Improving World Models using Deep Supervision with Linear Probes

Anonymous authors
Paper under double-blind review

Abstract

Developing effective world models is crucial for creating artificial agents that can reason about and navigate complex environments. In this paper, we investigate a deep supervision technique for encouraging the development of a world model in a network trained end-to-end to predict the next observation. While deep supervision has been widely applied for task-specific learning, our focus is on improving the world models. Using an experimental environment based on the Flappy Bird game, where the agent receives only LIDAR measurements as observations, we explore the effect of adding a linear probe component to the network's loss function. This additional term encourages the network to encode a subset of the true underlying world features into its hidden state. Our experiments demonstrate that this supervision technique improves both training and test performance, enhances training stability, and results in more easily decodable world features – even for those world features which were not included in the training. Furthermore, we observe a reduced distribution drift in networks trained with the linear probe, particularly during high-variability phases of the game (flying between successive pipe encounters). Including the world features loss component roughly corresponded to doubling the model size, suggesting that the linear probe technique is particularly beneficial in compute-limited settings or when aiming to achieve the best performance with smaller models. These findings contribute to our understanding of how to develop more robust and sophisticated world models in artificial agents, paving the way for further advancements in this field.

1 Introduction

Developing effective world models is a crucial aspect of artificial intelligence, as it can enable agents to make accurate predictions of how the world will unfold, as well as how their actions will influence the world, and plan their actions accordingly (Ha & Schmidhuber, 2018). World models allow agents to reason about the environment and its dynamics. In the context of AI, a ”world model” typically refers to a learned approximation of the environment's underlying dynamics, which may include physical laws, object interactions, and agent-environment relationships. By compressing sensory observations into a structured latent space, world models allow agents to reason about their surroundings and adapt to changing conditions. However, learning robust and accurate world models remains a significant challenge, especially in complex and partially observable environments.

Recent advancements in machine learning have shown promise in learning world models through end-to-end training of predictive recurrent neural networks (RNNs; Hafner et al. (2020)). These networks are trained to predict the next observation based on the current observation and action, essentially learning a model of the environment's dynamics. However, it is unclear whether such networks can develop implicit world models that capture the true underlying structure and features of the environment.

In this paper, we consider an example environment – Flappy Bird with LIDAR (Figure 1) – and investigate whether end-to-end trained predictive RNNs develop implicit world models. Crucially, we explore techniques to encourage the emergence of such models. Specifically, we focus on the following research question: Can an addition of a linear probe to the network's loss function, which encourages the decoding of true world features, improve the network's performance and the quality of the learned world model? This technique lies within the broader class of deep supervision (Lee et al., 2015; Li et al., 2022), but its application to the problem of world model learning remains relatively unexplored.

By investigating these questions and techniques, we aim to contribute to the understanding and development of more effective world models in artificial intelligence, ultimately leading to more capable and adaptable agents.

1.1 Related Work

Deep supervision techniques were first introduced by (Lee et al., 2015) and were extensively applied in training vision models (Al-Barazanchi et al., 2016; Ai et al., 2017; Zou et al., 2023), with prominent applications in the field of medical imaging (Zhou et al., 2017; Lin et al., 2021; Mishra et al., 2019; Lei et al., 2018), image super-resolution Kim et al. (2016); Lai et al. (2019) and segmentation (Peng & Wang, 2021; Yan et al., 2022). For a comprehensive review, see Li et al. (2022).

Learning recurrent world models to learn behaviors in the latent space for downstream applications was introduced by Ha & Schmidhuber (2018), and advanced by the Dreamer sequence of models (Hafner et al., 2020; 2022; 2024) as well as other techniques for learning world models such as contrastive learning (Kipf et al., 2020) and applications of pretrained models of the world to physical settings such as robotics (Wu et al., 2023; Firoozi et al., 2024)

Representations of continuous variables in recurrent neural networks (RNNs), as well as mechanisms for their storage and evolution are extensively studied in the field of computational neuroscience (Khona & Fiete, 2022; Noorman et al., 2024; Agmon & Burak, 2023; Burak & Fiete, 2009; Darshan & Rivkind, 2022), investigating stable attractor mechanisms (Cueva et al., 2021), as well as periodic or quasi-periodic ones (Park et al., 2023).

