Metric learning with multi-relational data

Jiajun Pan, Hoel Le Capitaine

Metric Learning with Multi-Relational Data

Jiajun Pan[1] and Hoel Le Capitaine[2*]

[1]Université de Lorraine, LORIA UMR 7503, Vandoeuvre les Nancy, France.
[2]Nantes Université, LS2N UMR 6004, Nantes, France.

*Corresponding author(s). E-mail(s): hoel.lecapitaine@ls2n.fr;
Contributing authors: jiajun.pan@inria.fr;

Abstract

Over the past decades, there has been a growing interest in metric learning, a type of representation learning that aims to learn a distance metric that can fit to the data being analyzed. Many metric learning algorithms have been designed for data lying in Euclidean spaces, where a parametric Mahalanobis metric can be learned. However, such algorithms are often unable to handle relational data, that is not independent and identically distributed (i.i.d.), or can only be used at an entity level. In contrast, relational data allows for the discovery of complex interactions between features and entities, which can lead to better models. In this paper, we introduce two novel metric learning algorithms tailored to handle relational data, that preserve the structural information of the graph and use the features of the nodes as well. The first one is supervised and makes full use of both the graph structure and node labels with a carefully designed loss function, while the second is unsupervised and only uses the graph structure. Our experimental results show that both methods outperform state-of-the-art learning algorithms. Interestingly, we also find that the proposed unsupervised method often performs better than traditional supervised metric learning approaches.

Keywords: Metric Learning, Multi-Relational Learning

1 Introduction

Currently, machine learning is extensively employed in many real-world domains, such as financial services, marketing forecasting, healthcare, and government analysis, among others. It is also used in several related artificial intelligence domains, including

natural language processing, machine vision, and pattern recognition [1]. Irrespective of the type of machine learning method employed, the evaluation and comparison of sample entities are typically required.

The effectiveness of machine learning algorithms is significantly influenced by the quality of the distance metric used to describe the dissimilarities between entities, also known as observations. The Euclidean metric is the most commonly used distance metric, although numerous other techniques are used to quantify the differences between feature vectors, as discussed in the study [2].

In real-world applications, no distance metric is capable of perfectly fitting complex data distributions. Thus, there is a need for an algorithm that can automatically learn a data-adapted metric. This is known as the metric learning problem [3], which is a subfield of machine learning and, like deep learning, falls under the category of representation learning approaches. Representation learning [4], encompasses a set of techniques for transforming raw data into features that can be employed by machine learning algorithms or to convert data into more efficiently learned features. It eliminates the need for manual feature extraction and enables computers to learn how to extract and use features simultaneously, thereby learning to learn.

The output of metric distance learning is not a model that is directly utilized for prediction, but rather a new metric that adapts to the task at hand. This metric can be regarded as a representation of the data in a new latent feature space or a self-adaptive feature space.

Metric distance learning has undergone several advancements in recent years, and there have been numerous interactions and combinations with transfer learning, deep learning, and other related fields, as presented in [3, 5].

However, there exist various types of data in real-world applications that do not conform to the standard flat structure, which are commonly referred to as non-flat data. These include strings, time-series, trees, graphs, and relational databases, among others. The majority of existing models for non-flat data are based on the corresponding non-structural data type of distance metric (e.g., Hamming distance for strings [6], dynamic time warping distance for time series [7], and edit tree distance for trees [8]). Once defined, conventional distance metric learning algorithms are typically employed. However, to the best of our knowledge, there are currently no metric learning algorithms specifically designed for relational data, which is where our proposals come in. More specifically, the key problem to be solved is the lack of a formal metric between entities in a relational dataset, where entities are described by individual features and by links to each others.

The purpose of our paper is to formalize and learn a metric that specifically takes into account multi-relational data [9]. Specifically, we aim to develop a proposed metric that provides an embedding space for classification or visualization. In other terms, we propose a representation learning process consisting in learning node embeddings of multi-relational data, which incorporates both edge and node label constraints. Compared to [9], we propose the following additional elements:

- we propose an unsupervised alternative approach to our proposition, where learning constraints are only obtained through the graph structure, hence facilitating its use in much broader applications,

- we modify and extend the link-strength based approach presented in [10], which is restricted to bipartite graph data, to multiple and valued edges between nodes,
- we provide an extensive experimental analysis with a) more results, b) more other recent deep approaches included into the study, and c) more datasets,
- we analyse the trade-off on the degree of supervision, and discuss the relative conditions under which methods perform better than existing ones.

2 Metric learning and classical algorithms for flat datasets

Metric learning is a field of study in machine learning that aims to create effective representations of entities by mapping them into spaces [11]. The resulting metric is tailored to the underlying data distribution and can be applied to various machine learning methods. The key to this representation is the ability to accurately capture the similarities and differences between entities. The most commonly used metric is the Mahalanobis distance:

$$d_M^2(x_i, x_j) = (x_i - x_j)^T M (x_i - x_j), \tag{1}$$

with a learned square matrix M of size $p \times p$, where p denotes the feature dimension and x_i is a point in $\mathbb{R}^p$.

A popular formulation of metric learning using the Mahalanobis distance d_M^2 is to find M such that it minimizes $L(M) = \ell(M, \mathcal{C}) + \lambda r(M)$, where ℓ is a loss function which penalizes unsatisfied constraints, with $\mathcal{C}$ the set of constraints, controlled by a trade-off parameter λ between the regularization term $r(M)$ and the loss. If feasible, this model is generally cast as a constrained optimization problem

$$\begin{aligned} &\min\ r(M) \\ &\text{s.t. } \ell(M, i) \leq 0, \forall i \in \mathcal{C} \end{aligned} \tag{2}$$

There are different types of constraints that can be used in metric learning, depending on the specific approach being employed. Two commonly used types of constraints are pairwise constraints and relative constraints. Pairwise constraints involve setting a maximum margin for the distance between two samples x_i and x_j sharing the same class label. Additionally, if a third sample x_k with a different label is considered, the distance between x_k and the first sample x_i should be greater than the margin. Relative constraints, on the other hand, involve comparing the distances between pairs of samples. Specifically, if two samples have the same class label, the distance between them should be lower than the distance between one of those samples and another sample with a different label.

Metric learning is a well-studied topic with a vast literature. Some of the seminal works include LMNN (Large Margin Nearest Neighbor)[12], ITML (Information-Theoretic Metric Learning)[13], and their various extensions. The field has progressed to recent deep metric learning approaches. Readers interested in an in-depth review can refer to [14] and [15], respectively.

Most linear metric learning algorithms use the Mahalanobis distance model, but recently, there has been interest in other measures such as similarity functions that are also parameterized by a matrix M. The difference is that M does not have to be positive semi-definite (PSD), which avoids the need to repeatedly project the matrix onto the PSD cone while learning M. Nonlinear metric distance learning algorithms have also evolved in recent years. The general idea for the nonlinear case is to use an embedding function ϕ before the linear projection, which can be any nonlinear function, a kernel being a popular choice. Consequently, some linear metric learning methods have been extended with kernelization, and GB-LMNN [16] is an example of LMNN where a non-linear projection is used. Gradient-boosting is applied to learn nonlinear mappings directly in a function space. Additionally, there are several kernel-based metric learning approaches based on KPCA [17], a nonlinear extension of PCA. KPCA projects the data into an induced nonlinear feature space and performs dimension reduction in this feature space. For example, the authors of [18] use linear metric learning algorithms based on Mahalanobis distance in the KPCA feature space.

