Max-Margin Weight Learning for Markov Logic Networks

Tuyen N. Huynh and Raymond J. Mooney

The University of Texas at Austin, Austin TX 78712, USA
{hntuyen,mooney}@cs.utexas.edu

Abstract. Markov logic networks (MLNs) are an expressive representation for statistical relational learning that generalizes both first-order logic and graphical models. Existing discriminative weight learning methods for MLNs all try to learn weights that optimize the Conditional Log Likelihood (CLL) of the training examples. In this work, we present a new discriminative weight learning method for MLNs based on a max-margin framework. This results in a new model, Max-Margin Markov Logic Networks (M3LNs), that combines the expressiveness of MLNs with the predictive accuracy of structural Support Vector Machines (SVMs). To train the proposed model, we design a new approximation algorithm for loss-augmented inference in MLNs based on Linear Programming (LP). The experimental result shows that the proposed approach generally achieves higher F_1 scores than the current best discriminative weight learner for MLNs.

1 Introduction

Statistical relational learning (SRL) concerns the induction of probabilistic knowledge that supports accurate prediction for multi-relational structured data [1]. Markov Logic Networks (MLNs) are a recently developed SRL model that generalizes both full first-order logic and Markov networks [2]. An MLN consists of a set of weighted clauses in first-order logic. Rather than completely ruling out situations that violate these logical constraints, possible worlds simply become exponentially less likely as the total weight of violated clauses increases. MLNs have been successfully applied to a variety of real-world problems ranging from collective classification of web pages [3] to extraction of bibliographic information from scientific papers [4].

Existing discriminative training algorithms for learning MLN weights attempt to maximize the conditional log likelihood (CLL) of a set of *target predicates* given evidence provided by a set of *background predicates* [5,3,6]. If the goal is to predict accurate target-predicate probabilities, this approach is well motivated. However, in many applications, the actual goal is to maximize an alternative performance metric such as classification accuracy or F-measure. Max-margin methods are a competing approach to discriminative training that are well-founded in computational learning theory and have demonstrated empirical success in

W. Buntine et al. (Eds.): ECML PKDD 2009, Part I, LNAI 5781, pp. 564–579, 2009.

many applications [7]. They also have the advantage that they can be adapted to maximize a variety of performance metrics in addition to classification accuracy [8]. Max-margin methods have been successfully applied to structured prediction problems, such as in Max-Margin Markov Networks (M3Ns) [9] and structural SVMs [10]; however, until now, they have not been applied to an SRL model that generalizes first-order logic such as MLNs.

In this paper, we develop Max-Margin MLNs (M3LNs) by instantiating an existing general framework for max-margin training of structured models [11]. This requires developing a new algorithm for approximating the "loss-augmented" inference in MLNs. Extensive experiments in the two real-world MLN applications referenced above demonstrate that M3LNs generally produce improved results when the goal involves maximizing predictive accuracy metrics other than CLL.

The remainder of the paper is organized as follows. Section 2 provides some background on MLNs and structural SVMs. Section 3 presents the max-margin approach for weight learning in MLNs. Section 4 shows the experimental evaluation of the proposed approach. Section 5 and 6 discuss the related work and future work. Section 7 concludes the paper.

2 Background

2.1 MLNs and Alchemy

An MLN consists of a set of weighted first-order clauses. It provides a way of softening first-order logic by making situations in which not all clauses are satisfied less likely but not impossible [2]. More formally, let X be the set of all propositions describing a world (i.e. the set of all ground atoms), $\mathcal{F}$ be the set of all clauses in the MLN, w_i be the weight associated with clause $f_i \in \mathcal{F}$, $\mathcal{G}_{f_i}$ be the set of all possible groundings of clause f_i, and $\mathcal{Z}$ be the normalization constant. Then the probability of a particular truth assignment $\mathbf{x}$ to the variables in X is defined as [2]:

$$P(X = \mathbf{x}) = \frac{1}{\mathcal{Z}} \exp\left(\sum_{f_i \in \mathcal{F}} w_i \sum_{g \in \mathcal{G}_{f_i}} g(\mathbf{x})\right)$$
$$= \frac{1}{\mathcal{Z}} \exp\left(\sum_{f_i \in \mathcal{F}} w_i n_i(\mathbf{x})\right) \quad (1)$$

where $g(\mathbf{x})$ is 1 if g is satisfied and 0 otherwise, and $n_i(\mathbf{x}) = \sum_{g \in \mathcal{G}_{f_i}} g(\mathbf{x})$ is the number of groundings of f_i that are satisfied given the current truth assignment to the variables in X.

