molecular pharmaceutics

Article

Cite This: Mol. Pharmaceutics 2018, 15, 4398−4405

pubs.acs.org/molecularpharmaceutics

Entangled Conditional Adversarial Autoencoder for de Novo Drug Discovery

Daniil Polykovskiy,*[,†,‡] Alexander Zhebrak,[†] Dmitry Vetrov,[‡] Yan Ivanenkov,[†,⊥,∥] Vladimir Aladinskiy,[†,∥] Polina Mamoshina,[†] Marine Bozdaganyan,[†] Alexander Aliper,[†] Alex Zhavoronkov,[†] and Artur Kadurin[†,§]

[†]Insilico Medicine, Rockville, Maryland 20850, United States

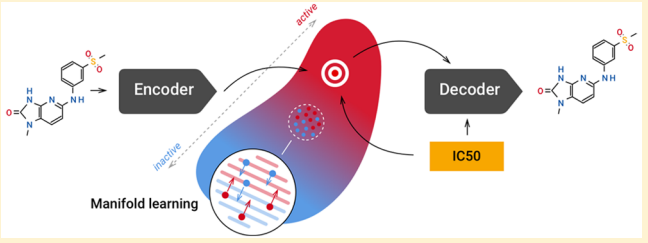
[‡]National Research University Higher School of Economics, Moscow 101000, Russia

[§]Insilico Taiwan, Taipei City 115, Taiwan R.O.C

[⊥]Institute of Biochemistry and Genetics Russian Academy of Science, Ufa, 450054, Russia

[∥]Moscow Institute of Physics and Technology (State University), Moscow Region, 141700, Russia

ABSTRACT: Modern computational approaches and machine learning techniques accelerate the invention of new drugs. Generative models can discover novel molecular structures within hours, while conventional drug discovery pipelines require months of work. In this article, we propose a new generative architecture, entangled conditional adversarial autoencoder, that generates molecular structures based on various properties, such as activity against a specific protein, solubility, or ease of synthesis. We apply the proposed model to generate a novel inhibitor of Janus kinase 3, implicated in rheumatoid arthritis, psoriasis, and vitiligo. The discovered molecule was tested in vitro and showed good activity and selectivity.



KEYWORDS: *adversarial autoencoders, disentanglement, conditional generation, Janus kinase*

INTRODUCTION

The latest advancements in deep learning are propagating into biomarker development, drug discovery, and drug repurposing.[1−7] The development of a new drug, from the original idea to the market approval, is a complicated process. It takes many years and can cost over $2.5 billion, with over a 90% failure rate in human clinical trials. While the standard drug discovery pipeline includes many stages, it is still an open problem to find an initial set of molecules that would change the activity of a specific protein or a signaling pathway.[8]

Researchers can improve the hit rate of new drug candidates by removing unpromising compounds at early stages with machine learning models to estimate properties of the compound and guide the drug optimization process.[9−11] Machine learning techniques also allow researchers to learn useful latent representations of molecules using variational autoencoders,[12] graph convolutions,[13,14] and graph message passing networks.[15] Gómez-Bombarelli et al.[16] used the latent representation of VAE to optimize chemical properties of encoded molecules using Bayesian optimization.

Generative adversarial networks (GAN)[17] and adversarial autoencoders (AAE)[18] have become prominent in the generative modeling of structured objects, such as text, speech, and images.[19−21] Generative models, trained on molecular descriptors, 3D structure, textual notation, or molecular graphs,[16,22−25] can create novel molecular structures with desired properties, such as the activity against a given target-protein.

Makhzani et al.[18] applied supervised adversarial autoencoders (SAAE) to generate new objects with given properties. The original model achieved good results with a few simple conditions, but the generation of complex objects (molecular structures or high-resolution images) requires dozens of complex conditions with thousands of variations. In this work, we improve SAAE architecture and demonstrate a significantly higher performance in the generation of novel chemical structures given complex conditions. Our contribution is 3-fold:

- We discuss disentanglement issues in SAAE and show that SAAE does not have any theoretical guarantees for the conditional generation.
- We propose an improved model with two different disentanglement approaches and a semisupervised extension.
- We validate our model by generating compounds against a specific protein and test one generated molecule in vitro. The discovered molecule (Figure 1) shows good inhibition activity and can be further developed as a primary hit.

Special Issue: Deep Learning for Drug Discovery and Biomarker Development

Received: August 7, 2018
Revised: September 3, 2018
Accepted: September 4, 2018
Published: September 4, 2018

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

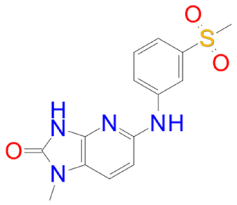


Figure 1. A novel molecule generated with the entangled conditional adversarial autoencoder (ECAAE). It was tested in vitro and showed high binding affinity and specificity toward the Janus kinase 3 (JAK3) protein.

CONDITIONAL ADVERSARIAL AUTOENCODER

Adversarial autoencoders[18] are generative models that model the data distribution $p_{\text{data}}(x)$ by training a regularized autoencoder. The regularizer forces a distribution of the latent code $q(z) = \int Q_E(z|x)p_{\text{data}}(x)dx$ to match a tractable prior $p(z)$. In this article, we will only consider deterministic autoencoders: the encoding distribution $Q_E(z|x)$ and decoding distribution $P_G(x|z)$ are parametrized by neural networks E and G, respectively, $z = E(x)$ and $x = G(z)$.

