Research Internship

Generative Replay for Continual Learning in Robotic

28/05/2019 to 09/08/2019

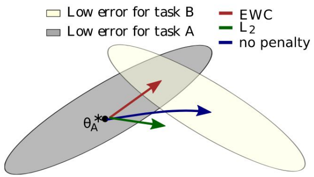
Author: Bunthet SAY
Promotion: 2020
Internship Tutor: Natalia Diaz Rodriguez
ENSTA Paris Tutor: David Filliat

Contents

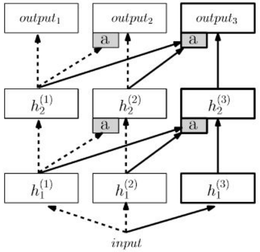
I. Introduction
II. Context
  A. Policy Distillation
  B. Generative Replay
III. Experiments and Results
  A. Training RL models
  B. Training Generative models
  C. Evaluate Generative model
  D. Policy Distillation
  E. Evaluate Distilled model
IV. Conclusion

Introduction

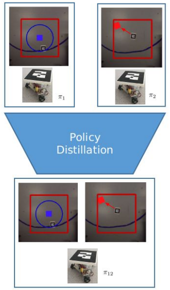
- “ Humans and animals have the ability to continually acquire, fine-tune, and transfer knowledge and skills throughout their lifespan.”
  [German Ignacio Parisi et al. “Continual Lifelong Learning with Neural Networks: A Review”]
- With the arrival of AI and Deep Learning, we have witness greate application in Robotic
- Catastrophic forgetting is a crucial challenge for Continual Reinforcement Learning.
- Three main approaches have contributed to continual learning



Elastic weights consolidation(EWC)



Progressive network



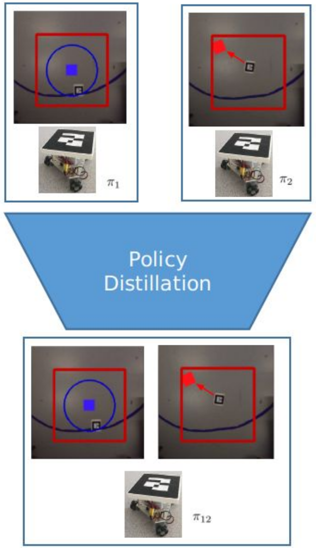
Policy distillation

Context

Policy Distillation

- **Approach**: Imitate behavior from pre-trained teacher models

$$
\begin{aligned}
1.\quad & \pi_1 \rightarrow D_{\pi_1} = \{O_t^1, P(a|O_t^1)\} \\
& \pi_2 \rightarrow D_{\pi_2} = \{O_t^2, P(a|O_t^2)\} \\
& \vdots \\
& \pi_k \rightarrow D_{\pi_k} = \{O_t^k, P(a|O_t^k)\} \\
2.\quad & D_{\pi_1} \cup D_{\pi_2} \cup \ldots \cup D_{\pi_k} = D_{\pi_{1,2,\ldots,k}} \xrightarrow{PolicyDistillation} \pi_{1,2,..,k}
\end{aligned}
$$



- **Challenge** :
  - Access to the environment is need for a basic policy distillation
  - Dataset stockage

Context

Generative Replay

1. Variational AutoEncoder (VAE)
2. Conditional Variational AutoEncoder (CVAE)
3. Generative Adversarial Network(GAN)
4. Conditional Generative Adversarial Network(CGAN)

Context

Generative Replay

1. Variational AutoEncoder (VAE)

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|X_i)}[\log p_\phi(X_i \mid z)] + \mathbb{KL}(q_\theta(z \mid X_i) \mid\mid p(z))$$

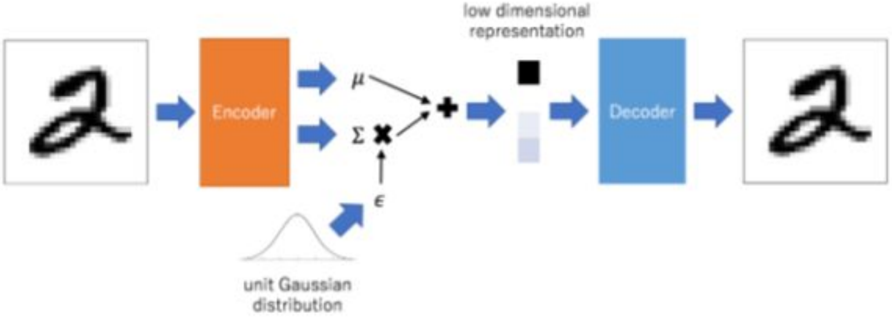


Figure 2.2: Variational Autoencoder schema

Context

Generative Replay

1. Conditional Variational AutoEncoder (CVAE)

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|X_i,c_i)}[\log p_\phi(X_i \mid z, c_i)] + \mathbb{KL}(q_\theta(z \mid X_i, c_i) \mid\mid p(z|c_i))$$

