Generative Modeling for Molecules

Sven Lüpke
Technical University of Munich
luepke@in.tum.de

1 Introduction

The goal of inverse molecular design is the discovery of novel molecular structures that fulfill a set of target properties. In this project we explore the application of generative models and molecular graph representations for the development of solutions to the inverse molecular design problem.

2 Probabilistic Inverse Design with Generative Models

Given target properties y, our objective is to find molecules x that satisfy those properties by sampling the distribution $p(x|y)$. Due to the discrete nature of x, i.e., a graph, sampling this density directly is challenging. Therefore we introduce a continuous latent variable [6] z and define the generative model

$$p_\theta(x, y) = \int p_\theta(x|z)p_\theta(y|z)p_\theta(z)dz,$$

where $p_\theta(x|z)$ is the decoder, $p_\theta(y|z)$ the property predictor and $p_\theta(z)$ the prior on z.

Once we have optimized the model parameters θ, we can sample $p(x|y)$ through the following steps:

1. Sample $\hat{z} \sim p(z|y) \propto p_\theta(y|z)p_\theta(z)$
2. Sample $\hat{x} \sim p_\theta(x|\hat{z})$

3 Graph Variational Autoencoder

3.1 Method

Our first implementation of the generative model is based on the GraphVAE model by Simonovsky et al. [13], which uses a fully connected decoder $p_\theta(x|z)$, a graph convolutional encoder $q_\phi(z|x)$ and a standard normal prior on the latent variable z, i.e. $p_\theta(z) = \mathcal{N}(0, I)$.

We introduce several modifications to the original model. Firstly, we increase the capacity of the model by replacing all ReLU activations with the more flexible parametric ReLU function, adding an additional graph convolutional layer in the encoder, and adding an additional fully connected layer in each of the decoder paths for the adjacency matrix, the edge feature matrix and the node feature matrix. Secondly, we follow Gómez-Bombarelli et al. [6] and add a fully connected property predictor containing two hidden layers with 69 neurons each to the model. The output values of the property predictor are the mean and standard deviation of a normal distribution, i.e., $p_\theta(y|z) = \mathcal{N}(y|\mu_\theta(z), \sigma_\theta(z)^2)$. Lastly, we partition the latent variable z into two parts z_1 and z_2, i.e. $z = (z_1, z_2)$, predict properties only from z_1 and define the joint generative model

$$p(x, y) = \int\int p_\theta(x|z_1, z_2)p_\theta(y|z_1)\mathcal{N}(z_1|0, I_n)\mathcal{N}(z_2|0, I_m)dz_1dz_2.$$

H-Atoms	# Parameters	log $\mathbf{p}_\theta(\mathbf{G}\|\mathbf{z})$ ↑	Valid ↑	Connected ↑	Unique ↑	Novel ↑
✓	2.2M	-0.151	0.014	0.049	0.979	1.000
	1.2M	-0.203	0.700	0.862	0.899	0.704

Table 1: Impact of including hydrogen atoms on the graph variational autoencoder.

Model	# Parameters		Properties	Valid ↑	Connected ↑	Unique ↑	Novel ↑
	Encoder	Decoder					
SMILES VAE	29K	3.8M		0.054	1.000	0.998	0.967
SMILES VAE	29K	3.8M	✓	0.093	1.000	0.993	0.912
Graph VAE	129K	1.1M		0.700	0.862	0.899	0.704
Graph VAE	129K	1.1M	✓	0.687	0.862	0.866	0.700
Mixture ($N = 9$)	90K	3.2K		0.027	0.083	0.974	0.765

Table 2: Comparison of generative models.

where $n = dim(z_1) = 16$ and $m = dim(z_2) = 112$.

To optimize the model, we utilize the encoder network q_ϕ to maximize the evidence lower bound using amortized variational inference, leading to the loss function

$$\begin{aligned}\mathcal{L} = &- \mathbb{E}_{z\sim q_\phi(z|x)}[log(p_\theta(x|z)) + log(p_\theta(y|z)] \\ &+ \beta_1 \cdot \mathbb{KL}(q_\phi(z_1|x)||\mathcal{N}(0, I_n)) \\ &+ \beta_2 \cdot \mathbb{KL}(q_\phi(z_2|x)||\mathcal{N}(0, I_m)).\end{aligned}$$

where $\beta_1 = 0.1$ and $\beta_2 = 0.01$.

Given the trained model, we perform inverse design for a given target property $\hat{y}$ through the following procedure:

1. Sample $z_1 \sim p(z_1|\hat{y}) \propto p_\theta(\hat{y}|z_1)\mathcal{N}(z_1|0, I_n)$ using a Hamiltonian Monte Carlo sampler [3]
2. Sample $z_2 \sim \mathcal{N}(z_2|0, I_m)$
3. Decode and sample $\hat{x} \sim p_\theta(x|z_1, z_2)$
4. Encode $\mathbb{E}[q_\phi(z|\hat{x})] = (\hat{z_1}, \hat{z_2})$
5. Predict the property distribution $p_\theta(y|\hat{z_1})$

By repeating this procedure K times we obtain a set of candidate molecules $\{\hat{x_k}\}_{k=1}^K$ and corresponding latent vectors $\{\hat{z_{1k}}\}_{k=1}^K$. The best candidate molecule x_i is then selected to be the one that most likely satisfies the target property, i.e.

$$\underset{i\in(1,K)}{\arg\max}\, log(p_\theta(\hat{y}|\hat{z_{1i}})).$$

3.2 Experiments and Results

For all of our experiments, we used the QM9 dataset [10, 9], which contains 133885 molecules with up to 9 heavy atoms. Following Simonovsky et al. [13], we do not explicitly generate Hydrogen atoms but instead add them as "padding" during the chemical validity check, which reduces the maximum graph size from 29 nodes to 9 nodes and improves the performance of our model (Table 1).

We trained the VAE jointly with the property predictor for 100 epochs using the Adam optimizer with a learning rate of 0.001 and a cyclic annealing schedule [5] of the weights β_1 and β_2 of the KL divergence terms.

The SMILES-based variational autoencoder by Gómez-Bombarelli et al. [6] served as a baseline for evaluating our generative model and the property predictor.

We evaluated the generative model using the fraction of chemically valid molecules according to RDKit [8], the number of generated graphs that are connected, the uniqueness of the generated molecules and their novelty, i.e., the fraction of generated molecules that are not present in the

	logP	SAS	QED
QM9	0.33 ± 0.95	4.20 ± 0.92	0.47 ± 0.07
SMILES VAE	0.46 ± 0.96	4.86 ± 0.91	0.49 ± 0.07
Graph VAE	0.84 ± 1.20	3.75 ± 1.07	0.47 ± 0.08
Mixture ($N = 9$)	1.23 ± 1.11	4.57 ± 1.32	0.51 ± 0.06

Table 3: Mean and standard deviation of chemical properties of the molecules generated by each model compared to the dataset.

Model	HOMO ↓	LUMO ↓	R2 ↓
Mean	0.439	1.050	198.5
SMILES VAE	0.320	0.482	123.7
Graph VAE	0.121	0.150	39.09

Table 4: Property prediction MAE of our models compared to two baselines, each trained jointly on all properties.

training data. The results are shown in Table 2. Additionally, we compared the distributions of various chemical properties of the generated molecules, including the octanol-water partition coefficient (logP), the synthetic accessibility score (SAS) [4], and the quantitative estimate of drug-likeness (QED) (Table 3). The property predictor was evaluated using the mean absolute error (MAE) between the predicted and the true property (Table 4).

We also studied the impact of the weight of the Kullback-Leibler divergence. Table 5 and Figure 11 show how a high weight results in a low variety of generated molecules but a high property coverage, whereas a low weight results in a high molecule variety but a low property coverage. This highlights the necessity of our split latent approach, which allows us to maintain both a high molecule diversity and a high property coverage.

