mLynx: Relational Mutual Information

Nicola Di Mauro, Teresa M.A. Basile, Stefano Ferilli, and Floriana Esposito

Department of Computer Science, LACAM laboratory
University of Bari "Aldo Moro", Via Orabona,4, 70125 Bari, Italy
{ndm,basile,ferilli,esposito}@di.uniba.it

Abstract. This paper represents a further steps in combining Information Theory tools with relational learning. We show how mutual information can be used to find relevant relational features.

1 Introduction

According to Tishby et al. [1], Information Theory provides a natural quantitative approach to the question of *relevant information* and an alternative view for Machine Learning because of the abstract and principled concept of *mutual information* (MI). For instance, they provided the Information Bottleneck (IB) method taking in mind that any learning process has to deal with the basic tradeoff between the complexity of the available data representation and the best accuracy that this complexity enables. The first approach using the IB method for Statistical Relational Learning (SRL) [2] has been proposed in [3] and this paper is a new step forward for information theoretic relational learning. We use the MI descriptor from Information Theory to propose a SRL method that learns the model by finding the most relevant features. Given a training dataset $\mathcal{D} = \{\mathbf{x}_i, c_i\}_{i=1}^n$ of n relational examples, characterized by a set of m relational features $X = \{f_i\}_{i=1}^m$, and a target discrete random variable c, generating class labels c_i, the aim of this paper is to find a subset of X that optimally characterizes the variable c minimizing the classifier's probability error. We want to find the maximal statistical dependency of the target class c on the data distribution in the selected subspace (*maximal dependency*), that usually corresponds to the *maximal relevance* of the features to the target class c.

Most of the Inductive Logic Programming (ILP) learning approaches builds models by searching for *good* relational features guided by a scoring function, such as in FOIL. In many SRL systems this *feature construction* process is combined with a discriminative/generative probabilistic method in order to deal with noisy data and uncertainty, such as in `kFOIL` [4], `rsLDA` [5], and Markov Logic Networks (MLNs) [6]. The combination may be *static* or *dynamic*. In the former case (*static propositionalization*), the constructed features are usually considered as boolean features and used offline as input to a propositional statistical learner; while in the latter case (*dynamic propositionalization*), the feature construction and probabilistic model selection are combined into a single process. We propose the `mLynx` system that, after a feature construction phase, stochastically searches, guided by the mutual information criterion, the set of the most relevant features minimizing a Bayesian classifier's probability error.

2 Feature construction and classification

This section reports the first components of the `mLynx` system, extending `Lynx` [7], that implements a probabilistic query-based classifier based on mutual information. Specifically, we start to report its feature construction capability and the adopted query-based classification model. The adopted mutual information feature selection approach will be presented in Section 3.

The first step of `mLynx` carries out a feature construction process by mining frequent Prolog queries (relational features) adopting an approach similar to that reported in [8]. The algorithm for frequent relational query mining is based on the same idea as the generic level-wise search method, performing a breadth-first search in the lattice of queries ordered by a specialization relation $\preceq$. The algorithm starts with the most general Prolog queries. At each step it tries to specialize all the candidate frequent queries, discarding the non-frequent ones and storing those whose length is less or equal to a user specified input parameter. Furthermore, for each new refined query, semantically equivalent patterns are detected, by using the θ_{OI}-subsumption relation, and discarded. In the specialization phase the specialization operator, basically, adds atoms to the query.

Now, having a set of relational features, we need a way to use them in order to correctly classify unseen examples. Given the training set $\mathcal{D} = \{\mathbf{x}_i, c_i\}_{i=1}^{n}$ of n relational examples, where c denotes the discrete class random variables taking values from $\{1, 2, \ldots, Q\}$, the goal is to learn a function $h : x \to c$ from $\mathcal{D}$ that predicts the label for each unseen instance. Let $\mathcal{Q}$, with $|\mathcal{Q}| = d$, be the set of features obtained in the first step of the `mLynx` system (the queries mined from $\mathcal{D}$). For each example $\mathbf{x}_k$ we can build a d-component vector-valued $\mathbf{x}_k = (x_k^1, x_k^2, \ldots, x_k^d)$ random variable where each $x_k^i \in \mathbf{x}_k$ is 1 if the query $q_i \in \mathcal{Q}$ subsumes example $\mathbf{x}_k$, and 0 otherwise, for each $1 \leq i \leq d$.

