity, (2) for computing DateTime similarity between facts, the possible similarities according to our metric are: $\{0, 0.33, 0.5, 0.67, 1\}$. These 3 selected thresholds are most influential on the selection of DateTime facts, thus can produce most conclusive results for showing the trade-off between novelty and recall. As shown in Table 10, even though there is a trade-off between *novelty* and *recall*, different values of τ do not have a strong influence on the evaluation metrics. One of the reasons is that, a large proportion of facts have non-literal (e.g. numeric) values. While our datatype-specific similarity computes a binary (0 or 1) score in these cases, it is not influenced by the selection of τ.

Consider $\tau = 0.5$ as an example. The $KnowMore_{nov}$ step improves novelty by 28.2% on average across datasets and KBs compared to the result (F_{ded}) of the $KnowMore_{ded}$ step, and is able to recall over 90% of the novel facts. Our final result F' shows a novelty of over 90% on average, what translates to a minor amount of near-duplicates and a sufficient novelty for augmenting the target KBs.

7.4. Coverage Gain

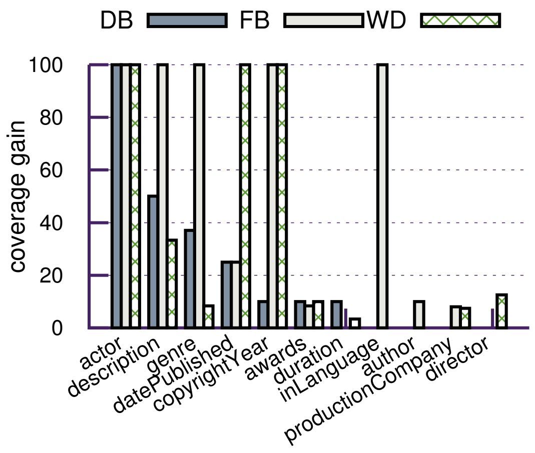
This section discusses the coverage gain, as an indicator of the *KnowMore* performance in the particular KBA task described in Section 3.2. While this constitutes a fairly narrow task, i.e. the augmentation of a selected set of attributes and entities, existing in all three investigated KBs, we discuss the overall KBA potential of Web markup data beyond our ground truth dataset in Section 9.1.

Figure 4 shows the coverage gain on the previously empty slots as shown in Figure 2 per predicate and KB for our selected 30 entities (per type). Based on

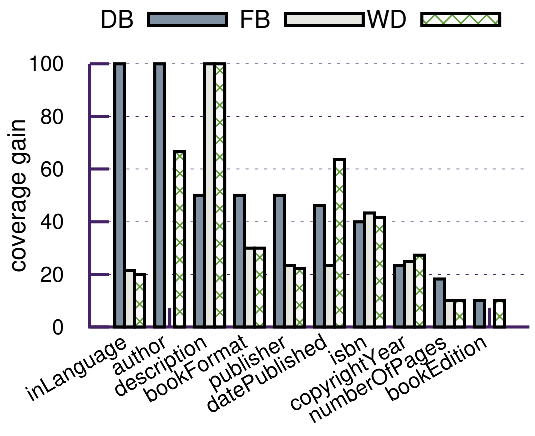
Table 10

Novelty of F_{ded} and F' with respect to target KBs.

	KB	Nov(F_{ded})	Nov($F', \tau = 0.3$)	Nov($F', \tau = 0.5$)	Nov($F', \tau = 0.7$)	R(F_{ded})	R($F', \tau = 0.3$)	R($F', \tau = 0.5$)	R($F', \tau = 0.7$)
Movie	**DBpedia**	0.631	**0.963**	0.962	0.962	1	0.927	**0.939**	**0.939**
	Freebase	0.527	**0.747**	0.742	0.742	1	0.942	**0.957**	**0.957**
	Wikidata	0.412	**0.929**	0.929	0.897	1	**0.963**	**0.963**	**0.963**
		Nov(F_{ded})	Nov($F', \tau = 0.3$)	Nov($F', \tau = 0.5$)	Nov($F', \tau = 0.7$)	R(F_{ded})	R($F', \tau = 0.3$)	R($F', \tau = 0.5$)	R($F', \tau = 0.7$)
Book	**DBpedia**	0.736	**0.962**	0.929	0.92	1	0.826	0.848	**0.870**
	Freebase	0.639	**0.915**	0.846	0.825	1	0.833	**0.846**	**0.846**
	Wikidata	0.705	**0.944**	0.933	0.923	1	0.791	0.814	**0.837**



