Towards Interpretable Probabilistic Classification Models for Knowledge Graphs

Nicola Fanizzi
Dipartimento di Informatica
Università degli studi di Bari Aldo Moro
Bari, Italy
nicola.fanizzi@uniba.it

Claudia d'Amato
Dipartimento di Informatica
Università degli studi di Bari Aldo Moro
Bari, Italy
claudia.damato@uniba.it

***Abstract*—Tackling the problem of learning probabilistic classifiers that can be used the context of knowledge graphs, we describe an inductive approach based on learning networks of Bernoulli variables. Namely, we consider the application of multivariate Bernoulli models, a simple one and a two-levels mixture model. In addition, we also consider a hierarchical model combining the multivariate Bernoulli model with a restricted Boltzmann machine as the first level. We show how such models can be converted into probabilistic rule bases ensuring more understandability. A preliminary empirical evaluation is presented to test the effectiveness of these models on a number of classification problems with different knowledge graphs.**

***Index Terms*—knowledge graph, semantic web, probabilistic model, rule base, classification, interpretability, XAI**

I. Introduction

Classification of individual resources in the context of *knowledge graphs* [1] is a fundamental task enabling several more complex applications. For lack of consensus on a standard notion of knowledge graph, we will focus on the general form of *Web ontologies*, i.e. knowledge bases of interlinked entities across the Web infrastructure. In this context, classification may turn out to be a difficult task, even in an approximate form, especially when it is required that the original semantics ought to be preserved as much as possible.

A popular research trend resorts to *representation learning* techniques [2] that aim at mapping entities and relations to embedding spaces where tasks like classification and link prediction can be defined in terms of linear algebra operations. The two main downsides with these methods are related to:

- the difficulty of incorporating in the model implicit knowledge that can be precisely extracted from the graph only through deductive reasoning;
- the scarce interpretability of the model, as the vector spaces representations of the embeddings turn out to be hardly relatable to the original features, so that tasks such as classification - or link prediction - manage the models as a sort of black-boxes.

Then, for the sake of interpretability, it is possible to resort to simpler yet effective probabilistic models (classifiers) ultimately relying on base functions of discrete variables that leverage on basic logic features.

Following some ideas appeared in the context of neural learning (e.g. see [5] [3]), it can be shown how to fit simple graphical models that can be also converted into probabilistic rule bases with a direct interpretation in terms of the original representation of the instances in the knowledge graph (ontology) and can be simplified to produce logical axioms.

Specifically, we will initially focus on the problem fitting multivariate Bernoulli classification models in which the common simplifying assumption of conditional independence of the features given the class is made, that is typical of the Naive Bayes classifiers [4]. Then we move on towards models that relax this assumption, considering hierarchical mixtures of (Bernoulli) variables. Both kinds of models are suitable for a direct usage for classification but, differently from more effective yet much less explainable statistical models, they also lend themselves to an easier interpretation, verification and even integration by domain experts, provided that the intended semantics of the ontology is known.

These generative models are suitable also for cases in which only incomplete data is available and may be exploited for multiple additional applications, such as knowledge graph completion, axiom (disjointness) discovery, clustering, anomaly detection, etc.

A preliminary experiment that aimed at testing feasibility and effectiveness of such models on a number of classification problems with different knowledge graphs (Web ontologies) is described. As a baseline, another probabilistic (yet less easily interpretable) model was also considered in a regularized form which enforces the sparseness of the coefficients of the linear combination

The remainder of the paper is organized as follows. We briefly recall some basics of the logical representation of the knowledge graphs expressed as Web ontologies in OWL-DL and then a simple Boolean encoding for the individual resources in terms of basic features / classes (Sect. II). Then the simpler model based on multivariate Bernoulli is presented in Sect. III providing a basis for more complex hierarchical models illustrated in Sect. IV. A preliminary empirical evaluation is described in Sect. V with a comparison of some instances of these models on a number of classification tasks on real ontologies. Finally Sect. VI discusses ongoing and future work.

II. Basics

Preliminarily, we assume familiarity with the basic notions and the standard notation of *Description Logics* (DL) [6] as in the following we will focus on *knowledge graphs* that can be represented through DL axioms and ultimately as OWL-DL ontologies.

