Attend, Infer, Repeat: Notes

Adam Kosiorek

January 4, 2018

1 Introduction

Attend, Infer, Repeat (AIR; Eslami et al., 2016) is a VAE-type model capable of decomposing a static scene into its constituent parts. This is useful, since it explicitly represent distinct parts of the scene, which provides useful and actionable representations for down-stream processing. An example application are proposal location for object detection algorithms. While not guaranteed to represent objects, parts of the scene represented by separate representations are bound to be statistically significant. The purpose of this document is to explore the model in more detail.

2 Generative Model and Priors

Let $\mathbf{x}$ an image, n a number of statistically interesting separate parts in the image, $\mathbf{z} = \{\mathbf{z}^i, \dots, \mathbf{z}^n\}$ a group of variables, where each variable describes a part of the scene. Let $p_\theta(\mathbf{z}, n) = p_\theta(\mathbf{z} \mid n) p_\theta(n)$ be a prior over latent variables describing our assumptions about possible layouts and complexity of the scene and let $p_\theta(\mathbf{x} \mid \mathbf{z})$ be a *generating* model. The prior and the generating model together form a generative model of the scene, where the prior describes it in terms of latent variables and the generating model uses the latent variables to paint the scene. We can write down the marginal distribution of images as

$$p_\theta(\mathbf{x}) = \sum_{n=1}^{N} p_{\theta,N}(n) \int p_\theta(\mathbf{z} \mid n) p_\theta(\mathbf{x} \mid \mathbf{z}) \, d\mathbf{z}. \tag{1}$$

We take the prior to have the following form:

$$p_\theta(n) = \text{Geom}(n \mid \theta), \tag{2}$$

$$p_\theta(\mathbf{z} \mid n) = \prod_{i=1}^{n} p_\theta\left(\mathbf{z}^i\right) = \prod_{i=1}^{n} \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{3}$$

which assumes that $\mathbf{z}^i$ are independent under the prior. The success probability for the geometric distribution is chosen to encourage sparse (as few steps as possible) solutions. In practice, it is annealed over 100k training steps to a very low value on the order of $p = 10^{-e}$, $e \in [5, 10]$.

3 Inference through an Approximate Posterior

Since the latent variables $\mathbf{z}^i$ are independent under the prior, they are exchangeable. It introduces symmetries, which increases the volume of the search space in a combinatorial way. To address this issue, the original work formulates inference as a sequential problem, where latent variables describing a part of a scene depend on previously inferred latent variables. Specifically, it parametrises n as a vector of n ones followed by a zero $\mathbf{z}^p$. With this representation and a sequential implementation of inference, it is enough to output a single *presence* indicator variable at each processing step. The approximate posterior is given by

$$q_\phi(\mathbf{z}, \mathbf{z}^p \mid \mathbf{x}) = q_\phi\left(z^{p,n+1} = 0 \mid \mathbf{z}^{1:n}, \mathbf{x}\right) \prod_{i=1}^{n} q_\phi\left(\mathbf{z}^i, z^{p,i} = 1 \mid \mathbf{z}^{1:i-1}, \mathbf{x}\right), \quad (4)$$

with q_ϕ implemented as a neural network. To avoid explaining the same object twice, it is vital to capture the dependency $\mathbf{z}^i, z^{p,i} \mid \mathbf{z}^{1:i-1}, \mathbf{x}$, which is implemented as a recurrent neural network R_ϕ with hidden state $\boldsymbol{h}^i$, where the initial hidden state $\boldsymbol{h}^0$ is randomly initialised and learnable, and specifically as

$$\boldsymbol{\omega}^i, \boldsymbol{h}^i = R_\phi(\mathbf{x}, \mathbf{z}^{i-1}, \mathbf{h}^{i-1}). \quad (5)$$

The variable ω^i specifies parameters of the probability distribution over $\mathbf{z}^i$ and $z^{p,i}$ and introduces a conditional independence property, namely $\mathbf{z}^i \perp\!\!\!\perp z^{p,i} \mid \omega^i$. We can use this property to factorise the approximate posterior distribution as

$$q_\phi(\mathbf{z}, \mathbf{z}^p \mid \mathbf{x}) = q_\phi\left(z^{p,n+1} = 0 \mid \omega^{n+1}\right) \prod_{i=1}^{n} q_\phi\left(\mathbf{z}^i \mid \omega^i\right) q_\phi\left(z^{p,i} \mid \omega^i\right). \quad (6)$$

Since the hidden state $\boldsymbol{h}^i$ is deterministic, it follows a Dirac distribution centred on $\mathbf{h}^i$ and there is no need to integrate over the it: the integral collapses to a single value. Latent distribution statistics $\boldsymbol{\omega}$ are computed iteratively and depend on the previous latent variables. This precludes analytical computation of the KL-divergence, which has to be approximated by sampling.

4 Learning by Maximising the ELBO

AIR is trained by maximising the evidence lower bound $\mathcal{L}_{AIR}$ (ELBO) given by

$$\mathcal{L}_{AIR}(\phi, \theta) = \mathbb{E}_{q_\phi}\left[\log \frac{p_\theta(\mathbf{z}, n, \mathbf{x})}{q_\phi(\mathbf{z}, n \mid \mathbf{x})}\right] \tag{7}$$

$$= \mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{x} \mid \mathbf{z})] - \mathrm{KL}\big(q_\phi(\mathbf{z}, n \mid \mathbf{x}) \parallel p_\theta(\mathbf{z}, n)\big). \tag{8}$$

The first term is a probabilistic analog of the reconstruction error, while the second term is a complexity penalty. In this case it encourages minimum-length encoding of every part of the scene as well as decomposing the scene into a minimal number of parts. The KL term can be rewritten according to the product rule as

$$\mathrm{KL}\big(q_\phi(\mathbf{z}, n \mid \mathbf{x}) \parallel p_\theta(\mathbf{z}, n)\big) = \tag{9}$$

$$= \mathrm{KL}\big(q_\phi(n \mid \mathbf{x}) \parallel p_\theta(n)\big) + \mathbb{E}_{q_\phi(n|\mathbf{x})}\big[\mathrm{KL}\big(q_\phi(\mathbf{z} \mid n, \mathbf{x}) \parallel p_\theta(\mathbf{z} \mid n)\big)\big]. \tag{10}$$

5 Policy Gradient Techniques

AIR implements a sequential inference procedure, where the number of steps is given by a discrete latent variable. Policy gradients are used here to backprop through samples from a discrete probability distribution.

