Chapter 2

Background

In this chapter, I start by briefly introducing the relevant Semantic Web concepts, such as the Linked Open Data, most prominent datasets, related technologies and tools that are employed in this thesis. Then, I will highlight the key concepts of the social media. Finally, the majority of this chapter is dedicated to outlining some of the fundamental principles and state-of-the-art approaches employed to represent textual and graph data that are used throughout this thesis. In addition to the content presented in this chapter, going forward I will also provide a more in-depth look at related work and relevant background in each chapter separately, where that information pertains specifically to the contents discussed in the chapter.

2.1 Semantic Web and Linked Open Data

Semantic Web is the initiative that aims to extend the World Wide Web with the goal of facilitating the rapid sharing and reuse of data. This goal is achieved by developing open standards and best practices under the World Wide Web Consortium (W3C)[1] making data in the web accessible, machine-readable and increasingly interlinked.

A centerpiece of the Semantic Web initiative is the Linked Open Data (LOD) cloud, which is a collection of interlinked datasets that are released under an open license. A LOD-compliant dataset has to adhere to a list of simple rules: (i) the instances described in the dataset have to be identifiable via URIs, (ii) when queried, descriptions have to be returned using open standards, such as RDF and SPARQL, and (iii) links have to be provided to other related URIs in other datasets from the LOD. Such principles were declared essential for the success of the Semantic Web project (Berners-Lee, 2006) and direct the community effort to share as many interlinked datasets as possible making all sorts of knowledge publicly available in a standardized machine-readable form. This effort has yielded billions of RDF triples accessible from thousands of different datasets over

[1] http://www.w3.org

the last years. With such vast array of knowledge readily available, the LOD cloud is routinely used as a source of background knowledge for tasks spanning across many fields, for example, in Computer Vision and Natural Language Processing.

Independently from the LOD concept, Wilkinson et al. (2016) have defined a set of principles, namely, Findability, Accessibility, Interoperability, Reusability (collectively known as FAIR principles) aimed at supporting the reuse of data produced by researchers. It has since been adopted by academia, including the major Semantic Web outlets, such as ISWC and ESWC conferences, and industry alike as guidelines to consider when building and sharing datasets to improve data discovery, access, integration, adequate citation, and reuse. One of the major contributions of this thesis, the SocialLink resource, is build both to be LOD-compliant and having FAIR principles in mind.

One of the largest datasets existing in the LOD cloud is DBpedia (Lehmann et al., 2015). DBpedia is built by automatically extracting structured multilingual knowledge from Wikipedia, the crowdsourced online encyclopedia, meaning that each entity contained in DBpedia is backed by the corresponding Wikipedia article. In total, DBpedia has 128 language chapters covering corresponding Wikipedia ones and provides over 1.46[2] billion RDF triples. Being the most interlinked dataset in the LOD cloud, it acts as a centerpiece providing a simple access point for the researchers and industry alike to the world of linked data.

On the other hand, the Wikidata[3] project of the Wikimedia Foundation is a collaborative effort aiming at creating a knowledge backbone for the Wikipedia and related projects accumulating structured knowledge in a LOD-compliant form. Wikipedia is able to populate its infoboxes and other structured parts from the knowledge contained in Wikidata. Much like DBpedia, Wikidata is multilingual and each Wikipedia article has its Wikidata counterpart entity. Following best LOD and FAIR practices, it is freely accessible and interlinked with various datasets within and outside the LOD cloud. Additionally, Wikidata aims at providing references for the claims it contains, allowing the user to verify the knowledge by checking where it comes from. Unfortunately, many of the claims in Wikidata are not appropriately sourced making SocialLink, which is a core contribution of this thesis, even more relevant as it allows using social media profiles as a source of references.[4] Additionally, even though SocialLink mainly targets DBpedia, Wikidata URIs are used where possible in the resource to identify entities linking them to DBpedia URIs via the owl:sameAs links.

