PluGeN: Multi-Label Conditional Generation from Pre-trained Models

Maciej Wołczyk[1*], **Magdalena Proszewska**[1*], **Łukasz Maziarka**[1],
Maciej Zieba[2,4], **Patryk Wielopolski**[2], **Rafał Kurczab**[3], **Marek Śmieja**[1†]
[1]Jagiellonian University
[2]Wroclaw University of Science and Technology
[3] Institute of Pharmacology PAS
[4] Tooploox

Abstract

Modern generative models achieve excellent quality in a variety of tasks including image or text generation and chemical molecule modeling. However, existing methods often lack the essential ability to generate examples with requested properties, such as the age of the person in the photo or the weight of the generated molecule. Incorporating such additional conditioning factors would require rebuilding the entire architecture and optimizing the parameters from scratch. Moreover, it is difficult to disentangle selected attributes so that to perform edits of only one attribute while leaving the others unchanged. To overcome these limitations we propose PluGeN (Plugin Generative Network), a simple yet effective generative technique that can be used as a plugin to pre-trained generative models. The idea behind our approach is to transform the entangled latent representation using a flow-based module into a multi-dimensional space where the values of each attribute are modeled as an independent one-dimensional distribution. In consequence, PluGeN can generate new samples with desired attributes as well as manipulate labeled attributes of existing examples. Due to the disentangling of the latent representation, we are even able to generate samples with rare or unseen combinations of attributes in the dataset, such as a young person with gray hair, men with make-up, or women with beards. We combined PluGeN with GAN and VAE models and applied it to conditional generation and manipulation of images and chemical molecule modeling. Experiments demonstrate that PluGeN preserves the quality of backbone models while adding the ability to control the values of labeled attributes. Implementation is available at https://github.com/gmum/plugen.

Introduction

Generative models such as GANs and variational autoencoders have achieved great results in recent years, especially in the domains of images (Brock, Donahue, and Simonyan 2018; Brown et al. 2020) and cheminformatics (Gómez-Bombarelli et al. 2018; Jin, Barzilay, and Jaakkola 2018). However, in many practical applications, we need to control the process of creating samples by enforcing particular features of generated objects. This would be required to regulate the biases present in the data, e.g. to assure that people of each ethnicity are properly represented in the generated set of face images. In numerous realistic problems, such as drug discovery, we want to find objects with desired properties, like molecules with a particular activity, non-toxicity, and solubility.

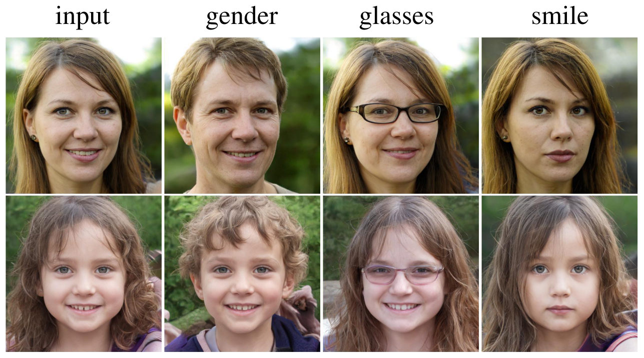
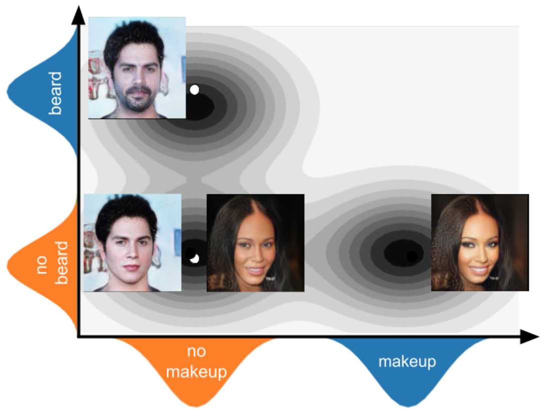


Figure 1: Attributes manipulation performed by PluGeN using the StyleGAN backbone.

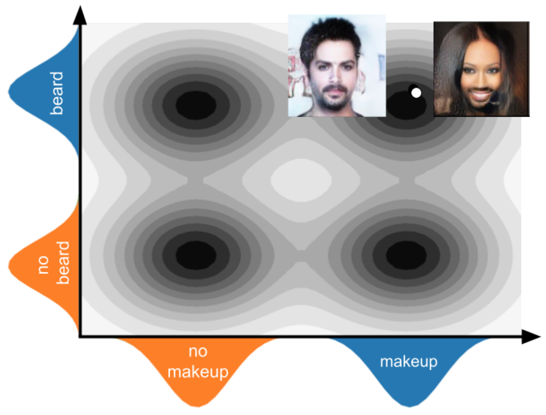
Designing the conditional variants of generative models that operate on multiple labels is a challenging problem due to intricate relations among the attributes. Practically, it means that some combinations of attributes (e.g. a woman with a beard) might be unobserved or rarely observed in the training data. In essence, the model should be able to go beyond the distribution of seen data and generate examples with combinations of attributes not encountered previously. One might approach this problem by building a new conditional generative model from the ground up or design a solution tailored for a specific existing unsupervised generative model. However, this introduces an additional effort when one wants to adapt it to a newly invented approach.

To tackle this problem while leveraging the power of existing techniques, we propose PluGeN (Plugin Generative Network), a simple yet effective generative technique that can be used as a plugin to various pre-trained generative models such as VAEs or GANs, see Figure 1 for demonstration. Making use of PluGeN, we can manipulate the attributes of input examples as well as generate new samples with desired features. When training the proposed module, we do not change the parameters of the base model and thus

*Equal contribution
†Corresponding author: marek.smieja@uj.edu.pl



(a) Factorization of true data distribution



(b) Probability distribution covered by PluGeN.

Figure 2: PluGeN factorizes true data distribution into components (marginal distributions) related to labeled attributes, see (a), and allows for describing unexplored regions of data (uncommon combinations of labels) by sampling from independent components, see (b). In the case illustrated here, PluGeN constructs pictures of men with make-up or women with beards, although such examples rarely (or never) appear in the training set.

we retain its generative and reconstructive abilities, which places our work in the emerging family of non-invasive network adaptation methods (Wołczyk et al. 2021; Rebuffi, Bilen, and Vedaldi 2017; Koperski et al. 2020; Kaya, Hong, and Dumitras 2019; Zhou et al. 2020).

