# rsLDA: a Bayesian Hierarchical Model for Relational Learning

Claudio Taranto, Nicola Di Mauro and Floriana Esposito
Dipartimento di Informatica
Università degli Studi di Bari "Aldo Moro", Bari, Italy
Email: {claudio.taranto,ndm,esposito}@di.uniba.it

***Abstract*—We introduce and evaluate a technique to tackle relational learning tasks combining a framework for mining relational queries with a hierarchical Bayesian model. We present the novel rsLDA algorithm that works as follows. It initially discovers a set of relevant features from the relational data useful to describe in a propositional way the examples. This corresponds to reformulate the problem from a relational representation space into an attribute-value form. Afterwards, given this new features space, a supervised version of the Latent Dirichlet Allocation model is applied in order to learn the probabilistic model. The performance of the proposed method when applied on two real-world datasets shows an improvement when compared to other methods.**

## I. Introduction

Learning in domains that cannot be adequately represented with a propositional language needs effective machine learning techniques. These domains where the data are strongly interrelated and structured can be elegantly described with Statistical Relational Learning (SRL) [1] or Probabilistic Inductive Logic Programming (PILP) [2] languages that combine statistical learning techniques with relational (or first order logic) representations. The vast interest in SRL has resulted in a wide variety of different formalisms, models and probabilistic programming languages, such as probability logic based formalisms or modelling approaches concerning the combination of relational database models and graphical models, such as Probabilistic Horn Abduction (PHA) [3], Probabilistic Logic Programming (PLP) [4], Bayesian Logic Programming (BLP) [5], Logic Programs with Annotated Disjunctions (LPADs) [6], [7], Probabilistic Relational Models (PRMs) [8], Relational Markov Networks (RMNs) [9], and Markov Logic Networks (MLNs) [10].

Another possible perspective towards SRL consists in restricting expressiveness, this way allowing for more efficient learning and inference algorithms. This category includes for instance *dynamic propositionalization approaches* such as nFOIL [11], that integrates the naive Bayes probabilistic model with a relational rule learner, kFOIL [12], where a relational kernel function is learnt and defined in terms of a small set of interpretable relational features, and Lynx [13], that combines the naive Bayes probabilistic model with relational queries mining.

If on the one hand an expressive representation formalism allows one to deal with complex and structured data, on the other hand modern Bayesian analysis provides the *hierarchical Bayesian method* as a powerful tool for representing rich statistical models. Hierarchical modelling is a fundamental concept in Bayesian statistics, where the parameters are endowed with distributions which may themselves introduce new parameters (some parameters are partly determined from distributions defined by other parameters, named *hyperparameters*) [14].

The goal of this paper is to combine a hierarchical Bayesian model with relational learning. In particular, we propose a propositionalization approach for relational learning integrating the *Latent Dirichlet allocation* [15] (LDA) model. A way to tackle the task of relational learning corresponds to reformulate the problem into an attribute-value form and then applying a propositional learner [16]. The reformulation process may be obtained adopting a *feature construction* method, such as mining relational queries that can then be successfully used as new Boolean features [17], [18], [19].

LDA is a *mixed membership model*, generalising a finite mixture model, in which each data point is associated with multiple draws from a mixture model. The mixture model is composed by two levels. LDA was originally proposed in [15] as a probabilistic model for uncovering the underlying semantic structure of a document collection based on a hierarchical Bayesian analysis of the original texts. There is a finite mixture whose components can be viewed as representations of *topics*, and then a latent Dirichlet variable that provides a random set of mixing proportions for the underlying finite mixture. The idea of LDA is to model documents as arising from multiple topics, where a topic is defined to be a distribution over a fixed vocabulary of terms. $K$ topics are associated with a collections, and each document exhibits these topics with different proportions.

Similar in the spirit to other propositionalization Inductive Logic Programming [20] (ILP) approaches, where a relational problem is turned into a propositional one by computing a set of features and then using a traditional statistical learning system on the resulting representation, here we propose the rsLDA algorithm combining a propositionalization technique and LDA. In particular, given a set of relational examples, we firstly construct a set of relational features used to propositionalize the examples, and then the LDA statistical framework is used to learn a probabilistic model. rsLDA has been experimentally evaluated on a benchmark ILP problem.

The paper is organized as follows. Section II reports some works that are related to the proposed method. Section III

shows the proposed rsLDA method and Section IV reports its evaluation on two real world datasets when compared to other methods. Finally, Section V concludes the paper.

## II. Related works

This work may be correlated to that in [19], where the authors presented one of the first Inductive Logic Programming feature construction method. They firstly construct a set of features adopting a declarative language to constraint the search space and to find discriminant features. Then, these features are used to learn a classification model with a propositional learner.

The approach presented in this paper is related to dynamic propositionalization approaches such as nFOIL [21] and kFOIL [12]. nFOIL integrates ILP and naive Bayes by performing a covering search in which one feature (in form of a clause) is learned after the other, until adding further features does not yield improvements. The search heuristic is based on class conditional likelihood and clauses are combined with naive Bayes. kFOIL is a statistical relational learner that greedily learns a set of clauses in a general-to-specif way. It learns a logic kernel to be used within a kernel machine learning algorithm. The set of learned clauses defines a feature space representation of the input examples and a statistical learning algorithm is trained with such a representation.

