Applications of Autoencoders along with Deep Learning Techniques to generate valid molecules

Sesha Sai Aparna T, Anuradha T

Department of Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, AP, India

aparnatatavarthi39@gmail.com,atadiparty@gmail.com

Abstract. From the moment of identifying the fundamental cause of an illness to its availability in the marketplace, it takes an average of 10 years and almost $2.6 billion dollars to develop a medication. We're actually hunting for a needle in a haystack, which takes a lot of time, effort, and money. In a solution space of between $10^{30} and $10^{100}$ synthetically viable compounds, we're seeking for the one molecule that can turn off a disease at the molecular level. The chemical solution space is just too large to adequately screen for the desired molecule. Only a small percentage of the synthetically viable compounds for wet lab research are stored in pharmaceutical chemical repositories. Computational de novo drug design can be used to explore this vast chemical space and develop previously undesigned compounds. Computational drug design can cut the amount of time spent in the discovery phase in half, resulting in a shorter time to market and lower drug prices. Deep learning and artificial intelligence (AI) have opened up new perspectives in cheminformatics, especially in molecules generative models. Recurrent neural networks (RNNs) trained with molecules in the SMILES text format, in particular, are very good at exploring the chemical space. Two baseline models were created for generating molecules, one of the model includes an encoder that takes SMILES as input and then develops a deep generative LSTM model which acts as a hidden layer and the output from layers acts as an input to the decoder. The other baseline model acts the same as the above-mentioned model but it includes latent space, it is simply a representation of compressed data that bring related data points closer together physically. To learn data properties and find simpler data representations for analysis, and weights which are obtained from the previous model to generate more efficient molecules. Then created a custom function to play with the temperature of the softmax activation function which creates a threshold value for the valid molecules to generate. This model enables us to produce new molecules through successful exploration.

Keywords: Autoencoders, Long Short Term Memory(LSTM), Recurrent Neural Network(RNN), Simplified Molecular Input Entry System(SMILES), RDKIT, Automatic Molecules Generation

1. Introduction
Apart from virtual screening, computational de novo molecular design is an important discipline of drug design that has made significant contributions to drug discovery. In the last two to three decades, many classic structure-based de novo drug design methods$^{[1]}$, including our LigBuilder methods, have been created to build novel molecular entities with desirable chemical and pharmacological properties$^{[2]}$. Deep generative models have recently been widely used and have demonstrated exceptional achievements in a variety of areas, including realistic musical improvisation, changing
facial emotions in photographs, making realistic-looking artworks, and translating between source and target images. In terms of representing and creating data in continuous domains, generative models are at the top of their game. There has been a growing interest in constructing generative models for generating realistic and valid data for more complicated and discrete data types, such as mathematical expressions, source code, and chemical compounds. The development of deep generative models has created a slew of potential solutions to address the problem of molecule generation, illuminating a promising new direction in de novo drug design. Simplified molecular input line entry specification (SMILES) is a string-based representation of chemical compounds obtained from molecular graphs commonly used. RNN-based generative models with one-hot encoding have become popular because recurrent neural networks (RNNs) are suitable candidates for these representations. With recent advances in the fields of deep learning on molecular graphs, training deep generative models directly on graphics has become a viable option. Machine learning is being used to create synthetically logical molecules with increasing success. A complete system capable of both generating valid molecules and optimizing multiple traits, on the other hand, has remained unknown. Molecules were encoded into a continuous vector space using autoencoders, allowing for rapid optimization. Encoding essentially discrete molecules into continuous space, on the other hand, poses some challenges. Molecules produced synthetically are often irrational. While evolutionary algorithms struggle to generate valid molecules, they do show promise in optimization. A variety of evolutionary selection mechanisms have shown promise in drug production as well as other multiobjective optimization problems. Through a method focused on natural language processing, recurrent neural networks have been effective in producing rational molecules. The SMILES then used to encode molecules as strings. After that, the recurrent neural network is trained to predict the next SMILES character from a collection of previous characters.

