UPPSALA
UNIVERSITET

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 2138

Heterotic Compactifications in the Era of Data Science

ROBIN SCHNEIDER

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2022

ISSN 1651-6214
ISBN 978-91-513-1477-8
URN urn:nbn:se:uu:diva-471719

Dissertation presented at Uppsala University to be publicly examined in Häggsalen, 10132, Ångström, Lägerhyddsvägen 1, Uppsala, Friday, 20 May 2022 at 09:00 for the degree of Doctor of Philosophy. The examination will be conducted in English. Faculty examiner: Professor Yang-Hui He (Royal Institution, London Institute for Mathematical Sciences and City University of London, Department of Mathematics and University of Oxford, Merton College and NanKai University, School of Physics).

Abstract

Schneider, R. 2022. Heterotic Compactifications in the Era of Data Science. *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 2138. 105 pp. Uppsala: Acta Universitatis Upsaliensis. ISBN 978-91-513-1477-8.

The goal of this thesis is to review and investigate recent applications of machine learning to problems in string theory. String theory, the leading candidate for a unification of gravity and the standard model of particle physics, requires the introduction of additional space-time dimensions. To match experimental observations of our universe, these additional dimensions need to be curled up on a compact space. The most common choice to describe this compact space are manifolds of Calabi-Yau type. These manifolds come with favourable mathematical and phenomenological properties.

In the first half of this thesis Calabi-Yau manifolds, which are complex Kähler manifolds admitting a Ricci-flat metric, are introduced. The popular construction as complete intersections in products of complex projective space is explained and the necessary mathematical machinery to compute their topological quantities presented. This part is followed by a review of machine learning applications to study their Hodge numbers and the cohomologies of line bundles. In a next step the new Python library cymetric is presented for modeling numerical approximations of the unknown Ricci-flat metric. The metric tensor is a required component in the calculation of Yukawa couplings. It is learned by a neural network trained against a custom loss function, that encodes all the necessary mathematical properties.

In the second half Calabi-Yau manifolds are used to compactify the heterotic string and con-struct standard model like vacua. Those are vacua which match the particle content and gaugegroup of a supersymmetric extension of the standard model. First, the popular compactificationprocedure utilising line bundle sums is reviewed and applied to the newly discovered construc-tions of generalized complete intersection Calabi-Yau manifolds. Second, an exploration ofsuch models is initiated in so far uncharted territories. This includes two Calabi-Yau manifoldswith more than 7 Kähler moduli, which are beyond systematic computational reach. In total19538 new models are found by using Actor-Critic agents from deep reinforcement learning.

Keywords: Calabi-Yau, string compactifications, machine learning

Robin Schneider, Department of Physics and Astronomy, Theoretical Physics, Box 516, Uppsala University, SE-751 20 Uppsala, Sweden.

ISSN 1651-6214
ISBN 978-91-513-1477-8
URN urn:nbn:se:uu:diva-471719 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-471719)

Dedicated to all hard-working
doctoral students at Uppsala University

List of papers

This thesis is based on the following papers, which are referred to in the text by their Roman numerals.

I M. Larfors and R. Schneider, "Line bundle cohomologies on CICYs with Picard number two," *Fortsch. Phys.* **67** no. 12, (2019) 1900083, arXiv:1906.00392 [hep-th].

II M. Larfors and R. Schneider, "Explore and Exploit with Heterotic Line Bundle Models," *Fortsch. Phys.* **68** no. 5, (2020) 2000034, arXiv:2003.04817 [hep-th].

III M. Larfors, D. Passaro, and R. Schneider, "Heterotic Line Bundle Models on Generalized Complete Intersection Calabi Yau Manifolds," *JHEP* **05** (2021) 105, arXiv:2010.09763 [hep-th].

IV H. Erbin, R. Finotello, R. Schneider, and M. Tamaazousti, "Deep multi-task mining Calabi–Yau four-folds," *Mach. Learn. Sci. Tech.* **3** no. 1, (2022) 015006, arXiv:2108.02221 [hep-th].

V M. Larfors, A. Lukas, F. Ruehle, and R. Schneider, "Learning Size and Shape of Calabi-Yau Spaces," *Machine Learning and the Physical Sciences, Workshop 35th NeurIPS* (12, 2021) , arXiv:2111.01436 [hep-th].

Papers not included in this thesis.

VI L. B. Anderson, J. Gray, M. Larfors, M. Magill, and R. Schneider, "Generalized Vanishing Theorems for Yukawa Couplings in Heterotic Compactifications," *JHEP* **05** (2021) 085, arXiv:2103.10454 [hep-th].

Reprints were made with permission from the publishers.

Contents

1. Introduction

The standard model of particle physics is arguably the most successful theory matching experimental observations. For example the best precision measurement of the fine-structure constant is within 10^{-8} of the theoretical prediction [1]. The standard model describes three of the four fundamental forces. Those are, the electromagnetic, the weak, and the strong force. Its mathematical description is encoded as a quantum field theory, realising particles as excitations of quantised fields and their interactions via exchange of gauge bosons.

The fourth and last fundamental force is gravity. It is mathematically described by Einstein's theory of general relativity. It accurately predicts the physics of our universe at large scale, such as the movement of planets in our solar system. It accounts for the mysterious dark energy responsible for the expansion of the universe by introducing a tiny positive cosmological constant. Moreover, when introducing additional mass via dark matter it accurately describes the dynamics of galaxies.

Together the two theories yield good predictions at their respective scales. A problem, however, arises when trying to combine the theories in a unifying framework. This is not only desired from an aesthetical point of view, but also necessary to solve the mysteries and inconsistencies of black holes, such as the information paradox. A combined theory of quantum gravity is needed to explain how information is preserved when a black hole emits energy through Hawking radiation. General relativity without serious modifications can not be quantised in four dimensions, which is a necessary step for unification of the four forces.

String theory is the leading candidate for a unifying theory of gravity and particle physics. Infinities plaguing the quantisation of gravity in four dimensions cancel exactly in string theory by introducing one-dimensional objects called strings. It is then possible to write down a consistent theory of quantised gravity, but at the costs of string theories own caveats.

These caveats include the introduction of additional space-time dimensions. The bosonic string requires a total of 26 dimensions. Fermions are realised by introducing an additional symmetry, called supersymmetry, matching a fermionic superpartner to any boson. A supersymmetric string theory is only consistent in ten dimensions. These additional dimensions are usually treated as a tiny, smaller than the energy range probed by the Large Hadron Collider, compact manifold. Consistency with low energy observations then puts several restricting constraints on the shape of this compact space. Nevertheless,

there are billions of manifolds which can still lead to semi-realistic compactifications. A second problem is the positive sign and small size of the cosmological constant, which does not naturally arise in the string theory context. There exist proposals for resolving and fine-tuning this problem, but also arguments why de Sitter space can not be realised straightforwardly in supergravity approximations of string theory. This active research direction goes under the name of the swampland program.

Research in string theory has not only led to progress in theoretical physics, but also in various areas of mathematics. The study of compact manifolds is one of those and will be the main focus of this thesis. Physical and phenomenological constraints such as the requirement of minimal supersymmetry or the existence of three fermion generations have strong implications on the topology of the compact manifold. The most successful compactifications are based on so called Calabi-Yau manifolds, which are studied in the first part of this thesis. In the second part these manifolds are utilised to construct string compactifications, which match the particle content and gauge group of the standard model. In both parts algorithms from machine learning are used to gain new insights of the physics on these manifolds. Machine learning can simplify resource expensive computations and find efficient algorithms to parse different string vacua generated by the various possibilities for the compact space. In the context of this thesis I will use these tools to shed new light on the old problem of finding interesting string theory compactifications.

1.1 Calabi-Yau compactifications

The journey to the first appearance of Calabi-Yau manifolds in physics starts with the heterotic string [2–4]. They also naturally arise when starting from other superstring theories, but the main physics applications of this thesis will be dealing with the heterotic string so we will follow the original historical records. Consider the heterotic string with gauge group $E_8 \times E_8$. The Lagrangian of this theory is given by coupling $N = 1$ supergravity to $N = 1$ super Yang-Mills theory in 10 dimensions:

$$\begin{aligned} e^{-1}L = &-\frac{1}{2\kappa_{10}}R - \frac{1}{4g^2\phi}\mathrm{tr}F_{MN}^2 - \frac{1}{\kappa_{10}^2}(\partial_M\phi/\phi)^2 - \frac{3\kappa_{10}^2}{8g^4\phi^2}H_{MNP}^2 \\ &+ \text{ fermion terms }. \end{aligned} \tag{1.1}$$

Here g is the Yang-Mills coupling constant, ϕ the dilaton, R encodes the curvature of space-time, F is the field strength of the non-abelian gauge group, and H is the field-strength of the antisymmetric B-field. H has to satisfy the modified Bianchi identity

$$dH = \frac{\alpha'}{4}\left(\mathrm{tr}(R\wedge R) - \mathrm{tr}(F\wedge F)\right), \tag{1.2}$$

where α' is the only free parameter and given in terms of the string length $l_s = \sqrt{\alpha'}$. The heterotic sting is interesting because a single of the E_8 factors already contains the most common Grand Unified Theory (GUT) gauge groups $E_6, SO(10)$ and $SU(5)$. The supersymmetry transformations of the fermionic fields λ, χ, ψ are given by [4]

$$\delta\psi_M = \frac{1}{\kappa} D_M \eta + \frac{\kappa}{32 g^2 \phi} (\Gamma_M^{NPQ} - 9\delta_M^N \Gamma^{PQ}) \eta H_{NPQ} + (\text{fermi})^2 = 0 \quad (1.3a)$$

$$\delta\chi^a = -\frac{1}{4g\sqrt{\phi}} \Gamma^{MN} F^a_{MN} \eta + (\text{fermi})^2 = 0 \quad (1.3b)$$

$$\delta\lambda = -\frac{1}{\sqrt{2}\phi} (\Gamma \cdot \partial\phi)\eta + \frac{\kappa}{8\sqrt{2} g^2 \phi} \Gamma^{MNP} \eta H_{MNP} + (\text{fermi})^2 = 0 \quad (1.3c)$$

where η is the ten-dimensional Majorana-Weil spinor of the supersymmetric variation and $\Gamma^{MN} = \frac{1}{2}(\Gamma^M \Gamma^N - \Gamma^N \Gamma^M)$ are antisymmetric products of ten-dimensional gamma matrices. To make the connection to our four dimensional universe we decompose the ten-dimensional space-time into a direct product $\mathscr{M}_{10} = \mathscr{M}_4 \times X$ of a four-dimensional maximally symmetric space $\mathscr{M}_4$ and a compact manifold X. As a consequence the ten dimensional spinor η splits into six and four dimensional parts. Imposing the further simplifying assumptions[1], that H vanishes, and that the dilaton ϕ is constant, one arrives at the following supersymmetric transformations in six dimensions [7]:

$$0 = \delta\psi_i = D_i \eta_6 \quad (1.4)$$

$$0 = \delta\chi^a = \Gamma^{ij} F^a_{ij} \eta_6 \quad (1.5)$$

where i, j are the compact indices and run from $1, \ldots, 6$ and η_6 is the six-dimensional supersymmetry generating spinor. The spinor fields must have a complex representation to match our experimental observations and thus allows for at most $N = 1$ supersymmetry. $N = 1$ supersymmetry comes with some desirable properties, such as natural dark matter candidates, the unification of gauge coupling constants at high energy scales, and solving the hierarchy problem. Nevertheless it has not been observed yet and hence if it exists must be broken at unprobed energy scales. In the rest of this thesis we will investigate low energy vacua of $N = 1$ supersymmetry and leave the questions of breaking supersymmetry unanswered for future work. The first equation (1.4) says that the supersymmetry generator η_6 has to be covariantly constant, which in turn implies the integrability condition

$$\Gamma^k R_{ik} \eta_6 = 0. \quad (1.6)$$

The ten-dimensional version of this equation combined with the assumption that $\mathscr{M}_4$ is maximally symmetric, implies the vanishing of the cosmological

[1]Not imposing these assumptions leads to a different system of more complicated equations called the Strominger-Hull system [5, 6].

constant[2]. The condition (1.6) also states that there has to be exactly one spinor which is covariantly constant in order to preserve $N = 1$ supersymmetry.
The spin connection ω on X is an $SO(6) \cong SU(4)$ gauge field. In general a field parallel transported around a contractible curve γ is acted on by some group element U of the holonomy group H, i.e. $\psi \to U\psi$, with $U \in H$. Now in order for $U\eta_6 = \eta_6$, we need to look for a subgroup of $SU(4)$ which leaves η_6 invariant. Since η_6 is a spinor field it transforms as $\mathbf{4}$ or $\bar{\mathbf{4}}$ depending on its chirality. Acting with an $SU(4)$ transformation on η_6 it can be brought into the following form

$$\eta_6 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \eta_0 \end{pmatrix}. \tag{1.7}$$

This makes it obvious that an $SU(3)$ transformation acting on the first three components leaves η_6 invariant, so if the holonomy group is $H = SU(3)$ there exists exactly one covariantly constant spinor. Complex, compact, Kähler manifolds of dimension n with holonomy group $SU(n)$ are commonly called Calabi-Yau manifolds. Calabi-Yau manifolds are not only interesting for physicists, but come with fascinating properties studied by mathematicians such as mirror symmetry [9].

1.2 Outline

The outline of this thesis is as follows: In chapter 2 the usage of machine learning to tackle problems in modern string theory is motivated. A short non exhaustive overview of recent progress at the intersection of these two fields is given in section 2.1. The next section 2.2 provides an introduction to neural networks and in section 2.3 the setting of reinforcement learning is explained.

In part I we cover all things Calabi-Yau (CY) related. These manifold are defined in chapter 3 which contains a brief discussion of their most common constructions. The defining property of a CY manifold is their unique Ricci-flat metric tensor, which is discussed in section 3.1. Complete Intersection Calabi-Yau (CICY) manifolds will be the targets of most studies presented in this thesis. They are introduced in section 3.2, a discussion of their topological invariants is included in section 3.2.1, the necessary machinery to compute these quantities can be found in section 3.2.2 and their recent generalised construction (gCICY) is given in section 3.2.3.

The next chapter 4 compares various studies investigating the Hodge numbers of tangent space cohomologies of CICYs in section 4.1. It includes a

[2]There are proposals for obtaining a small positive constant [8].

review of Inception-blocks (section 4.1.1) used in paper IV which set the state-of-the-art in learning Hodge numbers of three and four-folds. The next section 4.2 shows that analytic equations for line bundle cohomologies can be learned. It is based on the results of paper I.

Chapter 5 discuses recent progress in learning Calabi-Yau metrics using neural networks. The `cymetric` package published alongside paper V is presented in section 5.1. The learning of Ricci-flat metrics, requires a point sampling procedure (section 5.1.1), finding a good Ansatz for the metric tensor (section 5.1.3), and encoding all the necessary mathematical properties in a custom loss function presented in section 5.1.4. A couple new examples not discussed in the literature so far are presented in section 5.2.

In the second part II Calabi-Yau manifolds are utilised to construct standard like models (SLM) from the heterotic $E_8 \times E_8$ string theory. In particular the popular line bundle sum construction, introduced in section 6.1 of chapter 6, is used to construct several new examples. The next section 6.2 reviews the results of paper III for constructing such models on gCICYs.

The subsequent chapter 7 introduces reinforcement learning for effective parsing of heterotic line bundle configurations. Actor-Critic agents are reviewed in section 7.1 and the `gymCICY`-package developed for paper II is presented in section 7.2. New SLMs are found with deep reinforcement learning in section 7.2.1 on so far unexplored CICYs .

The thesis wraps up with a summary and conclusion in chapter 8 and gives a popular science summary in Swedish in chapter 9.

2. String theory meets machine learning

Machine learning and its modern applications based on deep learning underwent a meteoric rise in popularity over the past decade. It has itself established as an integral part and toolkit for many disciplines in the physical sciences, often leading to surprisingly accurate results and predictions [10]. Theoretical physics is no exception and researchers are beginning to utilize these new algorithms more regularly in their work [11]. In the context of string compactifications there are several obvious problems which can benefit from smart and more efficient algorithms.

Many string theoretical computations are (NP-) hard [12, 13]. In some cases they are even undecidable, for example when finding solutions to Diophantine equations. In other cases they scale double exponentially with the input, e.g. when computing Gröbner bases an often necessary computation in heterotic compactifications. Computations that take seconds to finish on a local machine might take hours or won't finish at all for minor modifications of the input data. Hence it is desirable to find short-cuts and good approximations to these computations.

There are also situations for which no exact solution is known as is the case for the Calabi-Yau metric tensor. In those instances one has to resort to numerical approximations. This is an application, where deep learning really shines as a universal function approximator with highly optimised hardware and software libraries.

On the other hand computations are plentiful. The largest dataset of Calabi-Yau three-folds compromises 473×10^6 reflexive polytopes [14]. Each polytope can have several triangulations, in the worst case leading up to 10^{428} distinct Calabi-Yau manifolds coming from a single polytope [15]. There are even more configurations to consider, when accounting for the vector bundle data on each Calabi-Yau. Starting from F-theory requires Calabi-Yau four-folds for which the upper bounds of flux vacua are estimated to be $10^{272\,000}$ [16]. Thus, string theory is in dire need of smart ways to parse this landscape of different vacua.

Machine learning can be categorized into four related subfields depending on the problem formulation and kind of learning algorithms involved. They are presented in fig. 2.1. Each of these subfields is again connected via deep learning which is responsible for the majority of state-of-the-art algorithms.

Supervised machine learning denotes the problem of training a model on a labelled dataset. Typical problems include image recognition in which a dataset of images exists with labels of the content. A neural network is

then trained to classify the images based on their labels. Machine learning researchers test the performance of their algorithms on established benchmarks, such as ImageNet [17]. Deep neural network revolutionised the field of image and pattern recognition [18] which started the recent artificial intelligence boom.

Unsupervised machine learning is often used in studying clusters of some unlabelled data. The other main application is dimensional reduction. Unsupervised learning algorithms utilise similarity metrics between data samples, such as their difference in feature space, to group samples. Typical examples based on neural networks are autoencoders to detect anomalies in a dataset. Variational Autoencoders [19] are employed to generate new synthetic data. Other generative algorithms are based on Generative Adversial Networks (GAN) [20] which also initially started as unsupervised learning techniques.

Self-supervised learning is a relatively new emerging subfield in between supervised and unsupervised learning. It gained traction in natural language processing (NLP) when training large models at scale against masked unlabeled data to recover the original input. In this way the neural networks are able to learn good generalisable representation of the data from which it is possible to quickly fine tune to related tasks. A famous example exhibiting these generalisation properties is GPT-3 [21] which was at its publication the largest neural network.

Reinforcement learning does not strictly rely on datasets. Instead it studies the design of algorithms looking for the optimal solution of a complicated environment. The algorithms are agents, which can explore a problem via interactions with the environment and get rewarded or punished for their performed actions. They have to balance exploration and exploitation to find an optimal strategy parsing the different states. Utilizing self-play it is possible to train these models at large scale and reach superhuman performance in a variety of settings. The most famous example is probably AlphaZero which outperformed the best humans in games such as shogi, chess, and Go [22].

There have been applications of algorithms from three of the four subfields within the string theory community. A non exhaustive list[1] of these is given in section 2.1. The main part of this chapter introduces neural networks and deep learning in section 2.2, while section 2.3 covers the basics of reinforcement learning. A more thorough review with a theoretical physics reader in mind has been written by Ruehle [11]. The reader familiar with these topics may skip these sections.

[1]The literature of machine learning applications to string theory has grown significantly over the last two years, making every attempt at a somewhat exhaustive list quickly fruitless. I apologise for all the papers that have not been mentioned.

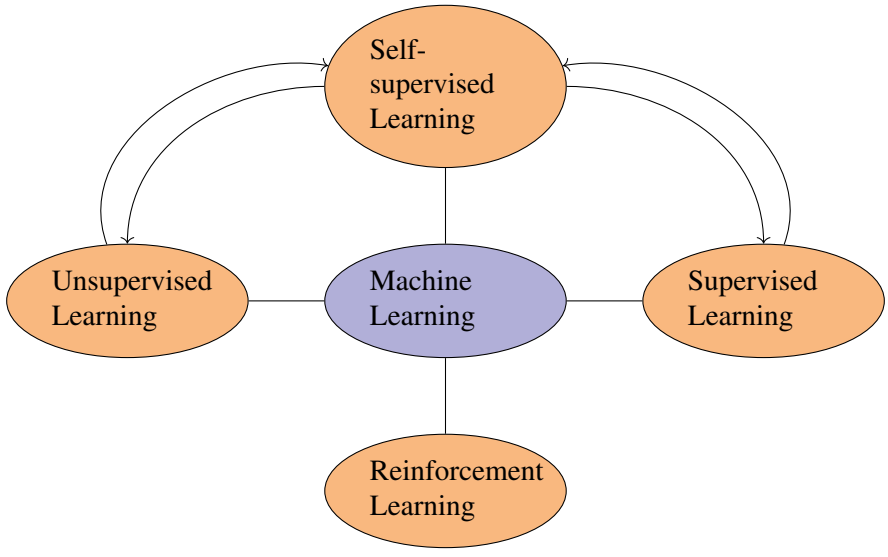


Figure 2.1. Different subfields of machine learning based on the available data and problem. Self-supervised learning lies in between supervised and unsupervised learning. The main training is done in an almost unsupervised manner against the input data. The final networks are subsequently fine tuned in a more supervised setting.

2.1 Related work

Most applications in string theory have been in the context of *supervised learning*. A popular target of these learning algorithms are topological quantities of Calabi-Yau manifolds due to well established existing datasets [23].

For example, He initiated studies learning the Hodge numbers of Calabi-Yau manifolds [24]. This was followed by applications of tree methods, support vector machines, and neural networks to predict the Hodge numbers of complete intersection Calabi-Yau (CICY) three-folds [24–28] and four-folds in [24, 29] as well as in paper IV. The elliptic fibrations of CICYs were studied in [30]. Calabi-Yau manifolds constructed as hypersurfaces in toric varieties were investigated with neural networks in [15, 31–33]. Furthermore, line bundle cohomologies have been predicted with deep learning and tree based methods [24, 34–37].

Calabi-Yau volumes were learned directly in [38]. They have also been computed as a necessary check when learning the unique Ricci-flat Calabi-Yau metric directly in [39–43] and in paper V. While there exists no labeled dataset for Calabi-Yau metric tensors, the neural network optimisation is done in a supervised manner by minimizing the Monge-Ampère equation and Ricci tensor directly. These numerical solutions have been used to investigate the swampland distance conjecture [43] and construct line bundle connections [44].

Other applications of supervised learning techniques were in string theory conjecture generation [45] and learning the volumes of knot invariants [46, 47]. Moreover, there exists a whole program to learn various kinds of mathematical structures [48], such as Lie structures [49], the Sato–Tate conjecture [50], number fields [51], arithmetic curves [52], and graph laplacians [53]. This program demonstrated that many advanced mathematical structures can be learned with accuracies of over 90% by throwing dense neural networks at the problem. These initial results suggest that there still remains a lot of hidden structure to be uncovered even in pure mathematics.

Popular applications of *unsupervised learning* are autoencoders to find clusters of SLMs in string theory compactifications [54–56]. The clusters are often identified with KMeans-clustering which has also been successfully applied to determine cones in the charge lattice of line bundle cohomology computations [36]. Persistent homology has been used to study the distributions of type IIB flux vacua [57].

Deep *reinforcement learning* is a popular tool to parse the string landscape in the search for interesting physics vacua. Initial studies using Actor-Critic agents considered intersecting brane configurations in the type IIA setting [58] and have been extended to heterotic line bundle models on complete intersection Calabi-Yau manifolds in paper II. More reinforcement learning applications of physics model building can be found in [59–61], of solving the unknot problem in [62], and of solving conformal field theories in [63, 64].

The other popular approach to search for physics models are genetic algorithms [65]. These algorithms have been employed in the same settings described above [66] which led to studies comparing bias and performance between genetic algorithms and reinforcement learning [67, 68]. It was found that genetic algorithms generally converge much faster and find solutions earlier, but the variance of solutions is larger for reinforcement learning leading to a broader class of different interesting physical models. Genetic algorithms have also been used to solve the tadpole problem [69, 70].

Finally, techniques from theoretical physics, such as perturbation theory and renormalisation group flow, are being used to study the properties of neural networks [71, 72]. A NN-QFT correspondence has been proposed [73] to describe the properties of neural networks at initialisation. This dictionary was refined in [74]. An effective theory of deep learning before and after training has been worked out in [75].

2.2 Deep learning

Deep learning describes algorithms which are based on neural networks. Neural networks have been known for a long time and they come with favourable theoretical properties such as the universal function approximation theorem [76]. In paraphrased language it states that a sufficient deep (or wide) neural

network can approximate any function with negligible error. Unfortunately it does not tell us how to find the correct neural network parameters θ.

