Noname manuscript No.
(will be inserted by the editor)

Discovering Similarity and Dissimilarity Relations for Knowledge Propagation in Web Ontologies

Pasquale Minervini[1], Claudia d'Amato[1], Nicola Fanizzi[1], Volker Tresp[2]

the date of receipt and acceptance should be inserted later

Abstract We focus on the problem of predicting missing class memberships and property assertions in Web Ontologies. We start from the assumption that related entities influence each other, and they may be either *similar* or *dissimilar* with respect to a given set of properties: the former case is referred to as *homophily*, and the latter as *heterophily*. We present an efficient method for predicting missing class and property assertions for a set of individuals within an ontology, by: *a*) Identifying relations that are likely to encode influence relations between individuals (learning phase) and *b*) Leveraging such relations for propagating property information across related entities (inference phase). We show that the complexity of both inference and learning is nearly-linear in the number of edges in the influence graph, and we provide an empirical evaluation of the proposed method.

1 Introduction

In the era of the Web of Data, the interest in using Semantic Web (SW) [?] formalisms for representing semantically annotated knowledge bases and resources is naturally increasing. An example of this trend is provided by the *Linked Open Data* (LOD) [?] Cloud, a set of interlinked knowledge bases (KBs) which includes, among others, DBpedia [?], Freebase [?] and YAGO [?]. As of April 2014, the LOD Cloud was composed of 1091 interlinked KBs, describing 8×10^6 entities connected by 188×10^6 relationships holding between them [?].

[1] Department of Computer Science - University of Bari, Italy
E-mail: {firstname.lastname}@uniba.it
[2] Siemens AG, Corporate Technology, Munich, Germany
E-mail: volker.tresp@siemens.com

However, despite the large scale and popularity, SW Knowledge Bases are often far from being complete. For instance, as of October 2013, 71% of the persons described in Freebase have no known place of birth and 75% of them have no known nationality; such percentages increase for less frequently used relationships [?]. For this reason, in this paper, we focus on the problem of predicting missing assertions, in the form of class-memberships and properties (such as the aforementioned *nationality*) of individuals, hereafter also called entities, in Web Ontologies. In the literature, this task is often referred to as *assertion prediction* or *Knowledge Graph completion* [?], and it is often tackled by using *Machine Learning* methods [?, ?, ?]. As an example of the usefulness of automatically completing SW Knowledge Bases, consider the following scenario:

Example 1 (Academic Domain) Let $\mathcal{K}$ be a knowledge base modeling an academic domain that contains the following set of assertions:

- Mark, Lucas and John are researchers.
- Mark is the supervisor of Lucas, and Lucas works with John.
- Mark and John are affiliated with the "Efficient Algorithms" (`EffAlg`) and "Complexity Management" (`CoM`) research groups, respectively.

Let us assume it is also known that Lucas is affiliated with a research group, but it is not known whether such a research group is `EffAlg`, `CoM` or none of them. By using *assertion prediction* methods, it is possible to determine the most likely research group Lucas is affiliated with, even if neither such knowledge is available as an assertion nor it can be inferred by deductive inference, as it often happens in the SW setting. ■

We propose a method for predicting missing class and property assertions in SW KBs. The method is grounded on the intuition that related entities (individuals) may influence each other. Depending on the nature of the influence relationship, this phenomenon is referred to as either *homophily* (i.e. "love of the same") or *heterophily* (i.e. "love of the different"): homophily indicates the tendency of individuals to associate with similar others, while heterophily indicates the tendency of individuals to relate with different ones.

Homophily and heterophily arise in a vast array of networked domains [?, ?]. For instance, in social networks, friends tend to have common characteristics, such as religious beliefs and political views (homophily) On the other hand, talkative persons tend to have silent friends, and vice-versa (heterophily) [?].

Unfortunately, identifying such relations and the type of influence they convey is not straightforward, also because individuals might not be related directly, but rather through chains of relations, such as the case of *friends of friends* in social networks.

In particular, we propose *Adaptive Knowledge Propagation with (Dis-)Similarities* (AKP^{D}), a method for predicting missing class and property assertions of individual resources in Web Ontologies. AKP^{D} first identifies *homophilic* and *heterophilic* relations, then efficiently leverages them for propagating knowledge through (chains of) related individuals. In particular:

1. It first *identifies* homophilic and heterophilic relations holding between a set of considered individuals, with respect to a given set of properties (such as *nationality*) – for example, by recognizing that friends tend to share the same nationality.
2. Then, it leverages such relations for constructing an *influence graph*, which is then used for efficiently *propagating* knowledge across chains of related individuals - e.g. from a single person to their friends, and the friends of their friends.

The proposed method is an extension of our previous work, which only considers homophilic relations [?, ?]. When constructing the influence graph, the method learns weights to be associated to properties. Each weight is interpreted in terms of how likely it is that a given property (hereafter also called *relation*) links two individuals sharing the same properties or belonging to the same class.

From this perspective, our approach is closely related to Graph-based *Semi-Supervised Learning* (SSL) [?]. Such methods rely on a *similarity graph*, defined over examples, to propagate label information across them. However, they have two main limitations:

1. They assume the similarity graph to be already provided, without tackling the problem of learning an optimal similarity graph from available training data.
2. The similarity graph can only represent similarity relations between examples, and cannot express *dissimilarity* relations (such as *John and Mary have different political views*).

The proposed method overcomes such limitations, by automatically learning an *optimal* influence graph by leveraging homophilic and heterophilic relations holding between individuals in a SW Knowledge Base.

We also provide an in-depth analysis of the complexity of our method (see Sect. 4 and Sect. 5). In particular, by leveraging recent results in numerical analysis, we show that: *a*) The result of the propagation process can be computed efficiently, with a nearly-linear complexity in the number of edges in the influence graph, and *b*) The proposed gradient-based learning algorithm also shares similar complexity properties.

Our method is especially useful with real-world *shallow ontologies* [?], such as social networks and citation networks, which are characterized by a relatively simple terminology and populated by very large amounts of instance data. A lesson learned, however, is that also considering heterophilic relations in the propagation process does not sensibly improve prediction results on prediction tasks discussed in the literature.

The remainder of this article is organized as follows. In Sect. 2, we review the basics of semantic knowledge representation and reasoning tasks, and we introduce the concept of *transductive learning* in the context of SW knowledge bases. In Sect. 3, we illustrate how influence relations between individuals can be used to define a *penalty term* over discriminant functions, which is then used in Sect. 4 for efficiently propagating label information among individuals across a given *influence graph*. In Sect. 5, we propose an efficient solution to the problem of *learning* an optimal influence graph for a given learning task, by leveraging relations holding between individuals in the ontology and in Sect. 6 we propose a solution to the problem of efficiently retrieving relations between individuals, and provide an overall summary of the proposed method. In Sect. 7 we survey related work on predicting missing assertions in Web Ontologies. In Sect. 8, we evaluate AKP^{D} on several learning tasks, involving real world KBs. In Sect. 9, we draw conclusions and discuss possible future research directions.

2 Basics

We assume that the considered KBs are encoded in the form of a *Description Logic* (DL) [?]. A DL KB describes a set of objects (or *entities*), their attributes and the relations between them. Fundamental elements are *atomic concept names* $N_C = \{C, D, \ldots\}$, interpreted as subsets of a domain of objects (e.g. `Person` or `Article`), and *atomic role names* $N_R = \{R, S, \ldots\}$, interpreted as binary relations on such a domain (e.g. `friendOf` or `authorOf`). Several constructors are available for building complex concept descriptions, starting from atomic concepts and roles. Domain objects are represented by *individuals* $N_I = \{a, b, \ldots\}$, each associated to a domain entity (such as a person in a social network, or an article in a citation network).

A DL knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is composed of two main components: a *TBox* $\mathcal{T}$, which contains terminological axioms (i.e. concept and role definitions, and the relations between them), and an *ABox* $\mathcal{A}$, which contains ground axioms (or *assertions*) about individuals. In the following, we will denote as $\mathsf{Ind}(\mathcal{K})$ the set of individuals occurring in $\mathcal{K}$.

It is possible to perform several inference tasks with DL KBs. *Instance Checking* consists in deciding whether $\mathcal{K} \models Q(a)$ holds, Q is a query concept and a is an individual. Because of the *Open-World Assumption*, instance checking may provide three possible outcomes, i.e. *a)* $\mathcal{K} \models Q(a)$, *b)* $\mathcal{K} \models \neg Q(a)$ and *c)* $\mathcal{K} \not\models Q(a) \wedge \mathcal{K} \not\models \neg Q(a)$. This means that failing to deductively infer the membership of an individual a to a concept Q in a KB $\mathcal{K}$ does not necessarily imply that a is a member of its complement $\neg Q$.

It is also possible to express more complex queries: given a (infinite) set of variables N_V, a *Conjunctive Query* q is a conjunction of concept and role atoms ($C(v)$ or $R(v, v')$, with $v, v' \in N_V \cup N_I$) built on the signature of $\mathcal{K}$. The set of its variables $\textsc{Var}(q)$ is composed of *answer variables* and (existentially) *quantified variables*. Informally, a binding of the variables w.r.t. some model of $\mathcal{K}$ determines the satisfiability of a query and a result via the answer variables values. $\mathcal{K} \models q$ denotes the satisfiability of q w.r.t. all models of $\mathcal{K}$.

We specifically consider KBs in the OWL 2 language[1] that is the standard representation language in the SW and is supported by DLs in its theoretical foundations. In OWL 2, concepts and roles are referred to as *classes* and *properties*, respectively.

Problem Definition

The problem we aim to tackle is predicting missing classes and property assertions for a given set of individuals in a SW KB. For this purpose, we cast it as a *transductive classification* problem [?].

In the transductive learning setting, one is given a (labeled) training set and an (unlabeled) test set of examples. The goal is to predict the labels only for the (known) test examples, without generalizing to unseen examples. This is different from the *inductive learning* setting, where the goal is to create a general model which can be used also for unseen examples. The approach in this article uses transductive learning but it is generalized to unseen examples by using a very simple method outlined in [?, Ch. 11].

We cast the problem of predicting concept/property assertions as a *binary classification* problem. Entities (represented by individuals in the KB) for which the value of a property (such as *American* or *not American*) can be observed [2] are considered as *labeled examples.* If the value cannot be observed, they are considered as *unlabeled examples.*

We propose a method for learning a *discriminant function* (also referred to as *labeling function*) such that, given an example (either labeled or unlabeled), it predicts its binary class, either positive or negative. Formally, the learning problem is defined as follows:

Definition 1 (Transductive Classification)
Given: A set of examples $X \subseteq \mathsf{Ind}(\mathcal{K})$, partitioned into:

– *positive examples* X_+,
– *negative examples* X_-,
– *neutral* (*unlabeled*) *examples* X_0.

Find: A *discriminant function* $\mathbf{f}^* : X \to \{+1, -1\}$ defined over examples X, where $+1$ corresponds to the *positive* class, and -1 to the *negative* class.

In particular, we aim at finding a discriminant function such that:

1. It is coherent with training labels, i.e. to what we know about the examples, and
2. It is likely to assign similar labels to similar examples, and dissimilar labels to dissimilar examples.

Similarity and dissimilarity relations between examples are extracted from the KB, by identifying *homophilic* and *heterophilic* relations holding between them.

Local Closed World Assumption

As highlighted at the beginning of this section, in SW knowledge bases it is not always possible to obtain *negative examples* for a given property (such as *not American*). This is mainly due to the Open-World Assumption and, sometimes, to the limited expressiveness of

[1] OWL 2 W3C Recommendation: `http://www.w3.org/TR/owl-overview/`

[2] Plase note that class memberships can be regarded as *type*, or *is-a*, properties: the assertion "x is American" can be encoded as American(x) or as nationality(x, American).

some SW knowledge representation formalisms (such as RDF Schema), which do not allow proving negated statements.

