PNAS Nexus, 2024, **3**, pgae168
**https://doi.org/10.1093/pnasnexus/pgae168**
Advance access publication 25 April 2024
**Research Report**

# Kernel-elastic autoencoder for molecular design

Haote Li[a], Yu Shee[a], Brandon Allen[a], Federica Maschietto[a], Anton Morgunov[a] and Victor Batista[a,*]

[a]Department of Chemistry, Yale University, New Haven, CT 06520, USA
*To whom correspondence should be addressed: Email: victor.batista@yale.edu
**Edited By:** Cristina Amon

## Abstract

We introduce the kernel-elastic autoencoder (KAE), a self-supervised generative model based on the transformer architecture with enhanced performance for molecular design. KAE employs two innovative loss functions: modified maximum mean discrepancy (m-MMD) and weighted reconstruction ($\mathcal{L}_{\text{WCEL}}$). The m-MMD loss has significantly improved the generative performance of KAE when compared to using the traditional Kullback–Leibler loss of VAE, or standard maximum mean discrepancy. Including the weighted reconstruction loss $\mathcal{L}_{\text{WCEL}}$, KAE achieves valid generation and accurate reconstruction at the same time, allowing for generative behavior that is intermediate between VAE and autoencoder not available in existing generative approaches. Further advancements in KAE include its integration with conditional generation, setting a new state-of-the-art benchmark in constrained optimizations. Moreover, KAE has demonstrated its capability to generate molecules with favorable binding affinities in docking applications, as evidenced by AutoDock Vina and Glide scores, outperforming all existing candidates from the training dataset. Beyond molecular design, KAE holds promise to solve problems by generation across a broad spectrum of applications.

**Keywords:** generative modeling, molecular optimization, molecular docking

## Significance Statement

Kernel-elastic autoencoder (KAE) is a powerful computational tool for molecular design that extends the capabilities of variational autoencoders (VAEs) as applied to drug discovery. With the widespread use of VAEs and the growing interest in generative models, KAE bears the potential to advance the field of generative modeling. Beyond applications of molecular designs, KAE introduces a generative approach that allows the enhancement of all VAE-based models, offering a practical and versatile solution for the broader machine learning community.

## Introduction

The advent of generative models has precipitated a revolutionary shift in the development of methods for drug discovery, revealing new opportunities to swiftly identify ideal candidates for specific applications (1–7). The variational autoencoder (VAE) model has emerged among these models as an approach with extraordinary capabilities that can be adapted for molecule generation via character, grammar, and graph-based representations (8–11).

Autoencoders (AEs) encode the input data by compression into a low-dimensional space (12). Though providing a high lower bound for accurate reconstruction, such space is not well structured and in some regions, the decoder does not generate output that resembles the training data, thereby limiting its generative capabilities. Sacrificing reconstruction performance, VAEs mitigate this disadvantage by enforcing encoded latent vectors to known prior distributions. Upon decoding samples from those distributions, VAEs generate outputs mimicking the training data. An outstanding challenge of great interest to drug discovery is to harness the power of VAEs to generate molecular candidates with optimal properties during the screening phase of molecular discovery while preserving AE's high reconstruction rate for precise lead candidate optimizations.

Generative models are typically evaluated for molecule generation using novelty ($N$), uniqueness ($U$), validity ($V$), and reconstruction ($R$) metrics. NUV-R metric, which is the product of them, captures the tradeoff between these four factors, the so-called NUV to $R$ tradeoff, as a model with high reconstruction ability usually does not achieve high metrics for novelty, uniqueness, and validity.

Optimizing the design of molecules near a reference molecule requires robust reconstruction, as proximity in latent space should correlate with proximity in the value of the desired property. Accurate reconstruction also allows for interpolation between molecular motifs with intermediate properties between promising lead compounds (13–16).

*Kernel-elastic autoencoder* (KAE) stands out as a new self-supervised generative model with a modified maximum mean discrepancy and weighted reconstruction loss functions. Leveraged by the transformer architecture(17–20), KAE (Fig. 1) effectively overcomes the NUV-R tradeoff by combining the merits of both autoencoder (AE) and variational autoencoder (VAE) models.

**Competing Interest:** The authors declare no competing interest.
**Received:** December 14, 2023. **Accepted:** April 4, 2024

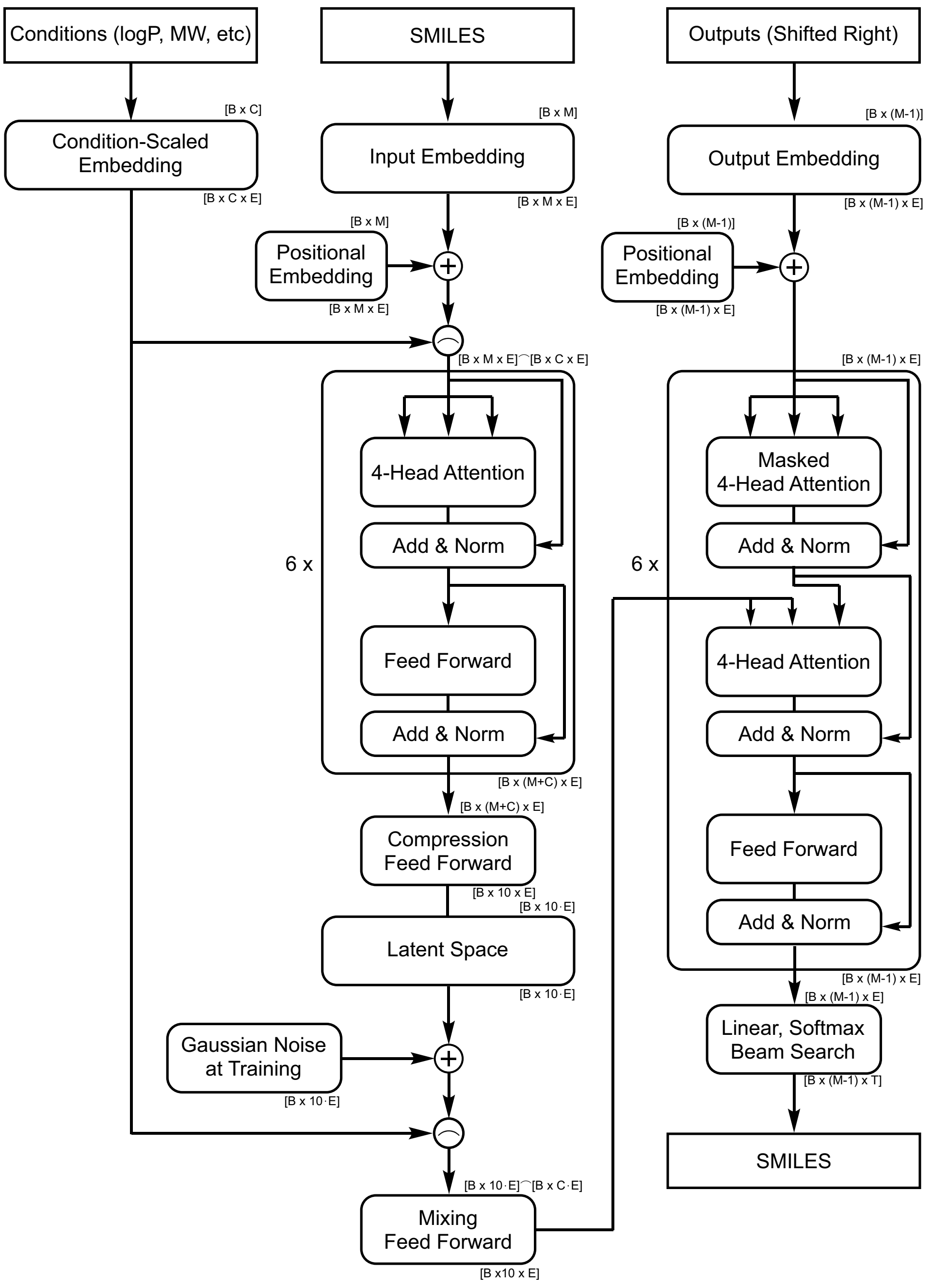


**Fig. 1.** CKAE architecture. KAE consists of six encoder layers, six decoder layers, and a latent space for conditional generations. During training, the condition is concatenated after positional embedding and provided as input to the 4-head attention encoder. The condition is also concatenated with the latent vector before a mixing layer. During training, Gaussian noise is added to the latent vectors. The decoder output is then passed through a linear layer and softmax function, producing the probabilities of output tokens for each character in the dictionary of size T.

