Available online at www.sciencedirect.com

ScienceDirect

Nuclear Physics B 940 (2019) 113–129

NUCLEAR PHYSICS B

www.elsevier.com/locate/nuclphysb

Deep learning in the heterotic orbifold landscape

Andreas Mütter *, Erik Parr, Patrick K.S. Vaudrevange

Physik Department T75, Technische Universität München, James-Franck-Straße, 85748 Garching, Germany

Received 2 December 2018; received in revised form 18 January 2019; accepted 20 January 2019
Available online 25 January 2019
Editor: Stephan Stieberger

Abstract

We use deep autoencoder neural networks to draw a chart of the heterotic $\mathbb{Z}_6$-II orbifold landscape. Even though the autoencoder is trained without knowing the phenomenological properties of the $\mathbb{Z}_6$-II orbifold models, it identifies fertile islands in this chart where phenomenologically promising models cluster. Then, we apply a decision tree to our chart in order to extract the defining properties of the fertile islands. Based on this information we propose a new search strategy for phenomenologically promising string models.

1. Introduction

It is widely assumed that string theory, being a unique and UV-complete theory of gravity, can incorporate the Standard Model (SM) of particle physics. However, strings are conveniently defined to live in ten-dimensional space-time. Thus, six spatial dimensions have to be hidden from observation. This process is called compactification. By choosing a specific compactification, the properties of the resulting effective four-dimensional (4D) string model are fully specified: all symmetries, the particle spectrum and all interactions are fixed by the choice of compactification. However, in most cases these models are strikingly different from the SM. In addition, the choice of compactification and thus the resulting 4D string model is far from being unique. This

* Corresponding author.
E-mail addresses: andreas.muetter@tum.de (A. Mütter), erik.parr@tum.de (E. Parr), patrick.vaudrevange@tum.de (P.K.S. Vaudrevange).

https://doi.org/10.1016/j.nuclphysb.2019.01.013

freedom yields a huge number of 4D string models that is called the string landscape. Indeed, soon after the dawn of string theory the number of inequivalent 4D string models was quoted to be at least of the order 10^{1500}, a huge but finite number [1], see also [2].

There have been many attempts to identify those 4D string models that come as close as possible to (Minimal Supersymmetric extensions of) the SM (MSSM), see e.g. [3–16] and references therein. The motivation for such searches has several aspects: First of all, in the most optimistic case an existence proof of a 4D string model that is in agreement with all current experimental and observational data would clearly be a milestone in the study of string theory. Even if the SM or the MSSM is not found in the near future by searching the string landscape (as one might expect due to the enormous size of the string landscape) finding MSSM-like models could be beneficial to high energy particle physics: For example, one might uncover common properties of (MSSM-like) string models, like the absence of certain quantum field theory models, or one might identify new mechanisms to address theoretical shortcomings of the SM or of the MSSM.

Yet, searches in the string landscape mainly focus on the gauge symmetry of the MSSM and on the representation content suitable for three generations of quarks and leptons plus (at least) one Higgs-pair. In addition, due to the enormous number of inequivalent 4D string models these searches have to be restricted to small corners of the entire string landscape. Thus, exhaustive classifications of 4D string models are typically out of reach. Instead random scans for inequivalent MSSM-like models in small corners of the string landscape are state-of-the-art, for other approaches see e.g. [17,18].

Typically, a 4D string model is specified by $\mathcal{O}(100)$ compactification parameters that specify, for example, the geometry of the six-dimensional compactification space, fluxes or world-sheet parameters. These parameters have to satisfy certain consistency conditions, e.g. quantization conditions, Bianchi identities or world-sheet modular invariance of the one-loop partition function. Hence, a (random) scan in the string landscape is often performed as follows: first, one chooses the $\mathcal{O}(100)$ compactification parameters (maybe randomly). Then, one checks that the consistency conditions are satisfied. Finally, if the parameters are consistent one computes the gauge group and the matter spectrum of the resulting 4D string model and compares this to the MSSM. While it is possible to find MSSM-like models in this way, it remains in general unclear whether some classes of compactification parameters are more likely to yield MSSM-like models than others, the reason for this being that in string theory the relation between the compactification parameters and the resulting particle spectrum is in general highly non-trivial and, additionally, computationally intensive. Moreover, this strategy suffers from the fact that a huge parameter space needs to be searched in order to find only a relatively small number of MSSM-like models.

In this paper we propose and demonstrate a new search strategy for MSSM-like models using techniques from machine learning.[1] As in the standard approach, we concentrate on one corner of the entire string landscape and start with a random scan in the corresponding parameter space of $\mathcal{O}(100)$ compactification parameters. However, we do not aim at an exhaustive random scan but stop searching after a rather small fraction of inequivalent 4D string models has been constructed. Furthermore, we keep all inequivalent 4D string models that we find and not only the MSSM-like models. By doing so, we obtain a coarse sample of this corner of the string landscape. Now, the hope is that one can identify islands in this coarse sample where

[1] See e.g. [19–25] for different approaches to study the string landscape using machine learning.

promising MSSM-like models accumulate. To uncover such islands we use a deep autoencoder neural network [26] – a concept from unsupervised machine learning. This way, we can obtain an approximate, lower-dimensional (e.g. two-dimensional) non-linear parametrization of the $\mathcal{O}(100)$-dimensional parameter space. Thus, we are able to draw two-dimensional charts of one corner of the string landscape. Indeed, it turns out that MSSM-like models cluster in islands within such two-dimensional charts of the string landscape – even though the autoencoder neural network had no information of a model being MSSM-like or not. Having identified these islands, the next step would be to perform finer scans (or even classifications) in these regions of the parameter space and, consequently, obtain a huge sample of MSSM-like models. Obviously, using this strategy it is by no means guaranteed that all promising models can be uncovered, and we comment on possible extensions of our search strategy to address this issue. In the following we exemplify our proposal at the landscape of heterotic $\mathbb{Z}_6$-II orbifolds.

2. Parameter space of heterotic $\mathbb{Z}_6$-II orbifolds

To be specific, we choose a promising corner in the string landscape: the so-called $\mathbb{Z}_6$-II orbifold compactification of the $E_8 \times E_8$ heterotic string [27,28]. This corner is chosen as there have been successful scans for MSSM-like $\mathbb{Z}_6$-II models, known in the literature as the $\mathbb{Z}_6$-II Mini-Landscape [9,10]. In particular, the search for MSSM-like orbifold models in the $\mathbb{Z}_6$-II Mini-Landscape was not performed as a completely random scan but it was based on a physical principle (i.e. the existence of local GUTs with complete matter representations) to identify particularly promising patches in the $\mathbb{Z}_6$-II parameter space. Our approach is in some sense complementary: we do not impose any physical principle but use a neural network and expect to identify those physical principles that yield MSSM-like orbifold models. By doing so in the $\mathbb{Z}_6$-II case, we can compare our findings with known results.

$\mathbb{Z}_6$-II models are parametrized by four 16-dimensional vectors (that describe boundary conditions on the world-sheet of closed strings): the so-called shift vector V and three Wilson lines W_3, W_2 and W_2'. Hence, $4 \times 16 = 64$ compactification parameters fully specify a single $\mathbb{Z}_6$-II model in this construction. We use the `orbifolder` [29] to randomly construct a coarse sample of inequivalent $\mathbb{Z}_6$-II models, i.e. to randomly generate consistent (i.e. quantized and modular invariant) sets of shifts and Wilson lines and to check for inequivalence of their gauge symmetries and massless matter spectra. Our coarse sample consists of $\mathcal{O}(700{,}000)$ models, i.e. less than 10% of the expected number of all $\mathbb{Z}_6$-II models [10].

