Can Wikipedia Teach Us About the World?
Impact of Language-Pretraining on Model-Based RL

Alex Gilbert
Department of Electrical Engineering
Stanford University
alexrg@stanford.edu

1 Introduction

With the recent frenzy of excitement around the complex reasoning capabilities of transformers [1] and large-scale foundation model's, there have been a growing number of efforts attempting to apply these skills to sequential decision-making and planning, especially for robotics tasks. However, current approaches which attempt to capture the sequence-modeling power of transformers for decision-making encounter two complimentary problems:

- Training agents from scratch with large-scale data requires huge task-diversity and often still fails to generalize to far out-of-domain tasks [2, 3, 4].
- Pre-trained LLM's benefit from massive, generalizable data, but aren't *grounded* in any task environment, and therefore struggle with generating realistic plans without the help of complicated, hand-crafted systems. [5, 6, 7].

Nonetheless, it is undeniable that LLM's contain extensive, general knowledge about the world and powerful modeling capacity. We therefore propose an alternative strategy for harnessing those strengths for planning: as a world-model in a model-based reinforcement learning (MBRL) framework. We posit that relying on LLM's purely for their information about the world—leaving the decision-making to a task-specific agent—will allow us to avoid relying on their limited ability to convert knowledge to action, while optimally utilizing the world-knowledge they contain.

In this project, we undertake an initial exploration into the utility of LLM's as world-models for MBRL. Specifically, we plug an off-the-shelf LLM into an existing model-based agent with a transformer world model [8], and apply task-specific fine-tuning. We then probe the weights and internal representations learned by the fine-tuned world-model compare with those learned by a randomly-initialized baseline trained from scratch. While our approach is relatively naive, the impacts of language-pretraining revealed by our analysis serve as an initial validation of our theory that LLM's indeed have some utility as agent world-models, and that representations learned from language can be helpful for sequential decision-making. In future work, we hope to build on this insight by designing a more principled integration of LLM's with MBRL which explicitly capitalizes on the language understanding they possess.

2 Prior Work

2.1 LLM's as Planning Agents

There have been a growing number of efforts attempting to utilize large-scale generative foundation models for sequential decision-making, particularly in embodied robotics contexts. Much of this work has focussed on applying their strengths in sequence-modeling directly to online planning [4, 2, 9, 3], which has enabled some impressive results in real robotic environments. However, as discussed in section 1, such methods generally require massive datasets with abundant task variation that are

Preprint. Under review.

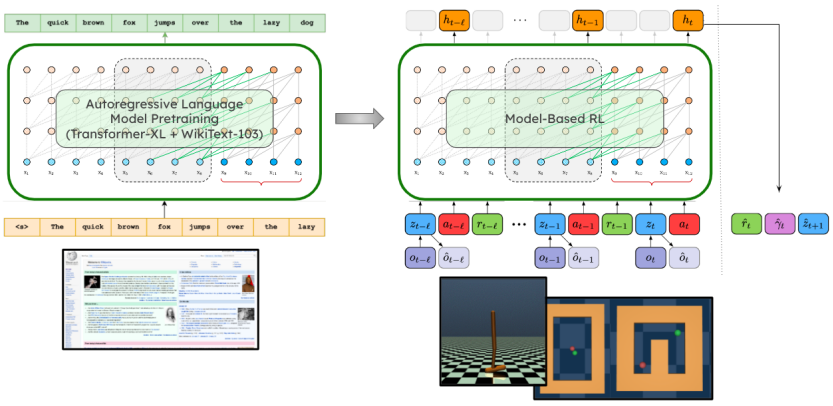

Figure 1: We fine-tune a language-pretrained transformer model to capture world-dynamics during task training.

prohibitively expensive to collect, and often still fail to enable robust generalization. Alternatively, various studies have tried to take advantage of the zero-shot and in-context learning capabilities of LLM's, attempting to coax out emergent planning behaviors [5, 6, 10, 7, 11, 12]. These solutions tend to involve complicated engineering to manually frame the environment in language for compatibility with the agent's domain, which limits their application beyond the specific task they are designed for. In this work we aim to maintain the generality of the directly learned approach, while also benefiting from the information in off-the-shelf LLM's.

