Deep Molecule Generation

Kobby Panford-Quainoo
Aisha Alaagib
African Institute of Mathematical Sciences

Outline

1. Representing molecules
2. Sequence models
3. Generative models
4. Applications

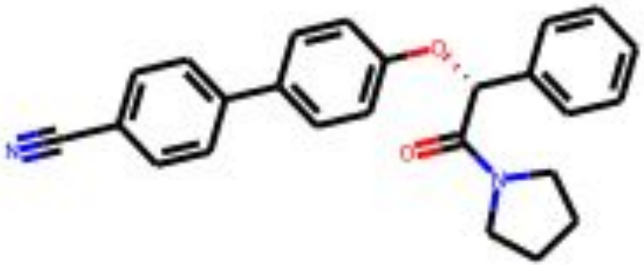
Motivation

- **Need for speed** from discovery to production
- **Avoid exhaustive search** for possible combinations of chemical atoms
- **Create creative and innovate innovative** drug generating machines

Machine Learning is the way to go! you?

How to represent Molecules

- String representation (with SMILES notations)
- Graph representation



SMILES notations

SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules

DAVID WEININGER

Medicinal Chemistry Project, Pomona College, Claremont, California 91711

Received June 17, 1987

SMILES (Simplified Molecular Input Line Entry System)

- Treat atoms and bonds as sequence of ASCII characters
    - c1ccccc1   Benzene
    - c1c(N(=O)=O)cccc1   Nitrobenzene

SMILES (Simplified Molecular Input Line Entry System)

- Treat atoms and bonds as sequence of ASCII characters
  - c1ccccc1   Benzene
  - c1c(N(=O)=O)cccc1   Nitrobenzene
- Each character can be given a unique index and/or one-hot encoded

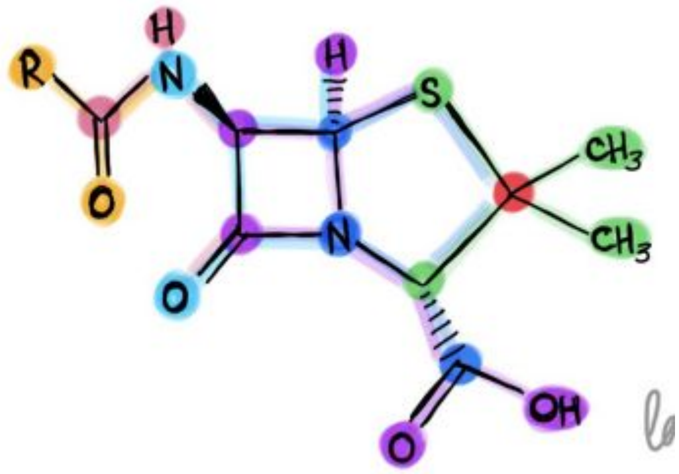
vocab = $\{0, 1, c, C, N, =, (,)\}$

$$
\begin{pmatrix} 0 \\ 1 \\ c \\ C \\ N \\ = \\ (\\ ) \end{pmatrix}
\quad
\begin{array}{c}
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \\
\begin{pmatrix} c & 1 & c & c & c & c & c & 1 \end{pmatrix}
\end{array}
$$

SMILES (Simplified Molecular Input Line Entry System)

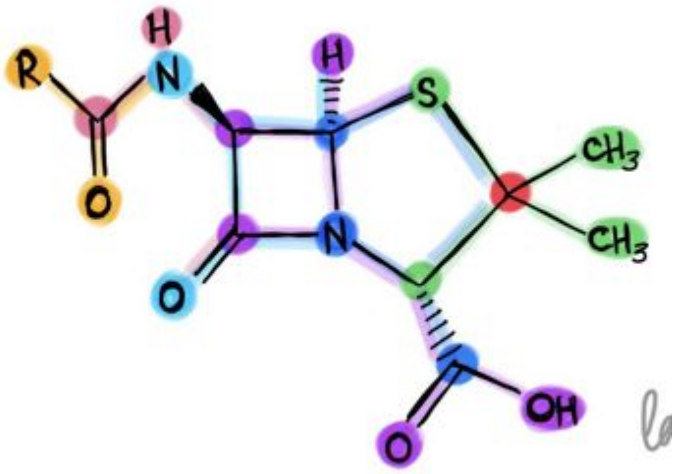
- Treat atoms and bonds as sequence of ASCII characters
    - c1ccccc1 Benzene
    - c1c(N(=O)=O)cccc1 Nitrobenzene
- Each character can be given a unique index and/or one-hot encoded
- Sequence models from NLP could be used to do learning on this data representation ie. RNN, LSTM etc
- This is a successful approach for property prediction, molecular generation etc

Graphs



Credit: Michael Bronstein

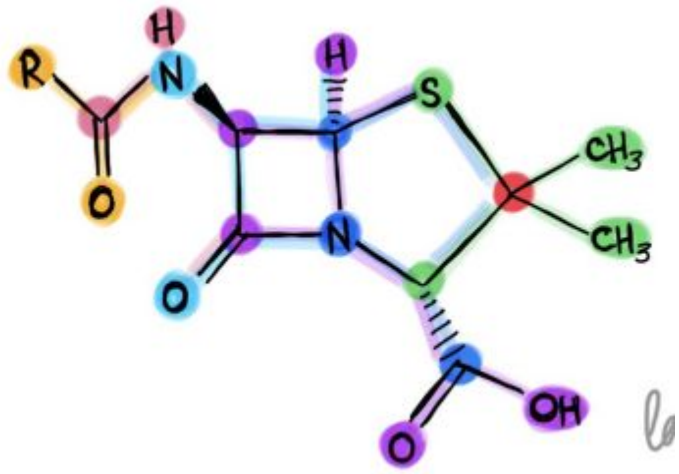
Graphs



Credit: Michael Bronstein

- Graph is an arbitrary data structure made up of entities called **nodes** and connected to each other by **edges**.

