Inferring Missing Categorical Information in Noisy and Sparse Web Markup

Nicolas Tempelmeier, Elena Demidova, Stefan Dietze
L3S Research Center, Leibniz Universität Hannover
Hannover, Germany
{tempelmeier,demidova,dietze}@L3S.de

ABSTRACT

Embedded markup of Web pages has seen widespread adoption throughout the past years driven by standards such as RDFa and Microdata and initiatives such as schema.org, where recent studies show an adoption by 39% of all Web pages already in 2016. While this constitutes an important information source for tasks such as Web search, Web page classification or knowledge graph augmentation, individual markup nodes are usually sparsely described and often lack essential information. For instance, from 26 million nodes describing events within the Common Crawl in 2016, 59% of nodes provide less than six statements and only 257,000 nodes (0.96%) are typed with more specific event subtypes. Nevertheless, given the scale and diversity of Web markup data, nodes that provide missing information can be obtained from the Web in large quantities, in particular for categorical properties. Such data constitutes potential training data for inferring missing information to significantly augment sparsely described nodes. In this work, we introduce a supervised approach for inferring missing categorical properties in Web markup. Our experiments, conducted on properties of events and movies, show a performance of 79% and 83% F1 score correspondingly, significantly outperforming existing baselines.

ACM Reference Format:
Nicolas Tempelmeier, Elena Demidova, Stefan Dietze. 2018. Inferring Missing Categorical Information in Noisy and Sparse Web Markup. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186028

1 INTRODUCTION

Web search in general and the interpretation of Web documents in particular are increasingly being supported through semi-structured, entity-centric knowledge. For instance, publicly available knowledge graphs (KGs) such as Freebase [4] or YAGO [23] as well as proprietary KGs used by Google or Microsoft [18, 21] are key ingredients when interpreting search queries as well as Web documents. More recently, Web markup facilitated through standards such as RDFa[1], Microdata[2] and Microformats[3] has become prevalent on the Web, driven by initiatives such as *schema.org*, a joint effort led by Google, Yahoo!, Bing and Yandex.

For instance, the Web Data Commons (WDC) project [16] that releases markup extracted from the Common Crawl[4], found that in 2016 39% out of 3.18 billion HTML pages from over 34 million pay-level-domains (plds) contain some form of embedded markup, resulting in a corpus of 44.24 billion RDF quadruples[5]. There is an upward trend of Web markup adoption, where the proportion of pages containing markup increased from 5.76% to 39% between 2010 and 2016.

To this extent, markup data provides an unprecedented and growing source of explicit entity annotations to be used when interpreting and retrieving Web documents, to complement annotations otherwise obtainable through traditional information extraction pipelines, or to train information extraction methods. In addition, while traditional KGs capture large amounts of factual knowledge, they still are incomplete, i.e. 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 are usually insufficiently represented [3]. In this context, markup also provides essential input when incrementally augmenting and maintaining KGs [29], in particular when attempting to complement information about long-tail properties and entities [30].

The specific characteristics of statements extracted from embedded Web markup pose particular challenges [28]. Whereas coreferences are very frequent (for instance, in the WDC 2013 corpus, 18,000 entity descriptions of type *schema.org:Product* are returned for the query `'Iphone 6'`), these are not linked through explicit statements. In contrast to traditional densely connected RDF graphs, markup statements mostly consist of isolated nodes and small subgraphs, each usually made up of small sets of statements per entity description. In addition, extracted RDF markup statements are highly redundant and are often limited to a small set of highly popular predicates, such as *schema.org:name*, complemented by a long tail of less frequent statements. Moreover, data extracted from markup contains a wide variety of errors [17], ranging from typos to the frequent misuse of vocabulary terms [15]. Hence, individual markup extracted from a particular Web document or crawl usually contains very limited or unreliable information about a particular entity. According to our analysis, out of 26 million annotated events in the WDC 2016 corpus, less than 257,000 (0.96%) indicate a more specific event subtype and 59% nodes provide less than six statements. This strongly limits the meaningfulness of Web markup, in particular for entities that cannot be mapped to a representation in an existing knowledge graph.

In this work, we introduce an approach to automatically infer missing categorical information for particular entities obtained from Web markup. Building on the Web-scale availability of markup, and hence, the abundance of potential training data for the task, we

WWW 2018, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5639-8/18/04..
https://doi.org/10.1145/3178876.3186028

introduce a supervised method to efficiently infer missing categorical information from existing entity markup describing coreferring or similar entities. Our experiments address the inference of entity (sub-)types, as well as inference of arbitrary non-hierarchical predicates, such as movie genres. We demonstrate superior performance compared to both naive baselines as well as specialised state-of-the-art methods for type inference and achieve F1 scores of 79% and 83% in two experimental tasks.

2 MOTIVATION AND PROBLEM DEFINITION

2.1 Motivation

Microdata and RDFa markup are used to embed semi-structured data about entities within Websites. While being leveraged to facilitate interpretation and retrieval of Websites by most major search engines, markup data is also used to maintain and augment knowledge graphs, where additional applications include Google Rich Snippets, Pinterest Rich Pins and search features for Apple Siri [11]. By today, Web markup data is available at an unprecedentedly large scale, which can be exemplarily observed on the *Web Data Commons* (WDC) [16] corpus. WDC[6] offers a large-scale corpus of RDF quadruples extracted from the *Common Crawl*[7]. The crawl of October 2016 contains $3.18 \cdot 10^9$ URLs of which 39% exhibit markup from which over $4.4 \cdot 10^{10}$ triples were extracted. In contrast, the crawl of November 2015 contains $1.77 \cdot 10^9$ URLs of which only 31% exhibit markup, resulting in only about $2.4 \cdot 10^{10}$ extracted triples[8]. *Schema.org* is a joint initiative from major search engines such as Bing, Google, Yahoo! and Yandex that provides a joint vocabulary and is the most commonly deployed vocabulary on the Web [3]. In the following we abbreviate the prefix of the *schema.org* vocabulary by *s:*, e.g. *s:Movie*. We refer to the WDC corpus from October 2016 as the WDC 2016 corpus.

While Web markup constitutes an unprecedented source of semi-structured knowledge, markup is usually sparse and highly redundant, consisting of vast amounts of coreferences and (near) duplicate statements [29]. Individual entities extracted from Web markup usually are sparsely described, such that only a fraction of the properties foreseen by *schema.org* for a specific type is provided, often only providing a label and a type for a specific node. Table 1 provides an overview of the number of quadruples per single node for specific types (*s:Event*, *s:Movie*) in the WDC 2016 corpus. The property distribution follows a power law, where a small set of terms is very prevalent, yet the majority of properties is hardly used across the Web. Figure 1 shows the top-20 most frequently used properties of movies, highlighting that certain properties occur very often (e.g. *s:actor*) while others are provided rarely, such as *s:productionCompany*.

