JOURNAL OF
EMERGING INVESTIGATORS

Article

Evaluating the feasibility of SMILES-based autoencoders for drug discovery

Colin Zhang[1], Kristine Govani[2]
[1] Carlmont High School, Belmont, California
[2] Department of Computer Science, Carlmont High School, Belmont, California

SUMMARY

The vast majority of molecules with desirable drug-like properties have not yet been discovered. With the advent of machine learning for de novo molecular generation, the process of designing these molecules has become increasingly efficient. However, to what extent are these machine learning models actually learning chemical properties versus memorizing the syntax of a training set? In this project, we trained a Simplified Molecular Input Line Entry System (SMILES)-based generative autoencoder for up to 200 epochs to investigate whether the latent space can separate molecules based on five chemical properties (partition coefficient, molecular weight, topological polar surface area, number of hydrogen bond donors, and number of hydrogen bond acceptors) and how generated molecules compare to the training set. We hypothesized that the model would preferentially encode molecular weight and that generated molecules would be similar to the training set. Consistent with our hypothesis, the model quickly learned to distinguish molecules primarily by their molecular weight, while other properties were considered to a lesser extent. Moreover, generated molecules were very similar to the training set both in terms of structure and properties. These results suggested that the model overfits the training set. In particular, the model best learns chemical properties that directly depend on atomic composition while it is difficult for the model to encode higher-level properties that rely on connectivity and structure. Our results may represent fundamental limitations of SMILES-based generative models and could assist in development of new research to mitigate these issues.

INTRODUCTION

Drug discovery is crucial for improving public health by providing new treatment options, preventing disease occurrence and addressing unmet medical needs. Drug-like chemical space is estimated to contain as many as 1060 possible organic compounds (1,2). Due to the resource-intensive process of drug development, virtual screening techniques have been developed to assess chemical space for its potential to contain effective drugs (3). High-throughput screening (HTS) is a technique that has been used in both structure-based and ligand-based approaches to scan millions of pharmaceutical compounds for potential drugs in a short period of time (4). However, the success rate of HTS in detecting candidate molecules is small, and scientists have looked to machine learning algorithms for better results.

The field of deep generative chemistry aims to locate novel molecules with enhanced chemical properties for drug design, such as binding affinity, partition coefficient, and aqueous solubility (3,5,6). Deep generative chemistry involves using neural networks to quickly generate new chemical compounds, thus speeding up the slow process of molecular design. A generative model is first trained on a large database of known molecules, allowing it to identify certain characteristics of known molecules and generate new ones not present in the original training set. These generated molecules can then be experimentally tested to determine their properties and potential as drug candidates. By using machine learning to identify potential drug candidates faster and more accurately than traditional methods, deep generative chemistry enables more efficient drug discovery and design (7).

Machine learning architectures in generative chemistry fall into three major categories: autoencoders (AEs), generative adversarial networks (GANs), and reinforcement learning (RL) (8). Autoencoders and their variations, such as variational autoencoders (VAEs) and adversarial autoencoders (AAEs), have been popular for their ability to generate novel molecules (9–11). The general structure of a three-layer autoencoder includes: encoder, latent space, and decoder. The input is a representation of a molecule, typically in the form of a SMILES string. The latent space encodes the input into a condensed latent vector generally possessing fewer dimensions than the input. The decoder converts the latent vector into the input format. In other words, the goal of the decoder is to replicate the input based on its latent representation (12). Ideally, these latent vectors should encode some useful properties of the input molecule. The generative capability of an autoencoder arises from traversing the latent space to produce novel molecules with diverse chemical properties. Thus, in order for autoencoders to be effective generative models, the latent representation of molecules must be carefully optimized (13).

There are several desirable characteristics of a chemical latent space. First of all, the latent space should be dense so that the majority of latent vectors correspond to candidate molecules. Another important characteristic is a wide diversity of generated molecules. A common problem in generative models is overfitting, where the model can only generate molecules that are very similar to the training set (14). Diversity is important for molecular generation because it facilitates exploration of the chemical space (15). In addition, vectors close together should produce similar molecules and distant vectors should produce different molecules. In particular, the latent space should exhibit smooth variations; that is, the chemical properties of a molecule should change gradually

as the latent space is traversed. Thus, the properties of generated molecules can be carefully controlled in relation to a seed molecule by varying the distance between their latent vectors.

These desired characteristics raise the question of which, if any, chemical properties can actually be encoded by a SMILES-based autoencoder. Jin *et al.* have raised concerns about the ability of SMILES strings to encode chemical properties, given that similar molecules can have very different SMILES representations (16). However, several studies have shown that there is indeed a correlation between certain chemical properties and SMILES-based latent vectors (17, 18). Specifically, researchers have found a link between the Tanimoto similarity and latent vector distances of molecules generated by SMILES-based architectures (17). Galushka and colleagues used latent vectors generated by a SMILES-based VAE to predict chemical properties, namely LogD (lipophilicity for ionizable compounds) and binding affinity and showed that the latent space learns some but not all chemical characteristics from SMILES strings (18).

Evidently, SMILES-based autoencoders can encode certain chemical properties, but no study has yet compared whether some properties are preferentially encoded. The latent space is constantly changing during training; therefore, by examining this latent space at various stages, we can determine the order and extent to which certain properties become encoded in the latent space. We trained a SMILES-based generative autoencoder to investigate how the following important chemical properties are represented in the latent space by probing the density, smoothness, and diversity of the latent space: the partition coefficient (LP), average molecular weight (MW), number of hydrogen bond donors (HBD), number of hydrogen bond acceptors (HBA), and topological polar surface area (TPSA) (**Figure 1**). With the exception of TPSA, these properties are cited in Lipinski's Rule of 5, a rule used for drug design to determine a molecule's suitability as a pharmaceutical drug (19). TPSA is also used to predict the ability of a drug to pass through the blood-brain barrier in the human body (20). We focus on the above properties due to their importance for assessing the viability of drug candidates.

