A de novo molecular generation method using latent vector based generative adversarial network

Oleksii Prykhodko[€,¥,Δ], Simon Johansson[€,¥,Δ,*], Panagiotis-Christos Kotsias[Δ], Esben Jannik Bjerrum[Δ], Ola Engkvist[Δ], Hongming Chen[Δ,*]

Δ Hit Discovery, Discovery Sciences, Biopharmaceutical R&D, AstraZeneca

Gothenburg, Sweden

¥ Department of Computer Science and Engineering

Chalmers University of Technology

Gothenburg, Sweden

€ Authors contribute equally

* Corresponding authors:

Simon.johansson@astrazeneca.com

Hongming.chen71@hotmail.com

Abstract

Recently deep learning method has been used for generating novel structures. In the current study, we proposed a new deep learning method, LatentGAN, which combine an autoencoder and a generative adversarial neural network for doing de novo molecule design. We applied the method for structure generation in two scenarios, one is to generate random drug-like compounds and the other is to generate target biased compounds. Our results show that the method works well in both cases, in which sampled compounds from the trained model can largely occupy the same chemical space of the training set and still a substantial fraction of the generated compound are novel. The distribution of drug-likeness score for compounds sampled from LatentGAN is also similar to that of the training set.

Introduction

The rise of deep learning has been witnessed in the cheminformatics domain in the last few years[1-5]. While deep learning methods have demonstrated a lot of impact on replacing traditional ML methods on building QSAR models, a more profound impact of deep learning on the area is probably the application of the deep generative models on de novo molecular design. Historically, the normal way to do de novo design is through searching against the virtual libraries created based on known chemical reactions alongside a set of available chemical building blocks[6]; another alternative is to use transformational rules based on the expertise of medicinal chemists to design analogues to a query structure[7]. While many successes using these techniques have been reported in the literature, it is worthwhile to point out that these methods heavily rely on the predefined rules for structure generation and doesn't have the concept of learning the prior knowledge of what a drug-like molecule should be. On the contrast, deep generative model can learn the underlying probability distribution over a large set of chemical structures in the training set, the structure generation step is basically a sampling process following the learned probability distribution among chemical space[8-10]. It is completely a data driven process and doesn't need any predefined rules for structure generation.

Bombarelli[11] et al reported the use of variational autoencoder (VAE) models for generating molecule SMILES strings. This represent the first effort on using deep generative models for structure generation. Some follow-up studies were then reported on variations of autoencoder type models. Inspired by the deep learning methods used in the natural language processing (NLP) area, Segler[9] et al reported another type of generative models using recurrent neural network (RNN) architecture to generate SMILES string. The RNN method is surprisingly simple and efficient for structure generation. Olivecrona[8] proposed the REINVENT method to combine RNN with reinforcement learning for generating structures with desirable properties. Studies using similar strategies were also reported[12].

Recently generative adversarial neural (GAN) networks[13] has become a very popular architecture for generating images. GAN has two components, a generator and a discriminator, that compete against each other during training. The discriminator network is able to distinguish true data and false data while the generator tries to generate false data to fool the discriminator. During the training, the generator gets improved on its output until the discriminator is unable to distinguish the artificial data from the true data. This method has been used in molecule generation purpose. Examples of such implementations are ORGAN[14] and ORGANIC[15]. The former was tested with both molecular generation as well as musical scores, whereas the latter was targeted directly at inverse design of molecules. The latter had trouble optimizing towards the discrete values from Lipinski's Rule of Five[16] heuristic score but showed some success in optimizing the QED[7] score. Algorithm combining GAN with RL was also used in RANC[17] and ATNC[18] where the central RNN was substituted by a differential neural computer (DNC)[19], a more advanced neural network architecture using an accessible external memory to better handle long range dependency of sequences. The authors demonstrated that the DNC-based architectures can handle longer SMILES and have more diversity in the output than the ORGANIC implementation. All the aforementioned algorithms generate molecule structures in form of SMILES string, however there are also recent efforts to generate structure on graph level[20].

In the current study, a new molecular generation strategy, LatentGAN, combining autoencoder and GAN is proposed. The difference between this method and previous GAN methods like ORGANIC and RANC is that the generator and discriminator network doesn't use SMILES strings as input, but instead n-dimensional vectors derived from the code-layer of an autoencoder trained as a SMILES heteroencoder. A pretrained heteroencoder[21] neural network was used to translate the generated n-dimensional vector into molecular structures. We first trained the GAN on a set of ChEMBL[22] compounds, after training, the GAN model was able to generate drug-like structures. Secondly, additional GAN models were trained on three target specific datasets (corresponding to EGFR, HTR1A and S1PR1 targets). Our results show that these

GAN model can generate compounds which are similar to the ones in the training set but are still novel structures. It is suggested that the LatentGAN method can be a useful tool for de novo molecule design.

Methods and Materials

Heteroencoder architecture. A heteroencoder is an autoencoder architecture trained on pairs of different representations of the same entity, i.e. different non-canonical SMILES of the same molecule. The implementation followed the architecture previous reported[21] with some changes as reported below. The heteroencoder was to map molecule structures into latent variables using the code-layer. It consists of two neural networks, namely, the encoder and decoder, which are jointly trained as a transformation pipeline. The encoder is responsible for translating one-hot encoded SMILES strings into a numerical latent representation whereas the decoder accepts this latent representation and attempts to reconstruct one of the possible non-canonical SMILES string that it represents.

Initially, the one-hot encoded SMILES string is propagated through a two-layer bidirectional encoder with 512 Long Short-Term Memory units per layer, half of which are used for the forward and half for the backward direction. The output of the first layer is fed to the second one whereas the outputs of both layers are thereafter concatenated and expanded by a feed-forward layer with 512 units for the latent vector. As a regularizing step, this vector is furthermore perturbed by applying additive zero-centered Gaussian noise with a standard deviation of 0.1. The latent representation of the molecule is fed to a feed-forward layer, the output of which is copied and inserted as hidden and cell states to the 4 layers of the decoder. Finally, the output of the ultimate decoding layer is processed by a feed-forward layer with softmax activation, to return the probability of sampling each character of the known character set of the dataset. All encoding and decoding layers consist of 512 Long Short-Term Memory units. Batch normalization with momentum with a value of 0.9 is applied on the output of every hidden layer, except for the noise layer.

During training the neural network is trained on pairs of randomly chosen non-canonical SMILES string of the molecules as data augmentation[23]. Due to the possible number of pairs, it is very unlikely that the network is ever shown the same specific pair for each molecule in different epochs. The autoencoder network was trained for 100 epochs with a batch size of 128 sequences, using a constant learning rate of 1e-3 for the first 50 epochs and an exponential decay following that, reaching a value of 1e-6 in the final epoch. The decoder was always trained using the Teacher's Forcing method.

The GAN architecture. Wasserstein GAN with gradient penalty (WGAN-GP)[24] has been chosen as a GAN model because it addresses the gradient vanishing and model collapse problems, improves stability of the training and shows good results on the generative tasks[25]. Every GAN consists of two neural networks – Generator and Discriminator – that train simultaneously.

The discriminator *D* (usually called the critic in the context of WGANs) tries to distinguish between the true data *h* and the fake data *h′*. The loss of the critic is defined as

$$L_D = E_{h' \sim P_g}[D(h')] - E_{h \sim P_r}[D(h)] + \lambda E_{\hat{h} \sim P_{\hat{h}}}\left[\left(\left\|\nabla_{\hat{h}} D(\hat{h})\right\|_2 - 1\right)^2\right], \qquad (1)$$

Where *h* is a real latent vector, *h′* is a fake latent vector generated by G. The combination of the first and second part of the equation (1) represents the difference between the mean values for the batch of fake data *h′* and true data *h*. The third part of the equation (1) corresponds to the gradient penalty described in the paper[24]. λ is a constant, which was kept to 10, following the experiments of the gradient penalty.

