Expressivity of Parameterized and Data-driven Representations in Quality Diversity Search

Alexander Hagg*
alex@haggdesign.de
Bonn-Rhein-Sieg University of Applied Sciences
Sankt Augustin, Germany

Sebastian Berns
Queen Mary University of London
London, United Kingdom

Alexander Asteroth
Bonn-Rhein-Sieg University of Applied Sciences
Sankt Augustin, Germany

Simon Colton†
Queen Mary University of London
London, United Kingdom

Thomas Bäck
Leiden Institute of Advanced Computer Science
Leiden, The Netherlands

ABSTRACT

We consider multi-solution optimization and generative models for the generation of diverse artifacts and the discovery of novel solutions. In cases where the domain's factors of variation are unknown or too complex to encode manually, generative models can provide a learned latent space to approximate these factors. When used as a search space, however, the range and diversity of possible outputs are limited to the expressivity and generative capabilities of the learned model. We compare the output diversity of a quality diversity evolutionary search performed in two different search spaces: 1) a predefined parameterized space and 2) the latent space of a variational autoencoder model. We find that the search on an explicit parametric encoding creates more diverse artifact sets than searching the latent space. A learned model is better at interpolating between known data points than at extrapolating or expanding towards unseen examples. We recommend using a generative model's latent space primarily to measure similarity between artifacts rather than for search and generation. Whenever a parametric encoding is obtainable, it should be preferred over a learned representation as it produces a higher diversity of solutions.

CCS CONCEPTS

• **Computing methodologies** → **Search methodologies**; **Learning latent representations**; *Continuous space search*;

KEYWORDS

Quality Diversity, Generative Models, Variational Autoencoder.

*Also with Leiden Institute of Advanced Computer Science.
†Also with SensiLab, Monash University.

GECCO '21, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8350-9/21/07…$15.00
https://doi.org/10.1145/3449639.3459287

ACM Reference Format:
Alexander Hagg, Sebastian Berns, Alexander Asteroth, Simon Colton, and Thomas Bäck. 2021. Expressivity of Parameterized and Data-driven Representations in Quality Diversity Search. In *2021 Genetic and Evolutionary Computation Conference (GECCO '21), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3449639.3459287

1 INTRODUCTION

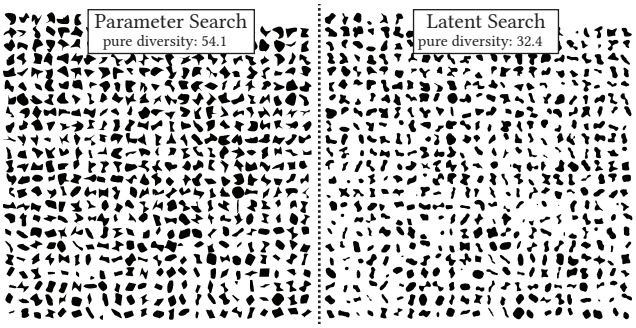

Figure 1: Searching the parameter space produces a more diverse set of artifacts than searching a VAE's latent space. In both cases, the same VAE's latent dimensions were used as niching dimensions of a quality diversity algorithm.

While engineering-driven design optimization looks for solutions to technical problems, artistic practices are usually more concerned with generating culturally valuable artifacts. However, these two approaches are more similar than the seeming difference in focus and objective would suggest. Architects and engineers often use the output of a design optimization tool in the beginning of the design process in order to survey the space of possibilities, where underlying parameters can have complicated correlations [4]. Candidate solutions are then expanded or contracted upon in an iterative design loop. Similarly, artists might set up an evolutionary system to find initial inspiration and continue to it towards a desired outcome through the iterative adjustment of the fitness function. In both workflows, the diversity of the generated population is key to illustrating the range of possibilities. We propose that initial diversity is the basis for the potential of later discoveries. Focusing on only

one optimal individual too early limits the chances of encountering unexpected candidate solutions.

Evolutionary multi-solution approaches such as quality diversity (QD) algorithms have been developed for the purpose of divergent search [19]. Defining QD descriptors by hand is a non-trivial task which requires expertise and, depending on the domain, often cannot compete with an automated solution [14]. Deep generative models (GM) such as variational autoencoders (VAE) [18] can extract patterns from raw data, learn meaningful representations for the data set and accurately produce more samples with similar properties. Disentangled representation learning can furthermore equip a model's latent space with linearly separated factors of variation [5], revealing the underlying factors of a generative process. The resulting feature compression model encodes descriptors to be used with QD algorithms [7, 9, 14]. While the advantage of learning from data lies in the recognition of complex patterns, the expressivity of the resulting GM is entirely dependent on the quality and representativeness of the data samples provided. This is especially critical when relying on such a model to produce novel examples and diverse sets of outputs. In fact, artists who employ generative adversarial networks (GANs) often use a variety of strategies to actively diverge from the intended purpose of these models and produce outputs significantly different from the original data [1].

We compare the performance of multi-solution evolutionary search in the parameter space of a generative system with the search in the latent space of a VAE that was trained with examples from the same system. An example of the resulting solution sets (see Sec. 4.2) produced by the two search methods is depicted in Fig. 1. While the latent space is built from a limited data set, the parameter space represents the full range of the system's possible output. The purpose of this work is to understand how expressive either of these search spaces are and, from this knowledge, to derive recommendations for their usage. We choose the simple, yet illustrative example problem of shape optimization, as previously introduced [14], for easy interpretation and visualization. While more complex domains might be closer to actual applications, they would make presentation of our results less accessible. We assume our findings generalize to those domains. Shape is an important basic design element in art, architecture, engineering, as well as graphic and industrial design. On the one hand, shapes can carry semantic meaning (e.g. letters of a font) and on the other hand, define the properties and visualize the form of a physical object in engineering-driven design (e.g. the cross-section of a wing optimized for aerodynamical flow).