Formally, let $\mathcal{K}$ be a DL *knowledge base* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, with the set $\mathcal{T}$ (the *TBox*) containing terminological axioms defining concepts and roles (relations) of the domain of interest, and the set $\mathcal{A}$ (the *ABox*) containing basic assertions, i.e. facts regarding the individuals whose collection will be indicated with $\mathsf{Ind}(\mathcal{K})$.

Boolean Encoding

Similarly to previous works on distance or kernel-based learning for these representations [7] [8], we will build the models upon a simplified representation of the individuals.

Let the Boolean set be $\mathbb{B} = \{0, 1\}$. Given an ordered set $\mathcal{C} = \{F_i \mid i = 1, \ldots, D\}$ of basic features, i.e. DL concepts occurring in in $\mathcal{K}$ or defined in terms of concepts and roles therein[1], we will consider individuals $\mathtt{a} \in \mathsf{Ind}(\mathcal{K})$ represented by the (sparse) Boolean vector $\mathbf{a} \in \mathbb{B}^D$ with:

$$a_i = \begin{cases} 1 & \mathcal{K} \vdash F_i(\mathtt{a}) \\ 0 & \mathcal{K} \vdash \neg F_i(\mathtt{a}) \end{cases} \quad i = 1, \ldots, D \tag{1}$$

where $\vdash$ denotes the underlying proof procedure. Note that a_i stands for the Boolean value corresponding to the definite *membership* of $\mathtt{a}$ to F_i (i.e. 1) or to its complement (0): in some cases a_i may be undefined as the logic procedure $\vdash$ could be unable to determine the individual membership under the standard semantics adopted in DL, which does not assume complete knowledge: *Open World Assumption* (OWA) [6].

These cases may be treated by replacing the missing values with a fixed constant $m \in]0, 1[$ reflecting the uncertain membership to F_i, e.g. one derived from *pseudo-counts* provided by experts or from observed frequencies of the instances of F_i. Alternatively, one might consider adopting an uninformative initialization for the missing values in the input vectors and then use a Bernoulli RBM (see Sect. IV-D) fitted on the dataset, as an encoder.

III. Multivariate Bernoulli Model

In line with similar approaches for continuous distributions (e.g. see [3]), we start from a simple model as a network of Bernoulli units, then we discuss how to fit them and other useful tasks.

With individuals represented as D-dimensional binary tuples, and a variable y indicating the membership w.r.t. a target class C, their distribution can be modeled as a *(Naive Bayes) Multivariate Bernoulli Model* (MBM), i.e. a joint distribution[2] of D Bernoulli variables $\mathbf{x} = (x_1, x_2, \ldots, x_D)^T$ supposed to be conditionally independent given y:

$$P(\mathbf{x}|y) = \mathrm{Ber}_D(\mathbf{x}|\mathbf{p}_y) = \prod_{i=1}^{D} \mathrm{Ber}(x_i|p_{yi}) = \prod_{i=1}^{D} (p_{yi})^{x_i}(1 - p_{yi})^{1-x_i} \tag{2}$$

with parameters $p_{yi} = P(x_i = 1|y)$, conditional probabilities of $x_i = 1$ (i.e. of individual $\mathtt{x}$ to belong to the i-th feature concept) given the value of y. More specifically, we define the conditional probability of the i-th input being set when $y = 1$ (indicating the membership to C) as $p_{1i} = P(x_i = 1|y = 1) = P(x_i = 1|C)$ and the probability of the i-th input being set when $y = 0$ (indicating the membership to $\neg C$) as $p_{0i} = P(x_i = 1|y = 0) = P(x_i = 1|\neg C)$.

Fig. 1 depicts the graphical representation of the classification model. Besides of the conditional distributions $P(\mathbf{x}|y)$, it requires a *prior* for the output variable y, i.e. an extra parameter $\pi = \pi^1 = P(y = 1)$, hence one can also write $\pi^0 = 1 - \pi = P(y = 0)$.

Note that the binary setting for the output y can be simplified by removing the subscript y as it is possible to focus on the likelihood of one value (typically $y = 1$). This naturally extends to a multiple-class setting where y indicates one of several mutually disjoint target classes, say $C_1, \ldots, C_T$. In the following we will keep the subscript even though we continue to focus on the binary classification setting.