KAE's loss function is a modified version of the maximum mean discrepancy (MMD), inspired by Refs. (21–23), that shapes the latent space and enables better performance than using Kullback–Leibler (KL) divergence loss used in VAEs. When coupled to the weighted cross-entropy loss ($\mathcal{L}_{\mathrm{WCEL}}$), KAE, without any checking for molecular grammar or chemical rules, outperforms both string and graphical-based models in generation tasks while exhibiting nearly flawless reconstruction, as demonstrated on the ZINC250k testing sets. The freedom to adjust KAE's behavior through the $\mathcal{L}_{\mathrm{WCEL}}$ gives the "Elastic" term in its naming.

When implemented to solve optimization problems, KAE outperforms the state-of-the-art by a substantial 28% (24). Additionally, KAE tackles the problem of molecular docking by finding suitable binding ligands with conditional generation, as demonstrated using the dataset from GFlowNet (5). Superior candidates from the baseline and the training data are independently verified by both Autodock Vina (25) and Schrodinger Glide (26, 27), demonstrating its practicality.

## Result

### KAE performance

The overall performance of the KAE (Fig. 1) compared to state-of-the-art generative models is shown in Table 1. Described in

**Table 1.** Comparison of performance of molecular generative models trained with the ZINC250K dataset.

| Method | N | U | V w/o | V | NUV | R | NUV-R |
|---|---|---|---|---|---|---|---|
| CVAE (9)[a] | 0.980 | 0.021 | 0.007 | N/A | 0.0001 | 0.446 | $5e-6$ |
| GVAE (10)[a] | 1.000 | 1.000 | 0.072 | N/A | 0.072 | 0.537 | 0.039 |
| JT-VAE (11)[a] | 1.000 | 1.000 | 0.935 | 1.000 | 0.935 | 0.767 | 0.717 |
| MoFlow (28) | 1.000 | 0.999 | 0.818 | 1.000 | 0.817 | 1.000[b] | 0.817 |
| Rebalanced (29) | 1.000 | 1.000 | 0.907 | 0.938 | 0.907 | 0.927 | 0.841 |
| GraphDF (30) | 1.000 | 0.992 | 0.890 | 1.000 | 0.883 | 1.000[b] | 0.883 |
| ALL SMILES (31)[a] | 1.000 | 1.000 | N/A | 0.985 | N/A | 0.874 | N/A |
| $\beta$-VAE (24) | 0.998 | 0.983 | 0.983 | 0.988 | 0.964 | N/A | N/A |
| KAE ($\lambda = 1$, $\delta = -1$) | 0.998 | 0.994 | 0.863 | N/A | 0.856 | 0.992 | 0.849 |
| KAE ($\lambda = 3.5$, $\delta = 1$) | **0.996** | **0.973** | **0.997** | **1.000** | **0.966** | **0.997** | **0.963** |

The bolded numbers represent the KAE and the best results.
[a]Results obtained from sampling 1,000 vectors from latent space. [b]Reconstruction rates were obtained on training datasets. Assessment of the capabilities of the models to generate novel (N), unique (U), valid (V), and properly reconstructed (R) molecules. Validity (V w/o) indicates that the generated strings have not been postprocessed using chemical knowledge to enforce corrections. NUV results were obtained from averaging over 5 iterations of sampling 10,000 random vectors from latent space, while the reconstruction rate was calculated using all molecules from the testing dataset. The two KAE models in the table were trained using loss functions with $\lambda = 1$ and 3.5 and $\delta = -1$ and 1. The choice of $\delta = -1$ is a special case of $\mathcal{L}_{\mathrm{WCEL}}$ and is equivalent to not using any AE objectives. Our validity check selects alternative candidates from the beam search.

the Methods section, KAE combines a modified-MMD (m-MMD) loss and the weighted cross-entropy loss ($\mathcal{L}_{\mathrm{WCEL}}$), with hyperparameters $\lambda$ and $\delta$, and exhibits the generative capabilities of VAEs as well as the exact reconstruction objectives of AEs.

KAE was evaluated according to the fraction of generated molecules that are novel (N), unique (U), and valid (V). A molecule is considered novel if it is not included in the training dataset. Uniqueness is defined as the absence of duplicates in the set of generated molecules. A molecule is counted as valid if its SMILES representation is syntactically correct and passes the RDKit chemical semantics checks (32). Additionally, reconstruction (R) is successful if and only if the decoder regenerates the input SMILES sequence matching every single token.

Maximum validity and reconstruction was achieved by using the $\mathcal{L}_{\mathrm{WCEL}}(\lambda, \delta)$ defined by Eq. 4 where the hyperparameter $\delta$ controls the AE-like objective (see supplementary material for a discussion of the effect of changing $\lambda$ and $\delta$). The best results for the NUV-R metric were obtained by using a combination of $\lambda = 3.5$ and $\delta = 1$.

## Learning behavior

We have analyzed the KAE behavior by comparing under the same architecture but with various loss functions (Fig. 2). The reconstruction was evaluated from 1,000 molecules from the validation set at every epoch. Figure 2 shows the improvement in validity, uniqueness, novelty, and reconstruction along the training process for models based on a loss that combines the $\mathcal{L}_{\mathrm{WCEL}}(\lambda,\delta)$, defined by Eq. 4, with m-MMD (m-MMD($\lambda$), Eq. 11), s-MMD, Eq. 10, or KL-divergence. All models were trained with the ZINC250K dataset for 200 epochs, with $\lambda = 1$ and $\delta = -1$. When $\lambda = -\delta$, the weighted cross-entropy loss ($\mathcal{L}_{\mathrm{WCEL}}$, Eq. 4) reduces to the standard cross-entropy loss ($\mathcal{L}_{\mathrm{CEL}}$). Additionally, we examine the effect of noise while training with the m-MMD loss. The results (Fig. 2) indicate that the KAE model using m-MMD loss with Gaussian noise added to the latent space exhibits the best performance. The models exhibit significant differences in their ability to generate valid SMILES strings and reconstruct input molecules. The m-MMD model trained with noise in latent space generated the highest percentage of valid SMILES strings, making it preferable to other models. For example, the model trained with KL-divergence exhibited much lower validity and a significantly slower learning rate. The assessment of novelty and uniqueness also shows that s-MMD and m-MMD models trained with Gaussian noise added in latent space (noisy models) performed better than the corresponding models without noise. Another reason to add noise is to prevent the model from overfitting the latent vectors so that the decoder has to see the multitude of possible outcomes related to the region of the decoding latent vector. Further, the decision to add a Gaussian noise on top of confining the latent vector to the same Gaussian through m-MMD is to maximize the overlap of the distribution of all latent vectors with respect to the distribution of any individual latent vector. This approach is different from VAE as the VAE has the option to output small variances for some latent vectors which could reduce the probability of sampling corresponding instances from its prior distribution.

## Conditional-KAE

In this section, the performance of the conditional-KAE (CKAE) (Fig. 1) on the constraint optimization task is investigated.

CKAE generates new candidates conditioned on properties such as PLogP or docking scores. Here, we first demonstrate the capabilities of CKAE as applied to the PLogP values defined, as follows (9, 11):

$$\mathrm{PLogP}(m) = \mathrm{LogP}(m) - \mathrm{SA}(m) - \mathrm{ring}(m), \qquad (1)$$

where LogP is the octanol–water partition coefficient of molecule $m$ calculated using Crippen's approach from the atom contributions (33). SA is the synthetic accessibility score (34), while ring($m$) corresponds to the number of rings with more than six members for the molecule $m$.

