Generate Novel Molecules With Target Properties Using Conditional Generative Models

Abhinav Sagar[*]
Vellore Institute of Technology
Vellore, Tamil Nadu, India
abhinavsagar4@gmail.com

Abstract

Drug discovery using deep learning has attracted a lot of attention of late as it has obvious advantages like higher efficiency, less manual guessing and faster process time. In this paper, we present a novel neural network for generating small molecules similar to the ones in the training set. Our network consists of an encoder made up of bi-GRU layers for converting the input samples to a latent space, predictor for enhancing the capability of encoder made up of 1D-CNN layers and a decoder comprised of uni-GRU layers for reconstructing the samples from the latent space representation. Condition vector in latent space is used for generating molecules with the desired properties. We present the loss functions used for training our network, experimental details and property prediction metrics. Our network outperforms previous methods using Molecular weight, LogP and Quantitative Estimation of Drug-likeness as the evaluation metrics.

1 Introduction

Deep learning has achieved tremendous success in many areas tackling a range of datasets from images to text. One of those areas is Chemoinformatics. Neural networks have been used to solve a variety of problems like molecule property prediction, drug design, chemical reaction prediction etc. In this work, we propose a novel network for de novo molecular design using deep learning. Neural networks clearly offers a better approach as it speeds up the process while also increasing the efficiency compared to the traditional methods.

The total number of potential organic molecules is very large while the chemical space contains more than 10^{60} molecules (Virshup et al., 2013). Of these only some 10^{8} molecules are discovered, All traditional methods discovered new molecules using the chemical space of the molecules which were already discovered (Kim et al., 2016). A lot of newly discovered molecules were born out of hit and trial methods.

The main challenge is to generate new molecules with desired property (molecular weight, toxicity, solubility etc). Deep learning optimizes the computational overhead to search for new molecules. This approach is not only fast but also cheaper and more efficient. Also the whole chemical space can be utilized to search for potential small molecules. The process of de novo molecular design can be separated into the following parts: 1. molecular generation; 2. approach to rank molecules; 3. function to optimize the molecular space in search of new molecules.

[*]Website of author - https://abhinavsagar.github.io/

2 Related Work

Generative models primarily driven by GANs and VAEs have been used successfully for efficient molecular design. (Gómez-Bombarelli et al., 2018) used VAEs to optimize the molecular properties in latent space where molecules are expressed as a real vector. Since the latent space is continuous and differentiable, hence gradient based optimization can be done. (Blaschke et al., 2018) used adversarial autoencoder and bayesian optimization to generate molecules according to a specific property. (Kadurin et al., 2017) compared AAE and VAE using reconstruction error and variability of the output molecular fingerprints as the evaluation metrics.

(Segler et al., 2018) and (Gupta et al., 2018) used generative models using Natural Language Processing(NLP) techniques. The molecules are represented as SMILES string and the model learns the probability distribution of the next character of a given piece of SMILES. Transfer learning was used using a pre trained backbone. (Popova et al., 2018) and (Segler et al., 2018) used Recurrent Neural Networks, (Kusner et al., 2017) and (Dai et al., 2018) used Variational Autoencoders. The models directly generating the graph structures of molecules are used in (Simonovsky and Komodakis, 2018) and (Jin et al., 2018).

Lately conditional molecular design has been used to generate molecules with properties close to pre-determined target conditions. (Segler et al., 2018) used recursive fine tuning while (Gómez-Bombarelli et al., 2018) used bayesian optimization. Here the need is to find a latent representation which is close to the target condition. Since an additional optimization procedure is used, this method is inefficient especially when there are multiple target conditions.

Our main contributions can be summarized as follows:

• We present a novel network for generating new but similar molecules to the ones it is trained on.

• Our network can be divided into three parts: encoder which converts the training data into a latent space, predictor is used to enhance the function of encoder using another latent space conversion while the decoder is responsible for generating new molecules from the latent space.

• We present the loss functions, experimental details and evaluation metrics for property prediction.

• Our network outperforms previous methods on most of the metrics.

