June 2018, DIRECT Capstone

DIRECT
Data Intensive Research
Enabling Clean Technologies

Generative Molecules: Automatic Molecules Design by Deep Generative Models

Yize Chen[1], Xiaoxiao Jia[2], Jiaxu Qin[3]
Department of Electrical Engineering[1], Department of Materials Science and Engineering[2] and Molecular Engineering Institute[3]
University of Washington, Seattle, WA

Introduction

Backgrounds

- Molecule discovery/drug design are of key interests
- Deep learning opens the door for learning molecular patterns/distribution

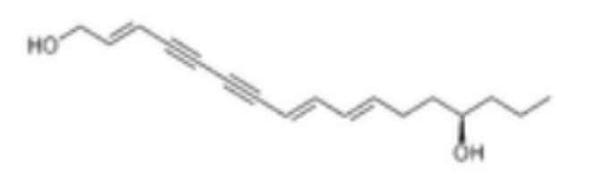
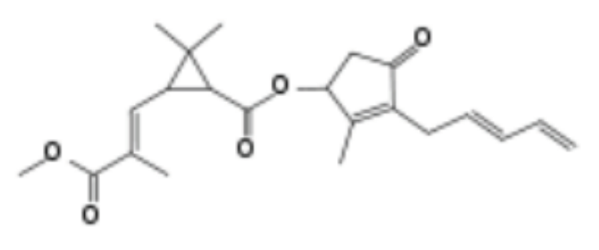
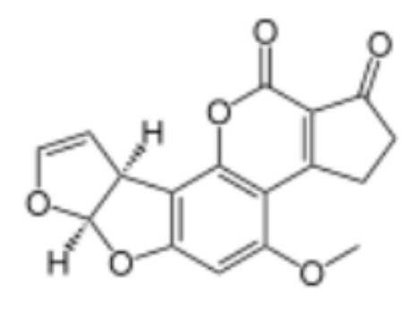
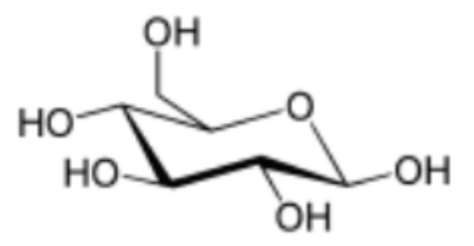
Challenges

- Discrete representation of molecules
- Hard ML problem: learn to generate rather than learn to classify
- Molecular design with targeted properties

Proposed Methods

- One-hot representation of molecules
- Deep Generative Adversarial Networks(GANs) for generative learning
- Deep Q network for reinforcement learning

Dataset Description

Graph	SMILES string
	CCC[C@@H](O)CC\C=C\C=C\C#CC#C\C=C\CO
	COC(=O)C(\C)=C\C1C(C)(C)[C@H]1C(=O)O[C@@H]2C(C)=C(C(=O)C2)CC=CC=C
	O1C=C[C@H]([C@H]1O2)c3c2cc(OC)c4c3OC(=O)C5=C4CCC(=O)5
	OC[C@@H](O1)[C@@H](O)[C@H](O)[C@@H](O)[C@@H](O)1