The rise in popularity of deep learning over the last decade is due to technological improvements in computational power, larger datasets, and improved software libraries. These days deep learning is responsible for the majority of progress attributed to artificial intelligence. Neural networks are predictors in the case of supervised learning, for example when guessing the labels of image data. They are realistic data generators in the form of variational autoencoders [77] or generative adversial networks [20] in unsupervised learning. They are also function approximators for policy and action-value functions [78] in reinforcement learning. Finally, the large foundation models [79] developed in the context of self-supervised learning are deep neural networks based on the transformer architecture [80]. There is a vast literature introducing neural networks to the interested reader. The canonical introduction is due to Bishop [81], more recent books can be found by Goodfellow et al. [82] or Zhang et al. [83]. The latter includes freely available well curated jupyter notebooks. Introductions to the topic from a (theoretical) physics perspective can be found in Refs. [11, 84].

Linear Regression

In this section we will briefly introduce neural networks and their optimisation in the context of supervised learning. It will be good to fix some notation. Consider a labelled dataset $\mathscr{D} = (\mathbf{X}, \mathbf{y})$ with observation variables $\mathbf{X}$ and labels $\mathbf{y}$. A linear regression model, $f(\mathbf{x}; \theta) = \mathbf{x} \cdot \mathbf{w} + b$, maps a single observation $\mathbf{x}$ to a label y, $f : \mathbf{x} \to y$. It can be understood as a trivial neural network consisting of a single output layer with no hidden layers. The neural network parameters θ have been decomposed into weights $\mathbf{w}$ and bias b. How does one find the optimal parameters $\hat{\theta}$? Statistics comes to the rescue. The likelihood $p(\mathscr{D}|\theta)$ is the probability of observing $\mathscr{D}$ given θ. The goal is to find the optimal parameters $\hat{\theta}$ that maximise the likelihood of observing y given an input $\mathbf{x}$. This procedure is denoted as *Maximum Likelihood Estimation* (MLE). Since $p(\mathscr{D}|\theta)$ is a probability distribution and strictly positive applying the logarithm does not change the position of the maximum. The optimal values are then found by optimising the log-likelihood

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta} \log p(\mathscr{D}|\theta). \tag{2.1}$$

To make this problem more explicit assume that the dataset $\mathscr{D}$ contains N independent and identically distributed random variables with some random Gaussian noise ε ($\mu_\varepsilon = 0$ and $\sigma^2 =$ fixed). The regression model depends on the input variables $\mathbf{x}$ with K features and its parameters θ. It learns a distribution for y. Assuming that this distribution is Gaussian then

$$p(y|\mathbf{x}, \theta) = \mathscr{N}(y|\mu(\mathbf{x}), \sigma^2(\mathbf{x})) \tag{2.2}$$

where the mean is given by $\mu = \mathbf{x} \cdot \mathbf{w}$ and the variance is taken to be constant $\sigma^2(\mathbf{x}) = \sigma^2$. The log-likelihood $l(\theta)$ becomes

$$l(\theta) = \sum_i^N \log p(y_i|\mathbf{x}_i, \theta) \overset{(2.2)}{=} -\frac{1}{2\sigma^2} \sum_i^N (y_i - \mathbf{x}_i \cdot \mathbf{w})^2 + \text{ const} \tag{2.3}$$

and one recovers the well-known formula for least-square fitting of a regression model. The optimal weights are found by finding the minimum of the negative log-likelihood. This objective function is commonly denoted as *loss* or cost function for neural networks.

The current description of the problem has taken a very data centric view. From a Bayesian perspective one would like to incorporate our prior beliefs $p(\theta)$ of the parameters θ into the analysis. The prior and likelihood together can be used to compute the posterior distribution $p(\theta|\mathscr{D})$ via Bayes theorem

$$p(\theta|\mathscr{D}) = \frac{p(\mathscr{D}|\theta)p(\theta)}{\int \mathrm{d}\theta' p(\mathscr{D}|\theta')p(\theta')}. \tag{2.4}$$

The posterior distribution is often of particular interest, because it contains all the information about the wanted parameters θ given the initial dataset $\mathscr{D}$. Unfortunately the denominator tends to be intractable in everything but the simplest toy examples. Hence, in order to draw samples from the posterior one has to rely on approximations such as Markov Chain Monte Carlo (MCMC) methods [81].

Ignoring the overall initialisation due to the denominator it is possible to consider a different loss function, *maximum a posteriori probability* (MAP). In this approach the log-likelihood is replaced with the log-posterior in (2.1) and leads to

$$\hat{\theta}_{\text{MAP}} = \arg\max_\theta \log p(\mathscr{D}, \theta) + \log p(\theta). \tag{2.5}$$

Depending on the prior believes this introduces further regularization terms for the parameters θ. For example take a Gaussian prior with mean zero and variance σ' then equation (2.5) leads to so-called L_2 or Ridge regularisation of the form

$$\hat{\theta}_{\text{MAP}} = \arg\max_\theta \left[-\frac{1}{2\sigma^2} \sum_i^N (y_i - \mathbf{x}_i \cdot \mathbf{w})^2 - \frac{1}{2\sigma'^2} \sum_k^K w_k^2 + \text{ const} \right] \tag{2.6}$$

where K are the total number of features and weights in the linear regression model. The squared weights in the loss function lead to an overall decay of the weights controlled by the parameter σ'.

Fully connected networks

Neural networks are based on repeated linear regression chained to some non-linearities called activation functions. In supervised learning tasks the output

of a neural network can be either a set of scalar predictions (regression) or a vector of zeros and ones (classification) representing each label. The intermediate n_h layers between input and output layer are called hidden layers. If there is more than one hidden layer, such a neural network is referred to as deep. As in the previous section a neural network learns the likelihood of observing $f(y;\theta) \simeq p(y|\mathbf{x},\theta)$:

$$f : \mathbb{R}^{\text{in}} \rightarrow \begin{cases} \mathbb{R}^{\text{nout}} & \text{regression} \\ \mathbb{R}^{\text{nlabels}} & \text{classification} \end{cases} . \tag{2.7}$$

At each hidden layer the following tensor manipulations are carried out

$$\begin{aligned} H_1 &: \mathbf{a}_1 = \sigma_1(\mathbf{z_1} = \mathbf{W_1}\mathbf{x} + \mathbf{b_1}) \\ &\vdots \\ H_{n_h} &: \mathbf{a}_{n_h} = \sigma_{n_h}(\mathbf{W_{n_h}}\mathbf{a_{n_h-1}} + \mathbf{b_{n_h}}) \end{aligned} \tag{2.8}$$

where σ_i is a non-linear function. Typical activation functions and their derivatives are shown in figure 2.2. The outputs of the neural network are

$$\mathbf{a_{n_h+1}} = \begin{cases} \mathbf{W_{n_h+1}}\mathbf{a_{n_h}} + \mathbf{b_{n_h+1}}, & \text{regression} \\ \text{softmax}(\mathbf{W_{n_h+1}}\mathbf{a_{n_h}} + \mathbf{b_{n_h+1}}) & \text{classification} \end{cases} . \tag{2.9}$$

Within the context of this thesis neural networks are frequently used to solve different problems. Figure 2.3 depicts a fully connected, also called dense, neural network of a regression task with a single scalar output. In order to find the optimal weights[2] one again has to introduce a loss function which will be minimized. Common choices for these loss functions are the mean-square loss, which was introduced in (2.3), for regression models, and the categorical cross-entropy for a classification task with P-labels:

$$\mathscr{L}_{\text{mean-square}} = \frac{1}{N}\sum_i^N (y_i - f(\mathbf{x}_i;\theta))^2 \tag{2.10}$$

$$\mathscr{L}_{\text{cross-entropy}} = -\sum_i^N \sum_j^P y_{ij} \log f(\mathbf{x}_i;\theta)_j + (1-y_{ij}) \log\left[1 - f(\mathbf{x}_i;\theta))_j\right] . \tag{2.11}$$

The derivatives of the loss functions $\frac{\partial \mathscr{L}}{\partial \theta}$ are computed with automatic differentiation. In practice most neural network optimisations rely on modifications

[2]Optimal is a bit of a stretch. While it is possible for the linear regression model to find optimal weights minimizing the least-square loss, the weights of deep neural networks will only find some local minimum in the parameter landscape. Empirical evidence shows though that these local minima often generalise very well beyond the training data [85].

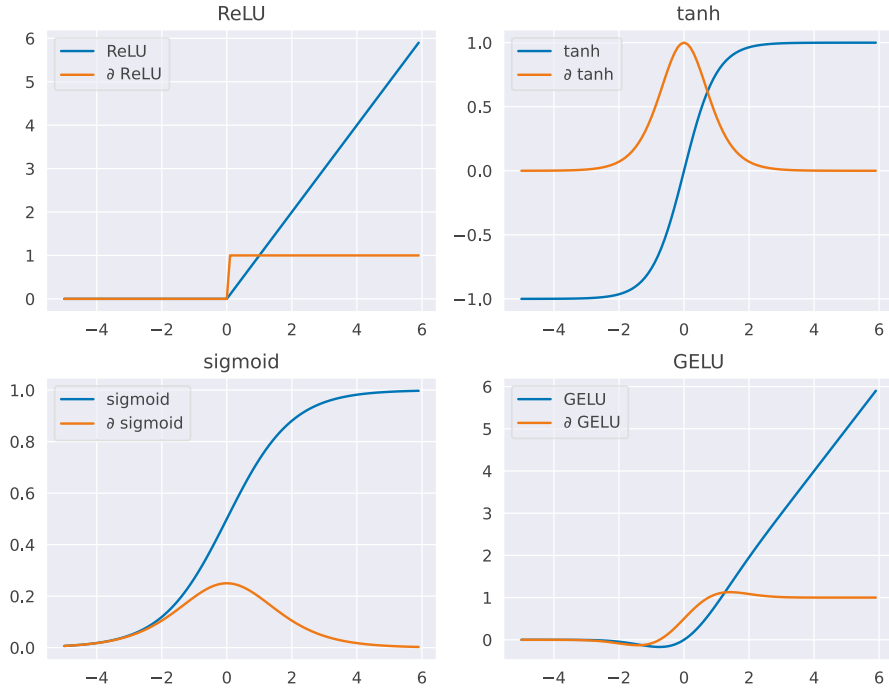


Figure 2.2. Typical activation functions: ReLU, tanh, sigmoid and GELU in blue, their derivatives in orange.

of a procedure called gradient descent. Gradient descent starts with initialising the weights with some non zero prior $p(\theta)$ for example a Gaussian. The weights are then subsequently updated with gradient updates while iterating over the whole dataset

$$g_t = \frac{\partial \mathscr{L}(\theta_t)}{\partial \theta_t}, \qquad \theta_{t+1} = \theta_t - \eta_t g_t. \tag{2.12}$$

The parameter η_t is called the learning rate and determines the step-size of the updates. The updates are computed by backpropagating the gradients through each layer, e.g. for the parameters in layer N one applies the chain rule and finds

$$\frac{\partial \mathscr{L}}{\partial \theta^i} = \frac{\partial \mathscr{L}}{\partial a^N} \frac{\partial a^N}{\partial z^N} \frac{\partial z^N}{\partial \theta^N}. \tag{2.13}$$

In that way all gradients are found by repeated application of the chain rule. The whole process is efficiently implemented in modern machine learning libraries simplifying the day-to-day life of machine learning researcher significantly. Standard gradient descent has some disadvantages: it is sensitive to

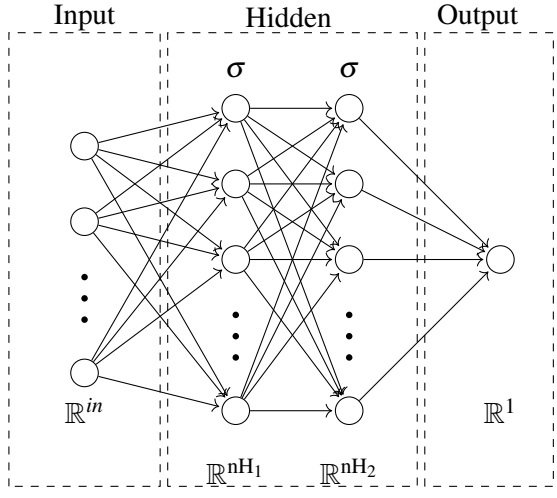


Figure 2.3. A fully connected neural network with two hidden layers and a single output of a regression problem.

the size of the gradient steps and choice of learning rate, it is computationally expensive to calculate gradients over the whole dataset, and the final results is highly sensitive to the initial starting position. There are common improvements which are used in practice.

SGD: Stochastic Gradient Descent is a modification of regular gradient descent by considering random mini-batches of n samples and summing over those in the loss functions (2.10) and (2.11). There are then N/n gradient updates, denoted as one epoch, when iterating over the whole dataset once. The more frequent updates reduce the computation time to find a minimum. A consequence of the stochastic updates is a lower chance of getting stuck in some isolated bad local minima. SGD is often further modified by adding an additional momentum contribution with hyperparameter γ

$$g_t = \gamma g_{t-1} + \frac{\partial \mathscr{L}(\theta_t)}{\partial \theta_t} \qquad \theta_{t+1} = \theta_t - \eta_t g_t \tag{2.14}$$

which takes values $\gamma \in [0,1]$. Adding momentum to the gradient updates improves convergence in relatively flat directions, while at the same time regularising directions with a high curvature.

RMSprop: Root Mean Squared propagation further refines standard gradient descent by keeping track of the second moment $s_t = \mathbb{E}[g_t^2]$ [86]. Introducing the hyperparameter $\beta (= 0.9,$ default value if not otherwise specified),

which controls the averaging of s_t, the gradient updates become:

$$g_t = \frac{\partial \mathscr{L}(\theta_t)}{\partial \theta_t}, \qquad s_t = \beta s_{t-1} + (1-\beta) g_t^2, \qquad \theta_{t+1} = \theta_t - \eta_t \frac{g_t}{\sqrt{s_t + \varepsilon}}, \tag{2.15}$$

where ε is a numerical stabilisation parameter. As the name suggest, the gradient updates are normalised with respect to the root of the mean square moment s_t. This reduces the learning rate in directions with a large gradient and allows the usage of a larger initial learning rate leading to faster convergence.

Adam: Adaptive moment estimation further improves RMSprop [87]. It keeps track of running averages of first and second moments m_t, s_t. They are used for adaptive changes to the learning rate

$$\begin{aligned} g_t &= \frac{\partial \mathscr{L}(\theta_t)}{\partial \theta_t}, \qquad m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \quad s_t = \beta_2 s_{t-1} + (1-\beta_2) g_t^2, \\ \hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \qquad\qquad \hat{s}_t = \frac{s_t}{1-\beta_2^t}, \quad \theta_{t+1} = \theta_t - \eta_t \frac{\hat{m}_t}{\sqrt{\hat{s}_t + \varepsilon}}. \end{aligned} \tag{2.16}$$

Two hyperparameters $\beta_1 (= 0.9)$ and $\beta_2 (= 0.999)$ controlling the momentum decay are introduced. The default values result in a momentum at initialisation close to zero, which is why the actual algorithms performs a bias correction to $\hat{m}_t, \hat{s}_t$. Adam will be the default optimisation algorithm used in this thesis if not stated otherwise.

More advanced building blocks

So far only fully connected building blocks have been discussed. However, there are many more possibilities for the hidden layers. In this section a couple different building blocks which have been used in the experiments of this thesis are introduced. Most of these layers are fairly standard in modern deep neural network architectures and improve performance significantly.

Convolutional layer: A convolutional layer consists of 2d filters with shape (width, height). Generalisation to higher dimensional filter is straightforward. They scan with a specified stride-size over the two-dimensional input, e.g. an image. This procedure is significantly more parameter efficient than fully connected neural networks and better in extracting characteristic features from the input data. It introduces a bias towards local features. Much of the initial progress in the recent AI revolution can be attributed to convolutional neural networks [18]. The repeated linear transformations are highly optimised on modern graphical processing units and make it possible to scale computations to much larger and deeper neural networks than plain fully connected ones.

Dropout layer: Dropout is used to combat overfitting [88]. A dropout layer includes a single hyperparameter, the dropout rate $p \in [0,1]$. At each iteration step during the training process every neuron has a chance of p to be

disabled. As a consequence signals have to be propagated via several neurons through the network, introducing a regularising effect. On the downside, the disabled neurons will not receive a gradient update and the additional signal propagation through different neurons takes additional training time. In the validation and test step all neurons are activated and the network is scaled appropriately to account for the additional signals.

Residual blocks: Residual blocks became very popular after their introduction in ResNet, which won the 2015 ImageNet competition [89]. Residual connections propagate the input signal $\mathbf{a}_k$ directly into the deeper layer. Consider any building block and denote its transformation as $\mathscr{F}(\mathbf{a}_k)$. This can be a dense or convolutional layer and their activation functions. The residual block then takes the form

$$\mathbf{a}_{k+1} = \mathscr{F}(\mathbf{a}_k) \oplus \mathfrak{h}(\mathbf{a}_k) \tag{2.17}$$

where $\mathfrak{h}$ is often chosen to be the identity map or an equivalent projection to match the output shapes of $\mathscr{F}$. This operation allows for much better gradient propagation into the earlier layers in very deep neural networks. Due to residual blocks it was possible to efficiently train networks with over a thousand layers in ResNet [89].

Attention Mechanism: The attention mechanism [90] has revolutionised NLP in the form of the flexible transformer architecture [80]. It is used to train deep neural networks at scale largely in a self-supervised manner. Fine-tuning on a specific downstream tasks often achieves state-of-the-art performance on a variety of problems, such as translations from English to German or French [91]. The attention mechanism works with tokenised data and no longer has the local bias of convolutional layers, but instead allows for global interactions of the input signals. The success in NLP tasks has motivated the use in other domains, such as image recognition via the vision-transformer [92].

2.3 Reinforcement learning

Reinforcement learning describes the process of goal-oriented learning in an environment, $\mathscr{E}$ [93]. The environment is explored with algorithms, denoted as agents, which observe a state S_t of the environment and interact via actions A_t. Each action is followed by a reward signal R_t and the transition into a new state S_{t+1} of the environment. The purpose of reinforcement learning is to study and possibly solve the environment by using different strategies for the agents that maximise the accumulated reward.

To name a few examples: the environment can be a board game such as chess or Go. The observation states are the board positions while the possible actions are the figures or tokens the two players can move and place. The reward signal might be zero for any move and ± 1 if an agent has won or lost

the game. These game settings have gotten a lot of attention in the past few years, when artificial intelligence agents modelled by neural networks were able to beat the world champions in both chess and Go. The neural networks were trained in self-play against different instances of each other at scale [22, 94]. This is a remarkable feat, as the observation space of Go admits 10^{178} possible configurations. It provides hope for the string theory community that such algorithms can also be utilized to find realistic compactifications in the huge configuration spaces described in the beginning of this chapter.

In the rest of this section the setting of reinforcement learning will be formalised and some necessary notation introduced. By using results from **dynamic programming** a simple toy example is solved from which we generalise to the application of neural networks. In chapter 7 deep reinforcement learning is revisited to study compactifications of the heterotic string on Calabi-Yau manifolds. For that purpose Actor-Critic agents are employed, which will further be introduced in section 7.1. The interested reader is referred to [93] for the standard textbook on reinforcement learning and the lecture series by Silver [95] on which this section is based.

A reinforcement learning environment is usually based on a Markov decision process.

Definition 1. *A Markov decision process (MDP) is a 4-tuple (S,A,P_a,R_a), where*

- *S is a set of states called the state space,*
- *A is a set of actions called the action space,*
- *$P_a(s,s') = Pr(S_{t+1} = s'|S_t = s, A_t = a)$ is the probability of transitioning into state s' given s and a,*
- *$R_a(s,s')$ is the immediate reward for state transition $s \to s'$ due to action a.*

A Markov decision process satisfies the Markov property, that is each state S_t contains all relevant information and is independent of the trajectory and history of past actions.[3] Note that $P_a(s,s')$ governs the whole dynamics of the environment. The agent selects an action according to some strategy, which from now on will be denoted as policy.

Definition 2. *A policy π is a distribution over actions given states,*

$$\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]. \tag{2.18}$$

[3]Recent results have demonstrated that deep reinforcement learning is also capable of solving environments, which do not satisfy the Markov property with superhuman performance, for example in games such as DotA2 [96, 97] and StarcraftII [98].

The goal of reinforcement learning is to determine a policy that maximises the accumulated reward. To this end it is helpful to introduce the discounted return G_t.

Definition 3. *The return G_t is the total discounted reward for time-step t.*

$$G_t = R_{t+1} + \gamma R_{t+2} + ... = \sum_k \gamma^k R_{t+k+1}, \qquad \text{with } \gamma \in [0,1] \tag{2.19}$$

This is a useful objective to be maximised by any policy. The parameter γ controls the long-term thinking of the deployed strategy. Note that there exists a recursive relationship

$$G_t = R_{t+1} + \gamma G_{t+1}. \tag{2.20}$$

In order to measure the value of a given action under policy π one introduces two further definitions.

Definition 4. *The action value function $q_\pi(s,a)$ of an MDP is the expected return starting from state s, using action a, and then following policy π*

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]. \tag{2.21}$$

The value of any state under policy π is given by the state value function.

Definition 5. *The state value function $v_\pi(s)$ of an MDP is the expected return starting from state s and following policy π*

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]. \tag{2.22}$$

With these definitions out of the way, it will be illuminating to solve a MDP using dynamic programming in a model based way. It is possible to write down recursive relationships for the state and action value functions. Those are called **Bellman equations**. They read

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + ... | S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(s_{t+1}) + ... | S_t = s] \end{aligned} \tag{2.23}$$

and

$$q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]. \tag{2.24}$$

This leads to the definition of the optimal state and action value functions.

Definition 6. *The optimal state-value function $v_*(s)$ for all $s \in S$ is the maximum value function over all policies*

$$v_*(s) = \max_\pi v_\pi(s). \tag{2.25}$$

The optimal action-value function $q_*(s,a)$ for all $s \in S$ and $a \in A(s)$ is

$$q_*(s,a) = \max_\pi q_\pi(s,a). \tag{2.26}$$

The Bellman equations (2.24) and (2.23) can be used to find v_* and q_* yielding the Bellman optimality equations

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')] \tag{2.27}$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_\pi q_*(s',a')]. \tag{2.28}$$

Finally, it is possible to obtain an optimal policy by picking actions in such a way that the state value function is equivalent to the optimal one found in equation (2.27).

Theorem 1. *Define a partial ordering over policies*

$$\pi \geq \pi' \text{ if } v_\pi(s) \geq v_{\pi'}(s) \; \forall \; s. \tag{2.29}$$

Then for any Markov decision process

- *exists an optimal policy π_* that is better than or equal to all other policies, $\pi_* \geq \pi \; \forall \; \pi$,*
- *all optimal policies achieve the optimal value function $v_{\pi_*}(s) = v_*(s)$.*

The whole process of applying the Bellman equations and finding an optimal policy is demonstrated with an example.

Example 1. *Consider the MDP depicted in fig. 2.4 showing the somewhat simplified life cycle of a graduate student at Uppsala University. It starts with a state S_0 at the beginning of the PhD studies and the student has two possible actions: either they are productive and do research leading to a publication or they sit back and watch the latest season of their favourite show on Netflix. Research is fun, but also exhausting giving a reward of $R = -1$, while watching Netflix is fun and not exhausting yielding a reward of $R = -0.5$. The latter, however, doesn't result in any progress and the student remains in the same state at the beginning of their studies. If they researched and managed to publish their work, the less fun part of writing their thesis begins. This action returns a reward of $R = -2$, meanwhile watching Netflix only incurs a penalty of $R = -0.5$. In the third possible thesis-state the student only has to (successfully) defend their work, which does not only give them a PhD but also $R = 10$ reward points. Postponing the defense to watch Netflix on the other hand gives the usual reward of $R = -0.5$.*

The Bellman equations (2.24) *and* (2.23) *are used to compute action and state value function for a policy π. Starting from a uniform policy which does*

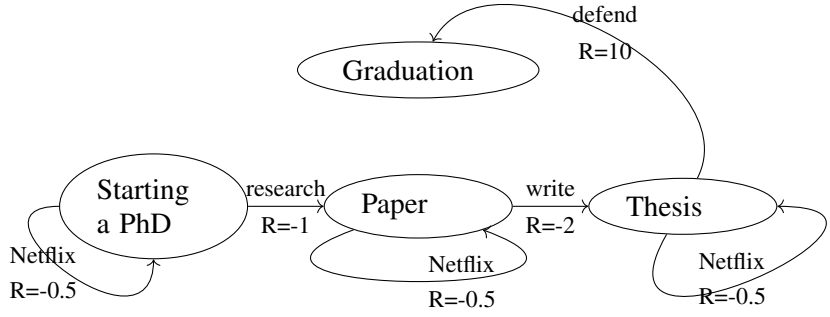


Figure 2.4. MDP of starting graduate studies in theoretical physics. In each state there are two possible actions, first performing research activities and second laying back and watching Netflix.

not discriminate between watching Netflix and doing research and discount factor of $\gamma = 1$ *the Bellman equation* (2.23) *yields*

$$v_\pi(\mathit{Thesis}) = \frac{1}{2}\cdot 10 + \frac{1}{2}\left[-\frac{1}{2} + v_\pi(\mathit{Thesis})\right] \Leftrightarrow v_\pi(\mathit{Thesis}) = 9.5$$
$$v_\pi(\mathit{Paper}) = \frac{1}{2}\left[-2 + v_\pi(\mathit{Thesis})\right] + \frac{1}{2}\left[-\frac{1}{2} + v_\pi(\mathit{Paper})\right] \Leftrightarrow v_\pi(\mathit{Paper}) = 7$$
$$v_\pi(\mathit{Start}) = \frac{1}{2}\left[-1 + v_\pi(\mathit{Paper})\right] + \frac{1}{2}\left[-\frac{1}{2} + v_\pi(\mathit{Start})\right] \Leftrightarrow v_\pi(\mathit{Start}) = 5.5 \quad .$$

These values show that doing graduate studies is quite a rewarding (MD) process. An equal probability for watching Netflix and performing research might be a realistic reflection of the real world, but it is hardly the optimal policy. Solving the Bellman optimality equations (2.27) *gives*

$$v_*(\mathit{Thesis}) = \max\left(10, -\frac{1}{2} + v_*(\mathit{Thesis})\right) \Leftrightarrow v_*(\mathit{Thesis}) = 10$$
$$v_*(\mathit{Paper}) = \max\left(-2 + v_*(\mathit{Thesis}), -\frac{1}{2} + v_*(\mathit{Paper})\right) \Leftrightarrow v_*(\mathit{Paper}) = 8$$
$$v_*(\mathit{Start}) = \max\left(-1 + v_*(\mathit{Paper}), -\frac{1}{2} + v_*(\mathit{Start})\right) \Leftrightarrow v_*(\mathit{Start}) = 7 \quad .$$

In summary the optimal policy π_* *to obtain a PhD is to never watch Netflix but focus on research instead.*

This example benefited from the fact that we had perfect information about the MDP, knew all transition functions, and the state space was quite small. In most real world applications those conditions are not always satisfied. For example when exploring the string landscape there are more possible states than can be stored on any hard drive and that doesn't even account for state-action pairs. Hence, model based approaches are less useful in solving these environments. So what should one do when the state space becomes too large and it is not possible to look up all future rewards? Neural networks come to the rescue. There are two common approaches, policy based ones learning a policy π directly, or value based ones learning the action-value function $q_\pi(s,a)$ directly. In the rest of this section a brief summary is given on how deep learning algorithms are applied in reinforcement learning. A more detailed derivation of the here presented algorithms can be found in [93].