A possible solution to this problem consists in resorting to a heuristic called the *Local Closed World Assumption* (LCWA) [?, ?] where the idea is to consider the knowledge about a specific property $\mathtt{r}$ (e.g. `nationality`) of an individual a to be *locally complete* if a value for $\mathtt{r}$ is already specified for the individual a.

More formally, given a KB $\mathcal{K}$, let $O(a, \mathtt{r}) = \{b \mid b \in \mathsf{Ind}(\mathcal{K}) \wedge \mathcal{K} \models \mathtt{r}(a, b)\}$ denote the set of individuals in $\mathcal{K}$ related to a given individual a by a given role $\mathtt{r}$. Given a candidate assertion $\mathtt{r}(a, c)$, its truth value (according to the LCWA) is assigned as follows: if $c \in O(a, \mathtt{r})$, we say the assertion is *correct.* If $c \notin O(a, \mathtt{r})$ and $|O(a, \mathtt{r})| \geq 1$, we say the assertion is *not correct*, since we assume that the KB $\mathcal{K}$ is *locally complete* for the $\langle a, \mathtt{r} \rangle$ pair. If $O(a, \mathtt{r}) = \emptyset$, we assume the truth value for the assertion $\mathtt{r}(a, c)$ is *missing*, and can be predicted by using assertion prediction methods.

This strategy of collecting negative examples by assuming *local completeness* is also adopted in [?, ?], two related works on assertion prediction. In the empirical evaluation in Sect. 8, we reproduced the experimental setting presented in [?, ?], which rely on the LCWA for collecting negative examples for a given property.

Example 2 (Academic Domain – cont.) Following Ex. 1, consider the task of predicting whether Lucas is affiliated to the *Efficient Algorithms* (`EffAlg`) research group. It can be cast as an *transductive classification* task, where Mark is a positive example (e.g. because we know that affiliatedTo(`Mark`, `EffAlg`) holds true) and Lucas an unlabeled example w.r.t. the membership to `EffAlg`, i.e. $X_+ = \{\mathtt{Mark}\}$, $X_0 = \{\mathtt{Lucas}\}$ whilst there are no negative examples for `EffAlg`.

However, since it is known that John is a member of the `CoM` group, we can resort to the LCWA and assume that the knowledge about its research group membership is *locally complete*: this allows considering John as a negative example, i.e. $X_- = \{\mathtt{John}\}$. Then, given $X_+ = \{\mathtt{Mark}\}$, $X_- = \{\mathtt{John}\}$ and $X_0 = \{\mathtt{Lucas}\}$, finding whether Lucas is a member of `EffAlg` can be cast to finding a discriminant function $\mathbf{f}^* : X \to \{-1, +1\}$, which associates a binary class (positive or negative) to all researchers, depending on the *predicted* value of their affiliation to the `EffAlg` research group. ■

It is also possible to use other strategies for collecting negative examples. For instance, *a*) one might ask *human experts* to provide for negative examples explicitly, or *b*) add logical axioms to the knowledge base (such as disjointness axioms), so to identify assertions whose truth value is *false.*

In the following section, we discuss how an *influence graph*, defined over examples in X and modeling their mutual influence relations, can be used to define a *penalty term* (i.e. a *cost function*) over discriminant functions. This is needed for encoding our preference towards discriminant functions which are both consistent with labeled examples (i.e. what we know about the world), and assign similar labels to similar examples, and different labels to dissimilar ones.

3 A Penalty Term over Discriminant Functions using Similarity and Dissimilarity Relations

In this section, we show how similarity and dissimilarity relations between examples in X can be used for defining a *penalty term* (or *cost function*) over discriminant functions. In Sect. 4 we show how a global minimum for such a penalty term can be calculated very efficiently, and in Sect. 5 we show how similarity and dissimilarity relations can be learned from the KB.

Let X be a set of $n \triangleq |X|$ examples, of which $l \triangleq |X_+ \cup X_-|$, with $l \leq n$, are labeled.

Graph-based SSL methods rely on a weighted graph defined over examples in X, which is referred to as *similarity graph.* If two examples are connected in the similarity graph, that means that they are considered *similar*, and should be assigned the same label. However, not all similar examples might be connected by an edge in the similarity graph: the underlying idea is that the similarity graph can be used for *propagating* label information from a small set of labeled examples to all the examples in the similarity graph.

The similarity graph is encoded by its non-negative symmetric adjacency (weight) matrix $\mathbf{W}^+ \in \mathbb{R}^{n \times n}_{\geq 0}$, where $\mathbf{W}^+_{ij} \geq 0$ is the edge weight between examples x_i and x_j, with $x_i, x_j \in X$. Specifically, $\mathbf{W}^+_{ij} > 0$ iff $x_i, x_j \in X$ are connected by an edge in the similarity graph.

Following [?], we can encode our assumption that *similar examples should be assigned similar labels* by defining a quadratic *penalty term* (or *cost function*) over discriminant functions in the form $\mathbf{f} : X \to \{+1, -1\}$:

$$E^+(\mathbf{f}) \triangleq \frac{1}{2} \sum_{x_i \in X} \sum_{x_j \in X} \mathbf{W}^+_{ij} \left[\mathbf{f}(x_i) - \mathbf{f}(x_j)\right]^2. \qquad (1)$$

Given an input discriminant function $\mathbf{f}$, the penalty term in Eq. 1 associates, for each pair of examples $x_i, x_j \in X$, a non-negative penalty $\mathbf{W}^+_{ij} \left[\mathbf{f}(x_i) - \mathbf{f}(x_j)\right]^2$: this quantity is 0 when $\mathbf{W}^+_{ij} = 0$ (i.e. x_i and x_j are not linked in the similarity graph) or when $\mathbf{f}(x_i) = \mathbf{f}(x_j)$

(i.e. they are assigned the same label). For such a reason, the penalty term in Eq. 1 favors discriminant functions that are more likely to assign the same labels to examples that are linked by an edge in the similarity graph, encoded by its adjacency matrix $\mathbf{W}^{+}$.

Exploiting Dissimilarity Relations

As for similarity relations, we assume that *dissimilarity relations* are encoded by a weighted graph defined over examples in X, which we will refer to as *dissimilarity graph*. Also the dissimilarity graph is represented by its non-negative symmetric adjacency (weight) matrix $\mathbf{W}^{-} \in \mathbb{R}^{n \times n}_{\geq 0}$, where $\mathbf{W}^{-}_{ij} = \mathbf{W}^{-}_{ji} \geq 0$ is the edge weight between two examples x_i and x_j, with $x_i, x_j \in X$: $\mathbf{W}^{-}_{ij} > 0$ iff $x_i, x_j \in X$ are connected by an edge in the dissimilarity graph. In this case, *dissimilar* examples are connected by edges with large weights.

Following [?], we can encode our assumption that *dissimilar examples tend to be associated to dissimilar labels* by defining a quadratic *penalty term* over discriminant functions in the form $\mathbf{f} : X \to \{+1, -1\}$:

$$E^{-}(\mathbf{f}) \triangleq \frac{1}{2} \sum_{x_i \in X} \sum_{x_j \in X} \mathbf{W}^{-}_{ij} \left[\mathbf{f}(x_i) + \mathbf{f}(x_j)\right]^2 . \tag{2}$$

Given an input discriminant function $\mathbf{f}$, the penalty term in Eq. 2 associates, for each pair of examples $x_i, x_j \in X$, a non-negative penalty $\mathbf{W}^{-}_{ij} \left[\mathbf{f}(x_i) + \mathbf{f}(x_j)\right]^2$: this quantity is 0 when $\mathbf{W}^{-}_{ij} = 0$ (i.e. x_i and x_j are not linked in the dissimilarity graph) or when $\mathbf{f}(x_i) \neq \mathbf{f}(x_j)$. This means that penalty term in Eq. 2 favors (by assigning a low penalty) discriminant functions that are likely to assign the different labels to examples that are linked by an edge with a large weight in the dissimilarity graph, encoded by its adjacency matrix $\mathbf{W}^{-}$.

Merging Similarity and Dissimilarity Relations in a Single Influence Graph

An intuitive way of combining the two penalty terms over discriminant functions $\mathbf{f} : X \to \{+1, -1\}$ defined in Eq. 1 and Eq. 2 is based on the following intuition: we can encode both similarity and dissimilarity relations between examples in X in a single weighted graph, which we will refer to as *influence graph*.

In the influence graph, represented by its symmetric adjacency (weight) matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$, edges with strictly *positive* weights are associated with *similarity* relations, while edges with strictly *negative* weights are associated with *dissimilarity* relations. This leads to the definition of the following penalty term over discriminant functions in the form $\mathbf{f} : X \to \{+1, -1\}$:

$$\begin{aligned} E(\mathbf{f}) &\triangleq \frac{1}{2} \sum_{x_i \in X} \sum_{x_j \in X} \begin{cases} \mathbf{W}_{ij}[\mathbf{f}(x_i) - \mathbf{f}(x_j)]^2 & \text{if } \mathbf{W}_{ij} \geq 0 \\ -\mathbf{W}_{ij}[\mathbf{f}(x_i) + \mathbf{f}(x_j)]^2 & \text{if } \mathbf{W}_{ij} < 0 \end{cases} \\ &= \mathbf{f}^T(\mathbf{D} - \mathbf{W})\mathbf{f} \\ &= \mathbf{f}^T \mathbf{M} \mathbf{f}, \end{aligned} \tag{3}$$

where $\mathbf{f}$ is a finite-size vector representation of the labeling function $\mathbf{f} = [\mathbf{f}(x_1), \ldots, \mathbf{f}(x_n)]^T$, where $\mathbf{f}_i = \mathbf{f}(x_i)$ is the label assigned to the i-th example $x_i \in X$, $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} \triangleq \sum_{x_j \in X} |\mathbf{W}_{ij}|$, and $\mathbf{M} \triangleq \mathbf{D} - \mathbf{W}$.

In the penalty term defined in Eq. 3, each edge in the influence graph between two examples $x_i, x_j \in X$ is considered as a *similarity relation* if its weight $\mathbf{W}_{ij}$ is strictly positive, and as a *dissimilarity relation* if its weight is strictly negative. If $\mathbf{W}_{ij} = 0$, it means there is no edge between the two examples x_i and x_j. For such a reason, the influence graph can be particularly convenient to interpret: it suffices to examine the sign of the weight associated to each edge for estimating whether it encodes a similarity or dissimilarity relationship.

Note that if all weights in $\mathbf{W}$ are positive, then $\mathbf{M}$ is the *graph Laplacian* of the graph whose adjacency matrix is $\mathbf{W}$. Also note that, unlike the penalty functions in Eq. 1 and Eq. 2 (which only consider either similarity or dissimilarity relations), non-zero elements in the adjacency matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ can take both positive and negative values, depending on whether they represent a similarity or dissimilarity relation.

In the following section, we show how the influence graph can be used for efficiently *propagating* label information from a limited set of labeled examples to all examples in X. Here we assume the influence graph is already given: a method for learning the influence graph from data by leveraging the knowledge base is proposed in Sect. 5 and 6.

4 Transductive Classification as a Convex Optimization Problem

The penalty term in Eq. 3 allows to encode our preference for discriminant functions assigning similar labels to similar examples, and different labels to dissimilar examples.