We hypothesized that the model would encode MW the best out of all tested chemical properties, because it is least dependent on molecular structure. We also predicted that the model would overfit the chemical properties of the training set. The results indeed showed that MW has the strongest correlation with the location of molecules within the latent space. Furthermore, generated molecules had similar structural and chemical properties as the molecules in the training set. We concluded that SMILES-based autoencoders may not be ideal for molecular generation due to their difficulty encoding more complex properties. In addition, the generative capabilities of these models are limited to the properties of the training set, even though such overfitting is not obvious during training. Our study suggests the need to develop alternative architectures for better results.

RESULTS

Model architecture & training

We used a basic autoencoder architecture to construct the model, where both the encoder and decoder contain a gated recurrent unit (GRU) and the latent vector is of size 64 (**Figure 2A**). The encoder GRU performs a sequence-to-vector operation, and the decoder GRU performs a sequence-to-

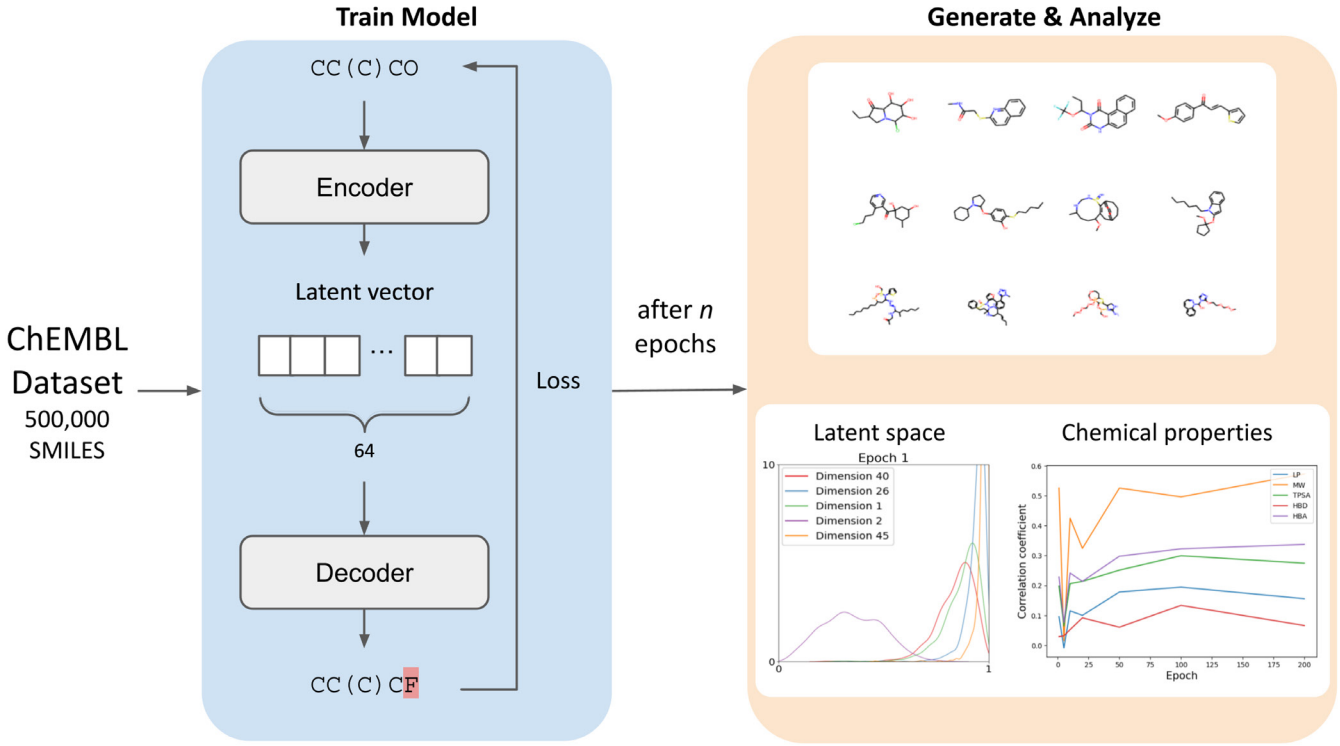

Figure 1: Graphical abstract of the study. We trained an autoencoder using the ChEMBL dataset to compress small molecule SMILES into a latent vector, and generated molecules by sampling random points in the latent space. We analyzed the latent space and chemical properties of generated molecules at different stages of training.