Our work is relevant in two scenarios: 1) when the generative process is manually defined but a VAE is used to compare artifacts (i.e. distance/similarity estimation), and 2) when only data is available and the underlying patterns are unknown or too difficult to extract manually and have to be learned by an appropriate model. The present study makes the following contributions:

(1) In the context of the first scenario, we give informed recommendations of how to use a VAE to its full capacity in combination with a QD algorithm. We test whether the latent space is suitable both for searching for artifacts and for evaluating artifacts' similarity or whether the two steps should be performed in separate spaces.

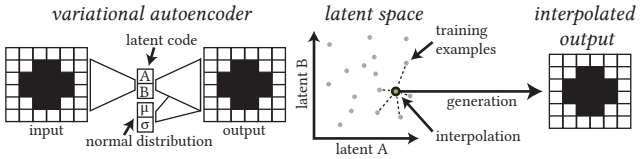

Figure 2: Left: variational autoencoder, center: sampling from latent space, right: interpolated output.

(2) For both scenarios, we give evidence for the limitations of VAEs in their ability to represent and generate examples beyond the original training data and, as a result, the diversity of their output.

2 BACKGROUND

In this section, we provide background knowledge on the two core methods used in our generative system, VAE and QD search. We briefly discuss related work.

2.1 Variational Autoencoders

VAEs are a likelihood-based method for generative modelling in deep learning. They follow the standard architecture of an autoencoder: a compressing encoder network, mapping data samples to latent space, and a decoder network which is trained to generate the original samples from the corresponding latent codes (Fig. 2). A VAE can generate new samples by interpolating between training locations in the latent space. While common autoencoders draw from an unrestricted range of latent code values, the latent space of a VAE is typically modelled to be a centered isotropic multi-variate Gaussian ($\mathcal{N}(0, I)$). The VAE training objective is to optimize a lower bound on the log-likelihood of the data. We use a beta-annealing variant of the loss term to improve disentanglement with improved reconstruction [5]. This variant of the evidence lower bound (ELBO) calculates the loss function L over the predicted output x and the ground truth $\hat{x}$ as follows:

$$L(x, \hat{x}) = C(x, \hat{x}) + \beta \cdot (KL(x, 0, 1) - \gamma) \tag{1}$$

Eq. 1 consists of a reconstruction loss term, in this case the binary cross-entropy C between prediction and ground truth, and a regularization term, which penalizes a latent distribution that is not similar to a normal distribution with $\mu = 0$ and $\sigma = 1$. The regularization term is calculated using Kullback-Leibler divergence and scaled by the parameter β. The annealing factor γ is increased from 0 to 5 during training to focus on improving the reconstruction error in the beginning of the training and then gradually improve the distribution in latent space. The internal latent space of a converged model provides meaningful representations in which distances between data points correspond to their semantic similarity. In this work, we use a VAE's internal representation to estimate the similarity of artifacts.

Previous work employed autoencoders for dimensionality reduction and the encoding of behavioral descriptors in a control task. In robotics, this approach allows robots to autonomously discover the range of their capabilities, without prior knowledge [7]. GM have been used to distinguish parameterized representations in shape

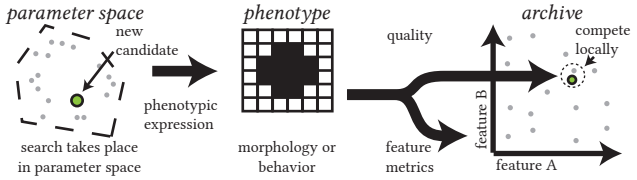

Figure 3: QD searches in parameter space and maintains diversity by only allowing solutions to compete based on their phenotypic similarity. Candidate artifacts are assigned to an archive based on features that are usually manually created a priori. Candidates are only placed inside the archive if they improve its quality value locally.

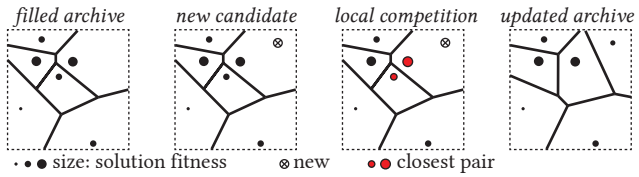

Figure 4: Updating a Voronoi archive. A new candidate artifact is compared to the closest artifacts. The worse of the two is rejected or removed from the archive and the artifact with higher fitness is kept or added to the archive. Here the maximum number of niches is set to six and the borders between niches are drawn to illustrate each item's range of influence and how it changes after an update.

optimization [13, 14]. They have also been employed to learn an encoding during optimization, using them as a variational operator [9]. Other GMs like GANs have been used in latent variable evolution [2] to generate levels for the video games Super Mario Bros. [23] and Doom [10]. A model's latent space is searched with an evolutionary algorithm for instances that optimize for desired properties such as the layout or difficulty of a level. While some authors view the generated levels as novel, none have studied exactly how novel or diverse of an output such a system can produce.

2.2 Quality Diversity Search

Optimality is not always the only goal in engineering or design. Finding a diverse set of ways to solve a problem increases potential innovation in the design process. Algorithms built around diversity as well as optimality enable engineers to use algorithms much earlier in the real world design process. Multi-solution optimization is a field that is getting more attention due to the advent of QD algorithms and GM. QD has been shown to produce more diverse sets of artifacts than classical approaches like multi-criterion and multimodal optimization [14].