Finally, another system similar to this reported in this paper is Lynx [13], originally proposed for relational sequence learning, that combines probabilistic feature construction and feature selection for relational learning. In a first phase it adopts a classical probabilistic feature construction approach, and then it adopts a wrapper feature selection approach, that uses a stochastic local search procedure, embedding a naive Bayes classifier to select an optimal subset of the constructed features. In particular, the optimal subset of patterns is searched using a Greedy Randomised Search Procedure (GRASP) and the search is guided by the predictive power of the selected subset computed using a naive Bayes approach.

## III. rsLDA: Relational Supervised Latent Dirichlet Allocation

In this section we report the components of our approach to combine ILP and LDA. Given a set of relational labelled examples, the first step is to adopt a propositionalization algorithm to reduce each relational example to an attribute-value description. This goal is obtained by a feature construction approach adopting a relational query mining algorithm, as reported in Section III-A. After having extracted the relevant relational features from the data, we can apply a supervised LDA to the corresponding propositionalized dataset.

### A. Relational query mining

Here we firstly briefly report the framework for mining relational queries introduced in [22] and then adopted in Lynx [13], that we use in this paper for feature construction from relational data.

*1) Logic Programming Concepts:* As a representation language we use first order logic. A first order *alphabet* consists of a set of *constants*, a set of *variables*, a set of *function symbols*, and a non-empty set of *predicate symbols*. Both function symbols and predicate symbols have a natural number (its *arity*) assigned to it. A *term* is a constant symbol, a variable symbol, or an $n$-ary function symbol $f$ applied to $n$ terms $t_1, t_2, \ldots, t_n$. An atom $p(t_1, \ldots, t_n)$ is a predicate symbol $p$ of arity $n$ applied to $n$ terms $t_i$. Both $l$ and its negation $\bar{l}$ are said to be (resp., positive and negative) *literals* whenever $l$ is an atom. Literals and terms are said to be *ground* whenever they do not contain variables.

A *substitution* $\theta$ is defined as a set of bindings $\{X_1 \leftarrow a_1, \ldots, X_n \leftarrow a_n\}$ where $X_i, 1 \leq i \leq n$ are variables and $a_i, 1 \leq i \leq n$ are terms. A substitution $\theta$ is applied to an expression $e$, obtaining the expression $(e\theta)$, by replacing all variables $X_i$ with their corresponding term $a_i$.

*2) Query mining and feature construction:* Query mining corresponds to the classical local pattern mining when applied to multi-relational database representations [23]. In this paper, Datalog is used to represent both queries and database.

We assume that there is a relational predicate $key(\boldsymbol{X})$ referring the examples to be characterised with queries, and a language $\mathcal{L}$ of patterns corresponding to the set of queries defined as $\{key(\boldsymbol{X}), l_1, \ldots, l_n\}$, where $l_i$ are positive atoms.

Query mining aims at finding all queries satisfying a set of selection functions $\phi_j$, and it can be formulated as follows [23]:

**Given** a language $\mathcal{L}$ containing queries of the form $\{key(\boldsymbol{X}), l_1, \ldots, l_n\}$, a database $\mathcal{D}$ including the relation $key(\boldsymbol{X})$, and a set of selection functions $\phi_j$

**Find** all queries $q \in \mathcal{L}$ such that $\phi_j(q, \mathcal{D}) = true$.

The classical selection function is the minimum frequency. In order to compute the frequency of a query it is important to define the concept of query subsumption.

Given $\Sigma = \mathcal{B} \cup U$, where $U$ is the set of conjunctive atoms corresponding to an example $e$, and $\mathcal{B}$ is a background knowledge, a query $q$ *subsumes* an example $e$ ($q \preceq e$), iff there exists an $\text{SLD}_{\text{OI}}$-deduction of $q$ from $\Sigma$.

An $\text{SLD}_{\text{OI}}$-deduction is an SLD-deduction under Object Identity [24]. In the Object Identity framework, within a clause, terms denoted by different symbols must be distinct, i.e. they must represent different objects of the domain.

One of the component of Lynx, that we used in this paper, is the feature construction process obtained by mining frequent queries with an approach similar to that reported in [19]. The algorithm for frequent query mining is based on the same idea as the generic level-wise search method, known in data mining from the Apriori algorithm.The level-wise algorithm performs a breadth-first search in the lattice of patterns ordered by a specialization relation $\preceq$.

Generation of the frequent queries is based on a top-down approach. The algorithm starts with the most general query $\{key(\boldsymbol{X})\}$. Then, at each step it tries to specialise all the candidate frequent queries, discarding the non-frequent queries

and storing those whose length is equal to the user specified input parameter `maxsize`.

For each new refined query, semantically equivalent queries are detected, by using the $\theta_{\text{OI}}$-subsumption relation, and discarded. In the specialization phase the specialization operator, basically, adds atoms to the query.