There are two inference tasks in MLNs. The first one is to infer the Most Probable Explanation (MPE) or the most probable truth values for a set of unknown literals $\mathbf{y}$ given a set of known literals $\mathbf{x}$, provided as evidence (also called MAP inference). This task is formally defined as follows:

$$\arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \arg\max_{\mathbf{y}} \frac{1}{\mathcal{Z}_x} \exp\left(\sum_i w_i n_i(\mathbf{x},\mathbf{y})\right)$$
$$= \arg\max_{\mathbf{y}} \sum_i w_i n_i(\mathbf{x},\mathbf{y}) \quad (2)$$

where $\mathcal{Z}_x$ is the normalization constant over all possible worlds consistent with $\mathbf{x}$, and $n_i(\mathbf{x},\mathbf{y})$ is the number of true groundings of clause f_i given the truth assignment $(\mathbf{x},\mathbf{y})$. MPE inference in MLNs is therefore equivalent to finding the truth assignment that maximizes the sum of the weights of satisfied clauses, a Weighted MAX-SAT problem. This is an NP-hard problem for which a number of approximate solvers exist, of which the most commonly used is MaxWalkSAT [12]. Recently, Riedel [13] proposed a more efficient and accurate MPE inference algorithm for MLNs called Cutting Plane Inference (CPI), which does not require grounding the whole MLN. However, the CPI method only works well for some classes of MLNs where the separation step of the CPI method returns a small set of constraints. In the worst case, it also constructs the whole ground MLN.

The second inference task in MLNs is computing the conditional probabilities of some unknown literals, $\mathbf{y}$, given some evidence $\mathbf{x}$. Computing these probabilities is also intractable, but there are good approximation algorithms such as MC-SAT [14] and lifted belief propagation [15].

Learning an MLN consists of two tasks: structure learning and weight learning. The weight learner can learn weights for clauses written by a human expert or automatically induced by a structure learner. There are two approaches to weight learning in MLNs: generative and discriminative. In discriminative learning, we know a priori which predicates will be used to supply evidence and which ones will be queried, and the goal is to correctly predict the latter given the former. Several discriminative weight learning methods have been proposed, all of which try to find weights that maximize the CLL (equivalently, minimize the negative CLL). In MLNs, the derivative of the negative CLL with respect to a weight w_i is the difference of the expected number of true groundings $E_w[n_i]$ of the corresponding clause f_i and the actual number according to the data n_i. However, computing the expected count $E_w[n_i]$ is intractable. The first discriminative weight learner [5] uses the voted perceptron algorithm [16] where it approximates the intractable expected counts by the counts in the MPE state computed by the MaxWalkSAT. Later, Lowd and Domingos [3] presented a number of first-order and second-order methods for optimizing the CLL. These methods use samples from MC-SAT to approximate the expected counts used to compute the gradient and Hessian of the CLL. Among them, the best performing is *preconditioner scaled conjugate gradient* (PSCG) [3]. This method uses the inverse diagonal Hessian as the preconditioner. Recently, Huynh and Mooney [6] proposed an efficient and accurate discriminative weight learner for MLNs when all clauses are non-recursive (A non-recursive clause is a clause that contains only one non-evidence literal). For information about previous work on structure learning see [17].

Alchemy [18] is an open source software package for MLNs. It includes implementations for all of the major existing algorithms for structure learning,

generative weight learning, discriminative weight learning, and inference. Our proposed algorithm is implemented using ALCHEMY.

2.2 Structural Support Vector Machines

In this section, we briefly review the structural SVM problem and an algorithmic schema for solving it efficiently. For more detail, see [11]. In structured output prediction, we want to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is the space of inputs and $\mathcal{Y}$ is the space of multivariate and structured outputs $\mathcal{Y}$, from a set of training examples S:

$$S = ((x_1, y_1), ..., (x_n, y_n)) \in (\mathcal{X} \times \mathcal{Y})^n$$

The goal is to find a function h that has low prediction error. This can be accomplished by learning a discriminant function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$, then maximizing f over all $y \in \mathcal{Y}$ for a given input x to get the prediction.

$$h_w(x) = \arg\max_{y \in \mathcal{Y}} f_w(x, y)$$

The discriminant function $f_w(x, y)$ takes the form of a linear function:

$$f_w(x, y) = \mathbf{w}^T \Psi(x, y)$$

where $\mathbf{w} \in \mathcal{R}^N$ is a parameter vector and $\Psi(x, y)$ is a feature vector relating an input x and output y. The features need to be designed for a given problem so that they capture the dependency structure of y and x and the relations among the outputs y . Then, the goal is to find a weight vector $\mathbf{w}$ that maximizes the margin:

$$\gamma(x_i, y_i; w) = \mathbf{w}^T \Psi(x_i, y_i) - \max_{y_i' \in \mathcal{Y} \setminus y_i} \mathbf{w}^T \Psi(x_i, y_i')$$

To find such a weight vector, Joachims *et al.* [11] proposed to solve the following optimization problem, called the "1-slack" structural SVM problem:

Optimization Problem 1 (OP1): 1-Slack Structural SVMs [11]

$$\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C\xi$$

$$s.t.\ \forall (\bar{y_1}, ..., \bar{y_n}) \in \mathcal{Y}^n : \frac{1}{n} \mathbf{w}^T \sum_{i=1}^{n} \delta\Psi(\bar{y_i}) \geq \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i, \bar{y_i}) - \xi$$

where $\delta\Psi(\bar{y_i}) = \Psi(x_i, y_i) - \Psi(x_i, \bar{y_i})$, and $\Delta(y_i, \bar{y_i})$ is the loss function. This optimization problem has an exponential number of constraints $|\mathcal{Y}|^n$, one for each possible combination of labels $(\bar{y_1}, ..., \bar{y_n}) \in \mathcal{Y}^n$, but it can be solved efficiently (provably linear runtime in the number of training examples) by Algorithm 1.

