Type Prediction Combining Linked Open Data and Social Media

Yaroslav Nechaev[1,2], Francesco Corcoglioniti[1], Claudio Giuliano[1]

(1) Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
(2) University of Trento, Via Sommarive 14, 38123 Trento, Italy
{nechaev,corcoglio,giuliano}@fbk.eu

ABSTRACT

Linked Open Data (LOD) and social media often contain the representations of the same real-world entities, such as persons and organizations. These representations are increasingly interlinked, making it possible to combine and leverage both LOD and social media data in prediction problems, complementing their relative strengths: while LOD knowledge is highly structured but also scarce and obsolete for some entities, social media data provide real-time updates and increased coverage, albeit being mostly unstructured.

In this paper, we investigate the feasibility of using social media data to perform type prediction for entities in a LOD knowledge graph. We discuss how to gather training data for such a task, and how to build an efficient domain-independent vector representation of entities based on social media data. Our experiments on several type prediction tasks using DBpedia and Twitter data show the effectiveness of this representation, both alone and combined with knowledge graph-based features, suggesting its potential for ontology population.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; *Supervised learning by classification*; *Knowledge representation and reasoning*;

KEYWORDS

Type Prediction; Ontology Population; Social Media; Linked Open Data; Machine Learning; Semantic Web

ACM Reference Format:
Yaroslav Nechaev[1,2], Francesco Corcoglioniti[1], Claudio Giuliano[1] . 2018. Type Prediction Combining Linked Open Data and Social Media. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271781

1 INTRODUCTION

The Linked Open Data (LOD) cloud over the years has become a prominent source of background knowledge for a large selection of tasks. DBpedia and Wikidata in particular cover a wide range of entities including significant amounts of people and organizations. The data in such knowledge graphs is highly structured and readily available. However, it is mainly populated and updated using the crowdsourced content-editing efforts of the Wikimedia community, making the contained knowledge often incomplete, noisy and stale. While a significant amount of research has been done to address those issues via ontology population from external sources, reasoning and knowledge graph completion, they are far from being solved. Social media, on the other hand, provide up-to-date information on an overwhelming amount of people, organizations, and brands: Facebook alone has more than 2.1B monthly active users. This data is hard to extract due to API limitations and privacy considerations, and difficult to process due to its semistructured nature. Despite difficulties, social media is steadily becoming a primary source of real-time knowledge: where a community-curated knowledge graph can take hours to months to update, the information in social media often appear immediately.

There is a considerable overlap of coverage between LOD and social media. Living people and existing organizations from LOD are likely to be found in the social media as well. Such entities have become increasingly interlinked. More than 50K links between DBpedia entities and corresponding Twitter profiles can be found in DBpedia, with more available in Wikidata. Current community efforts, such as the SocialLink [12] project, aim to expand the coverage of those links significantly — the version that was available at the time of writing (v2.0) provides additional 300K alignments from DBpedia to Twitter. The increased interlinking can enable knowledge transfer between the highly structured LOD cloud and the vibrant social media world, improving and simplifying the processing pipelines in both.

In particular, the ontology population task can benefit from such interlinking. In this task, the aim is to fill missing connections in the knowledge graph and to populate it with new data from an external resource, to improve the coverage on a particular set of attributes. This is particularly important, for example, for DBpedia, where such a fundamental attribute of a person as age has only 52,8% coverage. Even if a particular attribute has a good enough coverage, the problem of referencing appears. In Wikidata, for instance, each claim can be corroborated by references to external resources confirming the data. Given that many social media profiles of people and organizations are marked as "verified" and considered official, they can serve as references for existing facts.

In this paper, we consider and investigate the feasibility of using social media data to predict entity attributes and perform ontology population. To the best of our knowledge, no one has tried to tackle these tasks exploiting social media data before. To this end, we present a general purpose approach that, given a stream of social media data (Twitter) and some links from entities in a LOD

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271781

knowledge graph (DBpedia) to their social media profiles, is able to predict and fill entity type attributes. This is achieved by treating the knowledge graph as a source of ⟨entity, type⟩ training data, and the social media as a source of features for the considered type prediction task, which we use to build rich, domain-general entity representations by integrating different data in the social media entity profile (e.g., profile attributes, social graph, textual data).

An approach like the presented one poses different challenges, which we tackle in this paper. Firstly, the social graph is hard to acquire at scale. The Twitter API, like any social media API, imposes limits on the number of queries per fixed period, which makes it impractical to base our approach on the real social graph. Instead, we acquire an approximation of the social graph using retweet and mention relations extracted from the stream of tweets. Secondly, social media data is sparse and noisy: for some users, we might have a clean and complete profile and a significant amount of textual content, while for some, we might only have mentions of the user without any authored content. To overcome this issue, we have designed a feature space that efficiently combines different kinds of user-related data in a joint representation, aimed to be effective in different type prediction tasks.

To evaluate our approach, we compare the proposed feature space derived from Twitter data with a state-of-the-art entity representation model for DBpedia by Cochez et al [3], consisting of RDF entity embeddings. Our experiments show that our social media entity representation gives prediction performances competitive with the ones obtained using RDF embeddings, outperforming it in most of the considered type prediction tasks. Additionally, we show that by combining social media and RDF embeddings, performances can be further improved. Finally, in the same setting, we compare the usage of social media data to Wikipedia, a traditional source of knowledge for the DBpedia population task. We demonstrate that the social media data is able to complement the Wikipedia-based features effectively achieving up to 92% F_1.

While in this paper we experiment only with DBpedia and Twitter data, the presented approach is in principle social media and knowledge graph-agnostic. From the social media side, the same data as we use here (social graph, posts, minimal user profile) can be found in virtually any other social network. On the knowledge graph side, the distributional semantics-based features we use for DBpedia can be produced from an arbitrary knowledge graph [5].

The rest of the paper is organized as follows. In Section 2 we introduce the considered type prediction task and its application to ontology population in detail. Section 3 provides an overview of our approach, with Section 4 detailing the acquisition of ground truth data from LOD and Section 5 describing the use of social media features to represent entities. In Section 6 we evaluate our approach on several type prediction problems. Related work is presented in Section 7, while Section 8 concludes.

2 PROBLEM DEFINITION

In this work, we investigate the combined use of LOD and social media data for predicting entity attributes, and more specifically entity *types*. This task is often referred as *type prediction*, and is characterized by the fact that the types being predicted form a closed set whose size is typically small, whereas other attribute prediction tasks involve a large and possibly open set of predicted values, such as all the entities in a knowledge graph for *link prediction* tasks, or a continuous range of values for regression tasks.[1]

In our context, the types being predicted come from a LOD knowledge graph. Typically, these types are ontological classes *explicitly* defined in the graph, but they may be also *implicitly* derived from other kinds of information, such as age category types derived from a birth date property (e.g., young adult, see Section 4). In other words, we refer here to a generic notion of type that is compatible with different attribute prediction problems.