A. Classification

A classification problem amounts to estimating the state of the output y for the Boolean vector $\mathbf{x}$ corresponding to an input individual $\mathtt{x}$. Given the model parameters, i.e. prior $P(y)$ and $P(\mathbf{x}|y)$, the posterior of the class-membership distribution can be computed using Bayes' rule and Eq. (2):

$$P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} = \frac{\pi^y \mathrm{Ber}_D(\mathbf{x}; \mathbf{p}_y)}{P(\mathbf{x})} \tag{3}$$

Then the membership to be predicted is the one maximizing $P(y|\mathbf{x})$:

$$\hat{y}(\mathbf{x}) = \operatorname*{argmax}_{y \in \mathbb{B}} P(y|\mathbf{x})$$

which is equivalent to the decision procedure:

$$\mathcal{K} \mid\sim \begin{cases} C(\mathtt{x}) & \textit{if } P(y = 1|\mathbf{x}) > 1/2 \\ \neg C(\mathtt{x}) & \textit{otherwise} \end{cases}$$

Note that a different symbol $\mid\sim$ with respect to $\vdash$ is used to denote the classification procedure.

Here the mid-point 0.5 was adopted for simplicity, but a more complex decision procedure can be devised, including a case of *rejection* when the probability is close to this value. This can be defined in terms of a cost-sensitive decision using a threshold θ.

[1] A set of primitive concepts (i.e. classes), also including those obtained as restrictions on roles, such as $\exists R.C$. In the experiments we will use those at the bottom of the subsumption hierarchies determined by the axioms in $\mathcal{T}$.

[2] Also called the *Bernoulli product model*, or the *binary independence model*.

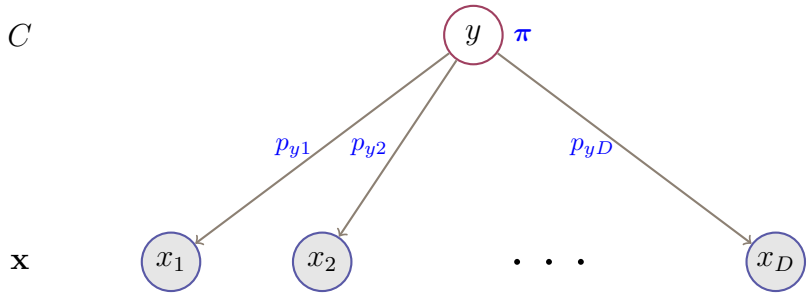

Fig. 1. Naive Bayes Multivariate Bernoulli Model network

B. Rules and Conjunctive Concepts Definition

A probabilistic *conjunctive classification rule* for C may be defined as follows:

IF $\mathcal{K} \mid\sim C(\mathrm{x})$ with prior π THEN
(the x_i's are independently Bernoulli distributed)
AND $x_1 = 1$ with probab. p_{11}
AND $x_1 = 0$ with probab. $1 - p_{11}$
AND $x_2 = 1$ with probab. p_{12}
AND $x_2 = 0$ with probab. $1 - p_{12}$
...
AND $x_D = 1$ with probab. p_{1D}
AND $x_D = 0$ with probab. $1 - p_{1D}$

which can be simplified considering for each $i = 1,\ldots,D$ only the more probable conjuncts, i.e. with values of x_i's corresponding to $\max(p_{1i}, 1 - p_{1i})$. Rules for the prediction of the complement $\neg C$ can be formed analogously, in terms of the parameters p_{0i} and $\pi^0 = 1 - \pi$.

Given a threshold $\theta \in [0.5, 1]$, a logic definition for the target concept may be extracted from a MBM considering the sets $F^+ = \{i \in \{1,\ldots,D\} \mid p_{1i} > \theta\}$ of feature concepts correlated with a positive membership and $F^- = \{i \in \{1,\ldots,D\} \mid 1 - p_{1i} > \theta\}$ of feature concepts correlated with a negative membership. Then we can define the axiom:

$$C \sqsubseteq \bigsqcap_{i \in F^+} F_i \sqcap \bigsqcap_{j \in F^-} \neg F_j$$

which can be proposed to a domain expert for its validation and possible inclusion to define the target class C. Note that only a subset of basic features will be used.

