IFT 6756 :Game Theory and ML

Karimulla Naik Saheb
Department of Computer Science
University of Montreal
karimulla.saheb.naik@umontreal.ca

Yassir Mamouni
Department of Computer Science
University of Montreal
yassir.mamouni@umonreal.ca

Abstract

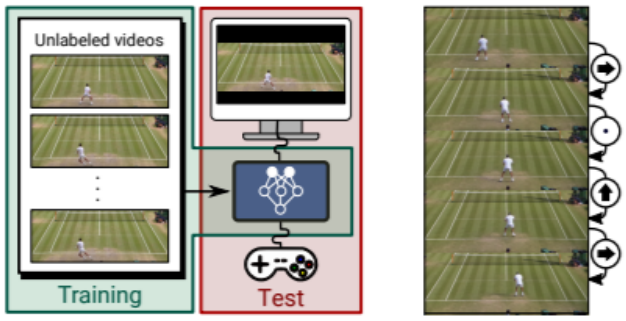
This paper is inspired by the research paper about playable video generation (PVG) [1]. In PVG, we aim at allowing a user to control the generated video by selecting a discrete action at every time step as when playing a video game. The difficulty of the task lies both in learning semantically consistent actions and in generating realistic videos conditioned on the input. What differs from this research paper is that we made hypothesis from the previous results we want to prove. Indeed, some of the metrics results are poorer than others depending on the datasets and/or the number of discrete action in the simulation. We want to make sure what parameters affect the quality of the result and how each one of them can change it.

Introduction

The concept of PVG or playable video generation, is to learn a set of distinct actions from real-world video clips in an unsupervised manner in order to interactively generate new videos. The user can provide a discrete action label at every time step and can see live its impact on the generated video, similarly to video games.

Most of the previous work before this paper on this problem have used data which frame-level action at every time step to train the network. This has limitations in terms of interactivity bound to what was trained on and also tedious process of video annotation.

The mentioned limitations are solved in an unsupervised manner. The approach discovers a set of distinct actions after watching multiple videos through a clustering procedure blended with the generation process that, at inference time, outputs playable videos. This approach is termed as Clustering for Action Decomposition and Discovery (CADDY).



(a) Playable Video Generation (b) Our results

Figure 1: Playable Video Game presentation from the original paper [1]

Approach

Objective

The objective here is to reproduce some of the training made from the inspired research paper. We noticed in the results from the paper that some of the metrics like the Δ-accuracy were pretty poor for some videos. Indeed for the *Tennis* dataset, the accuracy is under 50 comparing to 91 for the *Atari Breakout*. The inference made from these results could be that the model doesn't perform well on real image input with noisy demarcations between objects. Plus the fact that the *Tennis* training has more user agreements than the others (6 or 7 for the *Tennis* training comparing to 3 for the *Ataris Breakout*). We will explore interplay of number of user agreements vs Δ-accuracy.
Ablation studies were also made to show the importance the each key elements. The results shown in the paper were made only for the *BAIR* dataset, which we won't use for this project. We want to see how the results change for the *Ataris 2600* dataset and for the *Tennis* video. Try to see the results from another video picked and sample from *Youtube* can be a way to reinforce the conclusions we will make from the future results.

Model

CADDY consists of 4 modules. (i) an **encoder** employs a network E to extract frame representations; (ii) a **temporal model** estimates the label corresponding to the action performed in the current frame and predicts a state that describes the environment at the next time step, after performing the detected action. The action label is predicted via a network A that receives the frame representations from the current and next frames. To predict the next frame environment state , we employ a recurrent neural network R that we refer to as Dynamics network. (iii) a **decoder** module employs a network D to reconstruct each frame from the frame embedding predicted by the temporal model. (iv) the reconstructed frames are fed to the encoder to assess the quality of the estimated action labels.