3 Background

3.1 Recurrent Neural Networks

Recurrent Neural Networks are used where data in sequential in nature ie there is a connectivity with previous terms. Generating SMILES sequences requires understanding of the sequence of characters. The outputs in case of RNN depends on previous computations thus creating a loop where information at every stage is influenced by previous stages. RNNs is represented mathematically as in Equation 1:

$$\begin{gathered} h_t = \sigma_h\left(U_h x_t + V_h h_{t-1} + b_h\right) \\ o_t = \sigma_y\left(W_y h_t + b_h\right) \end{gathered} \tag{1}$$

Where x_t denotes input vector $(m \times 1)$, h_t denotes hidden vector $(n \times 1)$, o_t denotes output vector $(n \times 1)$, b_h denotes bias vector $(n \times 1)$, U, W denotes parameter matrices $(n \times m)$, V denotes parameter matrix $(n \times n)$ and σ_h, σ_y denotes activation functions.

Thus RNNs form a chain of repeating units which are all linked together for making sense of sequential information.

3.2 Gated Recurrent Unit

RNNs are very difficult to train. A lot of factors make training a challenge of which the most common is the vanishing gradient problem. As the training proceeds, the value of gradient which is used to update the weights of the network becomes less. In the earlier layers, the value becomes infinitely small thus there is a memory loss. Gated Recurrent Unit were used to counter this as it has an additional forget gate to forget the information when it is no longer needed. Together with the input gate, forget gate is used to decide which information to forget and which is important to make predictions (Cho et al., 2014). GRUs is represented mathematically as in Equation 2:

$$\begin{aligned} z_t &= \sigma\left(W_z \cdot [h_{t-1}, x_t] + b_z\right) \\ r_t &= \sigma\left(W_r \cdot [h_{t-1}, x_t] + b_r\right) \\ \tilde{h}_t &= \tanh\left(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h\right) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \end{aligned} \tag{2}$$

Where h_t denotes hidden layer vectors, x_t denotes output vector, b_z, b_r, b_h denotes bias vector, W_z, W_r, W_h denotes parameter matrices and σ, tanh denotes activation functions.

3.3 1D-CNN

Although CNNs have been mostly used with image data, however it can also be used for text based data. The need is to convert the text into one hot encoded format in the form of matrix. Each row of the matrix represents a word or character. In case of images, data is represented in two dimensional form along with a two dimensional filter. However text data is in a 1D format along with a one dimensional kernel. There is also a need to use zero padding at places where character is null.

3.4 Variational Autoencoder

A vanilla autoencoder is good at reconstructing the samples given as input to encoder. However it lacks the ability to generate new samples using existing samples. This is where Variational Autoencoder comes to rescue. They are a kind of generative model which takes a standard normal distribution as input to encoder and decoder generates new samples from the latent space (Kingma and Welling, 2013). The objective function of vanilla VAEs is defined in Equation 3:

$$E[\log P(X \mid z)] - D_{KL}[Q(z \mid X)||P(z)] \tag{3}$$

where E denotes expectation value, P, Q denotes probability distributions, D_{KL} denotes Kullback-Leibler divergence, X denotes data and z denotes latent spaces.

3.5 Conditional Variational Autoencoder

The decoder of VAE is not suitable since the goal is to design drugs with the desired property. However CVAE solves this problem using labels along with the input sample to encoder. The latent vector contains information about sampled data and hence the decoder is able to synthesize new samples with the desired property. Condition vector c is concatenated and is used for both the encoder and decoder (Sohn et al., 2015). The objective function of CVAE is changed accordingly and is defined in Equation 4:

$$E[\log P(X \mid z, c)] - D_{KL}Q(z \mid X, c)||P(z \mid c)] \tag{4}$$

where E denotes expectation value, P, Q denotes probability distributions, D_{KL} denotes Kullback-Leibler divergence, X denotes data, z denotes latent spaces and c denotes condition vector.

4 Method

4.1 Dataset

A sample of 100,0000 SMILES strings of drug like molecules was randomly sampled from ZINC database. We use 90,000 molecules for training and 10,000 molecules for testing the property prediction performance. A special sequence indicating end of sequence is appended at the end of every sequence. To evaluate the performance of our network, we used three properties molecular weight (MolWt), Wildman Crippen partition coefficient (LogP) and quantitative estimation of drug-likeness (QED).

4.2 Molecular Representation

A lot of molecular representations have been used in literature. The most common among them are the SMILES (Weininger, 1988) and the Graph representation. The more detailed the representation is, the more computational burden it demands. Simplified Molecular Input Line Entry System (SMILES) is a one dimensional representation of a two dimensional chemical drawing. It contains atoms and bond symbols with an easy vocabulary.