2 Methods

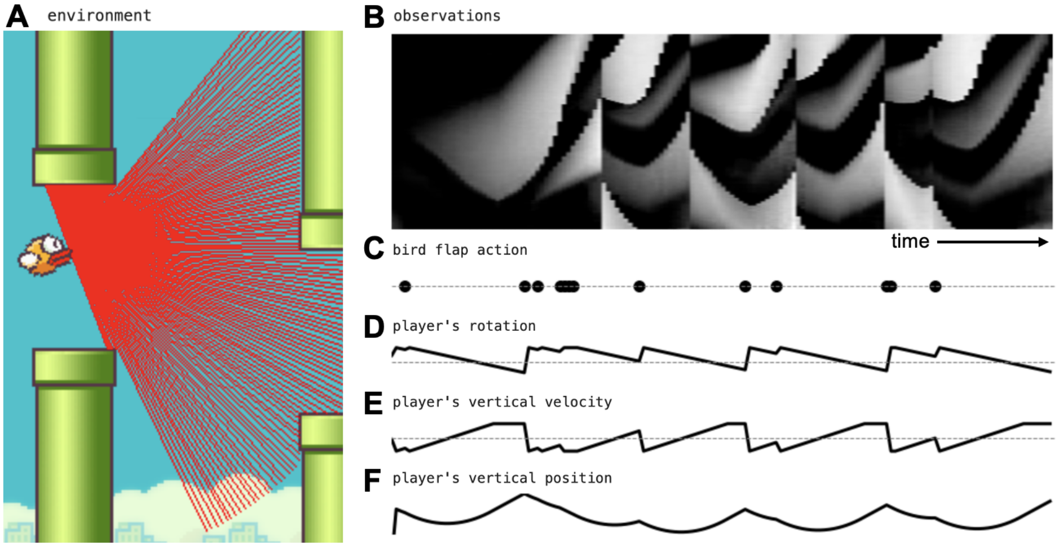

Figure 1: Flappy Bird environment with lidar. (A) The environment. (B) The agent only observes the lidar signal as a function of time. (C) The only available actions are no-op and flap. (D-F) The environment provides true variables of the world, such as the player's rotation angle, vertical velocity, and position.

2.1 Environment and Task Description

We use an experimental environment based on the Flappy Bird game[1] (Figure 1A), where the bird must successfully pass through layers of pipes by performing either a no-op action or a flap action, which propels the bird upward (Figure 1C). The agent receives only LIDAR measurements as observations, represented by a 180-dimensional vector indicating the distance to the nearest obstacle for each ray angle (Figure 1B). Separately, the environment also provides the true underlying world features, such as the bird's rotation, vertical velocity, and position (Figure 1D-F), as well as information about the pipes.

The agent (bird) receives reward every time it successfully passes through a pair of pipes. The episode ends whenever the bird touches any obstacle (pipe, floor, or ceiling). However, in this paper we focus exclusively on the world model aspect of this environment, and so the rewards are not considered.

For the experiments in this paper, we use 10,000 training rollouts of the environment collected by an expert DQN policy (which turns into a random policy with probability 1% every timestep; this randomness was included to encourage off-policy exploration). For testing, we use 2,000 rollouts of the DQN policy and 2,000 rollouts from a random policy (which chooses action ”flap” with probability 7.5% every timestep).

2.2 Network Architecture and Training

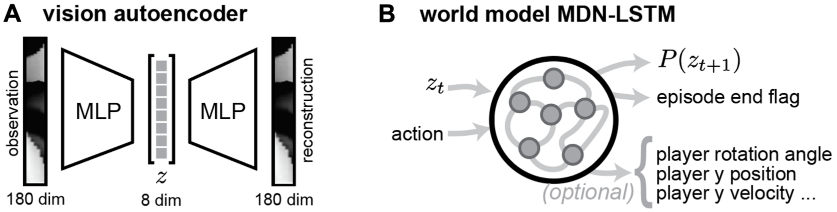

Figure 2: Network architecture and training setup. (A) The vision autoencoder compresses the 180-dimensional raw observations into an 8-dimensional latent space. (B) The world model MDN-LSTM takes the current latent observation vector and action as inputs, and predicts the distribution of the next latent vector and an episode end flag. Optionally, the LSTM's hidden state is encouraged to capture true world variables (such as player rotation angle, y position, y velocity, etc.) through a linear probe.