Multi-criterion optimization defines diversity w.r.t. solution fitness. Multimodal optimization uses parametric similarity to distinguish and protect novel solutions, creating a diverse set of artifacts. In contrast, QD compares solutions on the basis of phenotypic, not parametric or objective similarity, and combines optimality with solution diversity [12]. QD measures similarity between artifacts based on morphological or behavioral features that can usually only be obtained by expressing a solution to its phenotype or even placing the artifact in its environment, i.e. through expensive simulation.

QD searches in parameter space (Fig. 3), but solutions are evaluated based on their expressed phenotypes. Predefined feature metrics, which measure some aspects of behavior or morphology are used to assign an artifact to a niche in an archive, which keeps track of the artifacts found so far. Niching is commonly used in evolutionary approaches to protect novel solutions from not being selected. Examples of archive features are the proportion of time that each leg is in contact with the ground in a hexapod robot's walking gait [8], the turbulence in the air flow around a shape [15], or surface area of a shape [15]. Competition between artifacts only takes place when they are assigned to the same niche. An artifact is only added if it survives local competition in the niche. New artifacts are created by selecting surviving candidates from the archive and adding perturbations to their genome, e.g. through mutation and/or crossover with other genomes.

The QD algorithm used in this work is based on an alternative formulation to MAP-Elites [8]. *Elites* is the common nomenclature for high-performing archive members. In MAP-Elites, the archive consists of a fixed grid of niches, which leads to an exponential growth of niches with the number of phenotypic feature dimensions. CVT-Elites [22] dealt with this problem by predefining fixed niches using a Voronoi tessellation of the phenotypic space. Due to their fixed archive, both methods tend to reduce the variance of the solution set in the first iterations. Initial (random) samples tend to not cover the entire phenotypic space and thus competition is harsher, leading to the excluding of many solutions in the beginning. To maximize the number of available training samples for the VAE, the Voronoi-Elites (VE) [14] algorithm is therefore more appropriate. VE does not precalculate the niches. It accepts all new artifacts until the maximum number of niches is surpassed. Only then the pairs of elites that are phenotypically closest to each other are compared, rejecting the worst-performing pair members.

The VE archive's evolution is illustrated in Fig. 4. Selection pressure is applied based on artifact similarity. In effect, VE tries to minimize the variation of distances between artifacts in the (unbounded) archive. The total number of niches/artifacts is fixed, independent of the number of archive dimensions. Tournament selection is used to select artifacts from the archive. New artifacts are created by mutation, drawn from a normal distribution.

3 STUDY SETUP

When a VAE is used for generation or search, the diversity of its output is bound by the expressivity of its latent space. The objective of our study is to analyze the generative capabilities of a VAE's latent space and give empirical evidence for its limitations.

This section outlines the details of our study's subject domain, the generation of two-dimensional shapes, lists the general configurations of the VAE and VE algorithm (specific settings for experiments can be found in the experimental setups below) and explains how the two methods are combined to build two versions of a generative system which we compare in a series of experiments.

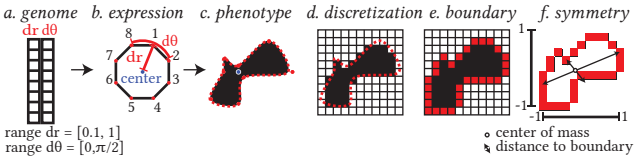

Figure 5: Generation of shapes: from (a) 16 genes to (b) eight control points, freely placed in two-dimensional space, to (c) a smooth interpolated spline and (d) a final bitmap rendering. Quality evaluation: (e) the boundary of the shape is determined and (f) the shape's symmetry is measured from its center of mass.

3.1 Shape Generation

For our study, we focus on the generation of two-dimensional shapes, similar to a data set which has been proposed for the evaluation for the quality of disentangled representations [16]. Here we explain the setup of our shape generating system in the context of its later use with the VE algorithm. The shapes are generated by connecting eight control points which can be freely placed in a two-dimensional space. Each control point is defined by two parameters, the radial (dr) and angular deviation ($d\theta$) from a central reference point (Fig. 5b). These 16 parameters serve as genomes, encoding the properties of each individual. To form a final smooth outline, the points are connected by locally interpolating splines [6] (Fig. 5c). A discretization step renders this smooth shape onto a square grid resulting in a bitmap of 64×64 pixels (Fig. 5d).

3.2 Fitness

As a simple objective and fitness criterion, we have chosen point symmetry, which acts as an exemplary problem in generative applications, is easy to understand, and is computationally inexpensive. To determine an artifact's quality, first, the boundary of the artifact is determined (Fig. 5e). Second, the coordinates of the boundary pixels are normalized to a range of -1 to 1 in order to remove any influence of the shape's scale (Fig. 5f). Third, the center of mass of the boundary is determined to serve as the center of point symmetry. Fourth, the distances to the center of pixels opposite of each other w.r.t. the center of mass are compared. Finally, the symmetry error E_s, the sum of Euclidean distances of all $n/2$ opposing sampling locations to the center, is calculated (Eq. 2). A maximally symmetric shape is one for which this sum equals zero. The fitness function $f_P(\boldsymbol{x})$ is calculated as follows:

$$f_P(\boldsymbol{x}) = \frac{1}{1 + E_s(\boldsymbol{x})} \qquad E_s(\boldsymbol{x}) = \sum_{j=1}^{n/2} \left\| \boldsymbol{x}_j - \boldsymbol{x}_{j+n/2} \right\| \tag{2}$$