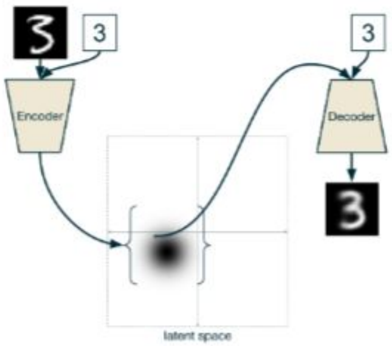


Figure 2.3: Conditional Variational Autoencoder schema

Context

Generative Replay

1. Generative Adversarial Network(GAN)

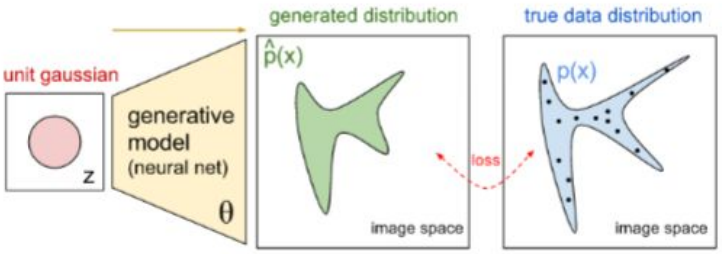


Figure 2.4: Generative Adversarial Network schema

The loss function of GAN is the Mini-Max loss between D and G:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

Context

Generative Replay

1. Conditional Generative Adversarial Network(CGAN)

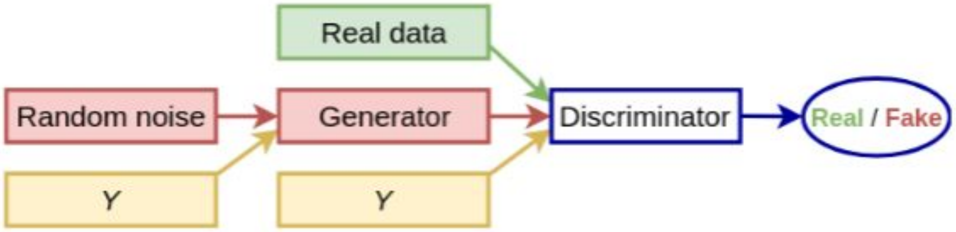
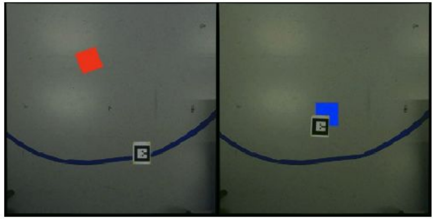


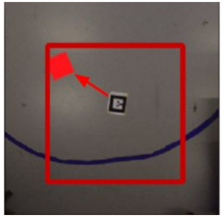
Figure 2.5: Conditional Generative Adversarial Network training schema

Experiments and Results

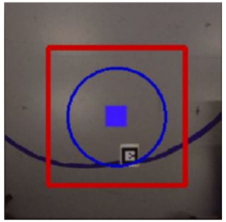
- Objective: Generate data set from 2 tasks to serve policy distillation
- Method: Generative replay
- 2 main experiments
- Experiments setup: **Experiment 1 and 2**
  - Environment: Omnirobot_env



Omnirobot_env



Task 1: Target Reaching (TR)



Task 2: Target Circling (TC)

Experiments and Results

- Experiments setup: **Experiment 1**

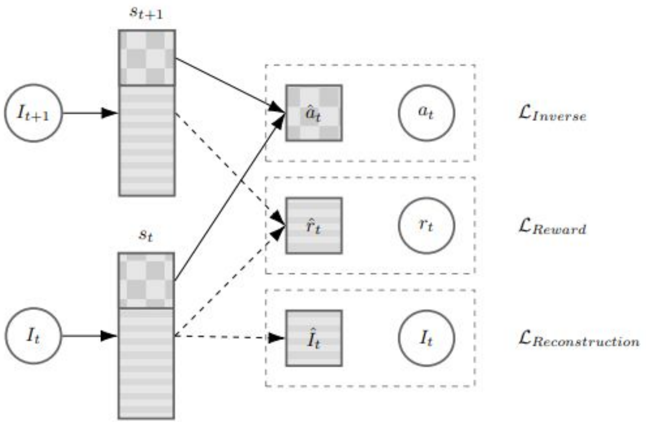
1. $\{O_t^{Task_1}\} \xrightarrow{Train\ SRL} \{s_t^{Task_1}\} \xrightarrow{Train\ RL} \pi_{Task_1}$
2. $\{O_t^{Task_1}\} \xrightarrow{Generation\ on\ policy\ \pi_{Task_1}} D_{\pi_{Task_1}} = \{O_t^{Task_1}, P_{\pi_{Task_1}}(a|O_t^{Task_1})\}$
3. $D_{\pi_{Task_1}} \xrightarrow{Training} GM_{\pi_{Task_1}}(z, a, t_{pos})$
4. $GM_{\pi_{Task_1}}(z, a, t_{pos}) \xrightarrow{Sampling} \{O^{G_{Task_1}}\} \xrightarrow{SRL} \{s^{G_{Task_1}}\} \xrightarrow{\pi_{Task_1}} D_{GM_{\pi_{Task_1}}}$
   where $D_{GM_{\pi_{Task_1}}} = \{O_t^{G_{Task_1}}, P_{\pi_{Task_1}}(a|O_t^{G_{Task_1}})\}$
5. Similarly for $Task_2$, we get : $D_{GM_{\pi_{Task_2}}}$
6. $D_{GM_{\pi_{Task_1}}} \cup D_{GM_{\pi_{Task_2}}} \xrightarrow{Train\ Policy\ Distillation} \pi_{Task_{12}}$

