JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

pubs.acs.org/jcim Article

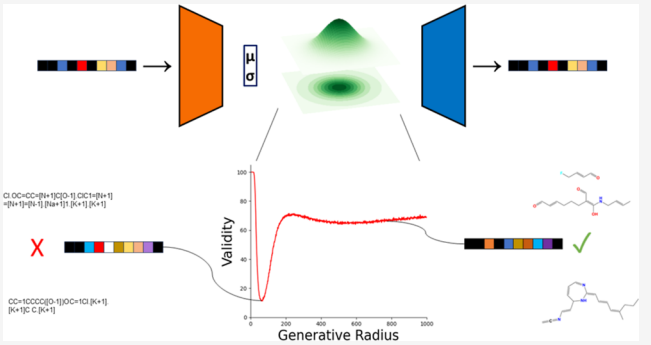
Fuzz Testing Molecular Representation Using Deep Variational Anomaly Generation

Victor H. R. Nogueira,[‖] Rishabh Sharma,[‖] Rafael V. C. Guido,[*] and Michael J. Keiser[*]

Cite This: *J. Chem. Inf. Model.* 2025, 65, 1911−1927 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Researchers are developing increasingly robust molecular representations, motivating the need for thorough methods to stress-test and validate them. Here, we use a variational auto-encoder (VAE), an unsupervised deep learning model, to generate anomalous examples of SELF-referencIng Embedded Strings (SELFIES), a popular molecular string format. These anomalies defy the assertion that all SELFIES convert into valid SMILES strings. Interestingly, we find specific regions within the VAE's internal landscape (latent space), whose decoding frequently generates inconvertible SELFIES anomalies. The model's internal landscape self-organization helps with exploring factors affecting molecular representation reliability. We show how VAEs and similar anomaly generation methods can empirically stress-test molecular representation robustness. Additionally, we investigate reasons for the invalidity of some discovered SELFIES strings (version 2.1.1) and suggest changes to improve them, aiming to spark ongoing molecular representation improvement.



INTRODUCTION

Anomalous or outlier data contaminate real-world data sets in domains from telecommunications to health care; these data can significantly deviate from the norm and require filtering.[1−3] This motivates the development of robust deployable anomaly detection models for data cleaning or raising alarms in dynamic information processing systems like browsing, spam, or credit card fraud detection.[4] Conversely, anomalies may be the subject of interest, pivoting the investigation from anomaly detection and removal to their active generation and incorporation, as in "fuzz testing" software. In software fuzz testing, invalid or anomalous inputs can expose hidden vulnerabilities as exceptions such as crashes.[5,6] Fuzz testing exposes exploits or limitations in software by intentionally injecting invalid, unexpected, or random inputs.

In some domains, anomaly detection modeling may suffer from training data scarcity that limits its predictive potential. In such cases, generating anomalies to populate synthetic training data sets may mitigate data scarcity and class imbalance.[7] In other cases, the nature of potentially anomalous data may be unknown due to the absence of real-world examples or the ineffectiveness of general formulation or conceptualization approaches for devising anomalous criteria.[8] For instance, manufacturing processes often progress through filtration steps to remove anomalous and defective pieces.[9] However, this relies on a definition of anomalous criteria despite the rarity of defective pieces. Here, machine learning (ML) for anomaly generation can reveal possible failure modes in the data. By extension, generative models can enable constructing and exploring an "anomalous data space" that reveals previously unknown or nonintuitive but actionable anomaly criteria. Similarly, generative exploration can reveal failure modes in data representations as unexpected behaviors that deviate from the norm in functionality. Such "representational anomalies" do not indicate semantic anomalies but expose representation definition or syntax loopholes. Representational anomalies are specific to their data.

Graphs[10] and strings including SMILES,[11] SELFIES,[12] and DeepSMILES[13] can represent molecular data. However, these representations may contain edge cases that fail to map the corresponding molecular data correctly. In addition to synthetic training set curation by generating (semantic) anomalies, such a scenario also presents an application domain for generating (representational) anomalies to support representation development efforts. In this work, we investigate the application of ML for representational anomaly generation to explore failure modes in a molecular string representation (SELFIES) and, consequently, test representational robustness. Specifically, we demonstrate how exploring a variational autoencoder (VAE) latent space, trained on purely normal (valid) data, can

Received: October 12, 2024
Revised: January 18, 2025
Accepted: January 24, 2025
Published: February 5, 2025

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

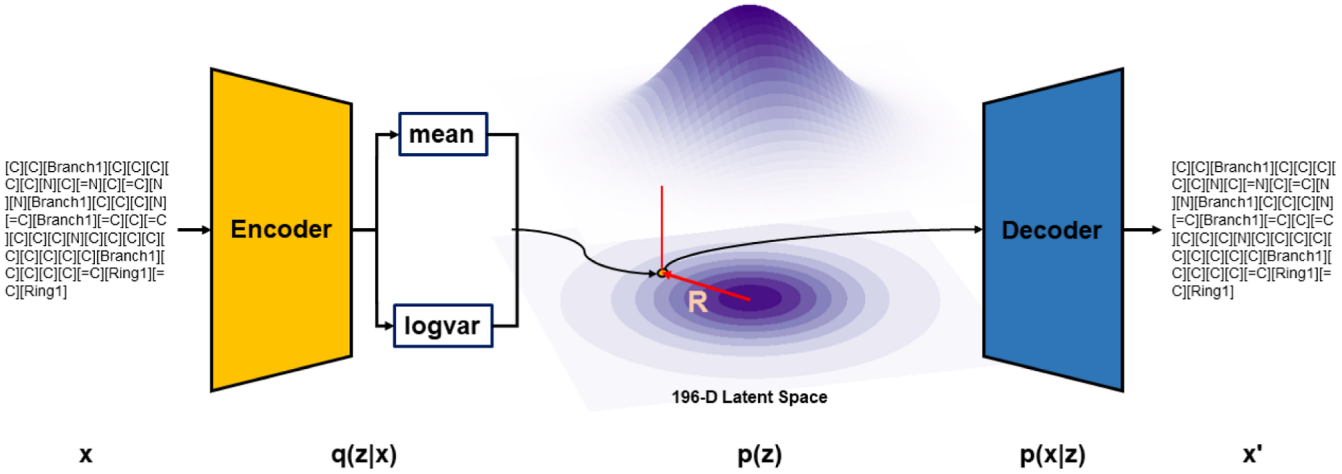


Figure 1. SELFIES-VAE workflow and illustration including the approximate posterior (encoder $q(z|x)$), prior ($p(z)$), and likelihood (decoder $p(x|z)$). The model encodes SELFIES strings into approximate posterior distributions in a 196-dimensional latent space. In the generative phase, points are sampled radially over hyper-sphere surfaces and decoded to SELFIES strings as a function of radius R.

effectively fuzz-test representational robustness by anomaly generation. While generative models have been extensively explored for de novo molecular design, we present a spatially aware analysis of their fuzz-testing abilities—which can guide representation development efforts. We also compare VAE-based anomaly generation against baseline generative null models to demonstrate how VAE fuzz-testing maximizes the number of representational anomalies.

Deep Learning Method: VAE. The VAE is a probabilistic variant of autoencoder[14] proposed by Kingma and Welling.[15] Instead of encoding inputs into a point estimate (degenerate distribution), the VAE encodes inputs into a conditional probability distribution to form a latent information bottleneck. This approach to latent space design enforces representational continuity and a generative interpretation.[16] By feed-forward propagation, the model may be decomposed into three parts (Figure 1). An encoder network approximates a posterior distribution over latent variable z conditioned on input data x, $q_\phi(z|x)$. This effectively distinguishes latent space regions by observation probability as a function of the conditioned data x. The second is an uninformative prior, $p(z)$, usually the standard multivariate Gaussian distribution. The third is a decoder that outputs a likelihood distribution over the data conditioned on the posterior-sampled latent variable z, $p_\theta(x|z)$, effectively generating a reconstruction of the input on sampling. The training jointly optimizes for encoder and decoder parameters ϕ and θ by maximizing the (evidence) lower bound (ELBO) on the log-likelihood of the data distribution.

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z))) \tag{1}$$

The two terms in this objective formulation achieve dual outcomes. The first term maximizes the log-probability of reconstructing the data x conditioned on the latent variable z, as expected by the posterior. Simultaneously, the second term minimizes the Kullback–Leibler (KL) divergence between the prior and posterior, which acts as a regularizer and minimizes variance by penalizing the encoder for approximating posteriors that diverge from the prior.

Given a specific hyperparameter configuration, a strongly enforced unit Gaussian prior should pull the training data's encoded latent embeddings, vectors sampled from approximated posterior distributions, to agglomerate near the zero origin. On the other hand, reconstruction optimization encourages the decoder to regenerate the training data by forward-propagating the latent embeddings. Effectively, the objective encourages the training data distribution to be both "encoded to" and "decoded from" the same latent regions, with their locality enforced by the prior. Additionally, we expect reconstruction optimization to encourage clustering over the input data's latent embeddings.[17,18] Consequently, we expect to observe latent space regions corresponding to "in-distribution" data separated from relatively anomalous "out-of-distribution" data on decoding, where the training data distribution defines "in-distribution". Consequently, by performing location-guided sampling-decoding analysis, we could determine which latent regions decode in-distribution versus out-of-distribution data. Given the VAE's generative property and density learning objective, this binary clustering and associated boundary identification should enable us to generate novel examples of either distribution on targeted decoding. Generative ML applications usually focus on the "in-distribution" regions to generate realistic and novel data that resemble training data. On the contrary, we investigate the VAE for anomalous "out-of-distribution" molecular strings with the aim of testing representational robustness.

Related Work. Various unsupervised, supervised, and semisupervised methods employ ML for anomaly detection. Common shallow methods include z-score and Mahalanobis distance threshold-based classification, local outlier factor (LOF), K-nearest neighbors, and support vector machines.[19−22] Deep neural networks such as VAEs can detect anomalies by learning to effectively reconstruct compressed representations of normal data while achieving only lossy reconstructions of anomalous data.[23−26]

Multiple ML methods can detect anomalies in biological data. For example, Michael-Pitschaze et al. detected anomalous proteins through representation learning using protein language models.[27] They extracted the penultimate layer of an encoder neural network to represent proteins, applied an anomaly scoring function to identify human prion-like proteins, and classified viral proteins from the host proteome. Similarly, Czibula et al. introduced AnomalP, a method to detect anomalous protein conformations using deep learning.[28]

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

Analogously, Ferre et al. proposed atyPeak, a stacked convolutional autoencoder, to detect anomalous peaks in genomic catalogs.[29] Similarly, to address experimental problems, Tiwari et al. applied advanced ML algorithms for anomaly detection in protein A chromatography column integrity.[30]

While ML use for anomaly generation is relatively rare in research settings, it is more common in software testing. In a research example, Laptev et al. generated synthetic anomalous time-series data by decoding outlying regions of a VAE's latent space.[7] In a biological context, Uzolas et al. deployed a conditional adversarial network to synthesize abnormal banded chromosome images and propose the method for cytogenetic data detection, simulation, and augmentation.[31] By contrast, multiple ML strategies effectively generate fuzz testing inputs for software robustness and safety testing. For instance, Godefroid et al. used LSTMs to generate PDF input grammars to develop generation-based fuzzers.[32,33] In another study, Rajpal et al. combined deep learning with the American Fuzzy Lop (AFL) fuzzer to enhance coverage by better choices of input bytes for mutation.[34] Similarly, She et al. proposed NEUZZ, a method for efficient fuzzing using neural program smoothing.[35] They developed surrogate neural network models capable of learning smooth approximations of a real-world program's behaviors. Proceeding to reinforcement learning, Becker et al. used the SARSA algorithm to fuzz the IPv6 protocol by introducing mutations to network packets sent to a host.[36−38] More recently, Deng et al. proposed FuzzGPT, a method to prime Large Language Models to generate unusual programs for fuzzing.[39] FuzzGPT found 76 bugs in the latest versions of PyTorch and TensorFlow at the time of testing. Analogous to generation-based software fuzzers, we study the VAE and associated radial latent exploration approach as a smart generative black-box fuzzing method for a molecular string representation.