C. Fitting the Parameters

We assume the availability of a complete *training set* $\mathbf{T} = \langle \mathbf{X}, \mathbf{y} \rangle = \{(\mathbf{x}^t, y^t)\}_{t=1}^N$ where $\mathbf{x}^t$ is the encoding of an individual $\mathrm{x}^t \in \mathsf{Ind}(\mathcal{K})$ and $y^t \in \mathbb{B}$ indicates the actual membership to the target class C.

This NB classifier can be trained by optimizing the MLE (or the MAP estimate) for the parameters $\mathbf{\Pi} = \{\boldsymbol{\pi}, \mathbf{p}_y\}$. The probability for a single example is given by:

$$P(\mathbf{x}^t, y^t | \mathbf{\Pi}) = P(y^t | \boldsymbol{\pi}) P(\mathbf{x}^t | y^t, \mathbf{p}_{y^t}) = \pi^{y^t} \mathrm{Ber}_D(\mathbf{x}^t; \mathbf{p}_{y^t})$$

hence the *log-likelihood* $\mathcal{L} = \log P(\mathbf{T} | \mathbf{\Pi})$ can be written as

$$\begin{aligned}\mathcal{L} &= \log \prod_{t=1}^N P(\mathbf{x}^t, y^t | \mathbf{\Pi}) \\ &= \sum_{y \in \mathbb{B}} N_y \log \pi^y + \sum_{y \in \mathbb{B}} \sum_{t : y^t = y} \sum_{i=1}^D \log \mathrm{Ber}(x_i^t; p_{y^t i})\end{aligned}$$

where $N_b = \sum_t 1(y^t = b),\ b \in \mathbb{B}$ are the counts of training examples for either membership case. Then, the MLE for the proportions of the prior $P(y)$ can be estimated as:

$$\hat{\pi}^y = N_y / N.$$

As all input features are binary, and $x_i | y \sim \mathrm{Ber}(p_{yi})$, the MLE for the parameters are

$$\hat{p}_{yi} = N_{yi}/N \qquad y = 0, 1;\ i = 1, \ldots, D$$

with $N_{yi} = \sum_t 1(x_i^t = 1, y^t = y)$.

Actually, one does not expect the features to be independent, even conditional on y. However, even if the assumption does not hold true, the model has been proven to be effective in terms of performance (as discussed, e.g., in [4]), given its only $O(D)$ parameters for a binary problem with D features, and hence it is less prone to overfitting [9]. To better avoid this problem, one may resort to a Bayesian approach considering Dirichlet and Beta conjugate priors [9].

Alternatively, the parameters can be adjusted to minimize the (training *squared*) *error*:

$$E(\mathbf{T}) = \sum_{t=1}^N \left[y^t - \hat{y}(\mathbf{x}^t) \right]^2 \tag{4}$$

To this purpose, a *Gradient Descent* optimization procedure can be used.

In case of incomplete training data, which is very likely in this context, an *Expectation-Maximization* (EM) procedure [10] may be exploited.

IV. Mixture Models

As the features for a class are likely to be correlated in general, $P(\mathbf{x}|y)$ cannot be described by a single model, then the resulting classifier may turn out to be inaccurate when the underlying assumptions cannot be made.

We may allow for more than a single approximation of the conditional distribution for either membership case. This can

be modeled by allowing additional multivariate Bernoullis to approximate the conditional distribution, introducing an additional K-dimensional[3] intermediate (hidden) layer of binary indicator variables $\mathbf{z}$, i.e. nodes $\{z_k\}_{k=1}^K$, corresponding to a different $P_k(\mathbf{x}|z_k)$.

The resulting network is a *Mixture of Multivariate Bernoulli Model* (MMBM) whose structure is depicted in Fig. 2. In this case, given the model parameters $\mathbf{\Pi} = \{\pi, \boldsymbol{\pi}_1, , \ldots, \boldsymbol{\pi}_K, \mathbf{p}_1, \ldots, \mathbf{p}_K\}$:

$$P(\mathbf{x}|\mathbf{\Pi}) = \sum_{k=1}^{K} P(\mathbf{x}, z_k|\mathbf{\Pi}) = \sum_{k=1}^{K} P(z_k|\mathbf{\Pi}) P_k(\mathbf{x}|z_k, \mathbf{\Pi}) = \sum_k \pi_k \text{Ber}_D(\mathbf{x}; \mathbf{p}_k) \tag{5}$$

Alternatively, this model can be also described in the form of a *Hierarchical Mixture of Experts* [5].