To demonstrate that CKAE generates molecules that are strongly correlated to the conditioned value, we analyzed the correlation between the properties of CKAE-generated molecules and the specified input condition. Figure 3 shows the mean PLogP value obtained from 1,000 CKAE-generated molecules, strongly correlated to the PLogP value used as a condition (correlation coefficient 0.9997). The distribution of PLogP values of the training set, rendered as a histogram in Fig. 3, shows the range of PLogP values used for CKAE training. We have also trained a separate model using the dataset from Lim et al. (35) who developed a conditional-VAE (CVAE) with recurrent neural network (RNN) architectures to sample molecules given five distinct pharmaceutically relevant properties. The comparison between CKAE and CVAE from Lim et al. further shows that CKAE generates candidates correlating to the asked conditions and outperforms the given baseline by a wide margin (Table 2).

Instead of using regressors to navigate in the latent space (11, 24, 29, 36), a procedure called similarity exhaustion search

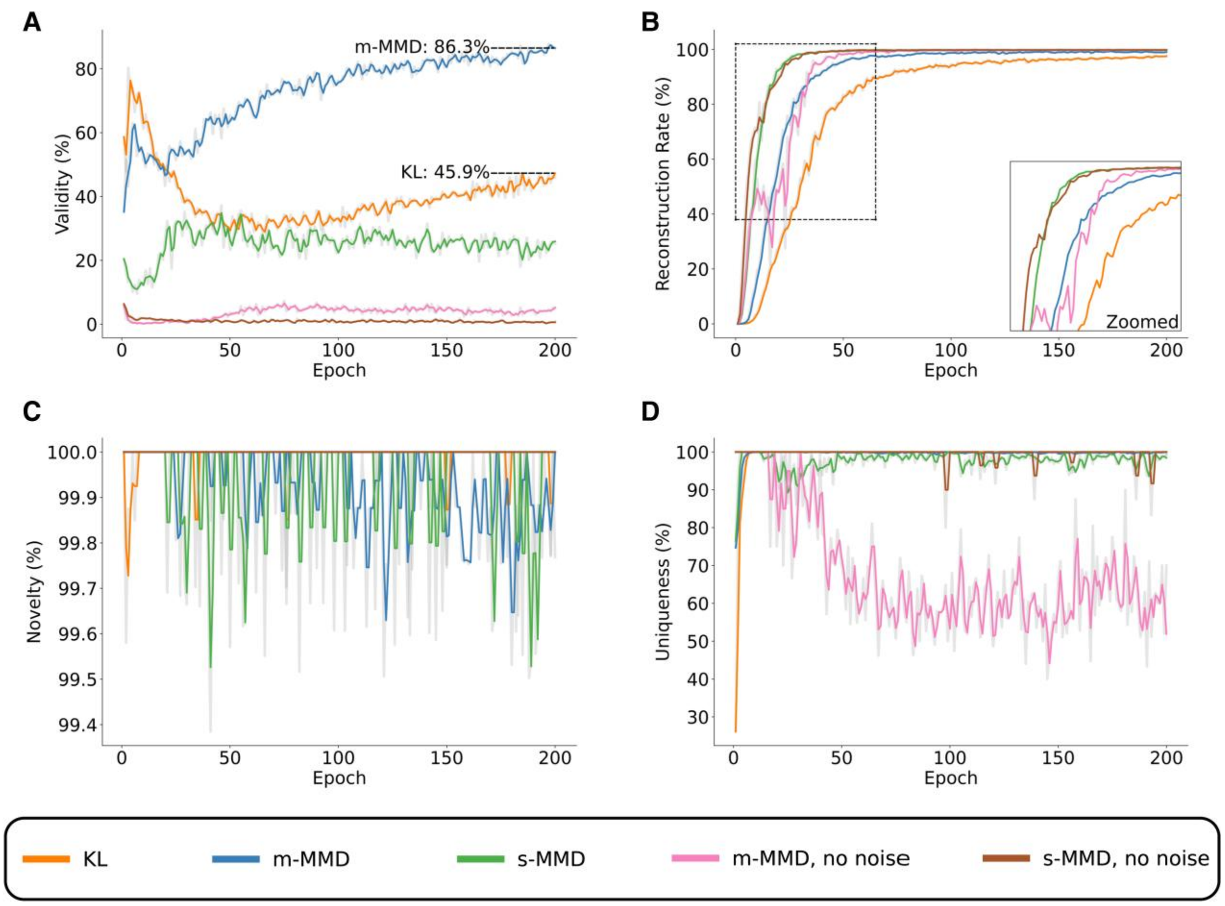


**Fig. 2.** Comparison of learning rates for models trained with m-MMD loss, s-MMD loss, and KL-divergence loss. A) Validity evaluated at each epoch. B) Fraction of molecules properly reconstructed as a function of epochs. C) Novelty evaluated at each epoch. D) The uniqueness at each epoch. The model labeled as KL includes an extra layer that estimates the SD of each latent vector. The models labeled with m-MMD are trained with the loss $\mathcal{L}_{\mathrm{CEL}}$ + m-MMD($\lambda = 1$), s-MMD with $\mathcal{L}_{\mathrm{CEL}}$ + s-MMD($\lambda = 1$), and KL with $\mathcal{L}_{\mathrm{VAE}} = \mathcal{L}_{\mathrm{CEL}} + \mathrm{KL}(\lambda = 1)$. "no noise" in the legends means no noise is added to the latent vectors during training.

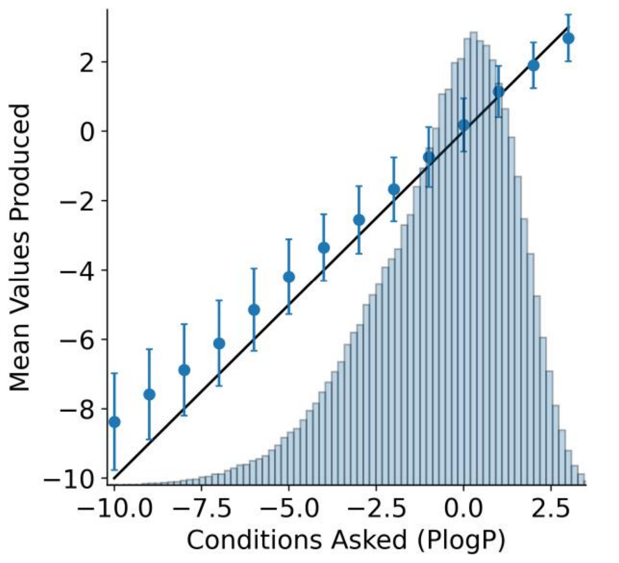


**Fig. 3.** CKAE correlation performance. The blue dots represent the mean PLogP values of 1,000 molecules generated by CKAE, as a function of the condition PLogP value. The error bars on each dot indicate the associated SD as estimations of errors. The black line shows the ground truth values. The histogram shows the underlying distribution of the training dataset over the entire range of PLogP values.

**Table 2.** Performance of CKAE compared to CVAE, as applied to conditional molecular generation.

| Method | Target | Attempts | Number of valid molecules | Success rate (%) |
|---|---|---|---|---|
| CVAE Lim et al. | Aspirin | 28,840 | 32,567 | 0.34 |
| CVAE Lim et al. | Tamiflu | 15,960 | 34,696 | 0.62 |
| CKAE | Aspirin | **4,743** | 4,743 | **2.11** |
| CKAE | Tamiflu | **3,715** | 3,715 | **2.63** |
| CKAE w. Beam | Aspirin | **671** | 4221 | **14.90** |
| CKAE w. Beam | Tamiflu | **436** | 3927 | **22.94** |

We impose counting criteria for the CKAE statistics so that the proposed molecule is counted as an attempt only if it is valid, novel, and unique. Therefore, the number of valid molecules is equal to the number of attempts. To further compare to the CVAE method by Lim et al. where there is more than one valid molecule per attempt, we have applied beam search with a beam size of 10 (labeled CKAE w. Beam). When beam search is used, the number of valid molecules reports the number of valid, novel, and unique candidates derived from all attempts. The success rate is defined as 100 times the rate of finding a candidate within a 10% error range of each property per attempt. The result shows CKAE is better than CVAE with RNN architectures. Further, if beam search is applied, CKAE significantly outperforms the given baseline. The bolded numbers represent the KAE and the best results.

