Conditional generation of molecules from disentangled representations

Amina Mollaysa
University of Geneva
University of Applied Sciences Western Switzerland
maolaaisha.aminanmu@etu.unige.ch

Brooks Paige
University College London
Alan Turing Institute
b.paige@ucl.ac.uk

Alexandros Kalousis
University of Geneva
University of Applied Sciences Western Switzerland
alexandros.kalousis@unige.ch

Abstract

Though machine learning approaches have shown great success in estimating properties of small molecules, the inverse problem of generating molecules with desired properties remains challenging. This difficulty is in part because the set of molecules which have a given property is structurally very diverse. Treating this inverse problem as a generative modeling task, we draw upon work in disentangled representations to learn a variational autoencoder (VAE) which includes latent variables representing both property information and molecular structure. By explicitly setting the property to a desired value at generation time, can perform conditional molecule generation via a "style transfer" process. We improve the disentangling behaviour relative to baseline VAE models by introducing a novel regularization term which constrains the generated molecules to have the property provided to the generation network, no matter how the latent factor changes.

1 Introduction

Conditional molecule generation is far from being solved. The main challenge is the enormous and discrete nature of the molecules space, and the fact that molecule properties are highly sensitive to molecular structure [11]. Many existing approaches to conditional generation are two-step, either using a model or genetic algorithm to generate candidates [17] which are later filtered, or learning a continuous embedding of the discrete molecules and optimizing in a real-valued representation space [12, 8, 4, 14**?**]. The former is computationally expensive, the latter performs conditional generation only obliquely. Alternatively, reinforcement learning approaches [6, 21, 5] and graph-to-graph translation [9] yield generators that are tuned to maximize certain metrics. This improves upon basic virtual screening methods by providing more targeted generation; however, these models must be retrained for any new reward functions.

We propose a conditional generative model that produces candidate molecules targeting a desired property in a single step. This approach builds on work in structured deep generative models [10, 18], which aim to learn a disentangled representation that factors into distinct latent variables: one, the observed properties we want to control for; and two, latent factors that account for the remaining features which are either hard to annotate or irrelevant to the properties we wish to optimize. We derive a regularizer for supervised variational autoencoders which exploits property information that we provide as supervision, ensuring that produced molecules adhere to target properties they are conditioned on. We demonstrate the ability of our model to perform accurate conditional generation

Machine Learning for Molecules Workshop at NeurIPS 2020. https://ml4molecules.github.io

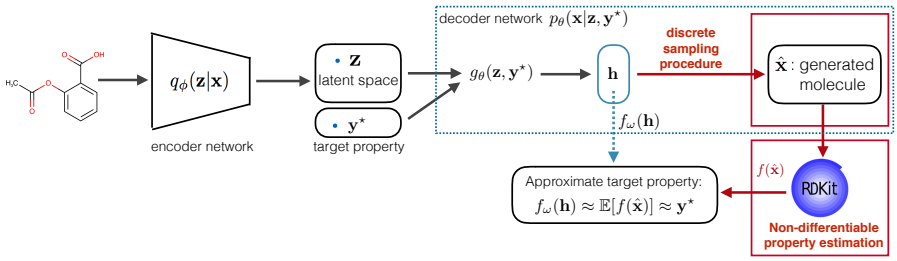

Figure 1: Setting and modeling pipeline for conditional generation of molecules, with supervision provided via an external property prediction oracle. Red lines correspond to non-differentiable components, the blue dashed line corresponds to the approximate property predictor.

as well as "style transfer" on molecules, where a latent representation for a single molecule can have its target properties perturbed relatively independently of its learnt structural characteristics, allowing direct and efficient generation of candidates for local optimization of molecules.

2 Background

Condtional generation and style transfer with supervised VAEs Suppose we are given a training set of pairs $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}, i = 1, \ldots, N$, where we refer $\mathbf{x}$ to molecules and $\mathbf{y}$ to some properties. To account for the diversity of molecules with a particular property, we model the conditional distribution with a latent variable $\mathbf{z}$, such that $p_\theta(\mathbf{x}|\mathbf{y}) = \int p_\theta(\mathbf{x}|\mathbf{z},\mathbf{y})p(\mathbf{z})d\mathbf{z}$. To do so we fit a joint generative model of the form $p_\theta(\mathbf{x},\mathbf{y},\mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{y},\mathbf{z})p(\mathbf{y})p(\mathbf{z})$, in which $\mathbf{y}$ and $\mathbf{z}$ are independent under the prior, To infer the latent variable $\mathbf{z}$ we introduce a multivariate normal posterior approximation, or encoding distribution, $q_\phi(\mathbf{z}|\mathbf{x})$ and optimize

$$\mathcal{L}_{ELBO}(\theta,\phi) = \sum_{i=1}^{N} \left\{ \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{y}_i,\mathbf{z}_i)] - D_{KL}(q_\phi(\mathbf{z}_i|\mathbf{x}_i)||p(\mathbf{z}_i)) \right\}. \tag{1}$$

This objective corresponds to learning a supervised VAE [10]. Once we learn the generative model $p_\theta(\mathbf{x}|\mathbf{y},\mathbf{z})$, we can perform both conditional generation and style transfer by setting $\mathbf{y}$ to target value and sampling $\mathbf{z}$ either from the prior or from a specific molecule's encoding distribution.