The network architecture is based on that of the world models in the paper by Ha & Schmidhuber (2018).

First, raw 180-dimensional observations are transformed into an 8-dimensional latent space using an autoencoder (Figure 2A). The autoencoder consists of an encoder multi-layer perceptron (MLP) E that maps the high-dimensional observations to the lower 8-dimensional latent space (outputting the mean and variance of an 8-dimensional factored gaussian distribution), and a decoder MLP D that reconstructs the original observations from the latent representation.

The autoencoder is trained on the collected rollouts, with L2 regularization, with the following loss:

$$L_{AE} = \sum_i (\hat{x}_i - x_i)^2 + \frac{1}{2} \sum_i \|E(x_i)\|^2, \tag{1}$$

where

$$\hat{x}_i = D(z_i); \quad z_i \sim \mathcal{N}(\mu_i, \sigma_i^2); \quad E(x_i) = (\mu_i, \sigma_i). \tag{2}$$

After training, the weights of the autoencoder are frozen.

[1] The original implementation of the environment can be found at https://github.com/markub3327/flappy-bird-gymnasium.

The world model is a long short-term memory RNN (LSTM; Hochreiter & Schmidhuber (1997); Figure 2B). This RNN takes as input the current latent observation vector (z_t) and action a_t, and outputs the distribution of the next latent vector ($P(\tilde{z}_{t+1})$) and an episode end flag $\tilde{e}_{t+1}$. The distribution of latent vectors is represented using a mixture of 5 Gaussian variables (following the implementation of the mixture density network; Graves (2014); Bishop (1994)) – that is, the network outputs 5 tuples $(\mu^{(i)}, \sigma^{(i)})$ to represent the distribution $P(\tilde{z}_{t+1})$.

The loss function is

$$L_{pred} = -\frac{1}{n}\sum_t \log\left[\frac{1}{5}\sum_{i=1}^{5} P(z_{t+1} \mid \mu_t^{(i)}, \sigma_t^{(i)})\right] + \frac{10}{n}\sum_t (e_t - \tilde{e}_t)^2, \tag{3}$$

where n is the number of timesteps in an episode.

Our key contribution is the (optional) addition of linear probes to the network, which attempt to decode the true underlying world features from the LSTM's hidden state. The loss function is modified to include a term proportional to the mean squared error of the linear probes, with a hyperparameter multiplier λ. The total loss is the sum of the predictive loss and the linear probe mean squared error:

$$\begin{aligned} L_{total} &= L_{pred} + \lambda \cdot MSE(\text{linear probe}) && (4) \\ &= L_{pred} + \lambda \sum_{k,t} \left(W_k r_t - f_{k,t}\right)^2, && (5) \end{aligned}$$

where W_k are the linear weights for world feature k; r_t are the features in the recurrent layer of the LSRM at time t, and $f_{k,t}$ is the value of the world feature k for the timestep t.

3 Results

3.1 Improved Next State Prediction Loss

To assess the effects of adding linear probes during training, we trained world model RNNs for various values of λ (20 random seeds for each choice of λ). If $\lambda = 0$, the setting is equivalent to training on the predictive loss only, without any linear probe component.

Perhaps paradoxically, increasing the hyperparameter λ, which put more weight on the linear probe component of the loss, did not hurt the predictive loss component. Instead, both the training and test predictive losses decreased as λ was increased (Figure 3A-C). In fact, the predictive loss as a function of λ continued to decrease through values as high as $\lambda = 64$.

This result suggested that the addition of the linear probe component to the loss function improved the network's performance without compromising its predictive capabilities. For the rest of this paper, only networks with $\lambda = 0$ and $\lambda = 64$ will be considered and compared together.

3.2 Improved Decodability of World Features

To assess the sophistication of the world model, we examined the performance of linear probes in decoding true world variables from the network's hidden state.

As expected, the world features that were included in the linear probe loss during training are more easily decodable for $\lambda = 64$ than for $\lambda = 0$ (Figure 4; last three panels – player's vertical position, velocity, and rotation angle).