The query mining algorithm uses a background knowledge $\mathcal{B}$ containing a set of constraints, similar to that defined in SeqLog [25], corresponding to selection functions $\phi_j$ that must be true. In particular, some of the constraints in $\mathcal{B}$ are (see [22] for more details):

- `maxsize(M)`: maximal query length;
- `minfreq(m)`: the frequency of the query must be greater than $m$;
- `type(p)` and `mode(p)`, denote, respectively, the type and the input/output mode of the predicate's arguments `p`, used to specify a language bias;
- `posconstraint([`$p_1, p_2, \ldots, p_n$`])` (resp. `negconstraint([`$p_1, p_2, \ldots, p_n$`])`) specifies a constraint that the query must (resp. must not) fulfil;
- `atmostone([`$p_1, p_2, \ldots, p_n$`])` discards all the queries that make true more than one predicate among $p_1$, $p_2$,..., $p_n$;
- `key([`$p_1, p_2, \ldots, p_n$`])` specifies that the $key$ predicate of the queries must be one among the predicates $p_1, p_2, \ldots p_n$.

Given a set of relational examples $\mathcal{D}$ defined over a set of classes $C$, the *frequency* of a query $q$, $\text{freq}(q, \mathcal{D})$, corresponds to the number of examples $e \in \mathcal{D}$ such that $q$ subsumes $e$. The *support* of a query $q$ with respect to a class $c \in C$, $\text{supp}_c(q, \mathcal{D})$ corresponds to the number of examples $e \in \mathcal{D}$ subsumed by $q$ whose class label is $c$. Finally, the *confidence* of a query $q$ with respect to a class $c \in C$ is defined as $\text{conf}_c(q, \mathcal{D}) = \text{supp}_c(q, \mathcal{D})/\text{freq}(q, \mathcal{D})$.

The refinement of queries is obtained by using a refinement operator $\rho$ that maps each query to a set of its specializations, i.e. $\rho(q) \subset \{q' | q \preceq q'\}$ where $q \preceq q'$ means that $q$ is more general than $q'$ or that $q$ subsumes $q'$.

For each specialization level, before starting the next refinement step, Lynx may record all the obtained queries. Hence, it might happen that the final set includes a query $q$ that subsumes many other queries in the same set. However, the subsumed queries may have a different support, contributing in different way to a classification model.

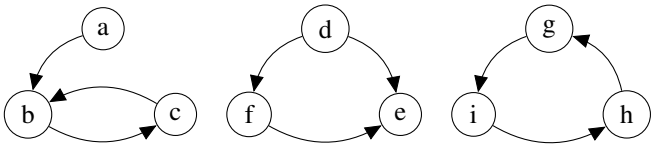
### B. Propositionalization step

Given a relational dataset $\mathcal{D}$, after having identified the set of frequent queries (relational features), now the task is how to use them as features in order to propositionalize $\mathcal{D}$.

Let $\mathcal{X}$ be the input space of relational examples, and let $\mathcal{Y} = \{1, 2, \ldots, C\}$ denote the finite set of possible class labels. Given a training set $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{D}$, where $X_i \in \mathcal{X}$ is a single relational example and $Y_i \in \mathcal{Y}$ is its corresponding label, the goal is to learn a function $h : \mathcal{X} \to \mathcal{Y}$ from $D$ that predicts the label for each unseen instance.

Let $\mathcal{Q}$, with $|\mathcal{Q}| = d$, be the set of features obtained as reported in the Section III-A (the queries mined from $\mathcal{D}$). For each example $X_k \in \mathcal{X}$ we can build a $d$-component vector-valued $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$ random variable where each $x_i \in \boldsymbol{x}$ is 1 if the query $q_i \in \mathcal{P}$ subsumes the example $x_k$, and 0 otherwise.

*Example* Suppose to have the following three examples:
$x_1$: { `arc(a,b), arc(b,c), arc(c,b)` }
$x_2$: { `arc(d,e), arc(d,f), arc(f,e)` }
$x_3$: { `arc(g,i), arc(i,h), arc(h,g)` }
corresponding to a logical representation of the following three graphs.




Suppose to have the following three queries:
$q_1$: { `arc(X,Y), arc(Y,Z)` }
$q_2$: { `arc(X,Y), arc(Y,X)` }
$q_3$: { `arc(X,Y), arc(Y,Z), arc(Z,X)` }.
Now, since $q_1 \preceq x_i$ $(i = 1, 2, 3)$, $q_2 \preceq x_1$ and $q_3 \preceq x_3$, we can build the following vector based representation for the examples $x_i$:

| | $q_1$ | $q_2$ | $q_3$ |
|---|---|---|---|
| $x_1$ | 1 | 1 | 0 |
| $x_2$ | 1 | 0 | 0 |
| $x_3$ | 1 | 0 | 1 |

### C. Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) has been proposed in [15] as a probabilistic model for uncovering the underlying semantic structure of a document collection based on a hierarchical Bayesian analysis of the original texts. Here we review the *supervised LDA* approach (sLDA), as proposed in [26], [27], that extends LDA to the supervised case, but referring to relational data and not just to documents. In particular, a relational example may be considered as a document, and the relational features subsuming the example will correspond to the words belonging to a document.

The idea of LDA is to model documents as arising from multiple topics, where a topic is defined to be a distribution over a fixed vocabulary of terms. $K$ topics are associated with a collection of documents, and each document exhibits these topics with different proportions.

LDA casts this intuition into a hidden variable model of documents. Hidden variable models are structured distributions in which observed data interact with hidden random variables. In a hidden variable model, one assumes that there is a hidden structure in the observed data, and the goal is to learn this structure using a posterior probabilistic inference approach. The interaction between the observed data and hidden structure is manifest in the probabilistic generative process associated with LDA, the random process that is assumed to have produced the observed data.

