Soft Adaptive Segments for Bio-Inspired Temporal Memory

Artem Prokhorenko[1][0009−0008−5194−8009], Evgenii Dzhivelikian[1][0000−0001−8110−1521], Petr Kuderov[1,2][0000−0001−7055−3426], and Aleksandr Panov[1,2,3][0000−0002−9747−3837]

[1] Moscow Institute of Physics and Technology, Dolgoprudny, Russia
[2] Artificial Intelligence Research Institute, Moscow, Russia
[3] Federal Reserarch Center "Computer Science and Control" of the Russian Academy of Sciences, Moscow, Russia

Abstract. World models for online learning in complex environments are increasingly essential, particularly in partially observable scenarios. Within this domain, biologically inspired models of temporal memory have emerged as a promising class of models. These models extend traditional artificial neural networks by incorporating the idea of dendritic segments into point neurons and leveraging the structural insights from neocortical layers to facilitate efficient online sequence learning. This paper introduces an innovative segment-growing strategy for enhancing the Distributed Hebbian Temporal Memory (DHTM) algorithm. The proposed modifications aim to boost the model's generalization capabilities and introduce a more efficient segment growth approach based on real-time training statistics. Our findings demonstrate that this novel strategy not only reduces the number of segments used, thereby enhancing the model's computational efficiency, but also enhances its predictive accuracy in challenging, noisy environments.

Keywords: temporal memory · bio-inspired learning · online learning.

1 Introduction

Temporal Memory (TM) algorithms play a key role in natural language processing, time-series analysis, world models and others [6, 22, 24]. The idea behind TM is to form a compressed representation of the input and subsequently transform this representation over time to predict future data.

Most known TM algorithms, such as Recurrent Neural Network (RNN) [28] and Hidden Markov Model (HMM) [16], have drawbacks. Loss of long-term dependencies, overfitting, and vanishing gradients are inherent in models with backpropagation learning [5]. Also, there are limitations to Baum-Welch [26] training for models such as HMM, which can lead to slow convergence and difficulties in finding the global maximum [9]. In addition, most traditional memory models do not support online learning, which is very important for tasks where data is constantly changing or where immediate, test time adaptation to new information is required.

There is an increasing necessity for architectures capable of online sequence learning in partially observable environments, which, as a consequence, should not rely on backpropagation or other learning algorithms that require huge sample buffers. Neurophysiological temporal memory models, such as Hierarchical Temporal Memory (HTM) [17] may serve as a basis for such architecture as they rely only on local learning rules, which are more biologically plausible than backpropagation [4]. HTM framework has already shown to be a prolific source of inspiration for anomaly detection [1, 31, 2] algorithms, hierarchical agent architectures [15], efficient sparse artificial neural networks [20], and networks able to mitigate catastrophic forgetting [21].

Distributed Hebbian Temporal Memory (DHTM) [13] represents a promising approach of bringing ideas of HTM framework into Hidden Markov Models (HMM) representing a hybrid sequence learning algorithm. DHTM maps the idea of Sparse Distributed Representations to categorical state hidden state variables, neuronal dendritic segments to components of transition matrix for partially coupled HMMs [19] and cortical columns to the emission probability function, similarly to CSCG [18]. DHTM demonstrates the capabilities of learning under highly non-stationary conditions and diminishes the limitations associated with the Baum-Welch classical learning algorithm for HMM, including its low computational and sample efficiency. One of the main drawbacks of the DHTM model is the excessive growth of dendritic segments in noisy environments, which impairs its computational efficiency and predictive quality due to a lack of generalization capabilities. Therefore, we introduce another approach for growing new segments, further incorporating ideas from HTM framework. It will be demonstrated that our proposed method for restricting the formation of segments in the DHTM model can reduce the number of grown segments and improve the model's quality in noisy environments.