Regularization of the latent space is implemented by an adversarial training procedure[17] with the discriminator model $D(z)$. The discriminator is trained to discriminate between samples from the latent distribution $q(z)$ and the prior $p(z)$. The encoder E is trained to modify the latent code so the discriminator can not distinguish the latent distribution from the prior. This results in a minimax game $\min_E \max_D \mathcal{L}_{\text{adv}}$, where

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{x\sim p_{\text{data}}} \log\ D(E(x)) + \mathbb{E}_{z\sim p(z)} \log\ (1 - D(z)) \quad (1)$$

The adversarial training with the reconstruction penalty constitutes the following optimization task:

$$\min_{E,G} \max_{D} \mathbb{E}_{x\sim p_{\text{data}}} \log\ D(E(x)) + \mathbb{E}_{z\sim p(z)} \log\ (1 - D(z)) - \mathbb{E}_{x\sim p_{\text{data}}} \log\ p(x|G(E(x))) \quad (2)$$

The framework of adversarial autoencoders can be extended to the conditional generation. Consider data points $x \in \mathcal{X}$ coupled with some properties $y \in \mathcal{Y}$. The conditional generation procedure produces samples from the distribution $p(x|y)$ for any fixed property y. Supervised adversarial autoencoders (SAAE)[18] modifies the reconstruction process by concatenating the property y with the latent code z at the input of the decoder (Figure 2). The training procedure becomes (new parts are in bold)

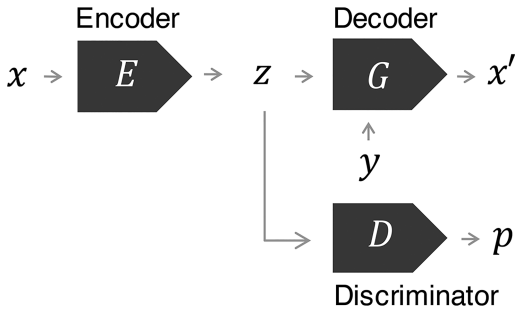


Figure 2. Supervised adversarial autoencoder (SAAE) model.

$$\min_{E,G} \max_{D} \mathbb{E}_{x\sim p_{\text{data}}} \log\ D(E(x)) + \mathbb{E}_{z\sim p(z)} \log\ (1 - D(z)) - \mathbb{E}_{(x,\boldsymbol{y})\sim p_{\text{data}}} \log\ p(x|G(E(x), \boldsymbol{y})) \quad (3)$$

In the original article, the authors suggested to generate new objects by first sampling $z \sim p(z)$ and then passing the latent code through the generator $x = G(z,y)$. This process implies independence of z and y, which is not always true. Sampling from $p(z)$ can be inconsistent, even if the model perfectly matches the latent distribution $p(z)$, and the reconstruction works well, as shown in the Experiments section. Intuitively, this may happen if the marginal distribution of latent codes is $p(z)$, but for any fixed y, we get a completely different distribution $p(z|y)$, as illustrated in Figure 3. In this case, we cannot generate useful samples from

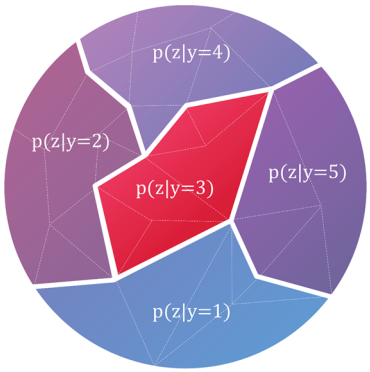


Figure 3. Motivation for the disentanglement. The marginal distribution of latent codes is $p(z)$, but conditional distributions differ from the marginals.

$p(z)$ for a specified y. Instead, we must sample from an intractable distribution $p(z|y)$. To overcome this inconsistency issue, we introduce two approaches: forcing conditional distributions $p(z|y)$ to be close to a marginal distribution $p(z)$, and learning $p(z|y)$ directly.

DISENTANGLEMENT OF *Z* AND *Y*

In this section, we describe predictive and joint approaches to disentangle latent codes z and properties y.

Predictive Disentanglement. We estimate the dependence between two random variables by computing their mutual information $\mathcal{I}(z, y) = \mathcal{KL}[p(z, y)||\, p(z)p(y)]$

$$= \int p(z, y) \log \frac{p(z,y)}{p(z)p(y)} dzdy$$

where $\mathcal{KL}$ is the Kullback–Leibler divergence. We can promote the independence between y and z by minimizing this mutual information. Since the density of the distribution $p(z,y)$ is unknown, we approximate $I(z,y)$ with a variational distribution $q(y|z)$

$$\mathcal{I}(z, y) = \mathcal{H}(y) + \underbrace{\mathbb{E}_{p(y,z)} \log\ p(y|z)}_{=-\mathcal{H}(y|z)} + \underbrace{\max_q -\mathbb{E}_{p(z)} \mathcal{KL}(p(y|z)||\, q(y|z))}_{=0,\ \text{achieved at } q(y|z)=p(y|z)}$$

$$= \mathcal{H}(y) + \max_q \mathbb{E}_{p(y,z)} \log q(y|z) \quad (4)$$

where $\mathcal{H}(y)$ is a constant entropy term, and q is a neural network trained to estimate $p(y|z)$. Implying that z is obtained from data points by a deterministic mapping, the regularizer takes the following form:

$$\mathcal{R}_{\text{predictive}} = \max_{q(y|E(x))} \mathbb{E}_{(x,y)\sim p_{\text{data}}} \log q(y|E(x)) \quad (5)$$

We optimize this loss in an adversarial manner by first training a neural network q to extract information about y from z, and then updating the encoder to eliminate extracted features from the latent code. We call this method the predictive

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

disentanglement (Figure 4). The optimization procedure with a new term becomes (predictive disentanglement is in bold)

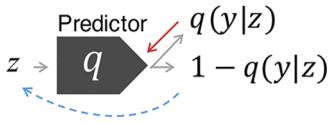


Figure 4. Predictive disentanglement. We first train $q(y|z)$ to extract property y from the latent code (solid red line) and then modify z to confuse the predictor (dashed blue line).

$$\min_{E,G}\max_{D,\boldsymbol{q}} \mathbb{E}_{x\sim p_{\text{data}}}\log D(E(x)) + \mathbb{E}_{z\sim p(z)}\log(1 - D(z)) - \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log p(x|G(E(x), y)) + \lambda \mathbf{E}_{(\boldsymbol{x},\boldsymbol{y})\sim \boldsymbol{p}_{\text{data}}}\log\ \boldsymbol{q}(\boldsymbol{y}|\boldsymbol{E}(\boldsymbol{x})) \quad (6)$$

Joint Disentanglement. In the predictive disentanglement, the variational distribution $q(y|E(x))$ has to be flexible enough to capture dependencies between components of y. This can be challenging. In the Generation of Structural Analogs section, we use 166-long binary vectors as properties y, which requires a neural network to estimate a probability of 2^{166} possible fingerprints.

