JMLR: Workshop and Conference Proceedings vol 52, 195-206, 2016 PGM 2016

On Stacking Probabilistic Temporal Models with Bidirectional Information Flow

Thomas Geier[1] THOMAS_GEIER@GMX.DE
Michael Glodek[2] MICHAEL@ZGUY.DE
Georg Layher[2] GEORG.LAYHER@UNI-ULM.DE
Heiko Neumann[2] HEIKO.NEUMANN@UNI-ULM.DE
Susanne Biundo[1] SUSANNE.BIUNDO@UNI-ULM.DE
Günther Palm[2] GUENTHER.PALM@UNI-ULM.DE
[1] *Institute of Artificial Intelligence*
Ulm University (Germany)
[2] *Institute of Neural Information Processing*
Ulm University (Germany)

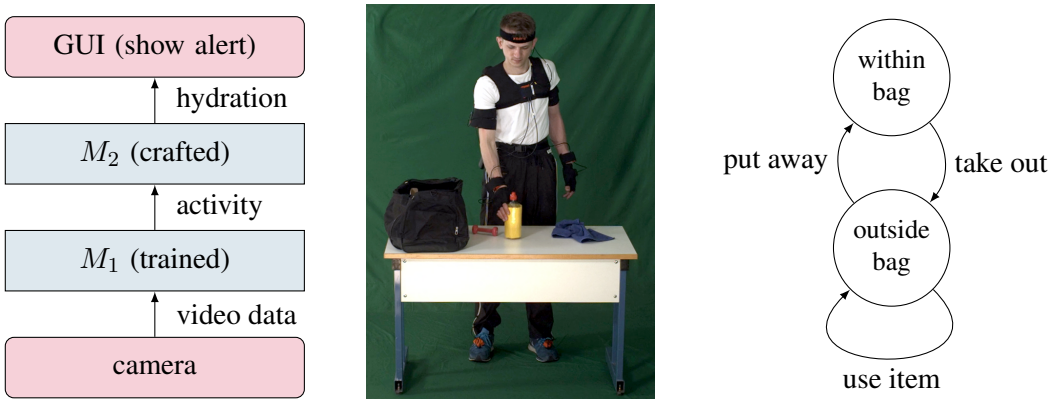
Abstract

We discuss hierarchical combinations of probabilistic models where the upper layer is crafted for predicting time-series data. The combination of models makes the naïve Bayes assumption, stating that the latent variables of the models are independent given the time-indexed label variables. In this setting an additional independence assumption between time steps and mildly inconsistent results are often accepted to make inference computationally feasible. We discuss how the application of approximate inference to the practically intractable joint model instead, shifts the need for these simplifications from model design time to inference time, and the application of loopy belief propagation to the joint model realizes bidirectional communication between models during inference. A first empirical evaluation of the proposed architecture on an activity recognition task demonstrates the benefits of the layered architecture and examines the effects of bidirectional information flow.

Keywords: graphical model; time-series; hidden Markov model; layered architecture; approximate inference; belief propagation; activity recognition.

1. Multi-Layer Models

Recently, multi-layered models are celebrating great successes in basically every application domain of machine learning under the name of *deep learning* (LeCun et al., 2015). In deep learning, models often are trained in a greedy, layer-wise way, where the next layer receives the output of the previous one. This stacking produces a hierarchy of models that represent increasingly abstract concepts found within the data. The training of early layers can be done unsupervised, i.e., without access to labelled data points. Then the aim of a layer is to transform a high-dimensional input into a lower-dimensional representation that preserves meaningful aspects and removes redundancy and noise. The absence of a teaching signal places no constraints on the representations found for the interface between layers, which are thus difficult to interpret. To obtain a defined output, higher layers are trained in a supervised way to predict the labels of the input data, e.g., the category of an object found in an image, or the word spoken in an audio sequence. Today there are many versions of this general approach differing in the details of the local learning algorithms and the theoretical



(a) example architecture (b) screenshot from data set (c) state machine for item dynamics

Figure 1: The images show a possible application architecture (a) that contains a trained model M_1 and a crafted model M_2. The application domain is monitoring the hydration level of a human user. Additionally a screen capture from the data set (b) and a state machine that describes item dynamics (c) are shown.

framework, which ranges from classical neural networks to restricted Boltzmann machines and Bayesian approaches (Bengio et al., 2012). In this paper we follow a probabilistic approach.