In the following, we assume we are provided with an *influence graph*, defined over examples in X, which encodes similarity and dissimilarity relations, and is represented by its adjacency matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$. In the influence graph, each edge encodes an *influence* (i.e. either a similarity or a dissimilarity) relation between a pair of examples. Specifically, if $\mathbf{W}_{ij} > 0$, examples $x_i, x_j \in X$ are linked by a *similarity* relation; if $\mathbf{W}_{ij} < 0$, x_i and x_j are linked by a *dissimilarity* relation.

Let $L \triangleq X_+ \cup X_-$ and $U \triangleq X_0$ denote labeled and unlabeled examples, respectively. The problem of finding an optimal discriminant function $\mathbf{f}^*$ can be cast as an optimization problem: given a labeling for labeled examples L, we want to find a discriminant function that: *a*) is consistent with training labels, and *b*) minimizes the penalty term $E(\cdot)$ in Eq. 3.

The problem of finding the optimal discriminant function $\mathbf{f}^*$ can be formalized as follows:

$$\begin{aligned} \underset{\mathbf{f} \in \{+1,-1\}^n}{\text{minimize}} \quad & E(\mathbf{f}) \\ \text{subject to} \quad & \forall x \in L : \ \mathbf{f}_i = \mathbf{y}_i, \end{aligned} \tag{4}$$

where $\mathbf{y} \in \{-1, 0, +1\}^n$ is a *label vector* containing labels for labeled examples, such that $\mathbf{y}_i = +1$ (resp. $\mathbf{y}_i = -1$) if $x_i \in X_+$ (resp. $x_i \in X_-$), and $\mathbf{y}_i = 0$ otherwise.

The optimization problem in Eq. 4 aims at minimizing the penalty term $E(\cdot)$, as proposed in Eq. 3, so to favor discriminant functions that are consistent with the influence graph. The constraint $\forall x \in L : \mathbf{f}_i = \mathbf{y}_i$ enforces the label of each labeled example $x_i \in L$ to $\mathbf{f}_i = +1$ if it is a positive example, and to $\mathbf{f}_i = -1$ if it is a negative example, so to achieve consistency with training labeled examples.

However, constraining discriminant functions $\mathbf{f}$ to only take discrete values (i.e. $\mathbf{f}(x) \in \{+1, -1\}, \forall x \in X$) has two main drawbacks [?]:

- Each discriminant function $\mathbf{f}$ can only provide *hard* classifications (i.e. $\mathbf{f} \in \{+1, -1\}^n$), without yielding any measure of confidence in the provided classification.
- The penalty term $E(\cdot)$ in Eq. 3 defines the energy function for a discrete Markov Random Field [3], and calculating the marginal distribution over labels of unlabeled examples $\mathbf{f}_U$ can be inherently difficult [?].

For overcoming the aforementioned limitations, in [?] authors propose a continuous relaxation of the discrete optimization problem in Eq. 4, by allowing the discriminant function $\mathbf{f}$ to yield *real values* as possible outcomes – i.e. $\mathbf{f}(x) \in [-1, +1]$ instead of $\mathbf{f}(x) \in \{+1, -1\}$, for every $x \in X$.

More formally, the relaxed version of the optimization problem in Eq. 4 can be defined as follows:

$$\begin{aligned} \underset{\mathbf{f} \in [-1,1]^n}{\text{minimize}} \quad & E(\mathbf{f}) + \epsilon \sum_{x_i} \mathbf{f}_i^2 \\ \text{subject to} \quad & \forall x \in L : \ \mathbf{f}_i = \mathbf{y}_i, \end{aligned} \tag{5}$$

where the term $\sum_{x_i \in X} \mathbf{f}_i^2 = \mathbf{f}^T\mathbf{f}$ is a L_2 regularizer over $\mathbf{f}$, weighted by a parameter $\epsilon > 0$ which ensures that the optimization problem has a unique global solution. The parameter ϵ can be interpreted as the *decay* of the propagation process: as the distance from a labeled example within the influence graph increases, the confidence in the classification (as measured by the continuous label) gets closer to zero. An in-depth analysis of such a relaxation (in terms of probabilistic models, or harmonic functions) can be found in [?] and [?].

Without loss of generality, assume that the vectors $\mathbf{f}$ and $\mathbf{y}$, and the matrices $\mathbf{W}$ and $\mathbf{M}$ are partitioned w.r.t. the membership of examples to the sets of labeled examples L and unlabeled examples U, i.e. $\mathbf{f} = [\mathbf{f}_L, \mathbf{f}_U]^T \in \mathbb{R}^n$, $\mathbf{y} = [\mathbf{y}_L, \mathbf{y}_U]^T \in \mathbb{R}^n$,

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_{LL} & \mathbf{W}_{LU} \\ \mathbf{W}_{UL} & \mathbf{W}_{UU} \end{bmatrix}, \ \mathbf{M} = \begin{bmatrix} \mathbf{M}_{LL} & \mathbf{M}_{LU} \\ \mathbf{M}_{UL} & \mathbf{M}_{UU} \end{bmatrix}. \tag{6}$$

The optimization problem in Eq. 5 has a unique, global solution that can be calculated in closed-form. Specifically, the optimal (relaxed) discriminant function $\mathbf{f}^* : X \to \mathbb{R}$ is given by $\mathbf{f}^* = [\mathbf{f}_L^*, \mathbf{f}_U^*]^T$, where $\mathbf{f}_L^* = \mathbf{y}_L$ (i.e. labels for labeled examples in L coincide with training labels), while $\mathbf{f}_U^*$ is given by:

$$\mathbf{f}_U^* = (\mathbf{M}_{UU} + \epsilon\mathbf{I})^{-1}\mathbf{W}_{UL}\mathbf{f}_L^*. \tag{7}$$

Note that the optimal discriminant function, in this context, can take continuous values. Given an example $x \in X$, the labeling is given by $\text{sgn}(\mathbf{f}^*(x))$, where $\text{sgn}(\cdot)$ denotes the *sign* function: $\mathbf{f}^*(x) \approx +1$ (resp. $\mathbf{f}^*(x) \approx -1$) means a high confidence that the example is in the positive (resp. negative) class, while $\mathbf{f}^*(x) \approx 0$ denotes a very low confidence in the labeling.

Complexity.
A solution for Eq. 7 can be computed efficiently in nearly-linear time w.r.t. the number of edges in the influence graph. Indeed computing $\mathbf{f}_U^*$ can be reduced to solving a linear system in the form $\mathbf{A}\mathbf{x} = \mathbf{b}$, with $\mathbf{A} = (\mathbf{M}_{UU} + \epsilon\mathbf{I})$, $\mathbf{b} = \mathbf{W}_{UL}\mathbf{f}_L^*$ and $\mathbf{x} = \mathbf{f}_U^*$.

A linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{A} \in \mathbb{R}^{n \times n}$ can be solved in nearly-linear time w.r.t. the number of non-zero elements in $\mathbf{A}$ if the coefficient matrix $\mathbf{A}$ is *Symmetric and Diagonally Dominant* (SDD) [4]. In Eq. 7, the matrix $(\mathbf{M}_{UU} + \epsilon\mathbf{I})$ is SDD: this follows from the fact that $\mathbf{M}_{UU}$ is a principal submatrix of $\mathbf{M}$, which is SDD by construction. However, it is important to note that, if the influence graph is *fully connected* or very dense, the number of edges in the influence graph grows *quadratically* with the number of examples.

[3] For instance, see the probabilistic interpretation of the penalty term in the end of this section.

[4] A matrix $\mathbf{A}$ is SDD iff $\mathbf{A}$ is symmetric (i.e. $\mathbf{A} = \mathbf{A}^T$) and $\forall i : \mathbf{A}_{ii} \geq \sum_{i \neq j} |\mathbf{A}_{ij}|$.

An efficient algorithm for solving SDD linear system is proposed in [?]: it has a $\approx \mathrm{O}\big(m \log^{\frac{1}{2}} n\big)$ time complexity, where m is the number of non-zero entries in $\mathbf{A}$ and n is the number of variables in the system of linear equations. An efficient parallel solver for SDD linear systems is proposed in [?]

Probabilistic Interpretation

The propagation model proposed in this section can be considered as a *probabilistic graphical model*: each continuous label is represented by a Gaussian variable, and the interactions between labels are modeled through a Markov network, where each edge corresponds to an edge in the influence graph.

In particular, the objective function in Eq. 5 associates a penalty, or *cost*, to an input (relaxed) discriminant function $\mathbf{f} \in \mathbb{R}^n$. Following [?], $E(\mathbf{f}) + \epsilon \mathbf{f}^T \mathbf{I} \mathbf{f}$ can be interpreted as the *energy function* of the following probability density p over discriminant functions:

$$\begin{aligned} p(\mathbf{f}) &= (2\pi)^{-\frac{1}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2} \left[E(\mathbf{f}) + \epsilon \mathbf{f}^T \mathbf{I} \mathbf{f} \right] \right\} \\ &= \mathcal{N}\Big(\mathbf{0}, (\mathbf{M} + \epsilon \mathbf{I})^{-1}\Big). \end{aligned} \tag{8}$$

The probability density function in Eq. 8 defines a finite-set Gaussian process [?] $\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Omega} = (\mathbf{M} + \epsilon \mathbf{I})$ and $\boldsymbol{\Sigma} = \boldsymbol{\Omega}^{-1}$ are respectively its *inverse covariance* (or *precision*) and *covariance* matrix, and $|\boldsymbol{\Sigma}|$ indicates the determinant of $\boldsymbol{\Sigma}$. The covariance matrix and its inverse fully determine the independence relations among variables in a multivariate Gaussian distribution [?]. In particular, if $\boldsymbol{\Omega}_{ij} \neq 0$, then there is an edge between variables $\mathbf{f}_i$ and $\mathbf{f}_j$ in the minimal I-map Gaussian Markov Random Field of the probability density p.

Finding the most likely discriminant function over unlabeled examples $\mathbf{f}_U^*$ given the labels $\mathbf{f}_L^* = \mathbf{y}_L$ for labeled examples in L (see Eq. 7) is equivalent to finding the expected value for $\mathbf{f}_U$ given $\mathbf{f}_L = \mathbf{y}_L$ according to p:

$$\mathbf{f}_U^* = \mathbb{E}\left[\mathbf{f}_U \mid \mathbf{f}_L = \mathbf{y}_L\right] = (\mathbf{M}_{UU} + \epsilon \mathbf{I})^{-1} \mathbf{W}_{UL} \mathbf{f}_L^*.$$

5 Identifying Similarity and Dissimilarity Relations

In Sect. 4 we show how, given an *influence graph* defined over a set of examples X, we can efficiently find an optimal (relaxed) discriminant function $\mathbf{f}^* : X \to [-1, +1]$. In this section we discuss how we can exploit the relations holding between examples X in the KB for learning the influence graph.

Recall that the underlying assumption in this work is that *related individuals influence each other*. In particular, some relations between example in the KB might encode influence relations w.r.t. a specific target set of properties and classes: identifying such relations can be used for propagating information among examples, and provide new knowledge about the domain.

In the literature, this phenomenon is also referred to as *homophily* (love of the same) and *heterophily* (love of the different). A relation between examples (such as *friendship* in a social network) can be correlated with those examples being similar, or dissimilar, w.r.t. a set of properties, such as political views, hobbies, and religious beliefs. Homophily and heterophily are known to occur in a vast array of networked domains [?, ?].

However, depending on the considered properties, not all relations encode the same type of influence between examples. For instance, in a social network, friends may tend to share common interests (homophily), while quiet people may tend to prefer talkative friends and vice-versa (heterophily) [?].

