Online Structure Learning with Dirichlet Processes through Message Passing

Bart van Erp[1,2], Wouter W. L. Nuijten[1], and Bert de Vries[1,2,3]

[1] Eindhoven University of Technology, 5612 AP Eindhoven, The Netherlands
[2] Lazy Dynamics, 5611 XD Eindhoven, The Netherlands
[3] GN Hearing, 5612 AB Eindhoven, The Netherlands

Abstract. Generative or probabilistic modeling is crucial for developing intelligent agents that can reason about their environment. However, designing these models manually for complex tasks is often infeasible. Structure learning addresses this challenge by automating model creation based on sensory observations, balancing accuracy with complexity. Central to structure learning is Bayesian model comparison, which provides a principled framework for evaluating models based on their evidence. This paper focuses on model expansion and introduces an online message passing procedure using Dirichlet processes, a prominent prior in non-parametric Bayesian methods. Our approach builds on previous work by automating Bayesian model comparison using message passing based on variational free energy minimization. We derive novel message passing update rules to emulate Dirichlet processes, offering a flexible and scalable method for online structure learning. Our method generalizes to arbitrary models and treats structure learning identically to state estimation and parameter learning. The experimental results validate the effectiveness of our approach on an infinite mixture model.

Keywords: Dirichlet processes · Factor graphs · Infinite mixture model · Message passing · Probabilistic inference · Scale factors · Structure learning.

1 Introduction

The task of generative or probabilistic modeling is fundamental in developing intelligent agents capable of reasoning about their environment. However, it is often infeasible for human engineers to manually design these models for complex tasks because of the involved intricacies. Structure learning addresses this challenge by automating the construction of models based on sensory observations, thus alleviating the burden on human engineers.

Structure learning is encapsulated in the task of Bayesian model comparison, which provides a principled framework for comparing models based on their evidence. This process facilitates the identification of better models that are either smaller or larger than a baseline, known, respectively, as model reduction [4, 14, 15] and model expansion [16, 31]. These techniques are critical for refining

and optimizing models, thus enhancing their performance and applicability in various tasks. This paper will focus on model expansion in particular.

In [9] the tasks of Bayesian model comparison [18], selection and combination [25] have been automated using message passing based on variational free energy minimization. This paper extends this set of methods with Dirichlet processes [6, 10, 26, 33], which are one of the most established priors in non-parametric Bayes. Effectively, we present an online message passing procedure based on Dirichlet processes which enables the model to grow automatically over time, providing a natural trade-off between model accuracy and complexity.

Our approach shows similarities with [36], yet offers more flexibility as a result of our commitment to message passing. Compared to [23], our approach leverages scale factors [27, 29, Ch.6] to track the model evidence rather than performing a partial mean-field approximation. In contrast to [16, 31] our approach is based on non-parametric priors, allowing for a message passing-based treatment of both state estimation, parameter learning and structure adaptation, which is not limited to discrete-space models.

This paper presents a novel and principled approach to online structure learning using message passing. Specifically, we make the following contributions:

- We present a generic and modular approach similar to the sequential updating and greedy search algorithm [36] for online structure learning utilizing Dirichlet processes;
- We derive novel message passing update rules to emulate Dirichlet processes, based on the mixture node recently introduced in [9];
- We demonstrate our approach on an infinite mixture model [28], with the potential for generalization to arbitrary models.

To provide a solid foundation for all readers, Section 2 introduces Forney-style factor graphs and message passing, the core methodology behind this paper. Readers unfamiliar with this methodology and its benefits are encouraged not to skip this section. Throughout the subsequent sections, we use the infinite mixture model [28], as specified in Section 3, as a running example to elucidate our approach. It should be noted, however, that the methods presented in this paper can be easily generalized to more complex graphs due to the inherent modularity of our approach. Using this model, Section 4 details how inference can be executed to ensure the message passing procedure emulates a Dirichlet process, facilitating online structure learning. The experimental results validating our approach are presented in Section 5, and Section 6 follows with a discussion of the presented approach, concluding the paper.

2 Technical background

This section provides a concise review of factor graphs and message passing algorithms, essential for understanding our core contributions. For a deeper understanding, we provide references rather than an exhaustive review. In Section 2.1,

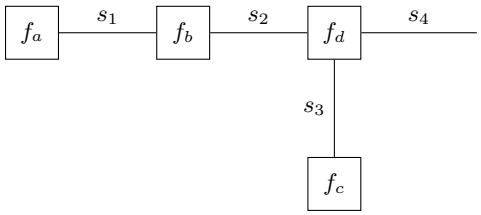


Fig. 1: A Forney-style factor graph representation of the factorization in (2).

we introduce factor graphs for visualizing factorizable probabilistic models, after which Section 2.2 covers efficient probabilistic inference through message passing. Section 2.3 explains how to track model evidence locally with message passing using scale factors.