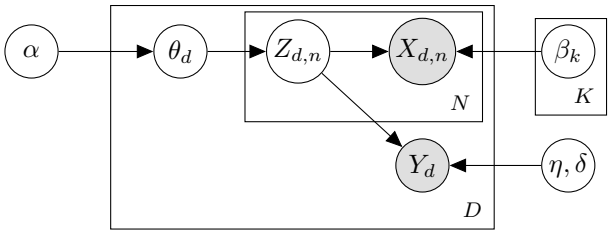



Fig. 1. The graphical model representing the generative process of sLDA

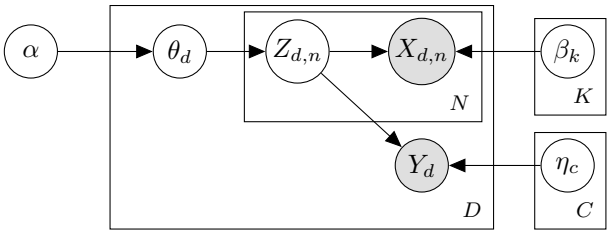



Fig. 2. The graphical model representing the generative process of Multi-Class sLDA.

*1) Relational topic model:* A *relational topic model* can be defined as a distribution over a relational dataset where each relational example is represented as a collection of discrete random variables $\boldsymbol{X} = X_{1:N}$, denoting its relational features. A *query* (resp., word in [15]) represents the basic feature of a relational example (resp., document in [15]). We treat the features of an example as arising from a set o latent relational topics. Examples in a dataset share the same set of $K$ topics, but each example uses a mixture of topics (the topic proportions) unique to itself. A random variable $Y$ is associated to examples denoting their response class value. The parameters of the model are the $K$ relational topics $\boldsymbol{\beta} = \beta_{1:K}$, a Dirichlet parameter $\alpha$, and the class response parameters $\eta$ and $\delta$.

*2) Supervised LDA:* The sLDA model [26] is represented by the following distributions:

$$\theta|\alpha \sim \mathsf{Dir}(\alpha) \quad (1)$$

$$z_n|\theta \sim \mathsf{Mult}(\theta) \quad (2)$$

$$x_n|z_n, \boldsymbol{\beta} \sim \mathsf{Mult}(\beta_{z_n}) \quad (3)$$

$$y|\boldsymbol{z}, \eta, \delta \sim \mathsf{GLM}(\bar{z}, \eta, \delta) \quad (4)$$

where $\bar{z} \triangleq \frac{1}{N}\sum_{n=1}^{N} z_n$ is the empirical topic frequency. The family of probability distributions corresponding to this generative process is depicted in the probabilistic graphical model reported in Figure 1. The distribution of the label is a generalised linear model (GLM) [28]:

$$p(y|\boldsymbol{z}, \eta, \delta) = h(y, \delta) \exp\left\{\frac{(\eta^\top \bar{z})y - A(\eta^\top \bar{z})}{\delta}\right\} \quad (5)$$

The sLDA generative process corresponds to:

1) draw topic proportions $\theta|\alpha \sim \mathsf{Dir}(\alpha)$;
2) for each feature:
   a) draw topic assignment $z_n|\theta \sim \mathsf{Mult}(\theta)$, and
   b) draw feature $x_n|z_n, \boldsymbol{\beta} \sim \mathsf{Mult}(\beta_{z_n})$;
3) draw label variable $y|\boldsymbol{z}, \eta, \delta \sim \mathsf{GLM}(\bar{z}, \eta, \delta)$.

As reported in [26], the GLM framework gives us the flexibility to model any type of label variable whose distribution can be written in exponential dispersion form.

Given the model parameters $\pi = \{\alpha, \boldsymbol{\beta}, \eta, \sigma\}$, the joint distribution of a topic mixture $\theta$, a set of $N$ topics $\boldsymbol{z}$, a set of $N$ features $\boldsymbol{x}$, and the label $y$ is given by:

$$p(\theta, \boldsymbol{z}, \boldsymbol{x}, y|\pi) = p(\theta|\alpha)\left(\prod_{n=1}^{N} p(z_n|\theta)p(x_n|z_n, \boldsymbol{\beta})\right) p(y|\boldsymbol{z}, \eta, \sigma) \quad (6)$$

The computational problems to be solved in order to analyse data with sLDA are the following. The first is the *posterion inference* to compute the conditional distribution of the latent variables at the example level given its features $\boldsymbol{x}$ and the corpus-wide model parameters. Second is the *parameter estimation* that estimates the Dirichlet parameters $\alpha$, the GLM parameters $\eta$ and $\delta$, and the topic multinomial $\boldsymbol{\beta}$ from a dataset of observed example-label pairs $\mathcal{D} = \{\boldsymbol{x}_d, y_d\}_{d=1}^{D}$. Finally is the *prediction* to predict the label $y$ from a newly observed example $\boldsymbol{x}$, given the model parameters.