5.1 NVIL and VIMCO

NVIL, introduced by Mnih and Gregor, 2014, is a method of variance reduction for the score-function estimator (REINFORCE, Williams, 1992). It constructs a control variate from a running mean of the learning signal and an input-dependent baseline, typically a neural network trained jointly with the main model. It also tracks the running standard deviation of the learning signal and normalises centred learning signal. I implemented it, but it doesn't work that well, presumably because of different effective weightings of the continuous and discrete components of the loss. Even Mnih himself discourages using it, and advocates his newer method, VIMCO, instead. VIMCO does indeed work much better and leads to stable training of the model.

5.2 Convex Relaxation

To be done... concrete distribution/gumbel Jang, Gu, and Poole, 2016; Maddison, Mnih, and Teh, 2016. It is a biased estimator, but Tucker et al., 2017 introduced a method to debias it called REBAR.

6 FIVO for Better Bounds

Introduced by Maddison, Lawson, et al., 2017; It's supposedly much better for sequential estimation such as AIR, and especially the sequential version, than the usual ELBO. The problem is it cannot be used with VIMCO, but it can be used with convex relaxation of the discrete latent variable.

7 Complex Backgrounds: Behaviour and Extenstions

AIR depends on statistical differences between pixels belonging to an object and the ones belonging to the background. Moreover, the way it has been defined, it can reconstruct only objects while neglecting the background completely. It is natural to expect that it simply wouldn't work in the case of non-empty backgrounds. To mitigate this issue, it is necessary to augment the model with a background-explaining component g. In a simple case, it could produce a background $\boldsymbol{b}$ based on the last hidden state of the controller RNN R_ϕ, specifically:

$$\boldsymbol{b} = g_\phi(\boldsymbol{h}^N) \tag{11}$$

$$y = h^{\text{dec}}(\mathbf{z}, \boldsymbol{b}) \tag{12}$$

For the case of MNIST, g can be implemented as a simple MLP. For more complicated model, it would have to be a more complicated model based on up-convolutions, but this is also true about the glimpse decoder.

Design an experiment to test this behaviour.

8 An Extension to Timeseries: Attend, Propagate, Discover, Repeat (APDR)

AIR displays an inconsistent behaviour depending on initialisation. It is randomly initialised and it tends to converge to different solutions when training is restarted. Only a small subset of the solutions is close to what the user would want, namely explaining one object at a time. This is understandable, since a dataset of single images cannot encode any *consistency* prior, whereby objects would have to be preserved when they move. Extending AIR to sequential data has a potential of solving this issue, since it should be much easier to explain an object by changing its position parameter than to infer its appearance description from scratch at every time-step. Moreover, it prevents explaining two objects moving separately with a single glimpse, since it would be impossible to preserve them consistently while reconstructing.

An extension to timeseries is not trivial, however. It requires some form of dependency between consecutive time-step. It is important to distinguish

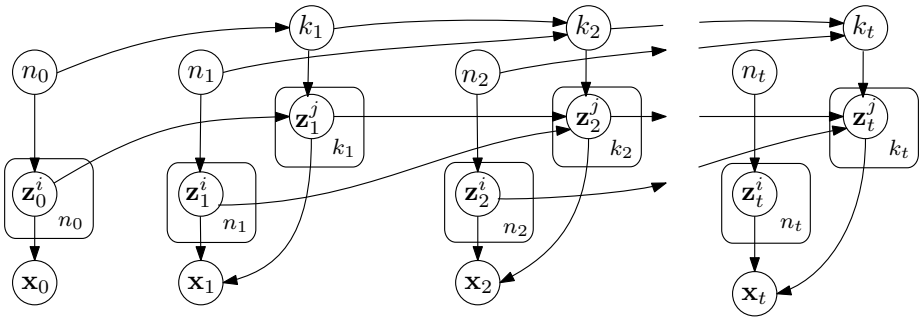

Figure 1: The generative story of sequential AIR.

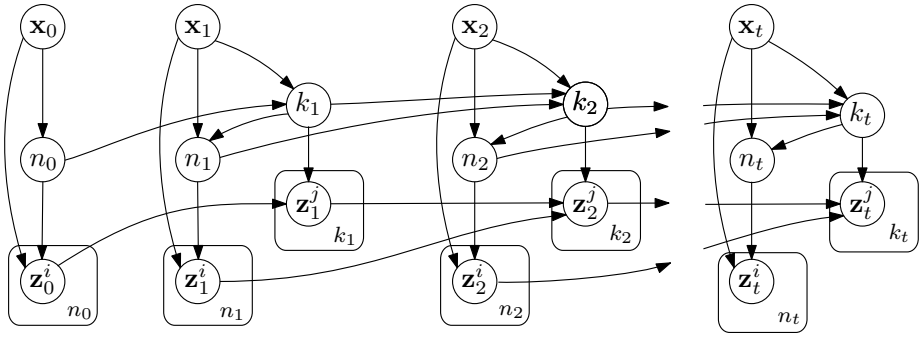

Figure 2: Graphical model for inference in sequential AIR.

temporal and within-time-step dependencies. We dub the latter *sequential* dependencies. In the original work, sequential dependencies exist between the deterministic hidden states and stochastic latent variables, namely $\boldsymbol{h}^{\mathrm{i}} \mid \boldsymbol{h}^{\mathrm{i}-1}, \mathbf{z}^{\mathrm{i}-1}, z^{\mathrm{p},\mathrm{i}-1}$, and $\mathbf{z}^{\mathrm{i}}, z^{\mathrm{p},\mathrm{i}} \mid \boldsymbol{h}^{\mathrm{i}-1}$. A temporal version of the algorithm requires (or might benefit from) the following:

- between-time-step hidden state dependence e. g., $\boldsymbol{h}_{\mathrm{t}} \mid \boldsymbol{h}_{\mathrm{t}-1}$, where this time-step might or might not be the same as the sequential hidden state
- temporal dependence on object position
- temporal dependence on object appearance
- temporal dependence on object presence

8.1 Sequential Generative Model and Prior

Let us start by describing the prior generative story. Images are generated by first assuming that, at every time-step, objects are propagated from the previous time-step and some new objects can be introduced (or discovered), where

propagation and discovery are handled by different parts of the model. Let $k \in \{0, 1, \ldots, K\}$, the number of objects propagated from the previous time-step and let $n \in \{0, 1, \ldots, N - k\}$ the number of objects discovered at the current time-step. The maximum of $K \in \mathcal{N}_+$ objects can be propagated and the model can handle up to $N \in \mathcal{N}_+$ total objects, therefore $K \leqslant N$. Moreover, at every time-step, the model will generate a set of latent variables, one for each object. If we use superscripts D and P to denote latent variables generated by discovery and propagation models, respectively, then the aggregated latent variable $\mathbf{z}_t = \mathbf{z}_t^P \cup \mathbf{z}_t^D$, where $\mathbf{z}_t^P = \{\mathbf{z}_t^{P,1}, \ldots, \mathbf{z}_t^{P,k_t}\}$ and $\mathbf{z}_t^D = \{\mathbf{z}_t^{D,1}, \ldots, \mathbf{z}_t^{D,n_t}\}$. Consequently, if we use $m_t = k_t + n_t$ to denote the total number of objects present at time t, then $\mathbf{z}_t = \{\mathbf{z}_t^1, \ldots, \mathbf{z}_t^{m_t}\}$. We note that the structure of the latent variables from the discovery and propagation models is the same, and in its simplest form it consists of $\mathbf{z}_t^i = \{\mathbf{z}_t^{\text{what},i}, \mathbf{z}_t^{\text{where},i}, b_t^i\}$. The components represent appearance, location and presence of an object at the given time-step.

At the first time-step $t = 1$ there are no objects to propagate, so we sample up to N objects from a discovery prior $p\big(n_1, \mathbf{z}_1^D \mid N\big)$. Starting from the second time-step $t = 2$, the model first propagates objects by sampling from a propagation prior $p\big(k_2, \mathbf{z}_2^P \mid n_1 + k_1, \mathbf{z}_1\big)$, where $\mathbf{z}_1 = \mathbf{z}_1^D$. The model also samples n_2 new objects from the prior $p\big(n_2, \mathbf{z}_2^D \mid N - k_2\big)$. From now on, that is for $t \geqslant 2$, we set the aggregated latent variable $\mathbf{z}_t = \mathbf{z}_t^P \cup \mathbf{z}_t^D$. This process continues up to the final time-step T. The images are generated by passing the latent variables $\mathbf{z}_{1:T}$ to the generating model $p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)$ one at a time. Note that $\mathbf{z}_t$ is a set of up to N latent variables, where each latent variable represents a separate object. The generating model acts separately on every latent variable in the set, and the output random variable of the generating model consists of the sum of outputs of the generative model for each latent variable in the set. Figure 1 shows the graphical model of the generative story. Please note that the prior for the number of new objects stays the same for every time-step and it is analogous to the prior used by AIR, with the exception that the number of maximum objects can differ.

To make it more formal, we will now give all equations governing the generative process. The prior for the discovery model is defined as follows.

$$\begin{aligned} &p_\theta\Big(n_t, \mathbf{z}_t^D \mid N\Big) = \\ &p(n_t \mid N) p_\theta(\mathbf{z}_t \mid n_t) = \text{TruncatedGeom}\,(n_t \mid N, \theta) \prod_{i=1}^{n_t} \mathcal{N}(\mathbf{0}, \mathbf{I}), \end{aligned} \tag{13}$$

where $\text{TruncatedGeom}\,(n_t \mid N, \theta)$ is a geometric distribution, whose support is truncated to $\{0, 1, \ldots, N\}$ and the success probability is given by θ. The

propagation model is defined by the following.

$$\begin{aligned} p\left(k_t, \mathbf{z}_t^P \mid m_{t-1}, \mathbf{z}_{1:t-1}\right) &= \prod_{j=1}^{m} p\left(b_t^{P,j} \mid \mathbf{z}_{1:t-1}^j\right) p\left(\mathbf{z}_t^j \mid \mathbf{z}_{1:t-1}^j\right) = \\ \prod_{j=1}^{m_{t-1}} &\text{Bernoulli}\left(b_t^{P,j} \mid \psi(\mathbf{z}_{1:t-1}^j)\right) \mathcal{N}\left(\mathbf{z}_t^j \mid \boldsymbol{\mu}(\mathbf{z}_{1:t-1}^j), \boldsymbol{\sigma}^2(\mathbf{z}_{1:t-1}^j)\right), \end{aligned} \tag{14}$$

with $m_t = n_t + k_t$ the total number of objects present at time t, $\mathbf{z}_{1:t-1}^j$ the history of latent variables belonging to object j and $\mathbf{z}_{1:0}^j = \emptyset$. In the above we reparamterise k_t as a sum of Bernoulli random variables $b_t^{P,j}$, that is $k_t = \sum_{i=1}^{m} b_t^{P,j}$. Since the probability of propagation ψ^j is different for every object, k_t follows a Poisson-Binomial distribution with m trials. Direct conditioning on the whole history of latent variables is impractical, and therefore we use a learned deterministic RNN R_θ^{prior} to estimate ψ, μ and σ^2. the RNN shares parameters but maintains a separate hidden state for every object j. We can rewrite Equation (14) in terms of a simple recurrence as