Sparsity is exacerbated by the lack of connectivity of markup data, where controlled vocabularies, taxonomies, and essentially, links among nodes are hardly present. Previous studies [7] on a specific markup subset find that, out of a set of 46 million quadruples involving transversal, i.e. non-hierarchical properties, approximately 97% actually refer to literals rather than URIs, that is object nodes. These findings underline that markup data largely consists of rather isolated nodes, which are linked through common schema terms (as provided by *schema.org*) at best, but commonly lack relations at the instance level. In particular for categorical information,

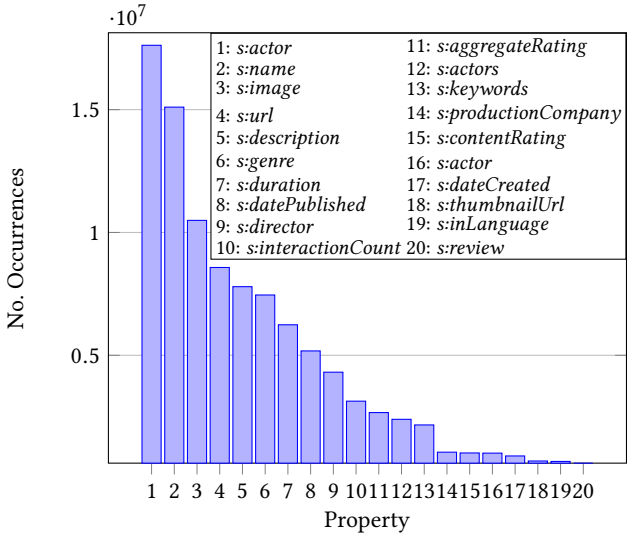

Figure 1: Top-20 most frequent properties for the type *s:Movie* in WDC 2016. The second entry of *s:actor* is caused by erroneous annotations in Web markup.

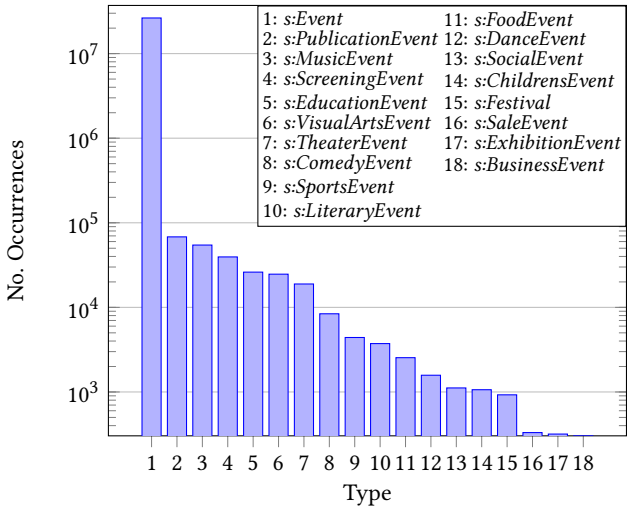

Figure 2: Number of occurrences of schema.org event types in WDC 2016 (Y-axis is logarithmic).

such as movie genres or product categories, this poses a crucial challenge when it comes to interpreting such information.

A particular instantiation of the aforementioned problem is the use of unspecific types. Figure 2 illustrates the number of instances of events annotated with respective event subtypes. Note that assignment of multiple types is theoretically possible, but rarely used in practice (i.e. less than 0.1% of events have multiple types). Apparently, most of the instances are assigned the generic type *s:Event*, while only 0.96% of nodes use more specific types like *s:TheaterEvent* or *s:Festival*, hindering data interpretation.

Table 1: Number of quadruples per node for specific types in WDC 2016.

Type	Total No. Quadruples	Total No. Nodes	Quadruples				Distinct Properties			
			Min.	Max	Avg.	Median	Min.	Max.	Avg.	Median
s:Event	$1.58 \cdot 10^8$	$2.66 \cdot 10^7$	1	2889	5.55	5	1	32	5.31	5
s:Movie	$1.25 \cdot 10^8$	$1.62 \cdot 10^7$	1	4547	7.71	6	1	26	5.77	6

Whereas individual markup nodes are usually sparsely annotated, markup as a whole provides a rich source of data, where in particular for categorical, i.e. discrete, properties a wide variety of instances can be drawn from the long tail. For instance, referring to Figure 2, while only 0.96% of all event nodes are typed with a meaningful subtype, this still corresponds to a set of 257,000 nodes available as training data to build supervised models to classify the remaining 26 million insufficiently typed events. Hence, we follow the intuition that markup data can significantly benefit from supervised approaches, which learn categorial or discretised properties as a means to infer missing categorical information for sparsely annotated nodes, i.e. to enrich markup entities. Overall, augmentation of sparse Web markup nodes can contribute to the improvement of the interpretability of the markup, the enrichment of knowledge graphs, and hence, to the effectiveness of the applications using the markup. This includes search and Web page classification, where in particular categorical and type information is essential to correctly interpret resources.

2.2 Problem Definition

This work aims at inferring missing categorical information in data sourced from Web markup. For a given corpus of Websites C, Q_C denotes the set of *RDF quadruples* of the form (s,p,o,u) extracted from the corpus, where s,p,o represent an RDF triple, i.e. a statement, of the form subject, predicate and object and u represents the URL of the Web document, from which the triple has been extracted.

A *vocabulary* V consists of a set of *types* T and *properties* P. A particular property $p_i \in P$ has a declared domain $d(p_i)$ that defines the set of expected types $T_i \subseteq T$ a subject involved in the same triple with p_i is meant to be an instance of. The range $r(p_i)$ of a property p_i defines the expected types an object involved in the same triple as p_i is meant to be an instance of.

For instance, within the *schema.org* vocabulary, the domain of the property *translator*[9] is defined as instances of type *Event*[10] and *CreativeWork*[11], while the declared range is defined as instances of type *Organization*[12] and *Person*[13].

Definition 2.1. Given a vocabulary V, a set of quadruples Q_C, for a particular node representing a subject $s_i \in Q_C$, this work aims at predicting quadruples $q = (s_i, p_i, o_i, u_i)$ which are: (a) not present in the markup corpus ($q \notin Q_C$), (b) valid according to the definition of vocabulary V, and (c) a valid statement about subject s_i in the context of u_i.

The last requirement of the aforementioned definition is experimentally evaluated according to a ground truth G, where an example is described in Section 4.1.

Note that our work focuses on *categorical* properties, i.e. we consider properties where the corresponding range $r(p)$ is *finite*.

