Progress in Artificial Intelligence manuscript No.
(will be inserted by the editor)

SocialLink: Exploiting Graph Embeddings to Link DBpedia Entities to Twitter Profiles

Yaroslav Nechaev · **Francesco Corcoglioniti** · **Claudio Giuliano**

Received: 1 March 2018 / Accepted: 14 August 2018

Abstract SocialLink is a project designed to match social media profiles on Twitter to corresponding entities in DBpedia. Built to bridge the vibrant Twitter social media world and the Linked Open Data cloud, SocialLink enables knowledge transfer between the two, both assisting Semantic Web practitioners in better harvesting the vast amounts of information available on Twitter and allowing leveraging of DBpedia data for social media analysis tasks. In this paper, we further extend the original SocialLink approach by exploiting graph-based features based on both DBpedia and Twitter, represented as graph embeddings learned from vast amounts of unlabeled data. The introduction of such new features required to redesign our deep neural network-based candidate selection algorithm and, as a result, we experimentally demonstrate a significant improvement of the performances of SocialLink.

Keywords Social Media · Linked Open Data · Machine Learning · DBpedia

1 Introduction

Today it is hard to imagine a public person or an organization that does not have a social media account. Such entities typically have a rich presence in the social media, sharing content, engaging with their audience, maintaining and expanding their popularity. They typically post new content frequently and keep all the information in their profiles as relevant and precise as possible, so that a potential consumer or a fan can be informed about the latest developments in no time. Thus, social media have become a primary source of information providing up-to-date knowledge on a wide variety of topics, from major events to the opening hours of stores or what books or songs a particular celebrity likes. Coincidentally, such people and organizations often have dedicated Wikipedia pages, and thus corresponding entries in knowledge bases (KB) related to Wikipedia, such as DBpedia [20], YAGO [18], or Wikidata [10].

Data in social media and KBs present opposite characteristics. On the one hand, KBs provide high-quality structured information (e.g., YAGO has 95% accuracy [18]) that is easily accessible, e.g., due the use of open formats (RDF) and online publishing as Linked Open Data (LOD), while data from social media accounts is often noisy, unstructured, and hidden behind restrictive APIs. To extract from social media as much information as typically contained in a KB entry, sophisticated pipelines have to be built implementing tasks like event detection, user profiling, and entity linking. These tasks typically exploit supervised learning [7, 14, 15, 21, 22, 23], which requires training sets that are scarcely available and expensive to create manually. On the other hand, social media provide up-to-date (real-time) information, while contents in KBs may lag behind from hours to months. For Wikipedia-related KBs, such lag comprises both the time for changing the page (hours to months [12], based on popularity) and, for automatically extracted KBs like DBpedia and YAGO, the time for that change to propagate in the KB (months to years). Such lag may prevent using these KBs in some application scenarios.

In light of these differences, we have started the SocialLink[1] project to bridge the KB and social media worlds by linking KB entities to their corresponding social media profiles. Our motivation is to enable knowledge transfer be-

Yaroslav Nechaev
Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
University of Trento, Via Sommarive 14, 38123 Trento, Italy
E-mail: nechaev@fbk.eu

Francesco Corcoglioniti · Claudio Giuliano
Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
E-mail: {corcoglio, giuliano}@fbk.eu

[1] http://sociallink.futuro.media

tween the two. Indeed, bringing high-quality knowledge from KBs into social media enables and significantly simplifies a wide variety of tasks including Entity Linking [7,23,6] and User Profiling [24]. On the other hand, the up-to-date nature of social media can be exploited to help keeping KBs updated, providing new data and references[2] to existing facts.

The original implementation of SocialLink targets DBpedia as KB and Twitter as social network, and consists of a linking approach and a public LOD dataset. The *linking approach* (described in [25]) is a three-phase pipeline that (i) gathers, indexes and stores the data necessary to perform the linking (data acquisition phase), (ii) proposes a set of candidate social media profiles for each entity in a KB (candidate acquisition phase) and (iii) uses a deep learning-based model to select (or abstain from selection) the best possible candidate for the target entity (candidate selection phase). The approach is trained and evaluated using the 56,133 existing links to Twitter found in DBpedia and Wikidata, and is able to leverage large amounts of unsupervised data both from DBpedia and Twitter to improve performances. Source code and documentation are available in our GitHub repository.[3] As the approach and most of the features it uses are general, it may be potentially expanded to support additional KBs and social media presenting characteristics similar to the ones considered here (e.g., availability of names, textual content and connections for both KB entities and social media profiles). Concerning the *LOD dataset* (described in [26]), it consists of almost 300K high quality (more than 90% precision) alignments, obtained by applying the above linking approach to 2M living people and 500K currently existing organizations in DBpedia (entities from multiple DBpedia language chapters are considered for the LOD dataset). Additionally, the dataset contains raw scores for each candidate alignment allowing end users to tune the precision / recall balance as they see fit. We distribute the dataset in accordance with LOD best practices, reusing existing vocabularies and providing a SPARQL endpoint. New versions are produced periodically covering the latest data and algorithm improvements. Relevant statistics and the latest dataset version can be found on our website and Zenodo.[4]

Graph-based features, both on the social media and KB sides, is an essential source of information little exploited in the original SocialLink. On the social media side, the *social graph* plays a crucial role for many social media-related tasks, where it reveals a significant amount of information about the users. The social graph is basically a vibrant network of connections between users that is typically represented by an explicit "follow" action, which manifests the intent of a user (follower) to read content written by the followed user (friend). In one of our recent papers [24], we have shown that by just using SocialLink along with a simple rule-based technique one can infer interests of a passive user (i.e., a user not generating any content on his own) by exploiting the social graph. The social graph has also been used to determine user's location [35], gender, and political affiliation [39]. Moving to the KB side, knowledge is often encoded in RDF triples and can naturally be represented as a knowledge graph with entities as vertices and relations as edges. These connections between entities are a powerful mechanism used in literature for solving a wide range of tasks both in the Semantic Web and many other domains [33]. Due to difficulties in acquiring and encoding such graph-based features, previous versions of SocialLink have utilized them only indirectly, through measures such as the number of friends and followers of a social media profile or the indegree and the outdegree of a KB entity.

In this paper, we build upon and significantly extend our previous works [25,26] by presenting an improved version of the SocialLink linking approach that addresses the above shortcoming. Our contributions are threefold:

1. we introduce graph-based features trained from the large amounts of unsupervised data available on both the social media and the KB sides;
2. we redesign our candidate selection step to accommodate those new features and a larger training set;
3. we provide an extensive evaluation of the algorithm improvements made since the original approach [25,26].

Concerning the first contribution, we explore the addition of graph-based features into the feature space in the form of *embeddings*, i.e., low-dimensional vector representations of nodes learned using large amounts of unsupervised data. Embeddings typically capture similarities among the objects they encode, a particularly useful trait for linking tasks like ours. On the social media side, we build on Swivel [36] to derive *social graph embeddings* for Twitter user profiles. In doing so, we address two fundamental challenges. Firstly, the social graph is typically very expensive to acquire at scale, as many social media obscure or hide the social graph altogether from third parties or, where it is available, significantly limit the number of user connections that can be sampled over a fixed period. In this paper, we show that the social graph can be efficiently approximated using retweet and mention relations mined from the sampled tweet stream provided by the Twitter Streaming API (the same stream we leveraged in [26] to overcome similar limitations affecting the *candidate acquisition* phase). Secondly, there are much more users in the social media than entities in the KB or words in any reasonable vocabulary, and most of the embedding generation approaches are not reasonably scalable to accommodate the complete social graph of any major social media. For example, Cochez et al. [5] provide embeddings with one of the largest vocabularies we

[2] https://meta.wikimedia.org/wiki/Grants:Project/Hjfocs/soweego

[3] https://github.com/Remper/sociallink

[4] https://zenodo.org/record/820160

have seen, yet it is still two orders of magnitude smaller than what is required by our task. We solve this issue by training embeddings only for the limited subset of the most followed users on Twitter and then treating other users as a weighted sum of the users they follow. Moving to the KB side, we leverage recent research results on RDF-based embeddings by Cochez et al. [5], using precomputed models made available by them [34] that cover almost 9M entities from the English DBpedia. In particular, we have found that among the models based on GloVe [29], the PageRank-based weighting schemes provide the best improvement in our task, consistently with the general findings reported by the authors.

Concerning the second contribution, the introduction of the new features inevitably required modifications to our original model. Initially, we have tried to use the embeddings by directly concatenating them with our old feature sets. However, since we are simultaneously adding vectors from two completely independent feature spaces, it becomes hard for the densely connected neural network to consistently find a solution that brings an improvement to the whole task. To this end, we show that by changing the topology of the network with the addition of a transformation layer followed by the multiplication of the transformed embeddings, instead of their simple concatenation, we are able to assist the training algorithm in finding a better solution overall.

Concerning the third and final contribution of this paper, it consists in an extensive evaluation of the new model in comparison with the old one [25,26]. This evaluation is performed on a new, larger gold standard for English DBpedia entities, which includes recent Twitter data and fixes and additions provided by the DBpedia and Wikidata communities, resulting in an overall growth from 35,149 [25] to 56,133 alignments. We evaluate both the improved model here presented and our old model [26] and other two baselines on this new gold standard, showing that the improvements introduced in this paper significantly increase performances.

The rest of the paper is structured as follows. In Section 2 we introduce the linking task in more details. Section 3 provides an overview of the SocialLink pipeline including the description of features in the old model. Section 4 introduces the new graph-based features, while Section 5 describes the new neural model. The evaluation of the new additions is provided in Section 6. Section 7 briefly present the LOD resource obtained from running the SocialLink pipeline. We present related work in Section 8 and conclude summarizing our findings in Section 9, with Appendix A comparing different word embeddings for our task.

2 Problem Definition

Our goal is to find a profile of an entity (person or organization) in a particular social network given the knowledge base (KB) entry for the entity.[5] In the following, we consider the DBpedia KB and the Twitter social network, although the task definition and most of the remarks here are general and may apply to other KBs and social media with similar characteristics.

The information available in a KB entry depends on the KB considered and, within the same KB, may be different from entity to entity. DBpedia itself, although based on Wikipedia that is being updated by millions of people every day, can have various issues including inconsistency, noisiness, obsolete knowledge and unavailability of entity attributes. An entry about the President of the United States can, for example, contain many attributes from different domains, while an entry for a regional-level politician in a non-English speaking country can basically contain a name, a description and the occupational class. This heterogeneity requires an approach that can work with the bare minimum of information known about the target entity. Here, we assume that a KB entry at least contains the name and a textual description, person vs organization type information, and some temporal information allowing the distinction between alive/existing entities and non-existing ones.

Working with social media from the outside also imposes a number of challenges. First, similar to entities, also the amount and quality of information available in a social media profile may vary. Specifically, profiles can be private, have limited attributes available, and / or contain confusing or inaccurate information. Therefore, a linking approach has to use the attributes that are most widely available in social media. They include, for example, user name, social graph, posting behavior, textual description and user-generated content and a special "verified" flag issued by the social media that certifies the identity of the profile owner.

Second, for famous people and organizations, there typically exist impersonating and fan profiles that can be very similar to the real one. Since most of the entities do not try to acquire the "verified" flag, it can be hard even for a human to distinguish them. Moreover, certain groups of people, e.g., politicians and athletes, tend to have multiple profiles that correspond to various periods in their life. A politician might create a new profile if he was elected, an athlete can do the same when changing teams. Some famous people tend to have an official and a personal profile. In all these cases, finding the right profile among very similar options is hard.

Third, for Twitter and many other social media it is not feasible to acquire the entire social network, due to its enormous size and the API limitations. Therefore, to acquire the candidate profiles for a target KB entity one has to use the available API request quota sparingly, which limits the amount of candidates and the types of candidate informa-

[5] We start from KB entries as they are entirely known in advance, differently from social network profiles that can be only queried or (partially) acquired via expensive crawling.