3.3 VAE Configuration

Throughout this work, we use a VAE with a beta-annealing loss term [3, 5] and its decoder as a mapping network from latent codes to phenotype bitmaps (see Fig. 6). The model's encoder network is made up of four downscaling blocks, each consisting of a convolution layer (8, 16, 32 and 64 filters respectively; kernel size 7×7; stride 2) followed by a ReLU activation function. The set of blocks is followed by a final fully-connected layer. The decoder network inversely maps from the latent space to bitmaps through five transposed convolution layers, which have 64, 32, 16, 8 and 1 filter respectively, kernel size 7×7 and stride 2, except for the first layer which has a kernel size of 14×14. The last layer is responsible for outputting the correct size (64×64 pixels). The weights of both networks are initialized with the Glorot initialization scheme [11]. The regularization term scaling factor β was set to 4 and the annealing factor γ was increased from 0 to 5 over the course of the training, in order to focus on improving the reconstruction error first, and improve the distribution in latent space later in the process. Each model was optimized with the Adam optimizer [17] with a learning rate $\mu = 0.001$ and a batch size of 128.

3.4 VE Configuration

We configure VE to start with an initial set of samples, generated from a Sobol sequence [21] in parameter space. Sobol sequences are quasi-random and space-filling. They decrease the variance in the experiments but ensure that the sampling is similar to a uniform random distribution and easily reproduced. In all experiments, VE runs for 1,024 generations, producing 32 children per generation. Children are produced by adding a small mutation vector, drawn from a normal distribution centered around zero with $\sigma = 0.1$, to selected parent individuals. The selection is drawn at random from the archive. The number of artifacts in the archive remains constant, identical to the initial population size, over the entire experiment.

3.5 Combining a VAE with VE into a Generative System

The VAE is combined with VE to form the AutoVE [14] generative system with the objective to produce point-symmetric two-dimensional shapes. The difference to the original formulation is the use of a VAE instead of a classical autoencoder, as a VAE creates a more even occupancy of training samples in latent space as well as allowing interpolating new examples and disentangling latent dimensions. The full generative process is illustrated in Fig. 7 and can be separated into two phases: 1) initialization and 2) an evolutionary optimization loop. At initialization time, a set of random genomes is drawn and translated into bitmaps, their phenotypic counterpart. The VAE is trained to convergence on this set of bitmap data. The learned latent space is then used in the following evolutionary process and the model's encoder and decoder networks serve as mapping functions between the phenotypic bitmap representations and the model's latent representations and vice versa.

In the evolutionary optimization loop, the VE algorithm iteratively updates the archive and tries to increase the diversity as well as the quality of the archive through local competition. To compare two candidates to each other, it relies on the VAE's low-dimensional latent representations, which preserve semantically meaningful distances. We perform this optimization process in two different search spaces for the central comparison of our study: 1) parameter space (the explicit genome encoding) and 2) the VAE's latent space (the learned representation). In this way, we can evaluate the expressivity of a VAE's latent space and its capability to generate a diverse set of artifacts in comparison to the full space of possibilities which is reflected by the 16 predefined genetic parameters. The performance of the two approaches is measured in terms

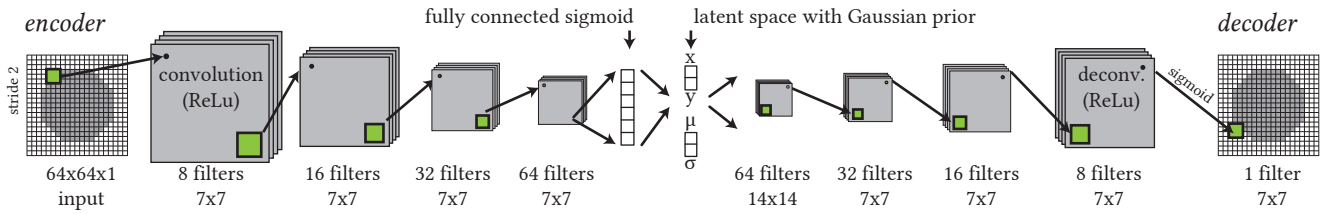

Figure 6: Architecture of a convolutional variational autoencoder.

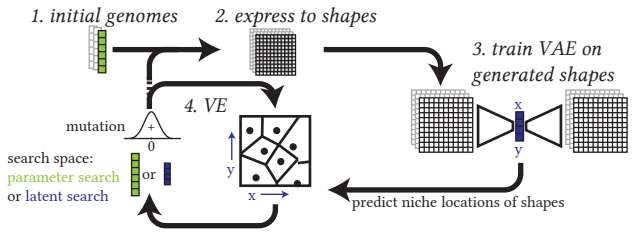

Figure 7: AutoVE combines a VAE and VE into a generative system in two phases. First, initialization: (1) a random set of genomes is generated and (2) converted into shape bitmaps which are used to (3) train a VAE. Second, optimization loop: (4) VE iteratively updates the archive of candidates. We compare two setups of this loop: the VE performs search either in parameter space or in the VAE's latent space.

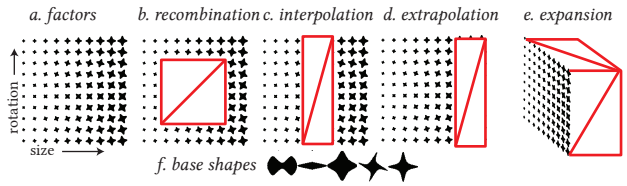

Figure 8: (a) generative factors used to create data sets in this work; (b–e) four tasks on which we compare the performance of latent space search with parameter search, the red rectangles indicate artifacts that either have been left out of a data set (b, c, d) or are not available (e); (f) all base shapes used in this work.

of diversity of the produced set (more on our diversity metric in the following Section 3.6). This setup allows us to study the limitations of the latent space of a VAE and compare it to the baseline diversity of searching for candidate solutions over the possible parameters.