In this work we investigate the problem of connecting a layer (M_2) to the output of a preceding layer (M_1) that is learned in a supervised way. This has the consequence of M_1 producing an interpretable output. While this approach also requires labels for the intermediate representation, it has at least two advantages: First, it enables training M_1 and M_2 separately, obtaining the joint model M_{1+2} by a simple operation without further training. Secondly and more importantly, it allows the higher layer M_2 to be crafted. By crafted we mean that the model is mostly based on human expert knowledge, and requires little or no training data at all. As will be explained below, such an expert model is well suited for interfacing to other modules of a technical system, and it can also expose additional meaningful variables for which no labelled data is required. This construction is also quite common in complex practical applications where the variations on the data level are of a completely different nature than the variations on the categorical or 'symbolic' level.

Further, a joint probabilistic model that layers M_2 on top of M_1 supports a two-way communication between models. In addition to the obvious stream $M_1 \rightarrow M_2$, a downward stream $M_2 \rightarrow M_1$ can be realized that allows to use the expert knowledge found in M_2 to enhance the predictions produced by M_1. Such bidirectional communication between probabilistic models is naturally implemented using the approximate Belief Propagation (BP) inference algorithm.

As our motivation is the usage of partially crafted models in cognitive technical systems, we concentrate on a temporal setting, where both the data, and the target variables are time-indexed. For the probabilistic setting this means that the layers are (temporal) Markov models.

2. An Activity Recognition Task

An application of activity recognition, which is also used in the evaluation of section 7, will serve as an example throughout this paper. A human subject is performing physical training activities using a

dumbbell. He/she can also drink from a bottle of water and use a towel. The items are stored in a bag prior to the exercise session. They have to be taken out before use, and they can also be stored back inside the bag. We are interested in determining the current activity of the subject, which is one of taking item I out of the bag, using item I, or storing item I away, for $I \in \{\text{bottle}, \text{towel}, \text{dumbbell}\}$ for a total of nine different activities. A possible motivation for recognizing these activities is the goal to remind the subject of having a steady intake of water to avoid dehydration. The lower layer M_1 is trained on sequences of sensory data $\boldsymbol{O}$ (video or acceleration data) with the current activity L^t as label for each time step t. Under realistic circumstances, M_1 will yield far from perfect performance when predicting the current activity. The second model M_2 is crafted by a human expert and contains additional background knowledge about the domain. This simple application architecture is depicted in fig. 1(a). Please note that we focus on using the joint model M_{1+2} for improving the prediction of the current activity L^t. Inference of the hydration state serves only as a motivation.

A simple hand-crafted model M_2 can be implemented as a discrete-valued graphical model (either a dynamic Bayesian network (DBN) or a dynamic Markov random field (MRF)) that realizes the state machine depicted in fig. 1(c). The state machine captures the dynamics of taking items from a bag, and putting them back, while also enforcing that an item that is used must be outside of the bag. Since such constraints hold for each item separately and independently, these dynamics can easily be expressed by a crafted graphical model, while it might be difficult to learn them from data. A flat hidden Markov model (HMM) will have severe problems modelling this behavior, as the state space grows exponentially with the number of items; and even with a large enough hidden state the available training data might not suffice. An additional benefit of the crafted variant is the fact that the item locations can be inferred without them being labeled in the data-set. And access to those might be useful for interacting with a higher-level model, e.g., a user interface. We will now describe the process of combining two temporal probabilistic graphical models, after covering some preliminaries.

3. Notation and Basic Concepts

Random variables are represented by capital letters O, X, etc. The range of a random variable X is written $\text{Val}(X)$, values from the range are written as lower-case letters $x \in \text{Val}(X)$. We also use elements from the range of a random variable as assignments, and write x as shorthand for $X = x$. All notations in bold stand for concepts relating to sets of random variables ($\boldsymbol{O}, \boldsymbol{X}, \boldsymbol{o}, \boldsymbol{x}, \ldots$). We shorten quantification expressions by writing x for $x \in \text{Val}(X)$ below $\sum$ and $\prod$.

An MRF (undirected graphical model, or Markov network) defines a probability distribution over set of discrete-valued random variables $\boldsymbol{X}$ via a product of A-indexed local factors $\phi_a \colon \text{Val}(\boldsymbol{X}_a) \to \mathbb{R}^+$ for $a \in A$. Each factor ϕ_a depends only on a subset of variables $\boldsymbol{X}_a \subseteq \boldsymbol{X}$:

$$p(\boldsymbol{x}) = \frac{1}{Z} \prod_{a \in A} \phi_a(\boldsymbol{x}_a) \tag{1}$$

The value Z is called *partition function* and normalizes the distribution.