Using the Bayes' theorem, if $p(c_j)$ describes the prior probability of class c_j, then the posterior probability $p(c_j|\mathbf{x})$ can be computed from $p(\mathbf{x}|c_j)$ as

$$p(c_j|\mathbf{x}) = \frac{p(\mathbf{x}|c_j)p(c_j)}{\sum_{i=1}^{Q} p(\mathbf{x}|c_i)p(c_i)}.$$

Given a set of discriminant functions $\{g_i(\mathbf{x})\}_{i=1}^{Q}$, a classifier is said to assign the vector $\mathbf{x}$ to class c_j if $g_j(\mathbf{x}) > g_i(\mathbf{x})$ for all $j \neq i$. Taking $g_i(\mathbf{x}) = P(c_i|\mathbf{x})$, the maximum discriminant function corresponds to the *maximum a posteriori* (MAP) probability. For minimum error rate classification, the following discriminant function will be used: $g_i(\mathbf{x}) = \ln p(\mathbf{x}|c_i) + \ln p(c_i)$. Given $\mathbf{x} = (x_1, \ldots, x_d)$, we define $p_{ij} = \text{Prob}(x_i = 1|c_j)$ with the components of $\mathbf{x}$ being statistically independent for all $x_i \in \mathbf{x}$. The estimator $\hat{p}_{ij}$ of the factor p_{ij} corresponds to the frequency counts on the training examples: $\hat{p}_{ij} = \eta_{i,j}(\mathcal{D}, \mathcal{Q}) = |\{\mathbf{x}_k, c_k \in \mathcal{D}|c_k = j \wedge q_i \in \mathcal{Q} \text{ subsumes } \mathbf{x}_k\}|/\eta_j(\mathcal{D})$, where $\eta_j(\mathcal{D}) = |\{\mathbf{x}_k, c_k \in \mathcal{D}|c_k = j|$. The estimator $\hat{p}(c_j)$ of $p(c_j)$ is $\eta_j(\mathcal{D})/|\mathcal{D}|$. By assuming conditional independence

$p(\mathbf{x}|c_j) = \prod_{i=1}^{d}(p_{ij})^{x_i}(1-p_{ij})^{1-x_i}$, yielding the discriminant function

$$g_j(\mathbf{x}) = \ln p(\mathbf{x}|c_j) + \ln p(c_j) = \sum_{i=1}^{d} x_i \ln \frac{p_{ij}}{1-p_{ij}} + \sum_{i=1}^{d} \ln(1-p_{ij}) + \ln p(c_j).$$

The minimum probability error is achieved by deciding c_k if $g_k(\mathbf{x}) \geq g_j(\mathbf{x})$ for all j and k.

3 Mutual Information Feature Selection

A formalization of the uncertainty of a random variable is the Shannon's entropy. Let x be a discrete random variable and $p(x)$ its probabilistic density function, then the entropy of x, a measure of uncertainty, is defined as usual by $H(x) = \mathbb{E}(I(x)) = \sum_i p(x_i)I(x) = -\sum_i p(x_i)\log p(x_i)$, assuming $p(x_i)\log p(x_i) = 0$ in case of $p(x_i) = 0$. For two random variables x and y, their joint entropy is defined as $H(x,y) = -\sum_{i,j} p(x_i,y_j)\log p(x_i,y_j)$, and the conditional entropy is defined as $H(x|y) = -\sum_{i,j} p(x_i,y_j)\log p(x_i|y_j)$. From the last definition, the chain rule for conditional entropy is $H(x,y) = H(x) + H(y|x)$.

The *mutual information* $I(x;y)$ measures how much (on average) the realization of the random variable y tells about the realization of x:

$$I(x;y) = H(x) - H(x|y) = \sum_{i,j} p(x_i,y_j)\log \frac{p(x_i,y_j)}{p(x_i)p(y_j)}. \tag{1}$$