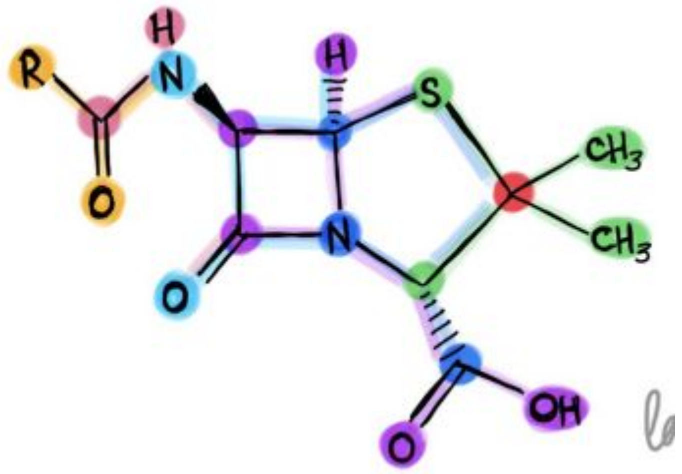
Graphs



Credit: Michael Bronstein

- Graph is an arbitrary data structure made up of entities called **nodes** and connected to each other by **edges**.
- Atoms are **nodes**, bonds are **edges**

Graphs

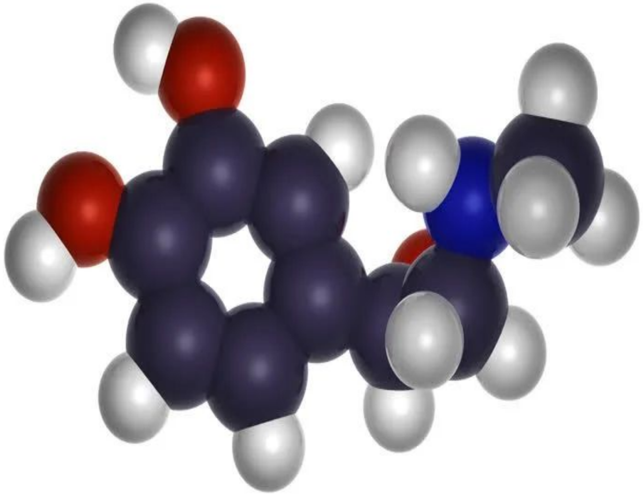


Credit: Michael Bronstein

- Graph is an arbitrary data structure made up of entities called **nodes** and connected to each other by **edges**.
- Atoms are **nodes**, bonds are **edges,** different bond types are the different **edge types**
- Graph Neural Networks presents a family of deep learning techniques applicable to graphs.
  - Graph Convolutional Networks
  - Message passing Neural Networks
  - Graph Attention Networks etc

Other forms of representation

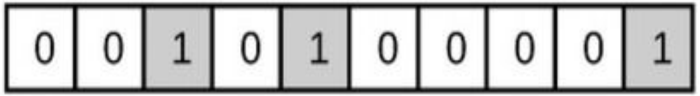
- 2D and 3D image representation



David, L., Thakkar, A., Mercado, R. *et al.* Molecular representations in AI-driven drug discovery: a review and practical guide. *J Cheminform* 12, 56 (2020). https://doi.org/10.1186/s13321-020-00460-5)

Other forms of representation

- fingerprints



David, L., Thakkar, A., Mercado, R. *et al.* Molecular representations in AI-driven drug discovery: a review and practical guide. *J Cheminform* 12, 56 (2020). https://doi.org/10.1186/s13321-020-00460-5)

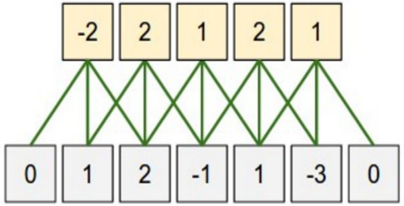
Other forms of representation

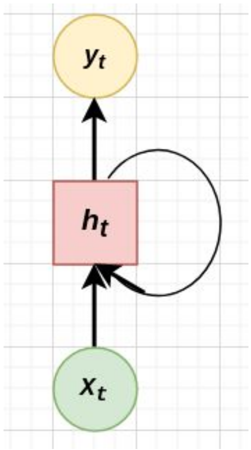
- 2D and 3D image representation
- fingerprints
- There are many other possible ways (ref. has more)

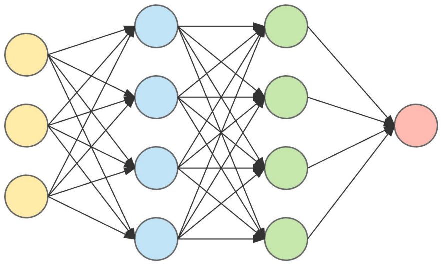
David, L., Thakkar, A., Mercado, R. *et al.* Molecular representations in AI-driven drug discovery: a review and practical guide. *J Cheminform* 12, 56 (2020). https://doi.org/10.1186/s13321-020-00460-5)

Learning molecular structures "deeply"

- Key tools
  - Feedforward Neural Networks
  - Recurrent Neural Networks
  - Convolutional Neural Networks