A. Fitting the Parameters

The network parameters can be determined using a *training set* $\mathbf{T} = \{(\mathbf{x}^t, y^t)\}_{t=1}^N$ where $\mathbf{x}^t$ is the encoding of an individual $\mathtt{x}^t \in \mathsf{Ind}(\mathcal{K})$ and $y^t \in \mathbb{B}$ indicates its actual membership to the target class C. With a complete training set, learning the parameters of the two layers in the model by means of the minimization of an *error* function would require a form of *backpropagation*.

Alternatively, one may aim at maximal likelihood estimates of the training data by means of an *Expectation-Maximization* procedure, as proposed also for the simpler model. Let us consider the general case of incomplete data in which $\mathbf{z}^t$ (and possibly some y^t) is unknown. In this case, given the model parameters $\mathbf{\Pi}$, the complete log-likelihood is:

$$\mathcal{L}(\mathbf{\Pi}|\mathbf{T}, \mathbf{Z}) = \log \prod_{t=1}^{N} P(\mathbf{x}^t|\mathbf{\Pi}) = \sum_{t=1}^{N} \sum_{k=1}^{K} z_k^t \left[\log \pi_k + P(\mathbf{x}^t|\mathbf{p}_k)\right]$$

Then the EM steps can be defined as follows:

- **E-step**: The values of the unknown variables and parameters are estimated from the current estimates. More specifically, the probability of $\mathbf{x}^t$ being generated by a certain component k has to be estimated (e.g. when $z_k = 1$, i.e. $y^t = 1$):

$$\hat{P}(z_k^t|\mathbf{x}^t) = \frac{\hat{\pi}_k P_k(\mathbf{x}^t|\hat{\mathbf{p}}_k)}{\sum_{h=1}^K \hat{\pi}_k P_k(\mathbf{x}^t|\hat{\mathbf{p}}_h)}. \tag{6}$$

[3]The structure depends on the choice of K, that can be made by exploiting a separate validation set, to maximize a score function like the BIC (*Bayesian Information Criterion*) score [9], for example.

- **M-step**: The values of the parameters are updated on the ground of the new estimates for $\hat{P}(z_k|\mathbf{x}^t)$ along the following rules:

$$\hat{\pi}_k = \frac{1}{N} \sum_{t=1}^{N} \frac{\hat{P}(z_k^t|\mathbf{x}^t)}{\sum_{h=1}^K \hat{P}(z_h^t|\mathbf{x}^t)} \tag{7}$$

$$\hat{\mathbf{p}}_k = \frac{\sum_{t=1}^N \hat{P}(z_k^t|\mathbf{x}^t)\mathbf{x}^t}{\sum_{t=1}^N \hat{P}(z_k^t|\mathbf{x}^t)} \tag{8}$$

B. Classification

To decide the membership to C for an input individual $\mathtt{x}$ with corresponding Boolean vector $\mathbf{x}$, using Bayes' rule and Eq. (5), the posterior is determined as:

$$P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} = \frac{\pi \sum_k \pi_k \text{Ber}_D(\mathbf{x}; \mathbf{p}_k)}{\pi \sum_k \pi_k \text{Ber}_D(\mathbf{x}; \mathbf{p}_k) + (1-\pi) \sum_k \pi_k \text{Ber}_D(\mathbf{x}; \mathbf{p}_k)} \tag{9}$$

Then, the decision procedure may be summarized as:

IF $P(y = 1|\mathbf{x}) > \theta$ THEN $\mathcal{K} \mid\sim C(\mathtt{x})$ ELSE $\mathcal{K} \mid\sim \neg C(\mathtt{x})$

In the following we will focus on the definition of disjunctive and conjunctive classification rules translating the whole hierarchical model.