2.1 Forney-style factor graphs

A factor graph is a type of probabilistic graphical model. We use the Forney-style factor graph (FFG) framework from [11] with notations from [21] to visualize our models. An FFG represents a factorized function:

$$f(s) = \prod_{a \in \mathcal{V}} f_a(s_a), \tag{1}$$

where s includes all variables, and $s_a \subseteq s$ includes the variables of factor f_a. In an FFG, nodes ($\mathcal{V}$) represent factors, and edges ($\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$) represent variables. An edge connects to a node if the variable is an argument of the factor at that node. The edges connected to node $a \in \mathcal{V}$ are denoted by $\mathcal{E}(a)$, and the nodes connected to edge $i \in \mathcal{E}$ are denoted by $\mathcal{V}(i)$. For example, consider

$$f(s_1, s_2, s_3, s_4) = f_a(s_1) f_b(s_1, s_2) f_c(s_3) f_d(s_2, s_3, s_4), \tag{2}$$

which represents a factorization whose FFG representation is shown in Figure 1. For a detailed review of factor graphs, see [21, 22].

2.2 Sum-product message passing

Consider the normalized probabilistic model

$$p(y, s) = \prod_{a \in \mathcal{V}} f_a(y_a, s_a), \tag{3}$$

where y represents observed variables and s represents latent variables. The subset $y_a \subseteq y$ can be empty, such as in prior distributions. Upon observing realizations $\hat{y}$, the model $p(y = \hat{y}, s)$ becomes unnormalized. Probabilistic inference involves computing the posterior distribution $p(s \mid y = \hat{y})$ and the model evidence $p(y = \hat{y})$ as the decomposition $p(y = \hat{y}, s) = p(s \mid y = \hat{y}) p(y = \hat{y})$.

Consider integrating over all variables in the model except s_j as $\int p(y = \hat{y}, s)\,\mathrm{d}s_{\setminus j}$. This integration can be performed through smaller local computations, whose results are termed messages, which propagate over the graph edges. The sum-product message $\vec{\mu}_{s_j}(s_j)$ flowing from node $f_a(y_a = \hat{y}_a, s_a)$ with incoming messages $\vec{\mu}_{s_i}(s_i)$ is given by [19]

$$\vec{\mu}_{s_j}(s_j) = \int f_a(y_a = \hat{y}_a, s_a) \prod_{\substack{i \in \mathcal{E}(a) \\ i \neq j}} \vec{\mu}_{s_i}(s_i)\,\mathrm{d}s_{a\setminus j}. \tag{4}$$

Edges in the graph are represented by directed arrows to distinguish between forward ($\vec{\mu}_{s_j}(s_j)$) and backward ($\overleftarrow{\mu}_{s_j}(s_j)$) messages. For acyclic models, the global integration reduces to the product of messages

$$\int p(y = \hat{y}, s)\,\mathrm{d}s_{\setminus j} = \vec{\mu}_{s_j}(s_j)\overleftarrow{\mu}_{s_j}(s_j). \tag{5}$$

Posterior distributions can be obtained by normalizing the resulting product. The computed normalization constant represents the model evidence. For derivations of the message passing update rules, see [37]. Variations of this approach also yields alternative algorithms such as variational message passing [35], expectation propagation [24], expectation maximization [8], and hybrid algorithms.

2.3 Scale factors

The previously discussed integration $\int p(y = \hat{y}, s)\,\mathrm{d}s_{\setminus j}$ can be expressed as

$$\int p(y = \hat{y}, s)\,\mathrm{d}s_{\setminus j} = p(y = \hat{y}) \int p(s \mid y = \hat{y})\,\mathrm{d}s_{\setminus j} = p(y = \hat{y})p(s_j \mid y = \hat{y}), \tag{6}$$

where $p(s_j \mid y = \hat{y})$ is the marginal distribution of s_j. This means that the product of two colliding sum-product messages $\vec{\mu}_{s_j}(s_j)\overleftarrow{\mu}_{s_j}(s_j)$ in an acyclic graph yields the scaled marginal distribution $p(y = \hat{y})p(s_j \mid y = \hat{y})$. Thus, we can obtain both the normalized posterior $p(s_j \mid y = \hat{y})$ and the model evidence $p(y = \hat{y})$ at any edge or node in the graph.

Theorem 1. *[9, Theorem 1] Consider an acyclic Forney-style factor graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The model evidence of the corresponding model $p(y = \hat{y}, s)$ can be computed at any edge in the graph as $\int \vec{\mu}_{s_j}(s_j)\overleftarrow{\mu}_{s_j}(s_j)\,\mathrm{d}s_j$ for all $j \in \mathcal{E}$ and at any node in the graph as $\int f_a(y_a = \hat{y}_a, s_a) \prod_{i \in \mathcal{E}(a)} \vec{\mu}_{s_i}(s_i)\,\mathrm{d}s_a$ for all $a \in \mathcal{V}$.*

This local computation of model evidence is enabled by the scaling of messages resulting from the equality in (4). Consequently, the messages $\vec{\mu}_{s_j}(s_j)$ can be decomposed as

$$\vec{\mu}_{s_j}(s_j) = \vec{\beta}_{s_j}\vec{p}_{s_j}(s_j), \tag{7}$$

where $\vec{p}_{s_j}(s_j)$ is the normalized probability distribution of the message $\vec{\mu}_{s_j}(s_j)$, and $\vec{\beta}_{s_j}$ is the scaling factor [29, Ch.6], [27]. These scale factors serve as local summaries of the model evidence passed along the graph.

