Comparison between Explicit Learning and Implicit Modeling of Relational Features in Structured Output Spaces

Ajay Nagesh[123], Naveen Nair[123], and Ganesh Ramakrishnan[21]

[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
{ajaynagesh,naveennair,ganesh}@cse.iitb.ac.in

Abstract. Building relational models for the structured output classification problem of sequence labeling has been recently explored in a few research works. The models built in such a manner are interpretable and capture much more information about the domain (than models built directly from basic attributes), resulting in accurate predictions. On the other hand, discovering optimal relational features is a hard task, since the space of relational features is exponentially large. An exhaustive search in this exponentially large feature space is infeasible. Therefore, often the feature space is explored using heuristics. Recently, we proposed a Hierarchical Kernels-based feature learning approach (StructHKL) for sequence labeling [**?**], that optimally learns emission features in the form of conjunctions of basic inputs at a sequence position. However, StructHKL cannot be trivially applied to learn complex relational features derived from relative sequence positions. In this paper, we seek to learn optimal relational sequence labeling models by leveraging a relational kernel that computes the similarity between instances in an implicit space of relational features. To this end, we employ relational subsequence kernels at each sequence position (over a time window of observations around the pivot position) for the classification model. While this method of modeling does not result in interpretability, relational subsequence kernels do efficiently capture relational sequential information on the inputs. We present experimental comparison between approaches for explicit learning and implicit modeling of relational features and explain the trade-offs therein.

Keywords: Subsequence Kernels, StructSVM, Sequence Labeling

1 Introduction

Structured output classification has gathered significant interest in the machine learning community during the last decade [**?**,**?**,**?**,**?**].The goal of such works is to classify complex output structures such as sequences, trees, lattices or graphs, in which the class label at each node/position of the structure has to be inferred

based on observed evidence data. The possible space of structured outputs tends to be exponential and thus structured output classification is a challenging research area. We, in our research work, focus on a specific structured output classification problem, popularly known as sequence labeling. As in any classification setting, the sequence labeling domain is also characterized by complex relationships among entities and uncertainties in their relationships. Efficient models can be constructed by exploiting these relationships. However, discovering relationships that enhance the discriminative power of classifiers is a hard task, since the relationship space is often too large. Therefore, most of the research in sequence labeling and other structured output space classification, either ignore the complex relationships or use heuristics to learn the relationships. In this work, we focus on exploiting complex relationships in both the input as well as the output space in an efficient way to improve sequence labeling models. We begin with a brief introduction to the task of sequence labeling.

The objective in sequence labeling is to assign a state (class label) to every instance of a sequence of observations. Typical sequence labeling algorithms learn probabilistic information about the neighboring states along with the probabilistic information about the observations. Hidden Markov Models (HMM) [**?**], Conditional Random Fields (CRF) [**?**] and StructSVM [**?**] are three models used popularly for sequence labeling problems. The training objective can be posed as learning feature weights that make the score F ($F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$), of the true output sequence Y greater than any other possible output sequence, given an input sequence X. The score is defined as:

$$F(X, Y; \mathbf{f}) = \langle \mathbf{f}, \boldsymbol{\psi}(X, Y) \rangle \tag{1}$$

where $\boldsymbol{\psi}$ is the feature vector (describing observations 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

$$\mathcal{F}(X; \mathbf{f}) = \underset{Y \in \mathcal{Y}}{\arg\max}\ F(X, Y; \mathbf{f}) \tag{2}$$

Recent works have shown that learning the relational structure between input features improves the efficiency of sequence labeling models [**?**,**?**,**?**]. However, the space of relational features is exponential in the number of basic observations, making the discovery of useful features a difficult task. For instance, the simple case of learning features that are conjunctions of basic observations at any single sequence position results in a feature space that is exponential. The problem is further exacerbated if we consider complex relational features built from observations at different relative positions. An exhaustive search in this exponentially large feature space is infeasible. Therefore, most systems that learn relational features follow a greedy search strategy based on heuristics to select useful features. These approaches start with an initial (possibly empty) set of features and iteratively search (using some ordering of the feature space) for refinements that improve the heuristic score.