Table 1: Information in gold standard and DBpedia 2016-04, including average # of names and description length (chars)

	Live Entities (per, org)	Persons percentage	Avg. # names	Avg. desc. length
Gold standard	56,133	72.98%	1.81	547
DBpedia (EN)	1,123,735	71.05%	1.89	525
DBpedia (All)	2,589,023	78.62%	1.61	518

tion that can be acquired. While these limits can be softened using more sophisticated API crawling techniques (in accordance with terms of use) and by leveraging possible information streams provided by the social network, like the sampled tweet stream we exploited in [26] and in this work, one cannot generally assume that all relevant candidates and their information can be accessible when linking an entity, if not only for the difficulty of processing such huge amount of information (e.g., running a classifier over millions of users for millions of KB entities). Therefore, in all the cases where it is impossible to link an entity to a profile we cannot infer that the entity is not present on the considered social network, as the right profile for the entity may just be not accessible and thus unknown (open world assumption).

The task that we solve in this paper is similar to the well-known problem of profile matching on social media, if we look at a KB entry as a special kind of profile. However, KBs do not contain attributes that were vital to matching profiles in previous studies, such as usernames, user-generated content, and social graph. Therefore, the techniques outlined in such studies cannot be directly applied in our case and cannot provide a baseline for evaluating our approach.

3 Approach Overview

In this section, we provide an overview of SocialLink approach for linking DBpedia entities to Twitter user profiles. We build on our previous work reported in [25,26], that we organize and summarize here pointing out the improvements contributed in this paper and further detailed in Section 4 (embeddings) and Section 5 (improved selection model).

Figure 1 highlights the three phases of the approach. Processing starts with the *data acquisition* phase (Section 3.1), where the required data from Twitter and DBpedia, enriched with fresh data from Wikidata and including preexisting gold standard alignments, are gathered, prepared, and indexed locally for further processing. Next, in the *candidate acquisition* phase (Section 3.2), for each DBpedia entity a list of candidate Twitter profiles matching the entity is obtained by querying the indexes. Finally, the *candidate selection* phase (Section 3.3) uses the gold standard to train a neural network that scores and selects the best matching candidate, or abstains if there is no suitable candidate.

3.1 Data Acquisition

During this phase, we gather and process large amounts of data to support further steps. This includes the retrieval and local indexing of entity data (RDF triples) from DBpedia and Wikidata and of user profile data from Twitter, as well as the generation of entity and profile *embeddings*, i.e., dense vector representations of objects learned from large amounts of unlabeled data and used as features in our approach.

Entity Indexing Entity information exploited in SocialLink consists of names, types, textual descriptions, live / not-alive status, relations to other KB entities, and preexisting alignments to social media profiles that form our gold standard. This data is mainly acquired from DBpedia[6] for *live* person and organization entities,[7] which account for the majority of the available DBpedia-Twitter alignments. We further enrich this data with more up-to-date alignments and entity death / closing dates from Wikidata, mapping from Wikidata entity identifiers to DBpedia entity identifiers using the owl:sameAs links from DBpedia and the RDFpro [8] tool for URI rewriting. To speed up processing, and overcome the limitations of public SPARQL endpoints, we build a local *entity index* consisting of a Virtuoso triplestore populated with all the required data. Here the 56,133 gold standard alignments (40,967 persons, 15,166 organizations, available on our website) are also extracted to be used for training the candidate selection phase.[8] Table 1 provides relevant statistics for the gold standard, compared to DBpedia in general (English chapter and all chapters, linkable live entities only).

Twitter Profile Indexing Profile information exploited in SocialLink consists of names, user-generated texts, and social relations (e.g., follow, retweet, reply and mention relations). This information can be obtained from the social media API, that for Twitter consists of either the ReST API or the Streaming API. We employed the first initially [25], but its rate limits (e.g., 180 user queries every 15 minutes) make difficult to acquire data for all the *candidate* Twitter profiles that may be linked to DBpedia. Consequently, we switched to the Streaming API [26] that provides a continuous (sampled) stream of tweets posted on Twitter, each one enriched with metadata and information about its author.

[6] English DBpedia version 2016-04, for what concerns the experiments reported in this paper (to enable comparison with original approach in [25]). The SocialLink LOD dataset released online is instead built using data from all language chapters of the most recent DBpedia.

[7] Entity alive status is gathered from temporal properties like `dbo:deathDate`, `dbo:deathYear`, `dbo:closingYear`, `dbo:closed`, `dbo:extinctionYear`, `dbo:extinctionDate`, `wikidata:P570`, `wikidata:P20`, `wikidata:P509`, or properties implying death like `dbo:deathPlace`, `dbo:deathCause`, `dbo:causeOfDeath`.

[8] Gold alignments derive from selected foaf:isPrimaryTopicOf and wikidata:P2002 triples of entities assumed living.

Fig. 1: The three processing phases of the SocialLink pipeline.

From the Twitter stream, we continuously collect and index the following data. First, the latest profile encountered for each user is recorded. Since the Streaming API is allowing us to observe only a subset of the complete Twitter stream, it is not guaranteed that all possible profiles will be extracted. However, this approach has yielded 241M most active users, which we consider sufficient for our task. Secondly, all the available text generated by each profile is gathered. This includes the tweets written by a user as well as the (time-varying) textual descriptions found in the user profile for the observed time period. Third, we extract the names related to a profile along with the frequency for each name. Finally, we record the interactions between users by extracting mentions and retweets. This information is needed to acquire an approximate social graph for each profile.

Collected user data is indexed in a PostgreSQL relational database and a basic full-text search index is built to allow efficient searching of profiles by name. Data processing is implemented using Apache Flink,[9] a framework providing reliability (via automatic checkpoints) and scalability (via automatic horizontal scaling). We have currently gathered more than four years of raw Twitter data, out of which 450 GB of indexed and accessible user data are produced. This setup of the system allows performing hundreds of queries per second on a single machine and enables frequent, reliable, and fully automatic population and update of the SocialLink dataset. Additionally, the user index provides much more user-related data compared to live querying of Twitter ReST API, thus increasing alignment performances in both candidate selection and candidate acquisition phases.

Embeddings To effectively exploit textual information of DBpedia entities (short abstracts) and Twitter profiles (user descriptions, tweets) and deal with lexical variability, in our work [25,26] we leverage low-dimensional vector representations of texts — i.e., embeddings — computed using a Latent Semantic Analysis (LSA) approach [19,9]. These word embeddings are derived from the term-by-document matrix of the English Wikipedia via dimensionality reduction (matrix factorization via singular value decomposition).[10] It is worth noting that a number of approaches were proposed to improve word embeddings in various ways over the last five years. We have conducted additional tests (see Appendix A) with some of them and, since such approaches did not introduce significant improvement on our task, we decided to stay with LSA to perform a proper comparison with the model proposed in our previous work [25].

In addition to LSA embeddings, which account for text information only, in this paper we introduce *graph embeddings* to account also for relational data, both in the KB and the social media, which we did not leverage before. On the KB side, we use the 'PageRank Split' variant of graph embeddings described in [5,4] and computed for the English DBpedia (version 2016-04),[11] which are a particular implementation of *RDF graph embeddings* [34]. On the social media side, we propose our own *social graph embeddings* that capture the information of a user's social relations, as detailed in Section 4. Together, these graph embeddings allow exploiting the large amounts of unlabeled RDF and Twitter data available online and in our indexes, to learn effective low-dimensional representations for entities and user profiles that can be exploited in the alignment task.

3.2 Candidate Acquisition

In this phase, given an entity to align to Twitter, we obtain a list of candidate Twitter profiles that is expected to contain the true candidate for the entity, if any. While we previously queried the Twitter ReST API [25] to produce the candidate list, we now employ a full-text search query targeted at our user index [26]. In both cases, the choice of query is significant: we want to maximize recall, i.e., the probability of finding the true candidate among the list, without introducing too much noise (i.e., unrelated candidates) that may decrease performances in the following selection phase.

[9] `http://flink.apache.org/`

[10] See [1, Section 3.2] for a detailed description of how LSA embeddings are computed.

[11] PageRank Split embeddings downloaded from `http://data.dws.informatik.uni-mannheim.de/rdf2vec/models/DBpedia/2016-04/GlobalVectors/`

Table 2: Example of candidate retrieval query.

Entity	`http://dbpedia.org/resource/Barack_Obama`
Names	`Barack Obama`, `Barack Hussein Obama`
Query	`(Barack Obama) OR (Barack Hussein Obama)`
Result	`@BarackObama` *(true candidate)*, `@ObamaNews` ... *other 38 candidates*

Based on our previous experience, where we developed and tested different query construction strategies [25], we currently adopt a strategy that combines all known names of a DBpedia entity as encoded by properties foaf:name and rdfs:label. Names consisting only of first or last name (i.e., foaf:name matching foaf:givenName or foaf:surname) are filtered out to prevent noisy results. Indeed, if for entity "John Smith" we were to keep the name "John", the list of potential candidates would contain all possible Johns, which is not desirable. The remaining names are deduplicated and the three most frequent ones (in collected foaf:name and rdfs:label properties) are OR-ed to form the query, as shown in the example in Table 2. If no results or too many results are obtained, the query is modified by selecting a different combination of names. The returned results are sorted based on the frequency with which we saw a name referring to a candidate profile in a stream of tweets. In the end, we save at most the top $k = 40$ results returned by the user index as candidates for the entity, this threshold empirically chosen with the goal of maximizing recall while reducing noise. So if the candidate with a name-based match is not mentioned or retweeted as much as other matching candidates, it would not appear in the resulting list.

Compared to live querying the Twitter ReST API, the current approach based on the user index allows a greater degree of flexibility in acquiring the candidate list for a DBpedia entity, as it enables different query strategies and allows bypassing API limitations in terms of query complexity and request rates (at most one request / 20 candidates per entity could be feasibly obtained with the ReST API), resulting in an increase of recall.

3.3 Candidate Selection

In this phase, given a DBpedia entity and the corresponding list of candidate Twitter profiles, we formulate a classification problem where the classifier has to provide a probability estimate of a candidate being a match of the target entity, for each considered ⟨candidate, entity⟩ pair.

As classifier we employ a deep neural network (DNN) trained on the gold standard DBpedia-Twitter alignments described earlier. Our initial DNN model [25,26] consisted of a stack (5 hidden layers with 256 units each) of densely-connected layers with *tanh* as activation function and *softmax* applied on top to acquire probability estimates. The DNN takes a feature vector as input consisting of the features of Table 3 and all their pairwise combinations, scaled to unit variance and zero mean and hereafter referred to as BASE features. The *Adam* algorithm is used to train the network, employing cross-entropy as cost function. Dropout with the probability of 0.5 is applied to each layer, and L2 regularization is used to prevent overfitting. In this paper, we revise and improve this architecture to include graph embeddings both for the entity and the candidate profile, as described in Section 5.

After the probability estimates are computed, the candidate with the best probability is selected. SocialLink can abstain from selection based on a minimal score threshold. Like in our previous paper [25], Section 6 provides precision / recall curves produced by changing this threshold during the evaluation on the gold standard dataset.

4 Graph-based Embeddings

The addition of graph-based embeddings in our case enables exploiting connections among entities and users, which is the data we have not used in our model before. Embeddings provide a dense low-dimensional representation of objects trained to reflect similarities between them. In case of neural language models, for example, individual embeddings typically follow the *distributional semantics* hypothesis: words that are used in the same contexts tend to have similar meanings, which means that in the resulting vector space such words will be close to each other according to some distance metric. Inspired by the overwhelming success of word embeddings in many tasks, similar approaches appeared for representing nodes in graphs. They follow the same principle: nodes that have similar neighborhood in a graph will have similar representations.

In this section, we motivate and describe the algorithm we have chosen to represent graph-based features from DBpedia, as well as present our novel approach for the social graph in Twitter. Both algorithms are based on neural language models, specifically, the global log-bilinear regression models that produce embeddings by factorizing[12] the co-occurrence matrix of objects (entities in case of DBpedia, users in case of Twitter).

4.1 Global vectors from the co-occurence matrix

In order to construct the aforementioned co-occurrence matrix following the distributional semantics hypothesis, one has to define the focus object and one or many context objects. In case of natural language, where the objects are words, for each focus word in a sentence, some neighboring words

[12] The regression models described in this section perform an approximate matrix factorization rather than the exact one used, for example, by LSA.