Value based algorithms attempt to find the optimal state or action value function. The optimal value function is found by following a policy that balances exploring new states and accumulating the largest return. A typical policy example is the ε-greedy policy, selecting random actions with probability ε and otherwise performing the optimal action. Popular value based algorithms are SARSA [93], and Q-learning and its variations [99, 100]. In Q-learning $q_*(s,a)$ is learned via iterative off-policy updates at each time step t for an episode with fixed length:

$$q(s_t,a_t) \leftarrow q(s_t,a_t) + \eta \left[R_{t+1} + \gamma \max_a q(s_{t+1},a) - q(s_t,a_t) \right] \tag{2.30}$$

This algorithm is off-policy because it uses the greedy policy prediction selecting the action that maximises $q(s_{t+1},a)$ in its value update, even though the actual followed policy need not necessarily pursue the same action a_{t+1}. The correction to $q(s_t,a_t)$ is usually denoted as TD error δ_{t+1}. Deep Q-learning introduces a function approximation in the form of a neural network with parameters θ for the optimal action value function $q_*(s_t,a) \approx Q(s_t,a;\theta)$ rather than storing the values in a look-up table. It is optimised with gradient descent by minimizing the least-square loss of the update rule (2.30)

$$L_i(\theta) = \delta_{t+1}^2 = \left(R_{t+1} + \gamma \max_a Q(s_{t+1},a;\theta) - Q(s_t,a_t;\theta) \right)^2 \tag{2.31}$$

at each time step t. There are further variations, based on adding a baseline, delaying the updates, or learning two state-value functions simultaneously, which allow for faster convergence and training.

Policy based methods of reinforcement learning learn a policy

$$\pi(a|s,\theta) = \frac{e^{h(s,a,\theta)}}{\sum_b e^{h(s,b,\theta)}} \tag{2.32}$$

directly by applying a softmax activation over the possible actions from a numerical function approximator $h(s,a,\theta)$ which is usually a neural network depending on the parameters θ. The most famous example is the REINFORCE

algorithm [101]. It learns a policy by gradient ascent with respect to a performance measure J. This performance measure is given by the (optimal) state value function $J(\theta) := v_{\pi_\theta}(s_0)$ where s_0 is the initial state of a finite episode. The policy gradient theorem [93] allows to re-express the gradients in terms of derivatives with respect to the original policy:

$$\begin{aligned}\nabla J(\theta) &\propto \mathbb{E}_\pi \left[\sum_a q_\pi(S_t, a) \nabla_\theta \pi(a|S_t, \theta) \right] \\ &= \mathbb{E}_\pi \left[G_t \nabla_\theta \ln(\pi(A_t|S_t, \theta)) \right] \end{aligned} \tag{2.33}$$

This gives the following gradient updates for the parameters θ with discount factor γ

$$\theta_{t+1} \leftarrow \theta_t + \eta \gamma^t G_t \nabla_\theta \ln(\pi(A_t|S_t, \theta)) \tag{2.34}$$

Note, that the computation of G_t requires full reward information of the whole episode making REINFORCE a Monte-Carlo method with gradient updates after the episode ends. This often leads to somewhat slower convergence then online algorithms.

Modern approaches combine these two methods and have one neural network learn the policy and another learn the state or action value function. These algorithms are commonly denoted as Actor-Critic models and will be further discussed in section 7.1.

Part I:
Calabi-Yau geometries

The first part of this thesis discusses Calabi-Yau manifolds and their topological properties in more details. It starts with a review of Complete Intersection Calabi-Yau manifolds and their generalised construction. For the purpose of studying CICYs and their topological properties the package `pyCICY` has been developed. This review is followed by a discussion of some recent results in learning their Hodge numbers and the cohomologies of line bundles. The second half focuses on learning the local metric tensor of Calabi-Yau manifolds directly using neural networks. This is a very promising and new field of research which finally allows to get accurate numerical approximations within a reasonable computation time. To that purpose the Python package `cymetric` is presented, which allows for computation of the metric tensor at specific points in complex and Kähler moduli space.

3. Calabi-Yau manifolds

This chapter introduces Calabi-Yau manifolds and fixes the notation for the rest of the thesis. The main part is centered around the well studied constructions of Complete Intersection Calabi-Yau manifolds in section 3.2. Calabi-Yau manifolds are defined as follows.

Definition 7. *A Calabi Yau $n-$fold is an $n-$dimensional compact complex Kähler manifold with trivial canonical bundle.*

There are several equivalent definitions, such as vanishing Ricci form or Chern class, requiring that the holonomy group of the metric g is contained in $SU(n)$ or that the manifold admits a globally defined nowhere vanishing holomorphic n-form. In this thesis only compact manifolds with these properties are considered. Calabi-Yau manifolds are named after E. Calabi [102] and S.T. Yau [103].

Conjecture 1. ***Calabi [102].*** *Let X be a compact complex manifold X with Kähler form J' and corresponding Kähler metric g'. Suppose that ρ is a real, closed $(1,1)$-form on X with $[\rho] = 2\pi c_1(X)$. Then there exists a unique Kähler metric g on X with Kähler form J, such that $[J] = [J'] \in H^2(X,\mathbb{R})$ and we have that the Ricci form of g is ρ.*

The existense of this unique Ricci flat metric was proven over twenty years later by Yau [103]. While this is great news for physicists and mathematicians alike, the proof does not include a method for constructing the metric. The problem of finding an analytic expression for the metric remains unsolved today almost half a century later. Only for 'trivial' CY examples such as tori, there are known closed form expressions.[1] On the other hand many global properties are known of Calabi-Yau manifolds. We have good control of topological quantities, such as Hodge numbers and Chern classes, and have developed several successful ways of constructing a wide set of different examples. The three main constructions of Calabi-Yau geometries are:

- Complete Intersection Calabi-Yau manifolds, in short CICY [106]. They are built from intersecting polynomial equations in products of projective spaces. They have been classified in three and four dimensions [107,

[1] There has been progress recently in constructing K3-metrics [104, 105]. This construction, however, requires Kähler properties unique to two dimensions and can not be straightforwardly extended to arbitrary n-folds.

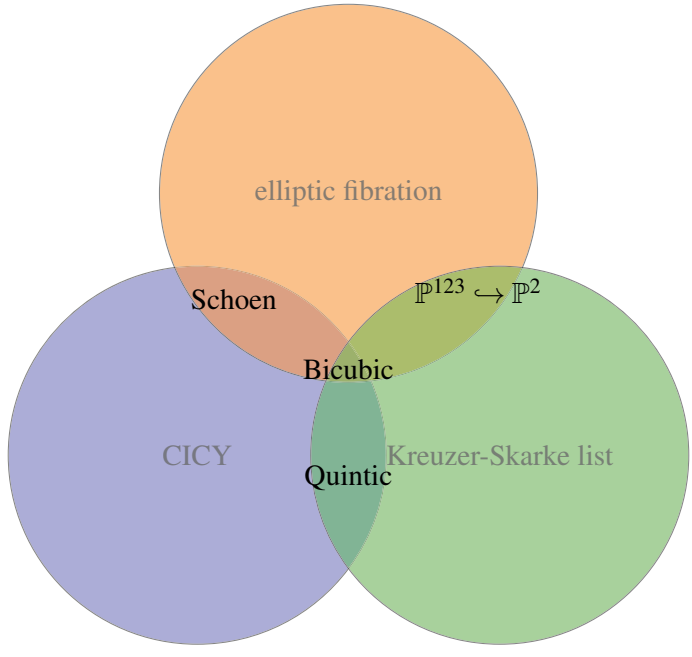


Figure 3.1. Venn diagram of the three most common Calabi-Yau three fold constructions with popular examples at the intersections.

108] and were recently generalised to incorporate even more examples by considering negative degrees in the defining polynomials [109].

- Three dimensional Calabi-Yau hypersurfaces in toric varieties given by reflexive polytopes have been classified by Kreuzer and Skarke [14]. There are a total of 473 million of them and they make up the largest class of CY manifolds. The construction can in general be extended to more than one hypersurface [110, 111] with first initial studies trying to classify such systems performed in [112]. Moreover, it is also possible to incorporate negative degrees in the toric hypersurface construction [113].
- Elliptic fibrations over weak Fano manifolds are popular constructions in the context of F-theory [114, 115]. 61 539 smooth toric bases that support elliptic fibrations have been classified in [116, 117].

These different constructions are presented by the Venn diagram in figure 3.1 with some famous selected examples. There is an overlap of the three methods as some manifolds can be realized in only one way and others in all three. The study of Calabi-Yau manifolds is worthy a book by itself and such books have been written [9, 118, 119]. In this chapter the most important details about some common constructions and the computation of relevant (topological) quantities for physics are stated. It requires basic knowledge of

complex geometry on the level of [120, 121]. The more interested reader is referred to [118, 119] for proofs and proper derivation of the results stated in the following.

3.1 Ricci-flat metrics

This section discusses the **Calabi conjecture** 1 in more detail and fixes notation. It is based on [119]. Assume X is a compact complex manifold of dimension n. A Kähler form J is a nowhere vanishing real $(1,1)$-form on X. If J is closed, i.e.

$$dJ = (\partial + \bar{\partial})J = 0, \tag{3.1}$$

X is said to be a Kähler manifold and one can define the cohomology class $[J] \in H^{1,1}(X)$. The Kähler form J further defines a metric g with non vanishing components

$$J = ig_{a\bar{b}} dz^a \wedge dz^{\bar{b}} \tag{3.2}$$

where z^a are the homogeneous coordinates on X. On any Kähler manifold the metric tensor is given locally by taking partial derivatives with respect to a Kähler potential K

$$g_{a\bar{b}} = \partial_a \bar{\partial}_{\bar{b}} K. \tag{3.3}$$

The Ricci curvature $R_{a\bar{b}}$ and Ricci form ρ on a Kähler manifold are given by

$$R_{a\bar{b}} = -\partial_a \bar{\partial}_{\bar{b}} \log\det g \tag{3.4}$$

$$\rho = -i\partial\bar{\partial} \log\det g. \tag{3.5}$$

The Ricci scalar is obtained from the curvature by contraction with the inverse metric $g^{a\bar{b}}$. ρ is a real $(1,1)$-form, but also a closed 2-form with cohomology class $[\rho] = 2\pi c_1(X) \in H^2(X,\mathbb{R})$. By definition 7 a Calabi-Yau manifold has vanishing first Chern class and thus vanishing Ricci curvature and scalar.

Example 2. *Consider the complex projective space $\mathbb{P}^n$. $\mathbb{P}^n$ is a complex compact Kähler manifold with a single Kähler class $[J]$. It has $n+1$ homogeneous coordinates $z_0, \ldots, z_i, \ldots, z_n$. The Kähler potential is given by*

$$K = \frac{1}{\pi} \ln \sum_{i=0}^{n} |z_i|^2 \tag{3.6}$$

and its corresponding metric is the so called Fubini-Study metric

$$g_{FS} := g_{i\bar{j}} = \partial_i \partial_{\bar{j}} \left[\frac{1}{\pi} \ln \sum_{i=0}^{n} |z_i|^2 \right] = \frac{1}{\pi} \frac{\sum_i |z_i|^2 \delta_{i\bar{j}} - z_i z_{\bar{j}}}{\sum_i |z_i|^4}. \tag{3.7}$$

The Ricci form is

$$\rho = 2\pi c_1(\mathbb{P}^n) = -(n+1)J \tag{3.8}$$

*Thus $\mathbb{P}^n$ is **not** a Calabi-Yau manifold.*

Computing the Ricci curvature in equation 3.4 gives rise to a fourth order partial differential equation with respect to the Kähler potential. Fortunately, it is possible to reduce the expression to a second order Monge-Ampère equation. Consider the closed $(1,1)$-form J associated to the Ricci-flat metric g. J and J', which is any Kähler form on X, are in the same Kähler class and are thus related by an exact correction from a scalar function ϕ:

$$J = J' + \partial\bar{\partial}\phi. \tag{3.9}$$

There are two ways to build the unique top-form on a CY. It is unique up to some complex constant number because $h^{n,n}(X) = 1$. The first construction considers the anti-symmetric product $J \wedge J \wedge J$ and the alternative relies on the nowhere vanishing holomorphic volume form Ω. This gives rise to

$$J \wedge J \wedge J = \kappa\Omega \wedge \bar{\Omega} \tag{3.10}$$

a non linear second order partial differential equation, where κ is a constant complex number.

3.2 Complete Intersection Calabi-Yau manifolds

A Complete Intersection Calabi-Yau manifold, X, is constructed as the vanishing loci of K homogeneous polynomial equations in products of projective spaces $\mathscr{A} = \Pi_{i=1}^{r}\mathbb{P}^{n_i}$. In the rest of this section it is assumed that the Calabi-Yau manifold is of dimension three, i.e. $\sum_i^r n_i - K = 3$, but the methods explained here generalise straightforwardly to other dimensions. The intersections of hypersurfaces commute and are encoded in a so called configuration matrix:

$$X \in \left[\begin{array}{c||ccc} n_1 & q_1^1 & \cdots & q_1^K \\ \vdots & \vdots & \ddots & \vdots \\ n_r & q_r^1 & \cdots & q_r^K \end{array}\right]_{\chi}^{h^{(1,1)},h^{(2,1)}} . \tag{3.11}$$

Each $q_i^a \in \mathbb{Z}_{\geq 0}$ corresponds to the degree of the a-th polynomial in the homogeneous coordinates of the i-th complex projective space. The super and sub script denote Hodge numbers $(h^{(1,1)}, h^{(2,1)})$, and the Euler number χ which are topological invariants of the manifold and uniquely determined from the configuration matrix. There are certain conditions which need to be satisfied which will be derived in section 3.2.1. The procedure for computing the Hodge numbers is presented in section 3.2.2.

3.2.1 Topological quantities

A Calabi-Yau manifold has vanishing first Chern class. The Chern class of a sub-manifold X embedded in some ambient space $\mathscr{A}$ can be obtained from the **adjunction formula**

$$0 \longrightarrow TX \longrightarrow T\mathscr{A}|_X \longrightarrow \mathscr{N}_X \longrightarrow 0 \tag{3.12}$$

where $\mathscr{N}$ is the normal bundle defining the embedding hypersurface(s). From this short exact sequence one finds the following formula for the total **Chern class**, c, of each bundle

$$c(TX) = \frac{c(T\mathscr{A}|_X)}{c(\mathscr{N}_X)}. \tag{3.13}$$

The right hand side can be expanded in terms of Kähler forms J_i spanning the ambient space

$$c(X) = c_1^i J_i + c_2^{ij} J_i J_j + c_3^{ijk} J_i J_j J_k. \tag{3.14}$$

Explicit calculation of the quotient in (3.13) yields

$$c_1^i = \left[n_i + 1 - \sum_{a=1}^{K} q_i^a \right] \stackrel{!}{=} 0 \tag{3.15}$$

$$c_2^{ij} = \frac{1}{2}\left[-\delta_{ij}(n_i+1) + \sum_{a=1}^{K} q_i^a q_j^a \right] \tag{3.16}$$

$$c_3^{ijk} = \frac{1}{3}\left[\delta^{ij}(n_i+1) - \sum_{a=1}^{K} q_i^a q_j^a q_k^a \right]. \tag{3.17}$$

Equation (3.15) is a condition on each row of the configuration matrix (3.11), only if it is satisfied a configuration matrix represents a Calabi-Yau manifold. This condition is not sufficient in bounding the number of possible CY configuration matrices. Finiteness of CICY manifolds can be shown by considering identities between different matrices [118]. The Euler characteristic is found by integrating the top (third-) Chern class

$$\chi = \int_X c_3^{ijk} J_i \wedge J_j \wedge J_k = c_3^{ijk} d_{ijk}, \tag{3.18}$$

where the integration over the CY three-fold is done by contraction with the **triple intersection numbers**, d_{ijk}. The triple intersection numbers are computed by lifting the integral to the ambient space

$$\begin{aligned} d_{ijk} &= \int_X J_i \wedge J_j \wedge J_k = \int_{\mathscr{A}} \mu \wedge J_i \wedge J_j \wedge J_k \\ &= \int_{\mathscr{A}} \bigwedge_{a=1}^{K} \left(\sum_l q_l^a J_l \right) \wedge J_i \wedge J_j \wedge J_k. \end{aligned} \tag{3.19}$$

It is common practice to contract the second Chern class with the triple intersection numbers to have an easily comparable vector representation

$$c_{2,i} = c_2^{kj} d_{ijk}. \tag{3.20}$$

Let's consider a simple example.

Example 3. ***Quintic.*** *Take the CY given by a single homogeneous polynomial constraint of degree five in* $\mathbb{P}^4$*. The configuration matrix reads*

$$Q \in [4||5]^{1,101}_{-200} \tag{3.21}$$

and it clearly satisfies equation (3.15)*. The single triple intersection number is by equation* (3.19)*,* $d_{111} = 5$*. The third Chern class is computed with equation* (3.17) *to be* $c_3^{111} = -40$*. Contraction of those two tensors yields the Euler characteristic in equation* (3.18)*:*

$$\chi = d_{ijk} c_3^{ijk} = -200. \tag{3.22}$$

The Hodge numbers of the quintic manifold are computed in the next section, after introducing some more machinery.

3.2.2 Sequence chasing

Computation of the Hodge numbers requires additional machinery from algebraic geometry. This section is based on results derived in [122, 123] with notation borrowed from the physics literature [118, 124]. The adjunction formula (3.12) expresses the tangent space cohomologies TX in terms of line bundle cohomologies $\mathscr{N}_X$ and ambient tangent bundle cohomologies $T\mathscr{A}$. The two latter cohomology groups can be computed with the introduction of two further exact sequences.

- The **Koszul resolution** expresses the cohomology of a vector bundle V on X in terms of cohomology groups on $\mathscr{A}$ by tensoring V with

$$0 \to \wedge^K \mathscr{N}^* \to \ldots \to \mathscr{N}^* \to \mathscr{O}_{\mathscr{A}} \to \mathscr{O}_{\mathscr{A}}|_X \to 0\,. \tag{3.23}$$

- The **Euler sequence** states that for an ambient space $\mathscr{A}$ given by a K-fold product of n_i-dimensional projective spaces there exists a short exact sequence

$$0 \to \mathscr{O}_{\mathscr{A}}^{\oplus k} \to \mathscr{O}_{\mathscr{A}}^{\oplus n_1+1}(1,0,\ldots,0) \oplus \ldots \oplus \mathscr{O}_{\mathscr{A}}^{\oplus n_k+1}(0,0,\ldots,1) \to T\mathscr{A} \to 0. \tag{3.24}$$

Combining these three sequences (3.12), (3.23) and (3.24) it is possible to compute $H^\bullet(TX)$ by appropriately splitting the Koszul resolution (3.23) into short exact sequences. After some cumbersome computations one is able to

express tangent space cohomology groups of X in terms of line bundle cohomology groups over the ambient space $H^\bullet(\mathscr{A},L)$. Such cohomology groups over direct products of projective spaces are computed with a theorem due to Bott-Borel-Weil [118, 124].

Theorem 2. ***Bott-Borel-Weil.*** *Let $\mathbb{F}$ be a flag space and $V \sim (a_1,\ldots,a_{n_0}|\ldots|d_1,\ldots,d_{n_r})$ a holomorphic homogeneous vector bundle over $\mathbb{F}$. Then:*

1. *Homogeneous vector bundles V over $\mathbb{F}$ are in 1-1 correspondence with the $U(n_1)\times\cdots\times U(n_r)$ representations.*
2. *The cohomology $H^i(\mathscr{A},V)$ is non-zero for* ***at most one value*** *of i, in which case it provides an irreducible representation of $U(N)$, $H^i(\mathbb{F},V)\approx(c_1,\ldots,c_N)\mathbb{C}^N$.*
3. *The bundle $(a_1,\ldots,a_{n_0}|\ldots|d_1,\ldots,d_{n_r})$, determines the cohomology group $(c_1,\ldots,c_N)$, according to the following algorithm:*

   1) Add the sequence $1,\ldots,N$ to the entries in $(a_1,\ldots,a_{n_0}|\ldots|\,d_1,\ldots,d_{n_r})$.

   2) If any two entries in the result of step 1 are equal, all cohomology vanishes; otherwise proceed.

   3) swap the minimum number $(=i)$ of neighbouring entries required to produce a strictly increasing sequence.

   4) Subtract the sequence $1,\ldots,N$ from the result of previous step, to obtain $(c_1,\ldots,c_N)$.

Without going into too much detail, the ambient space of any CICY consists of direct products of projective spaces, which can be written as 'generalised flag varieties' $\mathbb{P}^{n_i}=\left(\frac{U(n+1)}{U(1)\times U(n)}\right)$ [124]. The line bundles then take representations of $U(1)\times U(n)$ expressed as Young diagrams and the cohomology dimension is computed from their dimension. It is clear from step 3 that the majority of ambient space cohomologies are going to vanish. The whole procedure of splitting the Koszul resolution and expressing $H^\bullet(X,L)$ in terms of ambient space cohomologies $H^\bullet(\mathscr{A},L)$ is summarised in a Leray tableaux.

Definition 8. *A Leray tableaux is given by*

$$E_{i+1}^{j,k}=\frac{Ker(d_i:E_i^{j,k}(L)\to E_i^{j-i+1,k-i}(L))}{Im(d_i:E_i^{j+i-1,k+i}(L)\to E_i^{j,k}(L))} \tag{3.25}$$

where the first page is defined as

$$\begin{aligned} E_1^{j,k}(L) &:= H^j(\mathscr{A},L\otimes\wedge^k\mathscr{N}_X^*),\\ &\qquad with\ k=0,\ldots,K;\ j=0,\ldots,dim(\mathscr{A}). \end{aligned} \tag{3.26}$$

This is a complex defined by differential maps $d_i:E_i^{j,k}\to E_i^{j-i+1,k-i}$.

There are a few comments in order about the construction of the differential maps d_i. A complete discussion is out of the scope for this review and the

reader is referred to [118, 124]. From the Bott-Borel-Weil theorem 2 it is possible to express the ambient space cohomologies in terms of Young diagrams, which one can map to a basis of monomials. Starting at the first page $i = 1$, d_1 maps from one monomial basis $E_1^{j,k}$ to the next $E_1^{j,k-1}$. The explicit maps are generated from the defining polynomial equations and in that way complex structure dependent. Images, kernel and the rank of these maps are computed with computer algebra systems. The computation takes a considerable amount of time, because the number of monomials representing the ambient space cohomologies grow factorial with the line bundle charges. To summarize, if there are non trivial maps after splitting all sequences the computations can only be carried out with efficient computer implementations.

There are several theorems which can make our life easier. They improve the computation time significantly by relating Hodge numbers or constraining them.

Theorem 3. Kodaira vanishing theorem. *For any Kähler manifold, X, and positive line bundle $L \simeq \mathcal{O}(\vec{q})$ the cohomology groups*

$$H^i(X, L \otimes K_X) = 0 \qquad \forall i > 0 \tag{3.27}$$

vanish.