Interestingly however, only for $\lambda = 64$, even features which were not explicitly part of the loss function exhibited decodability above that of an untrained randomly-initialized network (Figure 4; all panels). This finding indicated that the addition of the linear probe component encouraged the network to develop a more comprehensive representation of the underlying world features, perhaps developing a more sophisticated world model.

3.3 Reduced Distribution Drift

Next, we tested the network's ability to predict how the latent state changes over time, without receiving constant feedback input from the environment. Specifically, we chose a certain timestep,

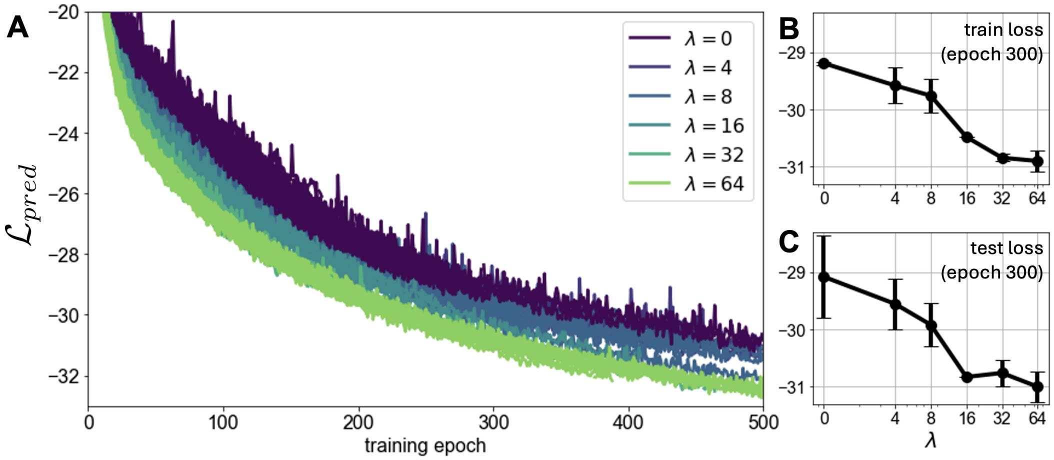

Figure 3: The effect of increasing the linear probe weight λ on the original (next latent state prediction) loss. Both training (A, B) and test (C) predictive losses decrease as λ increases. For every choice of λ, 20 RNNs were initialized from different random seeds. Error bars in panels (B) and (C) indicate the s.d.