In our previous work [**?**], we propose and develop a Hierarchical Kernels based approach for optimally learning features which are conjunctions of basic

features at a particular sequence position (*simple conjuncts* or $\mathcal{SC}$s) for each label. The approach is referred to as Hierarchical Kernel Learning for Structured Output Spaces (StructHKL) [4]. Although it optimally learns the most discriminative $\mathcal{SC}$s, its applicability in learning complex relational features that are derived from observations at different relative positions in a sequence, is non-trivial and challenging. To address this issue, our follow-up work [**?**], determines simple feature classes that can be composed to yield complex ones, with the goal of formulating efficient yet effective relational feature learning procedures. We identify feature classes called absolute features ($\mathcal{AF}$) and composite features ($\mathcal{CF}$) in increasing order of their complexity respectively [5]. It is posited that optimal relational features can be learned by enumerating $\mathcal{AF}$s and discovering their useful compositions ($\mathcal{CF}$) using StructHKL. However, the space of $\mathcal{AF}$s is prohibitively large and it is not feasible to enumerate all of them in a domain. To circumvent this issue, we propose to selectively enumerate $\mathcal{AF}$s based on some relevance criteria such as the support of $\mathcal{AF}$s in the training set.

An $\mathcal{AF}$ is formed by combining one or more predicates which share variables. The partial ordering of $\mathcal{AF}$s does not comply with the requirement of StructHKL that the descendant kernels should be summable in polynomial time. This limits the possibility of leveraging StructHKL to optimally learn features in the space of $\mathcal{AF}$s (and its super-space of $\mathcal{CF}$s). For this reason, in the current piece of work, we leverage a relational kernel that computes the similarity between instances in an implicit feature space of $\mathcal{CF}$s. To this end, we employ the relational subsequence kernel [**?**] at each sequence/pivot position (over a time window of observations around it) for the classification model. We would like to learn composite features which capture relational information about basic observations at positions relative to the pivot position for every sequence step. This sequence information would provide a rich feature space for the algorithm to learn a more expressive model. However, explicitly enumerating such a feature space is not feasible due to the high dimensionality of the feature space. Relational subsequence kernels implicitly capture the effectiveness of this rich feature space. We also show that the feature space of $\mathcal{CF}$s (*explicit* features) are captured by the *relational subsequence kernels* (*implicit* features). While this way of modeling does not result in interpretability, relational subsequence kernels do efficiently capture the relational sequential information on the inputs.

We evaluate the performance of our approaches on publicly available activity recognition datasets. Our experiments show improvements over other standard and state-of-the-art sequence labeling techniques. The paper is organized as follows.

Section 2 discusses background work. We discuss our approach in Section 3. Experimental setup and results are discussed in Section 4 and we conclude the paper in Section 5.

---

[4] StructHKL is derived from StructSVM in which we use sparsity inducing hierarchical regulariser for observation features.

[5] For the definitions and examples of $\mathcal{AF}$, $\mathcal{CF}$ and other feature classes, please refer to [**?**].

2 Background

Approaches to learning relationships for sequence labeling could be based on basic input features at a single sequence step or input features at multiple sequence steps and/or relationships among output variables. Some of these approaches are discussed below.

McCallum [**?**] as well as Nair *et. al* [**?**] propose feature induction methods that iteratively construct feature conjunctions that increase an objective. These approaches start with an initial set of features (conjunctions or atomic) and at each step, consider a set of candidate features that are refinements of the current set of features. Features whose inclusion will lead to maximum increase in the objective are selected. Weights for the new features are trained. The steps are iterated until convergence. While McCallum trains a CRF model and uses conditional log-likelihood as the objective for the greedy induction, Nair *et. al* train an HMM and use prediction accuracy on a held out dataset (part of the training data) as the objective. This effectively solves the problem of incorrect assumption, that individual observations are independent, while not dealing with exponential observation space. Although these greedy feature induction approaches have been shown to improve performance, they cannot guarantee an optimal solution. An exhaustive search to find the optimal solution is expensive due to the exponential size of the search space.

Kersting *et. al.* [**?**] discusses the Logical Hidden Markov Model which is a relational representation of HMM. However, this work does not investigate learning the input structure. Thon *et. al* ([**?**], [**?**]) 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 [**?**] with the goal of learning the parameters given the structure of the bayes-net.

