Report - lossfunk

Questions given in PS

1. **Why did you pick this particular project?**

   I selected the third project—extending Neural Atari—because I've been following the development of world models for quite some time. This area deeply aligns with my long-term interests in AI, particularly in building systems that can understand and simulate the dynamics of their environment. I believe world models will be a foundational component in the future of AI systems, especially in domains like game engines where temporal and spatial reasoning is essential. While I considered the other two projects, this one resonated most with my vision for AI and gave me an opportunity to explore a direction I genuinely care about.

2. **If you had more compute/time, what would you have done?**

   - **Train a simplified end-to-end world model**: I'd experiment with a stripped-down version where the VQVAE takes the last frame or last *N* frames along with the action as input and directly generates the next frame. This would help assess how well VQVAE can operate as a complete world model.
   - **Use a larger decoder**: Scaling up the decoder model could improve generation quality, helping it better reconstruct complex visual details and dynamics.
   - **Train a flow-matching model**: I'd explore using a flow-matching model conditioned on context as a world model, aiming to improve frame-to-frame coherence and better capture underlying transitions.
   - **Introduce KL loss**: Incorporating KL divergence into the training process would help ensure that the model captures the distributional nature of game dynamics, especially in environments with inherent randomness.
   - **Decouple action and world dynamics**: In the current latent action model, both action and environmental dynamics are entangled in the same latent

space. I would investigate methods to separate these two components for better interpretability and flexibility.

- **Add a memory module**: I'd incorporate a memory mechanism (like a recurrent or transformer-based module) to help the model estimate and represent time more explicitly—particularly useful in games where time is an implicit variable without an external clock.

3. **What did you learn in the project?**

   One of the key things I learned during this project was the concept of behavioral learning used in LAPA, which inspired our approach for building an action-to-latent model. Since the problem statement was quite open-ended, it pushed me to think deeply about the space of possible experiments. This included not only extending and analyzing the current model, but also understanding the reasoning behind existing models in the domain, like Genie and Dreamer. The process gave me insight into how different components of world models interact and evolve, and how design choices affect their ability to model dynamic environments.

4. **What surprised you the most?**

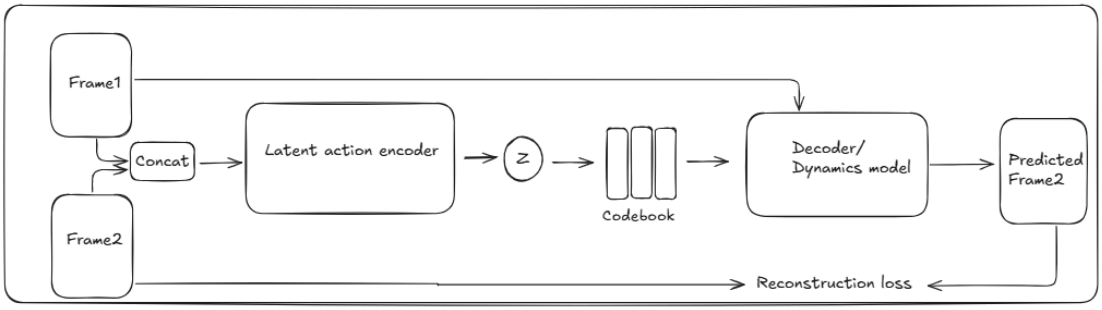
   What surprised me the most was how well the action-to-latent model performed. Given that the VQVAE encodes latent representations using both the current and next frame, it effectively captures not just the action itself but also the resulting change in the world. So, at first glance, it seemed counterintuitive that predicting this latent from only the action would work well empirically. To approximate the current and next frame, we provided the last two frames as input, but I still believe this pipeline introduces an inherent bias in the model. The fact that it still performs reasonably well despite this limitation was both unexpected and intriguing.