The predictive model usually assumes conditional independence of y components, as it allows us to optimize models independently for each component. Let us denote the family of factorized variational distributions as

$$Q = \left\{ q(y|z) \,\middle|\, q(y|z) = \prod_{i=1}^{d} q(y_i|z) \right\} \quad (7)$$

By optimizing in a narrow family of distributions, we will underestimate the remaining mutual information

$$I(z, y) = \mathcal{H}(y) + \max_{q} \mathbb{E}_{p(y,z)}\log q(y|z) \geq \mathcal{H}(y) + \max_{q\in Q} \mathbb{E}_{p(y,z)}\log q(y|z) \quad (8)$$

Let us denote the marginal distribution of a property y_i as $p(y_i)$. The predictive model will only make marginal distributions independent from z ($q(y_i|z) = p(y_i)$), which does not imply joint independence: $q(y|z) = p(y)$. Because of this, the joint distribution can retain arbitrarily complex dependencies of y and z and will not achieve independence.

We propose a disentanglement technique that avoids assumptions about $q(y|z)$, by discriminating pairs (z, y) instead of vectors z. We use the factorized prior $p(z)p(y)$ with labels independent from latent codes and sample the distribution $q(E(x), y)$ of real latent codes and their properties. Adversarial training brings the distribution $q(E(x), y)$ closer to $p(z)p(y)$, promoting independence. We call this method the joint disentanglement (Figure 5)

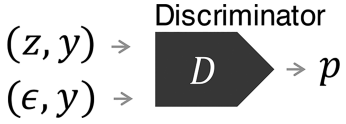


Figure 5. Joint disentanglement. We discriminate pairs (z, y) of latent codes and properties from pairs (ϵ, y), where $\epsilon \sim \mathcal{N}(0, I)$ are noise samples.

$$\min_{E,G}\max_{D} \mathbb{E}_{(x,\boldsymbol{y})\sim p_{\text{data}}}\log D(E(x), \boldsymbol{y}) + \mathbb{E}_{z\sim p(z)}\mathbf{E}_{\boldsymbol{y}\sim \boldsymbol{p}(\boldsymbol{y})} \log(1 - D(z, \boldsymbol{y})) - \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log p(x|G(E(x), y)) \quad (9)$$

Comparing Joint and Predictive Disentanglement. We introduced two methods of promoting independence between z and y. In our experiments, we found the joint disentanglement to be less stable than the predictive disentanglement at the beginning of training. It also requires a careful hyperparameters tuning. The predictive disentanglement, in contrast, is more stable and converges without an exhaustive hyperparameter search. However, as we mentioned above, the predictive disentanglement cannot achieve complete independence of z and y in complex cases. When working together, the predictive disentanglement forces the independence of marginals $p(y_i|z) = p(y_i)$, while the joint disentanglement reduces the remaining mutual information. As a result, we get a more stable technique that produces better results, as shown in the Experiments section. We call the method with both techniques the combined disentanglement.

$$\min_{E,G}\max_{D,q} \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log D(E(x), y) + \mathbb{E}_{z\sim p(z)}\mathbb{E}_{y\sim p(y)}\log(1 - D(z, y)) - \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log p(x|G(E(x), y)) + \lambda \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log q(y|E(x)) \quad (10)$$

■ ENTANGLED REPRESENTATION

The disentanglement of latent codes and labels is a powerful technique, but it imposes many constraints on the structure of a latent representation and may have a negative effect on the interpretability of latent features. For example, in ImageNet pictures, the distribution of object colors depends on a class label: cats usually have completely different colors than cars or trees. To improve the structure of the latent code, we add a dependence between y and z.

The probabilistic model becomes $p(y,z,x) = p(y)p(z|y)p(x|y,z)$. In this work, we learn $p(z|y)$ as a multivariate normal distribution with a diagonal covariance matrix parametrized by neural networks μ_θ and Σ_θ, $p(z|y) = \mathcal{N}(z|\mu_\theta(y), \Sigma_\theta(y))$, with θ optimized during training. To ensure that the parametrized posterior $p(z|y)$ matches the embeddings of the data, we train a discriminator to distinguish samples from $q(E(x)|y)$ and $\mathcal{N}(z|\mu(y), \Sigma(y))$. We also pass the property y to the discriminator to recognize which distribution is used as a reference for a specific object:

$$\min_{E,G}\max_{D,q} \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log D(E(x), y) + \mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim p(z|\boldsymbol{y})} \log(1 - D(z, y)) - \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log p(x|G(E(x), y)) \quad (11)$$

The discrimination between two learnable distributions is an unstable procedure, as for rare values of y, the discriminator poorly estimates the density $q(E(x)|y)$. To stabilize the training procedure, we apply the reparameterization trick, deterministically transforming latent codes into samples of the standard distribution, $\overline{z} = g_\theta(z,y)$ and discriminating samples from $p(\overline{z})p(y)$ and $q(g_\theta(E(x),y),y)$. Now, the distribution $p(\overline{z})p(y)$ does not depend on parameters θ and is fixed during training. For the normal distribution, the reparameterization trick becomes $g_\theta(z, y) = \Sigma_\theta^{-1/2}(y)(z - \mu_\theta(y))$ and a prior $p(\overline{z})$ is a standard normal distribution $\mathcal{N}(0, I)$. The optimization procedure after reparameterization becomes

$$\min_{E,G,\boldsymbol{\theta}}\max_{D,q} \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log D(\boldsymbol{g}_{\boldsymbol{\theta}}(E(x), \boldsymbol{y}), y) + \mathbb{E}_{y\sim p(y)}\mathbf{E}_{\overline{\boldsymbol{z}}\sim \boldsymbol{p}(\overline{\boldsymbol{z}})}\log(1 - D(\overline{\boldsymbol{z}}, y)) - \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log p(x|G(E(x), y)) \quad (12)$$

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

Since y and $\bar{z}$ are sampled independently, the discrimination procedure can be interpreted as a joint disentanglement of the reparameterized latent code and its property y. This leads us to the final idea to replace the joint disentanglement with the combined disentanglement. We call this model an entangled conditional adversarial autoencoder (ECAAE) model (Figure 6 and Chart 1). The underlying optimization task is

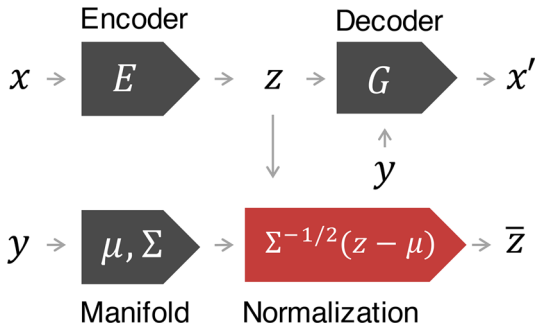


Figure 6. Entangled model. We encode an input x into a latent code z. We parametrize $p_\theta(z|y) \sim \mathcal{N}(\mu_\theta(y), \Sigma_\theta(y))$ and perform the reparameterization trick to obtain $\bar{z}$. We then apply disentanglement techniques to $\bar{z}$.

