2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Learning Latent Structure for Activity Recognition*

Ninghang Hu[1], Gwenn Englebienne[1], Zhongyu Lou[1] and Ben Kröse[1,2]

***Abstract*— We present a novel latent discriminative model for human activity recognition. Unlike the approaches that require conditional independence assumptions, our model is very flexible in encoding the full connectivity among observations, latent states, and activity states. The model is able to capture richer class of contextual information in both state-state and observation-state pairs. Although loops are present in the model, we can consider the graphical model as a linear-chain structure, where the exact inference is tractable. Thereby the model is very efficient in both inference and learning. The parameters of the graphical model are learned with the Structured-Support Vector Machine (Structured-SVM). A data-driven approach is used to initialize the latent variables, thereby no hand labeling for the latent states is required. Experimental results on the CAD-120 benchmark dataset show that our model outperforms the state-of-the-art approach by over 5% in both precision and recall, while our model is more efficient in computation.**

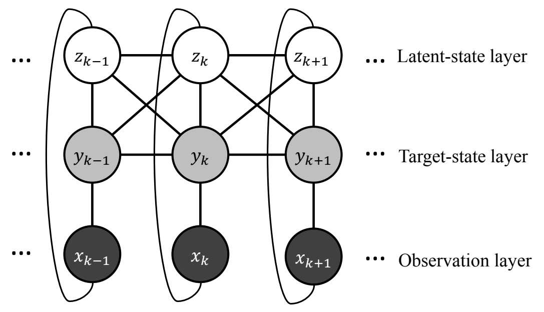


Fig. 1. The proposed graphical model. Nodes that represent the observations $\boldsymbol{x}$ are rendered in black, and they are observed both in training and testing. Grey nodes $\boldsymbol{y}$ are only observed during training but not testing, and they represent the target labels to be predicted, *e.g.* activity labels. White nodes $\boldsymbol{z}$ refer to the latent variables, which are unknown either in training or testing. Note that $\boldsymbol{x_k}$, $\boldsymbol{y_k}$, $\boldsymbol{z_k}$ are fully connected in our model, and also for nodes of transition.

I. Introduction

Robotic companions to help people in their daily life are currently a widely studied topic. In Human-Robot Interaction (HRI) it is very important that the human activities are recognized accurately and efficiently. In this paper, we present a novel graphical model for human activity recognition.

The task of activity recognition is to find the most likely underlying activity sequence based on the observations generated from the sensors. Typical sensors include ambient cameras, contact switches, thermometers, pressure sensors, and the sensors on the robot, *e.g.* RGB-D sensor and Laser Range Finder.

Probabilistic Graphical Models have been widely used for recognizing human activities in both robotics and smart home scenarios. The graphical models can be divided into two categories: generative models [1], [2] and discriminative models [3], [4], [5]. The generative models require making assumptions on both the correlation of data and on how the data is distributed given the activity state. The risk is that the assumptions may not reflect the true attributes of the data. The discriminative models, in contrast, only focus on modeling the posterior probability regardless of how the data are distributed. The robotic and smart environment scenarios are usually equipped with a combination of multiple sensors. Some of these sensors may be highly correlated, both in the temporal and spatial domain, *e.g.* a pressure sensor on the mattress and a motion sensor above the bed. In these scenarios, the discriminative models provide us a natural way of data fusion for human activity recognition.

The linear-chain Conditional Random Field (CRF) is one of the most popular discriminative models and has been used for many applications. Linear-chain CRFs are efficient models because the exact inference is tractable. However, they are limited in the way that they cannot capture the intermediate structures within the target states [6]. By adding an extra layer of latent variables, the model allows for more flexibility and therefore it can be used for modeling more complex data. The names of these models are interchangeable in the literature, such as Hidden-Unit CRF [7], Hidden-state CRF [6] or Hidden CRF [8].

In this paper, we present a latent CRF model for human activity recognition. For simplicity, we use *latent variables* to refer to the augmented hidden layer, as they are unknown either in training or testing. Intuitively, one can imagine that the latent variables represent subtypes of the activities. *e.g.* For the activity "opening", using latent variables we are able to model the difference between "opening a bottle" and "opening a door". The *target variables*, which is observed during training but not testing, represent the target states that we would like to predict, *e.g.* the activity labels. See Fig. 1 for the graphical model and the difference between latent variables and target variables. We evaluate the model using the RGB-D data from the benchmark dataset [3]. The results show that our model performs better than the state-of-the-art approach [3], while the model is more efficient in inference.

The contributions of this paper can be summarized as follows: We propose a novel Hidden CRF model for predicting underlying labels based on the sequential data. For each temporal segment, we exploit the full connectivity among

*The research has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 287624.