3 Model specification

In this section, we describe the probabilistic model that underpins our novel inference approach detailed in Section 4. As a running example, we employ the infinite mixture model [28]. This model leverages the unique properties of Dirichlet processes, allowing the model to expand dynamically over time. Consequently, it serves as an ideal and principled example for structure learning.

The infinite mixture model works as follows. Consider a single observation y_n, which is modelled by a likelihood model $p(y_n \mid \theta, c_n)$, with parameters θ. The model assumes multiple possible options or regimes for the parameters depending on the cluster assignment probability c_n. This cluster assignment probability c_n comprises a 1-of-K binary vector with elements $c_{nk} \in \{0, 1\}$ constrained by $\sum_{k=1}^{K} c_{nk} = 1$. Depending on the class, the observation is modelled by a different set of parameters. When the k^{th} class is active, the corresponding set of parameters is given by θ_k. As a result, the likelihood model can be further factorized as

$$p(y_n \mid \theta, c_n) = \prod_{k=1}^{\infty} p(y_n \mid \theta_k)^{c_{nk}}. \tag{8}$$

The infinite mixture model assumes we have an infinite amount of classes ($K = \infty$). Although this might seem computationally intractable, in practice only a limited number of classes is active as we will show in Section 4. The model's strength lies in its ability to grow the number of active classes over time, providing opportunities to expand the model in a principled manner.

In addition to the likelihood model, we define the prior over the cluster parameters as the base distribution G_0 as

$$p(\theta_k) = G_0(\theta_k) \quad \forall\, k. \tag{9}$$

Here independence across the clusters is implied by the characterization of [20] as

$$p(\theta) = \prod_{k=1}^{\infty} p(\theta_k), \tag{10}$$

and will also result into independence across the posteriors over the clusters [34].

The cluster assignment probabilities c_n are modeled using a categorical distribution

$$p(c_n \mid \pi) = \text{Cat}(c_n \mid \pi), \tag{11}$$

with event probabilities π. The prior on the event probabilities is in our case defined as

$$p(\pi) = \lim_{K \to \infty} \text{Dir}\left(\pi \,\middle|\, \frac{\alpha}{K} 1_K\right), \tag{12}$$

with α representing the concentration parameter and 1_K denoting a vector of ones of length K. Alternative definitions are also possible, e.g. using the Griffiths-Engen-McCloskey (GEM) distribution or using the stick-breaking representation, however, for ease of inference in Section 4 we use the former. Together,

the base distribution G_0 and the concentration parameter α characterize the underlying Dirichlet process.

With all the individual elements identified, the full generative model of the infinite mixture model can now be constructed for multiple observations. Given N observations $y = \{y_1, y_2, \ldots, y_N\}$ the total model factorizes as

$$p(y, \theta, c, \pi) = \underbrace{p(\pi)}_{(12)} \prod_{k=1}^{\infty} \underbrace{p(\theta_k)}_{(9)} \prod_{n=1}^{N} \underbrace{p(y_n \,|\, \theta, c_n)}_{(8)} \underbrace{p(c_n \,|\, \pi)}_{(11)}. \tag{13}$$

4 Probabilistic inference

With the model described in the previous section, we will now show how we can perform inference in this model. Specifically, we are interested in computing the marginal posterior distributions $p(\pi \,|\, y_{\leq n})$ and $p(\theta_k \,|\, y_{\leq n}) \,\forall\, k$ in the infinite mixture model. We focus here on online inference, where this inference task is performed using streaming data, as we highlight the importance of in-the-field structure learning. Ideally, we wish to solve the discrete equivalent of the Chapman-Kolmogorov integral [32, Ch.4]

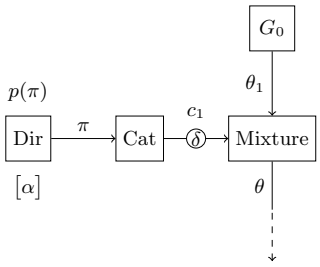
$$p(\theta, \pi \,|\, y_{\leq n}) \propto \sum_{c_n} p(y_n \,|\, \theta, c_n)\, p(c_n \,|\, \pi)\, p(\theta, \pi \,|\, y_{<n}) \tag{14}$$

recursively. However, the infinite dimensionality of c_n results in intractable inference. To circumvent this problem, all inactive components where the posterior beliefs over the parameters have not yet been updated from the prior belief, are grouped together. The components or cluster with indices $k > K^*$, where K^* denotes the number of active components, are grouped into a single component with concentration parameter $\lim_{K\to\infty} \sum_{k=K^*+1}^{K} \alpha/K = \alpha$. This particular grouping is very beneficial as the problem can now be tackled as a standard model comparison task as in [9].