Chart 1

Algorithm 1 Entangled Conditional AAE with semi-supervision

function Loss(x, y)
  $\epsilon \sim \mathcal{N}(0, I)$
  $\hat{y} \sim h(\hat{y} \mid x)$
  $m \leftarrow$ present positions of y
  $y = m * y + (1 - m) * \hat{y}$
  $z = E(x)$
  $\bar{z} = \Sigma_\theta^{-1/2}(y)(z - \mu_\theta(y))$
  $\mathcal{L}_{\text{discriminator}} = \log D(\bar{z}, y) + \log(1 - D(\epsilon, y))$
  $\mathcal{L}_{\text{predictive}} = \log q(y \mid \bar{z})$
  $\mathcal{L}_{\text{reconstruction}} = -\log p(x \mid G(z, y))$
  $\mathcal{L}_{\text{imputer}} = -\log p(\hat{y}|y)$
  return $\mathcal{L}_{\text{discriminator}} + \mathcal{L}_{\text{reconstruction}} + \lambda\mathcal{L}_{\text{predictive}} + \mathcal{L}_{\text{imputer}}$

repeat
  for $i = 1$ **to** discr_batches **do**
    $x, y \sim p_{\text{data}}$
    Optimization step to maximize Loss(x, y) w.r.t. D, q
  $x, y \sim p_{\text{data}}$
  Optimization step to minimize Loss(x, y) w.r.t. $E, G, \mu_\theta, \Sigma_\theta, h$
until converged

$$\begin{aligned}\min_{E,G,\theta}\max_{D,q}\ & \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log D(g_\theta(E(x), y), y) \\ & + \mathbb{E}_{y\sim p(y)}\mathbb{E}_{\bar{z}\sim p(\bar{z})}\log(1 - D(\bar{z}, y)) \\ & - \mathbb{E}_{(x,y)\sim p_{\text{data}}}\log p(x|G(E(x), y)) \\ & + \lambda\mathbb{E}_{(x,y)\sim p_{\text{data}}}\log q(y|g_\theta(E(x), y))\end{aligned} \tag{13}$$

SEMISUPERVISED EXTENSION

In medicine, and especially in drug discovery, we rarely know the whole set of properties. To fill in these missing values, one has to conduct expensive in vitro or in vivo experiments. One example of this property would be the activity of a molecule against a specific protein. Other properties may require computationally expensive simulations, such as molecular dynamics or docking. Utilization of partially labeled data sets should result in an improved performance of a drug discovery pipeline.

Proposed models can be naturally extended for a partially labeled data by training an imputer model $h(\hat{y}|x)$ that approximates the values of unknown properties. During backpropagation, gradients for h are passed through both known and unknown positions, allowing the imputer to train jointly with the generative model. The vector with imputed properties is computed as $m \cdot y + (1-m) \cdot \hat{y}$, where m is a binary mask vector with zeros in positions corresponding to unknown labels.

RELATED WORK

The generation of new objects according to a certain condition is a challenging yet popular task that widely employs modern generative models. The most commonly used generative frameworks are generative adversarial networks (GAN),[17] variational autoencoders (VAE),[12] and adversarial autoencoders (AAE).[18] The conditional generative models with some restrictions can be obtained by adding a label to the input of the generator network. The conditional GAN[27] also passes a property to the discriminator. If the generated object does not have a given property, the discriminator will identify this object as fake. Supervised AAE[18] adds a condition to the generator's input, which, as discussed above, is not sufficient for a true conditional generation. In a similar way, variational autoencoders (VAE) were extended to the conditional generation by Sohn et al.[26]

To address disentanglement, Cheung et al.[28] regularized the cross-covariance matrix between the property and the latent code by encouraging it to be close to the identity matrix. Another approach based on the domain adaptation technique[29] was proposed by Lample et al.[21] and Creswell et al.[30] This method resembles our predictive disentanglement method. Mathieu et al.[31] uses a method similar to the joint disentanglement on the produced objects, instead of the latent codes directly. This approach is not directly applicable for discrete objects, as it requires passing the gradient through sampling from discrete random variables.

Deep generative architectures have been used to discover new molecules with specified properties. Zhou[32] applied graph convolutions to generate new chemical structures from graph representations. Segler et al.[33] and Gupta et al.[34] implemented RNN on the SMILES representation.[35,36] VAE and AAE-based models[10,22,37] were used for the generation of molecular structures. Guimaraes et al.[38] and Putin et al.[39,40] combined GAN with the reinforcement learning objective on the generator network in order to generate molecules with specific properties. Olivecrona et al.[41] formulated the SMILES sequence generation process in terms of the reinforcement policy optimization.

Deep information bottlenecks[42,43] also require minimization of the mutual information. Our combined disentanglement extends and stabilizes the minimization of MI and can be used to improve current information bottleneck techniques.

EXPERIMENTS

Data. For our experiments, we used Clean Leads molecules from the ZINC database.[44] We performed an additional filtering to optimize the data set toward the potential drug candidates and increase the hit rate of novel drug compounds. For this purpose, we removed not drug-like molecules, charged molecules, and those that contained atoms other than C, N, S, O, F, Cl, Br, or H. The remaining set of molecules was filtered with additional drug-likeness filters to exclude toxic and insoluble structures. The final data set contained roughly 1.8 million molecules encoded as strings in the form of canonical SMILES.

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

We parsed SMILES notations to separate atoms as individual tokens. This led to a vocabulary of size 30, which contained atoms, SMILES-specific syntax elements, and special tokens. The median length of the token sequence was 36 tokens, the maximum length was 57.