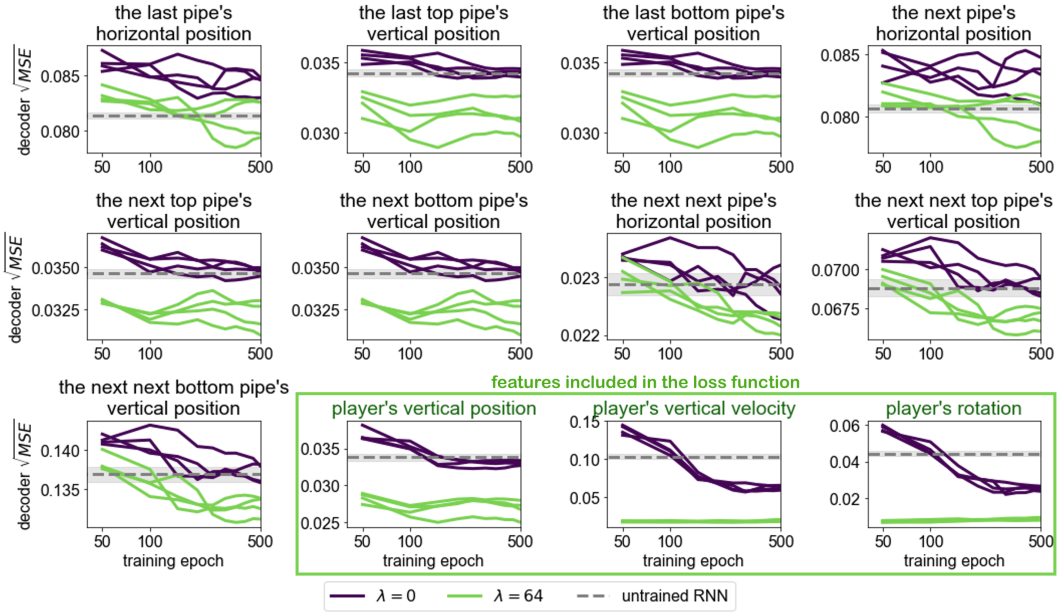

Figure 4: Decodability of world features from the network's hidden state for $\lambda = 0$ and $\lambda = 64$. Every panel represents one world feature. The last three panels (player's vertical position, velocity, and rotation; green) represent features which were explicitly included in the loss function for $\lambda = 64$. Note that even the features which were not explicitly part of the losss function exhibit decodability above that of an untrained randomly-initialized network, but only for $\lambda = 64$. For every choice of λ, four RNNs initialized from different random seeds are shown.

after which the network was cut off from the environment (stopped receiving the true input latent vector every timestep). Instead, after the cutoff timestep, we sampled the next predicted latent state from the network's output and fed it back as its input in the subsequent step.

Figure 5 compares the distribution drift between networks trained with and without the linear probe. Networks trained with the linear probe ($\lambda = 64$) exhibited smaller growth in the loss of predicting the real latent state, indicating reduced distribution drift, but only for timesteps when the bird was flying between successive pipe encounters (t=10 and t=60; Figure 5A, C). Interestingly, during the timesteps when the bird was flying through a pair of pipes, this result did not hold. This may have to do with the fact that the changes in game state are much less predictable in those timesteps.

The results were qualitatively equivalent for both the training policy (DQN) and a random policy (Figure 5D-F). This finding suggests effectiveness of the linear probes in maintaining a more accurate prediction of the latent observation state over time.

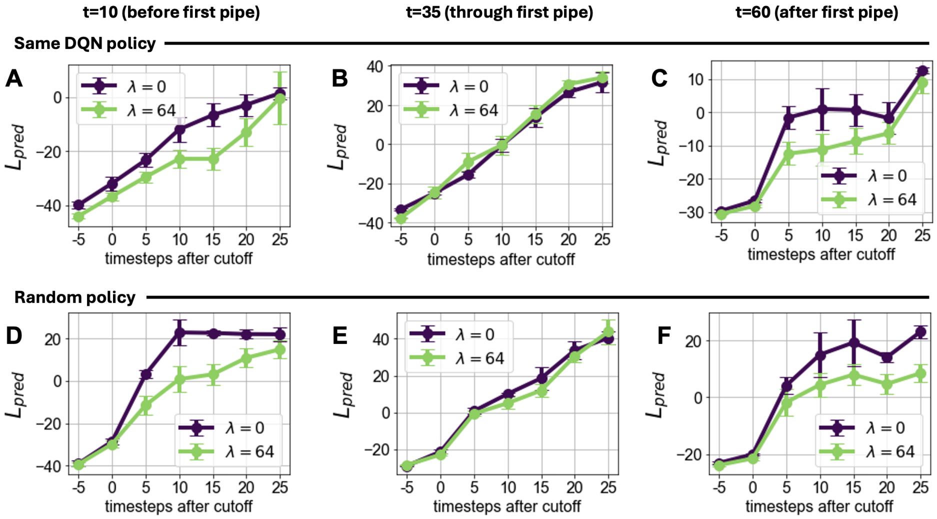

Figure 5: Distribution drift comparison. Networks trained with the linear probe ($\lambda = 64$) exhibit reduced distribution drift compared to those without the probe ($\lambda = 0$) for both the training policy and a random policy. However, note that this result does not hold for the timesteps near t=35, which correspond to the bird going through the first pair of pipes. The error bars represent s.d. across 10 random seeds.

3.4 Scaling Properties

The predictive component of the loss followed a scaling law with respect to training time (Figure 6), for both $\lambda = 0$ and $\lambda = 64$. However, the scaling law for $\lambda = 64$ was smoother, as well as shifted down, compared to the one for networks trained with $\lambda = 0$.

The loss of networks with and without the linear probe also followed a scaling law with respect to model size (Figure 7). However, the curve for networks trained with $\lambda = 64$ was shifted down, such that a network of a given size trained with the linear probe achieves performance roughly equivalent to a network twice the size trained without the probe. This result held true at both epoch 250 and 500.