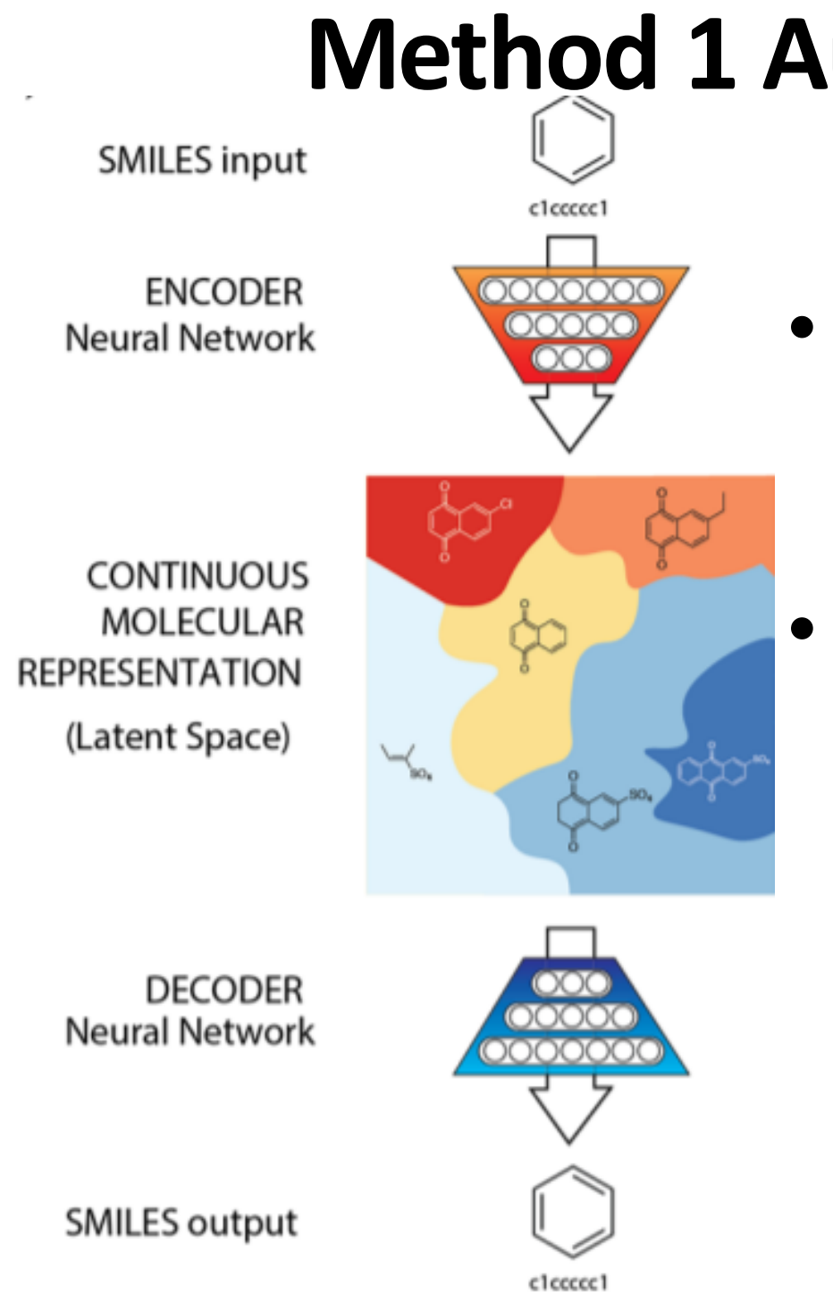
Training set
108,000 molecules from QM9 dataset;
120,000 molecules from ZINC dataset
One-hot encoding
Encode and unify length of encoded vectors.

From Discrete to Continuous

Motivation

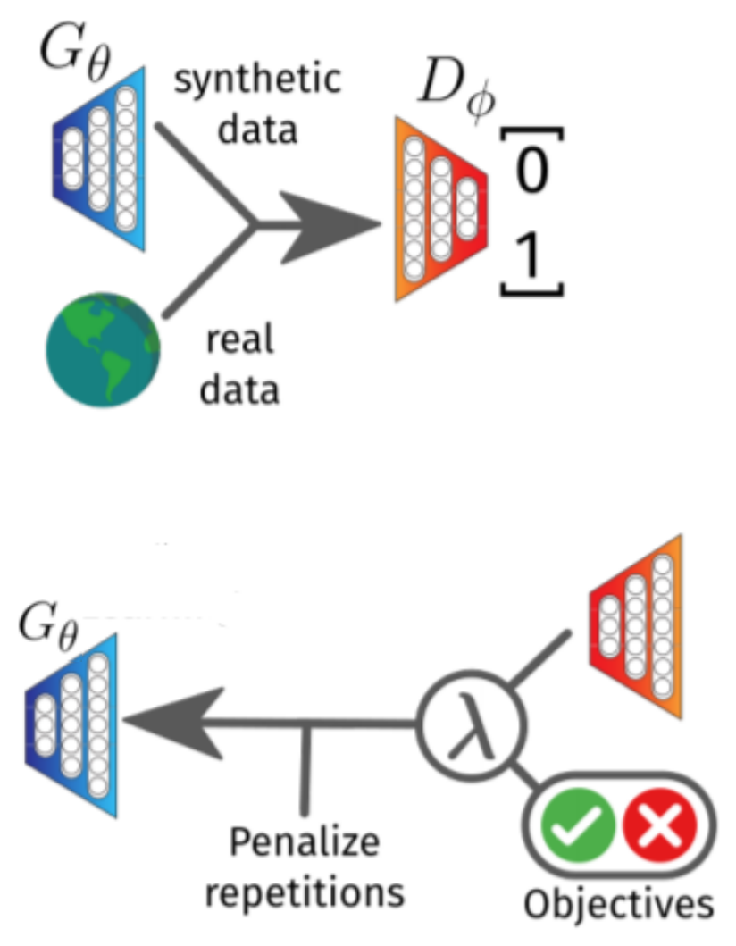
- Even with one-hot encoded molecular vector, data is sparse and discrete;
- Gradient descent method requires meaningful gradient updates.

