Unsupervised Relational Representation Learning via Clustering: Preliminary Results

Sebastijan Dumančić and **Hendrik Blockeel**
Computer Science Department KU Leuven, Belgium

Abstract

The goal of unsupervised representation learning methods is to learn a new representation of the original data, such that it makes a certain classification task easier to solve. Since their introduction in late 2000s, these methods initiated a revolution within machine learning. In this paper we present an unsupervised representation learning method for relational data. The proposed approach uses a clustering procedure to learn a new representation. Moreover, we introduce an adaptive clustering method, capable of addressing multiple interpretations of similarity in relational context. Finally, we experimentally evaluate the proposed approach. The preliminary results show the promise of the approach, as the models learned on the new representation often achieve better performance and are less complex than the ones learned on the original data representation.

Introduction

In the last decade, the unsupervised representation learning paradigm (Hinton and Salakhutdinov 2006; Bengio et al. 2007; Ranzato, Boureau, and LeCun 2007) initiated a revolution within the field of artificial intelligence. The key substance of these methods is learning a new set of features before addressing a given classification task. The core idea is to make a classifier more robust by modifying the representation of data, in a way that makes it easier to detect complex dependencies. Much research has been directed towards identifying properties of a good data representation (Bengio, Courville, and Vincent 2013). An important aspect to note here is that these methods avoid using any supervision when constructing the features, but instead find a representation that might be suitable for many tasks on the same dataset.

Intuitively, these methods can be viewed as multi-clustering procedures, where examples can belong to multiple clusters. A new representation typically takes the form of a set of *binary variables* which indicate cluster memberships of each example. Viewed in the context of neural networks, each hidden node is *activated* (i.e., its output is larger than zero) if an example belongs to the cluster indicated by the node. Once obtained, the new representation is given to a classifier to address a classification task, i.e., a classifier learns from cluster memberships, not the original data. One can further inspect the incoming weights associated with the hidden node to identify features relevant for the cluster. Moreover, recent results (Coates, Lee, and Ng 2011) show that, in principle, any clustering method (e.g., k-means) can be used to extract new features, so one is not tied to clustering methods formalized within neural networks.

Conceptually, a new representation is an abstraction of the original high-dimensional data into a lower dimensional space. The importance of abstracting the original data has been recognized within the relational learning community since its very beginnings. There the problem is known as *predicate invention* (Kramer 1995). The goal of predicate invention is to extend the original vocabulary given to a relational learner, by discovering novel concepts and relations from data. The objective of predicate invention is to either improve the performance of a classifier, or compact a model while not sacrificing the performance. Many approaches addressing this problem have been proposed (Muggleton and Buntine 1988; Wogulis and Langley 1989; Silverstein and Pazzani 1991; Srinivasan, Muggleton, and Bain 1992; Craven and Slattery 2001; Perlich and Provost 2003; Popescul and Ungar 2004; Davis et al. 2007; Kok and Domingos 2007a; Muggleton and Lin 2013; Wang, Mazaitis, and Cohen 2015). We would argue that representation learning shares the same goals of model compression and accuracy improvement.

As representation learning provides a relational learner with a new vocabulary abstracted from the original one, it is a form of predicate invention. However, in contrast to the existing work, an important difference has to be emphasized: whereas existing predicate invention work simply extends the original vocabulary, representation learning replaces the entire original vocabulary with a new one, abstracted from the original vocabulary. We focus on the latter case. This work presents an approach that leverages a relational clustering procedure to extract a new representation. Concretely, we apply relational clustering to cluster objects and relations in a given dataset (discussed in the next section), create new predicates to indicate the cluster membership, and treat this as the new representation of the same dataset. Thus, a predictive model will be learned from the cluster memberships of the relational objects, instead of the original data.

The potential of clustering for introducing novel and useful information in relation tasks has been utilized before. Popescul and Ungar (2004) apply k-means clustering to the objects of each type in a domain, create predicates for new clusters and add them to the original data. *Whereas Popescul and Ungar cluster the objects to enhance the original dataset representation, we are here interested in learning entirely from the newly extracted clusters and discard the original data. Moreover, we cluster both objects and relations between them. Though such representation could as well just be added to the original dataset, that would significantly increase the search space, while learning only from the abstract one can even simplify it.*

Kok and Domingos (2007b, 2008) introduce *multiple relational clustering (MRC)*, a relational probabilistic clustering framework based on Markov logic networks (Richardson and Domingos 2006). MRC simultaneously clusters both objects of each domain and their relations. This framework as well includes a probabilistic decision on the optimal number of clusters, based on *how well they explain the provided data*. This work might be considered as the first work on relational representation learning. However, the main drawback of this approach is that it is prone to overgenerating new predicates. Generating new representation with a large vocabulary makes the subsequent structure learning increasingly more difficult. Consequently, in their later work on structure learning (Kok and Domingos 2009; 2010) the authors used MRC as a pre-processing step with the intention to reduce the size of a given hypergraph. There MRC was used to identify interchangeable nodes in a hypergraph, which can later on be replaced by a new *super-node*. Consequently, this makes the subsequent *path-finding* task, which identifies the structure of an MLN, substantially easier. Thus, MRC was not used to provide new language constructs, but to simplify the search space over possible formulas.

