What's going on?
Discovering Spatio-Temporal Dependencies in Dynamic Scenes

Daniel Kuettel, Michael D. Breitenstein, Luc Van Gool, Vittorio Ferrari
Computer Vision Laboratory, ETH Zurich
<lastname>@vision.ee.ethz.ch

Abstract

We present two novel methods to automatically learn spatio-temporal dependencies of moving agents in complex dynamic scenes. They allow to discover temporal rules, such as the right of way between different lanes or typical traffic light sequences. To extract them, sequences of activities need to be learned. While the first method extracts rules based on a learned topic model, the second model called DDP-HMM jointly learns co-occurring activities and their time dependencies. To this end we employ Dependent Dirichlet Processes to learn an arbitrary number of infinite Hidden Markov Models. In contrast to previous work, we build on state-of-the-art topic models that allow to automatically infer all parameters such as the optimal number of HMMs necessary to explain the rules governing a scene. The models are trained offline by Gibbs Sampling using unlabeled training data.

1. Introduction

In this work, we address scene understanding and automatic behaviour mining in video footage from one static camera. Given a video of a dynamic scene, showing several different simultaneous activities interacting according to complex dependency patterns (Fig. 1), we want a system to automatically answer questions such as: "What are the typical actions in the scene? How do they relate to each other? What are the rules governing the scene?".

This is a challenging problem because both spatial and temporal dependencies between moving agents are relevant. While many researchers have focused on modeling isolated, independent behaviour of individual agents by analyzing their trajectories [1, 6, 9, 10], others have concentrated more recently on approaches based directly on the motion between consecutive frames [5, 7, 12, 13, 15]. This allows to model correlated behaviors of multiple agents. However, most methods still concentrate either on extracting spatially co-occurring motion patterns (*e.g.*, using topic models from document analysis [7, 12]), neglecting temporal dependencies, or on finding temporal dependencies only for previously defined events in a segmented scene (*e.g.*, using variants of Hidden Markov Models [4, 13]). Only very recently, Hospedales *et al*. [5] try to learn spatio-temporal patterns and dependencies jointly in a combined hierarchical model. However, they assume the whole scene is governed by only one Markov chain, which is problematic for complex scenes. Furthermore, they use a rather simple topic model that has several drawbacks (see Sec. 1.1).

(a) The optical flow from a scene. (b) Rules governing the scene.

Figure 1: A typical image from a dynamic scene with many agents. Many different activities with complex dependencies are possible. Our algorithm automatically finds sequences of co-occurring activities and the rules governing the scene.

In this paper, we propose two novel methods to extract spatio-temporal dependencies of moving agents in complex dynamic scenes. First, we present a method that builds on a state-of-the-art topic model (Hierarchical Dirichlet Processes [11]) and automatically learns dependencies between the motion patterns it extracts (*i.e*. *sequences of activities*). This allows to discover *local temporal rules* of the scene. As a second method, we present a novel model called DDP-HMM to *jointly* learn co-occurring activities and their time dependencies, enabling to discover *global temporal rules*. Using Dependent Dirichlet Processes, we learn an arbitrary number of Hidden Markov Models with an arbitrary number of states each. At the same time, the states (*i.e*. activities) of the HMMs are also learned, so that the mixture of activities from the different HMMs optimally explains the sequence of observations. When learned jointly, the resulting activities differ because they are forced to represent the movement in the scene rather than instantaneous states of the scene. We derive a Gibbs sampler for learning the joint model in an offline batch process.

We show experimentally on two datasets that our method

can extract spatio-temporal scene rules that answer queries like those described above. The first dataset has two videos of three hours each captured in Zurich. The second dataset has two shorter videos of London [5]. We qualitatively compare to the results of Hospedales *et al*. [5]. To our knowledge there is no standardized dataset available for activity mining, which is suitable for our setting. Thus, to encourage further work on this topic, we release matlab code for our method, along with the input videos and the exact flow features we used, on http://www.vision.ee.ethz.ch/~calvin/.

As a potential application, after training a model offline, it can be used to interpret new video data of the same scene in real-time (*e.g*. to explain what is currently going on and what will probably happen next) and to generate a textual description of the activities in the video. Another application is to trigger an alarm when a certain activity is observed or in the case of an unexpected sequence of activities.

1.1. Related Work

The two works most closely related to ours are the recently published methods from Wang *et al*. [12] and Hospedales *et al*. [5]. Both divide a video sequence into short clips (documents) and quantize moving pixels based on their direction (words) to learn a hierarchical topic model.

