Treatment-RSPN: Recurrent Sum-Product Networks for Sequential Treatment Regimes

Anonymous Author(s)
Affiliation
Address
email

Abstract

Sum-product networks (SPNs) have recently emerged as a novel deep learning architecture enabling highly efficient probabilistic inference. Since their introduction, SPNs have been applied to a wide range of data modalities and extended to time-sequence data. In this paper, we propose a general framework for modelling sequential treatment decision-making behaviour and treatment response using recurrent sum-product networks (RSPNs). Models developed using our framework benefit from the full range of RSPN capabilities, including the abilities to model the full distribution of the data, to seamlessly handle latent variables, missing values and categorical data, and to efficiently perform marginal and conditional inference. Our methodology is complemented by a novel variant of the expectation-maximization algorithm for RSPNs, enabling efficient training of our models. We evaluate our approach on a synthetic dataset as well as real-world data from the MIMIC-IV intensive care unit medical database. Our evaluation demonstrates that our approach can closely match the ground-truth data generation process on synthetic data and achieve results close to neural and probabilistic models while using a tractable and highly versatile model.

1 Introduction

Modelling of the sequential decision-making process of physicians, as well as the patient response to different treatments, has a wide range of useful applications. Transparent and interpretable models of human decision-making behaviour have been proposed as tools for understanding, quantifying and replicating policies in clinical practice [1, 2], while the treatment response models can help to inform clinician's choices when choosing among multiple available therapies [3].

Unfortunately, the currently widely used modelling methods suffer from considerable limitations. Neural-based models are highly expressive but do not represent tractable probabilistic distributions and are thus unable to support exact joint, marginal or conditional inference, greatly limiting their versatility. This also means that such models are inherently opaque and incapable of quantifying the uncertainty of their predictions in a principled way. Meanwhile, conventional probabilistic models are often insufficiently expressive to faithfully represent more complex systems.

Sum-product networks (SPNs) [4, 5], a deep learning architecture based on probabilistic circuits, largely address the above limitations, as they are highly expressive while still guaranteeing tractability. Due to these appealing properties, SPNs have been successfully applied in a variety of settings, including face image completion [4], robot navigation [6], image segmentation [7], speech modelling [8] or activity recognition in videos [9]. SPNs have also been generalized to time-series data of variable length, giving rise to recurrent sum-product networks (RSPNs) [10].

Submitted to NeurIPS 2022 Workshop on Learning from Time Series for Health. Do not distribute.

In this work, we introduce a probabilistic deep generative approach, Treatment-RSPN, that leverages RSPNs for efficient modelling of treatment decision-making as well as the dynamics of the patient's response to different treatments. As part of our framework, we develop a method for transforming conventional probabilistic graphical models (PGMs), such as dynamic Bayesian networks (DBNs) [11], into RSPNs, allowing us to bootstrap our models with a structure informed by domain knowledge and the specific task. Initializing the models in that way also benefits interpretability, as the random variables from the underlying PGM and the dependencies between them are faithfully reflected in the resulting RSPN. In order to train our models on data, we introduce a variant of the expectation-maximization (EM) algorithm optimizing the parameters of the RSPN and optionally updating its structure to more closely represent the considered distribution.

Treatment-RSPN provides *generative* and *tractable* models for sequential treatment regimes in partially observed settings. Additionally, these models can benefit from the full range of inference techniques devised for SPNs, including the evaluation of marginal, conditional and approximate or exact most probable explanation queries in a linear time with respect to their size, conditional or unconditional sampling and seamless handling of missing values. We describe our method in more detail in the following section.

2 Treatment-RSPN

2.1 Background

The models developed using the Treatment-RSPN framework are based on recurrent sum-product networks (RSPNs) [10], a generalization of SPNs specifically adapted for modelling time-series data. Each RSPN is composed of a top network, a template network and a bottom network. These networks can be stacked together to form a regular, unrolled SPN, with an arbitrary number of copies of the template network representing the variables over the full length of the considered sequence. The top and the bottom networks enable the special-case representation of the first and last time slices of the sequence, respectively.