*3) Multi-class sLDA:* In Equation 4 of the generative process for sLDA, a label variable for each example is assumed to be drawn from a GLM. In [26] the label variable is real valued and drawn from a linear regression. A continuous label (response) is not appropriate for our classification problem, where the class $c$ of each example is a discrete label. In this paper, we adopt the approach proposed in [29], named multi-class sLDA, where the class label response is drawn form a softmax regression:

$$y|\mathbf{z} \sim \mathrm{softmax}(\bar{z}, \eta), \quad (7)$$

that provides the following distribution

$$p(c|\bar{z}, \eta) = \exp(\eta_c^\top \bar{z}) / \sum_{i=1}^{C} \exp(\eta_i^\top \bar{z}), \quad (8)$$

where the set of parameters has been modified considering the set $C$ of class label coefficients $\eta_{1:C}$. Each $\eta_c$ is a $K$-vector or real values. The probabilistic graphical model corresponding to this modified generative process is reported in Figure 2. As we can see the difference regards the parameters generating the response variable, by adding the plate containing the $\eta_c$ parameters.

*a) **Posterior inference**:* As for LDA [15], the posterior distribution of the hidden variables given a model and a labelled example described as

$$p(\theta, \boldsymbol{z}|\boldsymbol{x}, y, \alpha, \boldsymbol{\beta}, \boldsymbol{\eta}) \quad (9)$$

is not efficiently computable, and a variational method to approximate it may be used [29]. A solution is to adopt a mean-field variational method that consider a simple family of distributions over the latent variables, indexed by free variational parameters, and try to find the setting of those parameters that minimises the Kullback-Leibler (KL) divergence to the true posterior [30].

Given $\pi = \{\alpha, \boldsymbol{\beta}, \boldsymbol{\eta}\}$, the *evidence lower bound* (ELBO) $\mathcal{L}(\cdot)$ to be maximised is

$$\log p(\boldsymbol{x}, y|\pi) \geq \mathcal{L}(\gamma, \phi, \pi) = \mathbb{E}_q[\log p(\theta, \boldsymbol{z}, \boldsymbol{x}, y|\pi)] + H(q) \tag{10}$$

where $H(q) = -\mathbb{E}_q[\log q(\theta, \boldsymbol{z})]$ is the entropy of the chosen variational distribution defined as

$$q(\theta, \boldsymbol{z}|\gamma, \phi_{1:N}) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{11}$$

where $\gamma$ is a $K$-dimensional Dirichlet parameter vector and each $\phi_n$ parametrises a categorical distribution over $K$ elements[1]. As reported in [15], [26], [29], by computing the derivative of $\mathcal{L}$ and by setting them equal to zero, it is possible to obtain the following ascent update equations:

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_n \tag{12}$$

$$\phi_{ni} \propto \exp(\Psi(\gamma_i) + \eta_{ci}/N - (h^\top \phi_n^{old})^{-1} h_i) \tag{13}$$

where

$$h^\top \phi_n = \sum_{l=1}^{C} \prod_{n=1}^{N} \left( \sum_{j=1}^{K} \phi_{ni} \exp(\eta_{lj}/N) \right),$$

$h = [h_1, \ldots, h_K]^\top$, and $\Psi(x)$ denotes the digamma function.

*b) **Parameter Estimation**:* [26] fits the parameters of the sLDA model with a variational expectation maximisation (EM) approach. Given a dataset $\mathcal{D} = \{(\boldsymbol{x}_d, y_d)\}_{d=1}^{D}$, variational EM optimises the corpus-level lower bound on the log likelihood of the data. Expectations are taken with respect to a example-specific variational distribution $q_d(\boldsymbol{z}, \theta)$.

The E-step estimates the approximate posterior distribution for each example-class pair using the variational inference algorithm, while the M-step maximises the corpus-level ELBO with respect to the model parameters.

The corpus loglikelihood to be maximised setting $\partial \mathcal{L}_{[\beta_{1:K}]}(\mathcal{D})/\partial \beta_{if} = 0$ and $\partial \mathcal{L}_{[\eta_{1:C}]}(\mathcal{D})/\partial \eta_{ic} = 0$, as reported in [26], [29], is

$$\mathcal{L}(\mathcal{D}) = \sum_{d=1}^{D} \log p(\boldsymbol{x}_d, y_d|\Theta). \tag{14}$$

Assuming a symmetric Dirichlet, by fixing $\alpha = t/K$ where K is the number of topics and $t \in \mathbb{Z}^+$, there is no need to estimate it. In the following experiments we set $t$ to 50.

*c) **Prediction**:* Given a new example $\boldsymbol{x} = x_{1:N}$ and a fitted model $\{\alpha, \boldsymbol{\beta}, \boldsymbol{\eta}\}$, we want to estimate the probability of the class label $y$ by replacing the true posterior $p(\boldsymbol{z}|\boldsymbol{x})$ with the variational approximation

$$p(y|\boldsymbol{x}) \geq \exp\left( \mathbb{E}_q[\eta_c^\top \bar{z}] - \mathbb{E}_q\left[ \log\left( \sum_{l=1}^{L} \exp(\eta_l^\top \bar{z}) \right) \right] \right) \tag{15}$$

[1]The topic assignment $Z_n$ is represented as a $K$-dimensional indicator vector, and hence $\mathbb{E}[Z_n] = q(z_n) = \phi_n$.