[1] N. Hu, G. Englebienne, Z. Lou and B. Kröse are with Intelligent System Lab Amsterdam, University of Amsterdam, 1098XH Amsterdam, The Netherlands {n.hu,g.englebienne,z.lou,b.j.a.krose} @ uva.nl
[2] B. Kröse is also with the Amsterdam University of Applied Science

observations, latent variables, and the target variables, from which we can avoid making inappropriate conditional independence assumptions. We show an efficient way of applying exact inference in our graph. By collapsing the latent states and the target states, our graphical model can be considered as a linear-chain structure. Applying exact inference under such a structure is very efficient. Our software is open source and will be fully available for comparison[1].

II. Related Work

Human activity recognition has been extensively studied in recent decades. Different types of graphical models have been applied to solve the problem, *e.g.* Hidden Markov Models (HMMs) [1], [2], Dynamic Bayesian Networks (DBNs) [9], linear-chain CRFs [10], loopy CRFs [3], Semi-Markov Models [4], and Hidden CRFs [11], [8].

As has been discussed in the introduction, the discriminative models are more suitable for data fusion tasks which are very common in HRI applications, where many different sensors are used. Here we focus on reviewing the most related work that uses discriminative models for activity recognition.

Recently Koppula et al. [3] presented a model for the temporal and spatial interactions between human and objects in loopy CRFs. More specifically, they built a model that has two types of nodes to represent sub-activity labels of the human and the object affordance labels of the objects. Human nodes and objects nodes within the same temporal segment are fully connected. Over time, the nodes are transited to the nodes with the same type. The results show that by modeling the human-object interaction, their model outperforms the earlier work in [2] and [12]. For inference in the loopy graph, they solve it as a quadratic optimization problem using the graph-cut method [13]. Their inference method, however, is less efficient compared with the exact inference in a linear-chain structure as the graph cut method takes multiple iterations before convergence, and usually more iterations are preferred to ensure of a good solution.

Other work [14] augments an additional layer of latent variables to the linear-chain CRFs. They explicitly model the new latent layer to represent the duration of activities. In contrast with [3], Tang et al. [14] solve the inference problem by reforming the graph into a set of cliques, so that the exact inference can be solved efficiently using dynamic programming. In their model, the latent variables and the observation are assumed to be conditionally independent given the target states.

Our work is different from the previous approaches in both the graphical model and the efficiency of inference. Firstly, similar to [14], our model also uses an extra latent layer. But instead of explicitly modeling what the latent variables are, we learn the latent variables directly from the data. Secondly, we do not make conditional independence assumptions between the latent variables and the observations. Instead, we add one extra edge between them to make the local graph fully connected. Thirdly, although our graph also presents a lot of loops as in [3], we are able to transform the cyclic graph into a linear-chain structure where the exact inference is tractable. The exact inference in our graph only needs two passes of messages across the linear chain structure which is much more efficient than [3]. Finally, we model the interaction between the human and the objects at the feature level, instead of modeling the object affordance as target states. In such a way, the parameters are learned to be directly optimized for activity recognition rather than making the joint estimation of both object affordance and the human activity. As we apply a data-driven approach to initialize the latent variables, hand labeling of the object affordance is not necessary in our model. Our results show that the model outperforms the state-of-the-art approaches on the CAD120 dataset [3].

[1]The source code will be available at https://github.com/ninghang/activity_recognition.git

III. Model

The graphical model of our proposed system is illustrated in Fig. 1. Let $\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K\}$ be the sequence of observations, where K is the total number of temporal segments in the video. Our goal is to predict the most likely underlying activity sequence $\boldsymbol{y} = \{y_1, y_2, \ldots, y_K\}$ based on the observations. We define $\boldsymbol{z} = \{z_1, z_2, \ldots, z_K\}$ to be the latent variables in the model. We assume there are N_y activities to be recognized and N_z latent states.

Each observation $\boldsymbol{x}_k$ itself is a feature vector within the segment k. The form of $\boldsymbol{x}_k$ is quite flexible. It can be collections of data from different sources, *e.g.* simple sensor readings, human locations, human pose, object locations. Some of these observations may be highly correlated with each other, *e.g.* the wearable accelerate meters and the motion sensors. Thanks to the discriminative nature of our model, we do not need to model such correlation among the observations.

A. Objective Function

Our model contains three types of potentials that in together form the objective function.