The unrolled SPN is a rooted directed acyclic graph with leaf nodes representing base distributions over random variables (such as Categorical or Gaussian) and internal nodes representing sums and products of probability functions of their children. Each edge leading from a sum node to a child node is associated with a non-negative parameter indicating the weight of the corresponding child. The weights of the edges originating at a single sum node must be normalized. A key component of the SPN inference and training methods is the SPN evaluation. Given a (partial or full) assignment to the variables in the scope of an SPN $\mathbf{x}$, the value $S(\mathbf{x})$ of an SPN can be computed as follows:

$$S(\mathbf{x}) = \begin{cases} P_S(X = \mathbf{x}) & \text{if } S \text{ is a leaf} \\ \sum_{C_i \in children(S)} w_i C_i(\mathbf{x}) & \text{if } S \text{ is a sum} \\ \prod_{C_i \in children(S)} C_i((x)) & \text{if } S \text{ is a product} \end{cases}$$

The value $S(\mathbf{x})$ corresponds to the probability of the assignment $\mathbf{x}$ under the distribution modelled by the given SPN [4].

The possibility of evaluating an SPN on a partial variables assignment provides a straightforward way for performing a marginal and conditional inference, as the variables not included in an assignment are being effectively marginalized over. This can be used for handling latent variables and missing values during both training and inference.

2.2 Treatment-RSPN Initialization

The first step of developing a model within our framework is the initialization of its structure. This step is of particular importance for RSPNs, as their structure directly affects the assumed dependencies between the variables in the scope of the model. In general, there are multiple ways in which the structure can be initialized, including learning the structure directly from the data using one of the structure learning algorithms (such as local search [10] or oSLRAU [12]), crafting the structure manually or basing it on a known probabilistic model.

In this work, we opt for the last approach, though we note that this is merely a detail of a particular instantiation of our framework. Specifically, we base the initial structure of our RSPN models on an input-output hidden Markov model (IOHMM) [13, 14]. IOHMMs model sequences composed of

Table 1: Log-likelihood of the synthetic test data under the reference IOHMM model used for their generation and the learned Treatment-RSPN model

Model	**Log-likelihood (↑)**
	Mean ± SD
Reference IOHMM	-18842.82 ± 131.44
Treatment-RSPN	-18885.07 ± 129.20

inputs $u_1, ..., u_n$, unobserved (latent) states $z_1, ..., z_n$ and outputs $x_1, ..., x_n$. This makes them highly suited for modelling sequential treatment regimes, as the treatment actions at each time step can be considered as inputs, while the patient physiological variables or test results can be considered as outputs. Additionally, the dependencies between the variables in an IOHMM model are rather simple and intuitive, leading to a more interpretable model.

In order to transform an IOHMM into an RSPN structure, we first consider an unrolled Bayesian network representation of the model over a fixed number of time slices. We then convert this network into an SPN structure, gradually constructing layers for the variables in the topological order of their appearance in the network graph. We give a detailed description of this procedure in the appendix A. After the initial conversion, the repeated portions of the resulting SPN structure modelling the variables in the intermediate time slices can be straightforwardly extracted into a template network, while the structures for the first and the last time slices can be turned into the top and bottom networks, respectively. This results in the initial form of the RSPN. Note that our transformation is fully general and could be applied to an arbitrary model, as long as it is expressible as a DBN. We provide a diagram of an example IOHMM and the associated RSPN in appendix C.

2.3 Treatment-RSPN Training

After deriving the initial structure for the RSPN model, we need to learn its parameters based on data. We opt to use an adapted version of the expectation-maximization (EM) algorithm. Compared to other formulations in the literature [4, 15], our variant of the EM algorithm does not require the use of differentiation and is straightforwardly applicable to RSPNs in addition to regular SPNs. Additionally, our approach is capable of adjusting the structure of the learned RSPN so that it more closely matches the distribution of the training data.

Our EM algorithm first unrolls the RSPN for a fixed number of time steps required to fit the given training data, internally assigning identical identifiers to the nodes repeated in the unrolled SPN. This ensures that the parameters of these nodes are updated jointly and do not diverge during the training process. The algorithm then alternates between the expectation and maximization steps. In an E-step, the procedure computes the likelihood of each data point "reaching" each node, i.e. the probability of this point having been generated by the node. In an M-step, the algorithm updates the weights on the edges leading from the sum nodes as well as the internal parameters of the leaves according to these likelihoods. Optionally, the algorithm can also replace the SPN leaves with a more complex learned structure by clustering the data points weighted by their likelihoods at the given leaf and creating a new sum node with multiple leaves modelling the clusters. We give a detailed description of our EM algorithm in the appendix B.