Experiments and Results

- Experiments setup: **Experiment 1 and 2**
  - Train SRL model

	$Task_1$	$Task_2$
name	SRL_Task_1	SRL_Task_2
model	*srl_plits: autoencoder/inverse/reward*	*srl_plit: autoencoder/inverse*
dimensionality	*autoencoder*: 198 *inverse*: 2 *reward*: 198	*autoencoder*: 198 *autoencoder*: 2
Losses' weight	*autoencoder*:1 *inverse*:1 *reward*:1	*autoencoder*1 *inverse*:1
Batch size	32	32
State's dimensions	200	200
Training epoch	20	20
optimizer	Adam	Adam
Adam parameters	*beta_1*=0.5,*beta_2*=0.9	*beta_1*=0.5,*beta_2*=0.9
Learning rate	0.005	0.005

Hyperparameters for SRL model training.



SRL splits model.

Experiments and Results

- Experiments setup: **Experiment 1**
    - Generative models architecture:
        - CVAE : Resnet
        - CGAN:
            - Deep Convolutional (DC)
            - Resnet

Experiments and Results

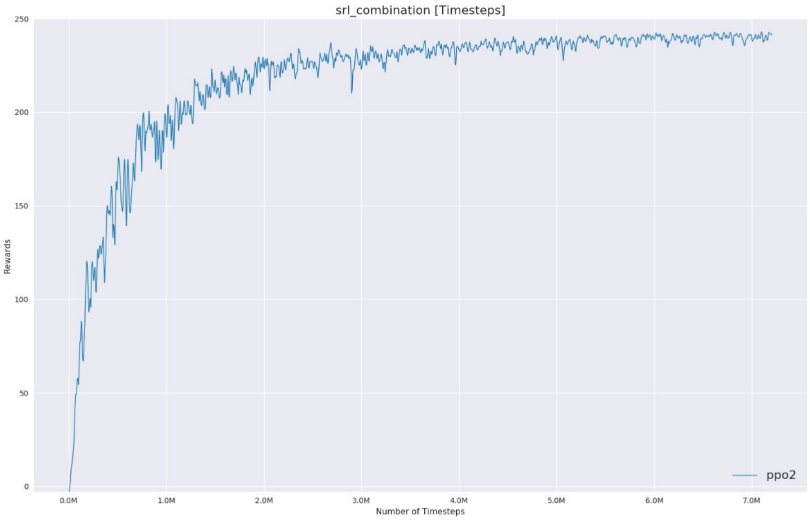
- Experiments setup: **Experiment 1 and 2**
  - Train RL model

	$Task_1$	$Task_2$
Algorithm	PPO2	PPO2
Srl_model	SRL_Task_1	SRL_Task_2
Number of timesteps	5 millions	5 millions
Image's size	(3,64,64)	(3,64,64)
Seed	0	0

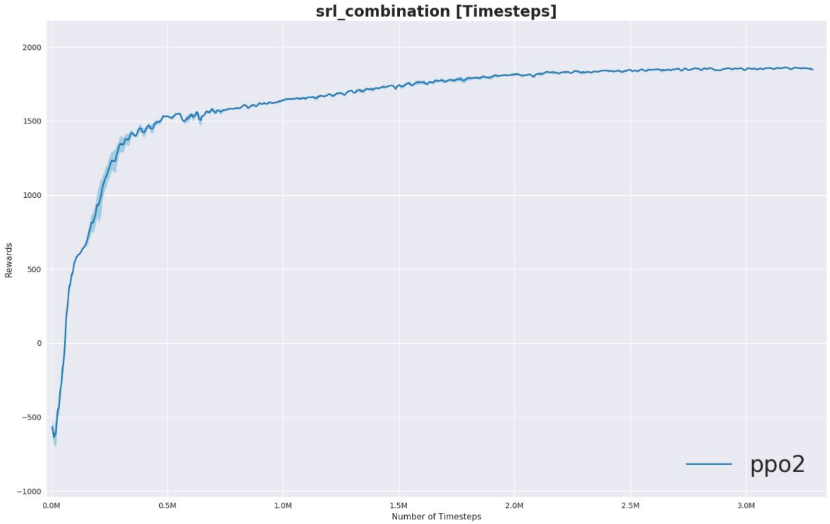
Setting for training RL.