Maximizing this conditional ELBO in Eq (1) will likely yield good reconstructions, but for properties which only weakly inform the generative model there is nothing to enforce that the variable $\mathbf{y}$ actually take part in the generative process. Since the value $\mathbf{y}$ is something we know is a derived property of the molecule $\mathbf{x}$, it is completely possible for all information about $\mathbf{y}$ to also be encoded in the representation $\mathbf{z}$, in which case there is no guarantee that the learnt likelihood $p_\theta(\mathbf{x}|\mathbf{y},\mathbf{z})$ actually takes into account the value of $\mathbf{y}$. Furthermore, for style transfer to be possible $\mathbf{z}$ must be disentangled from $\mathbf{y}$, as we need $\mathbf{z}$ to be relatively stable when we modify the $\mathbf{y}$.

3 Conditional generation by disentangling

Constrained ELBO Assume we have access to some oracle function f (possibly non-differentiable) which for any given $\mathbf{x}$ outputs a property estimate $\mathbf{y}$, e.g., RDKit [13]. Since for conditional generation our ultimate goal is to generate a molecule $\mathbf{x}$ for any given target property $\mathbf{y}_0$, which then actually has $f(\mathbf{x}) = \mathbf{y}_0$, we can reframe the problem by introducing hard constraints on the generated values, i.e. if restricting to values of $\mathbf{y}$ in the training set,

$$\max_{\theta,\phi} \mathcal{L}_{ELBO}(\theta,\phi) \quad \text{subject to } \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}|\mathbf{y}_i)}[\mathbb{I}[f(\mathbf{x}) = \mathbf{y}_i]] = 1, \quad \forall i = 1, \ldots, N$$

This is an unreasonably hard constraint, unlikely to be satisfied by any distribution other than one which simply places a point mass on the single $\mathbf{x}_i$ associated with $\mathbf{y}_i$. We can relax it by considering that the property space $\mathbf{y}$ is typically smooth, and reframe it as a soft penalty on the ELBO [15, 7]:

$$\mathcal{L}(\theta,\phi) = \mathcal{L}_{ELBO}(\theta,\phi) - \frac{\lambda_1}{2} \sum_{i=1}^{N} \mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y}_i)} \|f(\hat{\mathbf{x}}) - \mathbf{y}_i\|^2 \tag{2}$$

This constraint is expected to hold for any pair $(\mathbf{x}, \mathbf{y})$, not just those in the training data. We also show optimizing the relaxed constraint is equivalent to maximizing mutual information with the target $\mathbf{y}_i$ and generated molecule $\hat{\mathbf{x}}$ (appendix Section A.3).

Approximating the property predictor As it is shown in figure 1, f is often non-differentiable or CPU-bound, and $\hat{\mathbf{x}}$ are discrete samples from a categorical distribution. To enable gradient-based training on GPUs and avoid discrete sampling, we propose bypassing the discrete sampling step and learning a function f_ω that can map from a learned representation of the molecules $g_\theta(\mathbf{z}, \mathbf{y})$, which is the last hidden layer of the decoder network, directly to molecule property [7]. Ideally, f_ω would estimate the property distribution obtained by marginalizing out the discrete sampling step:

$$f_\omega(\mathbf{h} \equiv g_\theta(\mathbf{z}, \mathbf{y}_0)) \approx \mathbb{E}_{p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{y}_0)}[f(\mathbf{x})], \tag{3}$$

where $\mathbf{y}_0$ refers to an arbitrary input target property. With the approximation in Eq. (3), which can be used as a drop-in replacement for the non-differentiable penalty term in Eq. (2), we have

$$\mathbb{E}_{p_\theta(\hat{\mathbf{x}}|\mathbf{y}_i)} \|f(\hat{\mathbf{x}}) - \mathbf{y}_i\|^2 = \mathbb{E}_{p(\mathbf{z})} \left[\mathbb{E}_{p_\theta(\hat{\mathbf{x}}|\mathbf{y}_i, \mathbf{z})} \|f(\hat{\mathbf{x}}) - \mathbf{y}_i\|^2\right] \approx \mathbb{E}_{p(\mathbf{z})} \|f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i\|^2,$$

This encourages all the molecules that are generated from property $\mathbf{y}_i$ to have property close to $\mathbf{y}_i$, no matter how we vary $\mathbf{z}$. Therefore we refer it as disentangling regularizer:

$$\mathcal{L}_{disent}(\theta, \omega) = \frac{\lambda_1}{2} \sum_{i=1}^{N} \mathbb{E}_{p(\mathbf{z})} \|f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i\|^2 \tag{4}$$

yielding a candidate objective function

$$\mathcal{L}(\theta, \phi) \approx \mathcal{L}_{ELBO}(\theta, \phi) - \mathcal{L}_{disent}(\theta, \omega) \tag{5}$$