(SES) was developed for constraint optimizations. SES aims to find molecules that are both similar to the target molecule and have higher desired properties (e.g. PLogP) by using the same or slightly perturbed latent vector representations with gradually increasing

**Table 3.** Comparison of performance of various conditional generative models.

| Method[a] | PLogP-improvement | Tanimoto similarity | Success rate (%) |
|---|---|---|---|
| JT-VAE (11) | 0.84 ± 1.45 | 0.51 ± 0.1 | 83.6 |
| MHG-VAE (37) | 1.00 ± 1.87 | 0.52 ± 0.11 | 43.5 |
| GCPN (38) | 2.49 ± 1.30 | 0.47 ± 0.08 | 100 |
| Mol-CycleGAN (1) | 2.89 ± 2.08 | 0.52 ± 0.10 | 58.75 |
| MolDQboot (39) | 3.37 ± 1.62 | N/A | 100 |
| ZINC250K (this work) | 4.64 ± 2.33 | 0.48 ± 0.16 | 97.88 |
| MoFlow (28) | 4.71 ± 4.55 | **0.61 ± 0.18** | 85.75 |
| Random sample (this work) | 4.78 ± 2.08 | 0.43 ± 0.03 | 81.75 |
| MNCE-RL (40) | 5.29 ± 1.58 | 0.45 ± 0.05 | 100 |
| $\beta$-VAE (24) | 5.67 ± 2.05 | 0.42 ± 0.05 | 98.25 |
| CKAE (this work) | **7.67 ± 1.61** | 0.42 ± 0.02 | **100** |

The bolded numbers represent the KAE and the best results.
[a]Tanimoto similarity constraint of 0.4. The table presents the average PLogP improvements computed for the set of 800 lowest ranking molecules from the ZINC250K dataset as well as the mean Tanimoto similarities of the best candidate molecules compared to their respective starting molecules (SDs reported after ±). The success rate indicates the percentage of molecules for which the algorithm successfully achieved modifications resulting in higher PLogP values within the specified similarity constraint. The ZINC250K result corresponds to the highest PLogP improvement obtained by searching within the ZINC250K dataset itself. Our approach outperforms the search against the training data and demonstrates the highest performance when combining our model with the SES method.

conditions. Formally, $f(z, c) \approx f(z + \Delta_z, c + \Delta_c)$ for small values of $\Delta_z$ and $\Delta_c$ where $f(z, c)$ is the decoding output function of latent vector $z$ subject to the condition $c$ (e.g. PLogP $= c$). When the generative model has high enough NUV-R values, it is able to pinpoint the exact latent vector location and perform an exhaustive search for all possible $\Delta z$. Therefore, SES combines beam search with iterative sampling under various conditions to identify chemically similar molecules that closely resemble the target compound in the latent space. The details of SES can be found in the supplementary material.

Table 3 shows (i) the results of optimizing the 800 lowest PLogP-valued molecules from the ZINC250K dataset to generate similar molecules (Tanimoto similarity $\leq 0.4$) with larger PLogP values (39); (ii) the mean difference in PLogP values; and (iii) the Tanimoto similarity between the best candidate molecules and their starting molecules for each method. The success rate measures the percentage of molecules that achieved modifications with higher PLogP values within their similarity constraints.

Additionally, CKAE performance was assessed as compared to direct search from the ZINC250K training set. For each of the 800 molecules, its similarity value with respect to all other 250 K entries was calculated, and the compound with the highest PLogP value that remained within the 0.4 Tanimoto similarity constraint was identified. This particular outcome is labeled "ZINC250K" in Table 3.

We further compared CKAE to direct search using randomly sampled latent vectors with different conditions (PLogP values from −10 to 10 scanned with a step size of 0.1). At each step, instead of using encoder-provided latent vectors. Eight hundred vectors were randomly sampled from the latent space and decoded using beam search with a beam size of 15. The outcomes of this search are marked as "Random Search" in Table 3.

## CKAE for ligand docking

### *Comparison to GFlowNet*

Table 4 shows the performance of the CKAE model as applied to the generation of small molecule inhibitors that bind to the active site of the enzyme soluble epoxide hydrolase (sEH), as compared to results obtained with GFlowNet for the same active site (5, 41).

**Table 4.** Performance of the CKAE model on molecular docking as compared to GFlowNet.

| Method | Top 10 reward | Top 100 reward | Top 1,000 reward | Top-1,000 similarity |
|---|---|---|---|---|
| GFlowNet | 8.36 | 8.21 | 7.98 | 0.44 |
| Training data | 9.62 | 8.78 | 7.86 | 0.58 |
| CKAE (this work) | **11.15** | **10.46** | **9.63** | 0.63 |

Top 10, 100, and 1,000 rewards are the averages of the docking scores of molecules generated at the corresponding thresholds. The Top-1,000 similarity is the mean of all pairwise similarities. Lower similarity between generated molecules indicates greater diversity, which is desirable. For docking, the higher rewards are better.
The bolded numbers represent the KAE and the best results.

CKAE was trained using the same dataset of 3,00,000 molecules which GFlowNet (42) was trained from, each entry with a binding energy calculated using AutoDock (25) (see Glide anlysis section). Binding energies were converted to a reward metric, using a custom scaling function. Results in Table 4 correspond to the mean reward for the top 10, 100, and 1,000 best-scoring molecules from a pool of $10^6$ NUV molecules generated by the CKAE model. Rewards were computed from the Autodock Vina binding scores. Average Tanimoto similarities were computed using a Morgan Fingerprint with a radius of 2.

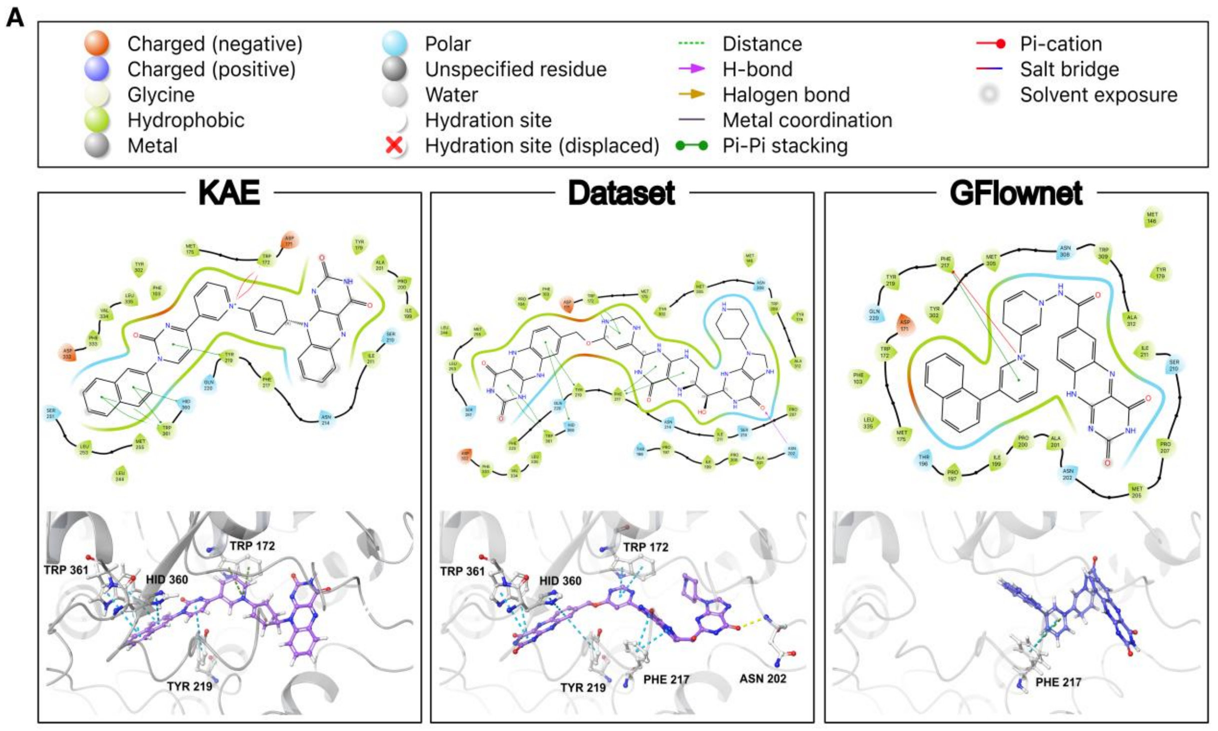
Table 4 shows that CKAE achieves similar performance to GFlowNet in molecular docking, and generates molecules with higher rewards at the top 10, 100, and 1,000 thresholds, without significantly sacrificing the similarity score. In fact, CKAE was able to generate molecules scoring as high as 11.45, which exceeds the maximum reward of 10.72 in the training database itself. This demonstrates the capabilities of CKAE for generative extrapolation, which allows for applications to generative dataset augmentation including molecules with scoring values beyond the range of the original dataset.

