Probing the Space of Optimal Markov Logic Networks for Sequence Labeling

Naveen Nair[1,2,3], Ajay Nagesh[1,2,3], and Ganesh Ramakrishnan[2,1]

[1] IITB-Monash Research Academy, Old CSE Building, IIT Bombay
[2] Department of Computer Science and Engineering, IIT Bombay
[3] Faculty of Information Technology, Monash University
{naveennair,ajaynagesh,ganesh}@cse.iitb.ac.in

Abstract. Discovering relational structure between input features in sequence labeling models has shown to improve their accuracies in several problem settings. The problem of learning relational structure for sequence labeling can be posed as learning Markov Logic Networks (MLN) for sequence labeling, which we abbreviate as Markov Logic Chains (MLC). This objective in propositional space can be solved efficiently and optimally by a Hierarchical Kernels based approach, referred to as StructRELHKL, which we recently proposed. However, the applicability of StructRELHKL in complex first order settings is non-trivial and challenging. We present the challenges and possibilities for optimally and simultaneously learning the structure as well as parameters of MLCs (as against learning them separately and/or greedily). Here, we look into leveraging the StructRELHKL approach for optimizing the MLC learning steps to the extent possible. To this end, we categorize first order MLC features based on their complexity and show that complex features can be constructed from simpler ones. We define a self-contained class of features called absolute features ($\mathcal{AF}$), which can be conjoined to yield complex MLC features. Our approach first generates a set of relevant $\mathcal{AF}$s and then makes use of the algorithm for StructRELHKL to learn their optimal conjunctions. We demonstrate the efficiency of our approach by evaluating on a publicly available activity recognition dataset.

Keywords: Feature Induction, Hierarchical Kernel Learning, Markov Logic Networks, Sequence labeling.

1 Introduction

Learning and prediction problems in real world have to deal with complex relationships among entities combined with uncertainties in these relationships. These complex relationships are quite often represented compactly in the form of first order logical statements. The uncertainties are typically captured in the form of probabilities or probabilistic weights. The systems capable of handling such complex logical relationships and their uncertainties are generally classified as Statistical Relational Learning (SRL) systems [1,2]. One of the most popular

F. Riguzzi and F. Železný (Eds.): ILP 2012, LNAI 7842, pp. 193–208, 2013.

SRL frameworks is the Markov Logic Network (MLN) [3,4,5,6], which combines the expressive representation formalism of first order logic and the ability of probabilistic graphical models to handle uncertainty [3,4]. Our focus, in this paper, is learning MLNs for the specific problem of sequence labeling. We first briefly introduce MLNs in the following paragraph and then proceed to define our sequence labeling problem.

Markov Logic Networks: MLNs [3] extend first order logical systems by incorporating probabilistic information to the clauses/rules. They are typically represented as a collection of first order clauses with real valued weights attached to each clause. Each vertex in the graphical representation is a first order predicate. The edges between these predicates represent the logical connectives in a first order formula. Therefore, a clique in the graph represents a first order clause. A grounded[1] MLN is a Markov Network in the propositional space.

MLNs define a probability distribution over a possible world I as,

$$P(I|H,B) = \frac{1}{Z} \prod_{C \in H \cup B} (\phi_C)^{n_C(I)} \tag{1}$$

where H is the hypothesis, B is the background knowledge, $\phi_C = e^{f_C}$, f_C is the weight attributed to the clause C, $n_C(I)$ is the number of true groundings of C in I and Z is the normalization constant. Therefore, an ideal MLN should have hypothesis,

$$H^* = \arg\max_H \prod_{I \in \hat{I}} P(I|H,B) \tag{2}$$

where $\hat{I}$ is the set of true interpretations.

Conventional MLN systems tend to learn the structure and parameters separately. There have been a few approaches recently to learn the structure and parameters simultaneously [7,8,9]. However since the feature space is exponential, all the MLN structure learning approaches are greedy and thus cannot guarantee optimal models. Therefore, learning optimal MLNs is a hard task. In this paper, we propose an approach that optimizes a substantial part of MLN structure learning, wherein we learn the final set of features and their parameters simultaneously from simpler features by leveraging an optimal feature learning algorithm.

We now briefly discuss the sequence labeling problem. The contributions of this paper can be extended to other acyclic structured output classification settings. Since we derive our approach for structured output spaces, which is more general, the derivations trivially cover simpler MLN settings.

Sequence Labeling: Sequence labeling is the task of assigning a class label to each instance in a sequence of observations. Typical sequence labeling algorithms learn probabilistic information about the neighboring states along with

[1] Grounding is replacing variables with constants/objects.

the probabilistic information about the observations. Hidden Markov Models (HMM) [10] and Conditional Random Fields (CRF) [11] are two traditional systems used popularly for sequence labeling problems. A HMM makes use of the assumption that, in a sequence, the label y_t at sequence step t is independent of all previous labels given y_{t-1} at time $t-1$ and the observation $\mathbf{x}_t$ at time t is independent of all other variables given y_t, and factorize the joint probability distribution in the form of the transition (label-label) distribution and the emission (label-observation) distribution. During the training phase, parameters that maximize the joint probability of the input and output sequences in the training data are learned. Whereas, a CRF maximizes the conditional probability of the output sequence given the input sequence to learn parameters. These parameters are later used to identify the (hidden) label sequence that best explains a given sequence of observations. Inference is typically performed efficiently by a dynamic programming algorithm called the Viterbi algorithm [12]. Recently, Tsochantaridis *et al.* proposed a maximum margin framework for structured output spaces such as sequence labeling, which is referred to as StructSVM [13]. It generalizes the standard Support Vector Machines (SVM) with the margin defined as the difference in the score of the original output sequence with any other possible output sequence.