Nonlinear machine learning algorithms frequently use neural networks, which also make them an appropriate choice for nonlinear metric learning. Among the first nonlinear metric learning method, LSMD (Learning Similarity Metric Discriminatively) [19], introduces a model that learns a nonlinear projection $\phi_W(\mathbf{x})$ parameterized by a vector W, with relative constraints in the latent space. It is worth noting that most deep metric learning methods rely on a two-step process, i.e., embedding and linear metric learning in the projected space. The learned metric is not deep in itself; rather, the embedding is.

Recently, another metric that does not depend on the Mahalanobis distance has been proposed. This distance is defined using the Lovasz extension and allows for learning weights on coalitions of features [20].

3 Learning with non-flat data

Most metric learning approaches are designed to work with data represented as feature vectors, where constraints are generated based on target labels or other supervised information. However, such distances are not well-suited for complex and/or structured non-iid data, which are prevalent in real-world datasets. Such data can take many forms, including string sequences, time series, trees, and graphs. These types of data are commonly referred to as non-flat data.

To address this issue, several metric learning algorithms have been proposed specifically for non-flat data in the past decades. One benefit of applying metric learning to non-iid data is that it can serve as a proxy for any metric-based algorithm that accesses data as if it were represented as feature vectors, without the need to handle these complex objects directly. Additionally, there are already many existing structural metrics associated with representing structured objects, such as edit distances and alignment indices. These metrics can be learned using metric learning strategies, just like learning metrics from feature vectors.

There are a variety of metrics available for comparing complex data. The simplest ones are based on an extension of Euclidean distance or alignment-based measures.

For example, the Needleman-Wunsch score [21] and the Smith-Waterman score [22]) are used for string sequences.

Edit distance is also a versatile tool for metric learning algorithms that is useful for non-flat datasets, including string sequences and other structured datasets such as tree or graph datasets [23]. Learning such a distance basically consists into learning an adapted cost function for each action. More recently, the authors of [24] proposed a variant based on the use of node embeddings on which the Euclidean distance can be used for classification.

Another metric learning method focuses on representing structural information for trees and groups [25]. In this approach, the authors propose a new graphical representation method for solving functional brain connection problems. They use a Siamese graph convolutional neural network (GCN [26]) to learn a graph similarity metric. The authors transfer the training set to a bipartite graph as the training pairs, and the inputs of Siamese GCN are the training pairs. The outputs are combined by an inner product layer followed by a single fully connected layer as the similarity estimate. The model is driven by the hinge loss of matching and non-matching graphs.

The authors of [27] propose a metric learning algorithm based on the adjacency matrix A of a network, SPML (Structure Preserving Metric Learning). SPML aims to learn a Mahalanobis distance metric by transforming the matrix M while maintaining the inherent connectivity structure of the network. The authors impose supervised constraints on the algorithm by requiring that the distances to all disconnected nodes x_j must be greater than the distance to the farthest connected neighbor of all neighbor nodes x_l. However, to alleviate the issue of considering every connected neighbor, they suggest introducing an additional input parameter K to limit the number of visited connected neighbor nodes x_l.

Recently, new embedding methods have been developed for graph structures that share the same objective as Word2Vec. These methods, such as Node2Vec, Struc2Vec, and Variational Graph Autoencoder, are discussed in a recent review [28]. They all begin by embedding a graph, or its nodes, into a feature vector that can be used for metric learning. While they give very interesting results, they share the same caveats of the vast majority of neural models : they are by essence black boxes that are not easy to interpret, and barely explainable.

While the objective is learning a metric, one may take advantage of the relations (or edges in graphs) existing between the samples. Relational learning, as described in [29], involves learning the uncertain relationships between target samples or internal associations within complex sample structures. These relationships can be either external or internal.

Different theories distinguish these two types of relationships, but the fundamental approach to learning relationships is essentially the same. Unlike other machine learning techniques, relational learning treats relationships as an additional source of information, in addition to the features of the samples themselves. The learning tasks of relational learning are focused on relationship information or predictive relationship information, and include collective classification, logical interpretations, link-based clustering, and link prediction.

Statistical relational learning (SRL) or probabilistic logic learning is the foundational theory of relational learning, and aims to learn the probability distribution of the uncertainty of relationships [30]. Probabilistic Relational Models (PRMs) are an extension of Bayesian Networks to relational data [31], and use the Entity-Relationship Model to represent the relationships between entities. Another well-known graphical model for relational learning is the Markov Logical Network (MLN) [32], which is defined as a set of weighted first-order logic formulas that constrain logical interpretations.

As an additional layer of information, one can consider to multi-relational data which can be described as a hyper-graph or a set of different graphs with the same batch of nodes. Entities may have several types of relations with same class entities, as well as different class entities. Graph-based data mining tends to focus more on the structure of graphs than on the properties of individual nodes or the expression of a single relational rule. For example, [33] focuses on sub-graph representation information, while [34] uses a graph embedding method to map the structure of a partial graph into the features of each sample.

The authors of [35] propose metric learning on graphs for domain adaptation. Their proposed method involves an iterative algorithm on the graph, where a new metric is learned from labeled nodes in the resource domain and then applied to the unlabeled nodes in the target domain. The graph is updated based on the learned distance, and low-entropy instances are chosen as constraints for the next iteration.

Several metric learning algorithms have been proposed to deal with relational data represented as heterogeneous networks. For example, in [36], a heterogeneous metric learning algorithm is presented that integrates the structure of different graphs into joint graph regularization. This algorithm uses two mapping functions for the feature space of the object entities and subject entities in one relation and introduces a joint graph regularization for iterative optimization of the loss function. Similarly, [37] uses meta-path-based random walks to incorporate heterogeneous network structures into skip-gram vectors for dealing with the relational graph.

In the last few years, embedding using methods have flourished. A number of them are using the concept of sequences obtained from random walks or variant of [38]. Inevitably, it has been applied to node embedding of graphs, entities of knowledge graphs [39] and tuple embeddings in databases [40]. In [41], the authors propose to encode the structural information of a multi-relational graph into a tree, on which unsupervised clustering algorithms are used, but without directly considering node features.

While these algorithms perform well in considering the structure information in the relational dataset, they do not take into account the side information carried by the links and the node features at the same time. They process the edge variables in the same way as the entities, but do not distinguish entity tables and association tables, which limit their scope of application and their performances. In contrast, our proposed method includes the value of different variables in the relationship and distinguishes them from entities. Furthermore, it has been shown that many node embedding methods based on random walk change considerably, even with constant parameter settings [42].

Note that we restrict in this study to metric measures, i.e. measures holding the four usual distance properties: symmetry, identity of indiscernibles, positivity and triangular inequality. It is worth noting that alleviating one of these properties may facilitate the learning process. Furthermore, the usefulness of each of these properties has been discussed, and some works show their flaws in practical situations [43]. Other approaches may use a non-metric definition of proximity, see [44] for details.