Since it is easy to understand and parse, hence Natural Language Processing (NLP) techniques can be used on them. More than one SMILES representation of a molecule is possible, however only one canonical form is used per molecule. The molecular latent space visualized in two dimensions using principal component analysis is shown in Figure 1:

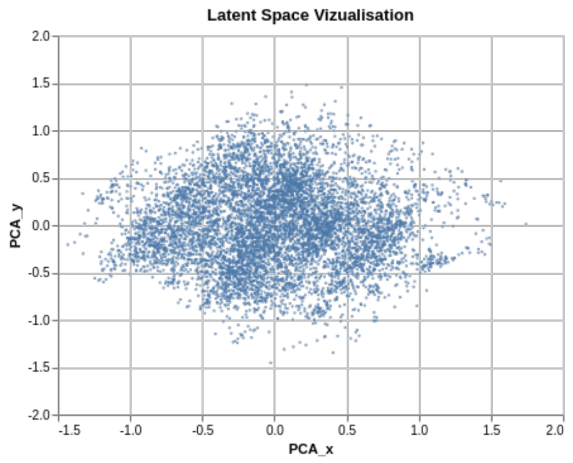

Figure 1: 2D visualization of around 8000 molecules encoded into the latent space.

The results are visualized using RDKit Python package.

4.3 Benchmarks

The following benchmarks was used to determine the performance of our network for generating molecules:

1. Validity: It assesses whether the molecules generated are realistic or not. Examples of not valid molecules are one with wrong valency configuration or wrong SMILES syntax.

2. Uniqueness: It assesses whether the molecules generated are different from one another or not.

3. Novelty: It assesses whether the molecules generated are different from the ones in the training set or not.

4.4 Network Architecture

Our network was trained on SMILES from ChEMBL database. Since the context is in small molecular generation, hence only SMILES string with less than 120 characters were used. The data was divided into 80% training data and 20% testing data. Bayesian optimization was done to optimize the hyper-parameters like number of hidden layers, activation functions, learning rate etc.

Our model is comprised of an encoder, predictor and decoder networks. The encoder in our network has three bi-GRU layers, flatten layer and a dense layer. The predictor is comprised of a dense layer and three 1D convolutional layers. The latent space dimensions were set to 292. The encoder was fed data from SMILES database after one hot encoding. The encoded data is sampled in the latent space using mean and standard deviation vectors. The latent vector produces new samples after passing through the decoder. The decoder has three uni-GRU layers followed by a dense and flatten layer.

The input variable x is generated from a generative distribution $p_\theta(x|y,z)$, which is conditioned on the output variable y and latent variable z. The prior distributions are denoted by $p(y) = N(y|\sigma y, \sum y)$ and $p(z) = N(z|\theta, I)$. In our case, x denotes molecules while y denotes properties. Standard deviation is used on both y and z terms before passing through the decoder. Our network is shown in Figure 2:

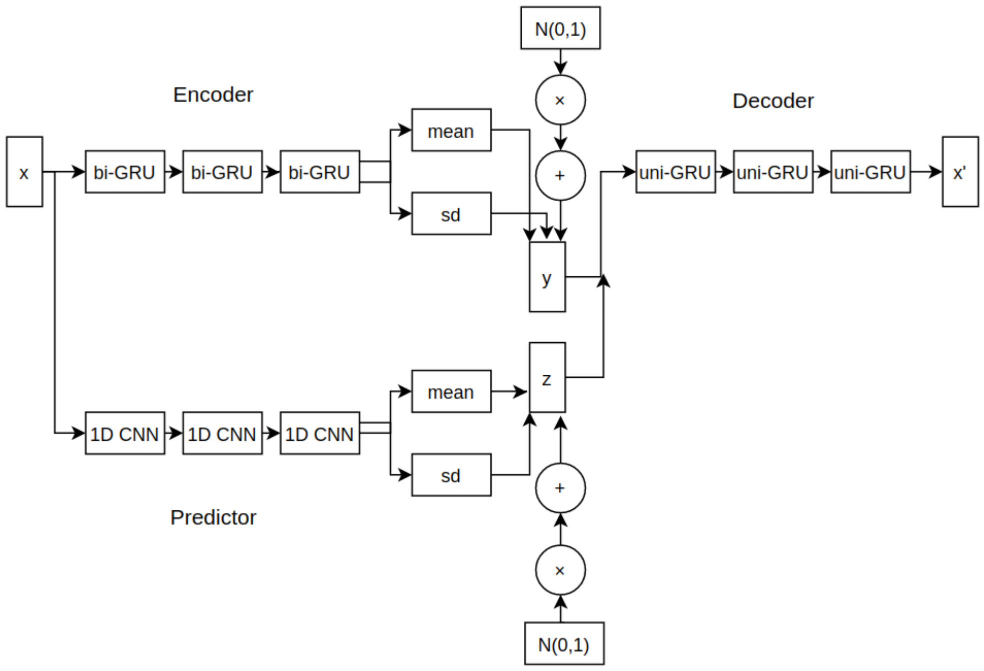

Figure 2: Illustration of our network architecture

4.5 Loss Functions

After the network is trained, property prediction is done using prediction network $q_\phi(y|x)$. The unlabeled instance is represented by x, the corresponding properties $\hat{y}$ are predicted as defined in Equation 5:

$$\hat{\mathbf{y}} \sim \mathcal{N}\left(\boldsymbol{\mu}_\phi(\mathbf{x}), \operatorname{diag}\left(\boldsymbol{\sigma}_\phi^2(\mathbf{x})\right)\right) \tag{5}$$

The point estimate $\hat{y}$ can be obtained by maximizing the probability which is represented by $\mu_\phi(x)$.

The decoder network $p_\theta(x|y,z)$ is used to generate a molecule. A molecule representation $\hat{x}$ is obtained from y and z as defined in Equation 6:

$$\hat{\mathbf{x}} = \arg\max \log p_\theta(\mathbf{x} \mid \mathbf{y}, \mathbf{z}) \tag{6}$$

Let a sequence S of symbols s_i at steps $t_i \in T$, our language model assigns a probability as defined in Equation 7:

$$P_\theta(S) = P_\theta(s_1) \cdot \prod_{t=2}^{T} P_\theta(s_t \mid s_{t-1}, \ldots, s_1) \tag{7}$$

where the parameters θ are learned from the training set. We use a GRU to estimate the probabilities of Equation 7.

The probability distribution $P_\theta(s_{t+1}|s_t, ..., s_1)$ of the next symbol given the already seen sequence is thus estimated using the output vector y_t of the recurrent neural network at time step t as defined in Equation 8:

$$P_\theta\left(s_{t+1}=k \mid s_t, \ldots, s_1\right)=\frac{\exp \left(y_t^k\right)}{\sum_{k^{\prime}=1}^K \exp \left(y_t^{k^{\prime}}\right)} \tag{8}$$

where y_t^k corresponds to the k^{th} element of vector y_t. Novel molecules can be generated by sampling from this distribution. This sampling procedure is repeated until the desired number of characters has been generated.

4.6 Experimental Details

An open source package named RDKit was used for testing the validity of the generated SMILES strings and calculating the properties of the molecules. Samples of dataset is drawn from ZINC database (Irwin et al., 2012). Learning rate in our experiments was set to 0.0001 with exponential decay of 0.99. The model was trained until it converged. The condition for generated molecules to be successful is if the target property of generated molecules was within 10% range of given target value. The molecules are encoded to the latent representation and gaussian noise is added to it.

The standard deviation value was important to tune as a lower value generated molecules similar to the ones in the training set. On the other hand using a larger standard deviation generated molecules very different from the ones it was trained on. The optimal value of standard deviation was found to be 0.05. During training, we normalize each output variable to have a mean of 0 and standard deviation of 1. Batch size value of 50 was used along with ADAM as the optimizer.

The property prediction performance is evaluated using mean absolute error (MAE) on the test set. The encoder, predictor and decoder networks consist of three hidden layers each having 50 gated recurrent units (GRU). The target values for MolWt, LogP, and QED are set as (250, 350, 450), (1.5, 3.0, 4.5), and (0.5, 0.7, 0.9), respectively.

4.7 Evaluation Metrics

Tanimoto Similarity: Several similarity and distance metrics are used for quantifying the distance between two molecules. The most common among them is Tanimoto similarity which has a value between 0 and 1. Measuring the similarity between two molecules is very important as new molecules are designed using the data of drugs which are already present. Designing drugs with new property but similar profile requires changing the previous structure nominally. Tanimoto Similarity metric is defined in Equation 9:

$$d(\mathbf{a}, \mathbf{b})=\frac{n(A \cap B)}{n(A)+n(B)-(A \cap B)} \tag{9}$$

where $n(A)$ denotes number of bits set to 1 in molecule $a's$ fingerprint, $n(B)$ denotes number of bits set to 1 in molecule $b's$ fingerprint, $(A \cap B)$ denotes number of bits set to 1 that molecule a and molecule b have in common.

