Deep Autoencoders in the Heterotic Orbifold Landscape

Andreas Mütter

Technical University of Munich

SUSY 2019
Corpus Christi, May 22nd 2019

in collaboration with

Erik Parr and Patrick Vaudrevange

arxiv:1811.05993

SFB 1258 Neutrinos Dark Matter Messengers

Motivation

The search for MSSMs in String Theory has a long history

- Type IIA/B with D-branes
- F-Theory
- Heterotic orbifolds

MSSM searches in heterotic orbifolds

- need to fix ~ 100 parameters
- usually done by random searches

Motivation

The search for MSSMs in String Theory has a long history

- Type IIA/B with D-branes
- F-Theory
- Heterotic orbifolds

MSSM searches in heterotic orbifolds

- need to fix ~ 100 parameters
- usually done by random searches

Very inefficient!

Outline

Goals of this talk

- Have a careful look at the parameter space of heterotic orbifold models
- Understand the connection between choices of parameters and "good" models
- Use this information to refine the usual random searches for MSSMs

Outline

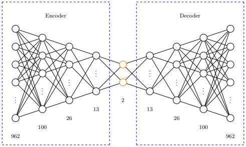
Goals of this talk

- Have a careful look at the parameter space of heterotic orbifold models
- Understand the connection between choices of parameters and "good" models
- Use this information to refine the usual random searches for MSSMs

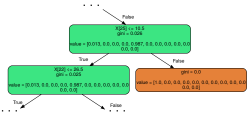
in order to achieve this: **Machine learning**

ORBIFOLDER

Data generation
Data preparation

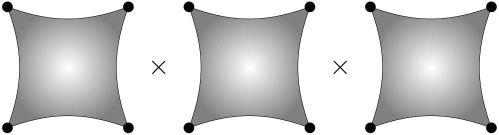


Neural Network
(find fertile patches)



Decision tree
(extract conditions)

Parameter Space of (Heterotic) Orbifolds

$\mathbb{R}^{1,3} \times$ 

Orbifolds are characterized by the *space group*

$$g \;=\; (\theta \,|\, n_\alpha \, e_\alpha)$$

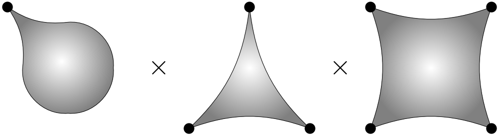
For the heterotic string: Embed into $\mathrm{E}_8 \times \mathrm{E}_8$
(e.g. using the `orbifolder` [1110.5229])

$$\left.\begin{array}{cccc} v_\theta & \to & V_\theta & \textbf{Shift} \\ e_\alpha & \to & W_\alpha & \textbf{Wilson lines} \end{array}\right\} \in \mathbb{Q}^{16}$$

subject to *modular invariance conditions*

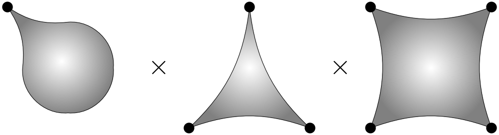
$\to$ spectrum, local gauge groups (local GUTs) fully determined

Compactification Parameters in $\mathbb{Z}_6$-II



- Geometric twist fixed as $v_\theta = \left(\frac{1}{6}, \frac{1}{3}, -\frac{1}{2}\right)$
- Gauge embeddings:
  - 1 shift of up to order 6
  - 1 order 3 Wilson line, 2 order 2 Wilson lines
- Local shifts: $V_g = kV_\theta + (n_3 + n_4)W_3 + n_5 W_5 + n_6 W_6$

Compactification Parameters in $\mathbb{Z}_6$-II

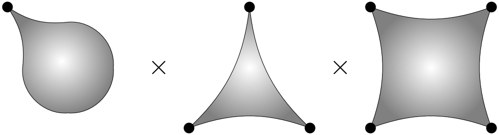


- Geometric twist fixed as $v_\theta = \left(\frac{1}{6}, \frac{1}{3}, -\frac{1}{2}\right)$
- Gauge embeddings:
  - 1 shift of up to order 6
  - 1 order 3 Wilson line, 2 order 2 Wilson lines
- Local shifts: $V_g = kV_\theta + (n_3 + n_4)W_3 + n_5 W_5 + n_6 W_6$