4 Metric learning for multi-relational data

The goal of this paper is to propose a metric learning algorithm that can be applied to a relational database with multiple entity tables and multiple relationships between entity tables. We first present different frameworks dealing with learning models adapted to relational data, and then describe our scientific proposal.

4.1 Learning with multi-relational data

Real-world datasets often contain multi-relational links between entities as the primary source of information, rather than just the structure or topological information. Social network analysis, for example, focuses on the relationships between users. Typically, such relational datasets have only one entity table with multiple relationships between them.

The authors of [45] present various relational frameworks, including first-order logic, relational database model, and set theory. An n_r-array relation $\mathbb{R}$ is defined as a subset of the Cartesian product of n_r sets, denoted by $\times$, see [31] :

$$\begin{aligned} &\mathbb{R} \subseteq \mathbb{V}_1 \times \cdots \times \mathbb{V}_{n_r} \\ &= \{(v_1, \cdots, v_{n_r}) | v_1 \in \mathbb{V}_1 \wedge \cdots \wedge v_{n_r} \in \mathbb{V}_{n_r}\}. \end{aligned} \tag{3}$$

The domain of $\mathbb{R}$, $dom(\mathbb{R})$, which also denotes the Cartesian product $\mathbb{V}_1 \times \cdots \times \mathbb{V}_{n_r}$, is the set of all possible relationships over the entities in their domains. For a set $\mathbb{X}$ and a subset $\mathbb{X}_{sub} \subseteq \mathbb{X}$, the characteristic function of $\mathbb{X}_{sub}$ is a boolean-valued function $f_{cha|\mathbb{X}_{sub}} : \mathbb{X} \to \{0, 1\}$ which indicates for all elements in $\mathbb{X}$, whether they are also an element of the subset $\mathbb{X}_{sub}$:

$$\forall x \in \mathbb{X} : f_{cha|\mathbb{X}_{sub}}(x) = \begin{cases} 1 \text{ if } x \in X_{sub} \\ 0 \text{ otherwise} \end{cases} \tag{4}$$

For a relation R, its characteristic function is a function $f_{cha|\mathbb{R}} : \mathbb{V}_1 \times \cdots \times \mathbb{V}_n \to \{0, 1\}$

$$f_{cha|\mathbb{R}} = \begin{cases} 1 \text{ if a relation exists} \\ 0 \text{ otherwise} \end{cases} \tag{5}$$

The focus of this work is on binary-valued (dyadic) relational data, where each relationship R_k is a subset of $\mathbb{V}_i \times \mathbb{V}j$. The subject and object of the relationship

are denoted by a and b respectively, following the RDF convention [46]. To formalize relational data, relational tensors are used, and combine n_r-tuples and set theory. Specifically, relational tensors rely on n_r-array relations, which are defined as sets of n_r-tuples. Relational learning is concerned with predicting the existence of a relationship between two individuals by learning the characteristic function of the relation from supervised information. In this context, a relational tensor stores the relationships of relational data with the characteristic function $f_{cha|\mathbb{R}}$. For modelling dyadic relational data, a labelled directed graph is used, where nodes represent entities and labelled directed edges denote relationships between them. The relational tensor is then the union of the characteristic function of the relations.

Given a multi-relational graph, a relational tensor T with n entities (nodes) and n_r different relations (edges) can be written as an extension of the square affinity matrix to the n_r relations, $T \in \mathbb{R}^{n \times n \times n_r}$:

$$t_{ijr} = \begin{cases} 1 & \text{if } R(i,j) \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The relational learning process in this work is based on a third-order tensor that includes the characteristic function of relationships between entities, as depicted in Figure 1. It is worth noting that this work focuses on a binary tensor, which indicates the presence or absence of a relationship, but it is also possible to consider valued relations (such as movie ratings by users) or vector-valued relations (such as actor ratings in a movie).

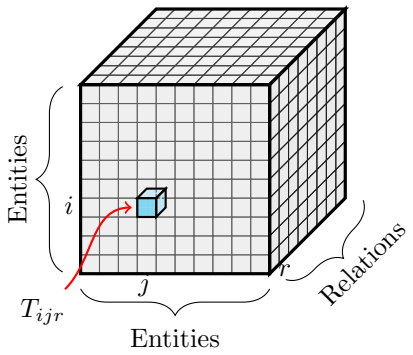


Fig. 1: Relation between entities seen as a third-order tensor T, e.g. T_{ijr} may denote that the user i and the movie j are related with the concept r.

Various relational learning approaches using relational tensors have been proposed in literature. The authors of [47] focus on existing link prediction models and extend

matrix factorization to use side information to overcome imbalance. They use the Tucker Decomposition (TD) model on a user-tag-item relational tensor to provide high-quality tag recommendations. Another approach is presented in [48], where they improve TD to PITF (Pairwise Interaction Tensor Factorization) using an adaptation of the Bayesian Personalized Ranking (BPR) criterion. PITF factors the tensor to a fixed diagonal core tensor, user matrix, item matrix, and pairwise tag matrix.

The authors of [49] and [50] propose RESCAL factorization, which decomposes the relational tensor $T \in \mathcal{R}^{n \times n \times n_r}$ into a core tensor $R \in \mathcal{R}^{a \times a \times n_r}$ and a matrix $A \in \mathcal{R}^{n \times a}$, where a is a user-given positive integer parameter with $0 < a < n$. The matrix A can be seen as an embedding of the entities into an a-dimensional latent space. Each slice k of the matrix R quantifies the similarity of the k-th relationships between entities and can be seen as a new latent feature space.

However, the matrix A only considers the relational information and not the original features, as mentioned in [50], [49] and [51], The use of a larger tensor that includes the original features during tensor factorization may cause a loss. Therefore, this work proposes using only the relational information for RESCAL factorization to obtain a latent space that represents the relational information and use it as a new "relational feature" for entities. Time and space complexity of RESCAL prevents it to be used on large scale data.

Relational Graph Convolutional Network (R-GCN) is a graph neural network architecture that is specifically designed to handle data with complex relationships between entities [52]. The novelty in R-GCN is the incorporation of relation-specific weights into the convolution operation, allowing the model to learn different weights for each type of relation in the graph. While the R-GCN model has the capability to capture more nuanced information about the relationships between nodes in a graph by incorporating relation-specific weights into the convolution operation, it does not take into account entity-specific features that could further enhance its ability to represent the graph data.

As there are few existing approaches, this work proposes a baseline approach that learns a metric in the embedding space, which is composed of the latent relational feature obtained through tensor factorization and the original feature space.

4.2 Metric learning with multi-relations

In this section, we formulate the problem of learning with multi-relational data under a metric learning framework, where we fit a metric in the entity space. The proposed method is called MRML, standing for Multi-Relational Metric Learning.

The goal of our approach is to take into account the three types of information available in the dataset, namely features, links, and labels. To achieve this, we use the Mahalanobis distance as the metric definition and incorporate relational constraints into the objective function. In essence, our objective function follows the same general model as other metric learning algorithms, but with specific modifications to account for these relational constraints.

$$L(M) = \sum_{(i,j,k) \in \mathbb{C}_{\mathbb{S}}} \ell_M(i,j,k) + \lambda r(M) \tag{7}$$

where $\mathbb{C}_{\mathbb{S}} = \{\mathbb{C}_{\mathbb{R}} \cup \mathbb{C}_{\mathbb{L}}\}$, i.e. the union of constraints obtained from links or edges, $\mathbb{C}_{\mathbb{R}}$, and constraints obtained from labels, $\mathbb{C}_{\mathbb{L}}$.