Since the second term of the exponent is constant with respect to the class label, the prediction rule is

$$c^* = \text{argmax}_c \; \mathbb{E}_q[\eta_c^\top \bar{z}] = \text{argmax}_c \; \eta_c^\top \bar{\phi}. \tag{16}$$

## IV. Experimental Results

In order to prove the validity of our proposed approach, we conducted experiments on the structural mutagenesis dataset [31] coming from the field of organic chemistry and on Alzheimer dataset [32].

Mutagenesis dataset describes a set of molecular compounds and the task is to predict whether a compound is mutagenic. The dataset is divided into two set: a regression friendly (r.f.) set consisting of 188 examples (125 positive and 63 negative examples) and a regression unfriendly (r.u.) consisting of 42 examples (13 positive and 29 negative examples). Here, the atom and bound structure only has been used. The structural representation is made up of atom and bond structures of the compounds described by the following predicates:

- `bond(compound,a1,a2,btype)`: stating that `compound` has a bond of `btype` between the atoms `a1` and `a2`;
- `atm(compound,atom,e,atype,c)`: stating that in `compound`, `atom` has element `e` of `atype` and partial charge `c`.

In Alzheimer dataset, the goal is to compare four desirable properties of drugs against Alzheimers disease. In each of the four subtasks, the aim is to predict whether a molecule is better or worse than another molecule with respect to the considered property: inhibit amine reuptake (686 examples), low toxicity (886 examples), high acetyl cholinesterase inhibition (1326 examples), and good reversal of scopolamine-induced memory deficiency (642 examples).

Each experiment involved four steps:

1) a feature construction phase via query mining conducted on the training examples;
2) a propositionalization step;
3) the sLDA model learning on the propositionalized training examples; and, finally,
4) the sLDA class prediction of the testing examples propositionalized with the feature obtained in the step 1.

For the query mining, in mutagenesis experiments we have set `maxsize` to 6, and `minfreq` to 0.1, while in alzheimer experiments we have set `maxsize` to 5, and `minfreq` to 0.01. Once the features have been extracted, then the propositionalization step has been done. sLDA[2] has been applied on the propositionalized representation to estimate the model and to perform the prediction on the testing examples.

Table I shows the parameters used for each experiment by sLDA: the value of the $\alpha$ paremeter, the maximum number of iterations or the convergence value for the EM stopping criterion, and the number of topics.

[2]We used the implementation of sLDA available at `http://www.cs.princeton.edu/~chongw/slda/`.

| Dataset | $\alpha$ | EM steps | EM conv. | topics |
|---|---|---|---|---|
| Mutagen. r.f. | 1 | 40 | $10^{-8}$ | 50 |
| Mutagen. r.u. | 3.3 | 40 | $10^{-8}$ | 15 |
| Alzh. amine | 0.5 | $\infty$ | $10^{-4}$ | 100 |
| Alzh. toxic | 0.25 | $\infty$ | $10^{-4}$ | 200 |
| Alzh. acetyl | 0.2 | $\infty$ | $10^{-4}$ | 250 |
| Alzh. memory | 0.3 | $\infty$ | $10^{-4}$ | 150 |

TABLE I
SETTINGS FOR SLDA.

| Dataset | kFOIL | nFOIL | rsLDA |
|---|---|---|---|
| Mutagen. r.f. | $77.0 \pm 14.5$ | $75.4 \pm 12.3$ | $85.6 \pm 1.0$ |
| Mutagen. r.u. | $85.7 \pm 35.4$ | $78.6 \pm 41.5$ | $85.0 \pm 3.0$ |
| Alzh. amine | $89.8 \pm 5.7$ | $86.3 \pm 4.3$ | $90.9 \pm 1.4$ |
| Alzh. toxic | $90.0 \pm 3.8$ | $89.2 \pm 3.4$ | $98.9 \pm 1.5$ |
| Alzh. acetyl | $90.6 \pm 3.4$ | $81.2 \pm 5.2$ | $95.3 \pm 1.4$ |
| Alzh. memory | $80.5 \pm 6.2$ | $72.9 \pm 4.3$ | $92.6 \pm 1.4$ |

TABLE II
AVERAGE PREDICTIVE ACCURACY RESULTS ON MUTAGENESIS FOR KFOIL, NFOIL AND RSLDA WITH A 10-FOLD CROSS VALIDATION.

Table II reports the average predictive accuracy obtained with a 10-fold cross validation on the two problems. We compared the results obtained by the preposed approach rsLDA to that of nFOIL and kFOIL, as that reported in [12]. We can note that rsLDA improves the accuracy values achieved by nFOIL and kFOIL.

Table III shows the number of features (queries), averaged on the 10 folds, obtained using query mining algorithm integrated in Lynx[3], we can notice a significant features reduction using rsLDA approach.

Table IV reports an example of the five top queries associate to two of the fifty topics used in the first fold of the mutagenesis r.f. problem, where $p(q_i|z_j)$ denotes the $\beta_{ij}$ value of the rsLDA model; $\mathrm{supp_p}$ and $\mathrm{supp_n}$ indicate, respectively, the support of the query on the positive and negative examples. The predicates `atm` and `bond` have been abbreviated in the table with `a` and `b` respectively.