C. Disjunctive Rules / Definitions

Given a MMBM, *disjunctive classification rules* for class C may be extracted by first defining the class in terms of the components:

$$C \sqsubseteq \bigsqcup_{\substack{k=1,\ldots,K:\\ z_k=1}} C_k$$

(this can be specified probabilistically with the class prior π and π_k) or also via the following rules:

IF $\mathcal{K} \mid\sim C_k(\mathtt{x})$ THEN
the x_i's are independently Bernoulli distributed
AND $x_1 = 1$ with p_{k1} AND $x_1 = 0$ with $1 - p_{k1}$
AND $x_2 = 1$ with p_{k2} AND $x_2 = 0$ with $1 - p_{k2}$
...
AND $x_D = 1$ with p_{kN} AND $x_D = 0$ with $1 - p_{kM}$

For the sake of interpretability, these rules may be simplified considering, for each i, only the more likely sub-case with a minimal threshold to be exceeded. A class definition can be given also for the complement $\neg C$ in terms of the sub-classes C_k corresponding to the cases when $z_k = 0$.

D. Bernoulli Restricted Boltzmann Machines

The bipartite graph at the bottom of the model can be regarded as the network structure of a Bernoulli *Restricted Boltzmann Machine* [11] (RBM). In this model, all units are binary stochastic units, hence the input data should either be binary, or real-valued between 0 and 1 signifying the probability that the visible unit would be turned on. The latent features extracted by a RBM can be fed into the top layer (or to a linear classifier such as a Perceptron or a SVM).

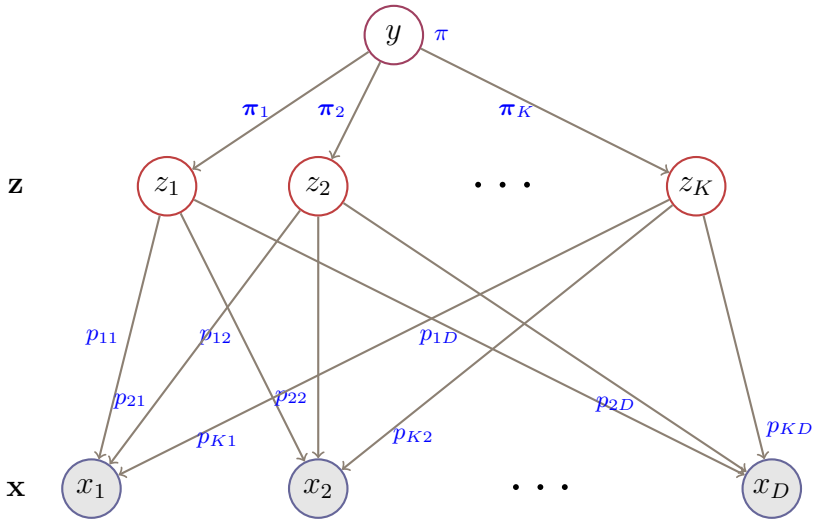

Fig. 2. Mixture of Multivariate Bernoulli Model Network

The parameter learning algorithm used to maximize the likelihood of the data, *Persistent Contrastive Divergence* [12], prevents the representations from straying far from the input data.

The joint probability of the model can be defined:

$$P(\mathbf{x}, \mathbf{z}) = \frac{1}{Z} \exp[-E(\mathbf{x}, \mathbf{z})]$$

where Z is a normalization constant and the *energy function* E measures the quality of a joint assignment

$$E(\mathbf{x}, \mathbf{z}) = -\sum_{i=1}^{D}\sum_{k=1}^{K} p_{ik} x_i z_k - \sum_i b_i x_i - \sum_k c_k z_k$$

with $\mathbf{b}$ and $\mathbf{c}$ as intercept vectors for the visible and hidden layers, respectively.

The conditional probability distribution of each unit is given by the logistic sigmoid activation function of the input it receives, i.e.: $P(x_i = 1|\mathbf{z}) = \sigma(\sum_k p_{ik} z_k + b_i)$ and $P(z_k = x|\mathbf{x}) = \sigma(\sum_i p_{ik} x_i + c_k)$.

We will exploit a combined classifier obtained by stacking a MBM on top of a Bernoulli RBM.

V. Preliminary Experiments

We will briefly present how the tested models have been implemented exploiting standard libraries then we will describe the setup of the hyper-parameters and finally the results of this empirical evaluation are presented.