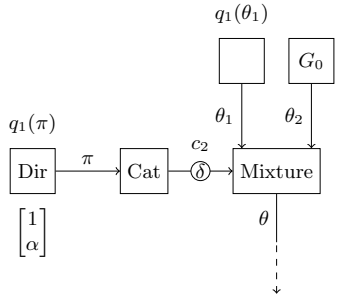
Furthermore, to limit the number of excitable components per observation, the class label c_n is constrained to correspond to a single class, such that each data point can only belong to a single class and therefore has the potential to initiate no more than a single component, as described in [36]. This constraint is reflected by constraining the approximate marginal distribution $q(c_n)$ [37] to

$$q(c_n) = \delta[c_n - e_k], \qquad \text{s.t. } k = \arg\max_k \vec{\mu}_{c_n}(c_n = e_k)\vec{\mu}_{c_n}(c_n = e_k), \tag{15}$$

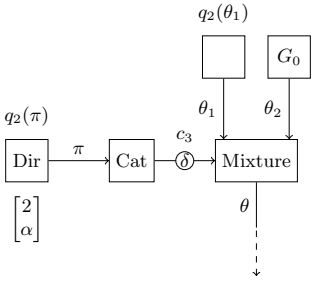
where $\delta[\cdot]$ denotes the Kronecker delta function and where we pick the component c_n as the maximum a posteriori estimate. This is similar to the Bayesian model selection setup as described in [9, Sec.5.2]. Using the Chinese restaurant metaphor, this constraint enforces that every customer can only sit at a single table at once. If we would not enforce this constraint, then there would always be a non-zero probability of the observation originating from the group of inactive components, which would be sufficient to activate a new component for each observation.



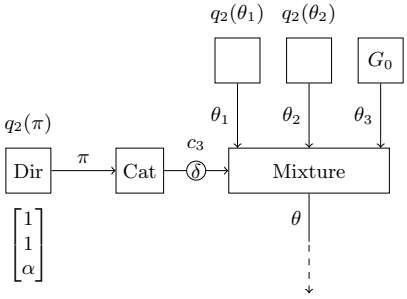
(a) Factor graph visualization for the first observation.



(b) Factor graph visualization for the second observation, where the first observation has initiated a new component.



(c) Factor graph visualization for the third observation, where the first and second observation are assigned to the same component.



(d) Factor graph visualization for the third observation, where the first and second observation have initiated distinct components.

Fig. 2: Factor graphs of the initial time slices of the infinite mixture model of Section 3. The edge denoted by θ here denotes the active or selected parameter settings. The edges connected to θ have been dashed to highlight its extensibility towards arbitrary observation models. The parameter vector below the Dir-node denotes the simplified vector of concentration parameters.

Based on this constraint, we approximate the marginal posterior distributions over $\theta_k \,\forall\, k$ and π with approximate posterior distributions $q_n(\theta_k)$ and $q_n(\pi)$, where the subscript n explicitly indexes the latest observation. Online inference proceeds using the iterative update procedure for θ_k as

$$q_n(\theta_k) \propto \begin{cases} p(y_n \,|\, \theta_k) q_{n-1}(\theta_k), & \text{if } q(c_n) = \delta[c_n - e_k], \\ q_{n-1}(\theta_k), & \text{otherwise,} \end{cases} \tag{16}$$

which effectively states that only the parameters gets updated which have been most likely to have generated the data. The posterior belief over π gets updated

as

$$q_n(\pi) \propto q_{n-1}(\pi) \underbrace{\sum_{c_n} q(c_n) p(c_n \,|\, \pi)}_{\overleftarrow{\mu}_\pi(\pi)} . \tag{17}$$

The initial conditions of this recursion find their origin in the model specification and are specified by

$$q_0(\theta_k) = p(\theta_k), \tag{18a}$$

$$q_0(\pi) = p(\pi). \tag{18b}$$

The above inference procedure can be automated using an adapted version of the mixture node from [9] as presented in Table 1. Effectively this node internally computes the model evidences of the individual combinations of inputs and output using the scale factors from Section 2.3, which are an indicator for how likely a data point y_n originated from one particular set of parameters θ_k. Through normalization of these evidences and together with the prior on the class label c_n, one can obtain the posterior distribution of the class label. In comparison to the mixture node as introduced in [9] the only adaptation occurs in the backward messages towards the parameters. This adaptation entails that only the parameters of the active component are being updated. Figure 2 visualizes the factor graphs corresponding to the initial time slices of the online training procedure. From this figure it can also be seen how the mixture node of [9] be used to represent the infinite mixture model.

The biggest benefit of this approach is that the system is inherently modular. The likelihood and priors can be extended to arbitrarily complex or hierarchical models to model more complex phenomena. By adding a temporal dependency $p(c_n \,|\, c_{n-1})$ to the model, one effectively creates a sticky Dirichlet process [12,13]. With the message passing updates rules from Table 1 together with rules derived in earlier works, e.g. [21, 27], one can build arbitrarily complex graphs tailored to any problem.