Recent works, including ours, have looked into the problem of learning better sequence labeling models by discovering the structure in the input space in the form of conjunctions [14,15]. However since these approaches employ a greedy search to discover useful conjunctions, an optimal model is not guaranteed. In [16], we proposed a Hierarchical Kernels based learning approach for Structured Output Spaces (StructRELHKL) for learning optimal conjunctive (propositional) features for sequence labeling. In this work, we look into leveraging the StructRELHKL framework to learn first order features. Before going into the details of our approach, we now give a brief introduction to one of the application areas of sequence labeling called the activity recognition domain, which is our motivating problem.

Activity Recognition: Activity recognition systems are ubiquitous in the modern era of smart systems. One specific example is the use of activity recognition systems and approaches to monitor the activities of users in domicile environments; for instance, to monitor the activities of daily living of elderly people living alone, for estimating their health condition [17,18,19]. Such non-intrusive settings typically have on/off sensors installed at various locations in a home. Binary sensor values are recorded at regular time intervals. The joint state of these sensor values at a particular time forms our observation. The user activity at a particular time forms the hidden state or label. The history of sensor readings and (manually) annotated activities can be used to train prediction models such as the Hidden Markov Model (HMM) [10], the Conditional Random Field (CRF) [11] or StructSVM [13], which could be later used to predict activities based on sensor observations [18,15].

Activity recognition datasets tend to be sparse; that is, one could expect very few sensors to be on at any given time instance. Moreover, in a setting such as activity recognition, one can expect certain combinations of (sensor) readings to be directly indicative of certain activities. While HMMs, CRFs and StructSVM attempt to capture these relations indirectly, Nair *et al.* illustrate that discovering activity specific conjunctions of sensor readings (as emission features in HMM) can improve the accuracy of prediction [15]. McCallum [20] follows a similar approach for inducing features for a CRF model to solve Natural Language Processing tasks. Both these approaches, since they employ a greedy search for discovering features, have the limitation that an optimal model is not guaranteed. An exhaustive search for optimal features is not feasible in real world settings. We, in [16], proposed a hierarchical kernels based learning approach (StructRELHKL) for learning optimal feature conjunctions for sequence labeling, which we build on for learning first order relational features in this work. We now present the sequence labeling problem in a Markov Logic framework.

Sequence Labeling as Markov Logic Network: The training objective in sequence labeling can be posed as learning features that make the score,

$$F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R} \tag{3}$$

of the original output sequence Y greater than any other possible output sequence, given an input sequence X. The score is defined as,

$$F(X, Y; \mathbf{f}) = \langle \mathbf{f}, \boldsymbol{\psi}(X, Y) \rangle \tag{4}$$

where $\boldsymbol{\psi}$ is the feature vector (features describing observation structure and transitions), and $\mathbf{f}$ is the weight vector. Inference is performed by the decision function $\mathcal{F} : \mathcal{X} \rightarrow \mathcal{Y}$ defined by

$$\mathcal{F}(X; \mathbf{f}) = \underset{Y \in \mathcal{Y}}{\arg\max}\ F(X, Y; \mathbf{f}) \tag{5}$$

In this work, we intend to learn first order features for the emission relationships and their weights, while learning the weights for all the transition relationships. Therefore, our $\boldsymbol{\psi}$ will include label specific emission features in first order along with the transition features. For example, rules of the form *activity(T, prepareBreakFast)* ← *microwave(T), previousItemMoved(T, X), breakFastItem(X)* can be an emission rule/feature. Whereas transition rules can include rules such as *activity(T, eatBreakFast)* ← *activity(T-1, prepareBreakFast)*.

This can be viewed as a Markov Logic Network. We refer to this MLN for sequence labeling as a Markov Logic Chain (MLC). In this context, HMM, CRF, StructSVM, *etc.* can be viewed as special instances of MLC. Learning first order MLC (or MLN) is challenging due to the huge feature space. Conventional MLN systems learn the structure using greedy search algorithms and thus could lead to local optimum models.

We have recently proposed a hierarchical kernels based approach for structured output classification to learn MLC features in propositional space, which is a sub-instance of general MLCs. The Hierarchical kernel learning approach, originally proposed by Bach [21] for binary classification problems, learns an optimal set of sparse and simple features that has an ordering defined. We have extended the HKL approach in [16] to structured output spaces, which we call Rule Ensemble Learning using Hierarchical Kernels in Structured Output Spaces (StructRELHKL). StructRELHKL, in sequence labeling, exploits the hierarchical structure in the (exponential) space of emission features and efficiently learns a sparse set of simple propositional features and their weights. One of the fundamental requirements of these approaches is the summability of kernels over descendants in polynomial time. Since our focus here is to learn features in first order settings, that has inherent complexities such as huge number of groundings, variable sharing, background knowledge, refinement operators using unifications and anti-unifications, subsume equivalence *etc.*, it is hard to sum the descendant kernels in polynomial time. Therefore, the first order extension of StructRELHKL is non-trivial and is a challenging problem. In this paper, we investigate the possibility of leveraging StructRELHKL in discovering first order features. We briefly introduce the intuitions for our approach next.

Leveraging StructRELHKL for Learning First Order MLCs: We intend to learn first order features for sequence labeling. An example of first order feature is a feature that capture input relationships across sequence steps. Gutmann and Kersting recently proposed a first order extension of CRF (TildeCRF) that captures input relationships across sequence steps [22]. However, this approach also employs a greedy search to discover useful features. To the best of our knowledge, no previous work has explored the possibility of incorporating optimal feature learning in any step of MLC structure learning.