The first potential measures the score of seeing an observation $\boldsymbol{x}_k$ with a joint-state assignment (z_k, y_k). We define $\Phi(\boldsymbol{x}_k)$ to be the function that maps the input data into the feature space. $\boldsymbol{w}$ is the vector of parameters in our model.

$$\psi_1(y_k, z_k, \boldsymbol{x}_k; \boldsymbol{w}_1) = \boldsymbol{w}_1(y_k, z_k) \cdot \Phi(\boldsymbol{x}_k) \qquad (1)$$

This potential models the full connectivity among y_k, z_k and $\boldsymbol{x}_k$, avoiding making any conditional independence assumptions. It is more accurate to have such a structure since z_k and $\boldsymbol{x}_k$ may not be conditionally independent over a given y_k in many cases. To make it more intuitive, one could imagine that y_k refers to the activity drinking coffee and z_k defines the progress level of drinking. The activity drinking coffee starts with human grasping the coffee cup ($z_k = 1$), then drinking ($z_k = 2$), and then putting the cup back ($z_k = 3$). Knowing it is a drinking activity, the observation $\boldsymbol{x}_k$ varies largely over different progress level z_k.

The second potential measures the score of coupling y_k with z_k. It can be considered as either the bias entry of (1) or the prior of seeing the joint state (y_k, z_k).

$$\psi_2(y_k, z_k; \boldsymbol{w}_2) = \boldsymbol{w}_2(y_k, z_k) \tag{2}$$

The third potential characterizes the transition score from the joint state (y_{k-1}, z_{k-1}) to (y_k, z_k). Comparing with the normal transition potentials [8], our model leverages the latent variable z_k for modeling richer contextual information over consecutive temporal segments. Not only does our model contain the transition between states y_k, but it also captures the sub-level context using the latent variables. Intuitively, our model is able to capture the fact that the start of reading a newspaper is more likely to be preceded by the end of the drinking activity rather than the middle part of the drinking activity.

$$\psi_3(y_{k-1}, z_{k-1}, y_k, z_k; \boldsymbol{w}_3) = \boldsymbol{w}_3(y_{k-1}, z_{k-1}, y_k, z_k) \tag{3}$$

Summing all potentials over the whole sequence, we can write the objective function of our model as follows

$$\begin{aligned} F(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x}; \boldsymbol{w}) = &\sum_{k=1}^{K} \{\boldsymbol{w}_1(y_k, z_k) \cdot \Phi(\boldsymbol{x}_k) + \boldsymbol{w}_2(y_k, z_k)\} \\ &+ \sum_{k=2}^{K} \boldsymbol{w}_3(y_{k-1}, z_{k-1}, y_k, z_k) \end{aligned} \tag{4}$$

The objective function evaluates the matching score between the joint states $(\boldsymbol{y}, \boldsymbol{z})$ and the input $\boldsymbol{x}$. The score equals to the un-normalized joint probability in the log space. The objective function can be rewritten into a more general linear form $F(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w} \cdot \Psi(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x})$. Therefore the model is in the class of the log-linear model.

Note that it is not necessary to model the latent variables explicitly, but rather the latent variables can be learned automatically from the training data. Theoretically, the latent variables can represent any form of data, *e.g.* time duration, action primitives, as long as it can help with solving the task. Optimization of the latent model, however, may converge to a local minimum. The initialisation of the random variables is therefore of great importance. We compare three initialization strategies in this paper. Details of the latent variable initialization will be discussed in Section VI-D.

One may notice that our graphical model has many loops, which in general makes the exact inference intractable. Since our graph complies with the semi-Markov property, next, we will show that how we benefit from such a structure for efficient inference and learning.

IV. Inference

Given the graph and the parameters, the inference is to find the most likely joint states $\boldsymbol{y}$ and $\boldsymbol{z}$ that maximizes the objective function.

$$(\boldsymbol{y}^*, \boldsymbol{z}^*) = \underset{(\boldsymbol{y},\boldsymbol{z}) \in \mathcal{Y} \times \mathcal{Z}}{\operatorname{argmax}} F(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x}; \boldsymbol{w}) \tag{5}$$

Generally, solving (5) is an NP-hard problem that requires evaluating the objective function over an exponential number of state sequences. Exact inference is usually preferable as it is guaranteed to find the global optimum. However, the exact inference usually can only be applied efficiently when the graph is acyclic. In contrast, approximate inference is more suitable for loopy graphs, but may take longer to converge and is likely to find a local optimum. Although our graph contains loops, we show that we can transform the graph into a linear-chain structure, in which the exact inference becomes tractable. If we collapse the latent variable z_k with the activity state y_k into a single node, the edges between z_k and y_k become the internal factor of the new node and the transition edges collapse into a single transition edge. This results in a typical linear-chain CRF, where the cardinality of the new nodes is $N_y \times N_z$. In the linear-chain CRF, the exact inference can be performed efficiently using dynamic programming [15].