$$\begin{aligned} p\left(k_t, \mathbf{z}_t^P \mid m_{t-1}, \mathbf{z}_{1:t-1}\right) &= p\left(k_t, \mathbf{z}_t^P \mid m_{t-1}, \boldsymbol{h}_t^{\text{prior, }j}\right) = \\ \prod_{j=1}^{m_{t-1}} &\text{Bernoulli}\left(b_t^{P,j} \mid \psi(\boldsymbol{h}_t^{\text{prior, }j})\right) \mathcal{N}\left(\mathbf{z}_t^j \mid \boldsymbol{\mu}(\boldsymbol{h}_t^{\text{prior, }j}), \boldsymbol{\sigma}^2(\boldsymbol{h}_t^{\text{prior, }j})\right), \end{aligned} \tag{15}$$

$$\boldsymbol{h}_t^{\text{prior, }j} = R_\theta^{\text{prior}}\left(\mathbf{z}_{t-1}^j, \boldsymbol{h}_{t-1}^{\text{prior, }j}\right). \tag{16}$$

Any parameters in the generating model, e. g., the parameters of the RNN, can be learnt jointly with the inference model (to be described shortly). Alternatively, one could use a non-parametric prior for the discrete latent variable $k_{1:T}$, e. g., the Indian Buffet Process (Gael, Teh, and Ghahramani, 2009), *cf.* Section 8.5.

I should actually use the IBP prior...

Images are created by decoding the latent variables into small 'glimpses' $\widehat{\mathbf{g}}_t^j \in \mathbb{R}^{h \times w \times c}$ that are then transformed into partial images $\mathbf{y}_t^j \in \mathbb{R}^{W \times H \times c}$, $w < W$, $h < H$, which match the size of the original image, and they are summed up. More formally, recall that each latent variable $\mathbf{z}_t^j$ contains $\mathbf{z}_t^{j,\text{where}}$ and $\mathbf{z}_t^{j,\text{what}}$, we have

$$\widehat{\mathbf{g}}_t^j = f_\theta^{\text{dec}}\left(\mathbf{z}_t^{\text{what,}j}\right), \tag{17}$$

$$\mathbf{y}_t^j = f^{\text{STN}}\left(\widehat{\mathbf{g}}_t^j, \mathbf{z}_t^{\text{where,}j}\right), \tag{18}$$

$$\boldsymbol{\mu}_x = \sum_{j=1}^{m_t} \mathbf{y}_t^j, \tag{19}$$

$$\widehat{\mathbf{x}}_t \sim \mathcal{N}\left(\mathbf{x}_t \mid \boldsymbol{\mu}_x, \boldsymbol{\sigma}_x^2\right), \tag{20}$$

with the decoder f^{dec}, spatial transformer f^{STN} and a fixed per-pixel standard deviation σ_x^2.

All components of the generative model taken together specify a joint probability $p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}, n_{1:T}, k_{1:T})$ over the sequence of images and the corresponding latent variables. We can expand it as

$$\begin{aligned} p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}, n_{1:T}, k_{1:T}) &= \prod_{t=1}^{T} p_\theta(\mathbf{x}_t, \mathbf{z}_t, n_t, k_t \mid \mathbf{z}_{1:t-1}, n_{1:t-1}, k_{1:t-1}) \\ &= \prod_{t=1}^{T} p_\theta(\mathbf{x}_t \mid \mathbf{z}_t) p_\theta(\mathbf{z}_t, n_t, k_t \mid \mathbf{z}_{1:t-1}, n_{1:t-1}, k_{1:t-1}) \\ &= \prod_{t=1}^{T} \underbrace{p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)}_{\text{generation}} \underbrace{p_\theta\left(\mathbf{z}_t^{\text{D}}, n_t \mid \mathbf{z}_t^{\text{P}}, k_t\right)}_{\text{discovery}} \underbrace{p_\theta\left(\mathbf{z}_t^{\text{P}}, k_t \mid \mathbf{z}_{1:t-1}, n_{1:t-1}, k_{1:t-1}\right)}_{\text{propagation}}. \end{aligned} \tag{21}$$

In the second line we used the fact that the observation at time t is conditionally independent of latent variables at previous time-steps given the current ones. In the third line we split $\mathbf{z}_t$ as the union of latent variables corresponding to the discovered objects $\mathbf{z}_t^{\text{D}}$ and propagated objects $\mathbf{z}_t^{\text{P}}$. Only the latter depend on the history of latent variables; the former depend only on what is captured in propagation to the current step. The last term can be decomposed as

$$\begin{aligned} &p_\theta\left(\mathbf{z}_t^{\text{P}}, k_t \mid \mathbf{z}_{1:t-1}, n_{1:t-1}, k_{1:t-1}\right) \\ &\quad= \int p_\theta\left(\mathbf{z}_t^{\text{P}}, k_t \mid \mathbf{z}_{t-1}, n_{t-1} + k_{t-1}, \boldsymbol{h}_t^{\text{P}}\right) p_\theta\left(\boldsymbol{h}_t^{\text{P}} \mid \boldsymbol{h}_{t-1}^{\text{P}}, \mathbf{z}_{t-1}\right) \mathrm{d}\boldsymbol{h}_t^{\text{prior}} \\ &\quad= p_\theta\left(\mathbf{z}_t^{\text{P}}, k_t \mid \mathbf{z}_{t-1}, m_{t-1}, \boldsymbol{h}_t^{\text{prior}}\right)\big|_{\boldsymbol{h}_t^{\text{prior}} = \mathrm{R}_\theta^{\text{prior}}\left(\boldsymbol{h}_{t-1}^{\text{prior}}, \mathbf{z}_{t-1}\right)}, \end{aligned} \tag{22}$$

where we introduce the hidden state $\boldsymbol{h}_t^{\text{P}}$ of the RNN used for propagating objects through time. Computation of the state is deterministic and therefore $p_\theta\left(\boldsymbol{h}_t^{\text{prior}} \mid \boldsymbol{h}_{t-1}^{\text{prior}}, \mathbf{z}_{t-1}\right)$ is a Dirac distribution centred on $\boldsymbol{h}_t^{\text{P}}$, which further simplifies computation. In the second line we also used the fact that object propagation depends only on the number of objects present at the previous time-step and it does not matter whether the objects were propagated from previous time-step or newly discovered.