Note that for any Calabi-Yau manifold the canonical bundle K_X is trivial, such that the above theorem simplifies to $\forall L \simeq \mathcal{O}(\vec{q})$ with $\vec{q} > 0 : h^i(X, L) = 0$, $\forall i > 0$. Another strong vanishing theorem relies on the computation of the slope. The **slope** of a vector bundle V is given by

$$\mu = \int_X c_1(V) \wedge J \wedge J = c_1^i(V) t^j t^k d_{ijk} \tag{3.28}$$

and has to vanish for polystable Hermitian Yang-Mills bundle [125]. These kind of bundles are also the ones physicists are mostly interested in when studying heterotic string compactifications. There exists a strong vanishing theorem due to Kobayasha [126] for such bundles. It is paraphrased here as stated in [127].

Theorem 4. Kobayasha. *A poly-stable, zero slope bundle V on a Calabi-Yau manifold X admits no global holomorphic sections, i.e. $H^0(X, V) = 0$.*

This statement can be made even stronger by invoking Serre duality.

Theorem 5. Serre duality. *On any $n-$dimensional Kähler manifold X with vector bundle V the following holds true:*

$$H^i(X, V) \simeq H^{n-i}(X, V^* \otimes K_X). \tag{3.29}$$

Thus on any Calabi-Yau three fold with a poly-stable vector bundle with vanishing slope one finds that also $H^3(X,V) = 0$. Finally, the index of a vector bundle V can be computed by either summing up the individual Hodge numbers or integrating the wedge product between Chern characters and Todd class

$$\text{ind}(V) = \sum_i (-1)^i h^i(X,V) = \int_X \text{ch}(V) \wedge \text{Td}(X). \tag{3.30}$$

This expression further simplifies for line bundles with $L \simeq \mathcal{O}(\vec{q})$ on Calabi-Yau manifolds to

$$\text{ind}(L) = d_{ijk}\left(\frac{1}{6}q^i q^j q^k + \frac{1}{12}q^i c_2(X)^{jk}\right). \tag{3.31}$$

It will be illuminating to apply this whole machinery and get some feeling for the sequence chasing done in heterotic string compactifications.

Example 4. ***Revisiting Quintic.*** *In example 3 we considered the Quintic manifold. The Hodge numbers are computed from the adjunction formula* (3.12)*, which leads to long exact sequence in cohomology*

$$\begin{aligned}
0 &\longrightarrow H^0(X,TX) \longrightarrow H^0(X,T\mathscr{A}|_X) \longrightarrow H^0(X,\mathscr{N}_X) \longrightarrow \\
&\hookrightarrow H^1(X,TX) \longrightarrow H^1(X,T\mathscr{A}|_X) \longrightarrow H^2(X,\mathscr{N}_X) \longrightarrow \\
&\hookrightarrow H^2(X,TX) \longrightarrow H^2(X,T\mathscr{A}|_X) \longrightarrow H^2(X,\mathscr{N}_X) \longrightarrow \\
&\hookrightarrow H^3(X,TX) \longrightarrow H^3(X,T\mathscr{A}|_X) \longrightarrow H^3(X,\mathscr{N}_X) \longrightarrow 0.
\end{aligned}$$

Applying Kodairas vanishing theorem 3 and the index formulae for line bundles in (3.31) *yields for the right column*

$$h^i(X,\mathscr{N}_X) = \begin{cases} ind(\mathscr{N}_X) = 125 & for\ i = 0 \\ 0 & else \end{cases}.$$

In a next step one has to consider the tensor product of Koszul sequence (3.23) *and the ambient tangent space cohomology group*

$$0 \to \mathscr{N}^* \otimes T\mathscr{A} \to T\mathscr{A} \to T\mathscr{A}|_X \to 0 \tag{3.32}$$

The first two columns in the long exact cohomology sequence diagram are computed on the ambient space. The ambient space cohomologies of $T\mathscr{A}$ are

found from the Euler sequence (3.24)

$$0 \to \mathscr{O}_{\mathscr{A}} \to \mathscr{O}_{\mathscr{A}}^{\oplus 5}(1) \to T\mathscr{A} \to 0 \tag{3.33}$$

The Bott-Borel-Weil theorem 2 reduces the long exact sequence in cohomology to

$$0 \to 1 \to 5 \cdot 5 \to h^0(\mathscr{A}, T\mathscr{A}) \to 0 \tag{3.34}$$

which reproduces the result found by the Bott formula for a single projective space [124]. Next ones computes $H^\bullet(\mathscr{A}, \mathscr{N}^* \otimes T\mathscr{A})$ *by tensoring the normal bundle with the Euler sequence* (3.33) *leading to*

$$0 \to h^3(\mathscr{A}, \mathscr{N}^* \otimes T\mathscr{A}) \to 1 \to 0$$

such that

$$h^0(\mathscr{A}, T\mathscr{A}) = 24 \quad \textit{and} \quad h^3(\mathscr{A}, \mathscr{N}^* \otimes T\mathscr{A}) = 1.$$

Plugging this back into the Koszul sequence (3.32) *shows that*

$$h^0(X, T\mathscr{A}|_X) = 24 \quad \textit{and} \quad h^2(\mathscr{A}, T\mathscr{A}|_X) = 1.$$

Collecting these intermediate results and inserting them into the long exact cohomology sequence of the adjunction formulae one finds that

$$0 \to h^0(X, TX) \to 24 \to 125 \to h^1(X, TX) \to 0$$

and

$$0 \to h^2(X, TX) \to 1 \to 0.$$

This gives in combination with the Euler characteristic (3.18) *the well known result that*

$$h^{(1,1)}(X) = 1 \quad \textit{and} \quad h^{(2,1)}(X) = 101.$$

This example of the simplest CICY demonstrates how involved the procedure of computing Hodge numbers becomes even for the most trivial examples. In the calculation above we did not have to make use of the Leray spectral sequence, nor did we have to compute any differential maps on the ambient space cohomologies. As mentioned earlier this procedure requires algorithmic implementation on a computer for when the cohomology sequences do not split back into short exact sequences and one has to compute the rank of these maps. For that purpose the `pyCICY`-package has been developed. It can be found on

`https://github.com/robin-schneider/CICY`

and contains routines for the computation of Hodge numbers of tangent and line bundles, triple intersection numbers, and Chern classes.

An interesting and so far unanswered question is to ask whether the number of Calabi-Yau manifolds is finite. Given the vast number of possible configuration matrices (3.11) and constructions outlined in the introduction of this chapter, it is important to know whether two different constructions of Calabi-Yau manifolds are equivalent.

Theorem 6. Walls [128]. *Two Calabi-Yau manifolds are said to be equivalent as six-dimensional real manifolds if the following set of their topological invariants match*

$$d_{ijk}, h^{\bullet}, c. \tag{3.35}$$

While this theorem appears to be rather straightforward to check, this is not actually the case. In particular, one has to consider the triple intersection numbers up to non trivial basis transformation of the Kähler forms J_i. Checking for all possible transformations of the shape

$$d_{ijk} = \Lambda_i^l \Lambda_j^m \Lambda_k^n d_{lmn} \text{ with } \Lambda \in SL(r,\mathbb{Z}) \tag{3.36}$$

is impossible. Nevertheless a finite set of CICY manifolds has been classified by utilising identities between different configuration matrices [118]. The topological data of this classification is visualised in fig. 3.2, which shows the distribution of Hodge numbers for $h^{(1,1)}$ and $h^{(2,1)}$. The shape of these two distribution differs quite significantly. This is an interesting observation with respect to mirror symmetry of Calabi-Yau manifold. Mirror symmetry of CY three-folds relates manifolds with opposite Hodge numbers and is naturally manifest in the construction of toric hypersurfaces, but not so in the CICY construction. CICY mirror pairs can in principle be constructed via nef-partitions in higher dimensional toric varieties [129]. The mean, maximum and minimum values for the Hodge numbers are

$$\langle h^{(1,1)} \rangle = 7.44^{19}_{1}, \quad \langle h^{(2,1)} \rangle = 28.83^{101}_{15}, \quad \langle \chi \rangle = -42.77^{0}_{-200} \quad . \tag{3.37}$$

3.2.3 Generalised constructions

It is possible to extend the construction of CICY manifolds to also incorporate negative degrees in the configuration matrices. This requires a careful tuning of the complex structure, such that the poles of the denominators are missing the other hypersurface equations [109]. Hence, these new generalised CICYs are constructed in a two-step manner. First, similar to CICYs, one considers a subvariety M constructed from K hypersurfaces in some ambient space $\mathscr{A} =$

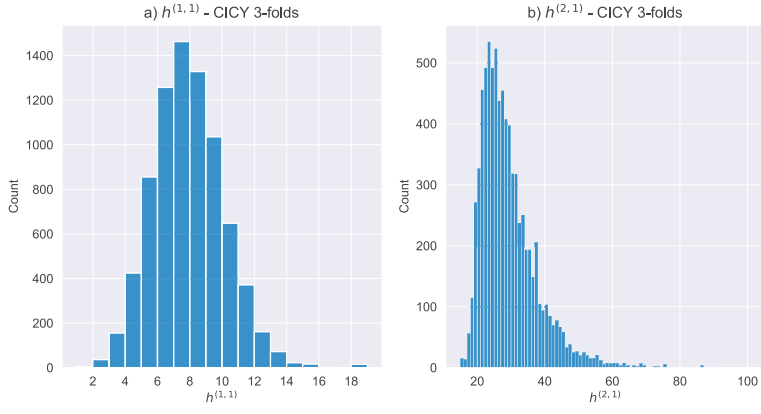


Figure 3.2. Histogram plot of a) $h^{(1,1)}$ and b) $h^{(2,1)}$ for all CICY three folds which are not direct products of lower dimensional manifolds.

$\mathbb{P}^{n_1} \times \mathbb{P}^{n_2} \ldots \times \mathbb{P}^{n_s}$. In a second step one adds L further constraints such that the first Chern class vanishes and arrives at a complete intersection X. Note, that these additional constraints have negative degrees and are built from sections of the previous constraints. The total variety X is still the common zero locus of a number of homogeneous equations. In contrast to the CICY construction the latter hypersurfaces do no longer commute. The whole information is again encoded in a configuration matrix:

$$X \in \left[\begin{array}{c||ccc|ccc} n_1 & a_1^1 & \ldots & a_1^K & b_1^1 & \ldots & b_1^L \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ n_r & a_r^1 & \ldots & a_r^K & b_r^1 & \ldots & b_r^L \end{array}\right]^{h^{(1,1)},h^{(2,1)}}_{\chi} . \tag{3.38}$$

Following the previously introduced notation one has the degrees n_i for each projective space. The integers $a^i_j \in \mathbb{Z}_{\geq 0}$ describe the first subvariety M and $b^i_j \in \mathbb{Z}$ encode the hypersurfaces with rational sections and negative degrees. The construction of these rational sections is highly non trivial and will not be further discussed in this thesis, for explicit examples and more detailed information see the theory section in paper III or for a more general treatment consult the original paper [109]. Computing topological invariants also has to be done in a two-step manner, first relating to quantities on the subvariety M and then back to the ambient space $\mathscr{A}$.

4. Learning Hodge numbers

Cohomology computations are an integral part of string theory. In the context of heterotic string compactifications the Euler characteristic χ of the embedded polystable vector bundle determines the number of massless fermion generations [4]. χ is given by the difference in dimensions of the first and second cohomology group, each respectively counting the number of families and anti-families. These cohomology computations rely on spectral sequences introduced in section 3.2.2. They are important ingredients of the dictionary between geometry and physical observables. This connection will be further investigated in chapter 6.

In chapter 3 the steps required to find the dimension of cohomology groups were introduced by studying the Leray tableaux of definition 8. The downside of this algorithmic procedure lies in the computation time required to determine the rank of the complex structure dependent differential maps on the ambient space. The size of these maps scales factorially with the input parameters. In this chapter recent work utilizing machine learning algorithms to compute the Hodge numbers of tangent and line bundle cohomologies is reviewed. This line of research has been initiated by He in Ref. [24]. He learned the Hodge numbers of CICY three- and four-folds, and line bundle cohomologies using fully connected neural networks. These initial benchmark studies have been extended to more systematic treatments of the two CICY datasets [24–29]. Moreover, they motivated the use of deep learning to determine line bundle cohomologies over toric varieties and other CY manifolds [34–37].

The next section 4.1 compares the accuracy of different supervised machine learning approaches in learning the Hodge numbers of Calabi-Yau manifolds. The focus is on CICYs for which the necessary topological information is encoded in a configuration matrix (3.11). The state-of-the-art results rely on so called inception blocks, which will be introduced in section 4.1.1. These blocks are also the main building components of the CICYMiner developed in the context of paper IV. In section 4.2 the cohomologies of line bundles will be further investigated. It has been postulated that their Hodge numbers on Calabi-Yau n-folds are described by polynomials of degree n [130]. The charge lattice can be divided into cones with each following an analytic expression. In paper I we learned these expressions on the set of CICY manifolds with Picard group two and two defining hypersurfaces.

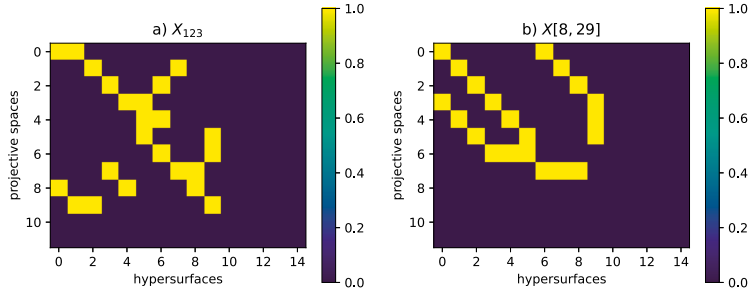


Figure 4.1. Two CICYs plotted as 2d-image: to the left the CICY with index 123 in the CICY list and to the right a CICY studied in a later chapter for model building and defined by the configuration matrix (7.7)*.*

4.1 Learning tangent bundle cohomologies

CICYs are a natural starting point for testing the performance of machine learning algorithms in solving problems of algebraic geometry. All topologically distinct CICY three- and four-folds have been compiled in two separate datasets [106–108, 131]. The three-folds contain 7890 data samples which can be efficiently parsed on a laptop, while the roughly 10^6 four-folds are large enough to test more complicated architectures requiring larger amounts of training data. These datasets form the basis of several supervised learning studies and accuracy benchmarks in learning their Hodge numbers. The original motivation initiating these benchmark studies was to see whether deep learning can be utilised to study various topological quantities required in string compactifications [24].

In case of the three-folds, the maximal dimensions of a configuration matrix (3.11) are 13 projective spaces and 15 hypersurfaces. Hence, to have a dataset with uniform shape all configuration matrices with matrix dimensions smaller than (13,15) are padded with zeros to the bottom and right.[1] An example of this padding is presented in fig. 4.1, which contains two CICYs, X_{123} of the CICY list to the left and another CICY with Hodge numbers $X[8,29]$ given by the configuration matrix (7.7) to the right. As evident from the colorbar of

[1]Note that in this 2d-image representation of the configuration matrix the first column containing the dimension of the projective space has been omitted. This information is redundant due to the Calabi-Yau condition $n_i + 1 = \sum_a^K q_i^a$. For some algorithms it was shown to improve the performance when the information was kept in the training data [28].

	$h^{(i,j)}$	accuracy (%)	training (%)	method	Ref.
3-folds	$h^{(1,1)}$	37	63	reg. FCNN	[24]
	$h^{(1,1)}$	75	70	reg. FCNN	[25]
	$h^{(1,1)}$	85	70	class. CNN	[25]
	$h^{(1,1)}$	(55,68)	(30,80)	SVM	[25]
	$h^{(1,1)}$	(97,98,99)	(30,50,80)	Inception-NN	[27, 28]
	$h^{(2,1)}$	(50,50,50)	(30,50,80)	Inception-NN	[27, 28]
4-folds	$h^{(1,1)}$	78	2	FCNN	[24]
	$h^{(1,1)}$	96	10	FCNN	[29]
	$h^{(3,1)}$	27	10	FCNN	[29]
	$h^{(1,1)}$	(99,100,100,100)	(10,30,50,80)	Inception-NN	Paper IV
	$h^{(2,1)}$	(87,91,94,95)	(10,30,50,80)	Inception-NN	Paper IV
	$h^{(3,1)}$	(59,67,68,70)	(10,30,50,80)	Inception-NN	Paper IV
	$h^{(2,2)}$	(62,73,75,75)	(10,30,50,80)	Inception-NN	Paper IV
	$h^{(1,1)}$	(100,100,100,100)	(10,30,50,80)	CICYMiner	Paper IV
	$h^{(2,1)}$	(90,97,99,100)	(10,30,50,80)	CICYMiner	Paper IV
	$h^{(3,1)}$	(62,81,92,96)	(10,30,50,80)	CICYMiner	Paper IV
	$h^{(2,2)}$	(36,49,66,83)	(10,30,50,80)	CICYMiner	Paper IV

Table 4.1. *Accuracies for various experiments investigating Hodge numbers of CICY three- and four-folds from the literature. FCNN denotes fully connected neural networks, CNN convolutional neural network, and SVM support vector machine.*

the pictures, the CICYs have a single color channel making them suitable for the application of standard techniques from image recognition.

The accuracies of different studies have been summarized in table 4.1. The best performing algorithms are based on Google's Inception network architecture [132], which is introduced in section 4.1.1. Using this architecture Erbin and Finotello were able to achieve an accuracy of 97% in predicting $h^{(1,1)}$ for CICY three-folds [27, 28] at different training ratios. Their work involved detailed ablation studies and feature enhancement of the images. Since the triple intersection numbers can be computed with combinatorics (3.19), an accuracy of 97% almost exactly solves the problem of predicting all non-trivial Hodge numbers. Finding a good direct predictor of $h^{(2,1)}$, however, has been unsuccessful so far.

The CICY four-folds allow for more detailed studies and experimentation due to the larger dataset. The histograms for the four Hodge numbers given in figure 1 of paper IV shows four distinct distributions. The histograms of $h^{(3,1)}$ and $h^{(2,2)}$ have significantly longer tails than $h^{(2,1)}$ has in the histograms shown in fig. 3.2. $h^{(2,1)}$ in the four-fold case has an interesting shape with roughly 70% of the values being zero, while $h^{(1,1)}$ is comparable to the $h^{(1,1)}$

histogram of the three-folds. In paper IV a new multi-task deep neural network based on inception modules was designed. It learns the four distinct Hodge number distributions of CICY four-folds simultaneously. The architecture labeled **CICYMiner** is inspired from state-of-the-art image recognition techniques [133] and achieves almost perfect accuracy on two of the four Hodge numbers. CICY four-folds admit a further linear constraint relating the different values [108]

$$44 = -4h^{(1,1)} + 2h^{(2,1)} - 4h^{(3,1)} + h^{(2,2)}, \tag{4.1}$$

such that with the Euler characteristic these almost perfect results extend to the prediction of all four Hodge numbers.

Related work

The clustering of CICY three- and four-folds was studied with Siamese neural networks [134]. These networks map the higher dimensional input space of the images from $\mathbb{Z}^{13} \times \mathbb{Z}^{15} \to \mathbb{R}^3$. Thus, significantly reducing the problem of finding similar manifolds. The similarity criterion was implemented with respect to $h^{(1,1)}$. The authors are able to show that drawing on respectively 2.67% and 0.62% training samples the networks can few-shot learn a majority of the test dataset.

The CICYs are not the only target of machine learning algorithms. The dataset of toric Fano three-folds has been analysed for their volume and reflexivity of the underlying polytopes [33]. An exploration study of the Kreuzer-Skarke dataset of reflexive 4d polytopes was initiated in [31]. It revealed two surprising linear equations for the Hodge numbers and an almost perfect fit to the training data. The triangulation dependent triple intersection numbers of selected polytopes have been predicted with deep residual style neural networks [15]. Finally, the subset containing a single hypersurface in weighted projective space of dimension four was studied with supervised and unsupervised learning techniques with almost perfect accuracy in [32].

In the next subsection we will review Inception modules and summarize the results found in paper IV.

4.1.1 Inception modules

GoogLeNet scored first place in the 2014 ImageNet competition [132] with a missclassification rate of 6.67%. It compromises 22 layers with a total of 6.7×10^6 parameters. It introduced new building blocks, called inception, utilising concatenated convolutional kernels to learn features at different scales in the image data. These blocks have been further refined by introducing batch normalisation [132, 135–137]. The *ImageNet* dataset contains 14×10^6 samples and over 21 000 classes. Hence, it is roughly one order of magnitude larger in both samples and classes than the CICY four-folds. Motivated by

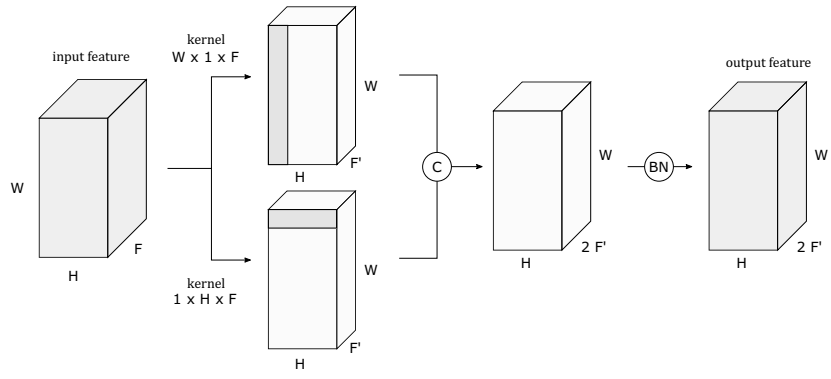


Figure 4.2. An Inception module can be decomposed into the different convolutional kernels scanning over the width (W) and height (H) with filters (F). They are subsequently concatenated (C) and followed by a batch normalization (BN) layer. The Inception module is the main building block of both the CICYMiner and the classification architectures. The figure is taken from paper IV.

these similarities and the promising results by Erbin and Finotello [27, 28] it seems suggestive to use similar inception blocks in the analysis of CICY four-fold data.

The architecture of an inception block is presented in fig. 4.2. It consists of three different operations. In a first step different convolutional kernels are applied in parallel to extract different features of the input. These blocks are then concatenated to a single output onto which batch normalisation (BN) is applied.

Batch normalisation: BN is a new building block for neural networks. It was initially proposed to reduce the internal covariate shift, that is the change in signal distributions at each layer [135]. Empirically it has been found to allow for better gradient propagation into the earlier layers in very deep neural networks and to introduce minor regularisation. This in turn allows for faster and more stable training.

Consider the signal $\mathbf{a}_{i,k}$ at the layer k with sample index i of some mini-batch $\mathfrak{B} = \{x_1, \ldots, x_m\}$. BN learns the statistics for a fixed batch size and normalises the output. Suppressing the index k these statistics read

$$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} \mathbf{a}_i \qquad \sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (\mathbf{a}_i - \mu_B)^2 \tag{4.2}$$

$$\hat{\mathbf{a}}_i \leftarrow \frac{\mathbf{a}_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \qquad \mathbf{y}_i \leftarrow \gamma \hat{\mathbf{a}}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(\mathbf{a}_i) \tag{4.3}$$

γ and β are learnable parameters and ε a constant stabilisation hyperparameter. During the training process the mini-batch mean and variance are used for the

signal propagation through the neural network. In inference mode of a batch $\mathfrak{B}_{\text{in}}$ a moving average of the training statistics $(\hat{\mu}, \hat{\sigma})$ are used. These are updated during training after every mini-batch $\mathfrak{B}_t$ with momentum α

$$\hat{\mu}_{t+1} \leftarrow \hat{\mu}_t \alpha + \mu_{\mathfrak{B}_{\text{in}}}(1-\alpha), \qquad \hat{\sigma}_{t+1} \leftarrow \hat{\sigma}_t \alpha + \sigma_{\mathfrak{B}_{\text{in}}}(1-\alpha). \tag{4.4}$$

The re-scaling gives best performance if the inference data has similar statistics as the training data.

In later studies it was revealed that the reduction of the internal covariate shift is not responsible for the advantages of BN. In fact BN does not necessarily lead to a reduction. The main benefit of BN is instead a smoothing of the loss landscape [138], making the gradient updates more predictive and allowing for higher learning rates during the training process.

Results

The results of paper IV are two fold: in a first exploratory step we considered deep neural networks based on chained inception blocks classifying the possible values of the Hodge numbers. In a second step we used a multi-task architecture with four arms predicting each Hodge number as a regression task. We proceeded as in earlier studies [27, 28] by considering different train:val:test splits with respectively 10 %, 30 %, 50 % and 80 % training and 10 % validation data to analyse the performance on limited training data. The results of these experiments are presented in table 4.1 and set the new state-of-the-art benchmarks for predicting Hodge numbers using machine learning.