The type prediction task can be seen as *classification* task whose labels are types, and whose flavor depends on the relations existing among the predicted types:

- a *binary classification* task arises whenever the considered types are independent, and thus a yes/no prediction may be independently produced for each type;
- a *multi-class classification* task arises when the considered types are mutually disjoint, so that a given entity must be assigned exactly to one of the considered types;
- a *multi-label classification* task occurs when types are not all disjoint (so an entity may have multiple types, i.e., labels) and there might exist dependencies among types (e.g., selected disjoint or sub-type constraints) that make inappropriate the use of independent binary classifiers whose output may be inconsistent. When these dependencies come in the form of a type hierarchy (e.g., via `rdfs:subClassOf` relation), the task is also referred as *hierarchical multi-label classification* [8, 18].

In this work, we consider the case of mutually disjoint types and thus multi-class classification. This variant is general enough to be useful in practice, and permits us to focus on validating the feasibility of our approach without considering the additional complexity of (hierarchical) multi-label classification, which we leave as future work. In particular, by requiring an entity to belong to exactly one of a closed set of disjoint types, we rule out the problem of dealing with incomplete type knowledge (assuming an open-world stance), as the fact that an entity e is not associated to type t_i in the knowledge graph can be considered as a negative example for t_i (instead of a case of missing information) if we know that e has another type t_j disjoint with t_i.

For predicting our types, we consider the use of social media data, either alone or combined with LOD data, to build the feature vector representations of predicted entities. We consider Twitter, but our approach and experiments can be in principle reproduced for other social networks. Social media data is obtained by aligning the LOD entity to its corresponding social media profile, from which different kinds of data can be extracted and mapped to features. A sizable amount of ⟨DBpedia entity, Twitter profile⟩ links are in DBpedia, and many more can be generated with adequate precision as recently shown [10]. As social media profiles are usually associated to living persons and organizations, these are the main kinds of entities for which our approach is applicable, and the only ones here considered for simplicity.

The main application we foresee for the type prediction task here investigated is *ontology population*, where a supervised type predictor is trained based on entities with known type information in

[1] Using social media data may be feasible also in these tasks but it is out-of-scope here.

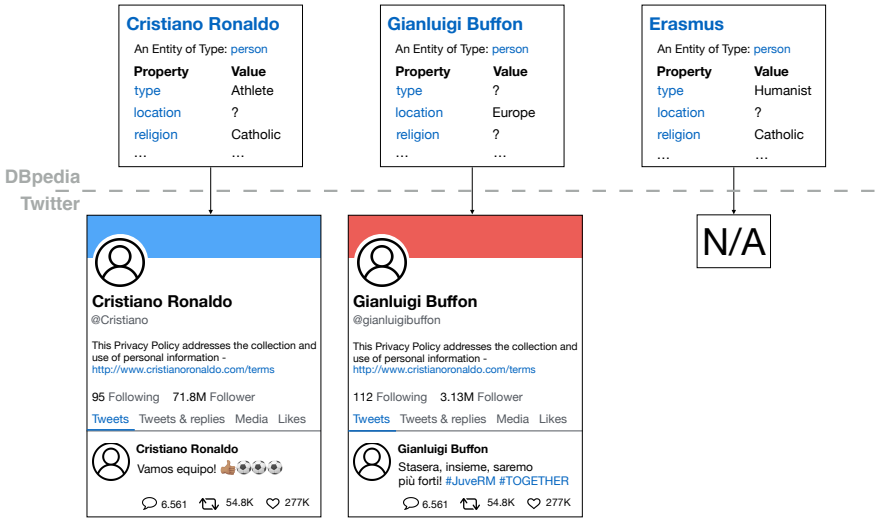

Figure 1: Using DBpedia-Twitter links for type prediction and ontology population.

LOD and then applied to classify and populate LOD entities without that information. This scenario is exemplified in Figure 1. It shows several DBpedia entities (top row) linked to their Twitter profiles (bottom row). Each of these links allows building a training example where the features are extracted from Twitter and the type label is derived, explicitly or implicitly, from one of the entity's properties. In the example, the entity `dbpedia:Cristiano_Ronaldo` provides labeled examples to train classifiers for types `dbo:Athlete` (from property `rdf:type`) and `dbo:Catholic` (property `dbo:religion`). Then, these classifiers can be used to predict the `rdf:type` and `dbo:religion` types for the entity `dbpedia:Gianluigi_Buffon`. No examples can be extracted from the entity `dbpedia:Erasmus` as there is no link to Twitter.

The same supervised type predictor can be used for the *user profiling* task. Here, instead of predicting a type for an existing entity in the knowledge graph, an arbitrary user can be placed into the knowledge graph connected to the known entities from this user's follow list. Then, the predicted type (e.g., location, age) can be assigned to this user. This direction has been partially investigated in the relevant literature [17] and is out of the scope of this paper.

3 APPROACH OVERVIEW

The two main steps for building a supervised solution for type prediction are (i) the acquisition of the necessary *ground truth* ⟨entity, type⟩ pairs, and (ii) the construction of an effective feature vector *entity representation*. These two steps are detailed respectively in Sections 4 and 5. Together, they permit to build a training set out of which a supervised classifier can be trained and/or evaluated, as shown graphically in Figure 2.

In the figure, the presence of the social link between entity `dbpedia:Cristiano_Ronaldo` and his `@Cristiano` Twitter account allows us to build an entity representation based on the information in Twitter, with different kinds of Twitter data (e.g., profile data with description and location, content posted, and mentions and retweets received, social graph) encoded in different sub-spaces of the entity feature vector. The connection to other resources, e.g., RDF2Vec, can be used to further enrich the feature space. Finally, the type *Athlete* from the LOD entity description is used to label the example. Iterated over multiple ⟨entity, profile⟩ pairs where the predicted type(s) are known, this approach permits to train a supervised type predictor for other, possibly unseen ⟨entity, profile⟩ pairs where those types are not known and can thus be populated.

In deriving the entity representation, we avoid any fine tuning for a specific prediction task, and strive for capturing all the available information in a *single*, *comprehensive*, *domain-independent* representation that can be effective in different type prediction tasks. So, for instance, the same vector representation shown in Figure 2 can be used to predict other types, e.g., related to the missing location property, in which case the training examples are extracted from all entities that have a valid location.

4 GROUND TRUTH ACQUISITION FROM LOD

In the considered multi-class prediction scenario, the acquisition of ground truth data consists in extracting one or more properly-defined *type prediction tasks* out of a given LOD knowledge graph (e.g., DBpedia) aligned to the considered social network (e.g., Twitter), each task consisting of a set of disjoint predicted types $T = \{t_1 \dots t_n\}$ and a dataset of ⟨entity, profile, type⟩ triples $\langle e, p_e, t_e \rangle$, $t_e \in T$ that is amenable to training a supervised classifier. Here, we describe the ground truth acquisition methodology that we manually applied to extract 8 type prediction tasks out of DBpedia, leaving the definition of an automated procedure for ground truth acquisition (possibly based on the methodology) as future work.