Table 3: Basic features used in the DNN classifier. Compared to [25], we dropped the Wikipedia-specific features that provide limited improvements.

Feature family	Feature type	Feature description
Name	4 scalars	edit distances are computed for pairs ⟨entity name, profile username⟩ and ⟨entity name, profile screen name⟩ using two metrics: Jaro-Winkler and Levenshtein resulting in four scores. Since an entity can have an arbitrary amount of names we compute average the scores across all of them. Such features are known to be useful when aligning profiles [38,37].
Description	2 scalars	two cosine similarity scores between the entity description (rdfs:comment) and the two texts derived from user-related textual content in Twitter: profile text (description, location, pinned tweet) and the content of tweets made by the user (as extracted from our stream of tweets). The average of tf-idf-weighted word embeddings was used to represent text.
Core profile metrics	4 scalars	logarithms of friends, followers, tweets and listed counts for the profile, to measure the popularity and activity level of the Twitter profile. These features capture the intuition that the true candidate profile for a KB entity is often the most popular and/or active one (this is especially the case for famous KB entities).
	1 binary	set to 1 if the profile has been 'verified' by the social media provider. Even if the percentage of verified entities is rather low, it is still one of the most effective features to help distinguishing a real profile from a fake one.
Homepage links	3 binary	we crawl links to Twitter profiles in DBpedia entity homepages (property foaf:homepage) and define three features to specify: (i) if the profile is contained in the list of profiles scraped for the entity; (ii) if the profile is the only one extracted; and (iii) if the profile contains a link back to the crawled homepage.
Entity type	2 binary	entity type chosen among person (dbo:Person type) and organization (dbo:Organization type), to permit the DNN classifier to learn a different strategy for persons and organizations.

are selected as its context. Each such occurrence is weighted based on a weighting function, such as $1/d$ if the two words are d words apart, and added to the co-occurrence matrix.

Following this approach through the unsupervised corpus results in the co-occurrence matrix $X \in \mathbb{R}^{v_f \times v_c}$, where v_f is the size of the vocabulary of focus words and v_c is the size of the vocabulary of context words (for simplicity, the same vocabulary size is typically used for both). Then, X is factorized into matrices $W \in \mathbb{R}^{v_f \times d}$ and $\tilde{W} \in \mathbb{R}^{v_c \times d}$, where d is the desired embedding size, by minimizing the objective:

$$J_{\text{GloVe}} = \sum_{i,j} f(x_{ij})(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \log x_{ij})^2 \quad (1)$$

where $f(x_{ij})$ is a weighting function and b_i and $\tilde{b}_j$ are biases. The model with objective (1) is referred to as the GloVe model [29] and effectively encourages $W\tilde{W}^\top$ to predict $\log X$. A similar model was subsequently investigated by Shazeer et al. [36]. Their model, Swivel, instead estimates the pointwise mutual information between the terms $\mathbf{pmi}(i;j)$, modifying the objective as follows:

$$\begin{aligned} J_{\text{swivel}} &= \frac{1}{2}\sum_{i,j} f(X_{ij})(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \mathbf{pmi}(i;j))^2 \\ &= \frac{1}{2}\sum_{i,j} f(X_{ij})(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \log x_{ij} \\ &\quad - \log|D| + \log x_{i*} + \log x_{*j})^2 \end{aligned} \quad (2)$$

where $x_{i*} = \sum_j x_{ij}$ and $x_{*j} = \sum_i x_{ij}$. The primary distinction between Swivel and GloVe is the introduction of the "soft hinge" loss for cases where $x_{ij} = 0$:

$$J_{\text{hinge}} = \log[1 + \exp(w_i^\top \tilde{w}_j - \log|D| + \log x_{i*} + \log x_{*j})] \quad (3)$$

This special case allows the unobserved co-occurrences to contribute to the learning process, producing more stable estimates for rare words.

For both GloVe and Swivel, the resulting embeddings W and $\tilde{W}$ are trained to estimate a particular word co-occurrence matrix. Given that the notion of focus and the context objects can be arbitrarily defined, the same approach can be applied to any task including the representation learning of nodes in a graph, provided a sufficiently large unsupervised dataset from which to extract the co-occurrence statistics is available (as in our case). The notion of co-occurrence between nodes and its weight can be defined differently to produce the co-occurrence matrix. For example, random walks [30, 17] have been used to obtain sequences of nodes, which are then treated as pseudo sentences similarly to language models. Cochez et al. [5] explore multiple weighting strategies to produce and weight co-occurrences in RDF graphs.

4.2 Social graph embeddings

SocialLink considers 241M Twitter users during the *candidate acquisition* step and this number continues to grow as we sample more data from Twitter. Each of those users can potentially end up being a candidate and would require a full set of features at the *candidate selection* step. Similarly to RDF graph embeddings, we would like to have vector representations for users in the social media based on the social graph capturing similarities between those users. Here we present an approach for computing such embeddings having in mind two fundamental issues that inevitably arise when trying to learn representations based on the social graph: the acquisition of the social graph and the expected coverage.

Social media APIs are incredibly restrictive: at the time of writing, Twitter allows only one request per minute covering at most 5,000 social graph edges at a time. To resolve this issue, instead of acquiring the exact social graph for each user, we record interactions, such as retweets and mentions, between users extracted from the Twitter Streaming API. In total, over 2.7B of such interactions were gathered. In Twitter, the users are unlikely to see a piece of content or a user that is not close to them in their social graph. This observation allows us to estimate the social graph by creating a "follow" edge for each such interaction. Additionally, we weight each edge in the graph by observing the locally normalized frequency of interaction: $\mathrm{sg}_{ij} = \mathrm{freq}_{ij} / \sum_k \mathrm{freq}_{ik}$. So if the user is interacting with a particular set of profiles more often, such profiles will have a more prominent weight.

The growing number of Twitter profiles in our user index poses unique requirements when designing user embeddings. Firstly, the learning approach has to be scalable. Learning embeddings for each of the 241M users would be simply impractical. Even though many of the methods would scale linearly with the size of the vocabulary, when increased by two orders of magnitude learning time and memory requirements would still become unreasonably high, not to mention the portability of the result. Indeed, most of the approaches would store embeddings in memory during training, which would quickly become a problem, especially if trained on a GPU. DeepWalk and node2vec use random walks, which are generated based on the number of nodes in the graph. The log-bilinear factorization approaches described above, in general, depend quadratically on the vocabulary size. However, GloVe in practice scales much better, since it is not utilizing unobserved co-occurrences in any way and the co-occurrence matrix is typically increasingly sparse. By comparison, Swivel would not benefit from such sparseness.

Scalability issues aside, the approach has to be able to reliably represent out-of-vocabulary users, which would inevitably appear as we continue to sample the stream of tweets. To address both issues, instead of learning embeddings for each user, we learn them only for profiles that are being followed (friends) and then represent other users as a weighted sum of their friends. This procedure allows us to build embeddings for any given user as long as he is following some profile for which we have the embedding.

To this end, for each user in our estimated social graph, the list of friends was retrieved and represented as a sequence. Then, each profile in this sequence was treated as a focus object while the rest were treated as co-occurring context objects with unit weights. This mimics natural language models by essentially treating users as documents and friends as words where the particular order of words does not matter. Since the computation of such co-occurrences is quadratic in the size of the sequence and we do not get much information by observing users with a large number of friends, we limit the maximum sequence size to 30,000 to speed up the computation. This procedure yields a dataset of almost 200M sequences from which the 500K most frequent users were chosen to form a vocabulary, and the co-occurrence matrix was calculated. Note that we consider each co-occurrence equally important at this stage.

The co-occurrence matrix is then factorized using Swivel. Swivel objective (2), compared to GloVe, bears two characteristics essential for our task. First, training to estimate the pointwise mutual information instead of the raw co-occurrence gives us a natural way to punish profiles that are very frequent and, therefore, their occurring in the list of friends do not bear much information. Second, co-occurrence matrices of such size tend to become increasingly sparse. Considering that our estimation of the social graph is based on an incomplete sample of Twitter, Swivel's additional term (3) for the unobserved co-occurrences is particularly important.

After the factorization, the matrix of embeddings $U \in \mathbb{R}^{v_{\mathrm{sg}} \times d_{\mathrm{sg}}}$ including $v_{\mathrm{sg}} = 499,712$ vectors with $d_{\mathrm{sg}} = 300$ is produced. For the other users, an embedding is calculated as a weighted average of the embeddings of their friends:

$$\mathrm{emb}_{\mathrm{sg}}(\mathrm{user}) = \begin{cases} \frac{\sum_{f \in \mathrm{fr}(\mathrm{user})} U_f * \mathrm{sg}_{uf}}{\sum_{f \in \mathrm{fr}(\mathrm{user})} \mathrm{sg}_{uf}}, & \text{if } \mathrm{fr}(\mathrm{user}) \neq \varnothing \\ \overline{u}, & \text{otherwise} \end{cases} \tag{4}$$

where $\mathrm{fr}(\mathrm{user})$ is a list of friends for which the embedding exists and sg_{uf} is the weight for a social graph edge from u to f. The mean embedding $\overline{u_j} = \sum_i u_{ij}$ is used if a target user does not follow anyone we have the embedding for.

4.3 RDF graph embeddings

KBs typically consist of RDF triples, providing a natural way of interpreting entities and relations between them as a graph. The goal of RDF-based embeddings is, therefore, to provide vector representations for each entity in the KB for computing similarities between entities in our task.

Cochez et al. [5] propose a number of approaches for building a co-occurrence matrix based on RDF data. They treat each node in the RDF graph as co-occurring with its neighbors and investigate the usage of Personalized PageRank (PPR) to determine the weights of such co-occurrences. They develop an efficient approximation of PPR based on the Bookmark-Coloring Algorithm and devise 12 weighting schemes to use along with it. After each edge in a graph is weighted, the co-occurrence matrix can be formed. Then the GloVe model is executed following the objective (1) and computing an embedding vector for each entity in the vocabulary (i.e., the RDF KB).

In our experiments, we tested the impact of those weighting schemes on our task using the precomputed embeddings provided by the authors, and we selected the "PageRank

Split" variant that provides optimal performances. We confirm the authors' findings that the weighting scheme significantly affects the performances on a given prediction problem. The provided embeddings $K \in \mathbb{R}^{v_{\text{kb}} \times d_{\text{kb}}}$ include $v_{\text{kb}} = 8,876,676$ vectors with $d_{\text{kb}} = 200$ and cover most of the entities required by our linking task. In cases where the embedding of an entity is not available, the mean embedding $\overline{k_j} = \sum_i k_{ij}$ is used.

5 The Updated Candidate Selection Model

Once embeddings from both the social graph and the RDF graph are available, we inject them into our *candidate selection* model introduced in Section 3.3.

The first aspect to consider concerns the way those embeddings are added to the model. The most trivial way of adding new features into the model is to just concatenate the old feature vector (BASE features) with both the embeddings for the candidate and the entity. However, considering that the BASE feature space contains just 136 dimensions, concatenating it with an additional 500-dimensional vector ($d_{\text{sg}} + d_{\text{kb}}$) would make it harder for the model training to arrive at a stable solution. Moreover, the BASE feature space contains features that were handcrafted for this specific task and can produce good performances on their own. We thus aim to add new features in a way that does not interfere with the network ability to learn from the old features too.

The second aspect to consider is that the candidate selection phase is a binary classification task, where the goal is to learn to match a particular entity with a particular candidate. So, the core motivation to have the graph-based features in the model in the first place is to exploit similarities between the RDF graph and the social graph. However, since both vector spaces of embeddings were trained separately, the model itself has to learn how to combine them in a useful way, which makes the optimization problem unnecessary harder. During our early experiments with the simple concatenation approach, we were unable to produce a consistent enough improvement over the BASE model.

The third and last aspect to consider is that representation learning from unsupervised data is typically used as a pre-training step. Then it is customary for a neural network to modify those representations during back propagation to trade the initial generality for performance in a particular prediction task. Unfortunately, we cannot use the same technique here. We have built our approach to be general enough to produce a SocialLink resource targeting at least 2.5M entities, while our gold standard dataset contains only 56,133. It is unlikely that training on such a gold standard will modify over 9M embeddings in a way that would produce a good general solution for the larger task.