Broadly, VAEs have been used for a variety of purposes and domains. For example, Sevgen et al. developed a transferable VAE to conditionally design protein sequences from low-dimensional latent embeddings.[40] Elsewhere, Wu and Xu proposed an adversarial VAE combined with residual learning to generate synthetic images of tomato leaf diseases to mitigate data scarcity.[41] In chemistry, Tempke and Musho generated chemical reactions by sampling the latent space of a VAE trained on gas-phase reactions data.[42]

Many molecular representations exist for molecular generation and de novo design. Lee and Min used graph matrices as model inputs and a multiobjective optimization process for molecular generation.[43] Jin et al. employed the junction tree VAE, a two-step process that creates a tree-based framework representing chemical substructures and then integrates them using a graphical message-passing neural network.[44] Hadipour et al. combined the SMILES representation and PCA dimensionality reduction in a VAE to embed molecular features and cluster small-molecule data sets.[45]

Case Study Molecular Representation: SELFIES. SELF-referencIng Embedded Strings (SELFIES) is a molecular string representation developed by Krenn et al.[12] to be 100% robust such that each SELFIES string converts to a valid molecular SMILES.[11] The representation addresses some fragility-inducing features commonly responsible for SMILES invalidity by removing them altogether, such as pairwise syntax constraints indicating ring and branch structures. To address the problem of bonded atom counts exceeding valences, SELFIES uses special parenthesized tokens that couple bonds with atoms to form a single token in a rule-abiding manner instead of treating bonds as separate tokens. SELFIES become SMILES by processing tokens through bond-order-correcting "rule vectors" for tokenized atoms that obey valence rules and maintain overall structural compliance. The module deploys a "derivation rule" table to process SELFIES tokens sequentially into SMILES tokens, building the corresponding SMILES string in parallel. Given this sequential processing, the corrective conversion process for a SELFIES token depends on its preceding and successive token's atomic composition and bond order or ring/branch type—comprising a "token neighborhood". The conversion process does not strictly preserve the atomic composition or empirical formula as the rules can discard misfitting or irreparable tokens to ensure validity. The authors demonstrate that SELFIES outperforms SMILES in maximizing the valid generative chemical space size.

The discovery of a SELFIES string that converts to an invalid SMILES string would demonstrate a representational anomaly that challenges nominal assumptions about 100% SELFIES validity. Such anomalies belong to an out-of-distribution space of hypothetical invalid SELFIES. Furthermore, these anomaly criteria focus exclusively on converting a SELFIES string to a valid SMILES through SELFIES's underlying mapping mechanisms. Therefore, we remove SELFIES token compositions from consideration and define a "normal versus anomalous" binary classification solely by the string's conversion success. Consequently, we explore SELFIES string anomalies solely at the representational level and accept chemically valid SMILES as a proxy for a valid molecule or molecular mixture, regardless of physicality. We test the hypothesis that SELFIES anomalies exist by using the generative properties of a VAE, conditioned on a training set, token set, model architecture choice, and latent space surveying approach, on the latest available SELFIES version 2.1.1.[46]

METHODS

VAE Architecture and Hyperparameters. We train and analyze a SELFIES-VAE in TensorFlow v. 2.10.0[47] with an architecture motivated by Gomez-Bombarelli et al.[48] The encoder consists of an embedding layer that processes SELFIES tokens represented as integer arrays, three one-dimensional convolutional layers (filters: 9,9,11; kernel size: 9,9,10), one flatten layer, and one dense layer that outputs parameters describing a 196-dimensional approximate posterior latent distribution. The decoder (Dec) receives a sampled latent vector z and consists of a dense layer followed by three sequence-returning GRU (Gated Recurrent Unit) containing 256 hidden units. Finally, a dense layer outputs a 91 × 54 likelihood matrix, $Dec(z) = L$ (Dec: $\mathbb{R}^{196} \rightarrow \mathbb{R}^{91\times 54}$), expressing model likelihood in the form of discrete unnormalized distributions (logits) of 54 SELFIES tokens for 91 sequence positions available in the sequence to be generated. The matrix element $L_{i,j}$ indicates the probability of observing token t_j in the ith position of the sequence. We employ a greedy decoding strategy to convert L to a SELFIES sequence by selecting the most probable token to occupy each position while ignoring the padding character in the final compilation. This is achieved by a *readout* function as follows

$$readout\ (L) = concat(t_{\mathrm{argmax}(L_{1,*})}, \ldots, t_{\mathrm{argmax}(L_{91,*})}) \quad \text{s. t.}\ t \neq pad \tag{2}$$

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

where *concat* indicates the string concatenation of tokens to produce a SELFIES sequence. This decoding process enforces a one-to-one mapping between latent points and SELFIES. Figure S1 pseudocode outlines the generation process. We compiled a SELFIES training set of 400,000 molecules to define encoder input dimensionality. We filtered sequences by size and converted from canonical SMILES in ChEMBL v.29[49,50] to obtain a data set token set size of 54, including a special padding token: ({"*pad*": 0, ".": 1, "[#Branch1]": 2, "[#Branch2]": 3, "[#*C* − 1]": 4, ..., "[*Si*]": 53}). This procedure yielded SELFIES with a maximum token count of 91 (Table 1 and Figure S2). Drawing from the same distribution as the training set, we compiled a validation set of 30,000 SELFIES strings. Figure S3 shows the distributions of various molecular properties for the training and validation sets. We trained the model using a sparse softmax cross entropy with logits loss (for reconstruction), KL divergence (for regularization), Adam optimizer,[51] and a learning rate of 0.001. We initialized parameters using a Glorot-uniform scheme and trained with a batch size of 128. We annealed the β-VAE's[52] β factor (KL loss weight) using a linear schedule after three epochs, with the final model trained in 15 epochs over 74 h (wall clock time) on a single NVIDIA RTX 3090 GPU. The annealing process dynamically updates β throughout training to mitigate KL vanishing and posterior collapse tendency.[53] We used early stopping with a patience of 8, meaning that training would stop if the best validation loss value did not decrease by a margin of 2.0 over 8 consecutive epochs, to determine convergence. We used the RDKit library for cheminformatics functions (RDKit: Open-source cheminformatics; http://www.rdkit.org).

Table 1. Data Set Metrics Before and After Filtering

	data set size (K)	max token count	token vocab (set) size
before filter	1000	1281	205
after filter	400	91	54

Setting Null Baselines for Deep Variational Anomaly Generation. ML methods may not be justified if they do not outperform simpler conventional algorithms. Therefore, we investigate and benchmark six null models for their ability to generate invalid SELFIES. Since SELFIES is a token sequence and the VAE's decoder outputs token probability distributions per sequence position, we analogously explore random position-wise SELFIES generation for null models using the training data set. These models test representational robustness because the authors claim that "each SELFIES corresponds to a valid molecule, even entirely random strings".[12] We construct six generative null models that increasingly leverage SELFIES training distribution information.

Naive Random. To generate SELFIES naively, we randomly sample a sequence size (token count): $size \sim Uniform(\{min\ tokencount, min\ tokencount + 1, ..., max\ tokencount - 1, max\ tokencount\})$, where *min tokencount* and *max tokencount* are the minimum and maximum number of tokens in any training set SELFIES, respectively. Second, we fill each sequence position by uniformly sampling a token from our token set with replacement: $SELFIES[idx] \sim Uniform(\{[C], [N], ..., [{=}O]\})$ (Figure 2A). This naive generator constructs sequences without prior knowledge of training-set SELFIES token distributions using randomized sequence size and token positional assignment. Figure S4 pseudocode outlines the naive random generation algorithm.

Shuffle Random. We second generate SELFIES by drawing from the training distribution of sequences and shuffling tokens internally, randomly placing tokens into new sequence positions (Figure 2B). Since we sample initial SELFIES from the training set, this generator contains more training distribution information than naive random because the shuffling process preserves composition while altering the positional arrangement of tokens.

Index-Token Distribution Random. As SELFIES strings essentially describe a distribution of tokens over positions in a sequence, a SELFIES data set establishes a distribution of tokens per position over multiple sequences. Within a sequence, the horizontal arrangement of tokens conveys relational characteristics about connectivity (bonds) and structure (opening and closing points for rings and branches). Analogously, a "vertical" arrangement of tokens would describe a distribution of tokens for each position over all sequences in a SELFIES data set. Therefore, each sequence position observes a certain frequency of tokens, which conveys global properties of the data set. For instance, a data set could be biased toward SELFIES sequences featuring long carbon chains in their tails. In this case, token distributions of the trailing positions would indicate a high frequency of carbon tokens, expressing a data set property. To adapt this interpretation for generation, we organize the SELFIES training set as a matrix with rows indicating sequences and columns indicating tokens per sequence position. This defines a multinomial distribution of tokens per position that we sample to generate SELFIES on a positional basis (Figure 2C). From the matrix, we generate sequences by uniformly sampling a token per position (column), ignoring padded characters, to compile the final sequence (Figure 2C). This generator is also relatively representative of the training distribution as it directly samples tokens from the position-wise data set distribution. Figure S5 pseudocode outlines the index-token distribution random generation algorithm.

N-Bitflip Random-Mutation. Motivated by the mutation-based robustness-testing approach by Krenn et al. in the SELFIES paper,[12] we also introduce random mutations in SELFIES sequences by randomly flipping or replacing constituent tokens (Figure 2D). As in the shuffle random approach, we first sample a SELFIES string from the training data set uniformly at random. Then, we replace n randomly existing sequence tokens with different random tokens from our token set. Figure S6 pseudocode outlines the N-bitflip random-mutation algorithm as a function of the number of bits, n. For smaller values of n, this generator is highly representative of the training distribution because it introduces fewer mutations in strings directly sampled from the training distribution. We use $n = 1$ and $n = 10$ to create two N-bitflip null-generators, varying by the number of mutations.

Full Random-Mutation. We also implement a full random-mutation generator that extends the N-bitflip random-mutation generator. In this approach, n equals the length of the SELFIES sequence, which replaces all sequence tokens, changing it entirely to generate a new string. Figure S7 pseudocode outlines the full random-mutation algorithm.

RESULTS

Our primary goal was to test the representational robustness of SELFIES through variational autoencoding. By seeking to minimize validity, we test SELFIES's 100% validity assumption and its robustness. Moreover, a large pool of invalid SELFIES strings would yield insights into the representation's failure

