澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

Reinforcement Learning for Generative AI: A Survey

GongYing
gong.ying@connect.umac.mo
2023.12.30

Contents

1. Introduction
2. Preliminary and background
3. Benefits
4. Challenges
5. Application
6. Future directions

1. Introduction

I. Great progress in generative AI recently:
   - Variational autoencoders
   - Autoregressive models
   - Adversarial generative nets
   - Energy-based models
   - Normalizing flows

II. Advancement above brought development of a broad range of applications from NLP to image generation and scientific research.
   - Large Language Models like ChatGPT changed the paradigm of developing next gen of machine learning systems.
   - Diffussion models.
   - Scientific research in molecular design and optimization.

1. Introduction

III. Training generative model is a cornerstone of generative AI, which studies to design object functions. The major objective is Maximum Likelihood Estimation (decreasing KL divergence between generated distribution and target distribution).

IV. Reinforcement learning is proposed as a useful optional training paradigm to improve the performance of generation models.
- Learns from interaction.
- Has flexible objectives in terms of reward functions.
- Can be utilized on generation problems, many of which can be redefined as decision-making problems.

2. Preliminary and background

I. Generative models

Generative models – how to represent a distribution for x or $\boldsymbol{x}$, the space of which is denoted as $\boldsymbol{X}$.

- Variational autoencoder (VAE)
- Generative adversarial networks (GANs)
- Energy-based models (EBM)
- Autoregressive models
- Normalizing flows

2. Preliminary and background

I. Generative models

- **Variational autoencoder (VAE)**

  VAE learns useful representations by reconstructing the input x with a latent variable z in consideration:

$$p(x) = \int p(x|z)p(z)dz.$$

  Latent variable refers to the variable in the model that cannot be observed, but is assumed to somehow impact the data that can be observed. For example, we are interested in students' academic performances which can be observed. But there can be many factors that cannot be observed impacting their academic performances (like motivation or family background) which are latent variables.

2. Preliminary and background

I. **Generative models**

- **Variational autoencoder (VAE)**

  VAE learns useful representations by reconstructing the input x with a latent variable z in consideration:

$$p(x) = \int p(x|z)p(z)dz.$$

  a) **Encoder**: encode the input (like image), and then get a set of params (usually mean or variance) describing the distribution in latent space, each dimension of which represents a characteristic of input.

  b) **Reparameterization**: VAE gets latent variable from the above distribution via reparameterization.

  c) **Decoder**: VAE decodes the latent variables via decoder, thus getting the reconstructed input. Decoder aims to make the constructed input data as close as possible to the raw input data.

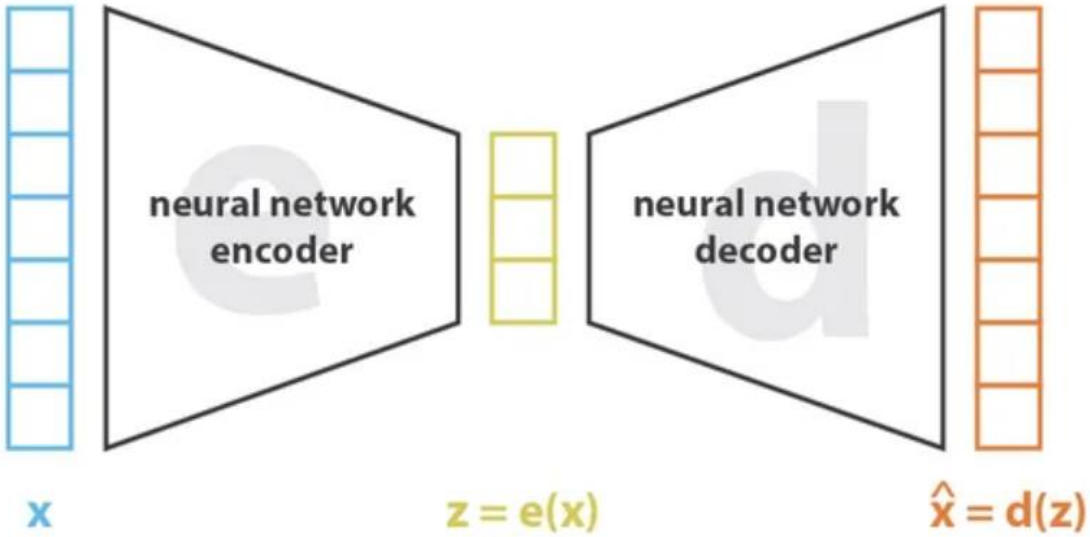
2. Preliminary and background

I. Generative models

- **Variational autoencoder (VAE)**

  VAE learns useful representations by reconstructing the input x with a latent variable z in consideration:

$$p(x) = \int p(x|z)p(z)dz.$$



2. Preliminary and background

I. Generative models

- **Generative adversarial networks (GANs)**

  A GAN is composed of a generator and a discriminator, where the generator aims to generate fake data while the discriminator aims to classify whether the data is from real datasets or generated.