In order to avoid these circumstances, in this work the model has been trained using autoencoders where LSTM is added as an hidden cell so that it can successfully identify to solve sequence related tasks for example, in the fields of natural language processing, text, voice, and recently into the field of molecular informatics. The outputs from the hidden layers are used as an inputs the decoder model and obtain generated smiles with weights.

2. Methodology
The following steps provides the methodology for the model to generate molecules.

2.1. Data Preprocessing
The European Bioinformatics Institute’s open-source ChEMBL collection of drug-like compounds was used to create a training dataset of 439,098 molecules. The SMILES string notation was used to represent molecules for easy interpretation by the recurrent neural network model we used. SMILES uses characters to represent atoms, bonds, and chemical structures, shown in Figure 1, and was developed with linguistic consistency and machine friendliness in mind. Aromatic and aliphatic carbon atoms, for example, are denoted by the symbols c and C. The symbols -, =, and # signify single, double, and triple bonds, respectively. Branching is denoted by parentheses, while rings are marked by digits directly after the atoms where the ring is closed. And also, ‘!’ and ‘E’ were appended to the end of the characters for each molecule yielding the character set into 45 unique characters. To be digitally processed by the neural network, the SMILES character strings must be vectorized into one-hot encoded arrays of 1s and 0s. A character set is created from all of the characters in the SMILES strings to prepare for this. The character set is then utilized to generate two Python dictionaries that help in character-to-number translation. These SMILES were given as a train data to generate valid molecules.
Figure 1. Simplified Molecular Input Line Entry Specification (SMILES) representation of different molecules after converting from molecular graphs existing in the database, basically the data consists in database will be in graphical structures then converted into smiles structures which can be easily understand by Recurrent Neural Network.

2.2. RDKIT - An open source cheminformatics package
RDKIT (Research and Development Kit) It is a series of C++ and Python-based cheminformatics and machine-learning applications. The BSD licence is a business-friendly open source licence. C++ data structures and algorithms are fundamental. It performs 2D and 3D molecule operations and it generates Machine Learning descriptors, basically it holds all the required packages for cheminformatics which makes a job of mankind easy with its libraries.

2.3. Design of Methodology
The neural network will be a sequential model for learning to copy molecules character by character. This model uses an encoder/decoder architecture. The encoder model limits the size of the rendering space to generate a compressed representation of the attributes of the input object. Then, the decoder model learns to generate the output of this compressed representation. Since these molecules are sequences of symbols, the AI model is generated using short-term memory units (LSTM). LSTM is a type of recurrent neural network (RNN) unit, used to solve many related AI problem sequences. Examples include natural language processing, language and text translation, music creation, and more recently the field of molecular computing.

The SMILES input data must be converted into a supervised learning-friendly format. Each SMILES sequence is transformed into a sequence of one-hot encoded vectors using a function that adds the starting (!) and ending character (E) markers, padding for constant sequence length, and transforms each SMILES sequence into a sequence of one-hot encoded vectors. Your input data should be a 3D tensor of shape for LSTM neural networks (number of samples, number of time steps, number of features). We’ve set a time step value of 100 in this case. Our molecular sequences have a maximum input and output length of 100 characters. All input data will be 100 characters long during training, with the ‘E’ character serving as padding from the end of the SMILES molecule to 100 characters. The number of features is equal to the number of characters in our character set, which is 45.
3. Proposed Method

In this work, proposed two baseline models, one of the model having LSTM as a processing layer where the input comes from encoder (Figure 2.). The encoder model uses a single LSTM layer to extract hidden and cell states based on the input tensor. It's worth noting that the LSTM layer's outputs are ignored, and only the state is captured. The initial context for the decoder model will be provided by this state. The states are then concatenated and fed into a dense layer, which returns a compressed version of our original input. The hidden and cell states of the LSTM are extracted from the compressed data by the decoder model. Those states are then fed into an LSTM processing layer as input conditioning. Finally, we generate the model output using a dense softmax layer. The Adam optimizer was used to create the model, and our loss function was categorical cross-entropy nothing but a softmax activation function. Python generators were used to feed the training data to the model in batches due to the large amount of data.