To understand the behavior of our model better, we also evaluated the impact of the property predictor on the latent distribution (Figure 2), how the discrete molecular structure changes while moving through the continuous latent space (Figure 1, Figure 3), and the consistency between the decoder and the encoder distribution (Figure 4). Our results show that in these aspects, our graph-based model exhibits a behavior very similar to the SMILES-based model by Gómez-Bombarelli et al. [6].

We evaluated our inverse design implementation separately on the LUMO (Figure 5), R2 (Figure 6) and HOMO (Figure 6) properties, each with 5 different target values. Our evaluation, which includes the lookup of the actual property values of non-novel molecules from the dataset, shows that our method can generate molecules whose properties are close to the target properties.

Figure 12 also highlights the importance of high β_1 values in the training loss for sampling the posterior of the latent variables given a target property with higher precision.

4 Mixture Model

4.1 Method

For a molecule with N heavy atoms, we define the following generative process:

1. Compute $\pi = softmax(\eta)$
2. For $n = 1, ..., N$:
   - Sample $c_n \sim Cat(\pi)$
   - Sample $z_n \sim \mathcal{N}(\mu_i, \sigma_i^2 I)$, where $i = c_n$
   - Sample $F_n \sim Cat(softmax(MLP(z_n; \theta)))$
3. For all pairs $n < m = 1, ..., N$
   - Sample $E_{nm} \sim Cat(softmax(g(z_n, z_m)))$, where $g_i(z_n, z_m) = z_n^T W_i z_m$

This model contains the following learnable parameters:

- $\eta \in \mathbb{R}^K$
- $\{\mu_i, \sigma_i\}_{i=1}^K$ each of which is of dimension C
- θ the parameters of a multilayer perceptron with one hidden layer
- $\{W_i\}_{i=1}^{|E|}$ each of which is a **symmetric** matrix of dimension $C \times C$

The generative process described above implies the latent variables $\{c_i\}_{i=1}^N$ and $\{z_i\}_{i=1}^N$ and the following generative model:

$$\begin{aligned} p_\theta(E, F, z_{1:N}, c_{1:N}) &= p_\theta(c_{1:N}) p_\theta(z_{1:N}|c_{1:N}) p_\theta(F|z_{1:N}) p_\theta(E|z_{1:N}) \\ &= \prod_{i=1}^N p_\theta(c_i) \prod_{i=1}^N p_\theta(z_i|c_i) \prod_{i=1}^N p_\theta(F_i|z_i) \prod_{i=1}^N \prod_{j=1}^{i-1} p_\theta(E_{ij}|z_i, z_j) \end{aligned}$$

We optimize this model by maximizing a lower bound on the evidence using amortized variational inference. Assuming that the approximate posterior factorizes as

$$\begin{aligned} q_\phi(z_{1:N}, c_{1:N}|E, F) &= q_\phi(z_{1:N}|E, F) p(c_{1:N}|z_{1:N}) \\ &= \prod_{i=1}^N q_\phi(z_i|E, F) \prod_{i=1}^N p(c_i|z_i), \end{aligned}$$

where $q_\phi(z_{1:N}|E, F)$ is a node-wise normal distribution whose mean and standard deviation are predicted by a graph convolutional neural network and the posterior over the cluster weights $p(c_i|z_i) = Cat(\pi^q)$ is computed in closed form

$$\pi_j^q = \frac{\pi_j \mathcal{N}(z_i|\mu_j, \sigma_j^2 I)}{\sum_{k=1}^K \pi_k \mathcal{N}(z_i|\mu_k, \sigma_k^2 I)}.$$

Derived from the ELBO, we arrive at the following loss function:

$$\begin{aligned} \mathcal{L} = &\mathbb{E}_{q_\phi(z)}\big[\sum_{i=1}^N ln(p_\theta(F_i|z_i))\big] + \mathbb{E}_{q_\phi(z)}\big[\sum_{i=1}^N \sum_{j=1}^{i-1} ln(p_\theta(E_{ij}|z_i, z_j))\big] \\ &- \sum_{i=1}^N \mathbb{E}_{q_\phi(z_i)}\Big[KL(p_\theta(c_i|z_i)||p_\theta(c_i))\Big] \\ &- \sum_{i=1}^N \mathbb{E}_{p_\theta(c_i|z_i)}\Big[KL(q_\phi(z_i|E, F)||p_\theta(z_i|c_i))\Big] \cdot \beta \end{aligned}$$

A full derivation of the loss can be found in Appendix A. All expectations over z are approximated with a single sample Monte Carlo estimator, whereas the Kullback-Leibler divergence terms, including categorical and Gaussian distributions, can be computed in closed form [1]. To backpropagate gradients through the full model, we use the Gumbel-Softmax to sample the categorical distribution with reparameterization [7]. β is a scaling additional constant set to 0.05 to avoid a posterior collapse.

4.2 Experiments and Results

For our experiments, we trained the mixture model for 25 epochs using the Adam optimizer and a cosine annealing schedule increasing β from 0 to 0.05 during the first 7 epochs. Unfortunately, our posterior approximation seems insufficient because our model always chooses to ignore all clusters but one independent of the number of clusters.

An advantage of this model with node-local latent variables over the previous variational autoencoder model with global latent variables is that it can synthesize molecules of arbitrary sizes, including sizes not present in the training data. However, the larger the number of atoms, the more challenging it becomes for the model to synthesize valid molecular structure (Table 6). Figure 13 shows examples of molecules generated by the mixture model.

β	**log** $\mathbf{p}_\theta(\mathbf{G}\|\mathbf{z})$ ↑	**Valid** ↑	**Connected** ↑	**Unique** ↑	**Novel** ↑	**LUMO MAE** ↓
1.0	-1.232	1.000	1.000	0.001	0.400	0.661
0.2	-1.197	0.982	0.986	0.003	0.539	0.155
0.1	-0.986	0.898	0.902	0.053	0.723	0.134
0.05	-0.656	0.794	0.828	0.377	0.709	0.137
0.02	-0.380	0.794	0.863	0.763	0.671	0.129
0.01	-0.307	0.699	0.858	0.850	0.701	0.130

Table 5: Impact of the weight $\beta = \beta_1 = \beta_2$ of the KL divergence on the generative performance of the graph variational autoencoder and the property predictor.

N	**Valid** ↑	**Connected** ↑	**Unique** ↑	**Novel** ↑
14	0.001	0.102	1.000	1.000
13	0.002	0.098	1.000	1.000
12	0.005	0.094	1.000	1.000
11	0.010	0.091	1.000	1.000
10	0.016	0.087	0.996	1.000
9	0.026	0.081	0.969	0.733
8	0.039	0.079	0.846	0.753
7	0.048	0.073	0.580	0.783
6	0.060	0.074	0.278	0.838
5	0.068	0.073	0.110	0.883
4	0.079	0.081	0.036	0.878
3	0.101	0.101	0.009	0.821
2	0.175	0.175	0.001	0.750

Table 6: Mixture model generative performance for different numbers of heavy atoms N.

5 Conclusion and Future Work

In this project, we developed two graph generative models and a procedure that combines a generative model with an additional property predictor to perform inverse molecular design. Even though the mixture model performed surprisingly well despite its simplicity, we believe that further improvements could be achieved with more accurate posterior approximations during training [11, 2]. In general, the fact that our graph generative models can generate graphs that contain more than one connected component proved to be a weakness compared to SMILES-based models. Thus, enforcing connectedness in our models could be an interesting direction for future work.

The accurate prediction of properties is a crucial component of our probabilistic inverse design framework, where a more precise property prediction will increase the precision of the posterior molecule distribution given a set of target properties. The inclusion of additional information about the molecules, such as the atomic positions [12], has the potential to significantly improve the accuracy of the property predictor.

Code

The code for the graph variational autoencoder and the mixture model, including notebooks for replicating the figures in this paper, can be found on here: `https://gitlab.lrz.de/maxstupp/idp_generative_modeling`.