The remainder of the paper is structured as follows. In the following section we present a high-level outline of our approach. We discuss the main issues arising when applying clustering to learn a new representation, and address desirable properties a clustering method should have for this task. Then, in the following section a new clustering method is presented, and its advantages over the existing clustering methods are discussed. In the next section we present preliminary results we obtain with TILDE (Blockeel and De Raedt 1998). We then conclude the paper.

Representation learning via clustering

As stated before, unsupervised representation learning can be seen as clustering the original data. However, it is not immediately clear how to generalize these ideas to the relational setting. Most of the issues arise from the representation of the relational data. In contrast to the i.i.d. case where one has a clear concept of an example, the relational case is more ambiguous.

Relational datasets can broadly be divided in two categories. In the first category, the data is simply a large network consisting of many interconnected entities, typically regarded as a mega-example. Such datasets are typically considered within SRL (Getoor and Taskar 2007) research. The datasets in the second category consist of many disconnected networks, where each individual network can be seen as an example. Such scenarios are typical for ILP (De Raedt 2008) research.

When developing relational representation learning methods, one has to address both scenarios. Therefore, several questions arise when clustering relational data: *(1) what should clusters consist of, (2) how should one interpret the similarity of relational objects, and (3) how to choose the number of clusters*. In the i.i.d. case, the dataset contains only the independent examples and their features, thus, one clusters the examples. The relational case, on the other hand, makes it more difficult to find a general approach that works for both of the discussed scenarios. ILP methods typically require a user to provide a set of examples and a background knowledge that describes the examples. For example, when learning a predictive model for mutagenicity of molecules, those molecules form examples, while atoms that form the molecules, as well as their properties, are considered to be the background knowledge. Though that gives a clear concept of an example, such information is not available within the unsupervised representation learning task.

To answer the first two questions, we will assume that relational data is provided as a labelled hypergraph, where examples form vertices and relations between them form hyperedges. We will make no further distinction between a single connected hypergraph or a composition of multiple hypergraphs, but consider both cases as a hypergraph. Formally, the data structure that we assume in this paper is a typed, labelled hypergraph $H = (V, E, \tau, \lambda)$ with V being a set of vertices, and E a set of hyperedges; each hyperedge is an ordered set of vertices. The type function τ assigns a type to each vertex and hyperedge. The set of all vertex types is denoted as T_V, whereas T_E denotes the set of all hyperedge types. A set of attributes $A(t)$ is associated with each $t \in T_V$. The labelling function λ assigns to each vertex a vector of values, one for each attribute of $A(\tau(v))$. If $a \in A(\tau(v))$, we denote $a(v)$ the value of a in v.

Accordingly, as the answer to the first question, we will learn a new representation by clustering both vertices and hyperedges of a certain type in a hypergraph. Moreover, considering that vertices have associated types, we will not allow mixing of types, i.e., clusters can contain only vertices of the same type. Similarly, only hyperedges that connect vertices of the same type can appear in the same cluster. In this work, we will also require a *role specification* for each predicate, i.e., the information whether a certain argument of a predicate identifies a feature value or a unique object. Consequently, the clustering is performed only on domains that represent objects, not feature values.

The second question is somewhat more difficult to answer. In the i.i.d., case, the features are the only source of similarity between examples. However, when one considers relational data, the interpretation of similarity becomes ambiguous. There, while formulating a similarity measure one can take into account attribute similarity, similarity of the relations an object participates in, similarity of neighbouring vertices (in terms of attributes of relationships) and interconnectivity or graph proximity of the objects. Different inter-

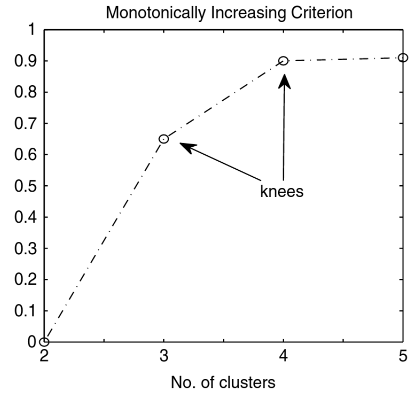

Figure 1: Example of criterion with monotonically increasing values as a function of k (from Vendramin, Campello, and Hruschka (2010)).

pretations might be needed for different problems, making relational representation learning much more complex than the i.i.d. case. Given the goal of unsupervised representation learning, the best solution would be to address multiple aspects simultaneously. However, existing relational clustering methods impose a fixed bias (as discussed in the later sections), and thus are not entirely appropriate for this task. *As the answer to the second question, in the next section we will describe an approach that accounts for multiple interpretations of relational similarity.* Whereas MRC relies on probabilistic model to find a good clustering, we explicitly find clusters taking different similarity interpretations into account.