Classification: The hyperparameters of the classification architectures have been optimized using Bayesian Optimisation HyperBand (BOHB) [139, 140] when predicting $h^{(3,1)}$ on 80% training data. The same hyperparameters were subsequently used in experiments on the three other Hodge numbers at the four training ratios. The neural network is built from five inception modules, containing two convolutional kernels scanning over the projective spaces (W=16) and hypersurfaces (H=20) as visualised in fig. 4.2. These kernels are then concatenated and followed by a batch normalisation layer. There are two more hidden dense layers before a softmax activation function over $\{h^{(i,j)}_{\text{min}}, \ldots, h^{(i,j)}_{\text{max}}\}$ classes predicts the output. Optimisation is done with Adam optimizer (2.16) and an initial learning rate of 10^{-4} on a mini-batch size of 128. The results improved initial studies of $h^{(1,1)}$ [24, 29] and demonstrated that it is also possible to learn the other Hodge numbers without feature engineering.

CICYMiner: Motivated by the success of using shared hyperparameters we designed the CICYMiner. This architecture contains a shared body of inception blocks with four arms each predicting one of the four non trivial Hodge numbers. Ablation studies revealed that introducing auxiliary loss branches improved the feature representation in the shared body. Moreover the introduction of a Huber loss [141] improved accuracies and robustness significantly. For a larger percentage of the training data the CICYMiner did

not only perform better than the classification network trained on a single task, but it also reached an accuracy of 100% on predicting $h^{(2,2)}$. When accounting for the linear relation (4.1) and the Euler characteristic which can be computed by integrating the fourth Chern class with the quadruple intersection numbers, it implies perfect accuracy for all Hodge numbers.

4.2 Learning line bundle cohomologies

In this section recent results in finding the cohomologies of line bundles over Calabi-Yau manifolds are presented. Monad, extension, and sum of line bundles are the most common choices for constructing vector bundles used in heterotic string compactifications. They all require the computation of line bundle cohomhologies with any possible charge combination. Theorems such as Kodaira (3) and Serre duality (5) in combination with the analytic formulae for the index (3.31) suggest that the Hodge numbers can be divided into cones within the charge lattice $q_i \in \mathbb{Z}$. Each such cone follows a simple n-fold polynomial description. These polynomials have been known for a long time in the case of a single projective ambient space. It will be useful to derive the polynomials for our favourite example.

Example 5. ***Revisiting the quintic (again).*** *Take the Calabi-Yau manifold given by a generic quintic polynomial in $\mathbb{P}^4$ encoded by the configuration matrix defined in* (3.21)*. Consider any line bundle $L \simeq \mathcal{O}(q)$. The Koszul sequence* (3.23) *and the Bott-Borel-Weil theorem 2 determine that only $h^0(X,L)$ and $h^3(X,L)$ can be non zero. Applying the index formulae* (3.31) *to compute the cohomology dimensions, one finds that*

$$h^0(X,\mathcal{O}(q)) = \max\left(\delta_{q,0} + \frac{5}{6}q^3 + \frac{25}{6}q, 0\right) \tag{4.5}$$

and h^3 is given via Serre duality in theorem 5.

Further evidence of these polynomial descriptions has been collected for all the $K = 1$ CICYs [130], for the tetra-quadric manifold [142], and finally for all CICYs with $K = 2 = h^{(1,1)}$ in paper I. The analysis of these papers relied on a combination of explicitly computing the Hodge numbers for a large set of possible charges, subdividing the charge lattice into cones depending on the structure of the underlying spectral sequence and fitting a degree three polynomial through the lattice points with linear regression. Rewriting the problem in terms of a monomial basis with degrees 0 to 3 the weights can be computed exactly with the maximum likelihood method in (2.3). For CICYs the line bundle comhomologies are computed with the `pyCICY` package [143], specifically developed for this purpose. For manifolds coming from the Kreuzer-Skarke list `cohomcalc` comes with the required functionality [144].

The analysis of paper I revealed that there exist recursive equations for some of the cones. Similar recursive patterns were afterwards rigorously proven in the case of two dimensional toric surfaces [145]. Finding such recursive relationships has very promising implications for future studies involving large scans over line bundle charges. The computations scale factorially with the absolute value of the charges and the total number of charges. It is, however, often relatively straightforward to find the Hodge number for line bundles close to the origin of the charge lattice. In that region many of the ambient space cohomologies vanish and the maps in the Leray spectral sequence (8) are between spaces of lower degree and computational feasible. Thus having recursive relationships connecting vectors with large norm to ones with small norm allows for significant computation time improvement.

Related work

The difficulties in computing the Hodge numbers motivate the use of more advanced tools from machine learning. In first exploratory studies the Hodge numbers of line bundles were predicted with neural networks trained on a small dataset of line bundle charges [24, 34]. Both papers demonstrated that neural networks are capable in learning the majority of the Hodge numbers. These results have been further supported by the capability of reinforcement learning agents to solve cohomology constraints as demonstrated in paper II.

Systematic scans of the landscape, however, require almost perfect accuracy since each potential model undergoes several cohomology checks. Hence, alternative machine learning applications focused on classifying and fitting the polynomial cones. The authors of [36] used the `cohomcalc` [144, 146] package to generate Hodge values on hypersurfaces of toric varieties. They showed that neural networks fail to compute the Hodge numbers with reliable accuracy at large charges. Instead they compute derivatives according to a degree three central difference scheme and then apply the unsupervised KMeans clustering algorithm to find the polynomial cones. This avoids the manual sequence chasing and fitting to determine the borders of the different polynomial cones.

In [35] the authors further tested the capabilities of deep learning to predict the Hodge numbers directly. They came to similar conclusions that the accuracy of dense neural nets to predict Hodge numbers is not sufficient for large scale systematic scans relying on these results. To circumvent this problem they designed a neural network architecture, in part relying on residual connections and feature enhanced input to learn the different cones and the polynomials at once. This neural network was shown to reproduce the same expressions found in paper I and also reproduced a master formula for del Pezzo surfaces [145].

Finally, decision trees have been employed to study the zeroth cohomology group of line bundles over del Pezzo surfaces [37]. In addition to previous studies the authors were interested in finding jumps in the dimension of the cohomology group by engineering the complex structure. It was shown that

decision trees are capable of predicting jumps in the spectrum with 95% accuracy when fed with the appropriate topological training data.

5. Learning Calabi-Yau metrics

Compactifications of the heterotic string on Calabi-Yau manifolds have led to a large amount of interesting standard-like-models [130]. To further reduce the number of possible models arising from string theory one has to compute the physical Yukawa couplings of these vacua. There exist topological arguments [147–150] and algebraic methods [151] to compute the vanishing of many Yukawa couplings (for more recent results see also paper VI and [152]). The physical Yukawa couplings, however, require the CY metric tensor directly in their normalisation and indirectly for determining an expression of the gauge-enhanced Laplacian.

Unfortunately there are no known analytic expressions for the metric tensor of Calabi-Yau manifolds with dimensions $n > 2$. For K3-surfaces recent progress allows for the expression of the metric in terms of a perturbative expansion around "semi-flat metrics" [104, 105]. These "semi-flat metrics" are derived by studying dualities between different string theories compactified on tori and orbifold geometries. They then arise as the metric of the moduli spaces in these different pictures, and can be computed by the counting of BPS states.

In dimensions larger than two one has to resort to numerical approximations. The first such approximations were based on the the Donaldson algorithm [153]. It approximates the Kähler potential with an iterative fixed-point method in terms of monomials with increasing degree $2k$, akin to a Fourier expansions. This method has been used extensively in the past to find numerical expressions on the Fermat quintic and quotients of the Schoen manifold [154–159]. The Donaldson algorithm comes with its own caveats. Instead of learning the metric directly, one has to work with the proxy quantity of fitting an expansion of the Kähler potential. This process requires exponential computation time and data samples for increases in accuracy.

More efficient fitting algorithms for finding an approximation of the Kähler potential are based on energy functionals [160]. These functionals are bounded from below and can be optimised to represent the Ricci-flat metric. They give an expression in terms of algebraic metrics derived from a Kähler potential with degree k monomials. These methods have been shown to scale exponentially in accuracy with the degree of the polynomial and thus are far more performant than the Donaldson algorithm. There are also more recent studies investigating the metric constructed from energy functionals at the conifold point [161].

Motivated by their role as universal function approximators, more recently neural networks have been employed to model the metric tensor [41, 42] and

the Kähler potential [40]. Applications of such neural network approximations already exist. They include studies of the swampland distance conjecture [43], random matrix theory [162], and the learning of a Hermitian Yang-Mills connection for line bundles [44]. An alternative approach learning the determinant of the metric based on a combination of machine learning methods and curve fitting is given in [39].

In paper V we developed a tensorflow [163] library, `cymetric`, to learn the metric tensor, and also the Kähler potential directly. It, furthermore, generalises to all CICYs and toric hypersurfaces, while preexisting papers were hard coded for the (Fermat) Quintic manifold. The library is written in a combination of Python and Mathematica and has been open sourced at:

`https://github.com/pythoncymetric/cymetric`

In addition to containing different metric Ansätze we proposed an architecture which gives the metric at the same point in Kähler moduli space as a reference metric, such as the Fubini-Study metric.

In the rest of this chapter the reader is introduced to the `cymetric` package and more generally how to model the metric tensor with the help of neural networks in section 5.1. The second half in section 5.2 contains additional results on a so far unstudied CICY.

5.1 Metrics as neural networks

In this section the implementation of the metric tensor as a neural network is reviewed. The training pipeline requires training data in the form of points on the manifold, which we will discuss in section 5.1.1, a neural network to model the metric tensor with possible Ansätze outlined in section 5.1.3, and a custom loss function introduced in section 5.1.4.

5.1.1 Point sampling

The training points for the neural network will be given in terms of the homogeneous ambient space coordinates z_i. To sample points on the CY one has to solve the defining hypersurface equations(s). That requires to initialise all but K ambient space coordinates and solve for the remaining ones. Each solution yields a point on the CY. The random starting points, drawn from some prior $p(z)$ will introduce a bias in the distribution of the points and not cover X uniformly. To find an unbiased distribution one should make use of Markov Chain Monte Carlo (MCMC) methods sampling with respect to the top form $\Omega \wedge \bar{\Omega}$. Modern MCMC algorithms have shown to work well in high-dimensional spaces [164, 165] avoiding pitfalls of conventional methods such as rejection sampling. They are, however, often an art by themselves requiring an extensive burn-in period, finding a good balance between rejection and

acceptance rate and anti-correlating all accepted solutions. Moreover, such an implementation tosses all but one solution of the defining hypersurface equations and thus becomes very sample inefficient.

A more elegant approach is to sample points using a theorem due to Shiffman and Zelditch [166]. The theorem studies the distribution of zeros from random sections over a complex compact space that can be mapped to $\mathbb{P}^N$. This theorem has been generalised to ambient spaces containing several projective spaces by Braun et al [155] and its application will be described in what follows.

Consider a CICY, given by K hypersurfaces defined in an ambient space $\mathscr{A} = \mathbb{P}^{n_1} \times \cdots \times \mathbb{P}^{n_r}$. Introduce free parameters $t_i \in \{t_1, \ldots, t_K\}$, that restrict the projective ambient space factors to subspaces and define a sub-manifold Y. In a next step one samples points by solving for the intersection with the defining polynomials of the CICY. The distribution of these random sections is then given by considering the anti-symmetric product of pullbacked ambient space measures restricted to the Calabi-Yau. It will be illuminating to illustrate this process with an example.

Example 6. *Consider the Calabi-Yau given by a homogeneous degree* $(2,2,3)$ *polynomial,* P_{223}*, in* $\mathscr{A} = \mathbb{P}^1_1 \times \mathbb{P}^1_2 \times \mathbb{P}^2_3$*. Then there is a single free parameter* t_1 *living in any of the three factors. Assuming that the parameter lives in the third projective space, one can define a three dimensional subspace Y*

$$Y = \mathbb{P}^1_1 \times \mathbb{P}^1_2 \times \{p_{3,0} + t_1 \cdot p_{3,1}\} \subset \mathscr{A} \tag{5.1}$$

where $p_{3,j}$ *are points living* $\in \mathbb{P}^2_3$ *and distributed according to the Fubini-Study metric. It is then straightforward to find points on the CY by studying the intersections*

$$\left(\mathbb{P}^1_1 \times \mathbb{P}^1_2 \times \{p_{3,0} + t_1 \cdot p_{3,1}\}\right) \cap X = \{3\ pts.\}. \tag{5.2}$$

Since the defining polynomial in the third projective space is of degree three there will be three solutions for generic points of Y defined in equation (5.1)*. The random points sampled from each* $\mathbb{P}^1$ *are also distributed with respect to the Fubini-Study measure. There are then three (1,1)-forms given by the pullbacks of the Fubini-Study metrics of each ambient space factor to the CY*

$$g^{(l)}_{a\bar{b}} := i^*(\omega_{\mathbb{P}^1_l}) = P^i_a g^{\mathbb{P}^1_l}_{i\bar{j}} \bar{P}^{\bar{j}}_{\bar{b}} \tag{5.3}$$

The pullback tensors are defined as

$$P^i_a = \left(\frac{\partial z_i}{\partial x_a}\right) \tag{5.4}$$

where z_i *are the homogeneous ambient space coordinates and* x_a *three local 'good' Calabi-Yau coordinates. Those three coordinates are selected from the*

ambient space coordinates by going to a patch, where all the three scaling relations set the largest ambient coordinates to $\max(z_i) = 1 + 0j$ *and then removing the coordinate with the largest value of* $\left|\frac{\partial P_{223}}{\partial z_i}\right|$ *with the defining polynomial* P_{223}*. Following this procedure one arrives in general at*

$$\left(\sum_{l=1}^{r} n_l + 1 - 1\right) - K = n$$

good local coordinates. The distribution of zero loci is then given by wedging the remaining degrees of freedom

$$dA \sim i^*(\omega_{\mathbb{P}^1_1}) \wedge i^*(\omega_{\mathbb{P}^1_2}) \wedge i^*(\omega_{\mathbb{P}^2_3}) = \varepsilon^{ace}\varepsilon^{\bar{b}\bar{d}\bar{f}} g^{(1)}_{a\bar{b}} g^{(2)}_{c\bar{d}} g^{(3)}_{e\bar{f}}. \tag{5.5}$$

If one had taken the free parameter to be in one of the $\mathbb{P}^1$*-factors, say* $l = 1$ *leading to a different subspace* Y'*, the zeros would have been distributed with respect to*

$$dA \sim i^*(\omega_{\mathbb{P}^1_2}) \wedge i^*(\omega_{\mathbb{P}^2_3}) \wedge i^*(\omega_{\mathbb{P}^2_3}) \tag{5.6}$$

instead. Moreover, the intersection (5.2) *would have only led to two rather than three solutions per points sampled from* Y'*.*

This example illustrates how to sample points on the Calabi-Yau which will be further investigated in section 5.2. A more general treatment on how to extend this algorithm to also toric hypersurfaces will appear in the future [167].

5.1.2 Error measures

There are two performance measures on which numerical approximations of Ricci-flat metrics on CY manifolds are evaluated. They have been introduced in the context of the Donaldson algorithm and are respectively called sigma and Ricci measure [157, 158]. The sigma measure is computing the failure of the Monge-Ampère equation (3.10) integrated over the whole manifold:

$$\sigma = \frac{1}{\mathrm{Vol}_{\mathrm{CY}}} \int_X \left| 1 - \frac{\frac{J^3}{\mathrm{Vol}_{\mathrm{K}}}}{\frac{\Omega \wedge \bar{\Omega}}{\mathrm{Vol}_{\mathrm{CY}}}} \right| \mathrm{dVol}_{\mathrm{CY}}. \tag{5.7}$$

The Ricci measure integrates the Ricci-scalar (3.4)

$$||R|| = \frac{\mathrm{Vol}_{\mathrm{K}}^{\frac{1}{n}}}{\mathrm{Vol}_{\mathrm{CY}}} \int_X |R| \, \mathrm{dVol}_{\mathrm{K}}. \tag{5.8}$$

In the rest of this section we will discuss, first how the integration is carried out in the above equations and second how the two different volumes are defined.

The numerical nature of our solutions makes it impossible to carry out the integration analytically Instead we will continue numerically by Monte-Carlo integrating any quantity f over randomly sampled points on the manifold:

$$\int_X \mathrm{dVol}_{\mathrm{CY}} f = \int_X \frac{\mathrm{dVol}_{\mathrm{CY}}}{\mathrm{d}A} \mathrm{d}A f = \frac{1}{N} \sum_l^N w_l f|_{p_i} \tag{5.9}$$

In the above step integration weights w_l have been introduced which depend on the auxiliary measure dA coming from the distribution of points in the ambient space (in the example above this was given by equation (5.5)). There are two ways to compute the volume of the manifold, either with respect to $\Omega \wedge \bar{\Omega}$, which one finds by setting $f = 1$:

$$\mathrm{Vol}_{\mathrm{CY}} = \int_X \mathrm{dVol}_{\mathrm{CY}} = \int_X \Omega \wedge \bar{\Omega} = \frac{1}{N} \sum_l^N w_l \tag{5.10}$$

or by integrating with respect to the Kähler forms $t^i J_i$:

$$\mathrm{Vol}_{\mathrm{K}} = \int_X \wedge^3 t^i J_i = \frac{1}{N} \sum_l^N \left. \frac{\det(g)}{\Omega \wedge \bar{\Omega}} \right|_{p_l} w_l \stackrel{!}{=} d_{ijk} t^i t^j t^k =: V(t) \tag{5.11}$$

where g is the CY metric and d_{ijk} the triple intersection numbers (3.19). Note that with equation (3.10) these two volumes are equivalent up to the factor κ

$$\kappa = \frac{\mathrm{Vol}_{\mathrm{K}}}{\mathrm{Vol}_{\mathrm{CY}}}. \tag{5.12}$$

Hence the appropriately normalised integration weights for each point sampled by the procedure introduced in example 6 is given by evaluating V at Kähler moduli $t^i = 1 \forall i$:

$$w_i = V(1) \left. \frac{\mathrm{dVol}_{\mathrm{CY}}}{\mathrm{d}A} \right|_{p_i}. \tag{5.13}$$

5.1.3 Neural network ansatz

The CY metric tensor is given by a Hermitian (n,n) matrix. A naïve Ansatz is to let a neural network predict the n^2 components of the metric. The studies in paper V revealed that more efficient approximations are based on physical domain knowledge by learning corrections to some already known Kähler metric. Hence, in this thesis only metrics of the form

$$g_{\mathrm{pr}} = \tilde{g}_{\mathrm{FS}} + \mathrm{correction} \tag{5.14}$$

are considered. Here $\tilde{g}_{\mathrm{FS}}$ is the pullbacked metric tensor of the direct product of Fubini-Study ambient spaces metrics

$$\tilde{g}_{\mathrm{FS}} := i^* \left(\partial_i \partial_{\bar{j}} K_{\mathrm{FS}} \right) \tag{5.15}$$

	Ansatz
free	$g_{\text{pr}} = g_{\text{NN}}$
addition	$g_{\text{pr}} = \tilde{g}_{\text{FS}} + g_{\text{NN}}$
mult-elementwise	$g_{\text{pr}} = \tilde{g}_{\text{FS}} + \tilde{g}_{\text{FS}} \odot g_{\text{NN}}$
mult-matrix	$g_{\text{pr}} = \tilde{g}_{\text{FS}} + \tilde{g}_{\text{FS}} \cdot g_{\text{NN}}$
ϕ-model	$g_{\text{pr}} = \tilde{g}_{\text{FS}} + \partial\bar{\partial}\phi$

Table 5.1. *Different Ansätze involving neural networks for the metric tensor.*

with

$$K_{\text{FS}} = \frac{1}{\pi} \log \prod_{l}^{r} e^{t^l} \sum_{i}^{n_r} |z_{li}|^2 \tag{5.16}$$

This bias is motivated by the Monge-Ampère equation eq. (3.9). Table 5.1 shows five different Ansätze studied in paper V. The experiments conducted in paper V demonstrated that an exact correction in the form of eq. (3.9), dubbed the ϕ-model, gives the best results. The explicit Ansatz reads:

$$g_{\text{pred}} := \tilde{g}_{\text{FS}} + i^*(\partial_i \bar{\partial}_{\bar{j}} \phi) \tag{5.17}$$

where ϕ is a patch invariant scalar function modeled by an underlying neural network. The learned metric remains in the same Kähler class as the reference metric $\tilde{g}_{\text{FS}}$ as long as we consider exact corrections. This Ansatz has the further advantage of being Kähler by construction. The two derivatives are taken with automatic differentiation efficiently implemented in the `tensorflow` library. In contrast to the other models, it no longer requires to enforce kählerity which normally introduces another derivative in the training process.

5.1.4 Custom loss function

By the theorems due to Calabi and Yau there exists a unique Ricci-flat Kähler metric on the Calabi-Yau which satisfies the Monge-Ampère equations 3.9 and 3.10. Hence, the neural network has to solve the complex second-order non linear partial differential equation. It has been shown in the past that neural networks are capable of learning solutions to high dimensional non-linear PDEs [168–170]. These PDEs are solved by customising a loss function capturing the evolution of the system.

In similar vein `cymetric` implements a custom loss function encoding all the geometrical properties of the metric tensor. This loss function is then optimised with stochastic gradient descent and its cousins, introduced in section 2.2. The total loss is given by summing up the different contributions. It reads

$$\mathscr{L} = \alpha_1 \mathscr{L}_{\text{MA}} + \alpha_2 \mathscr{L}_{\text{dJ}} + \alpha_3 \mathscr{L}_{\text{transition}} + \alpha_4 \mathscr{L}_{\text{Ricci}} + \alpha_5 \mathscr{L}_{\text{vol-K}} \tag{5.18}$$

where α_i are hyperparameters (by default set to $\alpha_i = 1.0$) scaling the various terms. The equations and conventions presented in this section are compatible with version `v-0.2alpha`[1]. They are subject to changes in the future [167]. Some of these changes will be discussed at the end of section 5.2.

The first term

$$\mathscr{L}_{\text{MA}} = \left|\left| 1 - \frac{1}{\kappa} \frac{\det g_{\text{pr}}}{\Omega \wedge \bar{\Omega}} \right|\right|_n \tag{5.19}$$

encodes that eq. (3.10) needs to hold at every point on the manifold. $||\cdot||_n$ refers to the Fröbenius norm of degree n (not to be confused with the dimension n of the CY). The metric has to be Kähler

$$\mathrm{d}J = 0 \Rightarrow c_{i\bar{j}k} = \partial_k g_{i\bar{j}} - \partial_i g_{k\bar{j}} = 0 \tag{5.20}$$

which is captured by the *Kähler-loss*

$$\mathscr{L}_{dJ} = \sum_{i,j,k} \left|\left| \text{Re}\left(c_{i\bar{j}k}\right) \right|\right|_n + \sum_{i,j,k} \left|\left| \text{Im}\left(c_{i\bar{j}k}\right) \right|\right|_n . \tag{5.21}$$

Moreover, the metric should be well defined over all patch transitions $s \to t$. Therefore the *transition-loss* has to vanish

$$\mathscr{L}_{\text{transition}} = \frac{1}{d} \sum_{(s,t)} \left|\left| g_{\text{pr}}^{(t)} - T_{(s,t)} \cdot g_{\text{pr}}^{(s)} \cdot T_{(s,t)}^{\dagger} \right|\right|_n \tag{5.22}$$

where $T_{(s,t)} = \frac{\partial \vec{z}^{(s)}}{\partial \vec{z}^{(t)}}$ are the transition matrices between two patches (s,t) and d is the total number of patch transitions. The ϕ-model has a different transition-loss. The correction (5.17) has to be globally exact, which implies that the scalar function ϕ is scale invariant. Hence, for each batch we draw ten vectors of random numbers for each projective scaling relation, $\varepsilon_l^s \in \text{Uniform}(0.1, 0.9)$, and compute

$$\mathscr{L}_{\phi-\text{transition}} = \frac{1}{10} \sum_{s=1}^{10} \left|\left| g_{\text{pr}}(\vec{z}) - g_{\text{pr}}(\vec{z}_{\varepsilon^s}) \right|\right|_n \tag{5.23}$$

$$\text{where} \quad \vec{z}_{\varepsilon^s} := (\underbrace{\varepsilon_1^s, \ldots, \varepsilon_1^s}_{n_1}, \ldots, \underbrace{\varepsilon_r^s, \ldots, \varepsilon_r^s}_{n_r}) \odot \vec{z}.$$

The fourth term

$$\mathscr{L}_{\text{Ricci}} = ||R||_n = \left|\left| g_{\text{pr}}^{i\bar{j}} \partial_i \bar{\partial}_{\bar{j}} \ln \det g_{\text{pr}} \right|\right|_n \tag{5.24}$$

enforces the Ricci-scalar to vanish. It is by default disabled, but can be activated for manifolds where training against the MA-loss (5.19) is not sufficient.

[1] accessed 2022-03-14, commit: fa15b967b49c51113a911559bd7cec4e5e2c20ca.