Experiments and Results

- Experiments setup:**Experiment 1 and 2**
  - Train RL model



Task 1: TR



Task 1: TC

Experiments and Results

- Experiments setup: **Experiment 1**
  - Train Generative model

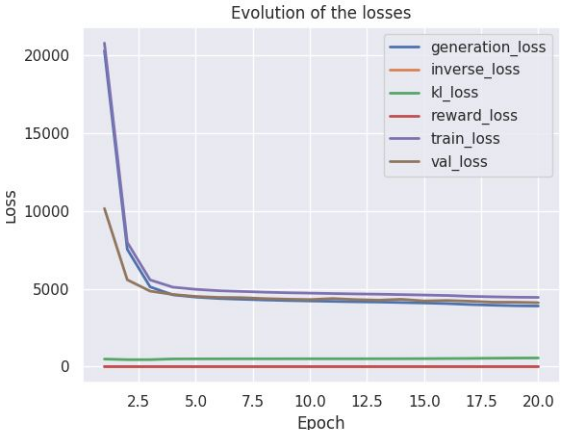
Hyperparameters	CVAE	CGAN
Batch size	128	128
Training epoch	20	100
optimizer	Adam	Adam (Both G and D)
Adam parameters	$beta\_1$=0.5,$beta\_2$=0.9	$beta\_1$=0.5,$beta\_2$=0.9

Soumit, in his repository, propose several tip and trick to train GAN to stabilize the training and prevent from mode collapse:

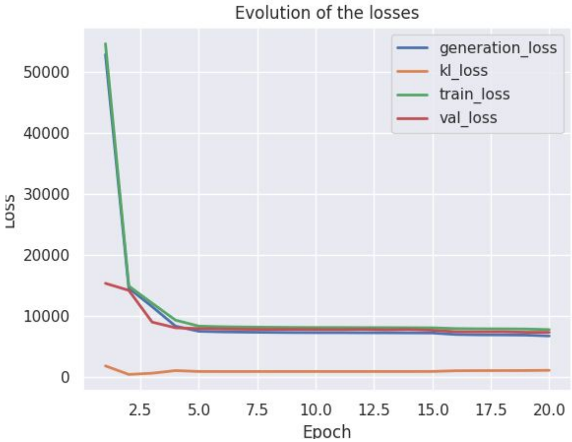
- Normalize input
- Use spherical Z
- Batch Normalization
- Avoid sparse gradients: Relu and Maxpool
- Use label smoothing
- Use Adam optimizer
- Use label for training (action and target position in our case)
- Spectral Normalization for both generator and discriminator (on top of Batch Normalization)

Experiments and Results

- Experiments setup: **Experiment 1**
    - Train Generative model



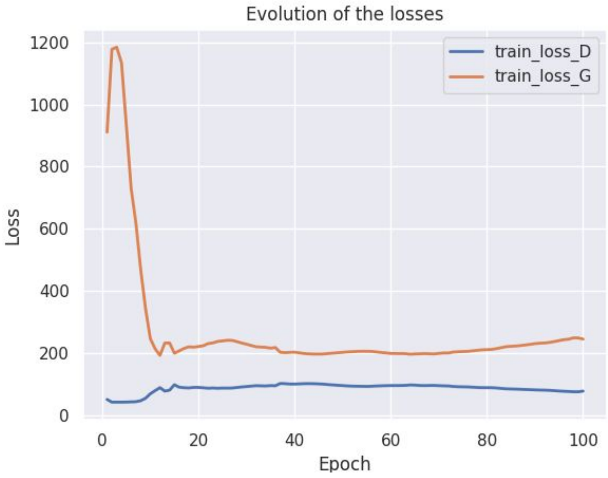
CVAE splits losses evolution for Task_1.
lr=0.0002



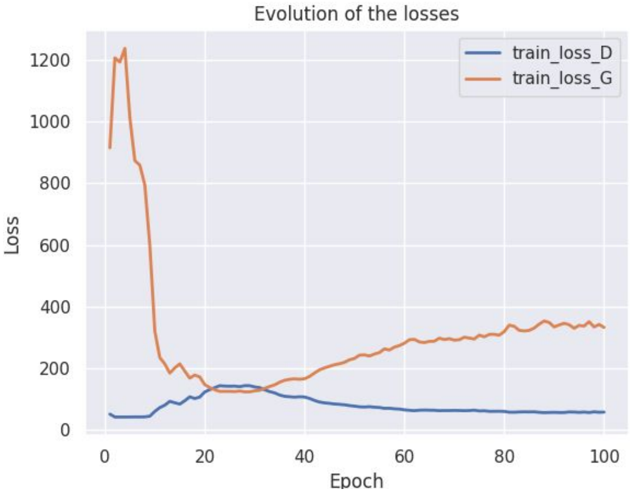
CVAE splits losses evolution for Task_2.
lr=0.0002