As the focus of this paper lies on the realization of cognitive processes for technical systems, we are dealing with temporal streams of data originating from sensors in real-time. To formalize this concept in a probabilistic setting, we predict a discrete-valued label sequence $\{\boldsymbol{L}^t : t \in \mathbb{N}\}$ based

on observations $\left\{\boldsymbol{O}^t : t \in \mathbb{N}\right\}$ for discrete time steps t. The task of calculating expectations of the current label L^t given all past observations $O^{1\ldots t}$ is called *filtering* (Koller and Friedman, 2009).

4. Layering of Probabilistic Models

Suppose that a model M_2 captures the constraints implied by the physical movement of the items out of and into the bag as captured by the state machine in fig. 1(c). The question we are trying to answer in this section is how to connect such a model M_2 to a low-level, data-driven model M_1, e.g., a HMM, in a way that supports the goals described in the previous section under the premise that both models are given as graphical models. For this case M_{1+2} can be obtained by multiplication of M_1 and M_2, although some care has to be taken: If both models are generative, their product $M_{1+2}(\boldsymbol{O}, \boldsymbol{L}) = M_1(\boldsymbol{O}, \boldsymbol{L}) \cdot M_2(\boldsymbol{L})$ double-counts the distribution over the labels $\boldsymbol{L}$. This problem can be avoided by making M_1 conditional on the labels: $M_{1+2}(\boldsymbol{O}, \boldsymbol{L}) = M_1(\boldsymbol{O} \mid \boldsymbol{L}) \cdot M_2(\boldsymbol{L})$.

Let the low-level model at time T be given as a distribution $M_1^T(O^{1\ldots T} \mid L^{1\ldots T})$ over the sequence of past observations $O^{1\ldots T}$ conditional on the label sequence $L^{1\ldots T}$. Then the high-level model can be interpreted as a prior over the label sequence $M_2^T(L^{1\ldots T})$, and the joint model M_{1+2}^T at time T is

$$M_{1+2}^T(O^{1\ldots T}, L^{1\ldots T}) = M_1^T(O^{1\ldots T} \mid L^{1\ldots T}) \cdot M_2^T(L^{1\ldots T}). \tag{2}$$

The joint model can then be used for prediction of the current label given past observations through

$$M_{1+2}^T(L^T \mid o^{1\ldots T}) \propto \sum_{l^{1\ldots T-1}} M_{1+2}^T(L^T, l^{1\ldots T-1}, o^{1\ldots T}), \tag{3}$$

where normalization is calculated over L^T. Summation over all past label sequences $l^{1\ldots T-1}$ is not tractable in general, and to make matters worse, both M_1 and M_2 usually have additional time-dependent latent states $H_1^{1\ldots T}, H_2^{1\ldots T}$. With latent states one obtains the following equation:

$$\begin{aligned} M_{1+2}^T(L^T \mid o^{1\ldots T}) \propto \sum_{l^{1\ldots T-1}} & \left(\sum_{h_1^{1\ldots T}} M_1^T(o^{1\ldots T}, h_1^{1\ldots T} \mid L^T, l^{1\ldots T-1}) \right) \\ & \cdot \left(\sum_{h_2^{1\ldots T}} M_2^T(h_2^{1\ldots T}, L^T, l^{1\ldots T-1}) \right) \end{aligned} \tag{4}$$

It is important to realize that joining models in this way makes a naïve independence assumption. The structure of M_{1+2} implies that their not-shared variables are independent of each other given the label variables. This is similar to the naïve Bayes assumption that can be made when fusing classifiers on the level of individual labels (Kuncheva, 2004). But even under this independence assumption, and with tractable models M_1 and M_2, the joint model M_{1+2} might not allow for exact evaluation. The following sections describes two common simplifications that address this problem.