Using the chain property, we can write the following recursion for computing the maximal score over all possible assignments of $\boldsymbol{y}$ and $\boldsymbol{z}$.

$$\begin{aligned} V_k(y_k, z_k) = &\boldsymbol{w}_1(y_k, z_k) \cdot \phi(\boldsymbol{x}_k) + \boldsymbol{w}_2(y_k, z_k) \\ &+ \max_{(y_{k-1}, z_{k-1}) \in \mathcal{Y} \times \mathcal{Z}} \{\boldsymbol{w}_3(y_{k-1}, z_{k-1}, y_k, z_k) \\ &+ V_{k-1}(y_{k-1}, z_{k-1})\} \end{aligned} \tag{6}$$

Knowing the optimal assignment at K, we can track back the best assignment in the previous time step $K-1$. The process keeps going until all $\boldsymbol{y}^*$ and $\boldsymbol{z}^*$ have been assigned, *i.e.* the inference problem in (5) is solved.

Computing (6) once involves $O(N_y N_z)$ computations. In total, (6) needs to be evaluated for all possible assignments of (y_k, z_k), so that it is computed $N_y N_z$ times. The total computational cost is, therefore, $O(N_y^2 N_z^2 K)$. Such computation is manageable when $N_y N_z$ is not very large, which is usually the case for the tasks of activity recognition.

Next, we show how we can learn the parameters using the max-margin approach.

V. Learning

We use the max-margin approach for learning the parameters in our graphical model. The observation sequences and ground-truth activity labels are given during training $(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_N, \boldsymbol{y}_N)$. The latent variables $\boldsymbol{z}$ are unknown from the training data. The goal of learning is to find the parameters $\boldsymbol{w}$ that minimize the loss between the predicted activities and the ground-truth labels. A regularization term is used to avoid over-fitting.

$$\min_{\boldsymbol{w}} \left\{ \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{N} \Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}}) \right\} \tag{7}$$

where C is a normalization constant and $\Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}})$ measures the loss between the ground-truth and the prediction. The loss function returns zero when the prediction is the same as the ground-truth, and counts the number of disagreed elements otherwise. $\hat{\boldsymbol{y}}$ is the most likely activity sequence computed from (5) based on $\boldsymbol{x}_i$.

Optimizing (7) directly is not possible as the loss function involves computing the argmax in (5). Following [16] and [17], we substitute the loss function in (7) by the margin rescaling surrogate which serves as an upper-bound of the loss function.

$$\min_{\boldsymbol{w}}\{\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n} \max_{(\boldsymbol{y},\boldsymbol{z})\in\mathcal{Y}\times\mathcal{Z}}[\Delta(\boldsymbol{y}_i,\boldsymbol{y}) + F(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{z};\boldsymbol{w})] - C\sum_{i=1}^{n}\max_{\boldsymbol{z}\in\mathcal{Z}} F(\boldsymbol{x}_i,\boldsymbol{y}_i,\boldsymbol{z};\boldsymbol{w})\} \tag{8}$$

The second term in (8) can be solved using the augmented inference, *i.e.* by plugging in the loss function as an extra factor in the graph, the term can be solved in the same way as the inference problem using (5). Similarly, the third term of (8) can be solved by adding $\boldsymbol{y}_i$ as the evidence into the graph and then applying inference using (5). As the exact inference is tractable in our graphical model, both of the terms can be computed very efficiently.

Note that (8) is the summation of a convex and a concave function. This can be solved with the Concave-Convex Procedure (CCCP) [18]. By substituting the concave function with its tangent hyperplane function, which serves as an upper-bound of the concave function, the concave term is changed into a linear function. Thereby (8) becomes convex again.

We can rewrite (8) in the form of minimizing a function subject to a set of constraints by adding slack variables

$$\min_{\boldsymbol{w},\boldsymbol{\xi}}\left\{\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i\right\} \tag{9}$$
$$s.t.\ \forall i\in\{1,2,\ldots,n\},\forall \boldsymbol{y}\in\mathcal{Y}:$$
$$F(\boldsymbol{x}_i,\boldsymbol{y}_i;\boldsymbol{w}) - F(\boldsymbol{x}_i,\boldsymbol{y};\boldsymbol{w}) \geq \Delta(\boldsymbol{y}_i,\boldsymbol{y}) - \xi_i$$

Note that there are exponential number of constraints in (9). This can be solved by the cutting-plane method [19].

Another intuitive way to understand the CCCP algorithm is to consider it as solving the learning problem with incomplete data using Expectation-Maximization (EM) [20]. In our training data, the latent variables are not given. We can start by initializing the latent variables. Once we have the latent variables, the data become complete. Then we can use the standard Structured-SVM to learn the model parameters (M-step). After that, we can update the latent states again using the parameters that are learned (E-step). The iteration continues until convergence.

The CCCP algorithm decreases the objective function in every iteration. However, it cannot guarantee of finding the global optimum. To avoid of being trapped in the local minimum, the latent variables need to be carefully initialized. In this paper, we present three different initialization strategies, and details will be presented in Section VI-D.