(a) Movie



(b) Book

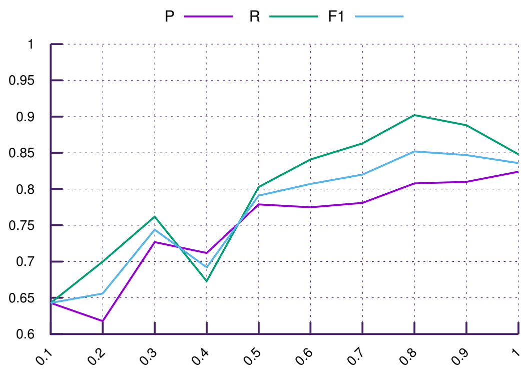
Fig. 4. Proportion of augmented entity descriptions with *KnowMore*. Only predicates which were augmented in at least one KB are shown.

the evaluation result, the *KnowMore* pipeline shows a coverage gain of 34.75% on average across different properties for DBpedia, 39.42% for freebase and 36.49% for Wikidata. We observe that the obtained gain varies strongly between predicates and entity types, with a generally higher gain for book-related facts. For instance, within the *Movie* case, for property *actor* we were able to gain 100% coverage in both DBpedia and Freebase, while the property *award* shows a coverage gain of 10% or less for all three KBs. Reasons behind low coverage gain for a particular property are 2-fold: 1) the lack of data in the Web markup data corpus, and 2) the lack of true facts in the real world for a particular attribute, e.g. only a small proportion of movies have won an award. On average, we obtained 2.8 (6.8) facts for each movie (book) entity in our experimental dataset.

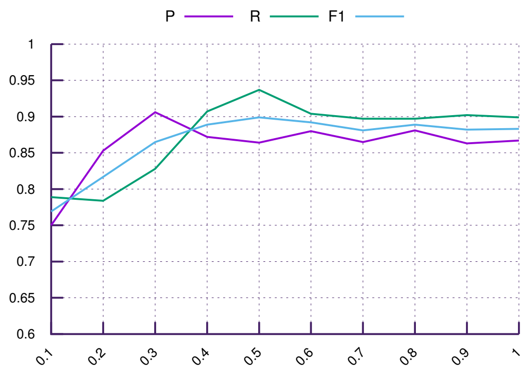
For a more thorough discussion of the KBA potential of the *KnowMore* approach, i.e. on tasks beyond our ground truth dataset, we refer the reader to Section 9.1.

8. Evaluation of Generalisation Potential

As introduced earlier, our approach has been trained on two specific types (*Book* and *Movie*). The intuition behind this choice is that (i) different properties have varied contribution on different types when computing the similarity between entity descriptions in the entity matching step and (ii) particular features such as predicate term t_1^p increase the feature space when introducing new types and associated properties. To this end, we have restricted our experiments to particular types to reduce the required training data. However, given the type-agnostic nature of most features, it seems reasonable to anticipate comparable performance even when applying our approach across types. In this section, we evaluate the generalisation of the *KnowMore* approach with respect to two aspects: 1) the scale of training data required for the supervised fusion step, 2) the performance of our approach when trained with cross-type training data, as opposed to type-specific sets.