2.2 Transformers for RL

The ability of transformers [1] to model high-dimensional distributions of semantic concepts at scale has enabled unprecedented success in versatile natural language processing (NLP) tasks when combined with internet-scale pre-training [13, 14]. However, the insight that planning in a reinforcement learning (RL) paradigm can be reduced to a supervised, sequence modeling task has enabled a number of methods to move beyond the NLP space and use transformer architectures to design agents for offline [15, 2] and online [16] RL. Inspired by their model-free counterparts, concurrent works [8, 17] applied a similar strategy to MBRL, building on the *Dreamer* framework [18, 19, 20] by implementing the world-model with a transformer-based architecture. Very relevant to our approach, the authors in [21, 22] experimented with replacing model-free transformer-based agents with LLM's. Here, we attempt to apply their methodology to MBRL.

3 Methods

3.1 Transformer-Based World Models

Robine et al. [8] augment the popular *Dreamer* [19] model-based RL framework with a TransformerXL [23] to model latent world dynamics. Note, TransformerXL is essentially a standard transformer augmented with a unique recurrence mechanism (to capture longer context information) and relative positional embeddings. We investigate the impact of language-pretraining on the learned dynamics.

Using a small number of environment interactions, the algorithm *iteratively* learns a stochastic latent representation of the observation space with a VQ-VAE,

$$\text{Observation Encoder: } z_t \sim p_\phi(z_t|o_t)$$
$$\text{Observation Decoder: } \hat{o}_t \sim p_\phi(\hat{o}_t|z_t),$$

a stochastic dynamics model (implemented with the TransformerXL architecture) which predicts the reward/episode termination/next latent-state given a state-action history,

$$\text{Aggregation Model (}\textit{TransformerXL}\text{): } h_t = f_\psi(z_{t-l:t}, a_{t-l:t}, r_{t-l:t-1}),$$
$$\text{Reward Predictor: } \hat{r}_t \sim p_\psi(\hat{r}_t|h_t),$$
$$\text{Discount Predictor: } \hat{\gamma}_t \sim p_\psi(\hat{\gamma}_t|h_t),$$
$$\text{Next Latent-State Predictor: } \hat{z}_{t+1} \sim p_\psi(\hat{z}_{t+1}|h_t),$$

and finally a policy and value function,

$$\text{Policy: } \hat{a}_t \sim \pi_\theta(\hat{a}_t|z_t)$$
$$\text{Value Function: } \hat{v}_t \sim V_\theta(\hat{v}_t|z_t),$$

using advantage actor-critic[24] *exclusively in 'imagination'* (i.e. replacing the environment with the learned dynamics model).

3.2 Language Pre-Training

In our method, we simply swap the randomly initialized aggregation model (see section 3.1) with a language-pretrained version. We use the *Transformer-XL Standard* checkpoint reported in [23] and implemented on the HuggingFace[25] repository (`https://huggingface.co/transfo-xl-wt103`). Note, we drop the learned token embeddings and learn new embeddings from scratch, as in [8].

Num. Layers	Num. Attention Heads	Num. Parameters
18	16	88M

Table 1: The dimensions of the `transfo-xl-wt103` checkpoint used in our experiments.

The model is trained for next-token prediction on *WikiText-103* [26], a word-level language modeling dataset with long-term dependencies.

Num. Wiki Articles	Avg Tokens per Article	Num. Total Tokens
28K	3.6K	103M

Table 2: Statistics of the *WikiText-103* dataset used for language pre-training.

3.3 Task-Based Fine-tuning

For this preliminary exploration, we restrict our experiment to an overly-simplistic toy task—*CartPole*, from the OpenAI-Gym[27]—which both agents (pre-trained and randomly-initialized) can solve easily, in order to focus primarily on analyzing the differences in learned parameters and representations.

Environment	Observation Space	Action Space	Max Episode length	Max/Expert Score
CartPole	`Box(4)`	`Discrete(2)`	500	500

Table 3: Parameters of the test environment used for our experiments.