Our idea is to find a mapping between the entangled latent representation of the backbone model and a disentangled space, where each dimension corresponds to a single, interpretable attribute of the image. By factorizing the true data distribution into independent components, we can sample from each component independently, which results in creating samples with arbitrary combinations of attributes, see Figure 2. In contrast to many previous works, which are constrained to the attributes combinations visible in the training set, PluGeN gives us full control of the generation process, being able to create uncommon combinations of attributes, such as a woman with a beard or a man with heavy make-up. Generating samples with unseen combinations of attributes can be viewed as extending the distribution of generative models to unexplored although reasonable regions of data space, which distinguishes our approach from existing solutions.

Extensive experiments performed on the domain of images and a dataset of chemical compounds demonstrate that PluGeN is a reusable plugin that can be applied to various architectures including GANs and VAEs. In contrast to the baselines, PluGeN can generate new samples as well as manipulate the properties of existing examples, being capable of creating uncommon combinations of attributes.

Our contributions are as follow:

- We propose a universal and reusable plugin for multi-label generation and manipulation that can be attached to various generative models and applied it to diverse domains, such as chemical molecule modeling.
- We introduce a novel way of modeling conditional distributions using invertible normalizing flows based on the latent space factorization.
- We experimentally demonstrate that PluGeN can produce samples with uncommon combinations of attributes going beyond the distribution of training data.

Related work

Conditional VAE (cVAE) is one of the first methods which includes additional information about the labeled attributes in a generative model (Kingma et al. 2014). Although this approach has been widely used in various areas ranging from image generation (Sohn, Lee, and Yan 2015; Yan et al. 2016; Klys, Snell, and Zemel 2018) to molecular design (Kang and Cho 2018), the independence of the latent vector from the attribute data is not assured, which negatively influences the generation quality. Conditional GAN (cGAN) is an alternative approach that gives results of significantly better quality (Mirza and Osindero 2014; Perarnau et al. 2016; He et al. 2019), but the model is more difficult to train (Kodali et al. 2017). cGAN works very well for generating new images and conditioning factors may take various forms (images, sketches, labels) (Park et al. 2019; Jo and Park 2019; Choi et al. 2020), but manipulating existing examples is more problematic because GAN models lack the encoder network (Tov et al. 2021). Fader Networks (Lample et al. 2017) combine features of both cVAE and cGAN, as they use encoder-decoder architecture, together with the discriminator, which predicts the image attributes from its latent vector returned from the encoder. As discussed in (Li et al. 2020), the training of Fader Networks is even more difficult than standard GANs, and disentanglement of attributes is not preserved. MSP (Li et al. 2020) is a recent auto-encoder based architecture with an additional projection matrix, which is responsible for disentangling the latent space and separating the attribute information from other characteristic information. In contrast to PluGeN, MSP cannot be used with pre-trained GANs and performs poorly at generating new images (it was designed for manipulating existing examples). CAGlow (Liu et al. 2019) is an adaptation of Glow (Kingma and Dhariwal

2018) to conditional image generation based on modeling a joint probabilistic density of an image and its conditions. Since CAGlow does not reduce data dimension, applying it to more complex data might be problematic.

While the above approaches focus on building conditional generative models from scratch, recent works often focus on manipulating the latent codes of pre-trained models. StyleFlow (Abdal et al. 2021) operates on the latent space of StyleGAN (Karras, Laine, and Aila 2019) using a conditional continuous flow module. Although the quality of generated images is impressive, the model has not been applied to other generative models than StyleGAN and domains other than images. Moreover, StyleFlow needs an additional classifier to compute the conditioning factor (labels) for images at test time. Competitive approaches to StyleGAN appear in (Gao et al. 2021; Tewari et al. 2020; Härkönen et al. 2020; Nitzan et al. 2020). InterFaceGAN (Shen et al. 2020) postulates that various properties of the facial semantics can be manipulated via linear models applied to the latent space of GANs. Hijack-GAN (Wang, Yu, and Fritz 2021) goes beyond linear models and designs a proxy model to traverse the latent space of GANs.

In disentanglement learning, we assume that the data has been generated from a fixed number of independent factors of underlying variation. The goal is then to find a transformation that unravels these factors so that a change in one dimension of the latent space corresponds to a change in one factor of variation while being relatively invariant to changes in other factors (Bengio, Courville, and Vincent 2013; Kim and Mnih 2018; Higgins et al. 2017; Brakel and Bengio 2017; Kumar, Sattigeri, and Balakrishnan 2017; Chen et al. 2019; Spurek et al. 2020; Dinh, Krueger, and Bengio 2014; Sorrenson, Rother, and Köthe 2020; Chen et al. 2016). As theoretically shown in (Locatello et al. 2019), the unsupervised learning of disentangled representations is fundamentally impossible without inductive biases on both the models and the data. In this paper, we solve a slightly different problem than typical disentanglement, as we aim to deliver an efficient plug-in model to a large variety of existing models in order to manipulate attributes without training the entire system. Creating compact add-ons for large models saves training time and energy consumption.

Plugin Generative Network

We propose a plugin generative network (PluGeN), which can be attached to pre-trained generative models and allows for direct manipulation of labeled attributes, see Figure 3 for the basic scheme of PluGeN. Making use of PluGeN we preserve all properties of the base model, such as generation quality and reconstruction in the case of auto-encoders, while adding new functionalities. In particular, we can:

- modify selected attributes of existing examples,
- generate new samples with desired labels.

In contrast to typical conditional generative models, PluGeN is capable of creating examples with rare or even unseen combinations of attributes, e.g. man with makeup.