4.1 Temporal Independence Assumption

A way to tackle the infeasibility of (4) is to approximate the M_1-likelihood of the observation $M_1^T(o^{1\dots T} \mid L^{1\dots T})$ using the product of the marginal likelihoods for every time step t:

$$\sum_{h_1^{1\dots T}} M_1^T(o^{1\dots T}, h_1^{1\dots T} \mid l^{1\dots T}) \approx \prod_{1\le t\le T} \sum_{h_1^{1\dots T}, \tilde{l}^{\{1\dots T\}\setminus t}} M_1^T(o^{1\dots T}, h_1^{1\dots T} \mid l^t, \tilde{l}^{\{1\dots T\}\setminus t}) \tag{5}$$

We call this approximation temporal independence assumption (TIA). If M_1 and M_2 allow tractable filtering, then (5) becomes also tractable by making TIA—as it only adds singleton factors to M_2.

4.2 Temporal Inconsistency

Usually another approximation is applied on top of TIA. Using (5), the marginal likelihoods $M_1^T(o^{1\dots T} \mid L^t)$ have to be computed on each new time step T *for every past time step* $t \le T$, which constitutes a quadratic effort in the length of the considered sequence, and is clearly not suitable for real-time applications. Thus a further approximation is made, where only the most recent likelihood $M_1^T(o^{1\dots T} \mid L^T)$ is computed on each time step, while reusing the old ones. This results in the following approximation, which we call temporal inconsistency (TI):

$$M_{1+2}^T(o^{1\dots T}, L^{1\dots T}) \approx M_2^T(L^{1\dots T}) \cdot \prod_{1\le t\le T} \sum_{h_1^{1\dots t}, l^{1\dots t-1}} M_1^t(o^{1\dots t}, h_1^{1\dots t} \mid L^t, l^{1\dots t-1}) \tag{6}$$

Work that makes the TIA in conjunction with TI when layering temporal probabilistic models is very common (Oliver et al., 2004; Glodek et al., 2011, 2014a).

5. Approximative Inference and Bidirectional Information Flow

Conceptually a two-way communication between a low-level model M_1 and a high-level model M_2 can improve the performance of the joint system. Within an architecture with unidirectional information flow, model M_1 processes the observations, and communicates information via the label variables to M_2 on each time step. Then M_2 uses this information to build the final estimate. Concerning the example of activity recognition, M_2 could rule out impossible sequences of activities/labels based on the state machine of fig. 1(c), which should clearly improve the estimate about the current activity. If information flow from M_2 to M_1 occurs, then M_1 might improve the information reported to M_2, because M_1 can already rule out certain sequences early, which might help keeping M_1's prediction on track. Another way of interpreting the downward flow of information is under the concept of *attention*: If M_2 reports to M_1 which regions of the belief space it considers more likely, then M_1 can focus resources there.

While there exist apparent benefits of a downward information flow, it can also lead to problems. If M_2 reports to M_1 which configurations of the labels it considers more likely, it must be avoided that M_1 simply shifts its own belief in the same direction. Otherwise a positive feedback loop is induced, where both models incite each other to firmly believe in configurations that were only slightly more probable initially. Interestingly the application of BP to (3) implements bidirectional message passing, and suppresses feedback loops at the same time. The observation that BP realizes bidirectional information propagation between parts of a dynamic MRF has been also been made by Wu et al. (2003) for the stochastic sequential BP algorithm.

5.1 Belief Propagation

BP is an inference algorithm for discrete-valued graphical models that computes marginal probabilities and the partition function (Koller and Friedman, 2009). The application of BP to problems with a cyclic graphical structure yields an approximation that works well in practise. BP can operate on the factor graph of any discrete-valued MRF as specified by (1). Let i, j be variable indices and a, b sets of variable indices, with ϕ_a being a factor with incident variables $\mathbf{X}_a$. Then BP iteratively optimizes the variable-to-factor messages μ_{ia} and factor-to-variable messages μ_{ai} for all factor scopes a and variables $i \in a$. Each message is a probability distribution over the according variable. The algorithm iteratively updates the message values according to the following equations, starting with random or uniform distributions:

$$\mu_{ia}(x_i) \propto \prod_{b \ni i, b \neq a} \mu_{bi}(x_i) \tag{7}$$

$$\mu_{ai}(x_i) \propto \sum_{\mathbf{x}_a \supseteq x_i} \phi_a(\mathbf{x}_a) \prod_{j \in a, j \neq i} \mu_{ja}(\mathbf{x}_a[j]) \tag{8}$$

Here $\mathbf{x}_a[j]$ is the restriction of the variable assignment $\mathbf{x}_a$ to the single variable x_j for $j \in a$. Given a set of message values, the marginal distribution for some variable X_i can be calculated by $b_i(x_i) \propto \prod_{a \ni i} \mu_{ai}(x_i)$. The updates are not guaranteed to converge to a fix point, and on convergence they yield only an approximation to the true marginals (Koller and Friedman, 2009).