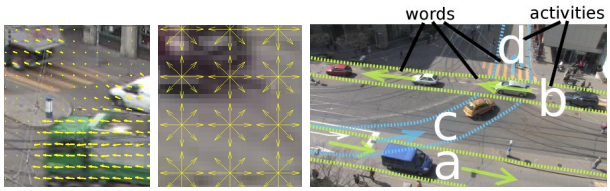
In contrast to us, [5] use Latent Dirichlet Allocation (LDA) [3] to model topics, which is less powerful than HDP as the number of topics is fixed manually beforehand. However, it is not obvious how to choose the optimal number of internal states necessary to represent everything that is going on in a complex scene. Furthermore, using LDA, the documents are not forced to share topics.[1] Thus, it is possible that individual activities are learned for many clips, making it very difficult to find repetitive sequences and temporal rules. Finally, their model only allows to find one Markov chain. This assumes that the scene is governed by one simple global rule.

We adopt the more powerful HDP model that is also used in [12]. Different to us, [12] extend HDP to Dual-HDP, which allows to additionally recover clusters of documents. However, their model only finds *instantaneous* rules (*e.g*., cars can simultaneously drive on two different lanes). In contrast, our model also finds *temporal* rules, for which *sequences* of documents are learned.

2. Model

Given an input video, we extract optical flow in each pair of consecutive frames using [14] (Figure 2(a)). First, the optical flow is thresholded to remove noise. The remaining optical flow vectors are then quantized using a *codebook*. The words $w = (x, y, u, v)$ of the codebook represent displacements (u, v), quantized into 8 directions, at positions (x, y) arranged on a grid with a spacing of 10 pixels (Fig. 2(b)). The video is divided into a sequence of 3-second clips. Each clip is represented by the words accumulated over its frames.

[1] *I.e.*, the mixture models in the different documents don't necessarily share mixture components [11].



(a) Optical flow. (b) Codebook. (c) Activities and temporal rules.

Figure 2: The optical flow (*a*) is quantized to build the codebook (*b*). (*c*) Activities a–d are represented by co-occurring words. A scene consists of several simultaneous sequences of co-occurring activities. Our method finds these temporal rules (*e.g*., "When c happens, d will follow. But if b is also active, d is deferred.").

We model *activities* as flow words co-occurring in the same clip (Fig. 2(c)). As these happen at different image positions, they form spatial *flow patterns*. More precisely, an activity is a distribution over flow words and thus independent of time, as the temporal ordering within a clip is ignored.

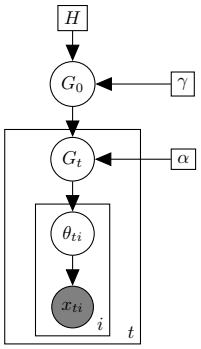
In the following subsections, we first recapitulate *Hierarchical Dirichlet Processes* (HDP) to model activities from video. Then we present the novel contributions of this paper: two methods for discovering temporal dependencies between activities. The first method (Sec. 2.1) first learns activities using HDP, and then finds dependencies between them in a second stage (*rules*). The second method consists of a new model combining HDP and infinite HMMs to *jointly* learn activities and their temporal dependencies.

The two methods discover complementary kinds of temporal dependencies. The two-stage approach finds rules of the scene that are spatially localized (*e.g*. rules between different lanes, Fig. 1(b)). The joint approach finds global temporal dependencies. It captures the state of the scene as a whole, and explains how it changes over time, which global state transitions are possible, and how likely they are. Therefore, both proposed approaches are appropriate, but for different levels of scene analysis.

2.1. Learning Activities using HDP

We model activities using the generative model in Figure 3. G_0 is the global list of activities, shared by all clips. For each clip t a subset of activities $G_t \subset G_0$ is active, based on which flow words are repeatedly generated by first drawing one activity θ_{ti} and then a word x_{ti}.

Mixture of Activities. As mentioned above, activities are represented by flow patterns θ. A flow pattern is a multinomial distribution over the flow words in the codebook. Typical flow patterns are caused by a car lane or a tram lane (Fig. 2(c)). Therefore, a clip t is explained by the mixture G_t of the activities (MoA) observed during t. A MoA G is



global list of activities

activities in clip t (mixture of flow patterns)

one activity (flow pattern)

one word (discretized flow)

Figure 3: Hierarchical Dirichlet Processes. Two Dirichlet Processes are employed. First, to genera ate the global list of activities. Second, to choose a subset of activities for a clip. Finally, words are generated from topics.

a multinomial distribution over flow patterns θ.

Dirichlet Processes. It is difficult to manually specify the activities θ or to fix their total number beforehand. Instead, here we infer both from data using a Hierarchical Dirichlet Process (HDP). HDP uses a *Dirichlet process (DP)* on two levels (Fig. 3). In our model, the first DP generates the global list of activities G_0. The second DP generates the different G_t, which are subsets of activities used to generate the clip t.[2] Thus, activities from the global set G_0 will be shared among different G_t. This allows the different clips to "share statistical strength".[3]

A sample from a DP $G_0 \sim DP(\gamma, H)$ with base distribution H and concentration parameter γ can be formulated using the stick-breaking construction [11]:

$$\begin{array}{ll} \pi'_k|\gamma, H \sim Beta(1,\gamma) & \theta_k|\gamma, H \sim H \\ \pi_k = \pi'_k \prod_{l=1}^{k-1} 1-\pi'_l & G_0 = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k} \end{array}$$

A DP is a stochastic process that generates a distribution G_0 in the form of an infinite mixture of atoms θ_k drawn from H. Thus, drawing from the mixture G_0 results in one of the atoms θ_k. The construction of π can be abbreviated with $\pi \sim GEM(\gamma)$.