In each iteration, this algorithm solves a Quadratic Programming (QP) problem (line 4) to find the optimal weights corresponding to the current set of

Algorithm 1. Cutting-plane method for solving the "1-slack structural SVMs" [11]

1: **Input:** $S = ((x_1, y_1), ..., (x_n, y_n)), C, \varepsilon$
2: $\mathcal{W} \leftarrow \emptyset$
3: **repeat**
4:

$$(\mathbf{w}, \xi) \leftarrow \min_{\mathbf{w}, \xi \geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi$$
$$s.t.\ \forall (\bar{y_1}, ..., \bar{y_n}) \in \mathcal{W} : \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} [\Psi(x_i, y_i) - \Psi(x_i, \bar{y_i})] \geq \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i, \bar{y_i}) - \xi$$

5: **for** $i = 1$ **to** n **do**
6: $\hat{y_i} \leftarrow \arg\max_{\hat{y} \in \mathcal{Y}} \{\Delta(y_i, \hat{y}) + \mathbf{w}^T\Psi(x_i, \hat{y})\}$
7: **end for**
8: $\mathcal{W} \leftarrow \mathcal{W} \cup \{(\hat{y_1}, ..., \hat{y_n})\}$
9: **until** $\frac{1}{n}\sum_{i=1}^{n} \Delta(y_i, \hat{y_i}) - \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} [\Psi(x_i, y_i) - \Psi(x_i, \hat{y_i})] \leq \xi + \varepsilon$
10: return $(\mathbf{w}, \xi)$

constraints $\mathcal{W}$ and a loss-augmented inference problem (line 6), also called a separation oracle, to find the most violated constraint to add to $\mathcal{W}$. Since the QP problem does not depend on the structure of a particular problem (the $\Psi(x, y)$ and $\Delta(y, \bar{y})$), it can be solved by any QP solver. In contrast, for each specific problem, one needs to come up with an efficient way to solve the loss-augmented inference problem.

In summary, to apply structural SVMs to a new problem, one needs to design a new feature vector function $\Psi(x, y)$, choose a loss function $\Delta(y, \bar{y})$, and solve two argmax problems:

Prediction: $\arg\max_{y \in \mathcal{Y}} \mathbf{w}^T\Psi(x, y)$
Separation Oracle: $\arg\max_{\bar{y} \in \mathcal{Y}} \{\Delta(y, \bar{y}) + \mathbf{w}^T\Psi(x, \bar{y})\}$

3 Max-Margin Weight Learning for MLNs

3.1 Max-Margin Formulation

All of the current discriminative weight learners for MLNs try to find a weight vector $\mathbf{w}$ that optimizes the conditional log-likelihood $P(\mathbf{y}|\mathbf{x})$ of the query atoms $\mathbf{y}$ given the evidence $\mathbf{x}$. However, an alternative approach is to learn a weight vector $\mathbf{w}$ that maximizes the ratio:

$$\frac{P(\mathbf{y}|\mathbf{x}, \mathbf{w})}{P(\hat{\mathbf{y}}|\mathbf{x}, \mathbf{w})}$$

between the probability of the correct truth assignment $\mathbf{y}$ and the closest competing incorrect truth assignment $\hat{\mathbf{y}} = \arg\max_{\bar{\mathbf{y}} \in \mathbf{Y} \setminus \mathbf{y}} P(\bar{\mathbf{y}}|\mathbf{x})$. Applying equation 1 and taking the log, this problem translates to maximizing the margin:

$$\begin{aligned}\gamma(\mathbf{x}, \mathbf{y}; \mathbf{w}) &= \mathbf{w}^T \mathbf{n}(\mathbf{x}, \mathbf{y}) - \mathbf{w}^T \mathbf{n}(\mathbf{x}, \hat{\mathbf{y}}) \\ &= \mathbf{w}^T \mathbf{n}(\mathbf{x}, \mathbf{y}) - \max_{\bar{\mathbf{y}} \in \mathbf{Y} \setminus \mathbf{y}} \mathbf{w}^T \mathbf{n}(\mathbf{x}, \bar{\mathbf{y}})\end{aligned}$$

Note that, this translation holds for all log-linear models. For example, if we apply it to a CRF [19] then the result model is an M3N [9]. In fact, this translation is the connection between log-linear models and linear classifiers [20].

In turn, the max-margin problem above can be formulated as a "1-slack" structural SVM as follows:

Optimization Problem 2 (OP2): Max-Margin Markov Logic Networks

$$\begin{aligned}&\min_{\mathbf{w}, \xi \geq 0} \ \frac{1}{2} \mathbf{w}^T \mathbf{w} + C\xi \\ &\quad s.t.\ \forall \bar{\mathbf{y}} \in Y : \mathbf{w}^T [\mathbf{n}(\mathbf{x}, \mathbf{y}) - \mathbf{n}(\mathbf{x}, \bar{\mathbf{y}})] \geq \Delta(\mathbf{y}, \bar{\mathbf{y}}) - \xi\end{aligned}$$

So for MLNs, the number of true groundings of the clauses $\mathbf{n}(\mathbf{x}, \mathbf{y})$ plays the role of the feature vector function $\Psi(x, y)$ in the general structural SVM problem. In other words, each clause in an MLN can be viewed as a feature representing a dependency between a subset of inputs and outputs or a relation among several outputs.