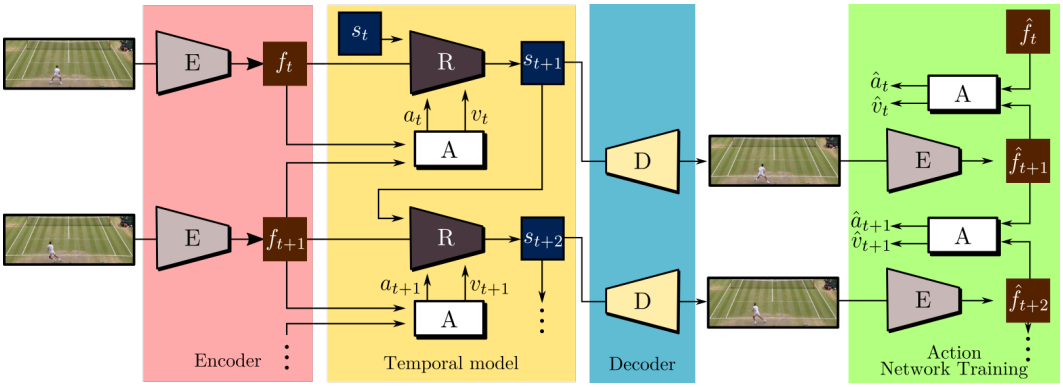


Figure 2: CADDY's training procedure for PVG

Training

Training consists of decreasing loss of objective which aims to get high quality output sequences and a discrete action space that captures the agent's high-level actions. To define such broad spectrum of objectives, losses are broken down into **Reconstruction loss** - for frame reconstruction loss based on the perceptual loss of Johnson et al. [2], **Action losses**, which allows us to measure action understanding. Total loss would be sum of these components as defined in the paper [1].

Experiments

Datasets

For the experiments, we had access to these already created datasets :

1. *Atari Breakout* dataset with 1407 sequences of about 32 frames and resolution 160x208 (358 for training, 546 sequences for validation and 503 for testing).
2. *Tennis* dataset with *YouTube* videos corresponding to two tennis matches from which 900 videos were extracted and resolution 256x96 (only the lower part of the terrain was extracted).

We also created two custom datasets :

- *Atari 2600 Pitfall* with 1712 sequence of about 40 frames and resolution 208x160 (500 for testing, 595 for validation and 617 for training).
- *Guardians* and 338 sequences (123 for testing, 135 for training and 80 for validating).

List of experiments

◇ **Experiment 1:** Reproduce the ablation studies made from the paper (to see the importance of the blocks from CADDY) and display the results not mentioned in the paper. Indeed, the *Atari Breakout* and the *Tennis* results are not shown on the paper.

◇ **Experiment 2:** Make a comparison from the previous results on the *Tennis* dataset ablation study with the original results from the paper. This aims to see and validate that our measurements from precedent experiment makes sense.

◇ **Experiment 3:** Analyze changes in Δ-accuracy by choosing Atari games with higher user agreements similar to real-time videos like Tennis. This should tell us how much the kind of content, noise, amount of objects in video affects accuracy for same count of user agreements.

Metrics to measure

The metrics we will measure for the experiments will be equivalent to the research paper :

◇ *The average Learned Perceptual Image Patch Similarity* (**LPIPS**). This metric evaluates the distance between the input frames and the generated sequence.

◇ *The average Fréchet Inception Distance* (**FID**): This metric summarizes the distance between the Inception feature vectors for input and generated frames in the same domain.

◇ *The Fréchet Video Distance* (**FVD**): This metric was introduced in 2018 from this paper [4] and works from the same principle of the FID, but consider the distribution from the entire input and generated video.

◇ *The* Δ *Mean Square Error* (ΔMSE): Δ here is represented as the center of the object of interest. This metrics evaluates the regression quality.

◇ *The* Δ *based Action Accuracy* (ΔAcc): This metric measures how the predicted action can be predicted from the displacement Δ.