Main motivation here

Mini-Landscape: [hep-th/0611095]
"Sweet spots" in the landscape,
based on physics considerations
(local GUTs w/full representations)

Compactification Parameters in $\mathbb{Z}_6$-II



- Geometric twist fixed as $v_\theta = \left(\frac{1}{6}, \frac{1}{3}, -\frac{1}{2}\right)$
- Gauge embeddings:
  - 1 shift of up to order 6
  - 1 order 3 Wilson line, 2 order 2 Wilson lines
- Local shifts: $V_g = kV_\theta + (n_3 + n_4)W_3 + n_5 W_5 + n_6 W_6$

Main motivation here

Mini-Landscape: [hep-th/0611095]
"Sweet spots" in the landscape, based on physics considerations (local GUTs w/full representations)

Machine Learning

reproduce and extend the idea of the Mini-Landscape in an automated fashion

Redundant Data: Weyl Reflections and Lattice Vectors

Naive idea: use shift + 3 Wilson lines (= 64 parameters) as input

but ...

Redundant Data: Weyl Reflections and Lattice Vectors

Naive idea: use shift + 3 Wilson lines (= 64 parameters) as input

but ...

There are various transformations that leave the physical projection conditions invariant

Weyl reflections	$V \mapsto V - 2\frac{\alpha_I \cdot V}{\alpha_I \cdot \alpha_I}\alpha_I$
addition of lattice vectors	$V \mapsto V + \lambda$

Problem: equivalent inputs can look completely different numerically

Redundant Data: Weyl Reflections and Lattice Vectors

Naive idea: use shift + 3 Wilson lines (= 64 parameters) as input

but ...

There are various transformations that leave the physical projection conditions invariant

Weyl reflections $$V \;\mapsto\; V - 2\frac{\alpha_I \cdot V}{\alpha_I \cdot \alpha_I}\alpha_I$$

addition of lattice vectors $$V \;\mapsto\; V + \lambda$$

Problem: equivalent inputs can look completely different numerically

Solution: use *breaking patterns* (= number of invariant roots)

Breaking patterns

For each gauge embedding V count the number of roots p that fulfill

$$V \cdot p = 0 \mod 1$$

Individually for each E_8, so each gauge embedding becomes a tuple of 2 integers

Breaking patterns

For each gauge embedding V count the number of roots p that fulfill

$$V \cdot p \ = \ 0 \quad \text{mod } 1$$

Individually for each E_8, so each gauge embedding becomes a tuple of 2 integers

Idea: Look at local gauge groups (local GUTs) in the θ twisted sector:

12 different local gauge groups

$$V_g = V_\theta + (n_3 + n_4) W_3 + n_5 W_5 + n_6 W_6$$

Breaking patterns

For each gauge embedding V count the number of roots p that fulfill

$$V \cdot p \;=\; 0 \mod 1$$

Individually for each E_8, so each gauge embedding becomes a tuple of 2 integers

Idea: Look at local gauge groups (local GUTs) in the θ twisted sector:

12 different local gauge groups

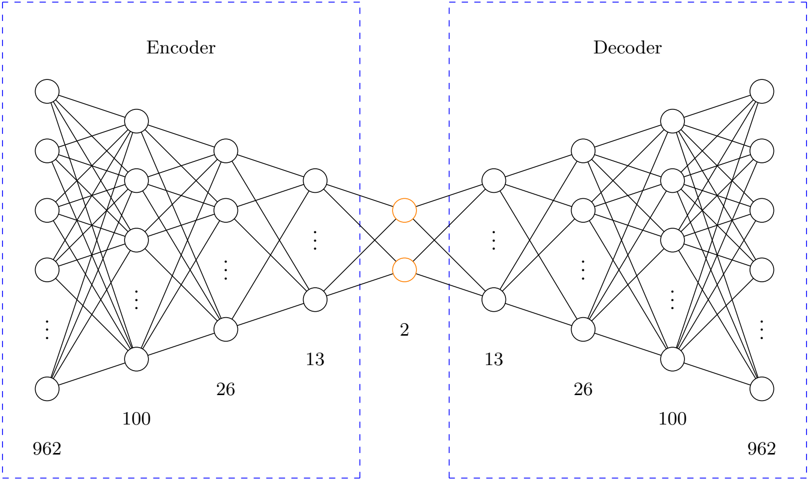
$$V_g = V_\theta + (n_3 + n_4) W_3 + n_5 W_5 + n_6 W_6$$