We have also adapted the official implementation of the baseline model by Gómez-Bombarelli et al. [6] to the QM9 dataset, which can be found here: `https://github.com/KnightTec/chemical_vae`.

References

[1] Dmitry I Belov and Ronald D Armstrong. Distributions of the kullback–leibler divergence with applications. *British Journal of Mathematical and Statistical Psychology*, 64(2):291–309, 2011.

[2] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[3] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

[4] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1:1–11, 2009.

[5] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*, 2019.

[6] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[7] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[8] Greg Landrum et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8:31, 2013.

[9] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[10] Lars Ruddigkeit, Ruud Van Deursen, Lorenz C Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17. *Journal of chemical information and modeling*, 52(11):2864–2875, 2012.

[11] Tim Salimans, Diederik Kingma, and Max Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International conference on machine learning*, pages 1218–1226. PMLR, 2015.

[12] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

[13] Martin Simonovsky and Nikos Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pages 412–422. Springer, 2018.

A Derivation of the mixture model loss

$$
\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q(z,c)}\left[ln\frac{p_\theta(E,F,z_{1:N},c_{1:N})}{q_\phi(z_{1:N},c_{1:N}|E,F)}\right] \\
&= \mathbb{E}_{q(z,c)}\left[ln\frac{p_\theta(c_{1:N})p_\theta(z_{1:N}|c_{1:N})p_\theta(F|z_{1:N})p_\theta(E|z_{1:N})}{q_\phi(z_{1:N}|E,F)p(c_{1:N}|z_{1:N})}\right] \\
&= \mathbb{E}_{q_\phi(z)}\big[ln(p_\theta(F|z_{1:N})p_\theta(E|z_{1:N}))\big] + \mathbb{E}_{q(z,c)}\left[ln\frac{p_\theta(c_{1:N})p_\theta(z_{1:N}|c_{1:N})}{q_\phi(z_{1:N}|E,F)p(c_{1:N}|z_{1:N})}\right] \\
&= \mathbb{E}_{q_\phi(z)}\big[ln(p_\theta(F|z_{1:N}))\big] + \mathbb{E}_{q_\phi(z)}\big[ln(p_\theta(E|z_{1:N}))\big] \\
&\qquad + \mathbb{E}_{q(z,c)}\left[ln\frac{p_\theta(c_{1:N})}{p(c_{1:N}|z_{1:N})}\right] + \mathbb{E}_{q(z,c)}\left[ln\frac{p_\theta(z_{1:N}|c_{1:N})}{q_\phi(z_{1:N}|E,F)}\right] \\
&= \mathbb{E}_{q_\phi(z)}\big[ln(p_\theta(F|z_{1:N}))\big] + \mathbb{E}_{q_\phi(z)}\big[ln(p_\theta(E|z_{1:N}))\big] \\
&\qquad + \mathbb{E}_{q_\phi(z)}\mathbb{E}_{p_\theta(c|z)}\left[ln\frac{p_\theta(c_{1:N})}{p(c_{1:N}|z_{1:N})}\right] \\
&\qquad + \mathbb{E}_{p(c|z)}\mathbb{E}_{q_\phi(z)}\left[ln\frac{p_\theta(z_{1:N}|c_{1:N})}{q_\phi(z_{1:N}|E,F)}\right] \\
&= \mathbb{E}_{q_\phi(z)}\big[ln(\prod_{i=1}^{N}p_\theta(F_i|z_i))\big] + \mathbb{E}_{q_\phi(z)}\big[ln(\prod_{i=1}^{N}\prod_{j=1}^{i-1}p_\theta(E_{ij}|z_i,z_j))\big] \\
&\qquad + \mathbb{E}_{q_\phi(z)}\mathbb{E}_{p_\theta(c|z)}\left[ln\prod_{i=1}^{N}\frac{p_\theta(c_i)}{p(c_i|z_i)}\right] \\
&\qquad + \mathbb{E}_{p(c|z)}\mathbb{E}_{q_\phi(z)}\left[ln\prod_{i=1}^{N}\frac{p_\theta(z_i|c_i)}{q_\phi(z_i|E,F)}\right] \\
&= \mathbb{E}_{q_\phi(z)}\big[\sum_{i=1}^{N}ln(p_\theta(F_i|z_i))\big] + \mathbb{E}_{q_\phi(z)}\big[\sum_{i=1}^{N}\sum_{j=1}^{i-1}ln(p_\theta(E_{ij}|z_i,z_j))\big] \\
&\qquad + \mathbb{E}_{q_\phi(z)}\mathbb{E}_{p_\theta(c|z)}\left[\sum_{i=1}^{N}ln\frac{p_\theta(c_i)}{p(c_i|z_i)}\right] \\
&\qquad + \mathbb{E}_{p(c|z)}\mathbb{E}_{q_\phi(z)}\left[\sum_{i=1}^{N}ln\frac{p_\theta(z_i|c_i)}{q_\phi(z_i|E,F)}\right] \\
&= \mathbb{E}_{q_\phi(z)}\big[\sum_{i=1}^{N}ln(p_\theta(F_i|z_i))\big] + \mathbb{E}_{q_\phi(z)}\big[\sum_{i=1}^{N}\sum_{j=1}^{i-1}ln(p_\theta(E_{ij}|z_i,z_j))\big] \\
&\qquad - \sum_{i=1}^{N}\mathbb{E}_{q_\phi(z_i)}\Big[KL(p(c_i|z_i)||p_\theta(c_i))\Big] \\
&\qquad - \sum_{i=1}^{N}\mathbb{E}_{p(c_i|z_i)}\Big[KL(q_\phi(z_i|E,F)||p_\theta(z_i|c_i))\Big]
\end{aligned}
$$

B Figures

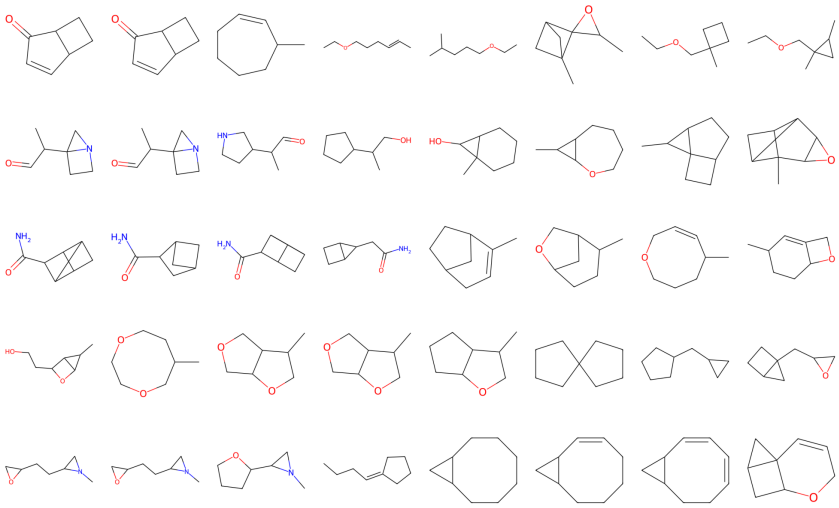
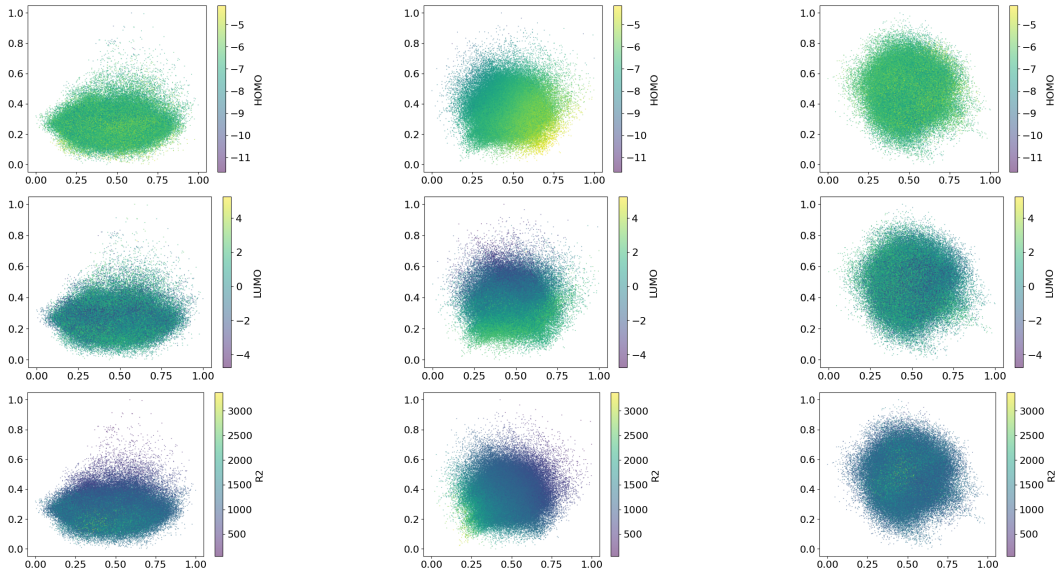