5 Experiments

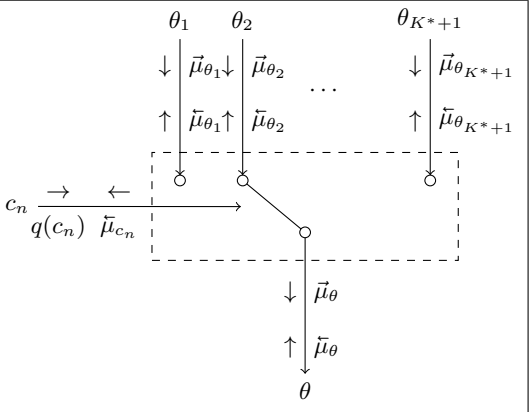
All experiments have been performed using the scientific programming language `Julia` [5] with the state-of-the-art probabilistic programming package `RxInfer.jl` [2]. The mixture node specified in Table 1 has been integrated in its dependency `ReactiveMP.jl` [1,3]. In addition to the results presented in the upcoming subsections, interactive `Pluto.jl` notebooks are made available online[4], allowing the reader to change hyperparameters in real-time.

For online learning of the infinite mixture model as described in Section 4, we generate observations from a two-dimensional normal mixture model with 8 clusters. As a model for these generated observations, we pick the infinite mixture model of (13). Here, the likelihood model is set to $p(y_n \,|\, \theta_k) = \mathcal{N}(y_n \,|\, \theta_k, \mathrm{I}_2)$

[4] All experiments are publicly available at `https://github.com/biaslab/OnlineMessagePassingDirichletProcess`.

Table 1: Table containing (top) the Forney-style factor graph representation of the mixture node of [9]. The edge denoted by θ here denotes the active or selected parameter settings. (bottom) The derived outgoing messages for the mixture node mimicing a Dirichlet process. It can be noted that the backward message towards c_n resembles a scaled categorical distribution and that the forward message towards θ represents only one of the incoming messages $\vec{\mu}_{\theta_k}(\theta)$.

Factor node



Messages	**Functional form**
$\overleftarrow{\mu}_{c_n}(c_n)$	$\prod_{k=1}^{K^*+1} \left(\int \vec{\mu}_{\theta_k}(\theta_k) \overleftarrow{\mu}_{\theta_k}(\theta_k) \mathrm{d}\theta_k \right)^{c_{nk}}$
$\vec{\mu}_\theta(\theta)$	$\vec{\mu}_{\theta_k}(\theta)$ if $q(c_n) = \delta[c_n - e_k]$
$\overleftarrow{\mu}_{\theta_k}(\theta_k)$	$\begin{cases} \overleftarrow{\mu}_\theta(\theta_k) & \text{if } q(c_n) = \delta[c_n - e_k] \\ const & \text{otherwise} \end{cases}$

where I_2 denotes a two-dimensional identity matrix. The priors over the mean parameters θ_k are set to be uninformative as $p(\theta_k) = \mathcal{N}(\theta_k \,|\, 0_2, 10\mathrm{I}_2)$, where 0_2 denotes a two-dimensional vector of zeros. The concentration parameter is initialized as $\alpha = 0.1$.

Figure 3 shows the inferred class assignments and posterior mean parameters, together with the posterior concentration parameters, as a function of the number of observations. From the figure we can validate that the inference procedure indeed recovers the 8 clusters that were used to generate the data. Furthermore, the mean parameters have converged to the data generating cluster means.

6 Discussion and conclusions

The presented approach in this paper enables model expansion by emulating Dirichlet processes through message passing. Through the use of scale factors, we