**Figure 2.** This Standalone baseline model architecture generally generates molecules. The one hot encoder acts as an input to the LSTM which had hidden and cell states. Then the network begins to start generating molecules till it reaches the end character ‘E’.

The general concept has now been split into individual elements that will be developed to generate a generative model. The primary model design consisted of two connected neural network models: an encoder and a decoder, as you may recall. We split the two-part baseline model apart and add a third, intermediary model now that they've been trained together. From our basic model, we first develop a fully featured encoder model. After that, we design an intermediate model that can decode the latent space created by our encoder model into the LSTM states that our decoder model requires as input. Latent space here (as shown in Figure 3.) is just a representation of compressed input in which it checks for the similar points in the closer space and from it learns the features of data for finding simple representations. A new input layer is defined for this model, which corresponds to the encoder’s
latent space output. The LSTM hidden and cell states are obtained using the layers of the base model. The weights from the previously trained model can be retained with this method. The independent decoder model should now be created.

**Figure 3.** This two-part Intermediary model having latent space which is partial representation of compressed input from LSTM which is treated as Encoder in which related data points are closer together. For learning data attributes and creating simpler data representation for analysis, Latent space is important and here this will be input for LSTM processing layer.

Unlike the base model, which was trained in a stateless batch mode, the decoder model was trained in a stateful batch mode, allowing us to forecast one SMILES character at a time. With the exception of a new input batch shape, the neural network model layers are defined precisely as previously (which is required for stateful LSTM model). This model predicts one SMILES character at a time, where the input layer shape is as follows samples=1,time steps=1,features=45.

**Table 1.** Molecules Generated during Training at different epochs

| Iteration | Generated Example Molecules                                                                 | Valid |
|-----------|-------------------------------------------------------------------------------------------|-------|
| 0         | Cc1ccc(Oc2ccccc(-c3nc4cc(C(N)=O)ccc4[nH]3)ccc2)ccc1F                                     | True  |
| 100       | CON(C)NC(=O)c1ccc([N+][=O][O])c2cnnHc(=O)c(=C(C)Nc3ccc(C[N+][C(C)c4cccc(Cl)c4])c3)c(O)c12 | True  |
| 225       | OCC1(c2ccccc2NC2CCC(CC3C(c4ccco4)Cc4cccc34)C2)CC1                                        | True  |

The generated molecules (Table 1.) were trained at different epochs and the molecules were valid.
4. Results and Discussion
The result shows the generation of smiles characters. In Standalone baseline model (as shown in Figure 2.) The data has been split into 70% of training and 30% as testing. The increase of epochs will lead to the loss of data. Hence, the number of epochs trained were 225, yielding 99.79% as a training accuracy and 99.65% as a validation accuracy and the total model achieved 99.63% test accuracy and also the second baseline model as an intermediate model where the data has been split into 70% as training and 30 % as testing data. With the help of Autoencoder and also the latent space was incorporated in between, and managed to run 30 epochs and achieved 99.8% of accuracy as discussed on Table 2. In order to produce new molecules from the latent space created by the encoder, the use of previously generated model to calculate the LSTM hidden and cell states and then utilize these for generative model input. In inference mode, one input character is entered in a generative model and the next character is tested repeatedly until a final character is met. Importing the trial dataset and run it through the encoder model to produce latent space in order to validate the ability to produce molecules and also constructing a custom function for producing new compounds that allows to experiment with the temperature of the Softmax activation function throughout the molecule generation. The model becomes more cautious with its samples as the temperature is reduced from 1 to a lower value (e.g. 0.5).