Figure 1: Spherical linear interpolation between 5 pairs of molecules in the latent space. The molecules to the far left and the far right are encoded and decoded at 6 equidistant steps during the interpolation.



(a) z (b) z_1 (c) z_2

Figure 2: Principle component analysis of the latent posterior distribution given all molecules in the dataset. a) shows the PCA of the latent space of a model without property prediction. b) and c) show the PCA plots of the two latent partitions, z_1 and z_2, of a model trained with property prediction. Because properties are only predicted from z_1, the latent space is structure w.r.t. the properties only in z_1 and not in z_2.

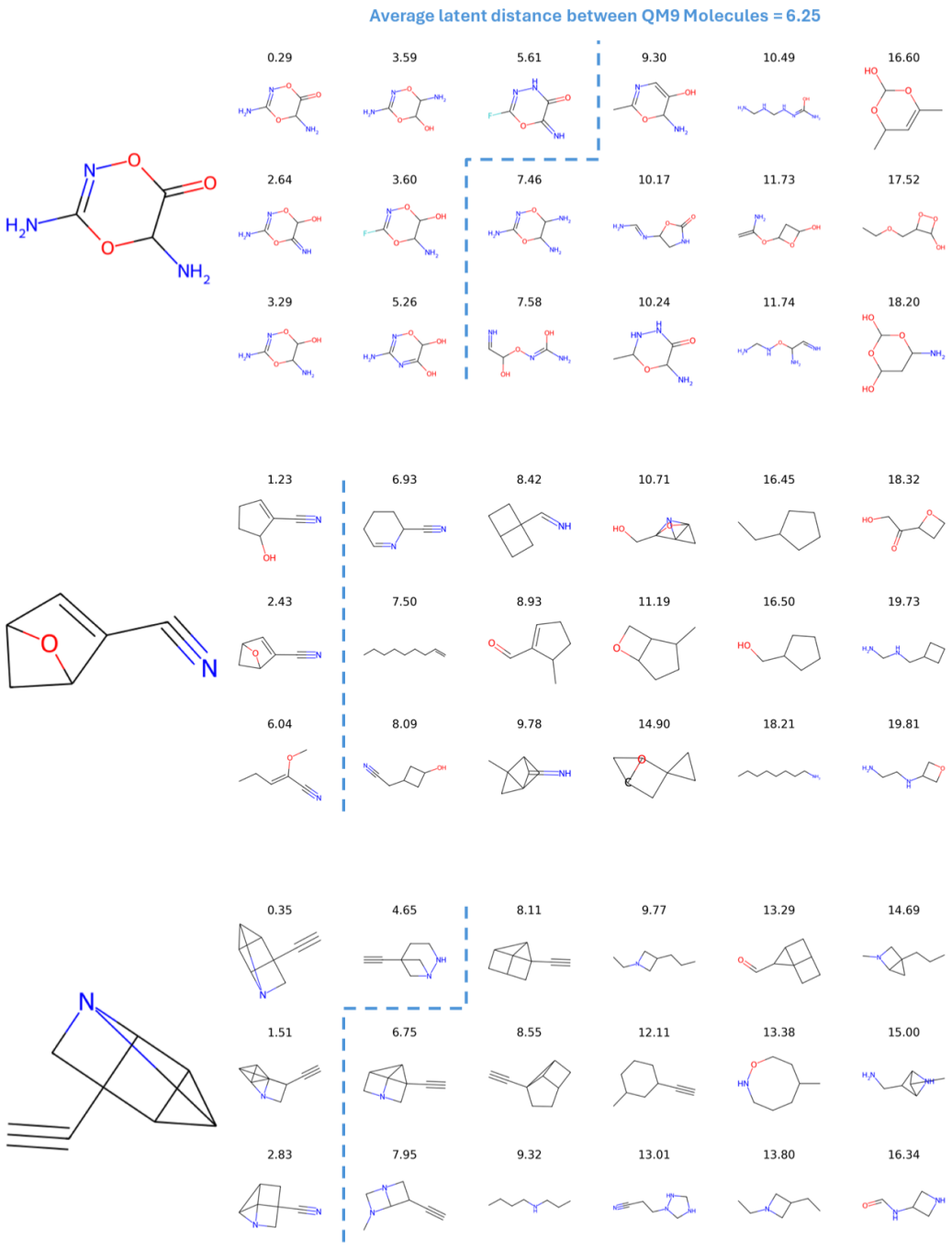


Figure 3: Molecules sampled in the latent neighborhood of the 3 molecules on the left. The values above the molecules are the latent L2 distances to the molecule on the left.