Two popular approaches are used for incorporating label information constraints: similar/dissimilar constraints and relative constraints, as described in [53]. In this work, we focus on relative constraints, given by the inequality:

$$d_M^2(x_i, x_j) + \gamma \leq d_M^2(x_i, x_k),\ \forall (i,j,k) \in \mathbb{C}_{\mathbb{L}},$$

where $\mathbb{C}_{\mathbb{L}}$ contains (i,j,k) triples of data, where (x_i, x_j) share the same label and (x_i, x_k) have different labels, and γ is a margin. The relative constraints ensure that entities with different labels are farther apart, with a margin, than entities with the same labels. We choose $\gamma = 1$ based on common usage in the literature [12].

The loss function $\ell_M(i,j,k)$ from Eq. (7) can be divided into two separate losses: ℓ_L for label constraints and ℓ_R for relational constraints. A hinge-loss functino is used to define the ℓ_L loss, which takes into account the label constraints and is defined as:

$$\ell_L = \frac{1}{|\mathbb{C}_{\mathbb{L}}|} \sum_{(i,j,k) \in \mathbb{C}_{\mathbb{L}}} \max(d_M^2(x_i, x_j) - d_M^2(x_i, x_k) + \gamma, 0) \tag{8}$$

On the other hand, for the relational constraints, we propose to use a multi-relationship tensor in place of the adjacency matrix. Each slice of the tensor represents an adjacency matrix, allowing for the consideration of all relational links. Thus, every slice provides a specific sum of loss functions, and adding up all slices leads to constraints from all relational links. We specifically focus on the connected neighbor constraints, which can be expressed as follows:

$$\forall (i,j),\ d_M^2(x_i, x_j) > (1 - R_r(i,j)) \times \max_l \left(R_r(i,l) d_M^2(x_i, x_l) \right),$$

where the matrix R_r is the slice r of the tensor T.

Summing up over all slices, and using a hinge loss, gives

$$\ell_R = \frac{1}{n_r} \sum_{z=1}^{n_r} \frac{1}{|C_z|} \times \sum_{(i,j,k) \in C_z} \max(d_M^2(x_i, x_j) - d_M^2(x_i, x_k) + \gamma, 0), \tag{9}$$

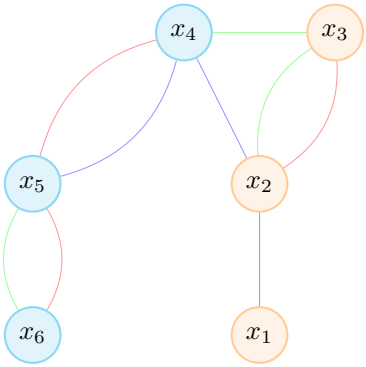
where C_z is the set of constraints obtained through the z-th relation $\mathbf{r}$ of the tensor cube T. More precisely,

$$C_z = \{(i,j,k) | \mathbf{r}_z(i,j) = 1, \mathbf{r}_z(i,k) = 0\}.$$

Note also that $\bigcup_{z=1}^{n_r} C_z = \mathbb{C}_{\mathbb{R}}$.

Taking the squared Frobenius norm as regularization term on M, the objective function becomes

$$L(M) = \frac{\lambda}{2} \|M\|^2 + \lambda' \ell_R + (1 - \lambda') \ell_L \tag{10}$$



$$\mathbb{C}_{\mathbb{L}} = \{(1,2,4),(4,5,3),(2,3,6),\cdots\}$$
$$C_1 = \{(4,5,6),(2,4,3),\cdots\}$$
$$C_2 = \{(2,3,1),(5,6,2),\cdots\}$$
$$C_3 = \{(3,4,6),(2,3,4),\cdots\}$$

Fig. 2: A multi-relational graph with binary labeled nodes (● and ●) and three different relations between nodes : r_1, r_2 , r_3 (left). Corresponding constraints drawn from the multi-relational graph (right).

where the introduced parameter λ' balances the importance of relational constraints and label constraints in the proposed approach. A value of 0 for λ' would mean that only label constraints are considered, while a value of 1 would mean that only relational constraints are used. It is worth noting that by applying an L_1 norm on M, the sparsity of the model could be promoted (such as in Lasso regularization) or a combination of L_1 and L_2 norms could be used (such as in elastic net regularization). However, exploring these different options is beyond the scope of this paper.

We adopt a stochastic sub-gradient descent with mini-batches to optimize the loss function of MRML, which has the benefit of making the complexity independent of the number of constraints. Using a convenient notation, we can express the difference between the two distances in the loss function using a sparse matrix S as follows:

$$d^2_M(x_i,x_j) - d^2_M(x_i,x_k) = S^{(i,j,k)} X^T M X, \tag{11}$$

where the different S are sparse matrices storing the parameters:
$S^{(i,j,k)}_{jj} = 1$, $S^{(i,j,k)}_{ik} = 1$, $S^{(i,j,k)}_{ki} = 1$, $S^{(i,j,k)}_{kk} = -1$, $S^{(i,j,k)}_{ij} = -1$ and $S^{(i,j,k)}_{ji} = -1$. Otherwise, $S^{(i,j,k)} = 0$. The matrices S are used to encode constraints directly into the loss function, whether to encode structural information (i.e. a link between nodes) or labels. In that case, a positive entry denotes the same node label, while a negative entry corresponds to a different node label. By design, the sparse matrix $S^{(i,j,k)}$ indexes the elements related to nodes i, j, and k, such that $tr(S^{(i,j,k)} X^T M X)$ is equal to $d^2_M(x_i,x_j) - d^2_M(x_i,x_k)$.

The sub-gradient of the objective function can then be written as:

$$\nabla L(M) = \lambda M + \frac{1-\lambda'}{|\mathbb{C}_{\mathbb{L}}|} \sum_{(i,j,k)\in \mathbb{C}_{\mathbb{L}}{}^+} X S^{(i,j,k)} X^T + \frac{\lambda'}{n_r} \sum_{z=1}^{n_r} \frac{1}{|C_z|} \sum_{(i,j,k)\in C_z^+} X S^{(i,j,k),z} X^T \tag{12}$$

where ${\mathbb{C}_{\mathbb{L}}}^{+}$ and C_z^+ are subset of $\mathbb{C}_{\mathbb{L}}$ and C_z, respectively, for which $d_M^2(x_i, x_j) - d_M^2(x_i, x_k) + 1 > 0$. Note that for each relation r, a corresponding sparse constraint matrix $S^{(i,j,k),r}$ is constructed.