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

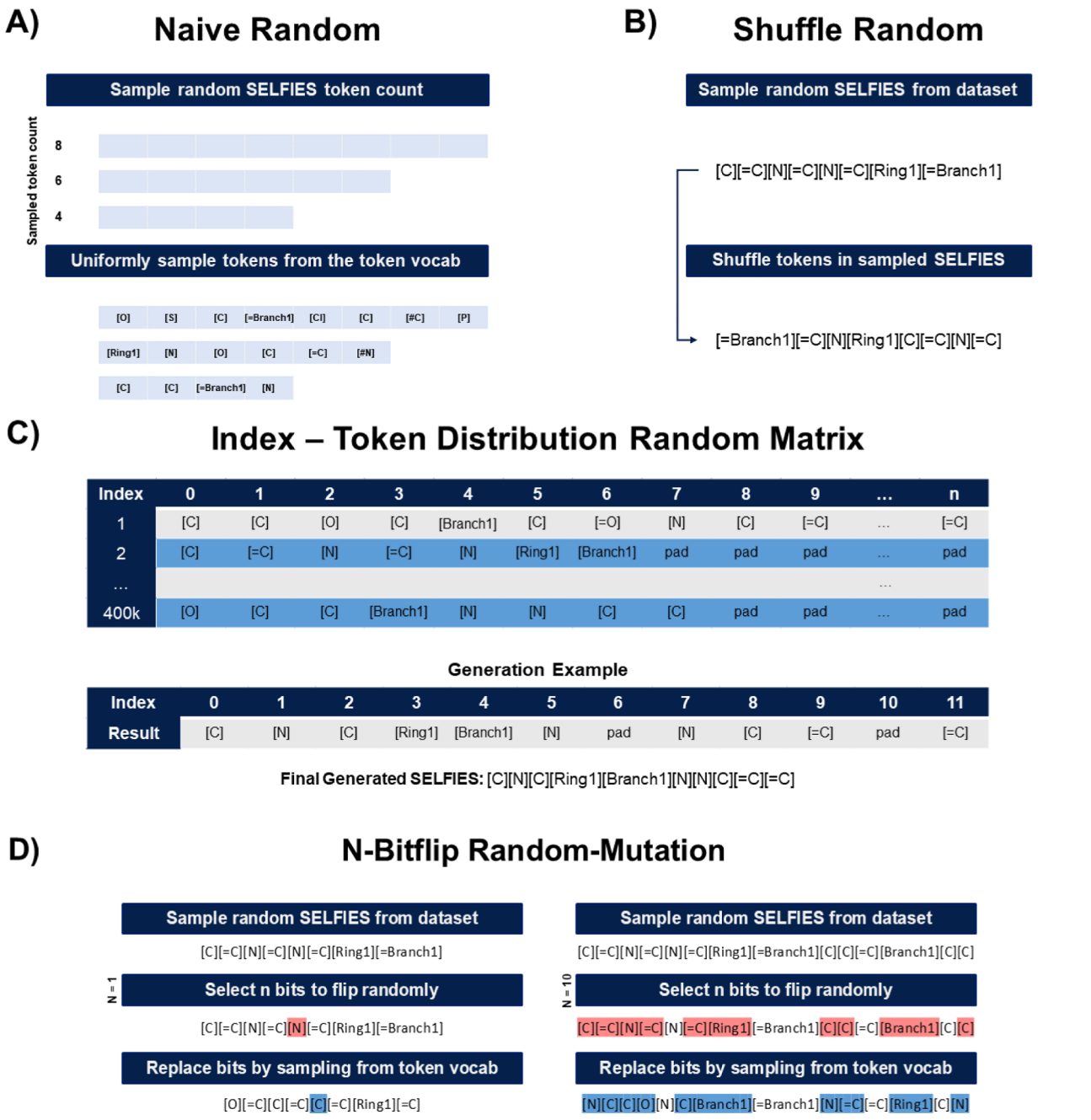


Figure 2. SELFIES generation process examples via null models. (A) Naive random generator samples a token count (uniformly between 1 and 91) and then fills each sequence position with a token sampled from the token set. (B) Shuffle random generator samples a SELFIES string from the training set and shuffles the tokens within the sequence to generate a new SELFIES. (C) Index-token distribution random generator applies a matrix interpretation to the SELFIES training set to randomly sample tokens from position-wise distributions, where n = *max tokencount*. Generated SELFIES discard padding characters. (D) N-Bitflip random-mutation samples SELFIES strings from the training set and then selects n bits (red) to be randomly replaced with a token from the token set (blue).

modes and factors contributing to anomalous criteria. Ideally, we expected to observe a self-organized latent space that clustered high- and low-validity regions.

We generated invalid SELFIES by radially surveying the VAE latent space. We randomly sampled points over 196-dimensional zero-centered hyper-sphere surfaces and decoded them to SELFIES strings. Figure S8 pseudocode outlines the algorithm used to perform radial sampling in the latent space. We assessed fixed-size-generated SELFIES sets for their validity percentage as a function of their generative (decoding) radius. To assess their validity, we converted SELFIES to SMILES and tested the SMILES's convertibility to molecular graphs or mixtures of molecular graphs using RDKit. For instance, if the SMILES violated a valence rule, RDKit raised an error identifying the atom and count by which the atom's explicit valence exceeded maximum permission. To ensure consistency and as a secondary check, we likewise assessed SELFIES validity using ChemWriter[a] and Smival[b] (Figure S9) along with manual inspection for selected strings (Figure 3, "Default Constraints"). This testing approach abides by the SELFIES definition of validity, which rests on its modules' converted SMILES complying with standard molecular rules.

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

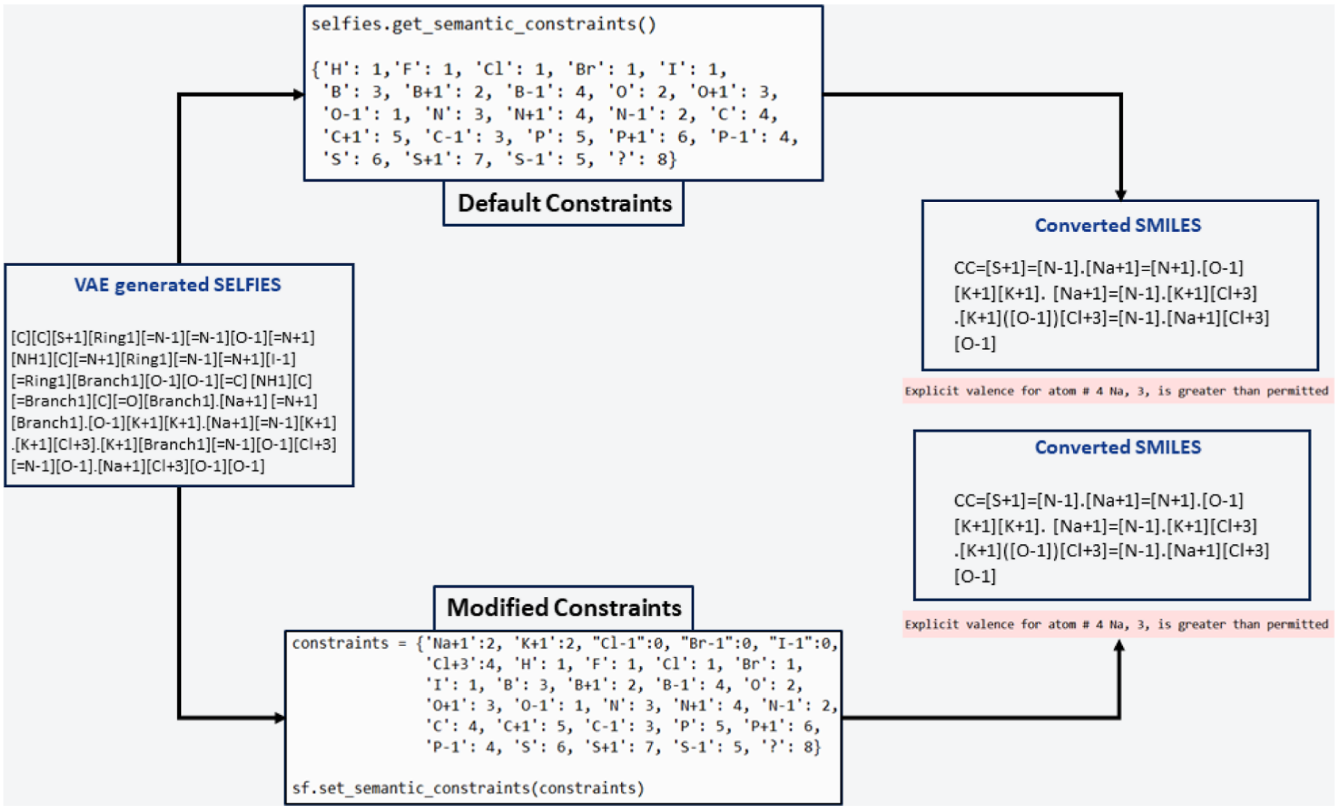


Figure 3. SELFIES validity checking. Example VAE-generated SELFIES string converted to SMILES using SELFIES module's default and modified constraints. The string validity is assessed using RDKit.

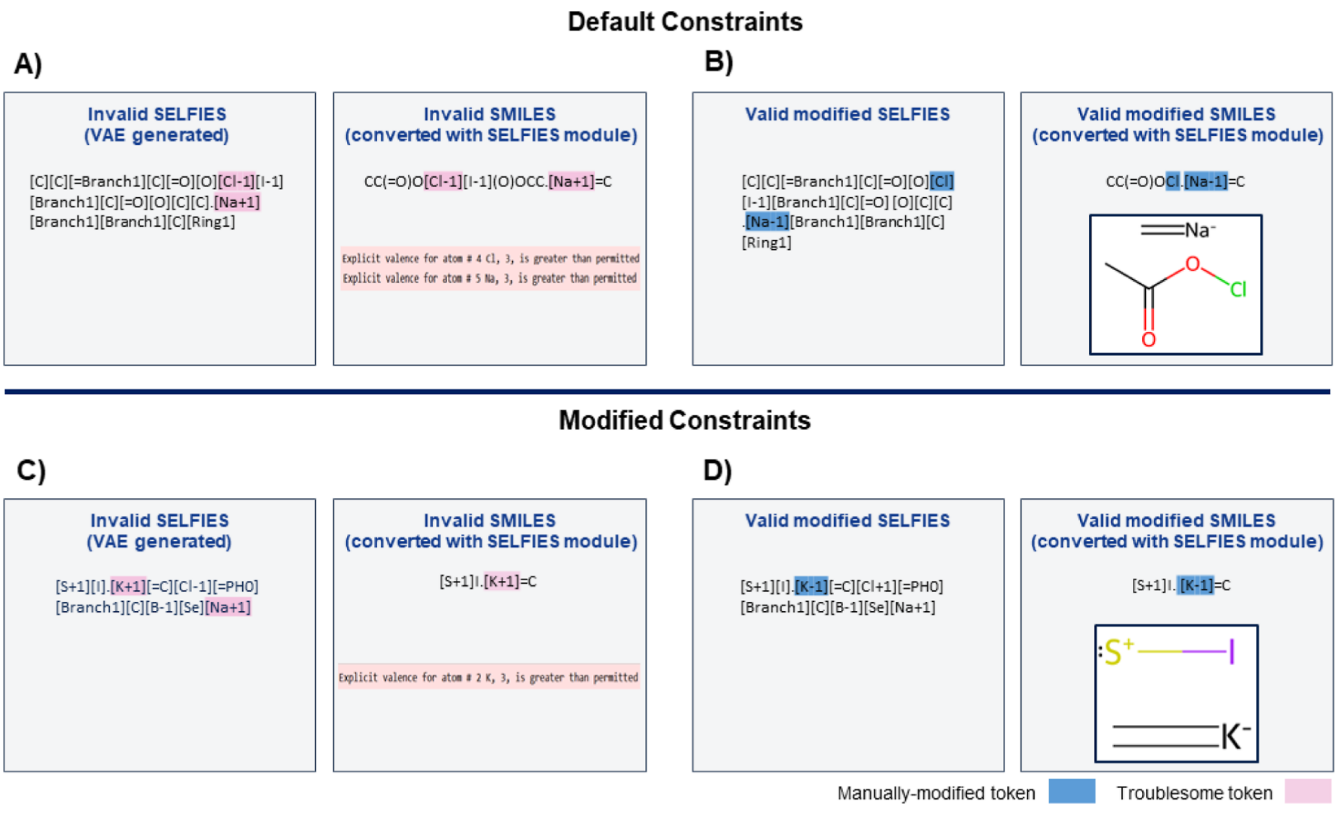


Figure 4. Evaluating SELFIES validity by token modifications and constraint settings. Default constraints: (A) SELFIES converted to SMILES with valence errors and (B) the same SELFIES with troublesome tokens manually modified to validate resultant converted SMILES. Modified constraints: (C) SELFIES converted to SMILES with valence errors and (D) the same SELFIES with troublesome tokens manually modified to validate resultant converted SMILES. Troublesome tokens and manually modified tokens are highlighted in pink and blue, respectively.

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

We identified SELFIES strings that generated atomic valence errors. Although the SELFIES-to-SMILES conversion module imposes corrections like atomic valence constraints to ensure SMILES validity, its default settings nonetheless raised explicit RDKit valence errors in some cases (Figure 4A). Although errors, most of these edge cases were easily correctable. We introduced a "Modified Constraints" correction (described below) that modifies invalid SELFIES by altering formal charges for atom tokens that raised errors in RDKit (Figure 4B). For instance, tokenized atom [Na + 1]'s explicit valence is 2, but within a generated invalid SELFIES (Figure 4A), the token's associated bonds and formal charge required the explicit valence to be 3, which exceeded maximum permission by 1. To remedy this, we manually modified [Na + 1]'s formal charge. Similarly, we neutralized [Cl − 1] to address the valence errors. The module converted the resultant modified SELFIES to valid SMILES, with the sodium double-bonded and ionized, the chlorine correctly bonded, but other tokens discarded, forming a disconnected structure (Figure 4B). This valence/charge modification process is consistent with the bond-order correction process the SELFIES module already employs, which alters the bond order of tokenized atoms to obey the valence rules.

By manually inspecting the VAE-generated invalid SELFIES strings, we identified six "troublesome" atom tokens (three cations and three anions) that consistently invalidated downstream SMILES strings (Table 2). Formal charge and neighboring atoms enforced valences on them, exceeding maximum permissions, raising RDKit flags. Conveniently, the SELFIES module allows users to set customized constraints using a specific function (*set_semantic_constraints*()). The module dynamically builds its derivation rules incorporating the custom constraints, which set the maximum permissible number of bonds each atom can form in a molecule.[46] We used this capability to set custom valence constraints for the six troublesome atom tokens, resolving four cases. However, this "Modified Constraints" customization failed to solve valence errors for the remaining two atom tokens, designated as our filtered troublesome token set (Figures 3 and 4C).