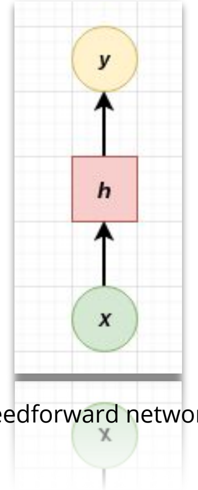




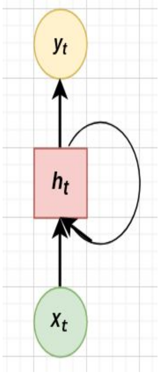


Recurrent Neural Networks

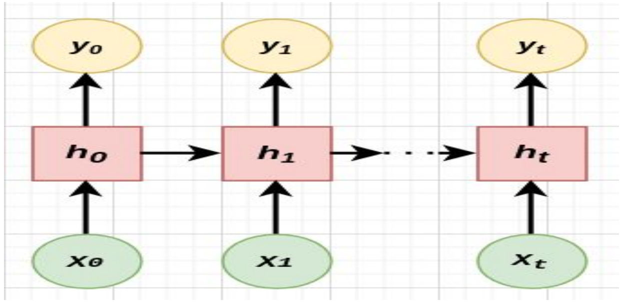
- Successful for modeling sequential data $S = \{s^1, s^2, \ldots, s^T\}$
- Feedforward neural networks that learn across timesteps through recurrent connections.



Feedforward network



RNN loop

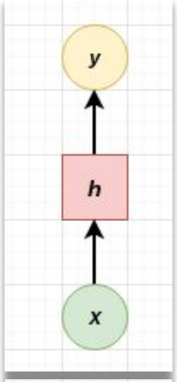


Unrolled RNN

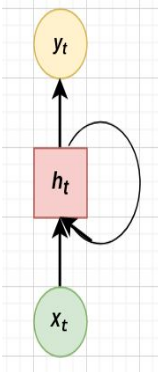
Recurrent Neural Networks

- Successful for modeling sequential data
- Made up of feedforward neural networks that learn across timesteps through recurrent connections.
- Given a sequence ie $S = \{s^1, s^2, \ldots, s^T\}$, RNN assigns a probability to the sequence as

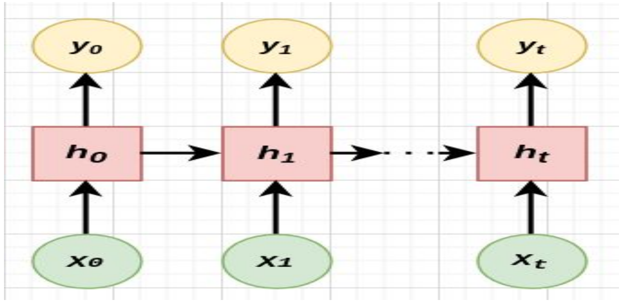
$$P_\theta(S) = P_\theta(s_1)\prod_{t=2}^{T} P_\theta(s_t|s_{t-1},\ldots,s_1)$$



Feedforward network



RNN loop



Unrolled RNN

Recurrent Neural Networks

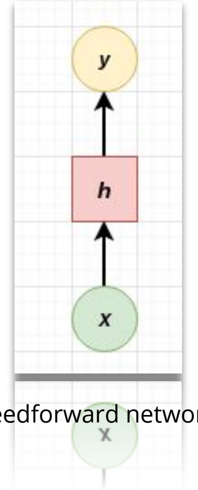
- Successful for modeling sequential data
- Made up of feedforward neural networks that learn across timesteps through recurrent connections.
- Given a sequence ie $S = \{s^1,\ s^2, \ldots, s^T\}$ RNN assigns a probability to the sequence as

$$P_\theta(S) = P_\theta(s_1) \prod_{t=2}^{T} P_\theta(s_t | s_{t-1}, \ldots, s_1)$$

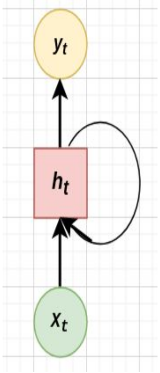
- RNN maintains a hidden state which is continuously updated at each timestep by a function f_w

$$h_t = f_w(h_{t-1}, s_t)$$

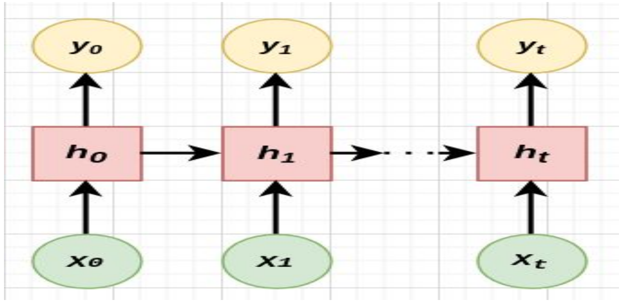
- There are other variants like LSTM and GRU (uses memory and gating to capture long term dependencies)