Training Details. The model was implemented in PyTorch.[45] We used a recurrent encoder and decoder with two LSTM layers of 256 units each. Hidden and cell states from the last time step of the encoder were linearly mapped onto a 64-dimensional space that we used as an embedding of the input sequence. The initial state of the decoder was obtained by a linear transformation from the embedding to the hidden and cell states of the recurrent decoder. At training time, we used the teacher forcing algorithm.[46] At evaluation time, we sampled tokens from the posterior distribution at each time step. We trained models using RMSProp[47] with an initial learning rate of 0.01, halving it after each 50 000 optimization steps. We used weight decay of 10^{-5} for g_θ and 10^{-6} for all other components. We used mini-batches of size 512 and trained all models for roughly 200 000 updates, which was sufficient for the model to converge. D, q, and h networks were represented by fully connected networks with two hidden layers of size 128. The network for μ_θ and Σ_θ was a fully connected network with 3 hidden layers of size 128. We tried different schedules for adversarial training and decided to use 4 updates of D, q, and h for each update of E, G, μ_θ and Σ_θ.

Generation of Structural Analogs. In the first experiment, we applied proposed models to generate structural analogs of known potent molecules. We measured the similarity between compounds by comparing their fingerprints (feature vectors describing the molecular structure with each bit of the fingerprint describing the presence or absence of different molecular substructures, such as acidic groups or aromatic rings). We trained conditional models to generate molecules using 166-bit long molecular access system (MACCS) binary fingerprints. To produce structural analogs of existing drugs, we generated 10 000 SMILES strings with each model by conditioning them on fingerprints that were excluded from the training data set.

We report Tanimoto similarity (Jaccard index for binary vectors) and Hamming distance between fingerprints of generated molecules and molecules used as a condition. We also report the percentage of molecules that exactly matched the condition value. Results in Table 1 suggest that the entangled representation satisfies conditions more often than other models. To compare different disentanglement techniques, we estimated mutual information (MI) between z and y using the mutual information neural estimation (MINE) method.[48] Results suggest that the predictive disentanglement eliminates more information than the joint disentanglement. However, as suggested in the Joint Disentanglement section, the predictive model cannot eliminate all mutual information, as it fits the predictor in a class of fully factorized distributions. Combining both methods halved the remaining MI. Finally, adding the predictive disentanglement to the entangled model also reduced the MI.

Table 1. Performance of Models Trained with Different Disentanglement Techniques Using Fingerprint Vectors as the Condition[a]

disentanglement	Tanimoto (%)	Hamming (%)	exact (%)	remaining MI (%)
no	80.0	10.49	4.4	2.75
predictive	86.2	7.13	11.4	0.64
joint	88.7	5.78	17.4	1.56
combined	91.8	4.18	27.8	0.32
entangled, no predictive	93.5	3.31	40.9	2.51
entangled	**93.6**	**3.28**	**41.3**	1.30

[a]Notice the large gap between the model with no disentanglement (corresponding to ref 18) and other models.

We also evaluated our models for diversity of generated molecules and generated 100 compounds for each fingerprint from the test set. Results reported in Figure 7 suggest that the diversity of generated structures uniformly improves for all proposed models.

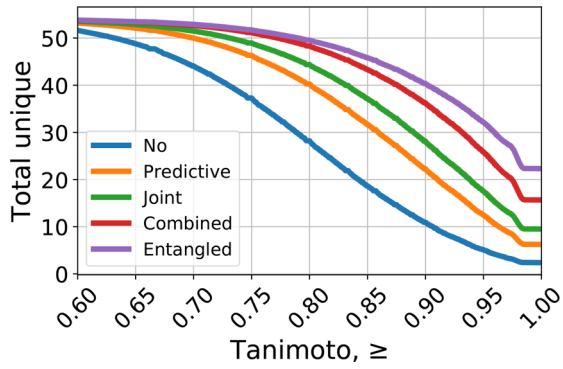


Figure 7. Average number of unique molecules (out of 100) for different Tanimoto score thresholds. Results show uniform contribution of different components across all thresholds.

We experimented with models conditioned on a more complex representation, Morgan[49] fingerprint of length 2048 and radius 4. With this input, all models showed similar performance: Tanimoto 0.8, Hamming 160, and an exact match of 60%. Since the large Morgan fingerprint almost uniquely describes the molecule, the decoder is likely to ignore the latent code, using only the fingerprint itself to generate the compound. This experiment suggests that the proposed ECAAE model is useful for moderate-size fingerprints that describe molecules incompletely.

Continuous Properties. We also evaluated the performance of our models on continuous properties: lipophilicity (logP) and synthetic accessibility (SA),[50] obtained from RDKit.[51] The ease of synthesis (low SA) is a desirable attribute of any prospective lead, while low logP is an important property of a potential oral drug candidate. For trained models, we jointly sampled logP and SA from the test data set and measured the Pearson correlation coefficient ρ between specified conditions and obtained properties of generated molecules. We removed generated molecules that were also present in the training data set when computing ρ. Results in Table 2 suggest that the entangled model balanced the quality of both logP and SA, while other models concentrated on the simpler property, logP. Table 3 contains examples of generated molecules for extreme values of the properties. On this

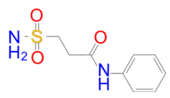
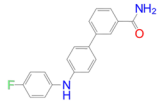
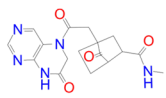
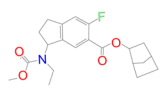
Table 2. Performance for Continuous Properties[a]

disentanglement	logP, r	SA, r
no	0.088 ± 0.005	0.004 ± 0.006
predictive	0.661 ± 0.005	0.060 ± 0.01
joint	0.432 ± 0.006	0.034 ± 0.01
combined	0.654 ± 0.004	0.113 ± 0.003
entangled	0.613 ± 0.004	0.431 ± 0.005

[a]We report the Pearson correlation r between the actual value for the generated molecules and the requested one.

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

Table 3. Generated with Entangled Model Molecules for Extreme Values of logP and SA[a]

	logP	SA	logP	SA	logP	SA	logP	SA
Requested	0.00	1.00	4.00	1.00	0.00	5.00	4.00	5.00
Actual	0.30	1.77	4.34	1.66	−0.12	5.03	4.25	4.58
Molecule								

[a]The left molecule has good logP and is easy to synthesize, while the bottom right is less soluble and harder to obtain.

data set, the difference between different disentanglement techniques is much lower than on MACCS fingerprints. This is presumably due to less interdependence between logP and SA than between 166 bits of the fingerprint, which was the limiting factor for the predictive disentanglement.