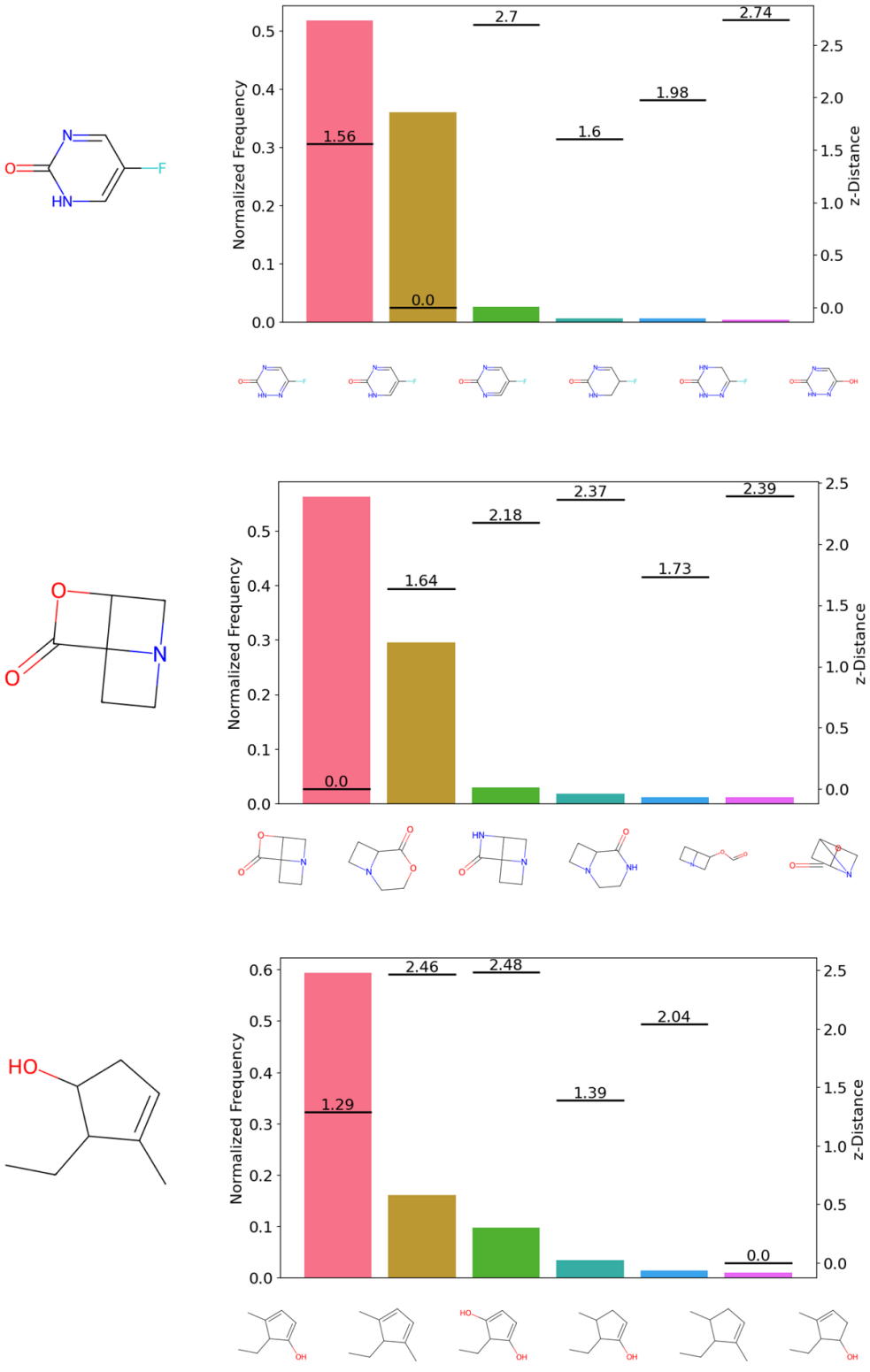


Figure 4: GraphVAE decoding distribution of the top-6 most frequently decoded molecules after encoding the molecule on the left, including the latent distance of the decoded molecules to the original molecule. This highlights the consistency between the decoder and the encoder.

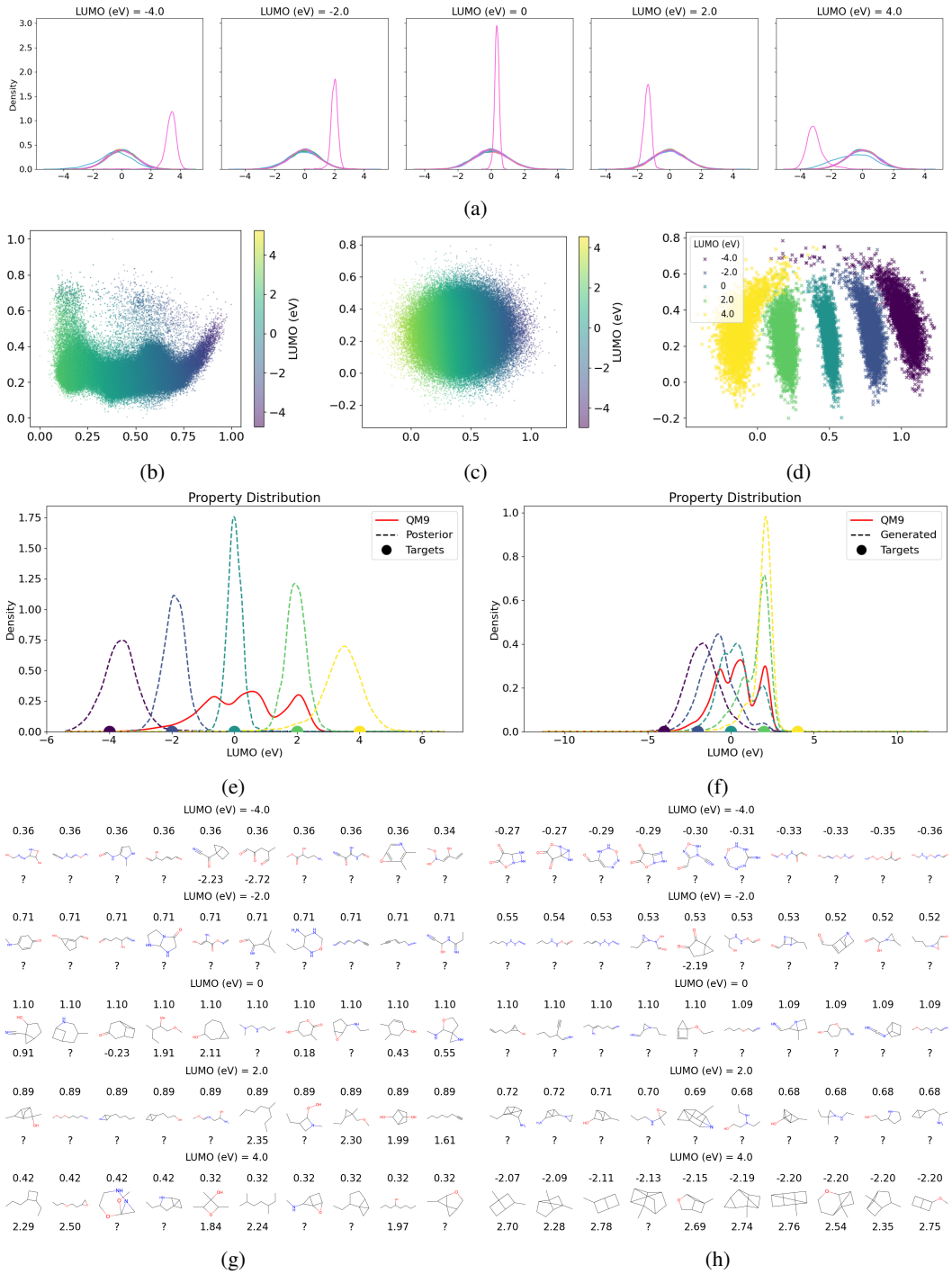


Figure 5: Inverse design using the LUMO property with 5 different target values. a) KDE of all dimensions of the latent posterior distribution. b) PCA of the encoded dataset with ground truth properties. c) PCA of samples from the prior with predicted properties. d) PCA of the latent posterior distributions. e) Property distribution predicted from the posterior. f) Property distribution predicted from the re-encoded molecules sampled from the posterior. g) Best candidate molecules selected according to their latent posterior without considering the decoder. The value above the molecules is their log-likelihood of fulfilling the target property. The value below the molecules is the true property value of molecules present in the dataset. h) Best candidate molecules selected according to our procedure described in subsection 3.1.