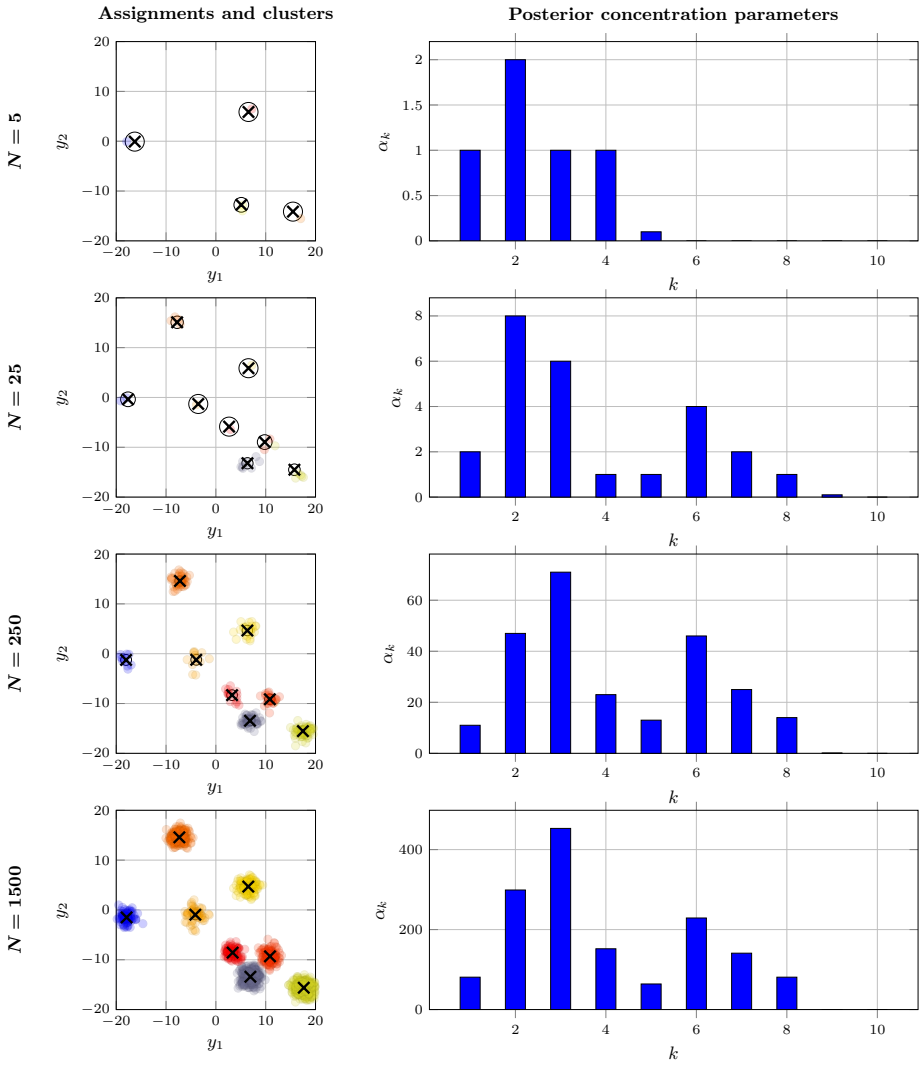


Fig. 3: Visualization of the results obtained by performing online inference in the infinite mixture model defined in Section 3 using the message passing implementation as described in Section 4. Each row represents the number of observations N. The left column shows the observations colored by their inferred cluster label and the inferred component means, denoted by square crosses. The right column denotes the concentration parameters of the approximate posterior distribution $q_N(\pi)$.

have effectively extended the task of model comparison to model expansion. The use of Dirichlet processes guarantees a well-grounded and principled approach

to the task of model expansion without any post-hoc treatment. A benefit of the online nature of the algorithm is that it is well-suited to the development of intelligent agents, which continuously perceive streams of information. Due to the online nature of the algorithm, its behavior also naturally depends on the ordering of the observations it perceives [36]. This is in contrast to sampling-based methods, however, these are significantly more computationally demanding.

It is important to note that the presented mixture node does not enforce any constraints on adjacent parts of the graph and can be used in both discrete and continuous spaces. A limitation of scale factors is that they can only be efficiently computed when the model submits to exact inference [27]. Extensions of the scale factors towards a variational setting would allow the use of the mixture node with a bigger variety of models. If this limitation is resolved, then the introduced approach can be combined with more complicated models, such as, for example, Bayesian neural networks, whose performance is measured by the variational free energy; see, e.g. [7, 17]. This provides a novel solution to multi-task machine learning problems where the number of tasks is not known beforehand [30]. Each Bayesian neural network can then be trained for a specific task, and additional components or networks can be added if appropriate.

Acknowledgments. The authors would like to thank the BIASlab team members for various insightful discussions related to this work. This publication is part of the project "ROBUST: Trustworthy AI-based Systems for Sustainable Growth" with project number KICH3.LTP.20.006, which is (partly) financed by the Dutch Research Council (NWO), GN Hearing, and the Dutch Ministry of Economic Affairs and Climate Policy (EZK) under the program LTP KIC 2020-2023.

Disclosure of Interests. The authors declare no conflict of interest.

References

1. Bagaev, D., van Erp, B., Podusenko, A., de Vries, B.: ReactiveMP.jl: A Julia package for reactive variational Bayesian inference. Software Impacts **12**, 100299 (May 2022). https://doi.org/10.1016/j.simpa.2022.100299, https://www.sciencedirect.com/science/article/pii/S2665963822000422
2. Bagaev, D., Podusenko, A., De Vries, B.: RxInfer: A Julia package for reactive real-time Bayesian inference. Journal of Open Source Software **8**(84), 5161 (Apr 2023). https://doi.org/10.21105/joss.05161, https://joss.theoj.org/papers/10.21105/joss.05161
3. Bagaev, D., de Vries, B.: Reactive Message Passing for Scalable Bayesian Inference. Scientific Programming **2023**, 6601690 (May 2023). https://doi.org/10.1155/2023/6601690, https://doi.org/10.1155/2023/6601690, publisher: Hindawi
4. Beckers, J., van Erp, B., Zhao, Z., Kondrashov, K., de Vries, B.: Principled Pruning of Bayesian Neural Networks Through Variational Free Energy Minimization. IEEE Open Journal of Signal Processing **5**, 195–203 (2024). https://doi.org/10.1109/OJSP.2023.3337718, https://ieeexplore.ieee.org/document/10334001, conference Name: IEEE Open Journal of Signal Processing

5. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: A Fresh Approach to Numerical Computing. SIAM Review **59**(1), 65–98 (Jan 2017). https://doi.org/10.1137/141000671, https://epubs.siam.org/doi/10.1137/141000671, publisher: Society for Industrial and Applied Mathematics
6. Blei, D.M., Jordan, M.I.: Variational inference for Dirichlet process mixtures. Bayesian Analysis **1**(1) (Mar 2006). https://doi.org/10.1214/06-BA104, https://projecteuclid.org/journals/bayesian-analysis/volume-1/issue-1/Variational-inference-for-Dirichlet-process-mixtures/10.1214/06-BA104.full
7. Blundell, C., Cornebise, J., Kavukcuoglu, K., Wierstra, D.: Weight Uncertainty in Neural Networks (May 2015). https://doi.org/10.48550/arXiv.1505.05424, http://arxiv.org/abs/1505.05424, arXiv:1505.05424 [cs, stat]
8. Dauwels, J., Korl, S., Loeliger, H.A.: Expectation maximization as message passing. In: Proceedings. International Symposium on Information Theory, 2005. ISIT 2005. pp. 583–586. IEEE, Adelaide, Australia (2005). https://doi.org/10.1109/ISIT.2005.1523402, http://ieeexplore.ieee.org/document/1523402/
9. van Erp, B., Nuijten, W.W.L., van de Laar, T., de Vries, B.: Automating Model Comparison in Factor Graphs. Entropy **25**(8), 1138 (Aug 2023). https://doi.org/10.3390/e25081138, https://www.mdpi.com/1099-4300/25/8/1138, number: 8 Publisher: Multidisciplinary Digital Publishing Institute
10. Ferguson, T.S.: A Bayesian Analysis of Some Nonparametric Problems. The Annals of Statistics **1**(2), 209–230 (Mar 1973). https://doi.org/10.1214/aos/1176342360, https://projecteuclid.org/journals/annals-of-statistics/volume-1/issue-2/A-Bayesian-Analysis-of-Some-Nonparametric-Problems/10.1214/aos/1176342360.full, publisher: Institute of Mathematical Statistics
11. Forney, G.: Codes on graphs: normal realizations. IEEE Transactions on Information Theory **47**(2), 520–548 (Feb 2001). https://doi.org/10.1109/18.910573
12. Fox, E.B., Sudderth, E.B., Jordan, M.I., Willsky, A.S.: An HDP-HMM for systems with state persistence. In: Proceedings of the 25th international conference on Machine learning. pp. 312–319. ICML '08, Association for Computing Machinery, New York, NY, USA (Jul 2008). https://doi.org/10.1145/1390156.1390196, https://dl.acm.org/doi/10.1145/1390156.1390196
13. Fox, E.B., Sudderth, E.B., Jordan, M.I., Willsky, A.S.: A Sticky Hdp-Hmm with Application to Speaker Diarization. The Annals of Applied Statistics **5**(2A), 1020–1056 (2011), https://www.jstor.org/stable/23024915, publisher: Institute of Mathematical Statistics
14. Friston, K., Parr, T., Zeidman, P.: Bayesian model reduction. arXiv:1805.07092 [stat] (Oct 2019), http://arxiv.org/abs/1805.07092, arXiv: 1805.07092
15. Friston, K., Penny, W.: Post hoc Bayesian model selection. NeuroImage **56**(4), 2089–2099 (Jun 2011). https://doi.org/10.1016/j.neuroimage.2011.03.062, http://www.sciencedirect.com/science/article/pii/S1053811911003417
16. Friston, K.J., Da Costa, L., Tschantz, A., Kiefer, A., Salvatori, T., Neacsu, V., Koudahl, M., Heins, C., Sajid, N., Markovic, D., Parr, T., Verbelen, T., Buckley, C.L.: Supervised structure learning (Nov 2023). https://doi.org/10.48550/arXiv.2311.10300, http://arxiv.org/abs/2311.10300, arXiv:2311.10300 [cs]
17. Haussmann, M., Hamprecht, F.A., Kandemir, M.: Sampling-Free Variational Inference of Bayesian Neural Networks by Variance Backpropagation (Jun 2019). https://doi.org/10.48550/arXiv.1805.07654, http://arxiv.org/abs/1805.07654
18. Hoeting, J.A., Madigan, D., Raftery, A.E., Volinsky, C.T.: Bayesian Model Averaging: A Tutorial. Statistical Science **14**(4), 382–401 (1999), https://www.jstor.org/stable/2676803