In our case, the atoms θ_k are multinomial distributions over words in the codebook (*i.e.* activities). Therefore, H needs to be defined as a distribution over multinomial distributions, for which we use the Dirichlet distribution:

$$\begin{array}{l} H = Dir(D_0) \\ \theta_k|\gamma, H \sim Dir(D_0) \end{array}$$

Thus, drawing from θ_k finally generates a word from the codebook (*i.e.* a single flow vector).

We introduce the second DP $G_t \sim DP(\alpha, G_0)$, where the base distribution G_0 itself is drawn from a DP. Thus, a sample G_t will be a subset of G_0. We illustrate this property using the stick-breaking construction again:

$$\begin{array}{ll} \hat{\pi}'_k|\alpha, G_0 \sim Beta(1,\alpha) & \\ \tau_k \sim Mult(\pi_1, \pi_2, \ldots) & \hat{\theta}_k|\alpha, G_0 = \theta_{\tau_k} \\ \hat{\pi}_k = \hat{\pi}'_k \prod_{l=1}^{k-1} 1-\hat{\pi}'_l & G_t = \sum_{k=1}^{\infty} \hat{\pi}_k \delta_{\hat{\theta}_k} \end{array}$$

Parameters. γ and α are hyper-parameters set by the user. Both are priors on the concentration of the word distributions within activities, and influence the number of activities in G_0 and G_t (not given explicitly to the system). Furthermore, D_0 is the parameter for the Dirichlet distribution $H = Dir(D_0)$. In general, higher elements in D_0 produce less variance in samples from H. We set D_0 experimentally (Sec. 3).

Summary. The model in Figure 3 first generates a global set of activities from the infinite mixture G_0. Then, each clip t is assigned a set of activities G_t. Finally, an (arbitrary) number of words are generated by repeatedly drawing first an activity θ_{ti} and then a flow word x_{ti}. The model is learned based on the observed flow words x_{ti} (obtained by assigning each optical flow vector to a word from the codebook, Fig. 2(b)). In summary, this HDP model clusters co-occurring flow words into activities representative for single clips of the video (Sec. 3).

2.2. Mining Local Rules

Our first method for finding temporal dependencies between activities is based on the output of the HDP model. We focus here on discovering *spatially local* rules of the scene. They describe temporal sequences of a few activities that recur frequently in a region of the scene (*e.g.* the local rule illustrated in Fig. 2(c)).

To simplify the notation, let $A = \{a_1, \ldots, a_n\}$ denote the set of activities found with HDP after training (elements in G_0). Furthermore, let P_k be the set of all subsets of A with size k, and S_k one element of P_k.

Markov Models on Sets of Activities. We use Markov Models (MMs) to represent rules. A $MM(S_k) = (V, T)$ describes a sequence of activities from S_k. The state set $V = \{v_1, v_2, \ldots\}$ is the power set of S_k, containing 2^k elements. Thus, a state represents a selection of activities. T is the transition matrix. The probability for a transition from state x to state y is T_{xy}.

As an illustration, assume a $MM(S_2 = \{a_1, a_2\})$ that has the state set $V = \{\{\}, \{a_1\}, \{a_2\}, \{a_1, a_2\}\}$ and a transition matrix T of size 4×4. The sequence of states $q = (q_1, q_2, \ldots, q_t, \ldots)$ is the output of a MM, where $q_i \in V$ (see Fig. 4).

Rule Prototypes. We first define prototypes for the rules that we want to find in the data. A prototype rule is defined by a prototype MM $PMM(k, U)$ involving k activities and a transition matrix U (see Fig. 5 for an example of such a prototype rule; more are shown in the results section). To

[2] Although a DP generates *infinite* mixtures, typically only a few elements have a significant probability. Thus it is sometimes convenient to think of G as a *set* of elements, omitting the elements with negligible probability [11].

[3] Without sharing, the model is ill-posed because there exist infinitely many combinations of activities and MoAs that generate the observed data. See [11] for a detailed discussion about the learnability.

