Incremental learning of action models as HMMs over qualitative trajectory representations

Maximilian Panzner and Philipp Cimiano

Semantic Computing Group, CITEC, Bielefeld University,
mpanzner@cit-ec.uni-bielefeld.de
http://www.sc.cit-ec.uni-bielefeld.de

Abstract. In this paper we present an incremental approach to learning generative models of object manipulation actions as HMMs over qualitative relations between two objects. We compare the incremental approach against a traditional batch training baseline and show that the resulting qualitative action models are capable of one-shot learning after just one seen example while displaying good generalization behavior as more data becomes available.

Keywords: hidden markov models, model merging, action recognition, online classification, qualitative trajectory calculus

1 Introduction

Acquiring representations or models of actions is important for many embodied intelligent systems that need to act in the world. We present a system which incrementally learns action models as Hidden Markov Models (HMMs) over qualitative relations between two objects. The incremental nature of the model building process allows the system to learn and adapt continuously. The resulting action specific models are capable of one–shot learning after just one seen example while displaying good generalization behavior as more and more data becomes available. There has been a lot of work in the field of intelligent systems on developing formalisms for learning and representing actions, ranging from task-space representations [1] to high-level symbolic description of actions and their effects on objects [2]. With our system we aim at bridging the gap between low-level representations and high-level object manipulation plans in a way that facilitates transfer of learned concepts between the representational levels. On the one hand, we abstract away from continuous task-space values such as joint angles and Euclidian positions. On the other hand, in contrast to purely symbolic descriptions like logic oriented action descriptions, we model actions as distributions and are thus able to capture variation in action performance. As an intermediate-level representation between two objects, we chose to build on the qualitative trajectory calculus as a formal foundation, which discretizes the relative position and movement of the objects into qualitative relations. The temporal progression of action instances is modeled using HMMs.

As we target applications in the field of human robot interaction, where the human tutors expect the robot to be responsive toward new tasks or stimuli, we adopt an incremental model merging scheme to estimate the HMM parameters on-line [3].

2 Method

In this approach action models are represented as Hidden Markov Models (HMM) over sequences of qualitative relations between a trajector and a landmark expressed in the Qualitative Trajectory Calculus (QTC). This model was already successfully applied to a co-development task using both action and linguistic cues to emerge structured, joint representations of action performances along with the grammatical structure of natural language sentences describing the respective action [4].

2.1 Qualitative action models

To describe the relative position and movement between landmark and trajector we build on the qualitative trajectory calculus - double cross (QTC_{C1}) [5] as a formal foundation. In general, QTC describes the interaction between two moving point objects k and l with respect to the reference line RL that connects them at a specific point t in time. The QTC framework defines 4 different subtypes as a combination over different basic relations between the two objects. As we only have one actively moved object in our experiments, we decided on QTC_{C1} to give the best trade off between generalization and specificity of the qualitative relations. QTC_{C1} consists of a 4-element state descriptor (C_1, C_2, C_3, C_4) where each $C_i \in \{-, 0, +\}$ represents a so called constraint with the following interpretation:

C_1 Distance constraint: Movement of k with respect to l at time t_1:
- $-$ k is moving towards l
- 0 k is not moving relative to l
- $+$ k is moving away from l

C_2 Distance constraint: Movement of l with respect to k at time t_1: same as above but with k and l interchanged

C_3 Side constraint: Movement of k with respect to RL at time t_1:
- $-$ k is moving to the left-hand side of RL
- 0 k is moving along RL or not moving at all
- $+$ k is moving to the right-hand side of RL

C_4 Side constraint: Movement of l with respect to RL at time t_1 analogously to C_3

As the positions in our dataset were sampled at a fixed rate, we could have missed some situations where one or more state descriptor elements transition through 0. These discretization artifacts are compensated by inserting the missing intermediate relations one at a time from left to right. QTC_{C1} is a rather

coarse discretization, leading to situations where the qualitative relation between the two objects can hold for a longer portion of the trajectory and is, due to the fixed rate sampling, repeated many times. Unlike many spatial reasoning systems, where repeating states are simply omitted, we use a logarithmic compression of repetitive subsequences:

$$|\hat{s}| = min(|s|, 10ln(|s| + 1)) \tag{1}$$

where $|s|$ is the original number of repeated symbols in the sequence and $|\hat{s}|$ is the new number of repeated symbols. By applying this compression scheme we preserve information about the acceleration along the trajectory, which increases the overall performance especially for very similar actions like "jumps over" and "jumps upon", while still allowing to generalize over high variations in relative pace of the action performances. The logarithmic compression of repetitive symbols in a sequence is in line with findings from psychophysics known as the Weber-Fechner law[6].