8.2 Inference

To infer the latent variables, we have to invert the generative model. To do so, the inference model has to take into account the observations $\mathbf{x}_{1:T}$ and produce sequences of sets of latent variables $\mathbf{z}_{1:T}$. Similar to the generative process, inference uses a two-step approach, whereby it propagates objects

from previous time-steps and then tries to discover new objects. Let us start by defining the approximate posterior distribution over all latent variables.

$$
\begin{aligned}
&q_\phi(\mathbf{z}_{1:T}, n_{1:T}, k_{1:T} \mid \mathbf{x}_{1:T}) \\
&\quad= \prod_{t=1}^{T} q_\phi(\mathbf{z}_t, n_t, k_t \mid \mathbf{x}_{1:t}, \mathbf{z}_{1:t-1}, n_{1:t-1}, k_{1:t-1}) \\
&\quad= \prod_{t=1}^{T} q_\phi(\mathbf{z}_t, n_t, k_t \mid \mathbf{x}_t, \mathbf{z}_{1:t-1}, n_{1:t-1}, k_{1:t-1}) \\
&\quad= \prod_{t=1}^{T} q_\phi(\mathbf{z}_t, n_t, k_t \mid \mathbf{x}_t, \mathbf{z}_{t-1}, n_{t-1}, k_{t-1}, \boldsymbol{h}_t^q)|_{\boldsymbol{h}_t^q = R_\phi^q(\mathbf{z}_{t-1}, \boldsymbol{h}_{t-1}^q)} \\
&\quad= \prod_{t=1}^{T} q_\phi^D\left(\mathbf{z}_t^D, n_t \mid \mathbf{x}_t, \mathbf{z}_t^P, k_t\right) q_\phi^P\left(\mathbf{z}_t^P, k_t \mid \mathbf{x}_t, \mathbf{z}_{t-1}, k_{t-1} + n_{t-1}, \boldsymbol{h}_t^q\right) \\
&\quad= \prod_{t=1}^{T} \underbrace{q_\phi^D\left(\mathbf{z}_t^D, n_t \mid \mathbf{x}_t, \mathbf{z}_t^P, k_t\right)}_{\text{discovery}} \quad \underbrace{q_\phi^P\left(\mathbf{z}_t^P, k_t \mid \mathbf{x}_t, \mathbf{z}_{t-1}, m_{t-1}, \boldsymbol{h}_t^q\right)}_{\text{propagation}}
\end{aligned}
\tag{23}
$$

In the second line we use the product rule to rewrite the joint distribution as a temporal chain. In the third line we introduce the assumption that latent variables are conditionally independent of the past observations given the corresponding latent variables. In the fourth line we substituted the dependency on the history of latent variables by introducing a Markovian hidden state $\boldsymbol{h}_t^q$. We assume that the hidden state follows a Dirac distribution and therefore the associated integral collapses to a single value. In line five we split the latent variable $\mathbf{z}_t$ into the ones introduced by the discovery and propagation steps, respectively. We also omit the value of the hidden state to reduce clutter. In line six we explicitly condition the propagation step on the number of latent variables in the previous time-step.

We start by describing the propagation model q_ϕ^P. We would like to condition propagation of objects on other propagated objects at the current time-step, and therefore we decompose inference into a sequence of steps.

$$
\begin{aligned}
&q_\phi^P\left(\mathbf{z}_t^P, k_t \mid \mathbf{x}_t, \mathbf{z}_{t-1}, m_{t-1}, \boldsymbol{h}_t^q\right) \\
&\quad= \prod_{i=1}^{m_{t-1}} q_\phi^P\left(\mathbf{z}_t^{P,i}, b_t^{P,i} \mid \mathbf{x}_t, \mathbf{z}_{t-1}^i, \boldsymbol{h}_t^q, \mathbf{z}_t^{P,1:i-1}\right) \\
&\quad= \prod_{i=1}^{m_{t-1}} q_\phi^P\left(\mathbf{z}_t^{P,i}, b_t^{P,i} \mid \mathbf{x}_t, \mathbf{z}_{t-1}^i, \boldsymbol{h}_t^q, \boldsymbol{h}_t^{P,i}\right)|_{\boldsymbol{h}_t^{P,i} = R_\phi^P\left(\mathbf{z}_t^{P,i-1}, b_t^{P,i-1} \boldsymbol{h}_t^{P,i-1}\right)}
\end{aligned}
\tag{24}
$$

$\mathbf{z}_t^{P,0}$ and $bt^{P,0}$ start as zeros and $\boldsymbol{h}_t^{P,0}$ is randomly initialised and learnable.

It is worth noting that $m_t = \sum_{i=1}^{m_{t-1}} b_t^{P,i} \leqslant m_{t-1}$, therefore propagation can forget objects, but it cannot introduce any new ones. At the end of the propagation stage, the hidden state $\boldsymbol{h}_t^{P,k_t}$ should contain information about all the propagated objects and can be used to condition the discovery stage.