Table 2. Set of Troublesome SELFIES Atom Tokens Discovered After SELFIES-to-SMILES Conversion Error Analysis Using Default Constraints[a]

	Troublesome SELFIES atom token
1.	[Na+1]
2.	[K+1]
3.	[Cl+3]
4.	[Cl-1]
5.	[Br-1]
6.	[I-1]

[a]The red cells indicate tokens that remained troublesome despite setting custom valence constraints ("Modified Constraints"); the blue cells indicate tokens whose customization resolved conversion issues.

The "troublesome" SELFIES tokens persistently resulted in silently invalid SMILES despite attempts to correct them using custom valence constraints, satisfying the definition of representational anomalies. We could manually correct them in a context-specific way but not automatically using the SELFIES representation's rule set or its module's capabilities (Figure 4D).

Invalid SELFIES strings contained at least one troublesome token, but valid SELFIES may contain them as well (Tables 3 and 4). The troublesome tokens only invalidate a SELFIES string based on neighboring tokens' associated valence and characteristics. Accordingly, editing the token itself or adjusting neighboring tokens' valence/charge or atomic composition can correct an invalid SELFIES string. SELFIES-to-SMILES conversion successfully corrects some troublesome token scenarios contingent on neighborhood factors, such as the arrangement of structural disconnections indicated by periods (Table 3). We quantified and compared the troublesome token containment (TTC; one or more present) in the training set and the VAE-generated set of 1.8 million SELFIES (Table 4). While less than 1% of the 100% valid training set SELFIES contained troublesome tokens, almost 59% of the VAE-generated SELFIES set (decoded in hyper-sphere of radius = 180.0) contained troublesome tokens.

We investigated the VAE as a representational anomaly generator (i.e., as a smart generative black-box fuzzer) by decoding SELFIES strings as a function of the latent generative radius and assessing the strings for validity. We also mapped validity vs radius trends evaluated for incompletely trained models at epochs 0 (untrained VAE initialized using Glorot-uniform scheme[54]), 1, 5, and 7 (Figure 5A). These illustrate the VAE's growing ability to discriminate and cluster SELFIES by validity over the course of training. Models at epochs 0 and 1 fall short of the best null model (full random-mutation) at minimizing validity, while models at epochs 5, 7, and 15 (final convergence) outperform the null model at varying radial domains. The training and validation loss plots for the VAE can be found in Figure S10. To assess the VAE's reconstruction performance on seen and unseen molecules, we also calculate mean categorical reconstruction accuracy (Figure S11, pseudocode) for training and validation set SELFIES. We found the VAE's mean categorical reconstruction accuracy equivalent in the training and validation sets. The accuracy value was 0.921 for the training and 0.920 for the validation set (Table S1). Figure S12 shows distributions of categorical accuracy values for the training and validation sets. Encoded-decoded example pairs of SELFIES molecules reconstructed by the VAE can be found in Figure S13.

To ensure expansive latent space coverage and a relatively global trend evaluation, we plot the SELFIES set validity percentage as a function of generative radius ranging from 0.0 to 1000.0, in the final model (Figure 5B). The "Modified Constraints" (customized, Figure 5B) procedure slightly increases validity but does not fully address the representational anomalies arising from troublesome tokens (Figure 6A,B).

Speaking of latent space organization and validating clustering, a two-dimensional projection (via principal component analysis, PCA) of the 196-dimensional latent space likewise shows that invalid SELFIES are more likely to arise from points decoded at high latent-space radii (Figure 5C). The final converged VAE model comprises several key latent space regions with consistent properties, as a function of radius R:

1. Normal-anomalous (valid-invalid SELFIES) boundary: $R = 13.0$
   - $R < 13.0$ is a purely valid SELFIES zone because strings decoded from this region demonstrate 100% validity, while points decoded in domain $R > 13.0$ generate SELFIES sets with varying validity percentages.
2. Global minima for validity percentage: $R = 61.0$

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

Table 3. Example SELFIES Strings from the Training and VAE Generated Sets by Validity and Troublesome Token Content (TTC), with Troublesome Tokens Bolded[a]

SELFIES Set	Validity	TTC	Example
Training	Valid	Yes	[C][C][C][O][C][=Branch1][C][=O][C][=C][C][=C][C][=C][Ring1][=Branch1][O][S][=Branch1][C][=O][=Branch1][C][=O][N-1][C][=Branch1][C][=O][N][C][=N][C][Branch1][Ring1][O][C][=C][C][Branch1][Ring1][O][C][=N][Ring1][#Branch2].[**Na+1**]
		No	[C][C][O][C][=C][C][=C][C][=C][Ring1][=Branch1][C][N][C][C][N][Branch1][N][C][C][=C][C][=C][Branch1][C][C][O][Ring1][=Branch1][C][Branch1][Ring2][C][C][O][C][Ring1][S]
VAE-generated SELFIES	Valid	Yes	[C][=C][C][=N][C][=Ring1][Ring1][N][Ring2][Ring1][C][=Branch1][O-1][=Ring1][Ring1][=Branch1][C][O-1][O-1][O-1][Ring1][Ring1][Branch1][C][Cl][=C][Ring2][NH1][C] [=N+1][Ring1][O-1][C][=O][O-1][NH1][**Na+1**][Ring1][C][=O][Ring1][O-1][O-1]
		No	[C][O][C][=Branch1][C][=O][C][=C][NH1][C][=N+1][C][=C][C][=C][Ring1][=Branch1][Cl][O][C][=Ring1][#Branch1]
	Invalid	Yes	[C-1][#N+1][C-1][=N-1][=N-1][=N-1][=N-1][=P][=Ring1][=N-1][=P][=N-1][=N-1][=N+1][=N-1][Cl+3][=N-1][O-1][#N][=C][Ring2][Ring2][Ring1][C][O-1][=N+1][O-1][=N+1][=N+1][=N-1][C][O-1].[O-1].[O-1]..[**K+1**][=N-1][=N-1][Cl][**K+1**].[Cl][Cl+3][Branch1][=N+1][O-1][=Ring1][=N-1][=N-1].[O-1][Ring1]..[O-1][=Ring1][Cl+3].[O-1].[**K+1**][**K+1**].[**K+1**].[O-1][O-1][O-1].[O-1][O-1]

[a]The training set SELFIES are 100% valid, regardless of the TTC. Valid and invalid SELFIES are highlighted in blue and red, respectively. The VAE-generated SELFIES were decoded by sampling within the volume of a hyper-sphere of radius = 180.0.

Table 4. Quantifying and Comparing TTC in Training and VAE-Generated SELFIES Sets[a]

SELFIES set	validity	TTC	number of SELFIES	
			subtotal	total
training	valid	yes	1986 (0.50%)	400,000
		no	398,014 (99.50%)	
VAE-generated SELFIES	valid	yes	239,656 (13.06%)	1,834,560
		no	850,458 (46.36%)	
	invalid	yes	744,446 (40.58%)	

[a]The training set SELFIES are 100% valid, regardless of the TTC. The VAE-generated SELFIES were decoded by sampling within the volume of a hyper-sphere of radius = 180.0.

- VAE minimizes decoded SELFIES set validity percentage in the final model to 11.2% at this radius, with invalidity percentage correspondingly maximized at 88.8%. Implicitly, this radius indicates the model's upper performance bound as a representational anomaly generator.

3. VAE applicability domain as a representational anomaly generator relative to the best-performing null generator: $R > 29.0$

- In this radial domain, the VAE outperforms the best null generator (full random-mutation) at minimizing validity percentage in generated SELFIES sets. SELFIES sets generated at every radius greater than 29.0 exhibit (variably) lower validity percentages than those generated by the full random-mutation null model, which generates SELFIES sets at a constant validity percentage of 72.9% (Table 5). Owing to superior performance at validity minimization, this radial domain effectively establishes the VAE's applicability domain as a representational anomaly generator.

We generated SELFIES via the six null models to benchmark the VAE model's anomaly generation performance and evaluated them for percentage validity. Of the 10,000 strings generated using the shuffle and index-token distribution random models, almost 100% are valid, while the 1-bitflip random-mutation and the 10-bitflip random-mutation methods generated SELFIES sets with 96.66% and 85.48% validity, respectively, making them weak robustness testers. In contrast, the naive random model, which contained the least training data set information in its formulation, resulted in a generated SELFIES set with no more than 74.35% validity, while the full random-mutation method generated 72.93% valid SELFIES strings—comprising a significantly better null test than the other baseline models for representational robustness (Table 5 and Figure 6C). Consequently, we chose the full random-mutation generator as the primary null baseline for representational anomaly generation. The null and VAE models generated both valid and invalid SELFIES containing troublesome tokens (Figures 7 and 8).

We observe that the SELFIES module deals with troublesome token-containing but valid SELFIES in two ways: (1) It transforms the string by discarding tokens sequentially until the troublesome token is correctly accommodated at a disconnection point in the resultant molecular mixture (Figure 9A); or (2) It discards the troublesome token altogether to validate the string (Figure 9B). By contrast, the module deals with out-of-scope SELFIES strings in either of two ways, depending on the validity of the characters: (1) It returns an empty string if the SELFIES sequence does not contain any atom token but only contains branch or ring-indicating tokens; or (2) It returns a Python None object if the character is an invalid, unrecognized token (Figure S14).