2.2 Learning qualitative action models

Differing from previous work [4], where we learned action models by batch training over all underlying trajectories, we now apply an incremental learning scheme utilizing the best first model merging [7,8] framework. Model merging is inspired by the observation that, when faced with new situations, humans and animals alike drive their learning process by first storing individual examples (memory based learning) when few data points are available and gradually switching to a parametric learning scheme to allow for better generalization as more and more data becomes available [9]. Our approach mimics this behavior by starting with simple models with just one underlying sequence, which evolve into more complex models generalizing over a variety of different sequences as more data becomes available.

The process to evolve simple models into complex ones relies on three basic operations. **Data incorporation** integrates a new observation sequence into an existing, possibly empty, model. **State merging** consolidates the resulting model in a way which allows it to generalize to yet unseen trajectories by intertwining paths corresponding to different action performances. **Model evaluation** approximates how well a given model fits its constituting dataset. This incremental learning scheme allows our models to display good generalization performance when faced with new samples while still being capable of one-shot learning after just one seen example. Learning, as generalization over the concrete observed examples, is driven by structure merging in the model in a way that we trade model likelihood against a bias towards simpler models, known as the Occam's Razor principle, which among equally well predicting hypothesis prefers the simplest explanation requiring the fewest assumptions. As graphical models, HMMs are particularly well suited for a model merging approach because data incorporation, state merging and model evaluation are straightforward to apply in this framework and implemented as graph manipulation operations:

Data incorporation: When a new sequence is to be integrated into a given model we construct a unique path between the initial and the final state of the model where each symbol in the sequence corresponds to a fresh state in the new path. Each of these states emits its respective symbol in the underlying sequence and simply transitions to the next state with probability 1, yielding a sub path in the model which exactly reproduces the corresponding sequence.
State merging: The conversion of the memory based learning scheme with unique sub paths for each sequence in the underlying dataset into a model which is able to generalize to a variety of similar trajectories is achieved by merging states which are similar according to their emission and transition densities. Merging two states q_1 and q_2 means replacing these states with a new state $\hat{q}$ whose transition and emission densities are a weighted mixture of the densities of the two underlying states.
Model evaluation: We evaluate the models resulting in the merging process using a mixture composed of a structural model prior $P(M)$ and the data dependent model likelihood $P(X|M)$:

$$P(M|X) = \lambda P(M) + (1-\lambda)P(X|M) \tag{2}$$

The model prior $P(M)$ acts as a data independent bias. Giving precedence to simpler models with fewer states makes this prior the primary driving force in the generalization process:

$$P(M) = e^{-|M|}, \tag{3}$$

where the model size $|M|$ is the number of states in the model. It is also possible to include the complexity of the transitions and emissions per state. For our dataset we found that using only the number of states generates the best performing models. While the structural prior favors simpler models, its antagonist, the model likelihood, has its maximum at the initial model with the maximum likelihood sub-paths. The exact likelihood of the dataset X given the model M is computed as:

$$P(X|M) = \prod_{x \in X} P(x|M) \tag{4}$$

with

$$P(x|M) = \sum_{q_1 \dots q_l \in Q^l} p(q_I \to q_1) p(q_1 \uparrow x_1) \dots \; p(q_l \uparrow x_l) p(q_l \to q_F) \tag{5}$$

where l is the length of the sample and q_I, q_F denote the initial and final states of the model. The probability to transition from a state q_1 to q_2 is given as $p(q_1 \to q_2)$ and $p(q_1 \uparrow x_1)$ denotes the probability to emit the symbol x_1 while being in state q_1. As we do not want to store the underlying samples explicitly, we use an approximation, which considers only the terms with the highest contribution, the Viterbi path:

$$P(X|M) \approx \prod_{q \in Q} \left(\prod_{q' \in Q} p(q \to q')^{c(q \to q')} \prod_{\sigma \in \Sigma} p(q \uparrow \sigma)^{c(q \uparrow \sigma)} \right) \tag{6}$$