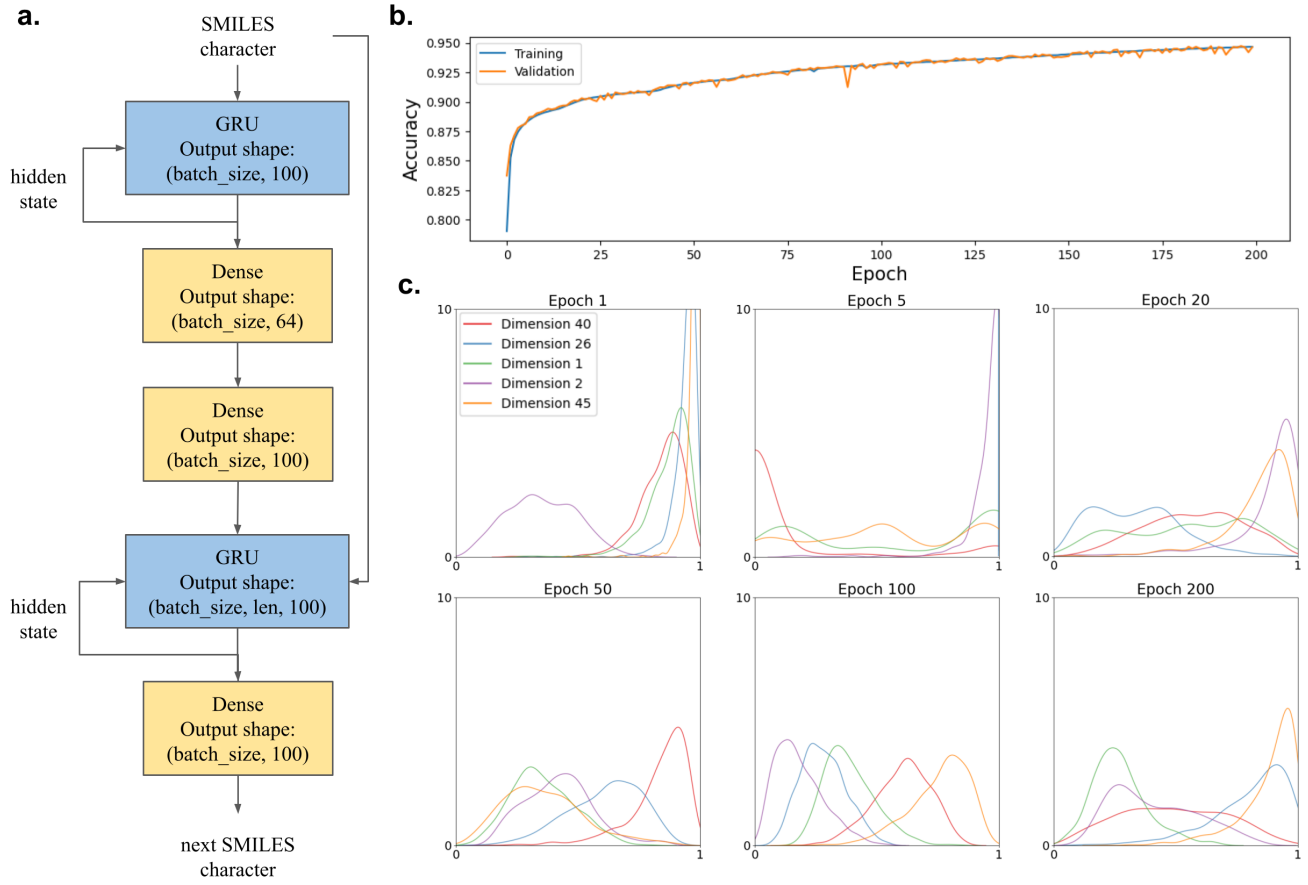

Figure 2: Architecture of SMILES-based GRU autoencoder, and evolution of model accuracy and latent space over epochs. A) Model architecture with layer dimensions, where len represents the length of the padded SMILES sequence. A latent space with 64 dimensions was empirically chosen. **B)** Training accuracy curve for the model over 200 epochs. **C)** Kernel density plots of the latent space at the specified epochs, showing 5 randomly chosen latent dimensions. As training time increases, the density of molecules along each latent dimension becomes more normally distributed. X axes: value along the latent dimension. Y axes: number of molecules.

sequence operation. The encoder GRU converts the one-hot encoded SMILES sequence to the latent vector. A dense layer reshapes and transforms the encoder GRU output into an intermediate representation referred to as the latent vector. A second dense layer modifies the latent vector prior to feeding into the decoder. The decoder recreates the original SMILES sequence one character at a time using contextual information from both the latent vector and characters earlier in the sequence. The latent vector represents context from the entire molecule and is used to set the initial hidden state of the decoder GRU. This hidden state is modified by each character of the predicted sequence as it is generated.

For model training and evaluation, we used a portion of the ChEMBL22 dataset containing 500,000 unique SMILES strings with a maximum length of 100 characters (21). The SMILES strings were tokenized, one-hot-encoded, and modified with start and end tokens as described in a previous study (14). The training process involved a reconstruction task where an input SMILES string was reconstructed after being passed through the model.

In order to investigate model characteristics at various stages in the training process, we saved a copy of the model at seven checkpoints: 1, 5, 10, 20, 50, 100, and 200 epochs. Analysis of the accuracy and loss showed a consistent improvement in model performance on both training (94.66%) and test (94.73%) sets (**Figure 2B**). The accuracy continued to increase at 200 epochs, suggesting that higher accuracy may be obtained given a longer training time. To visualize how the distribution of information across latent dimensions changed as the model trained, we plotted the kernel density of latent vector values for 1000 randomly chosen molecules from the training set at each checkpoint (**Figure 2C**). Values ranged from zero to one because of the sigmoid activation function in the latent dense layer. The distribution of values in the latent dimensions began highly skewed at earlier checkpoints but transitioned to a more normal distribution at later checkpoints. At 200 epochs, the latent dimension density plots had nearly all relatively normal distributions in contrast to the density plots at 1 epoch, which were skewed.

Visualizing the latent space

In order to determine whether the five chemical properties of interest (LP, MW, TPSA, HBD, HBA) are encoded by the latent vector, we visualized the latent space as the model was trained. For 200 randomly chosen molecules from the training set at each checkpoint, we plotted the pairwise Euclidean distance between latent vectors against the difference in their chemical property values. We then calculated the Pearson's correlation coefficient for each plot (**Figure 3A**). A high average correlation between the

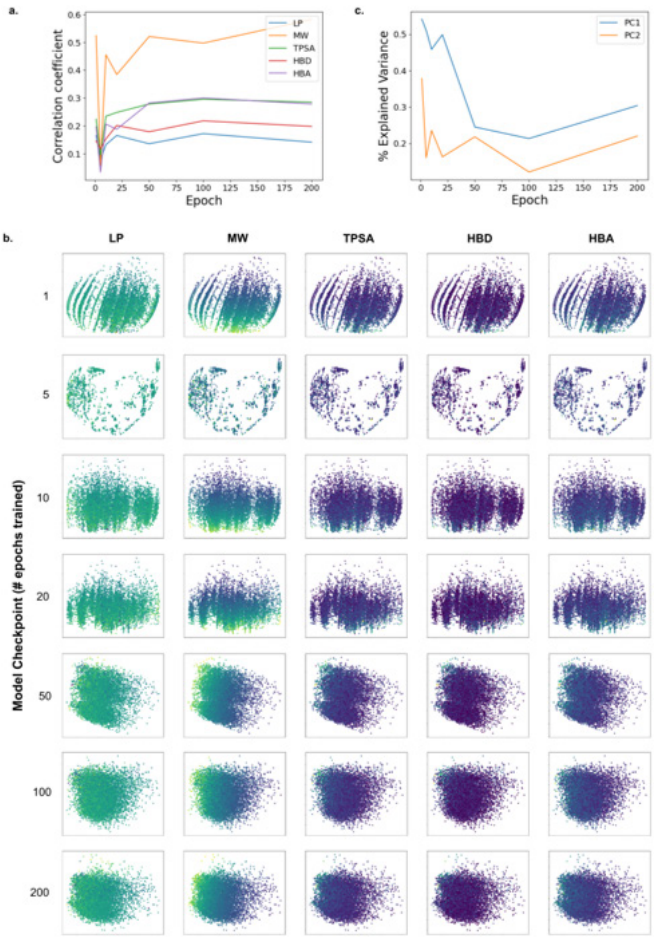

Figure 3: Distribution of chemical properties across the latent space. A) Correlation coefficients of pairwise latent vector distances with property differences over training time. **B)** Principal component analysis showing the first two principal components (x and y axes, respectively) for each chemical property of interest at each checkpoint. LP, partition coefficient; MW, average molecular weight; HBD, number of hydrogen bond donors; HBA, number of hydrogen bond acceptors; TPSA, topological polar surface area. **C)** Percentage explained variance for PC1 (principal component 1) and PC2 (principal component 2) over training epochs.