The $\hat{h}$ is sampled as an element-wise interpolation between h' and h with α being a vector of uniformly sampled numbers ∈ $[0,1]$,

$$\hat{h} = \alpha h' + (1 - \alpha)h. \qquad (2)$$

The discriminator is comprised of three fully connected layers with the leaky ReLU activation function in between, except for the last layer where no activation function was used.

The generator's loss is defined to be

$$L_G = E_{h' \sim P_g}[D(h')]. \qquad (3)$$

It is trained every fifth batch to keep the critic ahead and provide the generator higher gradients. The generator consists of five fully connected layers with batch normalization and leaky ReLU activation functions.

Workflow for training and sampling of the LatentGAN. The heteroencoder model was first pre-trained on the ChEMBL database for mapping structures to latent vectors. Then a set of training set compounds was selected for GAN training. The training set could be either a diverse set of random drug-like compounds in ChEMBL or a set of compounds which are active to certain targets. The latent vectors (vector *h* in Figure 1) of training set were generated from the encoder part of the heteroencoder and were used as the true data input for the discriminator, while a set of random variable ***Z*** (dimension size 512) sampled from a uniform distribution were taken as fake data input to the generator. Once the GAN training was finished, the Generator was sampled multiple times and the resulting latent vectors were decoded by the decoder to translate into SMILES strings. The process of training and sampling of GAN and decoding are summarized on the Figure 1.

Dataset and machine learning models for scoring. The heteroencoder was trained on around one million ChEMBL SMILES. A set of randomly selected 100,000 ChEMBL compounds were later selected for training a general GAN model. Three target datasets (corresponding to EGFR, S1PR1 and HTR1A) were extracted from ExCAPE-DB[26] for training target specific GANs. The ExCAPE-DB dataset were split into training and test set[27] according the chemical clustering so that chemical series will be assigned either to the training or to the test set, but not to both sets. The dataset information is listed in Table 1. To benchmark the performance of the targeted LatentGAN models, RNN based generative models (details can be seen in reference[8]) for the three targets were also generated by first training a prior RNN model based on the one million ChEMBL set (for training the heteroencoder model) and then making transfer learning on each focused target set. We built support vector machine learning (SVM) models on each of the dataset using the libsvm module in Scikit-learn package[28] and the 2048-length counted FCFP6 fingerprint in RDkit[29].

Table 1. Targeted data set and the performance of the SVM models.

Target	Training set	Test set	SVM Model ROC-AUC	SVM Model Kappa value
EGFR	2949	2326	0.85	0.56
HTR1A	48283	23048	0.993	0.9
S1PR1	49381	23745	0.995	0.91

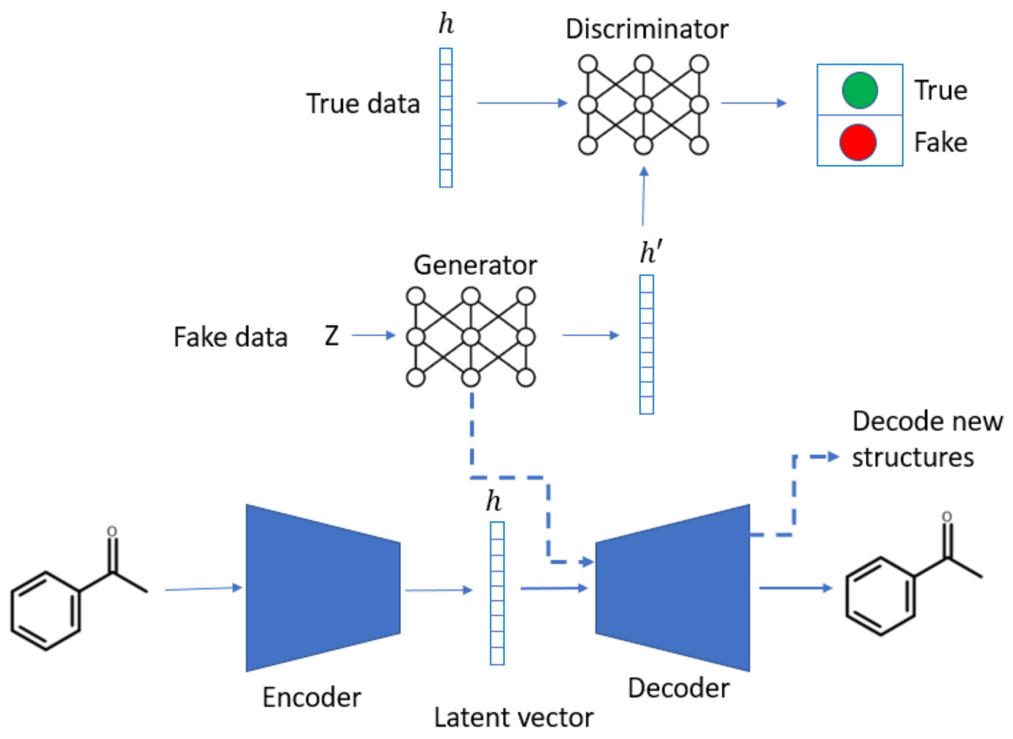


Figure 1. The workflow of LatentGAN

Results and discussion.

The training of heteroencoder. The heteroencoder was trained on the around one million compounds for 100 epochs. Its validity for the whole training set is round 99% and the reconstruction error is around 18%. Here the reconstruction error corresponds to not being able to rebuild the exact same molecule, whereas reconstruction to a different SMILES of the same molecule is counted as successes. Test set compounds were then taken as input to the encoder to get their latent variables and then decoded to SMILES string. The performance on the test set is listed in Table 2. Its validity and reconstruction error are 98% and 20% respectively.

Table 2. The performance of heteroencoder

Dataset	# compounds	Validity %	Reconstruction error[a] %
Training set	974105	99	18
Test set	10,823	98	20

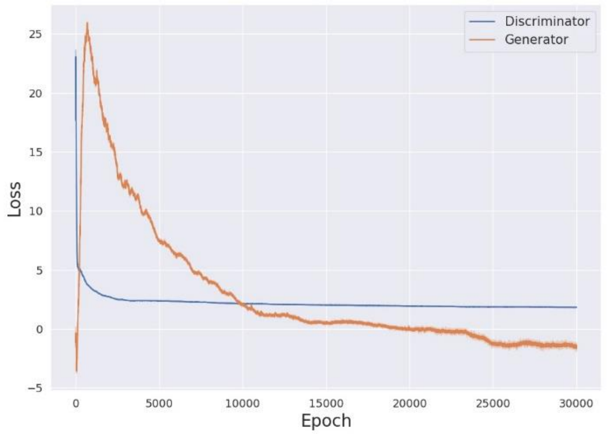
Note:

a) The reconstruction error is calculated based on valid structures.

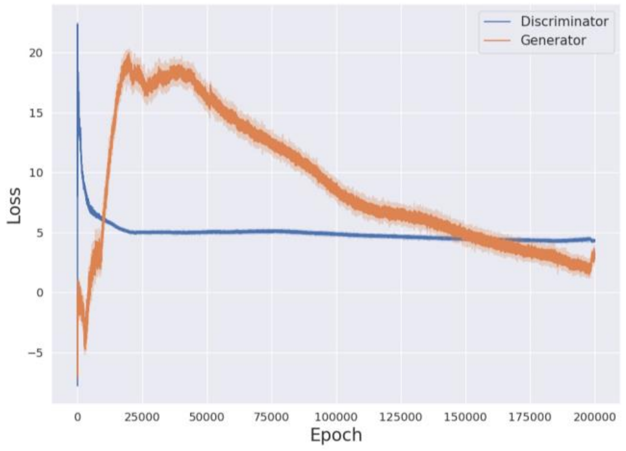
Training LatentGAN on random ChEMBL set. LatentGAN trained on a randomly selected 100K ChEMBL set to generate drug-like molecules in general. The loss curves of GAN training can be seen in Figure 2, the GAN model was trained 30,000 epochs. It seems that the loss curves of the discriminator decrease rapidly and become stable, while the generator curves first go up, then do down and gradually become stable. We stop training when both discriminator and generator models become stable. We sampled 200K compounds from the LatentGAN model and compared with the 100K ChEMBL training compounds in PCA plot to examine the coverage of their chemical space. The MQN descriptor set was generated for compounds in both sets and the top two principal components (the explained variance is around 74.70%) was demonstrated in Figure 4a. It can be seen that both compound sets cover the similar chemical space. Among the sampled set, 77% compounds are valid (Table 2) and 99.2% of them are unique. Also comparing with the training set, 99.6% of those unique compounds are actually novel structures.