Training procedure

We trained the model on an Nvidia Tesla T4 GPU with 16GB of Memory. Some of the ablation studies were also run on NVIDIA K80, 12GB GPU.

Some important hyperparameters which gave best results are: batch size of 8, applying Gumbel-Softmax, learning rate of 0.0004, learning rate scheduler for 300000 steps with gamma of 0.33. This is for all experiments.

Results

Results of Experiment 1

Ablation study reproduced on Tennis dataset

The first experiments made was to do the same ablation study made on the *BAIR* dataset but for *Tennis*. This experience aims to verify impact of three key elements of the proposed approach: Gumbel-Softmax sampling, the action loss, and action variability embeddings.

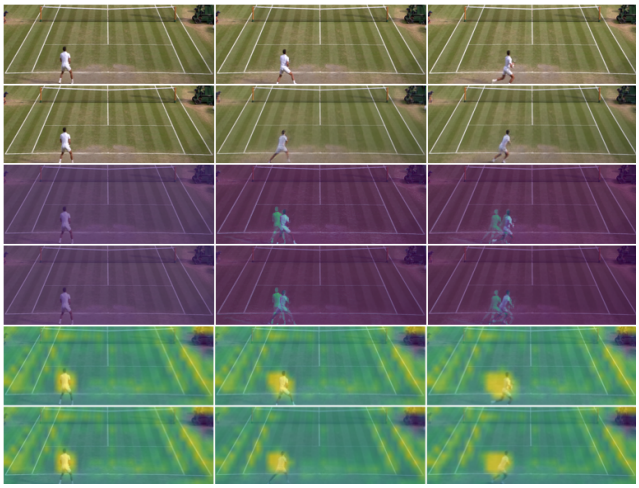


Figure 3: Model attention for training on *Tennis* dataset

Temporal model explanation

The temporal model contains probabilistic Action Network . Its goal is to estimate the action label a_t and the action variability embedding v_t .To do so, the action network uses an action state network A_s to estimate an action embedding e_t that represents the information f_t related to the action done in the frame. Following a probabilistic formulation the action state network predicts the distribution of the action embedding as a Gaussian such as :

$$e_t \sim \mathcal{N}(\mu_{e_t}, \sigma^2_{e_t})$$
$$\text{with } \mu_{e_t}, \sigma^2_{e_t} = A_s(f_t)$$

We can then estimate $e_{t+1} \sim \mathcal{N}(\mu_{e_{t+1}}, \sigma^2_{e_{t+1}})$ the same way from f_{t+1} and combined both to predict the action that is performed. And more precisely to model the distribution (a_t, v_t) we can use the difference between e_{t+1} and e_t which still is a Gaussian distribution such as:

$$d_t = e_{t+1} - e_t \sim \mathcal{N}(\mu_{d_t}, \sigma^2_{d_t})$$
$$\text{with } \mu_{d_t} = \mu_{e_{t+1}} - \mu_{e_t} \text{ and } \sigma^2_{d_t} = \sigma^2_{e_{t+1}} + \sigma^2_{e_t}$$

And finally, d_t is sampled and fed to a single layer classifier to estimate the probability of each action:

$$p_t = \{p_t^k\}_{k=1}^K \in [0, 1]^K.$$

To proceed this ablation, we specify which module to use before training the model and then we simply evaluate each ablated part the same way. Depending on the module use for ablation some parameters of the CADDY architecture are enabled or not :

- Gumbell-Softmax [5] : A Gumbell-softmax layer comes after the classifier to estimate the discrete action a_t from p_t
- Action variability v_t : To aim a meaningful learning of the action embedding a_t a set of K, *action direction centroids* $\{c_k\}_{k=1}^{K}$ are defined as the expected action direction for each action. The action variability embedding v_t is then define such as :

$$v_t = \sum_{k=1}^{K} p_t^k (d_t - c_k)$$

- The mutual information term loss $\mathcal{L}_{act}$.