Feedforward network

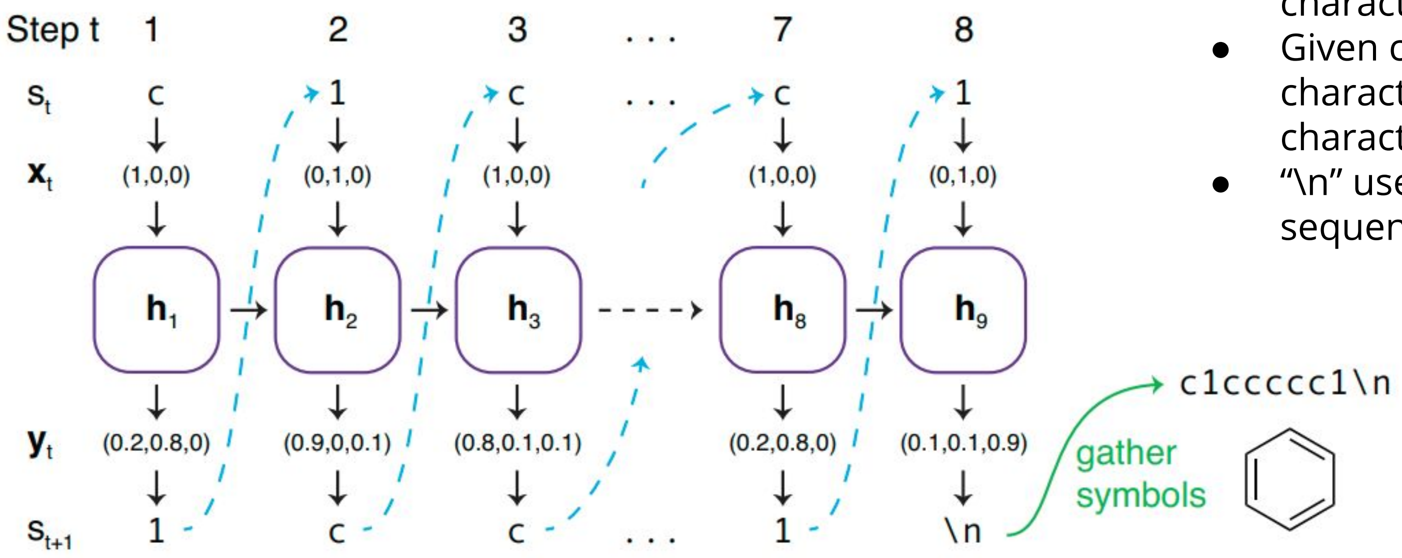


RNN loop



Unrolled RNN

Recurrent Neural Networks



- Molecule generation as next character prediction problem
- Given current token/ input character predict next character
- “\n” used to signify end of sequence

Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks, Marwin H. S. Segler et al, DOI: 10.1021/acscentsci.7b00512

Discriminative vs Generative Models

Discriminative

- Discriminates between features
- Computes conditional probability $P(Y|X)$
- Or basically predict labels given features:

  X $\rightarrow$ Y
- Used for supervised learning tasks
- Classification and regression models

Generative

- Computes joint distribution $P(X,Y)$ or $P(X)$

  Or a simulation of the data generation process
- Can generate features (belonging to a class)
- Useful for unsupervised learning tasks

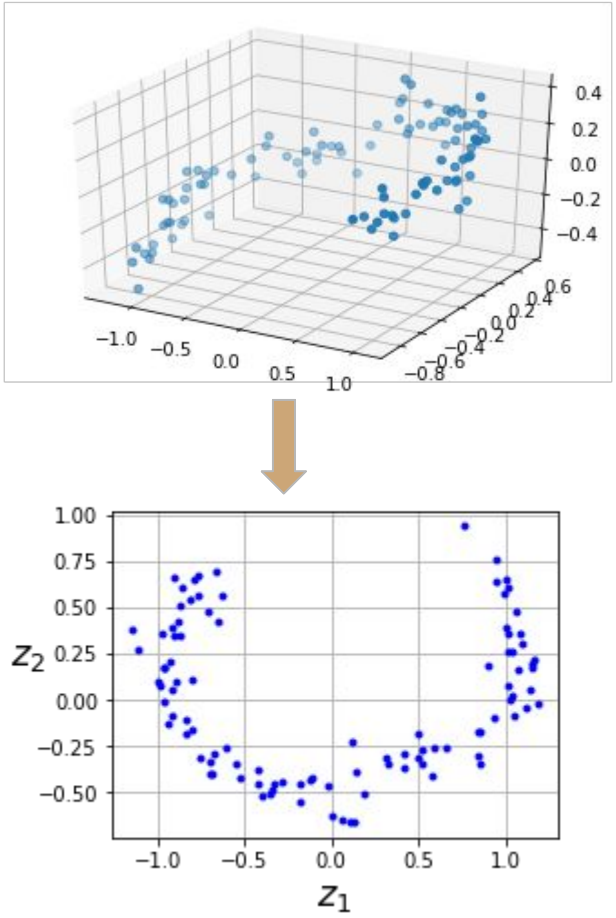
Generative Models

1. Commonly known deep generative models
    a. Variational Autoencoders (VAEs)
    b. Generative Adversarial Networks (GANs)
    c. Normalizing flows (NFs)

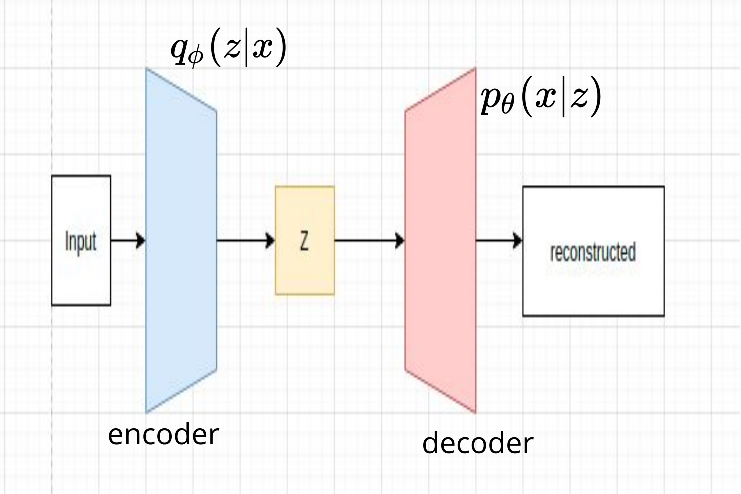
In the next slides, we'll discuss VAEs and GANs

Autoencoders

- The encoder encodes the data into a low-dimensional representation called *latent vector/ code* z