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

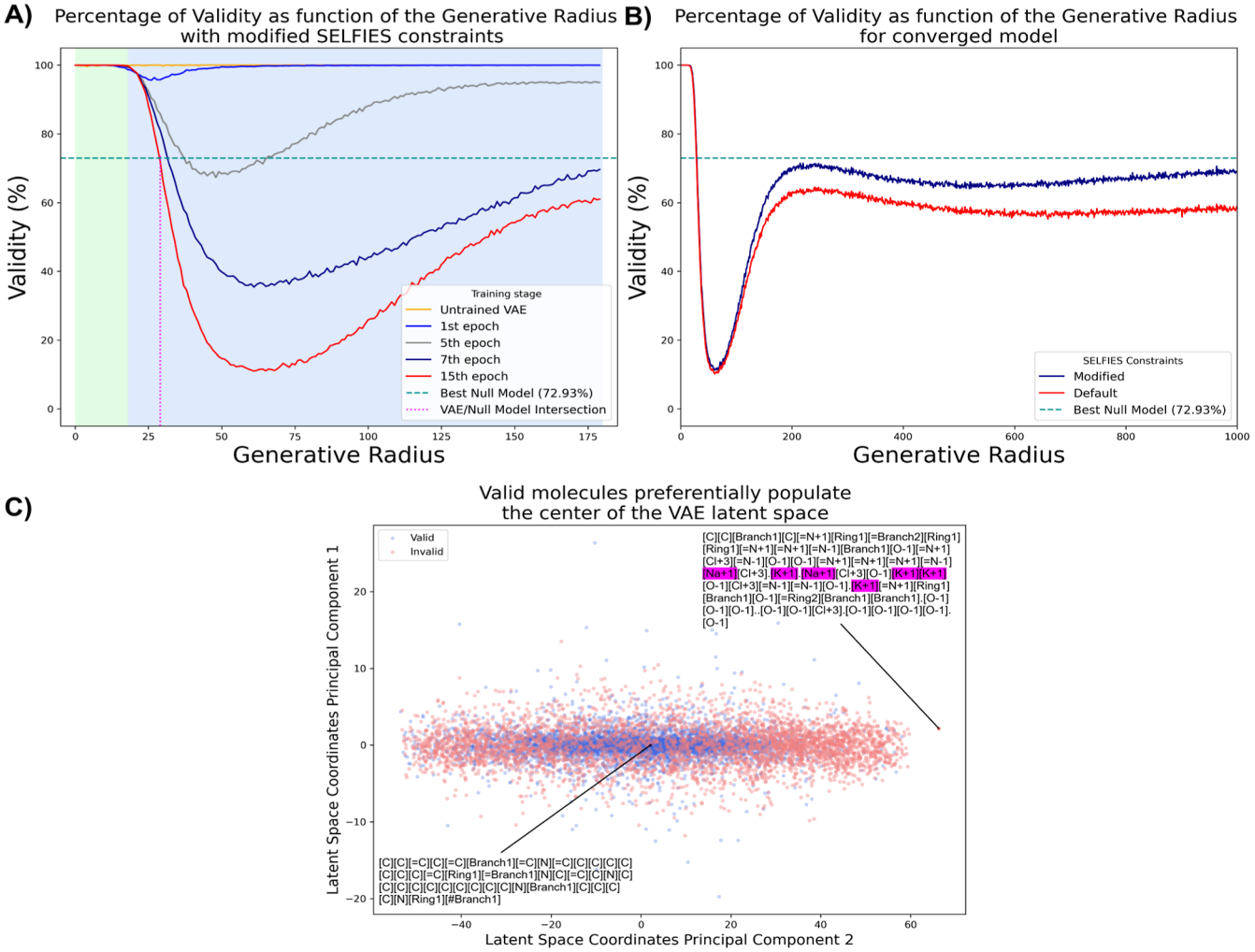


Figure 5. Generated SELFIES sets of size 10,000 for each radius (R) except $R < 6.0$ (sphere surface area and molecular density are too low to generate 10,000 unique SELFIES strings from this radial domain). (A) Percentage of validity in generated SELFIES sets as a function of radius from $R = 0.0$ to 180.0 over various stages of training (untrained model to model trained at1st, 5th, 7th, and 15th epochs). The green region indicates the normal purely valid SELFIES space, while the blue region indicates the anomalous invalid SELFIES space. The magenta dotted line indicates the region where the VAE begins to outperform the naive random model at validity minimization. (B) Percentage of validity by generative (decoding) radius from $R = 0.0$ to 1000.0 in the final converged model for both default (red) and modified (blue) constraints. The dark cyan dashed line indicates the naive random null model for both plots. (C) Two-dimensional PCA projection plot of 196-dimensional vectors (sampled within the volume of hyper-sphere of radius = 61.0) decoded to 10,000 SELFIES strings and color-coded by validity. Valid SELFIES are concentrated toward the center, and invalid SELFIES are scattered to the periphery. Black lines indicate examples of valid and invalid SELFIES (troublesome token highlighted) decoded from high-dimensional points.

DISCUSSION

Molecular string representations such as SMILES,[11] SELFIES,[12] and DeepSMILES[13] describe molecules using a linear sequence of tokens. This approach limits chemical expression given the inherently graphical nature of molecules, but tokens as building blocks support dimensionality reduction, memory efficiency, and ready machine readability. Invalid SMILES or graph outputs limit generative modeling's use; generative latent variable models learn larger useful chemical spaces if the representation ensures maximal validity. SELFIES is a major landmark in this effort because it fundamentally addresses common invalidity sources in the traditional SMILES representation by devising specialized tokens and corrective mechanisms.[12] Given the importance of SELFIES and progress in molecular representation development, we emphasize the complementary need for investigating advanced representational robustness testing methods.

SELFIES addresses the issue of invalid molecular representations, particularly for outputs of generative models, where relatively common but grammatically fragile string representations such as SMILES lead to wasted computational resources and reduced efficiency.[12] Ideally, by eliminating the potential sources of errors, SELFIES introduces a useful feature into a latent space: every point would correspond to a valid molecule. This allows the model to focus on generating molecules with the desired properties. While SELFIES largely succeeds, we found edge-case anomalies that suggest room for improvement. These anomalies can propagate through downstream workflows, emphasizing the importance of rigorous tests to ensure reliability.

Developing new molecular representations and methods for exploring chemical spaces is pivotal in chemical informatics. Hence, the rigorous evaluation of representations to verify intended functionality is essential. Indeed, Krenn et al.[12]

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

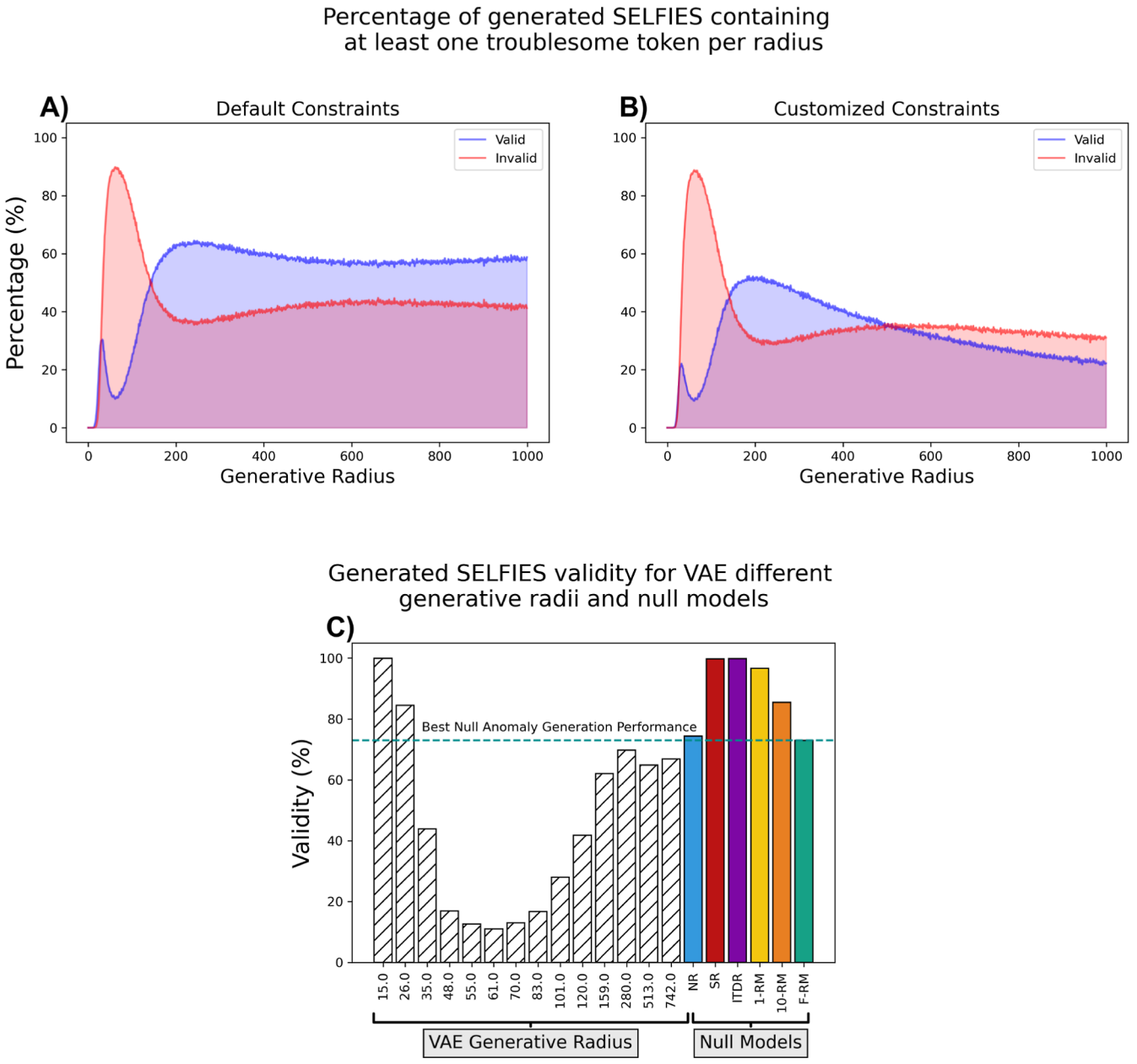


Figure 6. Percentage of generated SELFIES strings containing at least one troublesome token as a function of generative radius R and validity status using (A) default and (B) modified constraints. We generated 10,000 unique SELFIES strings at each radius, except for $R < 6.0$. (C) Bar chart summarizing generated SELFIES validity percentage for six models: hatched bars depict various VAE generative radii R, while colored bars represent the naive random model (NR), shuffle random model (SR), index-token distribution random model (ITDR), 1-bitflip random-mutation (1-RM), 10-bitflip random-mutations (10-RM), and full random-mutation (F-RM). A lower validity percentage indicates better performance at representational anomaly generation. Bars under the dashed red line outperform the best null model (F-RM).

conducted a series of empirical tests akin to fuzz tests to assess SELFIES robustness. By reapproaching robustness testing through the cybersecurity discipline of fuzz testing leveraging VAEs, we found that SELFIES has edge-case anomalies that can affect its intended use with generative models. For instance, the model generates SELFIES strings that map to chemically invalid SMILES strings without raising error messages. Examples highlight implausible chemical scenarios, such as a metal and a nonmetal in the SMILES/molecule forming a covalent bond and an explicit valence scenario that exceeds chemical acceptability. This creates inefficiencies in the chemical informatics processes and downstream molecule generation pipelines that assume, based on SELFIES' core claim, that every SELFIES string yields a valid molecule. Such behavior creates "silent" errors in calculations or could halt pipelines with RDKit SMILES reading errors such as "Explicit valence for atom #15 Na, 2, is greater than permitted". Subsequent operations on the now-missing generated string or molecule, such as property calculations, would fail.

Molecular representations encode molecular structure in chemical informatics tasks, including molecular generation and design, molecular property assessment, and computational chemistry. These numerical and statistical approaches enable the representation and design of chemical phenomena. Fragile representations that yield invalid molecules detract from this use. Verifying representational robustness by explicit, direct, and

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

Table 5. Generated SELFIES Sets Validity Percentage by Generator Types for a Sample Size of 10,000[a]

generator type	generator	validity (%)
null	naive random	74.35
	shuffle random	99.74
	index-token distribution random	99.88
	1-bitflip random-mutation	96.66
	10-bitflip random-mutation	85.48
	full random-mutation	**72.93**
deep learning	VAE (generative radius = 61.0)	**11.24**

[a]The best validity-minimizing null model type and the VAE's validity-minimizing generative radius are **bolded**.

thorough techniques like fuzz testing greatly impacts projects or processes that hinge on chemical representations. Fragile representations can directly affect chemical informatics tasks in several ways, such as

- Inefficiency in chemical library use and design: fragile representations can create low validity rates, compromising the effective exploration of chemical space. For example, the validity percentage of generated molecules, 71.9% with SMILES per Krenn et al.,[12] directly reflects computational resources wasted (28.1%) on invalid structures. This inefficiency becomes critical when valid molecules do not necessarily achieve desired properties, further limiting the discovery of synthesizable, useful compounds.
- Chemical informatics pipeline error: invalid or edge-case molecules can break or halt computational pipelines, causing software packages like RDKit to return Python errors. These issues can hinder workflows and demand significant debugging efforts.
- Silent invalidity: in pernicious cases, invalid structures might not trigger immediate errors but instead be skipped or misinterpreted, leading to incorrect conclusions about a model's distributions or generative-space capabilities. This "silent failure" can mislead scientific or discovery-oriented insights, compromising the reliability of one's work.
- Model interpretability: fragile representations that result in invalid generated molecules can negatively affect the explanation and interpretability of the internal working process of generative models in chemistry.