Hierarchical Kernel Learning for Structured Output Spaces (StructHKL) [**?**], optimally and efficiently learns discriminative features for multi-class structured output classification problems such as sequence labeling. StructHKL builds on the Support Vector Machines for Structured Output Spaces (StructSVM) model [**?**] for sequence prediction problems, wherein, all possible $\mathcal{SC}$s form the input features while the transition features are constructed from all possible transitions between state labels. A ρ-norm hierarchical regularizer is employed to select a sparse set of $\mathcal{SC}$s. Since there is a need to preserve all possible transitions, a conventional 2-norm regularizer is employed for state transition features. The exponentially large observation feature space is searched using an active set algorithm and the exponentially large set of constraints is handled using a cutting plane algorithm.

In our follow-up work [**?**], we learn complex relational features derived from relative sequence positions. We propose to enumerate $\mathcal{AF}$s and leverage StructHKL to learn their compositions, which are $\mathcal{CF}$s. However, it is noted that the space of $\mathcal{AF}$s is prohibitively large and therefore it is not feasible to enumerate all $\mathcal{AF}$s in a domain. As a solution we selectively enumerate $\mathcal{AF}$s based on some relevance criteria such as support of the $\mathcal{AF}$ in the training set. A feature is

considered to be *strongly relevant* if it helps the classification model to discern classes optimally. On the other hand, a feature is *weakly relevant* if it covers atleast a threshold percentage of examples. As discovering strongly relevant $\mathcal{AF}$s is a hard task, the focus is on discovering weakly relevant $\mathcal{AF}$s using Inductive Logic Programming tools. Pattern mining approaches are employed to discover a relevant set of $\mathcal{AF}$s. Specifically, a relational pattern miner called Warmr [**?**] is used. Warmr uses a modified version of Apriori algorithm [**?**] to find frequent patterns ($\mathcal{AF}$s) which have minimum support, as specified by the user. Once a set of relevant $\mathcal{AF}$s are enumerated, StructHKL is used to learn useful compositions of $\mathcal{AF}$s and their parameters to get the final model. This can be viewed as projecting the space of complex relational features such as $\mathcal{CF}$s into the space of $\mathcal{SC}$s and leveraging StructHKL.

TildeCRF [**?**] 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. Unlike in TildeCRF, in this work, we derive convex formulations for learning relational models.

In this paper, we provide operative definitions of the feature classes such as $\mathcal{AF}$ and $\mathcal{CF}$. For a more detailed exposition of the feature classes and the relationships between them, the reader is pointed to our previous work [**?**].

3 Implicit Modeling of Features for Sequence Labeling

In Section 1, we have stated our objective as exploiting complex relationships among input variables in sequence labeling problems to improve the efficiency of classification. We now formalize our intuitions and present our proposed approach in detail.

We have presented the training and inference objectives of sequence labeling problems in equations (1) and (2), where the features and feature weights are represented by $\boldsymbol{\psi}$ and $\mathbf{f}$, respectively. Elements of $\boldsymbol{\psi}$ correspond to the emission (basic input/observation) features and the transition features. We represent the emission and transition parts of the vector $\boldsymbol{\psi}$ as $\boldsymbol{\psi_E}$ and $\boldsymbol{\psi_T}$, respectively. We assume that both $\boldsymbol{\psi_E}$ and $\boldsymbol{\psi_T}$ are vectors of dimension equal to the dimension of $\boldsymbol{\psi}$ with zero values for all elements not in their context. That is, $\boldsymbol{\psi_E}$ has dimension of $\boldsymbol{\psi}$, but has zero values corresponding to the transition elements. In the dual space, we represent the kernels corresponding to transition and emission as κ_T and κ_T respectively. Our proposed approach is to leverage (implicitly or explicitly) discriminative observation features ($\boldsymbol{\psi_E}$) that capture complex relationships among input variables in an implicit manner.

In the previous sections, we have identified $\mathcal{CF}$s as the class of features that explicitly capture complex relationships among input variables at relative sequence positions. We have also defined $\mathcal{CF}$s as compositions of $\mathcal{AF}$s and that, since the partial ordering of $\mathcal{AF}$s does not comply with the requirements of StructHKL, it is not feasible to leverage StructHKL for learning features in the space of $\mathcal{AF}$s (and its super-space of $\mathcal{CF}$s). For this reason, in the sequence

labeling model, we leverage a relational kernel that computes the similarity between instances in an implicit feature space of $\mathcal{CF}$s. To this end, we employ the relational subsequence kernel [**?**] at each sequence position (over a time window of observations around the pivot position) for the classification model. We now briefly discuss about relational subsequence kernels in the following paragraph.