3 Experiments

We evaluate the performance of the Treatment-RSPN models on a synthetically generated dataset as well as real-world data from the MIMIC-IV intensive care unit medical database [16, 17].

In the synthetic data experiment, we test whether the Treatment-RSPN can closely match the parameters of a ground truth data generation process based on an IOHMM with two inputs, two states and four possible categorical observations. As shown in table 1, the log-likelihood of the Treatment-RSPN differs from the log-likelihood of the ground-truth model by less than 0.25%, indicating a near-perfect match and a high generative performance.

Table 2: Classification and calibration performance of the evaluated models on the one-step-ahead treatment action prediction task using the MIMIC-IV sepsis dataset

Model	**AUROC (↑)**	**F1 macro (↑)**	**Brier score (↓)**
	Mean ± SD	**Mean ± SD**	**Mean ± SD**
Interpole	0.88 ± 0.020	0.87 ± 0.023	0.042 ± 0.021
PO-MB-IL	0.58 ± 0.088	0.49 ± 0.010	0.108 ± 0.065
PO-IRL	0.57 ± 0.016	0.44 ± 0.020	0.437 ± 0.003
Off. PO-IRL	0.57 ± 0.054	0.49 ± 0.009	0.484 ± 0.003
LSTM	0.85 ± 0.030	0.85 ± 0.023	0.049 ± 0.021
Predict most common	0.50 ± 0.000	0.49 ± 0.009	0.098 ± 0.061
Treatment-RSPN	0.91 ± 0.010	0.86 ± 0.008	0.045 ± 0.024

Table 3: Regression performance of the evaluated models on the one-step ahead treatment response prediction task using the MIMIC-IV sepsis dataset

Model	**HR RMSE (↓)**	**MBP RMSE (↓)**	**RR RMSE (↓)**	**AVG RMSE (↓)**
	Mean ± SD	**Mean ± SD**	**Mean ± SD**	**Mean ± SD**
LSTM	10.81 ± 0.525	14.57 ± 0.938	5.35 ± 0.137	10.24 ± 0.487
Treatment-RSPN	9.54 ± 0.267	13.09 ± 0.453	4.46 ± 0.209	9.03 ± 0.286

On the MIMIC-IV dataset, we evaluate our models using the data from the ICU stays of patients suffering from sepsis. In the first MIMIC-IV experiment, we apply Treatment-RSPN to one-step-ahead treatment action prediction of the administration of vasopressors, with discretized minimum blood pressure and the total volume of intravenous fluids received by the patient serving as covariates. The task is set up as a binary classification problem, with the target values indicating whether any vasopressors were administered at the corresponding time step. We compare the performance of our model to baselines from [1], as well as a basic model always predicting the most common treatment action. The results of the experiment are captured in table 2.

In the second MIMIC-IV experiment, we evaluate the performance of Treatment-RSPN on one-step-ahead continuous treatment response prediction, using vasopressors and intravenous fluids administration indicators as inputs and heart rate, minimum blood pressure and respiratory rate as the predicted treatment outcome variables. The results of Treatment-RSPN and an LSTM baseline [18] on this task are shown in table 3.

In both MIMIC-IV experiments, the Treatment-RSPN achieved a performance competitive with the baselines, demonstrating the promise of the method.

4 Conclusion and Future Work

In this paper, we introduced a framework for generative and tractable modelling of clinical decision-making and treatment response mechanisms using recurrent-sum-product networks. We discussed the process for constructing the initial network structure for the RSPN models, as well as an adapted version of the EM algorithm suitable for their training. In the future, we would like to extend the structure-learning capabilities of our training algorithm and further explore how the tractability of our models could be used to provide more interpretable and informative predictions. We would also like to explore the potential use of Treatment-RSPN for simulation, fully exploiting its generative capabilities.

References

[1] Alihan Hüyük, Daniel Jarrett, Cem Tekin, and Mihaela Van Der Schaar. Explaining by imitating: Understanding decisions by interpretable policy learning. In *International Conference on*

Learning Representations, 2020.