Combining Multiple Influence Graphs

For identifying homophilic and heterophilic relations, we proceed as follows. For each relation type $\mathtt{rel}_i \in \{\mathtt{rel}_1, \ldots, \mathtt{rel}_r\}$, each representing a distinct binary predicate (such as `friendOf` or `advisorOf`), we create a corresponding *relational influence graph*, encoded by its adjacency matrix $\mathbf{R}_i \in \{0, 1\}^{n \times n}$. Each matrix $\mathbf{R}$, associated to a relation type `rel`, is structured as follows: $\mathbf{R}_{ij} = 1$ iff $\mathcal{K} \models \mathtt{rel}(x_i, x_j)$ or $\mathcal{K} \models \mathtt{rel}(x_j, x_i)$.

Note that, while the relation type `rel` may be *directed*, the corresponding influence graph (encoded by the adjacency matrix $\mathbf{R}$) is *undirected*, since pairwise similarity and dissimilarity relations between examples in X encoded by an influence graph are undirected by nature.

Following our intuition that *relations can be homophilic, heterophilic or none of the above* with respect to a given set of properties or classes, we propose the following model. Given a set of (relational) influence graphs $\mathcal{R} \triangleq \{\mathbf{R}_1, \ldots, \mathbf{R}_r\}$, according to the assumption that relations do not encode the same kind of influence between examples, we define the weight matrix $\mathbf{W}$ as a linear combination of the adjacency matrices in $\mathcal{R}$:

$$\mathbf{W} \triangleq \sum_{i=1}^{r} \mu_i \mathbf{R}_i, \qquad \text{with } \mu_i \in \mathbb{R}, \forall i \tag{9}$$

where each μ_i is a parameter representing the contribution of the matrix $\mathbf{R}_i$ in the construction of $\mathbf{W}$.

Note that the proposed model is particularly convenient to interpret: *homophilic* relation types are associated to strictly positive weights, *heterophilic* relation types are associated to strictly negative weight, and remaining relations are associated to a weight close to zero (thanks to the regularization term in Eq. 11). For such a reason, only checking the sign of the weight μ associated to a particular relation type to know whether it is homophilic or heterophilic w.r.t. a given property (such as nationality). In the following, we propose a solution to the problem of efficiently learning the parameters $\boldsymbol{\Theta} \triangleq \{\boldsymbol{\mu}, \epsilon\}$ from a set of examples X.

Parameters Learning

The parametric form of the influence graph $\mathbf{W}$ is fully specified by the parameters $\boldsymbol{\mu}$ in Eq. 9, which reflect whether each relation used in the construction of $\mathbf{W}$ is homophilic (links similar examples), heterophilic (links dissimilar examples), or none of the above.

In addition, the approach in Sect. 4 depends on the choice of a regularization parameter ϵ. In this work, we propose learning the parameters $\boldsymbol{\Theta} = \{\boldsymbol{\mu}, \epsilon\}$ by minimizing the *Leave-One-Out* (LOO) *Prediction Error* [?]. Provided that the propagation step can be performed efficiently (as proven in Sect. 4), we can compute the LOO Error for a set of examples X. The LOO Error is defined as the summation of reconstruction errors obtained by considering each labeled example, in turn, as unlabeled, and predicting its label.

More formally, let $U_i \triangleq U \cup \{x_i\}$ and $L_i \triangleq L - \{x_i\}$: w.l.o.g. we assume that the label of the left-out example $x_i \in L$ is in the first position of the new real valued labeling vector $\mathbf{f}_{U_i}$. Let $\ell(x, \hat{x})$ be a generic, differentiable loss function (e.g. $\ell(x, \hat{x}) = |x - \hat{x}|$ for the absolute loss, or $\ell(x, \hat{x}) = (x - \hat{x})^2/2$ for the quadratic loss). The LOO Error is defined as follows:

$$\mathcal{L}(\boldsymbol{\Theta} \mid \mathbf{f}_L) \triangleq \sum_{i=1}^{|L|} \ell(\mathbf{f}_i, \hat{\mathbf{f}}_i), \tag{10}$$

where $\mathbf{e}^T \triangleq (1, 0, \ldots, 0) \in \mathbb{R}^{u+1}$ and $\hat{\mathbf{f}}_i \triangleq \mathbf{e}^T(\mathbf{M}_{U_iU_i} + \epsilon\mathbf{I})^{-1}\mathbf{W}_{U_iL_i}\mathbf{f}_{L_i}$ represents the continuous label value assigned to x_i as if such a value was not known in advance. The vector $\mathbf{e}^T$ is needed to select the predicted label for the left-out example $x_i \in L$, which is assumed to be in the first position.

We can now define the following criterion for learning the parameters $\boldsymbol{\Theta}$:

Definition 2 (Minimum LOO Error Params.) Given a set of labeled (resp. unlabeled) examples L (resp. U) and a set of (relational) influence graphs $\mathcal{R}$, each corresponding to a relation type, the *minimum LOO Error Parameters* $\boldsymbol{\Theta}^*_{LOO}$ can be learned by solving the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{\Theta}}{\text{minimize}} \quad & \mathcal{L}(\boldsymbol{\Theta}) + \lambda ||\boldsymbol{\Theta}||_2^2 \\ \text{subject to} \quad & \epsilon > 0, \end{aligned} \tag{11}$$

where the function $\mathcal{L}$ is defined as in Eq. 10, and $\lambda > 0$ weights a L_2 regularization term over $\boldsymbol{\Theta}$, which controls the complexity of parameters $\boldsymbol{\Theta}$ [?]. In the following, we refer to the influence graph constructed by using the parameters in $\boldsymbol{\Theta}^*_{LOO}$ as the *optimal* influence graph.

The objective function in Def. 2 is differentiable and can be efficiently minimized by using gradient-based function minimization approaches, such as Gradient Descent, L-BFGS, or a stochastic optimization algorithm.

The gradient of the LOO Error $\mathcal{L}$ with respect to the parameters $\boldsymbol{\Theta}$ can be calculated as follows. Let $\mathbf{Z}_i = (\mathbf{M}_{U_iU_i} + \epsilon\mathbf{I})$: the gradient of $\mathcal{L}$ w.r.t. a parameter $\theta \in \boldsymbol{\Theta}$ is given by:

$$\begin{aligned} \frac{\partial \mathcal{L}(\boldsymbol{\Theta} \mid \mathbf{f}_L)}{\partial \theta} &= \sum_{i=1}^{|L|} \frac{\partial \ell(\mathbf{f}_i, \hat{\mathbf{f}}_i)}{\partial \hat{\mathbf{f}}_i} \left(\mathbf{e}^T \mathbf{Z}_i^{-1} \mathbf{z}_i\right), \\ \text{with } \mathbf{z}_i &= \left(\frac{\partial \mathbf{W}_{U_iL_i}}{\partial \theta}\mathbf{f}_{L_i} - \frac{\partial \mathbf{Z}_i}{\partial \theta}\mathbf{f}^*_{U_i}\right). \end{aligned} \tag{12}$$

Deriving the gradient is straightforward. The gradient of $\mathcal{L}$ w.r.t. a parameter $\theta \in \boldsymbol{\Theta}$ follows by the chain rule, and $\partial \ell$ depends on the choice of the loss function ℓ. The gradient of $\hat{\mathbf{f}}_i$ w.r.t. θ can be derived as follows:

$$\begin{aligned} \frac{\partial \hat{\mathbf{f}}_i}{\partial \theta} &= \frac{\partial}{\partial \theta}\left(\mathbf{e}^T \mathbf{Z}_i^{-1} \mathbf{W}_{U_iL_i}\mathbf{f}_{L_i}\right) \\ &= \mathbf{e}^T \mathbf{Z}_i^{-1}\left(\frac{\partial \mathbf{W}_{U_iL_i}}{\partial \theta}\mathbf{f}_{L_i} - \frac{\partial \mathbf{Z}_i}{\partial \theta}\mathbf{Z}_i^{-1}\mathbf{W}_{U_iL_i}\mathbf{f}_{L_i}\right) \\ &= \mathbf{e}^T \mathbf{Z}_i^{-1}\left(\frac{\partial \mathbf{W}_{U_iL_i}}{\partial \theta}\mathbf{f}_{L_i} - \frac{\partial \mathbf{Z}_i}{\partial \theta}\mathbf{f}^*_{U_i}\right), \end{aligned}$$

using the properties $\partial(\mathbf{X}^{-1}) = -\mathbf{X}^{-1}(\partial\mathbf{X})\mathbf{X}^{-1}$ and $\partial(\mathbf{XY}) = \mathbf{X}(\partial\mathbf{Y}) + (\partial\mathbf{X})\mathbf{Y}$.

Minimizing the LOO Error.
For solving the optimization problem in Eq. 11 by means of gradient-based optimization techniques, we propose using *Projected Sub-Gradient Descent* [?], with an intermediate line search to assess the step size.

Let $\mathcal{S}_{\boldsymbol{\Theta}} = \{\boldsymbol{\Theta} = \{\boldsymbol{\mu}, \epsilon\} \mid \boldsymbol{\mu} \geq \mathbf{0}, \epsilon > 0\}$ denote the space of valid parameter values for the set of parameters $\boldsymbol{\Theta}$. The algorithm is briefly summarized as follows:

A key component in Alg. 1 is the *projection operator* $P_{\mathcal{S}_{\boldsymbol{\Theta}}}[\tilde{\boldsymbol{\Theta}} = \{\tilde{\boldsymbol{\mu}}, \tilde{\epsilon}\}]$, which calculates the Euclidean projection of $\tilde{\boldsymbol{\Theta}}$ into the set of *valid* parameters $\mathcal{S}_{\boldsymbol{\Theta}}$. This is

ALGORITHM 1: Projected Gradient Descent for Minimum LOO Error Parameters Learning

```
Input: Training Labels $\mathbf{f}_L$, Adjacency Matrices $\mathcal{W}$
Output: Minimum LOO Error Parameters $\boldsymbol{\Theta}^*_{LOO}$.
// Randomly initialize parameters:
$\boldsymbol{\Theta}^{(0)} \leftarrow Init()$
for $t = 1, \ldots, \tau$ do
    // Gradient Descent Step:
    $\tilde{\boldsymbol{\Theta}}^{(t)} \leftarrow \boldsymbol{\Theta}^{(t-1)} - \eta_t \frac{\partial \mathcal{L}(\boldsymbol{\Theta}^{(t-1)} | \mathbf{f}_L, \mathcal{W})}{\partial \boldsymbol{\Theta}}$
    // Project $\tilde{\boldsymbol{\Theta}}^{(t)}$ in the space of parameters $\mathcal{S}_{\boldsymbol{\Theta}}$:
    $\boldsymbol{\Theta}^{(t)} = P_{\mathcal{S}_{\boldsymbol{\Theta}}}[\tilde{\boldsymbol{\Theta}}^{(t)}] = \arg\min_{\boldsymbol{\Theta} \in \mathcal{S}_{\boldsymbol{\Theta}}} ||\tilde{\boldsymbol{\Theta}}^{(t)} - \boldsymbol{\Theta}||_2$
end
$\boldsymbol{\Theta}^*_{LOO} \leftarrow \boldsymbol{\Theta}^{(\tau)}$
return $\boldsymbol{\Theta}^*_{LOO}$;
```

especially useful for enforcing constraints during learning, such as $\epsilon > 0$ or $\mu_i \geq 0$ in case we only need to identify *similarity relations* during learning (recall that edges with non-negative weights in the *influence graph* correspond to similarity relations).