The final question did not receive much attention within the deep learning community, where the number of clusters is fixed in advance. Consequently, these methods are known to be very sensitive to the parameter settings, and certainly require some domain knowledge in order to choose the optimal number of clusters. As our strategy is to cluster vertices and hyperedges depending on their domains, choosing the optimal number of clusters would be substantially more difficult, as separate numbers should be chosen for each domain. Therefore, in this work we choose to employ an adaptive method for clustering selection, i.e., learn the number of clusters from the data.

We propose to use a *difference-like criterion* Vendramin, Campello, and Hruschka (2010). Difference-like criteria assess relative improvements on some relevant characteristic of the data (e.g. within-cluster similarity) over a set of successive data partitions produced by a given iterative procedure. Such criteria can be monotonically increasing as the number of clusters varies from k=2 to k=N, as illustrated in Fig. 1. In these cases, one usually tries to identify a prominent *knee* or *elbow* that suggests the most appropriate number of clusters existing in the data. Following the suggestion in Vendramin, Campello, and Hruschka (2010), we choose the number of clusters as the one that achieves the highest value of the following formula:

$$S(k) = \left|\frac{C(k-1)-C(k)}{C(k)-C(k+1)}\right| - \alpha \cdot k \tag{1}$$

where $C(k)$ is the intra-cluster similarity, k is the number of clusters and α a user specified parameter. The second term is a penalty on the number of clusters, where the value of α specifies the preference on the number of clusters: a larger value states a preference for a smaller number of clusters.

Once the clustering are obtained, we will represent them in the following form. For each cluster of vertices we create a unary predicate in form of `cluster`ID`(vertex_type)` which takes an entity represented by the vertex as an argument. ID is a unique identifier created for each cluster. Such predicate is instantiated to `true` if a vertex belongs to a cluster, otherwise `false`. Similarly, for each cluster of hyperedges we create a n-ary predicate in form of `cluster`ID`(vertex_type`$_1$`,...,vertex_type`$_n$`)`, which takes an ordered set of n vertices as arguments. We refer to the cluster-induced representation as a $\mathcal{C}$-representation.

Type-based clustering over neighbourhood trees without explicit bias

This section introduces the clustering procedure employed to learn the new representation. Based on the hypergraph data representation discussed in the previous section, the clustering task we consider is the following: given a vertex type $t \in T_V$, partition the vertices of this type into clusters such that vertices in the same cluster tend to be similar, and vertices in different clusters dissimilar, for some subjective notion of similarity. In practice, it is of course not possible to use a subjective notion; one uses a well-defined similarity function, which hopefully approximates well the subjective notion that the user has in mind.

In the following two sections we describe a similarity measure that can be used in conjunction with any clustering algorithm. The main motivation behind the introduced similarity measure is a capability to address multiple aspects of relational similarity, as previously discussed, and control the bias with a small set of parameters. The similarity measure compares vertices according to their properties, as well as properties of their neighbourhoods. Therefore, we start by introducing a structure to compactly represent a neighbourhood of a vertex.

Neighbourhood tree

Consider a vertex v. A neighbourhood tree aims to compactly represent the neighbourhood of the vertex v and all relationships it forms with other vertices, and it is defined as follows. For every hyperedge E in which v participates, add a directed edge from v to each vertex $v' \in E$. Label each vertex with its attribute vector. Label the edge with the hyperedge type and the position of v in the hyperedge (recall that hyperedges are ordered sets). The vertices thus added are said to be at depth 1. If there are multiple hyperedges connecting vertices v and v', v' is added each time it is encountered. Repeat this procedure for each v' on depth 1. The vertices thus added are at depth 2. Continue this procedure up to some predefined depth d. The root element is never added to the subsequent levels.

Similarity measure

This section introduces a similarity measure for vertices of the hypergraph (we use the standard term *similarity measure*, though it actually reflects a dissimilarity: lower values indicate higher similarity). We assume the attributes have discrete domains.