By applying fuzz testing techniques to molecular representations, we aim to uncover weaknesses and systematically make these representations more robust. Additionally, robust representations enable closer integration with ML models, which are often sensitive to the quality of the input data. Ensuring the reliability of molecular representations (and chemical representations in general) enhances predictive

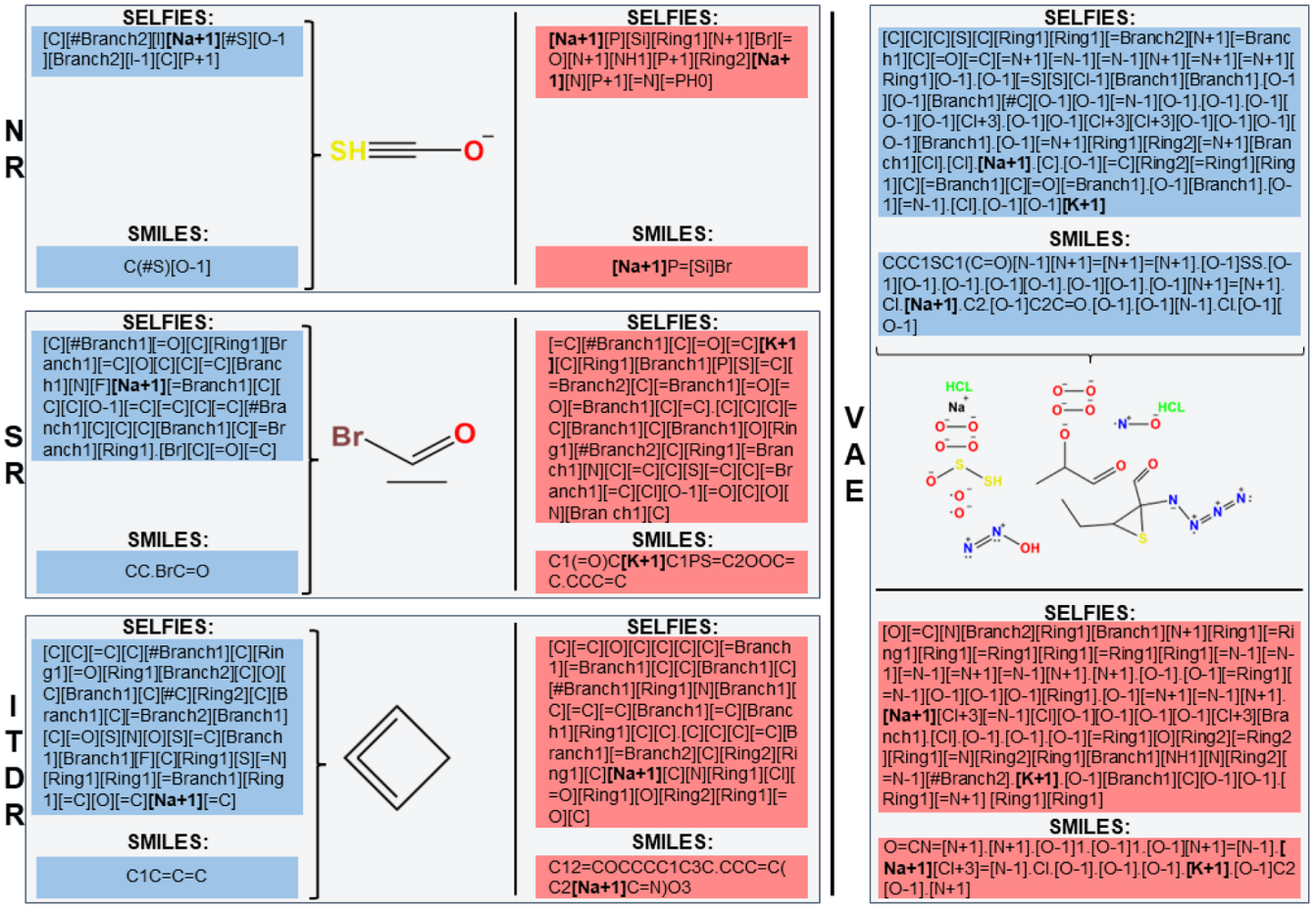


Figure 7. Examples of generated valid (blue highlight) and invalid (red highlight) SELFIES with converted SMILES containing troublesome tokens, using three null generators (NR = naive random, SR = shuffle random, ITDR = index-token distribution random). Troublesome SELFIES atom tokens are **bolded**. Molecular structures display where valid.

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

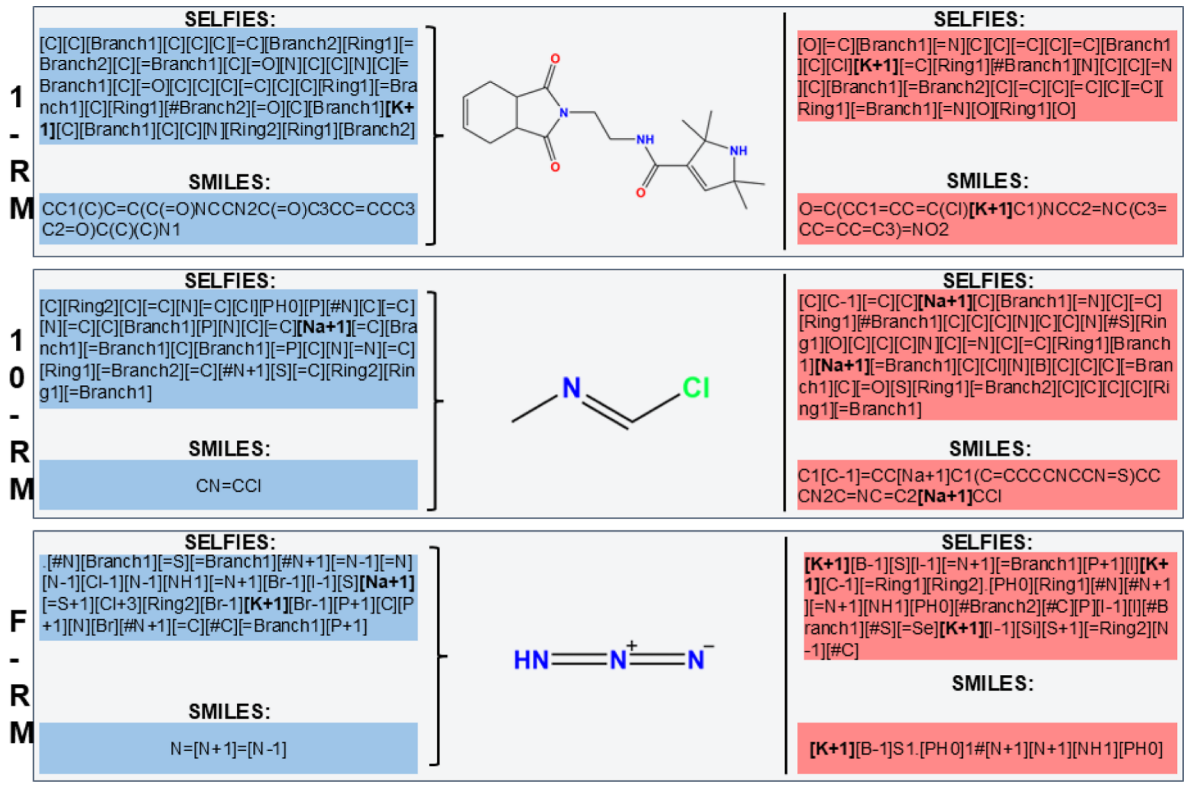


Figure 8. Examples of generated valid (blue highlight) and invalid (red highlight) SELFIES with converted SMILES containing troublesome tokens, using three null generators (1-RM = 1-bitflip random-mutation, 10-RM = 10-bitflip random-mutation, F-RM = full random-mutation), and the VAE. Troublesome SELFIES atom tokens are **bolded**. Molecular structures display where valid.

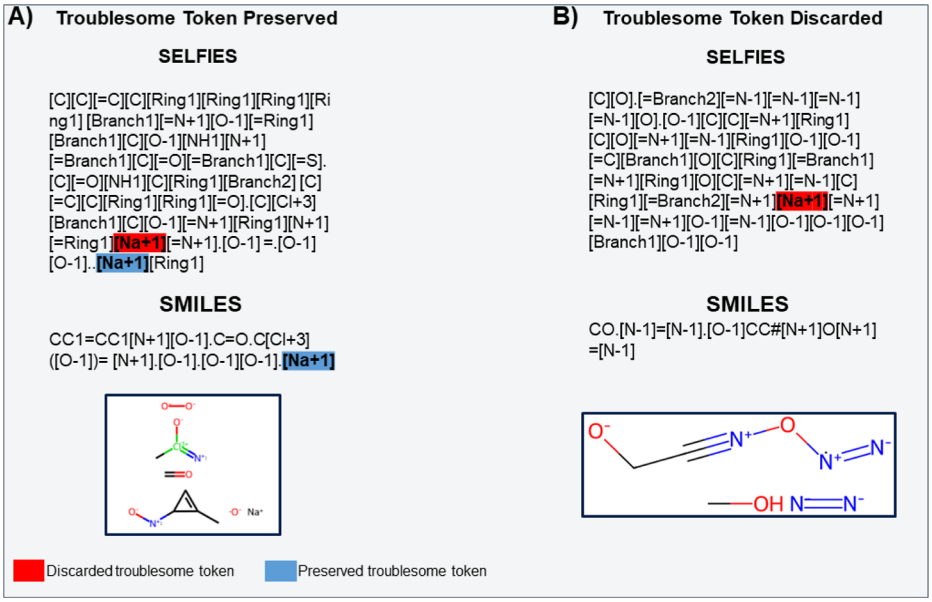


Figure 9. Two cases of valid SELFIES containing troublesome tokens. (A) VAE generated SELFIES (generative radius $R = 98.0$) with troublesome token preserved (in blue) on converting the SELFIES to SMILES. (B) VAE generated SELFIES (generative radius $R = 98.0$) with troublesome token discarded (in red) on converting the SELFIES to SMILES, displayed along with the molecular structure.

accuracy, streamlines molecular design workflows, and accelerates advancements in computational chemistry.

We used generative approaches of varying complexity to permute tokens within SELFIES sequences to stress string representation robustness by finding failure modes when converting to SMILES. Divergence from the purely valid training set imposed stressors on SELFIES's corrective conversion mechanisms. Real-world molecules range from complex natural products to synthetically accessible small molecule drugs, whose string representations explore token sequence permutations[48,55] in a space of 54^{91} strings at our token set size and maximum SELFIES length. Variational autoencoding converts inherently discrete chemical strings into continuous latent representations that capture salience and minimize

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

complexity for varied novel sequences. By sampling and decoding across hyper-spheres in the VAE's latent space, we discovered a radial domain (>29.0) that consistently outperformed the best null generator (full random-mutation) at minimizing validity in generated SELFIES sets of fixed size (10,000). The lowest validity percentage we found was 11.24% at a generative radius of 61.0. We infer the VAE distributes SELFIES validity unimodally over radius by concentrating training (of normal valid strings) within the unimodal unit Gaussian prior. This organized two key latent radial domains: (i) purely valid SELFIES in $R < 13.0$ and (ii) validity decreasing monotonically in $13.0 < R < 61.0$. Since the training objective's KL loss component pulled the training set's latent embeddings toward the origin and the training set was purely valid, observing pure validity in regions centered closely around the origin ($R < 13.0$) was sensible. Moreover, only 0.5% of the training set SELFIES contained troublesome tokens and the VAE only began to introduce troublesome tokens into SELFIES sets generated at radii greater than 13.0, with radius 61.0 decoding to the highest percentage of strings containing troublesome tokens (88.8%). Property distribution plots of generated SELFIES can be found in Figures S15 and S16 for a set of profile-representative generative radii. Additionally, Figure S17 presents a stacked plot showing the percentage distribution of valid and invalid SELFIES for 10,000 sets across different generative radii, with valid SELFIES further categorized into mixtures and nonmixtures. This highlights how radial sampling influences, besides validity, the proportion of mixtures versus nonmixtures on the latent space hyper-spheres. Figure S18 extends the property distribution analysis to 1 million decoded SELFIES for the set of profile-representative generative radii.

Naive random-mutation and full random-mutation significantly outperformed the other four null-generators at minimizing the validity percentage in generated SELFIES sets. This may be because the naively generated SELFIES were the least informative of the training distribution of purely valid SELFIES. Naive sampling lacked prior knowledge of the training set's position- and sequence-wise token distributions. Unlike the other two null generators, it randomly sampled from the token set, with the sequence token count selected uniformly randomly from the training strings' token count range. Given its degree of randomness, the naive random model maximally explored the space of possible sequences. The full random-mutation method performed only marginally better than the naive random method. While both methods followed identical procedures for inserting tokens into SELFIES sequences via random sampling from the token set, the full random-mutation method generated longer sequences. Unlike strings generated by naive-random, full random-mutation strings followed the distribution of the training string length. Consequently, the longer strings were more likely to include a troublesome token, which made them slightly more susceptible to invalidity (Figure S19). Furthermore, the 10-bitflip random-mutation method generated more invalid strings than its 1-bitflip counterpart, consistent with expectations. Unlike the VAE, there was no apparent advantage in incorporating nuanced token distribution information into the null models.

On tracing sources of error in invalid SELFIES, we identified two "troublesome" atom tokens ([Na + 1] and [K + 1]) that were consistently bonded beyond their valid valences and consequently yielded invalid SMILES. However, the errors were fixable using manual valence correction consistent with the module's otherwise automatic bond-correcting approach. TTC was one of the two prerequisites for string invalidity. Each invalid SELFIES contained at least one troublesome token whose bond count exceeded the valid valence in the converted SMILES. Hence, troublesome tokens alone were insufficient; their atomic neighborhood also contributed. Indeed, 13.1% of the VAE's *valid* SELFIES contained at least one troublesome token. Examples of SELFIES strings that are valid *despite* containing at least one troublesome token, along with at least one mixture-indicating disconnection point, can be found in Table S2. By tracing conversion logic, we observed that the SELFIES module successfully discarded troublesome tokens or their neighboring tokens to introduce a disconnection point in the valid SMILES. However, the conversion module could not universally correct the valences/charges associated with the two troublesome atom tokens. By contrast, SELFIES demonstrated 100% robustness for generated strings without troublesome tokens. We also observed the same errors in the module's earlier version, 2.1.0.

