molecular pharmaceutics

Article

pubs.acs.org/molecularpharmaceutics

druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico

Artur Kadurin,*[†,§,∥] Sergey Nikolenko,[‡,§,∥] Kuzma Khrabrov,[⊥] Alex Aliper,[†] and Alex Zhavoronkov*[†,#,¶]

[†]Pharmaceutical Artificial Intelligence Department, Insilico Medicine, Inc., Emerging Technology Centers, Johns Hopkins University at Eastern, Baltimore, Maryland 21218, United States

[‡]National Research University Higher School of Economics, St. Petersburg 190008, Russia

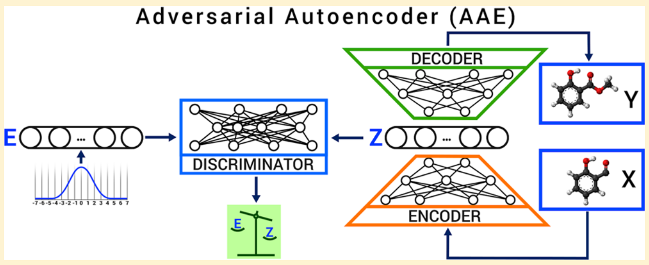
[§]Steklov Mathematical Institute at St. Petersburg, St. Petersburg 191023, Russia

[⊥]Search Department, Mail.Ru Group Ltd., Moscow 125167, Russia

[#]The Biogerontology Research Foundation, Trevissome Park, Truro TR4 8UN, U.K.

[¶]Moscow Institute of Physics and Technology, Dolgoprudny 141701, Russia

[∥]Kazan Federal University, Kazan, Republic of Tatarstan 420008, Russia



ABSTRACT: Deep generative adversarial networks (GANs) are the emerging technology in drug discovery and biomarker development. In our recent work, we demonstrated a proof-of-concept of implementing deep generative adversarial autoencoder (AAE) to identify new molecular fingerprints with predefined anticancer properties. Another popular generative model is the variational autoencoder (VAE), which is based on deep neural architectures. In this work, we developed an advanced AAE model for molecular feature extraction problems, and demonstrated its advantages compared to VAE in terms of (a) adjustability in generating molecular fingerprints; (b) capacity of processing very large molecular data sets; and (c) efficiency in unsupervised pretraining for regression model. Our results suggest that the proposed AAE model significantly enhances the capacity and efficiency of development of the new molecules with specific anticancer properties using the deep generative models.

KEYWORDS: *adversarial autoencoder, deep learning, drug discovery, variational autoencoder, generative adversarial network*

■ INTRODUCTION

The productivity of pharmaceutical research and development is declining,[1,2] partly due to an inefficient early lead discovery process which usually screens a large amount of compounds to identify potential leads for subsequent preclinical development.[3] *In silico* based approaches such as deep learning models which are able to generate reliable results at a reduced cost and time scale have served as a promising avenue of efficient screening.[4,5] Actually, deep learning methods have displayed substantial potential in many areas of biomedicine[6] such as biomarker development,[7,8] drug discovery,[4,9,10] predicting clinical trial outcomes,[11] and generating new molecules to a set of defined parameters.[12]

Deep learning methods started with generative models from 2006 to 2009, when deep Boltzmann machines were used for unsupervised pretraining of deep neural networks.[13−15] Shortly after that, unsupervised pretraining became unnecessary for most modern applications of deep learning due to optimization advances such as dropout,[16] batch normalization,[17] and adaptive gradient descent algorithms[18,19] and simply by virtue of making larger supervised data sets available and having more computational power to process them.

However, in the past two years there has been a revival of generative models as one of the major themes in deep learning. Apart from direct applications to new sample generation, which have been mostly successful in image processing,[20] there are reasons why generative models can lead to significant improvements in supervised tasks as well. By learning to sample data points that are similar to the training data, generative models can serve as very powerful feature extractors which can capture the essence of the underlying data set. Recent works suggest that generative models can serve as a basis for one-shot learning, i.e., learning new concepts with very few training examples based on previous general "understanding" of the data set.[21] Some researchers believe that generative models are one of the most promising approaches to the "common knowledge problem", i.e.,

Received: April 27, 2017
Revised: July 7, 2017
Accepted: July 13, 2017
Published: July 13, 2017