To address those three issues, we started with adding a densely connected layer after each of the embeddings. This

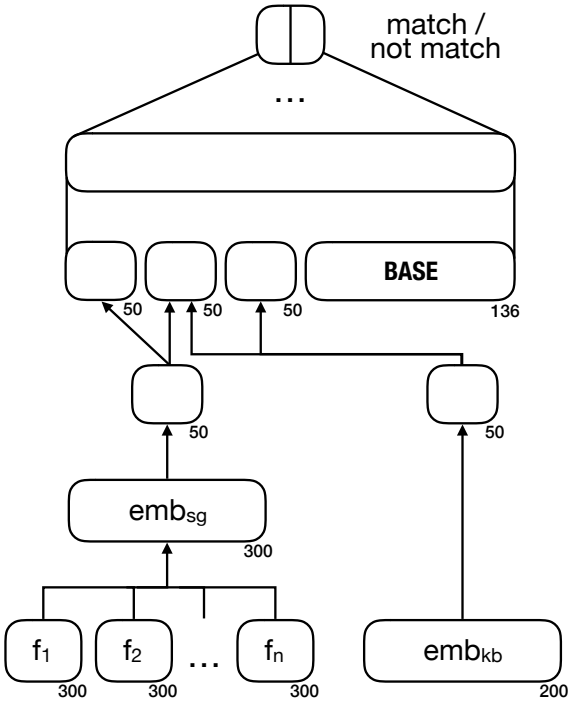


Fig. 2: Schematic view of the updated candidate selection model, showing the new neural network architecture.

transformation layer further reduces the dimensionality of embeddings to just $d_{\text{emb}} = 50$, acting as a global modifier of the entire embedding feature space instead of modifying individual vectors. To encourage the network to use this transformation layer to map the original embeddings into the same feature space, we introduce component-wise combinations of features for the embeddings by adding a *multiplication term*. Then the multiplication term and the transformed embeddings are concatenated with the BASE features, producing a total of 286 features ($d_{\text{emb}} * 3 + 136$), which go through the same densely connected layers as before. By modifying the topology of the network in this way we were able not only to improve consistency during training but also to further improve the results. Parameter d_{emb} was initially chosen so that embeddings terms together would be roughly the same size as BASE feature set. However, we did test larger values with an increment of 25 up to $d_{\text{emb}} = 150$. We did not notice significant performance changes with values above 100 requiring more epochs to converge. The rest of the network follows our previous approach described in Section 3.3. Figure 2 shows the updated network architecture.

Finally, we have changed the way we interpret the probability estimates we receive from this pairwise classification model. After acquiring each probability estimate p_i for candidates $i = 1 \ldots n$, we have to decide whether to align the entity to a candidate i or to abstain, i.e., decide that no candidate matches the entity, a case that we denote with

nil (analogously to *nil* in Entity Linking literature). Lacking a specific estimate for p_{nil}, we empirically set $p_{nil} = 1 - \max_i p_i$ and then re-scale p_i, $i = 1 \dots n$, so that after normalization we have a proper probability distribution satisfying $p_{nil} + \sum_i p_i = 1$ (note that $\sum_i p_i$ may be originally greater than 1 as estimates p_i are produced independently). We then select the candidate $i \in \{1 \dots n\}$ with the largest probability estimate, if greater than a probability threshold chosen to control the precision / recall balance. This estimate is released along the SocialLink dataset and provides an indication of the alignment reliability, permitting users to set a probability threshold to select only the most reliable alignments and thus operate different precision / recall balances.

6 Evaluation

We provide here an experimental evaluation of the improved linking system described in this paper, starting by introducing the experimental setting (Section 6.1) and then reporting on the system performances on the overall linking task (Section 6.2) and, separately, on the two main phases this task is organized in: candidate acquisition (Section 6.3) and candidate selection (Section 6.4). We also analyze the performances on different entity and profile types (Section 6.5) and conduct an error analysis (Section 6.6).

6.1 Experimental Setting

We denote the proposed improved system as BASE_KB_SG_TL, as it integrates base features (BASE) with knowledge base (KB) and social graph (SG) embeddings, combined using a translation layer (TL); we consequently concatenate different subsets of those feature identifiers to denote system variants obtained by ablation of selected features.

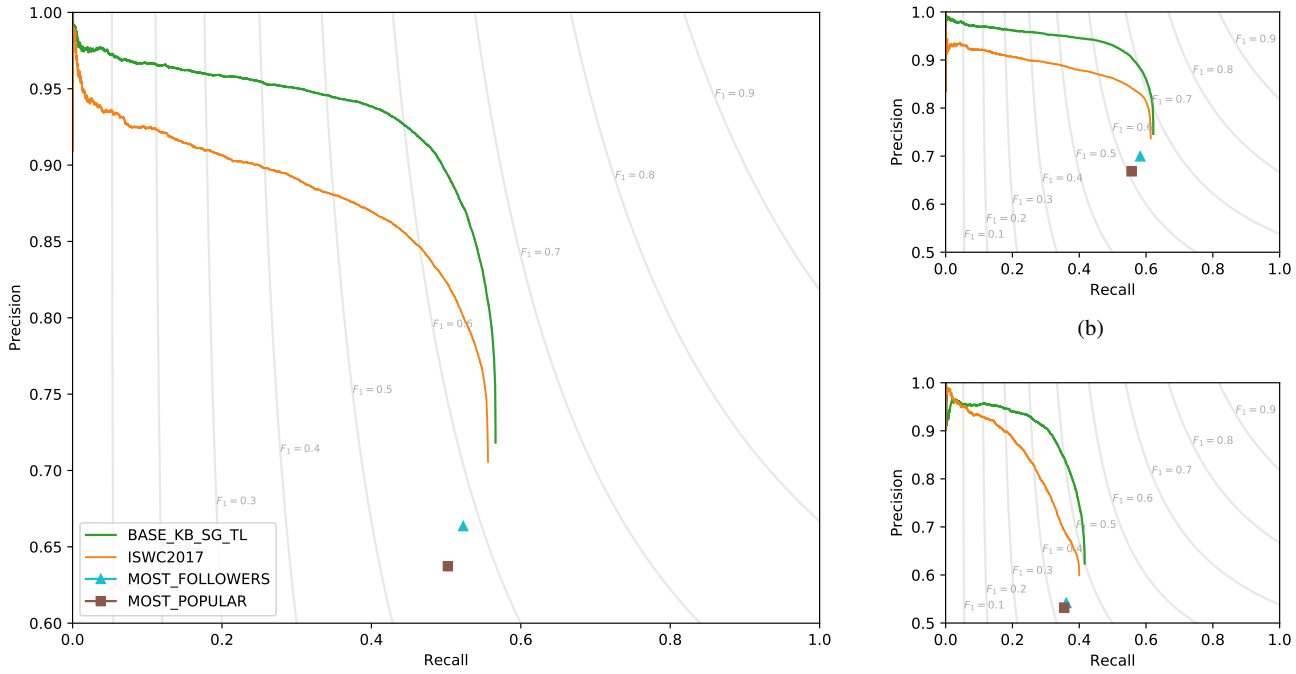
Since there are no publicly available datasets for our task, we evaluate BASE_KB_SG_TL and its variants on the gold standard collected from DBpedia and Wikidata, consisting of 56,133 alignments from English DBpedia entities (40,967 persons, 15,166 organizations) to corresponding Twitter profiles. Of these profiles, 94.69% are in our user index built from the tweet stream and may be successfully aligned by the system. We use stratified 5-fold cross validation, with alignments computed for each test partition being concatenated to form a single set of alignments for the whole gold standard, over which our analyses are performed.

As our main performance measures, we use *precision* (P), *recall* (R), and F_1 *score*. These measures are computed considering as true positive (TP) every alignment produced by the system that matches a gold standard alignment, as false positive (FP) every system alignment not present in the gold standard, and as false negative (FN) every gold standard alignment not found by the system, with $P = \frac{TP}{TP+FP}$, $R = \frac{TP}{TP+FN}$, and $F_1 = \frac{2 \cdot P \cdot R}{P+R}$. We report P, R, and F_1 scores together with their 95% confidence intervals computed via the *percentile bootstrap* method, and assess the statistical significance of the difference of those scores via the paired *approximate randomization* test [27] (significant if $pvalue \leq 0.05$). Since the output of the system is a set of alignments each one associated to a probability estimate, different precision / recall balances and thus P, R, F_1 scores can be obtained by setting a varying threshold on that estimate. This gives rise to a precision / recall curve that we report and analyze to study the performance of the system in different settings that differ by the relative importance given to precision and recall. From that curve, we pick the threshold producing the best F_1 score, and use the corresponding P, R, F_1 triple as the overall assessment of the system performances.

To assess the effectiveness of the improved system described in this paper, we compare it against three baselines:

1. ISWC2017. This baseline employs the linking model we previously proposed in [25] and reused in [26], which as described in Section 3.3 consists in a simpler DNN not leveraging relational information in the form of graph embeddings, and where the probabilities estimated by the DNN are directly used to select the best candidate without any normalization, differently from the new strategy that we described in Section 5.
2. MOST_POPULAR. This baseline implements a straightforward approach where a DBpedia entity is aligned to the most popular Twitter profile matching it. Here, *matching* is defined as having one of the entity names in the profile full name or screen name, and thus being in the candidate list obtained by querying our user index. *Most popular* is defined as being the matching profile "seen the most" (i.e., authoring and/or being mentioned) in the tweets collected from the Twitter stream.
3. MOST_FOLLOWERS. Here we select as a correct alignment the Twitter profile with the most followers among the ones in the user index matched to the entity.

Note that the MOST_POPULAR baseline replaces the one we used in [25] where we queried the Twitter ReST API (instead of the Streaming API) for a target entity name and we selected the first profile returned (if any) as the alignment for the entity. Both the old and the new baselines try to simulate a user's search for the matching profile, with the difference that we now search on our own user index built from Twitter data rather than querying Twitter directly, and we use the popularity notion defined above as a proxy for the ranking of query result provided by Twitter, which empirically appears to produce similar results. The newly added MOST_FOLLOWERS baseline similarly captures the popularity of a candidate profile with an explicit metric: the audience size. Concerning the DNN-based ISWC2017 baseline,



Model	Persons			Organizations			All entities		
	P	R	F1	P	R	F1	P	R	F1
BASE_KB_SG_TL	0.856 ±0.004	0.602 ±0.005	0.707 ±0.004	0.727 ±0.010	0.406 ±0.008	0.521 ±0.008	0.831 ±0.004	0.549 ±0.004	0.661 ±0.004
ISWC2017	0.808 ±0.004*	0.604 ±0.005	0.691 ±0.004*	0.630 ±0.010*	0.397 ±0.008*	0.487 ±0.008*	0.768 ±0.004*	0.548 ±0.004	0.639 ±0.004*
MOST_FOLLOWERS	0.700 ±0.005*	0.583 ±0.005*	0.636 ±0.005*	0.542 ±0.010*	0.361 ±0.008*	0.434 ±0.008*	0.664 ±0.004*	0.523 ±0.004*	0.585 ±0.004*
MOST_POPULAR	0.669 ±0.005*	0.557 ±0.005*	0.608 ±0.005*	0.532 ±0.010*	0.355 ±0.008*	0.426 ±0.008*	0.637 ±0.004*	0.502 ±0.004*	0.562 ±0.004*

(d)

Fig. 3: P/R curves of overall system: (a) all entities; (b) persons; (c) organizations; (d) precision, recall, and F1 scores for the setting maximizing F1, with confidence intervals and statistical significance (*) of difference wrt. best model

in [25] we have also investigated the use of an SVM for the candidate selection phase, obtaining slightly worse F_1 scores when tuning the system for F_1, and worse P scores when tuning the system for precision. We refer the reader to [25] for further details, as well as an analysis of BASE features.

6.2 Overall System Evaluation

Figure 3(a) reports the precision / recall curve of the improved BASE_KB_SG_TL system, compared on the same data to the curve of the ISWC2017 baseline and to the single $\langle P, R, F_1 \rangle$ points of the MOST_POPULAR and MOST_FOLLOWERS baselines. Figures 3(b) and 3(c) provide the same comparison restricted respectively to the alignment of person and organization entities only.