Learning the property estimator jointly with generative model While one could imagine attempting to learn f_ω jointly with ϕ, θ by direct optimization of Eq. (5), in practice this is very unstable, as values of $g_\theta(\mathbf{z}, \mathbf{y}_i)$ early in training may correspond to very poor generated molecules $\hat{\mathbf{x}}_i$ which may not have properties at all similar to $\mathbf{y}_i$. This can be sidestepped by training the property estimator jointly as part of an extended generative model on $[\mathbf{x}, \mathbf{y}]$. From Eq. 3, we note that the property estimator f_ω parameterizes a probability distribution $p_\omega(f(\mathbf{x})|\mathbf{z}, \mathbf{y}_0)$, where $\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{y}_0)$. With a Gaussian distribution over the error, we can consider

$$p_\omega(f(\mathbf{x})|\mathbf{z}, \mathbf{y}_0) = \mathcal{N}(f(\mathbf{x})|f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_0)), \lambda_2^{-1}\mathbf{I}) \tag{6}$$

for small, fixed λ_2. Therefore, we propose defining a new ELBO based on a joint autoencoder for $\{f(\mathbf{x}_i), \mathbf{y}_i)\}$, with a joint likelihood $p_{\{\theta,\omega\}}(\mathbf{x}_i, f(\mathbf{x}_i)|\mathbf{z}_i, \mathbf{y}_i) = p_\omega(f(\mathbf{x}_i)|\mathbf{z}_i, \mathbf{y}_i) p_\theta(\mathbf{x}_i|\mathbf{z}_i, \mathbf{y}_i)$. This yields a joint ELBO for the training set of

$$\begin{aligned}
\mathcal{L}_{ELBO}(\omega, \theta, \phi) &= \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)} \left[\log \frac{p_\omega(f(\mathbf{x}_i)|\mathbf{z}, \mathbf{y}_i) p_\theta(\mathbf{x}_i|\mathbf{z}, \mathbf{y}_i) p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right] \\
&= \mathcal{L}_{ELBO}(\theta, \phi) - \frac{\lambda_2}{2} \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)} \|f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i\|_2^2.
\end{aligned} \tag{7}$$

where we also see that $\mathcal{L}_{ELBO}(\omega, \theta, \phi) \leq \mathcal{L}_{ELBO}(\theta, \phi)$, allowing us to define an objective

$$\hat{\mathcal{L}}(\omega, \theta, \phi) = \mathcal{L}_{ELBO}(\omega, \theta, \phi) - \mathcal{L}_{disent}(\theta, \omega), \tag{8}$$

which is a lower bound on Eq. (5). Notice the two terms we have added to the original ELBO are quite similar, differing only in choice of distribution: for learning f_ω, we wish to use values of $\mathbf{z}$ simulated form the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$, whereas for enforcing a constraint across all possible generations we simulate $\mathbf{z}$ from the prior $p(\mathbf{z})$. However, during training, we instead sample from the marginal posterior over $\mathbf{z}$ by mirroring the style transfer process: for a target $\mathbf{y}_i$, we sample $\mathbf{z}$ from an encoding of an actual data point $\mathbf{x}_j$ where $i \neq j$. See detail in appendix A.5.

Model	QM9 Reconstruction %	QM9 Valid%	QM9 Unique %	QM9 Novel %	ZINC Reconstruction %	ZINC Valid%	ZINC Unique %	ZINC Novel %
CVAE [4]	3.61	10.3	-	90.0	44.6	0.70	-	100
GVAE [12]	96.00	60.20	-	80.90	53.70	7.20	-	100
SD-VAE [3]	97.84	**98.40**	99.28	91.97	76.20	**43.50**	-	-
Sup-VAE-1-GRU	97.53	93.66	91.30	92.05	74.12	32.84	94.61	100
CGD-VAE-1-GRU	99.27	95.61	93.65	87.87	**88.64**	29.00	**99.24**	100
Sup-VAE-3-GRU	97.81	97.90	95.09	89.47	82.40	36.16	86.26	100
CGD-VAE-3-GRU	**99.31**	97,80	**98.77**	**96.21**	81.80	37.78	98.75	100

Table 1: Reconstruction performance and generation quality (Valid, Unique, Novel).

	Model	$\mathbf{z} \sim \hat{q}_\sigma(\mathbf{z})$	$\mathbf{z} \sim q(\mathbf{z}\|\mathbf{x})$
QM9	Sup-VAE-1-GRU	0.5420	0.2526
	CGD-VAE-1-GRU	0.7185	0.5005
	Sup-VAE-3-GRU	0.6958	0.4204
	CGD-VAE-3-GRU	0.7414	0.4715
ZINC	Sup-VAE-1-GRU	0.2301	0.0481
	CGD-VAE-1-GRU	0.3877	0.0880
	Sup-VAE-3-GRU	0.3514	0.1808
	CGD-VAE-3-GRU	0.3966	0.1559

Table 2: Correlation between the desired input property and the obtained property .