Subsequence kernels have been used to extract relations between entities in natural language text [**?**], where the relations are between protein names in biomedical texts. The features are (possibly non-contiguous) sequences of word and word classes anchored by the protein names at their ends. They extend the string kernels [**?**] for this task.

We have defined $\mathcal{CF}$s as explicit features that capture the subset of features at the current position as well as its relative positions. To implicitly capture this feature space, we employ a relational subsequence kernel at each position of the input sequence, with the current position as the pivot position. Suppose we consider an input $\mathbf{x}_i^p$ at position p for example i. Let the previous k positions relative to p have inputs $\mathbf{x}_i^{p-1}, \ldots \mathbf{x}_i^{p-k}$ and next l positions relative to p have inputs $\mathbf{x}_i^{p+1}, \ldots \mathbf{x}_i^{p+l}$. Let there be N basic features at a time-step t denoted by $x^{1^t} \ldots x^{N^t}$.[6] Essentially our sequence for the particular time-step pivoted at p, denoted by Q^p, is as follows:

$$Q^p = \{x^{1^{p-k}}, \ldots x^{N^{p-k}}\}, \ldots, \{x^{1^{p-1}}, \ldots x^{N^{p-1}}\},$$
$$\{x^{1^p}, \ldots x^{N^p}\}, \{x^{1^{p+1}}, \ldots x^{N^{p+1}}\} \ldots \{x^{1^{p+l}}, \ldots x^{N^{p+l}}\}$$

Given two sequences Q^p and Q^q, we define the relational subsequence kernel $SSK(Q^p, Q^q)$ as elaborated in [**?**]. This kernel will implicitly enumerate all possible common subsequences between Q^p and Q^q. We now show that the feature space of $\mathcal{CF}$s are captured by our relational subsequence kernel.

Claim: Relational subsequence kernels implicitly enumerate all the features in the feature space defined by Composite Features ($\mathcal{CF}$) given a constant context window.

Proof. By their definition the relational subsequence kernel $SSK(Q^i, Q^j)$ will implicitly enumerate all possible common subsequences between Q^i and Q^j. $\mathcal{CF}$s are conjunctions of features in the present time-step with features present in time-steps before and after the current time-step, which can be represented by $\mathcal{AF}$s. Since we are considering all the sub-sequences in the given context (time) window in the relational kernel, we implicitly enumerate space of $\mathcal{CF}$s.

We now define the kernel for StructSVM framework below, which represents the kernel resulting from the difference in values for the original and the candidate sequences. This stands for the inner product, $\langle \boldsymbol{\psi_i}^{\delta}(Y), \boldsymbol{\psi_i}^{\delta}(Y^{'}) \rangle$ with $\psi_i^{\delta}(Y)$ defined as: $\psi_i^{\delta}(Y) = \psi(X_i, Y_i) - \psi(X_i, Y)$. The kernel, which is a combination of transition (κ_T) and emission (κ_E) kernels, is defined as follows:

[6] Ignoring the example number i for simplicity

$$\kappa\big((X_i, Y_i, Y), (X_j, Y_j, Y^{'})\big) = \kappa_T(Y_i, Y, Y_j, Y^{'}) + \kappa_E\big((X_i, Y_i, Y), (X_j, Y_j, Y^{'})\big) \tag{3}$$

where

$$\kappa_T\big(Y_i, Y, Y_j, Y^{'}\big) = \kappa_T(Y_i, Y_j) + \kappa_T(Y, Y^{'}) - \kappa_T(Y_i, Y^{'}) - \kappa_T(Y_j, Y), \tag{4}$$

$$\begin{aligned}\kappa_T(Y_i, Y_j) &= \sum_{p=1}^{l_i-1}\sum_{q=1}^{l_j-1} \Lambda(y_i^p, y_j^q)\Lambda(y_i^{p+1}, y_j^{q+1}) \\ &= \sum_{p=2}^{l_i}\sum_{q=2}^{l_j} \Lambda(y_i^{p-1}, y_j^{q-1})\Lambda(y_i^p, y_j^q),\end{aligned} \tag{5}$$

$\Lambda(y_i^p, y_j^q) = 1$ if $y_i^p = y_j^q$; 0 otherwise. and

$$\kappa_E\big((X_i, Y_i, Y), (X_j, Y_j, Y^{'})\big) = \sum_{p=1}^{l_i}\sum_{q=1}^{l_j} \kappa_E(x_i^p, x_j^q)\Big(\Lambda(y_i^p, y_j^q) + \Lambda(y^p, y^{'q}) - \Lambda(y_i^p, y^{'q}) - \Lambda(y^p, y_j^q)\Big) \tag{6}$$

In our setting of subsequence kernels for StructSVM, the kernel $\kappa_E(x_i^p, x_j^q)$ is the relational subSequence kernel, where we may be considering some window time steps before and after p and q, with p and q as pivots.