Experiments and Results

- Experiments setup: **Experiment 1**
  - Train Generative model



CGAN (Resnet) losses evolution for Task_1.
lr_G=5e-5
lr_D=e-5



CGAN (Resnet) losses evolution for Task_2.
lr_G=5e-5
lr_D=e-5

Experiments and Results

- Experiments setup:
  - Evaluate generative model

1. $GM_{\pi_{Task_1}}(z, a, t_{pos}) \xrightarrow{Sampling} O^{G_{Task_1}}_{z,a,t_{pos}}$
2. $O^{G_{Task_1}}_{z,a,t_{pos}} \xrightarrow{Forward\ to\ SRL\ model} s^{G_{Task_1}}_{z,a,t_{pos}} \xrightarrow{Forward\ to\ \pi_{Task_1}} P_{\pi_{Task_1}}(A_i | O^{G_{Task_1}}_{z,a,t_{pos}})$ where $A_i \in \{0, 1, 2, 3\}$
3. We check if $A_i = a$

Experiments and Results

- Results:
    - Task 1: TR

Model	LR	Overall Accuracy	Accuracy per Action
CVAE	10^{-4}	26.15%	[11.04%, 36.01%, 12.09%, 44.20%]
CVAE split	10^{-4}	25.10%	[07.20%, 39.20%, 14.00%, 40.00%]

Table 3.6: Benchmark of CVAE for $Task_1$ with $Random\_seed = 0$.

Model	LR_G	LR_D	Overall Accuracy	Accuracy per Action
DC	2.10^{-4}	2.10^{-4}	26.15%	[19.03%, 33.90%, 38.20%, 13.20%]
DC	10^{-5}	5.10^{-5}	24.90%	[04.10%, 50.60%, 01.40%, 43.50%]
Resnet	2.10^{-4}	2.10^{-4}	25.12%	[98.60%, 00.00%, 00.40%, 01.50%]
Resnet	10^{-5}	5.10^{-5}	24.60%	[03.40%, 58.90%, 25.30%, 10.80%]

Table 3.7: Benchmark of CGAN for $Task_1$ with $Random\_seed = 0$.

Experiments and Results

- Results:
  - Task 1: TC

Model	LR	Overall Accuracy	Accuracy per Action
CVAE	10^{-4}	21.95%	[77.00%, 00.10%, 10.30%, 00.40%]
CVAE split	10^{-4}	21.60%	[73.80%, 00.00%, 12.00%, 00.60%]

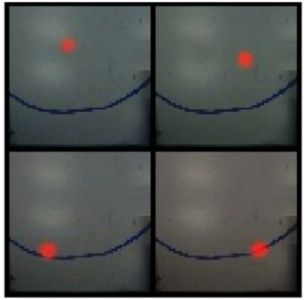
Table 3.8: Benchmark of CVAE for $Task_2$ with $Random\_seed = 0$.

Model	LR_G	LR_D	Overall Accuracy	Accuracy per Action
DC	2.10^{-4}	2.10^{-4}	25.78%	[05.90%, 02.70%, 00.75%, 93.80%]
DC	10^{-5}	5.10^{-5}	25.57%	[05.30%, 02.70%, 00.55%, 93.75%]
Resnet	2.10^{-4}	2.10^{-4}	24.75%	[00.00%, 34.60%, 62.00%, 02.40%]
Resnet	10^{-5}	5.10^{-5}	24.92%	[00.20%, 20.20%, 78.90%, 00.40%]

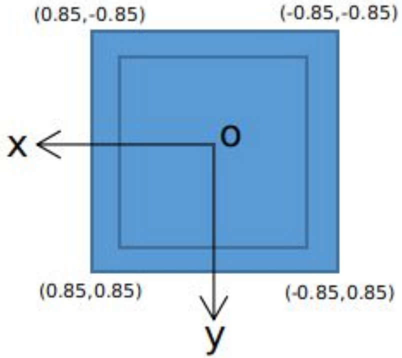
Table 3.9: Benchmark of CGAN for $Task_2$ with $Random\_seed = 0$.