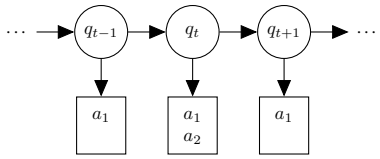


Figure 4: Example of a sequence of states q for $MM(\{a_1, a_2\}) = (V, T)$, where $V = \{\{\}, \{a_1\}, \{a_2\}, \{a_1, a_2\}\}$. Each state q_t depends on its previous state q_{t-1}. The output of a state is a selection of activities $\{a_i\}$.

$$U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0.5 & 0.5 & 0 \end{pmatrix}$$

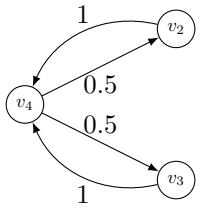


Figure 5: An exemplary prototype rule for a subset of two random activities, *i.e.*, $k = 2$ and $S_k = \{a_1, a_2\}$. The possible states are thus $v_1 = \{\}, v_2 = \{a_1\}, v_3 = \{a_2\}, v_4 = \{a_1, a_2\}$. U is the transition matrix that specifies the type of rule that can be found by mining the learned HMMs.

find an instance of a prototype rule in the video, we search for learned $MM(S_k) = (V, T)$, where T is similar to U, by iterating over all $S_k \in P_k$ (see next paragraphs).

Learning MMs. For a given S_k, we learn $MM(S_k) = (V, T)$ based on the output of HDP. Because V is given as the power set of S_k, only the transition matrix T remains to be learned. For this purpose, we build the sequence of states $q = (q_1, q_2, \ldots, q_t, \ldots)$, where one state q_t consists only of activities in S_k instead of all activities G_t in clip t. Then, we estimate T_{xy} by counting the number of transitions from state x to y in the sequence q, normalized by the number of occurrences of x in q.

Mining Rules. Given a $PMM(k, U)$, we iterate over all learned $MM(S_k) = (V, T)$ for $S_k \in P_k$. Each learned transition matrix T is compared to U using a distance measure. Then, all $MM(S_k)$ are ranked and those with a low distance to U are returned as found instances of a $PMM(k, U)$, *i.e.*, found scene rules of a certain type.

As a distance measure, we use the KL-Divergence D_{KL} to row-wise compare a prototype transition matrix U to a learned transition matrix T:

$$D = \textstyle\sum_x D_{KL}(U_{x\cdot}, T_{x\cdot})$$

Summary. The system is queried with a number of prototype rules $PMM(k, U)$. Based on the output of HDP, all possible Markov models (up to a certain number of states) are learned and compared to these prototype rules. Automatically selected instances of learned Markov models are then returned as scene rules (see results in Sec. 3).

The presented method is able to recover rules involving small subsets of activities. This is very useful to infer local rules that are often meaningful and easy to interpret, *e.g.*, the right of way between different lanes. However, the method is not practical for finding rules involving a large number of activities, *i.e.*, complex rules governing a complete, very complex scene. Furthermore, learning activities jointly with their temporal dependencies would ensure that they optimally explain a video of a scene rather than single clips. To this end, we introduce the model in the next section.

2.3. Mining Global Rules

Our second method jointly models activities and their temporal dependencies. For this purpose, we integrate an arbitrary number of infinite Hidden Markov Models M_c in HDP, such that they run in parallel. In the following, we gradually describe the resulting model called DDP-HMM.

Joint Model. One HMM $M_c = (K_c, T^c)$ consists of K_c states and a transition matrix T^c. The sequences of states from the different HMMs are aligned with the clips t. At each time step t, the HMM M_c is in state v_{ct}, as can be seen in the simplified model in Fig. 6. A state corresponds to a topic θ, from which words x are drawn.

The topics θ for all different states k of one HMM M_c are drawn from a Dirichlet distribution (similar to Sec. 2.1):

$$\theta_{ck} \sim Dir(D_0) \quad \text{topics } \theta \text{ for all different states } k \text{ of } M_c$$

The current state v_{ct} is defined by drawing from a multinomial distribution $Mult(\Pi)$[4], where Π denotes the probabilities of the next possible states (given by $T^c_{v_{c(t-1)}}$):

$$v_{ct} \sim Mult(T^c_{v_{c(t-1)}}) \quad \text{next state } v \text{ for } M_c \text{ based on last state}$$

The words x_{ti} of a clip t are finally generated by repeatedly drawing from the respective topics $\{\theta_{cv_{ct}}\}$ from the different HMMs, defined by multinomial distributions identical to Sec. 2.1. Since $M_{c_{ti}}$ is the HMM that generates word x_{ti} and thus currently is in state $v_{c_{ti}t}$, x_{ti} is sampled from the current topic $\theta_{c_{ti}v_{c_{ti}t}}$ (Fig. 6):

$$x_{ti} \sim Mult(\theta_{c_{ti}v_{c_{ti}t}}) \quad \text{word from topic of current state}$$

The assignment c_{ti} determines which HMM generates the word x_{ti}. This generative model is shown in Fig. 6. Here, c_{ti} as well as the number of HMMs and the number of states K_c for each HMM are inferred from the data (see subsequent paragraphs).

Infinite HMM with DPs. As seen before, in a HMM the transition probability T_{xy} from state x to state y is a finite mixture over the set of states $\{1, \ldots, K_c\}$. To model the transition to an unknown number of states, an infinite mixture is required. To this end, we introduce two levels of Dirichlet processes:

$$\begin{aligned} \Pi_0 &\sim GEM(\gamma) \\ \Pi_k &= DP(\alpha_0, Mult(\Pi_0)) \\ v_t | v_{t-1} &\sim \Pi_{v_{t-1}} \end{aligned}$$

[4]More precisely, we use $Mult(1, \cdot)$, which is sometimes also called categorical distribution.