[2] Alizée Pace, Alex Chan, and Mihaela van der Schaar. POETREE: Interpretable policy learning with adaptive decision trees. In *International Conference on Learning Representations*, 2022.

[3] Bryan Lim. Forecasting treatment responses over time using recurrent marginal structural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[4] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.

[5] Adnan Darwiche. A differential approach to inference in bayesian networks. *Journal of the ACM (JACM)*, 50(3):280–305, 2003.

[6] Kaiyu Zheng, Andrzej Pronobis, and Rajesh Rao. Learning graph-structured sum-product networks for probabilistic semantic maps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[7] Abram L Friesen and Pedro Domingos. Submodular sum-product networks for scene understanding, 2016.

[8] Robert Peharz, Georg Kapeller, Pejman Mowlaee, and Franz Pernkopf. Modeling speech with sum-product networks: Application to bandwidth extension. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3699–3703, 2014.

[9] Mohamed R Amer and Sinisa Todorovic. Sum product networks for activity recognition. *IEEE transactions on pattern analysis and machine intelligence*, 38(4):800–813, 2015.

[10] Mazen Melibari, Pascal Poupart, Prashant Doshi, and George Trimponias. Dynamic sum product networks for tractable inference on sequence data. In *Conference on Probabilistic Graphical Models*, pages 345–355. PMLR, 2016.

[11] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational intelligence*, 5(2):142–150, 1989.

[12] Agastya Kalra, Abdullah Rashwan, Wei-Shou Hsu, Pascal Poupart, Prashant Doshi, and Georgios Trimponias. Online structure learning for feed-forward and recurrent sum-product networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[13] Yoshua Bengio and Paolo Frasconi. An input output hmm architecture. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7. MIT Press, 1994.

[14] Y. Bengio and P. Frasconi. Input-output hmms for sequence processing. *IEEE Transactions on Neural Networks*, 7(5):1231–1249, 1996.

[15] Robert Peharz, Robert Gens, Franz Pernkopf, and Pedro Domingos. On the latent variable interpretation in sum-product networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(10):2030–2044, 2016.

[16] Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Leo Anthony Celi, Roger Mark, and S Horng IV. Mimic-iv. *PhysioNet*, 2021.

[17] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997.

Algorithm 1: BN to SPN conversion procedure

Given a Bayesian network N to convert to an SPN

1. Set Vs to be the variables (nodes) in N in the topological order.
2. Set $c = \emptyset$ to be the set of variables currently conditioned on.
3. Set $d = \{\}$ to be a map of unresolved variable dependencies (mapping from parent variables to lists of child variables).
4. For V in Vs:
   - 4.1. Construct a layer of sum nodes s, one for each possible assignment $\mathbf{x}_i^s$ of variables in c or a single sum node if c is empty. Attach the layer to the previous product layer p if any, with edges leading between nodes n_j^p and n_i^s whenever $\mathbf{x}_j^p$ and $\mathbf{x}_i^s$ for these nodes are compatible.
   - 4.2. If V has children in N:
     - a. Add V to c.
     - b. Remove all occurrences of v from d, remove any variables with empty child list in d from c.
     - c. Add $V \mapsto \text{children}_N(V)$ to d.
     - d. Add a product layer p with one node for each possible assignment $\mathbf{x}_j^p$ of variables in c, each connected to an indicator leaf for $V = v_j$ where v_j is the value of V in $\mathbf{x}_j^p$. Connect the product layer to the previous sum layer s, with edges leading between nodes n_i^s and n_j^p whenever $\mathbf{x}_i^s$ and $\mathbf{x}_j^p$ for these nodes are compatible and edge weights determined by $P_N(V = v_j|\mathbf{x}_i^s)$.

     Otherwise:
     - a. Construct a layer of leave nodes for V with distributions and connections to the previous layer l representing the distribution $P_N(V|\mathbf{x}_i^l)$ for each node n_i^l with assignment $\mathbf{x}_i^l$ in l.

A Conversion Procedure

We give our procedure for converting a Bayesian network into an SPN in algorithm 1.

B EM Algorithm

We detail our variant of the RSPN expectation maximization procedure in algorithm 2.