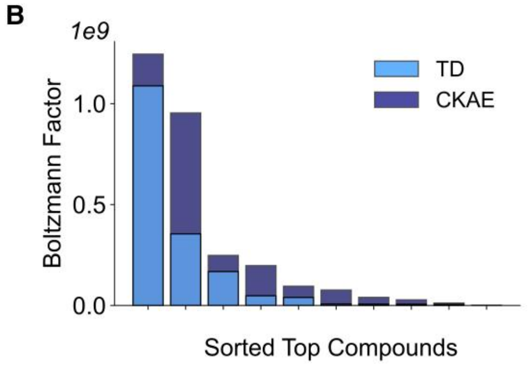
### *Glide analysis*

A comparison of the ligand–receptor interactions established by the top-scoring CKAE, TD, and GFlowNet candidates, respectively is shown in Fig. 4A. KAE's top candidate exhibits superior docking performance compared to top-scoring candidates in both the training dataset and GFlowNet. In terms of fitting within the pocket, the top CKAE candidate occupies a substantially larger volume within the receptor binding region when compared to the other two. The improved fit is also evidenced by the broader array of stabilizing interactions. These interactions include a series of $\pi$–$\pi$ stacking and $\pi$–cation interactions. In addition to occupying the pocket entirely, the CKAE-generated molecules are devoid of unfavorable clashes, further underscoring the effectiveness of the model in generating effective candidates in the context of molecular docking.

Figure 4A shows the analysis of the best-scoring molecules generated by CKAE and direct search from the training dataset (TD), as assessed by the Glide molecular docking program that is an integral part of the Schrödinger Suite of software (26, 27). Figure 4B thus provides an independent assessment of the quality of the best-scoring CKAE-generated molecules, showing that CKAE-generated molecules outperform the TD counterparts in terms of ranking as determined by the nature of the interactions established at the binding site.

The docking procedure employed an identical receptor grid size as used for Autodock Vina (25) calculations, and the candidates sourced from both the training dataset and CKAE, were docked onto the same receptor structure, using the highest scoring pose





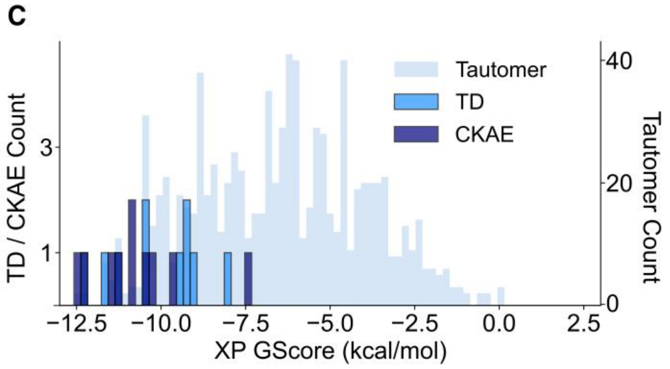


**Fig. 4.** Glide analysis of molecular inhibitors docked at the active site of sEH. A) Binding interactions of top-scoring molecules generated by CKAE (left), searched from the training dataset (middle) and generated by GFlowNet (right). B) Extra precision (XP) Glide score Boltzmann factors for the top 10 candidates obtained from the CKAE and training dataset (TD) show that the top-ranking CKAE-generated outperform the top molecules from the TD ensemble. C) Histogram of Glide XP docking scores, showing that top-scoring inhibitors generated by CKAE or TD outperform 869 tautomers generated from the top 10 candidates of the two datasets.

derived from Autodock Vina (25) calculations, as described in Section 4.5.1.

A dataset comprised of 869 tautomers was curated with high structural similarity, including the top 10 CKAE-derived molecules and the top 10 TD molecules, as well as a set of tautomers of the same molecules generated by changing protonation and enantiomeric states to analyze the quality of the top-performing hits relative molecular tautomers (molecules with different arrangements of atoms and bond). The results shown in Fig. 4C revealed that the top-ranking candidates from both CKAE and TD outperformed other contenders (tautomers) when compared against the dataset of tautomers. These results confirmed that the highest scoring molecular structures obtained from CKAE and TD remained superior, even when compared to a large number of structurally similar alternatives, confirming the reliability and quality of molecules generated by CKAE.

As examined by both Autodock Vina (25) and Glide (26, 27), it is clear that CKAE-generated molecules that bind better to the active site of sEH than those of the training dataset. The generated higher scoring molecules can then be used for dataset augmentation, for retraining purposes, allowing the model to generate even higher scoring molecules.

## Methods

### Model architecture

KAE treats molecule generation as a natural language processing task. Phrases in the "source language" (i.e. SMILES strings) are encoded and compressed into latent vectors and then decoded into the target output with corresponding labels. Major components for KAE include the encoder, compression layer, mixing layer, and decoder.

Source and target masks are created with specified padding tokens to ensure that the encoder and decoder do not attend to padding tokens during training. The SMILES tokens are separately passed through embedding layers of the encoder and decoder to become vectors of size 128. They are then added to the corresponding position embeddings of the same dimensions. Different from the original Transformer implementation that uses fixed sinusoidal functions in the representation, in this work, each positional token's embedding is learned and updated during training.

The input is encoded by the Transformer encoder and compressed into latent space. The compression layer is a single linear layer that applies to the sequence length dimensions. This layer takes in a dimension $M$, the maximum sequence length in the relevant dataset and outputs a dimension of 10. In the case of ZINC250k without using conditions, $M$ is 113 dimensional. The resulting latent tensor therefore has dimensions of $10 \times E$ where $E$ is the embedding size of 128. The latent vectors are then added with noise from a standard Gaussian distribution. In the CKAE variant, the conditions (i.e. molecule properties) are attached with additional embeddings. Condition-multiplied embeddings are concatenated with the input of the encoder and the latent representation along the sequence length dimension. This allows the model to generate molecules by either interpolating or extrapolating with a desired condition value. The mixing layer is a linear layer that takes in the compressed tensors with the size $(10 + \text{number of conditions}) \times E$ and maps them back to $10 \times E$ dimensions. These tensors are treated as the new encoder output which the decoder attends to without encoder masks. Each decoder layer attends to the encoder outputs through encoder–decoder multihead attention operations. The decoder outputs are contracted by a linear layer along the embedding dimension, producing a $T$-dimensional vector per token, where $T$ is the dictionary size. This $T$-dimensional vector is then softmaxed, resulting in a probability distribution (P) for each possible character (c).