For instance, consider the following markup triple, extracted from the URL http://www.imdb.com/title/tt0109830/?ref_=tt_trv_cnn describing the movie "Forrest Gump":

$$\left[\begin{array}{l|l} s: & \textit{\_:nodea73846c741abe988abf1c682f1fe26e7} \\ p: & \textit{rdf:type} \\ o: & \textit{s:Movie} \end{array}\right]$$

For the specific subtask of predicting movie genres (Section 4), we aim at predicting the quadruple involving the following triple (URL omitted) stating the genre of the movie:

$$\left[\begin{array}{l|l} s: & \textit{\_:nodea73846c741abe988abf1c682f1fe26e7} \\ p: & \textit{s:genre} \\ o: & \textit{"Drama"} \end{array}\right]$$

3 APPROACH

The characteristics of the data at hand suggest that, for most subjects s_i which are to be augmented, e.g. the movie mentioned in the previous example, sufficient training data can be obtained (Section 2). That means, we anticipate that a sufficient number of entity descriptions (instances) exist, which share the same missing categorical property p_i, e.g. a movie genre in the example above. Thus, we approach the inference problem as a supervised classification problem, where nodes which share the sought after property p_i are used as training data to build a model for the prediction of respective statements. This section describes our approach, namely the steps taken for data cleansing, feature extraction and building classification models.

3.1 Data Cleansing

Based on studies on common errors on deployed microdata [15], we applied the following heuristics proposed in [15], to improve the quality of the dataset by fixing the following errors:

Wrong namespaces: Many terms that deviate from the correct *schema.org* namespace can be corrected by adding missing slashes, changing *https://* to *http://*, removing additional substrings between *http://* and *schema.org* and fixing capitalisation errors.

Undefined properties and types: The use of wrong capitalisation of property and type names leads to the presence of undefined terms in markup data. We corrected the capitalisation by using the capitalisation defined by the *schema.org* vocabulary.

Applying these heuristics aids the feature extraction and classification steps described below by providing a larger amount of training data as well as by improving feature quality.

3.2 Feature Extraction

This section describes the considered features for our task and the applied feature extraction.

pld/tld: Based on the assumption that many Web domains are specialised on particular topics, e.g. concerts or documentary films, we employ domain-based features. The intuition is that any particular *pay-level-domain* (pld) and/or *top-level-domain* (tld) usually correlates with particular categorical properties, such as the types of covered events. Thus, for each node, we extract the pld and the tld from the URL of the Web page. For instance, taking into account the task of predicting event subtypes, consider the quadruple:

$$\begin{bmatrix} s: & \textit{\_:node396540c21b6fa0388c7293ebe216583} \\ p: & \textit{rdf:type} \\ o: & \textit{s:Event} \\ u: & \textit{<http://www.touristlink.com/india/cat/events.html>} \end{bmatrix}$$

From this quadruple we extract the pld "touristlink.com" and the tld ".com" from u and use these as features to predict the subtype "*s:MusicEvent*" of the described event. The plds and tlds are mapped into feature space via *1-hot-encoding*[14], resulting in one dimension for each pld and each tld.

node-vocab: The intuition behind this feature is that there is a correlation between the used vocabulary terms and the specific classes we aim to predict. For example, a composer (*s:composer*) is more likely to be provided for a music event (*s:MusicEvent*) than for a sports event. Following this intuition, Paulheim et al. [19] proposed an approach for entity type prediction using vocabulary term correlations. To this extent, they made use of the outgoing and incoming statements of the node n for type prediction of n in knowledge graphs (i.e. statements that have n either in the subject or the object position, respectively). In case of Web markup, it may not be feasible to determine all incoming statements for a given subject at Web scale. Therefore, in this work we make use of the outgoing statements only and use these statements to predict categorical properties of the entity described through the node n. More specifically, for all quadruples Q_n involving subject n, we extract all *schema.org* terms used as predicate. For each node n, we compute a frequency vector, where each dimension corresponds to a vocabulary term t_i and each value is the normalised number of times t_i occurs in a quadruple with n as a subject. The frequencies are normalised using the l^2 (euclidean) norm.

Example 3.1. For the node s and URL u

$$\begin{bmatrix} s: & \textit{\_:node3957c770b4f7c0bd1a17805dd8ca406} \\ u: & \textit{<https://gdssummits.com/nghealthcare/us/>} \end{bmatrix}$$

the following tuples are present:

$$\begin{bmatrix} p: & \textit{rdf:type} \\ o: & \textit{<http://schema.org/BusinessEvent>} \end{bmatrix}$$

$$\begin{bmatrix} p: & \textit{s:Event/name} \\ o: & \textit{"NG Healthcare Summit US"@en} \end{bmatrix}$$

$$\begin{bmatrix} p: & \textit{s:Event/location} \\ o: & \textit{"Omni Barton Creek Resort \& Spa, Austin, Texas"@en} \end{bmatrix}$$

These tuples result in the following node-vocab: {*rdf:type*:1, *s:Event/name*:1, *s:Event/location*:1}.

Note that we concatenated the predicate and the type used as the domain of the predicate. This way we ensure that: (a) types as well as terms are considered and (b) the connection between a predicate and its observed domain is preserved. The latter appears useful, considering that *schema.org* terms are used in a variety of contexts, often in ways other than recommended by the vocabulary definition, e.g. by violating domain and range definitions [7].

page-vocab: The vocabulary used on a Web page within which a subject appears intuitively correlates with categorical classes associated with nodes on the respective page. For instance, Websites discussing music albums are more likely to also contain music events rather than sports events. To take this context into account, we consider all *schema.org* vocabulary terms that appear as predicates on the same Web page as the node under consideration as a feature. Similar to the node-vocab, we create a frequency vector normalised using the l^2 (euclidean) norm.

Example 3.2. Assume that in addition to the quadruples in Example 3.1, the following triples are present on the same Web page:

$$\begin{bmatrix} s & \textit{\_:nodea9ff152514bcfb63c2714bc1336b2b3} \\ p: & \textit{s:Organization/url} \\ o: & \textit{<http://www.gdsinternational.com>} \end{bmatrix}$$

$$\begin{bmatrix} s & \textit{\_:node4ccbf7f34c95f14168f5fdb47b73ab} \\ p: & \textit{rdf:type} \\ o: & \textit{s:BusinessEvent} \end{bmatrix}$$

Then the terms from these quadruples are added to the node-vocab to form the page-vocab: {*rdf:type*:2, *s:Event/name*:1, *s:Event/location*:1, *s:Organization/url*:1}.

After computing the individual features, all features are concatenated to form a single feature vector. Finally, the feature vectors are normalised, i.e. the mean is removed and the features are scaled to unit variance. The feature vectors serve as input for supervised machine learning approaches that are detailed in Section 3.3.

3.3 Classification Models

We compare the use of the following classifiers:

Naïve Bayes: A Gaussian Naïve Bayes classifier that assumes that the likelihood of the features follows a Gaussian distribution. Since the features are normalised (i.e. may have negative values), a multinomial Naïve Bayes can not be applied. Naïve Bayes classifiers are known to be adoptable to many classification tasks.

Decision Tree: A classifier that successively divides the feature space to maximise a given metric (e.g. Gini Impurity, Information Gain). Decision Trees are able to identify discriminative features within high-dimensional data.