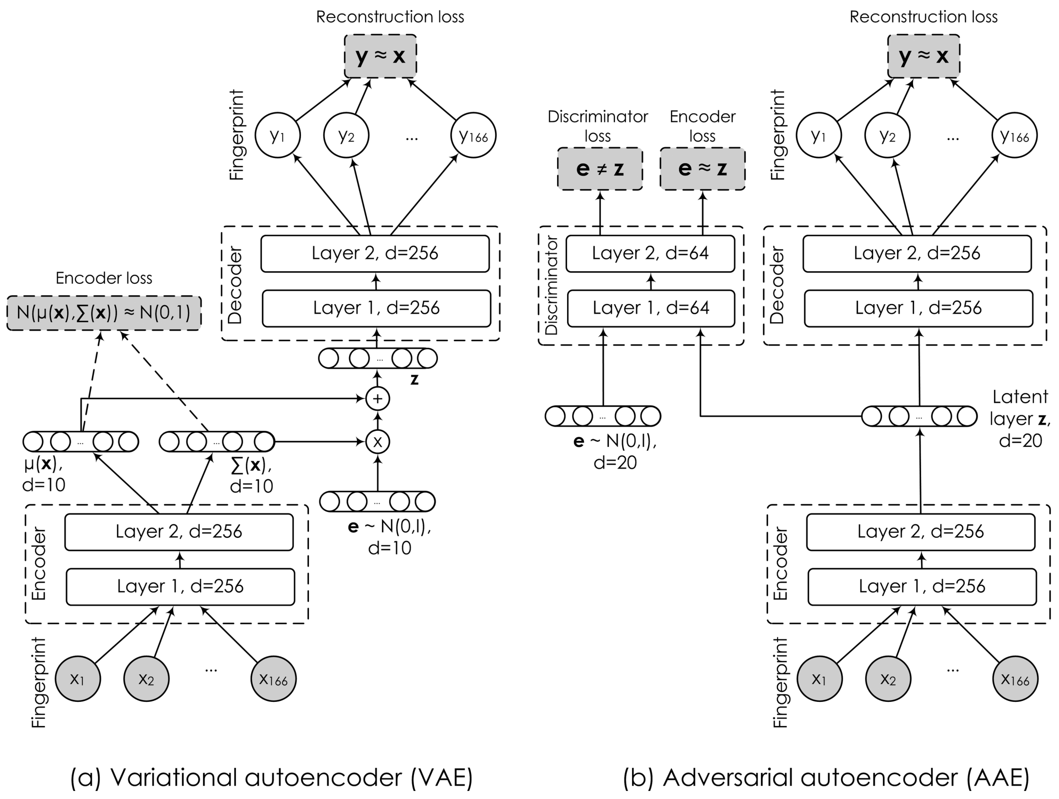


Figure 1. Architecture comparison between VAE and AAE. (a) Architecture of VAE. (B) Architecture of AAE.

making a model able to collect and process generic data about its environment like humans do.

Currently, the most popular generative models based on deep neural architectures include the variational autoencoder (VAE)[22−24] and generative adversarial network (GAN).[25,26] Our group previously presented a successful pharmaceutical application of generative models[12] using another generative model: the adversarial autoencoder (AAE),[27] which was derived from the GAN architecture. In that work, AAE was used to identify and generate new chemical compounds with predefined properties. In particular, the AAE learned to sample from a data set of molecular fingerprints augmented with their concentrations. Samples drawn from the trained model were screened against a database of known compounds, and it was found that a large percentage of them corresponded to known or suspected anticancer agents. That work also suggested that newly sampled untested molecules may be worth studying as potential anticancer agents.

However, the problem solved in our previous work was relatively small and limited, that is, the AAE was trained on a small data set with only 6252 available compounds profiled on the MCF-7 cell line. While it is positive that such a small data set can produce reasonable new samples with the AAE model, it would be a more striking result if improved deep generative models could capture the structure of large molecular databases. In this work, we present an adapted AAE model called druGAN (drug Generative Adversarial Network), and compare the capacity and efficiency between AAE and VAE with regard to molecular feature extraction problems in large data sets.

■ RESULTS

Structure of Inputs and Networks. Molecules are represented by binary vectors of dimension 166, referred to as molecular fingerprints, where each bit represents a certain chemically relevant feature. The *Tanimoto* similarity, which in this context is a synonym of *Jaccard* similarity, is believed to be a chemically reasonable measure of similarity between molecules. Hence, we used *Tanimoto* similarity to evaluate the models in our experiments. We train VAE and AAE as autoencoders based on the fingerprints.

Implementation of the AAE Model. Figure 1 shows the final structure of VAE and AAE, where shaded rectangles with dashed borders indicate components of the cost function minimized during training. Note that VAE directly trains the latent layer to match the standard normal distribution while AAE trains a separate (smaller) network, the discriminator, to distinguish between latent variables and the standard *Gaussian*. Each plate marked as "Layer ..., *d* = 256" consists of a fully connected layer of the specified dimension.