As mentioned, in order to apply Algorithm 1 to MLNs, we need algorithms for solving the following two problems:

Prediction: $\arg\max_{\mathbf{y} \in Y} \mathbf{w}^T \mathbf{n}(\mathbf{x}, \mathbf{y})$

Separation Oracle: $\arg\max_{\bar{\mathbf{y}} \in Y} \{\Delta(\mathbf{y}, \bar{\mathbf{y}}) + \mathbf{w}^T \mathbf{n}(\mathbf{x}, \bar{\mathbf{y}})\}$

The prediction problem is just the (intractable) MPE inference problem discussed in section 2.1. We can use MaxWalkSAT to get an approximate solution, but we have found that models trained with MaxWalkSAT have very low predictive accuracy. On the other hand, recent work [21] has found that fully-connected pairwise Markov random fields, a special class of structural SVMs, trained with overgenerating approximate inference methods (such as relaxation) preserves the theoretical guarantees of structural SVMs trained with exact inference, and exhibits good empirical performance. Based on this result, we sought a relaxation-based approximation for MPE inference. We first present an LP-relaxation algorithm for MPE inference, then show how to modify it to solve the separation oracle problem for some specific loss functions.

3.2 Approximate MPE Inference for MLNs

MPE inference in MLNs is equivalent to the Weighted MAX-SAT problem, and there has been significant work on approximating this NP-hard problem using LP-relaxation [22,23]. The existing algorithms first relax and convert the Weighted MAX-SAT problem into a linear or semidefinite programming problem, then solve it and apply a randomized rounding method to obtain an approximate integral solution. These methods cannot be directly applied to MLNs, since they require the weights to be positive while MLN weights can be negative

or infinite. So we modified the conversion used in these approaches to handle the case of negative and infinite weights.

Based on the evidence and the closed world assumption, a ground MLN contains only ground clauses (in clausal form) of the unknown ground atoms after removing all trivially satisfied and unsatisfied clauses. The following procedure translates the MPE inference in a ground MLN into an Integer Linear Programming (ILP) problem.

1. Assign a binary variable y_i to each unknown ground atom. y_i is 1 if the corresponding ground atom is $TRUE$ and 0 if the ground atom is $FALSE$.
2. For each ground clause C_j with infinite weight, add the following linear constraint to the ILP problem:
$$\sum_{i \in I_j^+} y_i + \sum_{i \in I_j^-} (1 - y_i) \geq 1$$
where I_j^+, I_j^- are the sets of positive and negative ground literals in clause C_j respectively.
3. For each ground clause C_j with positive weight w_j, introduce a new auxiliary binary variable z_j, add the term $w_j z_j$ to the objective function, and add the following linear constraint to the ILP problem:
$$\sum_{i \in I_j^+} y_i + \sum_{i \in I_j^-} (1 - y_i) \geq z_j$$
z_j is 1 if the corresponding ground clause is satisfied.
4. For each ground clause C_j with k ground literals and negative weight w_j, introduce a new auxiliary boolean variable z_j, add the term $-w_j z_j$ to the objective function and add the following k linear constrains to the ILP problem:
$$\begin{aligned} 1 - y_i &\geq z_j, \quad i \in I_j^+ \\ y_i &\geq z_j, \quad i \in I_j^- \end{aligned}$$

The final ILP has the following form:

Optimization Problem 3:

$$\begin{aligned} \max_{y_i, z_i} & \sum_{C_j \in C^+} w_j z_j + \sum_{C_j \in C^-} -w_j z_j \\ s.t. & \sum_{i \in I_j^+} y_i + \sum_{i \in I_j^-} (1 - y_i) \geq 1 \quad \forall\ C_j\ where\ w_j = \infty \\ & \sum_{i \in I_j^+} y_i + \sum_{i \in I_j^-} (1 - y_i) \geq z_j \quad \forall C_j \in C^+ \\ & 1 - y_i \geq z_j \quad \forall\ i \in I_j^+\ and\ C_j \in C^- \\ & y_i \geq z_j \quad \forall\ i \in I_j^-\ and\ C_j \in C^- \\ & y_i, z_j \in \{0, 1\} \end{aligned}$$

Algorithm 2. The modified ROUNDUP procedure

Input: The LP solution $\mathbf{y} = \{y_1, ..., y_n\}$
$F \leftarrow \emptyset$
for $i = 1$ **to** n **do**
  if y_i is integral **then**
    Remove all the ground clauses satisfied by assigning the value of y_i to the corresponding ground atom
  else
    add y_i to F
  end if
end for
repeat
  Remove the last item y_i in F
  Compute the sum w^+ of the unsatisfied clauses where y_i appears as a positive literal
  Compute the sum w^- of the unsatisfied clauses where y_i appears as a negative literal
  if $w^+ > w^-$ **then**
    $y_i \leftarrow 1$
  else
    $y_i \leftarrow 0$
  end if
  Remove all the ground clauses satisfied by assigning the value of y_i to the corresponding ground atom
until F is empty
return $\mathbf{y}$

where C^+ and C^- are the set of clauses with positive and negative weights respectively. This ILP problem can be simplified by not introducing an auxiliary variable z_j for unit clauses, where we can use the variable y_i directly. This reduces the problem considerably, since ground MLNs typically contain many unit clauses (Alchemy combines all the non-recursive clauses containing the query atom into a unit clause whose weight is the sum of all the clauses' weights). Note that our mapping from a ground MLN to an ILP problem is a bit different from the one presented in [13] which generates two sets of constraints for every ground clause: one when the clause is satisfied and one when it is not. For a clause with positive weight, our mapping only generates a constraint when the clause is satisfied; and for a clause with negative weight, the mapping only imposes constraints when the clause is unsatisfied. The final ILP problem has the same solution with the one in [13], but it has fewer constraints since our mapping does not generate unnecessary constraints. We then relax the integer constraints $y_i, z_j \in \{0, 1\}$ to linear constraints $y_i, z_j \in [0, 1]$ to obtain an LP-relaxation of the MPE problem.