Variant	G.S	v_t	$\mathcal{L}_{act}$	LPIPS↓	$FID\downarrow$	$FVD\downarrow$	$\Delta MSE\downarrow$	$\Delta Acc\uparrow$
(i)				0.128	25.97	523.29	nan	59.8
(ii)	✓			0.122	24.19	439.11	nan	49.11
(iii)	✓		✓	0.106	18.18	218.10	nan	37.6
(iv)	✓	✓		0.113	19.38	210.90	nan	36.6
CADDY(full)	✓	✓	✓	0.102	13.22	228.20	nan	42.28

Table 1: Ablation results on *Tennis* dataset

Te Δ*MSE* results were not normalized for comparison in the evaluation results so we decided to compare the other metrics we managed to access with the *BAIR* dataset results.

Variant	G.S	v_t	$\mathcal{L}_{\text{act}}$	LPIPS↓	FID↓	FVD↓	Δ-*MSE*↓	Δ-*Acc*↑
(i)				0.263	80.0	1300	69.7	51.2
(ii)	✓			0.209	42.3	571	64.8	37.9
(iii)	✓		✓	0.249	76.4	1130	92.7	24.1
(iv)	✓	✓		0.245	76.9	1130	93.7	27.6
CADDY (Full)	✓	✓	✓	**0.202**	**35.9**	**423**	**54.8**	**69.0**

Figure 4: Ablation study results on *BAIR* dataset

Here we can witness that we don't have the same high drop in the *FID* and *FVD* when adding the temporal model (ii) but it occurs when adding the the mutual information term loss of the Dynamic Network (iii). **We can deduce that adding L_{act} helps for a better generation of frames and of video in general**. The Δ-accuracy is also reduced when superposing two of these modules but adding the *action variability* and the loss in the full architecture(CADDY) helps to increase it.

One of the reasons for lower accuracy with *tennis* dataset compared to *BAIR* is that **latter has almost no background noise.**

Results of Experiment 2

Comparison with original metrics for *Tennis*

From the previous experiment results, we can compare the full CADDY metrics with the one made from the original paper.

Results	LPIPS↓	$FID\downarrow$	$FVD\downarrow$	$\Delta MSE\downarrow$	$\Delta Acc\uparrow$
CADDY (from paper)	0.102	13.7	239	72.2	45.5
CADDY(ours)	0.102	13.22	228.20	nan	42.28

Table 2: Comparison of results on *Tennis* dataset

Here we can see that the results are almost the same. We have a lower, but almost the accuracy that can be explained by the fact that we trained the model on less number of step (Around 175k compared to 300k). From the results we can also compare the qualitative evaluation of the action space learned :

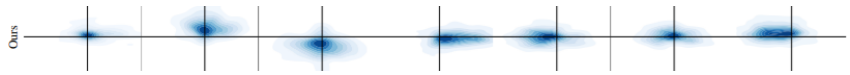


Figure 5: Original Action space

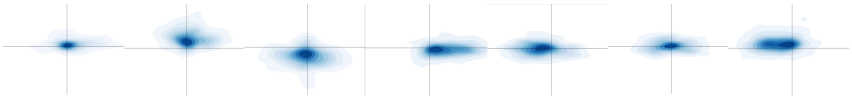


Figure 6: Action space measured during evaluation of our run

What is interesting is that for both results, each action corresponds to same movement. According to the paper the model was trained with 7 actions corresponding to *Stay, Forward, Backward, Right, Left, Hit the ball*. The seventh action corresponds to *Other* in any case the action can't be recognized. We can see then how each action corresponds to a certain direction of the tennis player for generated images:

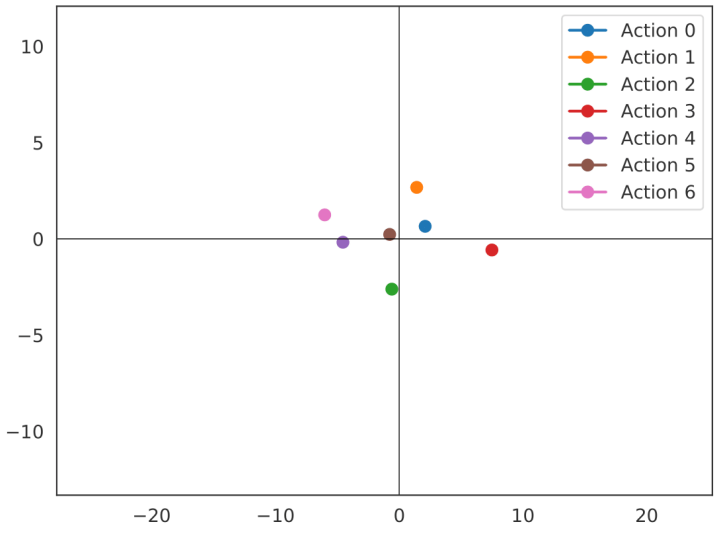


Figure 7: Visualisation of the learned space action direction on *Tennis*

Results of Experiment 3

Training on *Atari Pitfall* dataset

We train the model on 1712 sequence of about 40 frames with resolution 208x160 (500 for testing, 595 for validation and 617 for training) given in parameter the number of 4 discrete action. We choose this game because it is a platform game with simple movement and is a little more complex compared to *Breakout*.

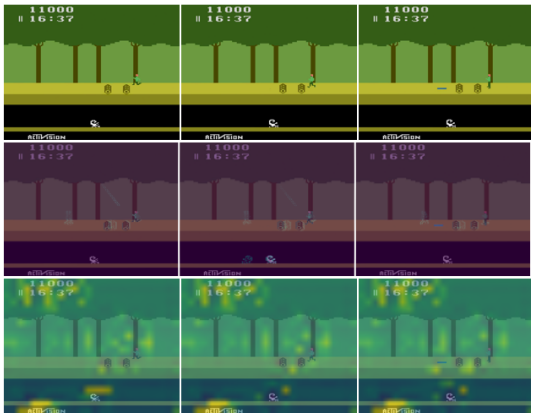
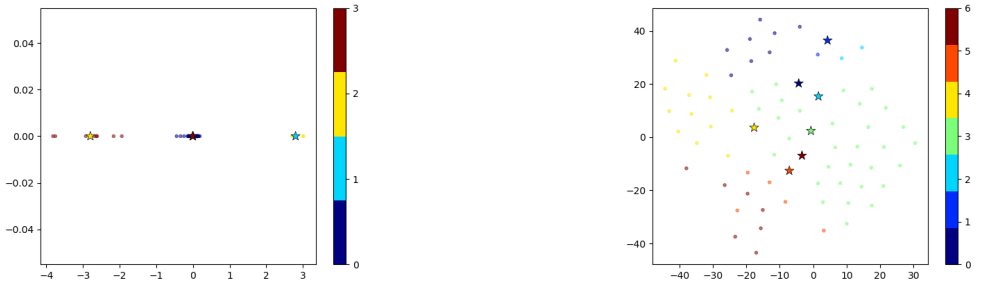


Figure 8: Model attention for training on *Pitfall* dataset

One of the interesting thing we observed by using this game is about an action we call "*Jump*". In the actual game, with combination of keys, we can jump forward or backward. Here, we see that it failed to detect combination of moves. However, it successfully recognised forward jump. Having said that, the recognized "*Jump*", isn't instantaneous. Same key needs to be pressed continuously to make the character jump. It is the same case with the detected action "*Go down the ladder*". We can say that, it has recognised the transition frame-by-frame, but it doesn't have a way to semantically consider entire sequence of frames as single action. There is scope of improvement here.