Given a set X of M features, mutual information feature selection corresponds to find a set $S = \{f_i\}_{i=1}^{m} \subset X$ whose elements jointly have the largest dependency on the class c, corresponding to optimize the *maximum dependency* condition:

$$\max_{S \subset X} I(S;c). \tag{2}$$

Directly computing (2) has some difficulties [9], and different approaches to approximate it have been proposed. The approach we used in this paper is the *minimal redundancy and maximal relevance* criterion (mRMR) [9]. An approximation of (2) can be obtained by optimizing the *maximal relevance* criterion:

$$\max_{S \subset X} \frac{1}{|S|} \sum_{f_i \in S} I(f_i;c). \tag{3}$$

In maximizing the relevance, the selected features f_i are required to have the largest mutual information $I(f_i;c)$ with the class c (i.e., the largest dependency on the class). Combinations of individually good features do not necessarily lead to good classification performance. Selecting features according to (3) could lead to a set containing high redundant features. Hence, in order to have mutually exclusive features the criterion of *minimal redundancy* should be optimized:

$$\min_{S \subset X} \frac{1}{|S|^2} \sum_{f_i, f_j \in S} I(f_i;f_j). \tag{4}$$

The mRMR combines these two last criteria by simultaneously optimizing (3) and (4). A possible technique may be to *incrementally search* near-optimal features as follows. Assume S is the subset of m features already selected. The incremental algorithm selects the next feature $f_j \in X \setminus S$ to be added to S by optimizing the following condition:

$$\max_{f_j \in \mathcal{X} \setminus S} \left(I(f_j; c) - \frac{1}{|S|} \sum_{f_i \in S} I(f_j; f_i) \right). \tag{5}$$

Starting from an empty set and using (5) to incrementally select the features to add, a first problem is how to determine the optimal number of features m. Furthermore, even fixing in advance the number m of features to be selected, another problem is that this incremental approach does not assure to find the global optimal solution, and repeated executions could lead the search to be trapped in the same local optimum solution.

One way to decrease the probability of being stuck in a local maximum and avoiding to test all the $2^M - 1$ possible subset solutions, is to consider the use of a *stochastic local search* (SLS) procedure. In this paper we propose a new SLS procedure, similar to the Randomized Iterative Improvement method, able to solve the above problems, as reported in Algorithm 1. Given X the set of available features, the algorithm starts by randomly selecting a feature $f_s \in X$, and setting $S = \{f_s\}$. Then, it iteratively adds a new feature $f_i \in X \setminus S$ to S according to (5) until the new information for the class variable c contributed by the feature f_j given S is greater than a threshold α. This helps the search to be immunized against noisy data, to overcome over-fitting problems, and to solve the problem of how to choose the number m of features to be selected. Furthermore, to implement diversification in the algorithm, the iterative construction phase can choose to make a random walk (by adding a random feature) with a walking probability wp.

After each construction phase, the found solution is evaluated according to the classifier's probability error, and the process is restarted hoping to find a better solution. Given S the selected features, for each example e_j we let the classifier find the MAP hypothesis $\widehat{h}_P(\mathbf{x}_j) = \arg\max_i g_i(\mathbf{x}_j)$ according to the Bayesian discriminant function reported in Section 2 where $\mathbf{x}_j$ is the feature based representation of the example e_j obtained using the queries in S. Hence the optimization problem corresponds to minimize the expectation $\mathbb{E}[\mathbf{1}_{\widehat{h}_P(\mathbf{x}_i) \neq c_i}]$ where $\mathbf{1}_{\widehat{h}_P(\mathbf{x}_i) \neq c_i}$ is the characteristic function of the training example e_i returning 1 if $\widehat{h}_P(\mathbf{x}_i) \neq c_i$, and 0 otherwise. Finally, the number of classification errors made by the Bayesian classifier using the queries S is $err_{\mathcal{D}}(S) = |\mathcal{D}|\mathbb{E}[\mathbf{1}_{\widehat{h}_P(\mathbf{x}_i) \neq c_i}]$.

4 Experiments

We tested the proposed `mLynx` approach on the well known Mutagenesis ILP dataset, and on the widely used UW-CSE SRL dataset [10]. The Mutagenesis

Algorithm 1: Randomized sequential forward feature selection

input : X: input features; *wp*: walking probability; *restarts*: number of restarts; α: threshold
output: $\hat{S}$: the optimal subset