Semisupervised Data. To evaluate the semisupervised models, we computed the binding energy of 140 000 molecules from the AID1022 bioassay to the leukemia-related protein *MCL1* with AutoDock Vina.[52] The binding energy E is an important value that shows how well a molecule can fit in an active site of a protein. The large negative value of E corresponds to the high binding affinity. We also added logP and SA values described in the previous section to the properties. The generation results are reported in Table 4. In the semisupervised

Table 4. Performance of Semisupervised Models on the Partially Labeled Binding Energy Dataset in Terms of the Pearson Correlation r between the Requested Value and the Generated One

disentanglement	logP, r	SA, r	E, r
no	0.311 ± 0.01	0.0522 ± 0.009	0.02 ± 0.04
predictive	0.687 ± 0.006	0.0893 ± 0.008	0.063 ± 0.05
joint	0.595 ± 0.007	0.0838 ± 0.008	0.109 ± 0.04
combined	0.677 ± 0.007	0.0896 ± 0.007	0.116 ± 0.04
entangled	**0.804 ± 0.005**	**0.593 ± 0.007**	**0.406 ± 0.04**

scenario, the entangled model often satisfies all three conditions, while other models seem to ignore SA and the binding energy. We also evaluated the coefficient of determination R^2 of the imputation quality $h(\hat{y}|x)$ and observed it to be similar for all models with values of 0.99 for logP, 0.95 for SA, and 0.6 for E.

Results suggest that the auxiliary task of predicting values of the condition helps improve the conditional generation by stabilizing the encoder training for most of the models. In this experiment, the entangled model was able to satisfy conditions significantly better than the others. Comparing different disentanglement techniques, the combined model compromised the performance on logP for the better statistics on E. We can also see that the joint disentanglement was not able to capture the correlation between different properties components.

Finally, we generated a few molecules conditioned on properties of the molecule with the lowest binding energy in the data set: $E = -11.1$, logP = 3.95, and SA = 1.8. Interestingly, two of the generated molecules had a binding energy of $E = -11.7$, demonstrating higher binding affinity toward the target. The simulated position of the generated molecule in the active site of the *MCL1* protein is shown on Figure 8.

In Vitro Validation. In this section, we apply our model to the drug discovery pipeline by generating a selective inhibitor of a Janus kinase 3 (JAK3). The Janus kinase (JAK) family contains

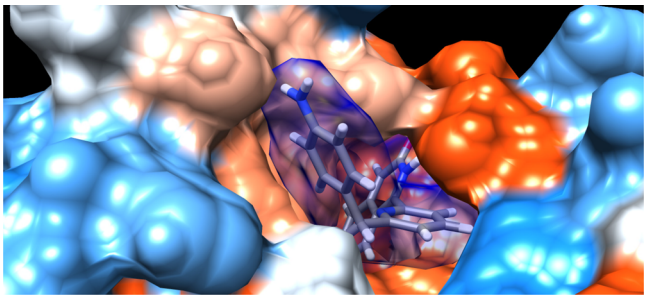


Figure 8. Position of the generated molecule in the active site of the *MCL1* protein. This molecule has lower binding energy than any other molecule in the training set.

four members, JAK1−3 and TYK2, with a different therapeutic significance. JAK3 is a promising biological target against rheumatoid arthritis, psoriasis,[53−55] alopecia,[56] and vitiligo. Currently, there are more than 10 novel small-molecule JAK inhibitors with an improved selectivity in different stages of clinical trails;[57−59] therefore, we are mainly focused on selective JAK3 kinase inhibitors.

To discover a selective compound, we collected a database of known inhibitors of JAK2 and JAK3 from the ChEMBL[60] database and trained a semisupervised entangled AAE model conditioned on the activity of molecules for JAK2 and JAK3. We specified high activity against JAK3 and low activity against JAK2 as a condition. We generated 300 000 molecules and passed them through a series of filters, including molecular docking,[52] prediction of side effects and chemical properties. This reduced the number of molecules to roughly 5000. Selected molecules were used for simulation of molecular dynamics, which resulted in a set of the 100 most promising molecules. Out of these molecules, medicinal chemists selected the most promising molecule, according to their experience. The chosen molecule was synthesized and tested in vitro against JAK2 and JAK3 as well as two other kinases, B-Raf and c-Raf. The activity was measured in terms of IC_{50}, a concentration of an inhibitor at which the enzyme is at the half of its maximal activity. A molecule is considered an initial hit if its IC_{50} against a target protein is less than 10 μM. The discovered molecule, presented in Figure 1, was shown to be active for JAK3 (IC_{50} = 6.73 μM) and inactive for JAK2 (IC_{50} = 17.58 mM), B-Raf (IC_{50} = 85.55 μM), and c-Raf (IC_{50} = 64.86 μM). Dose−response curves are shown in Figure 9.

CONCLUSION

In this work, we introduced an ECAAE with several disentanglement techniques to improve the generation quality. We applied our model to a generation of molecules with specified property descriptors, solubility, and synthetic accessibility

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

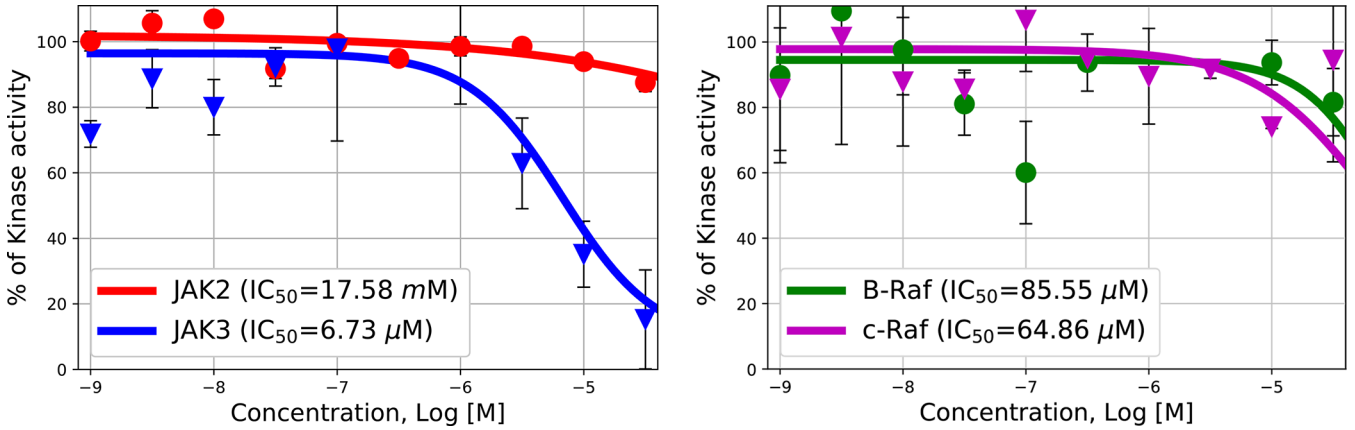


Figure 9. (Left) The effectiveness of the discovered molecule in inhibiting the JAK2 and JAK3 kinases. IC_{50} for JAK3 shows micromolar activity. The discovered molecule is active for JAK3 but not JAK2. (Right) Inhibition of B-Raf and c-Raf. The discovered molecule does not inhibit these proteins, which suggests its good specificity.