Autoencoders



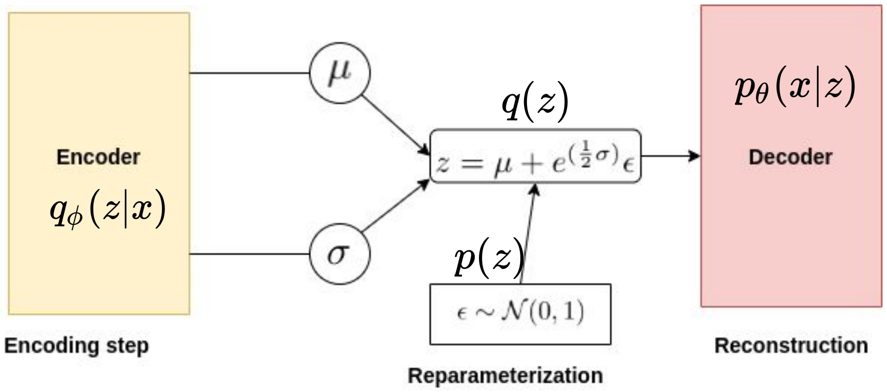
- An autoencoder has an encoder and a decoder
- The decoder attempts to reconstruct the input features from the latent code
- Training objective → Minimize reconstruction error

$$\min \underbrace{||\text{input} - \text{reconstructed}||}_{\text{Distance between original and reconstructed features}}$$

$$\mathcal{L}_{\text{REC}} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}\,|\,\mathbf{x})} \log p_\theta(\mathbf{x}\,|\,\mathbf{z})$$

Variational Autoencoders

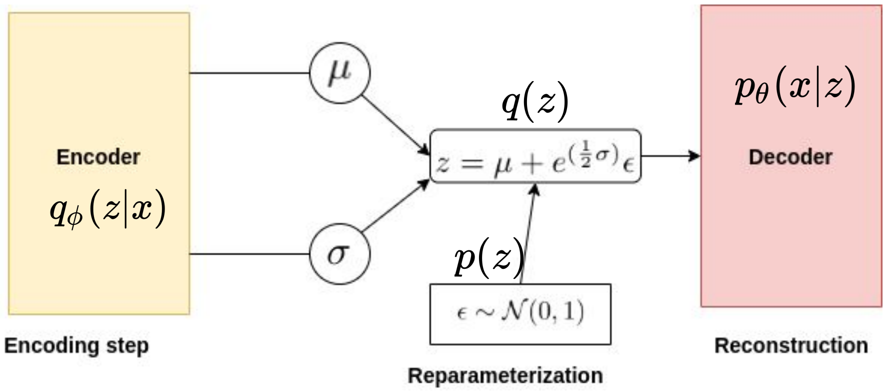
- Encoder + Decoder
- Learns a distribution instead of fixed latent vector/ code



Auto-Encoding Variational Bayes, Diederik P Kingma and Max Welling (2013)

Variational Autoencoders

- Encoder + Decoder
- Learns a distribution instead of fixed latent vector
- Training objectives: Evidence lower bound (ELBO)
    - Reconstruction loss
    - KL divergence loss



$$\arg\min_{\phi,\theta} \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} \mathcal{L}_{\text{ELBO}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} \mathcal{L}_{\text{REC}} + \mathcal{L}_{\text{KL}}$$

$$\mathcal{L}_{\text{REC}} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}\,|\,\mathbf{x})} \log p_\theta(\mathbf{x}\,|\,\mathbf{z})$$

$$\mathcal{L}_{\text{KL}} = \mathbb{KL}(q_\phi(\mathbf{z}\,|\,\mathbf{x})||p(\mathbf{z}))$$

Auto-Encoding Variational Bayes, Diederik P Kingma and Max Welling (2013)

Generative Adversarial Networks

I can create money

Generator

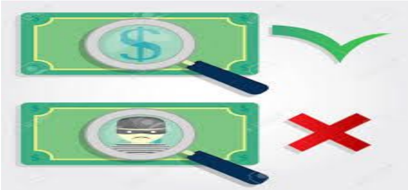
The MINIMAX game

I can detect fake money

Discriminator

Generative Adversarial Networks,   J. Goodfellow and Jean Pouget-Abadie and Mehdi Mirza and Bing Xu and David Warde-Farley and Sherjil Ozair and Aaron Courville and Yoshua Bengio (2014)