Note that the inference algorithm is extensively used in learning. As we are able to compute the exact inference by transforming the loopy graph into a linear-chain graph, our learning algorithm is much faster and more accurate compared with the other approaches with approximate inference.

VI. Experiments

Our system is built upon three parts, the graphical model, the inference part and the learning part. We construct the graphical model and build the CCCP algorithm in Matlab. For exact inference, we adopt the inference engine from libDAI [21]. For learning, we take the Structured SVM framework provided by [16]. We compare the results with the state-of-the-art approach in [3].

A. Data

We evaluate our model on the CAD-120 dataset [3]. The dataset has 120 RGB-D videos with 4 subjects performing daily life activities. Each video is annotated with one high-level activity label and a sequence of sub-activity labels. The ground-truth of the segments and object affordance labels are also provided. In this paper, we use the sub-activity labels for evaluation. But our model can be easily extended into a hierarchical approach that can recognize higher-level activities, which will be reported in our next paper. As in [3], we use the ground-truth segments that are provided by the dataset.

For comparison, the same input features[2] are used as in [3]. The features are human skeleton features $\phi_a(\boldsymbol{x}_k)\in\mathbb{R}^{630}$, object features $\phi_o(\boldsymbol{x}_k)\in\mathbb{R}^{180}$, object-object interaction features $\phi_{oo}(\boldsymbol{x}_k)\in\mathbb{R}^{200}$, object-subject relation features $\phi_{oa}(\boldsymbol{x}_k)\in\mathbb{R}^{400}$, and the temporal objection and subject features $\phi_t(\boldsymbol{x}_k)\in\mathbb{R}^{200}$. These features are concatenated into a single feature vector, which is considered as the observation of one sub-activity segment, *i.e.* $\Phi(\boldsymbol{x}_k)$.

B. Evaluation Criteria

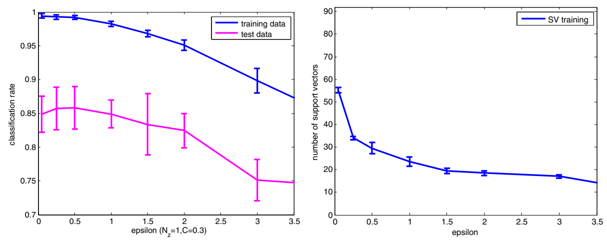
Our model is evaluated with 4-fold cross-validation. The folds are split based on the 4 subjects, *i.e.* the model is trained on videos of 3 persons and test on a *new person*. Each cross-validation is run for 3 times. To check the generalization of our model across different data, the results are averaged across the folds. In this paper, accuracy (classification rate), precision and recall are reported for comparing the results. In the CAD-120 dataset, more than half the instances are "reaching" and "moving". Therefore we consider precision and recall to be relatively better evaluation criteria than accuracy, as they remain meaningful despite class imbalance.

C. Baseline

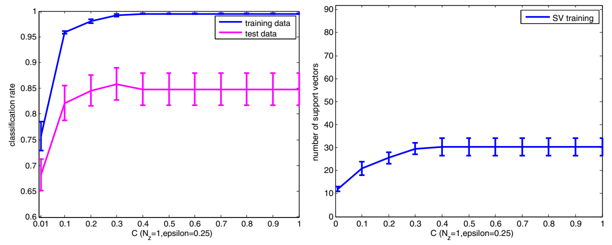
Our baseline approach uses only one latent state in our model ($N_z = 1$), which is equivalent to a linear-chain CRF. The parameters of the baseline model are learned with the standard Structured-SVM. We use the margin rescaling surrogate as the loss and L1-norm for the slacks. For optimization we use the 1-slack algorithm (primal) as being described in [22].

We apply a grid search for the best SVM parameters of C and ϵ. C is the normalization constant that is the trade-off between model complexity and classification loss. ϵ defines the stop threshold of optimization. When ϵ is small, the

[2]Input features can be downloaded from http://pr.cs.cornell.edu/humanactivities/data/features.tar



(a) $C = 0.30$, change ϵ



(b) $\epsilon = 0.25$, change C

Fig. 3. Another view of the grid search for the best C and ϵ. (a) shows the change of classification rate over ϵ when C is fixed to 0.3. When ϵ is small, a large number of support vectors is added and the model overfits. When ϵ is too large, the model is underfitting and the iterations stop too early, with too few support vectors. (b) shows the change of classification rate over C when ϵ is fixed to 0.25. When C is small, the learning algorithm tries to find a model as simple as possible, so that the performance is very low. When C is very large, the model overfits and the performance drops.

learning process takes longer time to converge and the trained model contains more support vectors. We show results of the grid search in Fig. 2. In Fig. 3 we show the curve of accuracy when keeping one of the parameters fixed.