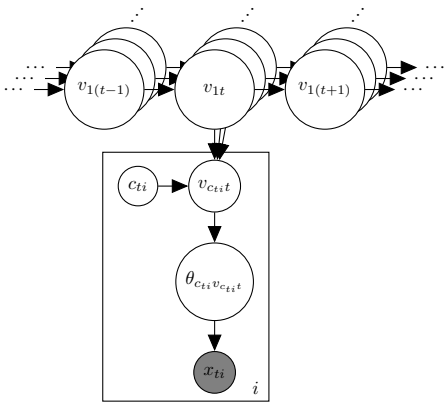


Figure 6: The states v_{ct} from the different HMMs M_c generate the words x_t in clip t. To this end, the assignment c_{ti} first selects one HMM $M_{c_{ti}}$ and its state $v_{c_{ti}t}$. Then, a topic θ and finally the word x_{ti} is drawn.

Here, Π_0 is the prior on the transition matrix, formulated as a stick-breaking construction (as in Sec. 2.1). Then, $\{\Pi_k\}_{k=1}^{\infty}$ are the (infinite) transition probabilities from state k to the next state, again defined by Dirichlet processes.

Such a model is shortly described by Teh *et al*. [11] and shown to be equivalent to Infinite Hidden Markov Models [2]. Although the number of states K_c in the model is infinite, the actual sequence of observed states during training is finite, thus determining K_c.

Infinite Mixture of infinite HMMs. As described before, a word is drawn from a mixture of states from infinitely many HMMs. To select one HMM from an infinite set of HMMs $\{M_c\}_{c=1}^{\infty}$, we use a Dirichlet process, again formulated as a stick-breaking construction:

$$\pi \sim GEM(\beta)$$
$$c_{ti} \sim Mult(\pi)$$

Thus, a sample π is an infinite vector of mixing weights, based on which one HMM $M_{c_{ti}}$ is selected to generate a word x_{ti}.

In this DP, the atoms themselves are stochastic processes, *i.e*., the HMMs M_c used to generate words. Hence, this is called a Dependent Dirichlet Process [8].

Complete DDP-HMM Model. Bringing everything together, Fig. 7 shows the complete model. We call it Dependent Dirichlet Process Hidden Markov Model (DDP-HMM). To sum up, first, an infinite mixture π of infinite HMMs M_c is generated. Each M_c gets an infinite mixture as transition prior Π_{c0}. Based on this prior, transition probabilities are generated as infinite mixtures Π_{ck}. Each state is associated with one topic θ_{ck}.

Then, the sequence of states for a HMM is generated by consecutively drawing from the respective transition probability vector Π_{ck}. To generate a word x_{ti}, first a HMM is selected by $c_{ti} \sim Mult(\pi)$. Second, the word is drawn from the corresponding topic $\theta_{c_{ti}v}$ at time t.

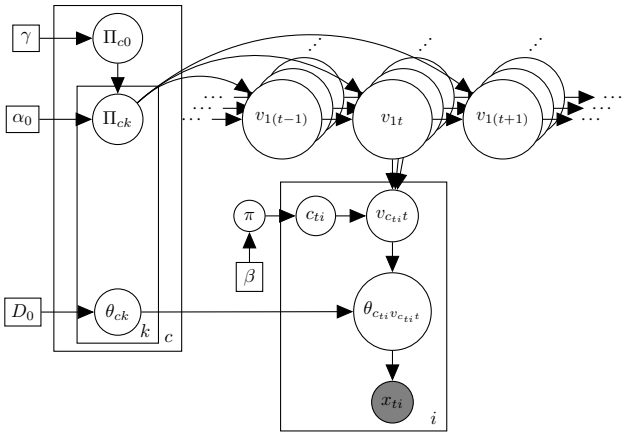


Figure 7: The complete DDP-HMM model integrates an infinite number of infinite HMMs running in parallel. They jointly generate the flow words x_{ti} in clip t. In contrast to the reduced model in Fig. 6, the DP is shown to select a HMM $M_{c_{ti}}$ for generating one word x_{ti}. Furthermore, the transition prior and transition probabilities for the infinite HMMs are shown, generated by two levels of DPs. Also, the Dirichlet distribution to generate all possible topics θ_{ck} is included.