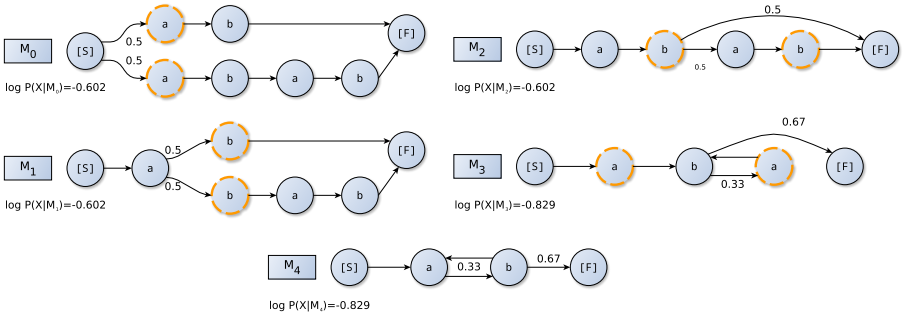

Fig. 1. Sequence of models obtained by merging samples from an exemplary language $(ab)^+$ and subsequently merging the highlighted states. Reproduced from [7].

where $c(q \to q\prime)$ and $c(q \uparrow \sigma)$ are the total counts of transitions and emissions occurring along the Viterbi path associated with the samples in the underlying dataset (see [7] for details).

The simplest model in our approach is a model which simply produces a single sequence. These models are called maximum likelihood models because they produce their respective sequences with the highest possible probability. Starting from maximum likelihood models over individual sequences we build more general HMMs by merging simpler ones and iteratively joining similar states to intertwine sub-paths constructed from different sequences, allowing them to generalize across different instances of the same action class. The first model M_0 of the example in figure 1 can be seen as a joint model of two maximum likelihood sequences $\{ab, abab\}$. When generating from such a model, the actual sequence which will be generated is determined early by taking one of the possible paths emanating from the start state. Only the transitions from the start state display stochastic behavior, the individual sub-paths are completely deterministic and generate either ab or $abab$. Intertwining these paths is done trough state merging, where we first build a list of possible merge candidates using a measure of similarity between state emissions and transition probability densities. In this approach we use the symmetrized Kullback-Leibler (KL) divergence. Then we greedily merge the best pair of states and re-evaluate the model likelihood. In the example above, the first two merges lead to model M_3 where we experienced a drop in log likelihood from -0.602 to -0.829. We continue the merging process until we reach a point where merging more states would deteriorate the model likelihood to a point where it is no longer compensated by the prior favoring simpler models (eq. 3). The final model M_4 is now able to generate the whole set of sequences from the exemplary language $(ab)^+$ the two initial samples where drawn from.

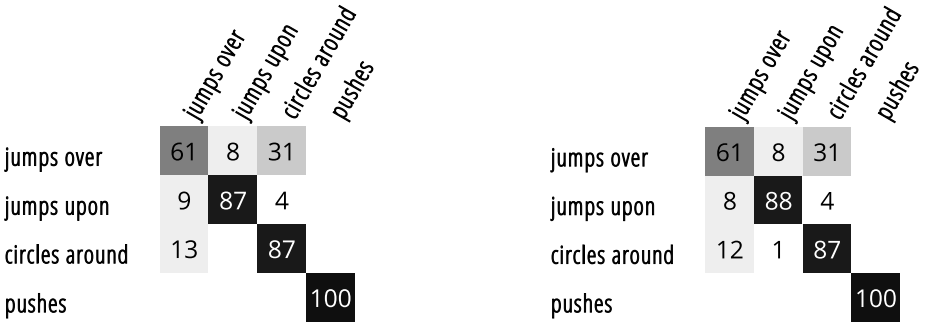

Fig. 2. Averaged class confusion matrix for the batch (left) and the incremental (right) approach.

3 Experiments

3.1 Dataset

To acquire a dataset we implemented a simple game where the test subjects where asked to perform an action with two objects according to a given instruction. The game screen was divided into two parts. The upper part was the actual gamefield with the two freely movable objects and below the gamefield was a textfield, where the test subjects could see the instruction describing the desired action performance. We had a total of 12 test subjects yielding a dataset with 1200 sample trajectories balanced over the four action classes "jumps over", "jumps upon", "circles around" and "pushes". See [4] for a complete description of the dataset.