Random Forest: A classifier that utilises an ensemble of uncorrelated decision trees. Random Forests can utilise a large amount of training data that is likely to be found in Web crawls.

SVM: A Support Vector Machine with a linear kernel. SVMs have been applied to a large variety of classification problems.

4 EVALUATION SETUP

While our approach is independent of the respective categorical information to be inferred, we conducted an evaluation in two specific tasks: (1) predicting subtypes of *s:Event* instances, and (2) predicting genres (*s:genre*) of *s:Movie* instances.

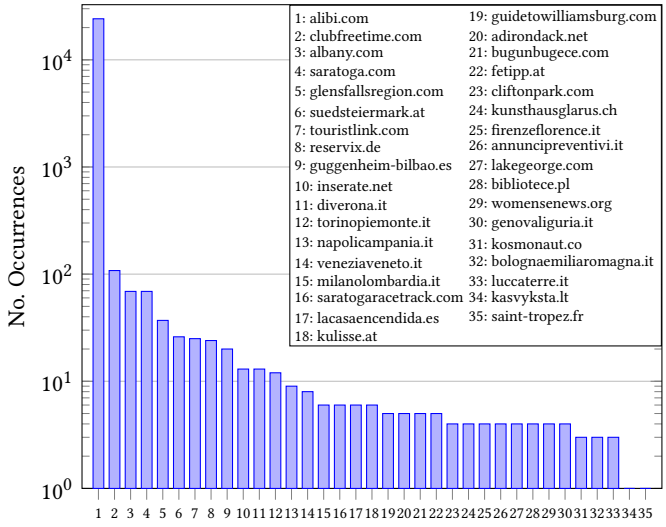

Figure 3: tld/pld-distribution of *s:VisualArtsEvent*s. Y-axis is logarithmic.

4.1 Datasets

Training and test datasets were extracted from the Web Data Commons dataset of October 2016.

Event Classification: This task deals with the prediction of event subtypes. *Schema.org* distinguishes between 19 different event subtypes, such as *s:BusinessEvent* or *s:SportsEvent*. Given a generic event, the goal of this task is to predict the correct subtype of the event, i.e. to predict the object of the *rdf:type* statement.

Movie Genre Classification: *Schema.org* allows annotation of movie genres via the *s:genre* property. The goal of this classification task is to predict statements describing the *s:genre* of respective movies. Since it is possible to assign multiple genres to a single movie by defining multiple *s:genre* properties, the classification of movie genres is a *multi-label problem*, i.e. a single movie entity can belong to multiple genre classes. We address this multi-label problem by extracting individual datasets for each genre upon which a binary classifier for each genre is trained.

4.1.1 Balancing and Sampling. We extracted quadruples that exhibit the respective property of interest by selecting quadruples which describe nodes of *rdf:type s:Event* (*s:Movie*) and are annotated with a more specific event subtype in the case of events and the *s:genre* predicate for movies. This results into a single *Events* dataset (containing instances of all considered subtypes) and an individual dataset for each movie genre. As illustrated in Figure 2, the class distribution is uneven.

To obtain a balanced dataset that is sufficiently large for training of a machine learning algorithm, we applied the following steps. For *Events*, we picked the top-7 classes with the highest number of instances. We introduced an additional class containing all events not included in the top-7 classes. The classes were balanced by limiting the size of all classes to c_e, which is the size of the smallest class. For *Movies*, we extracted 7 individual datasets corresponding to the top-7 most frequent movie genres. Each individual genre dataset includes all instances of the particular genre as well as all the remaining instances, which are labeled as "Other". The size of each genre datasets is limited to c_m, which is the size of the smallest class among all 7 datasets.

We employed two different sampling strategies:

1) *Stratified Random Sampling* simply chooses c_e (c_m) instances of each class at random from the whole dataset.

2) *pld-Aware Sampling*: Figure 3 depicts the pld distribution of *s:VisualArtsEvent*s. The distribution follows a power law, such that a small set of plds provides the majority of events. Random sampling may result in dropping some of the plds with fewer events and overfitting towards the patterns exhibited by very prominent plds. Therefore, we employ a sampling approach that ensures representation of long-tail entities in the sample. To this extent, we calculate a *fair share* in the sample by dividing the number of instances by the numbers of plds. We add all instances from plds that have fewer instances than the *fair share*. This process is repeated with recalculating the *fair share* with respect to the number of missing instances until the dataset contains c_e (c_m) instances of each class, where c_e (c_m) is the number of instances of the smallest class in the case of events (movies). If all remaining plds contain more instances than the *fair share*, each pld contributes the *fair share* to the final sample.

After the sampling, we split each resulting dataset in an individual training and test set (80% / 20% of the instances).

4.1.2 Labeling & Ground Truth. We follow a dataset-specific strategy to obtain class labels, i.e. a ground truth for training and testing. For assigning event types, we rely on the event subtypes defined within the *schema.org* type hierarchy. The class labels for events are thus explicitly given by the *rdf:type*-statements.

With respect to the prediction of movie genres, no controlled vocabulary is used consistently, whereas literals are used widely. Therefore, we map the literals to a unified genre taxonomy. We make use of the 22 genres defined by the International Movie Database (IMDB)[15]. To obtain the class labels, we check for string containment of the IMDB genre names in the literal values of the *s:genre* properties. If a genre name is a substring of the aforementioned property the genre is assigned as class label to the respective instance. Note that it is possible for one instance to exhibit multiple labels since multiple genre names may be substrings of a single *s:genre* property and, in addition, single instances may have multiple *s:genre* properties. Intuitively, this process leads to reasonable class labels for the majority of instances, such that a sufficiently large amount of correctly labeled training data can be obtained. Yet, we also anticipate a certain amount of noise. The cleansed and labeled datasets are made publicly available[16].

Table 2 provides an overview of the size of the extracted datasets as well as the amount of included plds. The event datasets are denoted by *Events* and contain the following classes: *PublicationEvent, MusicEvent, ScreeningEvent, ComedyEvent, TheaterEvent, EducationEvent, VisualArtsEvent, Other*. For movie genres, the genre-specific datasets are denoted by the first three letters of the respective genre as follows *{Drama, Comedy, Action, Thriller, Romance, Documentary, Adventure}* = {*Dra, Com, Act, Thr, Rom, Doc, Adv*}. *Movies* refers to average values for all genres. The sampling method is denoted by the subscript, where *s* represents *stratified random sampling* and *p pld-aware sampling*.

Table 2: Overview of the dataset size and contained plds. Movie genres are abbreviated by their first three letters. An own dataset for each genre is extracted since each genre is treated as a binary classification problem.