4.1 Methodology

We followed the three-step methodology described next.

Step 1: Type Information Identification This step deals with identifying the sources of type information available in the considered knowledge graph, leveraging TBox information and data querying to extract necessary ABox statistics. In a knowledge graph,

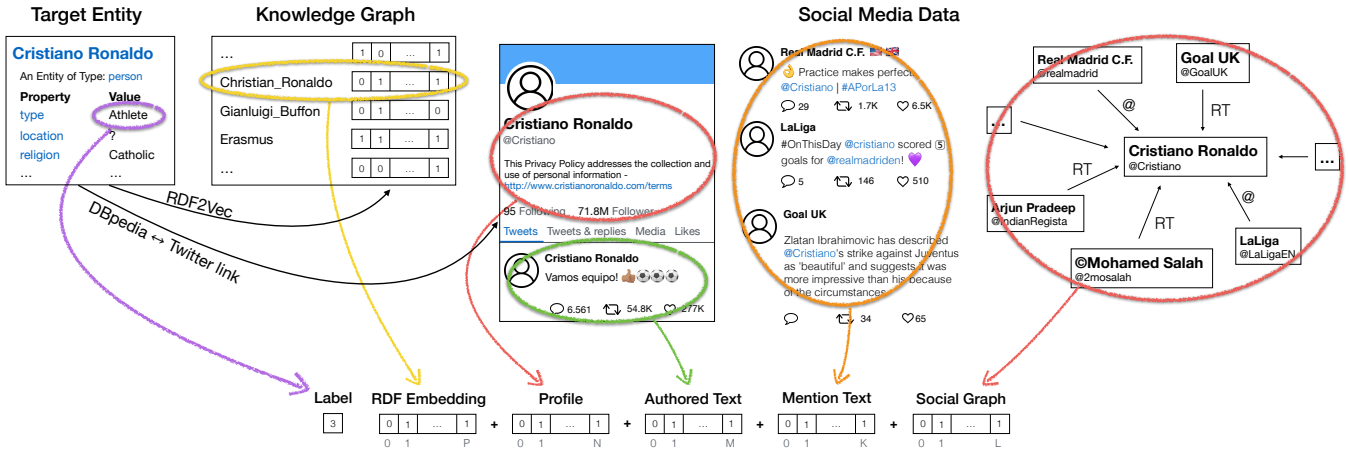

Figure 2: Example of training sample generation from DBpedia and Twitter data.

type-related information is available either explicitly or implicitly. Explicit type information comes in the form of `rdf:type` ABox triples assigning entities to well-known ontological classes. Implicit type information can be derived from other ABox triples involving the entity, in different ways. For instance:

- the object of the triple may already represent a type, as happens with values `dbpedia:Male` and `dbpedia:Female` of DBpedia property `dbo:gender`;
- the object of the triple may be part of a hierarchy whose upper nodes can be seen as types, as happens, e.g., with locations that can be spatially aggregated, or political parties that can be aggregated based on international affiliation;
- the object of the triple may be a numeric value or a date that can be discretized to different value ranges, an example being property `dbo:birthDate` that can be mapped to different age categories (e.g., child, young adult).

All the cases above can be treated as the materialization of implicit `rdf:type` triples via a rule-based strategy (e.g., via logical inference). These triples (if needed, see next steps) may be produced in a pre-processing step, so from now on we consider type information as fully available via `rdf:type` triples without loss of generality.

Step 2: Prediction Task Selection This step deals with the selection (here done manually based on the requirements below) of a set of prediction tasks based on the identified type information, each task defined by its set of types T. In our multi-class prediction context, the selection must satisfy two requirements:

- types $t_i \in T$ should be mutually disjoint; if not the case, overlapping types may be aggregated along a hierarchy or manually by adding super-classes, or alternatively finer-grained types representing their intersections may be introduced;
- for each type $t_i \in T$, there should be enough samples $\langle e, p_e, t_i \rangle$ for training a classifier; if it is not the case, types t_i may be aggregated along a hierarchy or manually to obtain coarser-grained types satisfying this requirement.

While these requirements are enough for our experiments, in a real ontology population scenario the selection should also satisfy two additional requirements:

- for each entity e in the populated knowledge graph and for each prediction task with types T, it must be possible to decide whether e may accept a type $t_i \in T$ (and thus the classifier may run on e). E.g., before predicting the age category of e, one should determine that e is a person. This check may be formulated based on coarser-grained entity types associated to e (e.g., a person/organization classification), either already known or predicted via an upstream classifier;
- for a prediction task to be useful, in addition to a large enough training set there must be a large enough amount of entities for which the predicted type information is missing and can thus be populated by the trained predictor.

Step 3: Dataset Extraction This step deals with extracting the $\langle e, p_e, t_e \rangle$ dataset for each selected prediction task. This involves implementing the pre-processing of Step 1 (if any) and the aggregations of Step 2 (if any). This step may involve also some *data normalization* — e.g., to convert organization revenues to the same currency, for predicting a revenue class — and *data clean-up* — e.g., to discard wrong data such as football teams being assigned a `dbo:gender`, or discard entities associated to multiple incompatible types, such as a person with multiple age categories.

4.2 DBpedia Type Prediction Tasks

We applied the aforementioned methodology to extract a non-exhaustive set of 8 type prediction tasks for our experiments (see Section 6) from the living persons and organizations in DBpedia 2016-04 that are aligned to Twitter in DBpedia (via property `dbo:wikiPageExternalLink`). Detailed information, code, datasets (including ground truth data) and additional experiments are available online.[2] We briefly describe these tasks below:

- *Category* — this task covers the specific category of person (e.g., artist, athlete) or organization (e.g., company, government agency) using a set of 17 types corresponding to DBpedia classes that can be reasonably considered as disjoint. Special types for "other person" and "other organization"

[2] http://sociallink.futuro.media/type-prediction

were added to aggregate persons and organizations belonging to types without enough training examples;
- *Location* — this task classifies entities (both persons and organizations) based on their location (property `dul:hasLocation`), aggregated geographically (property `geonames:parentFeature`) to obtain a 6-type continent-level classification;
- *Political Party* — this task classifies person entities based on political party (property `dbo:party`), aggregated along their affiliation to international party federations (property `dbo:internationalAffiliation`) to obtain a 6-type classification;
- *Religion* — this task classifies persons based on religion (property `dbo:religion`), manually aggregated in a 5-type classification (e.g., by merging different Christian divisions);
- *Age* — this task classifies person in 6 age categories (e.g., young adult 25-34 years old), computed based on their birth dates (property `dbo:birthDate`);
- *Org. Size* — this task classifies organizations based on their numbers of employees (property `dbo:numberOfEmployees`), which is discretized in a 4-type classification;
- *Revenue* — this task classifies organizations based on their revenue (property `dbo:revenue`), which is normalized to use US dollars as currency, cleaned-up discarding outlier values (e.g., organizations with only few hundreds dollars revenue) and then discretized to form a 3-type classification;
- *Music Skill* — this task classifies musical artists based on their specialty (singers, instrumentalists, other), on the basis of DBpedia datatype property `dbo:background`.