The introduced similarity measure compares two vertices by comparing their distributions over attribute values and edge types in their neighbourhoods. This offers two main benefits over existing approaches. Firstly, given that in general a vertex participates in a non-fixed number of relations, where neighbours are described by (often overlapping) sets of attributes and their relations to other vertices, distributions are necessary to reliably model the neighbourhood of a vertex. For example, in the Mutagenesis example it makes a big difference whether a compound consists of 1 positively and 1 negatively charged atom, or 5 positively and 1 negatively charged atom. However, such information is typically ignored by the existing approaches. Secondly, comparing distributions avoids issues arising when comparing the vertices with very different number of neighbours (for example, properly normalizing for such scenario).

The similarity measure we propose relies on the similarity measure between multisets. In principle, any measure over multisets of elements can be used, however, in our experiments, we chose to use the χ^2-distance between multisets (Zhao, Robles-Kelly, and Zhou 2011), which is defined as:

$$d(A,B) = \sum_{x \in A \cup B} \frac{(f_A(x) - f_B(x))^2}{f_A(x) + f_B(x)} \tag{2}$$

where A and B are multisets and $f_S(x)$ is the relative frequency of element x in multiset S (e.g., for $A = \{a, b, b, c\}$, $f_A(a) = 0.25$ and $f_A(b) = 0.5$).

For any neighbourhood tree g, let $B_l(g)$ be the multiset of vertices at depth l in g, and $B_{l,t}(g)$ the multiset of vertices of type t at depth l in g. All the vertices in $B_{l,t}$ have the same attributes, and each vertex assigns one value to each attribute; thus, for each attribute a, a multiset of values $B_{l,t,a}(g)$ is obtained. Let $B_{l,e}(g)$ be the multiset of labels of edges originating at vertices at depth l. Let $\mathcal{N}$ be the set of all neighbourhood trees corresponding to the vertices of interest in a hypergraph.

Let $norm(\cdot)$ be a *normalization operator*, defined as $norm(f(g_1, g_2)) = \frac{f(g_1,g_2)}{\max\limits_{g,g' \in \mathcal{N}} f(g,g')}$, i.e., the normalization operator divides the value of the function $f(g_1, g_2)$ of two neighbourhood trees g_1 and g_2 by the highest value of the function f obtained amongst all pairs of neighbourhood trees.

The similarity of two vertices v and v' is defined as the similarity of their neighbourhood trees g and g', equal to:

$$\begin{aligned} s(g,g) = & w_1 \cdot ad(g,g) + w_2 \cdot nad(g,g) + w_3 \cdot nd(g,g) \\ & + w_4 \cdot cs(g,g) + w_5 \cdot ed(g,g) \end{aligned} \tag{3}$$

where $\sum_i w_i = 1$ and

- *attribute-wise dissimilarity*

  $$ad(g,g') = norm\left(\sum_{a \in A(\tau(v))} \mathbb{I}[a(v) \neq a(v')] \right) \tag{4}$$

  measures the dissimilarity of the root elements v and v' according to their attribute-value pairs, where $\mathbb{I}$ is the indicator function,

- *neighbourhood attribute dissimilarity*

  $$nad(g,g') = norm\left(\sum_{l=1}^{d} \sum_{t \in T_V} \sum_{a \in A(t)} d(B_{l,t,a}(g), B_{l,t,a}(g')) \right) \tag{5}$$

  measures the dissimilarity of attribute-value pairs associated with the neighbouring vertices of the root elements, per level and vertex type,

- *neighbourhood dissimilarity*

  $$nd(g,g') = norm\left(\sum_{l=1}^{\#levels} \sum_{t \in T_v} d(B_{l,t}(g), B_{l,t}(g')) \right) \tag{6}$$

  measures the dissimilarity of two root elements according to the vertex identities in their neighbourhoods, per level and vertex type,

- *connection dissimilarity*

  $$cs(g,g') = 1 - norm\left(|\{v \in B_0(g) | v \in B_1(g')\}| \right) \tag{7}$$

  reflects how many different edges exists between the two root elements, and

- *edge distribution dissimilarity*:

  $$ed(g,g) = norm\left(\sum_{l=1}^{\#levels} d(B_{l,e}(g), B_{l,e}(g)) \right) \tag{8}$$

  measures the dissimilarity over edge types present in the neighbourhood trees, per level.

Each component is normalized to the scale of 0-1 by the highest value obtained amongst all pairs of vertices, ensuring that the influence of each factor is proportional to its weight. The weights w_{1-5} in Equation 3 allow one to formulate a bias through the similarity measure.