The MOSES[30] dataset was used to benchmark the performance of LatentGAN with some known structural generative models. A LatentGAN model was first trained on the 1.6 million compounds in the MOSES training set and 50,000 compounds were then sampled. Several metrics were generated based on the sampled set and the MOSES reference set (around 176k compounds) using the MOSES code[31] and the comparison results are shown in Figure 3. The detailed explanation of each metric can be found in reference 29. Metrics of four different known generative models and the training set were also listed for comparison and their values were directly taken from the reference 29. The validity of the LatentGAN is

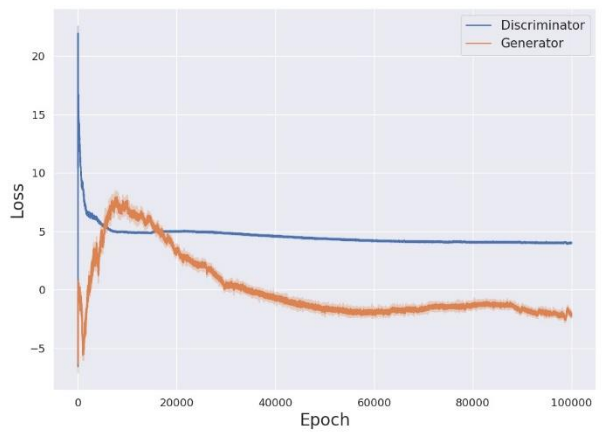
0.84 and it is lower than other methods due to probably the noise introduced in the latent vectors. For most of other metrics, the performance of LatentGAN is on-par to other models. Particularly in terms of the internal diversity, LatentGAN performs best.



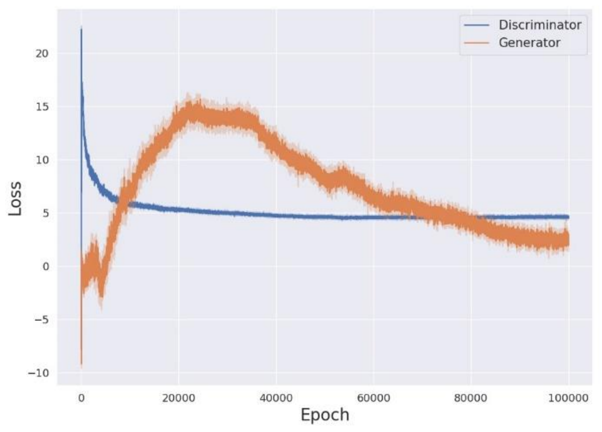
(a)



(b)



(c)



(d)

Figure 2. The loss curve during training for (a) ChEMBL, (b) EGFR, (c) HTR1A and (d) S1PR1 models.

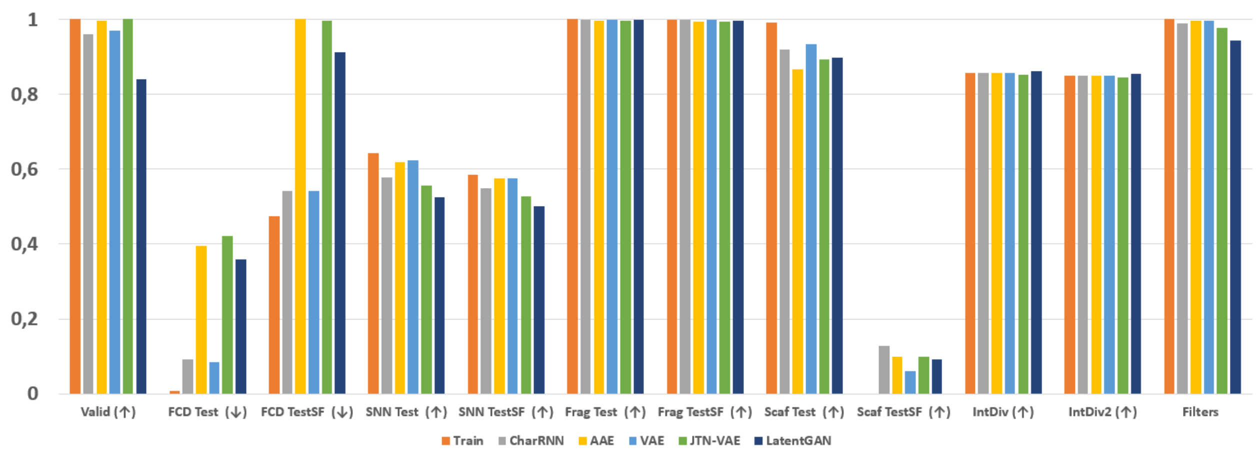


Figure 3. Comparison of different generative models on MOSES dataset. The upward arrow in the x-axis means the higher the metric value is, the more desirable the metric value is and the downward arrow means vice versa.

Training LatentGAN on biased dataset. Another interesting question to answer is if the LatentGAN can be trained to generate target specific/biased compounds. Three targets EGFR, HTR1A and S1PR1 were selected (Table 1) as the test cases. For evaluating target activity of GAN generated compounds. SVM models were built for the three targets using the training set and the model performance on the test set was shown in Table 1. It can be seen that all three models in general have good performance on the test sets. The active compounds of training set were then used as the true data to train the LatentGAN. Previous studies have used GAN architecture to train molecule generative models, such as ORGANIC[15], RANC[17]. But in their methods, SMILES strings were used directly as the input for both generator and discriminator and their output is also SMILES strings. One drawback of their methods is that a RNN model is needed in the GAN to embed the input SMILES to latent vectors during the training every time, this may decrease the efficiency of the training. Also, it was reported that the ratio of valid SMILES of ORGANIC can have large variation[15], this demonstrates the issues when training GAN using SMILES. In the LatentGAN method, we choose to use the latent variable of structures as the input and output of the GAN to avoid those limitations. The GAN can focus on training the latent variables and don't need to worry about the structure generation. The structure generation step is done after the GAN training is finished, therefore the training efficiency has been improved. The validity of the generated SMILES is largely bound by the quality of pretrained heteroencoder, but can be lower if latent space vectors that doesn't represent molecules are sampled.

The loss curves of GAN training can be seen in Figure 2, each GAN model was trained 10,000 epochs. Once the GAN training is done, 50,000 compounds were sampled from generator and decoded from heteroencoder. Their target activity was predicted by the SVM models. The overall performance can be seen in Table 2. Among the 50,000 sampled compounds, the ratio of valid SMILES were all above 80% and among the valid smiles, the uniqueness of the compound was 56%, 66% and 31% for EGFR, HTR1A and S1PR1 respectively. Comparing with the sample set of ChEMBL model, these numbers are much lower. It is probably due to the smaller size of training set, as the S1PR1 set has the smallest training set and the uniqueness of its sample set is also the smallest. The RNN models have higher percentage of validity, but their percentage of uniqueness is in general lower. Comparing with the GAN training set, the novelty for EGFR, HTR1A and S1PR1 are 97%, 95% and 98% respectively and it is quite similar to that of RNN models. This demonstrates that LatentGAN can not only generate valid SMILES but also most of them are novel to the training set, which is very important for de novo design task. All the sampled valid smiles were then predicted by the SVM models. It can be seen in Table 2 that high percentage of sampled compounds were predicted as active for these three targets (71%, 71% and 44% are for EGFR, HTR1A and S1PR1 respectively) and it is the same for the RNN models. The comparison between LatentGAN and RNN generated active (predicted by SVM model) structures was displayed in Figure 4. It seems that on the full structure and scaffold level, the overlap between LatentGAN and RNN active sets is very small, which highlights that both types of model are complementary to each other for structure generation.