To handle object discovery, we follow the implementation of AIR, with the difference that we condition on the hidden state of the propagation phase $\boldsymbol{h}_t^{P,k_t}$ and that there is a variable maximum number of steps. It defines a sequential inference procedure, where the model attends to and explains different regions of the image, one at a time. The approximate posterior distribution is defined by Equations (4) to (6). We redefine it here to account for temporal dependencies and conditioning. The approximate posterior distribution for discovered objects is given by

$$
\begin{aligned}
&q_\phi^D\left(\boldsymbol{z}_t^D, n_t \mid \boldsymbol{x}_t, \boldsymbol{z}_t^P, k_t\right) \\
&= q_\phi^D\left(\boldsymbol{z}_t^D, n_t \mid \boldsymbol{x}_t, \boldsymbol{h}_t^{P,k_t}, k_t\right) \\
&= q_\phi^D\left(b_t^{D,n_t+1} = 0 \mid \boldsymbol{x}_t, \boldsymbol{z}_t^{D,1:n_t}, \boldsymbol{h}_t^{P,k_t}, k_t\right) \prod_{i=1}^{n_t} q_\phi^D\left(b_t^{D,i}, \boldsymbol{z}_t^{D,i} \mid \boldsymbol{x}_t, \boldsymbol{z}_t^{D,1:i-1}, \boldsymbol{h}_t^{P,k_t}, k_t\right) \\
&= q_\phi^D\left(b_t^{D,n_t+1} = 0 \mid \boldsymbol{x}_t, \boldsymbol{h}_t^{D,n_t}, \boldsymbol{h}_t^{P,k_t}, k_t\right) \prod_{i=1}^{n_t} q_\phi^D\left(b_t^{P,i}, \boldsymbol{z}_t^{D,i} \mid \boldsymbol{x}_t, \boldsymbol{h}_t^{D,i-1}, \boldsymbol{h}_t^{P,k_t}, k_t\right),
\end{aligned}
\tag{25}
$$

where

$$
\begin{aligned}
&\boldsymbol{h}_t^{D,i} = R_\phi^D\left(\boldsymbol{z}_t^{D,i}, \boldsymbol{h}_t^{D,i-1}\right), \\
&q_\phi^D\left(b_t^{D,i} \mid \boldsymbol{x}_t, \boldsymbol{z}_t^{D,1:i}, \boldsymbol{h}_t^{P,k_t}, k_t\right) = \begin{cases} q^D\left(b_t^{D,i} \mid \boldsymbol{x}_t, \boldsymbol{z}_t^{D,1:i}, \boldsymbol{h}_t^{P,k_t}\right), & \text{if } i \leqslant N - k_t \\ 0, & \text{otherwise} \end{cases}, \\
&q_\phi^D\left(b_t^{D,i} = 0 \mid \boldsymbol{x}_t, \boldsymbol{z}_t^{D,1:i}, \boldsymbol{h}_t^{P,k_t}, k_t\right) = 1 - q_\phi^D\left(b_t^{D,i} \mid \boldsymbol{x}_t, \boldsymbol{z}_t^{D,1:i}, \boldsymbol{h}_t^{P,k_t}, k_t\right).
\end{aligned}
\tag{26}
$$

8.3 Variational Lower Bound

Given the specification of the prior distribution and the inference model, we can derive the evidence lower bound (ELBO).

$$
\begin{aligned}
\mathcal{L}_{APDR}(\phi, \theta) &= \mathbb{E}_{q_\phi}\left[\log \frac{p_\theta(\boldsymbol{z}_{1:T}, n_{1:T}, k_{1:T}, \boldsymbol{x}_{1:T})}{q_\phi(\boldsymbol{z}_{1:T}, n_{1:t}, k_{1:t} \mid \boldsymbol{x}_{1:T})}\right] \\
&= \mathbb{E}_{q_\phi}[\log p_\theta(\boldsymbol{x}_{1:T} \mid \boldsymbol{z}_{1:T})] \\
&\qquad - \mathrm{KL}\big(q_\phi(\boldsymbol{z}_{1:T}, n_{1:T}, k_{1:T} \mid \boldsymbol{x}_{1:T}) \,\|\, p_\theta(\boldsymbol{z}_{1:T}, n_{1:T}, k_{1:T})\big).
\end{aligned}
\tag{27}
$$

The KL divergence term cannot be computed analytically due to non-linear temporal dependencies, and we have to resort to a Monte-Carlo approxima-

tion. The latter can be rewritten as a sum of per time-step components. Let $\boldsymbol{\beta} = \{\mathbf{z}, n, k, \boldsymbol{h}\}$ denote all latent variables and all hidden states. We have that

Make sure the below is correct!

$$
\begin{aligned}
&\mathrm{KL}\big(q_\phi(\mathbf{z}_{1:T}, n_{1:T}, k_{1:T} \mid \mathbf{x}_{1:T}) \parallel p_\theta(\mathbf{z}_{1:T}, n_{1:T}, k_{1:T})\big) \\
&= \mathrm{KL}\big(q_\phi(\boldsymbol{\beta}_{1:T} \mid \mathbf{x}_{1:T}) \parallel p_\theta(\boldsymbol{\beta}_{1:T})\big) \\
&= \int q_\phi(\boldsymbol{\beta}_{1:T} \mid \mathbf{x}_{1:T}) \log \frac{q_\phi(\boldsymbol{\beta}_{1:T} \mid \mathbf{x}_{1:T})}{p_\theta(\boldsymbol{\beta}_{1:T})} \, \mathrm{d}\boldsymbol{\beta}_{1:T} \\
&= \int \prod_{t=1}^{T} \left[q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_{1:t}, \boldsymbol{\beta}_{1:t-1}) \log \frac{q_\phi(\boldsymbol{\beta}_{1:T} \mid \mathbf{x}_{1:T})}{p_\theta(\boldsymbol{\beta}_{1:T})} \right] \mathrm{d}\boldsymbol{\beta}_{1:T} \qquad (28) \\
&= \int \prod_{t=1}^{T} \left[q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) \log \frac{q_\phi(\boldsymbol{\beta}_{1:T} \mid \mathbf{x}_{1:T})}{p_\theta(\boldsymbol{\beta}_{1:T})} \right] \mathrm{d}\boldsymbol{\beta}_{1:T} \\
&= \int \prod_{t=1}^{T} q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) \left[\sum_{t=1}^{T} \log \frac{q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1})}{p_\theta(\boldsymbol{\beta}_t \mid \boldsymbol{\beta}_{t-1})} \right] \mathrm{d}\boldsymbol{\beta}_{1:T}
\end{aligned}
$$