LOD-compliant datasets typically rely on Resource Description Framework (RDF) specification to represent its knowledge. RDF 1.1 (Wood et al., 2014) presents data

[2] As described in Lehmann et al. (2015)
[3] https://www.wikidata.org
[4] See the soweego project discussed in Section 4.4.4.

as subject-predicate-object triples forming a knowledge graph. A node in such graph (a subject or an object of a triple) can either be a URI, a typed Unicode string literal or a blank node (denoting anonymous resources), while the predicate has to be a URI representing a relationship. Type information for arbitrary nodes can be expressed using the predicate rdf:type together with object URIs identifying types. The semantics of these predicates and types may be described using ontological languages such as RDFS (Guha and Brickley, 2014) and OWL (Motik et al., 2012), themselves expressible in RDF. In this thesis, I use OWL 2 to formalize the semantics of SocialLink data. RDF graphs can be serialized in many formats, such as Turtle, RDF/XML, JSON-LD, RDF/JSON, and many others. To perform various manipulations with RDF knowledge graphs, such as statistics extraction, smushing, filtering, deduplication, and other transformations, many tools have been developed over the last years. In this thesis, unless stated otherwise, I primarily employ the RDFpro (Corcoglioniti et al., 2015) tool to perform the large-scale local processing of RDF data. RDFpro is an open source Java library with a command line interface that implements many of the operations typically needed to produce and work with LOD datasets.

In order to query and manipulate the RDF-based knowledge graphs, the SPARQL Protocol and RDF Query Language (SPARQL) was introduced and standardized by the W3C (Harris and Seaborne, 2013). SPARQL endpoints offer access to datasets using a simple HTTP-based protocol supporting multiple output formats. The rich query language of SPARQL resembles other querying languages, such as SQL. It offers read queries that can be restricted with WHERE clauses to define patterns and constrain the query; then a variety of standard operations, such as joins, unions, intersections, aggregations, subqueries, and others exist to further transform query results. SPARQL also supports update queries that modify stored RDF triples. Additionally, the latest specification adds support for federated queries that allow accessing RDF data distributed across multiple endpoints. SPARQL has a number of open source implementations including RDF4J, Jena[5] and Virtuoso[6], GraphDB[7] and BlazeGraph.[8] In this thesis, I employ the KnowledgeStore[9] storage system with a Virtuoso backend to store, manage and query the RDF-based datasets. KnowledgeStore provides a SPARQL endpoint and offers a web-based user interface to manage the stored RDF triples and associated metadata.

[5] https://jena.apache.org
[6] https://virtuoso.openlinksw.com
[7] http://graphdb.ontotext.com
[8] https://www.blazegraph.com
[9] https://knowledgestore.fbk.eu

2.2 Social Media

Social media has attracted a considerable amount of research attention over the last decade due to its popularity and its provision of enormous amounts of real-time knowledge. Social media are mostly proprietary outlets that provide an opportunity for the users to create and post content, to search and interact with other users and the content created by them. Users create and fill their public or private profiles, which may or may not include machine-readable attributes describing them. Generally speaking, the users cannot verify if the profile they are interacting with is genuine, making it easy for an impostor to steal someone else's identity or invent an entirely new one. However, the social network can in some cases verify the identity of high-profile public entities partially alleviating this issue. The current major social networks include Facebook with $2.3B$ (statista.com, 2019a) monthly active users, Instagram with $1B$ (statista.com, 2019b) and Twitter with over $300M$ (statista.com, 2019c).

Not every social network, however, is proprietary and closed source. Since the publication of the OStatus[10] standard and its successor, ActivityPub,[11] by the W3C, there has been an increased community effort to create a new generation of social media based entirely on open standards increasing the end users' ability to control their data and improving compatibility between different social media. The most prominent examples include Mastodon[12] and GNU Social.[13]

The key concept of any social media is the social graph — a directed graph of users that captures explicit and implicit interactions between them. It may be implemented via some kind of friend, follow or subscribe action that generally puts the content written by one user to the other users content feed. However, even if the user does not explicitly state their desire to subscribe to someone, the mere fact of reaction to someone else's content (via like, share or comment functionality) may be enough to establish the link between two users. As will be repeatedly shown throughout this thesis, the social graph can reveal much and more about any given user.