However, at this point the compactification parameters are not yet ready for our machine learning purposes as it is strongly basis dependent. Therefore, we have to preprocess our 64 compactification parameters for each $\mathbb{Z}_6$-II model next: we map these 64 parameters to 26 so-called features, denoted by a vector X of integers such that two feature vectors $X_{(1)}$ and $X_{(2)}$ of the dataset cannot yield the same physical $\mathbb{Z}_6$-II model, unless $X_{(1)} = X_{(2)}$ – a fact that would not be given for shifts and Wilson lines, cf. [30]. In this way we render our input data of the neural network "invariant", i.e. basis independent. For details we refer to Appendix A. Now, we can use the autoencoder neural network on the dataset of 26-dimensional feature vectors $\{X\}$.

3. Machine learning in the $\mathbb{Z}_6$-II landscape

In this section, we give a detailed description of each step of our machine learning workflow. The overall idea is to identify patterns in the compactification parameters of $\mathbb{Z}_6$-II models that

lead to fertile islands in the string landscape, i.e. to patches in the parameter space of $\mathbb{Z}_6$-II models where the number of MSSM-like models is above average.

Let us start with an overview of the main points of the following discussion. We start with the preprocessing of our data, where we transform each $\mathbb{Z}_6$-II model into a suitable, machine-readable representation of 26 parameters X, also known as features. Then, we utilize a neural network to project each $\mathbb{Z}_6$-II model to a point in a two-dimensional image, yielding a "chart" of the $\mathbb{Z}_6$-II landscape. This is done such that the reconstruction error (i.e. the error when we map each point of the two-dimensional chart back to a feature vector X) is as small as possible. In this chart of the $\mathbb{Z}_6$-II landscape we can easily identify fertile islands where MSSM-like models appear to cluster – even though the neural network had no information of a model being MSSM-like or not during training. Afterwards, a decision tree is used to investigate these fertile islands, i.e. to find conditions on the 26 features X of a $\mathbb{Z}_6$-II model, such that one can directly decide if a given $\mathbb{Z}_6$-II model is located on a fertile island of the landscape or not. Finally, we discuss the performance of this procedure: we analyze how many MSSM-like models can be found if we restrict ourselves to search for MSSM-like models only on the fertile islands.

3.1. Data preprocessing

We start our machine learning workflow with the most basic, but crucial step: to define our training and validation sets. The training set is used in the machine learning algorithms to actually tune the weights and biases in the neurons, while the validation set is used to estimate the generalization properties of our machine learning model and can be exploited for hyperparameter search, e.g. to adjust the architecture of the neural network. Both of these sets contribute to the structure of the machine learning model.

In our case, we have a coarse sample of $\mathcal{O}(700{,}000)$ $\mathbb{Z}_6$-II models. This dataset is used to build our machine learning algorithm and is divided into 60% training and 40% validation data, all in a random procedure.

In order for the autoencoder to handle the data, we need a suitable numerical representation of the data. In our case, there exists a natural representation: the 26-dimensional feature vector of integers X, see Appendix A. However, it turns out that this representation does not perform well on the autoencoder. In fact, a more abstract representation, a so-called one-hot encoding, leads to a much better result. One-hot encoding is an approach for data that has no internal order like the values "green", "red", "blue". It generates a vector with n components where n equals the total number of possible values. Hence, in the example of three colors we have $n = 3$ and "green", "red" and "blue" have a one-hot encoding $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$, respectively. In our case of $\mathbb{Z}_6$-II models, each feature X_k of X can take 37 different values (i.e. there are in total 37 different breaking patterns for each E_8 factor). Thus, each component X_k of the 26-dimensional feature vector X is represented by a 37-dimensional vector. This 37-dimensional vector is zero except for the component, which corresponds to the given value of X_k. This component equals 1. Therefore, we obtain for each $\mathbb{Z}_6$-II model a $(26 \times 37 = 962)$-dimensional feature vector $X_{\text{one-hot}}$ as input to our neural network.

3.2. The autoencoder

The main effect of an autoencoder neural network is that redundancies in the feature vector $X_{\text{one-hot}}$ (such as irrelevant features) can be detected and reduced. Thus, an autoencoder

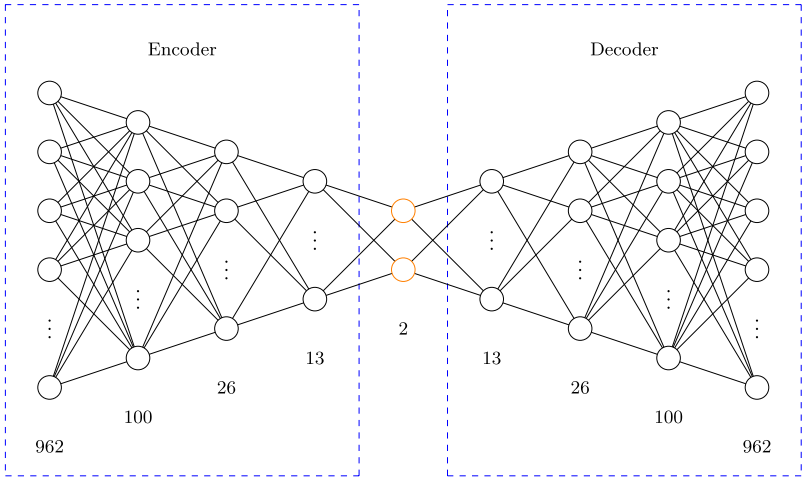

Fig. 1. Architecture of our autoencoder: For each $\mathbb{Z}_6$-II model the encoder takes 962 input features $X_{\text{one-hot}}$ (in a one-hot encoding of X) and reduces the number of features successively to 100, 26, 13 and finally to 2 – the so-called latent layer which is read out to draw a point in a two-dimensional chart of the landscape. The decoder is a mirrored version of the encoder with 962 output features. Now, the neural network is trained on $\mathcal{O}(400{,}000)$ $\mathbb{Z}_6$-II models such that the input features $X_{\text{one-hot}}$ match the output features.

yields a lower-dimensional, "compressed" representation of the feature vector $X_{\text{one-hot}}$. In order to achieve this, autoencoders are built as follows: starting from the input layer, the data is encoded through a number of hidden layers to the so-called latent layer. The latent layer is an information bottleneck: the number of neurons in this layer is much lower than the number of input nodes. Then, the encoding process is inverted in the second half of the network, the so-called decoder. The decoder leads to the output layer that has the same number of neurons as the input layer. Now, this network is trained such that the output features match the input features $X_{\text{one-hot}}$. This way, one ensures that the low-dimensional representation given in the latent layer is a compressed but valid representation of the high-dimensional feature vector $X_{\text{one-hot}}$, at least to an acceptable accuracy.

For our purposes, we implemented the autoencoder using TensorFlow [31]. By varying the architecture of the autoencoder we identify the following best setup: we use a fully connected autoencoder neural network with seven hidden layers and dimensionalities as indicated in Fig. 1. We choose the following activation functions: The latent layer uses the identity activation function, while we choose the `selu` activation function [32] for all other hidden layers, because it automatically accounts for batch normalization and hence makes the training process faster. Furthermore, we compute the L_2 loss and backpropagate the errors through the network.

Then, the autoencoder is trained on the training set of $\mathcal{O}(400{,}000)$ $\mathbb{Z}_6$-II models until the L_2 loss converges. Afterwards, the autoencoder is applied to the validation set. There, starting from the two-dimensional latent layer the decoder can reproduce on average 16.3 out of 26 features X, which corresponds to a L_2 loss of 0.013. This is a remarkable fact, since the information bottleneck was only two-dimensional and hence extremely narrow. Finally, we apply the encoder to all $\mathcal{O}(700{,}000)$ $\mathbb{Z}_6$-II models of the training and validation sets to obtain their two-dimensional parametrizations from the latent layer.