Table 1 provides relevant statistics for the 8 prediction tasks extracted. For each task, it reports the number of predicted types (i.e., classes) and the number of entities in the fragment of DBpedia linked to Twitter here considered for which the type information to predict is respectively available — in which case a training sample is obtained — or missing — in which case the entity becomes a target for ontology population. The table also shows the *parent type* (with respect to predicted types) that entities must have for a prediction task being applicable (e.g., the *Political Party* task applies only to persons). Due to ongoing efforts in the Semantic Web community in interlinking LOD entities and social media profiles [10–12], we expect a significant increase in both the number of training samples and potential ontology population targets (see Table 5).

Table 1: DBpedia type prediction tasks, with entity parent type (for the task being applicable), # of predicted types, and # of entities w/ type (training set) and w/o type (population target) in the DBpedia fragment linked to Twitter.

Task	Parent type	Predicted types	# Samples (training)	# Samples (population)
Category	`owl:Thing`	17	49,639	n/a
Location	`owl:Thing`	6	38,153	14,134
Political Party	`dbo:Person`	6	1,912	37,143
Religion	`dbo:Person`	5	1,858	37,197
Age	`dbo:Person`	6	31,998	6867
Org. Size	`dbo:Organisation`	4	1,062	12,171
Revenue	`dbo:Organisation`	3	412	12,821
Music Skill	`dbo:MusicalArtist`	3	7,085	277

Table 2: Coverage (i.e., percentage of entities having the feature) and dimensionality statistics for the social features extracted from Twitter. Differences in coverage stem from information unavailability in the Twitter stream.

Source	Feature	Coverage	# Dimensions
Profile	Language	79.5%	47
	Top-level domain of a URL	79.5%	288
	Followers count	79.5%	1
	Friends count	79.5%	1
	Listed count	79.5%	1
	Favorites count	79.5%	1
	Statuses count	79.5%	1
	Is protected	79.5%	1
	Is verified	79.5%	1
	Geo tagging enabled	79.5%	1
	Has profile images/tiling	79.5%	4
	Description (LSA)	79.5%	100
Social graph	RT+mentions (sparse)	87.9%	2,203,062
	RT+mentions (dense)	100.0%	300
Text of tweets authored by the user	Text (LSA)	79.5%	100
	Text (Bag-of-words)	79.5%	972,001
	Hashtags (Bag-of-words)	62.9%	169,679
Text of tweets mentioning the user	Text (LSA)	86.1%	100
	Text (Bag-of-words)	86.1%	972,001

5 ENTITY REPRESENTATION WITH SOCIAL FEATURES

As shown in Figure 2, we build an entity representation starting from social media data linked to the entity, possibly augmented with RDF features. As RDF features consist of existing RDF embeddings, our focus and contribution here is on the extraction of *social* features from the social network we consider in this work, i.e., Twitter.

We start by obtaining the list of Twitter accounts we are interested in by following the links from DBpedia to Twitter. Then we process the 4 TB of tweets gathered covering the period from 2013 until 2017 using the Streaming API, filtering out the tweets not related to users in the list. From the remaining tweets, we extract features based on four feature families: (i) social graph features; (ii) profile features; (iii) textual features from a user's own tweets; and (iv) textual features from tweets that mention the target user. Table 2 provides summary statistics for all the four feature families, which are detailed in the remainder of the section. The feature families that we chose provide a comprehensive representation of available user information and performed well in various social media analysis tasks before [7, 9, 12, 23]. For feature subspaces with increased sparsity, such as text and social graph, we also employ low-dimensional dense representations, i.e., embeddings.

Social Graph Features The social graph, while being regarded in the literature as one of the most prominent features for a variety of tasks including user profiling, community detection, and many others, is incredibly hard to obtain at scale. At the time of writing, Twitter allows the gathering of only 5,000 edges per minute via its REST API, which, in our case, translates to months of crawling time. Instead, we follow the procedure recently developed by Nechaev et al. [12], There the data extracted from the Twitter Streaming API is used to build an approximation of the real social graph. The

approximation is computed by extracting mention and retweet interactions between users, where a "follow" edge is generated from the mentioning/retweeting user to the target one for each such interaction, yielding a graph with 2.7B edges. We introduce the resulting social graph into our feature model in sparse form with 2.2M dimensions, encoding each node as a bag of adjacent nodes with the appropriate weight. As in [12], we also introduce a dense representation for each user. To this end, we learn embeddings of size 300 for the 500K most frequently followed users by building the co-occurrence matrix and estimating factorization using the Swivel algorithm [21]. This algorithm is inspired by the distributional semantics hypothesis for natural language, in that the users that interact with the same profiles will end up being similar in the resulting vector space. Then, to obtain the dense representation for an arbitrary user, we compute a weighted average of the embeddings of his/her friends in the top 500K list. If no friend has an embedding or if the user is not in the social graph, we emit a default representation by averaging all the available embeddings. This procedure provides perfect coverage allowing to produce a rough approximation even for users not seen in the Twitter stream.

Textual Features As seen in many studies tackling the user profiling task (see Section 7), the user-generated textual content can be a prominent feature exhibiting outstanding inference performance for attributes such as age, location, nationality, interests, and many others. We consider two sources of textual content for each user: the text that is authored by the user and the text of tweets that are mentioning the target user. Text from both sources is accumulated and tokenized. Then special entities in the text, such as URLs, hashtags, and mentions, are filtered out. The resulting term sets are converted to sparse bag-of-words representations, with each term weighted using the tf-idf scheme. As with the social graph, we emit a dense embedding of each sparse vector. To this end, we employ Latent Semantic Analysis (LSA) to map a sparse vector into a dense one of size 100. In addition to text, we build a simplified bag-of-words representation considering only the hashtags that were used in tweets authored by the user.

Profile-based Features Each tweet object contains a snapshot[3] of profile data for the author of the tweet. To produce a representation based on this data, we collect the latest snapshot for each user and extract a variety of features from it. Among those are categorical features, such as a top-level domain name of the URL field and the user's self-declared language encoded as one-hot vectors, binary features for each boolean attribute in the profile, and the dense LSA representation of the user-supplied description. In total, the user object is converted into a feature vector of size 447.

6 EXPERIMENTS

In this section, we investigate the feasibility of performing type prediction using social media data. Namely, we assess the use of Twitter-based features (as described in Section 5) alone and in conjunction with state-of-the-art DBpedia entity representations (RDF embeddings). Additionally, we compare those features with the ones derived from Wikipedia text as proposed in [1]. We conclude this section by providing an analysis of the contribution of dense social features to the overall performance.