The BASE_KB_SG_TL system outperforms the three baselines for all the P / R balances, except for low recall values for organizations. P and F_1 differences for the system configurations maximizing F_1 are always statistically significant, as shown in Table 3(d), while R differences are not statistically significant when compared to the ISWC2017 baseline.

Both figures and table show that persons are aligned better than organizations. This difference is exhibited also by the considered baselines and is consistent with our previous results [25], suggesting that the considered linking task is inherently more difficult for organization entities.

Both BASE_KB_SG_TL and the baselines exhibit a sensible loss in terms of recall. Part of the loss is explained by the fact that our user index contains only 94.69% of the profiles in the gold standard, which bounds the recall of any approach using the index to 94.69% and thus limits the F_1 score to 97.27%. The additional loss of recall can be explained by separately analyzing the two phases of the task.

6.3 Candidate Acquisition Evaluation

We analyze the internal behavior of the system starting from the candidate acquisition phase. Table 4 provides relevant

Table 4: Candidate acquisition statistics per entity type

Entity type	Has some candidate	Has true candidate	Avg. candidates	True candidate avg. index
Persons	83.3%	65.1%	9.8	1.7
Organizations	66.7%	46.1%	10.5	2.3
All entities	78.8%	60.0%	9.9	1.9

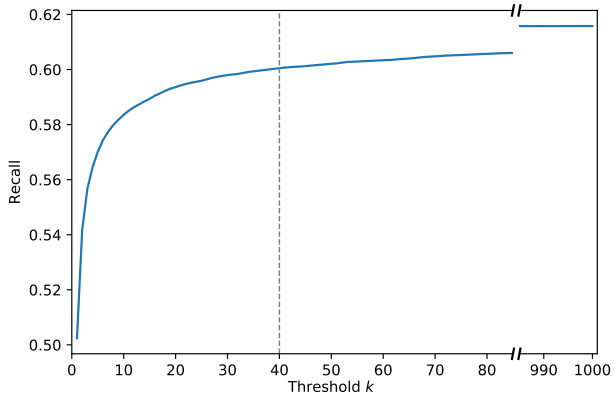


Fig. 4: Candidate acquisition recall, i.e., fraction of entities whose fetched candidate list contains the true candidate

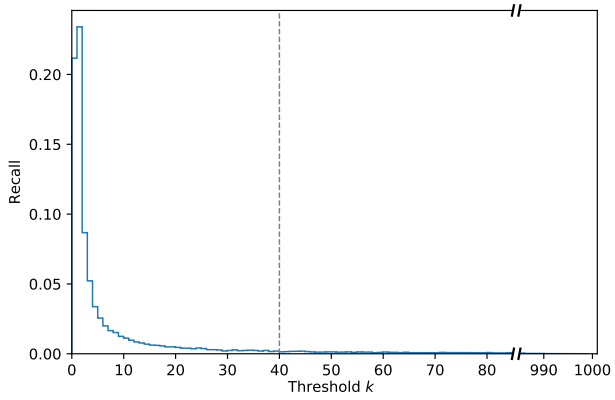


Fig. 5: Frequency distribution of the number of candidates fetched per entity using our candidate acquisition strategy

statistics for this phase computed for all gold standard entities and for persons and organizations only. They include: (i) the percentage of entities for which some candidate is returned by querying the user index; (ii) the percentage of entities for which the returned candidate list contains the true candidate for the entity, i.e., the candidate acquisition *recall*; (iii) the average length of the returned candidate list, when not empty; and (iv) the average position of the true candidate in the candidate list, when not empty.

Similarly to what observed in the overall evaluation, candidate acquisition for person entities exhibits better performances (recall, average true candidate index, non-empty candidate lists). For both kinds of entities and overall, the recall levels are limited, with the achieved recall of 60% (all entities) introducing a further loss of 34% to the 6% loss due to the right profile not being present in the user index. This additional loss reflects the difficulty of matching the entity names in the KB to the names used in the social media, which in some cases may be completely different. Since there is no way for the system to produce an alignment if the true candidate is not in the candidate list, the low recall levels obtained in this phase set hard limits to the recall achievable by the system on the overall task.

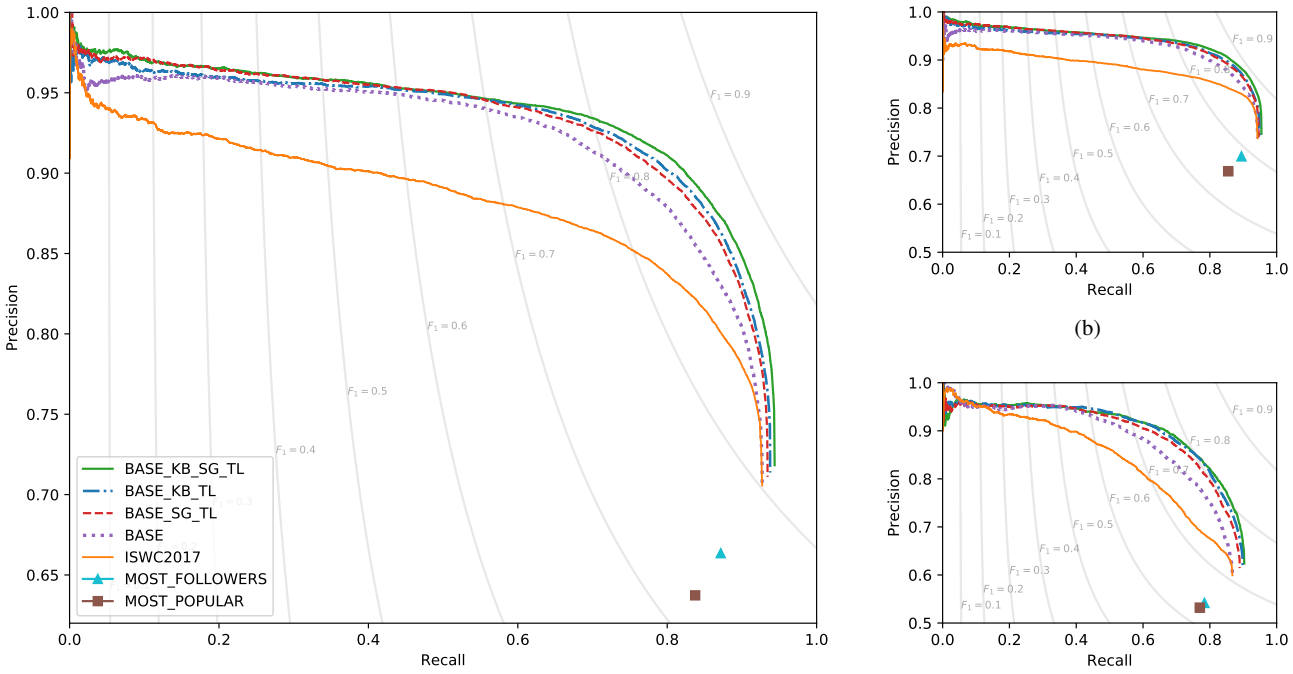
Table 4 is complemented by Figure 4 that shows the recall obtainable by varying the maximum number of candidates k fetched for each entity (see Section 3.2), and Figure 5 that shows the frequency distribution of the number of candidates fetched for an entity; the vertical lines correspond to the chosen threshold $k = 40$. The histogram shows that very few entities in the long tail have more candidates than the threshold $k = 40$. This is reflected in the recall plot of Figure 4, where going from the selected $k = 40$ to an hypothetical $k = 1000$ will bring only a 1.57% increase in recall (from 60.01% to the "plateau" value 61.58%), at the price however of a sensible increase in computation costs and a possible loss of precision in the overall task due to the introduction of noisy candidates in the input to candidate selection. Therefore, $k = 40$ represents a good trade-off solution.

6.4 Candidate Selection Evaluation

We now turn our attention to the candidate selection phase, whose performances are assessed in terms of precision / recall for the only entities for which a non-empty set of candidates is returned by the candidate acquisition phase.

Figure 6(a) shows the precision / recall curves of BASE_KB_SG_TL, the three baselines, and three variants of the proposed system: the BASE_KB_TL and BASE_SG_TL variants, obtained by removing the multiplication term from the DNN and, respectively, the social graph and the KB embeddings, and the BASE variant, obtained by removing both kinds of graph embeddings but keeping the normalization of probabilities and the other fine tunings applied to the DNN with respect to the ISWC2017 baseline. Figures 6(b) and 6(c) provide the same comparison restricted however to person and organization entities only, respectively. Table 6(d) reports P, R, and F_1 scores for the configurations with the best F_1.

As for the overall task, BASE_KB_SG_TL consistently outperforms the three baselines, with P, R, F_1 differences always statistically significant except overall recall with respect to the ISWC2017 baseline. Notably, recall is high in this phase, implying that the loss of recall in the overall task is largely due to the candidate acquisition phase. Also in this phase, performances are marginally better for persons.



Model	Persons			Organizations			All entities		
	P	R	F1	P	R	F1	P	R	F1
BASE_KB_SG_TL	0.873 ±0.004	0.908 ±0.004	0.890 ±0.003	0.802 ±0.009	0.822 ±0.009	0.812 ±0.008	0.856 ±0.004	0.892 ±0.003	0.874 ±0.003
BASE_KB_TL	0.866 ±0.004*	0.901 ±0.004*	0.883 ±0.003*	0.818 ±0.009*	0.797 ±0.010*	0.808 ±0.008	0.860 ±0.004*	0.876 ±0.004*	0.868 ±0.003*
BASE_SG_TL	0.863 ±0.004*	0.900 ±0.004*	0.881 ±0.003*	0.798 ±0.010	0.799 ±0.010*	0.798 ±0.008*	0.855 ±0.004	0.874 ±0.004*	0.864 ±0.003*
BASE	0.843 ±0.004*	0.901 ±0.004*	0.871 ±0.003*	0.783 ±0.010*	0.769 ±0.010*	0.776 ±0.008*	0.820 ±0.004*	0.885 ±0.003*	0.851 ±0.003*
ISWC2017	0.820 ±0.004*	0.914 ±0.003*	0.865 ±0.003*	0.671 ±0.010*	0.812 ±0.009*	0.735 ±0.009*	0.789 ±0.004*	0.890 ±0.003	0.836 ±0.003*
MOST_FOLLOWERS	0.700 ±0.005*	0.895 ±0.004*	0.785 ±0.004*	0.542 ±0.010*	0.784 ±0.010*	0.641 ±0.009*	0.664 ±0.004*	0.872 ±0.004*	0.753 ±0.004*
MOST_POPULAR	0.669 ±0.005*	0.855 ±0.004*	0.750 ±0.004*	0.532 ±0.010*	0.770 ±0.010*	0.629 ±0.010*	0.637 ±0.005*	0.837 ±0.004*	0.724 ±0.004*

(d)

Fig. 6: P/R curves of candidate selection phase: (a) all entities; (b) persons; (c) organizations; (d) precision, recall, and F1 scores for the setting maximizing F1, with confidence intervals and statistical significance (*) of difference wrt. best model

When comparing the full BASE_KB_SG_TL system to its BASE_KB_TL, BASE_SG_TL, and BASE variants obtained via ablation of graph-based features, both figures and table show that each ablated feature contributes positively to the performances of the system, with the best performances achieved by combining both types of graph embedding features. In particular, Table 6(d) shows that the F_1 improvements for all entities obtained by adding features are always statistically significant. Finally, the positive impact of the new probability normalization and DNN fine tuning is demonstrated by the difference in performances between the BASE system variant and the ISWC2017 baseline.