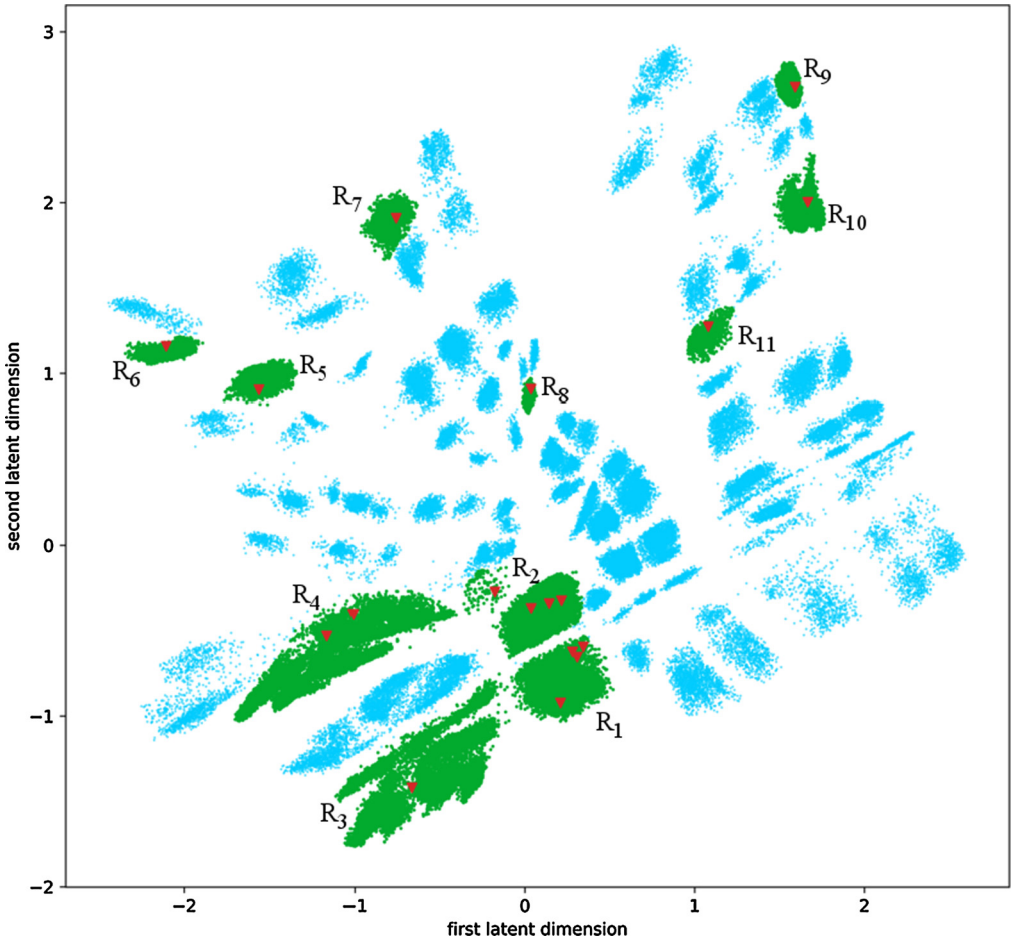

Fig. 2. The landscape of $\mathcal{O}(700{,}000)$ $\mathbb{Z}_6$-II models extracted from the autoencoder: Each point corresponds to a $\mathbb{Z}_6$-II model and MSSM-like models are highlighted as red triangles. It turns out that MSSM-like models populate eleven separated islands. We color these islands in green and label them by $R_1, \ldots, R_{11}$. In addition, all $\mathbb{Z}_6$-II models outside these islands are colored in blue and defined to live in the region R_0. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

3.3. A chart of $\mathbb{Z}_6$-II models and cluster selection

The result of the autoencoder is depicted in Fig. 2. It represents a chart of the landscape of $\mathcal{O}(700{,}000)$ $\mathbb{Z}_6$-II models of the training and validation sets, where the two-dimensional coordinates of each $\mathbb{Z}_6$-II model are extracted from the two-dimensional latent layer of the autoencoder.

The landscape turns out to be separated into various islands. We identify 18 MSSM-like models among the $\mathcal{O}(700{,}000)$ $\mathbb{Z}_6$-II models and highlight them as red triangles in Fig. 2. Interestingly, one can see that the MSSM-like $\mathbb{Z}_6$-II models cluster on a few islands and are not distributed over the entire chart. Note that during training, the autoencoder neural network had no information about a model being MSSM-like or not. Still, the MSSM-like $\mathbb{Z}_6$-II models are clustered. Hence, it seems that the autoencoder was able to identify common properties among the models and has grouped the models accordingly.

Next, we select those islands in Fig. 2 that contain MSSM-like $\mathbb{Z}_6$-II models (i.e. eleven islands) and highlight them. These eleven islands can act as a starting point for a refined search strategy for MSSM-like $\mathbb{Z}_6$-II models as we discuss in the next section. As a remark, we have verified that the clustering of MSSM-like $\mathbb{Z}_6$-II models on these islands is stable under a retraining of the autoencoder neural network with slightly different initial conditions.[2] Thus, we are confident that the autoencoder has identified some hidden patterns in the $\mathbb{Z}_6$-II landscape.

3.4. Towards a refined search strategy using a decision tree

Of course, drawing a chart that displays islands in the landscape containing MSSM-like orbifold models does not carry much insight by itself. Our aim is to learn something about the landscape of orbifold models. Hence, if one could understand the reason why a given orbifold model is located on a certain island in the landscape things would look different. This is precisely the next step which we take using a decision tree.

The decision tree works as follows: each $\mathbb{Z}_6$-II model (specified by 26 features X) is labeled according to which region R_i in the landscape it belongs to: either to one of the fertile islands or to the rest of the landscape R_0. Then, the decision tree is trained such that it splits the dataset according to whether or not a certain feature X_k is above or below a certain threshold value. As a result of the split, the data is then divided into two subsets. The feature X_k and its threshold are chosen such as to minimize the impurity in the two subsets that emerge as a consequence of the split. Each node is associated with the region R_i that is most dominant in this node. To measure the impurity of a node containing the dataset D, the Gini index $H(D)$ is a common choice. It is defined as

$$H(D) = \sum_i p_i(D)(1 - p_i(D)) \,, \tag{1}$$

where $p_i(D)$ is the percentage of points in D with label i and i sums over all labels, i.e. $i = 0, \ldots, 11$ in our case. In the end, one has a trained decision tree that can predict to which region R_i a given $\mathbb{Z}_6$-II model belongs to, using only simple True-or-False decisions like $X_k < X_{k,\,\text{threshold}}$.

For the decision tree we use the software scikit-learn [33]. To train the decision tree we split our set of $\mathcal{O}(700{,}000)$ $\mathbb{Z}_6$-II models again to a training and a validation set, where this time we assign 33% to the validation set. Additionally, to improve the performance of the decision tree on our fertile islands, we balance the dataset. In more detail, we weight the data points according to their regions R_i, such that the pure numerical superiority of the blue region R_0 does not bias the decision tree.