6.1 Experimental Setting

We consider the type prediction tasks introduced in Section 4 and summarized in Table 1, each task consisting of a set of mutually disjoint types (e.g., different person and organisation categories) and a dataset of entities from DBpedia (version 2016-04) having those types and a link to a corresponding Twitter profile.

For each task, we consider the following prediction approaches:

- *MF* — a most frequent baseline always predicting the type with the largest number of entities in the task dataset, whose performances represent a lower bound of the performances achievable on the task, and give an idea of its difficulty;
- *RDF* — a linear Support Vector Machine (SVM)[4] classifier using the *PageRank Split* RDF embeddings for DBpedia 2016-04 [3] as entity representations. This particular embedding weighting schema was selected based on preliminary experiments, the details of which are available online;
- *Social* — a linear SVM using our Twitter-based social features introduced in Section 5 as entity representations, produced for all the task entities based on the tweets sampled in 4 years from the Twitter Streaming API;
- *Social+RDF* — a linear SVM using a combination of social features and RDF embeddings as entity representations.

It is worth pointing out that the *RDF* embeddings we used[5] were computed from a knowledge graph that includes the entity types being predicted for the tasks *Category*, *Location*, *Political Party*, and *Religion*. Therefore, the performances of approach *RDF* in these tasks may be overestimated, influencing the comparison with other approaches. This problem does not affect the remaining datasets, which were generated starting from datatype properties not used for producing the RDF embeddings. The RDF embeddings [19] are used in many tasks in the literature outperforming other graph-based entity representations, both sparse and dense, therefore we employ them in our experiments as a baseline.

For each task dataset, we evaluate the performances of *RDF*, *Social* and *Social+RDF* classifiers via a 5-fold cross-validation protocol, where we collect predictions for all the entities in the task dataset by iteratively selecting one of the five partitions and applying on it the classifier trained on the remaining four partitions. During each training step (five in total), we apply a nested 3-fold cross-validation loop to select the optimal classifier hyper-parameters (the regularization parameter C) and the classifier score threshold below which we should abstain in order to balance precision and recall (i.e., optimize $F_{1\ micro}$). This threshold is then used at prediction time to abstain and discard the types predicted with low confidence. The *MF* baseline never abstains.

As evaluation measures we use Precision (P), Recall (R), and F1 scores in their micro-averaged (P_{micro}, R_{micro}, $F_{1\ micro}$) and macro-averaged (P_{macro}, R_{macro}, $F_{1\ macro}$) variants, as commonly defined

[3] Twitter documentation article about the user object: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/user-object

[4] We use the LibLinear [4] software package with L2 regularization and L1 (Hinge) loss. LibLinear was chosen as it copes well with large feature spaces. We tested other classifiers (SVM with Gaussian/polynomial kernels, Random Forests) obtaining similar performances but much higher run times.

[5] RDF embeddings downloaded from http://data.dws.informatik.uni-mannheim.de/rdf2vec/models/DBpedia/2016-04/GlobalVectors/

Table 3: Type prediction performances. Statistically significant differences w.r.t. *Social* are marked with $^+$ if better, $^-$ if worse; overestimated performances are marked with *.

Task	Approach	P_{macro}	R_{macro}	$F_{1\ macro}$	P_{micro}	R_{micro}	$F_{1\ micro}$
Category	*MF*	0.016^-	0.059^-	0.026^-	0.277^-	0.277^-	0.277^-
	RDF*	0.612^-	0.520^+	$\mathbf{0.539}^+$	0.759^+	0.721^+	$\mathbf{0.740}^+$
	Social	0.666	0.387	0.445	0.667	0.610	0.637
Location	*MF*	0.069^-	0.167^-	0.097^-	0.412^-	0.412^-	0.412^-
	RDF*	0.466^-	0.299^-	0.292^-	0.592^-	0.580^-	0.586^-
	Social	0.904	0.680	**0.763**	0.878	0.796	**0.835**
Political Party	*MF*	0.072^-	0.167^-	0.100^-	0.431^-	0.431^-	0.431^-
	RDF*	0.441^-	0.316^-	0.327^-	0.584^-	0.541^-	0.561^-
	Social	0.721	0.527	**0.596**	0.812	0.728	**0.767**
Religion	*MF*	0.122^-	0.200^-	0.152^-	0.610^-	0.610^-	0.610^-
	RDF*	0.723	0.358^-	0.374^-	0.688^-	0.681^-	0.684^-
	Social	0.651	0.474	**0.488**	0.726	0.721	**0.723**
Age Category	*MF*	0.054^-	0.167^-	0.082^-	0.326^-	0.326^-	0.326^-
	RDF	0.266^-	0.239^-	0.222^-	0.362^-	0.362^-	0.362^-
	Social	0.423	0.320	0.325	0.451	0.450	0.450
	Social+RDF	0.431	0.332^+	$\mathbf{0.339}^+$	0.462^+	0.456^+	$\mathbf{0.459}^+$
Org. Size	*MF*	0.076^-	0.250^-	0.117^-	0.304^-	0.304^-	0.304^-
	RDF	0.359^-	0.379^-	0.350^-	0.401^-	0.400^-	0.401^-
	Social	0.417	0.428	0.417	0.441	0.440	0.440
	Social+RDF	0.453^+	0.458^+	$\mathbf{0.447}^+$	0.481^+	0.477^+	$\mathbf{0.479}^+$
Revenue	*MF*	0.127^-	0.333^-	0.184^-	0.381^-	0.381^-	0.381^-
	RDF	0.514	0.509	0.480	0.522	0.515	0.518
	Social	0.555	0.533	0.531	0.548	0.544	0.546
	Social+RDF	0.539	0.530	0.519	0.541	0.539	0.540
Music Skill	*MF*	0.254^-	0.333^-	0.289^-	0.763^-	0.763^-	0.763^-
	RDF	0.255^-	0.333^-	0.289^-	0.765^-	0.762^-	0.763^-
	Social	0.664	0.424	0.436	0.804	0.792	0.798
	Social+RDF	0.653	0.434^+	$\mathbf{0.451}^+$	0.803	0.792	0.798

for multi-class classification problems.[6] We test the statistical significance of the difference of those scores via the *approximate randomization* test [13] (significant if $p\text{-}value \leq 0.05$), and produce precision-recall curves by varying the abstain threshold on the prediction score returned by the classifier (the SVM margin). Both evaluation scores and statistical significance are computed on the predictions for the whole task dataset obtained via cross-validation.

The whole pipeline required a couple of days to extract features from the stream of tweets (using Apache Flink), 14 hours of GPU time to produce dense *Social* features, and additional 7 hours for the nested cross-validation training and evaluation of all the classifiers on the 8 type prediction tasks (10-core E5-2630 machine with 192 GB RAM and GeForce GTX 1080 GPU).