The computation requires two additional derivatives with respect to the input coordinates, thus increasing computation time significantly. Finally, the metric should stay at the same overall volume as some reference metric. This is enforced by the *vol-k loss*

$$\mathscr{L}_{\text{vol-K}} = \left\| \int \det \tilde{g}_{\text{FS}} - \int \det g_{\text{pr}} \right\|_n , \tag{5.25}$$

where the integral is taken over the whole mini-batch.[2]

5.2 Experiments

This section contains the results of different experiments performed on a generic member of the CICY given by the following configuration matrix

$$X_{7880} = \left[\begin{array}{c||c} 1 & 2 \\ 1 & 2 \\ 2 & 3 \end{array} \right]^{3,75}_{-144} . \tag{5.26}$$

Motivated by the results of paper V only the two multiplicative ansätze and the ϕ-model presented in table 5.1 are used in the experiments. There are five experiments conducted for each Ansatz. The backbone neural network architecture consists of three fully connected hidden layers, initialised according to a normal distribution with $\sigma_\theta = 0.01$, GeLU-activation function, 128 hidden units and optimised with the Adam optimizer defined in eq. (2.16). The initial learning rate is $\eta = 0.001$ and the `tensorflow` default values for the momenta $\beta_{1/2} = \{0.9, 0.999\}$ are used. A learning rate decay schedule after 30 epochs of no improvement on the validation loss is initiated with a decay factor of 0.3.Training is done with a batch size of 64.[3] As a consequence of this small batch size the vol-k loss (5.25) has been disabled.

The experimental setup consists of 430000 points in the training set and respectively 20000 points in validation and test set. The validation data is used to track the training metrics, control the learning rate decay, examine the setup for overfitting, and some minor hyperparameter optimisation. The test data is a separate dataset on which the final performance is evaluated by computing

[2]The batch dependency makes this loss a bit more complicated to work with. The MC integral is not very sensible for small batch sizes, which in our experience work better in navigating the loss landscape. It is recommended to use an optimiser with momentum when training against this loss or employ a manual training loop which uses more points to compute the integral.

[3]While larger batch sizes better utilise the computation power of GPUs, such experiments showed a worse performance and often got stuck in bad local minima. This observation is in line with systematic studies of the batch size as a hyperparameter on image recognition tasks [171]. A batch size of 64 already gains computation time improvements from access to a GPU and still comes with favourable performance.

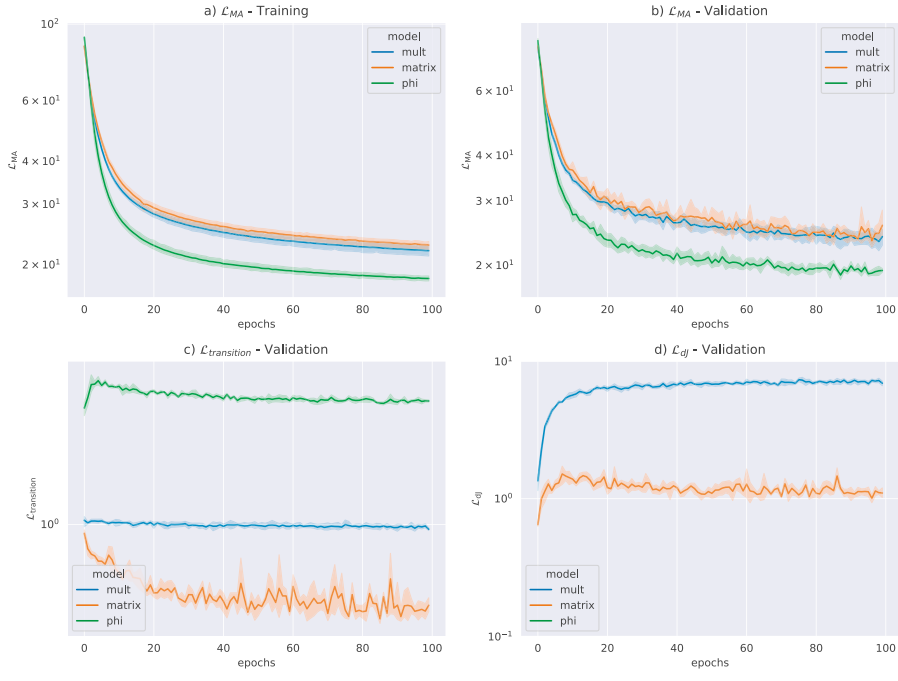


Figure 5.1. Experiments on X_{7880} defined in (5.26). The four subplots show a) Monge-Ampère loss on training b)+c)+d) Monge-Ampère, transition and Kähler loss on validation data.

the established benchmarks of sigma measure (5.7) and Ricci measure (5.8). The loss contributions of every sample in a mini-batch is scaled with respect to the integration weights (5.13).

In fig. 5.1 the loss contributions of five experimental runs for the different physical models are plotted against the training time of 100 epochs. 'phi' denotes the ϕ-model, 'mult' the element-wise multiplication and 'matrix' the matrix multiplication Ansatz. The y-axis uses a logarithmic scale in all four sub-plots. A comparison of the Monge-Ampère loss for the training data in subplot a) and for the validation data in subplot b) shows that the experiments are not overfitting to the training data. The ϕ-model outperforms both multiplication models. This observation matches the experiments in paper V.

Subplot c) shows the transition loss on the validation data. We recall that this loss is computed differently between the ϕ-model and the two multiplication models. Hence, only the latter two can be compared directly. The matrix model consistently decreases over the training process and has a lower transition loss than the element-wise multiplication. The latter remains approximately constant over the training. The ϕ-model first increases and then decreases again roughly to the same size it had at the beginning of the training

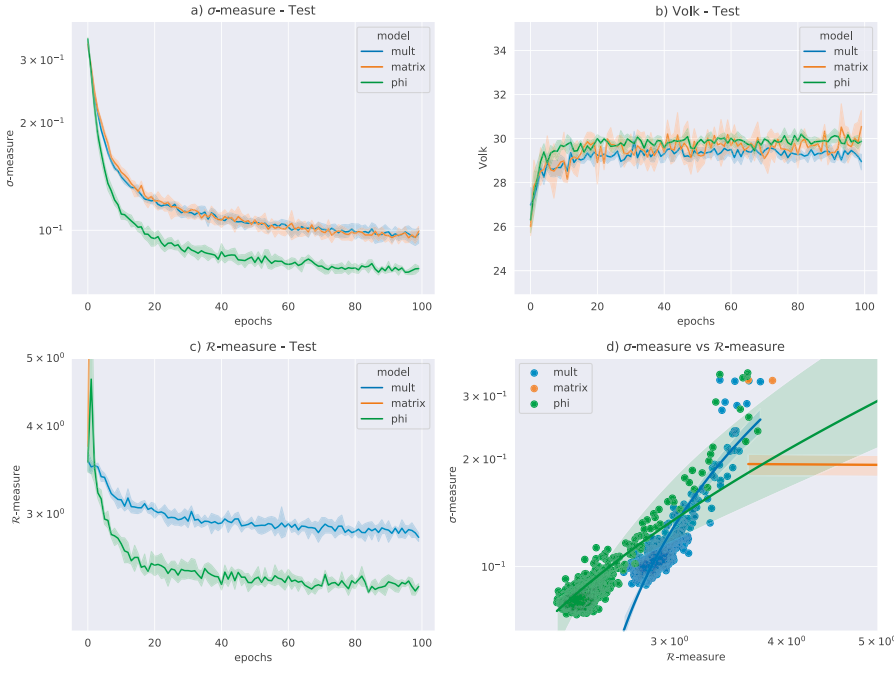


Figure 5.2. Experiments on X_{7880} defined in (5.26). The four subplots show a) σ-measure b) volume and c) $\mathscr{R}$-measure on test data and in d) the linear relationship between improvement in σ-measure and $\mathscr{R}$-measure.

process. Finally, subplot d) shows the Kähler loss for the two multiplication models: the matrix model is significantly better and also the only one that decreases over the training period. The ϕ-model is by construction Kähler such that this loss becomes trivial.

Figure 5.2 shows the σ- and Ricci-measure performance of the three models on a separate test set. The Monge-Ampère loss from the previous figure is mirrored in subplot a), which displays the σ-measure. Subplot b) shows the volume (5.11) computed from the test set over the course of the training. We observe that even though the volk-loss has been disabled the expected volume $\text{Vol}_K = 30$ is preserved simply by enforcing the Monge-Ampère equation with the correct κ value.

Subplot c) shows the Ricci measure on the test set. It comes with some surprising results. On the one hand the multiplication and ϕ-model become closer to the Ricci flat metric over the training process. On the other hand the matrix model does the opposite and increases its Ricci scalar over the training process. In fact even though the σ-measure is decreasing the Ricci measure increases monotonically as visible from the straight line in subplot d). Subplot d) plots the two benchmark measures against each other. Overall

we observe that the linear relation found in paper V does no longer describe the minimisation relationship between σ- and Ricci-measure accurately.

Outlook

The results displayed in fig. 5.1 and fig. 5.2 uphold that neural networks can be used to find numerical approximations of the unique Ricci-flat Calabi-Yau metric. Finding the right neural network weights is a multi-objective optimisation problem between the different loss contributions. This is an active area of research in machine learning. The Kähler and transition loss plots on the validation set in fig. 5.1 show that the experiments have not always been successful in minimizing all the different loss contributions individually. It would be interesting to test more modern approaches from the deep learning literature in handling the different optimisation objectives [172].

The increase in Ricci-measure for the matrix model further confirms that the loss landscape is highly non trivial. This is a new observation, which was absent for the Calabi-Yau manifolds investigated in paper V. It strongly suggests that future numerical experiments, which are not based on a formal convergence proof such as the Donaldson algorithm, have in addition to the σ-measure also keep track of the Ricci scalar. It also hints that optimisation against the Monge-Ampère loss (5.19) is not always sufficient in finding the Ricci-flat metric. Instead one has to also train against the Ricci-loss (5.24) for these models.

Overall our numerical solutions are closer to the Ricci-flat metric by factors of three to five compared to the Fubini-Study metric depending on benchmark and model. This is significantly worse than the improvements found in paper V. The discrepancy can in part be explained by the more complicated Calabi-Yau manifold investigated here. The chosen generic member of (5.26) does not admit a freely acting symmetry and contains significantly more complex structure moduli. With $h^{(1,1)} = 3$ it also comes with more Kähler moduli. Hence, one would expect that the underlying neural network also requires more parameters to accurately fit the Ricci-flat metric.[4]

It would be interesting to study scaling laws of the different models with respect to the number of parameters and dataset size. Scaling laws of neural networks have recently been investigated for natural language processing models, and different neural network architectures in image and speech recognition [173–177]. These studies revealed that the generalisation errors follow exact power-laws with respect to the neural network parameters if the experiments were not resolution limited by a too small dataset size.

Another promising approach for performance improvement is to modify the underlying neural network architecture. There are several modifications one could imagine, such as introducing feature engineering by adding monomials

[4]Doubling the width of the neural network to 256 units does lead to improved performance in some initial experiments not reported here.

of a fixed degree into the architecture or using more building blocks based on residual connections, batch normalisation or the attention mechanism. As described later in chapter 6 Calabi-Yau manifolds with a freely acting symmetry are particular important for physicists. Building architectures which respect these symmetries in the spirit of geometric deep learning [178] promise to be much more parameter and thus training efficient. Finally, it would be interesting to see whether one could find approximate analytic solutions via symbolic regression. Here, one should utilise methods presented in the context of AI Feynman [179] to generate these expressions which have a proven track record of learning symbolic equations.

In the experiments of this section the vol-k loss (5.25) has not been utilised. In the future [167] we intend to change the vol-k loss to also train against the slope of various line bundles (3.28). This modification fixes the Kähler class of the multiplicative models. Finally, the plan is to include new customised training loops which evaluate the Monte-Carlo integrals with a sensible number of points. These loops will then run concurrently over two different batch-sizes, one for the MC integral and another smaller one for the other loss contributions.

Part II:
Heterotic model building

The second part of this thesis covers heterotic model building. It focuses on so called heterotic line bundle models, which are the underlying construction of the largest class of SLMs. First, their basic construction is reviewed and demonstrated with an explicit example. This is followed by a summary of recent results obtained on gCICY manifolds. In the last chapter deep reinforcement learning is applied to the problem. In particular Actor-Critic agents are trained on so far unexplored CICYs discovering 19 538 new SLMs.

6. Heterotic model building

String theory model building has a long history since the introduction of the heterotic string [2, 3]. The first construction used the Quintic Calabi-Yau manifold (3.21) as compactification space and standard embedding of its tangent bundle into one of the E_8-factors to arrive at four generation E_6 GUT model [180]. This approach has been quickly refined to generate models with three fermion generations and further GUT breaking to the standard model [181–183]. Over the years more phenomenological constraints have been incorporated in fine tuned selected examples [184–186].

The process of finding semi-realistic string vacua, which match the gauge group and particle content of (minimal) supersymmetric extensions of the standard model, follows an established cooking recipe. In a first step one selects their favourite Calabi-Yau manifold X, which admits a freely acting symmetry Γ. Algorithmic classifications of freely acting symmetries on CICYs are found in the literature [187, 188]. In the next step one considers slope stable vector bundles V with vanishing first Chern class and rank $\text{rk}(V) = \{3,4,5\}$ to break one of the E_8-factors into the GUT groups $G = \{E_6, SO(10), SU(5)\}$. This vector bundle has to satisfy mathematical and phenomenological consistency requirements, such as anomaly cancellation and the existence of three massless fermion generations. In a last step one breaks the GUT group G via a Wilson line utilising Γ to the gauge group of the standard model.

This cooking recipe allows for systematic exploration of different vacua by considering a set of Calabi-Yau manifolds and scanning over different geometric configurations for the vector bundle. The scanning algorithm is depicted in algorithm 1. Early results yielded datasets of 300 phenomenological constrained MSSM models in $Z_6 - II$ orbifolds [189, 190] and 91 standard models using the positive Monad construction [191] over CICYs. The latter models turned out to not admit the precise particle content of the standard model after Wilson line breaking. This negative results led to the consideration of line bundle sums $V = \oplus L_a$, which would turn out to be the most successful approach for heterotic compactifications [192].

Large scale systematic scans of heterotic line bundle models over Calabi-Yau manifolds have led to the largest explicit dataset of semi-realistic string theory compactifications [192]. They have been performed over all CICYs with $h^{(1,1)} = \{1,\dots 7\}$ leading to 1 428 856 models [127, 192–194]. Further scans were carried out over all known hypersurfaces in toric varieties admitting a freely acting symmetry [195], and elliptically fibered Calabi-Yau manifolds [196]. In this chapter the line bundle sum construction is reviewed

Algorithm 1 Pseudo algorithm to find realistic heterotic compactification.

Require: X is a CY three-fold and $\tilde{X} = X/\Gamma$ s.t. $\pi(\tilde{X}) = \Gamma \neq 0$.
  $Vs \leftarrow$construct_all_V(p)
  $good \leftarrow []$
  while $\exists V$ in Vs **do**
    if V satisfies consistency conditions **then**
      if V satisifes phenomenological conditions **then**
        $good \leftarrow [V]$
      end if
    end if
  end while

in section 6.1. The next section 6.2 presents recent developments in finding such models on gCICYs based on paper III. The final chapter 7 uses deep reinforcement learning to explore new CICYs, which are out of computational reach for systematic studies with $h^{(1,1)} \geq 7$.

Heterotic model building is not the only subfield of string theory in which semi-realistic compactifications have been constructed. It has been estimated that one in a billion intersecting D6-brane models in the IIA setting leads to realistic particle content [197, 198]. Moreover, recently it was shown that there are F-theory compactifications which allow for a quadrillion realisations of the standard model. This feat was accomplished by finding vacua with the correct particle content on a Calabi-Yau four-fold admitting that many distinct triangulations [199].

6.1 Heterotic line bundle models

This section reviews the construction of heterotic standard like models (SLM) via line bundle sums as introduced in [127, 193, 194]. These heterotic line bundle models satisfy a set of different mathematical consistency checks and phenomenological constraints. In large scale scans they are implemented algorithmically as shown in algorithm 1 to collect all physical interesting models. The starting point is a smooth Calabi-Yau manifold X admitting a freely acting symmetry Γ. From this upstairs manifold one can get a downstairs manifold $\tilde{X}$ with a non-trivial fundamental group $\pi(\tilde{X}) = \Gamma$ by considering the quotient $\tilde{X} = X/\Gamma$. The cooking recipe presented in this section also generalises straightforwardly to elliptic and toric constructions [195, 196]. The vector bundle V is constructed as a sum of five line bundles L_a:

$$V = \bigoplus_{a=1}^{5} L_a = \bigoplus_{a=1}^{5} \mathscr{O}_X(q_a^1, ..., q_a^r) \tag{6.1}$$

The charges q_a^i need to satisfy $c_1^i(V) = \sum_{a=1}^5 q_a^i = 0$ such that $c_1(V) = 0$ vanishes. Then V has structure group $S(U(1)^5)$, which gives after embedding into one of the heterotic E_8-factors the intermediate GUT group

$$E_8 \supset SU(5) \times S(U(1)^5) \cong SU(5) \times U(1)^4. \tag{6.2}$$

The $SU(5)$ GUT group will be further broken with a Wilson line to the standard model. This leaves additional $U(1)$ vector bosons coming from the Abelian split-locus. They can become massive via the Green-Schwarz mechanism [193] and have the following mass matrix

$$M_{ab} = -c_1^i(L_a)c_1^j(L_b)\partial_i\partial_j \ln(K) \tag{6.3}$$

where K is the Kähler potential of the CY [193]. Hence, there are $4 - \text{rank}(V)$ additional non-anomalous massless $U(1)$-charges that have to acquire mass by switching on VEVs of the vector bundle moduli [127].

The 10D Bianchi identity introduces further integrability constraints in the form of anomaly cancellation. This anomaly cancellation is encaptured in a condition on the second Chern class for slope zero stable bundles

$$0 < c_2(V) \leq c_2(X) \tag{6.4}$$

$$\text{with:} \qquad c_{2i}(V) = -\frac{1}{2} d_{ijk} \sum_{a=1}^{5} q_a^j q_a^k \tag{6.5}$$

where d_{ijk} are the triple intersection numbers (3.19) and c_{2i} is to be compared with the vectorised second Chern class (3.20) of the underlying CICY. The lower bound arises by Bogomolov [200] and a direct matching is not required, since NS-5 branes can be added to the theory to cancel out the remaining anomalies [201]. These constraints hold for slope stable bundles. Slope stability is furthermore required by the Donaldson-Uhlenbeck-Yau theorem [125, 202] in order to preserve supersymmetry in the lower dimensional theory. Hence, at some supergravity compatible point in the Kähler cone the slope of V needs to vanish. This condition is expressed in terms of the Kähler moduli t^i as

$$\exists t^1, t^2, ..., t^{h^{(1,1)}} > 1 : \mu(L_a) \stackrel{!}{=} 0, \forall a \in \{1, ..., 5\} \tag{6.6}$$

and the slope is defined as in equation (3.28). As a consequence of this condition most CICYs with $h^{(1,1)} \leq 4$ won't admit any realistic line bundle models. For CICYs with $h^{(1,1)} > 4$ equation (6.6) is satisfied for generic values of the charges.

The **248** adjoint representation of E_8 decomposes under $G \times H = SU(5) \times SU(5)$ into

$$(\mathbf{1}, \mathbf{24}) \oplus (\mathbf{5}, \mathbf{10}) \oplus (\bar{\mathbf{5}}, \bar{\mathbf{10}}) \oplus (\bar{\mathbf{10}}, \mathbf{5}) \oplus (\mathbf{10}, \bar{\mathbf{5}}) \oplus (\mathbf{24}, \mathbf{1}) \tag{6.7}$$

multiplet	l.b. cohomologies counting multiplicities	contained in
10	$h^1(X,L_a)$	V
$\bar{\mathbf{10}}$	$h^1(X,L_a^*)$	V^*
5	$h^1(X,L_a\otimes L_b)$	$V\wedge V$
$\bar{\mathbf{5}}$	$h^1(X,L_a^*\otimes L_b^*)$	$V^*\wedge V^*$
1	$h^1(X,L_a\otimes L_b^*)$ and $h^1(X,L_a^*\otimes L_b)$	$V\otimes V^*$

Table 6.1. *Relevant particle content of the $SU(5)$ GUT theory and their multiplicities given by the dimensions of line bundle cohomology groups [193].*

The multiplicity of the particle representations is given by the Hodge numbers of certain line bundles as presented in table 6.1. Since the fundamental group Γ of the quotient manifold $\tilde{X} = X/\Gamma$ is non zero one can use a Wilson line to further break down the gauge group to the standard model [4]. The invariant part of $H^1(X,V)\otimes R_W$, with R_W a Wilson line representation, determines the particle multiplicities in the downstairs theory. Thus, for each $L_a \in V$

$$\chi(L_a) \mod |\Gamma| = 0 \tag{6.8}$$

$$\text{where } \chi(L_a) = d_{ijk}\left(\frac{1}{6}q_a^i q_a^j q_a^k + \frac{1}{12}q_a^i c_2^{jk}(X)\right) \tag{6.9}$$

has to hold for there to exist an equivariant structure on the quotient. The downstairs particle content is then

$$\#\text{fermion generations} = -\frac{\chi(V)}{|\Gamma|}. \tag{6.10}$$

Phenomenological observations impose the existence of exactly three fermion generations and no anti-families. This requirement translates to a constraint on the dimension of the cohomology groups

$$h^\bullet(V) = (0,3|\Gamma|,0,0). \tag{6.11}$$

Moreover, the MSSM particle content requires the existence of at least one pair of Higgs doublets, but no Higgs triplets. These conditions give rise to the following topological constraints [193]

$$\text{doublet: } h^2(\wedge^2 V) > 0, \qquad \text{triplet: } \chi(L_a\otimes L_b) \leq 0\ \forall a,b. \tag{6.12}$$

In principle one could impose more constraints, such as moduli stabilisation and the matching of Yukawa couplings. However, each additional constraint adds further computational complexity to the problem and at parts relies on technology which hasn't been sufficiently developed yet. For example, computation of the physical Yukawa coupling requires the Calabi-Yau metric and

solutions to the Hermitian Yang-Mills equation. The necessary toolkits for these computations are only now getting developed as outlined in chapter 5. Stabilisation of the moduli requires expensive Gröbner basis computations and additional scans over suitable vector bundle configurations in the hidden sector [203, 204]. To the best of my knowledge no analysis of SLMs with respect to their full moduli stabilisation and physical Yukawa couplings has been done yet. There are, however, promising initial results studying jumping spectra required for moduli stabilisation and topological constraints for the vanishing of Yukawa couplings in heterotic line bundle models [205].

Example 7. *Consider the following configuration matrix*

$$X_{7447} \in \left[\begin{array}{c||cc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right]^{5,45}_{-80} \tag{6.13}$$

belonging to the CICY with number 7447. It admits freely acting $\mathbb{Z}_2, \mathbb{Z}_2 \times \mathbb{Z}_2, \mathbb{Z}_5, \mathbb{Z}_{10}$ *and* $\mathbb{Z}_{10} \times \mathbb{Z}_2$ *symmetries [187]. The triple intersection numbers and second Chern class are*

$$d_{ijk} = 2 \cdot |\varepsilon_{ijk}| \quad \textit{and} \quad c_2^i = (24,24,24,24,24). \tag{6.14}$$

We want to study the low energy physics of the vector bundle given by a sum of five line bundles [193]

$$\begin{aligned} V = (0,1,0,-2,1) \oplus (0,1,-2,1,0) \oplus (0,0,1,1,-2) \\ \oplus (0,-1,1,0,0) \oplus (0,-1,0,0,1). \end{aligned} \tag{6.15}$$

This vector bundle has vanishing first Chern class and we can show that the slope (6.6) *vanishes simultaneously for all line bundles at e.g.:*

$$t^i = 1 \; \forall i \tag{6.16}$$

Hence, it is poly-stable. The Bianchi identity in equation (6.4) *is checked by comparing the second Chern class of the manifold in* (6.14) *and the second Chern class of V*

$$c_2(V) = (22,14,10,10,10) \leq c_2(X_{7447}) = (24,24,24,24,24). \tag{6.17}$$

The index of V is computed with equation (3.31) *to be*

$$\chi(X_{7447}, V) = \sum_a \chi(X_{7447}, L_a) = -4-4-4+0+0 = -12. \tag{6.18}$$

Thus leading to a three generation model when quotienting out with a discrete symmetry of rank $|\Gamma| = 4 = |\mathbb{Z}_2 \times \mathbb{Z}_2|$. *In fact the cohomology computations*

reveal that there are no antigenerations as required by constraint (6.11)*. The Hodge numbers of only the first and fourth line bundle are required as the others are permutations of those under which the configuration matrix is invariant. Applying the Koszul resolution* (3.23) *and splitting the long exact sequence into short exact sequences via the Leray tableaux 8 one finds*

$$h^\bullet(X_{7447}, \mathscr{O}(0,1,0,-2,1)) = (0,4,0,0) \tag{6.19}$$
$$h^\bullet(X_{7447}, \mathscr{O}(0,-1,1,0,0)) = (0,0,0,0) \tag{6.20}$$

which shows that there are no anti-generations. Finally the number of Higgs doublets and triplets are computed. The indices of all line bundle products reveal that the triplet condition is satisfied. There exists a single line bundle $L_1 \otimes L_5$ *for which the second cohomology group does not vanish*

$$h^\bullet(X_{7447}, \mathscr{O}(0,0,0,-2,2)) = (0,3,3,0). \tag{6.21}$$

leading to at least one pair of Higgs doublets. In summary, the proposed line bundle sum in (6.15) *satisfies all conditions outlined before.*

6.2 Model building on gCICY manifolds

In paper III we constructed heterotic line bundle models on gCICYs introduced in section 3.2.3. We proceeded as follows: First we identified two promising gCICY manifolds X with favourable configuration matrices and $h^{(1,1)} > 4$. The manifolds are given by

$$X_{1g} \in \left[\begin{array}{c||c|c} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 3 & -1 \end{array}\right]^{5,45}_{-80} \quad \text{and} \quad X_{2g} \in \left[\begin{array}{c||c|c} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 2 \\ 1 & 3 & -1 \end{array}\right]^{5,29}_{-48}. \tag{6.22}$$

A comparison of topological quantities identifies X_{1g} to be equivalent to two CICY realisation, which have been studied in the past

$$X_{7447} \in \left[\begin{array}{c||c|c} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right]^{5,45}_{-80} \quad \text{and} \quad X_{7487} \in \left[\begin{array}{c||c|c} 1 & 0 & 2 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right]^{5,45}_{-80}, \tag{6.23}$$

where X_{7447} is the same manifold as studied in example 7. X_{2g} on the other hand has no CICY realisation with the same Hodge numbers. There are four polytopes in the Kreuzer-Skarke list with a total of 19 triangulations matching $h^{(1,1)}$ and $h^{(2,1)}$. None of these four polytopes is known to admit a freely

acting symmetry and thus they haven't been utilised in the context of the heterotic line bundle program [195]. Moreover, a simple comparison of triple intersection numbers needed for Walls theorem 6 suggests that X_{2g} is indeed a topological new manifold not appearing in any of the standard CY datasets.