Euclidean distance and chemical property distance would suggest that the model had learned to differentiate molecules by the corresponding chemical property, i.e., the latent space encoded that property. Interestingly, the properties displayed high correlation coefficients at 1 epoch and then sharply decreased before increasing again. The magnitude of these coefficients differed significantly between properties. For example, MW consistently exhibited the highest correlation coefficient while the lowest correlation coefficients belonged to LP and HBD. However, no correlations were negative.

As another way to visualize the encoding of chemical properties throughout the latent space, we conducted principal component analysis (PCA). PCA plots provide a two-dimensional projection of the latent space, where molecules closer together in the PCA plot tend to have closer latent vectors. Chemical property values were represented as different shades of color (**Figure 3B**). The presence of a color gradient indicates that molecules in the same neighborhood of latent space have similar chemical properties. The gradients were the most visible for the property MW, indicating that molecules with similar molecular weight are close together in the latent space. Additionally, consistent with earlier results, the decrease in explained variance for the top two principal components (PC1 and PC2) shows that the variation in the data became distributed more evenly between latent dimensions as the model was trained for longer (**Figure 3C**).

Evaluating model overfitting

Next, we looked at the diversity of randomly generated molecules from the model. Latent vectors were randomly defined such that each element of the vector is between 0 and 1. By converting these latent vectors to hidden states assigned to the decoder, random SMILES strings were generated. The randomly generated molecular structures became subjectively more complex as training time increased, as shown by the greater diversity of substructures (**Figure 4A**). Despite this, the relative distribution of rings and functional groups in generated molecules remained relatively constant (**Figure 4B-C**). In fact, the relative frequencies of these rings and functional groups in the generated molecules matched the relative frequencies found in the training set, suggesting that the model had overfitted the chemical structures of the training set (**Figure 4B-E**).

The model can also generate variations of a target molecule, which we refer to as target-guided generation. From the validation set, a target molecule was chosen which

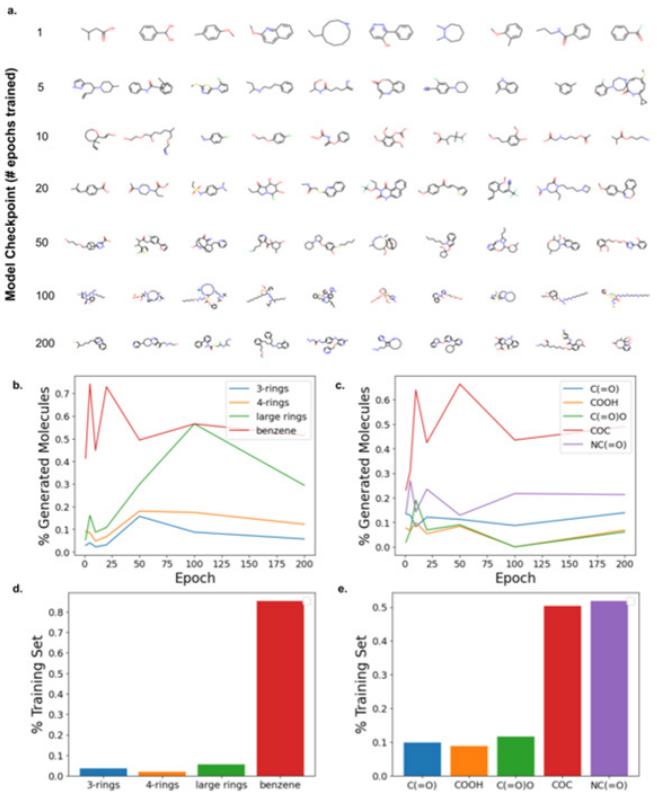

Figure 4: Molecules randomly generated by model. A) Randomly generated molecules from the latent space. 10 example molecules are shown for each checkpoint. Distribution of differently sized rings found in randomly generated molecules at each checkpoint **(B)** and in the training set **(D).** Large rings indicate rings with over 6 atoms. Distribution of common functional groups found in randomly generated molecules **(C)** and training set **(E).**

had very different properties from most of the training set (**Figure 5A-B**). This molecule was converted to a latent vector and nearby latent vectors were sampled by adding Gaussian noise with standard deviation 0.01. Thus, molecules similar to the target were generated (**Figure 5C**). Because latent vectors may yield invalid SMILES strings (e.g., incorrect syntax or incompatibility with RDKit), the success rate of generating molecules is less than 100%. During both target-guided and random generation, the highest success rates were achieved very early during training and then fluctuated widely (**Figure 5D**). The generated structures became more similar to the target as training time increased, although the ranges of Tanimoto values, a measure of chemical similarity, were fairly large (**Figure 5E**). Additionally, the distribution of