Probabilistic model. PluGeN works in a multi-label setting, where every example $\mathbf{x} \in \mathcal{X}$ is associated with a K-dimensional vector of binary labels[1] $\mathbf{y} = (y_1, \dots, y_K) \in \{0,1\}^K$. We assume that there is a pre-trained generative model $\mathcal{G} : \mathcal{Z} \to \mathbb{R}^D$, where $\mathcal{Z} \subset \mathbb{R}^N$ is the latent space, which is usually heavily entangled. That is, although each latent code $\mathbf{z} \in \mathcal{Z}$ contains the information about the labels $\mathbf{y}$, there is no direct way to extract or modify it.

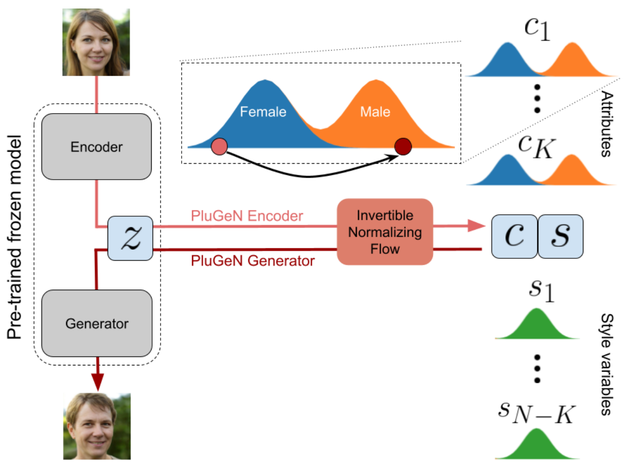


Figure 3: PluGeN maps the entangled latent space $\mathcal{Z}$ of pre-trained generative models using invertible normalizing flow into a separate space, where labeled attributes are modeled using independent 1-dimensional distributions. By manipulating label variables in this space, we fully control the generation process.

We want to map this entangled latent space $\mathcal{Z}$ into a separate latent space $\mathcal{D} \subset \mathbb{R}^N$ which encodes the values of each label y_k as a separate random variable C_k living in a single dimension of this space. Thus, by changing the value of C_k, going back to the entangled space $\mathcal{Z}$ and generating a sample, we can control the values of y_k. Since labeled attributes usually do not fully describe a given example, we consider additional $N-K$ random variables S_k, which are supposed to encode the information not included in the labels. We call $\mathbf{C} = (C_1, \dots, C_K)$ the label variables (or attributes) and $\mathbf{S} = (S_1, \dots, S_{N-K})$ the style variables.

Since we want to control the value of each attribute independently of any other factors, we assume the factorized form of the probability distribution of the random vector $(\mathbf{C}, \mathbf{S})$. More precisely, the conditional probability distribution of $(\mathbf{C}, \mathbf{S})$ given any condition $\mathbf{Y} = \mathbf{y}$ imposed on labeled attributes is of the form:

$$p_{\mathbf{C},\mathbf{S}|\mathbf{Y}=\mathbf{y}}(\mathbf{c},\mathbf{s}) = \prod_{i=1}^{K} p_{C_i|Y_i=y_i}(c_i) \cdot p_{\mathbf{S}}(\mathbf{s}), \tag{1}$$

for all $(\mathbf{c},\mathbf{s}) = (c_1, \dots, c_K, s_1, \dots, s_{N-K}) \in \mathbb{R}^N$. In other words, modifying $Y_i = y_i$ influences only the i-th factor C_i leaving other features unchanged.

Parametrization. To instantiate the above probabilistic model (1), we need to parametrize the conditional distribu-

[1]Our model can be extended to continuous values, which we describe in the supplementary materials due to page limit.

tion of C_i given $Y_i = y_i$ and the distribution of $\mathbf{S}$. Since we do not impose any constraints on style variables, we use standard Gaussian distribution for modeling density of $\mathbf{S}$:

$$p_{\mathbf{S}} = \mathcal{N}(0, I_{N-K}).$$

To provide the consistency with $p_{\mathbf{S}}$ and avoid potential problems with training our deep learning model using discrete distributions, we use the mixture of two Gaussians for modeling the presence of labels – each component corresponds to a potential value of the label (0 or 1). More precisely, the conditional distribution of C_i given $Y_i = y_i$ is parametrized by:

$$p_{C_i|Y_i=y_i} = \mathcal{N}(m_0, \sigma_0)^{(1-y_i)} \cdot \mathcal{N}(m_1, \sigma_1)^{y_i}, \qquad (2)$$

where $m_0, m_1, \sigma_0, \sigma_1$ are the user-defined parameters. If $y_i = 0$, then the latent factor C_i takes values close to m_0; otherwise we get values around m_1 (depending on the value of σ_0 and σ_1). To provide good separation between components, we put $m_0 = -1, m_1 = 1$; the selection of σ_0, σ_1 will be discussed is the supplementary materials.

Thanks to this continuous parametrization, we can smoothly interpolate between different labels, which would not be so easy using e.g. Gumbel softmax parametrization (Jang, Gu, and Poole 2016). In consequence, we can gradually change the intensity of certain labels, like smile or beard, even though such information was not available in a training set (see Figure 4 in the experimental section).

Training the model To establish a two-way mapping between entangled space $\mathcal{Z}$ and the disentangled space $\mathcal{D}$, we use an invertible normalizing flow (INF), $\mathcal{F} : \mathbb{R}^N \to \mathcal{Z}$. Let us recall that INF is a neural network, where the inverse mapping is given explicitly and the Jacobian determinant can be easily calculated (Dinh, Krueger, and Bengio 2014). Due to the invertibility of INF, we can transform latent codes $\mathbf{z} \in \mathcal{Z}$ to the prior distribution of INF, modify selected attributes, and map the resulting vector back to $\mathcal{Z}$. Moreover, INFs can be trained using log-likelihood loss, which is very appealing in generative modeling.