## KAE loss

The KAE loss function is defined, as follows:

$$\mathcal{L}(\lambda, \delta) = \mathcal{L}_{\text{WCEL}}(\lambda, \delta) + \text{m-MMD}(\lambda), \tag{2}$$

where m-MMD($\lambda$) is a modified version of the regularizing MMD loss, discussed in Section 4.3, and $\mathcal{L}_{\text{WCEL}}$ is a weighted cross-entropy loss ($\mathcal{L}_{\text{WCEL}}$) obtained from outputs generated by decoding the latent vector with and without Gaussian noise added to the latent vector. Based on the original definition of the cross-entropy loss (CEL):

$$\mathcal{L}_{\text{CEL}} = -\sum_{s}\sum_{c} Y_{s,c} \log(P_{s,c}), \tag{3}$$

where $P_{s,c}$ is the predicted softmax probability of token $c$ at sequence position $s$ and $Y_{s,c}$ is the ground truth label equal to one if the token belongs to class $c$ at position $s$, or zero otherwise. Accordingly, we define $\mathcal{L}_{\text{WCEL}}$, as follows:

$$\begin{aligned}\mathcal{L}_{\text{WCEL}}(\lambda, \delta) = \frac{-1}{\lambda + \delta + 1}\Bigg[&\sum_{s}\sum_{c} Y_{s,c}\log(P_{s,c}) \\ &+(\lambda + \delta)\sum_{s}\sum_{c} Y_{s,c}\log\left(P^{*}_{s,c}\right)\Bigg],\end{aligned} \tag{4}$$

where $P_{s,c}$ and $P^{*}_{s,c}$ are the predicted softmax values obtained upon decoding the latent vector with and without added Gaussian noise, respectively.

The hyperparameters $\lambda$ and $\delta$ control the significance of the second term in the r.h.s. of Eq. 4 (AE behavior) as well as the relative weight between the m-MMD term and the weighted cross-entropy loss, according to Eq. 2. The function of $\lambda$ is analogous to the $\beta$ parameter in $\beta$-VAE (43). By adjusting $\lambda$ and $\delta$, the learning objective can be positioned between the VAE and AE objectives. At the extremes, the objective becomes VAE-like (or AE-like) upon weighting more the term with (or without) noise. For example, when $\lambda = 1$ and $\delta = -1$, $\mathcal{L}$ is like the VAE loss except that we use m-MMD instead of the KL-divergence. For AE-like behavior, we choose $\lambda = 0$ and $\delta = 1$.

The inclusion of $\lambda$ in the second term of Eq. 4 allows larger $\lambda$ values to restrict the latent vectors closer together, penalized by the m-MMD loss. This effect increases the probability of sampling valid latent vectors but reduces distinctions between vectors. Further details on changing $\lambda$ are presented in the supplementary material. The normalization factor of $1/\lambda + \delta + 1$ is derived on the basis to make a linear interpolation between the $\mathcal{L}_{\text{CEL}}$ with and without noise.

During training, both the latent vector and the decoder outputs with and without noise are necessary for the calculation of the KAE loss. The latent vectors are penalized based on their differences from 1,000 randomly sampled Gaussian vectors ($\vec{G_i}$) using kernel-based metrics(21). During training, a noise vector $\epsilon \in \mathbb{R}^D$, with $D$ the dimension of the latent space, is added to the latent vector before passing it to the decoder. The noise vector is generated from a Gaussian normal distribution $\mathcal{N}(\mu, \sigma^2)$ with zero mean $\mu = 0$ and unit variance $\sigma = 1$.

In the two passes of the latent vectors to the decoder, one pass resembles an AE-like behavior without noise, while the other pass resembles a VAE-like behavior with added noise to the latent vector before decoding. The reconstructions of both are penalized by $\mathcal{L}_{\text{WCEL}}$. The parameter $\lambda$ controls the shape of the latent vector distribution and the relative weights between the MMD term and the cross-entropy loss. The parameter $\delta$ controls the relative weights between the AE and VAE objectives.

## KAE m-MMD loss

The MMD loss (44), between two distributions having $N_x$ and $N_y$ samples, is defined as their squared distance calculated in a space $\mathcal{F}$ through the transformation $\phi$:

$$\begin{aligned}\text{MMD}(\vec{x}, \vec{y}) &= \|\vec{\mu_x} - \vec{\mu_y}\|^2_{\mathcal{F}}, \\ &= \vec{\mu_x}^{T} \cdot \vec{\mu_x} + \vec{\mu_y}^{T} \cdot \vec{\mu_y} \\ &\quad - \vec{\mu_x}^{T} \cdot \vec{\mu_y} - \vec{\mu_y}^{T} \cdot \vec{\mu_x},\end{aligned} \tag{5}$$

where $\vec{\mu_x} = \frac{1}{N_x}\sum_i^{N_x} \phi(\vec{x_i})$. The space $\mathcal{F}$ is defined by its dot product which can be calculated using a kernel function $\mathcal{K}$. Introducing the kernel

$$\mathcal{K}(\vec{x_i}, \vec{y_j}) = \phi(\vec{x_i})^{T} \cdot \phi(\vec{y_j}), \tag{6}$$

we can write the inner products, as follows:

$$\vec{\mu_x}^{T} \cdot \vec{\mu_y} = \frac{1}{N_x N_y}\sum_i^{N_x}\sum_j^{N_y} \mathcal{K}(\vec{x_i}, \vec{y_j}), \tag{7}$$

so

$$\begin{aligned}\text{MMD}(\vec{x}, \vec{y}) = &\frac{1}{N_y^2}\sum_i^{N_y}\sum_j^{N_y} \mathcal{K}(\vec{y_i}, \vec{y_j}) \\ &+ \frac{1}{N_x^2}\sum_i^{N_x}\sum_j^{N_x} \mathcal{K}(\vec{x_i}, \vec{x_j}) \\ &- \frac{2}{N_x N_y}\sum_{i'}^{N_x}\sum_{j'}^{N_y} \mathcal{K}(\vec{x_{i'}}, \vec{y_{j'}}),\end{aligned} \tag{8}$$

where all $\vec{y}$ are sampled from the target Gaussian distribution, and the kernel is defined as follows:

$$\mathcal{K}(\vec{\alpha}, \vec{\beta}) = \exp\left(\frac{-\frac{1}{D}\sum_{d=0}^{D}(\alpha_d - \beta_d)^2}{2\sigma^2}\right), \tag{9}$$

where $D = 10 \times E$ is the size of the latent dimension and $\sigma = \sqrt{0.32}$ has been empirically chosen (see comparison in supplementary material).

The first term in the r.h.s. of Eq. 8 corresponds to $\vec{\mu_y}^{\mathrm{T}} \cdot \vec{\mu_y}$. It is typically dropped in the loss evaluations since this term does not contribute to the gradients of the loss with respect to the weights during backpropagation. So, the standard-MMD (s-MMD) loss is defined, as follows:

$$\text{s-MMD}(\lambda) = \lambda\left[\frac{1}{N_x^2}\sum_{i}^{N_x}\sum_{j}^{N_x}\mathcal{K}(\vec{x_i}, \vec{x_j}) - \frac{2}{N_x N_y}\sum_{i'}^{N_x}\sum_{j'}^{N_y}\mathcal{K}(\vec{x_{i'}}, \vec{y_{j'}})\right]. \tag{10}$$

For a zero-minimum inner product, the minimum of the first term is achieved at $\vec{\mu_x}$ equal zero. So, minimizing the first term promotes all $\phi(\vec{x_i})$ to spread out in the space $\mathcal{F}$, while the second term brings $\phi(\vec{x})$ to be similar to the distribution of $\phi(\vec{y})$.

Based on the s-MMD loss, introduced by Eq. 10, we define the m-MMD loss, as follows:

$$\text{m-MMD}(\lambda) = \lambda\left[1 - \frac{1}{N_x N_y}\sum_{i}^{N_x}\sum_{j}^{N_y}\mathcal{K}(\vec{x_i}, \vec{y_j})\right]. \tag{11}$$

The constant 1 is added to make m-MMD range from 0 to 1 before the $\lambda$ scaling.

## Decoding methods

KAE's generation process involves sampling a vector, $\vec{v} \in \mathcal{R}^{10\times E}$ from a $D$-dimensional Gaussian distribution and decoding it. For conditional generation (CKAE), the sampled vector is concatenated with a condition $C$, following its multiplication by its corresponding embedding vector. The resulting vector is subsequently mingled by a fully connected layer, yielding $\vec{L}$ again in $\mathcal{R}^{10\times E}$. The decoder then translates the SMILES string sequence, character by character, with decoder–encoder attention applied to $\vec{L}$.