6.2 Experimental Results

Table 3 reports the performance scores obtained by the different approaches on the 8 type prediction tasks, with indication of statistically significant differences with respect to *Social* ($^+$ if significantly better, $^-$ if significantly worse). The first four tasks — *Category*, *Location*, *Political Party*, *Religion* — are the ones where predicted types were included in the RDF embeddings of approach *RDF*, whose performances may be overestimated (marked with *); for these tasks, we do not consider the *Social+RDF* combination approach.

Compared to the baseline approach *MF*, *Social* always results in better performances (whereas approach *RDF* performs like the baseline in task *Music Skill*), with statistically significant differences that are higher for tasks *Location*, *Category*, and *Political Party*, somehow suggesting that these tasks are "easier" than the others.

Compared to approach *RDF*, *Social* outperforms *RDF* in all the 4 tasks where a proper comparison is possible (*Age*, *Org. Size*, *Revenue*, *Music Skill*), with statistically significant differences in 3 out of 4 tasks. In the other tasks, *Social* still manages to outperform *RDF* in 3 out of 4 tasks (*Location*, *Political Party*, *Religion*, with only exception of *Religion* P_{macro}), a remarkable result given that *RDF* performances may be overestimated in these tasks. We note that approach *RDF* significantly outperforms *Social* in task *Category*, whose data was obtained directly from DBpedia `rdf:type` triples.

The fact that approach *Social* is competitive with respect to approach *RDF* (outperforming it in most cases), suggests that the proposed social media entity representation may contribute positively in an ontology population task. This is confirmed by considering approach *Social+RDF* that combines *Social* and *RDF*. In the four tasks where we evaluate approach *Social+RDF*, it always outperforms *RDF* (expected, as *Social* outperforms it too on these tasks), and in 2 tasks out of 4 it also outperforms *Social* (score differences statistically significant except for P_{macro} on task *Age*), showing that the combined representation is generally better than its components taken separately.

[6] Given a prediction task with types $t_i \in T$, we define tp_i (true positives), fp_i (false positives), and fn_i (false negatives) as the numbers of entities respectively: of type t_i correctly classified as t_i; of type $t_j \neq t_i$ wrongly classified as t_i; of type t_i wrongly classified as $t_j \neq t_i$. Based on that, we have:

$$P_{micro} = \frac{\sum_i tp_i}{\sum_i tp_i + \sum_i fp_i} \quad R_{micro} = \frac{\sum_i tp_i}{\sum_i tp_i + \sum_i fn_i} \quad F_{1\ micro} = \frac{2 \cdot \sum_i tp_i}{2 \cdot \sum_i tp_i + \sum_i fp_i + \sum_i fn_i}$$

$$P_{macro} = \sum_i \frac{tp_i}{tp_i + fp_i} \quad R_{macro} = \sum_i \frac{tp_i}{tp_i + fn_i} \quad F_{1\ macro} = \sum_i \frac{2 \cdot tp_i}{2 \cdot tp_i + fp_i + fn_i}$$

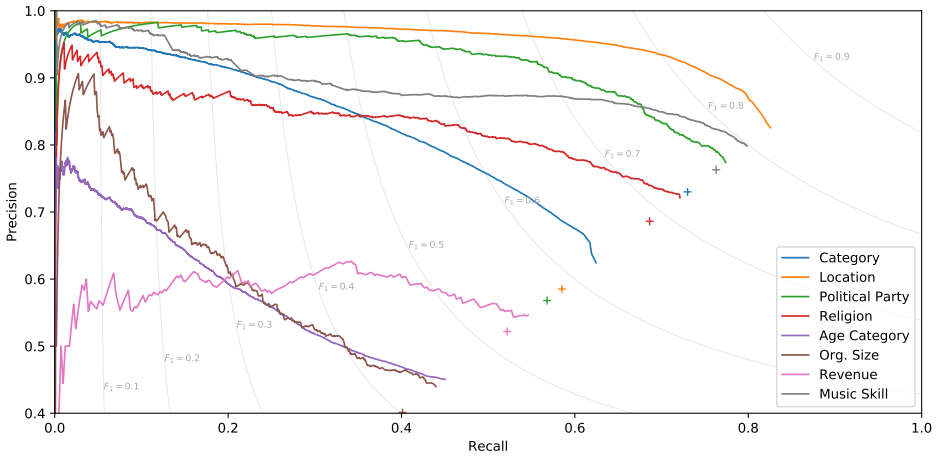

Figure 3: Precision-recall curves for different prediction tasks: lines correspond to approach *Social*, cross markers to approach *RDF* (best-$F_{1\ micro}$ setting).

As the precision scores in Table 3 (corresponding to the best $F_{1\ micro}$ score) may not be satisfactory in an ontology population task, we investigate whether better precision scores can be obtained at the expenses of recall, by changing the threshold on the classifier score to vary the precision/recall balance. The results are reported in Figure 3, which plots the precision/recall curves (P_{micro}/R_{micro}) for approach *Social* on the different tasks (performances of approach *RDF* for best-$F_{1\ micro}$ setting plotted with cross markers for reference). The plot shows that high precision levels ($P_{micro} > 0.9$) can be obtained for tasks *Category*, *Location*, and *Political Party*, thus opening up the possibility of performing ontology population from social media for these types.

6.3 Comparison to Wikipedia-based features

In addition to RDF embeddings, we investigate the usage of social media data along with Wikipedia-based features — a popular source of knowledge for ontology population. While DBpedia itself is primarily populated from Wikipedia infoboxes, the rest of the text on the corresponding Wikipedia page can still provide additional data for a given entity. Moreover, Wikipedia pages are typically available in many languages, allowing cross-language models to further boost the performance compared to the single language ones. We follow Aprosio et al. approach [1] for extracting features from Wikipedia articles to perform type prediction on our 8 tasks.

We thus add two additional approaches to the comparison:

- *Aprosio et al.* — a linear SVM using the $K_{\mathbf{combo}}$ features acquired from Wikipedia as described by Aprosio et al. [1]. This includes Wikipedia categories, sections, templates and words (both LSA-based dense and sparse) extracted from 6 language variants of each article: English, Italian, German, French, Spanish, and Portuguese;
- *All* — a linear SVM using a combination of *Aprosio et al.*, *Social* and *RDF* feature sets;

Table 4 reports the performances of these approaches when compared to the best performing models from Section 6.2: *Social+RDF*. The *Aprosio et al.* approach generally outperforms Social+RDF features (*Political Party*, *Age*, *Org. Size*, *Revenue* and *Music Skill*), while providing similar performance for the rest. The *All* approach achieve the best performances in all tasks, significantly better in five of them. Most notably, in *Location* task, this approach was able to reach 92% $F_{1\ micro}$, 7% higher than the next best model. This result shows that, given that the coverage may increase in the future, social media data is versatile enough to complement a variety of data sources for the ontology population task.