We have introduced an important modification to the default AAE architecture. In the GAN and DCGAN networks, batch normalization layers are essential. However, our experiments showed that adding batch normalization to discriminator layers in an AAE resulted in a mismatch between the discriminator and the generator, and we were not able to tune other parameters to fix this problem, suggesting that the standard batch normalization made the discriminator completely useless. In our opinion, this can happen because batch normalization layers in the discriminator mask the noise produced by the generator into target random noise, and the discriminator does not train at all. Therefore, we tried to use different normalization weights for positive and negative examples. In this case the opposite happened: the discriminator easily learned to distinguish between positive and negative examples, and the generator had absolutely no chance to fool it. This is due to an overly powerful normalization layer: for example, it suffices for the discriminator to train normalization with zero variance that sends all positive examples to one point and all negative examples to another.

After we removed BN from the discriminator, the same effect happened with the encoder: the scale of the first BN layer trained to be very close to zero, and the autoencoder did not work at all. This may require further study in other applications, but the solution that worked best in this molecular modeling task was to simply remove all batch normalization layers from the AAE architecture. Interestingly, we saw the same effect for the VAE model: batch normalization layers made the overall quality only worse. We believe this is an important result, as it shows that layers such as batch normalization, which usually do work well and improve the results, still can make things worse, sometimes even for principled reasons (as in the AAE case above), and it is worthwhile to investigate different architectures. However, due to fair model comparison on the regression task, batch normalization should be removed from the VAE network.

Note that while the entire resulting neural network was rather deep (7 layers in both VAE and AAE), the encoder, decoder, and discriminator (in AAE) networks had only two layers each; this architecture, which was also used in our previous work, follows the original AAE paper; for further work it would be interesting to experiment with deeper architectures.

In our previous work, AAE training was conducted in three phases, in each of which we used one minibatch to train first the discriminator, then the encoder, and then the autoencoder. However, in our experiments the scheme with alternating updates of the corresponding weights led to instability in the network.

We observed two main problems:

1. The competition between encoder and discriminator could, at an arbitrary point of the training, lead to a complete victory of one of the subnetworks whose error tended to zero. The situation could also become stable again and then again converge to the domination of one subnetwork.
2. When the encoder and discriminator were balanced in a draw, the autoencoder error could oscillate between the original random value and minimal value during the entire training.

Therefore, we changed the approach of AAE training through adding two conceptual novelties:

1. The experiments showed that we could stabilize reconstruction error by uniting the second and third phases into one, and thus, in the new model, the first phase remained the same while in the second phase we updated network weights according to the total error of generator and autoencoder.
2. To stabilize the errors of generator and discriminator, we used a training strategy with a hyperparameter p, which corresponded to the desired "discriminative power". Moreover, at each training step, if the discriminator correctly labeled samples generated by the generator with probability less than p, we trained the discriminator, otherwise the generator and autoencoder (together, as shown above). Thus, we strived to always preserve discriminator and generator errors on a certain level.

In our experiments, we made the assumption that the discriminator, as a "teacher" for the generator, must work better than a random classifier. Therefore, we compared different values of p in one of the experiments and then used $p = 3/5$ in the experiments below. In our study of the different values of p, we saw that the better the discriminator was, the worse the autoencoder was, but the coverage of the training/test samples with samples from the generator became better. With these two changes added to the training algorithm, we stabilized the AAE training and introduced a new hyperparameter that captured the trade-off between coverage of the original data set with generated samples and reconstruction quality.

Validation. In addition to experiments designed to test sample quality and compare between AAE and VAE, we demonstrated the experimental results suggesting that the deep generative models are able to produce features useful for subsequent supervised learning problems. To generate the experimental data set, we used the PubChem database of substances,[28] which contained more than 72 million molecules. We used the RDKit open-source chemoinformatics library (available from http://www.rdkit.org) to convert the SMILES string provided by PubChem into 166-bit Molecular ACCess System (MACCS) chemical fingerprints. These bit vectors comprised the data set that AAE and VAE were trained on. We compared VAE and AAE based on three sets of evaluation metrics.

In the first experiment, we compared the two models as autoencoders, with reconstruction errors on a held-out randomly sampled test set (1/1000 of the data set). The errors, plotted against training epochs, are shown in Figure 2. The VAE model

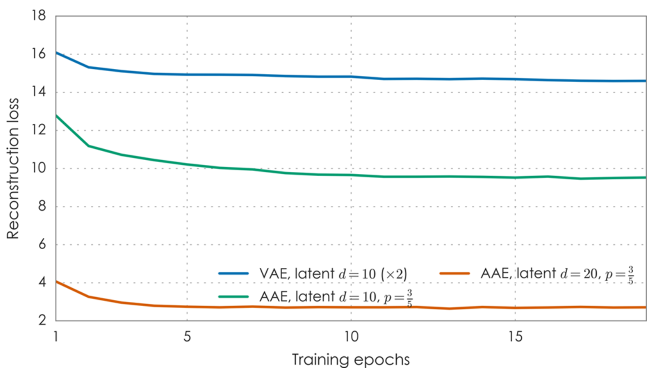


Figure 2. Comparison of reconstruction loss between VAE and AAE.

used in this experiment transforms samples from a normal distribution of dimension 10, but its latent layer has dimension 20 since its mean and variance are learned independently. We compared it against the proposed AAE model with latent layers of dimensions d = 10 and d = 20. The results (validated by multiple training runs) indicate that AAE performs much better in reconstruction than VAE and AAE with d = 20 far outperforms VAE.