Based on these results, we choose $C = 0.3$ and $\epsilon = 0.25$ for our experiments.

D. Initialize Latent Variables

In the our latent model, we choose the same C and ϵ as in the linear-chain CRF. Parameters of the model are initialized as zeros. To initialize the latent states, we adopt three different initialization strategies. a) Random initialization. b) A data-driven approach. We apply clustering on the input data $\boldsymbol{x}$. The number of clusters is set to be the same as the number of latent states. We run K-means for 10 times. Then we choose the best clustering results that with the minimal within-cluster distances. The labels of the clusters are assigned as the initial latent states. c) Object affordance. The object affordance labels are provided by the CAD120 dataset, which are used for training in [3]. We apply the K-means clustering upon the affordance labels. As the affordance labels are categorical, we use 1-of-N encoding to transform the affordance labels into binary values for clustering.

E. Results

Table I compares the activity recognition performance between our model and the state-of-the-art approach in [3]. We evaluate the model with different number of latent states, *i.e.* latent-2, latent-3 and latent-4, as well as the different initialization strategies, *i.e.* random, data-driven and affordance.

TABLE I
RESULTS OF ACTIVITY RECOGNITION

	Accuracy	Precision	Recall	F-score
Koppula, et al. [3]	86.0 ± 0.9	84.2 ± 1.3	76.9 ± 2.6	80.4 ± 1.5
latent-1 linear CRF	85.7 ± 2.9	86.4 ± 6.1	82.4 ± 4.0	82.6 ± 6.2
latent-2 random	84.0 ± 2.8	85.6 ± 4.6	79.5 ± 5.4	80.1 ± 6.5
latent-2 data-driven	**87.0 ± 1.9**	**89.2 ± 4.6**	**83.1 ± 2.4**	**84.3 ± 4.7**
latent-2 affordance	**87.0 ± 2.1**	**88.3 ± 4.3**	**84.0 ± 3.2**	**84.3 ± 5.1**
latent-3 random	83.1 ± 2.2	86.1 ± 4.5	76.3 ± 4.8	78.1 ± 6.1
latent-3 data-driven	86.0 ± 1.9	87.2 ± 2.9	82.3 ± 2.4	82.9 ± 4.2
latent-3 affordance	86.0 ± 2.0	88.0 ± 4.6	81.5 ± 3.4	82.1 ± 4.8
latent-4 random	82.8 ± 3.2	85.9 ± 5.0	76.3 ± 5.6	77.5 ± 6.9
latent-4 data-driven	85.9 ± 1.7	86.8 ± 2.7	82.4 ± 2.0	82.8 ± 3.7
latent-4 affordance	85.7 ± 1.6	86.4 ± 2.8	81.7 ± 2.9	82.0 ± 3.6

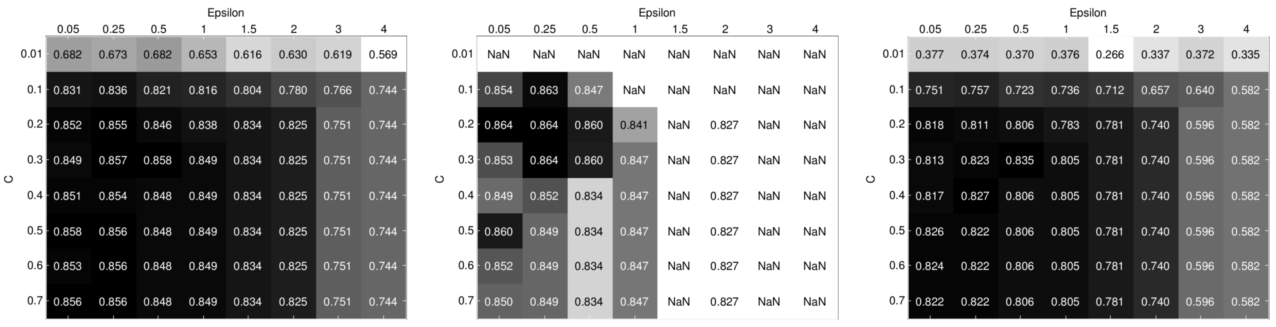
We show that with the optimal SVM parameters, the baseline performs better on the precision and recall compared with [3], but worse on the accuracy. This is because the baseline does not model the object affordance as target variables, and the parameters are optimized directly for minimizing the loss in activity recognition. The other reason is that the baseline model follows a linear-chain structure, and it is guaranteed to find the global optimal solution.