Algorithm 1 Metric learning based on relational tensor with stochastic sub-gradient descent

Input: X,$\mathbb{C}_{\mathbb{L}}$, $\mathbb{C}_{\mathbb{R}}$ and parameters λ, $n_c \leq |C|$, λ',t
Output: M

$M_0 = I_m$
for t_i from 1 to $t-1$ **do**
  $S = 0_{n,n}$, $S^z = 0_{n,n}$, $\forall z \in \{1, \cdots, n_r\}$
  $n_L = 0$, $n_R = 0$
  for b from 1 to n_c **do**
    sample (i, j, k) from $\mathbb{C}_{\mathbb{L}}$ with probability λ'
    if $d_M^2(x_i, x_j) - d_M^2(x_i, x_k) + 1 > 0$ **then**
      $n_L += 1$
      $S_{jj} += 1$, $S_{ik} += 1$, $S_{ki} += 1$,
      $S_{kk} += -1$, $S_{ij} += -1$, $S_{ji} += -1$.
    end if
    sample (i, j, k) from $\mathbb{C}_{\mathbb{R}}$ with probability $1 - \lambda'$
    for r from 1 to n_r **do**
      if $d_M^2(x_i, x_j) - d_M^2(x_i, x_k) + 1 > 0$ **then**
        $n_R += 1$
        $S^r_{jj} += 1$, $S^r_{ik} += 1$, $S^r_{ki} += 1$,
        $S^r_{kk} += -1$, $S^r_{ij} += -1$, $S^r_{ji} += -1$.
      end if
    end for
  end for
  $\nabla_{t_i} = \frac{1-\lambda'}{n_L} X S X^T + \frac{\lambda'}{n_R n_r} \sum_z X S^z X^T + \lambda M_{t_i}$
  $M_{t_{i+1/2}} = M_{t_i} - \frac{\nabla_{t_i}}{t_i \lambda}$
  $M_{t_{i+1}} = Proj_{\mathbb{S}_m^+}(M_{t_{i+1/2}})$ // projection to the closest PSD matrix on closed convex cone of $\mathbb{S}_m$, space of symmetric m-by-m matrices see [54].
end for
return M_t

Algorithm 1, also pictured in the flowchart of Figure 3, provides a detailed description of MRML with stochastic sub-gradient descent. A projection onto the PSD cone ensures the matrix M defines a true metric.

This algorithm is a variation of the PEGASOS algorithm [55] without projection, similar to the approach taken in [27]. Therefore, the running time does not scale with the input size n and the number $|T|$ of edges but with the dimensionality p.

With a probability of $1 - \delta$, it provides a bound on the optimization error ε, which is given by $\frac{84R^2 \ln(t/\delta)}{\lambda t}$, where t is the number of iterations, δ is a constant, and the norm of any input x is at most R. Consequently, the necessary number of iterations

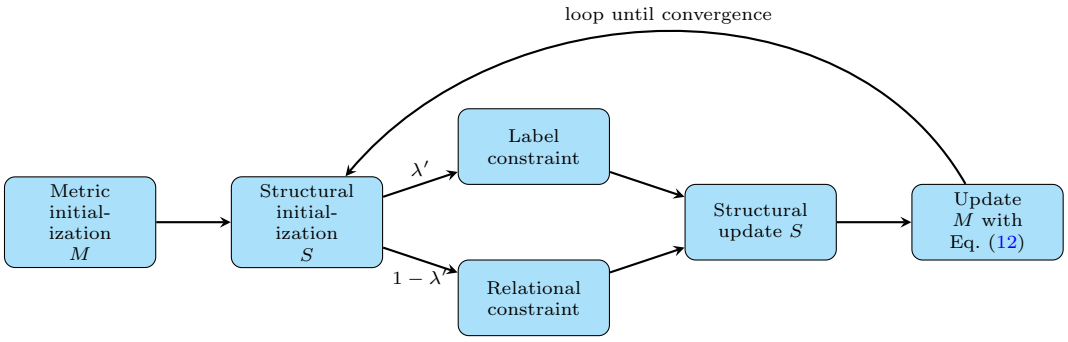


Fig. 3: A simple flow process of the proposed algorithm where M can be used in any metric based classification process.

to reach an error ε is $O(\frac{1}{\lambda\varepsilon})$. The size of the data, in terms of entities, does not play a role in the running time, because each iteration is $O(p^2)$.

It is worth noting that the algorithm can be terminated at any time and still produce a valid metric, thanks to the projection onto the convex cone of M in line 23.

5 Experiments, results and discussions

The evaluation of the proposition is done by comparing the effect of the learned metric with K-nearest neighbor (KNN) classification. More precisely, using the learned metric, we are able to compute a distance between any entities, so that the KNN algorithm allows to predict entity labels by majority vote of the k nearest entities.

We used K equal to 5 and score the performance with accuracy rate via randomly shuffled 3-fold cross-validation. Note that we tried different values for K (in particular 3, 5, 7 and 9), without notably different results. For each experiment, the number of constraints ranges from 100 to 500, and the average value of each set is taken as the final result. All the experiments were performed on an 8 cores Apple M1 chip, with 16 Go LPDDR4 as memory. The code is available online for research reproducibility[1]. We also give results obtained without learning a metric, i.e. using a Euclidean distance for the KNN algorithm (Euc).

5.1 Datasets

In this study, we use 6 real-world relational databases as benchmarks, whose properties are given in Table 1. n is the number of instances, n_r is the number of types of relations and m is the number of features.

- Elite: DutchElite dataset [56] contains the relational information of administrative elite in The Netherlands. The label distinguishes if the elite is top200 or not.
- Mondial: Mondial dataset [57], is the relational version of the geographical Web data sources. The labels are the classes of entities.

[1] https://gitlab.univ-nantes.fr/lecapitaine-h/relational-metric-learning/

Dataset	Entities n	Relations n_r	Features m	Edges	Classes
Elite	4747	41	7	5 221	2
Mondial	185	23	4	12 889	2
Movie	1804	26	5	1 237	18
UW	278	4	3	711	2
MG	4893	6	2	60 294	3
Wiki-CS	11701	1	300	297 110	10

Table 1: Dataset characteristics.

- Movie: Movie-Remark dataset [58] describes relations across several files of movie information, the labels are the genres of the movies.
- UW: The UW-CSE dataset, as presented in the standard version by the University of Washington (UW-std), describes the connections among professors and students in the Computer Science and Engineering Department at the university. The labels assigned to the data points are based on the academic stage of the individuals involved [59].
- MG: Mutagenesis dataset [60], describes trials of molecules for mutagenicity on Salmonella typhimurium. We use the atom and the bond between them as the relationship. The labels are the types of atom.
- Wiki-CS: The Wikipedia-Computer Science dataset [61] is built from Wikipedia categories, where the ten classes correspond to branches of computer science. Each node is an article whose features are pretrained with GloVe word embeddings.

Although the number of instances might appear quite low, it is worth noting that the number of relations between instances multiplies this number of instances, resulting in a much larger volume of data.

In the introduction, it was noted that traditional metric learning methods only utilize features and fail to account for the relationships between observations. To address this, we propose embedding the data into a space that accurately reflects the data's relationships to facilitate fair metric learning algorithms comparison.

For Rescal factorization-based metrics, the latent feature space is created using matrix A_l, which is generated by $T \approx R \times_1 A_l \times_2 A_l$, where A_l is a factor matrix with dimensions $n \times a$, and R is a core tensor with dimensions $a \times a \times m$. The values of A_l and R depend on the chosen positive integer parameter a. By increasing the value of a, we can reduce the approximation error; however, this comes at the cost of increased complexity. Our experiments indicate that increasing the dimensionality of A_l leads to better factorization, while increasing the rank results in a larger core tensor R and more arduous calculations.