The whole decision tree consists of 1,887 nodes. As an illustration, Fig. 3 shows an example node of our decision tree. The performance of the decision tree on the validation set estimates how well the rules found by the decision tree generalize to the whole $\mathbb{Z}_6$-II landscape. Having trained the decision tree, we therefore check its performance next. This is done by applying the decision tree to the validation set and counting how many MSSM-like $\mathbb{Z}_6$-II models are mapped to their correct regions. As a result, we obtain the so-called confusion matrix, cf. Table 1. We find that the decision tree performs extremely well, i.e. for most MSSM-like $\mathbb{Z}_6$-II models the region predicted by the decision tree agrees with the actual region.

[2] We thank Robert Helling for raising this question.

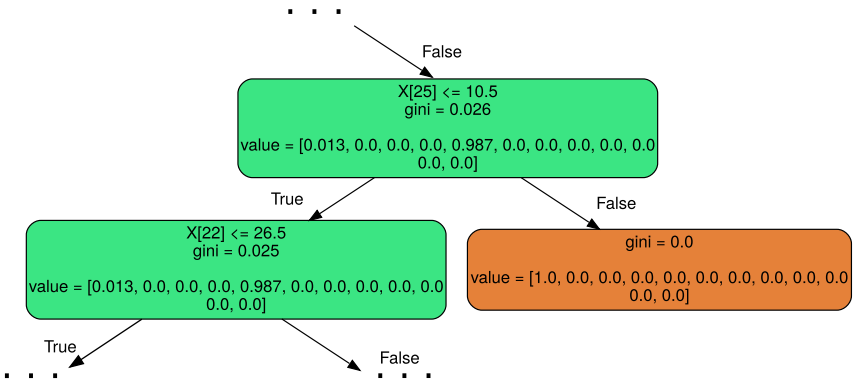

Fig. 3. Example of a decision node. Each node (containing the subset D of the training set) is labeled by a threshold, the Gini index $H(D)$, and the weighted percentages $p_i(D)$ (labeled by "value") of $\mathbb{Z}_6$-II models in this node that belong to the regions R_i for $i = 0, \ldots, 11$. The upper green node is the result from a previous decision. Now, this node enforces a threshold condition $X_{25} \leq 10.5$ on the $\mathbb{Z}_6$-II models that are part of this node. $\mathbb{Z}_6$-II models that do not fulfill this condition are directed to the orange node. This node has a Gini index 0.0 and a value 1.0 for the region R_0. Hence, no further splitting is necessary and we arrive at a so-called leaf node. On the other hand, if a given $\mathbb{Z}_6$-II model satisfies the threshold condition, it is directed to the lower green node. From here further splitting is necessary to separate the models in this node which belong to either region R_0 or R_4. After training the decision tree maps each $\mathbb{Z}_6$-II model to a leaf node and thus gives a prediction for its region R_i using the majority vote obtained from the training set.

Table 1
The confusion matrix of our decision tree evaluated for the validation set. The entries give the number of cases for a certain combination of the region predicted by the decision tree vs. true region that a given $\mathbb{Z}_6$-II model belongs to. For example, there are 11 cases where the decision tree predicted a model to be in region R_0, while the true region was R_1. As the numbers on the diagonal of the confusion matrix are by far larger than the off-diagonal entries, we see that our decision tree works very well.

		Predicted region											
		R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}	R_{11}
True region	R_0	198,994	10	39	10	24	1	7	17	3	16	4	13
	R_1	11	3,107	1	2	0	0	0	0	0	0	0	0
	R_2	19	3	9,667	2	1	0	0	0	0	0	0	0
	R_3	24	2	1	5,256	3	0	0	0	0	0	0	0
	R_4	31	2	4	1	6,430	0	0	0	0	0	0	0
	R_5	0	0	0	0	0	3,138	0	0	0	0	0	0
	R_6	3	0	0	0	0	0	994	0	0	0	0	0
	R_7	15	0	0	0	0	0	0	848	0	0	0	0
	R_8	0	0	0	0	0	0	0	0	1,139	0	0	0
	R_9	10	0	0	0	0	0	0	0	0	1,491	0	0
	R_{10}	2	0	0	0	0	0	0	0	0	0	3,333	0
	R_{11}	10	0	0	0	0	0	0	0	0	0	0	984

4. Evaluation of our results

In the previous section, we described our machine learning workflow and the performance of our algorithm on the validation set. This section is dedicated to determining how well our approach generalizes to the whole $\mathbb{Z}_6$-II landscape. In particular, we are interested in answering the following question: Do the MSSM-like models from the whole $\mathbb{Z}_6$-II landscape also cluster

Table 2
Number of MSSM-like $\mathbb{Z}_6$-II models from either the coarse sample or from the evaluation set within the various regions R_i of the $\mathbb{Z}_6$-II landscape, as predicted by our decision tree.

Region		Coarse sample	Evaluation set	Total
	R_0	0	65	65
Fertile islands	R_1	4	44	48
	R_2	4	17	21
	R_3	1	10	11
	R_4	2	16	18
	R_5	1	5	6
	R_6	1	2	3
	R_7	1	1	2
	R_8	1	1	2
	R_9	1	0	1
	R_{10}	1	11	12
	R_{11}	1	5	6
Total		18	177	195

on fertile islands even though during training the autoencoder neural network had no information of models being MSSM-like or not? How many MSSM-like models within the whole $\mathbb{Z}_6$-II landscape live on the eleven fertile islands and how many models do we lose if we restrict our search to the fertile islands only? To this end, we apply our algorithms to data that has not been considered before, namely to a dataset containing $\mathcal{O}(6{,}300{,}000)$ $\mathbb{Z}_6$-II models, which is hence around nine times as big as the dataset used for the autoencoder and the decision tree so far. We call this set of $\mathbb{Z}_6$-II models the evaluation set. In addition, we also consider a dataset of $\mathcal{O}(30{,}000)$ $\mathbb{Z}_6$-II models from the four patches of the Mini-Landscapes [9,10], in order to see how our approach compares to the search strategy employed there.

4.1. Evaluating the fertility of our islands

In the evaluation set we have 177 MSSM-like models, compared to only 18 in the training and validation sets. The mapping of these models into the chart of the $\mathbb{Z}_6$-II landscape is shown in Fig. 4. Hence, we see that the majority of MSSM-like models lives inside the fertile islands that we identified on the basis of 18 MSSM-like models only. To quantify this statement, we apply our trained decision tree to all $177 + 18 = 195$ MSSM-like $\mathbb{Z}_6$-II models and obtain the predictions as listed in Table 2.

There are MSSM-like $\mathbb{Z}_6$-II models in the evaluation set that are classified by the decision tree to belong to the region R_0, i.e. to the blue region in the chart that seemed to contain no MSSM-like models when considering the 18 MSSM-like $\mathbb{Z}_6$-II models from our coarse sample only. Hence, these models would be "lost" in the sense that they would be missed by our assignment of fertile islands in the chart of the $\mathbb{Z}_6$-II landscape. However, the decision tree maps 130 of all 195 MSSM-like $\mathbb{Z}_6$-II models to the fertile islands. Therefore, having used an extremely small set of only 18 MSSM-models, we reach $2/3$ of the MSSM-like models. We will comment on possible extensions of our approach in order to identify all/more MSSM-like orbifold models in the discussion section 5.