Table 2 The performance of LatentGAN models

Models	Training set	% Valid	% Unique[a]	% Novelty[b]	% Active[c]	# Recovered test set actives / # total test set actives	# Recovered NN of test set actives[d]
ChEMBL	100,000	77	99.2	99.6	NA	NA	NA
EGFR-LatentGAN	1387	86	56	97	71	69 / 1291	196
EGFR-RNN	1387	96	46	95	65	100/ 1291	238

HTR1A-LatentGAN	3485	86	66	95	71	95 / 1882	284
HTR1A-RNN	3485	96	50	90	81	137/1882	384
S1PR1-LatentGAN	795	89	31	98	44	3 / 323	24
S1PR1-RNN	795	97	35	97	65	12/323	43

Note:

a) The percentage of uniqueness is calculated based on the number of valid compounds
b) The percentage of novelty is calculated based on the number of unique compounds
c) The percentage of actives is calculated based on the unique compounds.
d) The Tanimoto similarity to the test set actives is larger than 0.7.

We also calculated both full compound similarity and Murcko scaffold[32] similarity between actives of sampled set and actives in training set. It can be seen in Figure 5 that in general there are around 5% of generated compounds are identical to the training sets for all three targets. There are around 25%, 24% and 21% compounds having similarity lower than 0.4 to the training set in EGFR, HTR1A and S1PR1 respectively. This means that the LatentGAN can at some extent generates very dissimilar compounds to the training set. In terms of scaffold similarity comparison, it is not surprising that the percentage of identical scaffolds to the training set is much higher for all the targets. Nevertheless, there is still around 15%, 14% and 13% scaffolds in the sample set having low similarity (<0.4) to the training set for EGFR, HTR1A and S1PR1.

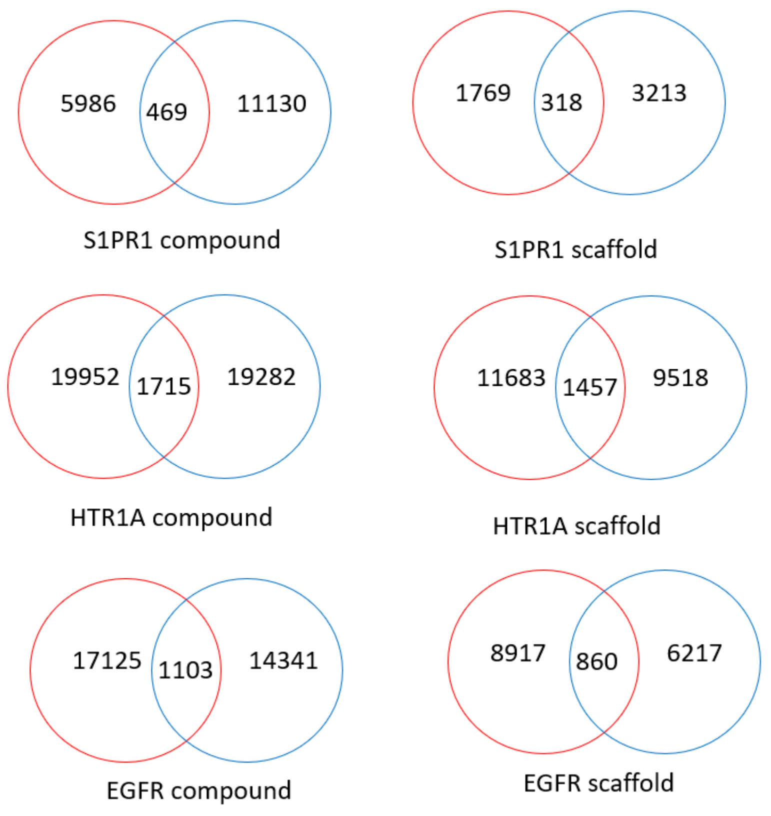
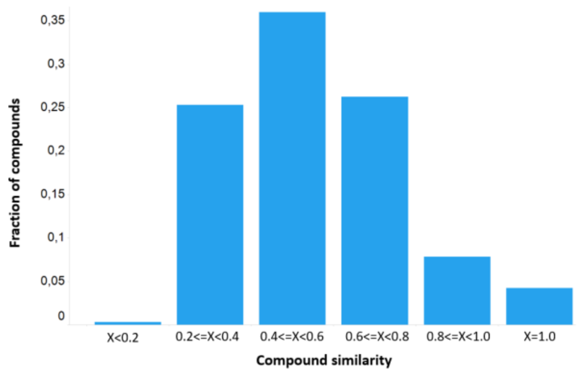


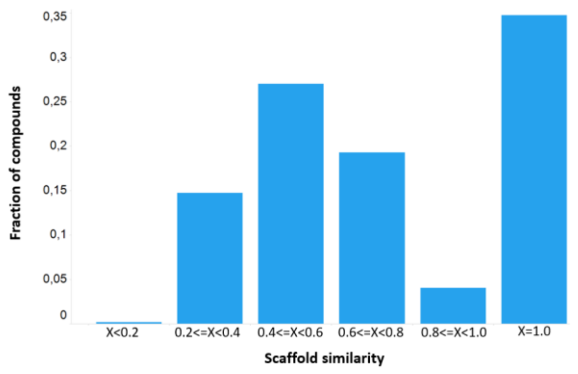
Figure 4. Venn diagram of LatentGAN and RNN generated compounds/scaffolds. The numbers in the red circles and blue circles are from LatentGAN and RNN model respectively. The numbers in the overlap of the two circles are the numbers of overlap structure/scaffold from the two models.

We did PCA analysis to compare the chemical space of sampled sets and training sets for all targets. The PCA analysis is calculated based on the MQN descriptors[33] of the compounds and the explained variance of the top two principal components is 83%, 75% and 79% for EGFR, HTR1A and S1PR1. The PCA plots are shown in Figure 6. It can be seen that the sampled compound sets cover most of the chemical space of the training sets. This means that LatentGAN has done a good job in generating data which can fool the discriminator. Interestingly, there are some regions in the PCA plots where most of the sampled compounds around the training compounds are predicted as inactive, for example the left lower corner in

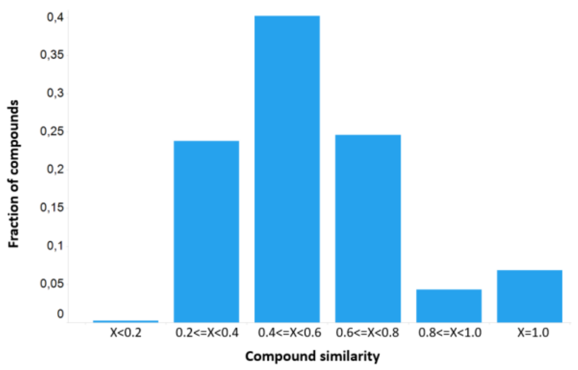
EGFR (Figure 6a) and right-hand side region in S1PR1 (Figure 5c). It was found that the training compounds in those regions are non-druglike compounds and outliers in the training set, SVM models predicted those similar sampled compounds as inactive compounds. Additionally, we also check how much of the actives in the test set are recovered by the sample set. The results are listed in Table 2, it seems that 69 (EGFR), 95 (HTR1A) and 3 (S1PR1) actives in test sets are recovered in the sampled sets and 196 (EGFR), 284 (HTR1A) and 24 (S1PR1) nearest neighbors, whose Tanimoto similarity (based on FCFP6 fingerprint) to the actives of test sets is larger than 0.7, are included in the sampled sets. It is interesting to note that there are more actives of test set recovered by RNN model for all three targets. Based on the analysis of the overlaps between two types of generative model, it could be a viable strategy to apply multiple types of generative model for structure generation. Figure 7 listed some examples generated by the LatentGAN model for three targets. QED score is a popular measurement for compound's drug-likeness and the higher the value is, the more drug-like the compound is. The distribution of QED score for LatentGAN, RNN and training set compounds were compared and displayed in Figure 8. It can be seen that in general the training set compound has a little bit higher fraction at high QED region, though, overall, the distribution of these three sets are quite similar. This highlights that LatentGAN can generate chemically reasonable structures.