These identified behaviors suggest that the SELFIES module's derivation table and self-correcting features, while efficient, lack a comprehensive set of chemical rules that ensure accurate 100% valid molecule generation. For instance, the VAE identified a set of troublesome tokens that reflect violations of chemical principles. This highlights areas where the SELFIES' understanding of molecular rules requires further refinement. For example, the model generated the SELFIES string [Na + 1][Branch1][Branch1][C][Ring1], which decodes to the SMILES string [Na + 1]=C. Sodium does not form covalent double bonds with carbon, representing a chemically implausible bonding scenario. Furthermore, this construction assigns sodium an explicit valence of 3, which exceeds its chemically acceptable range. Similarly, the same issue arises in other generated SELFIES strings such as "[C][C][S][=Branch1]-[C][O−].[K + 1].[O][=C][Branch1][C][=C][C]-[Branch1][=Branch1][C][#N][O][C][Ring1][=Branch1]-[C][K + 1]" or "[O][=C][Branch1][C][C][C][C][=C]-[NH1][C][=Ring1][Branch1][=C][=C][=Branch2]-[Ring1][Branch1][Cl].[C][=N + 1][Ring1][Ring1][N][C].[Na + 1][O][Ring1][Ring1]", decoding to the SMILES strings "CCS[O − 1].[K + 1].O=C(C)C(C#N)[K + 1]" and "O=C(C)C=1C=C[NH1]C=1C=CCl.C=[N + 1]NC2.[Na + 1]O2", respectively. These cases involve chemically implausible bonding scenarios as neither sodium nor potassium forms covalent bonds with carbon or oxygen. Furthermore, in both cases, the explicit valence of 2 exceeds the chemical acceptability. The fact that these SELFIES strings decode to SMILES strings without returning errors or warning messages reveals failures in the SELFIES module's derivation table and self-correcting features, which challenges its core claim that "every SELFIES string corresponds to a valid molecule".[12] Therefore, updating SELFIES to comprise a broader and more chemically accurate chemical space would benefit both its molecular representation framework and researchers relying on it.

In our findings, the full random-mutation and VAE-generated SELFIES sets both uncovered the troublesome token set, effectively stress-testing representational robustness and revealing the root cause of string invalidity. However, the VAE ($R = 61.0$) generated far more (88.76%) invalid SELFIES strings than the full random-mutation model (27.07%) and outperformed the full random-mutation model in an applicability domain of $R > 29.0$ onward. Since invalidity was doubly conditioned on TTC and local token neighborhood, the VAE generated more SELFIES-invalidating possibilities for regular tokens in the neighborhood of troublesome tokens. Furthermore, its radial

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

latent space organization enables tunable SELFIES set generation by the desired validity percentage along a continuum. The VAE can be interpreted as a collection of generators, each with its own generative property defined in terms of validity. The latent space exhibited radial organization partly because the KL loss component of the training objective operates radially by minimizing the magnitude of the encoded latent vectors.

Although the VAE was the most effective approach for validity minimization and offered advantages such as continuous radial organization, it has limitations. The generation process is computationally expensive, requiring inference via a million parameters in the decoder. In time complexity, random vector sampling and decoding are approximately 360 times slower than the fastest null generator, shuffle random. In space complexity, the decoder consumes approximately 5000 times the memory of the cheapest null generator, the index-token distribution random method (Table S3). Additionally, deep learning models require large amounts of training data to uncover meaningful patterns.[56] Hence, data availability can be a major obstacle to successfully adopting a VAE, and deep learning methods in general, to fuzz-test molecular representational robustness.[57] By contrast, null generators such as the naive-random method only require a token set and two parameters for the range (min−max) of sequence lengths to generate SELFIES strings for effective fuzz-testing. Moreover, deep learning models lack ready interpretability and transparency due to their black-box nature,[58] requiring substantial explainable AI analyses to interpret. Similarly, VAE models require effort to tune. For instance, an effective way to further leverage the VAE's fuzz-testing abilities might be to disentangle the conditioning of the latent space such that troublesome token composition in generated strings can be altered along specific latent dimensions[59] or to bias the validity-minimizing radius toward certain values. Furthermore, gradients dictate the generative potential of deep learning models. We generated SELFIES sets of size 10,000 at 994 evenly spaced radii from 6.0 to 1000.0 with a uniqueness of 1.0 (fraction of generated unique strings). However, we expect the model to approach saturation by degeneracy at higher radii, decreasing the uniqueness. This is because geometry (volume and surface area) and gradients (decoder output sensitivity to perturbations in decoded vector) limit the number of unique strings generated by sampling within or over hyper-spheres. We observe lower uniqueness at small radii ($R < 6.0$) and latent regions with small gradients, such as those at dramatically *high* radii. Although we can permutationally estimate the total string count (normal and anomalous) produced by the null generators, precisely estimating it for the VAE would require infinite sampling. Therefore, we compared anomaly generation methods by the percentage invalidity within a predefined sample size applied uniformly across the four approaches.

These results depended on our training set (ChEMBL v.29 subset), model hyperparameters, and the latent space surveying strategy. In the SELFIES paper,[12] the authors trained a SELFIES-VAE to demonstrate a 100% valid chemical latent space. They trained on the QM9 benchmark data set[60,61] and decoded molecules by hyperplanes in a 241-dimensional latent space. They tested the robustness by introducing random mutations locally at select positions. We offer an alternative case of a VAE that learns a latent space revealing representational failure modes by generating anomalous SELFIES strings on hyper-spherical decoding.

Future work could explore ways to optimize VAEs for anomaly generation. Exhaustive search is rarely feasible, but tools like Optuna could empirically tune model architecture and hyperparameter choices to minimize decoding validity.[62] This may be relatively straightforward, as validity *decreased* with model training (Figure 5A)—a counterintuitive result that may arise because the training solely focused on valid SELFIES, with no explicit feedback regarding invalidity. Furthermore, the validity-minimizing radius, and hence the VAE's applicability domain, also depends on hyperparameter choice. We hypothesize that increasing the beta factor would reduce the validity-minimizing radius due to the latent compression effects induced by the KL loss component.[52] Furthermore, if known beforehand, a conditional VAE could leverage normal and anomalous binary labels to generate data biased by class.[63] Alternatively, a triplet-VAE combines metric learning loss with VAE loss to cluster latent embeddings by labeled properties.[64] Finally, other generative latent variable models, like generative adversarial networks (GANs),[65] normalizing flow models, or latent diffusion models[66] could learn latent spaces over discrete sequences like SELFIES.[67]

CONCLUSIONS

This study applies seven generative models to search for anomalies in a popular string-based molecular representation, SELFIES, which is thought to be representationally incapable of encoding invalid molecular structures. We employed six null model types: naive, shuffle, index-token distribution random, 1-bitflip random-mutation, 10-bitflip random-mutation, full random-mutation, and one unsupervised deep learning type: a VAE. Leveraging a training data set of 400 K SELFIES strings, the VAE outperformed the null models and demonstrated radial organization in its latent space, defining a clear applicability domain in which it effectively functions as an efficient representational anomaly generator. Accordingly, we propose deep variational anomaly generation as an effective means to test molecular representational robustness, including anomalies with previously unknown criteria.

ASSOCIATED CONTENT

Data Availability Statement

All data and code can be found at https://github.com/keiserlab/vae-anomaly-paper.

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jcim.4c01876.

Token vocabulary from training data set ; radial sampling of high-dimensional latent vector oversphere of given radius pseudocode; training data set exploratory data analysis; training and validation data sets exploratory data analysis; naive random null model pseudocode; index-token distribution random pseudocode; N-bitflip random-mutation pseudocode; full random-mutation pseudocode; greedily generating a SELFIES string by radially sampling a latent vector pseudocode; converted SMILES validity check through different approaches; model losses curves; categorical accuracy pseudocode; categorical accuracy distribution; SELFIES encoded−decoded pairs; SELFIES decoding output when facing inconvertible string; property distribution of generated SELFIES as a function of generative radius; property distribution of generated SELFIES as a function of generative radius for

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

nonmixtures; percentage of valid nonmixtures, valid mixtures, and invalid SELFIES; property distribution of 1 M set generated SELFIES as a function of generative radius for nonmixtures; length of SELFIES sequences distribution; mean categorical accuracy; valid decoded SELFIES containing TTC and "."; and time and memory consumption by the four models (PDF)

AUTHOR INFORMATION

Corresponding Authors

Rafael V. C. Guido − *São Carlos Institute of Physics, University of São Paulo, São Paulo 13563-120, Brazil*; Email: rvcguido@ifsc.usp.br

Michael J. Keiser − *Department of Pharmaceutical Chemistry, Department of Bioengineering & Therapeutic Sciences, Institute for Neurodegenerative Diseases, Kavli Institute for Fundamental Neuroscience, Bakar Computational Health Sciences Institute, University of California, San Francisco, San Francisco, California 94158, United States*; orcid.org/0000-0002-1240-2192; Email: keiser@keiserlab.org

Authors

Victor H. R. Nogueira − *São Carlos Institute of Physics, University of São Paulo, São Paulo 13563-120, Brazil; Genebank Department, Leibniz Institute of Plant Genetics and Crop Plant Research (IPK), Seeland D-06466, Germany*

Rishabh Sharma − *Department of Pharmaceutical Chemistry, Department of Bioengineering & Therapeutic Sciences, Institute for Neurodegenerative Diseases, Kavli Institute for Fundamental Neuroscience, Bakar Computational Health Sciences Institute, University of California, San Francisco, San Francisco, California 94158, United States*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jcim.4c01876

Author Contributions

[‖]V.H.R.N. and R.S. contributed equally to this work. Conceptualization: V.H.R.N. and R.S. Methodology: R.S. Writing—original draft: R.S. and V.H.R.N. Writing—review and editing: all authors. Formal analysis: R.S. and V.H.R.N. Resource: M.J.K. Supervision: M.J.K. and R.V.C.G. Project administration: all authors. Funding acquisition: M.J.K. and R.V.C.G.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES)—Finance Code 001, a CAPES Ph.D. Fellowship and a CAPES Sandwich Doctorate Fellowship (grants 88887.357974/2019-00 and 88887.695330/2022-00 to V.H.R.N.), by FAPESP (grant 2013/07600-3), and by grant DAF2018-191905 (10.37921/550142lkcjzw) from the Chan Zuckerberg Initiative DAF, an advised fund of the Silicon Valley Community Foundation (10.13039/100014989). The authors thank Laura Shub, Alexandre Fassio, and Mahdi Ghorbani for their feedback and suggestions on the manuscript.

ABBREVIATIONS

ML	machine learning
DL	deep learning
VAE	variational autoencoder
NR	null random
SR	shuffle random
ITDR	index token distribution random
1-RM	1-bitflip random-mutation
10-RM	10-bitflip random-mutation
F-RM	full random-mutation
TTC	troublesome token content
SELFIES	SELF-referencIng Embedded Strings
ELBO	evidence lower bound
SMILES	simplified molecular-input line-entry system
KL	Kullback−Leibler
GRU	gated recurrent unit
RNN	recurrent neural network
PC	principal component
PCA	principal component analysis

ADDITIONAL NOTES

[a]https://chemwriter.com/smiles/.
[b]https://depth-first.com/images/posts/20200824/demo/.

REFERENCES

(1) Samariya, D.; Ma, J.; Aryal, S.; Zhao, X. Detection and explanation of anomalies in healthcare data. *Health Inf. Sci. Syst.* **2023**, *11*, 20.

(2) Shan, L.; Li, Y.; Jiang, H.; Zhou, P.; Niu, J.; Liu, R.; Wei, Y.; Peng, J.; Yu, H.; Sha, X.; Chang, S. Abnormal ECG detection based on an adversarial autoencoder. *Front. Physiol.* **2022**, *13*, 961724.