By adding the latent variables, our model can achieve better results than the baseline, but only when the latent variables are properly initialized. When the latent variables are randomly initialized, the average performance is much worse in most of the cases and shows a large variance as it most likely to have converged to a local minimum. We note that the data-driven initialization (clustering on $\boldsymbol{x}$) performs as good as the initialization with the hand-labeled object affordances.

We also compare the model when different numbers of latent states are used. We obtain better performance when we use only 2 latent states instead 3 or 4. This is partly because there are more parameters to be tuned when the model contains more latent states. The other reason is that the model may be too complex and overfits the data. Therefore choosing the number of latent states is also data related. If we use a more complex dataset, more latent states need to be used.

Fig. 4 shows the confusion matrix of activity classification. We can see that higher values present on the diagonal of the confusion matrix, and they represent the activities that are correctly classified. The most difficult classes are eating and scrubbing. Eating is sometimes confused with the drinking, and scrubbing is likely to be confused with reaching, drinking and placing.

Our best performance is obtained when we use 2 latent states and the model is initialized by clustering on the input data. We get 89.2% on the average precision and 83.1% on the average recall, which outperforms the state-of-the-art by over 5% on both precision and recall. We believe the performance can be further improved if we apply grid search for the optimal learning parameters of the latent-state model.



(a) Average Accuracy

C \ Epsilon	0.05	0.25	0.5	1	1.5	2	3	4
0.01	0.682	0.673	0.682	0.653	0.616	0.630	0.619	0.569
0.1	0.831	0.836	0.821	0.816	0.804	0.780	0.766	0.744
0.2	0.852	0.855	0.846	0.838	0.834	0.825	0.751	0.744
0.3	0.849	0.857	0.858	0.849	0.834	0.825	0.751	0.744
0.4	0.851	0.854	0.848	0.849	0.834	0.825	0.751	0.744
0.5	0.858	0.856	0.848	0.849	0.834	0.825	0.751	0.744
0.6	0.853	0.856	0.848	0.849	0.834	0.825	0.751	0.744
0.7	0.856	0.856	0.848	0.849	0.834	0.825	0.751	0.744

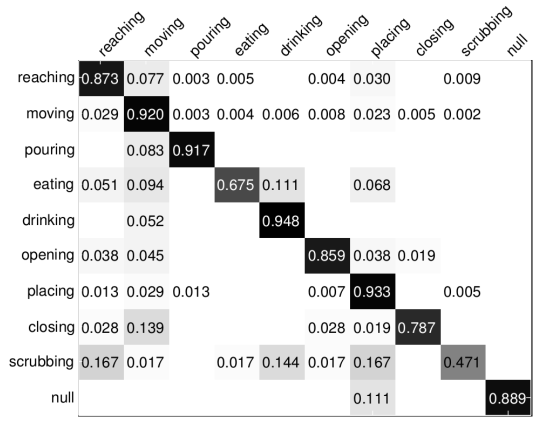
(b) Average Precision

C \ Epsilon	0.05	0.25	0.5	1	1.5	2	3	4
0.01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.1	0.854	0.863	0.847	NaN	NaN	NaN	NaN	NaN
0.2	0.864	0.864	0.860	0.841	NaN	0.827	NaN	NaN
0.3	0.853	0.864	0.860	0.847	NaN	0.827	NaN	NaN
0.4	0.849	0.852	0.834	0.847	NaN	0.827	NaN	NaN
0.5	0.860	0.849	0.834	0.847	NaN	0.827	NaN	NaN
0.6	0.852	0.849	0.834	0.847	NaN	0.827	NaN	NaN
0.7	0.850	0.849	0.834	0.847	NaN	0.827	NaN	NaN

(c) Average Recall

C \ Epsilon	0.05	0.25	0.5	1	1.5	2	3	4
0.01	0.377	0.374	0.370	0.376	0.266	0.337	0.372	0.335
0.1	0.751	0.757	0.723	0.736	0.712	0.657	0.640	0.582
0.2	0.818	0.811	0.806	0.783	0.781	0.740	0.596	0.582
0.3	0.813	0.823	0.835	0.805	0.781	0.740	0.596	0.582
0.4	0.817	0.827	0.806	0.805	0.781	0.740	0.596	0.582
0.5	0.826	0.822	0.806	0.805	0.781	0.740	0.596	0.582
0.6	0.824	0.822	0.806	0.805	0.781	0.740	0.596	0.582
0.7	0.822	0.822	0.806	0.805	0.781	0.740	0.596	0.582

Fig. 2. Performance of the baseline approach ($N_z = 1$). We apply a grid search to choose the best C and ϵ. The results are averaged on multiple runs of 4-fold cross-validation. The nan entry in (b) means that at least one of the classes gets no positive detection. Based on the grid search, we choose $C = 0.3$ and $\epsilon = 0.25$.