The dual of the primal SVM formulation as defined by Tsochantaridis et. al. [**?**] for structured output spaces with the new kernel can be written as,

$$\begin{aligned}&\max_{\alpha} \sum_i \sum_{Y\in S_i} \alpha_{iY} - \frac{1}{2}\sum_i \sum_{Y\in S_i}\sum_j \sum_{Y'\in S_j} \alpha_{iY}\alpha_{jY'}\Big(\kappa_T^{\delta}(Y_i, Y, Y_j, Y^{'}) + \kappa_E^{\delta}\big((X_i, Y_i, Y), (X_j, Y_j, Y^{'})\big)\Big) \\ &\quad s.t.\ \ \forall i, \forall Y \in S_i, \quad \alpha_{iY} \geq 0 \\ &\qquad \forall i, \quad m\sum_{Y\in S_i} \frac{\alpha_{iY}}{\Delta(Y_i, Y)} \leq C.\end{aligned} \tag{7}$$

where α is the Lagrange dual variable, Δ is the loss function, S_i and S_j are the active constraint sets for example i and j respectively.

Now the margin violation cost function for a candidate output sequence Y for example i (for the cutting plane algorithm) can be written as,

$$\begin{aligned}H(Y) &= \Big(1 - \langle \boldsymbol{\psi}_i^{\delta}(Y), \mathbf{f}\rangle\Big)\Delta(Y_i, Y) \\ . \qquad &= \Big(1 - \sum_j \sum_{y'\in S_j} \alpha_{jY'}\langle \boldsymbol{\psi}_i^{\delta}(Y), \boldsymbol{\psi}_j^{\delta}(Y^{'})\rangle\Big)\Delta(Y_i, Y) \\ . \qquad &= \Big(1 - \sum_j \sum_{y'\in S_j} \alpha_{jY'}\kappa\big((X_i, Y_i, Y), (X_j, Y_j, Y^{'})\big)\Big)\Delta(Y_i, Y)\end{aligned} \tag{8}$$

where S_j is the active constraint set for example j.

The dual objective and the margin violation cost function can be plugged into the cutting plane algorithm to solve the objective. While this way of modeling does not result in interpretability, relational subsequence kernels do efficiently capture the relational sequential information on the inputs.

As in typical sequence labeling systems, we perform inference using a dynamic programming approach called the Viterbi algorithm [**?**].

The next section discusses our experiments and results.

4 Experiments

Our entire implementation is in Java. Our experiments are carried out on two publicly available activity recognition datasets. The first is the data provided by [**?**]. The dataset is extracted from a household fitted with 14 binary sensors. Eight activities have been annotated for 4 weeks. Activities are daily house hold activities like *sleeping*, *usingToilet*, *preparingDinner*, *preparingBreakfast*, *leavingOut*, *etc.* A data instance is recorded for a time interval of 60 seconds and there are 40006 such data instances. Since the authors of the dataset are from the University of Amsterdam, we will refer to the dataset as the UA data. The second data is the relational activity recognition data provided by [**?**] of Katholieke University, Leuven. We refer to the data as KU data. The data has been collected from a kitchen environment with 25 sensors/RFID attached to objects. There are 19 activities annotated. The data has been divided into 20 sequences. In this data, we perform our experiments in a leave one out cross-validation setup and report average of the accuracies returned from each fold.

In UA data, We use 25% of data for training and the rest for testing and report all accuracies by average across the four folds (the dataset is split into different sequences and each sequence is treated as an example). We report both micro-average and macro-average prediction accuracies. The micro-average accuracy is referred to as time-slice accuracy by [**?**], and is the average of per-class accuracies, weighted by the number of instances of the class. Macro-average accuracy, referred to as class accuracy by [**?**], is simply the average of the per-class accuracies. Micro-averaged accuracy is typically used as the performance evaluation measure. However, in data that is biased towards some classes, too worse macro-average is an indicator of a bad prediction model.