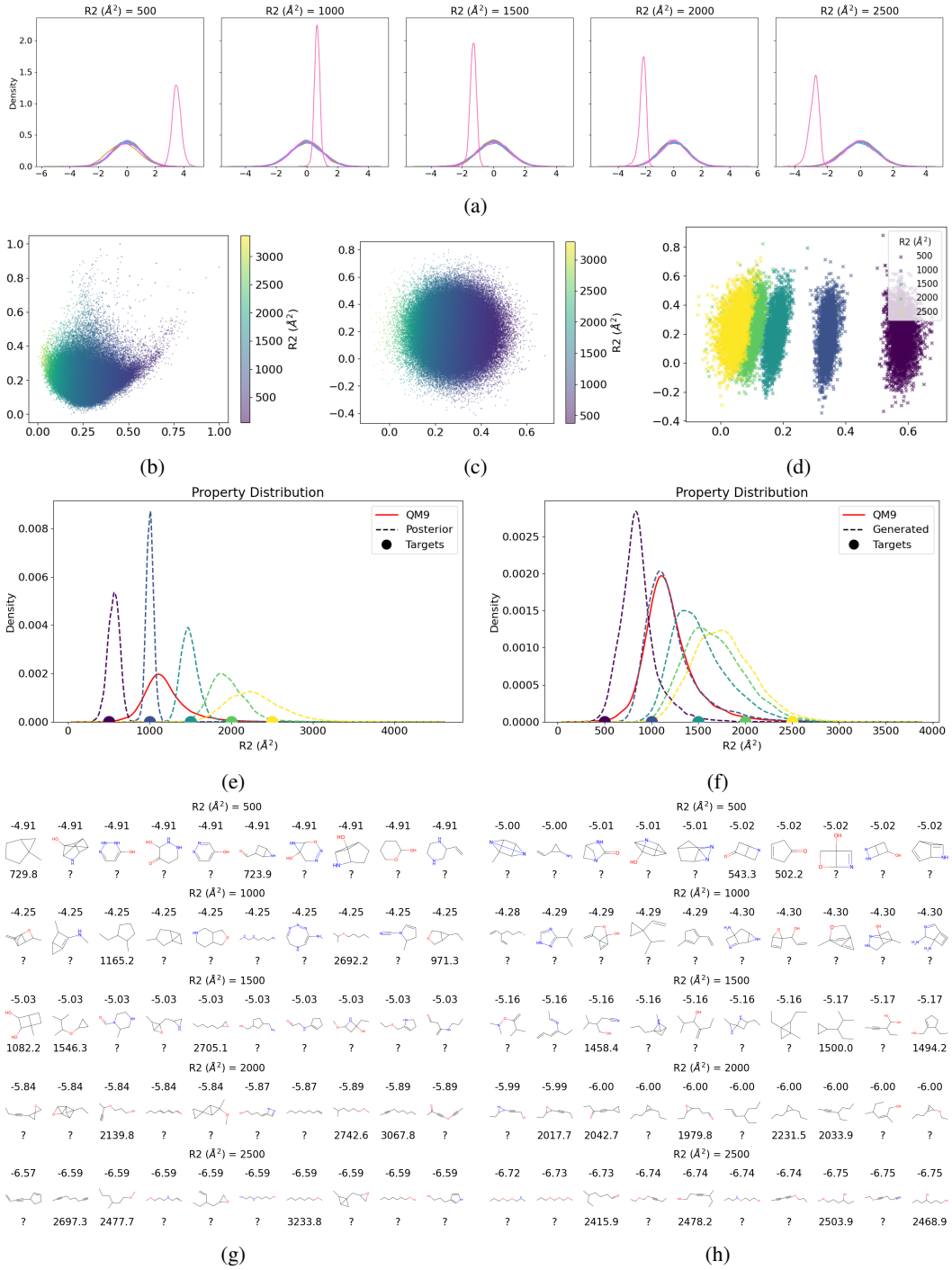


Figure 6: Inverse design using the R2 property with 5 different target values. a) KDE of all dimensions of the latent posterior distribution. b) PCA of the encoded dataset with ground truth properties. c) PCA of samples from the prior with predicted properties. d) PCA of the latent posterior distributions. e) Property distribution predicted from the posterior. f) Property distribution predicted from the re-encoded molecules sampled from the posterior. g) Best candidate molecules selected according to their latent posterior without considering the decoder. The value above the molecules is their log-likelihood of fulfilling the target property. The value below the molecules is the true property value of molecules present in the dataset. h) Best candidate molecules selected according to our procedure described in subsection 3.1.

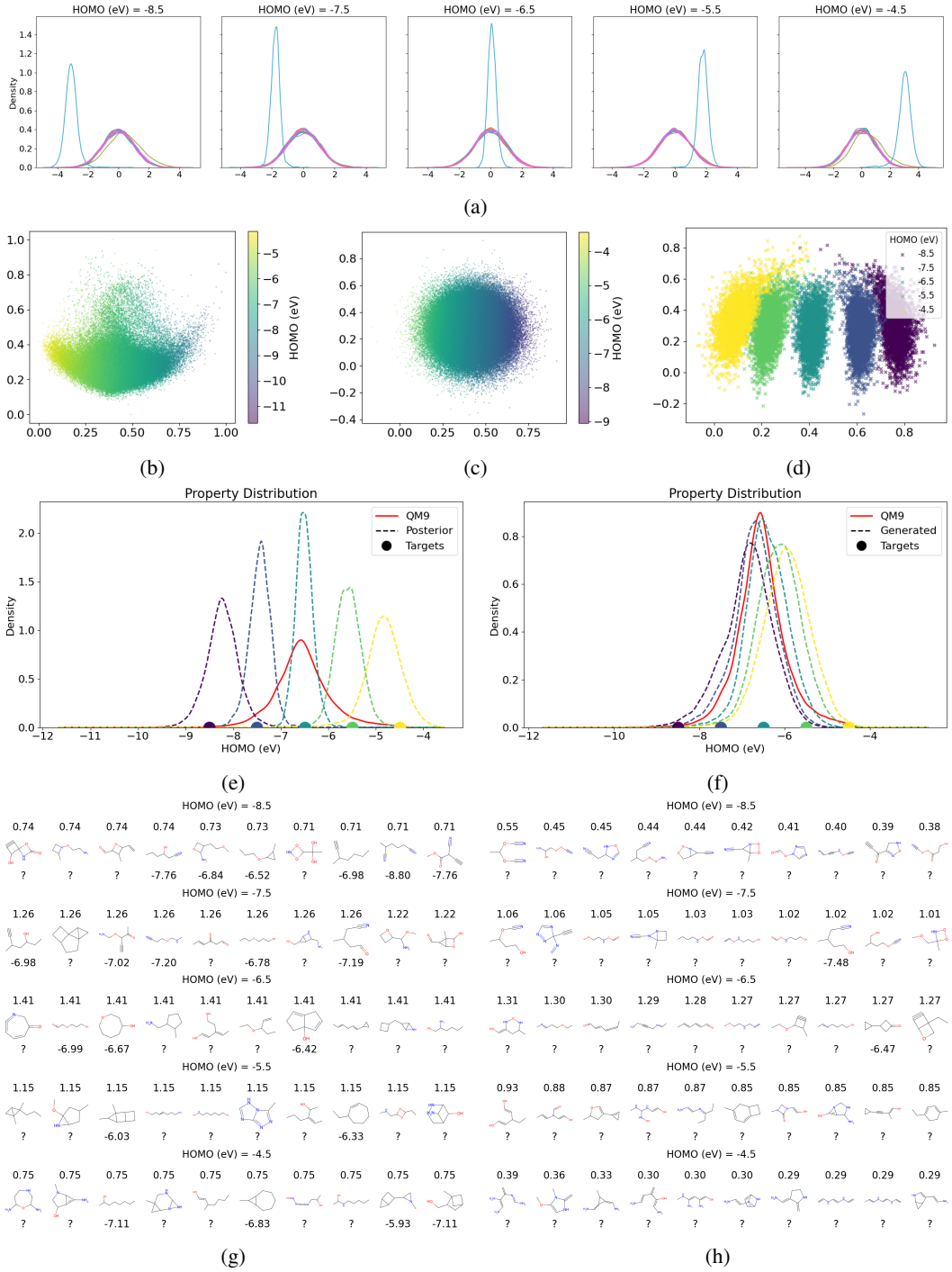


Figure 7: Inverse design using the HOMO property with 5 different target values. a) KDE of all dimensions of the latent posterior distribution. b) PCA of the encoded dataset with ground truth properties. c) PCA of samples from the prior with predicted properties. d) PCA of the latent posterior distributions. e) Property distribution predicted from the posterior. f) Property distribution predicted from the re-encoded molecules sampled from the posterior. g) Best candidate molecules selected according to their latent posterior without considering the decoder. The value above the molecules is their log-likelihood of fulfilling the target property. The value below the molecules is the true property value of molecules present in the dataset. h) Best candidate molecules selected according to our procedure described in subsection 3.1.