As discussed in the previous paragraphs, it is not feasible to apply StructRELHKL directly to learn first order MLCs. Although, grounding the first order predicates with all possible constants (according to the language bias) and leveraging StructRELHKL to learn optimal conjunctions of them seems to solve the problem, it is not feasible in large settings due to the huge search spaces. We therefore propose to employ StructRELHKL for learning optimal conjunctions of a powerful subclass of features, in the process of learning MLC features. To this end, we categorize first order MLN features based on their complexity and identify the class of features that can be efficiently constructed from simpler ones by StructRELHKL[2]. We identify a self-contained class of features called the absolute features ($\mathcal{AF}$) as the building blocks, whose unary/multiple conjunctions result in the final model. Our approach first generates $\mathcal{AF}$s that cover a threshold number of examples (weak relevance) and employs the StructRELHKL algorithm to simultaneously learn optimal conjunctions of $\mathcal{AF}$s and their weights

[2] In this work, we restrict our discussion to function-free first order definite features. A class specific feature can be constructed by conjoining the body literals of a definite clause whose head depicts the class label.

(as against learning them separately and/or greedily). We learn optimal MLCs with respect to weakly relevant $\mathcal{AF}$s. To summarize, we present the challenges and possibilities for optimally and simultaneously learning the structure as well as parameters of MLCs. We evaluate the efficiency of our approach on a publicly available activity recognition dataset and compare our results against TildeCRF [22], the state-of-the-art relational sequence tagging tool. A brief summary of some of the related approaches is given in the next paragraph.

Related Work: Huynh and Mooney in [23] propose an online weight learning algorithm for MLNs using a max-margin framework. The approach is for learning only the weights, whereas, our focus is on learning the structure as well as parameters for sequence labeling problems. In [24], Huynh and Mooney discusses their online algorithm for learning the structure and parameters of MLNs. In each iteration of their approach, the current model is used to predict the output, and if the prediction is wrong, the incorrect prediction is treated as a negative example, new clauses are learned that differentiate the true and false examples. The new clauses are learned by searching for atoms that are in the true example and not in the false one. This is performed by a relational path finding algorithm. Weight learning is then performed by an L1 regularized formulation, which also nullifies many non relevant clauses in the current structure. In contrast, our approach employs a batch learning algorithm. It learns a large number of candidate clauses called absolute clauses, which are then conjoined optimally to learn the final model. Our approach optimizes a convex formulation to learn structure and parameters with respect to the absolute clauses (features). A Logical Hidden Markov Model is discussed in [25], which deals with sequences over logical atoms. A model selection approach for Logical Hidden Markov Model is proposed in [26], which is based on Expectation Maximization algorithm and Inductive Logic Programming principles. Our approach differs from their approach in the sense that our objective is to explore the relationships among multiple observations at a sequence step to improve efficiency of sequence labeling. Thon *et al.* in [27] and [28] elaborate on relational markov processes which are concerned with efficient parameter learning and inference. They assume that a structure has been provided upfront. Similarly, a relational bayesian network learning is discussed in [29] with the goal of learning the parameters given the structure of the bayes-net.

TildeCRF [22] has an objective similar to our approach, where the relational structure and parameters of a CRF for sequence labeling are learned. TildeCRF uses relational regression trees and gradient tree boosting for learning the structure and parameters. The main difference of our approach with TildeCRF is that in our approach, unlike in TildeCRF, we derive convex formulations for a significant portion of learning steps.

Paper Organization: In section 2, the complexity based categorization of features is discussed. We discuss our approach in section 3. Experimental setup and results are discussed in section 4 and we conclude the paper in section 5.

2 First Order Definite Features for Markov Logic Chains

Most of the Inductive Logic Programming (ILP) systems and their statistical extensions (SRL systems) learn clauses by searching a space (often a lattice) of clauses in the domain. The search space is typically controlled by language restrictions, which define the type of clauses to be learned. One common way to solve a classification problem is to learn definite clauses (clauses having one head predicate conditioned on the values of zero or more body predicates). Since we are interested in such a setup, we confine our discussion to the space of definite clauses. We use the terms first order definite clause and first order feature interchangeably, as one can be derived from the other. We start by defining categories of predicates and then discuss the complexity based classification of features.

Similar to *structural* and *property* predicates in 1BC clauses [30], we define two types of predicates, *viz. (inter) relational* and *quality* predicates. A *relational* predicate is a binary predicate that represents the relationship between *types* or between a *type* and its parts, where a *type* is an entity described by its attributes. A *quality* predicate is a predicate that reveals a quality/property of a part (or subset) of a *type*. From the example clauses given below, `microwave(_,_)`, and `before(_,_)` are *relational* predicates and all other predicates are *quality* predicates.

1. `prepareDinner(T) :- microwave(T,X1), soak(X1)`
2. `prepareDinner(T) :- microwave(T,X1)`
3. `prepareLunch(T) :- microwave(T,X1), powdered(X1), cereal(X1)`
4. `prepareLunch(T) :- microwave(T,X1), dry(X1), microwave(T,X2), before(X1,X2), wet(X2), cereal(X2)`
5. `prepareDinner(T) :- microwave(T,X1), soak(X1), microwave(T,X2), nonSoak(X2)`

We now categorize first order definite features based on complexity into Absolute Features ($\mathcal{AF}$), Primary Features ($\mathcal{PF}$), Composite Features ($\mathcal{CF}$) and Definite Features ($\mathcal{DF}$). The definitions of $\mathcal{AF}$ and $\mathcal{CF}$ are used in this paper, while the other categories are presented for supporting the definitions for these.