$S = \emptyset$; $i = 0$; $bestValue = \infty$;
while $i < restarts$ **do**
  randomly select a feature f_s from X;
  $S = \{f_s\}$; $improve$ = true;
  while $improve$ **and** $|S| \neq |\mathcal{X}|$ **do**
    if $wp <$ `rand(0,1)` **then**
      $\max_{f_j \in \mathcal{X} \setminus S} \text{mRMR}(f_j, c, S) = \max_{f_j \in \mathcal{X} \setminus S} \left(I(f_j; c) - \frac{1}{|S|} \sum_{f_i \in S} I(f_j; f_i) \right)$;
      if $\text{mRMR}(f_j, c, S) > \alpha$ **then**
        $S = S \cup \{f_j\}$;
      else
        $improve$ = false;
    else
      randomly select $f_j \in \mathcal{X} \setminus S$;
      $S = S \cup \{f_j\}$;
  if $err_D(S) < bestValue$ **then**
    $\hat{S} = S$; $bestValue = err_D(S)$;
  $i = i + 1$;
return $\hat{S}$

dataset regards the problem to predict the mutagenicity of a set of compounds. As in [4] we used atom and bond information only. `mLynx` has been compared to `kFOIL` [4], whose results with a 10-fold cross validation are listed in Table 1. For `mLynx` we set $\alpha = 10^{-2}$, $restarts = 100$, and $wp = 0.05$. The accuracy obtained with `mLynx` is higher than that obtained with `kFOIL` with a difference that is statistically significant with p-value of 0.0455 for the Mutagenesis r.f. dataset.

Dataset	`mLynx`	`kFOIL`
Mutagenesis r.f.	83.94 ± 6.2	77.64 ± 6.5
Mutagenesis r.u.	80.90 ± 15.7	77.50 ± 18.44

Table 1. Average accuracy on the Mutagenesis dataset for `mLynx` and `kFOIL`.

The UW-CSE dataset [10] regards the Department of Computer Science and Engineering at the University of Washington, describing relationships among professors, students, courses and publications with 3212 true ground atoms over 12 predicates. The task is to predict the relationship `advisedBy(X,Y)` using in turn four of the five research areas (ai, graphics, language, theory and systems) for training and the remaining one for testing as in [10]. For `mLynx` we set $\alpha =$

10^{-4}, $restarts = 100$, and $wp = 0.05$. For Alchemy [6] we used the hand-coded MLN reported in [10] including formulas stating regularities, and the applying Alchemy to discriminative learn the weights and testing the resulting MLN on the testing set using the MC-SAT. Table 2 shows the AUC for ROC and Precision-Recall (PR) for `mLynx` and `Alchemy`. The results show that `mLynx` generally improves on `Alchemy` with a difference that is statistically significant with p-value of 0.095 for ROC and with p-value 0.052 for PR.

	`mLynx`		`Alchemy`	
	AUC ROC	AUC PR	AUC ROC	AUC PR
ai	0.929	0.295	0.903	0.286
graphics	0.960	0.697	0.967	0.313
language	0.980	0.797	0.823	0.188
systems	0.933	0.252	0.914	0.224
theory	0.922	0.427	0.867	0.184
mean	0.945	0.494	0.895	0.239

Table 2. AUC for ROC and PR on the UW-CSE dataset for `mLynx` and `Alchemy`.

References

1. Tishby, N., Pereira, F.C., Bialek, W.: The information bottleneck method. In: Proceedings of the 37th annual Allerton Conference on Communication, Control, and Computing. (1999) 368–377
2. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press (2007)
3. Riguzzi, F., Di Mauro, N.: Applying the information bottleneck to statistical relational learning. Machine Learning (2011)
4. Landwehr, N., Passerini, A., De Raedt, L., Frasconi, P.: kFOIL: Learning simple relational kernels. In: Proceedings of AAAI06, AAAI Press (2006)
5. Taranto, C., Di Mauro, N., Esposito, F.: rsLDA: A bayesian hierarchical model for relational learning. In Zhang, J., Livraga, G., eds.: International Conference on Data and Knowledge Engineering, IEEE (2011) 68–74
6. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning **62** (2006) 107–136
7. Di Mauro, N., Basile, T.M., Ferilli, S., Esposito, F.: Optimizing probabilistic models for relational sequence learning. In: 19th International Symposium on Methodologies for Intelligent Systems. Volume 6804 of LNAI., Springer (2011) 240–249
8. Kramer, S., Raedt, L.D.: Feature construction with version spaces for biochemical applications. In: Proceedings of the 18th International Conference on Machine Learning. Morgan Kaufmann Publishers Inc. (2001) 258–265
9. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence **27** (2005) 1226–1238
10. Singla, P., Domingos, P.: Discriminative training of markov logic networks. In: Proceedings of AAAI05, AAAI Press (2005) 868–873