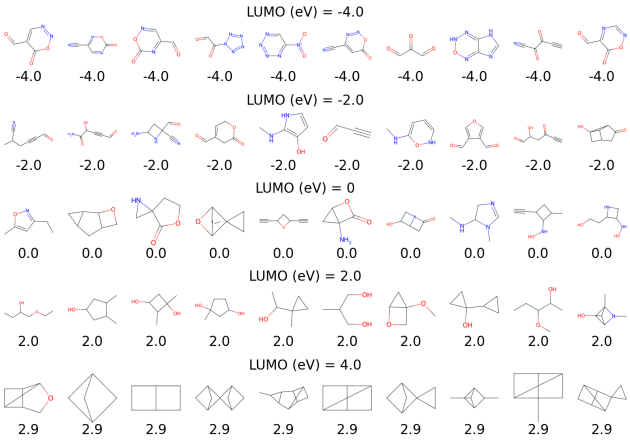


Figure 8: Molecules from the QM9 dataset with LUMO property values that are the closest to the objective values in our inverse design experiments.

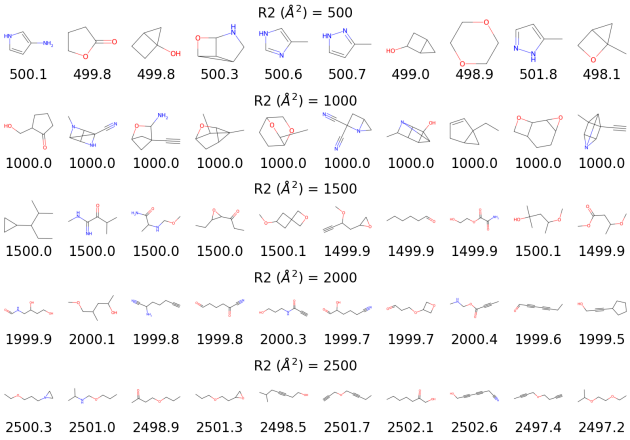


Figure 9: Molecules from the QM9 dataset with R2 property values that are the closest to the objective values in our inverse design experiments.

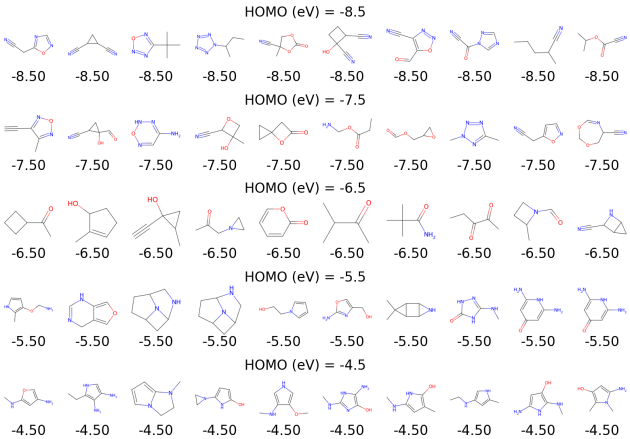
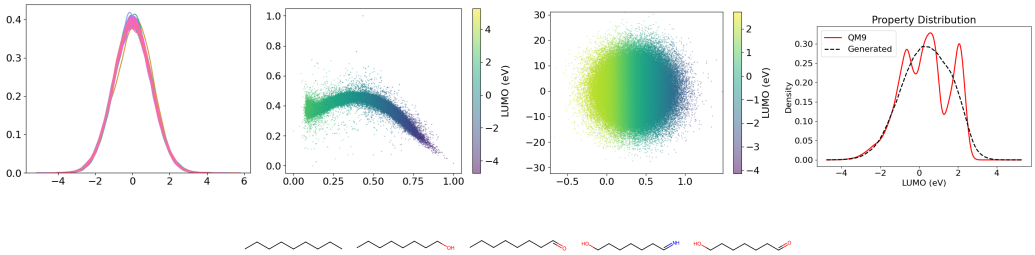
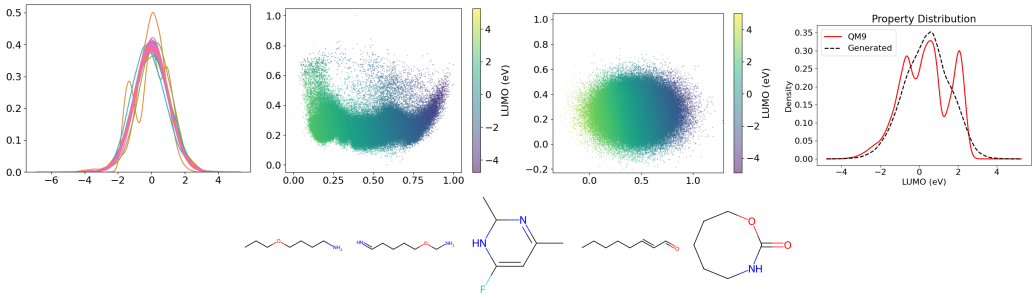


Figure 10: Molecules from the QM9 dataset with HOMO property values that are the closest to the objective values in our inverse design experiments.