Absolute Features ($\mathcal{AF}$):
In absolute features (clauses), new local variables can only be introduced in a *relational* predicate, where a local variable is a variable not present in the head predicate. Unlike in 1BC clauses, any number of new local variables can be introduced in a *relational* predicate. Any number of *relational* and *quality* predicates can be conjoined to form a $\mathcal{AF}$ such that the resultant $\mathcal{AF}$ is minimal and the local variables introduced in *relational* predicates are consumed by some other *relational* or *quality* predicates. Here a minimal clause is one which cannot be constructed from smaller clauses that share no common variables other than that in the head. So clauses 1, 3 and 4 above are $\mathcal{AF}$s whereas clauses 5 (since it is not minimal) and 2 (since variable `X1` is not consumed) are not.

Primary Features ($\mathcal{PF}$):
Primary features (clauses) are absolute clauses (features) that have at-most one *quality* predicate for every new local variable introduced. This is similar to elementary features in [30] except that elementary features allow only one new local variable in a *structural* predicate. Clause 1 is a $\mathcal{PF}$ whereas the other clauses do not conform to the restrictions imposed.

Composite Features ($\mathcal{CF}$):
Composite Features (clauses) are definite clauses that are formed by the conjunction of one or more $\mathcal{AF}$s without unification of body literals. Only the head predicates are unified. As in $\mathcal{AF}$s, every local variable introduced in a relational predicate should be consumed by other relational or quality predicates. Clauses 1, 3, 4 and 5 are $\mathcal{CF}$s where as 2 is not.

First Order Definite Features ($\mathcal{DF}$):
First order definite features (clauses) are features with none of the above restrictions. Therefore, all the given examples are $\mathcal{DF}$s.

We now state some of the relationships between these categories of features.

Claim 1. The set of primary features is a proper subset of the set of Absolute Features. That is, $\mathcal{PF} \subset \mathcal{AF}$.

Proof. From definition, $\mathcal{PF}$s are $\mathcal{AF}$ with the restriction that a new local variable introduced should be transitively consumed by a single *quality* predicate. Hence $\mathcal{PF} \subseteq \mathcal{AF}$. Now, consider the clause 3 above, which is an $\mathcal{AF}$ but not a $\mathcal{PF}$. Hence, $\mathcal{PF} \neq \mathcal{AF}$.

Claim 2. The set of absolute features is a proper subset of the set of composite features. That is, $\mathcal{AF} \subset \mathcal{CF}$.

Proof. From definition, $\mathcal{CF}$s are conjunctions of one or more $\mathcal{AF}$s. Therefore, all $\mathcal{AF}$s are $\mathcal{CF}$s (unary conjunctions). Now, consider the clause 5 above, which is a $\mathcal{CF}$ but not a $\mathcal{AF}$. Hence, $\mathcal{AF} \neq \mathcal{CF}$.

Claim 3. The set of composite features is a proper subset of the set of full first order definite features. $\mathcal{CF} \subset \mathcal{DF}$.

Proof. From definition, $\mathcal{DF}$s are first order definite clauses without any restrictions imposed for $\mathcal{CF}$s. Therefore, $\mathcal{CF} \subseteq \mathcal{DF}$. Now consider the clause 2 above, which is a $\mathcal{DF}$ but not a $\mathcal{CF}$. Therefore, $\mathcal{CF} \neq \mathcal{DF}$.

Claim 4. Every $\mathcal{AF}$ can be constructed from $\mathcal{PF}$s using unifications.

Proof. The difference an $\mathcal{AF}$ has with $\mathcal{PF}$ is that it can have more than one *quality* predicates for each local variable introduced. Let l_p be a *relational* literal in the body of a $\mathcal{AF}$ clause which introduces only one local variable. Let $l_1, l_2, \ldots, l_{p-1}$ be the set of *relational* literals in the body, which l_p depends on. Let there be $n \geq 0$ number of dependency chains starting from l_p to some *quality* predicates, each of which is represented as $l^i_{p+1}, \ldots, l^i_k$. We define l_p as a pivot

literal if $n > 1$. For simplicity, we assume there is only one pivot in a clause. Now, we can construct n $\mathcal{PF}$ clauses from this with the body of the i^{th} clause as $l_1, l_2, \ldots, l_{p-1}, l_p, l^i_{p+1}, \ldots, l^i_k$, where l^i_k is a *quality* predicate. It is trivial to see that these n clauses can be unified to construct the original $\mathcal{AF}$. For multiple pivot literal clauses, the above method can be applied recursively until $\mathcal{PF}$ clauses are generated. The proof can be extended to pivot literals with multiple new local variables by using a dependency tree structure in place of chain.

Claim 5. Every $\mathcal{CF}$ can be constructed from $\mathcal{AF}$s by conjunctions.

Proof. By definition.

We briefly discuss below some of the existing complexity based categorization of first order definite features in the following paragraph.

1BC clauses and elementary clauses introduced in [30] are similar to $\mathcal{AF}$s and $\mathcal{PF}$s respectively with the restriction that a structural predicate can have only one new local variable. Simple clauses are defined in [31] as the clause with at-most one sink literal, where a sink literal is one which has no other literal dependent on it. Simple clauses need not have a sink for a local variable and thus differs from $\mathcal{PF}$s. We present the proposed approach for learning first order features for sequence labeling in the next section.

3 Towards Optimal Feature Learning for Markov Logic Chains

We now formalize our intutions explained in the introduction section for leveraging StructRELHKL to learn first order MLC features.