In our experiments and as mentioned in Sections 1 and 5, the transformation layer turns out to be crucial for successfully training the network. Without it — i.e., with the system variant BASE_KB_SG that simply concatenates the BASE features with the RDF graph (KB) and social graph (SG) embeddings — good models ($P = 0.845 \pm 0.006$, $R = 0.883 \pm 0.006$, $F_1 = 0.863 \pm 0.005$, for all entities) can be trained only for two cross-validation folds out of five, while the models obtained for the other folds are significantly underperforming ($P = 0.744 \pm 0.006$, $R = 0.878 \pm 0.005$, $F_1 = 0.805 \pm 0.005$), bringing down the average performances of BASE_KB_SG over all folds ($P = 0.746 \pm 0.004$, $R = 0.900 \pm 0.003$, $F_1 = 0.815 \pm 0.003$). Even restricting to the two good folds, the performances of BASE_KB_SG there are significantly lower than the ones of BASE_KB_SG_TL, making not worthwhile any attempt at fixing the training issues with the simple concatenation model BASE_KB_SG.

6.5 Evaluation by Entity and Profile Type

To investigate for which types of entities and profiles the approach performs best, in Table 5 we report the performances

Table 5: Performances of candidate acquisition, candidate selection (best F_1 setting of BASE_KB_SG_TL), and joint task for subsets of the gold standard, with confidence intervals and statistical significance (*) of differences wrt. whole population

Gold standard subset	# Samples	Candidate acquisition recall	Candidate selection			Joint task		
			P	R	F1	P	R	F1
dbo:Person	41003	0.652 ±0.005*	0.865 ±0.004*	0.917 ±0.003*	0.890 ±0.003*	0.845 ±0.004*	0.608 ±0.005*	0.707 ±0.004*
dbo:Artist	17764	0.617 ±0.007*	0.861 ±0.006	0.919 ±0.005*	0.889 ±0.005*	0.836 ±0.007	0.578 ±0.007*	0.683 ±0.007*
dbo:Athlete	10207	0.632 ±0.010*	0.854 ±0.008	0.936 ±0.006*	0.893 ±0.006*	0.838 ±0.008	0.600 ±0.010*	0.699 ±0.009*
dbo:OfficeHolder	2911	0.725 ±0.016*	0.884 ±0.013*	0.913 ±0.012*	0.898 ±0.011*	0.874 ±0.014*	0.672 ±0.017*	0.760 ±0.015*
dbo:Politician	3628	0.714 ±0.015*	0.904 ±0.011*	0.946 ±0.009*	0.925 ±0.009*	0.895 ±0.012*	0.681 ±0.015*	0.773 ±0.013*
dbo:Writer	1176	0.702 ±0.026*	0.901 ±0.021*	0.892 ±0.022	0.896 ±0.018*	0.879 ±0.023*	0.641 ±0.027*	0.741 ±0.024*
dbo:Organisation	15175	0.461 ±0.008*	0.821 ±0.009*	0.800 ±0.010*	0.810 ±0.008*	0.775 ±0.009*	0.391 ±0.008*	0.520 ±0.009*
dbo:Broadcaster	1054	0.493 ±0.030*	0.778 ±0.040*	0.633 ±0.041*	0.698 ±0.036*	0.702 ±0.040*	0.337 ±0.029*	0.455 ±0.032*
dbo:Company	3363	0.543 ±0.017*	0.837 ±0.018*	0.752 ±0.020*	0.792 ±0.016*	0.790 ±0.018*	0.443 ±0.017*	0.568 ±0.017*
dbo:EducationalInstitution	1373	0.239 ±0.023*	0.881 ±0.038	0.744 ±0.048*	0.807 ±0.038*	0.851 ±0.040	0.195 ±0.021*	0.318 ±0.029*
dbo:Group	4705	0.529 ±0.014*	0.819 ±0.014*	0.903 ±0.012*	0.859 ±0.011*	0.769 ±0.015*	0.489 ±0.014*	0.598 ±0.014*
dbo:SportsTeam	2313	0.513 ±0.020*	0.802 ±0.022*	0.806 ±0.023*	0.804 ±0.019*	0.775 ±0.023*	0.440 ±0.021*	0.561 ±0.022*
more followers	26358	0.721 ±0.005*	0.918 ±0.004*	0.918 ±0.004*	0.918 ±0.003*	0.898 ±0.004*	0.675 ±0.006*	0.770 ±0.005*
less followers	26357	0.558 ±0.006*	0.832 ±0.006*	0.859 ±0.006*	0.846 ±0.005*	0.805 ±0.006*	0.496 ±0.006*	0.614 ±0.006*
more popular	26367	0.718 ±0.005*	0.919 ±0.004*	0.913 ±0.004*	0.916 ±0.003*	0.899 ±0.004*	0.668 ±0.006*	0.767 ±0.005*
less popular	26348	0.561 ±0.006*	0.832 ±0.006*	0.866 ±0.006*	0.849 ±0.005*	0.805 ±0.006*	0.502 ±0.006*	0.619 ±0.006*
verified	26194	0.755 ±0.005*	0.932 ±0.004*	0.939 ±0.003*	0.936 ±0.003*	0.915 ±0.004*	0.719 ±0.005*	0.805 ±0.004*
not verified	26521	0.526 ±0.006*	0.807 ±0.006*	0.826 ±0.006*	0.816 ±0.005*	0.778 ±0.007*	0.453 ±0.006*	0.573 ±0.006*

of candidate acquisition (recall), candidate selection (P, R, F_1), and joint task (P, R, F_1) on different subsets of the gold standard. As candidate selection model, we use the best performing model BASE_KB_SG_TL with the abstention score threshold that maximizes F_1 on the whole gold standard. As subsets of the gold standard, we consider:

- person and organization subclasses in DBpedia that have at least 1000 samples in the gold standard (to provide more accurate performance estimates);
- entities whose profile has more vs. less followers on Twitter, using the median number of followers measured on the gold standard as separation value;
- entities whose profile is more vs. less popular in Twitter, with popularity defined as the number of times we observed the profile in the tweet stream and using the median number of occurrences measured on the gold standard as separation value;
- entities with a verified vs. not verified profile.

The table also reports the size of each subset, 95% confidence intervals, and statistical significance of score differences with respect to the whole gold standard population.[13]

In terms of person and organization subclasses, Table 5 shows that each subset exhibits performances that differ from the average population, with differences always statistically significant for candidate acquisition and F_1 of candidate selection and joint task. Among persons, candidate selection performances are similar while lower candidate acquisition recall is observed for artists and athletes (more numerous), which maps to higher recall and thus F_1 in the joint task. Among organizations, candidate selection performs better for musical groups and worse for broadcasters, while educational institutions show a significantly low candidate acquisition recall that maps to very low joint recall and F_1 scores.

In terms of followers, popularity, and verified status, Table 5 shows that entities whose profiles have these characteristics are linked significantly better by our approach. Apart being directly used as candidate selection features, these characteristics generally imply less ambiguity as well as the availability of more abundant and accurate data on the social media side, which ease the linking task.

[13] We compare the performances of the subset with the ones of its complement using the non-paired approximate randomization test.

6.6 Error Analysis

Here we analyze the types of errors our joint pipeline commits, providing examples and estimating their impact so to inform possible extensions of the approach. We consider as error any answer by the system that differs from the one provided by the gold standard, including system abstention (since the alignment exists). On the whole gold standard, our pipeline with the best performing selection model BASE_KB_SG_TL (best F_1 setting) makes a total of $24,477$ errors, originating from all three constituent processing phases. In each of those phases we identify the types of errors, the detailed breakdown of which is provided in Table 6.

Firstly, the user index built during the data acquisition phase does not contain the entire population of Twitter. While we can be all but sure that popular entities will end up being in our index, passive users (i.e., the ones that do not tweet and retweet) will never be observed in the stream of tweets and will never be linked to an entity. For example,

Table 6: Error breakdown for the joint task using the best performing model (BASE_KB_SG_TL) as evaluated on a gold standard. Abstention counts as an error.

Error type	Share	Total	# per.	# org.
ALL	100.00%	24,477	15,566	8,911
STREAM	9.54%	2,334	1,499	835
CA_INDEX	78.57%	19,233	12,316	6,917
CA_CUTOFF	3.60%	882	473	409
CS_ABSTAIN	2.68%	656	349	307
CS_DISAMBIG	5.61%	1,372	929	443

at the time of writing, the correct profile for the American music band "The Spares" (dbr:The_Spares), @thespares, has never tweeted or interacted in any way with the rest of the network, therefore will never appear in our user index. Moreover, even when searched via Twitter itself, the correct profile is not returned as one of the options. This type of error, which we identify with STREAM in Table 6, is responsible for 9.54% of all errors.

Secondly, two types of errors, CA_INDEX and CA_CUTOFF, originate from the candidate acquisition (CA) phase. CA_INDEX errors amounts to 78.57% of all errors and occur if the profile, while being in the user index, cannot be retrieved by our full-text search candidate acquisition approach. This happens if there is a mismatch between the name contained in DBpedia (which we use for matching) and the name of the Twitter profile, or if the DBpedia name is too ambiguous (i.e., entity known with a popular name, such as John) which will timeout the query and force our candidate acquisition approach to rephrase it. An example of this error occurs with basketball player Walter Tavares (dbr:Walter_Tavares), who is also known as Edy Tavares, which is the name used in his official Twitter profile. This name is not in the English DBpedia and, therefore, his profile would not be returned as a candidate. In this specific case, however, the recall loss can be mitigated by importing names from other DBpedia language chapters or from Wikidata. Concerning CA_CUTOFF errors, they amount to 3.60% of all errors and occur when the correct profile is obtained from the index, but its position in the candidate list is past the cut-off $k = 40$ and it is thus discarded as part of our attempt to reduce potential ambiguity and computational costs.

In third, in case the correct alignment is within the list of candidates, the candidate selection model can commit a CS_ABSTAIN error by wrongly abstaining or commit a CS_DISAMBIG error by selecting the wrong candidate. CS_ABSTAIN errors happen in 2.68% of cases and are caused by the right alignment being rejected as having a score lower than the minimum score threshold (set to 0.169 to maximize F_1 on BASE_KB_SG_TL). For example, for the politician and radio host Jason Lewis (dbr:Jason_Lewis_(radio_host)), SocialLink correctly identified his two official Twitter accounts, @Jason2CD and @RepJasonLewis, with the DNN returning very high probabilities, 0.952 and 0.949 respectively. However, since the name has a high degree of ambiguity — there are other American politicians with the same name, not to mention an actor and a comedian, all of which has social media presence — the rescaled scores (see Section 5) for those two candidates decrease to 0.132 and 0.131, causing the model to abstain in this case.

CS_DISAMBIG errors represent 5.61% of errors and most frequently result from linking to (i) the profile of a related / topically similar entity, or (ii) an alternative profile of the target entity. An example of the first kind is Santiago Segura (dbr:Santiago_Segura), an actor and film director, which is aligned to the profile of a namesake who is also an actor. As often the case in such examples, the confidence scores for their respective profiles are very similar (0.952 vs. 0.953), and here, without using image data, even for a human it is hard to select a correct alignment. An example of the second kind is U.C. Sampdoria (dbr:U.C._Sampdoria), a football club, which is linked in the gold standard to its official Italian Twitter account @sampdoria. SocialLink, on the other hand, links to @sampdoria_en, which is the official English account of the same team. In this case, the DNN emits very similar probabilities: 0.712 and 0.728 respectively, which are both above the threshold after rescaling. The third ranked candidate alignment for this entity, @UCSampdoriaFeed, has a significantly lower score of 0.473 and is a fan account. Organizations, in particular, have a strong tendency towards having multiple social media profiles targeted at different audiences based on language, market or a group of products. Sometimes it is unclear what the main profile is (if unique), and linking to one of the alternative profiles may not necessarily be a mistake and may justify switching to a 1-to-N problem (and gold standard) where an entity may have multiple corresponding profiles.

As can be seen, due to the way the candidate acquisition approach is designed, SocialLink rarely confuses entities with different names but has a number of problematic cases especially when it has to choose between profiles within the same domain or associated to the same target entity. Additionally, when evaluated on the gold standard, SocialLink generally favors precision over recall: the abstention mechanism, designed for the open world case in which we cannot know in advance if the entity has a profile on Twitter, tends to abstain incurring in a false negative error rather than risking to select the wrong candidate and incur in both a false negative and a false positive errors.

7 Resource

The result of running the SocialLink pipeline is the SocialLink dataset, a public LOD dataset that provides alignments

for ∼270K entities and that can be exploited in different applications at the intersection of Semantic Web and Social Media. We provide some brief information about the dataset and its applications, referring the reader to SocialLink website for up-to-date information and statistics on the resource.