When BP is applied to the joint model M_{1+2}, it passes information in both directions across the label variables $\boldsymbol{L}$. The message going upward from L^t represents the marginal belief of M_1 over the variable L^t, and the downward message that of M_2. The role of the upward message is similar to the marginal likelihoods appearing in (5). BP assumes independence between variables locally, so TIA is still made in a weaker sense, but temporal inconsistency appears to be fixed when using a single joint model, though this consideration will be revised in section 5.2

Applying BP to the joint problem mitigates the issue of positive feedback loops. When computing the factor-to-variable message μ_{ai} via (8), the information of the backward message μ_{ia} is excluded, and the same holds for the variable-to-factor messages in (7). This means that BP tries to compensate feedback locally, although it can creep back into the result over loops. Translated into the model perspective, we can make the observation that a model M_i should not incorporate the information it obtains from some other model M_j into the information it transmits back to M_j. M_i may only use the information from M_j to improve the results reported to M_j, e.g., by spending more computational resources on relevant regions of the belief space to reduce the *expected* error.

5.2 Belief Propagation in Temporal Models

BP is agnostic to the special structure of temporal Markov models. If the aim is to solve the filtering task, the model has to be unrolled from the first to the current time step *after every new observation*, which makes effort quadratic in sequence length. To fix this, a special update schedule can be employed that constrains updates to a temporal window around the current time step. With a constant size window, and if BP is only run for a fixed number of iterations, real-time behavior can be achieved. But even when BP converges in every window, the forward-messages will be frozen once they exit the window—preventing a global convergence of BP on the whole problem. In some sense this reimplements the temporal inconsistency of section 4.2, but inconsistency vanishes when increasing the extent of the window into the past.

6. A Concrete Architecture

In this section a possible architecture with concrete graphical models for M_1 and M_2 is presented that can be applied to the example task of activity recognition. The lower level is realized by a conditioned hidden Markov model (CHMM) that models a distribution $M_1(\boldsymbol{H}, \boldsymbol{O} \mid \boldsymbol{L})$ conditional on the labels. The higher level is using Markov logic networks (MLNs)—a logic-based first-order language that compiles to MRFs.

6.1 Conditional Hidden Markov Models

As previously discussed, model M_1 should be a graphical model that is conditional on the labeled variables. A possible model class that adheres to this form is the CHMM (Glodek et al., 2011, 2014a). The CHMM extends the classic HMM by an additional sequence of causal states that influence the selection of hidden states. The model probability distribution of an observed sequence $O^{1...T}$ given a causal sequence $L^{1...T}$ is defined by

$$p(O^{1...T} \mid L^{1...T}) = \sum_{h^{1...T}} p(h^1 \mid L^1) \cdot \prod_{2 \leq t \leq T} p(h^t \mid h^{t-1}, L^t) \cdot \prod_{1 \leq t \leq T} p(O^t \mid h^t). \tag{9}$$

The causal states are assumed to be independent from each other, which makes the CHMM an inadequate model for highly correlated causal sequences without an additional label prior. The directed graphical model of the CHMM is depicted as the lower part of fig. 2. Given a dataset containing causal states $\boldsymbol{l}$ and observations $\boldsymbol{o}$, the parameter learning can be performed with the help of the expectation-maximization (EM) algorithm analogously to the HMM.

6.2 Markov Logic

The high-level model M_2 can be any generative graphical model. We have chosen to implement M_2 using MLNs (Richardson and Domingos, 2006). MLNs are a class of first-order probabilistic models (Milch and Russell, 2007), which allow the concise specification of discrete-valued MRFs using a syntax that resembles first-order logical sentences with weights. MLNs have been used to build models in numerous domains: An approach for office activity recognition which gets supported by the context of manipulated objects was presented by Biswas et al. (2007), who use MLNs to model the context relying on multiple weak feature sources, i.e. object information, body pose and body movement. A surveillance system for an outdoor parking lot was proposed by Tran and Davis (2008). MLNs are also used in our previous work (Glodek et al., 2014a, 2011) that describes a layered architecture based on the layered HMM of Oliver and Horvitz (2005).