In our implementation, $P_{\mathcal{S}_{\boldsymbol{\Theta}}}$ returns a $\boldsymbol{\Theta} = \{\boldsymbol{\mu}, \epsilon\}$ such that $\epsilon = \max\{\xi, \tilde{\epsilon}\}$, where $\xi > 0$ is a small, strictly positive scalar (in empirical evaluations, we use $\xi = 10^{-6}$): this is useful to enforce the constraint that $\epsilon > 0$.

At each iteration, we assess the step size η_t using a simple *line search*, choosing the value of η_t that provides the larger reduction of the LOO Error $\mathcal{L}$. The iterative process continues until no further improvement is possible.

For investigating the feasibility of learning the parameters using gradient-based optimization methods, we now analyze the complexity of calculating the gradient of the LOO Error $\mathcal{L}$ w.r.t. parameters in $\boldsymbol{\Theta}$.

Complexity.
Calculating the LOO Error $\mathcal{L}(\boldsymbol{\Theta})$ in Eq. 10 requires iterating over labeled examples in L. Then, for each labeled example $x_i \in L$, the calculation requires:

- Creating two new sets of labeled $L_i = L \setminus \{x_i\}$ and unlabeled $U_i = U \cup \{x_i\}$ examples.
- Propagating label information from labeled examples in L_i to unlabeled examples in U_i.
- Evaluating the *reconstruction error* $\ell(\mathbf{y}_i, \hat{\mathbf{f}}_i)$ between the real label $\mathbf{y}_i$ of x_i and its predicted label $\hat{\mathbf{f}}_i$.

It follows that the complexity of evaluating the LOO Error is dominated by $|L|$ propagation steps, and it is given by $\mathrm{O}\big(|L| m \log^{\frac{1}{2}} n\big)$ (using the algorithm in [?]), where m is the number of edges in the influence graph between examples in U_i, and $n = |U_i|$.

Similarly, evaluating the gradient of the LOO Error $\mathcal{L}$, as given in Eq. 12, requires iterating over the labeled examples in $x_i \in L$ and, at each iteration, performing a propagation step from examples in L_i to examples in U_i, and computing an additional term $\mathbf{e}^T \mathbf{Z}_i^{-1} \mathbf{z}_i$. Computing $\mathbf{Z}_i^{-1}$ might not be feasible if $\mathbf{Z}_i$ is large, since matrix inversion has a time complexity of approx. $\mathrm{O}\big(n^{2.3727}\big)$. However, once again, note that $\mathbf{Z}_i = (\mathbf{M}_{U_i U_i} + \epsilon \mathbf{I})$ is SDD: computing $\mathbf{Z}_i^{-1} \mathbf{z}_i$ can be reduced to solving a linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ such that $\mathbf{A} = \mathbf{M}_{U_i U_i} + \epsilon \mathbf{I}$ and $\mathbf{b} = \mathbf{z}_i$. It follows that also the complexity of calculating the gradient of the LOO Error is also $\mathrm{O}\big(|L| m \log^{\frac{1}{2}} n\big)$.

Summarizing, the evaluation of both the LOO Error and its gradient has a $\mathrm{O}\big(|L| m \log^{\frac{1}{2}} n\big)$ time complexity, thanks to the efficient algorithm proposed in [?]. However, it could still be not feasible if the number of labeled examples $|L|$ is very large. A possible solution to tackling this problem is replacing the LOO Error with the more general k-fold Cross Validation Error [?], which would limit the number of propagation steps to k. The LOO Error is a specific case of the k-fold Cross Validation Error, with $k = |L|$.

6 Retrieving the Relations Between Examples

As mentioned in Sect. 1, we rely on the relations holding between examples in the KB for constructing the influence graph $\mathbf{W}$. In Eq. 9 we expressed the influence graph $\mathbf{W}$ as a linear combination, with weights $\boldsymbol{\mu}$, of r (relational) influence graphs $\mathcal{R} = \{\mathbf{R}_1, \ldots, \mathbf{R}_r\}$, where $\mathbf{R}_i$ corresponds to the i-th relation type holding between examples in X (such as `friendOf` or `advisorOf`).

In AKP$^{\mathrm{D}}$, for retrieving the relations holding between examples from the KB, we rely on *Conjunctive Queries* (CQs), as introduced in Sect. 2. CQs allow representing a wide variety of relations between entities; for instance, the “co-authorship” relation between $x_i, x_j \in X$ can be expressed by the following CQ:

$$\exists z. \left[\texttt{authorOf}(x_i, z) \land \texttt{authorOf}(x_j, z) \right],$$

where `authorOf` is an atomic role, and $z \in N_V$ is a *non-distinguished* variable representing a work authored by both x_i and x_j.

It is also possible to express more complex relations between examples: for instance, the “co-authorship of an article in the field of Machine Learning (`ML`) which won the best paper award” can be retrieved by means of the following CQ:

$$\exists z. [\texttt{authorOf}(x_i, z) \land \texttt{authorOf}(x_j, z) \land \texttt{BestPaper}(z) \land \texttt{field}(z, \texttt{ML})].$$

However, the whole space of relations that can be expressed by CQs can be too large to be used in practical applications. Following [?], we propose capturing the following two phenomena that can apply in networked domains:

– **Homophily/Heterophily.** A direct link between two entities (such as *friendship* or *supervision*) is correlated with those entities being similar or dissimilar in nature. For instance, friends often tend to be similar in characteristic like age, social background and education level [?, ?], and different in characteristics like talkativeness [?].
– **Co-citation Regularity.** Similar entities tend to refer, or connect, to the same things. For instance, when two persons have similar tastes in music, literature or fashion, co-citation regularity suggests that they may be similar in other ways, or share other common interests [?].

Conjunctive Queries allow expressing even more complex relationships between examples. However, we empirically found that relying only on *homophily*, *heterophily* and *co-citation regularity* can be a balanced trade-off between the predictive accuracy of learned models, their understandability and the efficiency of the parameters learning process.

For such a reason, we rely on two types of Conjunctive Queries for expressing relations between examples:

Simple Queries representing direct links between two examples:

$$\texttt{relation}(x_i, x_j), \tag{13}$$

where $\texttt{relation} \in N_R$ is an atomic role. An example of a query in this form is $\texttt{friendOf}(x_i, x_j)$, which retrieves friendship relationships.

Symmetric Queries representing common links:

$$\begin{aligned} &\exists z.\, (\texttt{relation}(x_i, z) \land \texttt{relation}(x_j, z))\,, \\ &\exists z.\, (\texttt{relation}(z, x_i) \land \texttt{relation}(z, x_j))\,, \end{aligned} \tag{14}$$

where $\texttt{relation} \in N_R$ is an atomic role, and $z \in N_V$ is a non-distinguished variable. An example of a query in this form is:

$$\exists z.\texttt{authorOf}(x_i, z) \land \texttt{authorOf}(x_j, z),$$

which retrieves co-authorship relationships, where z represents the co-authored work.

However, although there is a general agreement on the correct formal interpretation of CQs, there not exist an official, formal specification [?]. For such a reason, we retrieve the relationships between examples by using SPARQL-DL [?] queries. This approach has the following advantages:

– SPARQL-DL queries generalize CQs by also allowing the use of variables in place of atomic roles.
– SPARQL-DL and SPARQL queries share the same syntax [?] [5]: for such a reason, applying the same method on RDF knowledge bases using SPARQL queries requires very little effort.

Example 3 (Retrieving Relations between Examples by using a SPARQL-DL query) Assume we need to retrieve all relations expressed by *Simple Queries*, as in Eq. 13. Such relations can be retrieved by the following simple SPARQL-DL query:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?x ?y ?r WHERE {
  ?x ?r ?y .
  ?r a owl:objectProperty .
}
```

In the case the set of examples X only comprises elements of a specific concept, such as `Person`, it is sufficient adding two triple patterns to the SPARQL-DL query, which constrain the type of the two examples involved in the relationship:

```
  ?x a ns:Person .
  ?y a ns:Person .
```

■

Summary of the Proposed Method

The method, which we refer to as *Adaptive Knowledge Propagation with Dissimilarities* (AKP$^{\mathrm{D}}$), can be summarized by the following steps:

1. Retrieve relations among examples in X by using SPARQL-DL queries, and create a set of binary adjacency matrices $\mathcal{R} = \{\mathbf{R}_1, \ldots, \mathbf{R}_r\}$, each corresponding to a distinct relation type, as discussed in Sect. 5.
2. Find the parameters $\boldsymbol{\Theta}^*_{LOO}$ that minimize the Leave-One-Out Error by using a gradient-based optimization process, as shown in Sect. 5 (specifically, by using the algorithm outlined in Alg. 1).
3. Use the learned parameters for finding an optimal labeling for unlabeled examples $\mathbf{f}^*_U$, by constructing the *influence graph* as defined in Sect. 5 (Eq. 9), and then propagating knowledge across such graph as presented in Sect. 4 (Eq. 7).

The current method is an extension of our previous work, which used a similar technique for propagating information across only homophilic relations [?, ?].

[5] The main difference between SPARQL and SPARQL-DL queries is that, in SPARQL-DL queries, one needs to specify whether a variable occurring in place of a role name refers to an object property or a data property.

7 Related Works

A variety of methods have been proposed for predicting the truth value of assertions in Web Ontologies, and Knowledge Graphs (graph-structured Knowledge Bases) in general [?].

Such methods can be distinguished into *kernel methods* (e.g., see [?, ?, ?]), *latent feature/factor models* (e.g., see [?, ?, ?, ?, ?, ?, ?, ?, ?]), and those based on the upgrading of propositional algorithms (e.g. [?]).

An issue with existing methods is that they either rely on a possibly expensive search process, or induce statistical models that are often not easy to interpret by human experts.

Kernel Methods. For instance, kernel methods induce models – such as separating hyperplanes – w.r.t. a high-dimensional feature space, which are implicitly defined by a kernel function. The underlying kernel function itself usually relies on purely structural features of the neighborhood graphs of two individual resources (such as the number of their common subtrees [?] or isomorphic subgraphs [?]), that do not necessarily have a direct translation in terms of domain knowledge.

In the proposed method, on the other hand, it suffices to examine the sign (and the weight magnitude) associated to each relation type to assess whether it is homophilic or heterophilic w.r.t. a given property: see also the experiments in Sect. 8 for a qualitative analysis of the learned models.

Latent Factor Models. In *latent factor models* (also referred to as *latent feature models* in the literature [?]), the properties of each entity are determined by a set of latent factors, whose parameters are not known in advance. Such models are based on probability theory (such as IHSM [?]), on tensor factorization (such as TripleRank [?], SUNS [?] and RESCAL [?]), or on an energy-based framework [?] (such as SE [?], NTN [?], SME [?], TransE [?] and TransH [?]). In many of these models (such as RESCAL, SE, SME, TransE and TransH), entities in the KB are represented into a relatively low-dimensional embedding vector space, and the *energy* (i.e. a measure inversely proportional to the probability) of each assertion (fact) results from an operation on such embeddings.

Models in this class have been proposed for query answering on factorized probabilistic triple databases [?], collective classification [?] and link prediction tasks (e.g. see [?]) on large-scale KBs.

Thanks to their scalability properties, such models have been attracting an increasing interest recently [?]. However, interpreting and understanding the latent factors in terms of domain knowledge is still an open research problem, since they do not always have an interpretable meaning [?].

First-Order Probabilistic Logic. Other methods, such as those described in [?, ?, ?], try to overcome the mentioned limitation by making use of first order logic rules extended with probabilistic information.

However, such methods rely on a search process spanning a possibly very large rules space, which might not be feasible in practice. In addition, in Markov Logic Networks [?], probabilistic inference is intractable on large KBs.

Heterogeneous Information Network Mining Models. The problem of learning from Web KBs is also related to mining *Heterogeneous Information Networks* [?, ?], which contain multiple types of entities and relations. The problem of propagating label information across relations in such networks has been discussed in the literature: in [?], the authors propose a method for propagating information across multiple types of nodes using a single type of relation. However, it is worthwhile to remark that:

1. Parameters are learned using a simple grid-search, which is infeasible if the parameter space is high-dimensional (as a solution, we propose gradient-based learning method for incrementally optimizing the influence graph);
2. They only account for a single type of (exclusively homophilic) relation, while AKP^D can make use of multiple types of relations, both homophilic and heterophilic.

In [?], the authors consider so-called *meta-paths*, representing relation paths in the network. However, in this work, the authors: *a*) Do not propose any efficient way to learn the weight of each meta-path, and *b*) Do not discuss the problem of efficiently retrieving the meta-paths holding among a set of examples.

In the following section, the proposed method is empirically compared with some of the aforementioned methods, and results are analyzed both quantitatively, terms of predictive accuracy, and qualitatively, with further specific considerations on the interpretability and understandability of the learned models.

8 Empirical Evaluations

In this section, we experimentally evaluate the AKP^D , as briefly summarized in Sect. 6, on a series of assertion prediction tasks. In empirical evaluations we used

Pellet [?], an open source DL reasoner [6], for answering the SPARQL-DL queries. Sources and datasets for reproducing the experiments are available on-line, with an open-source license: `https://code.google.com/p/akp/`. We now describe the setup of experiments and their outcomes.

Ontologies

We considered three real world ontologies: the DBpedia 3.9 Ontology [?], the AIFB Portal Ontology [7], and the British Geological Survey (BGS) Ontology [8]. The characteristics of these ontologies are outlined in Tab. 1.

- DBpedia [?] makes structured information extracted from Wikipedia available in the LOD cloud, providing unique identifiers for the described entities that can be dereferenced over the Web: The DBpedia 3.9 knowledge base, released in September 2013, describes approx. 4.0 million entities.
- The AIFB Portal Ontology relies on knowledge from the SWRC Ontology and metadata available from the Semantic Portal of the AIFB institute: it models key concepts within a research community, such as persons, articles, technical reports, projects and courses (e.g. ~ 500 individuals belong to the `Person` class and $\sim$2400 to the `Document` class).
- The British Geological Survey Ontology is part of an effort by the British Geological Survey, a premier center for earth science, to publish geological data (such as hydro-geological, gravitational and magnetic data) under OpenGeoscience [9]. In particular, the ontology models knowledge on 11697 "Named Rock Units".

Experimental Setting

In AKP$^{\mathrm{D}}$, summarized in Sect. 6, we used *Projected Gradient Descent* for finding the parameters that minimize the Leave-One-Out Error, using the *absolute loss* $\ell(x, \hat{x}) = |x - \hat{x}|$ as loss function (reconstruction error) of choice. During the gradient descent process, the optimal step size is assessed using a simple line search. The parameter λ in Eq. 11 was fixed to $\lambda = 10^{-6}$, for preventing the parameters to diverge.

For each of the considered, we evaluate AKP$^{\mathrm{D}}$ on a different prediction task, where we aim at completing the missing information about a given property of individual resources The properties to be predicted are already fully available in the initial ontologies, which serve as a gold standard.

Accordingly to the evaluation protocols in [?, ?], for each prediction task, we partially remove the information to be predicted from the ontology, following a k-fold Cross Validation procedure. At each iteration, this procedure creates a set of entities for which the property or class to be predicted is available (which will be the *positive examples*) and a set of entities for which it is missing (which will be the *unlabeled examples*).

As noted in [?, ?], it is often not possible to extract *negative examples* from a SW knowledge base. For instance, in the DBpedia 3.9 Ontology, we can extract all US Presidents affiliated with the Democratic Party, but we cannot extract the US Presidents that are *provably not affiliated* with the Democratic Party, since party affiliations are not mutually exclusive. For the sake of comparison, for collecting negative examples, also in this case we follow the procedure used in [?, ?]. This procedure is based on the Local Closed World Assumption, discussed in Sect. 1: this procedure consists in sampling negative examples from the examples where the property to be predicted is already valued (assuming that the knowledge about such property is *locally complete* for the considered examples). For example, this consists in sampling a set of the US Presidents affiliated to the Republican Party, and considering them as not affiliated with the Democratic Party. Following [?, ?], the set of sampled negative examples has the same cardinality as the set of positive examples.

In each experiment, we considered the problem of predicting the membership to each of several classes: for each class, we performed a distinct k-fold Cross Validation (CV) experiment, with $k = 10$.

AKP$^{\mathrm{D}}$ yields a continuous *confidence value* for each binary prediction, which can be considered as a sort of *ranking*. For such a reason, we evaluated the results in terms of Area Under the Precision-Recall Curve (AUC-PR). The AUC-PR has proven to be a suitable evaluation metric for evaluating rankings [?], and is widely used in the relational learning literature – for instance, see [?, ?].

For each binary classification method considered in our experiments, we used the same 10-folds partitioning of the dataset: for such a reason, we report statistical significance tests using a paired, non-parametric difference test (Wilcoxon T test). We also report diagrams showing how using a smaller sample of labeled training examples affects results.

[6] Pellet v2.3.1 – `http://clarkparsia.com/pellet/`
[7] Static dump version `V2012-02-21`, retrieved from `http://www.aifb.kit.edu/web/Wissensmanagement/Portal`
[8] `http://data.bgs.ac.uk/`, as of March 2014
[9] `https://www.bgs.ac.uk/opengeoscience/`

Table 1: Ontologies considered in the experiments

Ontology	DL Language	#Axioms	#Individuals	#Properties	#Classes
AIFB PORTAL [?]	$\mathcal{ALEHO(D)}$	268540	44328	285	49
DBPEDIA 3.9 [?] Fragment	$\mathcal{ALCH}$	78795	16606	132	251
BGS [?]	$\mathcal{ALI(D)}$	825133	87555	154	6

Setup of the Compared Methods

We compare AKP$^{\mathrm{D}}$ with several state-of-the-art methods proposed for learning from SW knowledge bases. Specifically, we considered two kernel methods: Soft-Margin SVM [?, pg. 223] (SM-SVM) and Kernel Logistic Regression [?] (KLR), together with different kernel functions suited for ontological KBs. In particular, we considered the *Intersection SubTree* [?] (IST) and the *Weisfeiler-Lehman* [?] (WL) kernels for ontological KBs. We also considered the SUNS [?] relational prediction model.

The RDF graph used by kernel functions and SUNS was materialized as follows: all $\langle \mathtt{s}, \mathtt{p}, \mathtt{o} \rangle$ triples were retrieved by means of SPARQL-DL queries (where $\mathtt{p}$ was either an object or a data-type property) together with all *direct type* and *direct sub-class* relations.