5.2 Unsupervised learning

In this section, we compare the performance of different metrics related on different combination of features information and relational information without using labels, but only relational constraints. More precisely, we consider the following loss function

$$L(M) = \frac{\lambda}{2}\|M\|^2 + \ell_R \tag{13}$$

The sub-gradient of the objective function then simplifies to:

$$\nabla L(M) = \lambda M + \frac{1}{n_r}\sum_{z=1}^{n_r}\frac{1}{|C_z|}\sum_{(i,j,k)\in C_z^+} XS^{(i,j,k),z}X^T \tag{14}$$

In Table 2, cross-validation accuracy, and their standard deviations, using the K-nearest-neighbour algorithm with different metrics are given. Euc. stands for the usual Euclidean distance in the feature space (i.e. $M = Id$). Res corresponds to the usual Euclidean distance in the latent space A_l obtained using RESCAL factorization, as described before. Finally, Agg corresponds to the use of the joint space of RESCAL embeddings A_l and the original feature space X, given by (X, A_l). Each observation i is described by $(x_{i1}, \cdots, x_{in}, al_{i1}, \cdots, al_{ir})$.

In order to provide a comparison with metric learning algorithms in a unsupervised setting, we adapt and extend the method proposed in [10], called LSCS (Link-strength Constraints Selection), which is restricted to relations formalized by bipartite graphs. In this method, instead of randomly selecting the constraints from labels, the probability of selecting a pair of nodes is a function of the structure of the relations. The strength of the link increase the probability of being selected for the corresponding pair.

The link-strength function is extended to multi-relational datasets by summing the link-strength for each relation. We also extend from the reference relation $\mathbb{R}_r$ to $\mathbb{R}_e$, considering the group structures as every edge between nodes is the side-information of common parents, but as a binary value. In that case, the link-strength function consider the additional term

$$\sum_{k}^{\ell_{ij}^k} P_k(i,j), \tag{15}$$

where $P_k(i,j)$ is the parent adjacency matrix of the k-relation in the group structure. Considering a simple binary adjacency matrix gives $P_k(i,j) = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ have common parents in relation $\mathbf{r}_k$, and 0 otherwise. The term ℓ_{ij}^k is the number of common parents of $\mathbf{x}_i$ and $\mathbf{x}_j$ in the relation $\mathbf{r}_k$.

LSCS-ITML and LSCS-LSML are using the classical metric learning algorithm ITML [13] and LSML (Least Squares Metric Learning) [62] respectively, together with LSCS, but only selecting constraints with relational side-information, as explained above. MRML is our proposed approach in a restricted unsupervised setting where it uses only relational constraints and no target labels. Best values are indicated in bold font. From Table 2, one can see that MRML no-label consistently performs better than the other approaches, except for the Elite dataset.

Dataset	Euc	Res	Agg	LSCS-ITML	LSCS-LSML	MRML
Elite	87.60±12.08	**91.14±0.80**	89.59±0.85	84.56±12.46	88.25±1.28	88.80±5.67
Mondial	68.66±7.99	61.31±7.83	58.59±5.30	64.66±8.81	59.57±5.97	**69.40±1.33**
Movie	38.56±1.86	33.31±7.84	39.56±2.25	38.21±1.21	39.42±1.52	**40.07±2.07**
UW	96.40±0.03	81.08±0.28	87.91±0.84	96.55±0.81	95.73±0.19	**98.27±0.35**
MG	83.92±0.79	62.22±1.04	75.39±2.19	79.06±19.96	82.94±9.33	**86.16±1.17**
Wiki-CS	76.50 ±0.15	71.01±0.29	77.59±0.21	77.89±0.17	77.12±0.09	**79.91±0.14**

Table 2: Cross-validation accuracy of KNN with different metrics without label information.

5.3 Fully supervised learning

Our focus now shifts to the full potential of our proposal by using both constraints and labels. To investigate the impact of λ' on the loss function $L(M) = \frac{\lambda}{2}|M|^2 + \lambda' \ell_R + (1-\lambda')\ell_L$ in MRML, we examine how it regulates the significance of relations and labels on the learned metric and performance. Consequently, we assess the learned metric for each dataset by varying λ' from 0 (no relations) to 1 (no labels), as shown in Figure 4. As can be seen, the optimal precision for all datasets is attained between these two extremes, indicating that both pieces of information are beneficial, as expected. It is worth noting that four datasets (Elite, Movie, UW, and MG) have similar performance whether there are no relationships or no labels. However, the experimental results suggest that the node classification in Mondial is more dependent on relationships, while Wiki-CS has more information in the labels. For each dataset, the optimum λ' is chosen according to this preliminary analysis.

We first consider two algorithms that directly incorporate relational information: an MLN (Markov Logic network) as the baseline, with default discrimination parameters [32], and SPML for structural preservation of the graph [27]. We also include the results of a graph neural network dedicated to relational data, R-GCN [52]. We use a 2-layer model with 16 hidden units, as in the original paper. Then, we consider state-of-the-art metric learning: ITML [13], LSML [62] and LFDA (Local Fisher Discriminant Analysis) [63]. The three algorithms are used on the latent space provided by RESCAL in order to take into account relationships between entities. We also provide results with ITML and LSML using the proposed LSCS process.

Dataset	MLN	SPML	R-GCN	ITML	LSML	LFDA	LSCS-ITML	LSCS-LSML	MRML
Elite	NT	86.71±1.99	89.07±2.17	89.28±0.33	90.76±0.87	90.20±.54	89.18±0.64	88.66±00.80	**91.19±1.33**
Mondial	67.71±4.82	57.16±9.45	69.63±1.89	61.29±4.64	58.54±9.28	59.46±7.11	64.66±5.90	59.35±6.90	**71.23±3.27**
Movie	40.84±0.84	38.64±1.63	**41.67±1.67**	39.16±2.76	38.54±1.81	39.86±1.53	38.16±1.25	39.38±1.62	40.80±1.26
UW	77.12±6.12	87.43±4.27	98.87±2.36	96.83±0.27	92.44±0.50	90.63±0.28	97.09±2.45	94.68±2.84	**99.28±0.22**
MG	81.45±2.71	84.88±5.86	83.28±2.25	79.74±1.48	70.61±1.58	72.03±1.27	82.98±10.10	84.51±5.17	**86.16±1.26**
Wiki-CS	NT	79.04± 0.56	79.07±1.00	77.58 ±0.20	77.34 ±0.11	74.56 ±0.34	78.67 ±0.45	74.89 ±0.13	**80.98±0.24**

Table 3: Cross-validation accuracy of KNN with different metric learning methods with both relational information and target labels.