In the fourth line we expanded the first q term by using the product rule. In line five we make use of the Markovian property of the aggregate hidden state $\boldsymbol{\beta}$. In line six we expand the logarithmic argument by using the product rule. To reduce clutter, we substitute the logarithm as

$$
l(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) = \log \frac{q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1})}{p_\theta(\boldsymbol{\beta}_t \mid \boldsymbol{\beta}_{t-1})}. \qquad (29)
$$

We obtain

$$
\begin{aligned}
&= \int \prod_{t=1}^{T} q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) \left[\sum_{t=1}^{T} l(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) \right] \mathrm{d}\boldsymbol{\beta}_{1:T} \\
&= \sum_{\tau=1}^{T} \int l(\boldsymbol{\beta}_\tau \mid \mathbf{x}_\tau, \boldsymbol{\beta}_{\tau-1}) \, q_\phi(\boldsymbol{\beta}_{1:T} \mid \mathbf{x}_{1:T}) \, \mathrm{d}\boldsymbol{\beta}_{1:T} \\
&= \sum_{t=1}^{T} \int l(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) \underbrace{q_\phi(\boldsymbol{\beta}_{1:t-2} \mid \mathbf{x}_{1:t-2})}_{\text{marginal over } \beta_{t-2}} \\
&\qquad q_\phi(\boldsymbol{\beta}_{t-1:t} \mid \mathbf{x}_{t-1:t}, \boldsymbol{\beta}_{t-2}) \underbrace{q_\phi(\boldsymbol{\beta}_{t+1:T} \mid \mathbf{x}_{t+1:T}, \boldsymbol{\beta}_t)}_{\text{after integrating}=1} \mathrm{d}\boldsymbol{\beta}_{1:T} \qquad (30) \\
&= \sum_{t=1}^{T} \int l(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1}) \, q_\phi(\boldsymbol{\beta}_{1:t} \mid \mathbf{x}_{1:t}) \, \mathrm{d}\boldsymbol{\beta}_{1:t} \\
&= \sum_{t=1}^{T} \int \left[\prod_{\tau=1}^{t} q_\phi(\boldsymbol{\beta}_\tau \mid \mathbf{x}_\tau, \boldsymbol{\beta}_{\tau-1}) \right] \log \frac{q_\phi(\boldsymbol{\beta}_t \mid \mathbf{x}_t, \boldsymbol{\beta}_{t-1})}{p_\theta(\boldsymbol{\beta}_t \mid \boldsymbol{\beta}_{t-1})} \, \mathrm{d}\boldsymbol{\beta}_{1:t} \\
&\approx \sum_{t=1}^{T} \mathrm{KL}_t(\cdot).
\end{aligned}
$$

In the first line we substitute from Equation (29). In line two we move the summation out of the integral and rearrange the terms. In line three and four we decompose the joint approximate posterior into three conditional distributions over temporal variables, grouped by time. Note that we can remove the last term since it integrates to one, the fact we use in line five. In line six, we back-substitute from Equation (29). The obtained KL-divergence is intractable, but can be approximated by sampling. Specifically, we can rewrite this it as a sum of temporal components, which we show in line seven, and describe in more detail below.

$$\mathrm{KL}_t(\cdot) = \frac{1}{L}\sum_{i=1}^{L} \log \frac{q_\phi\left(\mathbf{z}_t^{(l)}, n_t^{(l)}, k_t^{(l)} \mid \mathbf{x}_t, \mathbf{z}_{t-1}^{(l)}, n_{t-1}^{(l)}, k_{t-1}^{(l)}, \boldsymbol{h}_t^{q,\,(l)}\right)}{p_\theta\left(\mathbf{x}_t, \mathbf{z}_t^{(l)}, n_t^{(l)}, k_t^{(l)} \mid \mathbf{z}_{1:t-1}^{(l)}, n_{1:t-1}^{(l)}, k_{1:t-1}^{(l)}\right)}, \tag{31}$$

with

$$\mathbf{z}_{1:T}^{(l)}, n_{1:T}^{(l)}, k_{1:T}^{(l)} \sim q_\phi(\mathbf{z}_{1:T}, n_{1:T}, k_{1:T} \mid \mathbf{x}_{1:T}),$$
$$\boldsymbol{h}_t^{q,\,(l)} = R_\phi^q\left(\mathbf{z}_{t-1}(l), \boldsymbol{h}_{t-1}^{q,\,(l)}\right)$$

All latent variables can be obtained by ancestral sampling, since in MC approximation sampling at time-step t corresponds to multiplication by the value of the probability of the sample, or $q_\phi(\boldsymbol{\beta}_\tau \mid \mathbf{x}_\tau, \boldsymbol{\beta}_{\tau-1})$, *cf.* line six of Equation (30). The above approximation allows us to rewrite the ELBO as a sum of per time-step components, namely

$$\mathcal{L}_{\mathrm{APDR}}(\phi, \theta) \approx \sum_{t=1}^{T} \mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)] - \mathrm{KL}_t(\cdot), \tag{32}$$

which can be useful if we wanted to use a particle filter estimator of the ELBO, *cf.* Section 6.

8.4 Algorithm

Algorithm 1 details estimation of the variational lower-bound for APDR. After initialising latent variables and hidden states, it loops over temporal indices. It generates all latent variables for time-step t, and then updates or initialises hidden states for the propagation prior. Finally, it evaluates log-probabilities of previously generated latent variables under the approximate posterior and the prior. Note that it uses the partitioned latent variables ($\mathbf{z}_t^D, \mathbf{z}_t^P$ and n_t, k_t) to compute the probabilities under corresponding pdfs. Evaluating probabilities under the propagation models (both prior and posterior) is particularly cumbersome, since for every latent variable at time t we have to identify preceding hidden states and latent variables. To do so, we initialise an id for every latent variable introduced by the discovery posterior and we maintain it whenever we propagate the latent variable.

Algorithm 1: APDR lower-bound estimation

Input : Image sequence $\mathbf{x}_{1:T}$

$\mathcal{L}_{APDR} = 0$ // Initialise ELBO estimate to zero.

$m_0 = 0$ // Initialise the object count to zero.