C IOHMM and Treatment-RSPN Diagrams

In order to give more intuition for the structure of an IOHMM and Treatment-RSPN, we provide diagrams of a simple IOHMM model and the corresponding RSPN. The diagrams are shown in figures 1 and 2.

D Discrete Treatment Response Prediction Experiment

In addition to the experiments described in section 3, we also evaluated our model on the "mirror" task to MIMIC-IV experiment one, in which the model performs a one-step-ahead prediction of the discretized covariates (minimum blood pressure and the total volume of intravenous fluids). The results for this task are shown in table 4.

Algorithm 2: RSPN expectation maximization

Given an RSPN R and the training dataset D

1. Unroll R into a regular SPN S matching the length of the sequences in D and using shared node identifiers for repeated nodes.
2. Evaluate S on D using procedure described in section 2.1, store the value computed at each node.
3. Set $pl = \{(\text{nil}, \text{root}(R)) \mapsto \mathbf{1}\}$ to be the map of data point likelihoods passed to each node by its parents. The root node has no parents, and is thus passed likelihood of 1 for each point.
4. Set $l = \{(\text{id}(p), \text{id}(c)) \mapsto 0\}$ for all pairs of parents and children (p, c) in S. l stores the aggregated likelihoods of data points reaching nodes from their parents.
5. Set $lp, lw = \{\}, \{\}$ to store data points and their likelihoods at each leaf.
6. For n in nodes(S), processed from top to bottom:
   - 6.1. Set $dl = \sum_{p \in \text{parents}(n)} pl[p, n]$ to be the likelihoods of reaching the current node for each data point, passed to n by its parents.
   - 6.2. Increment $l[\text{id}(p), \text{id}(n)]$ by $\sum_{d \in D} pl[p, n, d]$ for all p in parents(n).
   - 6.3. If n is a sum:
     - a. Set $cl = \{c \mapsto \text{weight}(n, c) \times \text{value}(c)\}$ for all c in children(n) to store the base likelihood of each child.
     - b. Normalize the likelihoods of the children in cl
     - c. Set $pl[n, c] = cl[c] \times dl$

     If n is a product:
     - a. Set $pl[n, c] = dl$ for all c in children(n)

     If n is a leaf node:
     - a. Store the current data D and their likelihoods at this node dl in $lp[n]$ and $lw[n]$.
7. Update the weights of each edge between a sum node s and its child node c to $\frac{l[\text{id}(s), \text{id}(c)]}{\sum_{c' \in \text{children}(s)} l[\text{id}(s), \text{id}(c')]}$
8. Update the distributions in leaves according to the weighted data in lp and lw or cluster the weighted data and introduce a sub-structure to model the mixture.

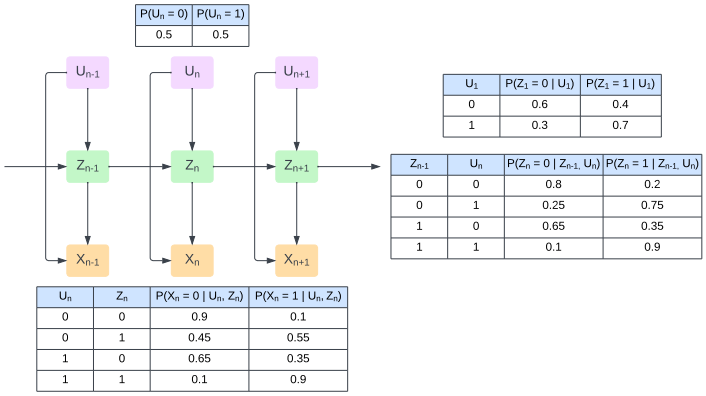

Figure 1: A simple IOHMM with discrete variables for the inputs, latent states and observations.