![Model Accuracy and Loss Graphs](image1.png)

**Figure 4.** Accuracy representing the baseline model and acquires about 99.6% which had epochs for 225 to generate valid molecules which includes both training and testing data

![Model Accuracy and Loss Graphs](image2.png)

**Figure 5.** Accuracy of Intermediary model shown in graphical way showing that it is slightly higher than the baseline model it classifies the valid model based on the custom function
A larger temperature (> 1), on the other hand, will enhance both the uniqueness of our samples and the amount of defective/invalid molecules created. This may be thought of as injecting different degrees of mutation. The created and tested molecules are then compared at different sampling temperatures. By playing with sampling temperature we got to know that, if the temperature >1, 9% of molecules which are wrongly formatted, where the temperature at 1.25 the wrongly formatted molecules has increased to 31%. The number of invalid molecules produced when temperature increases, as expected.

This model is achieving the maximum accuracy in 30 epochs compared to other models which reduces the cost of computation less and time investing in generating molecules also a bit low. Then generating a random Softmax temperature varying from 0.75 to 1.25 and tries to generate up to 9854 valid molecules around each seed in the vector space where the loop for molecule generation is executed for 3 times and saved for virtual screening.

Table 2. Comparison of methods in generating molecules with different techniques

| Technique | Purpose | Dataset | Generative Performance | Author |
|-----------|---------|---------|------------------------|--------|
| Recurrent Neural Network with Non-dominated Sorting Algorithm | Molecule Generation for Denovo Drug Design | ChemBL data with 500000 molecules | Accuracy-86% Valid-9417 Invalid-6295 | Yasonik J[1] |
| LSTM RNN Encoder (Standalone Baseline Model) | Automatic Generation of molecules | QM9 data with 133885 molecules | Accuracy-99.63% Valid-4915 Epochs-225 25% Wrongly formatted molecules | Sesa Sai Aparna T, Anuradha T |
| LSTM RNN Autoencoder with Latent Space Representation using Custom Function (Intermediary model) | Automatic generation of molecules | ChemBL data with 439098 molecules | Accuracy-99.8% Valid-9854 Epochs-30 31% Wrongly formatted molecules | Sesa Sai Aparna T, Anuradha T |

The above mentioned method (shown in Table 2.) proposed a Multiobjective Denovo drug design which includes Recurrent Neural Network to generate molecules and non-dominated sorting algorithm to select best molecules out of generated molecules[1]. This method achieved 77% of valid molecules whereas, 6,295 are invalid and 9,415 are valid molecules and unique and also doesn’t exist in the training set. After applying, transfer learning and optimization with five iterations 86% of molecules (shown in Table 2.) are valid. The drawback occurred from this method is from sorting algorithm where it classifies some unworthy molecules as worthy ones. This method suggested to use encodings and decodings to achieve good results.

Table 3. A summary of generative models based on simplified molecular input line entry

| Method | Dataset | Molecule Size | No of Molecules | Accuracy |
|--------|---------|---------------|----------------|----------|
| Grammar VAE | ZINC | <39 heavy atoms | 250000 | 53.7 |
| SD-VAE | CheMBL | <39 heavy atoms | 250000 | 76.2% |
| RNN- Only based model | CheMBL | 34 to 74 characters | 541555 | 93% |
| RNN based model with RL | CheMBL | 10 to 50 characters | 1.5 million | 94% |
The above mentioned (Table 3) tells about different techniques includes Reinforcement Learning, Transfer Learning etc., used to generate different valid molecules taken different datasets.