For this work, we have crafted two different variants of model M_2. The first variant (M_{potts}) only models an attractive dependency between two successive activities via a Potts interaction[1]:

$$M^T_{\text{potts}}(L^{1...T}) = \frac{1}{Z} \prod_{1 \leq t \leq T-1} \exp\left\{w \cdot 1[L^t = L^{t+1}]\right\} \tag{10}$$

The coupling parameter w has been found using the maximum likelihood method for MRFs with latent variables (Koller and Friedman, 2009). The second variant (M_{state}) extends M_{potts} by adding hard factors (either 0 or 1) that capture the state-machine of fig. 1(c) for every object.

1. The indicator function $1[\varphi]$ evaluates to one if φ is true, and to zero otherwise.

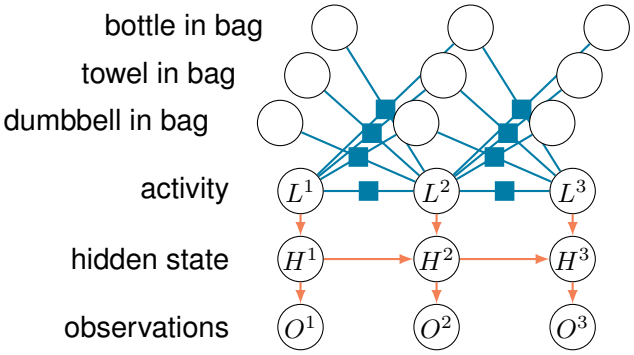


Figure 2: The MRF that is obtained via the combination of the CHMM (orange causal arrows) and the MLN model M_{state} (blue factor nodes) that encodes the state machine of fig. 1(c). Time progresses from left to right.

7. Empirical Evaluation

We have evaluated an instance of the described architecture on the example task of activity recognition using a CHMM as M_1 and both MLN variants M_{potts} and M_{state} as M_2. The goal of the evaluation is to demonstrate the value of adding a crafted layer to a model, and examine the effect of the bidirectional communication in contrast to just upward propagation.

7.1 The Data

The data-set consists of recordings of a subject standing behind a table with a bag containing a bottle of water, a towel and a dumbbell. Each item can be taken out of the bag, used and put back into the bag. A tenth no-op activity NN was labeled in extended pauses between activities. At the beginning all items are within the bag. The data-set consists of 62 sequences of 31 different human subjects performing a sequence of 10 activities with an average length of 104s (min/max/sd: 70/171/20). The subjects were recorded using a high-resolution camera, and an inertial motion capturing system. The recordings were taken in the same setting that is described in (Glodek et al., 2014b). Temporal resolution has been sub-sampled to 3hz for all further processing. The average number of activities per sequence is 34 (min/max/sd: 17/56/34). The length distribution for the different activities is roughly normal (mean/sd: 4.9s/1.8s) with the exception of the shorter NN (mean/sd: 0.9s/1.1s).

7.2 Training of Models

The features for the CHMM consist of classification results for the object in the right hand, and selected distances between joints of the skeleton model of the subject obtained from the motion capturing suite. To find the final CHMM model, the 62 sequences are used for developing, testing, training and validation with leave-one-sample-out cross-validation. The new sample rate represents a good compromise between information density and temporal progress. The multi-class ν-support vector machine (SVM) which recognizes the state of the hand, i.e., bare hand, dumbbell, bottle and towel, has been trained only on 30% of the available data. Sub-images are extracted using the projected location of the right hand. The SVM achieved an accuracy of 69.5% on the object

recognition task. The output of the multi-class SVM resembles a vector with four elements with object class memberships normalized to sum up to one. The CHMM has 65 hidden states which are shared over all actions and Gaussians with covariance matrices having only non-zero elements on the diagonal. An observation per time step is composed of a concatenated vector of Euclidean distances and angles between joints of the right body part as obtained from the motion capturing suite and the object class membership. Three distances are used, i.e. head and hand joints, the head and shoulder joints and the hand and hip joints. The angles are between the hand, elbow and shoulder joints and between the elbow, shoulder and neck joints.

The two MLN models were created by the authors and the single scalar parameter w of M_{potts} was found by likelihood maximization. A single rounded value $w = 4.32$ was used for the evaluation, as cross-validation showed only minimal sensitivity with respect to the different folds.