2 Related Works

Temporal memory models based on the concept of dendritic segments can be primarily found in the framework of Hierarchical Temporal Memory [17]. The HTM model is based on the structure and function of the neocortex and mimics the functioning of the brain through the use of dendritic segments. Dendritic segment with clustered synapses can trigger NMDA spikes, contributing to the prediction of patterns and sequences. This mechanism allows HTM to continually learn from data streams, mimicking some aspects of human memory and cognition.

There are also many developments in HTM ideas. Extensions to the HTM architecture may involve the addition of Gated Recurrent Unit (GRU) [30] neurons. This made it possible to improve the performance of the model with long-term dependencies for time series analysis [25]. In an unsupervised approach, you can combine Extreme Learning Machine [11] and HTM in one model [29]. HTM may also be used for model-based reinforcement learning algorithms [15]. The Dis-

tributed Hebbian Temporal Memory [14] is inherited from HTM and connects it to HMM-like models employing a probabilistic segment model.

Another temporal memory capable for online learning is an ecosystem with cognitive models such as Echo State Networks (ESN) [32] and Liquid State Machines (LSM) [34]. ESN and LSM use a reservoir computing approach, where neurons inside a reservoir process inputs and produce complex nonlinear responses. This approach allows models to adapt to changes in input data and learn online.

Many graph models are based on the belief propagation (BP) algorithm [33]. There are many adaptations of the belief propagation ideas in order to improve the quality of models. For example, integrates BP with GRU [7] and Graph Attention Networks [30], which allowing for efficient online learning and addressing issues with static damping factors [10]. Also, the performance of BP can be improved by removing edges [23].

3 Background

3.1 Belief propagation and Factorial HMM

Our contribution is a new message passing and graphical model learning algorithm based on belief propagation. The belief propagation algorithm is a message-passing mechanism for inference in graphical models. It is often used in probabilistic models to compute marginals and most likely assignments.

To compute marginal probabilities, a node X_i in a graph sends a message to a node X_j along the edge (i, j) using the following equation:

$$m_{i \to j}(x_j) = \sum_{x_i} \psi_{ij}(x_i, x_j) \prod_{k \neq j} m_{k \to i}(x_i), \tag{1}$$

where $\psi_{ij}(x_i, x_j)$— function representing the relationship between nodes X_i and X_j, and $m_{k \to i}(x_i)$— message from node X_k to X_i. Iterations continue until convergence [33].

The graphical structure learned by the proposed method belongs to the class of Factorial Hidden Markov Models, which extends the concept of HMM by allowing multiple hidden states variables to operate concurrently. The transition probabilities between hidden states in Factorial HMM are the following:

$$P(H_{t+1}|H_t) = \prod_{i=1}^{N} A_i(H_{t,i}, H_{t+1,i}), \tag{2}$$

where N is the number of hidden state variables, and $A_i(H_{t,i}, H_{t+1,i})$ is the transition matrix for each of the hidden variables. The more general formulation of the Factorial HMM also allows dependencies between different hidden variables [19].

The DHTM model is inspired by these approaches and provides powerful means for probabilistic modelling of sequential data.

3.2 Distributed Hebbian Temporal Memory

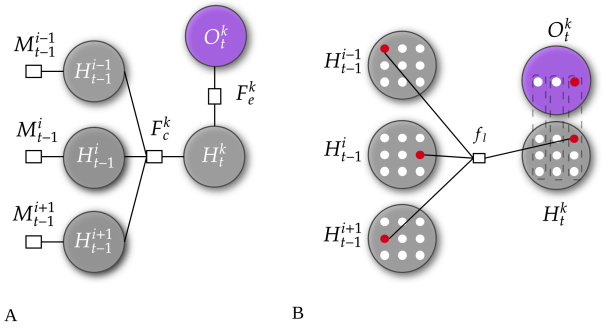

Fig. 1: A. Factor graph of the generative model DHTM learns. B. Internal structure of variables and factors. Each segment corresponds to a context factor value f_l for a combination of states (red circles). Emission factor is fixed that each observation state corresponds to a group of hidden states (column) which could be activated with equal probability, given the observation state active.