Experiments and Results

- Results
    - **Some generated observations**



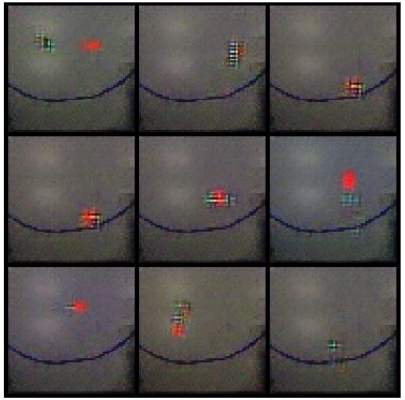
Four generated observations of Task_1 by CVAE split model corre-
spond to four target positions: (0.27,-0.63), (0.63,0.48),(-0.34,-0.39),(-0.59,0.46)
respectively form left to right and from up to down



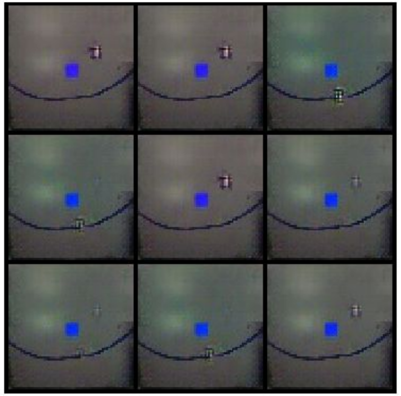
Coordinate of the arena

Experiments and Results

- Results
  - **Some generated observations**



Some generated observation by CGAN (Resnet) on Task_1 with
LR_D =5e-5, LR_D=e-5



Some generated observation by CGAN (Resnet) on Task_2 with
LR_D =5e-5, LR_D=e-5

Experiments and Results

- Experiments setup: **Experiment 2 (inspired by result of CVAE in Experiment 1)**

1. $\{O_t^{Task_1}\} \xrightarrow{Train\ SRL} \{s_t^{Task_1}\} \xrightarrow{Train\ RL} \pi_{Task_1}$
2. $\{O_t^{Task_1}, r_{pos}, t_{pos}\} \xrightarrow{Train\ generative\ model} GM_{Task_1}(z, r_{pos}, t_{pos})$
3. $GM_{Task_1}(z, r_{pos}, t_{pos}) \xrightarrow{Sampling} O^{G_{Task_1}}$
4. $\{O^{G_{Task_1}}\} \xrightarrow{SRL} \{s^{G_{task_1}}\} \xrightarrow{\pi_{Task_1}} D_{GM_{\pi_{Task_1}}}$

   where $D_{GM_{\pi_{Task_1}}} = \{O_t^{G_{Task_1}}, P_{\pi_{Task_1}}(a|O_t^{G_{Task_1}})\}$
5. Similarly for $Task_2$, we get : $D_{GM_{\pi_{Task_2}}}$
6. $D_{GM_{\pi_{Task_1}}} \cup D_{GM_{\pi_{Task_2}}} \xrightarrow{Train\ Policy\ Distillation} \pi_{Task_{12}}$

Experiments and Results

- Experiments setup: **Experiment 2**
  - Train SRL model (the same as in experiment 1)
  - Train RL model (the same as in experiment 1)
  - Train generative model:
    - Model Architecture: CVAE Resnet
    - Training data set :

	$D_{R_{Task_1}}$	$D_{R_{Task_2}}$
Policy	Random	Random
Number of episode	250	250
Image's size	(3,64,64)	(3,64,64)

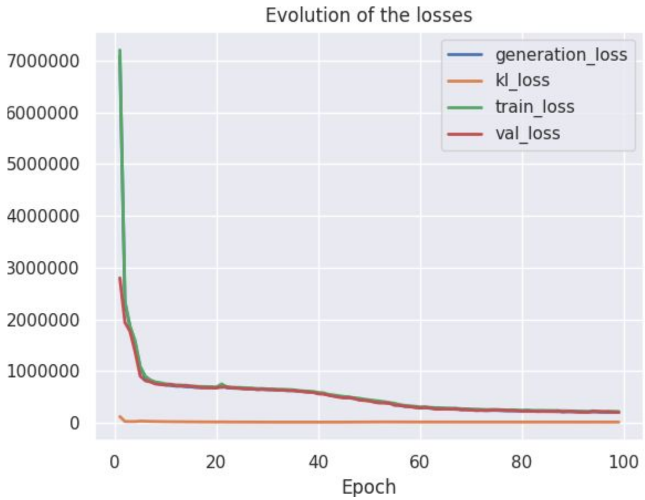
Experiments and Results

- Experiments setup: **Experiment 2**
  - Train generative model:

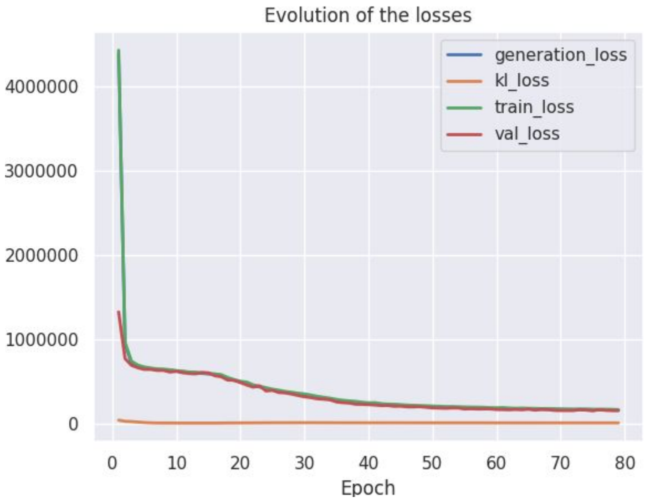
Hyperparameters	$CVAE_{Task_1}$	$CVAE_{Task_2}$
Training data set	$D_{R_{Task_1}}$	$D_{R_{Task_2}}$
Batch size	128	128
Training epoch	100	80
optimizer	Adam	Adam
Learning Rate	0.005	0.005
State dimension	200	200
Adam parameters	*beta_1*=0.5,*beta_2*=0.9	*beta_1*=0.5,*beta_2*=0.9
Losses weight	KL_loss: 100 Reconstruction_loss: 1	KL_loss: 100 Reconstruction_loss: 1

Experiments and Results

- Experiments setup: **Experiment 2**
    - Train generative model:



Losses evolution in CVAE_Task1 .

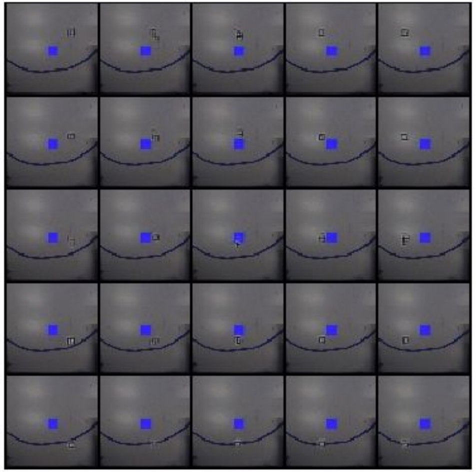


Losses evolution in CVAE_Task2 .

Experiments and Results

- Results: **Experiment 2**
  - Some generated observations:

Observations in an episode generated by CVAE_Task1 with a fixed value of latent variable z.The sampled target is at position (0.50,0.49) and the robot's position is sampled with grid walker of step 0.28. With this step there are 25 observations in each episode.



Observations in an episode generated by CVAE_Task2 with a fixed value of latent variable z.The sampled target is at the centre of the arena and the robot's position is sampled with grid walker of step 0.28. With this step there are 25 observations in each episode.

Experiments and Results

- Train policy distillation
  - Three step :
    1. Generate data set from replay of both tasks
    2. Merge data set
    3. Train policy distillation

Experiments and Results

- Train policy distillation : **experiment 1**
  1. Generate data set from replay of both tasks

	Constants of *Omnirobot_env*
Target's boundary	Target_MIN_X = -0.7 Target_MAX_X = 0.7 Target_MIN_Y = -0.7 Target_MAX_Y = 0.7
Robot's boundary	MIN_X = -0.85 MAX_X = 0.85 MIN_Y = -0.85 MAX_Y = 0.85

Table 3.10: Constants of *Omnirobot_env*.

	$D_{\pi_{Task_1}}$	$D_{\pi_{Task_2}}$
Number of actions	action "0": 2000 action "1": 2000 action "2": 2000 action "3": 2000	
State sampling	$z_i \sim Normal(0,1)$	
Target's position sampling	$x \sim Uniform[Target\_MIN\_X, Target\_MAX\_X)$ $y \sim Uniform[Target\_MIN\_Y, Target\_MAX\_Y)$	$x = 0$ $y = 0$
Robot's position sampling	$x \sim Uniform[MIN\_X, MAX\_X)$ $y \sim Uniform[MIN\_Y, MAX\_Y)$	
Image's size	(3,64,64)	

Table 3.11: Actions sampling and image's size of each generated data set.

Experiments and Results

- Train policy distillation : **experiment 1**
  1. Merge data set :

     We merge $D_{CVAE_{\pi_{Task_1}}}$ and $D_{CVAE_{\pi_{Task_2}}}$ as $D_{CVAE_{\pi_{Task_{12}}}}$.

  2. Train policy distillation:

     Setting:
     - Batch size: 8
     - Adaptive temperature: False
     - Learning rate: e-3
     - Training epoch: 20
     - Raw pixl
     - Policy model: Customs CNN
     - Fine Tuning : False

Experiments and Results

- Train policy distillation : **experiment 2**
    1. Generate data set from replay of both tasks

	Constants of *Omnirobot_env*
Target's boundary	Target_MIN_X = -0.7 Target_MAX_X = 0.7 Target_MIN_Y = -0.7 Target_MAX_Y = 0.7
Robot's boundary	MIN_X = -0.85 MAX_X = 0.85 MIN_Y = -0.85 MAX_Y = 0.85

Table 3.10: Constants of *Omnirobot_env*.