As explained in the introduction section, we focus on learning first order features for sequence labeling. An example class of first order features in sequence labeling domain is the features that capture input relationships across sequence steps. TildeCRF [22] is an existing approach that explores such feature space. However the approach pursued is greedy. Although the Hierarchical Kernels based feature learning approach for structured output spaces (StructRELHKL) [16], that we proposed recently, learns optimal conjunctive features in propositional settings, it has limitations in exploring first order space. The main limitation is that, due to the complex refinement operators using unification and anti-unification, any ordering of the first order features restricts the requirement of StructRELHKL to sum the descendant kernels in polynomial time. Alternatively, grounding the first order predicates with all possible constants and then leveraging StructRELHKL to learn optimal features is not feasible due to the huge feature space. Moreover, such settings could lead to less effective models, due to the redundant information present in the models learned. Here, we explore the space of first order features, wherein, our optimal conjunctive feature learning algorithm is leveraged to the extent possible in the learning step.

MLC Features: Our objective is to learn label specific relational features as observation features for sequence labeling. These types of features can be represented in the form of definite clauses. As discussed in section 2, Composite Features ($\mathcal{C}F$) are first order definite features with the restriction that local variables introduced in relational predicates have to be used by other relational or quality predicates. This category of features is particularly interesting in our case, because they are powerful to represent a large part of definite clauses. $\mathcal{C}F$s are not defined to include clauses such as clause 2 in section 2. In predicting smaller granular activities such as *prepareBreakFast, prepareLunch, etc.*, what is being cooked in the microwave oven becomes interesting. Absence of such information doesn't give meaning to the rule. On the other hand, if one wants to predict larger granular activities such as whether the current activity is cooking or not, then what is being cooked inside microwave is less important. To this effect, a new predicate *microwaveOn(T)* can be defined and we can construct $\mathcal{C}F$s from this. We have also proved that composite features ($\mathcal{C}F$) can be constructed from absolute features ($\mathcal{A}F$) with unary/multiple conjunctions without unifications and $\mathcal{A}F$s can be constructed from primary features ($\mathcal{P}F$) with unifications. Therefore, the space of $\mathcal{C}F$s can be defined as a partial order over $\mathcal{P}F$s with unifications and conjunctions. As discussed before, the optimal feature learning approach (StructRELHKL) is not applicable in a partial order with unifications. However, since the space of $\mathcal{C}F$s is a partial order over conjunctions of $\mathcal{A}F$s, it is possible to leverage the StructRELHKL framework to learn $\mathcal{C}F$s from $\mathcal{A}F$s. With proper language restrictions, $\mathcal{A}F$s can be generated by ILP methods. Our approach is to generate $\mathcal{A}F$s using ILP techniques and employ StructRELHKL to find optimal conjunctions of $\mathcal{A}F$s. The next paragraph gives some insight into the optimality of the resultant MLC.

We define an $\mathcal{A}F$ as strongly relevant if it is constructed from optimal unifications of $\mathcal{P}F$s, which is a hard task. On the other hand, we consider a feature to be weakly relevant if it covers at-least a threshold percentage of examples. We are interested in $\mathcal{A}F$s that are at-least weakly relevant and our approach learns optimal MLCs with respect to the weakly relevant $\mathcal{A}F$s. We now derive our approach in the following paragraph.

Optimizing MLC Feature Learning Steps: The objective is to first learn self contained weakly relevant $\mathcal{A}F$s by employing ILP methods and then learn optimal conjunctions of $\mathcal{A}F$s and their weights simultaneously by leveraging the StructRELHKL approach. Since there is a plethora of literature available on ILP approaches for searching clauses with language restrictions [2,1], we skip the discussion on generating $\mathcal{A}F$s and move on to give an overview of StructRELHKL [16] framework to construct $\mathcal{C}F$s.

An MLC should have features defining transition relationships between the labels as well as the observation relationships. Let $\boldsymbol{\psi_T}$ represent the feature vector corresponding to all transition features and $\boldsymbol{\psi_{CF}}$ represent the feature vector corresponding to all observation features (space of all possible conjunctions of $\mathcal{A}F$s). For the sake of visualization, lets assume there is a partial order of $\mathcal{C}F$s

for each label in a multi-class setup. As defined in the introduction section, $\boldsymbol{\psi}$ represents a combination of $\boldsymbol{\psi_T}$ and $\boldsymbol{\psi_{CF}}$. We assume that both $\boldsymbol{\psi_{CF}}$ and $\boldsymbol{\psi_T}$ are of dimension equal to the dimension of $\boldsymbol{\psi}$ with zero values for all elements not in their context. In similar spirit, the feature weight vector $\mathbf{f}$ can be constructed from $\mathbf{f_{CF}}$ and $\mathbf{f_T}$. Similarly, $\mathcal{V}$, the indices of the elements of $\boldsymbol{\psi}$, can be constructed from $\mathcal{V}_{\mathbf{CF}}$ and $\mathcal{V}_{\mathbf{T}}$. Our objective is to simultaneously select a sparse set of $\mathcal{C}F$s and their weights along with all the transition feature weights. To achieve this we build on the StructSVM framework [13] and employ a sparsity inducing hierarchical regularizer [21,32] on emission features and the standard 2-norm regularizer for transition features (as sparsity is desired only in the observation space). The margin of this Support Vector Machine (SVM) setup is defined as the difference in scores (defined in (4)) of the original output sequence and any other possible output sequence. Therefore, the objective can be stated as learning a sparse and simple set of observation features and all transition features that maximize the difference in scores of original sequences with any other possible sequence in the training data. These features can later be used during inference to find the best sequence among the set of possible sequences for a given observation. The SVM objective can be stated as,