Here, we briefly describe the Distributed Hebbian Temporal Memory algorithm, an extension of which is the subject of our contribution. For more details, see [14].

DHTM implements learning and inference in a factor graph shown in Figure 1-A for one observation variable. All random variables are categorical. The generative model DHTM may consist of several observation variables, O^k_t, processed in parallel, where t—the current time-step and k—the variable's number. Each hidden variable H^k_t forms an explanation model of the corresponding observation variable O^k_t and is considered independently of other hidden random variables for the current time step. The context factor F^k_c describes the dependency between a subset of previous time-step hidden variables and the current time-step k-th hidden variable H^k_t. Messages from previous time steps are represented by M^i_{t-1} factors. The emission factor F^k_e is fixed, and it sets the hidden variable posterior distribution update rule (Figure 1-B).

One of the main features of DHTM is that it handles context factor functions as a set of computational units called segments, which are the counterparts of biological dendritic segments of pyramidal neurons. Each segment corresponds to a factor value for a combination of variable states that are arguments of the factor function. Given that only a small fraction of possible combinations are actually used in the model, this design step is analogous to that used for efficient processing of sparse matrices [12]. However, as will be shown in the following sections, not every combination needs a separate segment, that is,

some of them may be grouped, further improving the computational efficiency of the algorithm.

4 Methods

In this section, we first outline some aspects of the DHTM algorithm as a setup for our contribution and then describe the method we developed to group factor values for different combinations of states.

Computations of the DHTM segment are depicted in Figure 2-A. The segment can be viewed as a Bernoulli random variable, the activation probability P_l of which depends on the factor value—f_l, synapse efficiencies—w_u and presynaptic cells (variables' states) activation probabilities m_u. The presynaptic cells form the segment's receptive field, $rec(l)$, which corresponds to state combinations of previous time-step hidden variables. The computations take place during the belief-propagation stage and represent a modified version of the sum-product algorithm to approximately account for dependency between H^i_{t-1} variables, which is not considered by the model itself.

The number of segments in use is determined by the segment-growth stage, which follows the belief-propagation stage. The states of the variables are sampled, and the overlaps between active cells and segments' receptive fields are determined. If there are no segments to match the combination of states, a new one is initialized.

That is, in DHTM, a new segment is constructed for each new combination of sampled states. However, in noisy environments, this results in an excessively large number of segments, which are rarely used and even negatively impact the model's quality.

Usually, in this scenario, pruning is applied to remove noisy segments based on their activation frequency, which, however, may delete important dependencies and deteriorate TM's performance. Moreover, it involves setting hyperparameters, such as a limit on the maximum number of segments or a pruning ratio coefficient, which depend on the environment and cannot be predetermined. In our method, we aimed to develop a more precise method for increasing DHTM's noise tolerance by modifying segments' computation and growth.

4.1 Merging Segments

The central idea involves merging segments with substantially overlapping receptive fields. This is done by setting the overlap activation threshold for a segment during the segment-growth stage lower than the size of the receptive field. In other words, when encountering a new combination of states, we search for the segment with a similar receptive field first. However, this approach requires modifications to the belief-propagation stage, since in that case the segment can be activated in several ways.