## V. CONCLUSION

In this paper we considered the problem of statistical relational learning introducing the rsLDA algorithm, combing a Bayesian hierarchical model with relational learning. In particular, we proposed to solve the relational learning problem adopting two main phases: mapping the relational representation problem to a propositional one and the applying a supervised Latent Dirichlet Allocation approach to induce the statistical learning model.

In the first phase we adopted the query mining algorithm included in Lynx in order to find the relevant features from the relational examples and to use them to propositionalize the relational problem. In the second step we cast the statistical relational learning problem to learning a supervised LDA model for a propositional problem.

The evaluation of the proposed approach has been made by applying our proposed approach to a real world dataset and proving that its predictive accuracy is better than that obtained by other established similar systems.

[3]Available at `http://www.di.uniba.it/~ndm/lynx/`.

| Dataset | Lynx features | rsLDA topic |
|---|---|---|
| Mutagenesis r.f. | 189.4 | 50 |
| Mutagenesis r.u. | 163.7 | 15 |
| Alzheimer amine | 1671.2 | 100 |
| Alzheimer toxic | 2839.7 | 200 |
| Alzheimer acetyl | 2998.6 | 250 |
| Alzheimer memory | 2070.2 | 150 |

TABLE III
FEATURES REDUCTION ADOPTING RSLDA.

| Topic 10 |
|---|
| $q_{160}$: `a(A,B,h,3,C), a(A,D,h,3,C), b(A,B,E,1), b(A,E,D,1)`<br>$p(q_{160}\|z_{10}) = 0.113$, $\mathrm{supp_p} = 33$, $\mathrm{supp_n} = 20$ |
| $q_{69}$: `a(A,_,c,22,_),a(A,_,c,10,_)`<br>$p(q_{69}\|z_{10}) = 0.112$, $\mathrm{supp_p} = 34$, $\mathrm{supp_n} = 19$ |
| $q_{96}$: `b(A,_,B,1),a(A,B,c,10,_)`<br>$p(q_{96}\|z_{10}) = 0.111$, $\mathrm{supp_p} = 34$, $\mathrm{supp_n} = 20$ |
| $q_{111}$: `a(A,_,n,38,_),a(A,_,c,10,_)`<br>$p(q_{111}\|z_{10}) = 0.111$, $\mathrm{supp_p} = 34$, $\mathrm{supp_n} = 20$ |
| $q_{21}$: `a(A,_,o,40,_),a(A,_,c,10,_)`<br>$p(q_{21}\|z_{10}) = 0.111$, $\mathrm{supp_p} = 34$, $\mathrm{supp_n} = 20$ |
| **Topic 42** |
| $q_{183}$: `a(A,B,c,22,C),a(A,D,c,E,F),b(A,D,B,7), a(A,_,c,22,C),a(A,_,c,E,F)`<br>$p(q_{183}\|z_{10}) = 0.500$, $\mathrm{supp_p} = 103$, $\mathrm{supp_n} = 24$ |
| $q_{174}$: `b(A,B,C,7),b(A,C,D,1),a(A,B,c,_,_), a(A,C,c,22,_),a(A,D,n,38,_)`<br>$p(q_{174}\|z_{10}) = 0.258$, $\mathrm{supp_p} = 66$, $\mathrm{supp_n} = 27$ |
| $q_{163}$: `a(A,B,o,40,C),a(A,D,_,_,C),b(A,B,_,2), b(A,D,_,_)`<br>$p(q_{163}\|z_{10}) = 0.070$, $\mathrm{supp_p} = 49$, $\mathrm{supp_n} = 15$ |
| $q_{182}$: `a(A,B,c,22,C),a(A,D,c,E,_),b(A,D,B,7), a(A,_,c,22,C),a(A,_,c,E,_)`<br>$p(q_{182}\|z_{10}) = 0.045$, $\mathrm{supp_p} = 65$, $\mathrm{supp_n} = 23$ |
| $q_{154}$: `a(A,B,c,C,_),a(A,D,c,_,_),b(A,D,B,7), a(A,_,c,C,_)`<br>$p(q_{154}\|z_{10}) = 0.042$, $\mathrm{supp_p} = 72$, $\mathrm{supp_n} = 23$ |

TABLE IV
THE FIVE TOP CONFIDENT FEATURES (QUERIES) ASSOCIATED TO TWO SELECTED TOPICS LEARNED WITH SLDA FOR THE FIRST FOLD ON THE MUATAGENESIS R.F. DATASET (ATM AND BOND ARE ABBREVIATED, RESP., WITH A AND B).

## REFERENCES


[1] L. Getoor and B. Taskar, Eds., *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[2] L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton, Eds., *Probabilistic Inductive Logic Programming*, ser. LNCS. Springer, 2008, vol. 4911.

[3] D. Poole, "Probabilistic horn abduction and bayesian networks," *Artificial Intelligence*, vol. 64, pp. 81–129, 1993.

[4] R. Ng and V. S. Subrahmanian, "Probabilistic logic programming," *Journal Information and Computation*, vol. 101, pp. 150–201, 1992.

[5] K. Kersting and L. De Raedt, "Towards combining inductive logic programming with bayesian networks," in *11th International Conference on Inductive Logic Programming*, ser. LNCS, C. Rouveirol and M. Sebag, Eds., vol. 2157. Springer, 2001, pp. 118–131.