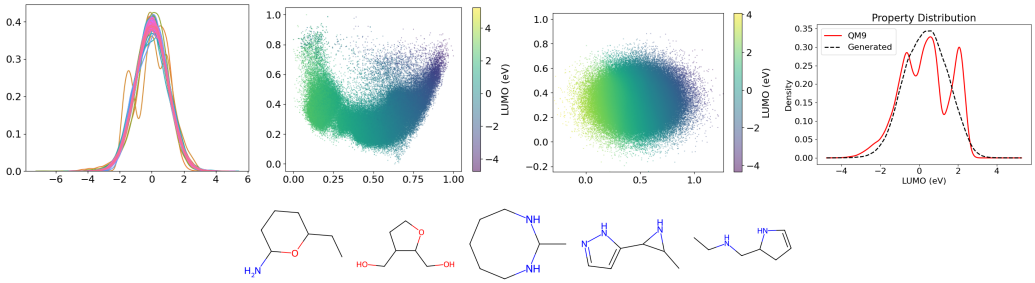
(a) $\beta = 1.0$



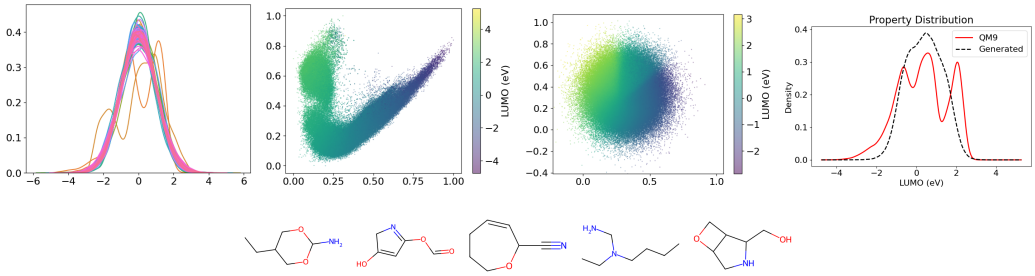
(b) $\beta = 0.1$



(c) $\beta = 0.05$



(d) $\beta = 0.02$



(e) $\beta = 0.01$

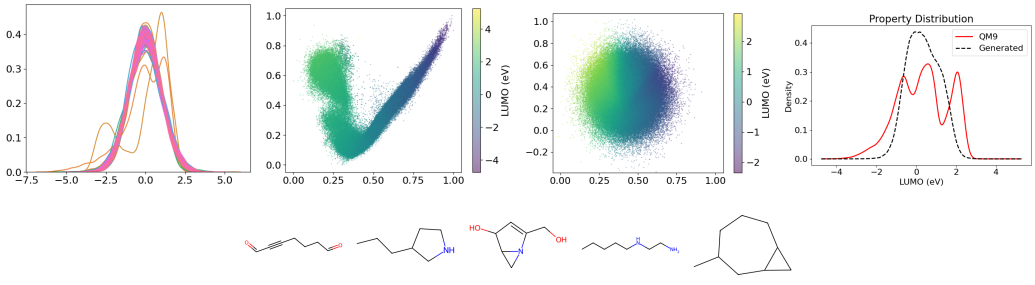
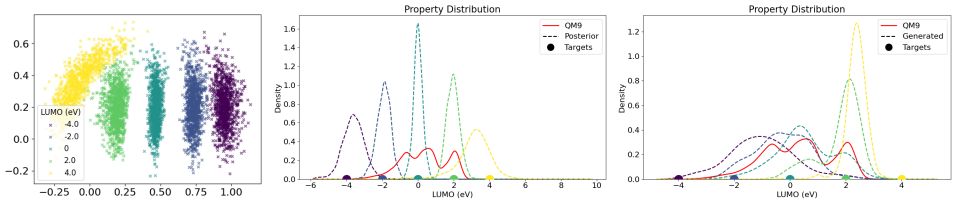
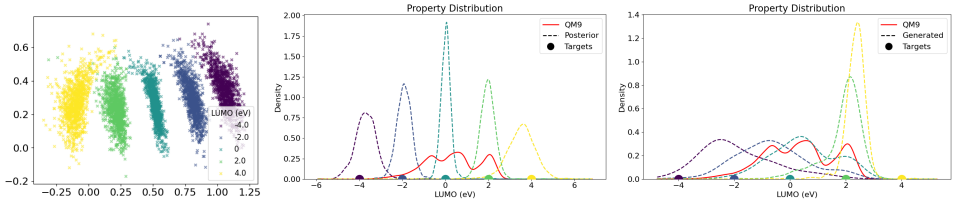


Figure 11: Impact of the KL divergence weight $\beta = \beta_1 = \beta_2$ on the latent space, the property distribution and the generated molecules. From left to right the figure shows a kernel density estimate of each latent dimension when encoding the validation set, a PCA plot of the encoded dataset, a PCA plot of samples from the prior with predicted properties and a KDE of the property distribution $p(y) = \int p_\theta(y|z)p(z)dz$.

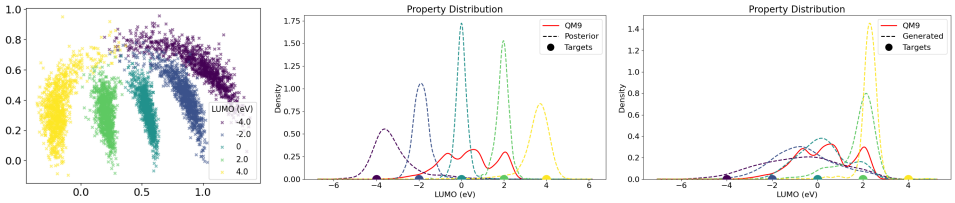
(a) $\beta = 0.2$



(b) $\beta = 0.1$



(c) $\beta = 0.05$



(d) $\beta = 0.02$

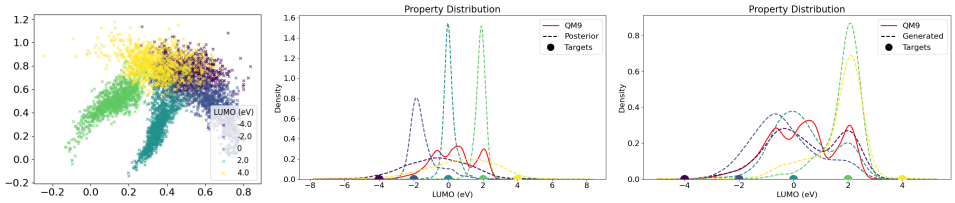
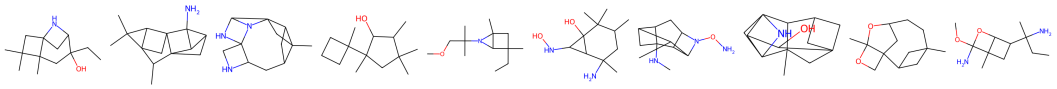
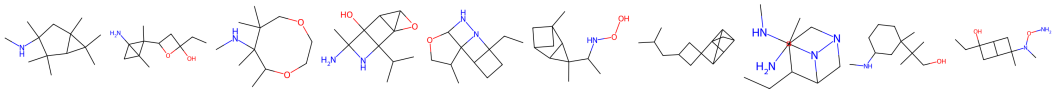


Figure 12: Samples from the latent posterior given 5 target property values with multiple models trained with different weights $\beta = \beta_1 = \beta_2$ of the KL divergence loss term.

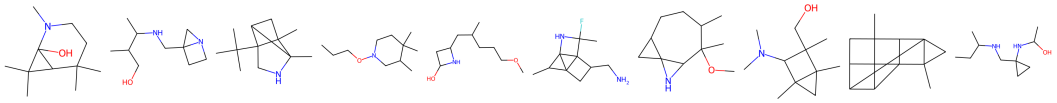
(a) $N = 15$



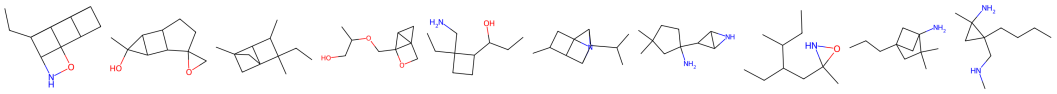
(b) $N = 14$



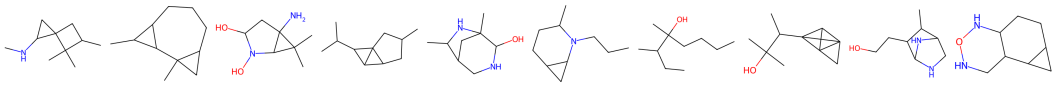
(c) $N = 13$



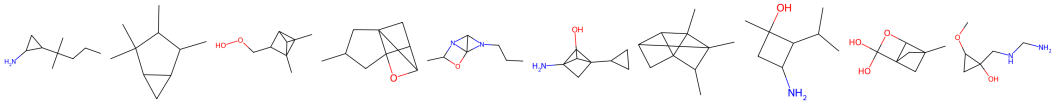
(d) $N = 12$



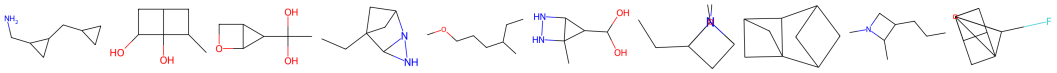
(e) $N = 11$



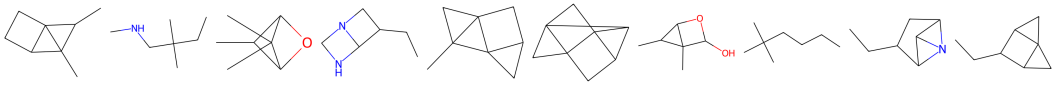
(f) $N = 10$



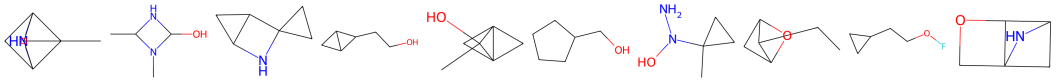
(g) $N = 9$



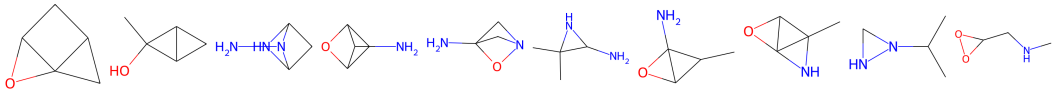
(h) $N = 8$



(i) $N = 7$



(j) $N = 6$



(k) $N = 5$

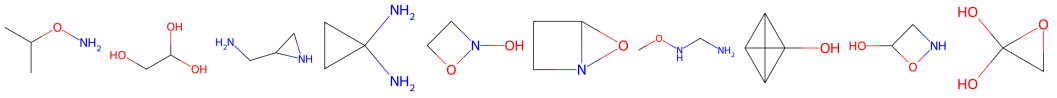


Figure 13: Molecules generated by the mixture model for various sizes N, including sizes beyond the maximum size ($N = 9$) in the training data.