6.4 Dense social representations

When processing, analyzing and potentially releasing the social media data, it is essential to consider the privacy aspect of such actions. Even though we used public data to build our entity representation, in case we ever release it, there is always a risk of reverse engineering of our sparse textual and social graph-based subspaces and profile features, which may ease access to personal (although public) data. One way to make such reverse engineering much harder to perform is to pack sparse encodings into low-dimensional embeddings: such dense representations are typically trained to reflect similarities between objects, so the original information is greatly corrupted and almost impossible to restore.

Since many of our sparse features have such dense counterparts to improve the performance of the system, we have conducted additional tests to see how much of the performance would have been lost if we only use the "safe" dense dimensions. The $F_{1\ micro}$ performance decreased for tasks *Category*, *Location*, *Age*, *Revenue* and *Music Skill* on average by 5.6%, while for tasks *Religion*, *Org. Size* and *Political Party* there was no statistically significant difference. The complete results are available online.[2] Given that profile features and hashtags have not been represented with a complementary embedding, we believe that a complete privacy-friendly dense representation can be developed without loss of performances. Not only such representation can be safely released in future for research purposes, but it will also allow the usage of machine learning algorithms not performing well on highly sparse input.

Table 4: Type prediction performances in comparison and in conjunction with Wikipedia-based features (*Aprosio et al. [1]*). Statistical significance is shown with respect to *All*.

Task	Approach	P_{macro}	R_{macro}	$F_{1\ macro}$	P_{micro}	R_{micro}	$F_{1\ micro}$
Category	*Social+RDF*	0.784^{-}	0.611^{-}	0.658^{-}	0.814^{-}	0.797^{-}	0.805^{-}
	Aprosio et al.	0.888^{+}	0.597^{-}	0.704^{-}	0.896^{+}	0.667^{-}	0.764^{-}
	All	0.876	0.767	**0.811**	0.883	0.872	**0.878**
Location	*Social+RDF*	0.893^{-}	0.717^{-}	0.781^{-}	0.865^{-}	0.843^{-}	0.854^{-}
	Aprosio et al.	0.955^{+}	0.674^{-}	0.782^{-}	0.961^{+}	0.755^{-}	0.846^{-}
	All	0.934	0.840	**0.880**	0.931	0.923	**0.927**
Political Party	*Social+RDF*	0.729^{-}	0.574^{-}	0.631^{-}	0.805^{-}	0.757^{-}	0.780^{-}
	Aprosio et al.	0.798	0.587	0.657	0.815^{-}	0.775^{-}	0.795^{-}
	All	0.801	0.621	0.687	0.871	0.807	**0.838**
Religion	*Social+RDF*	0.639^{-}	0.536^{-}	0.561^{-}	0.752^{-}	0.746^{-}	0.749^{-}
	Aprosio et al.	0.758	0.492^{-}	0.544^{-}	0.747^{-}	0.737^{-}	0.742^{-}
	All	0.758	0.569	**0.601**	0.791	0.771	**0.781**
Age Category	*Social+RDF*	0.416^{-}	0.320^{-}	0.320^{-}	0.455^{-}	0.452^{-}	0.453^{-}
	Aprosio et al.	0.460^{-}	0.382^{-}	0.391^{-}	0.498^{-}	0.494^{-}	0.496^{-}
	All	0.488	0.421	**0.434**	0.526	0.523	**0.525**
Org. Size	*Social+RDF*	0.425^{-}	0.438^{-}	0.425^{-}	0.456^{-}	0.455^{-}	0.455^{-}
	Aprosio et al.	0.497	0.512	0.490	0.518	0.518	0.518
	All	0.509	0.494	0.496	0.531	0.505	0.518
Revenue	*Social+RDF*	0.571	0.536^{-}	0.537^{-}	0.564^{-}	0.549^{-}	0.556^{-}
	Aprosio et al.	0.558	0.554	0.548	0.575	0.566	0.570
	All	0.606	0.584	0.586	0.608	0.595	0.601
Music Skill	*Social+RDF*	0.617^{-}	0.427^{-}	0.447^{-}	0.806^{-}	0.784^{-}	0.795^{-}
	Aprosio et al.	0.731	0.596	0.638	0.854	0.846	0.850
	All	0.736	0.598	0.638	0.851	0.844	0.848

Table 5: Amount of samples for each task when using data from SocialLink v2 compared to DBpedia (see Table 1).

Task	# Samples (training)		# Samples (population)	
	DBpedia	SocialLink	DBpedia	SocialLink
Category	49,639	498,842	n/a	n/a
Location	38,153	330,803	14,134	168,039
Political Party	1,912	16,941	37,143	380,232
Religion	1,858	10,587	37,197	386,586
Age	31,998	309,018	6,867	87,070
Org. Size	1,062	10,871	12,171	90,798
Revenue	412	5,452	12,821	96,217
Music Skill	7,085	25,011	277	1,050

7 RELATED WORK

Type prediction Type prediction is a well-studied task in the Semantic Web community. The goal is to significantly increase the coverage typically by reusing the data that is already present in the knowledge base. Latest studies on this topic cast the type prediction task as the multi-label classification problem.

Melo et al. [8] introduce SLCN (Scalable Local Classifier per Node) showing, to the best of our knowledge, the current state-of-the-art performances for type prediction. They compare SLCN to the previous statistical and heuristics-based approach called SDType [14] demonstrating improvements in all cases. Their system extracts features from the entities in the knowledge graph and uses off-the-shelf classifiers to generate predictions. Then a special procedure is introduced to produce the inferred types. Rico et al. [18] iterate on this idea trying to solve the partial depth problem of multi-class classifiers. As in Melo et al. they extract features from entities and use off-the-shelf classifiers, such as Naive Bayes and a densely connected neural network to infer types. Comparison against the SDType [14] approach shows same or better results. Kejriwal et al. [6] use low-dimensional dense representations of entities (embeddings) by Ristoski et al. [19] to acquire representations for types in the same vector space. The resulting shared vector space allowed them to perform the type recommendation task, where for each entity a ranked list of topically relevant entities is produced. They show that entity embeddings can be utilized efficiently to infer type information. This idea was also suggested as a possible future direction in Ristoski et al.

To summarize, multi-label classifiers based on vector entity representations extracted from the knowledge graph are exhibiting state-of-the-art results for the type prediction task.

Here we design additional features that can be used in conjunction with the ones employed in these works. Similarly, Aprosio et al. [1] brings the semi-structured data from different language chapters of Wikipedia to improve the type prediction performance. We provide the comparison to their features in Section 6.3.