The core concept is to sum segment activation probabilities over all possible combinations of states in the receptive field. Each such combination we

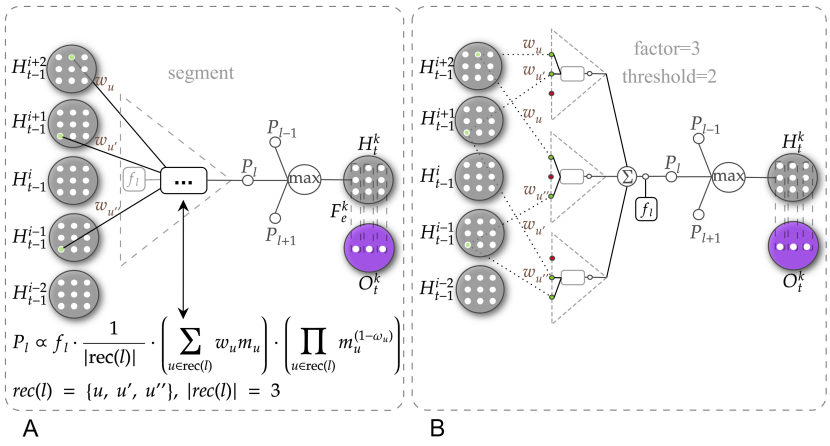

Fig. 2: A. Calculating segment activation probability using factor value f_l, to calculate effective cell excitation by segment l. B. Demonstration of accounting for combinations of subsegments, example for factor size = 3 and thresholds = 2.

call a subsegment. However, we leave out combinations that are bigger than the threshold (Figure 2-B). We do it for several reasons: 1. The contribution to the sum is usually lower the more variables the product includes; 2. it significantly reduces the number of operations. In this case, the number of subsegment combinations taken into account is C^{trsh}_{factor} Then the equation for segment activation probability takes the following form:

$$P_l \propto f_l \cdot \sum_{\substack{\text{rec}^i(l) \\ |\text{rec}^i(l)|=\text{trsh}}} \left(\frac{1}{|\text{rec}^i(l)|} \cdot \sum_{u \in \text{rec}^i(l)} w_u m_u \right) \cdot \left(\prod_{u \in \text{rec}^i(l)} m_u^{(1-\omega_u)} \right) \tag{3}$$

where $\text{rec}^i(l)$ is a subset of cells in the receptive field corresponding to i-th subsegment, $|\text{rec}^i(l)|$ is the number of cells in the subset.

4.2 Adaptive Thresholding

As mentioned, the essence of segments is that they represent a mapping of combinations of variable states into factor values. The collection of segments forms a factor function. By setting an activation threshold lower than the factor size, we simply map all state combinations of a certain size found in the receptive field into one factor value, thereby reducing the number of parameters that define the factor function. This, however, may be too restrictive, and our experiments show that the quality of the model may be further improved if the threshold is adaptively adjusted for each segment.

The threshold at the stage of segment growth determines active segments. The higher the threshold, the fewer active segments there are; thus, the higher the possibility of a new segment, and vice versa.

The approach is based on the idea that the lower the factor value, the less useful the segment is since it is proportional to the frequency of the corresponding combination of states activation. It means that if we have all segments with low factor values, we would like to form a new segment, hoping that it will capture a better combination of states. Raising a segment's threshold inversely to its factor value does exactly that; it increases the probability that a new segment will be grown in the event that there are no high-factor-value segments activated. The initial value for new segments is set to maximum, and therefore the threshold is minimal. If, during the learning process, the factor value decreases, the threshold increases, lowering the segment activation probability.

5 Experiments

DHTM is designed for online learning of the generative model. This is especially useful for Reinforcement Learning model based algorithms, where the test time adaptation is crucial. Therefore, test sequences for our new segment algorithm were gathered in the RL environments, the same, which were used for DHTM testing: Pinball and AnimalAI.

Pinball is a 2D environment created with Godot [3] game engine (Fig. 3-A). It consists of a ball, a field surrounded by walls, and a circle zone in the center capable of deflecting the ball in random directions, introducing stochasticity into the environment. At the beginning of each episode, the ball gets an initial impulse and continues to freely move, interacting with the environment.

Another set of observations was obtained from AnimalAI environment [8] (Fig. 3-B), using a fixed policy. AnimalAI is a configurable 3D environment specially designed to implement various psychological tests for animals.