- Amend local breaking patterns by global gauge group $\rightarrow$ $2 \times 12 \;+\; 2 \;=\; 26$ parameters
- For later purposes: one-hot encoding $\rightarrow$ $26 \times 37 = 962$ input dimensions

The Dataset(s)

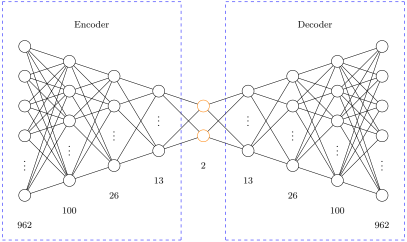
- **Training:** 700k from random search, by chance including some MSSMs
  $\rightarrow$ we will use this dataset in order to identify fertile patches
- **Evaluation:** 6.3M from random search
  $\rightarrow$ are our fertile patches chosen correctly?
- 30k from the **Mini-Landscape**
  $\rightarrow$ compare our approach to physics-motivated considerations

Autoencoders



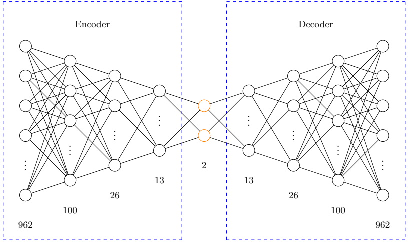
Autoencoders

- Feed inputs through network
- Train on `output=input`
- Information bottleneck in the latent layer



Autoencoders

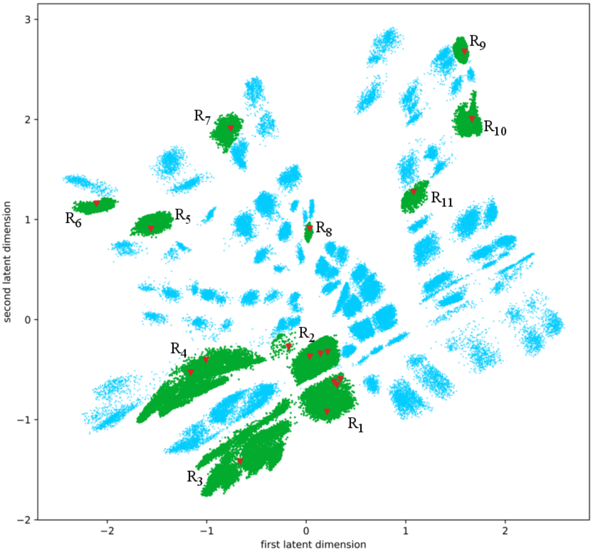
- Feed inputs through network
- Train on `output=input`
- Information bottleneck in the latent layer



Because of the information bottleneck, the network is forced to learn a compressed representation of the input by exploiting (nonlinear) redundancies

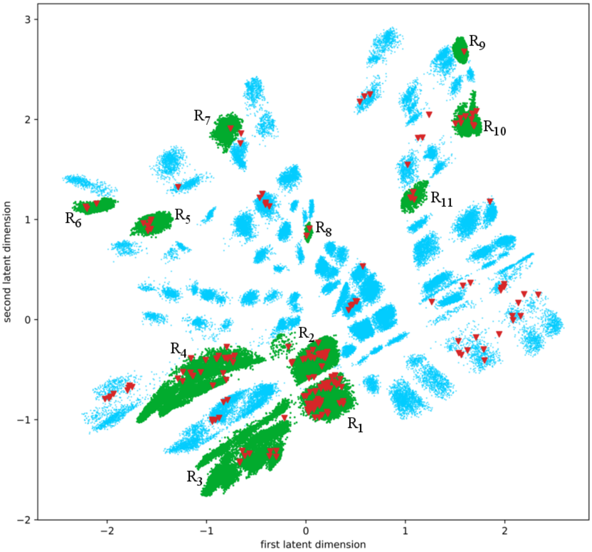
Draw a map: after training, use the first half of the network (=the encoder) to draw a 2-d map of the landscape