Many social media provide APIs for third parties to access a subset of the available users' data. This data may include the above-mentioned social graph, user's public profile, user's public content and some metadata, such as location and event data, profile settings and popularity statistics. Even though the released data is typically incomplete due to privacy or business reasons, it has been repeatedly shown[14] that a third party can exploit

[10] https://www.w3.org/community/ostatus
[11] https://www.w3.org/TR/activitypub
[12] https://mastodon.social
[13] https://gnu.io/social
[14] See, for example, Section 6.2 for the review of the user profiling task that specifically tackles this problem.

this data to fill out the blanks in the user's profile and gather the information that was not supposed to be available, making the privacy implications of the social media an increasingly important topic of public discussion. In case the API is not present, social media data can be collected using a web crawler, which, however, can in many cases violate the terms of use.

Social media are being routinely used in the academic community to assist in solving a wide variety of tasks spanning across many fields. I will provide a review of the tasks that are relevant to this thesis, namely user profiling (Section 6.2), and profile matching (Section 3.8), in the subsequent chapters.

2.3 Representation Learning

The performance of machine learning approaches significantly depends on the way we represent input data. Typically, for each data type, there is a set of conventional approaches to extract features from the raw input. For example, categorical input is usually represented as a sparse vector $\mathbf{x} \in \mathbb{R}^l$, where l is the number of possible values for this particular input. Numeric features are normalized in some way to range from -1 to 1 or from 0 to 1 depending on the approach. Specialized representations exist for more complex data types, such as text, images, graphs and other.

Additionally, representations for a particular set of features can be learned (see Chapter 15 in Goodfellow et al. 2016). The idea is to acquire a representation for a given set of features as a by-product of one learning algorithm and then use it as input for another algorithm, effectively sharing the learned knowledge across multiple tasks. In particular, suppose that we set up a feed-forward neural network that is trained to solve task A given some representation of an object as input. Regardless of the chosen initial feature set, by the final layer, the feed forward neural network will naturally learn to represent this object most suitably based on the task at hand. For example, if the classes in the original problem were not linearly separable given the input features, they may become separable by the final hidden layer. We can then use this representation at the last hidden layer and feed it as input to task B that expects features based on this object as input. In this case, we are effectively collecting what we have learned about the object at hand during solving the first task and reusing it (transferring this learning) in a different task.

The concept of learning representations becomes even more important considering that we might have plenty of data to solve task A but not enough to solve task B even if a researcher tries hard to extract the best possible features from the input object. In real scenarios, we would typically have a significant number of unlabeled samples and a relatively limited number of labeled ones. In this case, we would usually learn representations in an unsupervised way by designing an unsupervised objective to highlight

certain inherent properties of objects in a dataset. For example, for the text we might follow the *distributional semantics* hypothesis described below or corrupting the input sequence in some way and training the approach to correct the mistake making the learning algorithm to encode the meaning of the sequence into its internal state and then decoding the corrected version.

In the subsequent sections, I will describe the core representation learning techniques that I employed in this thesis. Tasks tackled here routinely use the notion of a social media user or an knowledge base entity as input for both of which a vast amount of unsupervised knowledge is available. Given that the size of a training set for each individual task is rather small to learn representations for such complex objects from the ground up, the idea of learning them using an unsupervised approach[15] is of paramount importance.

2.4 Text Representations

In this section, I briefly recap the algorithms and techniques used to represent written text in vectorial form throughout this thesis. Textual features are a cornerstone of any analysis done on social media and provide significant performance improvements for many of the tasks in the Semantic Web.