scores. We also conditioned the model on target-specific properties, such as the binding energy or IC_{50}. ECAAE discovered a promising hit compound with high selectivity against the JAK3 isoform over JAK2 and RAF kinases. The proposed architecture can be used to generate novel molecules with promising scaffolds. These results suggest that ECAAE can be integrated into the automated drug discovery pipelines to generate large sets of initial hypotheses for drugs in multiple disease areas.

AUTHOR INFORMATION

Corresponding Author

*E-mail: daniil@insilico.com.

ORCID

Daniil Polykovskiy: 0000-0002-0899-8368
Alexander Zhebrak: 0000-0003-1905-9996
Dmitry Vetrov: 0000-0001-6863-9028
Yan Ivanenkov: 0000-0002-8968-0879
Vladimir Aladinskiy: 0000-0002-3976-3991
Polina Mamoshina: 0000-0002-7705-4694
Marine Bozdaganyan: 0000-0003-1647-1627
Alexander Aliper: 0000-0002-4363-0710
Alex Zhavoronkov: 0000-0001-7067-8966
Artur Kadurin: 0000-0003-1482-9365

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

The authors would like to thank NVIDIA and Mark Berger personally for providing the early access to the NVIDIA V100 equipment used in this study. The authors thank the National Research University Higher School of Economics supported by the Russian Science Foundation Grant 17-71-20072 for discussion and commentary on the manuscript. The authors would like to thank the scientists from Moscow Institute of Physics and Technology supported by the Ministry of Education and Science of the Russian Federation, government Grant 20.9907.2017/VU and IBG RAS Ufa supported by the Russian Science Foundation 17-74-30012 for their valuable review of and comments on the manuscript.

REFERENCES

(1) Angermueller, C.; Parnamaa, T.; Parts, L.; Stegle, O. Deep learning for computational biology. *Mol. Syst. Biol.* **2016**, *12*, 878.

(2) Mamoshina, P.; Vieira, A.; Putin, E.; Zhavoronkov, A. Applications of Deep Learning in Biomedicine. *Mol. Pharmaceutics* **2016**, *13*, 1445−1454.

(3) Miotto, R.; Wang, F.; Wang, S.; Jiang, X.; Dudley, J. T. Deep learning for healthcare: review, opportunities and challenges. *Briefings Bioinf.* **2017**, *6*, 1.

(4) Chen, H.; Engkvist, O.; Wang, Y.; Olivecrona, M.; Blaschke, T. The rise of deep learning in drug discovery. *Drug Discovery Today* **2018**,.231241

(5) Ching, T. Opportunities and obstacles for deep learning in biology and medicine. *J. R Soc. Interface* **2018**, *15*, 20170387.

(6) 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* **2017**, *8*, 1021−1033.

(7) 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*, 210−222.

(8) Ozerov, I. V.; et al. silico Pathway Activation Network Decomposition Analysis (iPANDA) as a method for biomarker development. *Nat. Commun.* **2016**, *7*, 13427.

(9) Wallach, I.; Dzamba, M.; Heifets, A. AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery. 2015, https://arxiv.org/abs/1510.02855.

(10) Gomes, J.; Ramsundar, B.; Feinberg, E. N.; Pande, V. S. Atomic Convolutional Networks for Predicting Protein-Ligand Binding Affinity. 2017, https://arxiv.org/abs/1703.10603.

(11) Ragoza, M.; Hochuli, J.; Idrobo, E.; Sunseri, J.; Koes, D. R. Protein−Ligand Scoring with Convolutional Neural Networks. *J. Chem. Inf. Model.* **2017**, *57*, 942−957.

(12) Kingma, D. P.; Welling, M. Auto-encoding variational bayes. 2013, https://arxiv.org/abs/1312.6114.

(13) Duvenaud, D. K.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. Convolutional Networks on Graphs for Learning Molecular Fingerprints. 2015, https://arxiv.org/abs/1509.09292.

(14) Kearnes, S. M.; McCloskey, K.; Berndl, M.; Pande, V. S.; Riley, P. Molecular graph convolutions: moving beyond fingerprints. *J. Comput.-Aided Mol. Des.* **2016**, *30*, 595−608.

(15) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. Neural Message Passing for Quantum Chemistry. 2017, https://arxiv.org/abs/1704.01212.

(16) Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A.