As we discussed previously, we leverage a relational kernel that computes the similarity between instances in an implicit feature space of $\mathcal{CF}$s. To this end, we employ the relational subsequence kernel [**?**] at each sequence position (over a time window of observations around the pivot position) for the classification model. We refer to this approach as Relational Subsequence Kernels for StructSVM approach (SubseqSVM).

We have compared our approach against TildeCRF [**?**], StructSVM [**?**] and enum$\mathcal{AF}$ [**?**]. While we treat StructSVM as a baseline for our experiments, TildeCRF is a state-of-the-art approach for learning relational features for sequence labeling, and operates in the same feature space that we are interested in. In

our experiments with StructSVM, individual basic features are assumed to be conditionally independent given the label.

The comparison of results on the UA dataset is outlined in Table 1. Results show that enum$\mathcal{AF}$ and our approach for learning complex features for sequence labeling *viz.* SubseqSVM performed better than the baseline approach (StructSVM) and the state-of-the-art approach (TildeCRF). Although enum$\mathcal{AF}$ optimally finds $\mathcal{CF}$s as conjunctions of (selectively enumerated) $\mathcal{AF}$s, the step for selectively enumerating $\mathcal{AF}$s is based on heuristics. In contrast, SubseqSVM works on a convex formulation and learns an optimal model. This explains the better performance of SubseqSVM.

The comparison of results on the KU dataset is outlined in Table 2. As a single sequence step in this data has only one input feature, the feature space is not rich enough to evaluate the efficiency of our approach. The baseline reported the best performance. While the performance of SubseqSVM approach is slightly inferior to the baseline and the state-of-the-art, enum$\mathcal{AF}$ performed poorly on this dataset.

In the case of the UA dataset, both enum$\mathcal{AF}$ and SubseqSVM took 24 hours approximately to train the model. In comparison, TildeCRF and StructSVM took 0.5 hours and 20 hours, respectively. On the KU data, enum$\mathcal{AF}$ took around 24 hours and SubseqSVM took approximately 1.5 hours to train the model. In comparison, TildeCRF and StructSVM took 10 minutes and 15 hours, respectively. We now present an analysis of the progression of results on UA data, using different categories of features we have experimented with.

	Micro avg.	Macro avg.
`tildeCRF`	56.22(±12.08)	35.36 (±6.55)
`StructSVM`	58.02 (±11.87)	35.00 (±05.24)
`enum`$\mathcal{AF}$	60.36 (±6.99)	30.39 (±4.31)
`SubseqSVM`	65.25(±4.81)	29.34 (±2.78)

Table 1: Micro average accuracy and macro average accuracy of classification in percentage using various approaches on UA data.

	Micro avg.	Macro avg.
`tildeCRF`	66.04 (±13.50)	84.01 (±8.76)
`StructSVM`	66.35 (±17.16)	66.64 (±16.04)
`enum`$\mathcal{AF}$	33.24 (±15.72)	23.02 (±11.13)
`SubseqSVM`	64.66 (±8.42)	63.08 (±7.05)

Table 2: Micro average accuracy and macro average accuracy of classification in percentage using various approaches on KU data.

The progression on experiments on UA data based on feature categories is shown in Table 3. The baseline for sequence labeling can be one among the approaches that assume conditional independence among individual features, given the label. HMM, CRF, and StructSVM falls into this category. These approaches consider input features at a sequence step and assumes conditional independence among them given the label. Since StructSVM is the state-of-the-art in this category, we use StructSVM results for comparison. The next level of features is the set of simple conjuncts $\mathcal{SC}$, which are conjunctions of input features at a single sequence step. $\mathcal{SC}$s capture relationships among co-occurring features. We present the StructHKL results for this. Next is the category of $\mathcal{CF}$s,

which are capable of capturing input relationships across time steps in sequence labeling. We present the results of SubseqSVM in this category.

Feature	Approach	Micro avg.	Macro avg.
Basic	StructSVM	58.02 (±11.87)	35.00 (±05.24)
$\mathcal{SC}$	StructHKL	63.96 (±05.74)	32.01 (±03.04)
$\mathcal{CF}$	SubseqSVM	65.25(±4.81)	29.34 (±2.78)

Table 3: Progression on sequence labeling experiments on the UA dataset based on feature categories.