The most straightforward way to represent written text is a bag-of-words model. There, the input text is converted to a sparse vector $\mathbf{x} \in \mathbb{R}^v$ where v is the size of the vocabulary. Each non-zero index of this vector contains a score associated with a specific term that is present in the text. In the simplest case, each term in a text has a score of 1.0, yielding a so-called one-hot representation. In general, the score consists of two components: term frequency (TF) and the inverse document frequency (IDF). Semantically, TF is a frequency with which the term appears in the input text, while the IDF is the inverse frequency with which the term appears in the corpus. While there exist many weighting schemas for both TF and IDF, in this thesis I will employ the following formula for each element of $\mathbf{x}$:

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

where t is a term, d is a document in a corpus (e.g., a tweet in a corpus of tweets), D is a chosen corpus and freq_t is the number of occurrences of t in the document.

Such representation, however, is suboptimal for many applications (see Section 6 in Zhang et al. 2016). A high-dimensional sparse representation resulting from the bag-of-words approach is increasingly difficult to use as input with many of the machine learning

[15] This process usually referred to as *unsupervised pretraining*

algorithms, especially the ones based on neural networks, as each word index has to be associated with an individual weight and sparse tensor operations are much less performant than their dense counterparts on modern hardware.

Additionally, the bag-of-words model suffers from a *vocabulary mismatch* problem. The *vocabulary mismatch* arises when there is a need to group and compare semantically similar items. In a bag-of-words model, "cat" and "tiger" yield orthogonal representations even though both terms refer to felines. It means that "cat" would be as distant from "tiger" as it is from, let's say, "table", providing no extra semantic information besides a simple token match to the downstream models.

In order to reduce the dimensionality of the input while also solving the *vocabulary mismatch* issue, various algorithms were proposed. In these approaches, each term i is encoded into a dense low-dimensional continuous representation $\mathbf{w}_i \in \mathbb{R}^d$, where d is a chosen size of the representation, commonly referred to as *word embedding*. The word embeddings are typically acquired using an unsupervised algorithm that uses a large corpora to learn representations according to some hypothesis. Modern embedding algorithms used in practise, such as LSA, word2vec and GloVe, employ a so-called *distributional semantics* hypothesis which states that words that appear in similar contexts tend to have similar meanings. Given that we are interested in learning a vector for each word, we would like the words that appear in similar contexts to be close to each other in the resulting vector space according to some similarity metric, such as *cosine similarity*.[16] As can be seen, the vocabulary mismatch problem, that made all words orthogonal to each other, is solved since the words in a new vector space are arranged based on their meaning. Note that here and onwards I would use "word", "term" and "token" as synonyms, while it is assumed that in a text we can encounter other tokens such as punctuation, numbers and even emoji. Once the word embeddings are populated, dense low-dimensional representation of text can be acquired by simply multiplying the sparse TF-IDF vector $\mathbf{x}$ by the embedding matrix $\mathbf{M}$:

$$\mathbf{x}_{\text{dense}} = \mathbf{x}^T \cdot \mathbf{M} \tag{2.1}$$

Throughout this thesis, I employ various embedding algorithms from the literature that follow the distributional semantics hypothesis. Here, I describe three of them, LSA, GloVe and Swivel, in details as they are used in the proposed approaches and briefly discuss word2vec and fastText.

[16] $\text{sim}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}$

2.4.1 LSA

Latent Semantic Analysis (LSA, Deerwester et al. 1990) is a technique that allows learning of word representations from the co-occurrence matrix of words and documents. Given a corpus of n documents having a vocabulary of v unique words, we can build a co-occurrence matrix $\mathbf{X} \in \mathbb{R}^{v \times n}$ such that x_{ij} corresponds to some frequency measure (which can be, for example, the above-mentioned TF-IDF weight) of ith word of the vocabulary in jth document. From such matrix we can already derive word vectors of size n by taking the corresponding rows of $\mathbf{X}$. The resulting vector space will already be arranged according to distributional semantics and we can compute similarity score between the words using those vectors.