  Eventually, the output distribution approximates the real distribution and the discriminator tends to be more accurate in the process.

2. Preliminary and background

I. **Generative models**

- **Energy-based models (EBM)**

  An energy function E(x,y) is defined to describe the distribution of the data (where x is input and y is output) instead of directly defining a probability distribution. In EBM every possible output corresponds to an energy value which represents the possibility of the output (lower energy, higher possibility and vice versa).

  In learning process, minimizing the average value of the energy function on the training data and maximizing the value of it in other places are usually involved, in order to allow the model to better distinguish between the good and bad input-output pairs.

2. Preliminary and background

I. **Generative models**

- **Energy-based models (EBM)**

  Energy function is used to indirectly define the probability distribution because of its flexibility. The model only needs to be able to distinguish between the good and bad input-output pairs, but doesn't need to know exactly the probability of every pair.

  Moreover, it's easier for EBM to handle complex and high-dimensional data (like images or texts) for they don't directly depend on probability distribution and thus have more freedom in designing the form of energy function.

2. Preliminary and background

I. Generative models

- **Normalizing flows**

  The basic idea of Normalizing Flows is to start with a simple known distribution and then transform that distribution into the target distribution through a series of reversible transformations ("flows"). These transformations are carefully designed so that we can compute their Jacobi determinant, thus allowing us to compute the transformed probability density directly.

$$lnp(x_n) = lnp(x_1) - \sum_i ln \left| det \frac{f_i}{x_{i-1}} \right|,$$

  where a transformation is $x_i = f_i(x_{i-1})$.

2. Preliminary and background

II. Reinforcement learning methods

RL is a computational approach to automating goal-directed learning and decision-making.

- Markov decision process
- Model-free methods
- Model-based methods
- Dyna Q-learning

2. Preliminary and background

II. Reinforcement learning methods

- **Markov decision process**

  An MDP model contains 5 elements: S, A, T, R, γ.

  The agent learns by collecting new experience data and optimizing a policy π for action selection.

  The key assumption of MDP is the Markov property: the next state of the system depends only on the current state and actions taken, not on past states or actions.

  The goal of an MDP is usually to find a strategy for what actions should be taken in each state in order to maximize the desired cumulative reward. In RL, such a strategy is usually obtained through learning.

2. Preliminary and background

II. Reinforcement learning methods

- **Model-free methods**

  Model-free methods do not try learning environment models, but directly learning policy or value function of state or state-action pair. It's simpler and easier to implement but probably needs more data.

RL algorithms	Related works
Value-based	Q-learning, DQN, Soft Q-learning
Policy-based	Policy gradient/REINFORCE, TRPO, PPO
Actor-critic	DPG, DDPG, Actor-critic, SAC, A3C

2. Preliminary and background

II. Reinforcement learning methods

- **Model-based methods**

  In RL, models come from prior knowledge or learning. For example, for an agent playing Go, the rules of Go are fixed, thus it's possible to get a perfect environment model by programming.

  Model-based methods seek to construct an environment model (usually includes state transition function and reward function) to simulate new experiences instead of depending only on real interacitons, thus needing less data.

  The main disadvantage is that sometimes it's not feasible to get a perfect model especially in complicated or high-dimensional environments.

2. Preliminary and background

II. Reinforcement learning methods

- **Dyna Q-learning**

  Dyna Q-learning combines model-free (simplicity and stability) and model-based (generating new experiences via simulation) methods, which have advantages of both methods.

  Main steps of Dyna Q-learning:

  a) Take an action.
  b) Direct RL.
  c) Model learning.
  d) Simulating RL.

  The disadvantage is also inherited from model-based methods like needing an accurate model which is infeasible sometimes.