Figure 9 tells about action direction space given at each 1000 step during training for both pitfall and tennis datasets. We can witness here the direction space is flat even though we have *Jump* and *Go down the ladder* actions. We can compare with the *Tennis* action direction space where the classification of each action is more diverse.

We have also seen that action accuracy for *Pitfall* was very high, 82.88% compared to around 20% for tennis. Our best guess for this behaviour is that *Pitfall* video is not complex enough or there isn't much noise in it. The behaviour can be seen in Appendix, figure 12



(a) *Pitfall*'s Action direction space

(b) *Tennis*'s Action direction space

Figure 9: Comparison of action direction space between tennis and pitfall

Conclusion

In this work, we experimented ablation studies on a different dataset than the one made in the original Playable Video Game paper. We noticed that, with the *Tennis* dataset, metrics such as *FID* and *FVD* referring to the quality of generated images and videos and the accuracy change the same way depending on the module added or removed in the model. We also witnessed quality and accuracy can differ depending on the dataset. Comparison with previous work on the *Tennis* dataset evaluation results confirmed the relevance of our results.

One of the reasons for lower accuracy with *tennis dataset* compared to *BAIR* is that latter has almost no background noise.

Then we managed to train the model with a custom dataset and we noticed that depending on if we give the model the right minimum number of discrete action the generation can render differently. The action direction space differs from what we expected even though the game was playable.

We also observe that in the original video, if we have some action which spans large number of frames like jumping, the action detection is not smooth. Also, the generated videos of human characters are skewed.

As future work we can retrain the model with added losses of pose detection where videos have humans or animals. This should restrict distortion of human figures. There is also need for quantification of multiple frames combining into actions. This will help smooth transition of actions with just one key.

There is also scope for applying semantic segmentation methods to control characters better.

References

[1] Willi Menapace, Stephane Lathuilière, Sergey Tulyakov, Aliaksandr Siarohin, Elisa Ricci. Playable Video Generation.

[2] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In Computer Vision – European Conference of Computer Vision (ECCV), 2016.

[3] Marc G. Bellemare, Yavar Naddaf, Joel Veness, Michael Bowling. Arcade Learning Environment. 2019

[4] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, Sylvain Gelly. Towards Accurate Generative Models of Video: A New Metric & Challenges. 2018.

[5] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. 2017.

1 Appendix

1.1 Pitfall

The following plots are for pitfall dataset training:

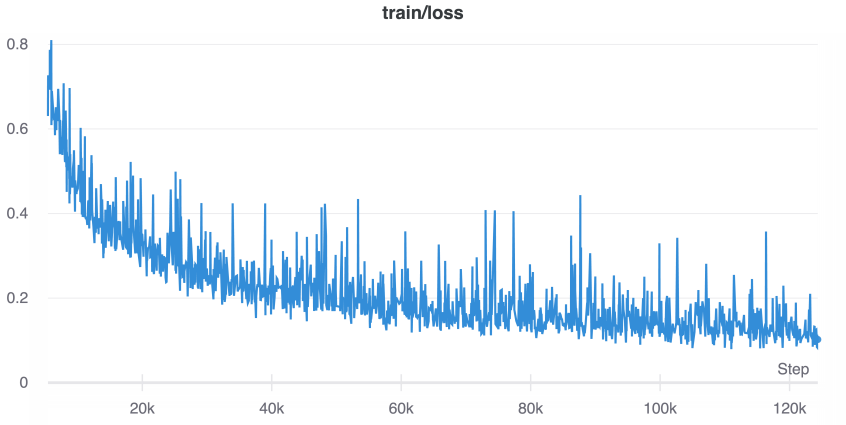


Figure 10: Training loss curve for pitfall dataset

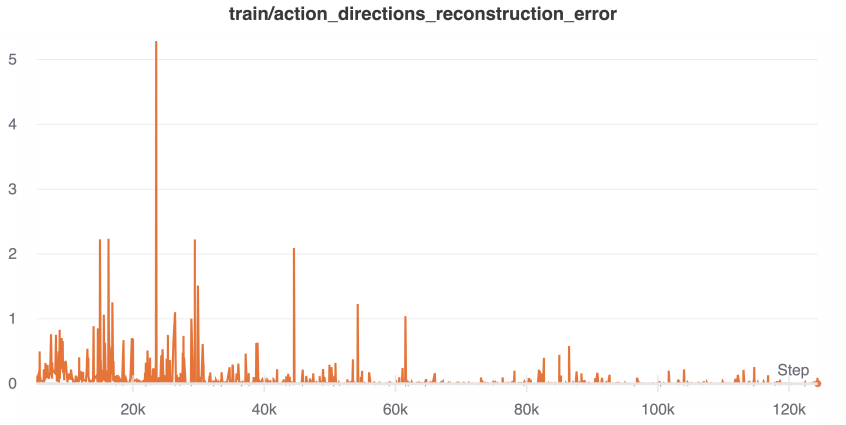


Figure 11: Reconstruction error for pitfall

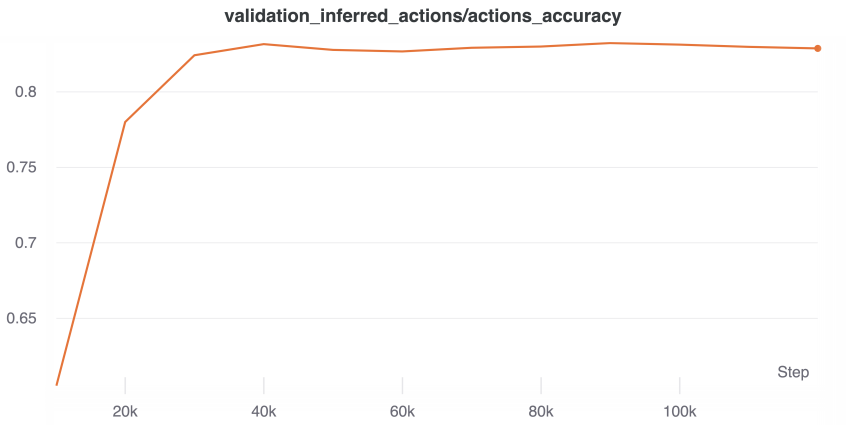


Figure 12: Action accuracy for pitfall

1.2 Tennis

This is combined loss curve for different ablation studies and full CADDY:

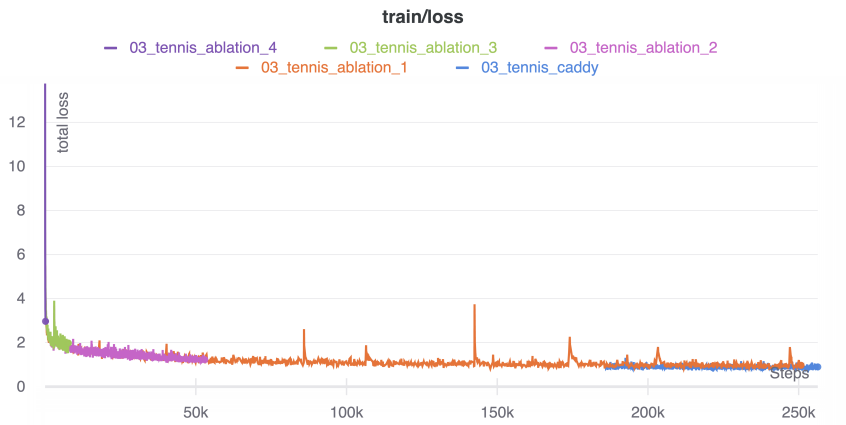


Figure 13: Training loss curve for pitfall dataset

Action variations mean for Full CADDY on Tennis data

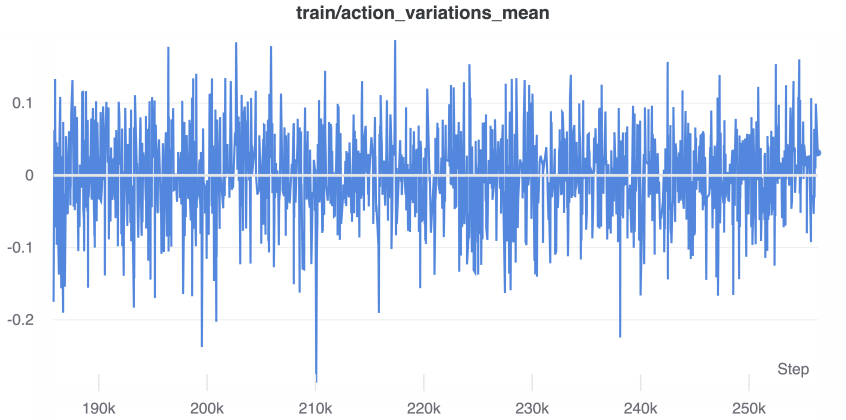


Figure 14: Action variations mean for Full CADDY on Tennis data

Reconstruction error for Full CADDY on Tennis data

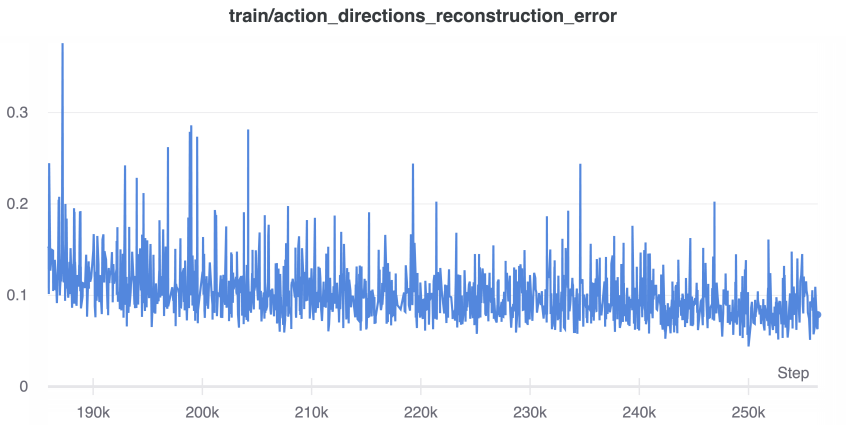


Figure 15: Reconstruction error for Full CADDY on Tennis data

Action accuracy for Full CADDY on Tennis data

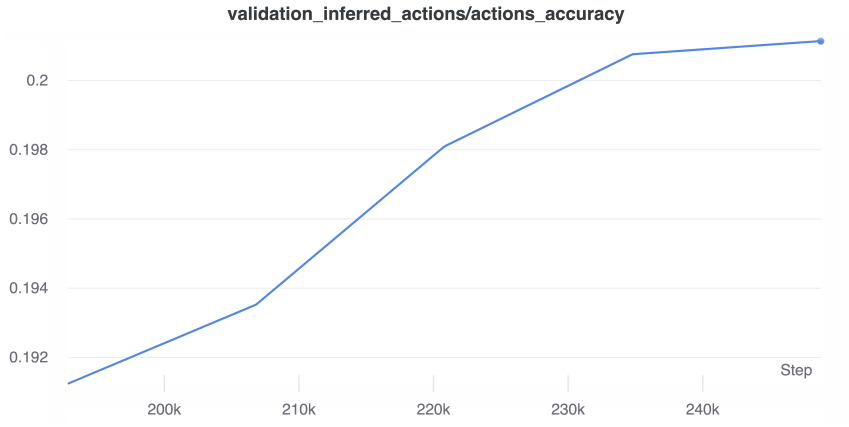


Figure 16: Reconstruction error for Full CADDY on Tennis data