This LP problem can be solved by any general LP solver. If the LP solver returns an integral solution, then it is also the optimal solution to the original ILP problem. In our case, the original ILP problem is an NP-hard problem, so

the LP solver usually returns non-integral solutions. Therefore, the LP solution needs to be rounded to give an approximate ILP solution. We first tried some of the randomized rounding methods in [23] but they gave poor results since the LP solution has a lot of fractional components with value 0.5. We then adapted a rounding procedure called ROUNDUP [24], a procedure for producing an upper-bound binary solution for a pseudo-Boolean function, to the case of pseudo-Boolean functions with linear constraints, which we found to work well. In each step, this procedure picks one fractional component and rounds it to 1 or 0. Hence, this process terminates in at most n steps, where n is the number of query atoms. Note that due to the dependencies between y_i's and z_j's (the linear constraints of the LP problem), this modified ROUNDUP procedure does not guarantee an improvement in the value of the objective function in each step like the original ROUNDUP procedure where all the variables are independent.

3.3 Approximation Algorithm for the Separation Oracle

The separation oracle adds an additional term, the loss term, to the objective function. So, if we can represent the loss as a linear function of the y_i variables of the LP-relaxation, then we can use the above approximation algorithm to also approximate the separation oracle. In this work, we consider two loss functions. The first one is the 0/1 loss function, $\Delta_{0/1}(\mathbf{y^T}, \mathbf{y})$ where $\mathbf{y^T}$ is the true assignment and $\mathbf{y}$ is some predicted assignment. For this loss function, the separation oracle is the same as the MPE inference problem since the loss function only adds a constant 1 to the objective function. Hence, in this case, to find the most violated constraint, we can use the LP-relaxation algorithm above or any other MPE inference algorithm. This 0/1 loss makes the separation oracle problem easier but it does not scale the margin by how different $\mathbf{y^T}$ and $\mathbf{y}$ are. It only requires a unit margin for all assignments $\mathbf{y}$ different from the true assignment $\mathbf{y^T}$. To take into account this problem, we consider the second loss function that is the number of misclassified atoms or the Hamming loss:

$$\begin{aligned}\Delta_{Hamming}(\mathbf{y^T}, \mathbf{y}) &= \sum_{i}^{n} [y_i^T \neq y_i] \\ &= \sum_{i}^{n} [(y_i^T = 0 \wedge y_i = 1) \vee (y_i^T = 1 \wedge y_i = 0)]\end{aligned}$$

From the definition, this loss can be represented as a function of the y_i's:

$$\Delta_{Hamming}(\mathbf{y^T}, \mathbf{y}) = \sum_{i:y_i^T=0} y_i + \sum_{i:y_i^T=1} (1 - y_i)$$

which is equivalent to adding 1 to the coefficient of y_i if the true value of y_i is 0 and subtracting 1 from the coefficient of y_i if the true value of y_i is 1. So we can use the LP-relaxation algorithm above to approximate the separation oracle with this Hamming loss function. Another possible loss function is $(1 - F_1)$ loss.

Unfortunately, this loss is a non-linear function, so we cannot use the above approach to optimize it. Developing algorithms for optimizing or approximating this loss function is an area for future work.

4 Experimental Evaluation

This section presents experiments comparing M3LNs to the current best discriminative weight learner for MLNs with recursive clauses, PSCG.

4.1 Datasets

We ran experiments on two large, real-world MLN datasets: WebKB for collective web-page classification, and CiteSeer for bibliographic citation segmentation. All the datasets and MLNs can be found at the ALCHEMY website[1].

The WebKB dataset consists of labeled web pages from the computer science departments of four universities. Different versions of this data have been used in previous work. To make a fair comparison, we used the version from [3], which contains 4,165 web pages and 10,935 web links. Each page is labeled with a subset of the categories: *course*, *department*, *faculty*, *person*, *professor*, *research project*, and *student*. The goal is to predict these categories from the words and links on the web pages. We used the same simple MLN from [3], which only has clauses relating words to page classes, and page classes to the classes of linked pages.

$$Has(+word, page) \rightarrow PageClass(+class, page)$$
$$\neg Has(+word, page) \rightarrow PageClass(+class, page)$$
$$PageClass(+c1, p1) \wedge Linked(p1, p2) \rightarrow PageClass(+c2, p2)$$

The plus notation creates a separate clause for each pair of word and page class, and for each pair of classes. The final MLN consists of 10,891 clauses, and a weight must be learned for each one. After grounding, each department results in an MLN with more than 100,000 ground clauses and 5,000 query atoms in a complex network. This also results in a large LP-relaxation problem for MPE inference.