Embeddings Recently, attempts have been made to obtain generic low-dimensional representations of entities in a knowledge graph to provide a better alternative to sparse binary representations. Such representations are typically learned using an unsupervised machine learning algorithm employing the entire knowledge graph for training. The Node2vec [5] approach has introduced the idea of generating generic embeddings of nodes in a graph based on the distributional semantics hypothesis used in the Natural Language Processing community. The RDF2Vec [19, 20] approach further refined this idea to produce specialized graph embeddings for the RDF graphs found in KBs. In this paper, we utilize a further iteration of this work [3] based on GloVe, which was made available for download by the authors on their website. We use a modification of

GloVe called Swivel [15] in this paper to learn embeddings for users from the approximated social graph. Previous studies about the generation of generic graph embeddings, such as DeepWalk [16], LINE [22], and Trans-E [2] do not exhibit performance gain compared to the node2vec or GloVe-based embeddings. Therefore, we consider Cochez et al. [3] a state-of-the-art entity representation approach and use it as a baseline for our experiments.

Social media analysis Social media analysis and user profiling, in particular, are being extensively researched topics both from the computer science and the societal standpoint. Over the years all possible combinations of features extracted from social media have been explored including social graph, textual, image and video content, semi-structured profile data and various metadata. Most notably, Li et al. [7] used all available user information to predict latent attributes of a user using weak supervision. Zheleva et al. [23] have demonstrated the importance of social graph-based features by inferring hidden attributes of completely private profiles. Defense mechanisms to protect users from such inference have also been investigated [9].

To facilitate the usage of social media data in the Semantic Web environments, the SocialLink project [10–12] was created, further interlinking the LOD cloud and the social media. SocialLink enables more than tenfold increase in the number of entities which we can use for training and population, which will make our approach even more performant (by providing more training samples) and useful. Coverage comparison is provided in Table 5.

8 CONCLUSIONS

In this paper, we showcase the use of social media data to perform entity type prediction on knowledge graphs. In particular, we show that Twitter data is able to complement existing RDF-based entity representations from DBpedia when used as input in the supervised type prediction setting. Our approach employs rich domain-independent representations derived from written text, social graph, and user profile attributes on Twitter, efficiently utilizing embeddings to further boost the performance and increase coverage. We demonstrate significant performance improvements (up to 11.7% $F_{1\ macro}$) of such hybrid approach on four different type prediction tasks compared to the performance of the state-of-the-art RDF-based representation by Cochez et al. [3].

By trading off the recall of such approach, the injection of social media data may allow expanding current ontology population efforts for knowledge bases, such as DBpedia, with the population of entity types from social media data. Moreover, more efficient and specialized machine learning techniques, such as SLCN [8], can be integrated in our approach to boost prediction results even further. In addition, we showed that Twitter data can be used in conjunction with Wikipedia text, significantly improving the performance in most of the considered prediction tasks.

In future work, Wikidata can be used both as the target knowledge graph and as an additional source of links, which may be gathered also from community efforts such as SocialLink [12]. These additional links will provide (i) more linked entities whose types (where missing) can be predicted and populated using our approach; and (ii) additional training (where types are known), which in turn may allow targeting a more extensive range of types and performing additional analyses, e.g., of the impact on performances of the amount of available social information. Finally, a joint embedding derived from all user-related data can be learned to address potential privacy concerns when making the data from social media available to the community.

REFERENCES

[1] Alessio Palmero Aprosio, Claudio Giuliano, and Alberto Lavelli. 2013. Automatic Expansion of DBpedia Exploiting Wikipedia Cross-Language Information. In *The Semantic Web: Semantics and Big Data (ESWC)*.

[2] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Proc. of 27th Conf. on Neural Information Processing Systems (NIPS)*. 2787–2795.

[3] Michael Cochez, Petar Ristoski, Simone Paolo Ponzetto, and Heiko Paulheim. 2017. Global RDF Vector Space Embeddings. In *Proc. of 16th Int. Semantic Web Conf. (ISWC)*. 190–207.

[4] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *J. Mach. Learn. Res.* 9 (June 2008), 1871–1874.

[5] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proc. of 22nd Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*. ACM, 855–864.

[6] Mayank Kejriwal and Pedro Szekely. 2017. Supervised typing of big graphs using semantic embeddings. In *Proc. of Int. Workshop on Semantic Big Data (SBD@SIGMOD)*. 3:1–3:6.

[7] Jiwei Li, Alan Ritter, and Eduard Hovy. 2014. Weakly Supervised User Profile Extraction from Twitter. In *Proc. of 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*. 165–174.

[8] André Melo, Heiko Paulheim, and Johanna Völker. 2016. Type Prediction in RDF Knowledge Bases Using Hierarchical Multilabel Classification. In *Proc. of 6th Int. Conf. on Web Intelligence, Mining and Semantics (WIMS)*. 14:1–14:10.

[9] Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2017. Concealing Interests of Passive Users in Social Media. In *Proc of Re-coding Black Mirror ISWC Workshop*.

[10] Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2017. Linking Knowledge Bases to Social Media Profiles. In *Proc. of 32nd Symposium on Applied Computing (SAC)*. 145–150.

[11] Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2017. SocialLink: Linking DBpedia Entities to Corresponding Twitter Accounts. In *Proc of Int. Semantic Web Conf. (ISWC)*. 165–174.

[12] Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2018. SocialLink: Exploiting Graph Embeddings to Link DBpedia Entities to Twitter Profiles. *Progress in Artificial Intelligence* (2018).

[13] Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

[14] Heiko Paulheim and Christian Bizer. 2013. Type Inference on Noisy RDF Data. In *Proc. of 12th Int. Semantic Web Conf. (ISWC)*. 510–525.

[15] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[16] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *Proc. of 20th Int. Conf. on Knowledge Discovery and Data Mining (KDD)*. 701–710.

[17] Guangyuan Piao and John G. Breslin. 2017. Inferring User Interests in Microblogging Social Networks: A Survey. *CoRR* abs/1712.07691 (2017).

[18] Mariano Rico, Idafen Santana-Perez, Pedro Pozo-Jimenez, and Asuncion Gomez-Perez. 2018. Inferring New Types on Large Datasets Applying Ontology Class Hierarchy Classifiers: The DBpedia Case. In *Proc. of 15th Extended Semantic Web Conference (ESWC)*. To appear.

[19] Petar Ristoski and Heiko Paulheim. 2016. RDF2vec: RDF graph embeddings for data mining. In *Proc. of 15th Int. Semantic Web Conf. (ISWC)*. 498–514.

[20] Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. 2017. RDF2Vec: RDF Graph Embeddings and Their Applications. *Semantic Web Journal* (2017).

[21] Noam Shazeer, Ryan Doherty, Colin Evans, and Chris Waterson. 2016. Swivel: Improving Embeddings by Noticing What's Missing. *CoRR* abs/1602.02215 (2016).

[22] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proc. of 24th Int. Conf. on World Wide Web (WWW)*. 1067–1077.

[23] Elena Zheleva and Lise Getoor. 2009. To Join or Not to Join: The Illusion of Privacy in Social Networks with Mixed Public and Private User Profiles. In *Proc. of 18th Int. Conf. on World Wide Web (WWW)*. 531–540.