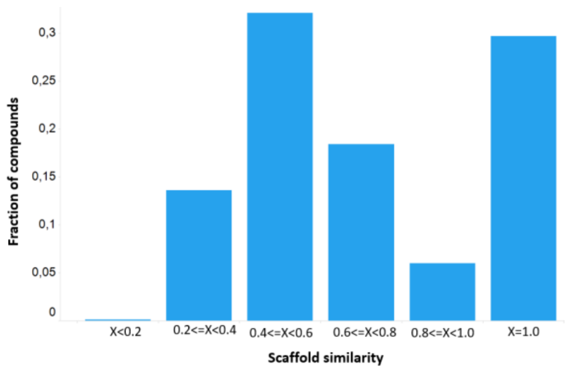
(a)



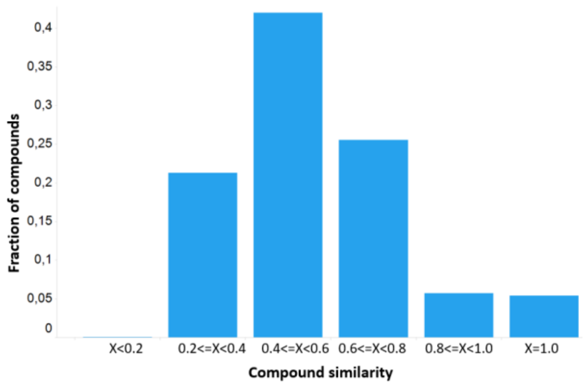
(b)



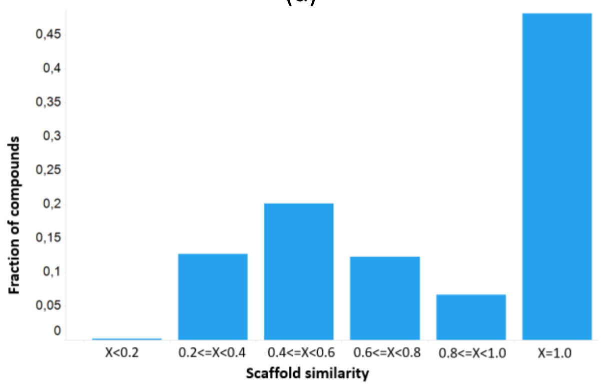
(c)



(d)

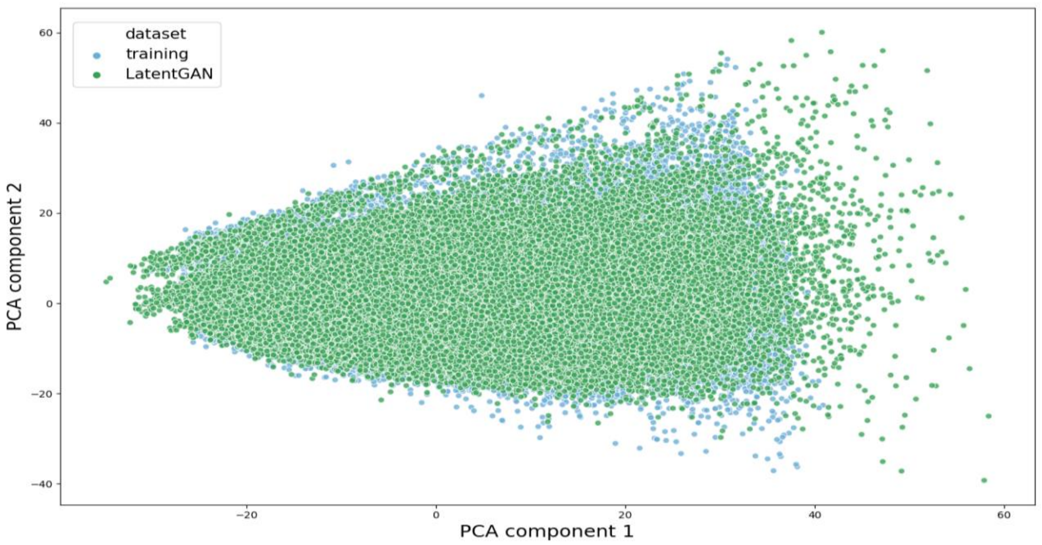


(e)

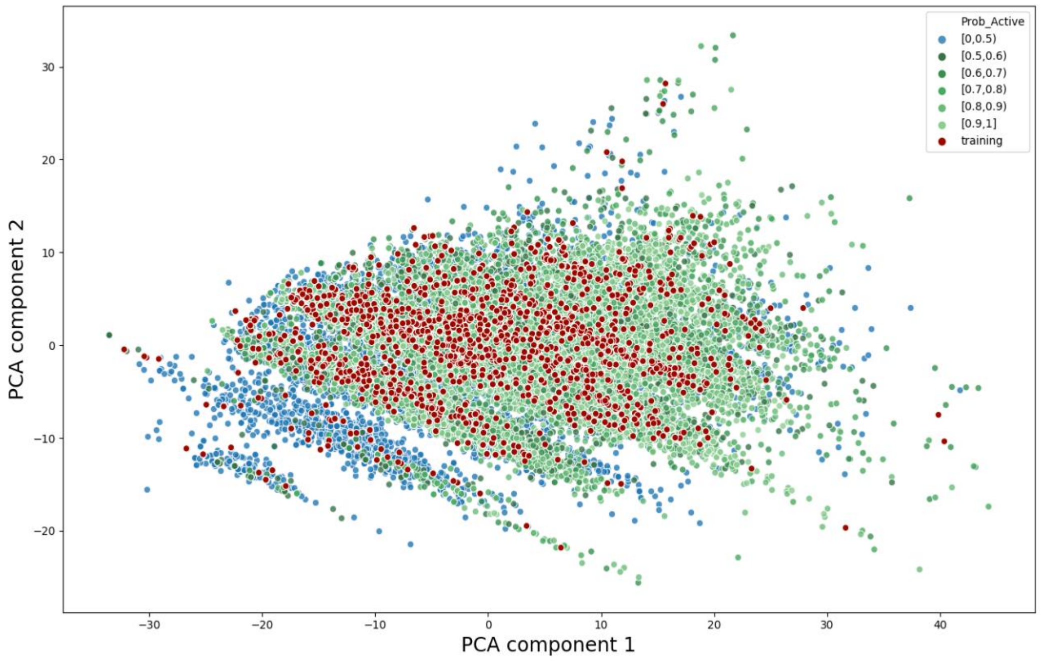


(f)