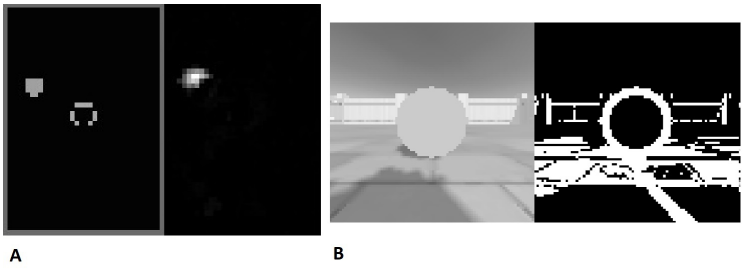

Fig. 3: Examples of raw outputs and its preprocessed version from (right) from A. the Pinball environment, B. AnimalAI.

The raw output of each episode step in the environments is an RGB image, which is preprocessed in a way to model the output of a simple event-based

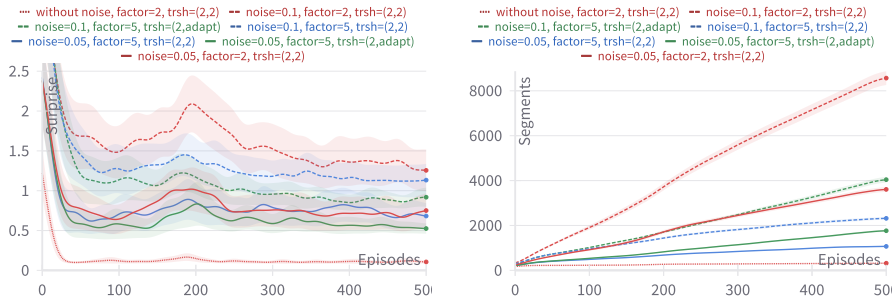

Fig. 4: Pinball environment. Different colors correspond to different variants of the factor size and thresholds: $factor = 2, trsh = (2, 2)$— optimal factor for the original version of DHTM; $factor = 5, trsh = (2, 2)$— fixed threshold (Section 4.1); $factor = 5, trsh = (2, adapt)$— adaptive threshold (Section 4.2). Differences in line style responsible for noise levels, according to legend

camera To test the model's robustness, we also add Gaussian noise to the raw image. This noise introduces random variations, simulating real-world conditions where data might contain some level of inherent noise. The preprocessed image is further fed into the same Winner-Takes-All Hebbian encoder that was used in DHTM experiments. The output of the encoder is 5 and 6 categorical random variables in the Pinball and AnimalAI environments, respectively. Those variables are then fed into the temporal memory.

For each environment, we determined the optimal factor size and threshold to compare the quality of the new segment's computational model with the best old one. We used surprise, which is the negative log-likelihood of observations, and number of segments as the performance metrics [27]. Lower surprise levels and the number of segments correspond to a better and more efficient generative model.

We compared three parameter sets for different levels of noise:

(1) Optimally determined factor size and thresholds for original DHTM segments - red line
(2) Maximum factor size and reduced thresholds, i.e., testing the idea from Section 4.1 - blue line
(3) Maximum factor size, reduced threshold at the belief-propagation stage (Section 4.1), and adaptive threshold proposed in Section 4.2 for segment-growth stage - green curve

The results for the Pinball and AnimalAI environments are presented in figures 4 and 5 respectively.