Summarizing, given a latent representation $\mathbf{z} \in \mathcal{Z}$ of a sample $\mathbf{x}$ with label $\mathbf{y}$, the loss function of PluGeN equals:

$$\begin{aligned} &-\log p_{\mathbf{Z}|\mathbf{Y}=\mathbf{y}}(\mathbf{z}) = \\ &\quad -\log\left(p_{\mathbf{C},\mathbf{S}|\mathbf{Y}=\mathbf{y}}(\mathbf{c},\mathbf{s}) \cdot \left|\det \frac{\partial \mathcal{F}^{-1}(\mathbf{z})}{\partial \mathbf{z}}\right|\right) = \\ &-\log\left(\prod_{i=1}^{K} p_{C_i|Y_i=y_i}(c_i) \cdot p_{\mathbf{S}}(\mathbf{s})\right) - \log\left|\det \frac{\partial \mathcal{F}^{-1}(\mathbf{z})}{\partial \mathbf{z}}\right| = \\ &-\sum_{i=1}^{K} \log p_{C_i|Y_i=y_i}(c_i) - \log p_{\mathbf{S}}(\mathbf{s}) - \log\left|\det \frac{\partial \mathcal{F}^{-1}(\mathbf{z})}{\partial \mathbf{z}}\right|, \end{aligned} \qquad (3)$$

where $(\mathbf{c}, \mathbf{s}) = \mathcal{F}^{-1}(\mathbf{z})$. In the training phase, we collect latent representations $\mathbf{z}$ of data points $\mathbf{x}$. Making use of labeled attributes $\mathbf{y}$ associated with every $\mathbf{x}$, we modify the weights of $\mathcal{F}$ so that to minimize the negative log-likelihood (3) using gradient descent. The weights of the base model $\mathcal{G}$ are kept frozen.

In contrast to many previous works (Abdal et al. 2021), PluGeN can be trained in a semi-supervised setting, where only partial information about labeled attributes is available (see supplementary materials for details).

Inference. We may use PluGeN to generate new samples with desired attributes as well as to manipulate attributes of input examples. In the first case, we generate a vector $(\mathbf{c}, \mathbf{s})$ from the conditional distribution $p_{\mathbf{C},\mathbf{S}|\mathbf{Y}=\mathbf{y}}$ with selected condition $\mathbf{y}$. To get the output sample, the vector $(\mathbf{c}, \mathbf{s})$ is transformed by the INF and the base generative network $\mathcal{G}$, which gives us the final output $\mathbf{x} = \mathcal{G}(\mathcal{F}(\mathbf{c}, \mathbf{s}))$.

In the second case, to manipulate the attributes of an existing example $\mathbf{x}$, we need to find its latent representation $\mathbf{z}$. If $\mathcal{G}$ is a decoder network of an autoencoder model, then $\mathbf{x}$ should be passed through the encoder network to obtain $\mathbf{z}$ (Li et al. 2020). If $\mathcal{G}$ is a GAN, then $\mathbf{z}$ can be found by minimizing the reconstruction error between $\mathbf{x}' = \mathcal{G}(\mathbf{z})$ and $\mathbf{x}$ using gradient descent for a frozen $\mathcal{G}$ (Abdal et al. 2021). In both cases, $\mathbf{z}$ is next processed by INF, which gives us its factorized representation $(\mathbf{c}, \mathbf{s}) = \mathcal{F}^{-1}(\mathbf{z})$. In this representation, we can modify any labeled variable c_i and map the resulting vector back through $\mathcal{F}$ and $\mathcal{G}$ as in the generative case.

Observe that PluGeN does not need to know what are the values of labeled attributes when it modifies attributes of existing examples. Given a latent representation $\mathbf{z}$, PluGeN maps it through $\mathcal{G}^{-1}$, which gives us the factorization into labeled and unlabeled attributes. In contrast, existing solutions based on conditional INF, e.g StyleFlow (Abdal et al. 2021), have to determine all labels before passing $\mathbf{z}$ through INF as they represent the conditioning factors. In consequence, these models involve additional classifiers for labeled attributes.

Experiments

To empirically evaluate the properties of PluGeN, we combine it with GAN and VAE architectures to manipulate attributes of image data. Moreover, we present a practical usecase of chemical molecule modeling using CharVAE. Due to the page limit, we included architecture details and additional results in the supplementary materials.

GAN backbone First, we consider the state-of-the-art StyleGAN architecture (Karras, Laine, and Aila 2019), which was trained on Flickr-Faces-HQ (FFHQ) containing 70 000 high-quality images of resolution 1024×1024. The Microsoft Face API was used to label 8 attributes in each image (gender, pitch, yaw, eyeglasses, age, facial hair, expression, and baldness).

PluGeN is instantiated using NICE flow model (Dinh, Krueger, and Bengio 2014) that operates on the latent vectors $\mathbf{w} \in \mathbb{R}^{512}$ sampled from the $\mathbf{W}$ space of the StyleGAN. As a baseline, we select StyleFlow (Abdal et al. 2021), which is currently one of the state-of-the-art models for controlling the generation process of StyleGAN. In contrast to PluGeN, StyleFlow uses the conditional continuous INF to operate on the latent codes of StyleGAN, where the conditioning factor corresponds to the labeled attributes. For evaluation, we modify one of 5 attributes[2] and verify the success of this operation using the prediction accuracy returned by Microsoft Face API. The quality of images is additionally assessed by calculating the standard Fréchet Inception Distance (FID) (Heusel et al. 2017).

Figure 1 (first page) and 4 present the effects of how PluGeN and StyleFlow manipulate images sampled by StyleGAN. It is evident that PluGeN can switch the labels to opposite values as well as gradually change their intensities. At the same time, the requested modifications do not influence the remaining attributes leaving them unchanged. One can observe that the results produced by StyleFlow are also acceptable, but the modification of the requested attribute implies the change of other attributes. For example, increasing the intensity of "baldness" changes the type of glasses, or turning the head into right makes the woman look straight.

The above qualitative evaluation is supported by the quantitative assessment presented in Table 1. As can be seen, StyleFlow obtains a better FID score, while PluGeN outperforms StyleFlow in terms of accuracy. Since FID compares the distribution of generated and real images, creating images with uncommon combinations of attributes that do not appear in a training set may be scored lower, which can explain the relation between accuracy and FID obtained by PluGeN and StyleFlow. In consequence, FID is not an adequate metric for measuring the quality of arbitrary image manipulations considered here, because it is too closely tied to the distribution of input images.