$$\begin{aligned} &\min_{\mathbf{f},\boldsymbol{\xi}} \frac{1}{2}\Omega_{CF}(\mathbf{f_{CF}})^2 \;+\; \frac{1}{2}\Omega_T(\mathbf{f_T})^2 \;+\; \frac{C}{m}\sum_{i=1}^{m}\xi_i, \\ &\forall i, \forall Y \in \mathcal{Y} \setminus Y_i : \; \langle \mathbf{f}, \boldsymbol{\psi}_i^{\delta}(Y)\rangle \geq 1 - \frac{\xi_i}{\Delta(Y_i, Y)} \\ &\forall i : \; \xi_i \geq 0 \end{aligned} \tag{6}$$

where $\Omega_{CF}(\mathbf{f_{CF}})$ is the hierarchical $(1,\rho)$ norm regularizer [32] defined as, $\sum_{v \in \mathcal{V}_{\mathbf{CF}}} d_v \parallel \mathbf{f_{CF}}_{D(v)} \parallel_\rho$, $\rho \in (1,2]$, $d_v \geq 0$ is a prior parameter showing usefulness of the feature conjunctions, $D(v)$ represents the set of descendants (including itself) of node v in the partial order (similarly $A(v)$ represents the set of ancestors of node v), $\mathbf{f_{CF}}_{D(v)}$ is the vector with elements as $\parallel f_{CFw} \parallel_2$, $\forall w \in D(v)$, and $\parallel . \parallel_\rho$ represents the ρ-norm , $\Omega_T(\mathbf{f_T})$ is the 2-norm regularizer $\big(\sum_i f_{Ti}^2\big)^{\frac{1}{2}}$, m is the number of examples, C is the regularization parameter, ξ's are the slack variables introduced to allow errors in the training set in a soft margin SVM formulation, $X_i \in \mathcal{X}$ and $Y_i \in \mathcal{Y}$ represent the i^{th} input and output sequences respectively, $\langle \mathbf{f}, \boldsymbol{\psi}_i^{\delta}(Y)\rangle$ represents the value $\langle \mathbf{f}, \boldsymbol{\psi}(X_i, Y_i)\rangle - \langle \mathbf{f}, \boldsymbol{\psi}(X_i, Y)\rangle$, and $\Delta(Y, \hat{Y})$ represents the loss when the true output is Y and the prediction is $\hat{Y}$.

The 1-norm in $\Omega_{CF}(\mathbf{f_{CF}})$ forces many of the $\parallel \mathbf{f_{CF}}_{D(v)} \parallel_\rho$ to be zero. Even in cases where $\parallel \mathbf{f_{CF}}_{D(v)} \parallel_\rho$ is not forced to zero, the ρ-norm forces many of node v's descendants to zero. This ensures a sparse and simple set of features.

The above SVM setup has to deal with two exponential spaces. The first is that of the exponential space of features and the second problem is the exponential number of constraints for the objective. The next paragraph outlines solution to this.

By solving (6), most of the emission feature weights are expected to be zero. As illustrated in [32,16], the solution to the problem when solved with the original

set of features is the same but requires less computation when solved only with features having non zero weights at optimality. Therefore, an active set algorithm can be employed to incrementally find the optimal set of non zero weights. In each iteration of the active set algorithm, since the constraint set in (6) is exponential, a cutting plane algorithm has to be used to find a subset of constraints of polynomial size so that the corresponding solution satisfies all the constraints with an error not more than ϵ. The MLC objective is solved by deriving a dual for (6) with the feature set reduced to active features and a sufficient condition to check for optimality. The active set algorithm starts with the top nodes in the partial order as active set and at each iteration, solves the dual of (6), checks a sufficient condition for optimality, adds those nodes in the sources (subset of a set of nodes in a partial order that have no parent in the set) of the active set's complement that violate the sufficient condition to the active set. The process continues until there are no nodes violating the sufficient condition. The dual of (6) is derived as,

$$\min_{\eta \in \Delta_{|\mathcal{V}|,1}} g(\eta) \tag{7}$$

where $g(\eta)$ is defined as,

$$\max_{\alpha \in S(\mathcal{Y},C)} \sum_{i,Y \neq Y_i} \alpha_{iY} - \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{\kappa}_{\mathbf{T}} \boldsymbol{\alpha} - \frac{1}{2}\Big(\sum_{w \in \mathcal{V}} \zeta_w(\eta) (\boldsymbol{\alpha}^\top \boldsymbol{\kappa}_{\mathbf{FC}w} \boldsymbol{\alpha})^{\bar{\rho}} \Big)^{\frac{1}{\bar{\rho}}}, \tag{8}$$

$$\zeta_w(\eta) = \big(\sum_{v \in A(w)} d_v^\rho \eta_v^{1-\rho} \big)^{\frac{1}{1-\rho}}, \quad \bar{\rho} = \tfrac{\rho}{2(\rho-1)}, \text{ and}$$

$$S(\mathcal{Y}, C) = \{ \boldsymbol{\alpha} \in \mathbb{R}^{m(n^l-1)} \mid \alpha_{i,Y} \geq 0,\ m \sum_{Y \neq Y_i} \tfrac{\alpha_{iY}}{\Delta(Y,Y_i)} \leq C,\ \forall i, Y \}$$

A sufficient condition for optimality, with the current active set $\mathcal{W}$, with a duality gap less than ϵ is derived as

$$\begin{aligned} \max_{u \in sources(\mathcal{W}^c)} \sum_{i,Y \neq Y_i} \sum_{j,Y' \neq Y_j} & \boldsymbol{\alpha}_{\mathcal{W}iY}^\top \sum_{p=1}^{l_i} \sum_{q=1}^{l_j} 2\Big(\prod_{k \in u} \frac{\psi_{FCk}(\mathbf{x}_i^p)\psi_{FCk}(\mathbf{x}_j^q)}{b^2} \Big) \\ & \Big(\prod_{k \notin u} \big(1 + \frac{\psi_{FCk}(\mathbf{x}_i^p)\psi_{FCk}(\mathbf{x}_j^q)}{(1+b)^2}\big) \Big) \boldsymbol{\alpha}_{\mathcal{W}jY'} \\ & \leq \Omega_{FC}(\mathbf{f}_{\mathbf{FC}\mathcal{W}})^2 + \Omega_T(\mathbf{f}_{\mathbf{T}\mathcal{W}})^2 + 2(\epsilon - e_{\mathcal{W}}) \end{aligned} \tag{9}$$

Inference is performed by applying dynamic programming methods [12]. We now discuss the experimental setup in the following section.