Results



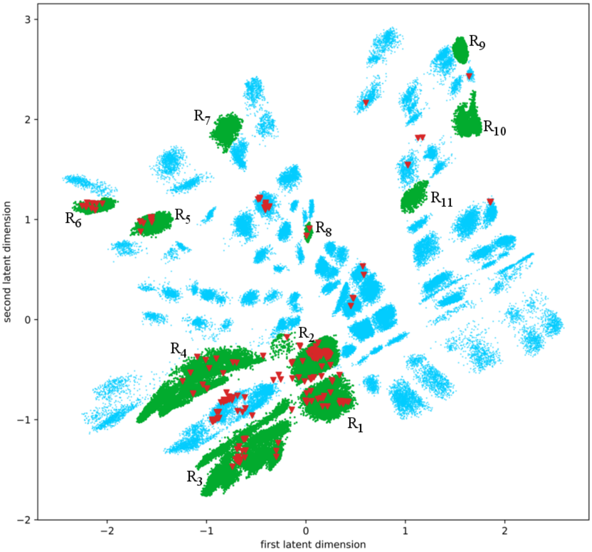
Training set: MSSMs in red, random models in blue, fertile islands in green

Results



Evaluation set: MSSMs in red, random models in blue, fertile islands in green

Results



Mini-Landscape: MSSMs in red, random models in blue, fertile islands in green

Decision Tree

Question: How can I tell the different clusters apart?

Decision Tree

Question: How can I tell the different clusters apart?
Decision tree: Classify clusters according to specifications like

> *If an entry x in the feature vector has value below T_x and an entry y in the feature vector has value below T_y and ...,*
> *then this data point belongs to cluster R_i.*

Decision Tree

Question: How can I tell the different clusters apart?
Decision tree: Classify clusters according to specifications like

> *If an entry x in the feature vector has value below T_x and an entry y in the feature vector has value below T_y and ...,*
> *then this data point belongs to cluster R_i.*

Main advantages:

- Almost trivial to feed information gathered in this way to a computer
- A first step to an interpretation in terms of physical quantities

Two direct applications

Two direct applications

Shortcuts during random search
The `orbifolder` works as follows

(i) Generate random, modular invariant gauge embedding
(ii) Compute spectrum
(iii) Check whether spectrum contains an MSSM

Using the decision tree, we can skip steps (ii) and (iii)

Two direct applications

Shortcuts during random search
The `orbifolder` works as follows

(i) Generate random, modular invariant gauge embedding
(ii) Compute spectrum
(iii) Check whether spectrum contains an MSSM

Using the decision tree, we can skip steps (ii) and (iii)

Generation of fertile models
Possible algorithm

(i) Generate random shift
(ii) Check whether it has the chance to be in a fertile patch:
    - if no, scrap the model
    - if yes, create random Wilson line
(iii) Repeat until one has a complete model

Conclusions

A role model for other orbifold geometries:

- Data preprocessing

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries
- Draw a map of the landscape using an autoencoder

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries
- Draw a map of the landscape using an autoencoder
- Identify fertile patches: every cluster containing an MSSM is a candidate

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries
- Draw a map of the landscape using an autoencoder
- Identify fertile patches: every cluster containing an MSSM is a candidate
- Use a decision tree in order to extract information on the fertile patches

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries
- Draw a map of the landscape using an autoencoder
- Identify fertile patches: every cluster containing an MSSM is a candidate
- Use a decision tree in order to extract information on the fertile patches
- Find more MSSMs ...

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries
- Draw a map of the landscape using an autoencoder
- Identify fertile patches: every cluster containing an MSSM is a candidate
- Use a decision tree in order to extract information on the fertile patches
- Find more MSSMs ...

Here, we have applied this method to the well-studied $\mathbb{Z}_6$-II orbifold and evaluated the results using both a very large dataset and models from the physically motivated Mini-Landscape

Conclusions

A role model for other orbifold geometries:

- Data preprocessing
  - Generate a coarse sample
  - Remove any redundancy in the data due to symmetries
- Draw a map of the landscape using an autoencoder
- Identify fertile patches: every cluster containing an MSSM is a candidate
- Use a decision tree in order to extract information on the fertile patches
- Find more MSSMs ...

Here, we have applied this method to the well-studied $\mathbb{Z}_6$-II orbifold and evaluated the results using both a very large dataset and models from the physically motivated Mini-Landscape

Thank You!