During decoding, the start-of-sequence token is initially supplied. The decoder subsequently generates a probability distribution across $T$ possible tokens for each input. One of the approaches is to continue the predictions using the token possessing the maximum probability, incorporating the token into the next-round input sequence and reiterating the procedure to obtain the next most probable token. This process is repeated until the end-of-sequence token is produced or the sequence length limit is achieved. Besides retaining only the token of highest probability, KAE employed beam search, guided by the hyperparameter beam size, to derive a broader array of interpretations of the same vector, $\vec{L}$. In our implementation, with a beam size, $B$, where $B \leq T$, a maximum of $B$ outputs are produced from a single decoding procedure.

The beam search records the probability at each decoding step for each of the $B$ sequences. For the first step, the top $B$ most probable tokens are selected. In subsequent steps, the model decodes from $B$ input sequences concurrently. Given that each of the $B$ sequences has $T$ potential outcomes for the succeeding token, the total number of potential next-step sequences equates to $B \times T$. These sequences are then ranked according to the sum of their probabilities for all S characters.

In a beam search, the probability of a sequence of tokens indexed from s, s − 1, s − 2 ⋯ to 0 can be represented, as follows:

$$\begin{aligned} &P(s, s-1, s-2, \ldots, 0) \\ &\quad = P(s \mid s-1, s-2, \ldots, 0) \times P(s-1, s-2, \ldots, 0). \end{aligned} \tag{12}$$

This can be interpreted as the product of individual probabilities,

$$\begin{aligned} P(s, s-1, s-2, \ldots, 0) &= P(s \mid s-1, s-2, \ldots, 0) \\ &\times P(s-1 \mid s-2, s-3, \ldots, 0). \times \cdots P(0) \end{aligned} \tag{13}$$

However, calculations of long sequences based on this equation yield impractically small numbers as every term is smaller than one. Therefore, we sum the log probabilities instead and these are also called the beam scores.

For the $B \times T$ sequences with equal sequence length S, the probability of the ith sequence at each position s is denoted as $P_{i,s}$. Excluding the probabilities of padding tokens, the sum of log probabilities, $P_i$ for the ith sequence is computed as

$$P_i = \frac{1}{\sqrt{N_i}}\sum_{s \neq \text{pad}}^{S} \text{Log}(P_{i,s}). \tag{14}$$

Here, $N_i$ represents the quantity of nonpadding tokens in sequence $i$.

To foster diversity in decoding, sequence lengths are factored into the computation of $P_i$. The $1/\sqrt{N_i}$ term counteracts the preference for shorter sequences over longer ones, as longer sequences typically yield smaller sums of log probabilities.

The top $B$ most probable tokens are selected and serve as the inputs for the subsequent iteration, which continues until the maximum sequence length $M$ is attained or all top $B$ candidates have produced the end-of-sequence token, signaling the cessation of decoding.

## Docking methods

The generated molecular structures were evaluated using Autodock Vina (25), following a procedure that ensures meaningful comparisons to other molecular generation models, such as GFlowNet (5). All results were independently tested by using Glide docking from Schrodinger Inc. (26, 27) to ensure the results are robust across different docking software packages.

Autodock Vina is known for its efficiency and speed, making it suitable for high-throughput screening. It employs an empirical scoring function for accurate prediction of binding affinities. On the other hand, Glide utilizes a force field-based scoring function that is widely recognized for its accuracy. In particular, Glide excels at predicting binding poses with high precision and has undergone extensive validation. Its efficacy in handling large and flexible ligands has established it as the gold standard in the field. To ensure meaningful comparisons to GFlowNet (5), we followed the same procedure implemented for Autodock Vina calculations. Specifically, 20 conformers were used per ligand, exhaustiveness was set to 32, and a maximum of 10 binding modes were generated.

### *Glide calculations*

The model protein receptor (PDB ID: 4jnc) was prepared by using the adept Protein Preparation Wizard tool in Maestro (45). The protonation states were defined at a neutral pH = 7.0. The protein was subsequently refined via energy minimization using the OPLS4 force field (46).

The 3D grid representation of the receptor binding site was prepared by using the Maestro Grid Generation tool, ensuring that the grid size and positioning was perfectly aligned with those used in GFlowNet calculations. All model structures for docking were prepared using the LigPrep tool of Maestro. Utilizing the Pre-Dock tool in Maestro, the docked molecules were prepared and assigned charges and protonation states via the OPLS4 force field (46). The XP (26, 27) flexible docking protocol was then implemented, employing a range of settings designed to optimize the docking accuracy and precision. These included a selection of all predefined functional groups for biased torsional sampling, the addition of Epik state penalties (47) to the docking scores (45), and the enhanced planarity setting for conjugated pi groups.

In the initial step of the docking procedure, 10,000 poses were filtered through the Glide screens, and the top 1,000 poses were selected for energy minimization. The expanded sampling option was utilized to maximize pose flexibility during the search. Ultimately, a single pose was retained for each ligand. The final stage involved refining the best docking poses. Two consecutive refinement steps were performed, each consisting of a postdocking energy minimization on the selected pose, eliminating the need for additional sampling. As a result, highly optimized and reliable docking poses were obtained and compared against those obtained with Autodock Vina (25) calculations.

## Conclusion

KAE allows the integration of the strengths of both VAE and AE frameworks in applications to molecular design. The KAE loss, with hyperparameters $\lambda$ and $\delta$, controls varying degrees of VAE and AE features as needed for the specific applications.

In the context of molecule generation, KAE is the top performer in terms of generation validity without the need for any chemical knowledge-based checks, and is the only one that synchronously reporting reconstruction performance near 100% accuracy akin to the AE. With beam search decoding (48–50) multiple candidates per latent vector can be derived. This enriched KAE's generation diversity and validity. In the context of conditional generation, CKAE generates molecules that exhibit excellent correlation with the input condition, including molecules with a desired property (e.g. specific value of PLogP, or reward value upon docking to a specific binding site of a biological target).

In the constrained optimization task, the CKAE model exhibits significant improvements, with an average increase of $7.67 \pm 1.61$ in PLogP units. CKAE achieves a 100% success rate, indicating that modifications leading to higher PLogP values were successfully achieved for all molecules within the defined similarity constraints. This improvement surpasses directly searching from the training dataset by over 65%. The comparison to "Random Search" shows the strength of KAE's accurate reconstruction which makes searching around the molecules much more efficient.

Using Glide (26, 27), the validation for CKAE's generated high binding affinity candidates reveals that they consistently outperform those from the training dataset as well as all structurally similar tautomers, demonstrating CKAE's ability of extrapolation and the quality of the generated molecules.

As demonstrated in this article, KAE can be used to tackle molecule generation problems such as docking with binding affinity and constrained optimization with PlogP labels. Outside of the context for molecular designs, KAE can be effectively employed to address a wide spectrum of problems, especially for those that are labeled by example–property pairs.

## Acknowledgments

V.B. acknowledges computational resources from the National Energy Research Scientific Computing Center (NERSC), a US Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under contract no. DE-AC02-05CH11231 using NERSC award BES-ERCAP0024372.

## Supplementary Material

Supplementary material is available at *PNAS Nexus* online.

## Funding

V.B. acknowledges support from the National Science Foundation (NSF) CCI grant, award no. 2124511.

## Author Contributions

H.L. developed the loss function and constrained optimization methods, built relevant model architectures, and wrote the initial manuscript. Y.S. analyzed model learning behaviors, performed ablations studies for combinations of hyperparameters, and edited the manuscript. B.A. performed docking validation of the generated candidates using Autodock Vina and edited the manuscript. F.M. performed docking validation of the top candidates from Autodock Vina in Glide and edited the manuscript. A.M. helped building the API interface and implementating of the code. V.B. analyzed the results, acquired funding, and edited the paper.

## Preprints

This manuscript was posted on a preprint: arxiv.org/abs/2310.08685.

## Data Availability

Pretrained KAE can be accessed through an API interface at kaemd.batistalab.com. For inquiries, please reach out to victor.batista@yale.edu.

## References

1 Maziarka L, *et al.* 2020. Mol-CycleGAN: a generative model for molecular optimization. *J Cheminform.* 12(1):1–18.

2 Moret M, Friedrich L, Grisoni F, Merk D, Schneider G. 2020. Generative molecular design in low data regimes. *Nat Mach Intell.* 2(3):171–180.