The above described similarity measure over multisets allows one to define a similarity of hyperedges, as well. As stated, a hyperedge is nothing more than an ordered set of vertices, and can therefore be transformed in an ordered set of neighbourhood graphs. One can extend the aforementioned similarity measure to operate on ordered sets of multisets, but an easier solution is to combine a set of multisets into a single multiset. In the experiments we will simple combine the multisets by concatenating all the elements of multisets into a single one.

In (Dumancic and Blockeel 2016), we compare the proposed similarity measure with a wide range of relational similarity measures , as well as a number of graph kernels. From the perspective of representation learning, the most attractive property of RCNT is the ability to formulate a bias through the similarity measure. As the goal of representation learning is to extract *better* representation of data before the classification task is known, it is important to abstract multiple aspects of the data. That is especially true for relational data where it is not known in advance whether attributes or relations are important.

Preliminary results with TILDE

Datasets. We have used the following five datasets to evaluate the potential of this approach. The IMDB[1] dataset describes a set of movies with people acting in or directing them. The UW-CSE[2] dataset describes the interactions of employees at the University of Washington and their roles, publications and the courses they teach. The Mutagenesis[3] dataset describes chemical compounds and atoms they consist of. The WebKB[4] dataset consists of pages and links collected from the Cornell University's webpage. Terrorists[5] (Sen et al. 2008), describes terrorist attacks each assigned one of 6 labels indicating the type of the attack.

Evaluation procedure. We are interested in answering the following research questions: (1) *does $\mathcal{C}$-representation improves the performance of a relational classifier, compared to using the original representation of a dataset?*, and (2) *how does it compare to MRC, which is the closest related work?* More concretely, we are interested whether $\mathcal{C}$-representation improves the accuracy of the model, or makes it more compact while not affecting the accuracy. In order to do so, we use TILDE (Blockeel and De Raedt 1998) on the original and the $\mathcal{C}$-representation of several classification datasets. When creating the $\mathcal{C}$-representation of each dataset, we use hierarchical and spectral clustering with the adaptive clustering selection procedure. For each dataset, we use the following set of weights w_{1-5} from Equation 3: (0.5,0.5,0.0,0.0,0.0), (0.0,0.0,0.33,0.33,0.34), (0.2,0.2,0.2,0.2,0.2). The main motivation behind using these parameter sets is to address multiple similarity aspect that might exist in the dataset, as previously discussed. Thus, the first set of weights uses only the attribute information, the second one only the link information, while the last one combines every component. The final $\mathcal{C}$-representation is then a concatenation of all clusterings obtained with different parameters.

As a complexity measure of a model we use the number of nodes a trained TILDE model has. We use the following values for the α parameter in Equation 1: $0.1, 0.05, 0.01$. In the case of MRC, we varied the λ parameter in the following range: $\{-1, -5, -10\}$[6]. The λ parameter has the same role as α in the proposed approach, affecting the number of clusters chosen for each domain. Both training and test accuracies are reported. These results are obtained by 10-fold cross validation. In that case, we use $N-1$ folds to extract the $\mathcal{C}$-representation, and assign the elements in the remaining fold to the closest clusters in the $\mathcal{C}$-representation.

For the IMDB dataset we use person's role (actor or director) as a target. For the UW-CSE dataset, a person's employee status (student or professor) is used. For the Mutagenesis dataset, we predict whether a molecule is mutagenic or not. For the WebKB and TerroristAttack datasets, we predict the type of a webpage (personal, departmental, staff,...) and an attack (kidnapping, arson, weapon,...), respectively. The predicates representing the target task were removed from the dataset when the $\mathcal{C}$-representations were created.

The IMDB and UW-CSE datasets are *easy* relational datasets, i.e., simple rules are enough for almost perfect performance. We use this dataset to see whether $\mathcal{C}$-representation will degrade the performance of the classifier when a classification task is relatively simple. The remaining datasets require complex models to address the classification tasks. Thus, when using the $\mathcal{C}$-representation, one hopes to gain performance, as well as obtaining a less complex model.

Results. The results are summarized in Table 1. The table shows the results when the original representation was used (white background), as well as the $\mathcal{C}$- and MRC representations (coloured background). Both training and test accuracies are presented, together with the complexity of the model. The table includes 6 $\mathcal{C}$-representations obtained with 3 values for the α parameter and 2 clustering algorithms, spectral (Ng, Jordan, and Weiss 2001) and hierarchical (Ward 1963), as well as 3 MRC representations obtained with different λ values.

By observing the results on the IMDB and UW-CSE datasets, one can see that when the classification task is relatively simple, $\mathcal{C}$-representation does not degrade the performance. One can see that the train and test accuracies are approximately the same regardless of which representation is used. Furthermore, the models trained on the $\mathcal{C}$-representation are consistently less complex than the ones trained on the original representation. The results with MRC are however less conclusive. On the IMDB dataset the accuracy is preserved while the complexity of a model is reduced. On the UWCSE dataset, however, the accuracy is reduced while the model complexity is significantly increased.