3. Benefits

I. Solving the non-differentiable learning problems

In application, RL algorithms may need to interact with non-differential modules like discrete values (e.g. rgb values in images). RL algorithms are utilized to optimize these modules via policy gradient methods.

For example, a robot needs to move to the target place and A={1 step to N, 1 step to S, 1 step to E, 1 step to W} where A is discrete. A policy is defined as a neural network which receives a state as input and outputs a probability distribution. Trajectories (including s, a, r) are generated to compute the gradient of the expected rewards of the policy and update the parameters of the policy until the performance meats our requirements.

3. Benefits

II. Introducing new training signal

The major benefit of RL is flexibility, which means we can incorporate various training signals into the generation process.

- **Reward by discriminator**

  Discriminator in GAN fulfills a role akin to that of a reward in context of reinforcement learning.
- **Reward by hand-designed rules**

  The majority of RL algorithm implementations fall within this branch
- **Reward and divergence**

  Divergence can be a useful signal to be integrated into rewards. Here divergence (like KL divergence) is a measure of policy or value function, reflecting the similarity between policies or value functions. To avoid the policy from changing too fast, a term related to divergence is imported into reward function.
- **Reward by data-driven model**

  It is feasible to incorporate various guidance into RL by training a reward model.

3. Benefits

III. Sampling

RL is utilized to train the sampler, which generates samples (actions) from a certain probability distribution (policy distribution) that are later used along with corresponding rewards to update parameters of the model.

In certain complex situations, like probability distribution in high-dimensional space or that is infeasible to directly sample, samplers that are more complex (like softmax) or trained by RL are supposed to be used to navigate efficiently in state space.

$$\text{softmax}\left(\boldsymbol{z}\right) \triangleq \frac{1}{\sum_{l=1}^{k}\exp(z_l)}\left[\exp\left(z_1\right),\ \exp\left(z_2\right),\ \cdots,\ \exp\left(z_k\right)\right]^T$$

3. Benefits

IV. Neural architecture search

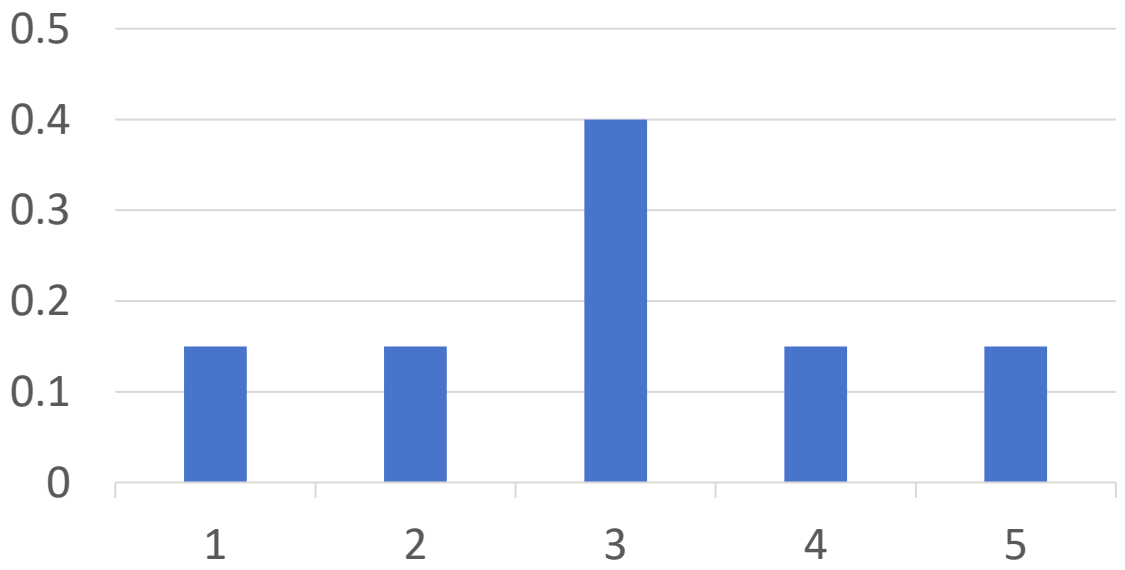
NAS is a technique used to optimize neural network architecture, seeking for the optimal neural network architecture by searching algorithms without manual designing or adjustment.

The “architecture” includes number of layers, connections between layers, number of neurons per layer and so on. While “search” means seeking for the optimal solution via a certain algorithm in the architecture space.