SocialLink Dataset The SocialLink dataset is represented in RDF as exemplified in Figure 7. DBpedia entities are identified using all their URIs, forming an owl:sameAs cluster rooted at a canonical URI possibly taken from the language-independent Wikidata, like wikidata:Q76 for person entity Barack Obama in the figure. Twitter profiles, like twitter:BarackObama in the figure, are modeled as foaf:OnlineAccount individuals, using properties foaf:accountName and dct:identifier to encode screen name and numeric identifier. The alignment between an entity canonical URI and the corresponding Twitter account is expressed using property foaf:account, and reified via sl:Candidate individuals (e.g., slr:Q76_BarackObama) that are enriched with properties sl:confidence and sl:rank to encode the candidate estimated probability and its rank among the entity candidates, to simplify querying.

The SocialLink dataset is indexed on DataHub[14] and Zenodo, is updated periodically to account for new data in Twitter and DBpedia, and is publicly available for download on the SocialLink website, together with the gold standard, the schema documentation, and source code and binaries of the SocialLink pipeline.[15] A SPARQL endpoint[16] is also available for use by end users and applications.

SocialLink Applications SocialLink establishes a link among DBpedia and Twitter, enabling transferring knowledge from one resource to another. Two examples of use cases benefiting from SocialLink, which we have explored, are *user profiling* and *entity linking* on Twitter.

User profiling aims at predicting user attributes and is a popular task in social media research. Most approaches are based on supervised learning techniques that require significant amounts of training data. SocialLink directly helps by providing accurate machine-readable descriptions for hundreds of thousands of Twitter profiles aligned to DBpedia, whose DBpedia attributes can now be modeled without relying on expensive manual annotation. Indirectly, SocialLink also allows injecting more features to existing classifiers, like providing DBpedia categories for the aligned followees of a Twitter user, to help inferring his / her interests [2].

Entity Linking (EL) aims at linking named entity mentions in a text to corresponding entities in a KB such as DBpedia, and is particularly challenging for short and noisy texts like Twitter posts. In this context, SocialLink directly helps by providing disambiguation for (aligned) `@username` mentions occurring in tweets. Through these mentions, SocialLink also indirectly supports injecting additional contextual information from DBpedia into tweets, which can then be leveraged for disambiguating other named entities occurring in the text. SocialLink was used in this capacity in the NEEL-IT 2016 challenge on EL for Italian tweets [7,23].

[14] `http://datahub.io/dataset/sociallink`
[15] `http://github.com/Remper/sociallink`
[16] `http://sociallink.futuro.media/sparql`

8 Related Work

Recently, some researchers have tried to link social media accounts to Wikipedia categories as a way of inferring user attributes, for example, interests. Faralli et al. [11] presents an approach that links Twitter profiles to Wikipedia pages using BabelNet and Babelfy and then to top 22 Wikipedia categories to determine a target user's or community's category. Besel et al. [2] introduced a similar interest inference pipeline using MediaWiki Web API and used a spreading activation technique on the Wikipedia Bitaxonomy to acquire interests. Piao and Breslin [31] iterated on this idea improving various steps of the pipeline. Nechaev et al. [24] used an earlier version of SocialLink to perform the linking and then propose a list of users to follow to conceal target user's interests as a way to defend against the inference approaches listed above. In these pipelines, the performance of the linking approach is not as important. The recall is mostly irrelevant in this case, and incorrect disambiguation of a Wikipedia page does not matter if it acquires correct or similar category in the end. In our case, we aim to disambiguate as best as we can for any given recall level, which makes SocialLink suitable for a wider array of tasks.

To the best of our knowledge, no one has tried to link KB entries to social media profiles. However, this task is closely related to the profile matching (or profile aligning) problem, whose goal is to align profiles of the same person in different social media. A lot of research has been done for this task, exploiting various attributes in profiles [28,15,22], user-generated content [28,21,14], and social graphs [22].

Some researchers have pointed out that most of the attributes that could theoretically be exposed in social media are unreliable for profile matching [15]. Attributes might not exist, might contain information of varying granularity, or they might even be false. Attempts were therefore made to use as little information as possible to align profiles, choosing only the most reliable attributes. In studies by Zafarani et al. [38,37] only the username (which is the unique identifier that exists in virtually any social media) was exploited to align profiles. The authors showed that people tend to be very consistent when choosing their usernames, which enables identification even if the rest of the profile is filled with incorrect information. Even though they proved that username is a powerful feature, KBs typically do not contain examples of usernames, which makes this feature unavailable for our task.

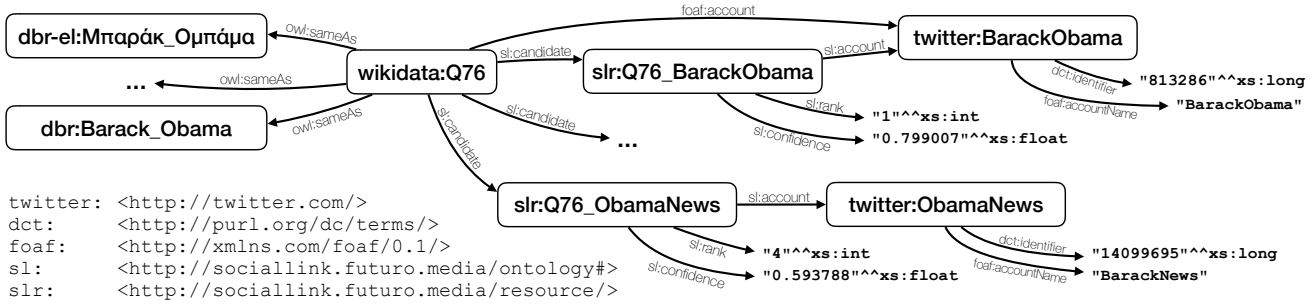


Fig. 7: RDF representation of alignments in the SocialLink LOD dataset.

Goga et al. [15] explored the reliability of attributes in various social media. According to their study, only few attributes like username and real name are available in social media reliably. For example, they reported that location is present in 54% of Twitter profiles and is not consistent across multiple social media. They exposed various methodological and technical challenges in this area related to the construction of ground truth datasets, attribute discriminability and impersonability. Goga's PhD thesis [13] provides a more in-depth look into those issues.

Liu et al. [21] reported the largest experiment on profile matching to date using a dataset of 10 millions profiles across 7 social media. Their approach leverages a wide variety of hand-crafted features based on textual and image user-generated content, and demonstrates its importance for profile matching. Note that user-generated content is usually missing in KBs, so it cannot be used in our task as it is used in the profile alignment task.

Goga et al. [14] showed that profiles can be matched robustly even if explicit attributes are hidden or intentionally falsified. Their approach uses only implicit features present in social media but generally unavailable in KBs, such as writing style, messaging behavior, and location metadata.

Social graph has proven to be hard to acquire in social media. It could be unavailable for crawling or there could be very strict restrictions on API (most notably, in Twitter). Lu et al. [22] were able to gather a small social graph dataset and proved that it can be effectively matched to improve the results of profile alignment. Entities in KBs often contain links to other entities which can be interpreted as a kind of social graph.

Finally, Peled et al. [28] gave an overview of the profile alignment task and presented their own approach that uses all available information in the profile to perform matching. They presented three main use cases for their system, one of which — searching for a user by similar name — is close to the candidate acquisition part of our system.

To summarize, even though some researchers expressed concerns [15, 13] regarding the use of some attributes, every piece of profile data contributes towards identifying the user.

In this work, we leverage and extend graph embeddings (or network embeddings), a particular type of embeddings that is an intensely researched topic in recent literature [16]. GloVe (Global Vectors for Word Representation) [29] is a method for training word embeddings based on word-word co-occurrence statistics from a corpus. Swivel (Submatrix-wise Vector Embedding Learner) [36] extends GloVe by introducing a soft hinge loss with special handling for unobserved co-occurrences that allows producing more stable estimates for rare words. Alternatively, in node2vec [17] and DeepWalk [30], random walks were employed to acquire sequences of nodes, which are then used to maximize the likelihood of preserving network neighborhoods. Both Node2vec and DeepWalk are based on word2vec neural models and do not compute explicit global co-occurrence statistics. This approach was adopted in RDF2Vec [32, 34] producing graph embeddings specialized for the RDF graphs found in KBs. Subsequent work [5], used in this paper, have investigated the usage of GloVe to produce embeddings for RDF graphs. While this approach did not exhibit performance improvement over RDF2Vec in most tasks, it is more efficient, allowing to incorporate larger portions of the graph without significant performance penalties.

9 Conclusions and Future Work

In this paper, we presented the latest version of SocialLink, our supervised approach and associated resource for automatically linking DBpedia entities to corresponding Twitter profiles. Building on our prior works [25, 26], we contributed a new linking approach that allows leveraging graph-based features both on the social media and KB sides, encoding them in the form of embeddings, i.e., low-dimensional representations of objects trained from massive amounts of unlabeled data, that we gather from both the DBpedia RDF graph and the Twitter social graph.

For the social graph embeddings, we devised a custom procedure to estimate the social graph from the publicly observable stream of tweets, followed by the application of the co-occurrence matrix factorization method called Swivel. We

showed that it is possible to acquire an approximation of the social graph indirectly and calculate embeddings at scale.

For the RDF graph embeddings, we used the precomputed embeddings provided by Cochez et al. [5]. Testing those embeddings on our task, we confirmed the findings of their authors about the significance and impact on performances of the weighting schema adopted in their production when applying such representations to a particular problem.

Additionally, we found that the combination of handcrafted features and pre-trained embeddings in a prediction task like ours requires significant redesign of the learning approach. By changing the topology of our neural network, we were able to achieve a stable performance improvement from the adoption of the new features.

Finally, we evaluated in detail the effect that graph-based features produce on our approach. We observed significant improvements when applying new features separately, which were still dominated by the complete model. In addition, we provided an evaluation of the candidate acquisition approach that was introduced in our resource paper [26].

In the future, we will continue to gradually update SocialLink by both improving the approach and expanding the scope to accommodate a larger subset of LOD entities. A significant goal in our current roadmap consists in the expansion of our approach to other social networks, such as Facebook and Instagram, by generalizing the approach used for Twitter and making SocialLink a resource truly connecting the social media world to the LOD cloud. By introducing more social media to SocialLink, we will be able not only to improve coverage but also to exploit cross-network information to validate our alignments, overall improving their precision. From the approach side, a number of possible improvements could be proposed. The most promising future direction would be to address the recall drop observed during the candidate acquisition phase, by redesigning it using machine learning-based techniques. Another interesting direction consists in learning joint embeddings for both social media profiles and knowledge base entities, to place them into the same vector space. Finally, our pairwise candidate selection solution could be reformulated to account for all candidate profiles at once, for example, via learning to rank instead of binary classification.