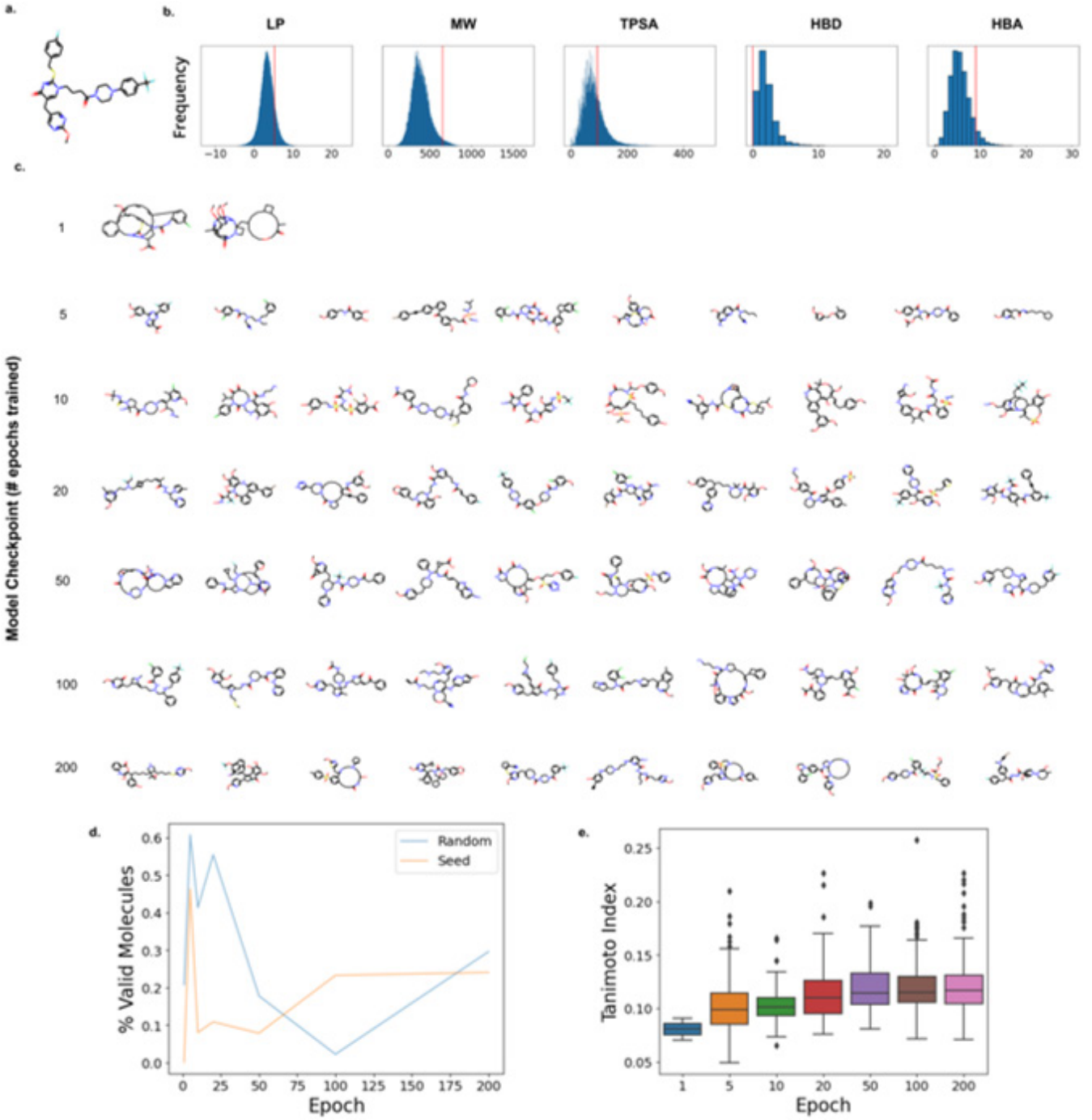

Figure 5: Variations of a target ligand produced via guided generation. A) Target molecule for guided generation. **B)** Properties of target molecule (red) in comparison with distribution of properties of train set (LP, the partition coefficient; MW, molecular weight; HBD, number of hydrogen bond donors; HBA, number of hydrogen bond acceptors; TPSA, topological polar surface area). **C)** Variants generated from the target molecule. 10 example molecules are shown for each checkpoint, if available. **D)** Success rates of both randomly generated (blue) and target-guided (orange) molecules, calculated based on the number of valid SMILES produced as a fraction of the total number of attempts. **E)** Tanimoto coefficient, a measure of molecular structure similarity, plotted for the target-guided molecules over training time.

chemical properties in the randomly generated molecules were very similar to the properties found in the training set (**Figure 6**). The target-guided molecules were most similar to the target molecule in terms of MW, compared with other properties. Overall, we showed that the autoencoder model effectively encodes MW in the latent space but tends to overfit on the training data, resulting in a limited range of generated molecules even when guided using a target molecule.

DISCUSSION

SMILES-based machine learning models are a promising tool for generating novel small molecules for drug discovery, but researchers suspect such models may not be properly grounded in chemical understanding. In this study, we analyzed the distribution of chemical properties in the latent space encoded by a simple SMILES-based autoencoder across different training durations. The model replicated the input SMILES string with increasing accuracy over 1, 5, 10, 20, 50, 100, and 200 epochs of training time. As the model was trained for longer periods, the latent representation of the molecule was more normally distributed across all latent dimensions. This study demonstrated two possible limitations of simple SMILES-based autoencoders. One, they poorly learn properties that rely on factors beyond the atomic composition of molecules. Two, the generative capability of these types of models may be restricted to the diversity of the training set. These results can inform the development of future molecular generative models which incorporate true chemical understanding.

We investigated the extent to which five chemical properties (LP, MW, TPSA, HBA, and HBD) are encoded in the latent space. A property can be considered to be encoded by the model if molecules possessing different values of that property are well-separated in the latent space. Our results demonstrate that MW was the only property effectively learned by the model. Other properties were generally poorly encoded regardless of training duration. The differential ability of the model to learn various chemical properties suggests that there are fundamental limitations to the way a SMILES-based autoencoder processes chemical information. In particular, we made a distinction between compositional properties (which depend mostly on the types and numbers of atoms present) and structural properties (which depend mostly on connectivity between atoms). Researchers have found that SMILES strings are not ideal for encoding structural information (16,22). For example, because TPSA is calculated additively using the PSA values of molecular fragments, in order to generate TPSA-relevant latent features a model must be able to encode the types and numbers of molecular fragments (20). Similarly, LP depends heavily on the presence of polar or nonpolar fragments, as lipophilicity is determined by intermolecular interactions and hydrogen bonds (23). However, the ability to reconstruct SMILES strings is independent of the ability to recognize fragments, because in many cases there is not a one-to-one correspondence between the presence of a fragment and the presence of a particular sequence of SMILES characters. Consequently, ways to mitigate this limitation have been proposed, including the use of heteroencoders, which reconstruct canonical SMILES strings to noncanonical SMILES strings or vice versa (24,25). Alternatively, the use of SMILES strings can be bypassed altogether with molecular graphs that directly encode atomic connectivity (26).