The fertile island R_1 contains in total 48 MSSM-like $\mathbb{Z}_6$-II models, i.e. 25% of all MSSM-like models. On the other hand, this island contains only 1.3% of the whole $\mathbb{Z}_6$-II landscape. Thus,

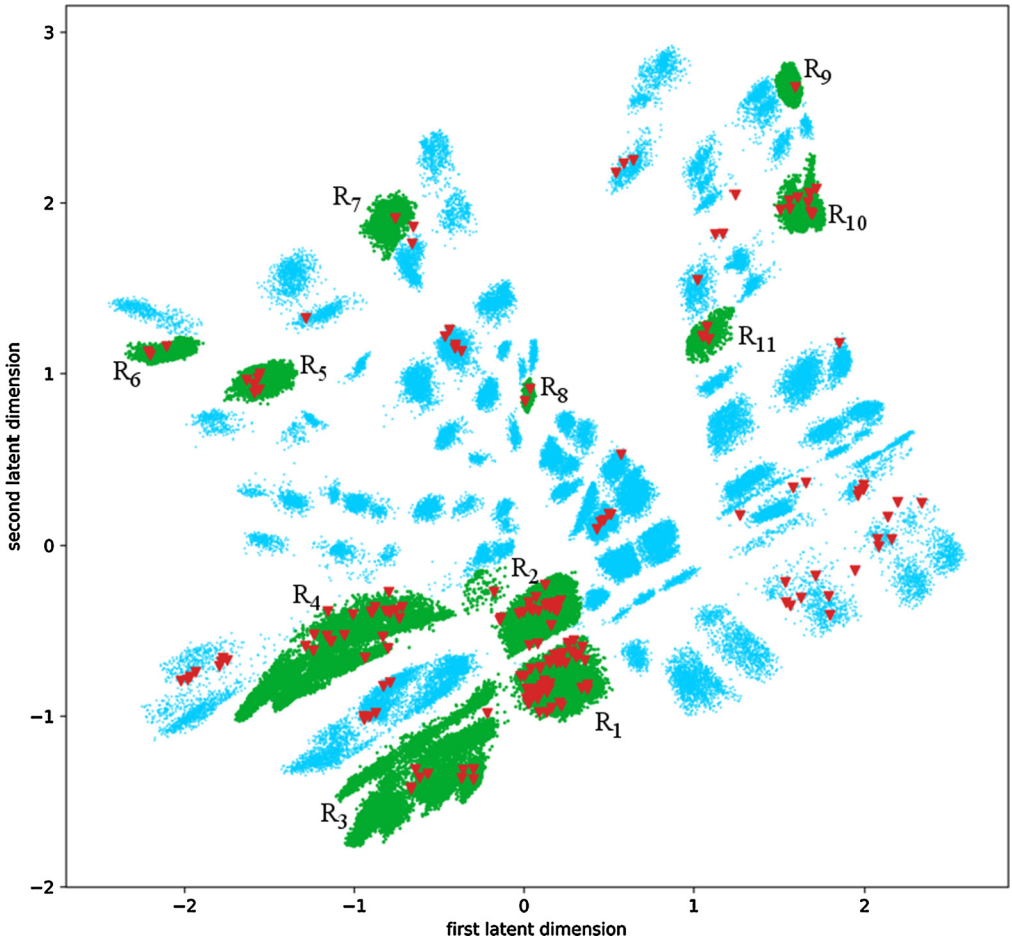

Fig. 4. Location of all 195 MSSM-like models from the evaluation set and the coarse sample (red triangles) within the eleven fertile islands R_i (green) and the whole $\mathbb{Z}_6$-II landscape (blue). Obviously, MSSM-like models prefer the fertile islands that were identified using our coarse sample only.

when searching on this fertile island only, the probability of finding an MSSM-like $\mathbb{Z}_6$-II model is 20 times higher than on a generic spot in the $\mathbb{Z}_6$-II landscape.

4.2. Location of the Mini-Landscape in the chart of the whole $\mathbb{Z}_6$-II landscape

An obvious question is how our approach connects to the Mini-Landscape found in ref. [9,10]. In Fig. 5 we observe that the MSSM-like $\mathbb{Z}_6$-II models from all four different Mini-Landscapes do not spread over the whole chart of the $\mathbb{Z}_6$-II landscape, but rather live on those fertile islands which we had identified using our coarse sample with only 18 MSSM-like $\mathbb{Z}_6$-II models.

Let us also analyze the performance of our decision tree on the MSSM-like $\mathbb{Z}_6$-II models of the Mini-Landscape. As one can infer from Table 3, almost 2/3 of the MSSM-like $\mathbb{Z}_6$-II models from the Mini-Landscape populate the fertile islands. It is interesting to observe that the numbers seem to indicate an approximate association of the two SO(10) patches of the Mini-Landscape to

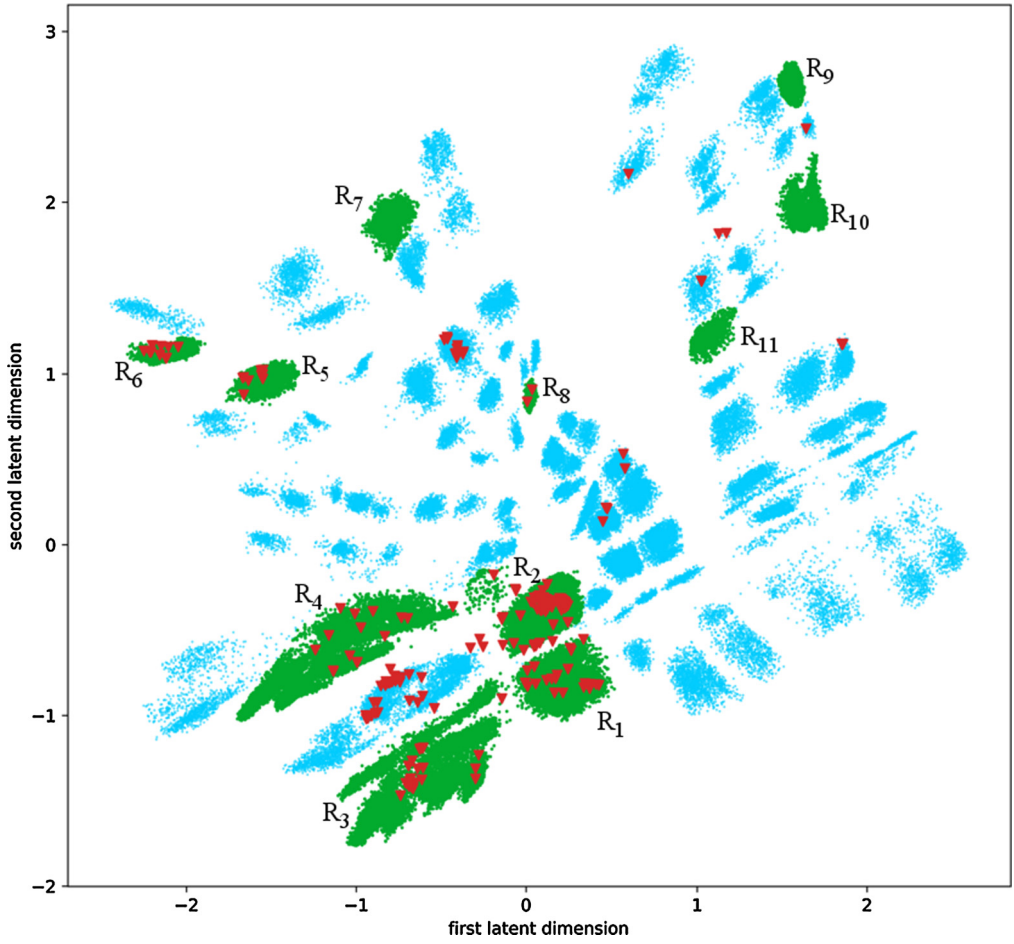

Fig. 5. Location of the MSSM-like models from the Mini-Landscape (red triangles) within the eleven fertile islands R_i (green) and the whole $\mathbb{Z}_6$-II landscape (blue). As in Fig. 4, the MSSM-like models from the Mini-Landscape clearly prefer the fertile islands, especially islands R_1, R_2 and R_3, that were identified using our coarse sample only.

certain regions: in particular, a $\mathbb{Z}_6$-II model with shift vector $V^{\mathrm{SO}(10),1}$ is most likely to be found on the island R_2, while the islands R_1 and R_3 contain most of the $\mathbb{Z}_6$-II models with shift vector $V^{\mathrm{SO}(10),2}$.