However, the original matrix $\mathbf{X}$ is large, noisy and sparse. To reduce the dimensionality of $\mathbf{X}$, LSA proposes to employ singular value decomposition (SVD), splitting $\mathbf{X}$ into a product of two orthogonal matrices and a diagonal matrix: $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \mathbf{U} \in \mathbb{R}^{v \times v}, \mathbf{\Sigma} \in \mathbb{R}^{v \times n}, \mathbf{V} \in \mathbb{R}^{n \times n}$. By taking the d largest singular values from $\mathbf{\Sigma}$ and their corresponding singular vectors from $\mathbf{U}$ and $\mathbf{V}$, we acquire an approximation of $\mathbf{X}$: $\mathbf{X}_d = \mathbf{U}_d\mathbf{\Sigma}_d\mathbf{V}_d^T, \mathbf{U}_d \in \mathbb{R}^{v \times d}, \mathbf{\Sigma}_d \in \mathbb{R}^{d \times d}, \mathbf{V}_d \in \mathbb{R}^{d \times n}$. This factorization procedure is referred to as *truncated SVD* and it is shown that $\mathbf{X}_d$ is the closest possible approximation of $\mathbf{X}$ with rank d matrices. According to the LSA procedure, if we then compute:

$$\mathbf{M}_{\text{lsa}} = \mathbf{U}_d \cdot \mathbf{\Sigma}_d, \quad \mathbf{M}_{\text{lsa}} \in \mathbb{R}^{v \times d} \tag{2.2}$$

the rows of $\mathbf{M}_{\text{lsa}}$ can be employed as embeddings of size d for the corresponding tokens yielding a convenient low-dimensional representation based on unsupervised co-occurrence statistics derived from the selected corpus. In this thesis, we employ the LSA model built basen on the corpus derived from seven language chapters of Wikipedia and detailed in Aprosio et al. (2013). I use this model to represent textual content in both Twitter and DBpedia across all the approaches presented in this thesis.

2.4.2 GloVe and Swivel

Recently proposed GloVe (Pennington et al., 2014) and Swivel (Shazeer et al., 2016) are the global log-bilinear regression models that implement the distributional hypothesis to learn word vectors. The co-occurrence matrix used in these methods is term-term instead of term-document, meaning that x_{ij} contains the frequency with which the term i appears in the context of the term j yielding the square matrix $\mathbf{X} \in \mathbb{R}^{v \times v}$, where v is the size of vocabulary. The context of any given word consists of all the words around it up to a certain distance. So, given a sentence $S = \{t_1, t_2, t_3, t_4, t_5\}$ and distance $l = 2$, t_1 would be co-occurring with t_2 and t_3, while t_4 would be co-occurring with t_2, t_3 and

t_5. Such definition of a context was initially proposed in Lund and Burgess (1996) for HAL approach. Typically, the weight of a single co-occurrence between any two words is defined as $1/l$ given that they are l words apart, which are then summed up over all co-occurrences of those two words in the corpus.

After the co-occurrence statistics are computed, the approximate factorization of $\mathbf{X}$ into matrices $\mathbf{W} \in \mathbb{R}^{v \times d}$ and $\tilde{\mathbf{W}} \in \mathbb{R}^{v \times d}$ is performed by formulating an optimization problem and minimizing the appropriate objective function treating $\mathbf{W}$ and $\tilde{\mathbf{W}}$ as weights. To do that GloVe defines the following objective function with the goal of encoraging $\mathbf{W}\tilde{\mathbf{W}}$ to approximate $\log(\mathbf{X})$:

$$J_{\text{GloVe}} = \sum_{i,j} f(x_{ij})(\mathbf{w}_i^\top \tilde{\mathbf{w}}_j + b_i + \tilde{b}_j - \log x_{ij})^2 \tag{2.3}$$

where $f(\cdot)$ is a weighting function and b_i and $\tilde{b}_j$ are the biases. The weighting function is selected in such way so that the frequent co-occurrences between words are not overweighted. Additionally, f is chosen so that $f(0) = 0$, meaning that sparse values in $\mathbf{X}$ (which usually constitute 75-95% of values in such matrix) can be skipped during training which significantly improves the performance of the approach for large v. Finally, one of the resulting matrices $\mathbf{W}$ or $\tilde{\mathbf{W}}$ can be defined as a matrix $\mathbf{M}_{\text{GloVe}}$ of word embeddings of size d, which can be used in eq. 2.1 to acquire a dense representation of a text.