We also observed the tendency for our SMILES-based autoencoder to overfit the training set, limiting the diversity of generated molecules. Several pieces of evidence indicate that this occurred. Firstly, the decreasing generative success rate after five epochs of training suggests the latent space becomes more sparse as training accuracy increases. In order to more accurately reproduce SMILES strings of the training set, the model may be increasing the distance between their latent representations. Secondly, there is considerable overlap between the distribution of structural fragments and chemical properties in the training set and a pool of randomly sampled molecules from the latent space. This means that the latent space is biased toward the chemical properties and molecular fragments found in the training set, perhaps by recreating verbatim segments of training SMILES strings. The overfitting phenomenon is intriguing in light of the fact that, during training, the training and validation losses do

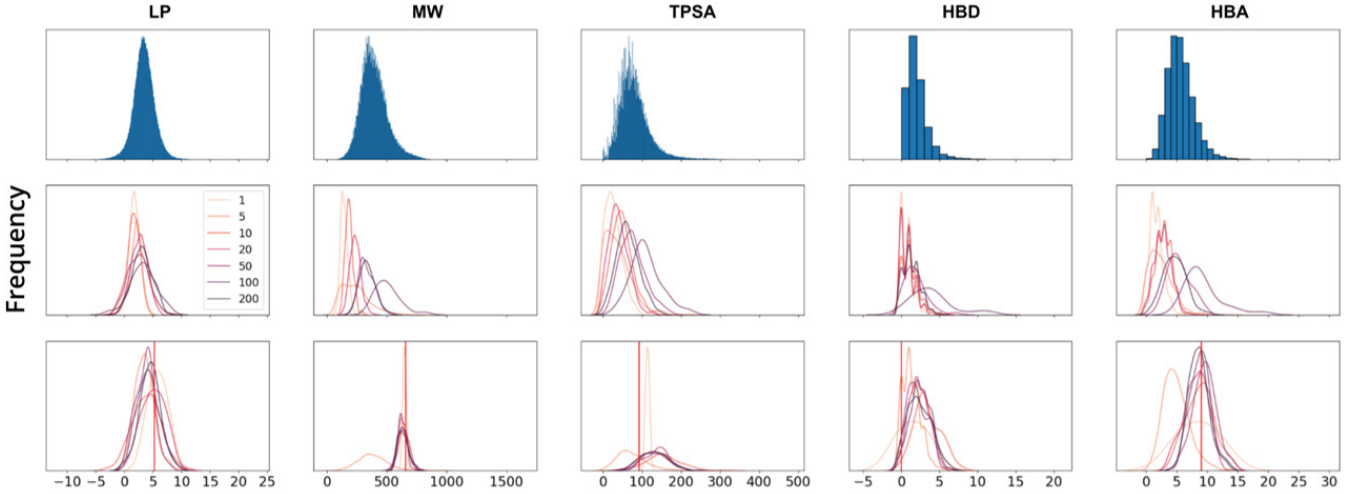

Figure 6: Property distribution of training set (top), randomly generated molecules (middle), and target-guided molecules (bottom). Red lines in the bottom row indicate property values of the target molecule (LP, the partition coefficient; MW, molecular weight; HBD, number of hydrogen bond donors; HBA, number of hydrogen bond acceptors; TPSA, topological polar surface area). The MW distribution of the target-generated molecules is very tightly clustered around the MW of the target molecule at 200 epochs, whereas other properties are less similar.

not diverge. A possible explanation is that the molecules in the validation set are very similar to those in the training set. Thus, when the model learns how to reconstruct SMILES strings from the training set, it can apply this equally well to the validation set.

Several changes could be made to this model to further analyze the capabilities of simple autoencoders. In order to keep this analysis simple, we designed this model with a single GRU layer in the encoder and decoder. Multiple stacked recurrent layers may allow the model to learn higher-order patterns in the SMILES strings and ameliorate some of the previously mentioned limitations in encoding chemical properties. Moreover, some studies have found that Long Short-Term Memory cells may be better than GRUs at encoding higher complexity sequences (27). As a result, they may provide better molecular latent representations from the SMILES strings. Newer versions of autoencoders implement more complex architectures compared to the simple encoder-decoder architecture found in our model. In a VAE, the addition of a special term to the loss function encourages the latent space to approximate a standard normal distribution (28). VAEs thus ensure that the properties being encoded in the latent space are not just a result of minimizing the reconstruction error between the input and the output, which could lead to overfitting. AAEs also use special loss functions to regularize the latent space (29). This makes it surprising that our vanilla autoencoder model generates a normally distributed latent space, despite the lack of explicit regularization. Examining the cause of this phenomenon, by decreasing the number of latent dimensions or changing the activation of the latent layer, for example, would be an interesting subject of future study. For example, a Gaussian layer inserted just before the decoder can add noise to the latent vector during training and make the decoder more robust to small deviations in the latent vector. Lastly, the training set can be expanded to reduce implicit bias. Although the ChEMBL dataset has been used as the training set in many generative models, other studies have used the subsets of the ZINC dataset, such as ZINC-250K, a dataset containing 250,000 molecules (5,11,14). The ChEMBL dataset is primarily generated from molecules used in disease research, those with high drug-likeness and compatible with the human body (21). On the other hand, the ZINC dataset is specifically focused on ligand discovery for protein docking and is limited to commercially available compounds (30). Combining multiple datasets may improve the generative diversity of the model.

As computational methods become increasingly important in the physical sciences, we must understand their abilities and limitations. Our work demonstrates that SMILES-based autoencoders can generate novel molecules, but their limited ability to control the chemical properties of generated molecules and go beyond the diversity of the training set may restrict their utility. Researchers should continue to develop and improve upon these computational tools in order to fully leverage their potential in drug discovery.