For CiteSeer, we used the dataset and MLN used in [4]. The dataset has 1,563 citations and each of them is segmented into three fields: *Author*, *Title* and *Venue*. The dataset has four disconnected segments corresponding to four different research topics. We used the simplest MLN in [4], which is the isolated segmentation model. Despite its simplicity, after grounding, this model results in a large network with more than 30,000 query atoms and 110,000 ground clauses.

4.2 Metrics

We used F_1, the harmonic mean of recall and precision, to measure the performance of each algorithm. This is the standard evaluation metric in multi-class text categorization and information extraction. For systems that compute marginal probabilities rather than MPEs, we predict that an atom is true iff its probability is at least 0.5.

[1] http://alchemy.cs.washington.edu

Table 1. F_1 scores on WebKB

	Cornell	Texas	Washington	Wisconsin	Average
PSCG-MCSAT	0.418	0.298	0.577	0.568	0.465
PSCG-LPRelax	0.420	0.310	0.588	0.575	0.474
MM-$\Delta_{0/1}$-MaxWalkSAT	0.150	0.162	0.122	0.122	0.139
MM-$\Delta_{0/1}$-LPRelax	0.282	0.372	0.675	0.521	0.462
MM-$\Delta_{Hamming}$-LPRelax	**0.580**	**0.451**	**0.715**	**0.659**	**0.601**

Table 2. F_1 scores of different inference algorithms on WebKB

	Cornell	Texas	Washington	Wisconsin	Average
PSCG-MCSAT	0.418	0.298	0.577	0.568	0.465
PSCG-MaxWalkSAT	0.161	0.140	0.119	0.129	0.137
PSCG-LPRelax	0.420	0.310	0.588	0.575	0.474
MM-$\Delta_{Hamming}$-MCSAT	0.470	0.370	0.573	0.481	0.473
MM-$\Delta_{Hamming}$-MaxWalkSAT	0.185	0.184	0.150	0.154	0.168
MM-$\Delta_{Hamming}$-LPRelax	0.580	0.451	0.715	0.659	0.601

4.3 Methodology

We ran four-fold cross-validation (i.e. leave one university/topic out) on both datasets. For the max-margin weight learner, we used a simple process for selecting the value of the C parameter. For each train/test split, we trained the algorithm with five different values of C: 1, 10, 100, 1000, and 10000, then selected the one which gave the highest average F_1 score on training. The ϵ parameter was set to 0.001 as suggested in [11]. To solve the QP problems in Algorithm 1 and LP problems in the LP-relaxation MPE inference, we used the MOSEK[2] solver. The PSCG algorithm was carefully tuned by its author. For MC-SAT and MaxWalkSAT, we used the default setting in ALCHEMY.

4.4 Results and Discussion

Table 1 and 3 present the performance of different systems on the WebKB and Citeseer datasets. Each system is named by the weight learner used, the loss function used in training, and the inference algorithm used in testing. For max-margin (MM) learner with margin rescaling, the inference used in training is the loss-augmented version of the one used in testing. For example, MM-$\Delta_{Hamming}$-LPRelax is the max-margin weight learner using the loss-augmented (Hamming loss) LP-relaxation MPE inference algorithm in training and the LP-relaxation MPE inference algorithm in testing.

Table 1 shows that the model trained by MaxWalkSAT has very low predictive accuracy. This result is consistent with the result presented in [13] which also found that the MPE solution found by MaxWalkSAT is not very accurate. Using the proposed LP-relaxation MPE inference improves the F_1 score from 0.139

[2] http://www.mosek.com/

Table 3. F_1 scores on CiteSeer

	Constraint	Face	Reasoning	Reinforcement	Average
PSCG-MCSAT	0.937	0.914	0.931	0.975	0.939
MM-$\Delta_{Hamming}$-LPRelax	0.933	0.922	0.924	0.958	0.934

Table 4. F_1 scores on CiteSeer with different parameter values

	Constraint	Face	Reasoning	Reinforcement	Average
PSCG-MCSAT-5	0.852	0.844	0.836	0.923	0.864
PSCG-MCSAT-10	0.937	0.914	0.931	0.973	0.939
PSCG-MCSAT-15	0.878	0.896	0.780	0.891	0.861
PSCG-MCSAT-20	0.850	0.859	0.710	0.784	0.801
PSCG-MCSAT-100	0.658	0.697	0.600	0.668	0.656
MM-$\Delta_{Hamming}$-LPRelax-1	0.933	0.922	0.924	0.955	0.934
MM-$\Delta_{Hamming}$-LPRelax-10	0.926	0.922	0.925	0.955	0.932
MM-$\Delta_{Hamming}$-LPRelax-100	0.926	0.922	0.925	0.954	0.932
MM-$\Delta_{Hamming}$-LPRelax-1000	0.931	0.918	0.925	0.958	0.933
MM-$\Delta_{Hamming}$-LPRelax-10000	0.932	0.922	0.919	0.968	0.935

to 0.462, the MM-$\Delta_{0/1}$-LPRelax system. Then the best system is obtained by rescaling the margin and training with our loss-augmented LP-relaxation MPE inference, which is the only difference between MM-$\Delta_{Hamming}$-LPRelax and MM-$\Delta_{0/1}$-LPRelax. The MM-$\Delta_{Hamming}$-LPRelax achieves the best F_1 score (0.601), which is much higher than the 0.465 F_1 score obtained by the current best discriminative weight learner for MLNs, PSCG-MCSAT.