$\mathbf{z}_0, \boldsymbol{h}_0^q, \boldsymbol{h}_0^{prior} = \text{init\_learnable}()$

for $t \in [1, \ldots, T]$ **do**

/* Generate latent variables */

$\boldsymbol{h}_t^q = R_\phi^q\left(\mathbf{z}_{t-1}, \boldsymbol{h}_{t-1}^q\right)$

$\mathbf{z}_t^P, k_t \sim q_\phi^P\left(\mathbf{z}_t^P, k_t \mid \mathbf{x}_t, \mathbf{z}_{t-1}, m_{t-1}, \boldsymbol{h}_t^q\right)$

$\mathbf{z}_t^D, n_t \sim q_\phi^D\left(\mathbf{z}_t^D, n_t \mid \mathbf{x}_t, \mathbf{z}_t^P, k_t\right)$

$\mathbf{z}_t = \mathbf{z}_t^P \cup \mathbf{z}_t^D$

$m_t = k_t + n_t$

/* Update the propagation prior. */

for $\mathbf{z}_t^j \in \mathbf{z}_t^P$ **do**

$\boldsymbol{h}_t^{prior,j} = R_\theta^{prior}\left(\mathbf{z}_{t-1}^j, \boldsymbol{h}_{t-1}^{prior,j}\right)$

/* Initialise hidden states of the prior for newly discovered objects. */

for $\mathbf{z}_t^i \in \mathbf{z}_t^D$ **do**

$\boldsymbol{h}_t^{prior,i} = \boldsymbol{h}_0^{prior}$

/* Estimate the log-likelihood. */

$\mathcal{L}_t^x = \log p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)$

/* Estimate the KL-divergence. */

$\mathcal{L}_t^k = \log q_\phi^P(k_t \mid \cdot) - \log p_\theta^P(k_t \mid \cdot)$

$\mathcal{L}_t^P = \log q_\phi^P\left(z_t^P \mid k_t, \cdot\right) - \log p_\theta^P\left(z_t^P \mid k_t, \cdot\right)$

$\mathcal{L}_t^n = \log q_\phi^D(n_t \mid \cdot) - \log p_\theta^D(n_t)$

$\mathcal{L}_t^D = \log q_\phi^D\left(z_t^D \mid n_t, \cdot\right) - \log p_\theta^D\left(z_t^D \mid n_t, \cdot\right)$

$KL_t = \mathcal{L}_t^k + \mathcal{L}_t^P + \mathcal{L}_t^n + \mathcal{L}_t^D$

/* Update the ELBO estimate. */

$\mathcal{L}_{APDR} = \mathcal{L}_{APDR} + \mathcal{L}_t^x - KL_t$

Output: $\mathcal{L}_{APDR}, \mathbf{z}_{1:T}, n_{1:T}, k_{1:T}, q_\phi(\mathbf{z}_{1:T}, n_{1:T}, k_{1:T} \mid \mathbf{x}_{1:T})$

The resulting estimate of the ELBO is differentiable in the continuous latent variables, but non-differentiable in the discrete latent variables. We use the reparametrisation trick to compute the gradient of the ELBO estimate with respect to parameters of all continuous pdfs, and we use the returned latent variables and probabilities to estimate gradients w. r. t. all discrete pmfs with VIMCO.

8.5 Related Work

IFHMM Our model decomposes a sequence of images into separate objects that evolve over time, and it uses discrete latent variables to encode presence or absence of an object. We parametrise all discrete latent variables as sums of Bernoulli random variables, and we use this parametrisation explicitly in the propagation part of the prior and in both discovery and propagation parts of the approximate posterior. This is similar to the Infinite Factorial Hidden Markov Model (IFHMM) model introduced by Gael, Teh, and Ghahramani, 2009. IFHMM is a non-parametric Bayesian model, which realises the Indian Buffer Process for temporal data.

write more about it

HDDP Neil's friend's paper; they use DP for unsupervised detection and tracking of multiple objects in videos; they perform background-subtraction to handle real data.

HART Supervised and single object but uses attention, explicitly maintains location as a hidden variable and propagates state of an object over time, similarly to the propagation phase of APDR.

Anything else?

References

Eslami, S. M. Ali et al. (2016). "Attend, Infer, Repeat: Fast Scene Understanding with Generative Models". In: *NIPS*. arXiv: 1603.08575. URL: http://arxiv.org/abs/1603.08575.

Gael, Jurgen Van, Yee Whye Teh, and Zoubin Ghahramani (2009). "The Infinite Factorial Hidden Markov Model". In: *NIPS*, pp. 1697–1704. ISBN: 9781605609492. URL: https://papers.nips.cc/paper/3518-the-infinite-factorial-hidden-markov-model.

Jang, Eric, Shixiang Gu, and Ben Poole (2016). "Categorical Reparameterization with Gumbel-Softmax". In: ISSN: 1611.01144. arXiv: 1611.01144. URL: http://arxiv.org/abs/1611.01144.

Maddison, Chris J., Dieterich Lawson, et al. (2017). "Filtering Variational Objectives". In: *NIPS*. arXiv: 1705.09279. URL: http://arxiv.org/abs/1705.09279.

Maddison, Chris J., Andriy Mnih, and Yee Whye Teh (2016). "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: arXiv: 1611.00712. URL: http://arxiv.org/abs/1611.00712.

Mnih, Andriy and Karol Gregor (2014). "Neural Variational Inference and Learning in Belief Networks". In: *ICML*. ISBN: 9781634393973. arXiv: arXiv:1402.0030v2. URL: http://arxiv.org/abs/1402.0030.

Tucker, George et al. (2017). "REBAR: Low-variance, unbiased gradient estimates for discrete latent variable models". In: *NIPS*. arXiv: 1703.07370. URL: http://arxiv.org/abs/1703.07370.

Williams, Ronald J. (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Mach. Learn.* 8.3-4, pp. 229–256. ISSN: 0885-6125. DOI: 10.1007/BF00992696. URL: http://link.springer.com/10.1007/BF00992696.