The complete DDP-HMM model can be written as the following process:

$\pi \sim GEM(\beta)$	mixing weights for HMMs
$c_{ti} \sim Mult(\pi)$	selected HMM $M_{c_{ti}}$
$\Pi_{c0} \sim GEM(\gamma)$	transition prior for M_c
$\Pi_{ck} = DP(\alpha_0, Mult(\Pi_{c0}))$	transition prob. from state k
$\theta_{ck} \sim Dir(D_0)$	topics θ for states k of M_c
$v_{ct} \sim \Pi_{cv_{c(t-1)}}$	next state for M_c
$x_{ti} \sim Mult(\theta_{c_{ti}v_{c_{ti}t}})$	word from topic of one state

Inference. We use Gibbs Sampling for inference in our DPP-HMM model. For this purpose, we alternate between sampling the assignment of words to HMMs (π, c_{ti}) and sampling the sequence of states (Π_{c0}, Π_{ck}, v_{ct}). The first step is similar to inference in HDP (described in [11]), *i.e*., the assignments of words to topics. Furthermore, as in [11], θ_{ck} is integrated out.

Summary. This model jointly learns activities and dependencies, while inferring different parameters such as the optimal number of HMMs and the number of their states from the data. Modelling activities and their time dependencies jointly over time results in different activities compared to those learned by HDP for single clips (Sec. 2.1). The resulting activities from DDP-HMM explain a whole state of the scene rather than small atomic activities (see Sec. 3). While this allows to find global rules that govern the whole scene rather than local, spatially limited rules, they are not as easy to interpret anymore.

3. Experiments

We experiment on videos of two different crowded, public scenes in Zurich (HD, 25 fps, 2x3 hours) and on two videos

Figure 8: The most important activities (topics) for the first sequence. The topics are shown in order of decreasing importance, as automatically determined by the model.

Figure 9: The most important activities (topics) for the second sequence. The single car and tram lanes are separated well.

of London traffic scenes from [5] (360x288, 25 fps, 2x1 hour)

Local Rules. For both sequences we show all activities (Figs. 8 and 9) found by HDP that explain at least 2% of all observations (Sec. 2.1). Based on them, we search rules of up to 4 activities (Sec. 2.2). Here, we show two exemplary results for each sequence.

The rule in Fig. 10 shows the relationship between trams that pass by straight and turning cars. The activities are never observed simultaneously, thus are mutually exclusive. The car activity (a) is observed more frequently than tram activities (b and c) (see HMM transition probabilities). After a tram passed by, it is very likely that cars are observed again afterwards. Only after a tram drove from right–left (c), a tram from left–right is sometimes observed next, never the other way around.

The turning car activity is divided into two separate activities by our model (Figs. 11(a), 11(c)). This is necessary because cars driving straight (Fig. 11(b)) have the right of way. Hence, turning cars sometimes have to wait (a) and let other cars (b) pass before continuing (c). They should

(a) Tram left–right. (b) Bus turns. (c) Tram right–left.

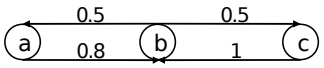


Figure 10: A blocking rule between trams and buses. A bus moving in (b) is blocked by a tram in either (a) or (c). Only transitions with probability higher than 0.2 are shown.

(a) Bus turns (start). (b) Car right–left. (c) Bus turns (end).

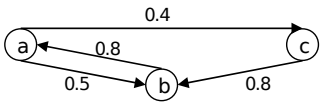


Figure 11: A bus moving on the sequence (a)-(c) is often interrupted by blocking traffic in (b).

(a) Cars left–right. (b) Cars right–left. (c) Cars top–down. (d) Cars right–down.

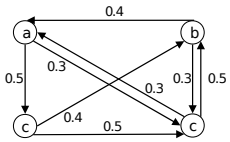


Figure 12: A local rule that explains a traffic light sequence.

wait behind a line (left border of the image), thus (a) and (b) should not be observed simultaneously, which however is not always the case. In contrast, (c) and (a) are never observed simultaneously, which would cause an accident.

For the second sequence, Fig. 12 shows a traffic light rule between cars with mutually exclusive activities. Although there exists a most probable sequence, almost every other sequence is possible, which is unusual for a traffic light situation. Thus, either the traffic light system is intelligent and adapts permanently to the current traffic situation, or a local rule involving four activities is not sufficient.

Global Rules. For the second scene, we show in Fig. 13 the 5 states that explain at least 5% of the observed flow in the scene, discovered by the DDP-HMM. Activities found with HDP and activities (*i.e.*, states) found with DDP-HMM on the same scene are different (compare Fig. 9 and Fig. 13). In general, DDP-HMM finds more global activities, since it has to explain the complete state of the scene, not just a locally restricted part.