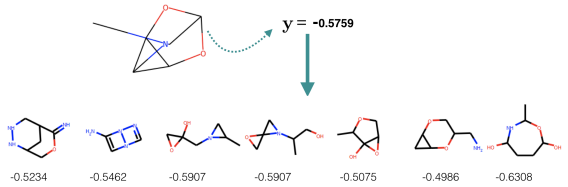

Table 3: Conditional generation given the desired logP=-0.5759, row molecules have a logP within a 15% range of the desired one.

4 Experiments

We experiment with the QM9 [16] and ZINC 250k [19] datasets. Our goal here is two-fold: we would like to understand (1) whether a supervised variational autoencoder is capable of learning suitable conditional distributions over molecules, and (2) to what extent this task is assisted by the additional regularization term corresponding to the soft constraint.

Experimental setting is kept the same as [3] with the only difference that our decoder takes as input the concatenation of $\mathbf{y}, \mathbf{z}$. We used logP [20] as the property to condition on. We give the details of the architecture in the appendix section A.4. We evaluate the reconstruction accuracy and the quality of the molecules generated by our method, which we denote by CGD-VAE (conditional generation with disentangling) and compare against CVAE [4], GVAE [12], and SD-VAE [3]. We explore its conditional generation performance in two settings: controlling only the property value and controlling both the property value and the molecule structure to what can be seen as style transfer. We also implemented supervised VAE versions of SD-VAE(Sup-VAE-X-GRU, X $\in \{1, 3\}$, denotes GRU layers) which can do conditional generation. For conditional generation, instead of marginalizing over the prior $p(z)$, we use $\hat{q}_\sigma(\mathbf{z})$ which is an approximation of the aggregated posterior.

In table 1, we report reconstruction performance and generation performance in terms of the percentage of valid, unique and novel molecules. We can see that our model has a better reconstruction performance compared to the baselines while in some cases generating slightly less valid molecules. To quantify the quality of the conditional generation and style transfer, we measure the correlation between the property value we obtain by the conditional generation and the property value on which we conditioned the generation. In Table 2, the two columns correspond to conditional generation and style transfer performance respectively. We randomly sample 1000 $\mathbf{y}$ values from the test set and 1000 $\mathbf{z}$ values from the approximate learned prior $\hat{q}_\sigma(\mathbf{z})$ or from learned posterior $q(\mathbf{z}|\mathbf{x})$. We decode each pair, obtain $\hat{\mathbf{x}} \sim p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})$, and then measure the correlation of the original $\mathbf{y}$ with the $\hat{\mathbf{y}}$ of generated $\hat{\mathbf{x}}$. As we can see our method has a considerably higher correlation score than Sup-VAE. Conditional generation seems considerably harder for the ZINC dataset for all methods.

In Figure 3, we visualize the conditional generation results where we randomly sample from the test set some molecule and obtain its property value $\mathbf{y}_0$. We then draw 50 random samples $\mathbf{z}_i$ from $\hat{q}_\sigma(\mathbf{z})$ and decode the $[\mathbf{z}_i, \mathbf{y}_0]$ vectors. Among the generated 46 valid molecules, we display those that have a property value $\mathbf{y}_i$ within a 15% range from the $\mathbf{y}_0$ property value. As we can see we get molecules that are structurally very different from the original one yet they have similar logP value. Visualization of style transfer and molecule property optimization are in Appendix figs. 2 through 5.

5 Conclusion

We presented a single-step approach for the conditional generation of molecules with desired properties. Our model allows also to condition generation on a prototype molecule with a desired high-level structure. We found that training the deep generative models conditional on target properties, following a supervised VAE approach, does not appreciably harm the quality of the unconditional generative model as measured by validity, novelty, and uniqueness of samples. Furthermore, we see that the additional act of regularizing the output using an approximate property predictor helps improve both reconstruction accuracy and property correlations in most combinations of tasks and datasets, particularly for the smaller QM9 dataset and for smaller models with fewer RNN layers.

References

[1] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016.

[2] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[3] H. Dai, Y. Tian, B. Dai, S. Skiena, and L. Song. Syntax-directed variational autoencoder for structured data. *arXiv preprint arXiv:1802.08786*, 2018.

[4] R. Gómez-Bombarelli, D. K. Duvenaud, J. M. Hernández-Lobato, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *CoRR*, abs/1610.02415, 2016.

[5] S. K. Gottipati, B. Sattarov, S. Niu, Y. Pathak, H. Wei, S. Liu, K. M. Thomas, S. Blackburn, C. W. Coley, J. Tang, et al. Learning to navigate the synthetically accessible chemical space using reinforcement learning. *arXiv preprint arXiv:2004.12485*, 2020.

[6] G. L. Guimaraes, B. Sanchez-Lengeling, C. Outeiral, P. L. C. Farias, and A. Aspuru-Guzik. Objective-Reinforced generative adversarial networks (ORGAN) for sequence generation models. May 2017.

[7] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org, 2017.

[8] W. Jin, R. Barzilay, and T. Jaakkola. Junction tree variational autoencoder for molecular graph generation. *arXiv preprint arXiv:1802.04364*, 2018.