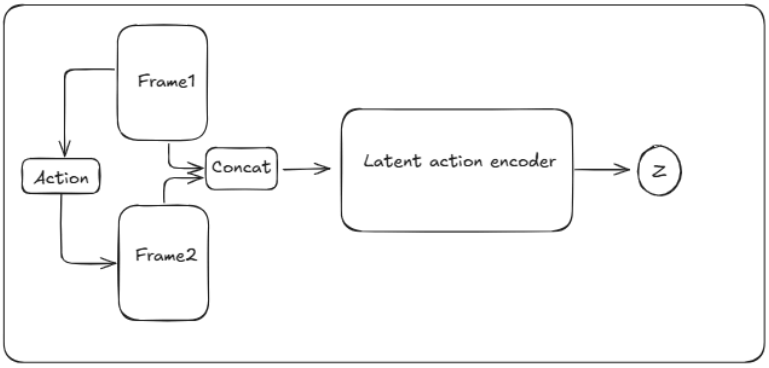
5. **If you had to write a paper on the project, what else needs to be done?**

   - **Multi-frame encoding**: Current results suggest that using only the last frame is insufficient, as a single frame can correspond to multiple plausible futures. We should experiment with encoding the last *N* frames to better capture temporal dynamics.

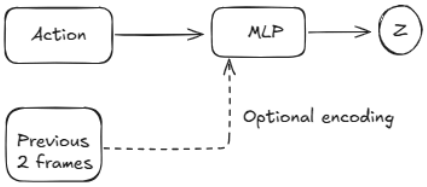
- **Establish a benchmark**: A standardized benchmark is needed to quantitatively evaluate model performance and compare different approaches fairly.
- **Modeling uncertainty in dynamics**: Many games feature stochastic elements—like enemy movement or random obstacle generation. The current VQVAE codebook is discretely one-dimensional and might be too limited. We should consider enhancing the codebook to better capture the underlying distribution of possible futures.
- **Generalizability**: It's important to test how well the model generalizes across different games. Results should not be overfitted to a single game environment.
- **Architectural improvements**: Future work should also explore more advanced or hybrid models for the action-to-latent prediction step, possibly incorporating attention mechanisms or transformers to better capture dependencies over time.

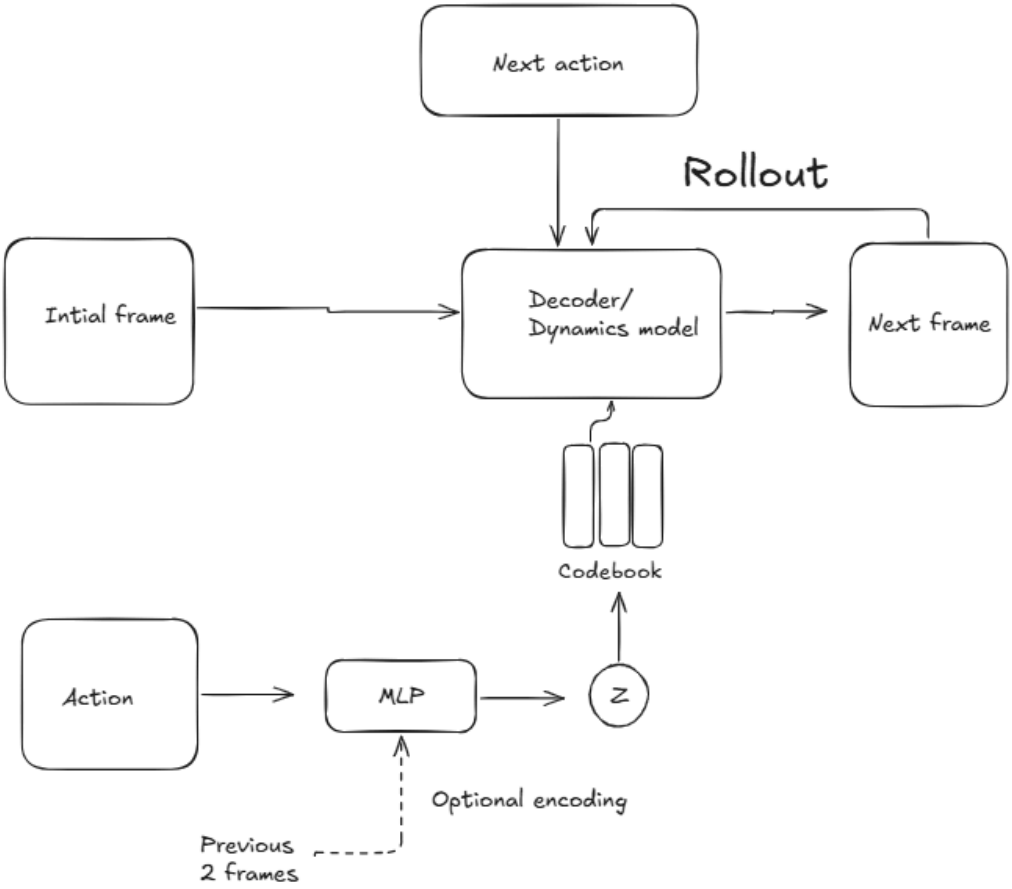
Understanding of present architecture





Collect action and latent pairs





Experiments

The following experiments were conducted from my side:

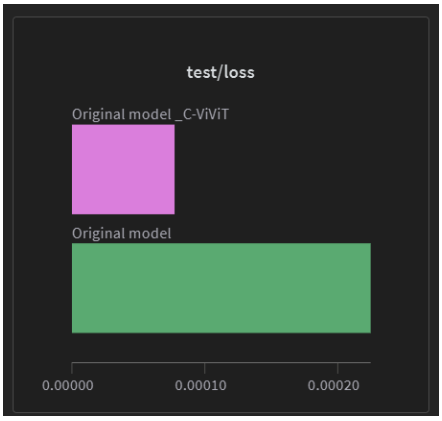
1. Convolution stems with temporal cross attention encoder inspired by C-ViViT
2. Exponential moving average for codebook updates to prevent codebook collapse
3. Decrease number of embeddings in codebook to 32
4. Try the model with different atari game with clock - Skiing-v5

Convolution stems with temporal cross attention encoder inspired by C-ViViT

Issue: As seen the video from YT, coherence between frames in missing, the score can been seen changed without an hit. The first idea came to my mind is to incorporate attention such that the model learns the relation on when to make changes.

Solution: As I was going through LAPA paper. I realized they also have used C-ViViT inspired encoder for same reason. So I have made an encoder with 2 convolution stems for both the frames and then applied cross attention to learn the relations which is then passed down to latent.

Result:

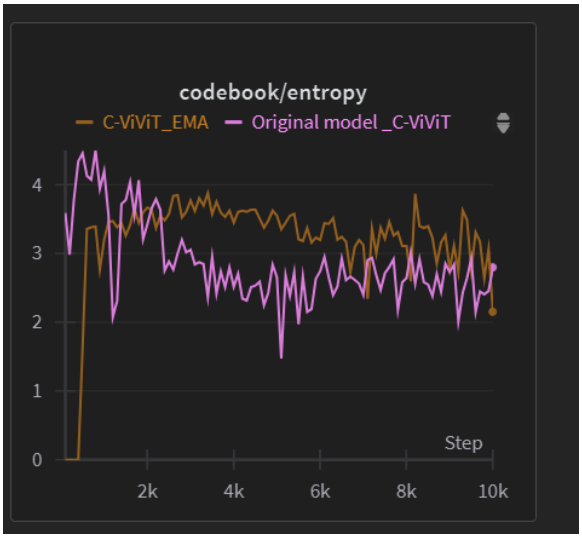


Exponential moving average for codebook updates to prevent codebook collapse

Issue: As we can see above, the codebook entropy is so chaotic, and Increasing entropy regularization is increasing, indicating codebook collapse.

Solution: To make the codebook updates more robust and prevent codebook collapse from happening, I have used Exponential moving average updates for the codebook Instead of gradient based updates.

Results :

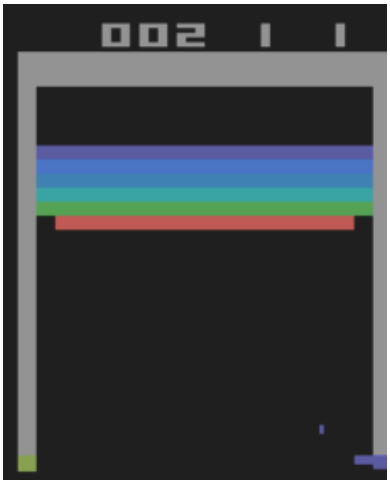


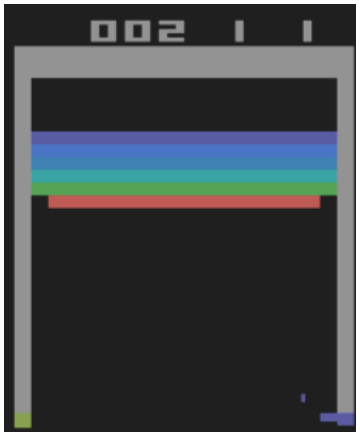
As we can see the EMA updates model is much more constant in the middle.