All hyper-parameters used in the considered methods (such as the depth of the neighborhood graph used by the kernel functions, or the factorization rank in SUNS) were selected through a 10-fold CV within the training set. As in [?], IST kernel parameters were selected in $d \in \{1, 2, 3, 4\}$ and $\lambda_{ist} \in \{0.1, 0.3, \ldots, 0.9\}$, while WL kernel parameters in $d, h \in \{1, 2, 3, 4\}$, where d represents the depth of the considered neighborhood graph. In SM-SVM, in order to obtain a ranking among instances, we applied the logistic function s to the decision boundary f instead of the sign function (which is commonly used in the classification context), thus obtaining $s(f(\cdot)) : X \to [0, 1]$. The parameter C in SM-SVM was selected in $C \in \{0.0, 10^{-6}, 10^{-4}, \ldots, 10^{4}, 10^{6}\}$, while in KLR the weight λ_k associated to the L_2 regularization term was selected in $\lambda_k \in \{10^{-4}, 10^{-3}, \ldots, 10^{4}\}$.

In the SUNS relational prediction model, the factorization rank t and the regularization parameter λ_s were selected in $t \in \{2, 4, 6, \ldots, 24\}$ and $\lambda_s \in \{0, 10^{-2}, 10^{-1}, \ldots, 10^{6}\}$.

Experiments with the DBPEDIA 3.9 Ontology

Similarly to [?], we evaluated the proposed approach on the task of predicting the value of missing political party affiliations to either the Democratic and the Republican party for 82 US Presidents and Vice-Presidents in the DBPEDIA 3.9 Ontology.

The experiment illustrated in [?] uses a small RDF fragment, only containing RDF triples with the predicates `president` and `vicePresident`: the former links each Vice-President with its Presidents, while the latter links each President with its Vice-Presidents.

For replicating a more realistic scenario, in this experiment we used a real fragment of the DBPEDIA 3.9 Knowledge Base: it was obtained by means of a crawling process, and contains a number of irrelevant and possibly noisy entities and relations. The DBPEDIA 3.9 fragment was extracted by a crawler, following the extraction procedure described in [?]. Specifically, the RDF graph was traversed starting from the resources representing US Presidents and Vice-Presidents. We retrieved all their immediate neighbors of the selected resources, together with all related schema information (direct classes and their super-classes, and the corresponding class hierarchy). All extracted knowledge was used for creating a new Knowledge Base, whose characteristics are summarized in Tab. 1.

In this experiment, for efficiency reasons, parameters in the WL kernel were $d = 1$ and $h = 1$.

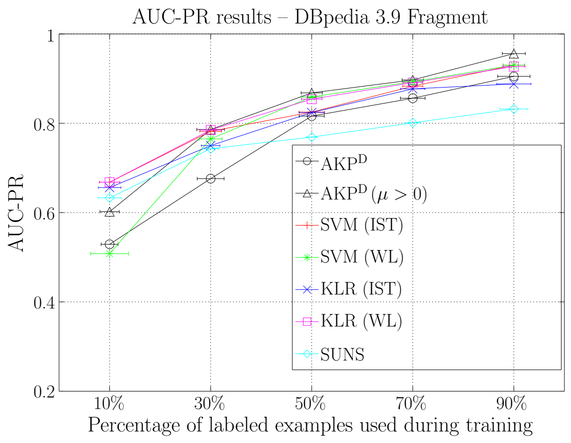
For this dataset, the total number of retrieved relations (both *simple* and *symmetric*) between US Presidents and Vice-Presidents was higher than the number of instances itself: 82 US presidents and vice-presidents were interlinked by 25 *simple* relations and 149 *symmetric* relations. This differs from the other experiments, where instance are only linked by a limited number of, exclusively *symmetric*, relations.

For such a reason, we investigated two variants of the proposed method: AKP$^{\mathrm{D}}{}_{\mathrm{S}}$, which only relies on *simple* relations, and AKP$^{\mathrm{D}}$, which uses both *simple* and *symmetric* relations. We also investigated the impact of enforcing the additional constraint $\boldsymbol{\mu} \geq \mathbf{0}$ in the parameters learning process, so to only assign *positive edge weights* in the influence graph: we denote the corresponding variant as AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$).

Experimental results are summarized in Fig. 1. The plot shows average AUC-PR values describes results obtained with a limited number of labeled training examples, and leaving the rest to the test: error bars (pictured horizontally) represent twice the standard deviation. AUC-PR results for each distinct political party are outlined in Tab. 2.

Table 2: DBpedia 3.9 – AUC-PR test values on the task of predicting the political party affiliations for all presidents and vice-presidents in the DBpedia 3.9 Ontology

DBpedia 3.9	AKP^{D} ($\boldsymbol{\mu} \geq \mathbf{0}$)	AKP^{D}	KLR (WL)	KLR (IST)	SUNS
Democratic	$\mathbf{.911 \pm .151}$	$.872 \pm .199$	$.879 \pm .102$	$.813 \pm .158$	$.835 \pm .174$
Republican	$\mathbf{1.00 \pm .000}$	$.938 \pm 0.111$	$.897 \pm .224$	$.850 \pm .122$	$.772 \pm .172$



Method	AUC-PR (mean ± var.)	$\boldsymbol{\mu} \geq \mathbf{0}$	AKP^{D}
AKP^{D} ($\boldsymbol{\mu} \geq \mathbf{0}$)	$\mathbf{.956 \pm .013}$		▲
AKP^{D}	$.905 \pm .026$	▼	
SUNS	$.832 \pm .019$	▼	▽
SMSVM (IST)	$.930 \pm .011$		
SMSVM (WL)	$.930 \pm .011$		
KLR (IST)	$.888 \pm .029$	▽	
KLR (WL)	$.927 \pm .012$		

Fig. 1: DBpedia 3.9 Ontology – Left: AUC-PR results (mean, st.d.) estimated by 10-fold CV, obtained varying the percentage of labeled examples used for training – Right: AUC-PR results estimated by 10-fold CV: ▼/▽ (resp. ▲/△) indicates that AKP^{D} 's mean is significantly higher (resp. lower) in a paired Wilcoxon T test with $p < 0.05$ / $p < 0.10$

From the results, we can see that AKP^{D} ($\boldsymbol{\mu} \geq \mathbf{0}$) yields higher AUC-PR values than every other method in the comparison. In particular, results obtained with AKP^{D} were significantly higher than the results yield by SUNS (with $p < 0.05$), and higher than those resulting from AKP^{D} and kernel methods relying on the WL and IST kernels. A possible explanation for AKP^{D} ($\boldsymbol{\mu} \geq \mathbf{0}$) yielding better results than (unconstrained) AKP^{D} is the following: most of the relationships between examples are *symmetric* (i.e. denote similar characteristics), and relationships of this kind are more likely to be *homophilic* (link similar entities) than *heterophilic* (link different entities).

This consideration is also supported by an analysis of the results yield by $\text{AKP}^{\text{D}}{}_{\text{S}}$ (which only relies on *simple* relationships), which show that modeling *heterophilic* relations – which are usually embodied by direct links between entities – can improve the predictive accuracy of the model.

Specifically, $\text{AKP}^{\text{D}}{}_{\text{S}}$ achieved a .935±.018 AUC-PR, while its constrained variant $\text{AKP}^{\text{D}}{}_{\text{S}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$) achieved a .922 ± .020 AUC-PR. An explanation for these results is the following. Recall that $\text{AKP}^{\text{D}}{}_{\text{S}}$ only relies on *simple* relationships, which can easily be both *homophilic* and *heterophilic*: for such a reason, also considering *heterophilic* relations within the model improved its empirical effectiveness.

Analysis of the Learned Model. AKP^{D} successfully identified which relations are likely to link presidents and vice-presidents in the same political party: some of such relations are summarized in Tab. 3.

The vast majority of the relationships with the largest and the lowerst μ weights were *simple*, which emphasizes the role of *homophily* and *heterophily* in this particular domain. For instance, AKP^{D} successfully recognized that US Presidents and Vice-Presidents linked by the relations `president` (which links a Vice-President to the corresponding President) and `vicePresident` (which links a President to the corresponding Vice-President) are very likely to be affiliated to the same political party, and were associated to high $\mu \gg 0$ weights. On the other hand, the predecessor or a successor of a US President or Vice-President (i.e. those linked by a `predecessor` or a `successor` relationship) are very likely to belong to *different* political parties: such relations were assigned a very low weight $\mu \ll 0$.

A number of relations were considered neither homophilic nor heterophilic: they were associated with a weight $\mu \approx 0$, meaning that they were considered not relevant in the knowledge propagation process. For instance, presidents and vice-presidents sharing their profession, religion or education were not considered more likely to be associated with the same political party.

Efficiency. The influence graph constructed for the DBpedia 3.9 Ontology fragment can be fairly dense, containing approx. 2775 non-zero edges in the case all μ weights are non-zero, resulting in an $\approx 83.6\%$ graph density. This result might motivate the adoption, in future works, of methods to enforce sparsity in the influence graph during the learning process, such as L_1

Table 3: Relations considered in the AIFB Portal and the DBpedia 3.9 Ontologies and a description of their corresponding weight

AIFB Portal	
$\mu_i > 0$	$\mu_i \leq 0$
`author`$^{-1} \circ$ `author`	`title` $\circ$ `title`$^{-1}$
`interest` $\circ$ `interest`$^{-1}$	`mobile` $\circ$ `mobile`$^{-1}$
`lecturer`$^{-1} \circ$ `lecturer`	`road` $\circ$ `road`$^{-1}$

DBpedia 3.9	
$\mu_i > 0$	$\mu_i \leq 0$
`vicePresident`	`successor`
`president`	`predecessor`
`region` $\circ$ `region`$^{-1}$	`profession` $\circ$ `profession`$^{-1}$
`state` $\circ$ `state`$^{-1}$	`award` $\circ$ `award`$^{-1}$

regularization [?], or by enforcing weights to be non-negative. For this experiment, the parameters learning process in AKP$^{\mathrm{D}}$ took an average of ~ 80 seconds on a single core of an Intel® Core$^{\mathrm{TM}}$ i7 processor. This shows that the proposed method is feasible for learning from real world KBs.

Experiments with the AIFB Portal **Ontology**

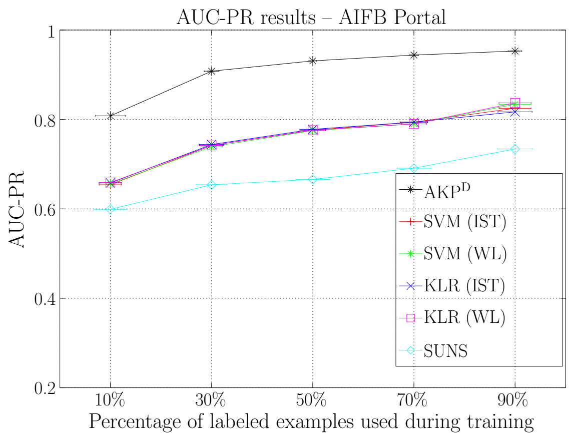
As in [?] and [?], this experiment consisted in predicting missing affiliations of AIFB Institute [10] staff members to research groups in the AIFB Portal Ontology. Specifically, in a set of 316 examples (each representing a distinct researcher), the task consisted in predicting the missing affiliations of researchers to 5 distinct research groups (each representing a distinct class).

The research groups described in the AIFB Portal Ontology are *Business Information Systems* (BIK, with 109 affiliates), *Complexity Management* (CoM, with 23 affiliates), *Efficient Algorithms* (EffAlg, with 49 affiliates), *Economics and Technology of eOrganizations* (EOrg, with 21 affiliates) and *Knowledge Management* (WBS, with 121 affiliates).

For each research group, we evaluated the proposed method on the task of predicting whether unlabeled examples, each representing a researcher, were members of the research group or not. As in [?, ?], following the Local Closed World Assumption, negative examples for each research group are sampled from the set of unlabeled examples that are also members of other classes (i.e. representing researchers in other research groups).

Empirical results are described in Fig. 2. The table summarizes the overall AUC-PR results on the research group affiliation prediction task, obtained via 10-fold CV (one per research group, in a *one-versus-all* setting). From the results summarized in Fig. 2, we can

[10] `http://www.aifb.kit.edu/`



Method	AUC-PR (mean ± var.)	
AKP$^{\mathrm{D}}$	$\mathbf{.953 \pm .004}$	
AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$)	$.952 \pm .004$	
SUNS	$.734 \pm .030$	▼
SMSVM (IST)	$.825 \pm .025$	▼
SMSVM (WL)	$.834 \pm .025$	▼
KLR (IST)	$.817 \pm .029$	▼
KLR (WL)	$.837 \pm .025$	▼

Fig. 2: AIFB Portal – Left: AUC-PR results (mean, std.dev.) estimated by 10-fold CV, obtained varying the percentage of labeled examples used for training – Right: AUC-PR results estimated by 10-fold CV: ▼/▽ (resp. ▲/△) indicates that AKP$^{\mathrm{D}}$'s mean is significantly higher (resp. lower) in a paired Wilcoxon T test with $p < 0.05$ / $p < 0.10$

clearly see that AKP$^{\mathrm{D}}$ yields significantly better AUC-PR results than every other method in the comparison, where statistical significance was calculated with a Wilcoxon T test with $p < 0.05$. Detailed results for each of the research groups are available in Tab. 4.

One of the main difficulties in this dataset is that there is very limited information for a subset of researchers, since they were not related with any other researcher in the ontology. In these cases, AKP$^{\mathrm{D}}$ correctly identified that it is hardly possible to predict the affiliation of such researchers, and assigned a $\mathbf{f}_i \approx 0$ continuous labeling to their research group membership. A labeling very close to zero means that the model does not have enough information for determining whether such researchers belong to the research group $(+1)$ or not (-1), thus expressing an high degree of uncertainty.

On the other hand, methods relying on the WL and IST kernels considered such researchers similar to each other, and were more likely to assign them to the same research group, often incorrectly.

Analysis of the Learned Model. All relations extracted in this experiment were *symmetric*: they indi-