A. Implementation

The implementation of two prototypical models to be tested has been carried out combining facilities provided by two Python packages, namely `Owlready2`[4] [13] and the well-known `Scikit-Learn`[5]. The former is intended for ontology-oriented programming and was used to manage the knowledge graphs, including loading and storage as well as reasoning (via an embedded version of the reasoner Pellet) which was used to initialize the binary encoding of the individuals in each graph. The latter library offered customizable versions of probabilistic models based on Bernoulli-distributed features. Besides various facilities for the preparation of the experiments were also employed (measures, cross-validation splittings, etc.) as described in the following subsection.

[4]https://pypi.org/project/Owlready2/

[5]https://scikit-learn.org/

B. Settings

Three models were tested in the experiment, namely a simple implementation of the MBM, its coupling with a Bernoulli RBM and, a baseline made up of a regularized *Logistic Regression* model with L1 penalization, enforcing

Three ontologies have been selected to provide the datasets employed as a testbed: Financial, New Testament Names (NTNames) and one generated from the Lehigh University Benchmark (LUBM).

For each ontology basic features (contained classes) have been extracted adding also classes as universal and existential restrictions each involving one the available relations (object properties). To define the final input Boolean encoding for the dataset a preliminary *feature selection* phase is performed to select the most informative features (those exhibiting a higher variance with respect to a definite threshold). Table I reports the numbers of classes and properties per KG as well as the numbers of features considered in the experiments.

TABLE I
Numbers of classes and properties per knowledge graph, generated and selected features

KGs	#classes	#obj.props.	#generated	#selected
Financial	59	16	75	13
NTNames	47	27	82	19
LUBM	43	25	75	13

TABLE II
OUTCOMES (AVERAGES ± STANDARD DEVIATIONS): PRECISION (P), RECALL (R), AND F_1-MEASURE (F_1)

		models		
knowledge graphs	*measures*	**MBM**	**RBM+MBM**	**LREG**
FINANCIAL	P	0.757 ± 0.050	0.734 ± 0.068	0.780 ± 0.086
	R	0.991 ± 0.024	0.942 ± 0.073	0.934 ± 0.097
	F_1	0.850 ± 0.037	0.812 ± 0.067	0.800 ± 0.132
NTNAMES	P	0.730 ± 0.138	0.702 ± 0.181	0.975 ± 0.042
	R	0.992 ± 0.014	0.975 ± 0.048	0.649 ± 0.337
	F_1	0.774 ± 0.122	0.747 ± 0.166	0.645 ± 0.335
LUBM	P	0.854 ± 0.177	0.850 ± 0.174	0.967 ± 0.098
	R	1.000 ± 0.000	0.994 ± 0.017	0.786 ± 0.295
	F_1	0.897 ± 0.119	0.892 ± 0.115	0.771 ± 0.310

Missing values in the input encoding have been treated by assuming an uninformative initialization for the likelihood of membership to the various basic F_i's.

Then 10 target concepts have been randomly defined, requiring that a minimal number of instances featuring positive, respectively negative, membership are available: this defines the correct classification of the individuals encoded in the target column.

The metrics that were measured are *precision*, *recall* and F_1-measure, considering the problems as binary (yet the test sets contained also unlabeled examples, i.e. individuals whose membership to the target concept cannot be determined by reasoning). The measures are averaged over the two classes weighted along the respective proportion of test examples.

For each learning problem, a 10-fold cross validation was performed randomly stratified splitting the examples (positive, negative and unlabeled). Ontologies, target concepts, code and output files of the experiments are publicly available[6].

The values for the hyper-parameters have been determined beforehand through grid-search on a per-ontology basis. A better performance can likely be achieved by performing randomized search on a per target concept basis (which is more time consuming).

C. Empirical Results

The outcomes of the various tests in terms of adopted measures are summarized in Table II. More details on the single experiments are contained in the output files.

Preliminarily, we observe that the results are averaged over the 10 problems per ontology. The diversity of the target concepts determined a relevant variance (standard deviation) especially in terms of precision and consequently of F_1-measure, particularly in the case of LUBM. Conversely, the results in terms of recall are quite high and stable for all cases considered. For a deeper insight in the outcomes of the single learning problems the output files can be consulted. Overall, it appears that the proposed models are more stable than the considered baseline.