In the second experiment, we compared VAE and AAE as generative models for the molecule vectors. We propose to evaluate sampling quality with coverage, i.e., share of the original data set that can be successfully sampled from the model. Specifically, for each model we (1) sampled 10,000 fingerprints from the model (note that these samples are probability estimates for each bit in the fingerprint); (2) 50 times randomly chose 1000 of them and sampled specific boolean vectors; and (3) counted how many samples from the test data set (approximately 70,000 molecules) had *Tanimoto* similarity ≥0.8 with the first vector, then with one of the first two vectors, and so on up to 1000 vectors. Figure 3 shows the mean and standard deviation for these coverage values. It is observed that in

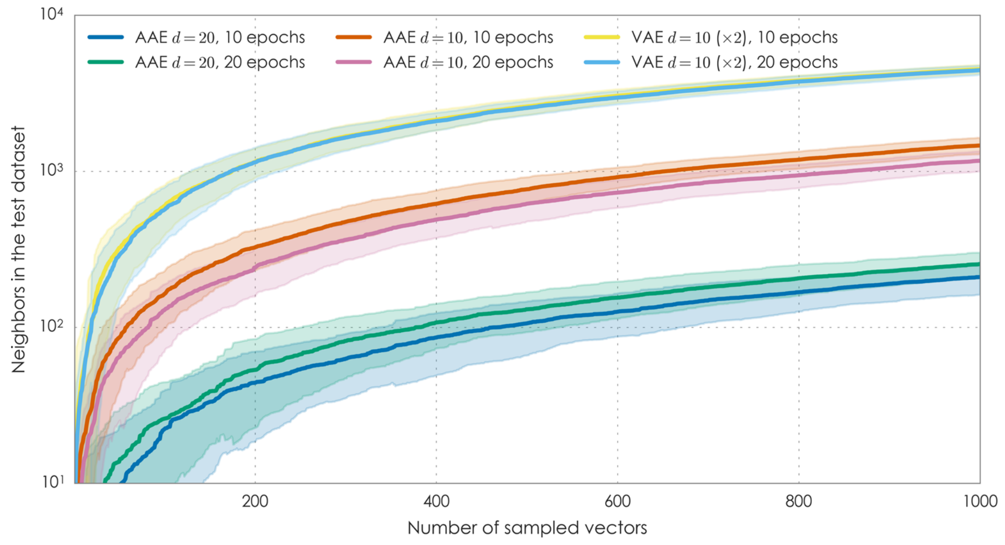


Figure 3. Comparison of coverage between VAE and AAE.

terms of coverage, i.e., for sampling diverse molecular fingerprints, the VAE model outperforms the AAE model with $p = \frac{3}{5}$.

We also used these two experimental settings to compare AAE models among different values of p. Figures 4 and 5 show the reconstruction loss and coverage plots for four different values of p with the same AAE model. There is a clear trade-off as expected: higher values of p make the discriminator stronger, so the reconstruction loss becomes larger; on the other hand, higher values of p present more of a challenge for the generator, so the sampling coverage improves. In other experiments, we used $p = \frac{3}{5}$ as a trade-off between these two objectives.

In the third experiment, we used the generative models as unsupervised feature extractors for further classification/regression. For this purpose, we used a well-established aqueous solubility data set consisting of 1144 compounds.[29] To use the features, we took the first two layers of each autoencoder architecture and added a new layer of linear regression on these features. The resulting neural network was then trained in a supervised way on the solubility data set, so the original features discovered by autoencoders served as unsupervised pretraining for the supervised regression network. The resulting mean absolute error plots are shown in Figure 6, which shows MAE on the test set with the shaded area as variances across 10 different runs. All autoencoders in our study produced satisfactory initial

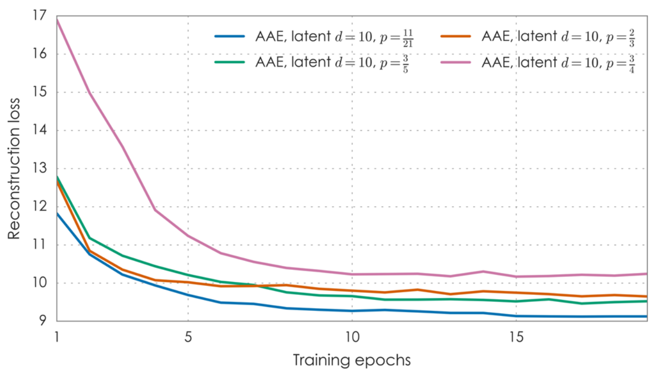


Figure 4. Comparison of reconstruction loss among AAE under different parameter settings.