3.2 Batch vs. Incremental

To validate the incremental model building scheme we evaluate the performance of the incremental process against a traditional Baum Welch parameter estimation baseline. We consider a setting where the system is trained using recorded action performances from 11 test subjects and is then presented with a 12th set of action trajectories recorded from a new performer. Following this scheme the dataset is partitioned into 12 folds with 1100 training examples and 100 test

	F_1	precision	recall	σ
Batch	0.86	0.82	0.90	0.11
Incremental	0.86	0.82	0.90	0.12

Table 1. Results of the 12-fold cross-validation for the batch and the incremental approach.

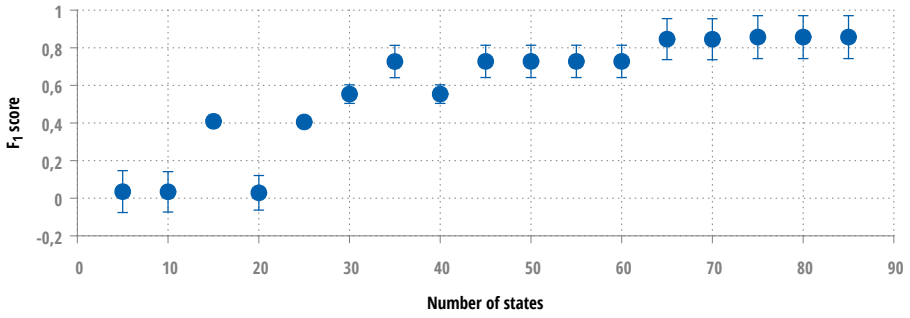

Fig. 3. Development of F_1 scores as the number of hidden states is increased for the batch training approach. The errorbars indicate the standard deviation of the F_1 scores across the 12 folds.

examples for each fold. We trained one HMM for each of the four action classes. Test sequences where classified according to the action specific HMM having the highest probability to have produced the respective sequence. Both approaches showed almost identical results (see table 1) with both having an averaged F_1 score of 0.86. The only difference is a slightly higher standard deviation between the results of the 12 folds for the incremental approach. The class confusion matrices in figure 2 show again only slight differences for both approaches. A noteworthy observation here is that the class confusions are not symmetric.

3.3 Parameter Sensitivity

In this experiment we evaluate the sensitivity of the batch and the incremental approach to their respective free parameters. As can be seen in figure 3 the classical Baum-Welch approach to HMM parameter estimation is highly sensitive to its structural parameter, the number of hidden states. F_1 scores range from 0.03 to 0.86 as the number of hidden states in the model is increased. The incremental approach on the other hand displays an almost linear response over

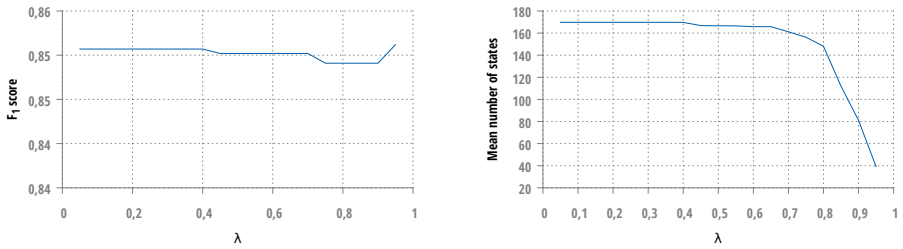

Fig. 4. Sensitivity to the λ parameter of the incremental approach.

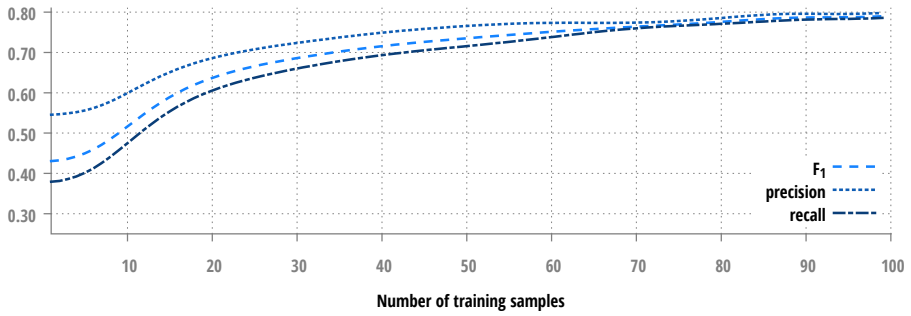

Fig. 5. Early learning behavior of the incremental model. The model performs notably well even with a single training example.

the whole range of values for the λ parameter (equation 2). While the F_1 score is stable with respect to λ, the number of hidden states in the model decreases drastically from 170 to 39 (at $\lambda = 0.95$). Setting $\lambda = 1$ and thus evaluating the model quality only by the prior favoring simpler models with less states leads to poorly performing models with just a single hidden state.