7.3 Inference

BP was applied for inference of the marginal probabilities $\boldsymbol{L}$ using the combined model M_{1+2}. We apply eight different message schedules to simulate unidirectional vs. bidirectional inference, and evaluate the effect of the window size, extending $p \in \{1, 5\}$ time steps into the past and $f \in \{0, 5\}$ time steps into the future around the current time. The marginal probabilities are extracted at appropriate times to prevent using future information beyond the extent of f. Note that the window has to extend over at least two time steps, so $p \geq 1$. Unidirectional inference is simulated by first computing a forward pass over the M_1-related variables, and then updating only the M_2-related messages when sliding the window. We used a maximum of 1000 iterations per window and some damping (0.2) to help convergence. BP converged for most configurations, where the ones with bidirectional inference, large window sizes, and the more expressive M_{state} took the most computational effort and sometimes did not converge.

7.4 Results

There are two sources of variance mostly unrelated to the goal of the evaluation. First, since the used frequency is rather high at 3hz, it is difficult to predict the exact transition point between the time-extended activities, even though the succession of activities is predicted correctly. And secondly, and closely related, the very short (less than 1s) and meaningless activity `NN` causes many wrong classifications which affect the reported error-probability. To mitigate the first issue, we present the edit distance between the predicted and the true sequence normalized to the length of the true sequence. For this calculation each activity segment is replaced by one instance of the activity, thus removing the temporal extent. This measure can become larger than one, if more errors than distinct activity segments are made. Note that, while both the error per time step and the edit distance assume similar values, a value of 0.5 for average error means a misclassification every second time step, while 0.5 for the edit distance means a misclassification every second activity (about every 15th time step). To mitigate the problem of misclassification of `NN` activities, both measures are also reported with `NN` ignored.

Figure 3 shows the four different quality statistics for the eight schedules applied to the joint model with both MLN variants. The most meaningful measure is the edit distance without `NN`. The results show that the more expressive MLN model M_{state} can improve over the Potts prior. Average edits per activity without `NN` go down from 1 to 0.5 for $p = 1, f = 0$, and from 0.5 to 0.3 for the largest window size $p = 5, f = 5$. Strangely the mean error shows an inverted behavior. The

Figure 3: This plot shows the mean quality of the activity prediction. Each facet shows results for the two variants of MLN model (along the x-axis), as well as the results for unidirectional and bidirectional filtering (color coded). The facet columns show different configurations for the active range of the BP updates per time step in the format $\langle p|f \rangle$. The facet rows show the different quality statistics as described within the text.

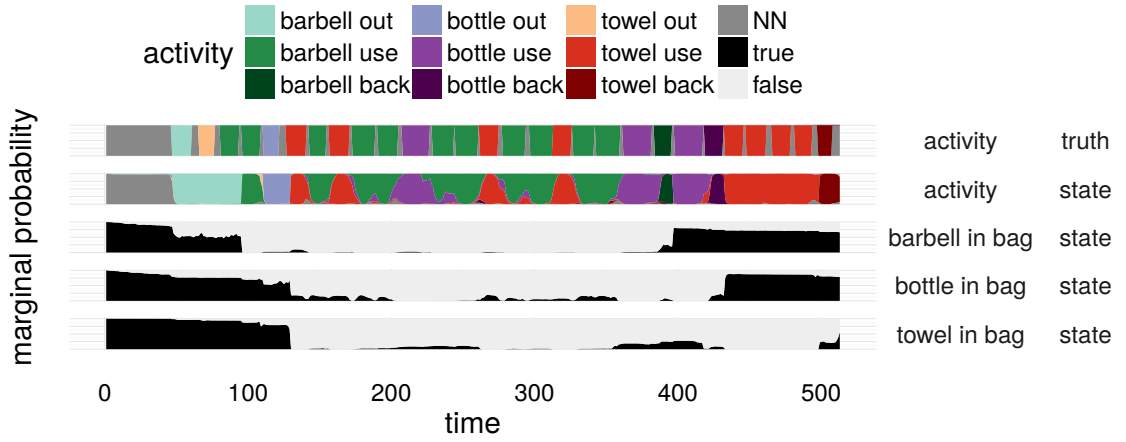


Figure 4: This plot shows marginal probabilities of the variables of the MLN model over time for one sequence. The top trace shows the ground truth for the current activity, while the second trace shows the estimated activity. The activity variable has nine possible values, while all other variables are binary.

presence of the following systematic error alone cannot explain the phenomenon: The transitions between activities appear delayed in the prediction, because it takes some steps until the lower model can convince the upper model to switch, violating M_{potts}. This effect decreases when the

look-ahead f increases. It can also be seen that increasing the window size in either direction improves the results.