Decrease the number of embeddings in the codebook to 32

Idea: It is an experiment to check how much we can compress the codebook dimension. It also checks which are the features prone to forget when the codebook is compressed more than it can.

Result:





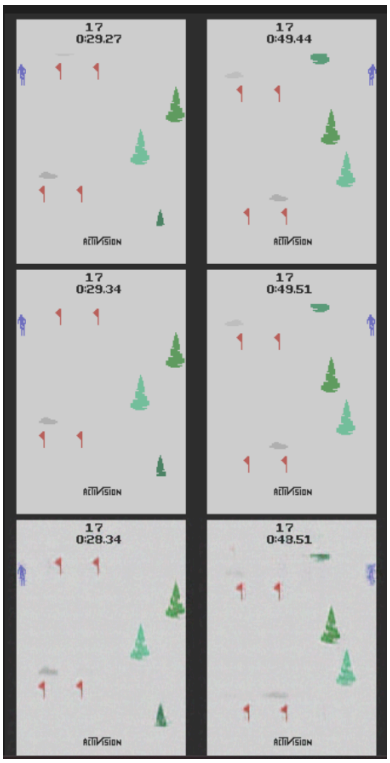
As we can see 32 is too less of a codebook size, that is starts to drop the ball from reconstruction.

Try the model with different atari game with a clock - Skiing-v5

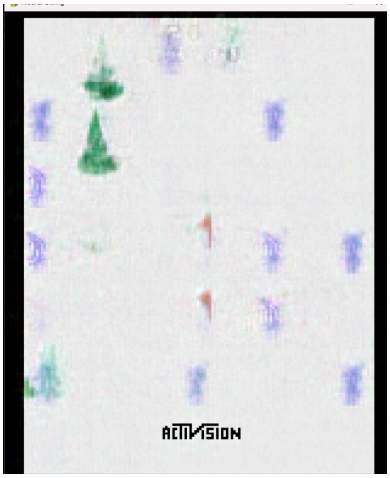
Idea: To check the generalizability of model architecture and also to answer the follow-up question.

I have chosen the Skiing game because it has less number of moving characters and also less number of actions among available games.

Result:



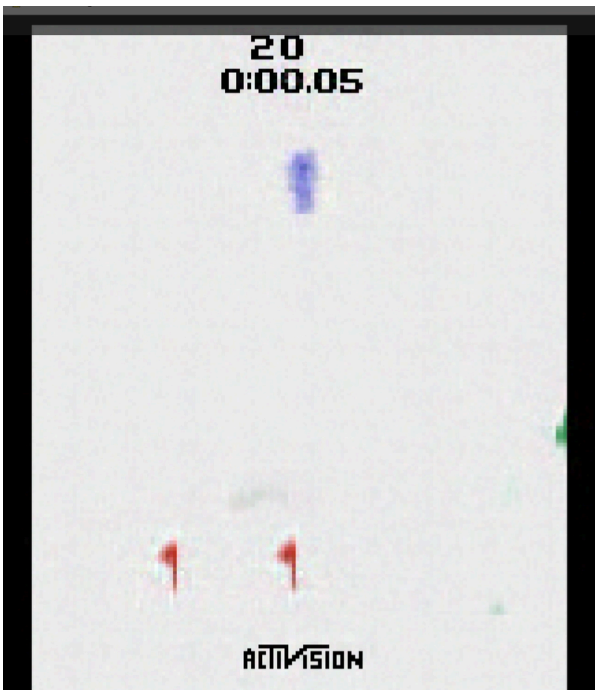
The model has learnt the reconstruction properly, although some parts are still blurry, suggesting for bigger and better generation model.



As the game continues, The model is not able to learn the distribution of the appearance of obstacles and hence move into chaos. This suggests we need to reconsider the design of codebook.

Follow up Question

one fun experiment is to train on games where time is kept (like super mario). What happens to time when you generate it via a neural network? There's no external ticking clock. How does it change?



frame1

frame2

The clock is moving forward, but with inconsistency in frame rate capture and display, we can see the delay in frames.

Theoretically, with a better and bigger model, we can get time ticking properly, as they can encode what comes after what frame. But that doesn’t mean the model has temporal understanding.

This can be justified if you take a game like Minecraft in which as time moves on, the weather changes from morning to evening, but there is no indication on display of time, and hence model never knows in this case when to change the weather unless it processes all the frames before or some kind of RNN. Or we are using a memory model to learn the time dimension.