4 Experiments

Our approach first learns a weakly relevant set of absolute features and then learns their optimal conjunctions using StructRELHKL. We use Warmr [33,34],

an ILP data mining algorithm that learns frequent patterns reflecting one to many and many to many relationships, to learn absolute features. Warmr uses an efficient level wise search through the pattern space. With proper language bias, absolute features can be generated by Warmr. The absolute features learned by Warmr are then input to StructRELHKL code to learn the structure and parameters of the final Markov Logic Chain. Our StructRELHKL is a java program, which implements the active set algorithm. In each iteration of the active set algorithm, the dual objective is solved using the current model, a sufficiency condition for optimality is checked on the nodes in the complement of active set that has parents only in the active set, and the violating nodes are added to the active set. The sub-problem of solving the dual is performed by a cutting plane algorithm, which starts with an empty set of constraints. In each iteration, the algorithm solves the objective with the current set of constraints and adds new constraints that violate the margin (defined by the current model) at-least by a threshold value.

Our experiments are carried out on a publicly available activity recognition dataset. This data, provided by Kasteren *et al.* [18], has been extracted from a household fitted with 14 binary sensors at *bedroomDoor*, *bathroomDoor*, *microwave*, $fridge$, *cupboards*, *etc.*. Eight activities have been annotated for 4 weeks by a subject. Activities are daily house hold activities like *sleeping*, $usingToilet$, $preparingDinner$, $preparingBreakfast$, $leavingOut$, *etc.* There are 40006 data instances. The dataset is skewed. For instance, the activity $leavingOut$ occurs about 56.4% of time, while the activity $preparingBreakfast$ occurs only 0.3% of time. Since the authors of the dataset are from the University of Amsterdam, we will refer to the dataset as the UA data.

The data is split into different sequences and each sequence is treated as an example. We perform our experiments in a four fold cross-validation setup. On each fold, we train our model on 25% of data and test on the remaining 75%[3]. We report performance in terms of micro-average and macro-average labeling accuracies. The micro-average accuracy, referred to as time-slice accuracy in [18], is the weighted average of per-class accuracies, weighted by the number of instances of the class. Macro-average accuracy, referred to as class accuracy in [18], is the average of the per-class accuracies.

We compare our approach with the TildeCRF [22]. The TildeCRF is the state-of-the-art ILP approach for learning relational features for sequence labeling, and works in the same feature space that we are interested in. The comparison is outlined in Table 1.

It can be observed that our results are competitive to the state-of-the-art sequence labeling approach, TildeCRF. Our approach returned better micro-average accuracy than TildeCRF, while reporting lesser macro-average accuracy. Micro-averaged accuracy is typically used as the performance evaluation measure. However in data that is biased towards some classes, macro-average

[3] Since in real world problems such as activity recognition, a trained model has to be used for a period much longer than the period training data is collected, here we considered training on a small part of the data and testing on the rest.

Table 1. Micro average accuracy (%), and macro average accuracy (%) using TildeCRF and the proposed MLC approach on UA dataset

	Micro avg.(%)	Macro avg.(%)
TildeCRF	56.22 ($\pm$12.08)	35.36 ($\pm$6.55)
MLC	60.36 ($\pm$6.99)	30.39 ($\pm$4.31)

accuracy being too low is considered to be a poor performance. Our standard deviation values are also less compared to the competitor algorithm. Our approach on average took about 25 hours for training while TildeCRF took 2.5 hours.

5 Conclusion

Recent works have shown the importance of learning the input relational structure (features) in sequence labeling problems [14,15]. Most of the existing feature learning approaches employ greedy search techniques to discover relational features. We have recently proposed a hierarchical kernels based approach for structured output classification (StructRELHKL) that is capable of learning optimum feature conjunctions to build propositional sequence labeling models [16]. However, StructRELHKL works in propositional domain and has limitations in exploring first order space. In this paper, we presented the challenges and possibilities of leveraging StructRELHKL to optimize first order feature learning steps. To this end, we categorized first order features based on complexity and identified the class of features that can be constructed using StructRELHKL from simpler ones. We therefore learn a simple and powerful sub-class of first order features called the absolute features using Inductive Logic Programming techniques and learn their optimal conjunctions using StructRELHKL to build the final model. Our experiments show competitive performance compared to the state-of-the-art relational sequence labeling tool, the TildeCRF.