In a second step freely acting symmetries on the two gCICYs leading to smooth quotient manifolds were identified. Both geometries admit a freely acting $\mathbb{Z}_2$ symmetry generated by

$$g : z_{i,j} \to (-1)^{j+1} z_{i,j} \tag{6.24}$$

acting on the homogeneous coordinates, $z_{i,j}$, of the $\mathbb{P}^1_i$ for $i = 0,\ldots,4,\ j = 0,1$ and also acting on the constraints with

$$g_{\text{constr}} : p_i \to (-1)^i p_i. \tag{6.25}$$

We then implemented an algorithm to compute the cohomology dimensions on $X_{1/2g}$ by relating them to the embedding hypersurfaces $M_{1/2g}$. This allowed us to scan over vector bundles in an iterative manner for all possible line bundles with a given maximal charge q_{max}. The scan was aborted when the number of allowed line bundles hadn't changed for three increases in q_{max} [194].

In total 99 and 33 models were found on X_{1g} and X_{2g} respectively. A comparison of the number of the Higgs doublets, anomaly cancellation conditions, and vector bundle singlets suggests that the 99 models on X_{1g}, and also the previously found models on X_{7447} are different realisations of the models found on X_{7487}. The 33 models on X_{2g} are genuinely new SLMs and the first constructed models on a gCICY manifold. In summary, we were able to show that gCICY manifolds can be used for model building, but might be less suitable for it, due to significantly more involved computations and less abundant number of models.

7. Exploring new heterotic line bundle models

Finding string theory solutions that match the particle content of the observable universe has been one of the main motivations for string phenomenology. However, the vast choices of possible configurations from selecting a Calabi-Yau manifolds to flux, brane or vector bundle data puts severe challenges on identifying the correct data describing our universe. Moreover, the Calabi-Yau landscape is dominated by manifolds with large values of $h^{(1,1)}$ [15], which makes it impossible to perform systematic exhaustive scans.

An alternative approach is to study the statistics of known realistic configurations to make some general statements. It requires a representative set of solution generated by some means. As brute force scans are not possible and random selection of configurations is highly unfeasible[1] string theorists are in looking for smart ways to find relevant configurations. The currently two most popular approaches to identify new SLMs are genetic algorithms [65, 67–70, 206] and reinforcement learning [58, 60, 61, 67, 68].

In the previous chapter a fruitful compactification scheme in the form of heterotic line bundle models has been introduced. The data on known solutions suggest that the number of SLMs scale exponentially with $h^{(1,1)}$ [192]. This is not surprising as the line bundle configurations to be scanned over also scale exponentially with $h^{(1,1)}$. Hence, for even relatively small values of $h^{(1,1)}$ it becomes impossible to perform systematic scans. In this chapter new SLMs are found with the help of deep reinforcement learning introduced in section 2.3.

The reinforcement learning experiments are implemented in an environment given by the `gymCICY` package. The package is build on the OpenAI gym [207] standard and has been designed and developed in the context of paper II. It is available on GitHub

`https://github.com/robin-schneider/gymCICY`

Reinforcement learning agents are tasked to explore the setting of heterotic line bundle models introduced in section 6.1 and find interesting configurations. After fixing an initial manifold, the agents observe an random line bundle sum (6.1). They interact with the environment by manipulation of the line bundle charges. They get punished or rewarded for finding increasingly more realistic configurations.

[1] The search for semi-realistic configurations can be compared to finding the needle in a haystack (or worse). Estimates for the occurrence of SLMs in string theory start at one in a billion [198].

In paper II Actor-Critic agents have been used to re-discover known SLMs on selected manifolds [194]. Advantage Actor-Critic agents are introduced in section 7.1. It was found that the experiments scale well with $h^{(1,1)}$, thus opening up the venue for studies on so far unexplored manifolds. Such experiments on CICYs with $h^{(1,1)} \geq 7$ are presented in section 7.2. Moreover, the experiments of paper II suggested that transfer learning can be used to speed-up the training process. This line of research is not further pursued in this thesis, as it was limited to CICYs of the same $h^{(1,1)}$ value.

7.1 Actor-Critic agents

In this section Actor-Critic agents are reviewed in the popular form of (Asynchronous) Advantage Actor-Critic agents denoted as A2/3C [78, 208]. The actor is given by a neural network that learns the policy function $\pi(a_t, s_t; \theta^a)$ while the critic network learns the state value function $v_\pi(s_t; \theta^c)$. Both networks interact with each other. The gradient updates for the actor depend on the prediction of the critic and the observed state for the critic is determined from the actions taken by the policy. Actor-Critic agents outperform most conventional reinforcement learning methods while being significantly more sample efficient and stable. The asynchronous variant runs efficiently on a multi-core CPU, while the synchronised version can utilise the computation power of a GPU. In the following the RL setting is fixed to have a finite episode length and gradient updates occur after $t_{\max}$-steps or at a terminal state.

The gradient updates are computed backwards from state t to $t - t_{\max}$. For each time step $i = t-1, \ldots, t-t_{\max}$ the gradient errors $d\theta$ are accumulated. The final gradient update is done with RMSprop, a variation of stochastic gradient descent introduced in section 2.2. Define

$$R_{i+1} \leftarrow r_i + \gamma R_i \tag{7.1}$$

with $R_t = v(s_t; \theta^c)$. The accumulated gradients for the critic are simply given by the squared difference between observed and expected reward

$$d\theta^c_{i+1} \leftarrow d\theta^c_i + \frac{\partial (R_i - v(s_i; \theta^c))^2}{\partial \theta^c}. \tag{7.2}$$

In contrast to the REINFORCE algorithm from section 2.3 the policy is also updated every $t_{\max}$ steps. The loss function introduces two modifications. First a baseline in the form of the advantage function

$$A(a_i, s_i; \theta^c) := R_i - v(s_i; \theta^c) \tag{7.3}$$

is used. Second the entropy over actions (2.11) is added to the loss to further encourage exploration. The accumulated gradients are then

$$d\theta^a_{i+1} \leftarrow d\theta^a_i + \nabla_{\theta^a} \left(\log \pi(a_i | s_i; \theta^a) A(a_i, s_i; \theta^c) + \beta H(\pi(a_i | s_i; \theta^a)) \right) \tag{7.4}$$

condition	reward
vanishing first Chern class	trivial
vanishing line bundle slope (3.28)	2
index constraint, three fermion generations (6.8)	10^2
Bianchi identity (6.4)	10^5
no Higgs triplets (6.12)	10^5
existence of Higgs doublets (6.12)*	10^6
no antigeneration (6.11)*	10^7
full stability (6.6)	10^7

Table 7.1. *Consistency checks for a line bundle sum and their rewards in* `gymCICY`*. Constraints marked with a (*) require lengthy cohomology computations and have been disabled for the experiments in section 7.2.*

where the hyperparameter β regulates the entropy contribution. There are multiple agents exploring different instances of the environment. The accumulated gradient updates are then averaged in the synchronous setting or alternatively they asynchronously update a global set of parameters for the two networks. This exploration of multiple instances makes the learning process more robust and further helps exploration if the initial state or the actions are chosen non deterministically.

7.2 Exploring uncharted territories

This section reports the results of RL studies investigating heterotic line bundle models on three CICYs with freely acting $\mathbb{Z}_3$-symmetry and $h^{(1,1)} \geq 7$. These CICYs have not been searched for SLM previously as their configuration matrices are not part of the regular CICY list. Furthermore, $h^{(1,1)} \geq 8$ is beyond systematic reach [192] which includes two of the CICYs considered in this section and is hence an interesting region for exploration with RL agents. The favourable configuration matrices investigated here admit $\mathbb{Z}_3-$symmetries, which are found by applying repeated effective splitting from the starting configuration matrix

$$X[3,48] = \left[\begin{array}{c||ccc} 2 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 \end{array}\right]^{3,48}_{-90} . \tag{7.5}$$

This manifold in turn is just an effective split away from the bi-cubic manifold from which the initial $\mathbb{Z}_3-$symmetry descends. This web of splitting has been studied extensively in Refs. [209, 210].

The A2C agents will explore the **flipping** environment of the `gymCICY` library. In paper II it was shown that this environment allows for an exploration

of a wider class of solutions. The RL agents manipulate the charges of each line bundle in the line bundle sum (6.1) by either increasing or decreasing a value with ± 1. Cyclic boundary conditions are applied when going beyond some maximal charge $q_{\max}$. The fifth line bundle is fixed in such a way that $c_1(V) = 0$ is always satisfied to improve the initial learning. The observation state is independent of the history and contains all necessary information, thus satisfying the Markov property of an MDP. There are a total of

$$(2 \cdot q_{max} + 1)^{4 \cdot h^{(1,1)}} \tag{7.6}$$

observation states.

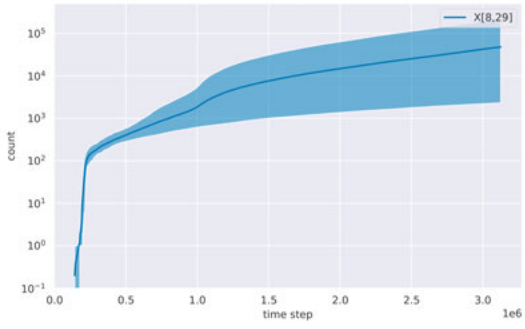
The default rewards for satisfying the different consistency conditions are presented in table 7.1. The constraints involving cohomology computations have been disabled in the following experiments. These take an extraordinary amount of time at larger $h^{(1,1)}$ values. In particular when using the synchronous version of A2C-agents it would result in plenty idle time, where the other agents are waiting on one agent to finish such a computation. Hence, all cohomology constraints have to be checked manually afterwards and the models presented here are less restrictive than what was investigated in paper II.

7.2.1 Experiments

This section presents results of five averaged runs with different seeds utilizing A2C agents implemented in `stable-baselines` [211][2]. The experimental configurations are as follows: the update rate is $t_{max} = 5$, two separate hidden layers are used for both policy and value function with $n_h = 128$ and ReLU activation function. The exploration parameter is $\beta = 1.$ and a learning rate of $\eta = 5 \times 10^{-4}$ with double linear decay is used for the RMSprop optimiser. Gradients are clipped with respect to five times the norm and a weight decay of 10^{-2} with momentum $\alpha = 0.99$ is used. Finally, the numerical stabilisation value is chosen to be $\varepsilon = 10^{-4}$. The max episode length is 300 for a total of 3.2×10^6 steps.

Hyperparameter optimisation was done with a combination of initial random and box search to bound the priors. Many of the default values used in `stable-baselines` [211] already provided good results. The exploration parameter β required more careful fine tuning, which was done with BOHB [139]. While hyperparameter optimisation was performed on all three manifolds, the results presented here share the same experimental configurations. Performance was not notably different between different configurations, which suggests that the selected values are relatively stable and should also work on other manifolds.

[2]Experiments with newer algorithms such as ACER [212] and ACKTR [208] had similar performance at the cost of increased training time.



run	all	unique
1	3576	2863
2	9577	4451
3	2296	1973
4	3554	2759
5	220071	9289
total	-	14374

Figure 7.1. Results of five experiments with same hyperparameter configurations but different seed on the manifold given by configuration matrix (7.7)*. Note the logarithmic scaling on the y-axis. The plots shows the total number of found models plotted against a global step counter.*

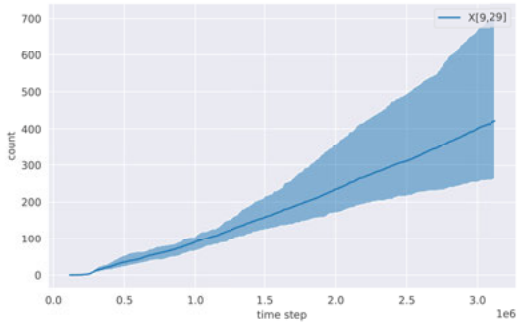
X[8,29]

Consider the CICY given by the following configuration matrix

$$X[8,29] = \left[\begin{array}{c||cccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]^{8,29}_{-42} \tag{7.7}$$

Figure 7.1 shows the found models at the global time step t for five experimental runs with different seeds. The plot has a logarithmic y-axis because of a large discrepancy in models found between the different runs. Four of the experiments find less than 10^4 models while the last one discovers 2.2×10^5. The majority of these models are, however, not unique as shown in the table to the right. After removing duplicates and permutations of the line bundles the last experiment still finds more SLMs than the other experiments but now at the same order of magnitude. The majority of models found by the other agents are unique.

The agent corresponding to the fifth seed has developed a strategy to recover successfully memorised solutions. The observed discrepancy in performance for the same hyperparameters is often criticised in deep reinforcement learning experiments [213]. It is the reason why experimental runs consisting of a set of at least five experiments are reported rather than showing the results of a single fortunate run [214, 215].



run	all	unique
1	705	571
2	263	246
3	452	413
4	363	344
5	321	305
total	-	1643

Figure 7.2. Results of five experiments with same hyperparameter configurations but different seed on the manifold given by configuration matrix (7.8)*. The plots shows the total number of found models plotted against a global step counter.*

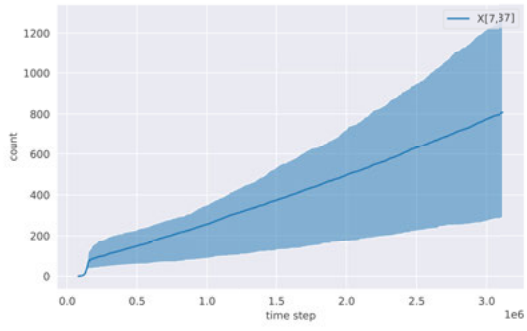
In total 14 374 new SLMs have been found in $5 \cdot 3.2 \times 10^6$ time steps. This is vastly more performant than the one-in-a-billion thumb rule. Note, that the total number of unique models also removes duplicates occurring in the other runs. To compare the performance of the RL agents a set of five further experiments was initiated with a uniform random policy. These random walkers were not able to detect a single SLM underlining the success of the A2C agents.

X[9,21]

The next CICY to be studied is

$$X[9,21] = \left[\begin{array}{c||cccccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right]^{9,21}_{-24} . \tag{7.8}$$

A similar plot of the results of five experimental runs can be found in fig. 7.2. The experiments are less successful than on the previous manifold. The performance of all five agents is comparable and they mostly find unique models in the search for viable vacua. The plot shows that the agents need a brief exploration period of 0.2×10^6 steps before they begin to consistently identify solutions. Afterwards the increase in models appears to be linear for the not so good performers and slightly more accelerated for the better agents. In total



run	all	unique
1	808	778
2	288	283
3	825	783
4	1275	1205
5	843	809
total	-	3521

Figure 7.3. Results of five experiments with same hyperparameter configurations but different seed on the manifold given by configuration matrix (7.9)*. The plots shows the total number of found models plotted against a global step counter.*

1643 new SLMs were discovered on this CICY. Again five random walkers running for the same number of total steps were unable to discover a single model.

X[7,37]
The last CICY investigated is

$$X[7,37] = \left[\begin{array}{c||cccccccc} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 2 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}\right]^{7,37}_{-60} \tag{7.9}$$

which has a smaller $h^{(1,1)}$-value still in the range of large distributed scans on a cluster [192]. The results are plotted in fig. 7.3. They show a sharp initial increase in found models at roughly 0.1×10^6 and slightly sub optimal performance afterwards with an approximated linear increase in found models per passed time. The agents predominantly find unique models, leading to a total of 3521 new SLMs. As was the case for the two previous manifolds random walkers were unable to identify a single good solution.

Summary
In this chapter the success of deep RL agents in finding interesting string theory compactifications was reported. These studies build up on the explorations initiated in paper II and showed that deep RL also works well in regimes with

larger values for $h^{(1,1)}$. In total 19 538 new SLMs have been found. Comparing the results to the plots of paper II A2C-agents appear to be more stable in their performance than A3C agents. This improved stability could, however, also be a consequence of the missing cohomology computations. These introduce computation time delayed rewards for good configurations in the asynchronous updates, which might break stable policies. Another difference is the larger $h^{(1,1)}$ values which in principle should allow for more models.

In this thesis CICYs with $\mathbb{Z}_3$-symmetries have been investigated. The results from systematic scans in the literature suggest, that configurations with $\mathbb{Z}_3$-symmetry are less suitable for finding SLMs [194]. Nevertheless, the agents were able to identify plenty of the solutions. While the results presented here do not account for a systematic analysis, they support the trend of increasing SLMs for larger Hodge numbers.

8. Conclusion

This thesis makes a case for using machine learning to solve problems in string theory compactifications. Three distinct problems with applications of deep learning have been discussed: first the learning of topological quantities, such as the Hodge numbers of Calabi-Yau manifolds, second the learning of the local metric tensor of Calabi-Yau manifolds and third using agents from deep reinforcement learning to find SLMs.

The multi-task CICYMiner demonstrated that deep neural networks are able to accurately predict dimensions of vector bundle cohomologies over a compactification space. These computations conventionally require advanced tools from algebraic geometry and topology. Using deep learning has the potential to improve computation time significantly. The success in learning Hodge numbers leads to the following question: which kind of topological invariants can a deep neural network learn? A natural follow up question to investigate is whether it is possible to identify identical representations of the triple intersection numbers. This would lead to new insights into the connected web of CY manifolds and possible vacua arising in string theory, possibly limiting the vast configuration space that need to be studied.

The `cymetric` package simplified the problem of finding Ricci-flat metrics to a straightforward machine learning task of finding good hyperparameters and a working neural network architecture. Deep learning is directly used to find numerical approximations of unknown quantities with much greater accuracy than previous methods in the literature. It would be interesting to study how this framework can be extended to more general SU(3)-structure metrics [42, 216] or to other manifolds of special holonomy. The modularity of `cymetric` allows for other use cases, such as learning symbolic approximations of the metric tensor or finding an expression of the Kähler potential. For the future it is planned to expand the functionality to also include solutions to the Hermitian Yang-Mills equation, which is the next step in computing heterotic Yukawa couplings. A regular user can already now use the package to test conjectures from the swampland program [217, 218] or probe the SYZ-conjecture [219].

In the last chapter deep reinforcement learning was used to find new SLMs. 19538 such models have been found on three so far unexplored manifolds. In this process neural networks efficiently learned to solve a long list of reward constraints including undecidable Diophantine equations. While RL will not find all possible models like a systematic scan does, this approach is nevertheless useful to get statistics over broad classes of manifolds which are out of

reach for systematic studies. Statistics of compactifications at very large $h^{(1,1)}$ have been collected previously in the literature [220].

To make generalisable predictions from datasets found by RL agents it is important to study the bias of these solutions. Initial studies in that directions were done by comparing them to the results found with genetic algorithms [67, 68]. In the first paper, the authors found that the solutions appear to be from quite different distributions and thus not necessarily generalisable to all realistic vacua. In the second paper though the authors found evidence for the opposite and an overwhelming match in the solutions. Certainly more comparison studies are required in the future to resolve this disagreement.

9. Svensk Sammanfattning

De fyra fundamentala naturkrafterna beskrivs av två extremt framgångsrika teorier. Partikelfysikens standardmodell redogör för de elektromagnetiska, svaga, och starka krafterna. Den beskriver partiklarnas interaktion på en mikroskopisk skala och förklarar atomers och molekylers sammansättning. Å andra sidan förklarar Einsteins allmänna relativitetsteori gravitationen och stjärnors och galaxers rörelser och universums expansion.

Dessvärre är dessa teorier inte direkt kompatibla med varandra, och fysiker har länge försökt hitta en förenande teori för alla fyra krafter. Strängteori ger ett sådant förenande ramverk för gravitationen och kvantfältteorier, såsom standardmodellen, på bekostnad av att introducera sex ytterligare rumsdimensioner. För att passa vara experimentella observationer, t.ex. att vi i verkligheten enbart upplever fyra rumtidsdimensioner, måste de övriga sex våra ihoprullade i ett mycket litet kompakt rum. Storleken på detta rum måste vara mindre än vad våra bästa experiment, t.ex. Large Hadron Collider-experimentet vid CERN, kan observera.

Det finns en uppsjö av möjliga beskrivningar av detta kompakta rum, fler än det finns atomer i universum. Genom att kräva matematisk följdriktighet och introducera fysikaliska villkor som motsvarar våra experimentella observationer kan antalet tillåtna konfigurationer skäras ned med många storleksordningar. Dock är det inte enbart beräkningsmässigt dyrt att införa dessa villkor, det måste också göras på ett smart sätt eftersom det är beräkningsmässigt omöjligt att gå igenom alla tillåtna konfigurationer.

Nyligen har fysiker börjat använda maskininlärning för att ta sig an båda dessa problem. Till exempel används algoritmer från förstärkningsinlärning för att hitta smarta sätt att utforska möjliga strängkompaktifikationer. Jag har bidragit till denna forskning i artikel II genom att undersöka den heterotiska strängen kompaktifierad på en viss klass av mångfalder som kallas Calabi-Yau-rum. På så sätt kunde jag hitta flera intressanta modeller som passar vissa av våra experimentella observationer med avseende på exempelvis partikelinnehåll eller kraftförmedlare.

Än så länge kan inte alla experimentella villkor införas eftersom det matematiska maskineri som krävs inte är tillräckligt utvecklat eller eftersom beräkningsmässigt ofördelaktiga skalningslagar innebär att nya superdatorer är nödvändiga. I artiklarna I och IV använde jag övervakad maskininlärning för att hitta numeriska och snabbare analytiska lösnngar som ersätter de dyra kohomologiska beräkningarna som krävs i strängteoretiska kompaktifikationer. Hittills har det varit omöjligt att hitta strängvakua som ger korrekta Yukawakopplingar och massor för standardmodellens partiklar eftersom det kräver att vi

känner till det kompakta rummets metriska tensor som inte är analytiskt känd. I artikel V designade jag ett paket som uppskattar denna metriska tensor med djupa neurala nätverk och som i framtiden förhoppningsvis gör det möjligt att beräkna partikelmassor numeriskt i heterotiska strängkompaktifikationer.

På det hela taget bidrar denna avhandling och min forskning främst till den beräkningsmässiga sidan av strängteori. Totalt har jag designat fyra paket med öppen källkod som förbättrar och etablerar nya algoritmer för att hantera de ofta svåra beräkningar som uppstår i strängteoretiska kompaktifikationer.

10. Acknowledgements

First, I would like to thank my supervisor Magdalena for taking me into her research group and introducing me to the world of heterotic string compactifications. I'm especially grateful for giving me the chance to explore my own little corner in a, at the beginning, rather speculative research direction. I am thankful for the opportunities to go abroad, for the many shared research projects, and the overall freedom during my studies. Finally, I really appreciate the support when it comes to navigating the bureaucratic sides of academia. I would also like to thank Guido for being my co-supervisor during master and PhD studies an Yuji who supervised my first attempts at theoretical physics.

Over the course of my PhD studies I had the chance to collaborate with many people across the world. I'd like to thank Lara and James for hosting Matt and me in the US, many interesting discussions about heterotic string theory, and a successful collaboration even through initial set-backs. I'd like to thank Riccardo, Mohamed and Harold for our work on applying computer vision to Calabi-Yau manifolds. I'm especially grateful to Harold for not only bringing this collaboration together, but also being a great host during my visit to MIT and sharing his excitement about physics insights to better understand deep learning. I'm thankful for the collaboration with Andre and Fabian, in particular to Fabians commitment in shipping this project in time for the NeurIPS deadline. Finally, I'd like to thank Davide for the great work together and taking on such an ambitious master thesis project.