5. Discussion

In this work, we have proposed a new search strategy for MSSM-like string models, with the goal of finding an alternative to random searches. The main steps of this strategy are summarized as follows:

1. Create a coarse, random sample of compactification parameters of the landscape under consideration.

Table 3
Number of MSSM-like $\mathbb{Z}_6$-II models from the four patches of the Mini-Landscape dataset (with local GUT shift vectors $V^{\mathrm{SO}(10),1}$, $V^{\mathrm{SO}(10),2}$, $V^{\mathrm{E}_6,1}$ and $V^{\mathrm{E}_6,2}$, see [9]) within the various regions R_i of the $\mathbb{Z}_6$-II landscape as predicted by our decision tree. As before, our approach "finds" around 2/3 of the MSSM-like models.

Region		$V^{\mathrm{SO}(10),1}$	$V^{\mathrm{SO}(10),2}$	$V^{\mathrm{E}_6,1}$	$V^{\mathrm{E}_6,2}$
	R_0	50	37	2	1
Fertile islands	R_1	12	16	0	0
	R_2	60	1	0	0
	R_3	2	24	2	0
	R_4	3	8	4	0
	R_5	10	0	0	0
	R_6	0	8	0	4
	R_7	0	0	0	0
	R_8	0	1	0	1
	R_9	0	0	0	0
	R_{10}	0	0	0	0
	R_{11}	0	0	0	0
% found		64%	61%	75%	83%

2. Train an autoencoder neural network on this sample and draw a two-dimensional chart of the landscape.
3. Identify MSSM-like string models, locate them on the chart of the landscape and define the corresponding fertile islands.
4. Train a decision tree to identify those fertile islands.

This search strategy has been tested successfully at the well-known Mini-Landscape of heterotic $\mathbb{Z}_6$-II orbifold models and we propose that it should be applied to other regions of the string landscape.

In more detail, we have used unsupervised machine learning techniques (i.e. an autoencoder neural network) with the aim to drastically reduce the complexity of model-searches in the heterotic orbifold landscape. In order to do so, it was crucial to find an invariant representation of the compactification parameters given by shifts and Wilson lines. As a result, we were able to draw a two-dimensional chart of the $\mathbb{Z}_6$-II heterotic orbifold landscape. By examining this chart we could verify visually that there are "fertile" islands in the landscape where the density of MSSM-like $\mathbb{Z}_6$-II models is significantly higher than in the remainder of the landscape.

The existence of fertile patches in the $\mathbb{Z}_6$-II landscape was already discovered in the $\mathbb{Z}_6$-II Mini-Landscape [9,10]. However, the fertile patches in the $\mathbb{Z}_6$-II Mini-Landscape were built in by hand, motivated by physical considerations (i.e. the existence of local GUTs like SO(10) or E_6 with complete matter representations). Our complementary search strategy is not based on such considerations, but identifies the fertile islands with MSSM-like models automatically. In particular, the autoencoder neural network was trained without the knowledge of whether a model is MSSM-like or not. Comparing our fertile islands to the Mini-Landscape, we observe that our most promising islands in the landscape consist to some extent of the SO(10) patches of the Mini-Landscapes described in ref. [9,10].

In a second step, we have extracted useful information from the two-dimensional chart of the $\mathbb{Z}_6$-II landscape. To do so, we have employed supervised machine learning, i.e. a so-called

decision tree. This decision tree is trained to predict whether a given $\mathbb{Z}_6$-II model belongs to a certain fertile island of the $\mathbb{Z}_6$-II landscape or not, using easy and fast True-or-False decisions. Thus, in some sense the decision tree is trained to predict the result of the autoencoder and we have shown that this prediction works with very high precision. The benefit of using the decision tree is twofold: Its simple form yields a significant time advantage compared to the autoencoder. Furthermore, the decision tree allows for an easier interpretation compared to the neural network of the autoencoder. These benefits will be used later in the proposed steps 5.a) and 5.b).

We think that our results can provide a valuable guideline when searching for MSSM-like models in the heterotic orbifold landscape: one should first search in the fertile islands that were discovered by a coarse sample of models. To be more specific, we propose to extend our search strategy for MSSM-like models as follows:

5. a) In the traditional approach, the search for MSSM-like orbifold models using the `orbifolder` is divided into three steps: i) a consistent set of shift vector(s) and Wilson lines is created randomly, ii) the spectrum is computed and iii) it is checked whether a given spectrum resembles the MSSM or not. The second and the third steps turn out to be much more time consuming compared to the first step. Now, using our decision tree, it is possible to decide easily whether or not a given set of shift vector(s) and Wilson lines yields an orbifold model inside a fertile island without computing and analyzing the full spectrum. Hence, if an orbifold model turns out to be outside the fertile island, it can be disregarded immediately. Consequently, this step is supposed to be much faster than the traditional one.
5. b) It is conceivable to use the decision tree together with the `orbifolder` in order to generate only consistent sets of shift vector(s) and Wilson lines from the fertile islands in the first place. This exploits the fact that orbifold models are generated step-by-step, i.e. first the shift vector is generated and then Wilson lines are added one by one. Hence, whenever a new shift vector or Wilson line is added, it can be checked quickly whether or not the resulting orbifold model can be inside a fertile island. Again, if the orbifold model fails to be inside such an island, much time can be saved by not further expanding the search in that direction.

As we have seen explicitly, it is not guaranteed that all MSSM-like models reside on those fertile islands of the orbifold landscape that were discovered using the coarse sample of models only (in our example this coarse sample consists of $\mathcal{O}(700{,}000)$ models compared to $\mathcal{O}(7{,}000{,}000)$ models of the full random scan). Hence, one should extend the search algorithm even further. There are many ways how one could proceed:

6. a) One possibility could be to repeat steps 1.– 5. however this time not sampling the full landscape but only the region R_0 outside of the fertile islands. In detail, one creates a new (smaller) coarse, random sample of 4D string models outside the fertile islands and analyzes this region for new fertile islands using a new autoencoder and a new decision tree. This iterative procedure can be repeated until the number of newly identified MSSM-like models goes below a limit to be defined.
6. b) Another possibility could be to combine the new search strategy with the traditional one as follows: In most cases the new search strategy is used to create and analyze models from the fertile islands only. However, in some cases (maybe every hundredth model or so) the traditional approach is used and a fully random model is created and analyzed. If this 4D string model turns out to be MSSM-like and outside the known fertile islands, the decision tree has to be updated (i.e. trained again) such that it includes the newly discovered fertile

island. Then, the search for MSSM-like models is continued by scanning all known fertile islands.

In summary, we think our new search strategy presented in this work may serve as a new paradigm for systematic searches for MSSM-like models in other corners of the string landscape as well. In our case, we could identify a fertile island in the $\mathbb{Z}_6$-II landscape, where the probability of finding an MSSM-like model is 20 times higher than on average. Furthermore, autoencoder neural networks have proven to be an extremely powerful tool in analyzing the string landscape. They can reduce the number of compactification parameters significantly such that one can even draw two-dimensional charts of the string landscape. Surprisingly, MSSM-like models turn out to cluster on separated islands in the string landscape – a fact that has been learned by the autoencoder itself without knowing the definition of the MSSM. Hence, these charts seem to contain a lot of information on the string landscape. However, a full understanding remains an open question. Work along these directions is in progress.

Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (SFB1258). We acknowledge the support by the DFG Cluster of Excellence "Origin and Structure of the Universe", especially by the computing facilities of the Computational Center for Particle and Astrophysics (C2PAP). We would like to thank Fabian Rühle, Christoph Mühlmann, Wolfgang Waltenberger and Robert Helling for useful discussions.

Appendix A. Invariant features of $\mathbb{Z}_6$-II orbifold models

The 64 compactification parameters (i.e. one shift vector and three Wilson lines) needed to specify a single $\mathbb{Z}_6$-II model are not free of ambiguities, i.e. there can be two different sets of compactification parameters that yield exactly the same physical $\mathbb{Z}_6$-II model. In our case, there are two sources for ambiguities: i) There are symmetry transformations acting on the parameters (i.e. lattice translations and Weyl reflections acting on the shifts and the Wilson line) and ii) One can redefine the origin of the orbifold and permute its fixed points systematically. The fact that two seemingly distinct sets of shifts and Wilson lines can yield the same 4D string model can be seen as an equivalence relation. The existence of such equivalences in the dataset is a problem because the network cannot distinguish whether two given models are truly different or differ only up to an equivalence relation. In general, there are two main strategies how to deal with this situation:

1. Amend the training set by transformed compactification parameters, such that the network "learns" that there can exist more than one set of compactification parameters for one and the same 4D string model.
2. Map the compactification parameters of each 4D string model to unique features, i.e. where all equivalence relations are modded out.

As the set of transformations acting on our compactification parameters is huge ($> \mathcal{O}(10^{19})$), the first strategy must be discarded, and we have to transform our original 64 compactification parameters for each 4D string model to unique features, denoted in our case by a 26-dimensional

feature vector X. In this appendix we describe how this can be achieved – however, at the cost that in a few cases two distinct $\mathbb{Z}_6$-II models are mapped to the same feature vector X.

A.1. Invariance under lattice translations and Weyl reflections

As already indicated, our 64 compactification parameters of a $\mathbb{Z}_6$-II model depend on the choice of $E_8 \times E_8$ basis vectors and the addition of arbitrary $E_8 \times E_8$ lattice vectors [34]. Given that the Weyl group for each E_8 is of order $\approx 7 \cdot 10^8$, this is a huge ambiguity. That is, two $\mathbb{Z}_6$-II models with different sets of 64 parameters yield the same 4D string model if their sets of parameters are related by such a symmetry transformation, although their 64 parameters may look (numerically) very different.

We solve this apparent problem by only feeding quantities in our neural network that are manifestly invariant under Weyl transformations and the addition of lattice vectors: from the shift V and the Wilson lines W_3, W_2 and W_2' we compute the so-called local shifts V_g and thereby the number of surviving roots of E_8. This number is invariant.

In detail, we consider the 12 fixed points in the θ-twisted sector of a $\mathbb{Z}_6$-II orbifold (see e.g. [35] for further details). Each fixed point corresponds to a so-called constructing element

$$g_a \;=\; (\theta, n_i^{(a)}\, e_i)\,, \qquad a \;=\; 1,\ldots,12\,, \tag{2}$$

where summation over $i = 1,\ldots,6$ is implied and for certain choices of $n_i^{(a)} \in \mathbb{Z}$. For each constructing element g_a we define the corresponding local shift vector

$$V_{g_a} \;=\; V + (n_3^{(a)} + n_4^{(a)})\, W_3 + n_5^{(a)}\, W_2 + n_6^{(a)}\, W_2'\,. \tag{3}$$

This sixteen-dimensional vector is split into two eight-dimensional vectors $V_{g_a} = (V_{g_a}^{(1)}, V_{g_a}^{(2)})$ corresponding to the first and second E_8 factor. Then, at the fixed point associated to g_a we compute the gauge group $G_a^{(\alpha)}$ ($\alpha = 1, 2$), the so-called local GUT [35,36] as follows: A root vector p of E_8 contributes to the local GUT $G_a^{(\alpha)}$ if

$$V_{g_a}^{(\alpha)} \cdot p \;=\; 0 \bmod 1\,. \tag{4}$$

Note that for the first twisted sector of $\mathbb{Z}_6$-II orbifolds local GUTs can be computed without taking the centralizer of g_a into account. For each of these 24 local GUTs, $G_a^{(\alpha)}$ for $a = 1,\ldots,12$ and $\alpha = 1, 2$, we count the number of non-zero roots p (e.g. 6 for SU(3) and 240 for E_8) and store these numbers in a 24-dimensional vector X of integers, one integer for each E_8 factor at each of the 12 fixed points.

Furthermore, for $\mathbb{Z}_6$-II orbifolds the four-dimensional gauge group $G_{4D} = G_{4D}^{(1)} \times G_{4D}^{(2)}$ is given by the intersection of the 12 local GUTs. Hence, we append the number of surviving roots of the 4D gauge group (i.e. two integers, one integer for each E_8) and, finally, obtain a 26-dimensional feature vector of integers X that is invariant under the addition of $E_8 \times E_8$ lattice vectors and Weyl reflections.

This mapping from 64 compactification parameters to 26 features X does not need to be one-to-one (i.e. injective). Hence, it is worthwhile to check how many different feature vectors X are obtained from all $\mathbb{Z}_6$-II models under consideration. It turns out that our transformation works very well: out of $\mathcal{O}(7{,}000{,}000)$ $\mathbb{Z}_6$-II models, only 0.5% are identified by this transformation.

A.2. Invariance under geometric redefinitions

The feature vector X, introduced in the previous section, is not yet free from all ambiguities: a 4D string model is invariant under i) the exchange of the two E_8 gauge groups and ii) under certain permutations of the fixed points. This results in certain permutations of the first 24 entries of the feature vector X. To be more precise, these permutations are generated as follows (see e.g. [35] for a visualization of the fixed points of the $\mathbb{Z}_6$-II orbifold): In the $\mathbb{Z}_3$ plane, it is possible to shift the origin and to redefine the Wilson line W_3 in such a way that the three fixed points are permuted. The three fixed points in the $\mathbb{Z}_3$ plane correspond to three choices of (n_3, n_4), corresponding to $n_3 + n_4 = 0, 1, 2$ in eq. (3), and in this basis the allowed permutations are generated by the transformations

$$\begin{matrix}(0,0)\\(1,0)\\(1,1)\end{matrix} \mapsto \begin{matrix}(1,0)\\(1,1)\\(0,0)\end{matrix} \quad \text{and} \quad \begin{matrix}(0,0)\\(1,0)\\(1,1)\end{matrix} \mapsto \begin{matrix}(0,0)\\(1,1)\\(1,0)\end{matrix}. \tag{5}$$

This yields the permutation group S_3 (of order 6). In the $\mathbb{Z}_2$ plane, the situation is more involved: here, it is possible to exchange the two Wilson lines $W_2 \leftrightarrow W_2'$, and to shift the origin such that the fixed points are exchanged pairwise. Hence, these permutations are generated by the following permutations of (n_5, n_6)

$$\begin{matrix}(0,0)\\(1,0)\\(0,1)\\(1,1)\end{matrix} \mapsto \begin{matrix}(1,0)\\(0,0)\\(1,1)\\(0,1)\end{matrix}, \quad \begin{matrix}(0,0)\\(1,0)\\(0,1)\\(1,1)\end{matrix} \mapsto \begin{matrix}(0,1)\\(1,1)\\(0,0)\\(1,0)\end{matrix}, \quad \text{and} \quad \begin{matrix}(0,0)\\(1,0)\\(0,1)\\(1,1)\end{matrix} \mapsto \begin{matrix}(0,0)\\(0,1)\\(1,0)\\(1,1)\end{matrix}. \tag{6}$$

These transformations can be summarized as $(n_5, n_6) \mapsto (n_5 + 1, n_6)$, $(n_5, n_6) \mapsto (n_5, n_6 + 1)$ and $(n_5, n_6) \mapsto (n_6, n_5)$ (all modulo 2), respectively. They generate the group D_8 (of order 8), i.e. only a subgroup of the full permutation group S_4 is a symmetry of the generic $\mathbb{Z}_2$ plane.