Table 2 compares the performance of the proposed LP-relaxation MPE inference algorithm against MCSAT and MaxWalkSAT on the best trained models by PSCG and MM on the WebKB dataset. In both cases, the LP-relaxation MPE inference achieves much better F_1 scores than those of MCSAT and MaxWalkSAT. This demonstrates that the approximate MPE solution found by the LP-relaxation algorithm is much more accurate than the one found by the MaxWalkSAT algorithm. The fact that the performance of the LP-relaxation is higher than that of MCSAT shows that in collective classification it is better to use the MPE solution as the prediction than the marginal prediction.

For the WebKB dataset, there are other results reported in previous work, such as those in [9], but those results cannot be directly compared to our results since we use a different version of the dataset and test on a more complicated task (a page can have multiple labels not just one).

On the Citeseer results presented in Table 3, the performance of max-margin methods are very close to those of PSCG. However, its performance is much more stable. Table 4 shows the performance of MM weight learners and PSCG with different parameter values by varying the C value for MM and the number of iterations for PSCG. The best number of iterations for PSCG is 9 or 10. In principle, we should run PSCG until it converges to get the optimal weight vector. However, in this case, the performance of PSCG drops drastically on

both training and testing after a certain number of iterations. For example, from Table 4 we can see that at 10 iterations PSCG achieves the best F_1 score of 0.939, but after 15 iterations, its F_1 score drops to 0.861 which is much worse than those of the max-margin weight learners. Moreover, if we let it run until 100 iterations, then its F_1 score is only 0.656. On the other hand, the performance of MM only varies a little bit with different values of C and we don't need to tune the number of iterations of MM. On this dataset, [4] achieved a F_1 score of 0.944 with the same MLN by using a version of the voted perceptron algorithm called Contrastive Divergence (CD) [25] to learn the weights. However, the performance of the CD algorithm is very sensitive to the learning rate [3], which requires a very careful tuning process to learn a good model.

Regarding training time, on both of the datasets, the max-margin weight learner is comparable to PSCG. It usually took about 200 iterations on the WebKB and less than 50 iterations on Citeseer to find the optimal weights, which resulted in a few hours of training for WebKB and less than an hour for Citeseer.

5 Related Work

Our work is related to various previous projects. Among them, M3N [9] is probably the most related. It is a special case of structural SVMs where the feature function $\Psi(x, y)$ is represented by a Markov network. When the Markov network can be triangulated and the loss function can be linearly decomposed, the original exponentially-sized QP can be reformulated as a polynomially-sized QP [9]. Then, the polynomially-sized QP can be solved by general QP solvers [26], decomposition methods [9], extragradient methods [27], or exponentiated gradient methods [28]. As mentioned in [9], these methods can also be used when the graph cannot be triangulated, but the algorithms only yield approximate solutions like our approach. However, these algorithms are restricted to the cases where a polynomially-sized reformulation exists [11]. That's why in this work we used the general cutting plane algorithm which imposes no restrictions on the representation. The ground MLN can be any kind of graph. On the other hand, since an MLN is a template for constructing Markov networks [2], the proposed model, M3LN, can also be seen as a template for constructing M3Ns. Hence, when the ground MLN can be triangulated and the loss is a linearly decomposable function, the algorithms developed for M3Ns can be applied. Our work is also closely related to the Relational Markov Networks (RMNs) [29]. However, by using MLNs, M3LNs are more powerful than RMNs in term of representation [2]. Besides, the objectives of M3LNs and RMNs are different. One tries to maximize the margin between the true assignment and other competing assignments, and one tries to maximize the conditional likelihood of the true assignment. Another related system is Rumble [30], a margin-based approach to first-order rule learning. In that work, the goal is to find a set of weighted rules that maximizes a quantity called margin minus variance. However, unlike M3LNs, Rumble only applies to independent binary classification problems and

is unable to perform structured prediction or collective classification. In terms of applying the general structural SVM framework to a specific representation, our work is related to the work in [31] which used CRFs as the representation and graph cuts as the inference algorithm. In the context of discriminative learning, our work is related to previous work on discriminative training for MLNs [5,3,6,32]. We have mentioned some of them [5,3,6] in previous sections. The main difference between the work in [32] and ours is that we assume the structure is given and apply max-margin framework to learn the weights while [32] tries to learn a structure that maximizes the conditional likelihood of the data. Extending the max-margin framework to structure learning is an area for future work.

6 Future Work

Currently, our loss-augmented LP-relaxation MPE inference algorithm requires grounding the entire network first. It would be useful to develop new loss-augmented MPE inference algorithms which do not require grounding the full network. On the other hand, the max-margin framework allows optimizing different kinds of loss functions. Therefore, it would also be interesting to extend the current algorithm to optimize other loss functions. Besides, we also want to apply M3LNs to other structured prediction tasks and compare to other structured prediction models.