3.6 Diversity Metric

In the QD community, metrics that measure the diversity of a solution set are usually domain-dependent or require to take one of the QD algorithms as a baseline [14]. Archive-dependent metrics do not generalize well and introduce biases. We therefore only use distance-based diversity metrics that are calculated on the expressed shapes directly. Pure diversity (PD) measures diversity within a set of artifacts. We use the $L^{0.1}$-norm, which is suitable for high-dimensional cases [24], to find the minimum dissimilarity between an individual item and the items in a set X (Eq. 3). The PD value of a set X is calculated recursively and is equal to the maximum of the sum of its value on all but one of the members and the minimum distance of that member to the set (Eq. 4).

$$d(s, X) = \min_{s_i \in X}(L^{0.1}(s, s_i)) \tag{3}$$

$$PD(X) = \max_{s_i \in X}(PD(X - s_i) + d(s_i, X - s_i)) \tag{4}$$

PD was first proposed in the context of many-objective optimization [24] and has been applied to high-dimensional phenotypes [14]. PD can deal with a high number of dimensions and is consistent with some other widely used diversity metrics. By calculating PD on a set of bitmaps, we can measure diversity directly, independent of the representation in parameter space or the VAE's latent space.

4 EXPERIMENTS

It is commonly assumed that GM, such as a VAE, have good interpolative and reasonable extrapolative capabilities, which makes their latent space a potentially appealing search space. But how well a search in this space performs in terms of generating a diverse output, to our knowledge, has not yet been adequately investigated. In the setup of our generative system[1] the latent space of a VAE is used to search for and generate two-dimensional shapes in the form of square bitmaps. We compare the output diversity of this process to the baseline diversity of a search performed on the explicit genome encoding (parameter space). We aim to gain insight into two questions: 1) how accurately can a VAE represent a variety of shapes, that is to say how useful are its latent representations, and 2) how well can a VAE generate unknown shapes?

All data sets in our first experiment consist of samples which have been produced by varying two generating factors: scale and rotation (Fig. 8). We present here a series of corresponding tasks that we evaluate in two experiments:

a) With a complete set of samples as a baseline data set we evaluate the standard reconstruction error of the model in order to determine the general quality of latent representations.
b) In the recombination task, we leave out a subset of artifacts in the center of the ranges of values of both generating factors, leaving sufficient examples at either end of the ranges.
c) In the interpolation task, the left-out subset of artifacts covers the complete range of one of the two generating factors,

[1]The code to reproduce the experiments can be found at https://github.com/alexander-hagg/ExpressivityGECCO2021.

while for the other some examples remain at both ends of its range of values.

d) The extrapolation task consists in omitted samples at one end of values of one factor of variation, which affects the complete range of the other factor.

e) The expansion tasks focuses on generating artifacts beyond the two given generating factors from the complete data set.

The VAE is expected to perform reasonably well in recombining (b), interpolating between (c) and extrapolating beyond the available variations (d) to reproduce the samples missing from the training data. In the expansion task (e), we expect the VAE's latent space to only produce artifacts of poor quality outside of the generating factors present in the training data.

4.1 Recombination, Interpolation and Extrapolation

We train one baseline VAE on the complete set of variations of a base shape (Fig. 8a,f) (256 shapes, scaled by factors of 0.1 to 1.0 and rotated by 0 to $\frac{\pi}{2}$ in 16 steps each) and three additional models, each one on the data set of one special task (b–d) with held-out samples. The VAEs are trained for 3,000 epochs, after which we choose the models with the lowest validation error (calculated on 10% of the input data).

To determine whether the VAE can correctly reproduce, and thus properly represent, the given shape, we measure the models' reconstruction errors. For the baseline model this is done over the complete data set. For the task models (b–d) the reconstruction error is calculated only on the held-out examples. We define the reconstruction error as the Hamming distance between an input bitmap and a generated bitmap, normalized by the total number of pixels. The Hamming distance is useful to measure differences between bitmaps, due to their high dimensionality. A high reconstruction error would indicate that the model cannot properly generate the shapes and that its latent space does not provide an adequate search space for VE. Generating shapes to which there are no corresponding training examples, the reconstruction errors of unseen shapes that can be created with recombination and interpolation (b and c) are expected to be lower than for extrapolation (d).

To determine the resolution of the models, we measure the distances in the latent space between the training examples for the baseline model and between the training and the unseen examples for the task models (b–d). If the latter are of a similar order of magnitude as the first, the models are able to distinguish unseen shapes from the training examples and from each other. This would indicate that the model's resolution is high enough to provide features of sufficient quality to perform a VE search.

This experiment was performed separately on each of the five base shapes (Fig. 8f) and for three different sizes of the VAE's latent space (4, 8, and 16 dimensions), as we assumed that the model would not able to perfectly learn the two generating factors. The results are reported as averages over the resulting 15 total runs.

Results. Fig. 10 shows the reconstruction, KL and total β-loss on the validation data, during training of the models. The training does not need much more than 1,000 epochs to converge.

Fig. 11 (left) shows the reconstruction errors for the training, recombination, interpolation and extrapolation sets. The error the models produce on the training samples is lower than when reproducing the recombination and interpolation sets. As expected, the error on the extrapolated shapes is highest. All significant differences (two-sample t-test, $p < 0.01$) between the reconstruction error on the whole data set and a hold-out set are marked with an asterisk. The latent distances between the shapes in the four sets are shown in Fig. 11 (right). The distance distributions are similar.