Swivel builds on the GloVe approach but instead of approximating $log(\mathbf{X})$ it estimates the *pointwise mutual information* (Church and Hanks, 1990) between the terms $\text{pmi}(i; j)$, which is defined as follows:

$$\text{pmi}(i; j) = \log \frac{P(i, j)}{P(i)P(j)} = \log \frac{x_{ij}|D|}{x_{i*}x_{*j}} \tag{2.4}$$

where $x_{i*} = \sum_j x_{ij}$, $x_{*j} = \sum_i x_{ij}$ and $|D| = \sum_{i,j} x_{ij}$. Which modifies the objective function (eq. 2.3) as follows:

$$\begin{aligned} J_{\text{swivel}} &= \frac{1}{2} \sum_{i,j} f(X_{ij})(\mathbf{w}_i^\top \tilde{\mathbf{w}}_j + b_i + \tilde{b}_j - \text{pmi}(i; j)))^2 \\ &= \frac{1}{2} \sum_{i,j} f(X_{ij})(\mathbf{w}_i^\top \tilde{\mathbf{w}}_j + b_i + \tilde{b}_j - \log x_{ij} - \log |D| + \log x_{i*} + \log x_{*j})^2 \end{aligned} \tag{2.5}$$

The primary distinction between Swivel and GloVe is the introduction of the "soft hinge" loss for cases where $x_{ij} = 0$:

$$J_{\text{hinge}} = \log[1 + \exp(\mathbf{w}_i^\top \tilde{\mathbf{w}}_j - \log |D| + \log x_{i*} + \log x_{*j})] \tag{2.6}$$

This special case allows the unobserved co-occurrences to contribute to the learning process, producing more stable estimates for rare words. It is worth noting, however, that the introduction of the special case for zero values of matrix $\mathbf{X}$ negatively affects performance making it impractical to use Swivel for learning embedding matrices with large vocabularies. To mitigate this issue, the authors of Swivel propose to split the co-occurrence matrix into shards making the whole training step easily parallelizable. Even though, Swivel and GloVe were originally designed to represent words, it is shown (Pennington et al., 2014; Nechaev et al., 2018b; Cochez et al., 2017b) that the same log-bilinear models can be exploited for any type of objects — it is just the matter of defining and computing the appropriate co-occurrence matrix. In particular, these approaches will be used in this thesis to represent nodes in a graph.

2.5 Graph Embeddings

Recently, learning of dense low-dimensional representations for nodes in a graph (onwards *graph embeddings*) to provide a better alternative to sparse representations have become a hot topic in the community (Cai et al., 2018). Such representations are typically learned using an unsupervised approach employing the entire graph for training. The task is similar to the one of learning word embeddings in NLP: given the graph $G = (V, E)$, where V are the nodes and $(v_i, v_j) \in E$ are the edges, we would like to learn an embedding matrix $\mathbf{X}_G \in \mathbb{R}^{|V| \times d}$ so that rows give d-dimensional representations for each node in a shared vector space. Alternatively, if the edges can be typed (labeled), the graph definition changes to $G = (V, E, R)$ with a set of possible relation types R and triples representing edges $(v_i, r_k, v_j) \in E, r_k \in R$. In this case representations can also be learned for the relation types themselves (Nickel et al., 2011; Guu et al., 2015; Bordes et al., 2013; Socher et al., 2013). However, even when the type information is available (e.g., knowledge graphs in the LOD), I will only employ representations for nodes for tasks presented in this thesis. Here, I provide the review of the approaches employed over the last years to acquire graph embeddings including the ones that are either directly used (Cochez et al., 2017b; Ristoski et al., 2017) in this thesis or that influenced (Tang et al., 2015; Nickel et al., 2011) the design decisions made in Chapter 3 to produce embeddings for Twitter users derived from the social graph.