	$D_{\pi_{Task_1}}$	$D_{\pi_{Task_2}}$
Random seed	0	0
Number of episode	100	100
State sampling	$z_i \sim Normal(0,1)$	
Target's position sampling in each episode	$x \sim Uniform[Target\_MIN\_X, Target\_MAX\_X)$ $y \sim Uniform[Target\_MIN\_Y, Target\_MAX\_Y)$	$x = 0$ $y = 0$
Robot's position sampling in each episode	grid walker	grid walker
Grid walker's step	0.1	0.1
Image's size	(3,64,64)	

Table 3.12: Sampling method of each data set in experiment 2.

Experiments and Results

- Train policy distillation : **experiment 2**
  1. Merge data set :

     We merge $D_{CVAE_{\pi_{Task_1}}}$ and $D_{CVAE_{\pi_{Task_2}}}$ as $D_{CVAE_{\pi_{Task_{12}}}}$.

  2. Train policy distillation:

     Setting:
     - Batch size: 8
     - Adaptive temperature: False
     - Learning rate: e-3
     - Training epoch: 20
     - Raw pixl
     - Policy model: Customs CNN
     - Fine Tuning : False

Experiments and Results

- Evaluate distilled model
  - Task 1: TR

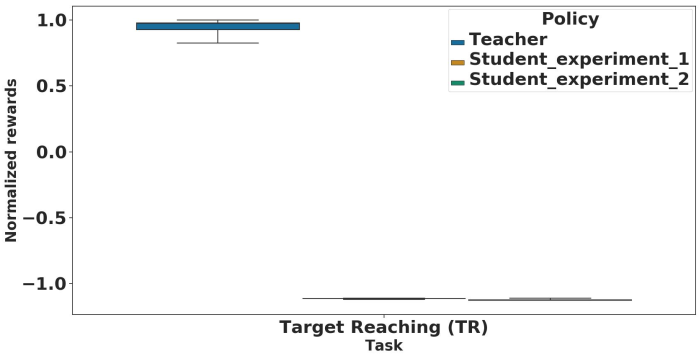
	2 episodes	4 episodes	6 episodes	8 episodes	9 episodes
$Teacher\ \pi_{Task1}$	211.5	201	211.83	216.75	179.22
$Student\_experiment\_1$	-242.5	-240.75	-241	-241.5	-241
$Student\_experiment\_2$	-240.5	-243.75	-244.33	-243.75	-244.35

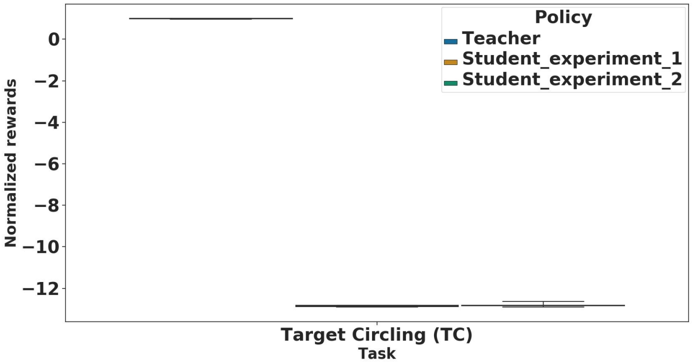
  - Task 2: TC

	2 episodes	4 episodes	6 episodes	8 episodes	9 episodes
$Teacher\ \pi_{Task2}$	1824.92	1868.9	1874.85	1878	1881.69
$Student\_experiment\_1$	-24134.34	-24290.95	-24130.59	-24145.42	-24219.86
$Student\_experiment\_2$	-23758.02	-24261.76	-24111.13	-24129.34	-24150.26

Experiments and Results

- Evaluate distilled model





Conclusion

- Experiment 1:
    - Non satisfied for policy distillation
    - Generate observation with target at any position we want which is indispensable for policy distillation (and it influence experiment 2)
    - Train generative model:
        - Without condition
        - With one condition: action
        - With two condition: action and target position
- Experiment 2:
    - Generate observation with target and observation at any position we want in the arena
- General
    - Scalable model (SRL, RL, Policy Distillation, Generative Replay)
    - Image size (3,64,64) instead of (3,224,224) => speed up the training from to 4 to 6 time (depend on the architecture)

Conclusion

- Challenges
    - Experiment 1
        - Training GAN
        - Complicate data set distribution
- Improvement and future work
    - Experiment 1:
        - Try with deterministic policy
        - Implement FID and Inception score for GAN
        - Try with other less realistic environments (ex: mobile_robot_extreme)
        - Evaluate distilled multiple time with different random seed (at least 5) and average them

Question?