Dataset	Size	Distinct plds	Avg. Instances/pld
$Events_s$	67,744	1,482	45.71
$Events_p$	67,744	2,064	32.82
Dra_s	239,030	360	663.97
Dra_p	239,030	476	502.16
Com_s	239,030	342	698.92
Com_p	239,030	476	502.16
Act_s	239,030	361	662.13
Act_p	239,030	476	502.16
Thr_s	239,030	342	698.92
Thr_p	239,030	476	502.16
Rom_s	239,030	347	688.85
Rom_p	239,030	476	502.16
Doc_s	239,030	337	709.29
Doc_p	239,030	476	502.16
Adv_s	239,030	340	703.03
Adv_p	239,030	476	502.16
$Movies_s$	239,030	347	689.30
$Movies_p$	239,030	476	502.16

4.2 Metrics

To evaluate the performance of the different classifiers, we compute the following metrics:

Precision: The fraction of the correctly classified instances among the instances assigned to one class.

Recall: The fraction of the correctly assigned instances among all instances of the class.

F1 score: The harmonic mean of recall and precision. This work considers the F1 score to be the most relevant metric since it reflects both recall and precision.

4.3 Baselines

We compare our approach to the following baselines:

RANDOM: This baseline chooses a class at random.

SD-TYPE: This baseline leverages conditional probabilities to infer the subject types using the *SD-Type* approach [19]. The probabilities are based on the incoming and outgoing statements of a particular node. Since *SD-Type* was not originally designed to be applied to Web markup, we adapted it by only considering outgoing statements. This is motivated by the fact that a complete set of incoming statements can not be obtained for Web markup, where links might (but are unlikely to) originate from any Web page.

KG-B: This baseline employs a knowledge graph to obtain class labels. The *s:name* of a subject is used as input for *DBpedia Spotlight* [5] to obtain candidate entities from *DBpedia* (*dbp*). If the markup is annotated in one of the 12 languages supported by Spotlight[17], the corresponding Spotlight model is used. For all other cases we employ the English Spotlight model. Labels obtained from DBpedia may be different from labels found in Web markup (e.g. the genre of the movie "Forrest Gump" is stated to be *Drama* and *Comedy* in DBpedia, but marked as *Drama* and *Romance* on *imdb.com*). In order to avoid noisy and costly matching process, we address this issue by considering all candidates with a confidence of at least 0.5 as true positives as long as the matching entity shows the correct type (*dbp:Event or s:Event* respectively *dbp:Movie* or *s:Movie*), independent of whether or not the entity actually shows the expected categorical property. If no candidate with a suitable type is found, the instance is assigned to the "Other"-class. Note that this simplification significantly boosts the performance of this otherwise naive baseline, yet serves the purpose of illustrating the lack of sufficient coverage (Section 5).

5 RESULTS

This section presents the results on the classification performance, the influence of the sampling methods and the individual features.

5.1 Classification Performance

Table 3 summarises the overall results of the baselines (Random, SD-Type, KG-B) as well as our proposed classification models (Naïve Bayes, Decision Tree, Random Forest, SVM). For both tasks, we report the macro averages of the results with respect to precision, recall and F1 scores for both *stratified random sampling* and *pld-aware sampling*. We observe that, for *Movies*, Random Forest, closely followed by Decision Tree, performs best across all evaluation metrics, except for precision/$Movies_s$, where it is slightly outperformed by KG-B. This is caused by the underlying assumption of the KG-B baseline that any entity match is considered as successful information inference, which unfairly boosts the baseline performance, in particular for popular entities. For *Events*, Random Forest shows the highest Recall and F1, closely followed by Decision Tree, whereas highest precision is achieved by Naïve Bayes in this case. The use of a single Decision Tree already results in relatively high F1 scores, e.g. 81.86% for $Movies_p$. Considering a Random Forest as an ensemble of Decision Trees, we conclude that additional trees only slightly improve the outcome (F1 of 83.14%). The SD-Type baseline achieves F1 scores of 56.99% for *Events*. This significant difference in performance between the baseline and our approach reflects the fundamental difference between knowledge graphs and data sourced from markup and the need to consider features beyond the structural connections of entity descriptions when dealing with markup data. For both *Events* and *Movies*, KG-B assigns the vast majority of the instances to the "Other"-class, resulting in high recall and low precision for the aforementioned class. Due to the design of the baseline, all classes different from "Other" exhibit 100% precision but very low recall, which ultimately results in low F1 scores after computing the macro average across classes.

For *Movies*, Table 3 reports the average scores of the individual genre-specific classifiers. It is worth to mention that the boundary of the classes (genres) might be fuzzy, e.g. it could be hard to differentiate a movie of genre "Thriller" from a movie of genre "Action". Since the classification of each genre is formulated as a binary classification problem, the random-baseline performance is close to 50% for all classes. The highest F1 score achieved by SD-Type is 67.62%, indicating that the subject properties used by this baseline might not be sufficient to classify movie genres precisely. Overall performance of the KG-B baseline is better in this task,

Table 3: Macro averages for precision, recall, and F1 score [%] over all datasets.

Classifier	$Events_s$			$Events_p$			$Movies_s$			$Movies_p$		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Random	12.72	12.71	12.71	12.81	12.82	12.81	50.00	50.00	50.00	49.87	49.87	49.86
SD-Type	58.35	49.56	40.98	58.71	62.83	56.99	61.67	58.92	56.36	68.34	67.77	67.62
KG-B	39.06	12.59	02.96	39.06	12.59	02.96	**76.52**	55.70	44.82	76.94	57.16	47.42
Naïve Bayes	**86.04**	44.24	40.04	**84.06**	50.51	47.78	69.06	50.29	33.98	61.55	50.39	34.19
Decision Tree	70.60	70.26	70.15	78.70	77.78	77.25	72.95	72.88	72.85	82.01	81.89	81.86
Random Forest	73.34	**72.46**	**71.67**	80.75	**79.71**	**79.59**	74.62	**74.49**	**74.46**	**83.27**	**83.16**	**83.14**
SVM	75.51	70.10	67.64	81.45	78.67	77.34	72.84	72.42	72.27	81.75	81.37	81.27

Table 4: Hyperparameters considered for optimisation.

Classifier	Parameter	Range
Decision Tree	Criterion	Gini Impurity, Information Gain
	Min.Impurity Decrease	[0,1]
Random Forest	Criterion	Gini Impurity, Information Gain
	Min.Impurity Decrease	[0,1]
	No. Estimators	[5,20]
SVM	Penalty	[0,5]
	Stopping Tolerance	$[0,10^{-3}]$

driven by higher recall for instances of type movie, which are better represented in knowledge bases. Similar to our observations in the event classification task, Random Forest performs best, closely followed by Decision Tree. The F1 score of 83.14% for Random Forest significantly outperforms the baselines (paired t-test with $p < 0.01$) when comparing Random Forest against the baselines in all configurations. Overall Random Forest classification using the features proposed in this paper clearly outperforms the baselines in both tasks.