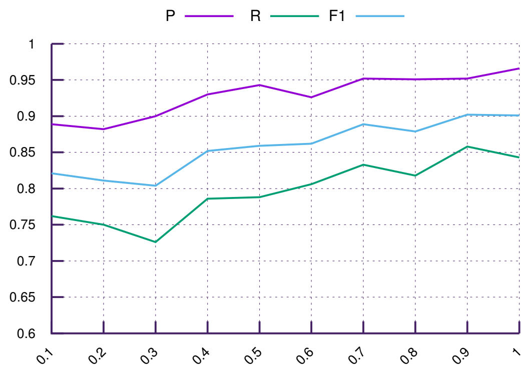


(a) *Movie*

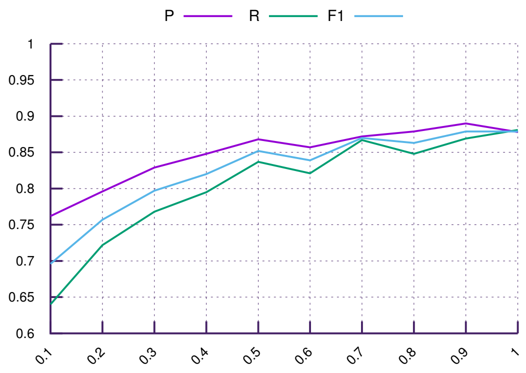


(b) *Book*

Fig. 5. P, R and F1 score using different size of the training data for $KnowMore_{match}$. X-axis shows the percent of training data, Y-axis shows the P/R/F1 value.



(a) *Movie*



(b) *Book*

Fig. 6. P, R and F1 score using different size of training data for $KnowMore_{class}$. X-axis shows the percent of training data, Y-axis shows P/R/F1 value.

8.1. *Scale of required Training Data*

As described in Section 6.2, our ground truth consists of judgments for 217 (298) entity descriptions for *Movie* (*Book*) entities for the entity matching step $KnowMore_{match}$, and 456 (371) facts of *Movie* (*Book*) entities for the data fusion step $KnowMore_{class}$. The experimental evaluation in Section 7 is based on the averaged P/R/F1 scores from a 10-fold cross validation (90% training and 10% testing). To evaluate how performance is affected by the scale of the training data, we have conducted our experiments with subsets of the data which vary in size. In particular, for each type, we run 10 experiments where the subset of the training data uses n percent of the original training data set and n is in the range of [10, 100]. 10-fold cross-validation is performed for each n.

Figure 5 presents the results for $KnowMore_{match}$, where the X-axis indicates the size of the training data and the Y-axis shows the P/R/F1 scores. The F1 score reaches 0.8 (0.88) at 60% (40%) percent of training data, and the P, R and F1 curves become steady with training data sets of the size of 80%(50%) for the *Movie* (*Book*) type, i.e. training data sets of at least 130 (119) entity descriptions for the *Movie* (*Book*) type.

Similar characteristics can be observed for the $KnowMore_{class}$ approach in Figure 6, where the F1 score reaches 0.89 (0.87) at 70% (70%) of training data for *Movie (Book)*. Hence, results suggest that even with comparably limited amounts of training data, reasonable performance can be achieved, thereby supporting the application across types and datasets. For instance, the cost for retrieving 80% (70%) of our entity matching (data fusion) ground truth from CrowdFlower with the approach as described in Section 6.2, is less than 15 USD and the time required is less than 24 hours for each type.

8.2. Model Performance across Types

In this section, we assess the performance of *KnowMore* across different types, i.e. without a type-specific training phase. Thus, we merge the aforementioned type-specific datasets *Book* and *Movie* and perform a 10-fold cross-validation using the query sets for both types. The averaged performance of $KnowMore_{match}$ on this cross type dataset is P = 0.782, R = 0.892, F1 = 0.833, where the precision is lower than the type-specific results (Section 7), but the overall performance and F1 score is still comparable, the latter being slightly above the F1 score of the type-specifically trained *Movie* model (0.83). The result of $KnowMore_{class}$ is P = 0.902, R = 0.825, F1 = 0.862. Where the precision is higher than the type-specific result for *Book* model (0.886) and lower than the one from the *Movie* model (0.967). This suggests that our approach can work on models trained on cross-type data. Further studies are required with more diverse query as well as datasets, involving larger amounts of types, to fully validate this finding.

9. Discussion & Limitations

9.1. Potential of KBA from Web Markup

Beside the specific KBA task evaluated in this paper, where we aim at (i) augmenting existing entities in a given KB by (ii) populating a given set of properties from a given KB schema for these entities, markup data shows large potential to augment KBs with properties and entities not yet present in KBs. Investigating the data from our two datasets (Movie, Book) and another set of 30 randomly selected entities of type *Product*, we observe that a large proportion of statements in the WDC dataset involve properties not yet present in any of the KB schemas. For instance, for movies (books), 62.5% (66.8%) of entity descriptions in F contain facts not yet present in our set of mapped predicates. Comparing product descriptions from F, we detect 20.6% statements containing properties not yet present in the DBpedia ontology at all (verified through manual inspection).