References

1. Getoor, L., Taskar, B.: Statistical relational learning. MIT Press (2006)
2. Nienhuys-Cheng, S.H., de Wolf, R.: Foundations of Inductive Logic Programming. Springer-Verlag New York, Inc., Secaucus (1997)
3. Richardson, M., Domingos, P.: Markov logic networks. Mach. Learn. 62(1-2), 107–136 (2006)
4. Domingos, P., Kok, S., Poon, H., Richardson, M., Singla, P.: Unifying logical and statistical AI. In: Proceedings of the 21st National Conference on Artificial Intelligence, AAAI 2006, vol. 1, pp. 2–7. AAAI Press (2006)
5. Muggleton, S., De Raedt, L., Poole, D., Bratko, I., Flach, P., Inoue, K., Srinivasan, A.: ILP turns 20. Mach. Learn. 86(1), 3–23 (2012)
6. Zhuo, H.H., Yang, Q., Hu, D.H., Li, L.: Learning complex action models with quantifiers and logical implications. Artif. Intell. 174(18), 1540–1569 (2010)

7. Kok, S., Domingos, P.: Learning the structure of markov logic networks. In: Proceedings of the 22nd International Conference on Machine Learning, ICML 2005, pp. 441–448. ACM, New York (2005)
8. Biba, M., Ferilli, S., Esposito, F.: Structure learning of markov logic networks through iterated local search. In: Proceedings of the 2008 Conference on ECAI 2008: 18th European Conference on Artificial Intelligence, pp. 361–365. IOS Press, Amsterdam (2008)
9. Khot, T., Natarajan, S., Kersting, K., Shavlik, J.: Learning markov logic networks via functional gradient boosting. In: Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM 2011, pp. 320–329. IEEE Computer Society, Washington, DC (2011)
10. Rabiner, L.R.: Readings in speech recognition, pp. 267–296. Morgan Kaufmann Publishers Inc., San Francisco (1990)
11. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML 2001, pp. 282–289. Morgan Kaufmann Publishers Inc., San Francisco (2001)
12. Forney, G.J.: The viterbi algorithm. Proceedings of IEEE 61(3), 268–278 (1973)
13. Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: Proceedings of the Twenty-First International Conference on Machine Learning, ICML 2004, pp. 104–111. ACM, New York (2004)
14. McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, CONLL 2003, vol. 4, pp. 188–191. Association for Computational Linguistics, Stroudsburg (2003)
15. Nair, N., Ramakrishnan, G., Krishnaswamy, S.: Enhancing activity recognition in smart homes using feature induction. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2011. LNCS, vol. 6862, pp. 406–418. Springer, Heidelberg (2011)
16. Nair, N., Saha, A., Ramakrishnan, G., Krishnaswamy, S.: Rule ensemble learning using hierarchical kernels in structured output spaces. In: Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)
17. Wilson, D.H.: Assistive intelligent environments for automatic health monitoring. PhD Thesis, Carnegie Mellon University (2005)
18. van Kasteren, T., Noulas, A., Englebienne, G., Kröse, B.: Accurate activity recognition in a home setting. In: Proceedings of the 10th International Conference on Ubiquitous Computing, UbiComp 2008, pp. 1–9. ACM, New York (2008)
19. Gibson, C., van Kasteren, T., Krose, B.: Monitoring homes with wireless sensor networks. In: Proceedings of the International Med-e-Tel Conference (2008)
20. McCallum, A.K.: Efficiently inducing features of conditional random fields. In: Proceedings of the Nineteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (2003)
21. Bach, F.: High-dimensional non-linear variable selection through hierarchical kernel learning. Technical report, INRIA, France (2009)
22. Gutmann, B., Kersting, K.: TildeCRF: Conditional random fields for logical sequences. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 174–185. Springer, Heidelberg (2006)
23. Huynh, T.N., Mooney, R.J.: Online max-margin weight learning with markov logic networks. In: Proceedings of the AAAI 2010 Workshop on Statistical Relational AI (Star-AI 2010), Atlanta, GA, pp. 32–37 (July 2010)

24. Huynh, T.N., Mooney, R.J.: Online structure learning for markov logic networks. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part II. LNCS, vol. 6912, pp. 81–96. Springer, Heidelberg (2011)
25. Kersting, K., De Raedt, L., Raiko, T.: Logical hidden markov models. Journal of Artificial Intelligence Research 25 (2006)
26. Kersting, K.: Say em for selecting probabilistic models for logical sequences. In: Proceedings of the Twenty First Conference on Uncertainty in Artificial Intelligence, pp. 300–307. Morgan Kaufmann (2005)
27. Thon, I.: Don't fear optimality: Sampling for probabilistic-logic sequence models. In: De Raedt, L. (ed.) ILP 2009. LNCS, vol. 5989, pp. 226–233. Springer, Heidelberg (2010)
28. Thon, I., Landwehr, N., De Raedt, L.: Stochastic relational processes: Efficient inference and applications. Mach. Learn. 82(2), 239–272 (2011)
29. Schulte, O., Khosravi, H., Kirkpatrick, A., Man, T., Gao, T., Zhu, Y.: Modelling relational statistics with bayes nets. In: Proceedings of 22nd International Conference on Inductive Logic Programming (ILP 2012). Springer (2012)
30. Flach, P., Lachiche, N.: 1BC: A first-order bayesian classifier. In: Džeroski, S., Flach, P. (eds.) ILP 1999. LNCS (LNAI), vol. 1634, pp. 92–103. Springer, Heidelberg (1999)
31. McCreath, E., Sharma, A.: LIME: A system for learning relations. In: Richter, M.M., Smith, C.H., Wiehagen, R., Zeugmann, T. (eds.) ALT 1998. LNCS (LNAI), vol. 1501, pp. 336–374. Springer, Heidelberg (1998)
32. Jawanpuria, P., Nath, J.S., Ramakrishnan, G.: Efficient rule ensemble learning using hierarchical kernels. In: Getoor, L., Scheffer, T. (eds.) ICML, pp. 161–168. Omnipress (2011)
33. Dehaspe, L., Toivonen, H.: Discovery of frequent datalog patterns. Data Min. Knowl. Discov. 3(1), 7–36 (1999)
34. Dehaspe, L., Toironen, H.: Relational data mining, pp. 189–208. Springer-Verlag New York, Inc., New York (2000)