Four exemplary latent spaces are shown in Fig. 9 from models with a latent space dimensionality of 8, which has been projected to two dimensions with the dimensionality reduction method t-distributed stochastic neighbourhood embedding (t-SNE) [20]. The first latent space (a) corresponds to the baseline model, trained on the complete training set. The other visualizations (b, c, and d) show the three tasks in which some shapes have been omitted.

4.2 Expansion

The last task, expansion (e), cannot be treated as per the previous experiment, because we cannot easily define an a priori ground truth shape set "outside of latent space". Instead, we compare the two search spaces (parameter: PS, and latent: LS) using the framework proposed in Fig. 7. We measure which one of the two search spaces produces the most diverse set of artifacts using the PD metric explained above. The experiment is split up into two configurations.

In the first configuration (**R**), both of the compared search approaches start from the same random initial set of genomes, which is common in many optimization problems. We increased the size of the set to 512, as this experiment poses a more difficult optimization problem. The genomes are translated into bitmaps, which serve as the training data for a VAE model. VE is then performed in both search spaces to fill two separate archives of 512 shapes each. The resulting shape sets are compared w.r.t. their diversity and average fitness, which are often in conflict with each other. As the translation from genome to bitmap always produces a contiguous shape, it is reasonable to expect that a VAE would learn to produce shapes, and not only random noise, even when starting with a randomly generated set of examples.

Often a generative system does not start from a random set of data, but rather a set of examples that has been observed in the real world. A second configuration, continuation (**C**), is defined to reflect this. We ask whether the diversity improves when training a VAE with a set of high-quality generated artifacts from a previous VE search. We use the archive of shapes produced by PS from the random initial set (**R**) as training data for a new VAE model. Both PS and LS are then performed again with this improved model.

It is expected that LS will interpolate between training samples, but not be able to expand beyond the generative factors in the data, except through modeling errors. Since PS is performed in the encoding's parameter space, this search approach should be able to produce a higher artifact diversity in both configurations **R** and **C**.

The number of latent dimensions of the VAE has been set to 8, 16 and 32 to analyze the influence of the degrees of freedom in latent space, when it is lower than, equal to, or higher than the number of parameters of the genome representation. A higher number of degrees of freedom gives an advantage to the latent

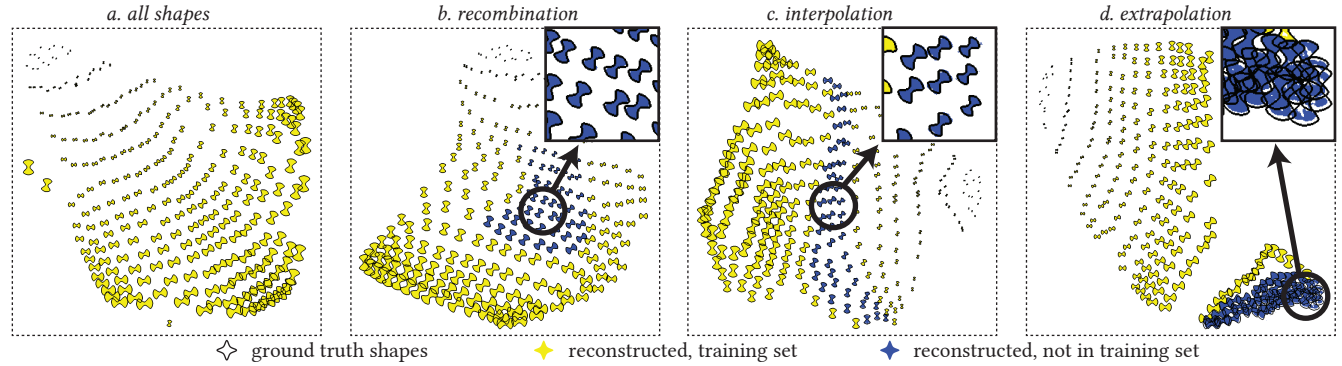

Figure 9: Examples of samples and latent spaces produced by the VAE with a latent dimensionality of eight (projected to two dimensions by t-SNE). Shapes in yellow represent samples that were present in the data set, while blue ones were not and have been generated by the model. Black outlines show the ground truth shapes, the difference between the outlines and shapes accounts for errors in reconstruction.

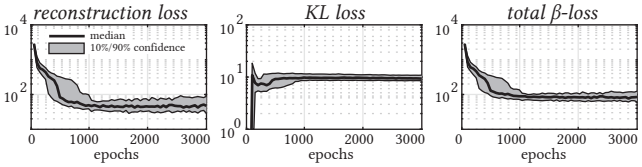

Figure 10: VAE validation losses during training.

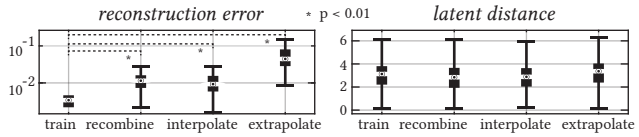

Figure 11: Reconstruction errors and latent distances for tasks a-d.