MATERIALS AND METHODS

Dataset

The ChEMBL22 dataset used in this investigation was downloaded from Kaggle, a community data science platform (31). 500,000 SMILES strings were randomly selected, with a maximum length of 100 characters. The dataset was split into a 9:1 ratio and filtered for compatibility with RDKit (an open-source toolkit for cheminformatics), resulting in a training set containing 449,685 molecules and a validation set containing 49,951 molecules. Similar to the process of tokenizing words or subsets of words in Natural Language Processing, dictionaries were constructed with an integer key for each unique SMILES character (with start and end tokens) in preparation for one-hot encoding. The final dictionary contained 39 unique characters (not including start and end tokens). One-hot encoding represents categorical data in a way that reduces bias in the model during training towards higher-value tokens, since everything becomes either 0 or 1. The train and validation sets were one-hot encoded, and each SMILES string was padded with start and end tokens to a total length of 102 characters.

Model Details

The model consisted of encoder and decoder portions, each containing a GRU. The input was an array of one-hot encoded SMILES characters with two dimensions: the length of the SMILES sequence and the number of unique characters in the dictionary (the length of each one-hot encoded character). A third dimension, the batch size of SMILES strings being fed into the model, was automatically prepended by Keras. The input layer was fed into the encoder GRU one character at a time, over a number of time steps equal to the number of characters in the SMILES sequence. At each time step, a hidden state was returned by the GRU that retained some information from previous characters. Consequently, when the entire SMILES sequence was passed through the encoder GRU, the final hidden state generated contained vital information pertinent to the entire SMILES sequence. This state is essentially a representation of the latent vector of the SMILES string. The output shape of the encoder GRU has two dimensions: the batch size and the recurrent unit size. The recurrent unit size represented the number of neurons in the GRU and was set as 100 for all occurrences of the unit in the model. The final hidden state of the encoder GRU was passed through another dense layer that had a dimensionality of 64, equal to the size of the latent vector. The output of this dense layer was the latent vector of the SMILES sequence.

The only constraint placed on the latent space is a sigmoid activation function, which limits the values to be between 0 and 1. We chose to use the sigmoid activation function in the latent layer of the autoencoder because it limits the range of latent values, discouraging overfitting. Empirically, the effect of this constraint is similar to the regularization of the latent space in a VAE.

The decoder portion of the model was used to convert the modified latent vector produced by the encoder model into an array of probabilities that determined the next character of the SMILES sequence. To convert the latent vector to a SMILES sequence, a second dense layer was required to transform the latent vector into the initial hidden state of the decoder GRU. The decoder GRU requires a starting input to base its prediction on, so the first input of the encoder (the start token) was reused as the input of the decoder GRU. The return_sequences parameter of the decoder GRU was set as "True" since the unit has to output a character in every timestep up to the length of the entire SMILES sequence. Finally, the decoder GRU was passed into a dense layer with softmax activation.

The final output was the array of probabilities representing the next character in the generated SMILES sequence. The model was trained for 1, 5, 10, 20, 50, 100, and 200 epochs with the Adam optimizer, learning rate of 0.001, categorical cross-entropy loss, and batch size of 256. The loss function calculates the error between the predicted next character and actual next character, allowing the model to learn to correctly reconstruct the SMILES string.

In order to generate novel molecules, the original model was split into two separate portions: the encoder, which can convert a target molecule to a latent vector, and the decoder, which can translate latent vectors back to SMILES strings. The encoder portion was redefined with input and output layers extracted directly from the original model using the model. getlayer() method in Keras that preserved the original weights in these layers. The decoder was defined similarly, except that the hidden states of the decoder GRU were preserved across batches because characters were inputted one at a time in independent batches. In contrast, during training, entire molecules were processed in a batch so that the states did not need to be preserved from previous molecules. The weights of the decoder GRU were transferred from the trained model using the model.get_weights() and model. set_weights() methods. To generate a full SMILES string, the decoder model was run in a loop after setting the start token and latent vector. Generated characters were appended to the string until the model encountered an end token.

Two important characteristics of this training process were teacher forcing and input-output shift. Teacher forcing refers to the ground truth sequence (input) being fed character-wise into the decoder, so that every timestep the decoder uses the ground truth rather than its own output to generate the next prediction (32). Teacher forcing mitigated the problem of allowing errors to compound across timesteps. Input-output shift was necessary in order for the model to satisfy a generative purpose. Specifically, the input and output strings were shifted by one, so that the input lacks the last end token and the output lacks the start token. When the model is used to generate novel molecules, it does so one character at a time, and the last generated character can be fed back into the decoder to generate the next character. Without shifting the input and output during training, a valid latent space may still be learned, but the decoder will always output the same character as the input.

Python Packages

For this project, we used Python 3.7 and Tensorflow 2.8.2 running on Google Colaboratory with GPU acceleration. Scikit-learn 1.0.2 was used for PCA and Scipy 1.7.3 was used to compute the Pearson's correlation coefficients. Plots were constructed in Python using Matplotlib 3.2.2 and Seaborn 0.11.2. In addition to drawing the molecules, RDKit 2021.09.4 was used to calculate chemical properties and identify functional groups with the Descriptors, Lipinski, and Fragments modules. Tanimoto similarity values were calculated using the AllChem and DataStructs modules of RDKit.

Received: September 20, 2022
Accepted: February 20, 2023
Published: June 19, 2023