7 Conclusions

We have presented a max-margin weight learning method for MLNs based on the framework of structural SVMs. It resulted in a new model, M3LN, that has the representational expressiveness of MLNs and the predictive performance of SVMs. M3LNs can be trained to optimize different performance measures depending on the needs of the application. To train the proposed model, we developed a new approximation algorithm for loss-augmented MPE inference in MLNs based on LP-relaxation. The experimental results showed that the new max-margin learner generally has better or equally good but more stable predictive accuracy (as measured by F_1) than the current best discriminative MLN weight learner.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. We also thank Daniel Lowd and Hoifung Poon for useful discussions and helping with the experiments. This research is sponsored by the DARPA and managed by the AFRL under contract FA8750-05-2-0283. The project is also partly support by ARO grant W911NF-08-1-0242. Most of the experiments were run on the Mastodon Cluster, provided by NSF Grant EIA-0303609. The first author also thanks the Vietnam Education Foundation (VEF) for its sponsorship.

References

1. Getoor, L., Taskar, B. (eds.): Statistical Relational Learning. MIT Press, Cambridge (2007)
2. Richardson, M., Domingos, P.: Markov logic networks. MLJ 62, 107–136 (2006)
3. Lowd, D., Domingos, P.: Efficient weight learning for Markov logic networks. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 200–211. Springer, Heidelberg (2007)
4. Poon, H., Domingos, P.: Joint inference in information extraction. In: AAAI 2007, pp. 913–918 (2007)
5. Singla, P., Domingos, P.: Discriminative training of Markov logic networks. In: AAAI 2005, pp. 868–873 (2005)
6. Huynh, T.N., Mooney, R.J.: Discriminative structure and parameter learning for Markov logic networks. In: ICML 2008, pp. 416–423 (2008)
7. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, Cambridge (2000)
8. Joachims, T.: A support vector method for multivariate performance measures. In: ICML 2005, pp. 377–384 (2005)
9. Taskar, B., Guestrin, C., Koller, D.: Max-margin Markov networks. In: NIPS 2003 (2003)
10. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. JMLR 6, 1453–1484 (2005)
11. Joachims, T., Finley, T., Yu, C.N.: Cutting-plane training of structural SVMs. MLJ (2009), `http://www.springerlink.com/content/h557723w88185170`
12. Kautz, H., Selman, B., Jiang, Y.: A general stochastic approach to solving problems with hard and soft constraints. In: Dingzhu Gu, J.D., Pardalos, P. (eds.) The Satisfiability Problem: Theory and Applications, AMS, pp. 573–586 (1997)
13. Riedel, S.: Improving the accuracy and efficiency of MAP inference for Markov logic, UAI, 468–475 (2008)
14. Poon, H., Domingos, P.: Sound and efficient inference with probabilistic and deterministic dependencies. In: AAAI 2006, Boston, MA (July 2006)
15. Singla, P., Domingos, P.: Lifted first-order belief propagation. In: AAAI 2008, 1094–1099 (2008)
16. Collins, M.: Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In: CONLL 2002, pp. 1–8 (2002)
17. Domingos, P., Kok, S., Lowd, D., Poon, H., Richardson, M., Singla, P.: Markov logic. In: De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S.H. (eds.) Probabilistic Inductive Logic Programming. LNCS (LNAI), vol. 4911, pp. 92–117. Springer, Heidelberg (2008)
18. Kok, S., Singla, P., Richardson, M., Domingos, P.: The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington (2005), `http://www.cs.washington.edu/ai/alchemy`
19. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML 2001, Williamstown, MA, pp. 282–289 (2001)
20. Collins, M.: Parameter estimation for statistical parsing models: Theory and practice of distribution-free methods. In: Harry Bunt, J.C., Satta, G. (eds.) New Developments in Parsing Technology. Kluwer, Dordrecht (2004)

21. Finley, T., Joachims, T.: Training structural SVMs when exact inference is intractable. In: ICML 2008, pp. 304–311 (2008)
22. Asano, T., Williamson, D.P.: Improved approximation algorithms for MAX SAT. J. of Algorithms 42(1), 173–202 (2002)
23. Asano, T.: An improved analysis of Goemans and Williamson's LP-relaxation for MAX SAT. Theoretical Computer Science 354(3), 339–353 (2006)
24. Boros, E., Hammer, P.L.: Pseudo-Boolean optimization. Discrete Applied Mathematics 123(1-3), 155–225 (2002)
25. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
26. Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., Ng, A.: Discriminative learning of Markov random fields for segmentation of 3D scan data. In: CVPR 2005, pp. 169–176 (2005)
27. Taskar, B., Lacoste-Julien, S., Jordan, M.I.: Structured prediction, dual extragradient and Bregman projections. JMLR 7, 1627–1653 (2006)
28. Collins, M., Globerson, A., Koo, T., Carreras, X., Bartlett, P.L.: Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. JMLR 9, 1775–1822 (2008)
29. Taskar, B., Abbeel, P., Koller, D.: Discriminative probabilistic models for relational data. In: UAI 2002, Edmonton, Canada, pp. 485–492 (2002)
30. Rückert, U., Kramer, S.: Margin-based first-order rule learning. MLJ 70(2-3), 189–206 (2008)
31. Szummer, M., Kohli, P., Hoiem, D.: Learning CRFs using graph cuts. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 582–595. Springer, Heidelberg (2008)
32. Biba, M., Ferilli, S., Esposito, F.: Discriminative structure learning of Markov logic networks. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 59–76. Springer, Heidelberg (2008)