The results on the remaining datasets, suggest the following conclusions. Firstly, the results suggest that the $\mathcal{C}$-representation obtained by spectral clustering yields the overall best results with regards to the two goals of representation learning. This combination consistently yields sub-

---

[1] Available at http://alchemy.cs.washington.edu/data/imdb

[2] Available at http://alchemy.cs.washington.edu/data/uw-cse/

[3] Available at http://www.cs.ox.ac.uk/activities/machlearn/mutagenesis.html

[4] Available at http://alchemy.cs.washington.edu/data/webkb/

[5] Available at http://linqs.umiacs.umd.edu/projects//projects/lbc/

[6] Note that it is difficult to exactly match the values of α and λ as both methods operate on very different scales

Table 1: Accuracy comparison. The first row in the table indicates the performance when the original dataset representation was used. The coloured rows indicate the performance when the $\mathcal{C}$-representation was used. The first column then specifies the setup parameters: clustering algorithm (**S** for spectral, **H** for hierarchical clustering, **M** for MRC), and the α and λ parameter values. Both train and test accuracies are reported, as well as the complexity of the model.

Setup	**UWCSE**			**Muta**			**WebKB**			**Terror**			**IMDB**		
	Train	Test	Cplx	Train	Test	Cplx	Train	Test	Cplx	Train	Test	Cplx	Train	Test	Cplx
Original	1.0	0.99	3.0	0.85	0.75	21.4	0.77	0.43	54.8	0.72	0.64	51.2	1.0	1.0	2.0
S, $\alpha = 0.01$	0.99	0.99	1.9	0.84	0.73	21.4	0.79	0.58	38.9	0.64	0.63	43.2	1.0	1.0	1.0
S, $\alpha = 0.05$	0.99	0.99	1.8	0.81	0.81	15.2	0.80	0.58	39.2	0.63	0.62	34.8	1.0	1.0	1.0
S, $\alpha = 0.1$	0.99	0.99	1.9	0.8	0.8	3.0	0.78	0.57	42.9	0.64	0.63	37.4	1.0	1.0	1.0
H, α=0.01	0.99	0.98	1.9	0.79	0.75	8.9	0.68	0.52	35.1	0.49	0.48	29.2	1.0	1.0	1.0
H, α=0.05	0.99	0.99	1.8	0.79	0.76	8.2	0.66	0.52	31.4	0.49	0.47	29.2	1.0	1.0	1.0
H, $\alpha = 0.1$	0.99	0.99	1.9	0.79	0.76	8.2	0.67	0.51	34.1	0.49	0.47	29.2	1.0	1.0	1.0
M, λ=-1	0.95	0.93	21.0	0.6	0.6	0.0	0.71	0.5	67.6	0.74	0.64	138.7	1.0	1.0	1.0
M, λ=-5	0.97	0.95	25.9	0.74	0.63	23.5	0.73	0.5	67.3	0.58	0.5	126.5	1.0	1.0	1.0
M, λ=-10	0.98	0.96	13.7	0.72	0.72	35.0	0.72	0.53	50.7	0.57	0.51	102.1	1.0	1.0	1.0

Table 2: Vocabulary size for each representation, including the original one (S: spectral, H: hierarchical, M: MRC)

Setup	**UWCSE**	**Muta**	**WebKB**	**Terror**	**IMDB**
Original	10	12	56	107	5
S, $\alpha = 0.01$	88	36	26	39	31
S, $\alpha = 0.05$	64	28	20	29	28
S, $\alpha = 0.1$	49	19	18	28	28
H, α=0.01	91	22	23	22	26
H, α=0.05	86	22	16	22	26
H, $\alpha = 0.1$	73	25	14	22	26
M, λ=-1	183	535	253	318	49
M, λ=-5	140	346	111	116	38
M, λ=-10	49	224	56	91	18

stantially less complex models compared to when the original representation was used, as well as better performance, except on the TerroristAttack dataset when the performance is comparable. Moreover, these models often have lower or approximately the same training accuracies, while producing equally good or better results on the test set. *This suggests that the models trained on the $\mathcal{C}$-representation have better generalization properties.* Secondly, $\mathcal{C}$-representation obtained by hierarchical clustering yield no performance gain, but produces substantially less complex models. This suggests that the choice of the clustering algorithm itself plays a significant role in relational representation learning. Thirdly, though the results suggest that the performance with $\mathcal{C}$-representation is mostly not affected by the value of the α value, α influences the complexity of a model, especially when the spectral clustering was used. Finally, the representation obtained with MRC mostly does not improve the performance (except on the WebKB dataset) and consistently yields substantially more complex models.