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405

Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent. Sci.* **2018**, *4* (2), 268.
(17) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. 2014, https://arxiv.org/abs/1406.2661.
(18) Makhzani, A.; Shlens, J.; Jaitly, N.; Goodfellow, I. Adversarial Autoencoders. 2016, https://arxiv.org/abs/1511.05644.
(19) Radford, A.; Metz, L.; Chintala, S. *Unsupervised representation learning with deep convolutional generative adversarial networks*, International Conference on Learning Representations, 2016.
(20) Karras, T.; Aila, T.; Laine, S.; Lehtinen, J. Progressive Growing of GANs for Improved Quality, Stability, and Variation. 2017, https://arxiv.org/abs/1710.10196.
(21) Lample, G.; Zeghidour, N.; Usunier, N.; Bordes, A.; DENOYER, L.; Ranzato, M. A. Fader Networks:Manipulating Images by Sliding Attributes. 2017, https://arxiv.org/abs/1706.00409.
(22) 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*, 10883.
(23) Jin, W.; Barzilay, R.; Jaakkola, T. Junction Tree Variational Autoencoder for Molecular Graph Generation. *Proceedings of the 35th International Conference on Machine Learning* **2018**, *80*, 2323−2332.
(24) Kuzminykh, D.; Polykovskiy, D.; Kadurin, A.; Zhebrak, A.; Baskov, I.; Nikolenko, S.; Shayakhmetov, R.; Zhavoronkov, A. 3D Molecular Representations Based on the Wave Transform for Convolutional Neural Networks. *Mol. Pharmaceutics* **2018**, *15*, 1.
(25) Sanchez-Lengeling, B.; Aspuru-Guzik, A. Inverse molecular design using machine learning: Generative models for matter engineering. *Science* **2018**, *361*, 360−365.
(26) Sohn, K.; Lee, H.; Yan, X. Learning structured output representation using deep conditional generative models. *Advances in Neural Information Processing Systems* **2015**, 3483−3491.
(27) Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. 2014, https://arxiv.org/abs/1411.1784.
(28) Cheung, B.; Livezey, J. A.; Bansal, A. K.; Olshausen, B. A. Discovering Hidden Factors of Variation in Deep Networks. 2014, https://arxiv.org/abs/1412.6583.
(29) Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; Lempitsky, V. S. Domain-Adversarial Training of Neural Networks. *J. Mach. Learn. Research* **2016**, *17*, 1.
(30) Creswell, A.; Bharath, A. A.; Sengupta, B. Conditional Autoencoders with Adversarial Information Factorization. 2017, https://arxiv.org/abs/1711.05175.
(31) Mathieu, M. F.; Zhao, J. J.; Zhao, J.; Ramesh, A.; Sprechmann, P.; LeCun, Y. In *Advances in Neural Information Processing Systems 29*; Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., Garnett, R., Eds.; Curran Associates, Inc., 2016; pp 5040−5048.
(32) Zhou, Z.; Li, X. Convolution on Graph: A High-Order and Adaptive Approach. 2017, https://arxiv.org/abs/1706.09916.
(33) Segler, M. H. S.; Kogej, T.; Tyrchan, C.; Waller, M. P. Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks. *ACS Cent. Sci.* **2018**, *4*, 120−131.
(34) Gupta, A.; Müller, A. T.; Huisman, B. J. H.; Fuchs, J. A.; Schneider, P.; Schneider, G. Generative Recurrent Networks for De Novo Drug Design. *Mol. Inf.* **2018**,.371700111
(35) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Model.* **1988**, *28* (1), 31.
(36) Weininger, D.; Weininger, A.; Weininger, J. L. SMILES. 2. Algorithm for generation of unique SMILES notation. *J. Chem. Inf. Model.* **1989**, *29*, 97−101.
(37) Blaschke, T.; Olivecrona, M.; Engkvist, O.; Bajorath, J.; Chen, H. Application of Generative Autoencoder in de Novo Molecular Design. *Mol. Inf.* **2018**, *37*, 1.
(38) Guimaraes, G. L.; Sanchez-Lengeling, B.; Farias, P. L. C.; Aspuru-Guzik, A. Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models. 2017, https://arxiv.org/abs/1705.10843.
(39) 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. Pharmaceutics* **2018**, *15*, 1.
(40) 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*, 1194−1204.
(41) Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular de-novo design through deep reinforcement learning. *J. Cheminf.* **2017**, *9*, 48.
(42) Wieczorek, A.; Wieser, M.; Murezzan, D.; Roth, V. *Learning Sparse Latent Representations with the Deep Copula Information Bottleneck*, International Conference on Learning Representations, 2018.
(43) Alemi, A. A.; Fischer, I.; Dillon, J. V.; Murphy, K. *Deep variational information bottleneck*, International Conference on Learning Representations, 2017.
(44) *Zinc*[12]; University of California, San Francisco, http://zinc.docking.org/subsets/clean-leads.
(45) Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. *Automatic differentiation in PyTorch*; NIPS-W, 2017.
(46) Williams, R. J.; Zipser, D. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation* **1989**, *1*, 270−280.
(47) Hinton, G.; Srivastava, N.; Swersky, K. Lecture 6a-Overview of mini-batch gradient descent. *Neural Networks for Machine Learning*, 2012.
(48) Belghazi, M. I.; Baratin, A.; Rajeshwar, S.; Ozair, S.; Bengio, Y.; Hjelm, D.; Courville, A. Mutual Information Neural Estimation. *PMLR* **2018**, *80*, 531−540.
(49) Morgan, H. L. The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service. *J. Chem. Doc.* **1965**, *5*, 107−113.
(50) Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *J. Cheminf.* **2009**, *1*, 8.
(51) Landrum, G. *RDKit: Open-source cheminformatics*; http://www.rdkit.org (accessed March 2012).
(52) Trott, O.; Olson, A. J. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *J. Comput. Chem.* **2009**, *31*, 455−461.
(53) Schwartz, D. M.; Kanno, Y.; Villarino, A.; Ward, M.; Gadina, M.; O'Shea, J. J. JAK inhibition as a therapeutic strategy for immune and inflammatory diseases. *Nat. Rev. Drug Discovery* **2017**, *16*, 843.
(54) Ivanenkov, T.; Balakin, K. V.; Tkachenko, S. E. New Approaches to the Treatment of Inflammatory Disease. *Drugs R&D* **2008**, *9*, 397−434.
(55) Ivanenkov, Y. A.; Balakin, K. V.; Lavrovsky, Y. Small Molecule Inhibitors of NF-kB and JAK/STAT Signal Transduction Pathways as Promising Anti-Inflammatory Therapeutics. *Mini-Rev. Med. Chem.* **2011**, *11*, 55−78.
(56) Samadi, A.; Ahmad Nasrollahi, S.; Hashemi, A.; Nassiri Kashani, M.; Firooz, A. Janus kinase (JAK) inhibitors for the treatment of skin and hair disorders: a review of literature. *J. Dermatol. Treat.* **2017**, *28*, 476−483.
(57) Verstovsek, S. Therapeutic potential of JAK2 inhibitors. *ASH Education Program Book* **2009**, *2009*, 636−642.
(58) Lancman, G.; Mascarenhas, J. Should we be treating lower risk myelofibrosis patients with a JAK2 inhibitor? *Expert Rev. Hematol.* **2017**, *10*, 23−28.
(59) Jain, T.; Mesa, R. The development, safety and efficacy of pacritinib for the treatment of myelofibrosis. *Expert Rev. Anticancer Ther.* **2016**, *16*, 1101−1108.
(60) Gaulton, A.; et al. The ChEMBL database in 2017. *Nucleic Acids Res.* **2017**, *45*, D945−D954.

DOI: 10.1021/acs.molpharmaceut.8b00839
Mol. Pharmaceutics 2018, 15, 4398−4405