[9] W. Jin, K. Yang, R. Barzilay, and T. Jaakkola. Learning multimodal graph-to-graph translation for molecular optimization. *arXiv preprint arXiv:1812.01070*, 2018.

[10] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[11] P. Kirkpatrick and C. Ellis. Chemical space. *Nature*, 432(7019):823, 2004.

[12] M. J. Kusner, B. Paige, and J. M. Hernández-Lobato. Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*, 2017.

[13] G. Landrum. Rdkit: Open-source cheminformatics.

[14] Q. Liu, M. Allamanis, M. Brockschmidt, and A. Gaunt. Constrained graph variational autoencoders for molecule design. In *Advances in Neural Information Processing Systems 31*, pages 7806–7815. 2018.

[15] T. Ma, J. Chen, and C. Xiao. Constrained generation of semantically valid graphs via regularizing variational autoencoders. In *Advances in Neural Information Processing Systems 31*, pages 7113–7124. 2018.

[16] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1:140022, 2014.

[17] M. H. Segler, T. Kogej, C. Tyrchan, and M. P. Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1):120–131, 2017.

[18] N. Siddharth, B. Paige, J.-W. Van de Meent, A. Desmaison, N. Goodman, P. Kohli, F. Wood, and P. Torr. Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems*, pages 5925–5935, 2017.

[19] T. Sterling and J. J. Irwin. Zinc 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.

[20] S. A. Wildman and G. M. Crippen. Prediction of physicochemical parameters by atomic contributions. *Journal of chemical information and computer sciences*, 39(5):868–873, 1999.

[21] J. You, B. Liu, R. Ying, V. Pande, and J. Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In *Advances in Neural Information Processing Systems 31*, pages 6412–6422. 2018.

A Appendix

A.1 Style transfer demo

To visualise the *style transfer* behavior of our model we randomly sample two molecules $\mathbf{x}_A, \mathbf{x}_B$ from the test set. We then sample $\mathbf{z}_A$ from the learned posterior $q_\phi(\mathbf{z}|\mathbf{x}_A)$. We subsequently decode $[\mathbf{z}_A, \mathbf{y}_B]$, $\mathbf{y}_B$ is the property of $\mathbf{x}_B$, and get a new molecule $\hat{\mathbf{x}}_{AB}$. Ideally, the obtained molecule $\hat{\mathbf{x}}_{AB}$ should have a property value (logP) close to the target $\mathbf{y}_B$ and be similar to $\mathbf{x}_A$. In Figure 4 we give one such example. To put the results into context, in Figure 3 we give the results of a virtual screening method, where we select from the full dataset five molecules which are structurally similar to $\mathbf{x}_A$ and have logP values close to $\mathbf{y}_B$. Our model generates novel molecules.

We further explore the style transfer and visualize how our model covers the combined space of molecule structure and properties. We sample nine molecules from the QM9 test set, and get their $\mathbf{z}$ encodings. For each such encoding we decode the vectors $[\mathbf{z}, y], y \in [-4.9, 4.9]$, with the y (logP) interval sampled at 11 points. We give in Figure 2 the resulting valid molecules, each column corresponding to one of the nine original molecules (surrounded by a dotted rectangle), and their decodings with different logP values. For each original molecule we order the generated molecules along the y axis according to the property that they actually exhibit. The x-axis is ordered based on the original molecules' logP values. As we can see not all $(\mathbf{z}, y)$ combinations produce a result. These holes can be explained either by the physical infeasibility of the combination and/or a limitation of the learned model.

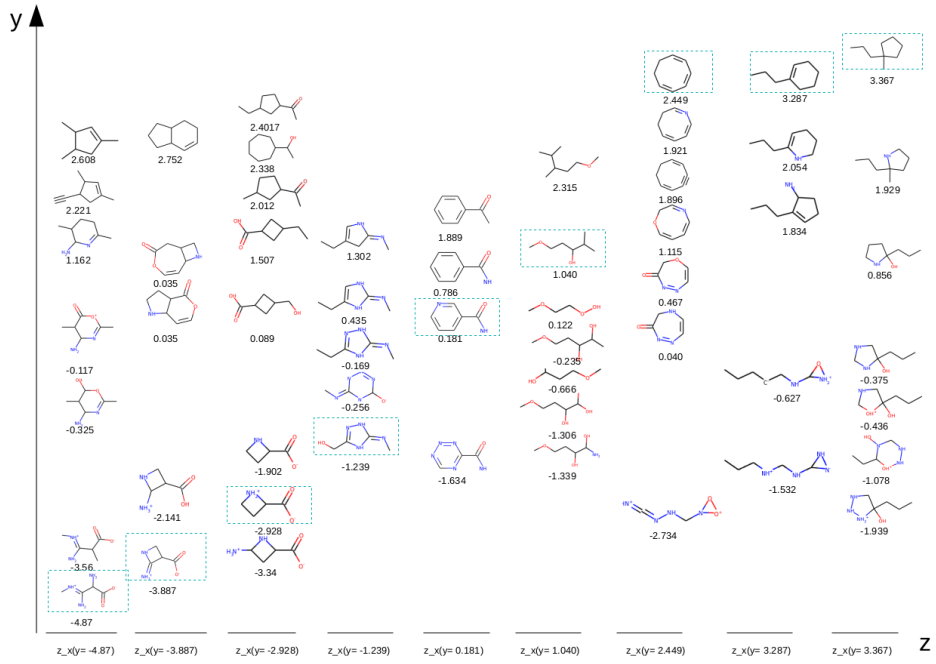

Figure 2: Style transfer. The $\mathbf{z}$ of the nine real molecules placed in the x-axis is combined with 11 $\mathbf{y}$ property values, sampled in [-4.9, 4.9], the resulting pair is decoded to a molecule.