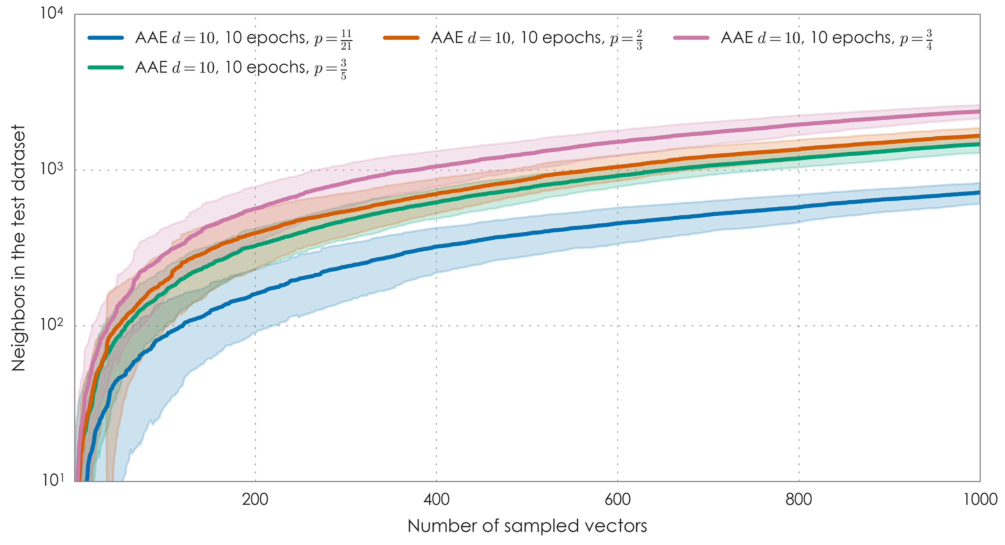


Figure 5. Comparison of coverage among AAE under different parameter settings.

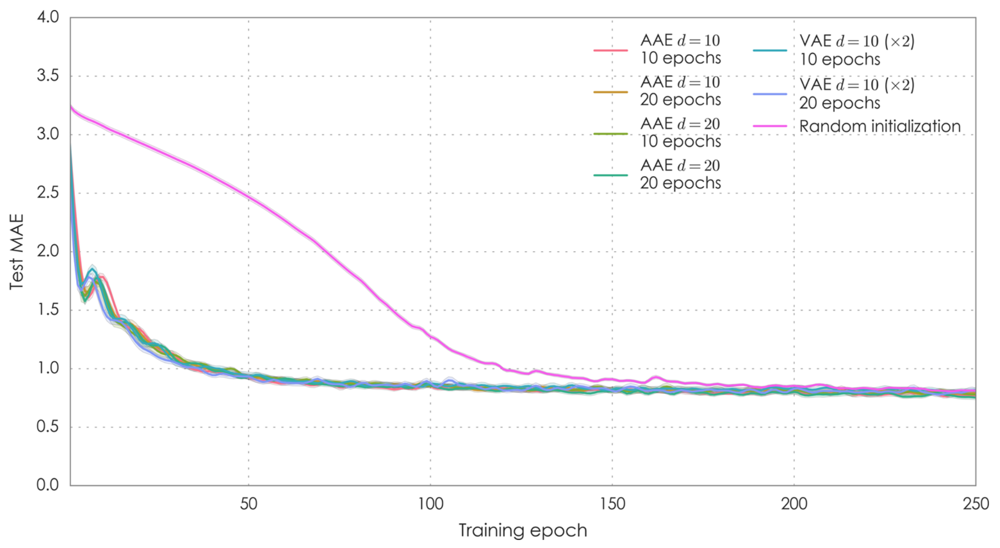


Figure 6. Comparison of mean absolute error between VAE and AAE.

Table 1. Summary of Experimental Validations Based on Comparisons between AAE and VAE

	performance summarization		
model	reconstruction	generation	feature extraction
AAE	(a) AAE with the same architecture is significantly better.	(a) Our improvements uncover a trade-off between coverage and reconstruction quality; moreover, potentially we can achieve better reconstruction with equal coverage.	(a) Approximately the same improvement compared to random network initialization as with VAE.
	(b) Reconstruction error: 9.52.	(b) Coverage: depends on the hyperparameter.	(b) Convergence: after 1 epoch instead of ~5 for random initialization.
VAE	(a) Reconstruction error: 14.60.	(a) Basic configuration is stable and covers many more test molecules than AAE.	(a) Approximately the same improvement compared to random network initialization as with VAE.
		(b) Coverage: 4500.	(b) Convergence: after 1 epoch instead of ~5 for random initialization.

approximations for the regression problem, and the pretrained network converged much faster than the same network with randomly initialized weights using Xavier initialization. Moreover, the horizontal axis on Figure 6 shows training minibatches with 32 training examples each rather than training epochs. This means that all of the models have already converged after a first pass over the data, and the advantage of pretrained models is very significant, as they converge not just faster than with random initialization but with less data.