2.5.1 General Graph Embedding Methods

The importance of representing graph-based features in the context of this thesis is hard to overstate. In the Semantic Web community, given that any LOD dataset is essentially a knowledge graph, graph-based features play a key role in any machine learning-based

task there. In social media analysis, the usage of social graphs enables tasks that wouldn't be possible otherwise (e.g., see interests prediction for passive users in Chapter 6) and significantly improves performance of approaches for tasks like user profiling, profile matching, recommendation and event detection.

After the overwhelming success of learning embeddings to represent text in Natural Language Processing (see Section 2.4), researchers have started to try to use the same ideas, namely the distributional hypothesis, to learn graph embeddings. In such works, the graph is seen as a document, while the nodes are viewed as tokens in this document. However, documents have a linear structure, meaning that the sequence of tokens can be scanned sequentially and each word has a clear notion of context, while the graphs do not naturally posses this property. In order to resolve this issue, the graph has to be linearized in some way to resemble a sequence from which the context for each node can be derived.

DeepWalk (Perozzi et al., 2014) approach was one of the first to linearize the graph to learn embeddings exploiting the methods designed to work on text. Its authors proposed to use uniform random walks on graph to model it as a set of sequences to run a Skipgram model of word2vec on them. Their experiments on multiple datasets showed the effectiveness of the approach and pawed the way for other researchers to further explore this direction. One of the extensions of the DeepWalk was the Deep Graph Kernels (Yanardag and Vishwanathan, 2015) system that learned embeddings for structured objects, such as graphlets and strings instead of nodes. The authors of DGK have also explored a number of alternative linearization mechanisms other than random walks. Direct usage of the Skipgram objective is not the only way to learn graph embeddings. Instead, LINE (Tang et al., 2015) proposes to directly model the first and second order proximity using *breadth-first search* starting from the sampled node. LINE approach is conceptually similar to matrix factorization methods but is designed to preserve the original network structure instead of just node similarity.

Node2vec (Grover and Leskovec, 2016) builds upon and generalizes the DeepWalk approach. Node2vec authors further formalize the random walk generation procedure by introducing two parameters. The return parameter p controls the probability with which the random walk procedure is able to use the edge it had just traversed and the in-out parameter q controls the probability of leaving or staying in the tightly connected groups of nodes. Different configurations of those two parameters control the way with which the random walking procedure balances *breadth-first search* and the *depth-first search*. Authors note that DeepWalk is effectively a special case of node2vec with $p = 1$ and $q = 1$. The additional flexibility in controling the linearization procedure allowed Node2vec to outperform both the LINE and DeepWalk approaches.

Additionally, approaches designed to learn embeddings for both nodes and relations in knowledge graphs specifically have been explored. RESCAL (Nickel et al., 2011) factorize

the tensor of relations and nodes learning representations for both as a result. One of the possible variations of this model was proposed by Guu et al. (2015). HolE (Nickel et al., 2016) offers a more efficient yet a more expressive version of RESCAL by simplifying the model and the objective via the usage of circular correlation operator to obtain the composition of embeddings. Similar compositional approach but in a pure neural network-based scenario was also proposed in Socher et al. (2013) (Neural Tensor Networks). There they learn low-dimensional representations for entities and a combination of a rank 2 and rank 3 tensors to represent relation. TransE (Bordes et al., 2013), on the other hand, learns representations for entities and relations by contrasting the real subject-relation-object triples with the corrupted ones assigning the loss penalty if the corrupted triple is scored higher than the correct one. TransE has a multitude of extensions including TransH (Wang et al., 2014) and TransR (Lin et al., 2015) that aimed to solve various limitations of the original approach. TransE has been routinely used in the last years for the *link prediction* task, which is a generalized version of the *type prediction* task explored in Chapter 5 of this thesis.