Table 2 summarizes the vocabulary size of each representation. Though it might be difficult to draw general conclusions from these results, they point out when representation learning might be most helpful. When the original vocabulary is relatively small (IMDB, UW-CSE and Mutagenesis datasets), the proposed method increases the vocabulary size. This is the consequence of clustering each domain separately, and treating edges as an ordered set of vertices. Both the proposed approach and MRC generate larger vocabularies, with MRC typically generating a much larger vocabulary than the approach proposed here. On the datasets with larger vocabularies, $\mathcal{C}$-representations produce substantially smaller vocabularies, while MRC-representations typically produce vocabularies larger than the original one. The datasets with large vocabularies might benefit the most from the representation learning approaches, as smaller vocabularies make structure learning easier.

Conclusion

In this paper we present a method for unsupervised representation learning for relational data. To the best of our knowledge, this is the first approach that addresses that question. Our approach clusters both objects and relations to obtain a new representation, addressing multiple possible aspects of similarity interpretation within relational data. We introduce a novel similarity measure for relational similarity, developed with the intention of capturing multiple interpretations of similarity based on a hypergraph representation of relational data. Finally, an adaptive method for deciding on the number of clusters is presented. We then present preliminary results demonstrating the potential of representation learning with the TILDE learner. The results show that with the representation extracted by the clustering procedure, TILDE often yields more accurate and less complex models. We believe these results show potential of relational unsupervised representation learning.

Future work includes developing more sophisticated methods for choosing the optimal number of clusters and experiments with wider range of (probabilistic) relational learners. An interesting extension would be to create more layers of features created with the same method. Finally, an interesting question for future work is: *what makes a good relational representation?*

References

Bai, L.; Ren, P.; and Hancock, E. R. 2014. A Hypergraph Kernel from Isomorphism Tests. In *Proceedings of the 22nd International Conference on Pattern Recognition*, 3880–3885. IEEE.

Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H.; Montral, U. D.; and Qubec, M. 2007. Greedy layer-wise training of deep networks. In *NIPS*. MIT Press.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* 35(8):1798–1828.

Blockeel, H., and De Raedt, L. 1998. Top-down induction of first-order logical decision trees. *Artificial Intelligence* 101(12):285 – 297.

Coates, A.; Lee, H.; and Ng, A. 2011. An analysis of single-layer networks in unsupervised feature learning. In Gordon, G.; Dunson, D.; and Dudk, M., eds., *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *JMLR Workshop and Conference Proceedings*, 215–223. JMLR W&CP.

Craven, M., and Slattery, S. 2001. Relational learning with statistical predicate invention: Better models for hypertext. *Mach. Learn.* 43(1-2):97–119.

Davis, J.; Ong, I. M.; Struyf, J.; Burnside, E. S.; Page, D.; and Costa, V. S. 2007. Change of representation for statistical relational learning. In Veloso, M. M., ed., *IJCAI*, 2719–2726.

De Raedt, L. 2008. *Logical and relational learning*. Cognitive Technologies. Springer.

Dumancic, S., and Blockeel, H. 2016. An efficient and expressive similarity measure for relational clustering using neighbourhood trees. *ArXiv e-prints, 1604.08934*.

Emde, W., and Wettschereck, D. 1996. Relational instance based learning. In Saitta, L., ed., *Proceedings 13th International Conference on Machine Learning (ICML 1996), July 3-6, 1996, Bari, Italy*, 122–130. Morgan-Kaufman Publishers, San Francisco, CA, USA.

Fonseca, N. A.; Santos Costa, V.; and Camacho, R. 2012. Conceptual clustering of multi-relational data. In Muggleton, S. H.; Tamaddoni-Nezhad, A.; and Lisi, F. A., eds., *Inductive Logic Programming: 21st International Conference, ILP 2011, Windsor Great Park, UK, July 31 – August 3, 2011, Revised Selected Papers*, 145–159. Berlin, Heidelberg: Springer Berlin Heidelberg.

Frasconi, P.; Costa, F.; Raedt, L. D.; and Grave, K. D. 2014. klog: A language for logical and relational learning with kernels. *Artif. Intell.* 217:117–143.

Getoor, L., and Taskar, B. 2007. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Haussler, D. 1999. Convolution kernels on discrete structures. Technical Report UCS-CRL-99-10, University of California at Santa Cruz, Santa Cruz, CA, USA.