Moreover, we compared the original DHTM (1 - red line), the adaptive version (3 - green line), and the original DHTM using pruning (yellow line). Pruning was based on the maximum number of segments, set according to the number of segments at the last measured episode for the adaptive version. Segments subject

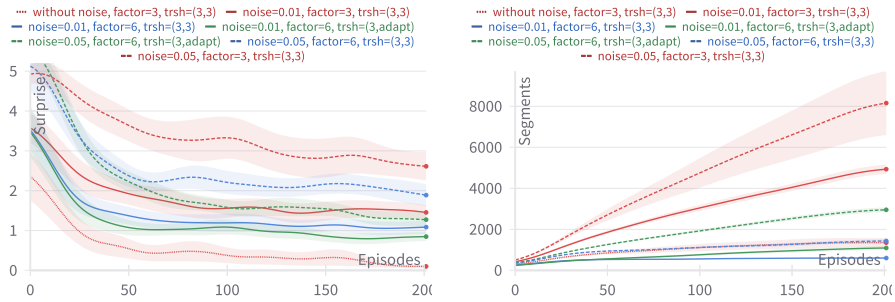

Fig. 5: AnimalAI environment. Color differentiates variants of the factor size and thresholds, the same as in pinball environment. Line style corresponds to different noise levels.

to pruning were selected based on the frequency of their use. The corresponding results are presented in Figures 6 and 7. These results demonstrate a fundamental difference between the method we propose and pruning. We achieve not only a reduction in the number of segments but also an improvement in quality, whereas pruning only degrades it.

Thus, we managed to significantly reduce the number of segments using thresholds under the size of the factor in noisy environments, and it also improved the quality of the model. It means that using a larger factor size with a lower threshold increases the model's tolerance to noise. Figures also show that the adaptive threshold results in a more robust model but slightly increases the number of grown segments in comparison to a fixed threshold.

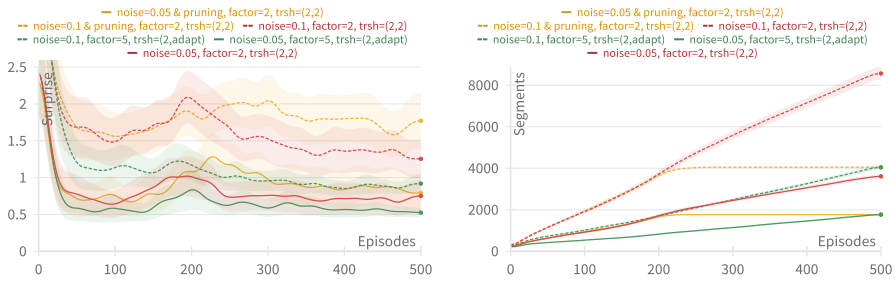

Fig. 6: Pinball environment. Comparison of the original DHTM (red line), the adaptive version (green line), and the original DHTM using pruning (yellow line). Line style corresponds to different noise levels.

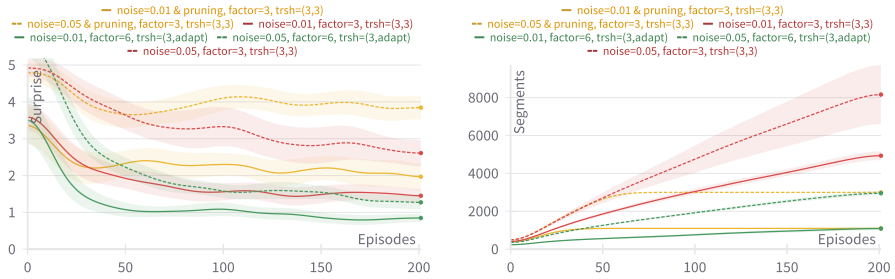

Fig. 7: AnimalAI environment. Comparison of the original DHTM, the adaptive version, and the original DHTM using pruning. Line style corresponds to different noise levels.

6 Conclusion

In this study, we have introduced innovative strategies to enhance the logical framework for constructing and utilizing segments within a biologically plausible DHTM model. Our proposed adjustments focus on augmenting the segment's capacity for generalization through a refined belief propagation threshold mechanism. Additionally, by enabling adaptability in the segment activation threshold, we have successfully amplified the segment count for receptive fields that demand heightened processing power. The implementation of these modifications has yielded significant improvements in the overall quality of the model while simultaneously curtailing the proliferation of new segments. It is evident that the enhancements we have incorporated have greatly fortified the model's efficacy and performance metrics in noisy environments. Moreover, the robustness of our introduced modifications in segment handling within this model paves the way for their seamless integration into other biologically inspired models that share the foundational logic of DHTM.