4.8 Property Prediction

The fraction of invalid molecules using our network was less than 1%. The fraction of unique molecules generated is 90.2%. The average and standard deviation values are reported in Table 1 using MAE as the evaluation metric with the varying fractions of labeled molecules. Our network outperforms others in most of the cases.

An important tool for evaluating the performance of network is done using properties distribution. The following three properties are used:

1. Molecular weight (MW): It is the sum of atomic weights in a molecule. To figure out if the generated samples are biased towards lighter or heavier molecules histograms of molecular weight for the generated and test sets are plotted.

2. LogP: It is the ratio of a chemical's concentration in the octanol phase to its concentration in the aqueous phase.

3. Quantitative Estimation of Drug-likeness (QED): It is a measure of how likely a molecule is a viable candidate for a drug. It's value lies between 0 and 1 both included.

5 Results

The property prediction performance with varying fractions of labeled molecules compared with networks: ECFP (Rogers and Hahn, 2010), GraphConv (Kearnes et al., 2016), VAE (Gómez-Bombarelli et al., 2018) and SSVAE (Kang and Cho, 2018) is shown in Table 1:

Table 1: Property prediction performance with varying fractions of labeled molecules.

frac(%)	property	ECFP	GraphConv	VAE	SSVAE	Ours
5	MolWt	17.713±0.396	6.723±2.116	3.463±0.971	1.639±0.577	1.215 ± 0.383
5	LogP	0.380±0.009	0.187±0.015	0.125±0.013	0.120±0.006	0.135 ± 0.005
5	QED	0.053±0.001	0.034±0.004	0.029±0.002	0.028±0.001	0.014 ± 0.004
10	MolWt	15.057±0.358	5.255±0.767	2.464±0.581	1.444±0.618	1.104 ± 0.279
10	LogP	0.335±0.005	0.148±0.016	0.097±0.008	0.090±0.004	0.086 ± 0.012
10	QED	0.045±0.001	0.028±0.003	0.021±0.002	0.021±0.001	0.023 ± 0.002
20	MolWt	12.047±0.168	4.597±0.419	1.748±0.266	1.008±0.370	0.796 ± 0.105
20	LogP	0.249±0.004	0.112±0.015	0.074±0.006	0.071±0.007	0.068 ± 0.010
20	QED	0.033±0.001	0.021±0.002	0.015±0.001	0.016±0.001	0.016 ± 0.001
50	MolWt	9.012±0.184	4.506±0.279	1.350±0.319	1.050±0.164	0.994 ± 0.081
50	LogP	0.180±0.003	0.086±0.012	0.049±0.008	0.047±0.003	0.058 ± 0.011
50	QED	0.023±0.000	0.018±0.001	0.009±0.002	0.010±0.001	0.009 ± 0.001

Sample of molecules generated using standard deviation value of 0.05 is presented in Figure 3:

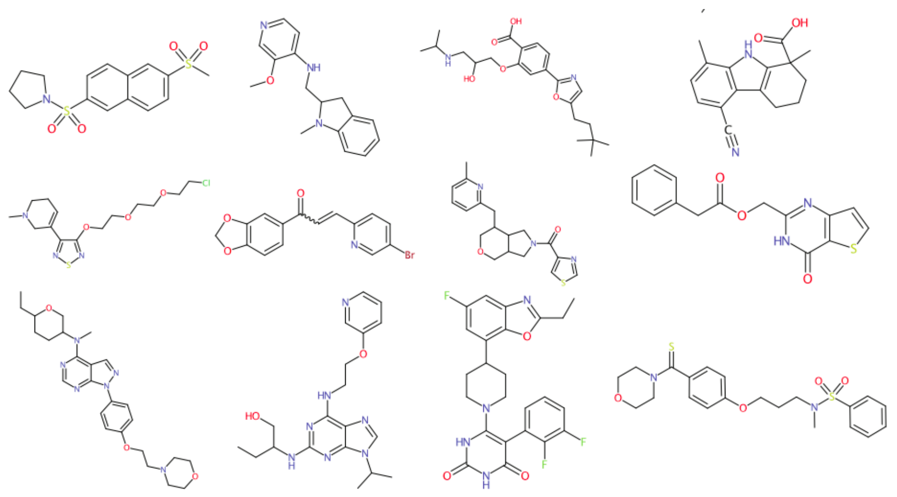

Figure 3: A few randomly selected, generated molecules. Sampled molecules using different signature and standard deviation of 0.05