It is worth mentioning that PluGeN obtains these very good results using NICE model, which is the simplest type of INFs. In contrast, StyleFlow uses continuous INF, which is significantly more complex and requires using an ODE solver leading to unstable training. Moreover, to modify even a single attribute, StyleFlow needs to determine the values of all labels, since they represent the conditioning factors of INF. In consequence, every modification requires applying an auxiliary classifier to predict all image labels. The usage of PluGeN is significantly simpler, as subsequent coordinates in the latent space of INF correspond to the labeled attributes and they are automatically determined by PluGeN. Finally, our approach is less computationally expensive as we verified that, using the same hardware, PluGeN can be trained 3 times faster than StyleFlow and is around 100 times faster in inference.

Image manipulation on VAE backbone In the following experiment, we show that PluGeN can be combined with autoencoder models to effectively manipulate image attributes. We use CelebA database, where every image of the size 256×256 is annotated with 40 binary labels.

We compare PluGeN to MSP (Li et al. 2020), a strong baseline, which uses a specific loss for disentangling the latent space of VAE. Following the idea of StyleFlow, we also consider a conditional INF attached to the latent space of pre-trained VAE (referred to as cFlow), where conditioning factors correspond to the labeled attributes. The architecture of the base VAE and the evaluation protocol were taken from the original MSP paper. More precisely, for every input image, we manipulate the values of two attributes (we inspect 20 combinations in total). The success of the requested manipulation is verified using a multi-label ResNet-56 classifier trained on the original CelebA dataset.

Requested value	PluGeN	StyleFlow
female	**0.95**	**0.95**
male	**0.92**	0.87
no-glasses	**1.00**	0.99
glasses	**0.90**	0.70
not-bald	**1.00**	**1.00**
bald	0.53	**0.54**
no-facial-hair	**1.00**	**1.00**
facial-hair	**0.72**	0.65
no-smile	**0.99**	0.92
smile	0.96	**0.99**
Average Acc	**0.90**	0.86
Average FID	46.51	**32.59**

Table 1: Accuracy and FID scores of attributes modification using StyleGAN backbone.

The sample results presented in Figure 5 demonstrate that PluGeN attached to VAE produces high-quality images satisfying the constraints imposed on the labeled attributes. The quantitative comparison shown in Table 2 confirms that PluGeN is extremely efficient in creating uncommon combinations of attributes, while cFlow performs well only for the usual combinations. At the same time, the quality of images produced by PluGeN and MSP is better than in the case of cFlow. Although both PluGeN and MSP focus on disentangling the latent space of the base model, MSP has to be trained jointly with the base VAE model and it was designed only to autoencoder models. In contrast, PluGeN is a separate module, which can be attached to arbitrary pre-trained models. Due to the use of invertible neural networks, it preserves the reconstruction quality of the base model, while adding manipulation functionalities. In the following experiment, we show that PluGeN also performs well at generating entirely new images, which is not possible using MSP.

Image generation with VAE backbone In addition to manipulating the labeled attributes of existing images, PluGeN generates new examples with desired attributes. To verify this property, we use the same VAE architecture as before trained on CelebA dataset. The baselines include cFlow and two previously introduced methods for multi-label conditional generation[3]: cVAE (Yan et al. 2016) and Δ-GAN (Gan et al. 2017). We exclude MSP from the comparison because it cannot generate new images, but only manipulate the attributes of existing ones (see supplementary materials for a detailed explanation).

Figure 6 presents sample results of image generation with the specific conditions. In each row, we fix the style variables $\mathbf{s}$ and vary the label variables $\mathbf{c}$ in each column, generating

[2]The remaining 3 attributes (age, pitch, yaw) are continuous and it is more difficult to assess their modifications.

[3]For cVAE and Δ-GAN we use images of the size 64×64 following their implementations.

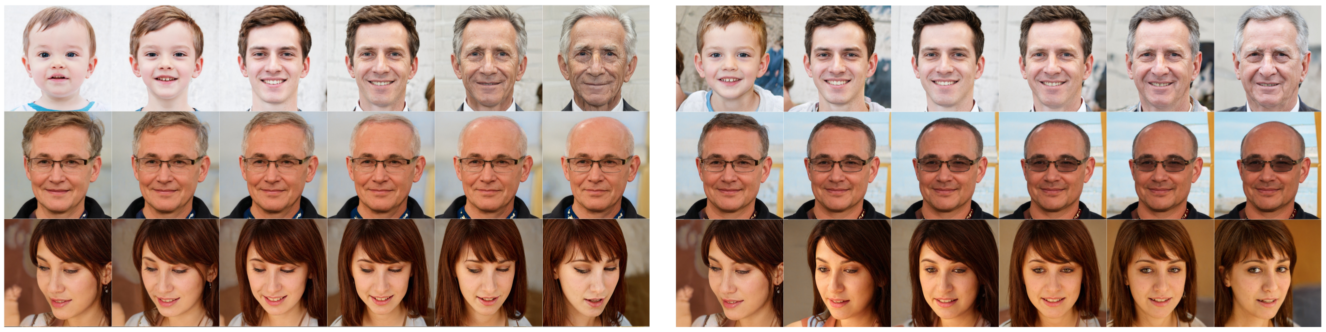

(a) PluGeN (b) StyleFlow

Figure 4: Gradual modification of attributes (age, baldness, and yaw, respectively) performed on the StyleGAN latent codes.