Generative Adversarial Networks

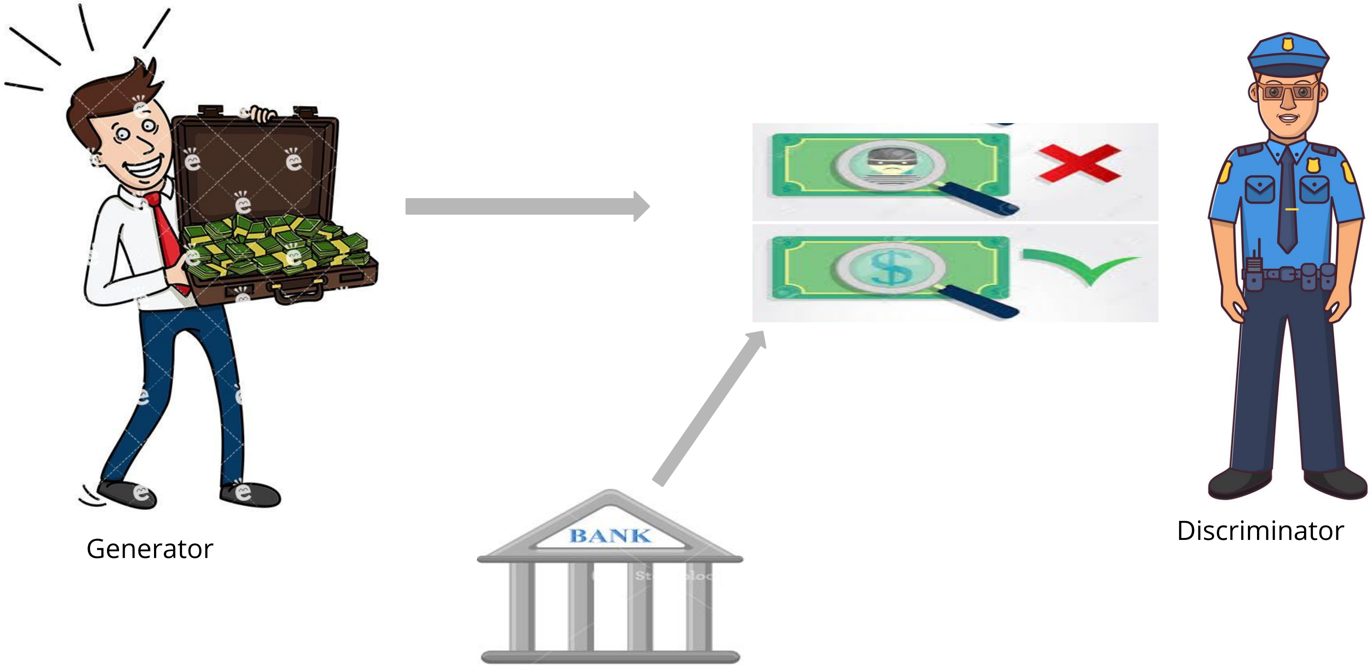


Discriminator

Discriminator needs to trains to detect fake currencies from real ones

Generative Adversarial Networks

I got the difference between true and fake currencies



Generator

Discriminator

- Trains on currencies from both sources (Bank and Generator)

Low generator loss

Generative Adversarial Networks

Hey, I got this money from the bank

Generator

Discriminator

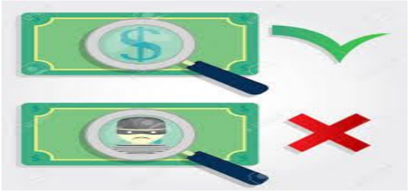
I’m convinced → Low generator loss

I’m not convinced → High generator loss

- Generator tries to fool the discriminator by labeling the fake currencies as true ones

Generative Adversarial Networks

Generator

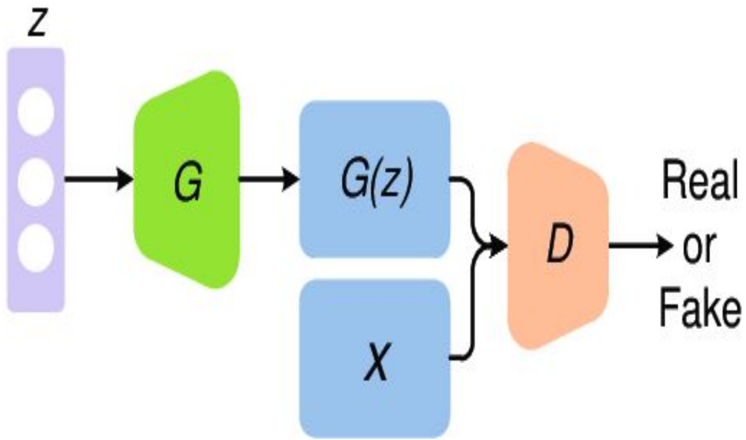


Discriminator

- Game continues till they both become good at their tasks.
- Now, if you need new kind of money, Go to the generator! :)

Generative Adversarial Networks - More formally

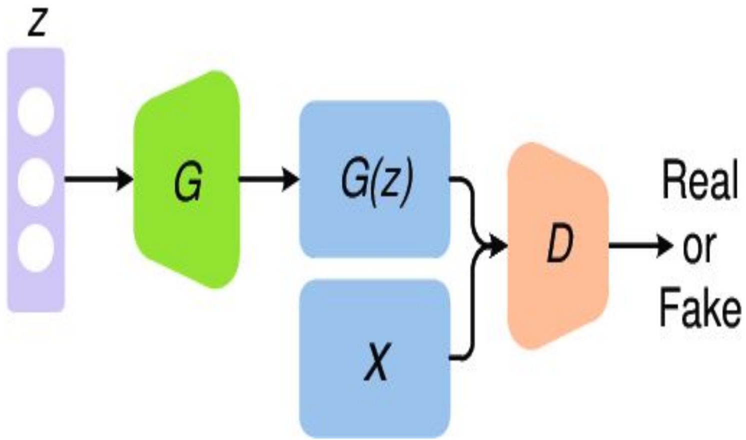
- Generative model made up of a **generator** and a **discriminator**



img: Zhaoqing Pan

Generative Adversarial Networks

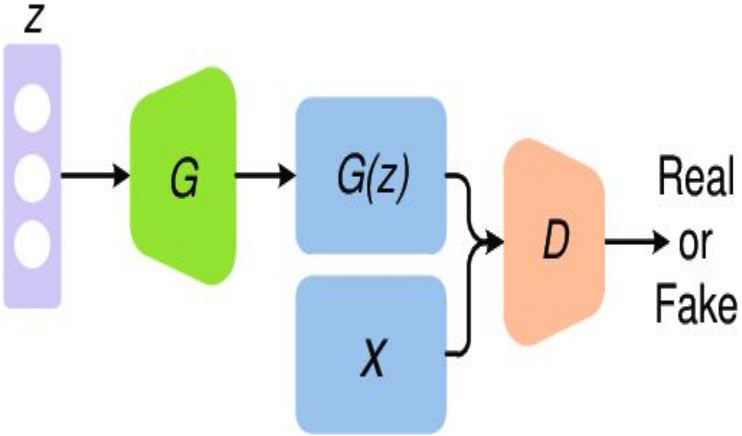
- Generative model made up of a **generator** and a **discriminator**
    - The generator tries to generate data (from arbitrary input) and wants it to look like an instance of the *true* data