In order to better highlight the potential of Web markup data to support knowledge base augmentation, we apply *KnowMore* on all the movie and book entities from DBpedia. Out of all the 106,613 movie entities in DBpedia (10 Jan., 2017 version), we found coreferences in our markup corpus for 101,069 distinct entities (94.8%). Out of 35,577 book entities in DBpedia, we found coreferences for 34,964 entities in our markup corpus M (98.3%). In total, this resulted in 4,412,337 (1,783,231) instances, i.e. markup nodes, and 42,624,281 (16,580,862) candidate facts for the selected set of movies (books). On average, we found 5.06 (7.98) facts for each movie (book) instance. Based on the experimental results, our KnowMore approach obtained 511,409 (279,013) new facts for all DBpedia movie (book) entities. Note that this includes only entities already present in DBpedia. Although there is a wide variety of instances of both types which are not present in DBpedia, this could be used to populate DBpedia, in particular with less popular and long-tail entities.

In addition, we observe a considerable potential for augmentation of KBs with new entities, as opposed to augmenting existing ones. To assess performance in such cases, we randomly select 30 names of products under the requirement that each appears in at least 20 different PLDs in WDC, to ensure that there is sufficient consensus on the name being a legitimate product title. Manual inspection confirmed that none of such randomly selected products is represented in DBpedia.

By running the $KnowMore_{class}$ approach on 30 selected product entities, we found 136 correct facts in total, resulting in 4.53 facts for each entity on average. Table 11 shows the performance of $KnowMore_{class}$ and our baselines on this dataset.

Table 11
Data fusion performance for *Product* entities.

Approach	**P**	**R**	**F1**
KnowMore$_{class}$	**0.983**	**0.927**	**0.954**
PrecRecCorr	0.827	0.485	0.611
CBFS	0.876	0.686	0.769

Results indicate that the performance gain of our approach is particularly evident on such long-tail entities as represented in our *Product* dataset.

9.2. Limitations

Results demonstrate that *KnowMore* is able to exploit Web markup data for KBA tasks. Further improvement can be gained by applying our approach on a focused crawl, targeted towards a specific KBA task, such as movie enrichment, rather than a cross-domain Web crawl such as the WDC/Common Crawl.

In contrast to related KBA approaches such as [16] or [39], it is worth noting that our approach is trained for particular entity types only, not towards particular properties, as is the case with the aforementioned approaches. Hence, *KnowMore* can be adapted to a wider range of scenarios with less effort than previous KBA approaches. In addition, we have demonstrated in Section 8.2 that our models can potentially generalise across types.

Performance strongly differs between query sets, and hence, type-specific markup datasets what presumably is caused by the variance in quality and quantity of facts in the WDC corpus between distinct types. Particular challenges arise from entities with a large amount of coreferences, where data usually originates from a wide variety of sources with varying degrees of quality. Compared to the baselines, our results indicate a particular strong performance gain of our approach in such cases.

One limitation is our exclusive focus on *schema.org* statements. This constraint is motivated by the costliness of providing high-quality schema mappings between markup statements and three KBs and the fact that *schema.org* is the vocabulary of most widespread use [23]. While *schema.org* adopters usually are motivated by the goal to improve their search result rankings, one assumption is that other vocabularies might show a different distribution of types and predicates, due to distinct motivations. This deserves deeper investigation as part of future work.

It is also worth noting that our KBA task setup ignored a large part of the markup data, i.e. 49.3% of facts in our type-specific subsets do not involve any of our selected *schema.org* properties. To consider other vocabularies, we are currently aiming at including a preliminary schema matching step with the intention of improving recall further.

Another important aspect concerns the temporal nature of fact correctness, specifically for highly dynamic predicates, such as the price tag of a particular product. While we do not consider temporal features as such, we argue that the dynamic nature of markup annotations is well-suited to augment particularly dynamic statements. This suggests particular opportunities for updating or complementing KBs with dynamic knowledge sourced from Web markup.