2.5.2 RDF Embeddings

LOD knowledge bases consist of RDF triples subject-relation-object, which can naturaly be represented as a graph with typed edges defined above. While any of the approaches described in Section 2.5 can be used to represent nodes in such graphs, many researchers (Ristoski et al., 2017; Ristoski and Paulheim, 2016a; Cochez et al., 2017a,b) have tried to tune their approaches for RDF-based knowledge graphs specifically resulting in improved performance for downstream tasks in the LOD. I will refer to embeddings produced by such specialized approaches as RDF embeddings.

One of the approaches for training RDF embeddings, RDF2Vec (Ristoski et al., 2017; Ristoski and Paulheim, 2016a), uses RDF graph kernels from the recent literature, specifically the Weisfeiler-Lehman Subtree RDF graph kernels (de Vries, 2013; de Vries and de Rooij, 2015), to perform linearization of nodes in a graph. This combined with the breadth-first search random walks from DeepWalk (Perozzi et al., 2014) allowed them to train embeddings using the usual Skipgram objective. The two linearization strategies are complementary meaning that one or the other can be skipped based, for example, on the task performance requirements. RDF2Vec authors have provided pretrained embeddings for DBpedia and Wikidata allowing other researchers to employ them in downstream tasks without reimplementing the approach or even retraining. The resulting embeddings semantically group entities of a KB. For example, dbr:France have dbr:Germany and dbr:Italy as nearest neighbours, while dbr:Paris is close to dbr:Berlin and dbr:Rome. Additionally, we can see that the chosen linearization procedure preserves other properties that can

typically be found in distributional semantics-based language models: linear substructures that capture relations between words in methods, such as GloVe, can also be found in embeddings produced by RDF2Vec. For example, if we substract dbr:Germany vector from dbr:Berlin and then add dbr:Italy, the nearest entity would be dbr:Rome meaning that for such entity the property dbo:Country and the complementary property dbo:Capital are implicitly preserved.

While the RDF2Vec approach focuses on using Skipgram objective to learn RDF graph embeddings for DBpedia and Wikidata, the subsequent work by Cochez et al. (2017b) employs the GloVe approach I described in Section 2.4.2. The usage of a method based on factorization of the co-occurrence matrix allowed the authors to explore approaches that do not rely on linearizing a graph into a set of sequences. Instead, reliance on the co-occurrence matrix only requires some notion of importance to be computed between a target and any other node in order for the approach to function. Cochez et al. (2017b) employ the Personalized PageRank (PPR) (Page et al., 1999) algorithm to produce such importance statistics. However, since the PPR had to be computed for each node separately and the number of nodes is high in the target knowledge graphs, the authors opted to use the Bookmark-Coloring Algorithm (BCA) (Berkhin, 2006) to produce an approximation of PPR. The authors further improve the performance of this method by reusing many of the values for the previously computed nodes.

Finally, the BCA approach can take into account edge weights. As DBpedia does not have a readily available notion of weight, Cochez et al. (2017b) propose twelve different weighting schemas based on raw frequences (both edge-centric and node-centric), different variations of Wikipedia-based PageRank and uniform weights. Such weighting schemas for RDF graphs were originally introduced in a prior work by the same authors (Cochez et al., 2017a) and applied as an extension to the original RDF2Vec. Throughout this thesis, I will employ the precomputed versions of their RDF embeddings with different weighting schemas to represent knowledge graph nodes in DBpedia and Wikidata.

2.6 Conclusions

Here I have presented relevant background I will refer to throughout this thesis: basic Semantic Web and Social Media concepts; the brief review of the state-of-the-art data representation approaches including the ones used to represent text and various graph data. As the subsequent chapters cover different topics, additionally I will provide the review of those topics in the corresponding sections. Specifically, Section 3.8 provides relevant background for the linking task that is at the core of this thesis including the review of the profile matching task and some of the similar linking techniques used in social media analysis. Section 5.7 talks about recent type prediction approaches and feature extraction

from social media content. Section 6.2 covers the user profiling task in general with the emphasis on interests inference, privacy issues, and binarized neural networks. Finally, Section 7.3.1 talks about follower acquisition strategies and recommendation techniques.