In summary, the combined symmetry group of the twelve fixed points of the θ-twisted sector of $\mathbb{Z}_6$-II orbifolds is $S_3 \times D_8$ with $6 \times 8 = 48$ elements.

Thus, one and the same string model may be represented by different feature vectors X. We remove these ambiguities by sorting the feature vector X as follows: First, we decide which E_8 is the first and which is the second one, by choosing the E_8 with the lower breaking patterns as the first one. Then, we remove the permutation ambiguity by sorting the 12 local GUTs associated to the 12 fixed points, while respecting the $S_3 \times D_8$ permutation symmetry of the fixed points, in ascending order of the number of surviving roots in the first E_8, with the second E_8 as tiebreak if two values are equal.

Again, this transformation does not need to be one-to-one and (formerly distinct) models can get mapped to the same feature vector X. However, we find that the majority of the $\mathcal{O}(7{,}000{,}000)$ $\mathbb{Z}_6$-II models yield distinct feature vectors X, i.e. 84% of the models are mapped to distinct feature vectors X.

References

[1] W. Lerche, D. Lüst, A.N. Schellekens, Nucl. Phys. B 287 (1987) 477.
[2] M.R. Douglas, J. High Energy Phys. 05 (2003) 046, arXiv:hep-th/0303194 [hep-th].
[3] A.E. Faraggi, Nucl. Phys. B 387 (1992) 239, arXiv:hep-th/9208024 [hep-th].
[4] T.P.T. Dijkstra, L.R. Huiszoon, A.N. Schellekens, Nucl. Phys. B 710 (2005) 3, arXiv:hep-th/0411129 [hep-th].
[5] V. Braun, Y.-H. He, B.A. Ovrut, T. Pantev, Phys. Lett. B 618 (2005) 252, arXiv:hep-th/0501070 [hep-th].

[6] F. Gmeiner, R. Blumenhagen, G. Honecker, D. Lüst, T. Weigand, J. High Energy Phys. 01 (2006) 004, arXiv:hep-th/0510170 [hep-th].
[7] K.R. Dienes, Phys. Rev. D 73 (2006) 106010, arXiv:hep-th/0602286 [hep-th].
[8] R. Blumenhagen, B. Körs, D. Lüst, S. Stieberger, Phys. Rep. 445 (2007) 1, arXiv:hep-th/0610327 [hep-th].
[9] O. Lebedev, H.P. Nilles, S. Raby, S. Ramos-Sánchez, M. Ratz, P.K.S. Vaudrevange, A. Wingerter, Phys. Lett. B 645 (2007) 88, arXiv:hep-th/0611095 [hep-th].
[10] O. Lebedev, H.P. Nilles, S. Ramos-Sánchez, M. Ratz, P.K.S. Vaudrevange, Phys. Lett. B 668 (2008) 331, arXiv:0807.4384 [hep-th].
[11] L.B. Anderson, J. Gray, A. Lukas, E. Palti, Phys. Rev. D 84 (2011) 106005, arXiv:1106.4804 [hep-th].
[12] L.B. Anderson, J. Gray, A. Lukas, E. Palti, J. High Energy Phys. 06 (2012) 113, arXiv:1202.1757 [hep-th].
[13] D.K. Mayorga Peña, H.P. Nilles, P.-K. Oehlmann, J. High Energy Phys. 12 (2012) 024, arXiv:1209.6041 [hep-th].
[14] H.P. Nilles, P.K.S. Vaudrevange, Mod. Phys. Lett. A 30 (10) (2015) 1530008, arXiv:1403.1597 [hep-th].
[15] M. Cvetič, D. Klevers, D.K. Mayorga Peña, P.-K. Oehlmann, J. Reuter, J. High Energy Phys. 08 (2015) 087, arXiv:1503.02068 [hep-th].
[16] Y. Olguín-Trejo, R. Pérez-Martínez, S. Ramos-Sánchez, Phys. Rev. D 98 (2018) 106020, arXiv:1808.06622 [hep-th].
[17] S. Abel, J. Rizos, J. High Energy Phys. 08 (2014) 010, arXiv:1404.7359 [hep-th].
[18] J. Carifio, W.J. Cunningham, J. Halverson, D. Krioukov, C. Long, B.D. Nelson, Phys. Rev. Lett. 121 (10) (2018) 101602, arXiv:1711.06685 [hep-th].
[19] Y.-H. He, arXiv:1706.02714 [hep-th].
[20] D. Krefl, R.-K. Seong, Phys. Rev. D 96 (6) (2017) 066014, arXiv:1706.03346 [hep-th].
[21] F. Ruehle, J. High Energy Phys. 08 (2017) 038, arXiv:1706.07024 [hep-th].
[22] J. Carifio, J. Halverson, D. Krioukov, B.D. Nelson, J. High Energy Phys. 09 (2017) 157, arXiv:1707.00655 [hep-th].
[23] Y.-N. Wang, Z. Zhang, J. High Energy Phys. 08 (2018) 009, arXiv:1804.07296 [hep-th].
[24] K. Bull, Y.-H. He, V. Jejjala, C. Mishra, Phys. Lett. B 785 (2018) 65, arXiv:1806.03121 [hep-th].
[25] D. Klaewer, L. Schlechter, Phys. Lett. B 789 (2019) 438–443, arXiv:1809.02547 [hep-th].
[26] G.E. Hinton, R.R. Salakhutdinov, Science 313 (5786) (2006) 504, http://science.sciencemag.org/content/313/5786/504.full.pdf.
[27] L.J. Dixon, J.A. Harvey, C. Vafa, E. Witten, Nucl. Phys. B 261 (1985) 678.
[28] L.J. Dixon, J.A. Harvey, C. Vafa, E. Witten, Nucl. Phys. B 274 (1986) 285.
[29] H.P. Nilles, S. Ramos-Sánchez, P.K.S. Vaudrevange, A. Wingerter, Comput. Phys. Commun. 183 (2012) 1363, arXiv:1110.5229 [hep-th], Web page: http://projects.hepforge.org/orbifolder/.
[30] K.R. Dienes, M. Lennek, Phys. Rev. D 75 (2007) 026008, arXiv:hep-th/0610319 [hep-th].
[31] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: large-scale machine learning on heterogeneous systems, Software available from: tensorflow.org, 2015.
[32] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, CoRR, arXiv:1706.02515, 2017.
[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, J. Mach. Learn. Res. 12 (2011) 2825.
[34] J.A. Casas, M. Mondragon, C. Muñoz, Phys. Lett. B 230 (1989) 63.
[35] W. Buchmüller, K. Hamaguchi, O. Lebedev, M. Ratz, Nucl. Phys. B 712 (2005) 139, arXiv:hep-ph/0412318 [hep-ph].
[36] S. Förste, H.P. Nilles, P.K.S. Vaudrevange, A. Wingerter, Phys. Rev. D 70 (2004) 106008, arXiv:hep-th/0406208 [hep-th].