Figure 6. SMILES generated from decoder once the process reaches end

Figure 7. SMILES generated under predefined conditions
5. Conclusion:
In this work, a new baseline models were proposed in order to generate molecules for denovo drug design based on text-based molecules which are also called as structural based molecules, we basically used encoding and decoding as a main theme using LSTM RNN and generated a good amount of molecules. The methods are trained using ChemBL and QM9 datasets which performed better and also given better performance compared to other SMILES methods mentioned in (Table 4.). The second baseline method called as Intermediary method uses latent space representation and that output is used for the input of Decoding part LSTM processing layer and also introduced a custom function which works with temperature and makes a threshold value for molecules to fall in.
This work completely ignores stereochemistry, though it is useful for drug design. So this work needs to extend to bring molecules properties having stereo chemistry. There are several ways for this approach to molecular design to be further improved. The increase in generation of molecules are further examined in our future work by incorporating any learning method and also providing the properties to that how a molecule must be. We used a text-based molecular encoding in this work, but there would be many benefits of using a graph-based autoencoder. The learning problem is excessively difficult to compel the decoder to generate valid SMILES strings, because the decoder must also implicitly learn which strings SMILES are valid. An autoencoder that outputs molecular graphs directly is desirable as it could resolve problems of graph isomorphism and the issue of strings that do not correspond to valid molecular graphs specifically.
Although providing insights to existing solutions, Computational de novo drug design plays an important role. These days when deep learning came into existence in cheminformatics it makes the work of human easy in the way, to search the synthetic chemical solution space and develop novel compounds for target binding, deep learning AI can be applied.

6. References
[1] Yasonik J. Multiobjective de novo drug design with recurrent neural networks and nondominated sorting. J Cheminform 12, 14. 2020.
[2] Lim J, Ryu S and Kim J W et al. Molecular generative model based on conditional variational autoencoder for de novo molecular design. J Cheminform 10, 31 2018.
[3] Yibo Li, Liangren Zhang and Zhenming Liu. Multi-objective de novo drug design with conditional graph generative model. Journal of ChemInformatics, 2018.
[4] Zheng S, Yan X and Gu Q et al. QBMG: quasi-biogenic molecule generator with deep recurrent neural network. J Cheminform 11, 5 2019.
[5] Kwon Y, Yoo J, Choi YS et al. Efficient learning of non-autoregressive graph variational autoencoders for molecular graph generation. J Cheminform 11, 70 2019.
[6] Li X and Fourches D. Inductive transfer learning for molecular activity prediction: Next-Gen QSAR Models with MolPMoFiT. J Cheminform 12, 27 2020.
[7] Chengkang Tang, Feng Qiu and Xiaojun Zhao. Molecular Design and Applications of Self-Assembling Surfactant-Like Peptides. Journal of Nanomaterials, vol. 2013.
[8] Philippe Gantzer, Benoit Creton and Carlos Nieto-Drighi. Inverse-QSPR for de novo design: A review. Molecular Informatics. 4, Wiley Online Library, 2018.
[9] Stephen Wu, Guillaume Lambard, Chang Liu, Hirona, Yamada and Ryo Yoshida. iQSPR in XenonPy: A Bayesian Molecular Design Algorithm. Molecular Informatics. 1-2, Wiley Online Library, 2019.
[10] Peter B. Jørgensen, Mikkel N. Schmidt and Ole Winther. Deep Generative Models for Molecular Science. Molecular Informatics. 1-2, Wiley Online Library, 2018.
[11] Daniel Merk, Lukas Friedrich, Francesca Grisoni and Gisbert Schneider. De Novo Design of Bioactive Small Molecules by Artificial Intelligence. Molecular Informatics. 1-2, Wiley Online Library, 2019.
[12] D d Marchi and A Budhiraja. Augmenting Molecular Images with Vector Representations as a Featurization Technique for Drug Classification. ICASSP 2020 - 2020 IEEE International Conference on Acoustics. Speech and Signal Processing (ICASSP). pp. 956-60, 2020.
[13] F Shi, S You, and C Xu. *Reinforced Molecule Generation with Heterogeneous States*. IEEE International Conference on Data Mining (ICDM). pp. 548-57, 2019.

[14] Samuel S Y Wong, Weimin Luo and Keith C C Chan. *EvoMD: An Algorithm for Evolutionary Molecular Design*. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2011.

[15] Hongbing Fang, Qian Wang, Yi-Cheng Tu and Mark F Horstemeyer. *An Efficient Non-dominated Sorting Method for Evolutionary Algorithms*. Researchgate, 2017.

[16] Shuang Wang, Zhen Li, Shugang Zhang, Mingjian Jiang, Xiaofeng Wang and Zhiqiang Wei. *Molecular Property Prediction Based on a Multichannel Substructure Graph*. IEEE Access, 2020