[6] J. Vennekens, S. Verbaeten, and M. Bruynooghe, "Logic programs with annotated disjunctions," in *Proceedings of 20th International Conference on Logic Programming*. Springer, 2004, pp. 431–445.

[7] F. Riguzzi and N. Di Mauro, "Applying the information bottleneck to statistical relational learning," *Machine Learning Journal*, 2011.

[8] L. Getoor, "Learning probabilistic relational models," in *Abstraction, Reformulation, and Approximation*, ser. LNCS, B. Choueiry and T. Walsh, Eds. Springer, 2000, vol. 1864, pp. 322–323.

[9] B. Taskar, P. Abbeel, M. Wong, and D. Koller, "Relational markov networks," in *Introduction to Statistical Relational Learning*, L. Getoor and B. Taskar, Eds. MIT Press, 2007.

[10] M. Richardson and P. Domingos, "Markov logic networks," *Machine Learning*, vol. 62, pp. 107–136, 2006.

[11] N. Landwehr, K. Kersting, and L. De Raedt, "nFOIL: integrating Naive Bayes and FOIL," in *Proceedings of the 20th national conference on Artificial intelligence*. AAAI Press, 2005, pp. 795–800.

[12] N. Landwehr, A. Passerini, L. De Raedt, and P. Frasconi, "Fast learning of relational kernels," *Machine Learning*, vol. 78, pp. 305–342, 2010.

[13] N. Di Mauro, T. M. Basile, S. Ferilli, and F. Esposito, "Optimizing probabilistic models for relational sequence learning," in *19th International Symposium on Methodologies for Intelligent Systems*. Springer, 2011, pp. 240–249.

[14] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis, Second Edition (Chapman & Hall/CRC Texts in Statistical Science)*, 2nd ed. Chapman and Hall/CRC, 2003.

[15] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, Jan 2003.

[16] S. Kramer, N. Lavrac, and P. Flach, "Propositionalization approaches to relational data mining," in *Relational Data Mining*, S. Dzeroski and N. Lavrac, Eds. Springer, 2001, pp. 262–291.

[17] L. Dehaspe, H. Toivonen, and R. King, "Finding frequent substructures in chemical compounds," in *4th International Conference on Knowledge Discovery and Data Mining*, R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, Eds. AAAI Press., 1998, pp. 30–36.

[18] R. D. King, A. Srinivasan, and L. DeHaspe, "Warmr: A data mining tool for chemical data," *Journal of Computer-Aided Molecular Design*, vol. 15, no. 2, pp. 173–181, 2001.

[19] S. Kramer and L. D. Raedt, "Feature construction with version spaces for biochemical applications," in *Proceedings of the 18th International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., 2001, pp. 258–265.

[20] S. Muggleton and L. De Raedt, "Inductive logic programming: Theory and methods," *Journal of Logic Programming*, vol. 19/20, pp. 629–679, 1994.

[21] N. Landwehr, K. Kersting, and L. D. Raedt, "Integrating naive bayes and foil," *Journal of Machine Learning Research*, vol. 8, no. 1, pp. 481–507, 2007.

[22] F. Esposito, N. Di Mauro, T. Basile, and S. Ferilli, "Multi-dimensional relational sequence mining," *Fundamenta Informaticae*, vol. 89, no. 1, pp. 23–43, 2008.

[23] L. Dehaspe, H. Toivonen, and R. D. King, "Finding frequent substructures in chemical compounds," in *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, R. Agrawal, P. E. Stolorz, and G. Piatetsky-Shapiro, Eds. AAAI Press, 1998, pp. 30–36.

[24] N. Di Mauro, T. Basile, S. Ferilli, F. Esposito, and N. Fanizzi, "An exhaustive matching procedure for the improvement of learning efficiency," in *Inductive Logic Programming: 13th International Conference (ILP03)*, ser. LNCS, T. Horváth and A. Yamamoto, Eds., vol. 2835. Springer, 2003, pp. 112–129.

[25] S. Lee and L. De Raedt, "Constraint based mining of first order sequences in SeqLog," in *Database Support for Data Mining Applications*, ser. LNCS, R. Meo, P. Lanzi, and M. Klemettinen, Eds. Springer, 2004, vol. 2682, pp. 154–173.

[26] D. M. Blei and J. D. McAuliffe, "Supervised topic models," in *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. MIT Press, 2007.

[27] ——, "Supervised topic models," *coRR*, no. arXiv:1003.0783v1, 2010.

[28] P. McCullagh and J. A. Nelder, *Generalized linear models (Second edition)*. London: Chapman & Hall, 1989.

[29] C. Wang, D. M. Blei, and F. fei Li, "Simultaneous image classification and annotation," in *Computer Vision and Pattern Recognition*, 2009, pp. 1903–1910.

[30] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine Learning*, vol. 37, pp. 183–233, 1999.

[31] A. Srinivasan, S. H. Muggleton, M. J. E. Sternberg, and R. D. King, "Theories for mutagenicity: a study in first-order and feature-based induction," *Artificial Intelligence*, vol. 85, pp. 277–299, 1996.

[32] R. D. King, M. J. E. Sternberg, and A. Srinivasan, "Relating chemical activity to structure: An examination of ilp successes," *New Generation Comput*, vol. 13, no. 3-4, pp. 411–433, 1995.