We provide a summary of experimental validations in Table 1. These experimental validations have suggested that AAE outperforms VAE and significantly enhanced the capacity and efficiency of deep learning models with regard to applications to oncological molecular development.

Limitation of the AAE Approach for Generating New Molecular Structures. The stated objectives of this research study were to improve previous results for generating molecules with AAE and stabilize its training process. To provide qualitative evaluation of the proposed technique we compare the new AAE to another popular architecture: VAE. Comparison with the other modern approaches for predicting solubility should be considered in the future studies that will involve intense search in hyperparameter space, such as number of layers, size of each layer, regularization terms, and learning rates. While the AAE approach may outperform VAE in many tests, VAE represents a valuable and promising alternative that should be used in the drug discovery pipelines on fingerprints as well as on other representations of the molecular structure. There may be applications where VAE outperforms AAE.

While the AAE architecture presented in this paper represents a significant improvement over our previously published work, it has multiple limitations that can be overcome by using a pipeline of several GAN and reinforcement learning architectures and different representations of the molecular structure to be able to generate genuinely novel molecules for specific disease signatures and sets of protein targets. The AAE approach presented in this study uses the MACCS molecular fingerprints, which are not ideal representations of molecular structure. Direct SMILES, InChI, molecular graphs, and other more chemically and biologically relevant representations of the molecular structures may serve as better types of training and discrimination data for the drug discovery pipelines incorporating the various flavors of generative adversarial models. The model serves as a proof of concept and as the example of one application of GANs that can be further improved to generate novel molecules with many more desired features by using better and larger training data sets and expanded into the applications for combination of molecules.

■ DISCUSSION

In this work, we have also found new effects that are important for training adversarial architectures in deep learning. It is clear that in training autoencoders as generative models we constantly encountered a trade-off between reconstruction error and variability of the sampling. For example, in our experiments VAE exhibited the largest reconstruction error together with the best coverage of the test set with generated samples. Our proposed AAE training procedure enabled us to be flexible about setting up this trade-off, and at the same time we saw that, despite significantly better reconstruction, AAE with latent layer dimension 10 and $p = \frac{3}{4}$ virtually did not lose in coverage to VAE. These promising first results warrant further study of these and similar techniques. We have also shown that the trained generative models can be used for further supervised training with very small data sets, virtually one-shot learning.

Aqueous solubility is an important property of a molecule. Low aqueous solubility is directly linked to low bioavailability and is one of the first properties considered during the drug development process.[30] Our experiment with predicting aqueous solubility has shown that a model with unsupervised pretraining can converge several times faster (i.e., with a data set several times smaller) than the same model with random initialization.

■ METHODS

Variational Autoencoder. Variational autoencoder (VAE)[22−24] looks for the generative model

$$p(x) = \int p(x|z;\, \theta)\, p(z)\, \mathrm{d}z$$

where x is a point from the data set, z is the vector of latent variables, and θ are model parameters. VAE adopts the assumption that both $p(x|z;\theta)$ and $p(z)$ are normal distributions,

$$p(x|z;\, \theta) = N(x|f(z;\, \theta),\, \sigma^2 I), \;\; p(z) = N(0,\, I)$$

and then approximates the deterministic function $f(z;\theta)$ with a neural network. To approximate the integral, VAE implements an autoencoder structure to optimize a variational bound on the data distribution $p(x)$, from which VAE learns to sample.

The basic structure and specific architecture of VAE used in this work are shown in Figure 1A. An input x is first transformed with an encoder network into a distribution on latent variables (modeled directly with the mean and variance), then a latent vector z is sampled from that distribution (in reality, VAE samples e from $N(0,I)$ and then transforms it into $z{:}N(\mu(x),\Sigma(x))$ with the corresponding linear transformation), and then it is transformed with a decoder network into a reconstruction y of the original vector x. The VAE loss function thus consists of two parts: VAE tries to make the latent variable distribution close to $N(0,I)$ (minimizing KL distance between them) and at the same time makes y as similar to x as possible (minimizing reconstruction loss). To sample from a trained VAE, one samples the latent vector z from $N(0,I)$ and then transforms it into a sample with the decoder network.