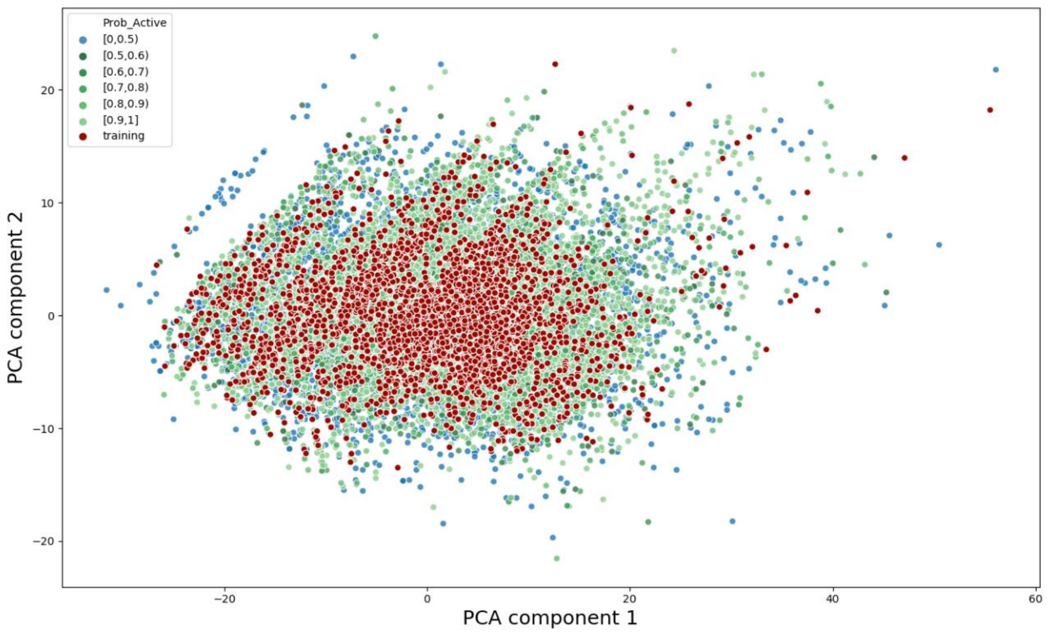
Figure 5. The distribution for (a) EGFR compound similarity, (b) EGFR scaffold similarity, (c) HTR1A compound similarity, (d) HTR1A scaffold similarity, (e) S1PR1 compound similarity, (f) S1PR1 scaffold similarity.


dataset
training
LatentGAN
PCA component 1
PCA component 2

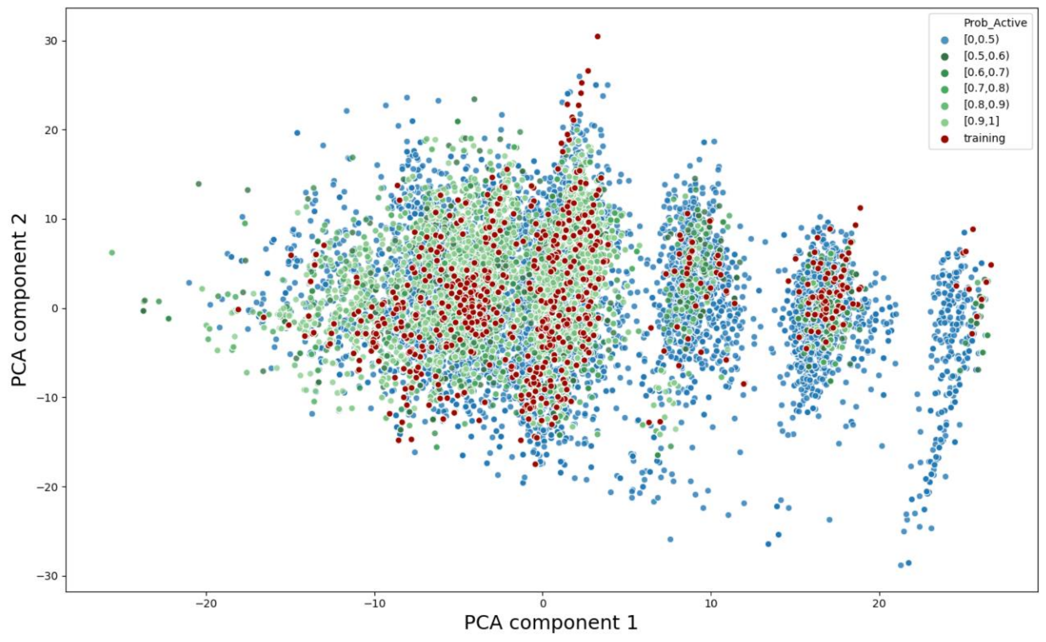
(a)


Prob_Active
[0,0.5)
[0.5,0.6)
[0.6,0.7)
[0.7,0.8)
[0.8,0.9)
[0.9,1]
training
PCA component 1
PCA component 2

(b)



(c)



(d)

Figure 6. PCA analysis for (a) ChEMBL, (b)EGFR, (c) HTR1A and (d) S1PR1 dataset. In figure (a), the green dots are the compounds sampled from the LatentGAN model; In figures b-d, the red dots are the training

set, the blue dots are the predicted inactive compounds in the sampled set and other dots are the predicted actives in the sampled set with different level of probability of being active.