The scene is a crossing governed by a traffic light. Thus, the sequence of activities is theoretically strictly defined. The transition matrix learned by DDP-HMM reflects this

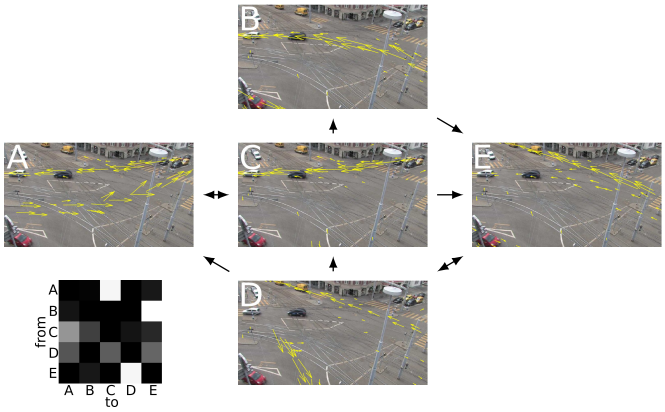


Figure 13: Transition graph and matrix for a traffic light controlled scene. The usual loop for the traffic lights is: $(A, C) - (B, E) - D$.

well with its simple structure (Fig. 13): State A represents the two lanes from left to right and vice versa. Often however, there are more cars coming from the right, thus state C usually follows A, as the lane from right to left tends to be active for a longer time. Then either state B or E follow, which both correspond to the same signal light configuration. Depending on whether there are more cars going left (B) or straight (E), a different state is used. Finally, state B follows, representing cars going up and down.

The usual loop for the traffic lights is: $(A, C) - (B, E) - D$. We group the states (A, C) and (B, E) because they correspond to the same traffic light configuration. Even though the model is not restricted to only one HMM, here it has learned that one is sufficient, because the traffic light can be represented by a single HMM.

This fact is supported by an evaluation on β. β is the concentration parameter that acts as a prior on the number C of HMMs. We performed a grid search on $\beta \in \{0.1, 0.5, 1.5, 5.0\}$. Although the concentration parameter is chosen very high, the learned model always converges to one HMM that explains at least 90% of the scene. Only if very rare motion needs to be explained, a single HMM is not sufficient anymore.

To further validate that the model is able to find more than one HMM we first generated (artificial) data according to the model and then applied our inference algorithm. Averaged over 100 runs with different generated data from on average 3.4 HMMs about 73% of the words are assigned correctly. For 6 HMMs it scores at 59%.

In contrast, the other traffic scene is not governed by a traffic light (Fig. 14). Thus, the transition matrix is much more complex. There is no single typical sequence of activities. Figure 14 shows a few selected states along with the transition matrix.

Based on a trained model, new video data of the same scene can be annotated with states along the timeline in real-time. At each time step, the state that explains best the current flow is chosen. Consequently, if an observation

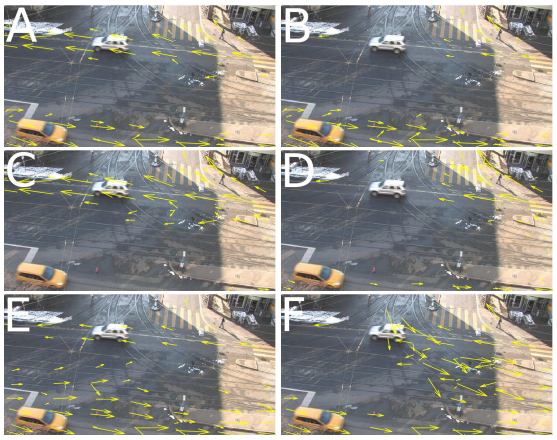


Figure 14: A selection of states and the transition matrix of a global rule found by the joint DDP-HMM for an erratic scene, *i.e.* not controlled by a traffic light. As can be seen, there are many different sequences of activities because the transition probabilities to different states are often similar.

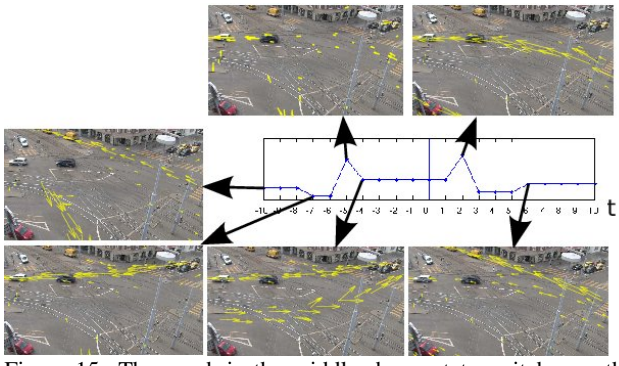


Figure 15: The graph in the middle shows state switches as the sequence progresses along the time line. A time step is 3 seconds. At each state change, we show the flow explaining the scene until the next state change.

cannot be explained well, an alarm could be triggered automatically. Figure 15 illustrates this on a short extract from the video of the traffic light scene (Fig. 13), where the current state is plotted for some time steps.

Hospedales *et al*. [5]. We apply our models to the 2 traffic scenes of [5] and compare to their results qualitatively.

First, we learn activities. Fig. 17 shows only a few of the activities found, due to space limitations. Our method finds activities similar to those shown in [5], but a full comparison is not possible, as also [5] show only a few activities.