10. Conlusions and Future Work

We have introduced *KnowMore*, an approach towards knowledge base augmentation from large-scale Web markup data, based on a combination of entity matching, data fusion and deduplication techniques. We apply our method to the WDC2015 corpus as largest publicly available Web markup crawl (approx. 20 billion quads) and augment three established knowledge base<s, namely Wikidata, Freebase and DBpedia. Evaluation results suggest superior performance of our approach with respect to non-redundant as well as correctness compared to state-of-the-art data fusion baselines, with an F1 score increase of 11.7% respectively 6.5% compared to the two baselines across datasets. Our experimental results indicate comparably consistent performance across a variety of types, whereas the performance of baseline methods tends to vary strongly.

Our evaluation of the KBA task on two types demonstrates a strong potential to complement traditional knowledge bases through data sourced from Web markup. We achieve a 100% coverage for particular properties, while providing substantial contributions to others. In addition, we demonstrate the capability to augment KBs with additional entity descriptions, particularly about long-tail entities, where for randomly selected entities of type *Product* from WDC, we are able to generate new entity descriptions with an average size of 6.45 facts.

While our experiments have exploited the WDC corpus, we will consider more targeted Web crawls, which are better suited to augment entities (or properties) of a particular type or discipline. Here, targeted datasets which are retrieved with the dedicated aim to suit a particular KBA task are thought to further improve the KBA performance. Another identified direction for future research is the investigation of the complementary nature of other sources of entity-centric Web data, for instance, data sourced from Web tables, when attempting to augment KBs.

Additional objectives for future work have surfaced during the experiments. For instance, identity resolution problems might occur during the matching step originating from different meanings of a particular entity. Current work aims at pre-clustering result sets into distinct entity meanings, from which we will be able to augment distinct disambiguated entity descriptions. Finally, we are investigating an iterative approach which enables the generation of entity-centric knowledge graphs of a certain length (*hop-size*), rather than flat entity descriptions. This would further facilitate research into the generation of domain or type-specific knowledge graphs from distributed Web markup.

11. Acknowledgments

This work has been partially funded by the European Union's Horizon 2020 research and innovation program (Grant No. 687916 - AFEL project). The authors would like to thank Robert Meusel for his inspiring feedback and support with the presented work.

References

[1] A. P. Aprosio, C. Giuliano, and A. Lavelli. Extending the coverage of DBpedia properties using distant supervision over wikipedia. In *NLP-DBPEDIA*, 2013.

[2] J. Bleiholder and F. Naumann. Data fusion. *ACM Computing Surveys (CSUR)*, 41(1):1, 2009.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD/PODS*, 2008.

[4] V. Bryl and C. Bizer. Learning conflict resolution strategies for cross-language Wikipedia data fusion. In *WWW*, 2014.

[5] L. Bühmann and J. Lehmann. Universal OWL axiom enrichment for large knowledge bases. In *EKAW*. Springer, 2012.

[6] L. Bühmann and J. Lehmann. Pattern based knowledge base enrichment. In *ISWC*, 2013.

[7] V. Christophides, V. Efthymiou, and K. Stefanidis. Entity resolution in the web of data. *Synthesis Lectures on the Semantic Web*, 5(3):1–122, 2015.

[8] T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

[9] S. Dietze, D. Taibi, R. Yu, P. Barker, and M. d'Aquin. Analysing and improving embedded markup of learning resources on the web. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, pages 283–292, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee.

[10] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *SIGKDD*, 2014.

[11] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, K. Murphy, S. Sun, and W. Zhang. From data fusion to knowledge fusion. *VLDB*, 2014.

[12] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang. Knowledge-based trust: Estimating the trustworthiness of web sources. *Proceedings of the VLDB Endowment*, 8(9):938–949, 2015.

[13] A. Dutta, C. Meilicke, and H. Stuckenschmidt. Enriching structured knowledge with open information. In *WWW*, 2015.

[14] D. Gerber, S. Hellmann, L. Bühmann, T. Soru, R. Usbeck, and A.-C. N. Ngomo. Real-time RDF extraction from unstructured data streams. In *ISWC*, 2013.