5.1.1 Classification Hyperparameter. For each classifier used with an exception of the Naïve Bayes classifier, we determine the parameters that maximise the F1 score by employing the random search algorithm proposed by Bergstra and Bengio [2]. The Naïve Bayes classifier does not exhibit parameters that could be optimised. Table 4 gives an overview of the parameters that were considered during the optimisation, whereas Table 5 summarises the hyper-parameters that were determined using random search. All previously shown performance results were obtained using the specified hyper-parameters.

5.2 Influence of Sampling Methods

In this section, we discuss the influence of the different sampling methods. Since the Random Forest classifier achieves the best results, we investigate the effects of sampling methods on our Random Forest configuration.

Table 5: Summary of classifier hyperparameters determined with random search for the following parameters: Crit: Criterion, Imp: Min. Impurity Decrease, No: No. Estimators, Pen: Penalty, Tol: Stopping Tolerance.

Dataset	Decision Tree		Random Forest			SVM	
	Crit	Imp	Crit	Imp	No	Pen	Tol
$Events_s$	ent.	0.192	ent.	0.892	13	3.53	0.0043
$Events_p$	gini	0.527	ent.	0.892	13	1.88	0.0098
Dra_s	gini	0.360	ent.	0.938	16	0.66	0.0037
Dra_p	gini	0.360	gini	0.414	18	0.66	0.0037
Com_s	gini	0.360	gini	0.414	18	0.66	0.0037
Com_p	ent.	0.608	gini	0.160	20	0.66	0.0037
Act_s	gini	0.360	gini	0.160	20	0.66	0.0037
Act_p	ent.	0.558	gini	0.160	20	0.66	0.0037
Thr_s	gini	0.360	ent.	0.482	16	0.66	0.0037
Thr_p	ent.	0.608	ent.	0.482	13	0.66	0.0037
Rom_s	ent.	0.608	ent.	0.482	13	0.66	0.0037
Rom_p	ent.	0.287	gini	0.160	20	0.66	0.0037
Doc_s	ent.	0.192	gini	0.160	20	0.66	0.0037
Doc_p	ent.	0.099	gini	0.068	16	0.66	0.0037
Adv_s	ent.	0.287	gini	0.160	20	0.66	0.0037
Adv_p	ent.	0.287	gini	0.068	16	0.66	0.0037

Figure 4 shows the F1 scores with respect to the sampling method for *Events* and the individual *Movies* genre datasets. The use of *pld-aware sampling* yields up to 17% percentage points better results than the use of *stratified random sampling*.

We observe that the use of a more diverse training set (i.e. a dataset including more data from long-tail domains e.g. obtained through the *pld-aware sampling*) has a significant and beneficial effect on the classification outcome (paired t-test with $p < 0.03$).

5.3 Influence of Features

In this section, we discuss the influence of the proposed features. We focus on the best performing classifier (Random Forest) while investigating the effects of varying the feature set.

Table 6 presents the F1 scores obtained through Random Forest on the *Events* dataset with respect to different feature combinations. Our results indicate that the influence of features varies strongly

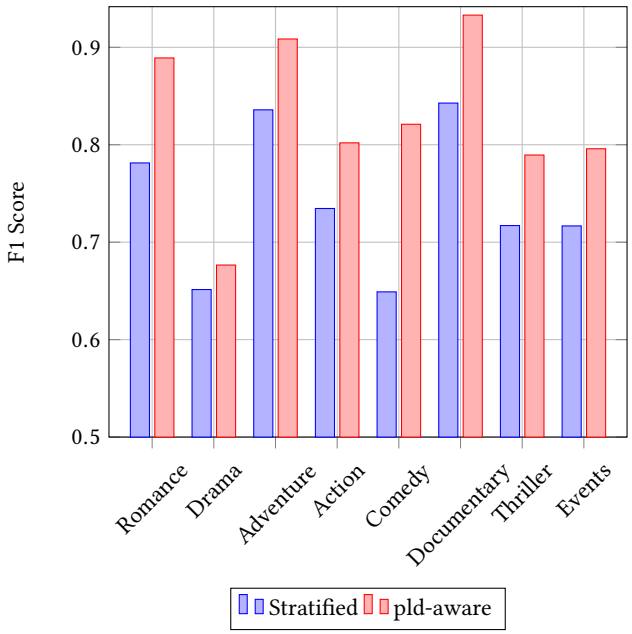

Figure 4: F1 scores macro averages [%] for the RANDOM FOREST classifier with respect to dataset and sampling method.

Table 6: Random Forest F1 scores macro averages [%] for different feature combinations (*Events* datasets).

Features	$Events_s$	$Events_p$
tld/pld	65.30	76.29
page-vocab	62.57	79.8
node-vocab	60.66	68.09
tld/pld,page-vocab	71.01	80.03
tld/pld,node-vocab	65.38	77.65
page-vocab,node-vocab	71.70	80.27
tld/pld,page-vocab,node-vocab	71.67	79.59

Table 7: Random Forest F1 scores macro averages [%] for different feature combinations (*Movies* datasets).

Features	$Movies_s$	$Movies_p$
tld/pld	66.32	80.56
page-vocab	72.27	82.14
node-vocab	73.96	82.18
tld/pld,page-vocab	72.59	82.68
tld/pld,node-vocab	74.28	82.94
page-vocab,node-vocab	74.33	82.59
tld/pld,page-vocab,node-vocab	74.46	83.14

dependent on the respective types and classes. This seems intuitive, given that some classes might be more specifically characterised by certain features, such as a set of plds. The *tld/pld* features alone result in a reasonable performance for *Events* but not for *Movies*. This indicates that the source of the markup node is stronger correlated with its actual type or category for *Events* than for *Movies*. This seems intuitive, given that event-centred Websites tend to be more focused on certain event types than movie-centred Websites are focused on particular genres. However, these observations are likely to vary strongly dependent on the actual classification task. In contrast, the *node-vocab* alone is not sufficient to determine the event subtype with high F1 score. This observation corresponds to the insufficient performance of the *SD-Type* baseline.

The combination of *tld/pld* and *node-vocab* results only in a slight improvement of the results for $Events_p$. A dependence between the two features seems intuitive as pages extracted from the same pld are likely to be maintained by the same organisation and thus typically use the same set of *schema.org* terms. For instance, an event database is likely to assign the same set of properties to each event resulting in a characteristic *node-vocabulary* for the events of a single pld. Since the *page-vocab* considers the terms that occur on the whole page, the number of considered terms is higher, which results in better chances to find usage of the same terms on other Web pages. This is reflected by the fact that both combinations of *tld/pld, page-vocab* and *page-vocab, node-vocab* lead to an improvement while the performance of *tld/pld, node-vocab* is roughly the same as *tld/pld* only. The combination of all three features yields in a slight decrease of the F1 score compared to *tld/pld, page-vocab* only, indicating once more that the information contributed by *node-vocab* is already provided by *tld/pld*.