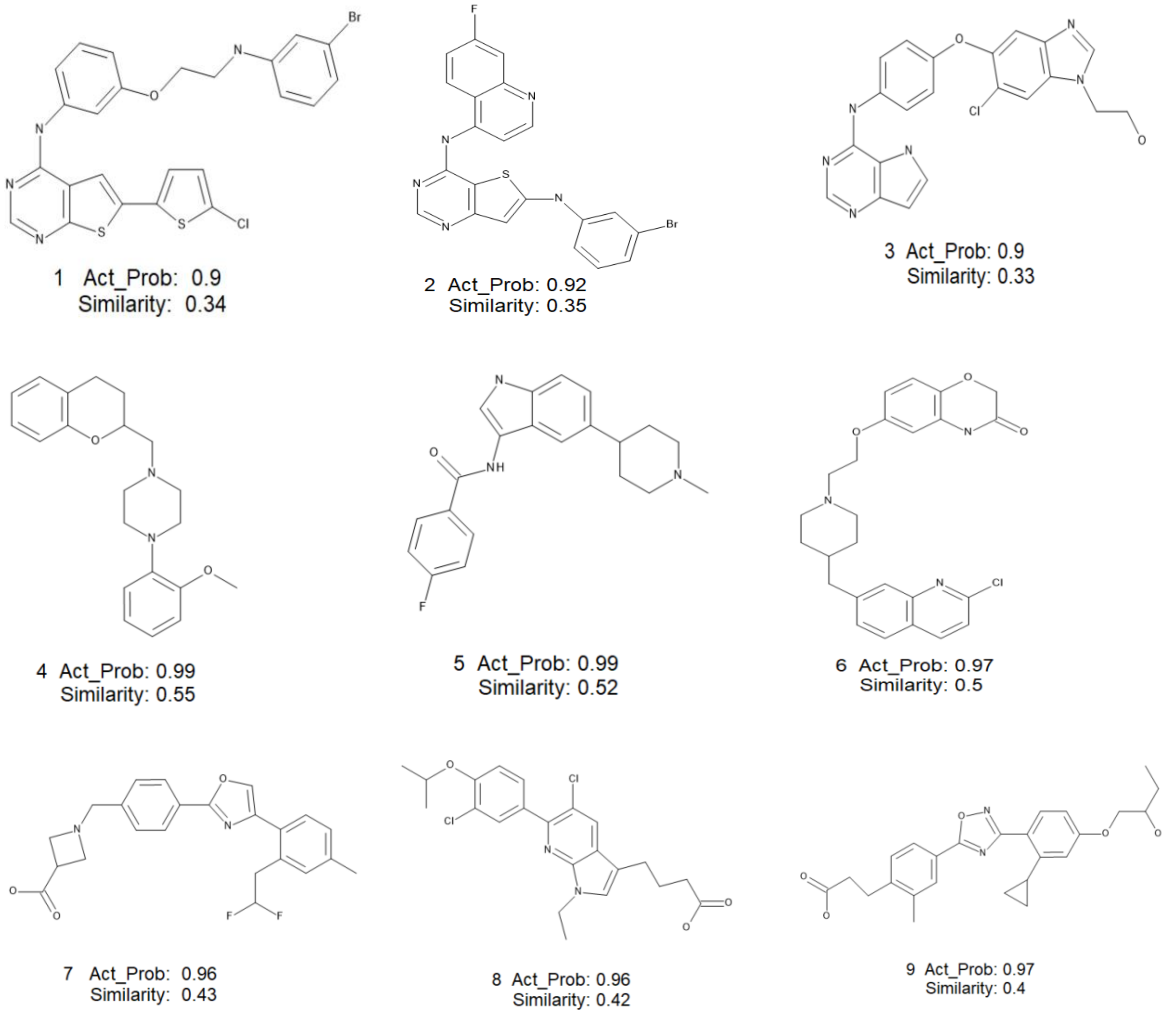
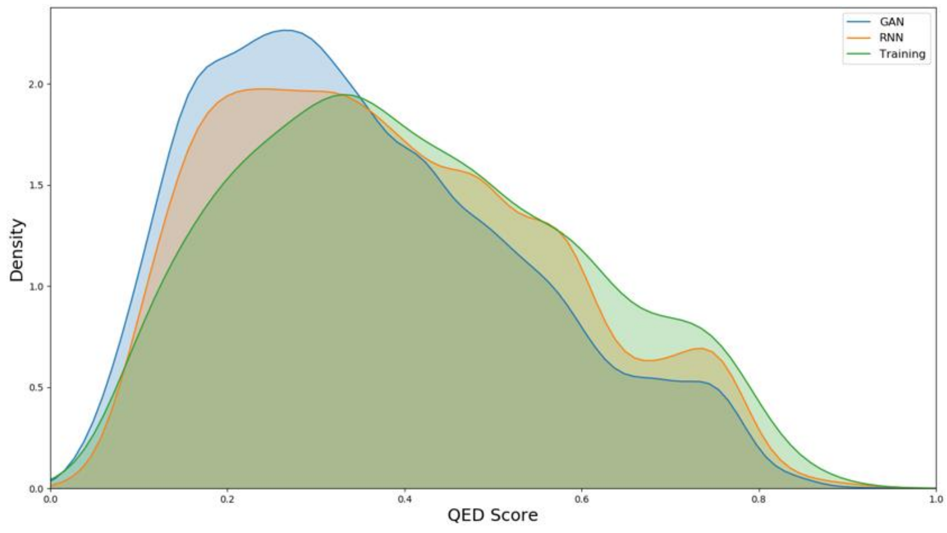
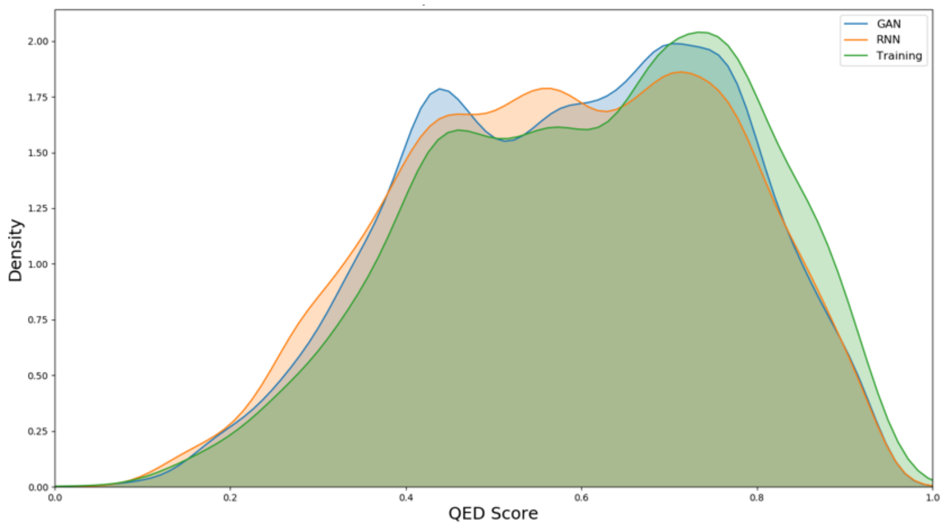

1 Act_Prob: 0.9 Similarity: 0.34
2 Act_Prob: 0.92 Similarity: 0.35
3 Act_Prob: 0.9 Similarity: 0.33
4 Act_Prob: 0.99 Similarity: 0.55
5 Act_Prob: 0.99 Similarity: 0.52
6 Act_Prob: 0.97 Similarity: 0.5
7 Act_Prob: 0.96 Similarity: 0.43
8 Act_Prob: 0.96 Similarity: 0.42
9 Act_Prob: 0.97 Similarity: 0.4

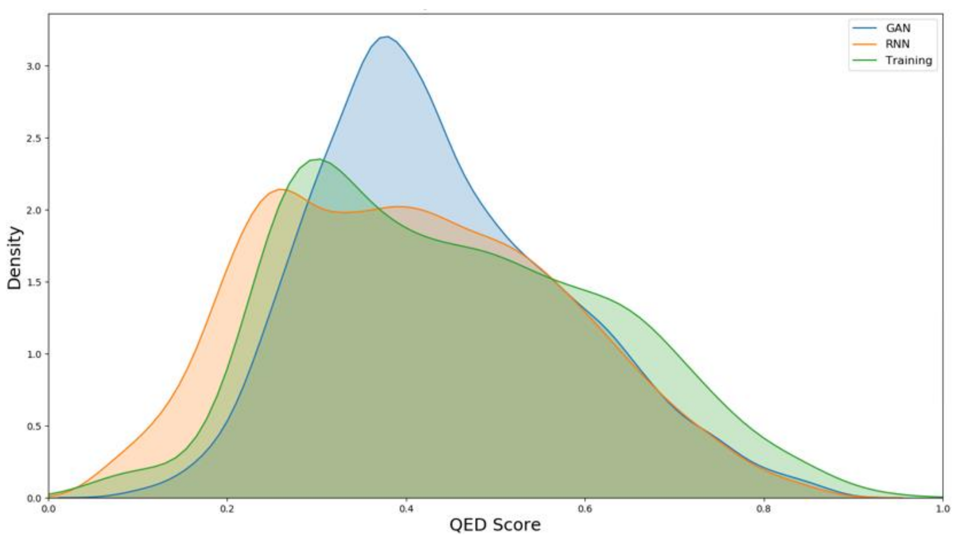
Figure 7. Examples generated by the LatentGANs. Compound 1-3 are generated by the EGFR model, 4-6 are generated by HTR1A model and 7-9 are generated by S1PR1 model.


GAN
RNN
Training
Density
QED Score
0.0
0.5
1.0
1.5
2.0
0.2
0.4
0.6
0.8

(a)


GAN
RNN
Training
Density
QED Score
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0.2
0.4
0.6
0.8

(b)



(c)

Figure 8 The QED distribution of compound set for (a) EGFR (b) HTR1A and (c) S1PR1

Conclusions

A new molecule de novo design method, LatentGAN, was proposed by combining an autoencoder neural network trained as a heteroencoder and a generative adversarial network. In our method, the pretrained autoencoder was used to translate the molecule structure to latent vectors and vice versa and the GAN was trained using latent vectors as input as well as output, all in separate steps. Once the training of GAN was finished, the sampled latent vectors was then mapped back to structures through the decoder of the autoencoder neural network. We first applied this method to train on a subset of ChEMBL compounds and LatentGAN can generate similar drug-like compounds. The benchmark study on MOSES dataset demonstrate that the performance of LatentGAN is on-par to known structural generative models. We later applied the method on three target biased datasets (EGFR, HTR1A and S1PR1) to investigate the capability of the LatentGAN for generating biased compounds. Encouragingly, our results show that most of the sampled compounds from the trained LatentGAN model are predicted to be active to the target which it was trained against and there are substantial portion of the sampled compounds are novel to the training set. Additionally, there are some unseen active compounds in the test set can also be identified in the sampled set. We also compared the structures generated from LatentGAN and the RNN based model for the three targets, it seems that there is very little overlap among the two sets and the two types of model can be complementary to each other. In summary, these results show that LatentGAN can be a valuable tool for doing de novo molecule design.