This finding suggested that the linear probe component provides a consistent performance benefit across different stages of training and different model sizes.

3.5 Training Stability

Finally, adding the linear probe mean squared error component to the loss function improved training stability (Figure 8). It resulted in fewer instances of exploding gradients and training divergences, with the divergences additionally happening later in training for $\lambda = 64$ than for $\lambda = 0$. Networks trained with the linear probe are more likely to reach epoch 500 without diverging (30% vs 7.5% for RNNs of size 128, 62.5% vs 10% for RNNs of size 256; Figure 8A-B, right).

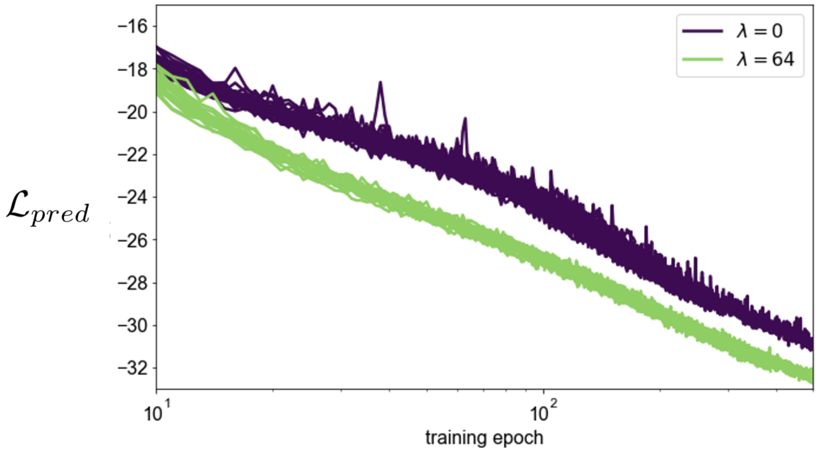

Figure 6: Scaling law of predictive loss with respect to training time. The predictive loss (L_{pred}) is plotted against the training epoch on a log scale for two different values of the linear probe weight: $\lambda = 0$ (purple) and $\lambda = 64$ (green). This plot highlights that while the linear probe improves the network's performance, it does not increase the slope of the scaling law of predictive loss with training time. Training curves of 20 randomly-initialized networks is plotted for every choice of λ.

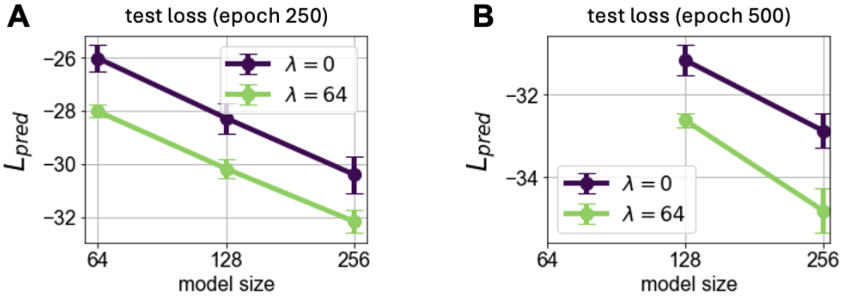

Figure 7: Scaling laws for model size. The performance of networks with and without the linear probe follows a scaling law, but the curve for networks trained with $\lambda = 64$ is shifted down, indicating better performance for a given model size. This holds true at both (A) epoch 250 and (B) epoch 500.

4 Discussion

In this study, we sought to encourage the development of world models in a recurrent neural network trained end-to-end to predict the next observation of an agent. Specifically, we considered adding a term to the loss function of the network, which was proportional to the mean squared error of linear probes trying to decode the true world features from the hidden state of the model.

In our experiments, this addition demonstrated improved training and test loss, enhanced training stability, and more easily decodable world features. We also observed reduced distribution drift in certain cases for models trained with the linear probe. The scaling laws for both model size and training time maintained the same slope both with and without the additional of linear probe loss, consistent with findings in the literature on scaling laws for neural networks (Kaplan et al., 2020). However, the networks trained with the linear probes had the loss curve shifted down, such that, for example, a network of a given size trained with the linear probe achieved performance roughly equivalent to a network twice the size trained without the probe. This technique may prove especially advantageous in compute-limited settings or when aiming to achieve the best performance with smaller models, for example when deploying models on-site in medical imaging settings (Zhou et al., 2017; Lin et al., 2021; Mishra et al., 2019; Lei et al., 2018).