References

1. Aprosio, A.P., Giuliano, C., Lavelli, A.: Automatic expansion of DBpedia exploiting Wikipedia cross-language information. In: The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings, *Lecture Notes in Computer Science*, vol. 7882, pp. 397–411. Springer (2013). DOI 10.1007/978-3-642-38288-8_27
2. Besel, C., Schlötterer, J., Granitzer, M.: Inferring semantic interest profiles from Twitter followees: Does Twitter know better than your friends? In: ACM SAC, pp. 1152–1157 (2016)
3. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
4. Cochez, M., Ristoski, P., Ponzetto, S.P., Paulheim, H.: Biased graph walks for RDF graph embeddings. In: Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics, WIMS 2017, pp. 21:1–21:12 (2017)
5. Cochez, M., Ristoski, P., Ponzetto, S.P., Paulheim, H.: Global RDF vector space embeddings. In: The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I, *Lecture Notes in Computer Science*, vol. 10587, pp. 190–207. Springer (2017). DOI 10.1007/978-3-319-68288-4_12
6. Corcoglioniti, F., Giuliano, C., Nechaev, Y., Zanoli, R.: Pokedem: An automatic social media management application. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17, pp. 358–359. ACM, New York, NY, USA (2017). DOI 10.1145/3109859.3109980
7. Corcoglioniti, F., Palmero Aprosio, A., Nechaev, Y., Giuliano, C.: MicroNeel: Combining NLP tools to perform named entity detection and linking on microposts. In: EVALITA (2016)
8. Corcoglioniti, F., Rospocher, M., Mostarda, M., Amadori, M.: Processing billions of RDF triples on a single machine using streaming and sorting. In: ACM SAC, pp. 368–375 (2015)
9. Cristianini, N., Shawe-Taylor, J., Lodhi, H.: Latent semantic kernels. Journal of Intelligent Information Systems **18**(2), 127–152 (2002). DOI 10.1023/A:1013625426931
10. Erxleben, F., Günther, M., Krötzsch, M., Mendez, J., Vrandeăić, D.: Introducing wikidata to the linked data web. In: Proceedings of the 13th International Semantic Web Conference - Part I, ISWC '14, pp. 50–65. Springer-Verlag New York, Inc., New York, NY, USA (2014). DOI 10.1007/978-3-319-11964-9_4
11. Faralli, S., Stilo, G., Velardi, P.: Large scale homophily analysis in twitter using a twixonomy. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, pp. 2334–2340 (2015)
12. Fetahu, B., Anand, A., Anand, A.: How much is Wikipedia lagging behind news? In: Proceedings of the ACM Web Science Conference, WebSci '15, pp. 28:1–28:9. ACM, New York, NY, USA (2015). DOI 10.1145/2786451.2786460
13. Goga, O.: Matching user accounts across online social networks: methods and applications. Ph.D. thesis, LIP6-Laboratoire d'Informatique de Paris 6 (2014)
14. Goga, O., Lei, H., Parthasarathi, S.H.K., Friedland, G., Sommer, R., Teixeira, R.: Exploiting innocuous activity for correlating users across sites. In: Proc. of WWW, pp. 447–458. ACM (2013)
15. Goga, O., Loiseau, P., Sommer, R., Teixeira, R., Gummadi, K.P.: On the reliability of profile matching across large online social networks. In: Proc. of KDD, pp. 1799–1808. ACM (2015)
16. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. arXiv preprint arXiv:1705.02801 (2017)
17. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: The 22th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 855–864. ACM (2016)
18. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: YAGO2: A spatially and temporally enhanced knowledge base from wikipedia. Artificial Intelligence **194**, 28–61 (2013). DOI 10.1016/j.artint.2012.06.001
19. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse processes **25**(2-3), 259–284 (1998)
20. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web **6**(2), 167–195 (2015). DOI 10.3233/SW-140134

21. Liu, S., Wang, S., Zhu, F., Zhang, J., Krishnan, R.: HYDRA: Large-scale social identity linkage via heterogeneous behavior modeling. In: Proc. of SIGMOD, pp. 51–62. ACM (2014)
22. Lu, C.T., Shuai, H.H., Yu, P.S.: Identifying your customers in social networks. In: Proc. of CIKM, pp. 391–400. ACM (2014)
23. Minard, A., Qwaider, M.R.H., Magnini, B.: FBK-NLP at NEEL-IT: Active learning for domain adaptation. In: EVALITA (2016)
24. Nechaev, Y., Corcoglioniti, F., Giuliano, C.: Concealing interests of passive users in social media. In: Proceedings of the Re-coding Black Mirror 2017 Workshop co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 22, 2017. (2017)
25. Nechaev, Y., Corcoglioniti, F., Giuliano, C.: Linking knowledge bases to social media profiles. In: ACM SAC, pp. 145–150 (2017)
26. Nechaev, Y., Corcoglioniti, F., Giuliano, C.: Sociallink: Linking dbpedia entities to corresponding twitter accounts. In: The Semantic Web – ISWC 2017, pp. 165–174. Springer International Publishing (2017). DOI 10.1007/978-3-319-68204-4_17
27. Noreen, E.W.: Computer-intensive methods for testing hypotheses. Wiley New York (1989)
28. Peled, O., Fire, Rokach, Elovici: Matching entities across online social networks. Neurocomputing (2016)
29. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
30. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, pp. 701–710 (2014)
31. Piao, G., Breslin, J.G.: Inferring user interests for passive users on twitter by leveraging followee biographies. In: Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2017, pp. 122–133 (2017)
32. Ristoski, P., Paulheim, H.: Rdf2vec: Rdf graph embeddings for data mining. In: International Semantic Web Conference, pp. 498–514. Springer (2016)
33. Ristoski, P., Paulheim, H.: Semantic Web in data mining and knowledge discovery: A comprehensive survey. Web Semantics: Science, Services and Agents on the World Wide Web **36**, 1 – 22 (2016). DOI https://doi.org/10.1016/j.websem.2016.01.001
34. Ristoski, P., Rosati, J., Di Noia, T., De Leone, R., Paulheim, H.: Rdf2vec: Rdf graph embeddings and their applications. Semantic Web Journal (2017)
35. Sadilek, A., Kautz, H., Bigham, J.P.: Finding your friends and following them to where you are. In: Proc. of 5th ACM Int. Conf. on Web Search and Data Mining (WSDM), pp. 723–732. ACM, New York, NY, USA (2012). DOI 10.1145/2124295.2124380
36. Shazeer, N., Doherty, R., Evans, C., Waterson, C.: Swivel: Improving embeddings by noticing what's missing. CoRR **abs/1602.02215** (2016)
37. Zafarani, R., Liu, H.: Connecting corresponding identities across communities. In: Proc. of ICWSM. AAAI Press (2009)
38. Zafarani, R., Liu, H.: Connecting users across social media sites: A behavioral-modeling approach. In: Proc. of KDD, pp. 41–49. ACM (2013)
39. Zheleva, E., Getoor, L.: 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), pp. 531–540. ACM, New York, NY, USA (2009). DOI 10.1145/1526709.1526781

Table 7: Precision, recall, F1 scores for the setting maximizing F1 with confidence intervals and statistical significance (*) wrt. ALL model

Model	P	R	F1
ALL	0.854 ±0.004	0.880 ±0.003	0.867 ±0.003
LSA	0.842 ±0.004*	0.884 ±0.003*	0.862 ±0.003*
GloVe	0.849 ±0.004*	0.870 ±0.004*	0.859 ±0.003*
fastText	0.842 ±0.004*	0.877 ±0.004*	0.859 ±0.003*

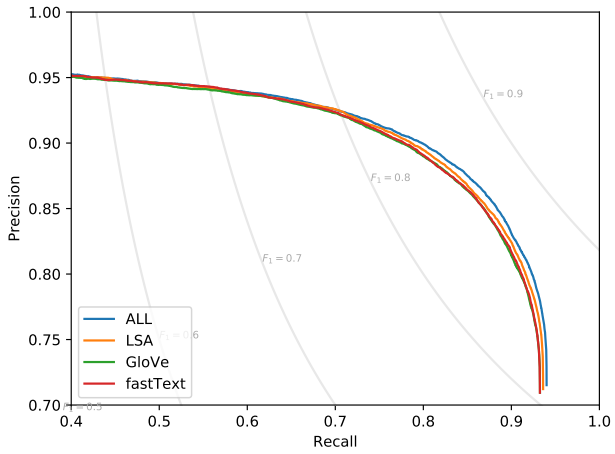


Fig. 8: P/R curves of four models using the BASE_KB_SG_TL model

A On the choice of word embeddings

Pre-trained word representations, or embeddings, have become a staple technique for modeling textual data in a convenient low-dimensional form. Such word representations are typically allocated in the resulting vector space according to the distributional semantics hypothesis, i.e., the words that appear in similar contexts tend to have similar meanings and will be placed close to each other. Pre-trained word representations allow the usage of large unsupervised corpora to model the target languages efficiently. Over the last five years, since the introduction of the word2vec algorithm, many new approaches were proposed to improve different aspects of such representations yielding better performance than the original word2vec in many tasks. Before word2vec, methods, such as LSA, HAL and autoencoders were also widely used. However, to the best of our knowledge at the time of writing, there is still no consensus in the community about whether any of the proposed approaches is clearly superior to others and should be used by default to represent text. This conclusion is consistent with the famous "no free lunch" concept, meaning that for each task the choice of particular word embeddings should be justified and supported with experiments.

In this appendix, we measure the impact of different pre-trained word representations on our task. In our original paper, as mentioned in Section 3.1, we chose the Latent Semantic Analysis (LSA) approach to represent text. The choice was mainly driven by our confidence in our LSA-based model that was used in DBpedia-related tasks before. In this paper, in order to have a precise evaluation of the new feature sets, we have opted to use the same model. However, we believe that an additional set of experiments to measure the impact of different embeddings could serve as an extra data point for the community, not to mention potentially improve the performance of our approach.

A.1 Experimental setting

We compare the previously chosen LSA model to two recent embedding types: GloVe [29] and fastText [3]. To this end, we modify the

computation of the "Description" scalars in our original BASE feature set (see Table 3). Each scalar is computed as follows. Each user text and entity text is converted into the sparse vector $\mathbf{x}_{\text{sparse}} \in \mathbb{R}^v$, where v is the size of the vocabulary for the given language model. Each vector contains tf-idf scores for each token t present in a text:

$$x_t = \text{tf}(t) \cdot \text{idf}(t, D) = \log(1 + \text{freq}_t) \cdot \log\left(1 + \frac{|D|}{1 + |\{d \in D : t \in d\}|}\right)$$

where D is a corpus on which a chosen language model was trained. As can be seen, the computation of such vector requires IDF statistics from the corpus. Here we use precomputed vectors provided by the authors of the respective approaches which do not include such data along with the embeddings. Therefore, we use the same IDF scores acquired from Wikipedia for all models. Each approach is represented by the embedding matrix $M \in \mathbb{R}^{v \times d}$, where d is the embedding size. Then the dense text vector is acquired: $\mathbf{x}_{\text{dense}} = \mathbf{x}^T_{\text{sparse}} \cdot M$. Finally, the Description scalars are computed as a cosine similarity between the user text $\mathbf{u}_{\text{dense}}$ and the entity text $\mathbf{e}_{\text{dense}}$. In short, in order to test other representation approaches we substitute the embedding matrix M_{lsa} we employed originally with $M_{\texttt{fastText}}$ and M_{GloVe}.

We test four different models and measure their impact on the performance of the BASE_KB_SG_TL approach described in the paper:

1. `LSA` ($v = 972{,}001; d = 100$). The same LSA-based approach we used throughout the paper. The model is derived from Wikipedia and is described in [1].
2. `GloVe` ($v = 1{,}917{,}494; d = 300$). The model is trained on 42B token Common Crawl corpus and provided by the authors on their website.[17] This approach is also described in Section 4.1 and used to populate the RDF embeddings we used.
3. `fastText` ($v = 2{,}519{,}370; d = 300$). The word2vec-based model exploiting subword information. The model was provided[18] by the authors and is trained on Wikipedia.
4. `ALL`. Description scalars produced by each model used together. Effectively an ensemble of embeddings.

Additionally, we slightly modify our *data acquisition* phase. During this phase, we would typically gather the textual content for each candidate from the stream of tweets. As a consequence, we would have a lot of textual content for more popular users and much less (as little as a description field in the profile) for others. This makes it so the "Description" features get a strong implicit notion of the user popularity instead of capturing only the textual similarity. To alleviate this bias, we freshly captured the most recent 200 tweets for each candidate for each entity in our gold standard using Twitter REST API.

All approaches are evaluated using stratified 5-fold cross validation, additionally computing 95% confidence intervals and performing statistical significance test using the approximate randomization test (see Section 6.1).

A.2 Experimental results

Evaluation results for the four models are provided in Table 7, while Figure 8 provides the precision / recall curves. As can be seen, the difference between the four models is minimal. However, the `ALL` model does provide statistically significant improvement over the `LSA` model we employed originally, and similarly over `fastText` and `GloVe`.

The absence of significant differences between separate models shows that our pipeline is not particularly sensitive towards the choice of the particular word embeddings. In future, the model can be modified to include textual representations in a similar way we have incorporated the graphical ones, to give more opportunity for the neural network to utilize textual data efficiently.

[17] `http://nlp.stanford.edu/data/glove.42B.300d.zip`

[18] `https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md`