Adversarial Autoencoder. Generative adversarial network (GAN)[25,26] has been recently proposed as a conceptually simple but very efficient adversarial approach to train generative models. The idea is to train two networks at once: the generator G transforms latent vectors z which are sampled from a prior distribution $p(z)$ into the data space, while the discriminator D distinguishes data points which are produced by G from the data points sampled from the actual data distribution. GAN assumes that if G has learned to fool a well-trained D, it probably has actually learned to sample from the data, since any imperfections in the samples can be amplified and used by D. This scheme can be formalized as a game-theoretic problem

$$\min_G \max_D E_{x:p_{\text{data}}}[\log D(x)] + E_{x:p_{\text{data}}}[\log(1 - D(G(z)))]$$

and G and D, both usually modeled as neural networks, can then be trained with alternating stochastic gradient descent.

The structure of an adversarial autoencoder (AAE)[27] used in this work is shown in Figure 1B. AAE applies the adversarial scheme to the basic idea of VAE. An input x is transformed via an encoder network into a vector of latent variables z, and then z is decoded back with the decoder network. At the same time, a separate discriminator network compares the latent vector z with a vector e sampled from $N(0,I)$, and tries discrimination between them. Similar to the regular GAN network, AAE alternates in optimizing the same two loss functions: (1) optimize generator weights with a loss function that tries to "fool" the discriminator into mixing randomly sampled e and latent z produced by the generator; (2) optimize discriminator weights with a loss function that tries to distinguish between e and z.

But AAE also adds to this scheme a third kind of loss function, the reconstruction loss that attempts to reconstruct x as accurately as possible. All three loss functions are optimized one after another: on each training iteration, sample e randomly, then train the discriminator on a minibatch from the training data, then train the encoder on the next minibatch, and finally train the autoencoder on the next. Sampling from a trained AAE also proceeds very similarly to VAE: first sample the latent variables z from $N(0,I)$ and then transform them with the decoder network.

Despite similar mathematical structure and similar practical results in terms of likelihood, current state of the art suggests that GANs are better in actually generating samples for such applications as generating images, which is currently the most common application of both AAE and VAE.[31]

All code was written for Python2.7 using TensorFlow framework. The number of weights in AAE for each subnetwork was as follows:

- Encoder: 113684
- Decoder: 113830
- Discriminator: 5568

Since we used ADAM optimizer to train neural networks, there were two more parameters for each weight of network, so the total count is 699246.

The networks were trained on a custom GPU cluster consisting of 16 NVIDIA Titan Xp graphics cards providing ∼160Tflop of the total computing power.

■ AUTHOR INFORMATION

Corresponding Author

*B301, ETC, JHU, 1101 33rd Street, Baltimore, MD 21218. Phone: (410) 733 5097. E-mail: alex@insilicomedicine.com.

ORCID

Alex Zhavoronkov: 0000-0001-7067-8966

Author Contributions

A.K., A.A., and A.Z. conceived the study, A.K. and K.K. performed the experiments, A.K. and S.N. analyzed the experimental data, and A.K., S.N., and A.Z. wrote the manuscript. All authors have reviewed and agreed to this information before submission.

Notes

The authors declare the following competing financial interest(s): Insilico Medicine is using the Generative Adversarial Networks (GANs) as part of the pipeline for developing new molecules with specific characteristics.

ACKNOWLEDGMENTS

The authors would like to thank NVIDIA for providing the Titan X Pascal graphics processing units (GPUs) for the latest Insilico Medicine GPU cluster and Mark Berger and Andrea de Souza for supporting the research activities at Insilico Medicine. We would like to thank Dr. Yupeng Wang from BDX Research and Consulting for editing the manuscript. The work of A.K. and S.N. was performed according to the Russian Government Program of Competitive Growth of Kazan Federal University and partially supported by the Government of the Russian Federation (Grant 14.Z50.31.0030).

ABBREVIATIONS USED

AAE, adversarial autoencoder; BN, batch normalization; GAN, generative adversarial network; VAE, variational autoencoder; GPU, graphics processing unit