Acknowledgements This work was supported by the Ministry of Science and Higher Education of the Russian Federation under Project 075-15-2024-544.

References

1. Ahmad, S., Lavin, A., Purdy, S., Agha, Z.: Unsupervised real-time anomaly detection for streaming data. Neurocomputing **262**, 134–147 (2017). https://doi.org/https://doi.org/10.1016/j.neucom.2017.04.070, https://www.sciencedirect.com/science/article/pii/S0925231217309864, online Real-Time Learning Strategies for Data Streams
2. Bamaqa, A., Sedky, M., Bosakowski, T., Bastaki, B.B.: Anomaly detection using hierarchical temporal memory (htm) in crowd management. In: Proceedings of the 2020 4th International Conference on Cloud and Big

Data Computing. p. 37–42. ICCBDC '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3416921.3416940, https://doi.org/10.1145/3416921.3416940
3. Beeching, E., Debangoye, J., Simonin, O., Wolf, C.: Godot reinforcement learning agents. arXiv preprint arXiv:2112.03636 (2021)
4. Bengio, Y., Lee, D.H., Bornschein, J., Mesnard, T., Lin, Z.: Towards biologically plausible deep learning (2016)
5. Bianchi, F.M., Maiorino, E., Kampffmeyer, M.C., Rizzi, A., Jenssen, R.: An overview and comparative analysis of recurrent neural networks for short term load forecasting. arXiv preprint arXiv:1705.04378 (2017)
6. Chowdhary, K., Chowdhary, K.: Natural language processing. Fundamentals of artificial intelligence pp. 603–649 (2020)
7. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
8. Crosby, M., Beyret, B., Shanahan, M., Hernández-Orallo, J., Cheke, L., Halina, M.: The animal-ai testbed and competition. In: Neurips 2019 competition and demonstration track. pp. 164–176. PMLR (2020)
9. Dedieu, A., Gothoskar, N., Swingle, S., Lehrach, W., Lázaro-Gredilla, M., George, D.: Learning higher-order sequential structure with cloned hmms. arXiv preprint arXiv:1905.00507 (2019)
10. Deng, Y., Kong, S., Liu, C., An, B.: Deep attentive belief propagation: Integrating reasoning and learning for solving constraint optimization problems. Advances in Neural Information Processing Systems **35**, 25436–25449 (2022)
11. Ding, S., Xu, X., Nie, R.: Extreme learning machine and its applications. Neural Computing and Applications **25**, 549–556 (2014)
12. Duff, I.S., Erisman, A.M., Reid, J.K.: Direct methods for sparse matrices. Oxford University Press (2017)
13. Dzhivelikian, E., Kuderov, P., Panov, A.I.: Learning hidden markov model of stochastic environment with bio-inspired probabilistic temporal memory. In: Biologically Inspired Cognitive Architectures 2023. Studies in Computational Intelligence. vol. 1130, pp. 330–339
14. Dzhivelikian, E., Kuderov, P., Panov, A.I.: Learning successor representations with distributed hebbian temporal memory. arXiv preprint arXiv:2310.13391 (2023)
15. Dzhivelikian, E., Latyshev, A., Kuderov, P., Panov, A.I.: Hierarchical intrinsically motivated agent planning behavior with dreaming in grid environments. Brain Informatics **9**(1), 8 (2022)
16. Eddy, S.R.: What is a hidden markov model? Nature biotechnology **22**(10), 1315–1316 (2004)
17. Fan, D., Sharad, M., Sengupta, A., Roy, K.: Hierarchical temporal memory based on spin-neurons and resistive memory for energy-efficient brain-inspired computing. IEEE transactions on neural networks and learning systems **27**(9), 1907–1919 (2015)
18. George, D., Rikhye, R.V., Gothoskar, N., Guntupalli, J.S., Dedieu, A., LÃ¡zaro-Gredilla, M.: Clone-structured graph representations enable flexible learning and vicarious evaluation of cognitive maps. Nature Communications **12**(1), 2392 (Apr 2021). https://doi.org/10.1038/s41467-021-22559-5, https://doi.org/10.1038/s41467-021-22559-5
19. Ghahramani, Z., Jordan, M.: Factorial hidden markov models. Advances in neural information processing systems **8** (1995)