Method 1 Autoencoder



- Encode discrete vector to continuous space;
- Utilize continuous vector for inference

Method 2 Policy Gradients



- Monte Carlo search with rollout policy;
- Deterministic policy gradients on actions

Generative Adversarial Networks

Minimax game **for generator and discriminator**

$$\min_{\phi} \mathbb{E}_{Y\sim p_{\text{data}}(Y)}\left[\log D(Y)\right] + \mathbb{E}_{Y\sim p_{G_\theta}(Y)}\left[\log(1-D(Y))\right]$$

- Wasserstein distance to distinguish generated distribution from training data distribution
- Once trained, generator is able to generate realistic encoded vectors

Policy Gradients **for evaluating generated molecules**

Calculate the cumulated reward from discriminator

$$J(\theta) = E[R(Y_{1:T})|s_0,\theta] = \sum_{y_1\in Y} G_\theta(y_1|s_0)\cdot Q(s_0,y_1)$$

Update generator based on rewards:

$$\nabla_\theta J(\theta) \simeq \frac{1}{T}\sum_{t=1,\ldots,T} \mathbb{E}_{y_t\sim G_\theta(y_t|Y_{1:t-1})}[\,\nabla_\theta \log G_\theta(y_t|Y_{1:t-1})\cdot Q(Y_{1:t-1},y_t)]$$

Design Highlights:

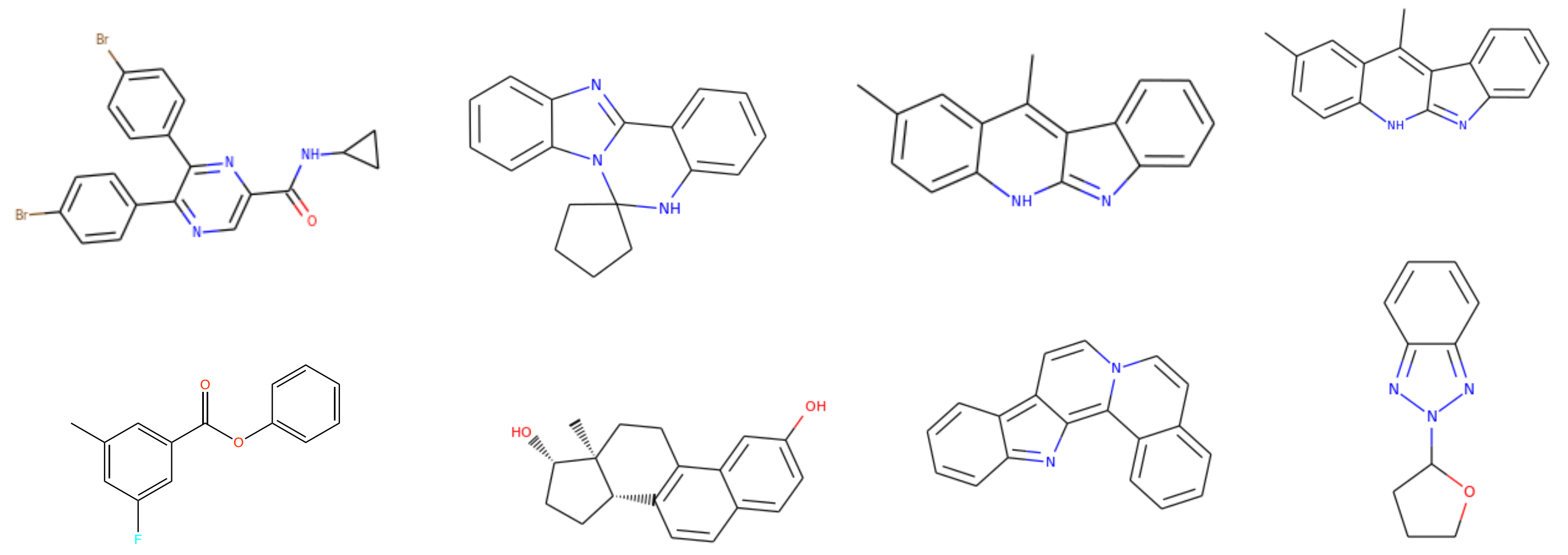
- Early stopping during training;
- Stochastic policy update w.r.t long-term rewards;
- Advanced network structures:
  - deep convolutional layers for encoder-decoder networks;
  - LSTM units for policy for discriminator