Table 7 shows average F1 scores using the RANDOM FOREST classifier on the *Movies* datasets. In contrast to the *Events* datasets we can achieve relatively good performance by employing only the *node-vocab* feature. Another difference is that we can observe a slightly larger margin between the exclusive use of *node-vocab* and *tld/pld*. This indicates that markup of movies of certain genres tend to exhibit the same *schema.org* terms. Any combination of two or more features results in similar outcomes (with approximatly 1 percentage point difference). In both domains we can see a substantial difference in the performance with respect to the sampling methods for all feature combinations. *pld-aware sampling* consistently achieves higher F1 scores than *stratified random sampling*, leading to the conclusion that individual features and feature combinations benefit from *pld-aware sampling*.

5.4 Discussion

Our experiments illustrated that traditional knowledge graph completion approaches that are not specifically designed for Web markup data may not be directly applicable to this kind of data, mainly due to the sparsity of individuals and the lack of connectivity in Web markup. Moreover, we observed that it is not sufficient to consider only node-specific features such as *node-vocab* to infer missing categorical information in Web markup. In contrast, contextual features such as *tld/pld* and *page-vocab* provide important information to infer missing statements.

In particular, our experiments demonstrated that contextual features such as *tld/pld* and *page-vocab* are discriminative for both tasks under consideration. These features are effective because many Websites focus on a particular topic, e.g. theater or music events. We observed that the *page-vocab* feature is especially useful

in both tasks, as it describes the context of the particular node in a more specific way. Whereas the use of the *tld/pld* feature can naturally only be applied to instances from known plds, i.e. plds that are contained in the training data, performance drops are expected when classifying data from unknown plds. However, our results indicate that features representative for certain kinds of plds, such as *page-vocab*, can serve as potent substitute able to efficiently classify markup from unknown sources.

Limitations arise from the focus on two particular tasks only. We anticipate variation in performance of particular features when applying this approach to other kinds of categorical information. Similarly, considering that our ground truth has been constructed by relying on markup nodes where the sought-after information was present already on the Web, one might argue that this constraint has led to a bias towards markup nodes of generally higher quality. Additional experiments on an unconstrained and randomly selected ground truth will investigate this assumption further as part of future work.

6 RELATED WORK

In this section we discuss related work in the areas of knowledge graph completion and schema inference for traditional knowledge graphs along with works focused directly on Web markup.

Existing approaches to knowledge graph completion and dataset profiling including its applications to schema inference have been summarised in recent survey articles [1, 18]. These approaches include in particular entity type inference, relation prediction and relation validation. In the context of KG completion, entity type inference is most commonly addressed as a multi-class prediction problem. [19] makes use of properties and conditional probabilities to infer entity types, building the baseline for our approach. Schemex is an approach to extract and index schema information from Linked Open Data (LOD) [12]. In YAGO+F instance-based matching enables to enrich Freebase entities with YAGO concepts [6]. [10] made use of Schemex to analyse schema information of LOD and found that properties provide information about subject types. In our work, we use properties as features for inferring missing categorical information in general. [9] predicts relations between two nodes by leveraging random walk inference methods using sub-graphs to improve the path ranking algorithm (PRA), initially proposed in [13]. [25] also builds on PRA and extends it to a multi-task learning approach.

All of the works discussed above have been applied to traditional KGs such as DBpedia, NELL and YAGO. In contrast, in this work we aim at inferring information on the Web markup data. Web markup is distinguished from the aforementioned knowledge graphs by specific characteristics, i.e. annotations are often very sparse or noisy, vocabularies are not used correctly in many cases and the overall RDF graph is connected very loosely [7, 16]. For these reasons, existing KG completion methods are not likely to perform well on Web markup. For instance, KG completion approaches based on graph topology (e.g. relation prediction discussed above) rely on the presence of relations, which are not widely available in markup.

Various approaches employ embeddings for KG completion in traditional knowledge graphs. [26] conducted a survey on KG embeddings for applications such as link prediction, entity classification and triple classification. [27] makes use of embeddings and rules. [14] propose the *TransR* model that builds separate entity and relation embeddings to compute the plausibility of missing triples. [22] predicts relations between entities by employing neural tensor networks. Embeddings techniques have not yet been applied to Web markup yet lend themselves as direction for future research.

Several recent studies focused on analysing the characteristics, evolution and coverage of markup [7, 20, 24] and on addressing specific tasks in the context of Web markup. Meusel et al. proposed heuristics that can be employed to fix common errors in Web markup [15, 16]. In this work, we applied the heuristics proposed in [15] for pre-processing and data cleansing. [29, 31] provide pipelines for data fusion and entity summarisation on Web markup, involving heuristics, clustering and supervised approaches for entity matching and classification of markup statements. [30] builds on these works by utilising fused markup data to augment existing knowledge bases, showing the complementarity of markup data and its potential to significantly complement information from traditional reference KGs.

While these works demonstrate the use of markup data, they suffer from the sparsity of individual nodes. The inference approach proposed in our work can augment markup nodes and is likely to boost the performance on both fusion as well as KG augmentation tasks. In particular, considering the impact of the use of controlled vocabularies on data reuse [8], we anticipate that inference of crucial categorical information can facilitate reuse of markup data.

7 CONCLUSION & FUTURE WORK

In this work, we addressed the problem of interpreting noisy and sparse Web markup by proposing an approach for automatically inferring categorical information for particular entities, thereby augmenting sparse markup nodes with information, which often is essential when interpreting markup and the corresponding Web pages. We leveraged the large amount of publicly available data as training data for a supervised machine learning approach. We employed Web markup specific features such as *tld/pld*, *node vocabulary* and *page vocabulary* and conducted an extensive evaluation of different classification algorithms, sampling methods and feature sets. Our proposed configuration outperforms existing baselines significantly, with Random Forest providing the most consistent performance across classes and datasets.

By applying our approach to the problem of inferring event types and movie genres, we demonstrated that supervised inference can uncover entity-centric categorical information, which is essential when interpreting markup or Websites in general. Potential applications include knowledge base augmentation from Web markup [30], Website classification or Web search in general. Considering the still limited experiments and the limitations of our dataset (Section 5.4), future work will include the conduction of additional experiments, involving more diverse datasets and tasks. With respect to the latter, current experiments are being conducted on discretised information rather than properties, which are apriori categorical. In addition, we aim at investigating the impact of our approach when being included into data fusion and KG augmentation pipelines, such as [29, 30].

ACKNOWLEDGMENTS

This work was partially funded by the European Commission ("AFEL" project, grant ID 687916) and the BMBF ("Data4UrbanMobility" project, grant ID 02K15A040).

REFERENCES

[1] Mohamed Ben Ellefi, Zohra Bellahsene, Breslin John, Elena Demidova, Stefan Dietze, Julian Szymanski, and Konstantin Todorov. 2017. RDF Dataset Profiling - a Survey of Features, Methods, Vocabularies and Applications. *Semantic Web Journal* (2017). to appear.