We demonstrate that our network can generate with target properties for Aspirin and Tamiflu. The condition vector was made up of custom values for target properties. The molecules generated by our network for Aspirin is shown in Figure 4:

The molecules generated are considerably different from original molecules since the latent vectors were chosen randomly. The molecules generated by our network for Tamiflu is shown in Figure 5:

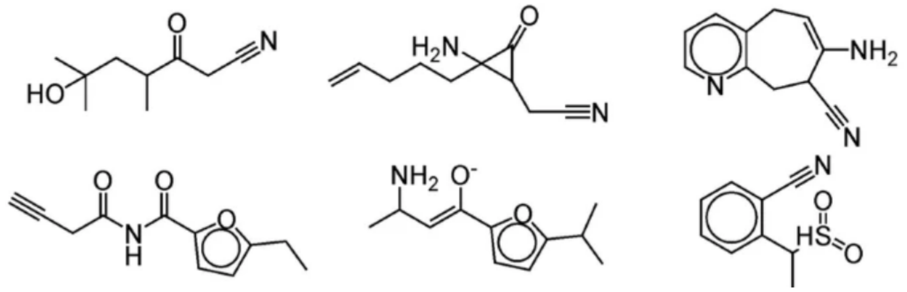
Figure 4: Molecules generated by our network with the condition vector made of the five target properties of Aspirin

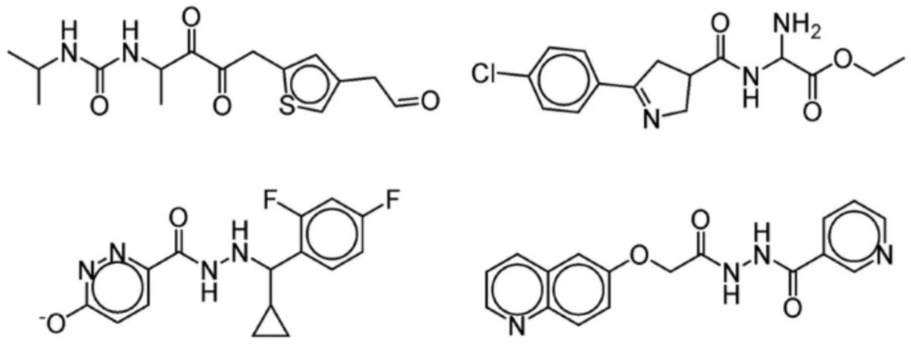
Figure 5: Molecules generated by our network with the condition vector made of the five target properties of Tamiflu

5.1 Interpolation

The generative models like VAE and CVAE are able to learn smooth latent representation of input data and perform interpolations on them. For carrying out the interpolations, starting and end points are needed. Both of these points should represent a molecule in chemical space. The interpolation samples between Aspirin and Paracetamol is shown in Figure 6:

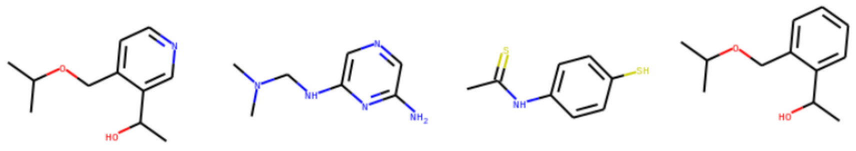
Figure 6: Generated samples using interpolation.

6 Conclusions

Drug discovery or new molecule generation has garnered a lot of attention from the deep learning community. Since it a very important but challenging problem in Cheminformatics, hence a lot of work has been done using a variety of neural networks. In this paper, we propose a novel network for generating similar but new molecules to the one it has been trained on. The network is made up of three parts: encoder, predictor and decoder networks. We present the loss functions, molecular representation and experimental details. Using Molecular weight, LogP and Quantitative Estimation of Drug-likeness as the property prediction metrics, our network performs better than previous state of the art approaches.

Acknowledgments

We would like to thank Nvidia for providing the GPUs for this work.

References

E. J. Bjerrum and B. Sattarov. Improving chemical autoencoder latent space and molecular de novo generation diversity with heteroencoders. *Biomolecules*, 8(4):131, 2018.