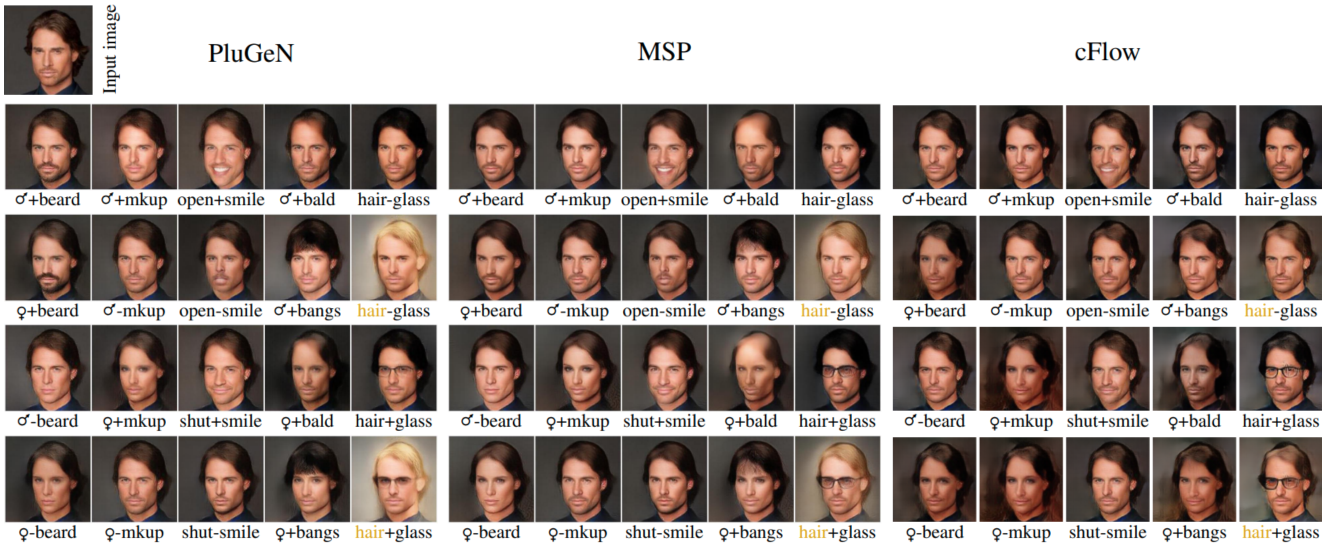

Input image
PluGeN
MSP
cFlow
♂+beard ♂+mkup open+smile ♂+bald hair-glass
♀+beard ♂-mkup open-smile ♂+bangs hair-glass
♂-beard ♀+mkup shut+smile ♀+bald hair+glass
♀-beard ♀-mkup shut-smile ♀+bangs hair+glass

Figure 5: Examples of image attribute manipulation using VAE backbone.

the same person but with different characteristics such as hair color, eyeglasses, etc. Although cVAE manages to modify the attributes, the quality of obtained samples is poor, while Δ-GAN falls completely out of distribution. PluGeN and cFlow generate images of similar quality, but only PluGeN is able to correctly manipulate the labeled attributes. The lower quality of generated images is caused by the poor generation abilities of VAE backbone, which does not work well with high dimensional images (see supplementary materials for a discussion). For this reason, it is especially notable that PluGeN can improve the generation performance of the backbone model in contrast to MSP.

Disentangling the attributes The attributes in the CelebA dataset are strongly correlated and at times even contradictory, e.g. attributes 'bald' and 'blond hair' cannot both be present at the same time. In this challenging task, we aim to disentangle the attribute space as much as it is possible to allow for generating examples with arbitrary combinations of attributes. For this purpose, we sample the conditional variables c_i independently, effectively ignoring the underlying correlations of attributes, and use them to generate images. Since the attributes in the CelebA dataset are often imbalanced (e.g. only in 6.5% of examples the person wears glasses), we calculate F1 and AUC scores for each attribute.

The quantitative analysis of the generated images presented in Table 3 confirms that PluGeN outperforms the rest of the methods with respect to classification scores. The overall metrics are quite low for all approaches, which is due to the difficulty of disentanglement mentioned above, as well as the inaccuracy of the ResNet attribute classifier. Deep learning models often fail when the correlations in the training data are broken, e.g. the classifier might use the presence of a beard to predict gender, thus introducing noise in the evaluation (Beery, Horn, and Perona 2018).

Chemical molecules modeling Finally, we present a practical use-case, in which we apply PluGeN to generate chemical molecules with the requested properties. As a backbone model, we use CharVAE (Gómez-Bombarelli et al. 2018), which is a type of recurrent network used for processing SMILES (Weininger 1988), a textual representation of molecules. It was trained on ZINC 250k database (Sterling and Irwin 2015) of commercially available chemical com-

Requested value	PluGeN	MSP	cFlow
male x beard	0.80	0.79	**0.85**
female x beard	**0.59**	0.33	0.31
male x no-beard	0.88	**0.92**	0.91
female x no-beard	0.85	0.82	**0.95**
male x makeup	**0.44**	0.43	0.29
male x no-makeup	0.72	0.92	**0.96**
female x makeup	0.42	0.41	**0.58**
female x no-makeup	0.55	0.40	**0.85**
smile x open-mouth	0.97	**0.99**	0.79
no-smile x open-mouth	0.79	**0.82**	0.77
smile x calm-mouth	0.84	**0.91**	0.72
no-smile x calm-mouth	0.96	0.97	**0.99**
male x bald	0.26	**0.41**	0.34
male x bangs	0.58	**0.74**	0.45
female x bald	0.19	0.13	**0.39**
female x bangs	0.52	0.49	**0.60**
no-glasses x black-hair	0.92	**0.93**	0.74
no-glasses x golden-hair	**0.92**	0.91	0.81
glasses x black-hair	0.76	**0.90**	0.58
glasses x golden-hair	0.75	**0.85**	0.61
Average Acc	0.69	**0.70**	0.67
Average FID	**28.07**	30.67	39.68

Table 2: Accuracy and FID scores of image manipulation performed on the VAE backbone.

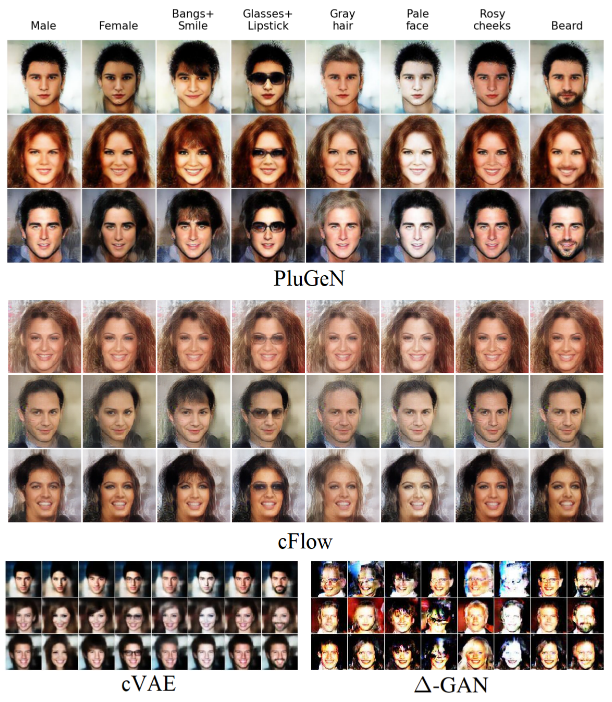


Figure 6: Examples of conditional generation using VAE backbone. Each row contains the same person (style variables) with modified attributes (label variables).