[2] James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-parameter Optimization. *J. Mach. Learn. Res.* 13 (Feb. 2012), 281–305.

[3] Christian Bizer, Kai Eckert, Robert Meusel, Hannes Mühleisen, Michael Schuhmacher, and Johanna Völker. 2013. *Deployment of RDFa, Microdata, and Microformats on the Web – A Quantitative Analysis.* Springer Berlin Heidelberg, 17–32.

[4] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD '08)*. ACM, 1247–1250.

[5] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. 2013. Improving Efficiency and Accuracy in Multilingual Entity Extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-SEMANTICS '13)*. ACM, 121–124.

[6] Elena Demidova, Iryna Oelze, and Wolfgang Nejdl. 2013. Aligning Freebase with the YAGO Ontology. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management (CIKM '13)*. ACM, 579–588.

[7] Stefan Dietze, Davide Taibi, Ran Yu, Phil Barker, and Mathieu d'Aquin. 2017. 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)*. International World Wide Web Conferences Steering Committee, 283–292.

[8] Kemele M. Endris, José M. Giménez-García, Harsh Thakkar, Elena Demidova, Antoine Zimmermann, Christoph Lange, and Elena Simperl. 2017. Dataset Reuse: An Analysis of References in Community Discussions, Publications and Data. In *Proceedings of the Ninth International Conference on Knowledge Capture (K-CAP 2017)*.

[9] Matt Gardner and Tom M. Mitchell. 2015. Efficient and Expressive Knowledge Base Completion Using Subgraph Feature Extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*. 1488–1498.

[10] Thomas Gottron, Malte Knauf, Stefan Scheglmann, and Ansgar Scherp. 2013. A Systematic Investigation of Explicit and Implicit Schema Information on the Linked Open Data Cloud. In *Proceedings of the ESWC 2013*. Springer Berlin Heidelberg, 228–242.

[11] R. V. Guha, Dan Brickley, and Steve Macbeth. 2016. Schema.Org: Evolution of Structured Data on the Web. *Commun. ACM* 59, 2 (Jan. 2016), 44–51.

[12] Mathias Konrath, Thomas Gottron, Steffen Staab, and Ansgar Scherp. 2012. SchemEX - Efficient Construction of a Data Catalogue by Stream-based Indexing of Linked Data. *Web Semant.* 16 (Nov. 2012), 52–58.

[13] Ni Lao and William W. Cohen. 2010. Relational Retrieval Using a Combination of Path-constrained Random Walks. *Mach. Learn.* 81, 1 (Oct. 2010), 53–67.

[14] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 2181–2187.

[15] Robert Meusel and Heiko Paulheim. 2015. Heuristics for Fixing Common Errors in Deployed Schema.Org Microdata. In *Proceedings of the 12th European Semantic Web Conference on The Semantic Web*. Springer-Verlag New York, Inc., 152–168.

[16] Robert Meusel, Petar Petrovski, and Christian Bizer. 2014. The WebDataCommons Microdata, RDFa and Microformat Dataset Series. In *Proceedings of the 13th International Semantic Web Conference - Part I (ISWC '14)*. Springer-Verlag New York, Inc., 277–292.

[17] Robert Meusel, Dominique Ritze, and Heiko Paulheim. 2016. Towards More Accurate Statistical Profiling of Deployed schema.org Microdata. *ACM Journal of Data and Information Quality* 8, 1 (2016).

[18] Heiko Paulheim. 2016. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web* Preprint (2016), 1–20.

[19] Heiko Paulheim and Christian Bizer. 2013. Type Inference on Noisy RDF Data. In *Proceedings of the 12th International Semantic Web Conference - Part I (ISWC '13)*. Springer-Verlag New York, Inc., 510–525.

[20] Pracheta Sahoo, Ujwal Gadiraju, Ran Yu, Sriparna Saha, and Stefan Dietze. 2016. Analysing Structured Scholarly Data Embedded in Web Pages. (April 2016).

[21] Amit Singhal. 2012. Introducing the Knowledge Graph: things, not strings. Official Google Blog. (May 2012). https://googleblog.blogspot.de/2012/05/introducing-knowledge-graph-things-not.html accessed on 01/20/2018.

[22] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., 926–934.

[23] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web (WWW '07)*. ACM, 697–706.

[24] Davide Taibi and Stefan Dietze. 2016. Towards embedded markup of learning resources on the Web: a quantitative Analysis of LRMI Terms Usage. In *Proceedings of the WWW Companion 2016.*

[25] Quan Wang, Jing Liu, Yuanfei Luo, Bin Wang, and Chin-Yew Lin. 2016. Knowledge Base Completion via Coupled Path Ranking. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*. ACL - Association for Computational Linguistics, 1308–1318.

[26] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2724–2743.

[27] Quan Wang, Bin Wang, and Li Guo. 2015. Knowledge Base Completion Using eddings and Rules. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 1859–1865.

[28] Ran Yu, Besnik Fetahu, Ujwal Gadiraju, and Stefan Dietze. 2016. A Survey on Challenges in Web Markup Data for Entity Retrieval. In *Proceedings of the ISWC 2016 Posters & Demonstrations Track*.

[29] Ran Yu, Ujwal Gadiraju, Besnik Fetahu, and Stefan Dietze. 2017. FuseM: Query-Centric Data Fusion on Structured Web Markup. In *Proceedings of the IEEE 33rd International Conference on Data Engineering (ICDE), 2017*. IEEE Computer Society, 179–182.

[30] Ran Yu, Ujwal Gadiraju, Besnik Fetahu, Oliver Lehmberg, Dominique Ritze, and Stefan Dietze. 2017. KnowMore - Knowledge Base Augmentation with Structured Web Markup. *Semantic Web Journal, IOS Press* (2017).

[31] Ran Yu, Ujwal Gadiraju, Xiaofei Zhu, Besnik Fetahu, and Stefan Dietze. 2016. Towards Entity Summarisation on Structured Web Markup. *The Semantic Web: ESWC 2016 Satellite Events,* (June 2016).

NOTES

[1] RDFa W3C recommendation: http://www.w3.org/TR/xhtml-rdfa-primer/

[2] https://www.w3.org/TR/microdata/

[3] http://microformats.org

[4] http://commoncrawl.org/

[5] http://webdatacommons.org/structureddata/2016-10/stats/stats.html

[6] http://webdatacommons.org/

[7] http://commoncrawl.org/

[8] Detailed numbers can be found at http://webdatacommons.org/structureddata/index.html

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

[10] http://schema.org/Event

[11] http://schema.org/CreativeWork

[12] http://schema.org/Organization

[13] http://schema.org/Person

[14] For a brief description of 1-hot-encoding see: http://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-categorical-features

[15] http://www.imdb.com/genre/

[16] The datasets can be found at http://markup.l3s.de.

[17] http://www.dbpedia-spotlight.org/faq