img: Zhaoqing Pan

Generative Adversarial Networks

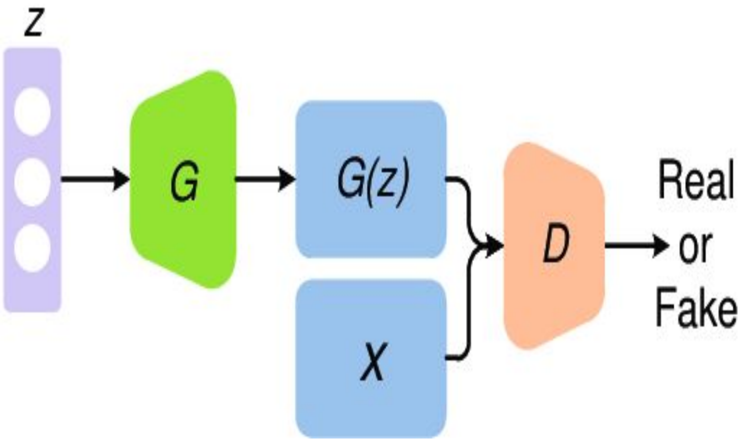
- Generative model made up of a **generator** and a **discriminator**
  - The generator tries to generate data (from arbitrary input) and wants it to look like an instance of the *true* data
  - The discriminator discriminates between a true instance of the data and a fake one created by the generator.



img: Zhaoqing Pan

Generative Adversarial Networks

- Generative model made up of a **generator** and a **discriminator**
    - The generator tries to generate data (from arbitrary input) and wants it to look like an instance of the *true* data
    - The discriminator discriminates between a true instance of the data and a fake one created by the generator.
- They are both trained alternatingly so each one gets better and better at doing their jobs.

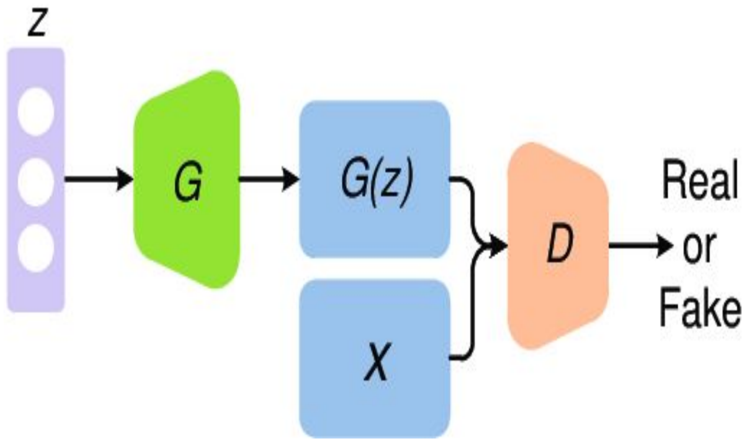


img: Zhaoqing Pan

Generative Adversarial Networks

- Generative model made up of a **generator** and a **discriminator**
  - The generator tries to generate data (from arbitrary input) and wants it to look like an instance of the *true* data
  - The discriminator discriminates between a true instance of the data and a fake one created by the generator.
- They are both trained alternatingly so each one gets better and better at doing their jobs.
- Training objective (generator minimizes, discriminator maximizes)

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

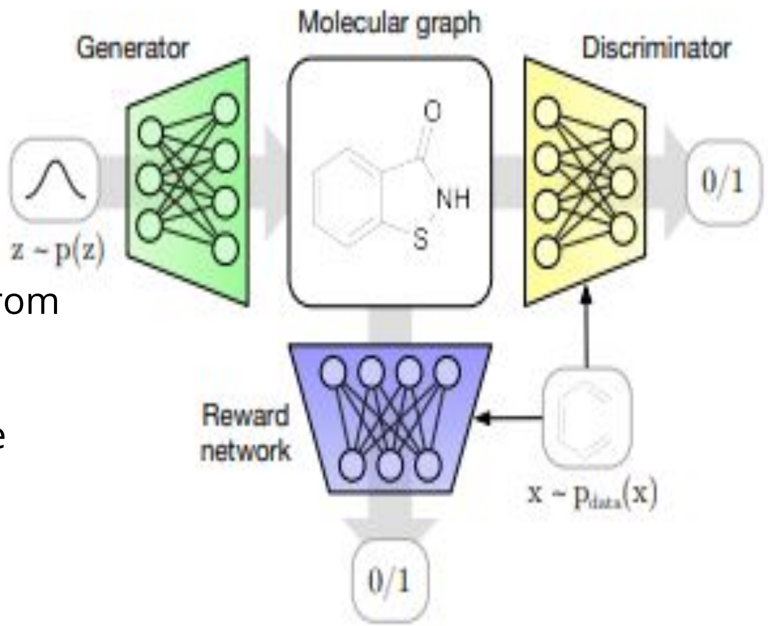


img: Zhaoqing Pan

Applications of Generative models in Drug design

MolGAN (GAN + RL + graph rep.)

- Represent SMILES molecules as graphs
- Adversarial training (Generator + Discriminator)
- Generator generates molecular graph from prior
- Discriminator tells whether the input it receives is from generator or data distribution
- Reward network ensures generated molecules have desired chemical properties