Considering the F_1-measure, the general outcomes show that the MBM had the best performance, immediately followed by the MBM-RBM combination. This can be explained with the number of individuals involved in each dataset which was not extremely large, hence the simplest model, i.e. the one with less parameters to be fitted, was also less prone to overfitting. One can expect the combined model to perform with datasets containing more examples. The lower performance of the Logistic Regression was due to the amount of penalty adopted to produce simpler models (i.e. with few non-zero parameters). Separate experiments showed it would be easy to get optimal performance tuning the amount of regularization (or adopting a L2 penalization). However, there is cost to be paid with these models in terms of lesser interpretability.

Taking into account the precision measure, instead, we observe that Logistic Regression had better performance compared to the two other models, but this depends also on the much lower amount of recall (except for the case of the experiments with FINANCIAL).

As a final general consideration regarding the interpretability of the models learned, we recall that the basic features involved are those selected among those extracted per knowledge graph (see Table I). They correspond to the nodes at the input level of the related networks. Hence the number of boolean features that can be set in a model (i.e. those whose probability may exceed the threshold) is bounded above by the number of these features, D. Models with the additional level of latent features are more complex and hence less easily interpretable. A trade-off must be found between interpretability and effectiveness of the classification model, that is also determined by the number of examples available. This can be accomplished by fine-tuning hyperparameter K.

VI. CONCLUSIONS AND POSSIBLE EXTENSIONS

We proposed methods to generate probabilistic classifiers as models based on base functions of discrete variables by leveraging on basic logic features. They can easily be converted into probabilistic rules bases with a straightforward interpretation, hence enforcing the model explainability A preliminary experiment, aiming at testing the effectiveness of these models together with a related baseline, proved them promising and worth of further investigations.

[6]drive.google.com/drive/folders/1ynzv0idYHAfgNfHEwYomtRU7FtNxPeBy

The underlying generative models are suitable also for cases in which only incomplete data is available and may have multiple additional applications such as clustering for axiom discovery, anomaly detection, etc.

Various extensions are possible along different lines including:

- a deeper investigation of the related problem (due to the *open-world* semantics) of handling incomplete data, also considering decision procedures that admit rejection cases;
- the incorporation of existing rules in the probabilistic models (similarly to [3] for Gaussian models);
- tackling the problems of adapting the structure of the model and fitting the parameters through a Bayesian approach, considering Beta conjugate priors;
- adding Gaussian units (like in the Gaussian-Bernoulli RBMs) to be able to integrate restrictions on numerical datatypes.

REFERENCES

[1] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutiérrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A. N. Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," 2020, arXiv:2003.02320.

[2] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[3] V. Tresp, J. Hollatz, and S. Ahmad, "Representing probabilistic rules with networks of Gaussian basis functions," *Machine Learning*, vol. 27, pp. 173–200, 1997.

[4] P. M. Domingos and M. J. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Mach. Learn.*, vol. 29, no. 2-3, pp. 103–130, 1997.

[5] M. Jordan and R. Jacobs, "Hierarchical mixtures of experts and the EM algorithm," in *Proceedings of IJCNN*, vol. 2, 1993, pp. 1339–1344.

[6] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation and Applications*, 2nd ed. Cambridge University Press, 2007.

[7] C. d'Amato, N. Fanizzi, and F. Esposito, "Query answering and ontology population: An inductive approach," in *Proceedings of ESWC 2008*, ser. LNCS, vol. 5021. Springer, 2008, pp. 288–302.

[8] A. Rettinger, U. Lösch, V. Tresp, C. d'Amato, and N. Fanizzi, "Mining the Semantic Web - Statistical learning for next generation knowledge bases," *Data Min. Knowl. Discov.*, vol. 24, no. 3, pp. 613–662, 2012.

[9] K. P. Murphy, *Probabilistic Machine Learning: An introduction*. MIT Press, 2022, probml.ai.

[10] Z. Ghahramani and M. Jordan, "Supervised learning from incomplete data via an EM approach," in *Advances in Neural Information Processing Systems*, J. Cowan, G. Tesauro, and J. Alspector, Eds., vol. 6. Morgan-Kaufmann, 1993.

[11] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[12] T. Tieleman, "Training restricted boltzmann machines using approximations to the likelihood gradient," in *Proceedings of ICML 2008*, 2008, pp. 1064–1071.

[13] J. Lamy, "Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies," *Artificial Intelligence In Medicine*, vol. 80, pp. 11–28, 2017.

[14] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, 2nd ed. Wiley, 2007.