To train our agents, we follow the training procedure outlined by Robine et al. [8]. We iterate between alternating phases of world model optimization and actor-critic training. During the world modeling phase, we sample trajectories collected during exploration and optimize the world model components for a set of predictive and variational objectives. During agent optimization, we follow the steps of standard actor-critic methods, however trajectories are exclusively executed in imagination. For *CartPole* we allow the agent 20K total interactions with the environment, and utilize a transformer context length of 16 state-action pairs. Pseudocode is provided in figure 3.3.

Algorithm 1 Training the world model and the actor-critic agent.

```
function train_world_model()
   // sample sequences of observations,
   // rewards, actions and discounts
   o,a,r,d = sample_from_dataset()
   z = encode(o)
   o_hat = decode(z)
   h = transformer(z,a,r)
   r_hat,d_hat,z_hat = predict(h)

   // optimize world model via
   // self-supervised learning
   optim_observation(o,z,o_hat,z_hat)
   optim_dynamics(r,d,z,r_hat,d_hat,z_hat)

   // z will be used for imagination
   return z
```

```
function train_actor_critic(z)
   // imagine trajectories of states,
   // rewards, actions and discounts;
   // use z as starting point
   imag = [z]
   for t = 0 until H do
      a = actor(z)
      imag.append(a)
      h = transformer(imag)
      r,d,z = predict(h)
      imag.extend([r,d,z])

   // optimize actor-critic via
   // reinforcement learning
   optim_actor_critic(imag)
```

3.4 Post-Training Analysis

3.4.1 Model Activations

To compare the representations learned by randomly-initialized and language-pretrained models during task fine-tuning, we measure the centered kernel alignment (CKA) [28] of each layer (including FFN's within each transformer attention block) of each model's activations from before and after fine-tuning. To do this, we use a sampling of offline CartPole trajectories from [29], feed them through the original and fine-tuned models (for both initialization methods), and record the output activations for the final 'action' token in each trajectory (note, we use the fine-tuned state and action embedding networks for both pre- and post- fine-tuning models). The CKA is computed between the $R^{mtimesd}$ matrices, where m is the number of sampled trajectories, and d is the length of a given layer's vectorized activation matrix.

3.4.2 Model Parameters

In order to identify differences in the learned weights between the randomly-initialized and language-pretrained world models, we perform a similar calculation as described in section 3.4.1, replacing the activations with the actual vectorized layer parameters, and measuring L2-norm and cosine similarity instead of CKA.

4 Results

4.1 Task Performance

We compare our method (i.e. using a language-pretrained transformer as a world-model) with the baseline, *TWM*, from Robine et al. [8]. Rewards are collected by running the converged policy for 100 evaluation episodes, taking the average total reward, and normalizing by the maximum/expert score for the particular environment. Results are shown in table 4.1. As mentioned above (section 3.3), because our environment is so simple, both methods are able to achieve nearly perfect performance (i.e. maintaining a balanced pole for the entire episode length in almost all episodes). We therefore focus primarily on analyzing the learned representations.

	Normalized Avg. Evaluation Reward	
Environment	**TWM**	**Ours**
CartPole	96.0	95.0

4.2 Learning Analysis

4.2.1 Weight Change

The comparison between each layer's initialized and fine-tuned *parameters* for both initialization methods, as described in section 3.4.2, is plotted (by relative layer depth) in figure 4.2.1. From

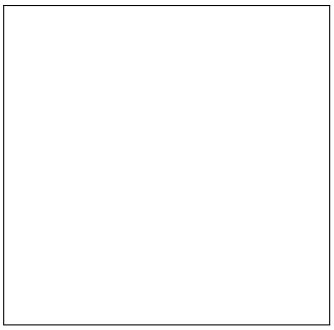

Figure 2: L2-Norm and cosine similarities between pre- and post-fine-tuning world model weights at each layer.