Implementation

- Keras, Tensorflow as deep learning platforms for LSTM, Conv-net;
- Stochastic gradient descent for optimization;
- RDKit for validating and visualizing generated molecules;

Results

Examples of Generated Molecules

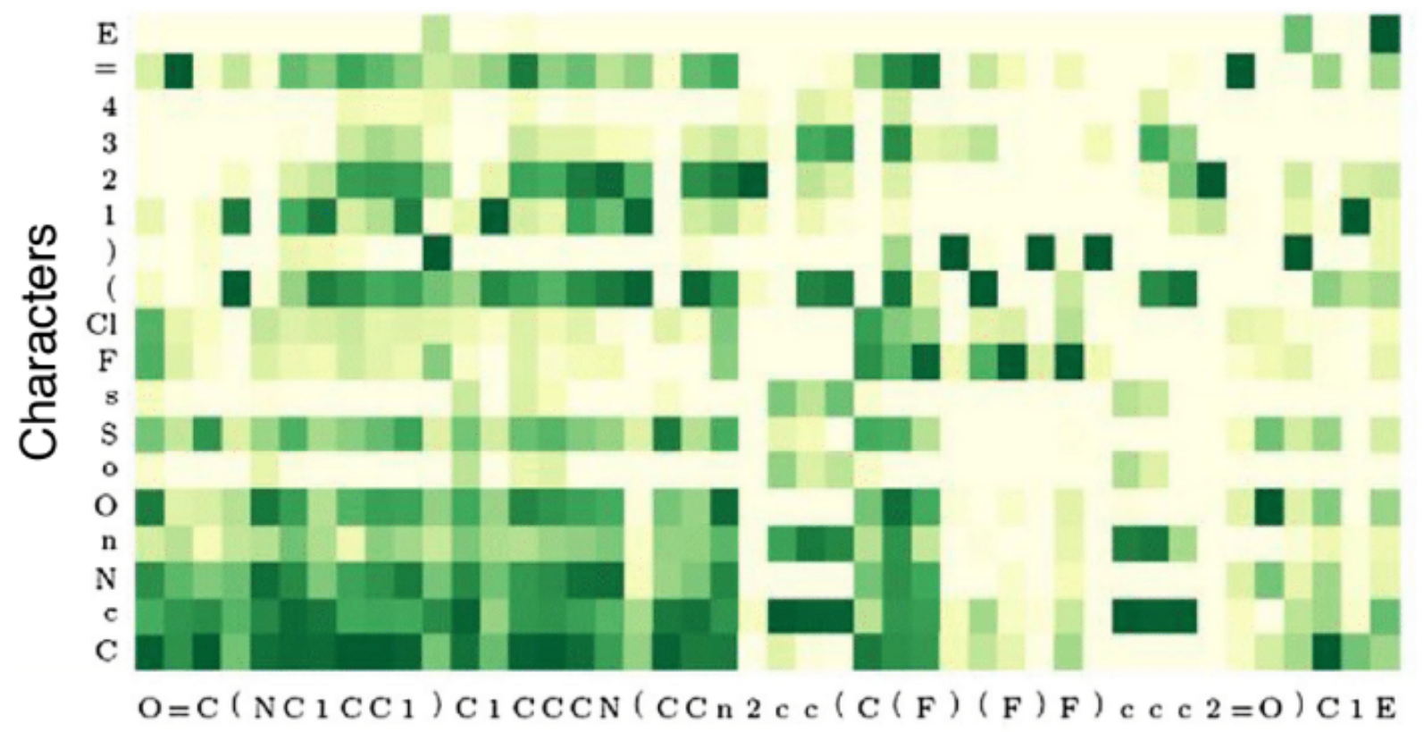


Advantages

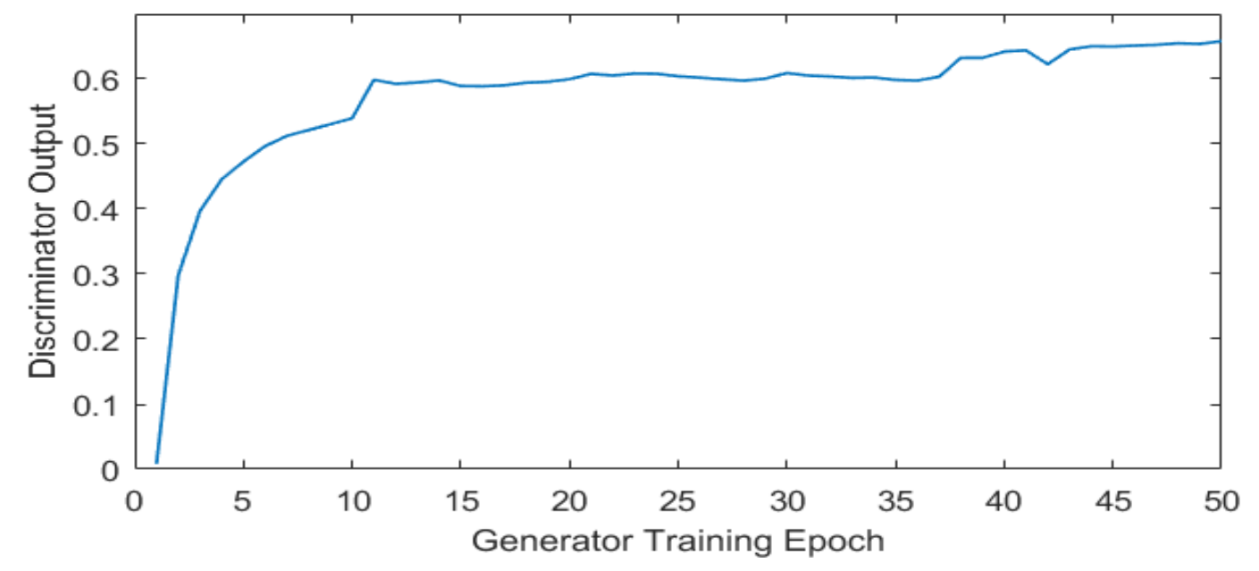
- Generated molecules with various modes and structures;
- Generate new structures unique from training sets;
- Conditional generation based on input labels;

Drawbacks

- Invalid generated molecules;
- Hard to generate larger and longer molecules;
- Some generations, e.g., CH_4, are repeated multiple times



Output example from generator; deeper color denotes higher probability



Conclusions & Future Work

Conclusion

- GAN is a powerful generative model for automatically learning molecules patterns and properties;
- Reinforcement learning and autoencoder would help for representing discrete original data;

Future Works:

- Interpolation, conditional generation will help for targeted generation;
- Adding loss terms during training to avoid invalid generated molecules

References:
[1] Gómez-Bombarelli, Rafael, et al. "Automatic chemical design using a data-driven continuous representation of molecules." *ACS Central Science* (2016).
[2] Guimaraes, Gabriel Lima, et al. "Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models." *arXiv preprint arXiv:1705.10843* (2017).
[3] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016, November). TensorFlow: A System for Large-Scale Machine Learning. In *OSDI* (Vol. 16, pp. 265-283).
[4] Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.

**We would like to thank Prof. Baosen Zhang for precious advice and discussions.*