	reaching	moving	pouring	eating	drinking	opening	placing	closing	scrubbing	null
reaching	0.873	0.077	0.003	0.005		0.004	0.030		0.009	
moving	0.029	0.920	0.003	0.004	0.006	0.008	0.023	0.005	0.002	
pouring		0.083	0.917							
eating	0.051	0.094		0.675	0.111		0.068			
drinking		0.052			0.948					
opening	0.038	0.045				0.859	0.038	0.019		
placing	0.013	0.029	0.013			0.007	0.933		0.005	
closing	0.028	0.139				0.028	0.019	0.787		
scrubbing	0.167	0.017		0.017	0.144	0.017	0.167		0.471	
null							0.111			0.889

Fig. 4. Confusion matrix over different activity classes. Rows are ground-truth labels and columns are the detections. Each row is normalized to sum up to one, as one data object can only be associated with a single class label.

VII. Conclusion and Future Work

In this paper, we present a novel Hidden-state CRF model for human activity recognition. We use the latent variables to exploit the underlying structures of the target states. By making the observation and state nodes fully connected, the model do not require any conditional independence assumption between latent variables and the observations. The model is very efficient in that the inference algorithm is applied to a linear-chain structure. The results show that the proposed model outperforms the state-of-the-art approach. The model is very general that it can be easily extended for other prediction tasks on sequential data.

References

[1] C. Zhu and W. Sheng, "Human daily activity recognition in robot-assisted living using multi-sensor fusion," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2009, pp. 2154–2159.

[2] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from rgbd images," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 842–849.

[3] H. Koppula and A. Saxena, "Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation," *International Journal of Robotics Research (IJRR)*, 2013.

[4] T. van Kasteren, G. Englebienne, and B. J. Kröse, "Activity recognition using semi-markov models on real world smart home datasets," *Journal of Ambient Intelligence and Smart Environments*, vol. 2, no. 3, pp. 311–325, 2010.

[5] N. Hu, G. Englebienne, and B. Kröse, "Posture recognition with a top-view camera," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[6] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell, "Hidden conditional random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, vol. 29, no. 10, pp. 1848–1852, 2007.

[7] L. Maaten, M. Welling, and L. K. Saul, "Hidden-unit conditional random fields," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 479–488.

[8] Y. Wang and G. Mori, "Max-margin hidden conditional random fields for human action recognition," in *Proc. Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009, pp. 872–879.

[9] Y.-c. Ho, C.-h. Lu, I.-h. Chen, S.-s. Huang, C.-y. Wang, L.-c. Fu, *et al.*, "Active-learning assisted self-reconfigurable activity recognition in a dynamic environment," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2009, pp. 1567–1572.

[10] D. L. Vail, M. M. Veloso, and J. D. Lafferty, "Conditional random fields for activity recognition," in *Proc. International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM, 2007, p. 235.

[11] S. B. Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell, "Hidden conditional random fields for gesture recognition," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, vol. 2. IEEE, 2006, pp. 1521–1527.

[12] B. Ni, P. Moulin, and S. Yan, "Order-preserving sparse coding for sequence classification," in *Proc. European Conference on Computer Vision (ECCV)*. Springer, 2012, pp. 173–187.

[13] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer, "Optimizing binary mrfs via extended roof duality," in *Proc. Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2007, pp. 1–8.

[14] K. Tang, L. Fei-Fei, and D. Koller, "Learning latent temporal structure for complex event detection," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 1250–1257.

[15] R. Bellman, "Dynamic programming and lagrange multipliers," *The Bellman Continuum: A Collection of the Works of Richard E. Bellman*, p. 49, 1986.

[16] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," pp. 1453–1484, 2005.

[17] C.-N. Yu and T. Joachims, "Learning structural svms with latent variables," in *Proc. of International Conference on Machine Learning (ICML)*. ACM, 2009, pp. 1169–1176.

[18] A. L. Yuille and A. Rangarajan, "The concave-convex procedure (cccp)," *Advances in Neural Information Processing Systems (NIPS)*, vol. 2, pp. 1033–1040, 2002.

[19] J. E. Kelley, Jr, "The cutting-plane method for solving convex programs," *Journal of the Society for Industrial & Applied Mathematics*, vol. 8, no. 4, pp. 703–712, 1960.

[20] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*. John Wiley & Sons, 2007, vol. 382.

[21] J. M. Mooij, "libDAI: A free and open source C++ library for discrete approximate inference in graphical models," *Journal of Machine Learning Research*, vol. 11, pp. 2169–2173, Aug. 2010.

[22] T. Joachims, T. Finley, and C.-N. J. Yu, "Cutting-plane training of structural svms," *Machine Learning*, vol. 77, no. 1, pp. 27–59, 2009.