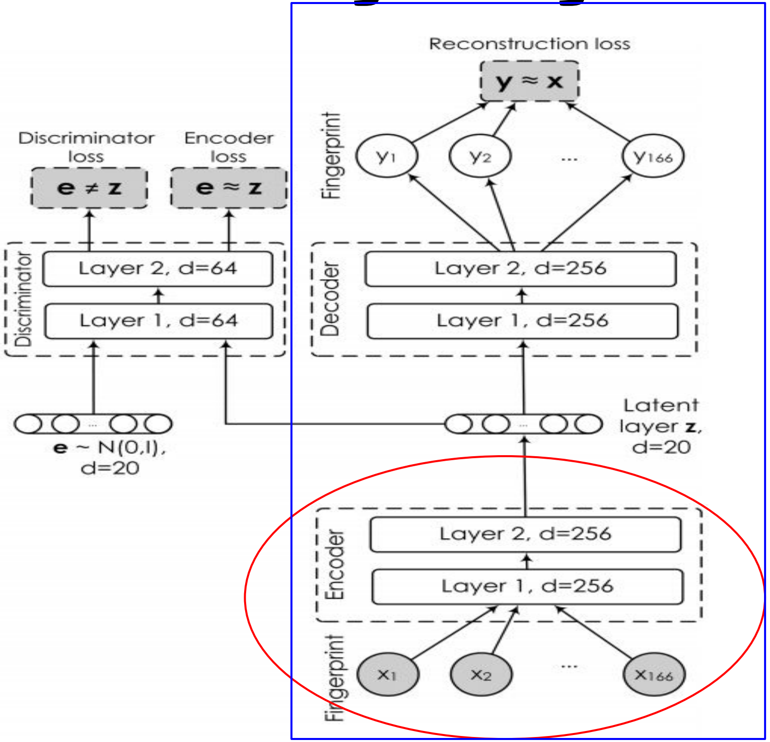


MolGAN: An implicit generative model for small molecular graphs, Nicola De Cao and Thomas Kipf (2018)

Applications of Generative models in Drug design

druGAN (GAN + Autoencoder = Adversarial Autoencoder)

- Consists of a generator and a discriminator
- The generator is an autoencoder
    - **encoder** + decoder

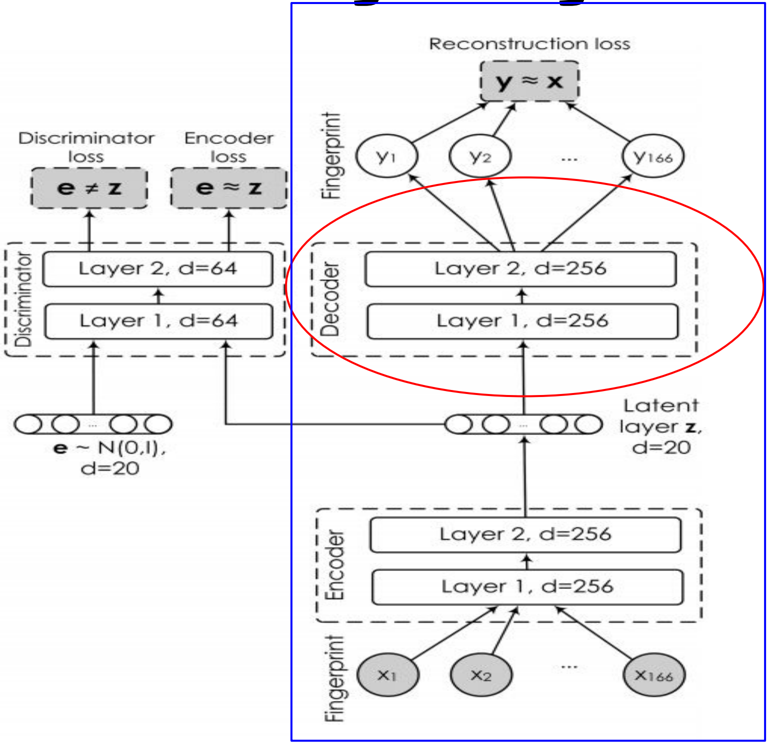


druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico

Applications of Generative models in Drug design

druGAN (GAN + Autoencoder = Adversarial Autoencoder)

- Consists of a generator and a discriminator
- The generator is an autoencoder
    - encoder + **decoder**



druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico

Applications of Generative models in Drug design

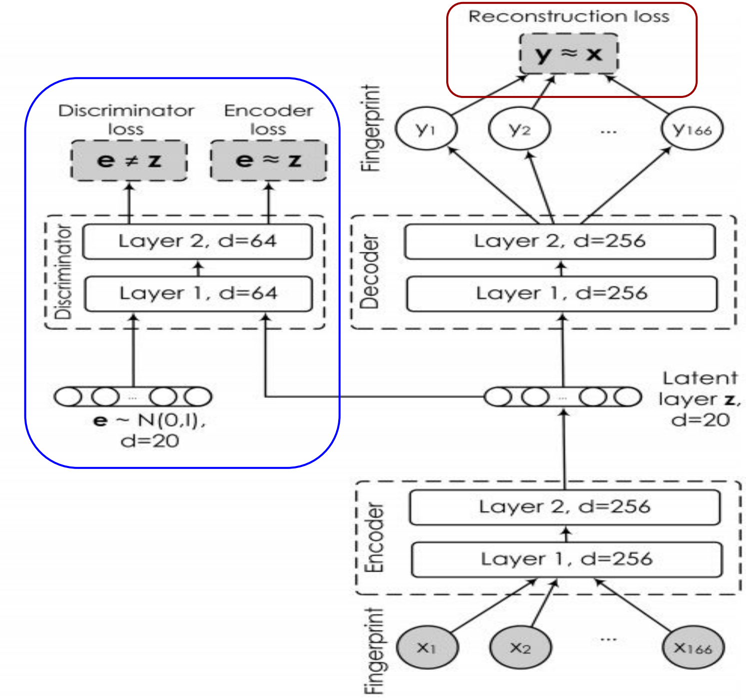
druGAN (GAN + Autoencoder = Adversarial Autoencoder)

- Consists of a generator and a discriminator
- The generator is an autoencoder (encoder + decoder)

Training Phases

- Reconstruction Phases
  - Train generator (encoder & decoder) to minimize reconstruction loss

- Regularization Phase
  - Train discriminator and generator's encoder with binary cross entropy loss



druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico

Lets go to the notebooks now