(3) Derakhshan, A.; Harris, I. G.; Behzadi, M. Detecting telephone-based social engineering attacks using scam signatures. In *Proceedings of the 2021 ACM Workshop on Security and Privacy Analytics*: New York, NY, USA, 2021, pp 67−73..

(4) Ketepalli, G.; Tata, S.; Vaheed, S.; Srikanth, Y. M. Anomaly detection in credit card transaction using deep learning techniques. In *2022 7th International Conference on Communication and Electronics Systems (ICCES)*, 2022, pp 1207−1214..

(5) Bekrar, S.; Bekrar, C.; Groz, R.; Mounier, L. Finding software vulnerabilities by smart fuzzing. In *2011 Fourth IEEE International Conference on Software Testing, Verification and Validation*, 2011, pp 427−430.

(6) Ding, Z. Y.; Le Goues, C. An empirical study of oss-fuzz bugs. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*, 2021, pp 131−142..

(7) Laptev, N. AnoGen: deep anomaly generator. In *Meta Research*, 2018.

(8) Du, H.; Wang, S.; Huo, H. Xfinder: detecting unknown anomalies in distributed machine learning scenario. *Front. Comput. Sci.* **2021**, *3*, 710384.

(9) Schlegl, T.; Schlegl, S.; West, N.; Deuse, J. Scalable anomaly detection in manufacturing systems using an interpretable deep learning approach. *Proc. CIRP* **2021**, *104*, 1547−1552.

(10) David, L.; Thakkar, A.; Mercado, R.; Engkvist, O. Molecular representations in AI-driven drug discovery: a review and practical guide. *J. Cheminf.* **2020**, *12*, 56.

(11) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31−36.

(12) Krenn, M.; Häse, F.; Nigam, A.; Friederich, P.; Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): a 100% robust molecular string representation. *Mach. Learn.: Sci. Technol.* **2020**, *1*, 045024.

(13) O'Boyle, N.; Dalke, A. DeepSMILES: an adaptation of SMILES for use in machine-learning of chemical structures. **2018**, ChemRxiv:1903.10145. ChemRxiv preprint.

(14) Bank, D.; Koenigstein, N.; Giryes, R. Autoencoders. **2020**, arXiv:2003.05991. arXiv preprint.

(15) Kingma, D. P.; Welling, M. Auto-encoding variational Bayes. **2013**, arXiv:1312.6114. arXiv preprint.

(16) Doersch, C. Tutorial on variational autoencoders. **2016**, arXiv:1606.05908. arXiv preprint.
(17) Kopf, A.; Fortuin, V.; Somnath, V. R.; Claassen, M. Mixture-of-experts variational autoencoder for clustering and generating from similarity-based representations on single cell data. *PLoS Comput. Biol.* **2021**, *17*, No. e1009086.
(18) Graving, J.; Couzin, I. VAE-SNE: a deep generative model for simultaneous dimensionality reduction and clustering. *bioRxiv* **2020**.
(19) Kamoi, R.; Kobayashi, K. Why is the mahalanobis distance effective for anomaly detection?. **2020**, arXiv:2003.00402. arXiv preprint.
(20) Alghushairy, O.; Alsini, R.; Soule, T.; Ma, X. A review of local outlier factor algorithms for outlier detection in big data streams. *Big Data Cogn. Comput.* **2021**, *5*, 1.
(21) Rani, S.; Tripathi, K.; Arora, Y.; Kumar, A. Analysis of anomaly detection of Malware using KNN. In *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, 2022, pp 774−779..
(22) Zhang, A.; Song, S.; Wang, J.; Yu, P. S. Time series data cleaning: from anomaly detection to anomaly repairing. *Proc. VLDB Endow.* **2017**, *10*, 1046−1057.
(23) Iqbal, T.; Qureshi, S. Reconstruction probability-based anomaly detection using variational auto-encoders. *Int. J. Comput. Appl.* **2022**, *45*, 231−237.
(24) Ramakrishna, S.; Rahiminasab, Z.; Karsai, G.; Easwaran, A.; Dubey, A. Efficient out-of-distribution detection using latent space of β-vae for cyber-physical systems. *ACM Trans. Cyber-Phys. Syst.* **2022**, *6*, 1−34.
(25) Cozzatti, M.; Simonetta, F.; Ntalampiras, S. Variational autoencoders for anomaly detection in respiratory sounds. In *International Conference on Artificial Neural Networks*, 2022, pp 333−345..
(26) Prifti, E.; Buban, J. P.; Thind, A. S.; Klie, R. F. Variational convolutional autoencoders for anomaly detection in scanning transmission electron microscopy. *Small* **2023**, *19*, 2205977.
(27) Michael-Pitschaze, T.; Cohen, N.; Ofer, D.; Hoshen, Y.; Linial, M. Detecting anomalous proteins using deep representations. *bioRxiv* **2023**.
(28) Czibula, G.; Codre, C.; Teletin, M. AnomalP: an approach for detecting anomalous protein conformations using deep autoencoders. *Expert Syst. Appl.* **2021**, *166*, 114070.
(29) Ferré, Q.; Chèneby, J.; Puthier, D.; Capponi, C.; Ballester, B. Anomaly detection in genomic catalogues using unsupervised multi-view autoencoders. *BMC Bioinf.* **2021**, *22*, 460.
(30) Tiwari, A.; Bansode, V.; Rathore, A. S. Application of advanced machine learning algorithms for anomaly detection and quantitative prediction in protein A chromatography. *J. Chromatogr. A* **2022**, *1682*, 463486.
(31) Uzolas, L.; Rico, J.; Coupé, P.; SanMiguel, J. C.; Cserey, G. Deep anomaly generation: an image translation approach of synthesizing abnormal banded chromosome images. *IEEE Access* **2022**, *10*, 59090−59098.
(32) Godefroid, P.; Peleg, H.; Singh, R. Learn&fuzz: machine learning for input fuzzing. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2017, pp 50−59..
(33) Saavedra, G. J.; Rodhouse, K. N.; Dunlavy, D. M.; Kegelmeyer, P. W. A review of machine learning applications in fuzzing. **2019**, arXiv:1906.11133. arXiv preprint.
(34) Rajpal, M.; Blum, W.; Singh, R. Not all bytes are equal: neural byte sieve for fuzzing. **2017**, arXiv:1711.04596. arXiv preprint.
(35) She, D.; Pei, K.; Epstein, D.; Yang, J.; Ray, B.; Jana, S. Neuzz: efficient fuzzing with neural program smoothing. In *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp 803−817..
(36) Kaelbling, L. P.; Littman, M. L.; Moore, A. W. Reinforcement learning: a survey. *J. Artif. Intell. Res.* **1996**, *4*, 237−285.
(37) Arulkumaran, K.; Deisenroth, M. P.; Brundage, M.; Bharath, A. A. A brief survey of deep reinforcement learning. **2017**, arXiv:1708.05866. arXiv preprint.
(38) Becker, S.; Abdelnur, H.; State, R.; Engel, T. An autonomic testing framework for IPv6 configuration protocols. In *Mechanisms for Autonomous Management of Networks and Services: 4th International Conference on Autonomous Infrastructure, Management and Security; AIMS 2010*: Zurich, Switzerland, June 23−25, 2010. Proceedings 4, 2010, pp 65−76..
(39) Deng, Y.; Xia, C. S.; Yang, C.; Zhang, S. D.; Yang, S.; Zhang, L. Large language models are edge-case fuzzers: testing deep learning libraries via fuzzgpt. **2023**, arXiv:2304.02014. arXiv preprint.
(40) Sevgen, E.; Moller, J.; Lange, A.; Parker, J.; Quigley, S.; Mayer, J.; Srivastava, P.; Gayatri, S.; Hosfield, D.; Korshunova, M.; Livne, M.; Gill, M.; Ranganathan, R.; Costa, A. B.; Ferguson, A. L. ProT-VAE: protein transformer variational autoencoder for functional protein design. *bioRxiv* **2023**.
(41) Wu, Y.; Xu, L. Image generation of tomato leaf disease identification based on adversarial-VAE. *Agriculture* **2021**, *11*, 981.
(42) Tempke, R.; Musho, T. Autonomous design of new chemical reactions using a variational autoencoder. *Commun. Chem.* **2022**, *5*, 40.
(43) Lee, M.; Min, K. MGCVAE: multi-objective inverse design via molecular graph conditional variational autoencoder. *J. Chem. Inf. Model.* **2022**, *62*, 2943−2950.
(44) Jin, W.; Barzilay, R.; Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. **2019**, arXiv:1802.04364. arXiv preprint.
(45) Hadipour, H.; Liu, C.; Davis, R.; Cardona, S. T.; Hu, P. Deep clustering of small molecules at large-scale via variational autoencoder embedding and K-means. *BMC Bioinf.* **2022**, *23*, 132.
(46) Lo, A.; Pollice, R.; Nigam, A.; White, A. D.; Krenn, M.; Aspuru-Guzik, A. Recent advances in the Self-Referencing Embedding Strings (SELFIES) library. **2023**, arXiv:2302.03620. arXiv preprint.
(47) Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M. et al. Tensorflow: large-scale machine learning on heterogeneous distributed systems. **2016**, arXiv:1603.04467. arXiv preprint.
(48) 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. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent. Sci.* **2018**, *4*, 268−276.
(49) Davies, M.; Nowotka, M.; Papadatos, G.; Dedman, N.; Gaulton, A.; Atkinson, F.; Bellis, L.; Overington, J. P. ChEMBL web services: streamlining access to drug discovery data and utilities. *Nucleic Acids Res.* **2015**, *43*, W612−W620.
(50) Mendez, D.; Gaulton, A.; Bento, A. P.; Chambers, J.; De Veij, M.; Félix, E.; Magariños, M.; Mosquera, J. F.; Mutowo, P.; Nowotka, M.; et al. ChEMBL: towards direct deposition of bioassay data. *Nucleic Acids Res.* **2019**, *47*, D930−D940.
(51) Jimmy, B.; Diederik, P. Adam: a method for stochastic optimization. **2014**, arXiv: 1412.6980. arXiv preprint.
(52) Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.
(53) Fu, H.; Li, C.; Liu, X.; Gao, J.; Celikyilmaz, A.; Carin, L. Cyclical annealing schedule: a simple approach to mitigating kl vanishing. **2019**, arXiv:1903.10145. arXiv preprint.
(54) Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*; Chia Laguna Resort: Sardinia, Italy, 2010, pp 249−256.
(55) Kusner, M. J.; Paige, B.; Hernández-Lobato, J. M. Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp 1945−1954.
(56) Sarker, I. H. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Comput. Sci.* **2021**, *2*, 420.
(57) Gangwal, A.; Ansari, A.; Ahmad, I.; Azad, A. K.; Wan Sulaiman, W. M. A. Current strategies to address data scarcity in artificial

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927

intelligence-based drug discovery: a comprehensive review. *Comput. Biol. Med.* **2024**, *179*, 108734.
(58) Dobson, J. E. On reading and interpreting black box deep neural networks. *Int. J. Digit. Humanit.* **2023**, *5*, 431−449.
(59) Ding, Z.; Xu, Y.; Xu, W.; Parmar, G.; Yang, Y.; Welling, M.; Tu, Z. Guided variational autoencoder for disentanglement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp 7920−7929.
(60) Ramakrishnan, R.; Dral, P. O.; Rupp, M.; von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Sci. Data* **2014**, *1*, 140022.
(61) Ruddigkeit, L.; van Deursen, R.; Blum, L. C.; Reymond, J.-L. Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *J. Chem. Inf. Model.* **2012**, *52*, 2864−2875.
(62) Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: a next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019, pp 2623−2631..
(63) Richards, R. J.; Groener, A. M. Conditional β-VAE for de novo molecular generation. **2022**, arXiv:2205.01592. arXiv preprint.
(64) Ishfaq, H.; Hoogi, A.; Rubin, D. TVAE: Triplet-based variational autoencoder using metric learning. **2018**, arXiv:1802.04403. arXiv preprint.
(65) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139−144.
(66) Xu, M.; Powers, A. S.; Dror, R. O.; Ermon, S.; Leskovec, J. Geometric latent diffusion models for 3d molecule generation. In *International Conference on Machine Learning*, 2023, pp 38592−38610.
(67) Ziegler, Z.; Rush, A. Latent normalizing flows for discrete sequences. In *International Conference on Machine Learning*, 2019, pp 7673−7682.

https://doi.org/10.1021/acs.jcim.4c01876
J. Chem. Inf. Model. 2025, 65, 1911−1927