4. Challenges

I. Peaked distribution

In the context of generative models, peaked distribution refers to the probability distribution of the data generated by the model where there is an obvious peak value. For example, it's a model generating a number from 1 to 5.



4. Challenges

I. **Peaked distribution**

For generative models, peaked distribution can be negative in many situations, like in a text generation model where diverse outputs are usually wanted instead of always similar sentences or paragraphs.

Possible reasons can be over-fitting, data bias, problem of architecture or parameters, optimization process and lack of training.

4. Challenges

II. Exploitation and exploration

In RL, an agent must balance a trade-off between exploitation and exploration: to maximize the expected rewards, the agent must exploit the best actions used before. While to learn the potential best actions, the agent must explore all possible situations where some are not experienced before.

Over-exploring may lead to missing the known optimal policy, while over-exploiting possibly causes stucking in a local optimum but missing better policies.

An example of the solutions of this is Proximal Policy Optimization (PPO), the target function of which contains an extra term used to punish the new policies changing too much to avoid large oscillations.

4. Challenges

III. Sparse rewards

Obviously, rewards are training signals needed by the agent to learn how to take good actions. However, in many situations the agent may need to take lots of steps to get one single reward and between these steps it doen't reveice any feedbacks.

Image a maze environment where the agent only gets a reward when finding the exit, before which it won't receive any rewards or penalties. In this circumstance the agent hardly knows what actions are helpful considering lack of feedbacks.

Sparse rewards make the agent quite hard to find effective policies unless a great deal of exploration is taken.

Solutions are shaped rewards, imitation learning and so on.

4. Challenges

IV. Long-term credit assignment

It's difficult for a model to learn and understand causal relationships in input sequences across long time spans. In other words, if the result of an action only appears after many steps, the model hardly can attribute the result to that early action.

One solution to it is Hierarchical RL, one framework of which is called options framework. In this framework, manager (in higher level) decides which sub-task or option is to be excuted. While worker is supposed to execute it in a lower level. This structure helps the model to explore and learn more efficiently especially when facing complex tasks and large state-action spaces.

4. Challenges

IV. Generalization

In generative models, generalization measures the performance of the model on training dataset and new unseen dataset. If it performs well on training dataset and badly on new unseen dataset, we can tell that this model is over-fitting and weak in generalization.

The solutions include regularization, early stopping, dropout, etc.

5. Application

I. **Neural language processing**

- **Text summarization**

  It is the process of automatically generating or extracting summaries from a given input document without losing important information.

- **Machine translation**

  RL is applied to bridge the train and test metric gap, to direct translation with sentiment preserved, to simplify sentences…

  Bilingual evaluation understudy (BLEU) is used to compare the results of machine translation and human translation to evaluate the performance of machine translation, the main idea of which is comparing the similarity of their word choices and sentence structures via “n-gram”.

5. Application

I. Neural language processing

- **Dialog system**

  Dialog system is an interactive system that is able to communicate with human using natural language via speech or text, including:

  a) Open-domain dialog system
  b) Task-oriented dialog system
  c) Hybrid dialog system

- **Human value alignment and constraints**

  Sometimes the output of a model is not matched with human values, like fake information and output that do not match human values. RL is used to adjust the model to address this problem.

- **Text, queries and knowledge graph**

  RL is used to translate natural language into structured queries and knowledge graphs.

- **Sequence generation**

5. Application

II. Code generation

III. Computer vision

- Image captioning:
  describe an image
- Visual question answering (multimodal task):
  given an image and answer the question
- Visual dialog system:
  handle dialog continuity and context dependency
- Text-to-image generation
- 3D generation

IV. Speech and music generation

V. AI for science

- Molecule design
- Reaction optimization
- Quantum architecture design

5. Application

VI. Recommender system and information retrieval

- States: user interaction history
- Actions: items
- Rewards： user feedback

VII. Robotics

Robotics can be treated as an interactive agent that generates responses to humans' orders. Large language models plus visual foundation models might lead to a large step towards better robotics control policies.

IV. Other areas

6. Future directions

I. Reward function design and multi-objective optimization
II. Model enhancement and control
III. Human preference modeling and interpretability
    - Human preference modeling: understand and predict humans' behaviors and decisions
    - Interpretablity: model's predictive ability and transparency of its working principle
IV. Sample-efficiency and generalization
V. Introducing of novel RL methods
VI. LLM and foundation models

Thank you.