Next, we use our model of Sec. 2.3 to find global rules, akin to the Markov Models of [5]. Fig. 16 shows the states along with the transition matrix. For the top scene, our model accurately recovered the traffic light cycle governing the scene. The transition matrix shows our method has found two alternating cycles DB... and DCAB..., whereas the transition matrix in [5] does not support any cycle. For

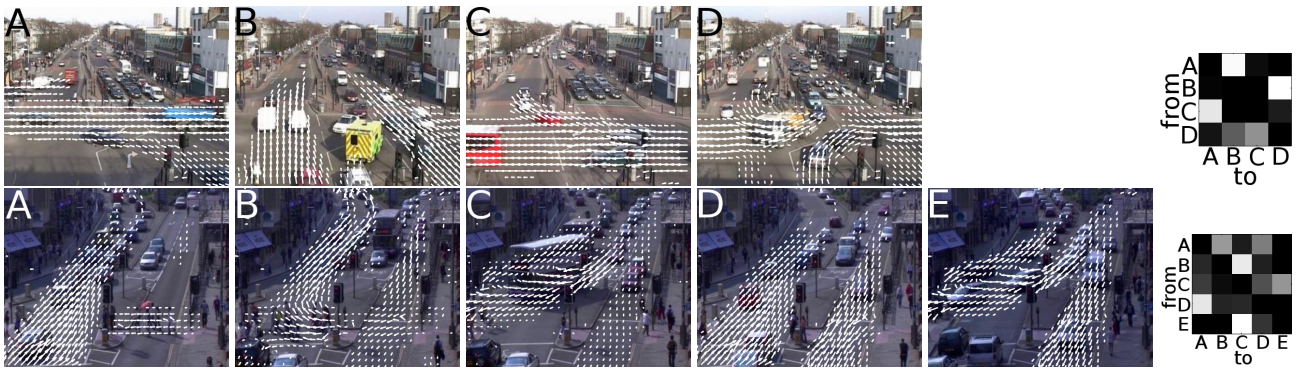


Figure 16: Top: A scene with a traffic light. 4 states are automatically found. The transition matrix accurately reflects the traffic light cycle. Bottom: Traffic scene with crossing pedestrians. 5 states are automatically found. After state A (pedestrian crossing the street on the right), either state B or C follows with pedestrians either crossing the street on the left or waiting in between.

Figure 17: A selection of found activities on the data of [5]. We find the same lanes. Also, on the second row, the pedestrians and cars are recognized as different activities.

the bottom scene, every car lane and pedestrian crossing is covered by some of the 5 states found by our model. In contrast, the 3 states found in [5] do not cover all the movement in the scene. As in [5], our transition matrix suggests that states A and B (pedestrians crossing) occasionally interrupt the flow of traffic. Moreover our transition matrix is more complete than that of [5], as it also explains transitions between states covering movement missed by the states found by [5]. Finally, note that our model learns both the number of activities and the number of states automatically, whereas in [5] they are manually set for each video.

4. Conclusion

To understand the behaviour of agents in a scene, both spatial and temporal dependencies between moving agents are relevant. To analyze them, we presented two methods to automatically learn co-occurring activities and temporal rules between them, first stepwise and then jointly. The first model allows to mine local rules. The second DDP-HMM model is more powerful and automatically infers all parameters from the data, such as the optimal number of HMMs necessary to explain the rules governing a scene. It finds global rules. We demonstrated experiments using two long video sequences of complex Zurich scenes and discussed the differences between the two proposed approaches. Moreover, we qualitatively compared our results with [5] on their London videos. As a possible application, a trained model can annotate new video data from the same scene with states along the timeline in real-time for an automatic interpretation of video, *e.g.* to detect unusual activity.

References

[1] A. Basharat, A. Gritai, and M. Shah. Learning object motion patterns for anomaly detection and improved object detection. In *CVPR*, 2008.

[2] M. Beal, Z. Ghahramani, and C. Rasmussen. The infinite hidden markov model. In *NIPS*, 2002.

[3] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[4] T. Duong, H. Bui, D. Phung, and S. Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *CVPR*, 2005.

[5] T. Hospedales, S. Gong, and T. Xiang. A markov clustering topic model for mining behaviour in video. In *ICCV*, 2009.

[6] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *PAMI*, 28(9):1450–1464, 2006.

[7] J. Li, S. Gong, and T. Xiang. Global behaviour inference using probabilistic latent semantic analysis. In *BMVC*, 2008.

[8] S. MacEachern. Dependent nonparametric processes. In *ASA Proc. of the Section on Bayesian Statistical Science*, 1999.

[9] I. Saleemi, K. Shafique, and M. Shah. Probabilistic modeling of scene dynamics for applications in visual surveillance. *PAMI*, 31(8):1472–1485, 2009.

[10] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *PAMI*, 22(8):747–757, 2000.

[11] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[12] X. Wang, X. Ma, and W. Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *PAMI*, 31(3):539–555, 2009.

[13] T. Xiang and S. Gong. Video behaviour profiling for anomaly detection. *PAMI*, 30(5):893–908, 2008.

[14] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. In *DAGM*, 2007.

[15] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *CVPR*, 2004.