20. Hunter, K., Spracklen, L., Ahmad, S.: Two sparsities are better than one: unlocking the performance benefits of sparseâ€“sparse networks. Neuromorphic Computing and Engineering **2**(3), 034004 (jul 2022). https://doi.org/10.1088/2634-4386/ac7c8a, https://dx.doi.org/10.1088/2634-4386/ac7c8a
21. Iyer, A., Grewal, K., Velu, A., Souza, L.O., Forest, J., Ahmad, S.: Avoiding catastrophe: Active dendrites enable multi-task learning in dynamic environments. Frontiers in Neurorobotics **16** (2022). https://doi.org/10.3389/fnbot.2022.846219, https://www.frontiersin.org/articles/10.3389/fnbot.2022.846219
22. Lara-Benítez, P., Carranza-García, M., Riquelme, J.C.: An experimental review on deep learning architectures for time series forecasting. International journal of neural systems **31**(03), 2130001 (2021)
23. Leisenberger, H., Pernkopf, F., Knoll, C.: Fixing the bethe approximation: How structural modifications in a graph improve belief propagation. In: Uncertainty in Artificial Intelligence. pp. 1085–1095. PMLR (2022)
24. Moerland, T.M., Broekens, J., Plaat, A., Jonker, C.M., et al.: Model-based reinforcement learning: A survey. Foundations and Trends® in Machine Learning **16**(1), 1–118 (2023)
25. Qin, T., Chen, R., Qin, R., Yu, Y.: Improved hierarchical temporal memory for online prediction of ocean time series data. Journal of Marine Science and Engineering **12**(4), 574 (2024)
26. Rabiner, L., Juang, B.: An introduction to hidden markov models. ieee assp magazine **3**(1), 4–16 (1986)
27. Rodkin, I., Kuderov, P., Panov, A.I.: Stability and similarity detection for the biologically inspired temporal pooler algorithms **213**, 570–579
28. Salehinejad, H., Sankar, S., Barfett, J., Colak, E., Valaee, S.: Recent advances in recurrent neural networks. arXiv preprint arXiv:1801.01078 (2017)
29. Sekh, A.A., Dogra, D.P., Kar, S., Roy, P.P., Prasad, D.K.: Elm-htm guided bio-inspired unsupervised learning for anomalous trajectory classification. Cognitive Systems Research **63**, 30–41 (2020)
30. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
31. Wang, C., Zhao, Z., Gong, L., Zhu, L., Liu, Z., Cheng, X.: A distributed anomaly detection system for in-vehicle network using htm. IEEE Access **6**, 9091–9098 (2018). https://doi.org/10.1109/ACCESS.2018.2799210
32. Yang, C., Nie, K., Qiao, J., Wang, D.: Robust echo state network with sparse online learning. Information Sciences **594**, 95–117 (2022)
33. Yedidia, J.S., Freeman, W.T., Weiss, Y., et al.: Understanding belief propagation and its generalizations. Exploring artificial intelligence in the new millennium **8**(236-239), 0018–9448 (2003)
34. Zhang, Y., Li, P., Jin, Y., Choe, Y.: A digital liquid state machine with biologically inspired learning and its application to speech recognition. IEEE transactions on neural networks and learning systems **26**(11), 2635–2649 (2015)