References

1. Chen, H.; Engkvist, O.; Wang, Y.; Olivecrona, M.; Blaschke, T., The rise of deep learning in drug discovery. *Drug Discov Today* **2018,** *23(6)*, 1241-1250.

2. Chen, H.; Kogej, T.; Engkvist, O., Cheminformatics in Drug Discovery, an Industrial Perspective. *Mol Inform* **2018,** *37(9-10)*, e1800041.

3. Ekins, S., The Next Era: Deep Learning in Pharmaceutical Research. *Pharm Res* **2016,** *33(11)*, 2594-603.

4. Gawehn, E.; Hiss, J. A.; Schneider, G., Deep Learning in Drug Discovery. *Mol Inform* **2016,** *35(1)*, 3-14.

5. Hessler, G.; Baringhaus, K. H., Artificial Intelligence in Drug Design. *Molecules* **2018,** *23(10)*, pii: E2520.

6. Schneider, G.; Geppert, T.; Hartenfeller, M.; Reisen, F.; Klenner, A.; Reutlinger, M.; Hahnke, V.; Hiss, J. A.; Zettl, H.; Keppner, S.; Spankuch, B.; Schneider, P., Reaction-driven de novo design, synthesis and testing of potential type II kinase inhibitors. *Future Med Chem* **2011,** *3* (4), 415-24.

7. Bickerton, G. R.; Paolini, G. V.; Besnard, J.; Muresan, S.; Hopkins, A. L., Quantifying the chemical beauty of drugs. *Nature chemistry* **2012,** *4* (2), 90-98.

8. Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular De Novo Design through Deep Reinforcement Learning *ArXiv e-prints* [Online], 2017. arXiv:1704.07555.

9. Segler, M. H. S.; Kogej, T.; Tyrchan, C.; Waller, M. P. Generating Focussed Molecule Libraries for Drug Discovery with Recurrent Neural Networks *ArXiv e-prints* [Online], 2017. arXiv:1701.01329 .

10. Arus-Pous, J.; Blaschke, T.; Ulander, S.; Reymond, J. L.; Chen, H.; Engkvist, O., Exploring the GDB-13 chemical space using deep generative models. *J Cheminform* **2019,** *11* (1), 20.

11. Gomez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernandez-Lobato, J. M.; Sanchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A., Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent Sci* **2018,** *4* (2), 268-276.

12 Popova, M.; Isayev, O.; Tropsha, A., Deep Reinforcement Learning for de Novo Drug Design. *Sci. Adv.* **2018**, *4* (7), eaap7885.

13. Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y., Generative Adversarial Networks. **2014**.

14. Lima Guimaraes, G.; Sanchez-Lengeling, B.; Outeiral, C.; Cunha Farias, P. L.; Aspuru-Guzik, A. Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models *arXiv e-prints* [Online], 2017. arXiv:1705.10843.

15. Benjamin, S.-L.; Carlos, O.; Gabriel L., G.; Alan, A.-G., *Optimizing distributions over molecular space. An Objective-Reinforced Generative Adversarial Network for Inverse-design Chemistry (ORGANIC)*. 2017. DOI: 10.26434/chemrxiv.5309668.v3

16. Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J., Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv Drug Deliv Rev* **2001**, *46* (1-3), 3-26.

17. Putin, E.; Asadulaev, A.; Ivanenkov, Y.; Aladinskiy, V.; Sanchez-Lengeling, B.; Aspuru-Guzik, A.; Zhavoronkov, A., Reinforced Adversarial Neural Computer for de Novo Molecular Design. *J Chem Inf Model* **2018**, *58* (6), 1194-1204.

18. Putin, E.; Asadulaev, A.; Vanhaelen, Q.; Ivanenkov, Y.; Aladinskaya, A. V.; Aliper, A.; Zhavoronkov, A., Adversarial Threshold Neural Computer for Molecular de Novo Design. *Mol Pharm* **2018**, *15* (10), 4386-4397.

19. Graves, A.; Wayne, G.; Reynolds, M.; Harley, T.; Danihelka, I.; Grabska-Barwinska, A.; Colmenarejo, S. G.; Grefenstette, E.; Ramalho, T.; Agapiou, J.; Badia, A. P.; Hermann, K. M.; Zwols, Y.; Ostrovski, G.; Cain, A.; King, H.; Summerfield, C.; Blunsom, P.; Kavukcuoglu, K.; Hassabis, D., Hybrid computing using a neural network with dynamic external memory. *Nature* **2016**, *538* (7626), 471-476.

20. Li, J.; Cai, D.; He, X. Learning Graph-Level Representation for Drug Discovery *ArXiv e-prints* [Online], 2017, arXiv:1709.03741.

21. Bjerrum, E. J.; Sattarov, B., Improving Chemical Autoencoder Latent Space and Molecular De Novo Generation Diversity with Heteroencoders. *Biomolecules* **2018**, *8* (4), pii:E131.

22. Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B.; Overington, J. P., ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Res.* **2012,** *40* (Database issue), D1100-D1107.

23. Bejerrum, E. J. SMILES Enumeration as Data Augmentation for Neural Network Modeling of Molecules. *arXiv e-prints* [Online], 2017. arXiv:1703.07076

24. Guljarani, I; Ahmed, F; Arjovsky, M; Dumoulin, V; Courville, A., Improved training of Wasserstein GANs. *arXiv e-prints* [Online], 2017. arXiv:1704.00028

25. Luo, Y.; Lu, B. L., EEG Data Augmentation for Emotion Recognition Using a Conditional Wasserstein GAN. *Conf Proc IEEE Eng Med Biol Soc* **2018**, *2018*, 2535-2538.

26. Sun, J.; Jeliazkova, N.; Chupakhin, V.; Golib-Dzib, J.-F.; Engkvist, O.; Carlsson, L.; Wegner, J.; Ceulemans, H.; Georgiev, I.; Jeliazkov, V.; Kochev, N.; Ashby, T. J.; Chen, H., ExCAPE-DB: an integrated large scale dataset facilitating Big Data analysis in chemogenomics. *J Cheminf* **2017**, *9* , 17.

27. Industry-scale Application and Evaluation of Deep Learning for Drug Target Prediction, to be submitted.

28. Scikit-learn machine learning package. https://scikit-learn.org/stable/.

29. RDKit: Open-source cheminformatics; http://www.rdkit.org.

30. Polykovskiy,D.; Zhebrak, A.; Sanchez-Lengeling, B.; Golovanov, S.; Tatanov, O.; Belyaev, S.; Kurbanov, R.; Artamonov, A.; Aladinskiy, V.; Veselov, M.; Kadurin, A.; Nikolenko, S.; Aspuru-Guzik, A.; Zhavoronkov, A., Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. *arXiv e-prints* [Online], 2018. arXiv:1811.12823

31. MOSES code. https://github.com/molecularsets/moses

32. Bemis, G. W.; Murcko, M. A., The properties of known drugs. 1. Molecular frameworks. *J Med Chem* **1996**, 39(15), 2887-93.

33. Nguyen, K. T.; Blum, L. C.; van Deursen, R.; Reymond, J. L., Classification of organic molecules by molecular quantum numbers. *ChemMedChem* **2009,** *4* (11), 1803-5.