REFERENCES

(1) Munos, B. H.; Chin, W. W. How to revive breakthrough innovation in the pharmaceutical industry. *Sci. Transl. Med.* **2011**, *3* (89), 89cm16.
(2) Mignani, S.; Huber, S.; Tomas, H.; Rodrigues, J.; Majoral, J. P. Why and how have drug discovery strategies in pharma changed? What are the new mindsets? *Drug Discovery Today* **2016**, *21* (2), 239−249.
(3) Yu, M. J. Druggable chemical space and enumerative combinatorics. *J. Cheminf.* **2013**, *5* (1), 19.
(4) Vanhaelen, Q.; Mamoshina, P.; Aliper, A. M.; Artemov, A.; Lezhnina, K.; Ozerov, I.; Labat, I.; Zhavoronkov, A. Design of efficient computational workflows for in silico drug repurposing. *Drug Discovery Today* **2017**, *22* (2), 210−222.
(5) LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521* (7553), 436−444.
(6) Mamoshina, P.; Vieira, A.; Putin, E.; Zhavoronkov, A. Applications of Deep Learning in Biomedicine. *Mol. Pharmaceutics* **2016**, *13* (5), 1445−1454.
(7) Putin, E.; Mamoshina, P.; Aliper, A.; Korzinkin, M.; Moskalev, A.; Kolosov, A.; Ostrovskiy, A.; Cantor, C.; Vijg, J.; Zhavoronkov, A. Deep biomarkers of human aging: Application of deep neural networks to biomarker development. *Aging* **2016**, *8* (5), 1021−1033.
(8) Ozerov, IV; Lezhnina, K. V.; Izumchenko, E.; Artemov, A. V.; Medintsev, S.; Vanhaelen, Q.; Aliper, A.; Vijg, J.; Osipov, A. N.; Labat, I.; West, M. D.; Buzdin, A.; Cantor, C. R.; Nikolsky, Y.; Borisov, N.; Irincheeva, I.; et al. In silico Pathway Activation Network Decomposition Analysis (iPANDA) as a method for biomarker development. *Nat. Commun.* **2016**, *7*, 13427.
(9) Gawehn, E.; Hiss, J. A.; Schneider, G. Deep Learning in Drug Discovery. *Mol. Inf.* **2016**, *35* (1), 3−14.
(10) Aliper, A.; Plis, S.; Artemov, A.; Ulloa, A.; Mamoshina, P.; Zhavoronkov, A. Deep Learning Applications for Predicting Pharmacological Properties of Drugs and Drug Repurposing Using Transcriptomic Data. *Mol. Pharmaceutics* **2016**, *13* (7), 2524−2530.
(11) Artemov, A. V.; Putin, E.; Vanhaelen, Q.; Aliper, A.; Ozerov; Zhavoronkov, A. Integrated deep learned transcriptomic and structure-based predictor of clinical trials outcomes. *bioRxiv* **2016**, DOI: 10.1101/095653.
(12) Kadurin, A.; Aliper, A.; Kazennov, A.; Mamoshina, P.; Vanhaelen, Q.; Khrabrov, K.; Zhavoronkov, A. The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology. *Oncotarget* **2017**, *8* (7), 10883−10890.
(13) Hinton, G. E.; Osindero, S.; Teh, Y. W. A fast learning algorithm for deep belief nets. *Neural computation* **2006**, *18* (7), 1527−1554.
(14) Salakhutdinov, R.; Hinton, G. Deep Boltzmann Machines. *PMLR* **2009**, *5*, 448−455.
(15) Salakhutdinov, R. Learning Deep Generative Models. *Annu. Rev. Stat. Its Appl.* **2015**, *2*, 361−385.
(16) Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929−1958.
(17) Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *PMLR* **2015**, *37*, 448−456.
(18) Kingma, D. P.; Ba, J. L. Adam: A method for stochastic optimization. *arXiv* **2014**, 1412.6980.
(19) Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* **2011**, *12*, 2121−2159.
(20) Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, 1511.06434.
(21) Rezende, D. J.; Mohamed, S.; Danihelka, I.; Gregor, K.; Wierstra, D. One-shot generalization in deep generative models. *arXiv* **2016**, 1603.05106.
(22) Kingma, D. P.; Welling, M. Auto-encoding variational Bayes. *arXiv* **2013**, 1312.6114.
(23) Doersch, C. Tutorial on variational autoencoders. *arXiv* **2016**, 1606.05908.
(24) Rezende, D. J.; Mohamed, S.; Wierstra, D. Stochastic back-propagation and approximate inference in deep generative models. *arXiv* **2014**, 1401.4082.
(25) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, 2672−2680.
(26) Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X. Improved techniques for training GANs. *Adv. Neural Inf Process. Syst.* **2016**, 2226−2234.
(27) Makhzani, A.; Shlens, J.; Jaitly, N.; Goodfellow, I.; Frey, B. Adversarial autoencoders. *arXiv* **2015**, 1511.05644.
(28) Kim, S.; Thiessen, P. A.; Bolton, E. E.; Chen, J.; Fu, G.; Gindulyte, A.; Han, L.; He, J.; He, S.; Shoemaker, B. A.; Wang, J.; Yu, B.; Zhang, J.; Bryant, S. H. PubChem Substance and Compound databases. *Nucleic Acids Res.* **2016**, *44* (D1), D1202−1213.
(29) Delaney, J. S. ESOL: estimating aqueous solubility directly from molecular structure. *Journal of chemical information and computer sciences* **2004**, *44* (3), 1000−1005.
(30) Savjani, K. T.; Gajjar, A. K.; Savjani, J. K. Drug solubility: importance and enhancement techniques. *ISRN Pharm.* **2012**, *2012*, 195727.
(31) Goodfellow, I. NIPS 2016 Tutorial: Generative Adversarial Networks. *arXiv* **2016**, 1701.00160.