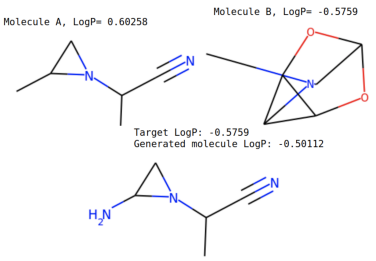

Figure 4: Style transfer

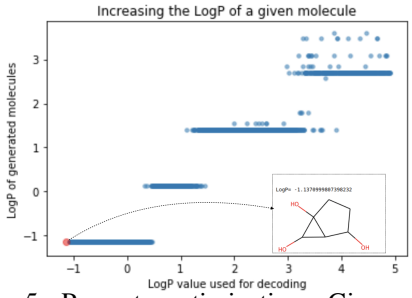

Figure 5: Property optimization. Given a start molecule (red), we combine its $\mathbf{z}$ with 1000 logP values and decode (blue)

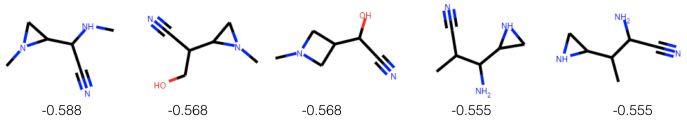

Figure 3: Molecules selected with virtual screening over the full dataset in a manner that they are structurally similar to the A molecule of figure 4 while they have a logP value which is close to the logP value of the B molecule also of figure 4

A.2 Molecule property optimization

We can use conditional generation to control in a fine manner the value of the desired property, to what can be seen as direct *property optimization*. We visualize the level of control we have on an experiment with a single molecule (with logP is -1.137), which we randomly sample from the test set. We obtain its $\mathbf{z}$ encoding and perform generations with increased logP taking values in 1000 point grid in $[-1.137, 4.9]$. We then decode $[\mathbf{z}, \mathbf{y}_i]$ and compute the logP value of the generated molecules. Among the 1000 generated molecules only 19 are unique. We get an increase of logP of a very discrete nature, Figure 5. As already discussed not all combinations of structure and properties are possible.

A.3 The regulariser and its relation to the mutual information maximization

The soft constraint in the loss 2, in fact, is equivalent to a simple maximizing mutual information formulation between generated molecules $\hat{\mathbf{x}}$ and the target property $\mathbf{y}$ provided to the generator. Assume the true conditional distribution is $\tilde{p}(\mathbf{y}|\mathbf{x})$:

$$\begin{aligned}\mathbf{I}(\mathbf{y}; \hat{\mathbf{x}}) &= H(\mathbf{y}) - H(\mathbf{y}|\hat{\mathbf{x}}) \\ &= H(\mathbf{y}) + \mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y})}\mathbb{E}_{\mathbf{y}'\sim\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})}[\log \tilde{p}(\mathbf{y}'|\hat{\mathbf{x}})] \end{aligned} \tag{9}$$

We do not know the true distribution $\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})$. However, RDKit provides a molecule property estimator f which, if we assume a Gaussian distribution over error, gives us $p(\mathbf{y}|\hat{\mathbf{x}}) = \mathcal{N}(\mathbf{y}|f(\hat{\mathbf{x}}), \lambda_1^{-1}\mathbf{I})$. We have:

$$\begin{aligned}\mathbf{I}(\mathbf{y}; \hat{\mathbf{x}}) &= H(\mathbf{y}) + \mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y})}\mathbb{E}_{\mathbf{y}'\sim\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})}[\log \frac{\tilde{p}(\mathbf{y}'|\hat{\mathbf{x}})}{p(\mathbf{y}'|\hat{\mathbf{x}})} p(\mathbf{y}'|\hat{\mathbf{x}})] \\ &= H(\mathbf{y}) + \mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y})}[D_{kl}(\tilde{p}(\mathbf{y}'|\hat{\mathbf{x}})||p(\mathbf{y}'|\hat{\mathbf{x}})) \\ &+ \mathbb{E}_{\mathbf{y}'\sim\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})} \log p(\mathbf{y}'|\hat{\mathbf{x}})] \\ &\geq H(\mathbf{y}) + \mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y})}[\mathbb{E}_{\mathbf{y}'\sim\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})} \log p(\mathbf{y}'|\hat{\mathbf{x}})] \end{aligned} \tag{10}$$

By following the Lemma that is been proven in [1]:

Lemma A.1 *For random variables* $\mathbf{X}$, $\mathbf{Y}$ *and function* $f(\mathbf{x}, \mathbf{y})$ *under suitable regularity conditions:*

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{X},\mathbf{y}\sim\mathbf{Y}|\mathbf{x}}[f(\mathbf{x}, \mathbf{y})] = \mathbb{E}_{\mathbf{x}\sim\mathbf{X},\mathbf{y}\sim\mathbf{Y}|\mathbf{x},\mathbf{x}'\sim\mathbf{X}|\mathbf{y}}[f(\mathbf{x}', \mathbf{y})] \tag{11}$$

we have:

$$\begin{aligned}&\mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y})}[\mathbb{E}_{\mathbf{y}'\sim\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})}\log p(\mathbf{y}'|\hat{\mathbf{x}})]\\&\qquad=\mathbb{E}_{\mathbf{y}\sim p(\mathbf{y}),\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y})}[\log p(\mathbf{y}|\hat{\mathbf{x}})]\end{aligned}\tag{12}$$

This means, the mutual information is lower bounded by:

$$\mathbf{I}(\mathbf{y};\hat{\mathbf{x}})\geq H(\mathbf{y})-\frac{\lambda_1}{2}\sum_{i=1}^{N}\mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y}_i)}\|f(\hat{\mathbf{x}})-\mathbf{y}_i\|^2\tag{13}$$

$H(\mathbf{y})$ is constant, therefore, minimizing our regularizer $\sum_{i=1}^{N}\mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta(\mathbf{x}|\mathbf{y}_i)}\|f(\hat{\mathbf{x}})-\mathbf{y}_i\|^2$ is equivalent to maximizing $\mathbf{I}(\mathbf{y};\hat{\mathbf{x}})$ under the assumption that $p(\mathbf{y}|\hat{\mathbf{x}})$ is close to $\tilde{p}(\mathbf{y}|\hat{\mathbf{x}})$, i.e., the RDKit oracle provides a good estimation of the molecules' true properties.

A.4 Architecture and Training procedure description

We represent molecules using the one-hot encoding of their SMILES production rules [12] and add a semantic constraint [3] on the decoder network to avoid generating syntactically correct but semantically invalid molecules. We use 80 production rules to describe molecules and set the maximum SMILES sequence length to 100 for the QM9 dataset and 278 for the Zinc dataset. We experiment with the logP property of the molecules [20]. We use the same encoder and decoder network structure as [3] with the only difference that our decoder takes as input the concatenation of $\mathbf{y},\mathbf{z}$.

We evaluate the reconstruction accuracy and the quality of the molecules generated by our method, which we denote by CGD-VAE (conditional generation with disentangling) and compare against CVAE [4], GVAE [12], and SD-VAE [3]. We explore its conditional generation performance in two settings: controlling only the property value and controlling both the property value and the molecule structure to what can be seen as property transfer. We took the results of CVAE, GVAE from the literature. For SD-VAE we used the authors code with the default values to generate results for QM9 since these were not available for QM9. We also implemented supervised VAE versions of SD-VAE which we denote Sup-VAE-X-GRU (X $\in\{1,3\}$, denotes GRU layers) and which can do conditional generation.

We use the same encoder and decoder network structure as [3] with the only difference that our decoder takes as input the concatenation of $\mathbf{y},\mathbf{z}$. As GRU layers become computationally expensive when the sequences length increase, we also examined the model using less layer GRU. To be precise, the decoder in [3] takes the from of a dense hidden layer with ReLU activation followed by three layers GRU [2]. We tried two different settings of decoder: in the first setting, we feed the concatenation of $\mathbf{y},\mathbf{z}$ to dense layer then apply one layer GRU, in the second setting, to enhance the effect of $\mathbf{y}$ in the decoder, we feed $\mathbf{y}$ not only to the dense layer but also to each layer of GRU. Furthermore, we set the dimension of the latent representation to 56. For the oracle function estimator f_w, we use the same network architecture as the encoder (there is no parameter sharing) and we add one more fully connected layer followed by a Tanh transformation.

To speed up convergence, we initialize the f_w from a pre-training, where we train f_w on the well-trained (maximum 500 epochs with early stopping) supervised VAE's decoder output to predict the molecules property value. We also initialize the parameters of the encoder/decoder networks with the partially trained supervised VAE model (after 40 epochs for QM9, 100 epochs for ZINC). We do not update ω and ϕ,θ simultaneously, instead we do an alternate optimization. We update ω continuously for five epochs while holding ϕ,θ and do the same for updating ϕ,θ. We set the hyper-parameter value λ_1 to 50 and λ_2 to 1. The mini-batch size is set to 300 for QM9 and 100 for ZINC. We use ADAM optimizer with learning rate 0.0001 and pytorch lr-scheduler on the validation loss. The general training algorithm is described in below algorithm block1. In our experiment, to train f_ω, we skipped the second term in step 7, which means we only train f_ω on the training data but not the newly generated molecules obtained by permuting the property. The reason for this is that, during the training, we found that it is easy for the model to learn to reconstruct but hard to conditionally generate the molecules with given properties while we have no guidance of what the molecules should look like. Further more, some combination of $\mathbf{z}$ and $\mathbf{y}$ are physically not feasible. In this case, when the conditional generation is not good enough yet during the training, we end up fitting f_ω on the miss represented molecules representations and it makes the optimization harder.