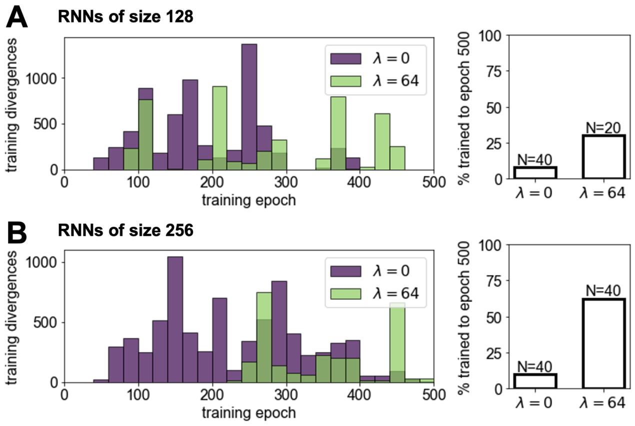

Figure 8: Training stability with and without the linear probe. Networks trained with the linear probe are more likely to reach epoch 500 without diverging. Here, results are shown for RNNs of hidden size 128 (A) and 256 (B).

Our findings support the annealing technique, wherein the linear probe loss term is applied only during the initial stages of training and gradually diminished to zero. The analysis of the loss curve (Figure 6) shows that the slope stabilizes after a certain number of steps, indicating that the linear probes have the most pronounced impact during the early stages of training when the network is actively learning its foundational representations. This suggests that the technique might be particularly beneficial at the beginning of training and less critical in later stages. By focusing the influence of the probes early on, this approach enhances the decodability of world features without imposing a lasting computational burden. Furthermore, transitioning to normal training procedures after the initial phase would remove the additional data requirement, making this technique both efficient and scalable. Testing this phased application of the technique is a promising direction for future work.

Moreover, the concept of leveraging additional sensory inputs aligns with applications in robotic systems (Wu et al., 2023). When a robot is augmented with inexpensive, additional sensors, decoding these inputs through the network's latent representations can enhance its ability to interpret and model the environment. This process facilitates improved learning by enriching the internal state of the model with diverse sensory information. This technique is thus promising for scalable and cost-effective advances in robotics, where this addition of sensors can unlock significant performance gains, while allowing the models to be smaller and therefore run faster (Firoozi et al., 2024).

Overall, this study contributes to our understanding of how to encourage the development of robust world models in end-to-end trained predictive networks. Further research in this direction may yield valuable insight for advancing the field of artificial intelligence and developing more capable agents.

References

Haggai Agmon and Yoram Burak. Simultaneous embedding of multiple attractor manifolds in a recurrent neural network using constrained gradient optimization. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 16770–16782. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/361e5112d2eca09513bbd266e4b2d2be-Paper-Conference.pdf`.

Baole Ai, Yu Zhou, Yao Yu, and Sidan Du. Human pose estimation using deep structure guided learning. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1224–1231, 2017. URL `https://api.semanticscholar.org/CorpusID:27366452`.

Hussein A. Al-Barazanchi, Hussam Qassim, and Abhishek Verma. Novel cnn architecture with residual learning and deep supervision for large-scale scene image categorization. *2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 1–7, 2016. URL `https://api.semanticscholar.org/CorpusID:6454063`.

Christopher M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Neural Computing Research Group, Dept. of Computer Science and Applied Mathematics, Aston University, Birmingham, B4 7ET, U.K., February 1994. URL http://www.ncrg.aston.ac.uk/. Previously issued as NCRG/94/4288.

Yoram Burak and Ila R. Fiete. Accurate path integration in continuous attractor network models of grid cells. *PLoS Computational Biology*, 5(2):e1000291, February 2009. doi: 10.1371/journal.pcbi.1000291. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000291.

Christopher J. Cueva, Adel Ardalan, Misha Tsodyks, and Ning Qian. Recurrent neural network models for working memory of continuous variables: activity manifolds, connectivity patterns, and dynamic codes, 2021. URL https://arxiv.org/abs/2111.01275.