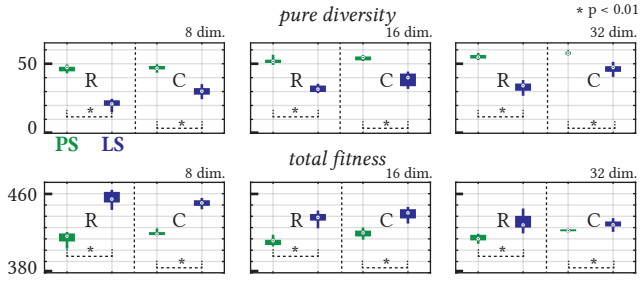

Figure 12: Diversity (top) and total sum of fitness (bottom) of artifact sets of both parameter (PS, green) as well as latent search (LS, blue). VAEs were trained with 8, 16 and 32 latent dimensions respectively. Both the random initialization (R) and continuation (C) configurations of the experiments are shown (in every box the two left-hand bars correspond to R and the two right-hand to C). Significant differences (two-sample t-test, $p < 0.01$) are marked with an asterisk.

model, a lower number would give it a disadvantage. When using 16 latent dimensions, VE deals with the same dimensionality in PS and LS. The number of filters in the VAE is quadrupled to give the model a better chance at learning the larger number of variations.

This experiment has been repeated 10 times per configuration: 1) random initial set **R** in PS, 2) continuation **C** in PS, 3) **R** in LS and 4) **C** in LS.

Results. Fig. 12 compares the PD and total fitness of the generated artifact sets. The diversity of PS is significantly higher than LS. In turn, LS produces artifacts with higher levels of fitness. Although the difference between PS and LS gets smaller when continuing search from an updated model (configuration **C**), it is still significant. Again, all significant differences (two-sample t-test, $p < 0.01$) between random initialization and continuation in all configurations are marked with an asterisk. Fig. 13 shows the expansion away from the latent surface achieved by PS, analogous to our previous hypothesis (Fig. 8e). For this visualization, the PS and LS artifacts' position in the 16-dimensional latent space is reduced to two dimensions using t-SNE. The reconstruction error of the model's prediction of the PS artifacts is used as a distance measure to the latent surface. An example of the resulting shape sets of PS and LS is shown in Fig. 1 to illustrate the effective difference in pure diversity.

5 DISCUSSION

VAEs are able to produce previously unseen examples through recombination and interpolation as expected (Section 4.1). The more difficult task, extrapolation beyond the extremes of the generative factors, results in a higher reconstruction error (Fig. 11). The distributions of latent distances between all four data set variants are similar. This suggests that, even when VAEs are not able to reproduce the extrapolated shapes, they can still distinguish them from the training data and from each other. That is to say, the position of examples in the latent space reflects their semantic relationship, as visualized in Fig. 9. Shapes are not properly reconstructed in the extrapolation task, but they are still positioned in a well-structured

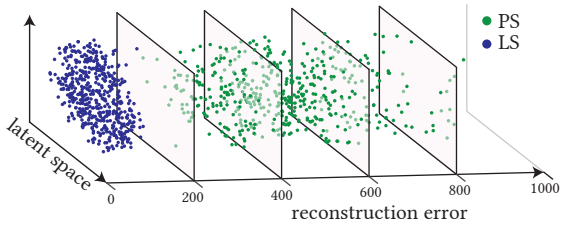

Figure 13: Expansion in a 16-dimensional latent model (projected to two dimensions with t-SNE). We interpret the reconstruction error of a shape as its distance from the latent surface. Samples from parameter search (PS, green) tend to extrapolate away from the latent distribution (LS, blue).

relation to others. The presented evidence leads to the hypothesis that expansion away from the latent surface is more difficult when searching the latent space directly.

We further examine the results of the expansion task (Section 4.2). The ability of a VAE to find new shapes is indirectly measured by comparing the diversity of the artifact sets created by a parameter (PS) and a latent search (LS). The diversity of PS is significantly higher than that of LS, as is shown in Fig. 12. This holds as the number of latent dimensions is increased beyond the number of degrees of freedom in the original encoding, or when the VAE is updated after a first VE run (**C**). Although a trade-off between diversity and fitness is expected, it becomes less pronounced in the 32-dimensional model. This provides evidence for the conclusion that PS indeed finds a more diverse set of artifacts than LS. We therefore recommend to use a more expressive predefined parametric encoding, whenever it is available, rather than the extracted feature space of a GM such as a VAE. Yet, a GM's latent space is still useful for its ability to distinguish shapes and its semantically meaningful mapping to lower-dimensional space.

6 CONCLUSIONS

In this work, we have presented a systematic study on the limitations of latent spaces of deep GMs as a base for divergent search methods, specifically the VE algorithm. Our findings quantify a VAE's ability to generate samples through recombination, interpolation and extrapolation within and expansion beyond the distribution of a given data set. We compare the diversity of generated artifacts when VE is run either in latent space or parameter space. Our findings show that the pure diversity of artifact sets generated by latent space search is significantly lower than that of parameter space search. Based on these observations we recommend using a VAE's latent space as an approximate measure of similarity. Evolutionary search for the generation of diverse outputs should, however, preferably be performed on explicitly expressed genome parameters, whenever these are available. The expressivity of a VAE, when used as a generator for diverse artifact sets, is limited by the generative factors in the training data.

Our findings are limited to multimodal continuous domain optimization. The presented conclusions are only meaningful to problem settings which allow for multiple solutions. Neither did we extend them to combinatorial search. The presented problem setting is kept simple in order to remain illustrative and easy to interpret. Most application domains are much more complex and more work is required to confirm or refute our assumptions on generalization.

We plan to extend these first results with a systematic study of the individual parts of our setup and their influence on expressivity. We will look at different priors for a VAE latent distribution, the size of the training data set, and mapping to a higher-dimensional latent space. Another opportunity for further study is the architecture of the generative model. Comparing the performance of a VAE to that of an autoregressive or flow-based model, a GAN or a transformer could highlight strengths or weaknesses of individual modelling methods, and allow for a more general understanding of their generative capabilities.