Hinton, G. E., and Salakhutdinov, R. R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Kok, S., and Domingos, P. 2007a. Statistical predicate invention. In *Proceedings of the 24th annual international conference on machine learning (ICML-2007*, 433–440.

Kok, S., and Domingos, P. 2007b. Statistical predicate invention. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, 433–440. New York, NY, USA: ACM.

Kok, S., and Domingos, P. 2008. Extracting semantic networks from text via relational clustering. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, ECML PKDD '08, 624–639. Berlin, Heidelberg: Springer-Verlag.

Kok, S., and Domingos, P. 2009. Learning markov logic network structure via hypergraph lifting. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, 505–512. New York, NY, USA: ACM.

Kok, S., and Domingos, P. 2010. Learning markov logic networks using structural motifs. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 551–558.

Kramer, S. 1995. Predicate Invention: A Comprehensive View. *Technical report.*

Muggleton, S., and Buntine, W. L. 1988. Machine invention of first-order predicates by inverting resolution. In Laird, J., ed., *Proceedings of the 5th International Conference on Machine Learning (ICML'88)*, 339–352. Morgan Kaufmann.

Muggleton, S., and Lin, D. 2013. Meta-interpretive learning of higher-order dyadic datalog: Predicate invention revisited.

Neville, J.; Adler, M.; and Jensen, D. 2003. Clustering relational data using attribute and link information. In *Proceedings of the Text Mining and Link Analysis Workshop, 18th International Joint Conference on Artificial Intelligence*, 9–15.

Ng, A. Y.; Jordan, M. I.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. In *Advances in neural information proicessing systems*, 849–856. MIT Press.

Perlich, C., and Provost, F. 2003. Aggregation-based feature invention and relational concept classes. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, 167–176. New York, NY, USA: ACM.

Popescul, A., and Ungar, L. H. 2004. Cluster-based concept invention for statistical relational learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, 665–670. New York, NY, USA: ACM.

Ranzato, M.; Boureau, Y.-L.; and LeCun, Y. 2007. Sparse feature learning for deep belief networks. In Platt, J. C.; Koller, D.; Singer, Y.; and Roweis, S. T., eds., *NIPS*, 1185–1192. Curran Associates, Inc.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Mach. Learn.* 62(1-2):107–136.

Sen, P.; Namata, G. M.; Bilgic, M.; Getoor, L.; Gallagher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI Magazine* 29(3):93–106.

Shervashidze, N., and Borgwardt, K. 2009. Fast subtree kernels on graphs. In Bengio, Y.; Schuurmans, D.; Lafferty, J.; Williams, C.; and Culotta, A., eds., *Advances in Neural Information Processing Systems 22 – Proceedings of the 2001 Neural Information Processing Systems Conference NIPS 2009, December 7-10, 2009 Vancouver, British Columbia, Canada*. Neural Information Processing Systems Foundation. 1660–1668.

Shervashidze, N.; Schweitzer, P.; van Leeuwen, E. J.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-lehman graph kernels. *J. Mach. Learn. Res.* 12:2539–2561.

Silverstein, G., and Pazzani, M. J. 1991. Relational clich'es: Constraining constructive induction during relational learning. In *Proceedings of the Eighth International Workshop on Machine Learning*, 203–207. Morgan Kaufmann.

Srinivasan, A.; Muggleton, S.; and Bain, M. 1992. Distinguishing exceptions from noise in non-monotonic learning. In *Proceedings of the 2nd International Workshop on Inductive Logic Programming*, 97–107.

Vendramin, L.; Campello, R. J. G. B.; and Hruschka, E. R. 2010. Relative clustering validity criteria: A comparative overview. *Stat. Anal. Data Min.* 3(4):209–235.

Wachman, G., and Khardon, R. 2007. Learning from interpretations: a rooted kernel for ordered hypergraphs. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, 943–950.

Wang, W. Y.; Mazaitis, K.; and Cohen, W. W. 2015. A soft version of predicate invention based on structured sparsity. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, 3918–3924.

Ward, J. H. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association* 58(301):236–244.

Witsenburg, T., and Blockeel, H. 2011. Improving the accuracy of similarity measures by using link information. In *Foundations of Intelligent Systems - 19th International Symposium, ISMIS 2011, Warsaw, Poland, June 28-30, 2011. Proceedings*, 501–512.

Wogulis, J., and Langley, P. 1989. Improving efficiency by learning intermediate concepts. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'89, 657–662. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Zhao, H.; Robles-Kelly, A.; and Zhou, J. 2011. On the use of the chi-squared distance for the structured learning of graph embeddings. In *Proceedings of the 2011 International Conference on Digital Image Computing: Techniques and Applications*, DICTA '11, 422–428. Washington, DC, USA: IEEE Computer Society.