	PluGeN	cFlow	Δ-GAN	cVAE
F1	**0.44**	0.29	0.39	0.39
AUC	**0.76**	0.68	0.70	0.73

Table 3: Results of the independent conditional generation using VAE backbone.

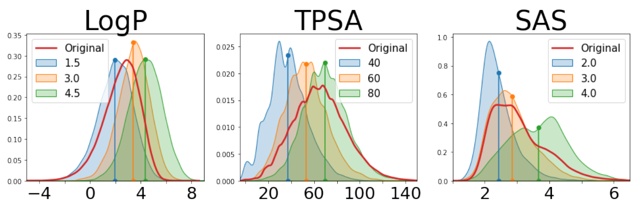


Figure 7: Distribution of attributes of generated molecules, together with distribution for the training dataset. Each color shows the value of a labeled attribute that was used for generation. PluGeN is capable of moving the density of generated molecules' attributes towards the desired value. The average of every distribution is marked with a vertical line.

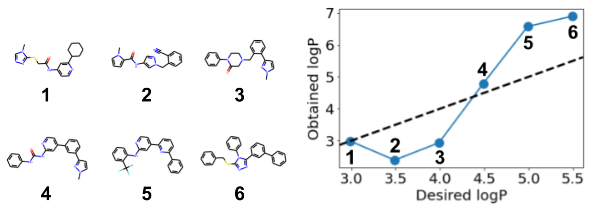
pounds. For every molecule, we model 3 physio-chemical continuous (not binary) labels: logP, SAS, TPSA, which values were calculated using RDKit package [4]. Additional explanations and more examples are given in the supplementary materials.

First, we imitate a practical task of de novo design (Olivecrona et al. 2017; Popova, Isayev, and Tropsha 2018), where we force the model to generate new compounds with desirable properties. For every attribute, we generate 25k molecules with 3 different values: for logP we set the label of generated molecules to: 1.5, 3.0, 4.5; for TPSA we set generated labels to: 40, 60, 80; for SAS we set them to: 2.0, 3.0, 4.0, which gives 9 scenarios in total. From density plots of labels of generated and original molecules presented in Figure 7, we can see that PluGeN changes the distribution of values of the attributes and moves it towards the desired value. A slight discrepancy between desired and generated values may follow from the fact that values of labeled attributes were sampled independently, which could make some combinations physically contradictory.

Next, we consider the setting of lead optimization (Jin et al. 2019; Maziarka et al. 2020), where selected compounds are improved to meet certain criteria. For this purpose, we encode a molecule into the latent representation of INF and force PluGeN to gradually increase the value of logP by 3 and decode the resulting molecules. The obtained molecules together with their logP are shown in Figure 8. As can be seen, PluGeN generates molecules that are structurally similar to the initial one, however with optimized desired attributes.

Obtained results show that PluGeN is able to model the physio-chemical molecular features, which is a non-trivial task that could speed up a long and expensive process of

[4]https://www.rdkit.org/



(a) Molecules decoded from path (b) LogP of presented molecules

Figure 8: Molecules obtained by the model during an optimization phase (left side), and their LogP (right side).

designing new drugs.

Conclusion

We proposed a novel approach for disentangling the latent space of pre-trained generative models, which works perfectly for generating new samples with desired conditions as well as for manipulating the attributes of existing examples. In contrast to previous works, we demonstrated that PluGeN performs well across diverse domains, including chemical molecule modeling, and can be combined with various architectures, such as GANs and VAEs backbones.

Acknowledgements

The research of M. Wołczyk was supported by the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund in the POIR.04.04.00-00-14DE/18-00 project carried out within the Team-Net programme. The research of M. Proszewska, Ł. Maziarka, M. Zieba and R. Kurczab was supported by the National Science Centre (Poland), grant no. 2018/31/B/ST6/00993, 2019/35/N/ST6/02125, 2020/37/B/ST6/03463 and the Polish National Centre for Research and Development (Grant LIDER/37/0137/L-9/17/NCBR/2018). The research of M. Śmieja was funded by the Priority Research Area DigiWorld under the program Excellence Initiative –- Research University at the Jagiellonian University in Kraków.

Ethical Impact

We did not identify ethical issues concerning our work, as we do not collect data, and we do not foresee malicious applications or societal harm. However, we believe that disentangling factors of variation can have a positive effect on reducing unjust correlations in the data. For example, even though CelebA dataset contains twice as many old men as old women, our method can generate an equal proportion of samples from those classes, thus avoiding amplifying bias in the data.

References

Abdal, R.; Zhu, P.; Mitra, N. J.; and Wonka, P. 2021. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (TOG)*, 40(3): 1–21.

Beery, S.; Horn, G. V.; and Perona, P. 2018. Recognition in Terra Incognita. In *ECCV*.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828.

Brakel, P.; and Bengio, Y. 2017. Learning independent features with adversarial nets for non-linear ica. *arXiv preprint arXiv:1710.05050*.

Brock, A.; Donahue, J.; and Simonyan, K. 2018. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations*.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Chen, R. T.; Li, X.; Grosse, R.; and Duvenaud, D. 2019. Isolating Sources of Disentanglement in VAEs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2615–2625.

Chen, X.; Duan, Y.; Houthooft, R.; Schulman, J.; Sutskever, I.; and Abbeel, P. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657*.

Choi, Y.; Uh, Y.; Yoo, J.; and Ha, J.-W. 2020. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8188–8197.

Dinh, L.; Krueger, D.; and Bengio, Y. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*.