Results are given in Table 3. NT means that the running of the algorithm is out of the time limit due to memory explosion. As can be observed in Table 3, MRML

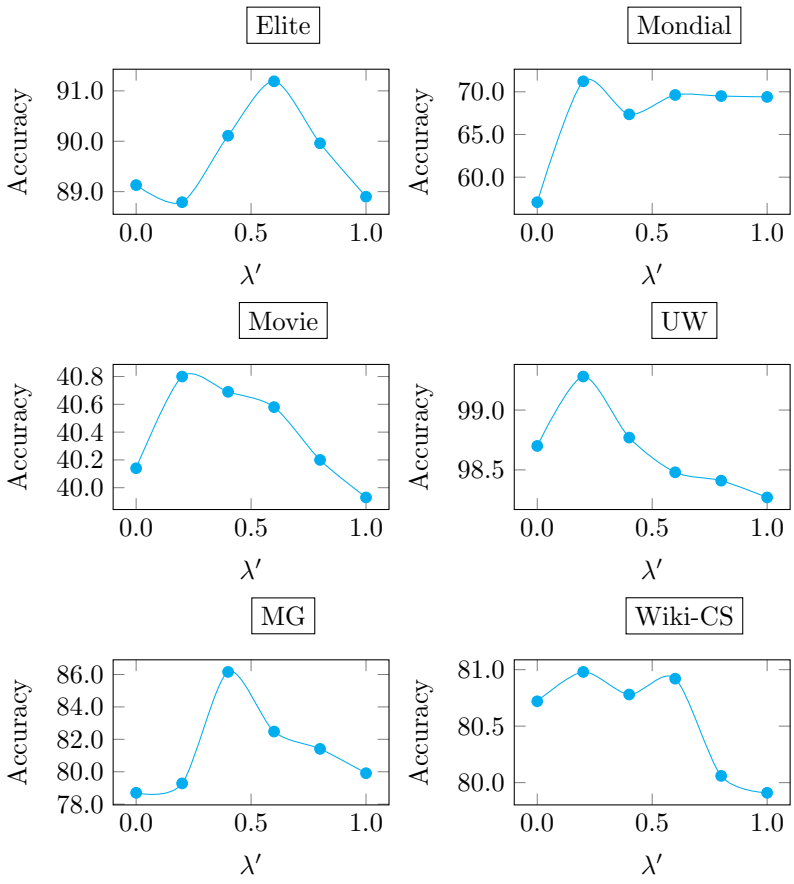


Fig. 4: Performance of the proposed MRML with respect to various levels of supervision, controlled by λ'. Best trade-offs are generally found with in-between values.

performs better than other approaches, and the Movie dataset on which R-GCN and MLN are better (although both of them are closely followed by MRML). It is worth noting that the unsupervised method proposed in the previous section performs better than the supervised algorithms in 5 out of 6 datasets, closely followed by R-GCN.

More precisely, the Mondial, Movie and UW datasets seem to give better results when the labels are prioritized, without the structure. Conversely, Elite seems to favor structure. This is also confirmed by the performance of SPML, an algorithm based solely on structure. Conversely, the ITML and LSML algorithms, inheriting the projection of attributes from RESCAL, behave better on data where the structure is less important. We can categorize the datasets considered: {Mondial, Movie, UW} where the label has great importance, {Elite} where the structure has great importance, and {MG , Wiki-CS} where the two aspects are more balanced. Interestingly, R-GCN seems to perform better for the first group (labels), and less for data where structure matters.

6 Conclusion, discussion and perspectives

Our paper introduces a novel approach to metric learning, named MRML, that leverages both features and relationships among entities in multi-relational data. We extend the standard adjacency matrix of a network to a relational tensor where entities are represented by feature vectors. Additionally, we utilize labels to generate extra constraints, similar to traditional metric learning techniques. We then present a stochastic sub-gradient descent algorithm to learn this metric, which includes a parameter λ' to regulate the degree of supervision, ranging from no use of labels to no use of relations. Our experimental results demonstrate that a trade-off between the two extremes produces the best outcomes.

Furthermore, we propose a baseline for relational metric learning utilizing tensor factorization. The resulting embedded space serves as the foundation for typical metric learning algorithms, where relations are encoded. Real-world dataset experiments demonstrate the efficacy of our approach concerning both accuracy and complexity, compared to state-of-the-art metric learning techniques.

Usual relational models, from statistical relational learning to graph-based relational mining, aims to learn a probability distribution of relationships or the structure of the graph, forgetting the characteristics of the nodes. Our method diverge from this approach, but could use the learned probability distribution on uncertain relationships in order to generate constraints related to the structure instead of randomly drawing them during the learning algorithm.

As future directions, we plan to extend our approach to non-binary relations, such as the user-movie rating relation, which can be a valued feature. Additionally, we aim to incorporate vector-valued relations into our approach. The current approach only uses binary relations in order to set relational constraints $\mathbb{C}_{\mathbb{R}}$, so that we may propose two alternatives to enhance the handling of information links. The first would be to use a mapping from $\mathbb{R}^k \to [0, 1]$ where k is the dimension of the edge vector, using e.g. a distance. Alternatively, one could use recent works on edge embeddings [64] to get more interesting constraint sets.

As an additional perspective, we would like to mention the use of local metrics within the graphs. Promising works have been proposed in this area, but generally restricted to flat data [65]. Defining locally adapted metrics for graphs, where the notion of proximity in the graph can be defined through the neighborhood of nodes, would be of great interest.

Finally, testing the proposed method on larger graphs, with millions of nodes and links would demonstrate the real potency of application.

References

[1] Mitchell, T.M., *et al.*: Machine learning. 1997. Burr Ridge, IL: McGraw Hill **45**(37), 870–877 (1997)

[2] Lenz, H.-J.: Proximities in statistics: Similarity and distance. Preferences and Similarities, 161–177 (2008)

[3] Kulis, B.: Metric learning: A survey. Foundations and Trends in Machine Learning **5**(4), 287–364 (2012)

[4] Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence **35**(8), 1798–1828 (2013)

[5] Kaya, M., Bilge, H.Ş.: Deep metric learning: A survey. Symmetry **11**(9), 1066 (2019)

[6] Hamming, R.W.: Error detecting and error correcting codes. The Bell system technical journal **29**(2), 147–160 (1950)

[7] Kruskall, J.B.: The symmetric time warping algorithm: From continuous to discrete. Time warps, string edits and macromolecules (1983)

[8] Bernard, M., Habrard, A., Sebban, M.: Learning stochastic tree edit distance. In: European Conference on Machine Learning, pp. 42–53 (2006). Springer

[9] Pan, J., Le Capitaine, H.: Metric learning with relational data. In: 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 367–372 (2019)

[10] Pan, J., Le Capitaine, H., Leray, P.: Relational constraints for metric learning on relational data. In: 8th International Workshop on Statistical Relational AI @ ICML (2018)

[11] Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning with application to clustering with side-information. Advances in neural information processing systems **15**, 505–512 (2003)

[12] Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. JMLR **10**(Feb), 207–244 (2009)

[13] Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: Proceedings of the 24th International Conference on Machine Learning, pp. 209–216 (2007). ACM

[14] Bellet, A., Habrard, A., Sebban, M.: A survey on metric learning for feature vectors and structured data. arXiv preprint arXiv:1306.6709 (2013)

[15] Ghojogh, B., Ghodsi, A., Karray, F., Crowley, M.: Spectral, probabilistic, and deep metric learning: Tutorial and survey. arXiv preprint arXiv:2201.09267 (2022)

[16] Kedem, D., Tyree, S., Sha, F., Lanckriet, G.R., Weinberger, K.Q.: Non-linear metric learning. In: Advances in Neural Information Processing Systems, pp. 2573–2581 (2012)