3.4 Early learning behavior

In this experiment we evaluate the one–shot learning capability of the incremental model. We prepared training sets with only 1 to 100 examples per action class and evaluated the classification performance as in section 3.2. As can be seen in figure 5 the precision is with 54% already notably good after the classifier had seen only one training example per action class.

3.5 Early classification behavior

To evaluate how well the system performs when it is presented with incomplete sequences, we trained the system as in section 3.2 but truncated the test sequences. As can be seen in figure 3.5, the sequences for the push action are rather distinctive. After the classifier has seen only 5% of the sequence, over 70% of the respective sequences are correctly classified as belonging to that action class. The “circles around” and “jumps over” actions get classified rather late in the sequence because a jumping trajectory could easily look like the start of a circles around action.

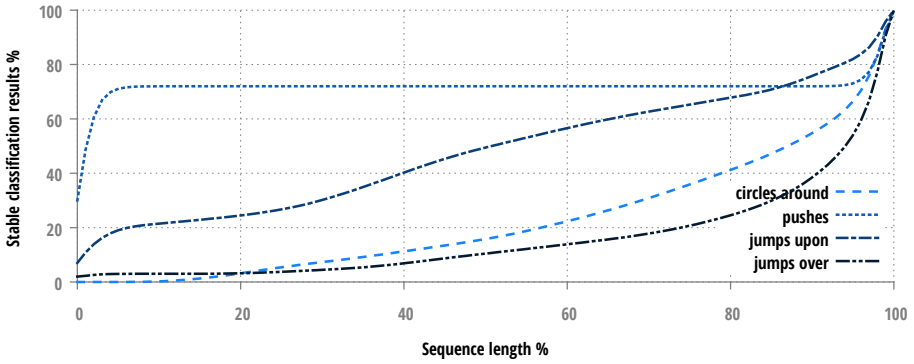

Fig. 6. Early sequence classification behavior. The x-axis represents the length of the sequence presented to the classifier and the y-axis the percentage of sequences for which their classification result will not change if more of the sequence was presented.

References

1. Komei Sugiura, Naoto Iwahashi, Hideki Kashioka, and Satoshi Nakamura. Learning, generation and recognition of motions by reference-point-dependent probabilistic models. *Advanced Robotics*, 25(6-7):825–848, 2011.
2. Moritz Tenorth and Michael Beetz. KnowRob – Knowledge Processing for Autonomous Personal Robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4261–4266, 2009.
3. Andreas Stolcke and Stephen Omohundro. Hidden markov model induction by bayesian model merging. *Advances in neural information processing systems*, pages 11–11, 1993.
4. Maximilian Panzner, Judith Gaspers, and Philipp Cimiano. Learning linguistic constructions grounded in qualitative action models. IEEE International Symposium on Robot and Human Interactive Communication, 2015.
5. Nico Weghe, Bart Kuijpers, Peter Bogaert, and Philippe Maeyer. A Qualitative Trajectory Calculus and the Composition of Its Relations. *GeoSpatial Semantics SE - 5*, 3799(Dc):60–76, 2005.
6. Thomas Bruss and Ludger Rüschendorf. On the perception of time. *Gerontology*, 56(4):361–370, 2010.
7. Stephen Omohundro. Best-first model merging for dynamic learning and recognition. In *Advances in Neural Information Processing Systems 4*, pages 958–965. Morgan Kaufmann, 1992.
8. Andreas Stolcke and Stephen Omohundro. Inducing probabilistic grammars by bayesian model merging. In *Grammatical inference and applications*, pages 106–118. Springer, 1994.
9. Roger N Shepard. Toward a universal law of generalization for psychological science. *Science*, 237(4820):1317–1323, 1987.

This research/work was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).