[15] H. Ji and R. Grishman. Knowledge base population: Successful approaches and challenges. In *ACL*, 2011.

[16] P. H. Kanani and A. K. McCallum. Selecting actions for resource-bounded information extraction using reinforcement learning. In *WSDM*, 2012.

[17] T. Knap, J. Michelfeit, J. Daniel, P. Jerman, D. Rychnovskỳ, T. Soukup, and M. Nečaskỳ. ODCleanstore: A framework for managing and providing integrated Linked Data on the web. In *WISE*. Springer, 2012.

[18] K. Krippendorff. Computing krippendorff's alpha reliability. *Departmental papers (ASC)*, page 43, 2007.

[19] J. Lehmann, S. Auer, L. Bühmann, and S. Tramp. Class expression learning for ontology engineering. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1):71–81, 2011.

[20] P. N. Mendes, H. Mühleisen, and C. Bizer. Sieve: Linked data quality assessment and fusion. In *EDBT/ICDT Workshops*. ACM, 2012.

[21] R. Meusel and H. Paulheim. Heuristics for fixing common errors in deployed schema.org microdata. In *ESWC*, 2015.

[22] R. Meusel, P. Petrovski, and C. Bizer. The WebDataCommons microdata, RDFa and microformat dataset series. In *ISWC*. 2014.

[23] R. Meusel, D. Ritze, and H. Paulheim. Towards more accurate statistical profiling of deployed schema.org microdata. In *ACM Journal of Data and Information Quality*, volume 8, 2016.

[24] J. Michelfeit and T. Knap. Linked Data fusion in ODCleanstore. In *ISWC*, 2012.

[25] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL & AFNLP*, 2009.

[26] H. Paulheim and S. P. Ponzetto. Extending DBpedia with Wikipedia list pages. In *NLP-DBPEDIA*, 2013.

[27] D. Pelleg, A. W. Moore, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML*, 2000.

[28] R. Pochampally, A. Das Sarma, X. L. Dong, A. Meliou, and D. Srivastava. Fusing data with correlations. In *SIGMOD*. ACM, 2014.

[29] P. Ristoski and P. Mika. Enriching product ads with metadata from HTML annotations. In *ISWC*, 2016.

[30] D. Ritze, O. Lehmberg, and C. Bizer. Matching HTML tables to DBpedia. In *WIMS*, 2015.

[31] D. Ritze, O. Lehmberg, Y. Oulabi, and C. Bizer. Profiling the potential of web tables for augmenting cross-domain knowledge bases. In *WWW*, 2016.

[32] P. Sahoo, U. Gadiraju, R. Yu, S. Saha, and S. Dietze. Analysing structured scholarly data embedded in web pages. In *WWW Companion*, 2016.

[33] A. Schultz, A. Matteini, R. Isele, C. Bizer, and C. Becker. Ldif-linked data integration framework. In *COLD*, 2011.

[34] R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, 2013.

[35] A. L. Strauss. *Qualitative Analysis for Social Scientists*. Cambridge University Press, 1987.

[36] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *WWW*, 2007.

[37] A. Tonon, V. Felder, D. E. Difallah, and P. Cudré-Mauroux. VoldemortKG: Mapping schema. org and web entities to linked open data. In *ISWC*, 2016.

[38] G. Weikum and M. Theobald. From information to knowledge: Harvesting entities and relationships from web sources. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 65–76. ACM, 2010.

[39] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin. Knowledge base completion via search-based question answering. In *WWW*, 2014.

[40] R. Yu, B. Fetahu, U. Gadiraju, and S. Dietze. A survey on challenges in web markup data for entity retrieval. In *ISWC*, 2016.

[41] R. Yu, U. Gadiraju, B. Fetahu, and S. Dietze. FuseM: Query-centric data fusion on structured web markup. In *ICDE*, pages 179–182. IEEE Computer Society, 2017.

[42] R. Yu, U. Gadiraju, X. Zhu, B. Fetahu, and S. Dietze. Entity summarisation on structured web markup. In *ESWC: Satellite Events*, 2016.

[43] B. Zhao, B. I. Rubinstein, J. Gemmell, and J. Han. A bayesian approach to discovering truth from conflicting sources for data integration. *VLDB*, 2012.