3 Skalic M, Jiménez J, Sabbadin D, De Fabritiis G. 2019. Shape-based generative modeling for de novo drug design. *J Chem Inf Model.* 59(3):1205–1214.

4 Wang J, *et al.* 2021. Multi-constraint molecular generation based on conditional transformer, knowledge distillation and reinforcement learning. *Nat Mach Intell.* 3(10):914–922.

5 Bengio Y, *et al.* 2023. GFlowNet foundations. *J Mach Learn Res.* 24(210):1–55.

6 Hoogeboom E, Satorras VG, Vignac C, Welling M. 2022. Equivariant diffusion for molecule generation in 3D. In: International Conference on Machine Learning. Baltimore, Maryland: PMLR. p. 8867–8887.

7 Prykhodko O, *et al.* 2019. A de novo molecular generation method using latent vector based generative adversarial network. *J Cheminform.* 11(1):1–13.

8 Kingma DP, Welling M. 2013. Auto-encoding variational bayes, arXiv, arXiv:1312.6114, preprint: version 11, not peer reviewed.
9 Gómez-Bombarelli R, *et al.* 2018. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent Sci*. 4(2):268–276.
10 Kusner MJ, Paige B, Hernández-Lobato JM. 2017. Grammar variational autoencoder. In: International Conference on Machine Learning. Sydney, Australia: PMLR. p. 1945–1954.
11 Jin W, Barzilay R, Jaakkola T. 2018. Junction tree variational autoencoder for molecular graph generation. In: International Conference on Machine Learning. Stockholm, Sweden: PMLR. p. 2323–2332.
12 Kramer M. 1991. Nonlinear principal component analysis using autoassociative neural networks. AICHE J. 37(2): 233–243.
13 Hoffman SC, Chenthamarakshan V, Wadhawan K, Chen P-Y, Das P. 2022. Optimizing molecules using efficient queries from property evaluations. *Nat Mach Intell*. 4(1):21–31.
14 Van De Waterbeemd H, Smith DA, Beaumont K, Walker DK. 2001. Property-based design: optimization of drug absorption and pharmacokinetics. *J Med Chem*. 44(9):1313–1333.
15 He J, *et al.* 2021. Molecular optimization by capturing chemist's intuition using deep neural networks. *J Cheminform*. 13(1):1–17.
16 Chen Z, Min MR, Parthasarathy S, Ning X. 2021. A deep generative model for molecule optimization via one fragment modification. *Nat Mach Intell*. 3(12):1040–1049.
17 Vaswani A. 2017. Attention is all you need. *Adv Neural Inf Process Syst*. 30:3–10.
18 Dollar O, Joshi N, Beck DAC, Pfaendtner J. 2021. Attention-based generative models for de novo molecular design. *Chem Sci*. 12(24): 8362–8372.
19 Jiang J, Xia GG, Carlton DB, Anderson CN, Miyakawa RH. 2020. Transformer VAE: a hierarchical model for structure-aware and interpretable music representation learning. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Georgia: IEEE. p. 516–520.
20 Wang T, Wan X. 2019. T-CVAE: transformer-based conditioned variational autoencoder for story completion. In: IJCAI. Macao, China. p. 5233–5239.
21 Zhao S, Song J, Ermon S. 2019. InfoVAE: balancing learning and inference in variational autoencoders. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. Hawaii, USA. p. 5885–5892.
22 Ucar T. 2019. Bridging the ELBO and MMD, arXiv, arXiv:1910.13181, preprint: not peer reviewed.
23 Louizos C, Swersky K, Li Y, Welling M, Zemel R. 2015. The variational fair autoencoder, arXiv, arXiv:1511.00830, preprint: not peer reviewed.
24 Richards RJ, Groener AM. 2022. Conditional $\beta$-VAE for de novo molecular generation, arXiv, arXiv:2205.01592, preprint: not peer reviewed.
25 Trott O, Olson AJ. 2010. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *J Comput Chem*. 31(2):455–461.
26 Halgren TA, *et al.* 2004. Glide: a new approach for rapid, accurate docking and scoring. 2. enrichment factors in database screening. *J Med Chem*. 47(7):1750–1759.
27 Friesner RA, *et al.* 2006. Extra precision glide: docking and scoring incorporating a model of hydrophobic enclosure for protein-ligand complexes. *J Med Chem*. 49(21):6177–6196.
28 Zang C, Wang F. 2020. Moflow: an invertible flow model for generating molecular graphs. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. California, USA. p. 617–626.
29 Yan C, Yang J, Ma H, Wang S, Huang J. 2022. Molecule sequence generation with rebalanced variational autoencoder loss. *J Comput Biol*. 30(1):82–94.
30 Luo Y, Yan K, Ji S. 2021. Graphdf: a discrete flow model for molecular graph generation. In: International Conference on Machine Learning. PMLR. p. 7192–7203.
31 Alperstein Z, Cherkasov A, Rolfe JT. 2019. All smiles variational autoencoder, arXiv, arXiv:1905.13343, preprint: not peer reviewed.
32 Landrum G, *et al.* 2016. RDKit: open-source cheminformatics software.
33 Wildman SA, Crippen GM. 1999. Prediction of physicochemical parameters by atomic contributions. *J Chem Inf Comput Sci*. 39(5):868–873.
34 Ertl P, Schuffenhauer A. 2009. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *J Cheminform*. 1(1):1–11.
35 Lim J, Ryu S, Kim JW, Kim WY. 2018. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *J Cheminform*. 10(1):1–9.
36 Ma C, Zhang X. 2021. Gf-VAE: a flow-based variational autoencoder for molecule generation. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. Queensland, Australia. p. 1181–1190.
37 Kajino H. 2019. Molecular hypergraph grammar with its application to molecular optimization. In: International Conference on Machine Learning. California, USA: PMLR. p. 3183–3191.
38 You J, Liu B, Ying Z, Pande V, Leskovec J. 2018. Graph convolutional policy network for goal-directed molecular graph generation. *Adv Neural Inf Process Syst*. 31:3–8.
39 Zhou Z, Kearnes S, Li L, Zare RN, Riley P. 2019. Optimization of molecules via deep reinforcement learning. *Sci Rep*. 9(1):1–10.
40 Xu C, Liu Q, Huang M, Jiang T. 2020. Reinforced molecular optimization with neighborhood-controlled grammars. *Adv Neural Inf Process Syst*. 33:8366–8377.
41 Bengio E, Jain M, Korablyov M, Precup D, Bengio Y. 2021. Flow network based generative models for non-iterative diverse candidate generation. *Adv Neural Inf Process Syst*. 34:27381–27394.
42 Korablyov M, Precup M, Bengio D, Bengio Y, Jain E. 2022. Flow network based generative models for non-iterative diverse candidate generation. https://github.com/GFNOrg/gflownet.
43 Higgins I. 2017. beta-VAE: learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations. Toulon, France.
44 Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A. 2012. A kernel two-sample test. *J Mach Learn Res*. 13(1):723–773.
45 Schrödinger, LLC. 2023. Schrödinger release 2023-2: Maestro. New York (NY): Schrödinger, LLC.
46 Lu C, *et al.* 2021. OPLS4: improving force field accuracy on challenging regimes of chemical space. *J Chem Theory Comput*. 17(7): 4291–4300.
47 Greenwood JR, Calkins D, Sullivan AP, Shelley JC. 2010. Towards the comprehensive, rapid, and accurate prediction of the favorable tautomeric states of drug-like molecules in aqueous solution. *J Comput Aided Mol Des*. 24(6–7):591–604.
48 Graves A. 2012. Sequence transduction with recurrent neural networks, arXiv, arXiv:1211.3711, preprint: not peer reviewed.
49 Bengio Y, Boulanger-Lewandowski N, Pascanu R. 2013. Advances in optimizing recurrent networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. California, USA: IEEE. p. 8624–8628.
50 Guo J, Schwaller P. 2023. Beam enumeration: probabilistic explainability for sample efficient self-conditioned molecular design, arXiv, arXiv:2309.13957, preprint: not peer reviewed.