R. Darshan and A. Rivkind. Learning to represent continuous variables in heterogeneous neural networks. *Cell Reports*, 39(1):110612, April 2022. doi: 10.1016/j.celrep.2022.110612.

R. Firoozi, J. Tucker, S. Tian, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, 0(0), 2024. doi: 10.1177/02783649241281508.

Alex Graves. Generating sequences with recurrent neural networks, 2014. URL https://arxiv.org/abs/1308.0850.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination, 2020. URL https://arxiv.org/abs/1912.01603.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models, 2022. URL https://arxiv.org/abs/2010.02193.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL https://arxiv.org/abs/2301.04104.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

M. Khona and I. R. Fiete. Attractor and integrator networks in the brain. *Nature Reviews Neuroscience*, 23:744–766, 2022. doi: 10.1038/s41583-022-00642-0. URL https://doi.org/10.1038/s41583-022-00642-0.

Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1637–1645, 2016. doi: 10.1109/CVPR.2016.181.

Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models, 2020. URL https://arxiv.org/abs/1911.12247.

Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2599–2613, 2019. doi: 10.1109/TPAMI.2018.2865304.

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-Supervised Nets. In Guy Lebanon and S. V. N. Vishwanathan (eds.), *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pp. 562–570, San Diego, California, USA, 09–12 May 2015. PMLR. URL https://proceedings.mlr.press/v38/lee15a.html.

Haijun Lei, Tao Han, Feng Zhou, Zhen Yu, Jing Qin, Ahmed Elazab, and Baiying Lei. A deeply supervised residual network for hep-2 cell classification via cross-modal transfer learning. *Pattern Recognition*, 79:290–302, 2018. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2018.02.006. URL https://www.sciencedirect.com/science/article/pii/S0031320318300608.

Renjie Li, Xinyi Wang, Guan Huang, Wenli Yang, Kaining Zhang, Xiaotong Gu, Son N. Tran, Saurabh Garg, Jane Alty, and Quan Bai. A comprehensive review on deep supervision: Theories and applications, 2022. URL https://arxiv.org/abs/2207.02376.

M. Lin, S. Momin, Y. Lei, H. Wang, W. J. Curran, T. Liu, and X. Yang. Fully automated segmentation of brain tumor from multiparametric mri using 3d context deep supervised u-net. *Medical Physics*, 48(8):4365–4374, August 2021. doi: 10.1002/mp.15032. Epub 2021 Jul 11.

D. Mishra, S. Chaudhury, M. Sarkar, and A. S. Soin. Ultrasound image segmentation: A deeply supervised network with attention to boundaries. *IEEE Transactions on Biomedical Engineering*, 66(6):1637–1648, June 2019. doi: 10.1109/TBME.2018.2877577. Epub 2018 Oct 22.

M. Noorman, B. K. Hulse, V. Jayaraman, et al. Maintaining and updating accurate internal representations of continuous variables with a handful of neurons. *Nature Neuroscience*, 27: 2207–2217, 2024. doi: 10.1038/s41593-024-01766-5. URL https://doi.org/10.1038/s41593-024-01766-5.

Il Memming Park, Ábel Ságodi, and Piotr Aleksander Sokół. Persistent learning signals and working memory without continuous attractors, 2023. URL https://arxiv.org/abs/2308.12585.

Jialin Peng and Ye Wang. Medical image segmentation with limited supervision: A review of deep network models. *IEEE Access*, 9:36827–36851, 2021. doi: 10.1109/ACCESS.2021.3062380.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 2226–2240. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/wu23c.html.

B. Yan, M. Cao, W. Gong, et al. Multi-scale brain tumor segmentation combined with deep supervision. *International Journal of Computer Assisted Radiology and Surgery*, 17:561–568, 2022. doi: 10.1007/s11548-021-02515-w.

Yuyin Zhou, Lingxi Xie, Elliot K. Fishman, and Alan L. Yuille. Deep supervision for pancreatic cyst segmentation in abdominal ct scans, 2017. URL https://arxiv.org/abs/1706.07346.

Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey, 2023. URL https://arxiv.org/abs/1905.05055.