Gan, Z.; Chen, L.; Wang, W.; Pu, Y.; Zhang, Y.; Liu, H.; Li, C.; and Carin, L. 2017. Triangle generative adversarial networks. *arXiv preprint arXiv:1709.06548*.

Gao, Y.; Wei, F.; Bao, J.; Gu, S.; Chen, D.; Wen, F.; and Lian, Z. 2021. High-Fidelity and Arbitrary Face Editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16115–16124.

Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; and Aspuru-Guzik, A. 2018. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2): 268–276.

Härkönen, E.; Hertzmann, A.; Lehtinen, J.; and Paris, S. 2020. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*.

He, Z.; Zuo, W.; Kan, M.; Shan, S.; and Chen, X. 2019. Attgan: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 28(11): 5464–5478.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 6626–6637.

Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; and Lerchner, A. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*.

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Jin, W.; Barzilay, R.; and Jaakkola, T. 2018. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, 2323–2332. PMLR.

Jin, W.; Yang, K.; Barzilay, R.; and Jaakkola, T. 2019. Learning multimodal graph-to-graph translation for molecular optimization. *International Conference on Learning Representations*.

Jo, Y.; and Park, J. 2019. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1745–1753.

Kang, S.; and Cho, K. 2018. Conditional molecular design with deep generative models. *Journal of chemical information and modeling*, 59(1): 43–52.

Karras, T.; Laine, S.; and Aila, T. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4401–4410.

Kaya, Y.; Hong, S.; and Dumitras, T. 2019. Shallow-Deep Networks: Understanding and Mitigating Network Overthinking. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, 3301–3310. PMLR.

Kim, H.; and Mnih, A. 2018. Disentangling by factorising. In *International Conference on Machine Learning*, 2649–2658. PMLR.

Kingma, D. P.; and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*.

Kingma, D. P.; Rezende, D. J.; Mohamed, S.; and Welling, M. 2014. Semi-supervised learning with deep generative models. *arXiv preprint arXiv:1406.5298*.

Klys, J.; Snell, J.; and Zemel, R. 2018. Learning latent subspaces in variational autoencoders. *arXiv preprint arXiv:1812.06190*.

Kodali, N.; Abernethy, J.; Hays, J.; and Kira, Z. 2017. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*.

Koperski, M.; Konopczynski, T.; Nowak, R.; Semberecki, P.; and Trzcinski, T. 2020. Plugin Networks for Inference under Partial Evidence. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2883–2891.

Kumar, A.; Sattigeri, P.; and Balakrishnan, A. 2017. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*.

Lample, G.; Zeghidour, N.; Usunier, N.; Bordes, A.; Denoyer, L.; and Ranzato, M. 2017. Fader networks: Manipulating images by sliding attributes. *arXiv preprint arXiv:1706.00409*.

Li, X.; Lin, C.; Li, R.; Wang, C.; and Guerin, F. 2020. Latent space factorisation and manipulation via matrix subspace projection. In *International Conference on Machine Learning*, 5916–5926. PMLR.

Liu, R.; Liu, Y.; Gong, X.; Wang, X.; and Li, H. 2019. Conditional adversarial generative flow for controllable image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7992–8001.

Locatello, F.; Bauer, S.; Lucic, M.; Raetsch, G.; Gelly, S.; Schölkopf, B.; and Bachem, O. 2019. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, 4114–4124. PMLR.

Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; and Warchoł, M. 2020. Mol-CycleGAN: a generative model for molecular optimization. *Journal of Cheminformatics*, 12(1): 1–18.

Mirza, M.; and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Nitzan, Y.; Bermano, A.; Li, Y.; and Cohen-Or, D. 2020. Disentangling in latent space by harnessing a pretrained generator. *arXiv preprint arXiv:2005.07728*, 2(3).

Olivecrona, M.; Blaschke, T.; Engkvist, O.; and Chen, H. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1): 1–14.

Park, T.; Liu, M.-Y.; Wang, T.-C.; and Zhu, J.-Y. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2337–2346.

Perarnau, G.; Van De Weijer, J.; Raducanu, B.; and Álvarez, J. M. 2016. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*.

Popova, M.; Isayev, O.; and Tropsha, A. 2018. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7): eaap7885.

Rebuffi, S.; Bilen, H.; and Vedaldi, A. 2017. Learning multiple visual domains with residual adapters. In Guyon, I.; von

Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 506–516.

Shen, Y.; Yang, C.; Tang, X.; and Zhou, B. 2020. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence*.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28: 3483–3491.

Sorrenson, P.; Rother, C.; and Köthe, U. 2020. Disentanglement by nonlinear ica with general incompressible-flow networks (gin). *arXiv preprint arXiv:2001.04872*.

Spurek, P.; Nowak, A.; Tabor, J.; Maziarka, Ł.; and Jastrzebski, S. 2020. Non-linear ICA based on Cramer-Wold metric. In *International Conference on Neural Information Processing*, 294–305. Springer.

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

Tewari, A.; Elgharib, M.; Bernard, F.; Seidel, H.-P.; Pérez, P.; Zollhöfer, M.; and Theobalt, C. 2020. Pie: Portrait image embedding for semantic control. *ACM Transactions on Graphics (TOG)*, 39(6): 1–14.

Tov, O.; Alaluf, Y.; Nitzan, Y.; Patashnik, O.; and Cohen-Or, D. 2021. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4): 1–14.

Wang, H.-P.; Yu, N.; and Fritz, M. 2021. Hijack-gan: Unintended-use of pretrained, black-box gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7872–7881.

Weininger, D. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1): 31–36.

Wołczyk, M.; Wójcik, B.; Bałazy, K.; Podolak, I.; Tabor, J.; Śmieja, M.; and Trzciński, T. 2021. Zero Time Waste: Recycling Predictions in Early Exit Neural Networks. *arXiv preprint arXiv:2106.05409*.

Yan, X.; Yang, J.; Sohn, K.; and Lee, H. 2016. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, 776–791. Springer.

Zhou, W.; Xu, C.; Ge, T.; McAuley, J. J.; Xu, K.; and Wei, F. 2020. BERT Loses Patience: Fast and Robust Inference with Early Exit. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.