REFERENCES

1. Bohacek, Regine S., et al. "The art and practice of structure-based drug design: a molecular modeling perspective." *Medicinal Research Reviews,* vol. 16, no. 1, Jan. 1996, doi: 10.1002/(SICI)1098-1128(199601)16:1<3::AID-MED1>3.0.CO;2-6.
2. Reymond, Jean-Louis, et al. "Chemical space as a source for new drugs." *MedChemComm,* vol.1, Apr. 2010, doi: 10.1039/C0MD00020E.
3. Hughes, James P., et al. "Principles of early drug discovery." *British Journal of Pharmacology*, vol. 162, no. 6, Mar. 2011, doi: 10.1111/j.1476-5381.2010.01127.x.
4. Pereira, D. A., and Williams, J.A., "Origin and evolution of high throughput screening." *British Journal of Pharmacology*, vol. 152, no.1, Sep. 2007, doi: 10.1038/sj.bjp.0707373.
5. Maziarka, Łukasz, et al. "Mol-CycleGAN: a generative model for molecular optimization." *Journal of Cheminformatics*, vol. 12, no. 1, Jan. 2020, doi: 10.1186/s13321-019-0404-1.
6. Bilodeau, Camille, et al. "Generating molecules with optimized aqueous solubility using iterative graph translation." *Reaction Chemistry & Engineering*, vol. 7, no. 2, Nov. 2021, doi: 10.1039/D1RE00315A.
7. Vamathevan, Jessica, et al. "Applications of machine learning in drug discovery and development." *Nature Reviews Drug Discovery*, vol. 18, no. 6, Jun. 2019, doi: 10.1038/s41573-019-0024-5.
8. Merz Jr, Kenneth M., et al. "Generative Models for Molecular Design." *Journal of Chemical Information and Modeling*, vol. 60, no. 12, Dec. 2020, doi: 10.1021/acs.jcim.0c01388.
9. Samanta, Soumitra, et al. "VAE-Sim: a novel molecular similarity measure based on a variational autoencoder." *Molecules*, vol. 25, no. 15, Jul. 2020, doi: 10.3390/molecules25153446.
10. Kadurin, Artur, et al. "druGAN: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico." *Molecular Pharmaceutics*, vol. 14, no. 9, Jul. 2017, doi: 10.1021/acs.molpharmaceut.7b00346.
11. Prykhodko, Oleksii, et al. "A de novo molecular generation method using latent vector based generative adversarial network." *Journal of Cheminformatics*, vol. 11, no. 1, Dec. 2019, doi: 10.1186/s13321-019-0397-9.
12. Gómez-Bombarelli, Rafael, et al. "Automatic chemical design using a data-driven continuous representation of molecules." *ACS Central Science*, vol. 4, no. 2, Jan. 2018, doi: 10.1021/acscentsci.7b00572.
13. Bojanowski, Piotr, et al. "Optimizing the latent space of generative networks." *arXiv preprint arXiv:1707.05776*, Jul. 2017, doi: 10.48550/arXiv.1707.05776.
14. Bjerrum, Esben Jannik, and Richard Threlfall. "Molecular generation with recurrent neural networks (RNNs)." *arXiv preprint arXiv:1705.04612*, May 2017, doi: 10.48550/arXiv.1705.04612.
15. Rupakheti, Chetan, et al. "Strategy to discover diverse optimal molecules in the small molecule universe." *Journal of Chemical Information and Modeling*, vol. 55, no. 3, Jan. 2015, doi: 10.1021/ci500749q.
16. Jin, Wengong, et al. "Junction tree variational autoencoder for molecular graph generation." *Proceedings of the 35th*

International Conference on Machine Learning, vol. 80, Feb. 2018, doi: 10.48550/arXiv.1802.04364.
17. Schiff, Yair, et al. "Characterizing the latent space of molecular deep generative models with persistent homology metrics." *Topological Data Analysis and Beyond Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020),* vol. 34, Oct. 2020, doi: 10.48550/arXiv.2010.08548.
18. Galushka, Mykola, et al. "Prediction of chemical compounds properties using a deep learning model." *Neural Computing and Applications*, vol. 33, no. 20, Jun. 2021,. doi: 10.1007/s00521-021-05961-4.
19. Lipinski, Christopher A., et al. "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings." *Advanced Drug Delivery Reviews*, vol. 46, no. 1-3, Mar. 2001, doi: 10.1016/s0169-409x(00)00129-0.
20. Prasanna, S., and R. J. Doerksen. "Topological polar surface area: a useful descriptor in 2D-QSAR." *Current Medicinal Chemistry*, vol. 16, no. 1, 2009, doi: 10.2174/092986709787002817.
21. Gaulton, Anna, et al. "The ChEMBL database in 2017." *Nucleic Acids Research*, vol. 45, no. 1, Jan. 2017, doi: 10.1093/nar/gkw1074.
22. Vanhaelen, Quentin, et al. "The advent of generative chemistry." *ACS Medicinal Chemistry Letters*, vol. 11, no. 8, Aug. 2020, doi: 10.1021/acsmedchemlett.0c00088.
23. Rutkowska, Ewelina, et al. "Lipophilicity--methods of determination and its role in medicinal chemistry." *Acta Poloniae Pharmaceutica*, vol. 70, no. 1, Jan-Feb. 2013, url: PMID: 23610954
24. Arús-Pous, Josep, et al. "Randomized SMILES strings improve the quality of molecular generative models." *Journal of Cheminformatics*, vol. 11, no. 1, Nov. 2019, doi: 10.1186/s13321-019-0393-0.
25. Bjerrum, Esben Jannik, and Boris Sattarov. "Improving chemical autoencoder latent space and molecular de novo generation diversity with heteroencoders." *Biomolecules*, vol. 8, no. 4, Oct. 2018, doi: 10.3390/biom8040131.
26. Kearnes, Steven, et al. "Molecular graph convolutions: moving beyond fingerprints." *Journal of Computer-Aided Molecular Design*, vol. 30, no. 8, Aug. 2016, doi: 10.1007/s10822-016-9938-8.
27. Cahuantzi, Roberto, et al. "A comparison of LSTM and GRU networks for learning symbolic sequences." *arXiv preprint arXiv:2107.02248,* Jun. 2021, doi: 10.48550/arXiv.2107.02248.
28. Kingma, Diederik P., and Max Welling. "An introduction to variational autoencoders." *Foundations and Trends in Machine Learning*, vol. 12, no. 4, 2019, doi: 10.1561/2200000056.
29. Makhzani, Alireza, et al. "Adversarial autoencoders." *arXiv preprint arXiv:1511.05644*, Nov. 2015, doi: 10.48550/arXiv.1511.05644.
30. Irwin, John J., et al. "ZINC: a free tool to discover chemistry for biology." *Journal of Chemical Information and Modeling*, vol. 52, no. 7, May 2012, doi: 10.1021/ci3001277.
31. "Drug Design with Small Molecule Smiles." *Kaggle*. www.kaggle.com/datasets/art3mis/chembl22. Accessed 26 Jun. 2022.
32. Williams, Ronald J., and David Zipser. "A learning algorithm for continually running fully recurrent neural networks." *Neural Computation*, vol. 1, no. 2, 1989, doi: 10.1162/neco.1989.1.2.270.