**Molecular Generation Using Convolutional Variational Autoencoders and Generative Adversarial Networks**

Contents

* Dataset
* Preprocessing
* Training
  + CVAE
  + GAN
  + Other Architectures
* Deployment
* Applications
* Research

Dataset

<https://www.nature.com/articles/s41597-022-01142-7>

Dataset consists of CSV file of molecules and their SMILES Representation, along with some other data.

Pre-Processing

* Rescales all molecules to standardized size
* Gets the width and height of all newly generated molecules
* Get 99th Percentile of these dimensions
* Standardizes the scale throughout images to fit in target size image
* Any in the top 1% of size are rotated through 45 Degrees as an approximation fix

Training

Hyper parameters – 24 EPOCHS, 48,000 Training Images, 64 Batch Size

* Convolutional Variational Auto Encoder
* Generative Adversarial Network

Deployment

Applications

Research

Process

* Preprocess RDKit Generated Images – Remove Glitched Images
* Encode Smiles Using Language Model/Vector/Bert
* Use Smiles Vector as Input to Variational Autoencoder, x\_train
* Train it off Using RDKit Generated Images as y\_train
* Different Perspectives
* Remove smiles truncating names
* Remove Molecules that Do not Generate Properly

Diffusion Models

Smile + Other Information (As Vector) -> Skeleton (Using VAE) (loss from RDKit Images)

Skeleton Image to Smile -> Trained off RDKit Images

Trained

Model 1 – Smile to Image Representation

Model 2 – Image to Smile Representation

Goal, Input a smile with some parameters like make it long and user is generated image of skeletons

Pipeline

SMILES Representation -> ChemBERTa or RDKit Pretrained Embedding -> Vector

Vector -> Auto-Encoder -> Skeleton (target is RDKit Generated Version) -> Conv Net -> SMILEs

<https://jcheminf.biomedcentral.com/articles/10.1186/s13321-018-0286-7>

Zinc Dataset

<https://arxiv.org/pdf/2209.01712.pdf>

CUDA – for optimisation