Paper: SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient

Summary

In this paper, the authors address two issues in the applying Generative Adversarial Network (GAN) to generate sequences. Firstly, GAN is designed for generating real-values, continuous data but has difficulties indirectly generating sequences of discrete tokens, such as texts. Second, GAN can only give the score/loss for an entire sequence when it has been generated; for a partially generated sequence, it is non-trivial to balance how well as it is now and the future score as the entire sequence. To address these issues, they consider the sequence generation procedure as a sequential decision-making process. The generative model is treated as an agent of reinforcement learning (RL); the state is the generated tokens so far and the action is the next token to be generated. Unlike other approaches that use a task-specific sequence score such as BLUE, to give the reward, they employ discriminator to evaluate the sequence and feedback the evaluation to guide the learning of the generative model. To solve the problem that gradient cannot pass back to the generative model when the output is discrete as suffered by GAN, they regard the generative model as a stochastic parameterized policy. In their policy gradient, they employ a Monte Carlo search to approximate the state-action value.

The first train a generative model $G_\Theta$on a real-world structured sequence, the dataset to produce a new sequence. They interpret this problem based on reinforcement learning. Additionally, they also train a Ø-parameterized discriminative model $D_\emptyset$ to guide for improving generator G_Θ. The discriminative model $D_\emptyset$ is trained by providing positive examples from the real sequence data and negative examples from the synthetic sequences generative from the generative model G_Θ. At the same time, the generative model G_Θ is updated by employing a policy gradient and MC search based on the expected end reward received from the discriminative model $D_\emptyset$. The reward is estimated by the likelihood that it would fool the discriminative model $D_\emptyset$. For estimating the action-value function, they use REINFORCE algorithm and consider the estimated probability of being real by the discriminator $D_\emptyset$ as a reward. A benefit of using a discriminator $D_\emptyset$ as a reward function is that it can be dynamically updated to further improve the generative model iteratively. They use RNNs as the generative model and CNN as the discriminative model.

To test the efficacy of their model, they conducted a simulated test with synthetic data. They use the randomly initialized LSTM as the true model, which they call oracle, to generate the real data distribution. The benefit of using such an oracle is that it provides the training dataset and also evaluates the exact performance of the generative models, which will not be possible with real data. They found out that their SeqGAN significantly outperforms other baselines. They analyzed the learning curves which also illustrated the superiority of SeqGAN. After 150 training epochs, both the maximum likelihood estimation and the schedule sampling methods converge to a relatively high NLLoracle score, whereas SeqGAN can improve the limit of the generator with the same structure as the baselines significantly. This indicates the prospect of applying adversarial training strategies to discrete sequence generative models to break through the limitations of MLE.

They also tested their model on various real-world examples such as poem composition, speech-language generation, and music generation. For text generation, they applied their proposed model to generate Chinese poems and Barak Obama political speeches. They used the

BLEU score as an evaluation metric to measure the similarity degree between the generated texts and the human-crated texts. They found out the significant advantage of SeqGAN over the MLE in text generation. In poem composition, their model performs comparably to real human data.

Strengths

- The paper is very well organized with enough sections and subsections, each with relevant details.
- They also give enough details about their algorithm and parameters, which makes it easy for researchers to try to apply or extend their work.
- They present comprehensive results with synthetic as well as real-world examples.

Weaknesses

- I would have loved to see some demos. For example, poems, music, or speech generated by their model.

Confusions

- What is oracle and what is it used for?
- How does considering the generative model as a stochastic parameterized policy solve the problem of gradient not passing back to the generative model when the output is discrete?

Discussions

- What is there are no structured real-world data? Can this procedure still be applied to generate sequences?
- Can these models be applied to combinatorial optimization problems such as TSP?