The usefulness of a GM's ability to interpolate, extrapolate or expand has to be discussed in the context of its application. In one setting, it might be ideal to perfectly reproduce a given domain and a model might be considered as working well if it can fit a distribution accordingly. In another context, however, and here we do include applications with a focus on exploration of possibilities and discovery, a systems's ability to surprise through its unexpected outputs can be of value and a desirable quality.

In the real world, the assumption that a GM can learn a "perfect" representation does not hold. Latent search spaces are not guaranteed to cover the whole possibility space as defined by the underlying parameters of the system that produced the training examples. The idea that a GM can fit the "true" distribution overlooks the fact that the data might not cover the entire range of parameters. Furthermore, a perfect model does not produce anything unexpected, only creating high quality artifacts with low diversity. A broken model, on the other hand, might not produce anything useful. The use of modeling errors to find novel artifacts is certainly a mechanism that can allow us to find novel solutions within the model. An important question for future work is whether we can use early stopping when training models to create novel yet useful artifacts: is there a correlation between training loss and diversity?

We can assume that GMs are always limited by the data we give them, which biases models towards the most prominent features therein. Yet, in very high-dimensional domains for which we can collect large data sets, like image and video data, a search in latent space already affords a vast amount of possible outcomes, which might be sufficient for some applications.

Surprisingly, using generative models to understand diversity and guide evolutionary computation produces more diverse sets of artifacts than having the models generate the artifacts themselves. With this work, we hope to have contributed some inspiration to using generative models in novel ways.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their valuable comments and helpful suggestions. S. Berns is funded by the EPSRC Centre for Doctoral Training in Intelligent Games & Games Intelligence (IGGI) [EP/S022325/1].

REFERENCES

[1] Sebastian Berns and Simon Colton. 2020. Bridging Generative Deep Learning and Computational Creativity. In *Proceedings of ICCC*.

[2] Philip Bontrager, Aditi Roy, Julian Togelius, Nasir Memon, and Arun Ross. 2018. Deepmasterprints: Generating Masterprints for Dictionary Attacks via Latent Variable Evolution. In *Proceedings of BTAS*.

[3] Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 10–21.

[4] Erin Bradner, Francesco Iorio, and Mark Davis. 2014. Parameters Tell the Design Story: Ideation and Abstraction in Design Optimization. In *Proceedings of SimAUD*.

[5] Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. 2017. Understanding Disentangling in Beta-VAE. In *NIPS Workshop on Learning Disentangled Representations*.

[6] Edwin Catmull and Raphael Rom. 1974. A Class of Local Interpolating Splines. In *Computer Aided Geometric Design*.

[7] Antoine Cully. 2019. Autonomous Skill Discovery with Quality-Diversity and Unsupervised Descriptors. In *Proceedings of GECCO*.

[8] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. 2015. Robots that can adapt like animals. *Nature* 521, 7553 (2015).

[9] Adam Gaier, Alexander Asteroth, and Jean-Baptiste Mouret. 2020. Discovering Representations for Black-box Optimization. In *Proceedings of GECCO*, Vol. 11.

[10] Edoardo Giacomello, Pier Luca Lanzi, and Daniele Loiacono. 2019. Searching the latent space of a generative adversarial network to generate Doom levels. In *2019 IEEE Conference on Games (CoG)*. IEEE, 1–8.

[11] Xavier Glorot and Yoshua Bengio. 2010. Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proceedings of AIStats*.

[12] Alexander Hagg. 2021. Phenotypic Niching using Quality Diversity Algorithms (accepted). In *Metaheuristics for Finding Multiple Solutions*, M. Epitropakis, X. Li, M. Preuss, and J. Fieldsend (Eds.). Springer Press.

[13] Alexander Hagg, Alexander Asteroth, and Thomas Bäck. 2020. A Deep Dive Into Exploring the Preference Hypervolume. In *Proceedings of ICCC*.

[14] Alexander Hagg, Mike Preuss, Alexander Asteroth, and Thomas Bäck. 2020. An Analysis of Phenotypic Diversity in Multi-Solution Optimization. In *Proceedings of BIOMA*.

[15] Alexander Hagg, Dominik Wilde, Alexander Asteroth, and Thomas Bäck. 2020. Designing Air Flow with Surrogate-assisted Phenotypic Niching. In *International Conference on Parallel Problem Solving from Nature*. Springer, 140–153.

[16] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. Beta-VAE: Learning Basic Visual Concepts With a Constrained Variational Framework. In *Proceedings of ICLR*.

[17] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*.

[18] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*.

[19] Joel Lehman and Kenneth O Stanley. 2011. Evolving a Diversity of Virtual Creatures Through Novelty Search and Local Competition. In *Proceedings of GECCO*.

[20] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data Using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008).

[21] Ilya Meerovich Sobol. 1967. On the Distribution of Points in a Cube and the Approximate Evaluation of Integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki* 7, 4 (1967).

[22] Vassilis Vassiliades, Konstantinos Chatzilygeroudis, and Jean-Baptiste Mouret. 2017. Using Centroidal Voronoi Tessellations to Scale Up the Multidimensional Archive of Phenotypic Elites Algorithm. *IEEE Transactions on Evolutionary Computation* 22, 4 (2017).

[23] Vanessa Volz, Jacob Schrum, Jialin Liu, Simon M. Lucas, Adam Smith, and Sebastian Risi. 2018. Evolving Mario Levels in the Latent Space of a Deep Convolutional Generative Adversarial Network. In *Proceedings of GECCO*.

[24] Handing Wang, Yaochu Jin, and Xin Yao. 2016. Diversity Assessment in Many-objective Optimization. *IEEE Transactions on Cybernetics* 47, 6 (2016).