[17] Schölkopf, B., Smola, A., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural computation **10**(5), 1299–1319 (1998)

[18] Chatpatanasiri, R., Korsrilabutr, T., Tangchanachaianan, P., Kijsirikul, B.: A new kernelization framework for mahalanobis distance learning algorithms. Neurocomputing **73**(10-12), 1570–1579 (2010)

[19] Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference On, vol. 1, pp. 539–546 (2005). IEEE

[20] Pan, J., Le Capitaine, H.: Metric learning with submodular functions. Neurocomputing **416**, 328–339 (2020)

[21] Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of molecular biology **48**(3), 443–453 (1970)

[22] Smith, T.F., Waterman, M.S., *et al.*: Identification of common molecular subsequences. Journal of molecular biology **147**(1), 195–197 (1981)

[23] Ristad, E.S., Yianilos, P.N.: Learning string-edit distance. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(5), 522–532 (1998)

[24] Paaßen, B., Gallicchio, C., Micheli, A., Hammer, B.: Tree edit distance learning via adaptive symbol embeddings. In: International Conference on Machine Learning, pp. 3976–3985 (2018). PMLR

[25] Ktena, S.I., Parisot, S., Ferrante, E., Rajchl, M., Lee, M., Glocker, B., Rueckert, D.: Metric learning with spectral graph convolutions on brain connectivity networks. NeuroImage **169**, 431–442 (2018)

[26] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Representation Learning (2017)

[27] Shaw, B., Huang, B., Jebara, T.: Learning a distance metric from a network. In: Advances in Neural Information Processing Systems, pp. 1899–1907 (2011)

[28] Hamilton, W.L.: Graph representation learning. Synthesis Lectures on Artifical Intelligence and Machine Learning **14**(3), 1–159 (2020)

[29] Struyf, J., Blockeel, H.: Relational learning. Encyclopedia of Machine Learning and Data Mining, 1090–1096 (2017)

[30] Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. MIT press, 1st edition (2007)

[31] Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: IJCAI, vol. 99, pp. 1300–1309 (1999)

[32] Richardson, M., Domingos, P.: Markov logic networks. Machine learning **62**(1-2), 107–136 (2006)

[33] Horváth, T., Ramon, J., Wrobel, S.: Frequent subgraph mining in outerplanar graphs. Data Mining and Knowledge Discovery **21**(3), 472–508 (2010)

[34] Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems **151**, 78–94 (2018)

[35] Dhillon, P., Talukdar, P., Crammer, K.: Metric learning for graph-based domain adaptation. In: Proceedings of COLING 2012: Posters, pp. 255–264. The COLING 2012 Organizing Committee, Mumbai, India (2012)

[36] Zhai, X., Peng, Y., Xiao, J.: Heterogeneous metric learning with joint graph regularization for cross-media retrieval. In: Twenty-seventh AAAI Conference on Artificial Intelligence (2013)

[37] Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 135–144 (2017). ACM

[38] Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)

[39] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in neural information processing systems **26** (2013)

[40] Lubarsky, Y.L., Tönshof, J., Grohe, M., Kimelfeld, B.: Selecting walk schemes for database embedding. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 1677–1686 (2023)

[41] Peng, H., Zhang, J., Huang, X., Hao, Z., Li, A., Yu, Z., Yu, P.S.: Unsupervised social bot detection via structural information theory. ACM Transactions on Information Systems **42**, 1–42 (2024)

[42] Schumacher, T., Wolf, H., Ritzert, M., Lemmerich, F., Grohe, M., Strohmaier, M.: The effects of randomness on the stability of node embeddings. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 197–215 (2021). Springer

[43] Duin, R.P., Pekalska, E.: Non-euclidean dissimilarities: Causes and informativeness. In: Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, SSPR&SPR 2010, Cesme, Izmir, Turkey, August 18-20, 2010. Proceedings, pp. 324–333 (2010). Springer

[44] Schleif, F.-M., Tino, P.: Indefinite proximity learning: A review. Neural computation **27**(10), 2039–2096 (2015)

[45] Džeroski, S.: Relational data mining. Data Mining and Knowledge Discovery Handbook, 887–911 (2010)

[46] Antoniou, G., Van Harmelen, F.: A Semantic Web Primer. MIT press, 1st edition (2004)

[47] Menon, A.K., Elkan, C.: Link prediction via matrix factorization. In: ECML-PKDD, pp. 437–452 (2011). Springer

[48] Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 81–90 (2010). ACM

[49] Nickel, M., Tresp, V.: Tensor factorization for multi-relational learning. In: ECML-PKDD, pp. 617–621 (2013). Springer

[50] Nickel, M., Tresp, V., Kriegel, H.-P.: A three-way model for collective learning on multi-relational data. In: ICML, vol. 11, pp. 809–816 (2011)

[51] Nickel, M., Tresp, V., Kriegel, H.-P.: Factorizing yago: scalable machine learning for linked data. In: Proceedings of the 21st International Conference on World Wide Web, pp. 271–280 (2012). ACM

[52] Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15, pp. 593–607 (2018). Springer

[53] Le Capitaine, H.: Constraint selection in metric learning. Knowledge-Based Systems **146**, 91–103 (2018)

[54] Higham, N.J.: Computing a nearest symmetric positive semidefinite matrix. Linear algebra and its applications **103**, 103–118 (1988)

[55] Shalev-Shwartz, S., Singer, Y., Srebro, N., Cotter, A.: Pegasos: Primal estimated sub-gradient solver for svm. Mathematical programming **127**(1), 3–30 (2011)

[56] Nooy, W.D.: The network data on the administrative elite in the netherlands in April- June 2006. De Volkskrant (2008)

[57] May, W.: Information extraction and integration with florid: The mondial case study. Technical Report 131, Universität Freiburg, Institut für Informatik (1999)

[58] Lichman, M.: UCI Machine Learning Repository (2013)

[59] Khosravi, H., Schulte, O., Hu, J., Gao, T.: Learning compact Markov logic networks with decision trees. Machine Learning **89**(3), 257–277 (2012)

[60] Debnath, A.K., Lopez de Compadre, R.L., Debnath, G., Shusterman, A.J., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. Journal of medicinal chemistry **34**(2), 786–797 (1991)

[61] Mernyei, P., Cangea, C.: Wiki-cs: A wikipedia-based benchmark for graph neural networks. arXiv preprint arXiv:2007.02901 (2020)

[62] Liu, E.Y., Guo, Z., Zhang, X., Jojic, V., Wang, W.: Metric learning from relative comparisons by minimizing squared residual. In: Data Mining (ICDM), 2012 IEEE 12th International Conference On, pp. 978–983 (2012). IEEE

[63] Sugiyama, M.: Local fisher discriminant analysis for supervised dimensionality reduction. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 905–912 (2006). ACM

[64] Jo, J., Baek, J., Lee, S., Kim, D., Kang, M., Hwang, S.J.: Edge representation learning with hypergraphs. Advances in Neural Information Processing Systems **34**, 7534–7546 (2021)

[65] Wang, J., Kalousis, A., Woznica, A.: Parametric local metric learning for nearest neighbor classification. Advances in Neural Information Processing Systems **25** (2012)