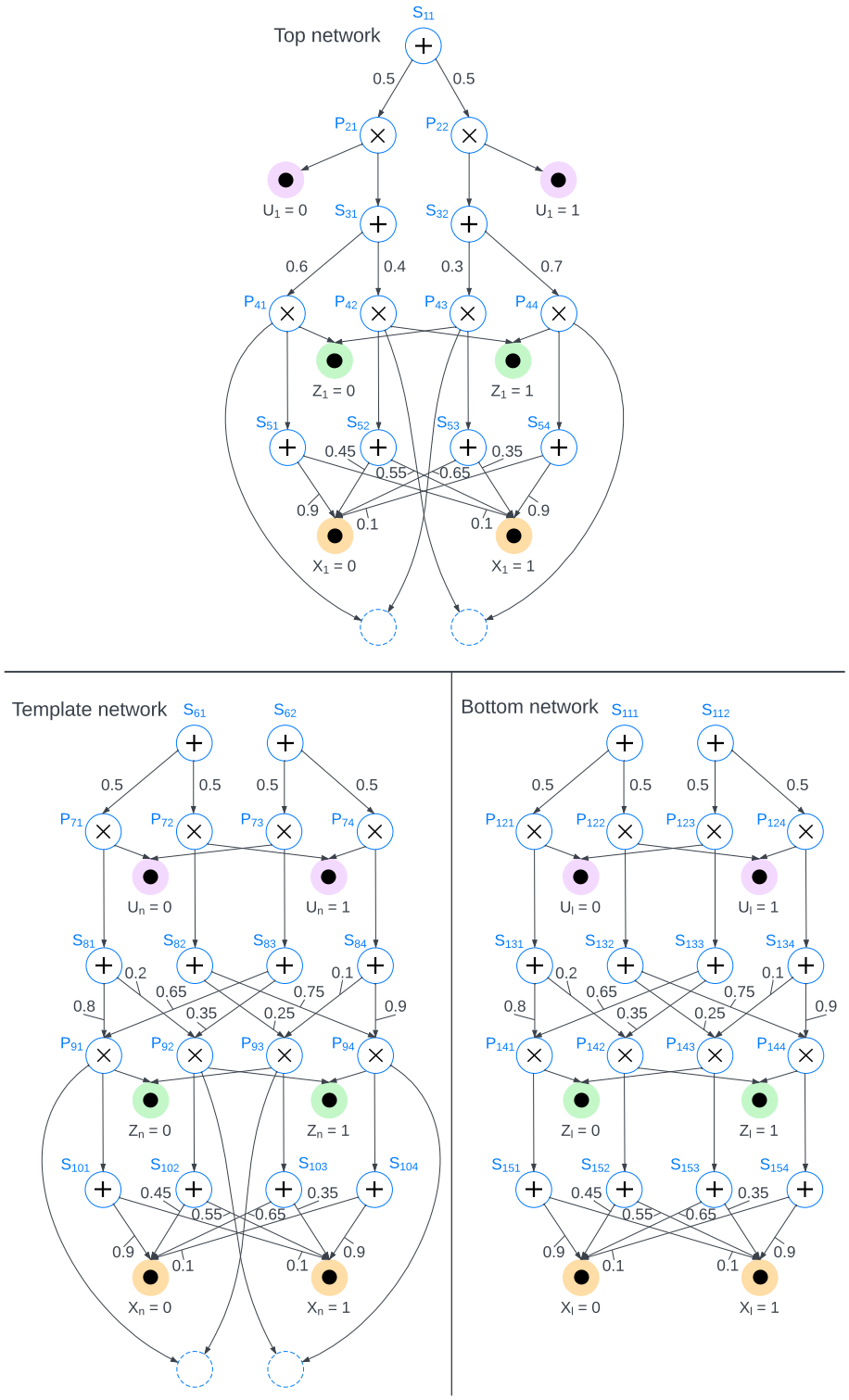

Figure 2: An RSPN corresponding to the IOHMM model shown in figure 1. This model can be obtained by running our initialization procedure described in section 2.2.

Table 4: Classification performance of the evaluated models on the one-step ahead treatment response prediction task using the MIMIC-IV sepsis dataset

Model	**F1 weighted (↑)**	**AUROC weighted (↑)**	**Brier score (↓)**
	Mean ± SD	**Mean ± SD**	**Mean ± SD**
LSTM	0.64 ± 0.054	0.93 ± 0.012	0.49 ± 0.058
Predict most common	0.10 ± 0.062	0.50 ± 0.000	0.87 ± 0.031
Treatment-RSPN	0.60 ± 0.059	0.93 ± 0.011	0.52 ± 0.062