A.5 Gradient estimation

As the regularizer $\mathcal{L}_{disent}(\theta, \omega)$ encourages disentangling by constraining the molecules generated from $\mathbf{y}_i$ to have property $\mathbf{y}_i$ no matter what value $\mathbf{z}$ takes, we found that it does not necessarily evaluate at meaningful values of $\mathbf{z}$ when sampled randomly from $p(\mathbf{z})$. This roughly corresponds to the notion that not all combinations of "style" and property are physically attainable; ideally for style transfer we would like the generated molecule to stay "close" in structure to the original molecule that we intended to modify. When estimating (gradients of) the soft constraint term $\mathcal{L}_{disent}(\theta, \omega)$, we found it advantageous to use samples of $\mathbf{z}$ which correspond to encodings of actual data points, as opposed to random samples from the prior. We approximate expectations with respect to $p(\mathbf{x})$ by looking at the so-called marginal posterior; we note that

$$p(\mathbf{z}) = \int p_\theta(\mathbf{z}|\mathbf{x}) p_\theta(\mathbf{x}) d\mathbf{x} \approx \frac{1}{N} \sum_j p_\theta(\mathbf{z}|\mathbf{x}_j) \approx \frac{1}{N} \sum_j q_\phi(\mathbf{z}|\mathbf{x}_j),$$

where the first approximation uses the empirical data distribution as an approximation to the model marginal $p_\theta(\mathbf{x})$, and the second uses our variational posterior approximation $q_\phi(\mathbf{z}|\mathbf{x})$. We define this quantity as $q(\mathbf{z}) = \frac{1}{N} \sum_j q_\phi(\mathbf{z}|\mathbf{x}_j)$, a mixture of Gaussians, which we can sample from by drawing random values from our dataset and then drawing from their encoding distributions.

When we use this in estimating gradients of the soft constraint, we can use samples from the same minibatch, exactly corresponding to a property transfer task. That is, for any particular y_i in the dataset, we can estimate

$$\mathbb{E}_{p(\mathbf{z})} \nabla_{\theta,\omega} \| f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i \|^2 \approx \mathbb{E}_{q(\mathbf{z}_j|\mathbf{x}_j)} \nabla_{\theta,\omega} \| f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i \|^2.$$

for any uniformly randomly sampled $j \neq i$. By sampling $\mathbf{z}_j$ from $q(\mathbf{z}_j|\mathbf{x}_j)$ where $j \neq i$, we make sure that all the label information decoder is receiving comes from the actual $\mathbf{y}_i$ that is feed to the decoder and $\mathbf{z}_j$ does not include any information about label. This can be evaluated easily by simply evaluating the penalty term of Eq. (7) twice per minibatch; once as in Eq. (7), and once to approximate $\mathcal{L}_{disent}(\theta, \omega)$ by permuting the properties in the minibatch to be assigned to incorrect molecules. We detail the training algorithm in the following algorithm block.

Algorithm 1 Training algorithm

1: Initialize $p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{y})$, $q_\phi(\mathbf{z}|\mathbf{x})$, f_w
2: **for** $i = 1, 2, \ldots, N$ (maximum epoch number) **do**
3: **for** $j = 1, 2, \ldots, L$ **do**
4: sample a minibatch $D = (\mathbf{X}, \mathbf{Y}) = \{\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^M$ of M samples, where $L \times M = N$,
5: Randomly permute the property set $\mathbf{Y}$ to obtain $\mathbf{Y}^{'}$ and define a label permuted mini batch $D^{'} = (\mathbf{X}, \mathbf{Y}^{'}) = \{\mathbf{x}_i, \mathbf{y}_i^{'}\}_{i=1}^M$,
6: Update generative model parameter θ with:

$$\theta^j = \theta^{j-1} - \alpha(-\nabla_\theta \mathcal{L}_{ELBO}(\theta, \phi, \omega) + \frac{\lambda_1}{2} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in D^{'}} \mathbb{E}_{q(\mathbf{z}|\mathbf{x}_i)} \nabla_\theta \| f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i \|^2)$$

7: Update recognition model parameter ϕ with: $\phi^j = \phi^{j-1} - \alpha(-\nabla_\phi \mathcal{L}_{ELBO}(\theta, \phi, \omega))$
8: Update property predictor model parameter ω with:

$$w^j = w^{j-1} - \alpha(\frac{\lambda_2}{2} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in D} \mathbb{E}_{q(\mathbf{z}|\mathbf{x}_i)} \nabla_\omega \| f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i \|^2 + \frac{\lambda_1}{2} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in D^{'}} \mathbb{E}_{q(\mathbf{z}|\mathbf{x}_i)} \nabla_\omega \| f_\omega(g_\theta(\mathbf{z}, \mathbf{y}_i)) - \mathbf{y}_i \|^2)$$