T. Blaschke, M. Olivecrona, O. Engkvist, J. Bajorath, and H. Chen. Application of generative autoencoder in de novo molecular design. *Molecular informatics*, 37(1-2):1700123, 2018.

H. Chen, O. Engkvist, Y. Wang, M. Olivecrona, and T. Blaschke. The rise of deep learning in drug discovery. *Drug discovery today*, 23(6):1241–1250, 2018.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

H. Dai, Y. Tian, B. Dai, S. Skiena, and L. Song. Syntax-directed variational autoencoder for structured data. *arXiv preprint arXiv:1802.08786*, 2018.

N. De Cao and T. Kipf. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

E. Gawehn, J. A. Hiss, and G. Schneider. Deep learning in drug discovery. *Molecular informatics*, 35(1):3–14, 2016.

G. B. Goh, C. Siegel, A. Vishnu, N. O. Hodas, and N. Baker. Chemception: a deep neural network with minimal chemistry knowledge matches the performance of expert-developed qsar/qspr models. *arXiv preprint arXiv:1706.06689*, 2017.

R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

A. Gupta, A. T. Müller, B. J. Huisman, J. A. Fuchs, P. Schneider, and G. Schneider. Generative recurrent networks for de novo drug design. *Molecular informatics*, 37(1-2):1700111, 2018.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.

W. Jin, R. Barzilay, and T. Jaakkola. Junction tree variational autoencoder for molecular graph generation. *arXiv preprint arXiv:1802.04364*, 2018.

A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper, and A. Zhavoronkov. drugan: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Molecular pharmaceutics*, 14(9):3098–3104, 2017.

S. Kang and K. Cho. Conditional molecular design with deep generative models. *Journal of chemical information and modeling*, 59(1):43–52, 2018.

S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30(8):595–608, 2016.

S. Kim, P. A. Thiessen, E. E. Bolton, J. Chen, G. Fu, A. Gindulyte, L. Han, J. He, S. He, B. A. Shoemaker, et al. Pubchem substance and compound databases. *Nucleic acids research*, 44(D1): D1202–D1213, 2016.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

M. J. Kusner, B. Paige, and J. M. Hernández-Lobato. Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*, 2017.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

J. Lim, S. Ryu, J. W. Kim, and W. Y. Kim. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *Journal of cheminformatics*, 10(1):1–9, 2018.

M. Popova, O. Isayev, and A. Tropsha. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7):eaap7885, 2018.

K. Preuer, P. Renz, T. Unterthiner, S. Hochreiter, and G. Klambauer. Fréchet chemnet distance: a metric for generative models for molecules in drug discovery. *Journal of chemical information and modeling*, 58(9):1736–1741, 2018.

K. Preuer, G. Klambauer, F. Rippmann, S. Hochreiter, and T. Unterthiner. Interpretable deep learning in drug discovery. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 331–345. Springer, 2019.

O. Prykhodko, S. V. Johansson, P.-C. Kotsias, J. Arús-Pous, E. J. Bjerrum, O. Engkvist, and H. Chen. A de novo molecular generation method using latent vector based generative adversarial network. *Journal of Cheminformatics*, 11(1):74, 2019.

D. Rogers and M. Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

B. Sanchez-Lengeling and A. Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018.

M. H. Segler, T. Kogej, C. Tyrchan, and M. P. Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1):120–131, 2018.

M. Simonovsky and N. Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *International Conference on Artificial Neural Networks*, pages 412–422. Springer, 2018.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015.

A. M. Virshup, J. Contreras-García, P. Wipf, W. Yang, and D. N. Beratan. Stochastic voyages into uncharted chemical space produce a representative library of all possible drug-like compounds. *Journal of the American Chemical Society*, 135(19):7296–7303, 2013.

J. Wang, H. Cao, J. Z. Zhang, and Y. Qi. Computational protein design with deep learning neural networks. *Scientific reports*, 8(1):1–9, 2018.

D. Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

D. Xue, Y. Gong, Z. Yang, G. Chuai, S. Qu, A. Shen, J. Yu, and Q. Liu. Advances and challenges in deep generative models for de novo molecule generation. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 9(3):e1395, 2019.

X. Yang, J. Zhang, K. Yoshizoe, K. Terayama, and K. Tsuda. Chemts: an efficient python library for de novo molecular generation. *Science and technology of advanced materials*, 18(1):972–976, 2017.