Table 4: AIFB Portal – AUC-PR test values on the task of predicting the research group affiliation for all researchers in the AIFB Portal Ontology

AIFB Portal	AKP$^{\mathrm{D}}$	AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$)	SVM (WL)	SVM (IST)	SUNS
EffAlg	$\mathbf{.962 \pm .040}$	$.961 \pm .032$	$.838 \pm .137$	$.836 \pm .130$	$.855 \pm .098$
EOrg	$\mathbf{.958 \pm .088}$	$.958 \pm .088$	$.956 \pm .094$	$.928 \pm .099$	$.764 \pm .173$
BIK	$\mathbf{.917 \pm .071}$	$.916 \pm .069$	$.824 \pm .106$	$.825 \pm .094$	$.628 \pm .131$
WBS	$\mathbf{.971 \pm .018}$	$.969 \pm .017$	$.875 \pm .063$	$.874 \pm .067$	$.839 \pm .080$
CoM	$\mathbf{.958 \pm .088}$	$.958 \pm .088$	$.678 \pm .222$	$.661 \pm .226$	$.586 \pm .183$

cate shared relations with other entities in the ontology, such as co-authored articles, shared office rooms and co-worked projects. Such relations were either homophilic or not relevant to the knowledge propagation process, but never heterophilic. For such a reason, there is very little difference between results obtained with AKP$^{\mathrm{D}}$, and its constrained variant AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$), which only assumes homophily relations.

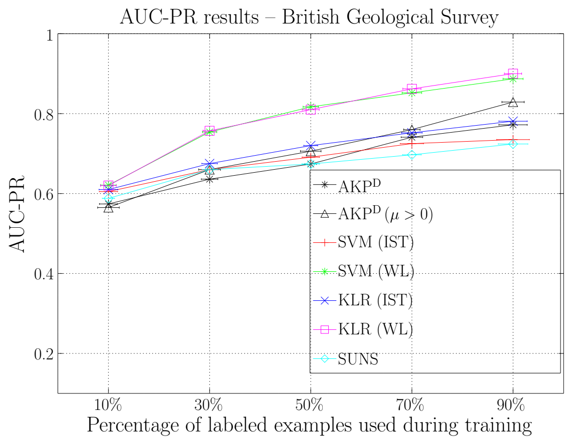
Tab. 3 shows a sample of the homophilic and non-homophilic relation types discovered during the experiments, from a total of 77 retrieved relation types. Recall that each was associated with a (learned) weight μ_i, which weighs its role in the construction of the influence graph $\mathbf{W}$, and thus in the knowledge propagation process. Specifically, if $\mu_i \gg 0$, AKP$^{\mathrm{D}}$ identified that the relation is homophilic, and can be used for propagating knowledge. For instance, AKP$^{\mathrm{D}}$ correctly identified that researchers co-authoring the same publications, sharing their interests, teaching the same classes and working in the same offices are very likely to be affiliated with the same research group.

Efficiency. The influence graph constructed for the AIFB Portal Ontology can also be fairly dense, containing approx. 46330 non-zero edges in the case all μ weights are non-zero, resulting in an $\approx$ 89.1% graph density. During this experiment, the parameters learning process in AKP$^{\mathrm{D}}$ took an average of $\sim$500 seconds on a single core of an Intel® Core™ i7 processor.

Experiments with the BGS Ontology

As in [?], we evaluated AKP$^{\mathrm{D}}$ on the *Lithogenesis* prediction problem in the British Geological Survey Ontology. The problem consisted in predicting the value of the property `hasLithogenesis` in a set of 159 named rock units labeled with their corresponding lithogenetic type. Following [?], we focus on two learning tasks, consisting in the prediction of two major lithogenetic types: "Alluvial" and "Glacial".

Results are summarized in Fig. 3, and grouped for each lithogenetic type in Tab. 5. AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$) yields better AUC-PR results than SUNS, and kernel methods relying on the Intersection SubTree (IST) kernel.



Method	AUC-PR (mean ± var.)	
AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$)	$\mathbf{.829 \pm .012}$	
AKP$^{\mathrm{D}}$	$.772 \pm .020$	▽
SUNS	$.724 \pm .022$	▼
SMSVM (IST)	$.735 \pm .026$	▼
SMSVM (WL)	$.887 \pm .010$	▲
KLR (IST)	$.781 \pm .020$	▽
KLR (WL)	$\mathbf{.900 \pm .007}$	▲

Fig. 3: BGS Ontology – Left: AUC-PR results (mean, st.d.) estimated by 10-fold CV, obtained varying the percentage of labeled examples used for training – Right: AUC-PR results estimated by 10-fold CV: ▼/▽ (resp. ▲/△) indicates that the mean of AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$) is significantly higher (resp. lower) in a paired Wilcoxon T test with $p < 0.05$ / $p < 0.10$

In general, AKP$^{\mathrm{D}}$ ($\boldsymbol{\mu} \geq \mathbf{0}$) achieved better results also compared with its unconstrained version AKP$^{\mathrm{D}}$: a possible explanation for this phenomenon is that all relations between examples extracted in this experiment were *symmetric*, and thus more likely to encode homophily relations. All relations between examples extracted in this experiment were *symmetric*, and more likely to be homophilic (i.e. link similar entities) than heterophilic. Kernel method relying on the Weisfeiler-Lehman (WL) kernel achieved AUC-PR results comparable to AKP$^{\mathrm{D}}$. This result confirms the effectiveness of the WL kernel on this specific dataset and task [?]. However, the statistical models produced with the WL kernel are not trivial to interpret in terms of domain

Table 5: British Geological Survey – AUC-PR test values on the task of predicting the lithogenetic type for all Named Rock Units in the BGS Ontology

BGS	AKP^{D} ($\boldsymbol{\mu} \geq \mathbf{0}$)	AKP^{D}	SM-SVM (WL)	SM-SVM (IST)	SUNS
Fluvial	$\mathbf{.871 \pm .106}$	$.773 \pm .146$	$.853 \pm .115$	$.760 \pm .146$	$.703 \pm .164$
Glacial	$\mathbf{.788 \pm .102}$	$.771 \pm .142$	$\mathbf{.922 \pm .068}$	$.709 \pm .180$	$.744 \pm .133$

knowledge. On the other hand, models learned by the proposed method explicitly represent the importance of each relation in the knowledge propagation process.

Also in this case, AKP^{D} was able to extract relations between rock units that are likely to link rocks with similar lithogenetic types. For example, among a total of 23 relations (all *symmetric*) it emerged that rocks with similar geographical distributions, thickness and lithological components were likely to share their lithogenetic type, while their geological theme and oldest geological age were not considered informative.

Efficiency. For this experiment, the parameters learning process in AKP^{D} took an average of ~ 100 seconds on a single core of an Intel® Core^{TM} i7 processor.

Scalability

In AKP^{D} the result of the knowledge propagation process can be calculated using the closed-form solution provided in Eq. 7. In particular, we show that computing the closed-form solution can be reduced to solving a linear system with an SDD coefficient matrix: this problem has a nearly-linear complexity in the number of non-zero elements in the coefficient matrix (edges of the influence graph). Similarly, in Sect. 5 we show that calculating the Leave-One-Out Error or its gradient has nearly-linear complexity in the number of edges in the influence graph, and in the number of labeled examples.

For solving SDD linear systems, in our experiments, we used an open-source implementation [11] of Lean Algebraic Multigrid (LAMG) [?], a fast numerical algorithm for solving linear systems with an SDD coefficient matrix. LAMG has a nearly-linear time and space complexity in the number of non-zero elements in the coefficient matrix, and has been shown to scale to graphs up to 47 million edges.

For evaluating the proposed model on relational domains with a growing number of entities and relations between them, we considered a network as the one in Fig. 4. It is structured as a squared grid, where each row is composed by nodes (entities) with the same label. Each entity in the grid-structured network is connected with the four adjacent entities. *Horizontal* and

[11] `https://code.google.com/p/lamg/`

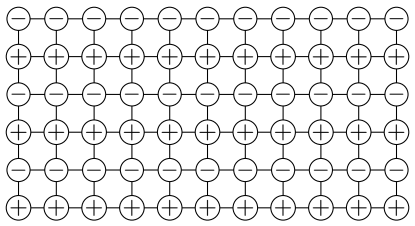


Fig. 4: Grid-structured network: Nodes labeled as + (resp. −) represent *positive* (resp. *negative*) examples, and horizontal and vertical links correspond to two distinct relations types

vertical links belong to two different types of relations, and only the former is homophilic, since it links entities with the same label.

In Fig. 5 we report the time required for both inference (propagating knowledge across chains of related entities) and learning (computing the gradient of the Leave-One-Out Error), using a varying number of nodes in the grid and randomly sampled labeled examples $|L|$. From the plots we can see that, even for a very large number of nodes (10,000 entities), the closed form solution allows propagating knowledge to the whole graph in less than 3.5 seconds. We can also see that computing the gradient of the Leave-One-Out Error is feasible for very large sets of nodes (10,000 entities). However, since this operation requires a propagation step for each labeled example, its complexity grows with the number of labeled examples $|L|$, and might be intractable if $|L|$ is very large. As discussed in Sect. 5, a possible solution consists in minimizing the k-fold Cross Validation Error instead of the Leave-One-Out Error, which would reduce the number of propagation steps from $|L|$ to k.

Improving the Scalability of the Method. Despite its interesting complexity properties, AKP^{D} can still be not completely feasible for very large and Web-scale graphs with billions of unlabeled examples. Several approaches have been proposed to tackle this problem: they can be used in conjunction with the method proposed in this article for further improving its efficiency and scalability. In [?], the authors propose sub-sampling the examples in the similarity graph, so to reduce the

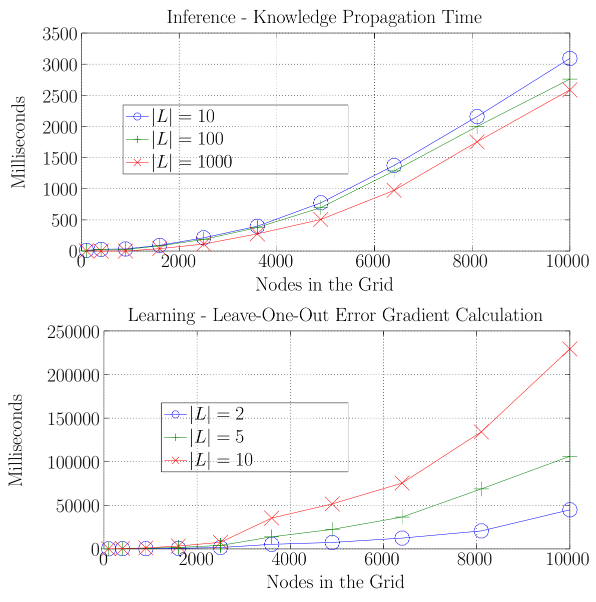


Fig. 5: Timings of *knowledge propagation* (inference) and Leave-One-Out Error *gradient calculation* (learning), for a varying number of nodes in the network and labeled examples

global graph size. In [?], the authors propose using the Nyström low-rank approximation for representing the Laplacian of the similarity graph. In [?], the authors use smooth eigenvectors of the Laplacian of the similarity graph for computing the discriminant function. In [?], the authors propose using *anchors* (landmarks) for representing groups of nodes in the similarity graph, significantly reducing the size of the graph and thus the complexity of the propagation process. In [?], the authors rely on a minimum spanning tree for approximating the similarity graph, and minimum tree cut for propagating information across (chains of) similar examples.

9 Conclusions

In this article, we proposed a method for predicting missing assertions in Web Ontologies.

We ground on the assumption that *related entities influence each other*. In particular, relations may be *homophilic* (i.e. more likely to link similar entities) or *heterophilic* (i.e. more likely to link dissimilar entities), with respect to a given set of properties, such as gender, talkativeness or nationality.

We propose a method, named *Adaptive Knowledge Propagation with Dissimilarities* (AKP$^{\mathrm{D}}$), which is capable of:

1. Automatically identifying homophilic and heterophilic relations holding between examples (learning step),
2. Efficiently propagating knowledge across chains of related examples (inference step).

AKP$^{\mathrm{D}}$ is inspired by Graph-based Semi-Supervised Learning methods, which rely on a *similarity graph* defined over examples for propagating knowledge across them. However, such methods were characterized by two major limitations: *a*) They assume that the similarity graph is already provided, and *b*) They do not make use of *dissimilarity* relations between examples.

In this work, we propose a solution to these problems, by: *a*) Automatically constructing the optimal *influence graph* for a given learning task, by identifying homophilic and heterophilic relations between examples, and *b*) Relying both similarity and dissimilarity relations, encoded in the influence graph, for propagating knowledge across examples. A lesson learned, however, is that considering both homophilic and heterophilic relations does not necessarily yield more accurate results (in terms of AUC-PR) in comparison to the solution based only on homophilic relations, at least in the prediction tasks taken into account.

Furthermore, by leveraging recently developed methods for efficiently solving Symmetric and Diagonally Dominant linear systems, we show that AKP$^{\mathrm{D}}$ has a nearly-linear complexity in the number of edges in the influence graph, both in the inference and learning step.

We empirically show that the proposed method successfully identifies homophilic and heterophilic relations in real world KBs, and that such information can provide new knowledge about a domain of interest. We also show that AKP$^{\mathrm{D}}$ provides significantly better or competitive AUC-PR results in comparison with several methods proposed in the literature.

We are currently investigating methods for further improving the efficiency of AKP$^{\mathrm{D}}$, and for identifying more complex forms of relationships between entities in a SW knowledge base.

bibliography,solvers,meta,dblp,lod,querying,gp,srl,kernel,embeddi