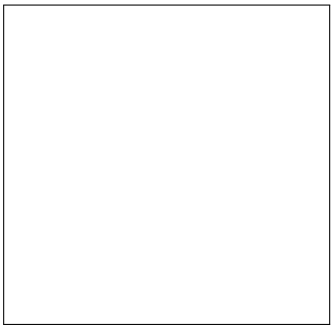

Figure 3: Centered kernel alignment between pre- and post-fine-tuning world model activations at each layer.

the plot we can see that for the randomly-initialized baseline, model parameters exhibit significant changes (i.e. high L2-norm, low cosine similarity) uniformly throughout the model. In contrast, the language-pretrained weights show consistently less adaptation as a result of fine-tuning, with the majority of change concentrated in a few shallow layers.

4.2.2 Activation Similarity

The CKA of each layer's activations before vs. after task training for both methods (calculated with the procedure described in section 3.4.1) is plotted in figure 4.2.2. For model representations, the opposite relationship with pre-training is apparent–the activations tend to become significantly less aligned much earlier in the model stack than they do for the randomly-initialized model.

5 Discussion and Conclusions

5.1 Impacts of Language Pre-Training

In the plots presented in section 4.2, we observe two key trends:

- Pre-trained models show notably less adjustment to their weights during fine-tuning, relative to the randomly initialized baseline—primarily confined to relatively early layers in the model.
- Randomly-initialized world models lead to less dramatic differences in activation similarity throughout most of the model depth.

While further analysis is certainly required to gain more confidently explanation these phenomena, one plausible explanation which addresses both observations is that the higher-level representations learned during language-based training do in fact have utility for this decision-making task. This fact would explain why the parameter changes were minimal outside a few early layers (potentially transforming the environment specific features into the language modeling mid-level latent space).

Moreover, these critical, early adjustments would likely cause the significant change in activations that we see starting early on and cascading through the remaining model layers, whereas the more uniform parameter adjustments in the randomly-initialized world-model would result in more gradual effects on model activations. If this explanation is indeed the case, then it stands to reason that language-pretraining can be a valuable tool for learning efficient and effective world-models.

5.2 Limitations and Future Work

With this initial exploration, we introduce the idea of using LLM's as a world-model, and find evidence in a toy example that language-pretraining can be useful for decision-making performance. However, the naive method with which we integrate LLM's into our model-based framework, and the overly simplistic environment that we test, leave many questions about this strategy open. In follow-up works, we first plan to repeat these procedures on more complicated environments and with more powerful world-models to further test the hypothesis put forth in this work, before ultimately designing a more principled approach to LLM and MBRL fusion.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[2] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent, 2022.

[3] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2022.

[4] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023.

[5] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can, not as i say: Grounding language in robotic affordances, 2022.

[6] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents, 2022.

[7] Wenlong Huang, Fei Xia, Dhruv Shah, Danny Driess, Andy Zeng, Yao Lu, Pete Florence, Igor Mordatch, Sergey Levine, Karol Hausman, and Brian Ichter. Grounded decoding: Guiding text generation with grounded models for robot control, 2023.

[8] Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions, 2023.

[9] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts, 2023.

[10] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time, 2022.

[11] Boyuan Chen, Fei Xia, Brian Ichter, Kanishka Rao, Keerthana Gopalakrishnan, Michael S. Ryoo, Austin Stone, and Daniel Kappler. Open-vocabulary queryable scene representations for real world planning, 2022.

[12] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation, 2022.

[13] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[14] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[15] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021.

[16] Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer, 2022.

[17] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models, 2023.

[18] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination, 2020.

[19] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models, 2022.

[20] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels, 2019.

[21] Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can wikipedia help offline reinforcement learning?, 2022.

[22] Shiro Takagi. On the effect of pre-training for transformer in different modality on offline reinforcement learning, 2022.

[23] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019.

[24] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.

[25] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.

[26] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[27] Mark Towers, Jordan K. Terry, Ariel Kwiatkowski, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G. Younis. Gymnasium, March 2023.

[28] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited, 2019.

[29] Takuma Seno and Michita Imai. d3rlpy: An offline deep reinforcement learning library. *Journal of Machine Learning Research*, 23(315):1–20, 2022.