19. Kschischang, F., Frey, B., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory **47**(2), 498–519 (Feb 2001). https://doi.org/10.1109/18.910572
20. Lo, A.Y.: On a Class of Bayesian Nonparametric Estimates: I. Density Estimates. The Annals of Statistics **12**(1), 351–357 (1984), https://www.jstor.org/stable/2241054, publisher: Institute of Mathematical Statistics
21. Loeliger, H.A.: An introduction to factor graphs. IEEE Signal Processing Magazine **21**(1), 28–41 (Jan 2004). https://doi.org/10.1109/MSP.2004.1267047
22. Loeliger, H.A., Dauwels, J., Hu, J., Korl, S., Ping, L., Kschischang, F.R.: The Factor Graph Approach to Model-Based Signal Processing. Proceedings of the IEEE **95**(6), 1295–1322 (Jun 2007). https://doi.org/10.1109/JPROC.2007.896497, http://ieeexplore.ieee.org/document/4282128/
23. Lu, X., Zhang, C., Wang, Z.: Combined Belief Propagation-Mean Field Message Passing Algorithm for Dirichlet Process Mixtures. IEEE Signal Processing Letters **26**(7), 1041–1045 (Jul 2019). https://doi.org/10.1109/LSP.2019.2918680, https://ieeexplore.ieee.org/document/8721567, conference Name: IEEE Signal Processing Letters
24. Minka, T.P.: Expectation Propagation for Approximate Bayesian Inference. In: Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence. pp. 362–369. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001), http://dl.acm.org/citation.cfm?id=2074022.2074067
25. Monteith, K., Carroll, J.L., Seppi, K., Martinez, T.: Turning Bayesian model averaging into Bayesian model combination. In: The 2011 International Joint Conference on Neural Networks. pp. 2657–2663. San Jose, CA, USA (Jul 2011). https://doi.org/10.1109/IJCNN.2011.6033566, iSSN: 2161-4407
26. Neal, R.M.: Markov Chain Sampling Methods for Dirichlet Process Mixture Models. Journal of Computational and Graphical Statistics **9**(2), 249–265 (Jun 2000). https://doi.org/10.1080/10618600.2000.10474879, https://www.tandfonline.com/doi/abs/10.1080/10618600.2000.10474879, publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/10618600.2000.10474879
27. Nguyen, H.M., van Erp, B., Şenöz, İ., de Vries, B.: Efficient Model Evidence Computation in Tree-structured Factor Graphs. In: 2022 IEEE Workshop on Signal Processing Systems (SiPS). pp. 1–6 (Nov 2022). https://doi.org/10.1109/SiPS55645.2022.9919250, iSSN: 2374-7390
28. Rasmussen, C.: The Infinite Gaussian Mixture Model. In: Advances in Neural Information Processing Systems. vol. 12. MIT Press (1999), https://papers.nips.cc/paper/1999/hash/97d98119037c5b8a9663cb21fb8ebf47-Abstract.html
29. Reller, C.: State-space methods in statistical signal processing: New ideas and applications. Ph.D. thesis, ETH Zurich (2013), http://hdl.handle.net/20.500.11850/65488
30. Ruder, S.: An Overview of Multi-Task Learning in Deep Neural Networks (Jun 2017), http://arxiv.org/abs/1706.05098, arXiv:1706.05098
31. Smith, R., Schwartenbeck, P., Parr, T., Friston, K.J.: An Active Inference Approach to Modeling Structure Learning: Concept Learning as an Example Case. Frontiers in Computational Neuroscience **14** (2020). https://doi.org/10.3389/fncom.2020.00041, https://www.frontiersin.org/articles/10.3389/fncom.2020.00041/full, publisher: Frontiers
32. Särkkä, S.: Bayesian Filtering and Smoothing. Institute of Mathematical Statistics Textbooks, Cambridge University Press, Cambridge (2013). https://doi.

org/10.1017/CBO9781139344203, https://www.cambridge.org/core/books/bayesian-filtering-and-smoothing/C372FB31C5D9A100F8476C1B23721A67
33. Teh, Y.W., Jordan, M.I.: Hierarchical Bayesian nonparametric models with applications. In: Bayesian Nonparametrics, pp. 158–207. Cambridge University Press, 1 edn. (Apr 2010). https://doi.org/10.1017/CBO9780511802478.006, https://www.cambridge.org/core/product/identifier/CBO9780511802478A043/type/book_part
34. Wang, L., Dunson, D.B.: Fast Bayesian Inference in Dirichlet Process Mixture Models. Journal of computational and graphical statistics : a joint publication of American Statistical Association, Institute of Mathematical Statistics, Interface Foundation of North America **20**(1), 10.1198/jcgs.2010.07081 (Jan 2011). https://doi.org/10.1198/jcgs.2010.07081, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3812957/
35. Winn, J.M.: Variational Message Passing and its Applications. Ph.D. thesis, University of Cambridge, Cambridge, United Kingdom (2004)
36. Zhang, X., Nott, D.J., Yau, C., Jasra, A.: A Sequential Algorithm for Fast Fitting of Dirichlet Process Mixture Models. Journal of Computational and Graphical Statistics **23**(4), 1143–1162 (2014), https://www.jstor.org/stable/43304802, publisher: [American Statistical Association, Taylor & Francis, Ltd., Institute of Mathematical Statistics, Interface Foundation of America]
37. Şenöz, İ., van de Laar, T., Bagaev, D., de Vries, B.: Variational Message Passing and Local Constraint Manipulation in Factor Graphs. Entropy **23**(7), 807 (Jul 2021). https://doi.org/10.3390/e23070807, https://www.mdpi.com/1099-4300/23/7/807, number: 7 Publisher: Multidisciplinary Digital Publishing Institute