Contrary to our expectation, the experiments show no statistically significant difference between unidirectional and bidirectional communication. While the edit distance without NN improves slightly under bidirectional communication in all configurations, the effect is minimal. Clearly, we had hoped for a stronger improvement in the bidirectional case, because unidirectional inference just prevents BP from convergence on certain message equations—and it can be expected that a totally converged result is superior. It would be interesting to investigate under which circumstances bidirectional inference yields improvements as compared to the unidirectional variant.

To demonstrate the inference of unlabeled variables, and to give a general impression of the results and the data-set, a sequence of marginal probabilities for all variables of M_{state} is shown in fig. 4. Note that the marginal beliefs show inconsistencies with respect to the used M_{state} model where the towel is used although the model thinks it is still within the bag due to missing the take-out activity. This is a consequence of restricting the BP updates to a time window. In experiments with full forward/backward propagation over the whole sequence (not shown) such problems are resolved using knowledge about the future.

8. Conclusion

We have discussed how to combine two temporal graphical models via shared variables, and identified two common approximations already made on the modelling level. When performing approximate inference on the intractable joint model instead of exact inference on the approximate model, the application of BP implements a bidirectional communication "in the correct way", i.e., avoiding positive feedback loops. While the experimental evaluation shows the benefit of combining a low-level model with a crafted model containing domain-knowledge, no significant improvement could be shown in the case of bidirectional vs. unidirectional inference. Because an improvement is to be expected theoretically, it remains as future work to characterise problems where bidirectional inference is clearly beneficial against those where it is not.

Acknowledgements

This research was conducted in the Transregional Collaborative Research Centre SFB/TRR 62 "A *Companion*-Technology for Cognitive Technical Systems", which is funded by the German Research Foundation (DFG).

References

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. abs/1206.5538, June 2012. URL http://arxiv.org/abs/1206.5538.

R. Biswas, S. Thrun, and K. Fujimura. Recognizing activities with multiple cues. In *Human Motion – Understanding, Modeling, Capture and Animation*, pages 255–270. Springer-Verlag, Springer, 2007. ISBN 3540757023. doi: 10.1007/978-3-540-75703-0_18.

M. Glodek, L. Bigalke, G. Palm, and F. Schwenker. Recognizing human activities using a layered HMM architecture. In B. Hammer and T. Villmann, editors, *Machine Learning Reports*, number 05, pages 38–41, 2011.

M. Glodek, T. Geier, S. Biundo, and G. Palm. A layered architecture for probabilistic complex pattern recognition to detect user preferences. *Biologically Inspired Cognitive Architectures*, 9: 46–56, jul 2014a. ISSN 2212-683X. doi: 10.1016/j.bica.2014.06.003. Neural-Symbolic Networks for Cognitive Capacities.

M. Glodek, G. Layher, F. Heilemann, F. Gawrilowicz, G. Palm, F. Schwenker, and H. Neumann. uulmmad–a human action recognition dataset for ground-truth evaluation and investigation of view invariances. In *Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction*, pages 77–91. Springer International Publishing, 2014b.

D. Koller and N. F. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. ISBN 9780262013192.

L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Blackwell, sep 2004. doi: 10.1002/9781118914564.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, may 2015. doi: 10.1038/nature14539.

B. Milch and S. Russell. First-order probabilistic languages: Into the unknown. In S. Muggleton, R. Otero, and A. Tamaddoni-Nezhad, editors, *Inductive Logic Programming*, volume 3, pages 10–24, Berlin, 2007. Springer. doi: 10.1007/978-3-540-73847-3_3.

N. Oliver and E. Horvitz. A comparison of HMMs and dynamic bayesian networks for recognizing office activities. In *User Modeling*, pages 199–209. Springer, 2005. doi: 10.1007/11527886_26.

N. Oliver, A. Garg, and E. Horvitz. Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, 96(2):163–180, nov 2004. doi: 10.1016/j.cviu.2004.02.004.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, jan 2006. ISSN 0885-6125. doi: 10.1007/s10994-006-5833-1.

S. Tran and L. Davis. Event modeling and recognition using Markov logic networks. *Computer Vision–ECCV 2008*, pages 610–623, 2008. doi: 10.1007/978-3-540-88688-4_45.

Y. Wu, G. Hua, and T. Yu. Tracking articulated body by dynamic Markov network. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1094–11012, Oct 2003. doi: 10.1109/ICCV.2003.1238471.