5 Conclusion

Recent works have shown the importance of learning the input structure, in the form of relational features, for sequence labeling problems [**?**,**?**,**?**]. Most of the existing feature learning approaches employ greedy search techniques to discover relational features. In this work, we discussed approaches that looked into learning optimal relational features for sequence labeling. We identify that the relational feature space is exponentially large and therefore, learning explicit features of arbitrary complexity in our most general feature subspace, is a hard task. To this end, we presented an approach that learns relational sequence labeling models (capturing the richness of relational features implicitly) by leveraging relational subsequence kernels in the dual objective of the StructSVM framework. From our discussions and empirical analysis, we conclude that it is desirable to use powerful kernels that capture the relational features implicitly, although the resulting model may not be interpretable.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. of 20th Intl. Conf. on VLDB. pp. 487–499 (1994)
2. Bunescu, R., Mooney, R.J.: Subsequence kernels for relation extraction. In: Submitted to the Ninth Conference on Natural Language Learning (CoNLL-2005). Ann Arbor, MI (July 2006)
3. Dehaspe, L., Toivonen, H.: Discovery of frequent datalog patterns. Data Min. Knowl. Discov. 3(1), 7–36 (Mar 1999)
4. Forney, G.J.: The viterbi algorithm. Proceedings of IEEE 61(3), 268–278 (1973)
5. Gutmann, B., Kersting, K.: Tildecrf: conditional random fields for logical sequences. In: Proceedings of the 17th European conference on Machine Learning. pp. 174–185. ECML'06, Springer-Verlag, Berlin, Heidelberg (2006)
6. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural svms. Mach. Learn. 77(1), 27–59 (Oct 2009)
7. 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. pp. 1–9. UbiComp '08, ACM, New York, NY, USA (2008)

8. Kersting, K., Raedt, L.D., Raiko, T.: Logical hidden markov models. Journal of Artificial Intelligence Research 25, 2006 (2006)
9. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data (2001), iCML
10. Landwehr, N., Gutmann, B., Thon, I., Raedt, L.D., Philipose, M.: Relational transformation-based tagging for activity recognition. Progress on Multi-Relational Data Mining 89(1), 111–129 (2009)
11. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. J. Mach. Learn. Res. 2, 419–444 (Mar 2002)
12. Mauro, N.D., Basile, T.M.A., Ferilli, S., Esposito, F.: Feature construction for relational sequence learning (2010)
13. 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 - Volume 4. pp. 188–191. CONLL '03, Association for Computational Linguistics, Stroudsburg, PA, USA (2003)
14. McCallum, A.K.: Efficiently inducing features of conditional random fields (2003), proceedings of the Nineteenth Conference Annual Conference on Uncertainty in Artificial Intelligence
15. Miao, X., Rao, R.P.: Fast structured prediction using large margin sigmoid belief networks. Int. J. Comput. Vision 99(3), 302–318 (Sep 2012)
16. Nair, N., Nagesh, A., Ramakrishnan, G.: Probing the space of optimal markov logic networks for sequence labeling. In: Proceedings of the 22nd international conference on Inductive logic programming. Springer-Verlag, Berlin, Heidelberg (2012)
17. Nair, N., Ramakrishnan, G., Krishnaswamy, S.: Enhancing activity recognition in smart homes using feature induction. In: Proceedings of the 13th international conference on Data warehousing and knowledge discovery. pp. 406–418. DaWaK'11, Springer-Verlag, Berlin, Heidelberg (2011)
18. Nair, N., Saha, A., Ramakrishnan, G., Krishnaswamy, S.: Rule ensemble learning using hierarchical kernels in structured output spaces. In: AAAI (2012)
19. Rabiner, L.R.: Readings in speech recognition. chap. A tutorial on hidden Markov models and selected applications in speech recognition, pp. 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1990)
20. 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)
21. Taskar, B., Lacoste-Julien, S., Jordan, M.I.: Structured prediction, dual extragradient and bregman projections. J. Mach. Learn. Res. 7, 1627–1653 (Dec 2006)
22. Thon, I.: Don't fear optimality: sampling for probabilistic-logic sequence models. In: Proceedings of the 19th international conference on Inductive logic programming. pp. 226–233. ILP'09, Springer-Verlag, Berlin, Heidelberg (2010)
23. Thon, I., Landwehr, N., Raedt, L.: Stochastic relational processes: Efficient inference and applications. Mach. Learn. 82(2), 239–272 (Feb 2011)
24. 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. pp. 104–. ICML '04, ACM, New York, NY, USA (2004)