There are many people at Uppsala university who have made my life better. Thank you, Gregor and Konstantina, for the board game evenings. Thanks to Lorenzo, for showing me the Gran Paradiso, Guilia, for all the insights into proper Italian lifestyle, and my PhD twin Matt, for the trips to Corfu, Virginia, and even Stockholm, but all three more generally for coffee, food, drinks, company, Palermo, and the shared PhD struggles. My computer support team consisted of Ansgar (Python), Gregor (everything that is not Python), and Olga (for random ML questions). Also thanks to Paul for many conversations about the life and gossip of a string theorist. I'd like to further thank Alessandro, Alex, Alex, Suvendu, Lucia, Paolo, Oliver, Rebecca, Thales, Joe, Simon, Roman, Giuseppe, Luca, Andrea, and all the others for providing a great atmosphere at work, (secret) meetings, and seminars.

Outside of work I'm happy to have met Joel and Alfred at Gotlands nation. I'm thankful for the regular game nights with Viktor, Tom, Eva, Raz, and most importantly Ansgar, who owns *all* the boardgames (and for visiting me in Japan). I'm grateful to have met Tom who is not only the most loyal

roommate but also taught me so much about the UK, its holidays, customs, food, and language*s*. Viktor has been another great roommate with whom we shared many exceptional evenings before he left us for Olga. I'm thankful for Anna for merging her household into our apartment, Calle for taking over when Anna left, and also Paul for when Calle left. I'm grateful for the Pazza physics dinners with further members, Melissa, who made me switch from material science to theoretical physics, Hans, who has the greatest gifts at every party, and Jorge, for the shared experiences from masters to Riga to Gredos to doctors and beyond.

Also thanks to all the people back home in Düsseldorf who always welcome me back with open arms on our annual Christmas meetings, Martin, Marten, Arne, Domi, Caro, Chris, Alex, Philipp, Theo, Stefan and Roman. I want to tank the Tuesday DnD group, consisting of Arne, Christian, Colin and Martin for many adventures. Finally I'm especially thankful to Sandra for listening to all my complaints, to Moritz for being the best carry, and to Martin for countless nights of gaming and discussions about gods and worlds.

This work would have not been possible without the support from family. They have always believed in me: my parents, who supported my every decision, even if it means moving to the other side of the world; my siblings for holding the position back home; my grandparents for their relentless support during my education even though string theory has no clear job perspectives. Last but not least Nguyên for all the support during the last two years and keeping me sane in the pandemic.

Throughout the development of this thesis and during the work of my PhD I made extensive use of the following python packages: `pandas` [221, 222] and `numpy` [223] for data operations, `matplotlib` [224] and `seaborn` [225] for visualisation, `Scikit-learn` [226] for ML prototyping, and `tensorflow` [163] for the deep learning algorithms, `sagemath` [227] and `topcom` [228] for toric geometry. For symbolic manipulations of equations I have used `sympy` [229] and `singular` [230]. Computations presented in this work were in part enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at the HPC clusters *Tetralith* and *Rackham*, partially funded by the Swedish Research Council through grant agreement no. 2018-05973 with project numbers 2019/8-105, 2019/3-546, 2020/15-133 and 2020/5-662.

Bibliography

[1] R. H. Parker, C. Yu, W. Zhong, B. Estey, and H. Müller. "Measurement of the fine-structure constant as a test of the Standard Model". In: *Science* 360.6385 (2018), pp. 191–195. DOI: 10.1126/science.aap7706. URL: https://doi.org/10.1126%2Fscience.aap7706.

[2] D. J. Gross, J. A. Harvey, E. Martinec, and R. Ryan. "Heterotic string theory (I). The free heterotic string". In: *Current Physics–Sources and Comments*. Vol. 4. Elsevier, 1989, pp. 76–107.

[3] D. J. Gross, J. A. Harvey, E. Martinec, and R. Rohm. "Heterotic string theory: (II). The interacting heterotic string". In: *Nuclear Physics B* 267.1 (1986), pp. 75–124. ISSN: 0550-3213. DOI: https://doi.org/10.1016/0550-3213(86)90146-X. URL: https://www.sciencedirect.com/science/article/pii/055032138690146X.

[4] M. B. Green, J. H. Schwarz, and E. Witten. *Superstring theory: volume 2, loop amplitudes, anomalies and phenomenology*. Cambridge university press, 2012.

[5] A. Strominger. "Superstrings with Torsion". In: *Nucl. Phys.* B274 (1986), p. 253. DOI: 10.1016/0550-3213(86)90286-5.

[6] C. Hull. "Compactifications of the heterotic superstring". In: *Physics Letters B* 178.4 (1986), pp. 357–364. ISSN: 0370-2693. DOI: https://doi.org/10.1016/0370-2693(86)91393-6. URL: https://www.sciencedirect.com/science/article/pii/0370269386913936.

[7] L. B. Anderson and M. Karkheiran. "TASI Lectures on Geometric Tools for String Compactifications". In: *PoS* TASI2017 (2018), p. 013. DOI: 10.22323/1.305.0013. arXiv: 1804.08792 [hep-th].

[8] S. Kachru, R. Kallosh, A. D. Linde, and S. P. Trivedi. "De Sitter vacua in string theory". In: *Phys. Rev. D* 68 (2003), p. 046005. DOI: 10.1103/PhysRevD.68.046005. arXiv: hep-th/0301240.

[9] K. Hori et al. *Mirror symmetry*. Vol. 1. American Mathematical Soc., 2003.

[10] G. Carleo et al. "Machine learning and the physical sciences". In: *Rev. Mod. Phys.* 91.4 (2019), p. 045002. DOI: 10.1103/RevModPhys.91.045002. arXiv: 1903.10563 [physics.comp-ph].

[11] F. Ruehle. "Data science applications to string theory". In: *Phys. Rept.* 839 (2020), pp. 1–117. DOI: 10.1016/j.physrep.2019.09.005.

[12] F. Denef and M. R. Douglas. "Computational complexity of the landscape. I." In: *Annals Phys.* 322 (2007), pp. 1096–1142. DOI: 10.1016/j.aop.2006.07.013. arXiv: hep-th/0602072.

[13] J. Halverson and F. Ruehle. "Computational Complexity of Vacua and Near-Vacua in Field and String Theory". In: *Phys. Rev. D* 99.4 (2019), p. 046015. DOI: 10.1103/PhysRevD.99.046015. arXiv: 1809.08279 [hep-th].

[14] M. Kreuzer and H. Skarke. "Complete classification of reflexive polyhedra in four-dimensions". In: *Adv. Theor. Math. Phys.* 4 (2002), pp. 1209–1230. DOI: 10.4310/ATMP.2000.v4.n6.a2. arXiv: hep-th/0002240.

[15] M. Demirtas, L. McAllister, and A. Rios-Tascon. "Bounding the Kreuzer-Skarke Landscape". In: *Fortsch. Phys.* 68 (2020), p. 2000086. DOI: 10.1002/prop.202000086. arXiv: 2008.01730 [hep-th].

[16] W. Taylor and Y.-N. Wang. "The F-theory geometry with most flux vacua". In: *JHEP* 12 (2015), p. 164. DOI: 10.1007/JHEP12(2015)164. arXiv: 1511.03209 [hep-th].

[17] O. Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[19] D. P. Kingma and M. Welling. *Auto-Encoding Variational Bayes*. 2014. arXiv: 1312.6114 [stat.ML].

[20] I. Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014). arXiv: 1406.2661 [stat.ML].

[21] T. B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[22] D. Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587 (2016), pp. 484–489.

[23] Y.-H. He. *The Calabi–Yau Landscape: From Geometry, to Physics, to Machine Learning*. Lecture Notes in Mathematics. May 2021. ISBN: 978-3-030-77561-2, 978-3-030-77562-9. DOI: 10.1007/978-3-030-77562-9. arXiv: 1812.02893 [hep-th].

[24] Y.-H. He. "Deep-Learning the Landscape". In: (June 2017). arXiv: 1706.02714 [hep-th].

[25] K. Bull, Y.-H. He, V. Jejjala, and C. Mishra. "Machine Learning CICY Threefolds". In: *Phys. Lett. B* 785 (2018), pp. 65–72. DOI: 10.1016/j.physletb.2018.08.008. arXiv: 1806.03121 [hep-th].

[26] K. Bull, Y.-H. He, V. Jejjala, and C. Mishra. "Getting CICY High". In: *Phys. Lett. B* 795 (2019), pp. 700–706. DOI: 10.1016/j.physletb.2019.06.067. arXiv: 1903.03113 [hep-th].

[27] H. Erbin and R. Finotello. "Inception neural network for complete intersection Calabi–Yau 3-folds". In: *Mach. Learn. Sci. Tech.* 2.2 (2021), 02LT03. DOI: 10.1088/2632-2153/abda61. arXiv: 2007.13379 [hep-th].

[28] H. Erbin and R. Finotello. "Machine learning for complete intersection Calabi-Yau manifolds: a methodological study". In: *Phys. Rev. D* 103.12 (2021), p. 126014. DOI: 10.1103/PhysRevD.103.126014. arXiv: 2007.15706 [hep-th].

[29] Y.-H. He and A. Lukas. "Machine Learning Calabi-Yau Four-folds". In: (Sept. 2020). arXiv: 2009.02544 [hep-th].

[30] Y.-H. He and S.-J. Lee. "Distinguishing elliptic fibrations with AI". In: *Phys. Lett. B* 798 (2019), p. 134889. DOI: 10.1016/j.physletb.2019.134889. arXiv: 1904.08530 [hep-th].

[31] P. Berglund, B. Campbell, and V. Jejjala. "Machine Learning Kreuzer–Skarke Calabi–Yau Threefolds". In: (Dec. 2021). arXiv: 2112.09117 [hep-th].

[32] D. S. Berman, Y.-H. He, and E. Hirst. "Machine Learning Calabi-Yau Hypersurfaces". In: (Dec. 2021). arXiv: 2112.06350 [hep-th].

[33] J. Bao, Y.-H. He, E. Hirst, J. Hofscheier, A. Kasprzyk, and S. Majumder. "Polytopes and Machine Learning". In: (Sept. 2021). arXiv: 2109.09602 [math.CO].

[34] F. Ruehle. "Evolving neural networks with genetic algorithms to study the String Landscape". In: *JHEP* 08 (2017), p. 038. DOI: 10.1007/JHEP08(2017)038. arXiv: 1706.07024 [hep-th].

[35] C. R. Brodie, A. Constantin, R. Deen, and A. Lukas. "Machine Learning Line Bundle Cohomology". In: *Fortsch. Phys.* 68.1 (2020), p. 1900087. DOI: 10.1002/prop.201900087. arXiv: 1906.08730 [hep-th].

[36] D. Klaewer and L. Schlechter. "Machine Learning Line Bundle Cohomologies of Hypersurfaces in Toric Varieties". In: *Phys. Lett. B* 789 (2019), pp. 438–443. DOI: 10.1016/j.physletb.2019.01.002. arXiv: 1809.02547 [hep-th].

[37] M. Bies, M. Cvetič, R. Donagi, L. Lin, M. Liu, and F. Ruehle. "Machine Learning and Algebraic Approaches towards Complete Matter Spectra in 4d F-theory". In: (June 2020). arXiv: 2007.00009 [hep-th].

[38] D. Krefl and R.-K. Seong. "Machine Learning of Calabi-Yau Volumes". In: *Phys. Rev. D* 96.6 (2017), p. 066014. DOI: 10.1103/PhysRevD.96.066014. arXiv: 1706.03346 [hep-th].

[39] A. Ashmore, Y.-H. He, and B. A. Ovrut. "Machine Learning Calabi–Yau Metrics". In: *Fortsch. Phys.* 68.9 (2020), p. 2000068. DOI: 10.1002/prop.202000068. arXiv: 1910.08605 [hep-th].

[40] M. R. Douglas, S. Lakshminarasimhan, and Y. Qi. "Numerical Calabi-Yau metrics from holomorphic networks". In: (Dec. 2020). arXiv: 2012.04797 [hep-th].

[41] V. Jejjala, D. K. Mayorga Pena, and C. Mishra. "Neural Network Approximations for Calabi-Yau Metrics". In: (Dec. 2020). arXiv: 2012.15821 [hep-th].

[42] L. B. Anderson, M. Gerdes, J. Gray, S. Krippendorf, N. Raghuram, and F. Ruehle. "Moduli-dependent Calabi-Yau and SU(3)-structure metrics from Machine Learning". In: *JHEP* 05 (2021), p. 013. DOI: 10.1007/JHEP05(2021)013. arXiv: 2012.04656 [hep-th].

[43] A. Ashmore and F. Ruehle. "Moduli-dependent KK towers and the swampland distance conjecture on the quintic Calabi-Yau manifold". In: *Phys. Rev. D* 103.10 (2021), p. 106028. DOI: 10.1103/PhysRevD.103.106028. arXiv: 2103.07472 [hep-th].

[44] A. Ashmore, R. Deen, Y.-H. He, and B. A. Ovrut. "Machine Learning Line Bundle Connections". In: (Oct. 2021). arXiv: 2110.12483 [hep-th].

[45] J. Carifio, J. Halverson, D. Krioukov, and B. D. Nelson. "Machine Learning in the String Landscape". In: *JHEP* 09 (2017), p. 157. DOI: 10.1007/JHEP09(2017)157. arXiv: 1707.00655 [hep-th].

[46] V. Jejjala, A. Kar, and O. Parrikar. "Deep Learning the Hyperbolic Volume of a Knot". In: *Phys. Lett. B* 799 (2019), p. 135033. DOI: 10.1016/j.physletb.2019.135033. arXiv: 1902.05547 [hep-th].

[47] J. Craven, V. Jejjala, and A. Kar. "Disentangling a Deep Learned Volume Formula". In: (Dec. 2020). DOI: 10.1007/JHEP06(2021)040. arXiv: 2012.03955 [hep-th].

[48] Y.-H. He. "Machine-Learning Mathematical Structures". In: (Jan. 2021). arXiv: 2101.06317 [cs.LG].

[49] H.-Y. Chen, Y.-H. He, S. Lal, and S. Majumder. "Machine learning Lie structures & applications to physics". In: *Phys. Lett. B* 817 (2021), p. 136297. DOI: 10.1016/j.physletb.2021.136297. arXiv: 2011.00871 [hep-th].

[50] Y.-H. He, K.-H. Lee, and T. Oliver. "Machine-Learning the Sato–Tate Conjecture". In: (Oct. 2020). arXiv: 2010.01213 [math.NT].

[51] Y.-H. He, K.-H. Lee, and T. Oliver. "Machine-Learning Number Fields". In: (Nov. 2020). arXiv: 2011.08958 [math.NT].

[52] Y.-H. He, K.-H. Lee, and T. Oliver. "Machine-Learning Arithmetic Curves". In: (Dec. 2020). arXiv: 2012.04084 [math.NT].

[53] Y.-H. He and S.-T. Yau. "Graph Laplacians, Riemannian Manifolds and their Machine-Learning". In: (June 2020). arXiv: 2006.16619 [math.CO].

[54] A. Mütter, E. Parr, and P. K. S. Vaudrevange. "Deep learning in the heterotic orbifold landscape". In: *Nucl. Phys.* B940 (2019), pp. 113–129. DOI: 10.1016/j.nuclphysb.2019.01.013. arXiv: 1811.05993 [hep-th].

[55] R. Deen, Y.-H. He, S.-J. Lee, and A. Lukas. "Machine Learning String Standard Models". In: (Mar. 2020). arXiv: 2003.13339 [hep-th].

[56] H. Otsuka and K. Takemoto. "Deep learning and k-means clustering in heterotic string vacua with line bundles". In: *JHEP* 05 (2020), p. 047. DOI: 10.1007/JHEP05(2020)047. arXiv: 2003.11880 [hep-th].

[57] A. Cole and G. Shiu. "Topological Data Analysis for the String Landscape". In: *JHEP* 03 (2019), p. 054. DOI: 10.1007/JHEP03(2019)054. arXiv: 1812.06960 [hep-th].

[58] J. Halverson, B. Nelson, and F. Ruehle. "Branes with Brains: Exploring String Vacua with Deep Reinforcement Learning". In: *JHEP* 06 (2019), p. 003. DOI: 10.1007/JHEP06(2019)003. arXiv: 1903.11616 [hep-th].

[59] T. R. Harvey and A. Lukas. "Quark Mass Models and Reinforcement Learning". In: *JHEP* 08 (2021), p. 161. DOI: 10.1007/JHEP08(2021)161. arXiv: 2103.04759 [hep-th].

[60] A. Constantin, T. R. Harvey, and A. Lukas. "Heterotic String Model Building with Monad Bundles and Reinforcement Learning". In: (Aug. 2021). arXiv: 2108.07316 [hep-th].

[61] S. Krippendorf, R. Kroepsch, and M. Syvaeri. "Revealing systematics in phenomenologically viable flux vacua with reinforcement learning". In: (July 2021). arXiv: 2107.04039 [hep-th].

[62] S. Gukov, J. Halverson, F. Ruehle, and P. Sułkowski. "Learning to Unknot". In: *Mach. Learn. Sci. Tech.* 2.2 (2021), p. 025035. DOI: 10.1088/2632-2153/abe91f. arXiv: 2010.16263 [math.GT].

[63] G. Kántor, V. Niarchos, and C. Papageorgakis. "Conformal Bootstrap with Reinforcement Learning". In: (Aug. 2021). arXiv: 2108.09330 [hep-th].

[64] G. Kántor, V. Niarchos, and C. Papageorgakis. "Solving Conformal Field Theories with Artificial Intelligence". In: (Aug. 2021). arXiv: 2108.08859 [hep-th].

[65] S. Abel and J. Rizos. "Genetic Algorithms and the Search for Viable String Vacua". In: *JHEP* 08 (2014), p. 010. DOI: 10.1007/JHEP08(2014)010. arXiv: 1404.7359 [hep-th].

[66] A. Cole, A. Schachner, and G. Shiu. "Searching the Landscape of Flux Vacua with Genetic Algorithms". In: *JHEP* 11 (2019), p. 045. DOI: 10.1007/JHEP11(2019)045. arXiv: 1907.10072 [hep-th].

[67] S. Abel, A. Constantin, T. R. Harvey, and A. Lukas. "Evolving Heterotic Gauge Backgrounds: Genetic Algorithms versus Reinforcement Learning". In: (Oct. 2021). arXiv: 2110.14029 [hep-th].

[68] A. Cole, S. Krippendorf, A. Schachner, and G. Shiu. "Probing the Structure of String Theory Vacua with Genetic Algorithms and Reinforcement Learning". In: *35th Conference on Neural Information Processing Systems*. Nov. 2021. arXiv: 2111.11466 [hep-th].

[69] I. Bena, J. Blåbäck, M. Graña, and S. Lüst. "The Tadpole Problem". In: (Oct. 2020). arXiv: 2010.10519 [hep-th].

[70] I. Bena, J. Blåbäck, M. Graña, and S. Lüst. "Algorithmically solving the Tadpole Problem". In: (Mar. 2021). arXiv: 2103.03250 [hep-th].

[71] E. Dyer and G. Gur-Ari. "Asymptotics of Wide Networks from Feynman Diagrams". In: (Sept. 2019). arXiv: 1909.11304 [cs.LG].

[72] S. Yaida. "Non-Gaussian processes and neural networks at finite widths". In: (Sept. 2019). arXiv: 1910.00019 [stat.ML].

[73] J. Halverson, A. Maiti, and K. Stoner. "Neural Networks and Quantum Field Theory". In: *Mach. Learn. Sci. Tech.* 2.3 (2021), p. 035002. DOI: 10.1088/2632-2153/abeca3. arXiv: 2008.08601 [cs.LG].

[74] H. Erbin, V. Lahoche, and D. O. Samary. “Nonperturbative renormalization for the neural network-QFT correspondence”. In: (Aug. 2021). arXiv: 2108.01403 [hep-th].

[75] D. A. Roberts, S. Yaida, and B. Hanin. “The Principles of Deep Learning Theory”. In: (June 2021). arXiv: 2106.10165 [cs.LG].

[76] G. Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[77] D. P. Kingma and M. Welling. “An Introduction to Variational Autoencoders”. In: *Foundations and Trends® in Machine Learning* 12.4 (2019), 307–392. ISSN: 1935-8245. DOI: 10.1561/2200000056. URL: http://dx.doi.org/10.1561/2200000056.

[78] V. Mnih et al. “Asynchronous methods for deep reinforcement learning”. In: *International conference on machine learning*. PMLR. 2016, pp. 1928–1937. arXiv: 1602.01783 [cs.LG].

[79] R. Bommasani et al. *On the Opportunities and Risks of Foundation Models*. 2021. arXiv: 2108.07258 [cs.LG].

[80] A. Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008. arXiv: 1706.03762 [cs.CL]. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[81] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, Jan. 2006. URL: https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/.

[82] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[83] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola. *Dive into Deep Learning*. 2021. arXiv: 2106.11342 [cs.LG].

[84] P. Mehta, M. Bukov, C.-H. Wang, A. G. Day, C. Richardson, C. K. Fisher, and D. J. Schwab. “A high-bias, low-variance introduction to Machine Learning for physicists”. In: *Physics Reports* 810 (May 2019), pp. 1–124. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2019.03.001. URL: http://dx.doi.org/10.1016/j.physrep.2019.03.001.

[85] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio. *Fantastic Generalization Measures and Where to Find Them*. 2019. arXiv: 1912.02178 [cs.LG].

[86] T. Tieleman, G. Hinton, et al. “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.

[87] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[88] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[89] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778. arXiv: 1512.03385 [cs.CV].

[90] D. Bahdanau, K. Cho, and Y. Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL].

[91] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[92] A. Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[93] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. URL: http://incompleteideas.net/book/the-book-2nd.html.

[94] D. Silver et al. “A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play”. In: *Science* 362.6419 (2018), pp. 1140–1144. ISSN: 0036-8075. DOI: 10.1126/science.aar6404. URL: https://science.sciencemag.org/content/362/6419/1140.

[95] D. Silver. *Lectures on Reinforcement Learning*. URL: https://www.davidsilver.uk/teaching/. 2015.

[96] OpenAI et al. *Dota 2 with Large Scale Deep Reinforcement Learning*. 2019. arXiv: 1912.06680 [cs.LG].

[97] OpenAI. *OpenAI Five*. https://blog.openai.com/openai-five/. 2018.

[98] O. Vinyals et al. “Grandmaster level in StarCraft II using multi-agent reinforcement learning”. In: *Nature* 575.7782 (2019), pp. 350–354.

[99] C. J. C. H. Watkins. “Learning from delayed rewards”. PhD thesis. King’s College, Cambridge United Kingdom, 1989.

[100] C. J. Watkins and P. Dayan. “Q-learning”. In: *Machine learning* 8.3-4 (1992), pp. 279–292.

[101] R. J. Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. In: *Machine learning* 8.3 (1992), pp. 229–256.

[102] E. Calabi. “On Kähler manifolds with vanishing canonical class, Algebraic geometry and topology. A symposium in honor of S. Lefschetz”. In: *Princeton Math. Ser., Princeton University Press, Princeton* 78 (1957), p. 89.

[103] S.-T. Yau. “Calabi’s conjecture and some new results in algebraic geometry”. In: *Proceedings of the National Academy of Sciences* 74.5 (1977), pp. 1798–1799.

[104] S. Kachru, A. Tripathy, and M. Zimet. “K3 metrics from little string theory”. In: (Oct. 2018). arXiv: 1810.10540 [hep-th].

[105] S. Kachru, A. Tripathy, and M. Zimet. “K3 metrics”. In: (June 2020). arXiv: 2006.02435 [hep-th].

[106] P. Candelas, A. M. Dale, C. A. Lutken, and R. Schimmrigk. “Complete Intersection Calabi-Yau Manifolds”. In: *Nucl. Phys.* B298 (1988), p. 493. DOI: 10.1016/0550-3213(88)90352-5.

[107] J. Gray, A. S. Haupt, and A. Lukas. "All Complete Intersection Calabi-Yau Four-Folds". In: *JHEP* 07 (2013), p. 070. DOI: 10.1007/JHEP07(2013)070. arXiv: 1303.1832 [hep-th].

[108] J. Gray, A. S. Haupt, and A. Lukas. "Topological Invariants and Fibration Structure of Complete Intersection Calabi-Yau Four-Folds". In: *JHEP* 09 (2014), p. 093. DOI: 10.1007/JHEP09(2014)093. arXiv: 1405.2073 [hep-th].

[109] L. B. Anderson, F. Apruzzi, X. Gao, J. Gray, and S.-J. Lee. "A new construction of Calabi–Yau manifolds: Generalized CICYs". In: *Nucl. Phys.* B906 (2016), pp. 441–496. DOI: 10.1016/j.nuclphysb.2016.03.016. arXiv: 1507.03235 [hep-th].

[110] V. V. Batyrev and L. A. Borisov. "On Calabi-Yau complete intersections in toric varieties". In: (Dec. 1994). arXiv: alg-geom/9412017.

[111] V. V. Batyrev and L. A. Borisov. "Dual cones and mirror symmetry for generalized Calabi-Yau manifolds". In: *arXiv preprint alg-geom/9402002* (1994).

[112] F. Schöller and H. Skarke. "All Weight Systems for Calabi–Yau Fourfolds from Reflexive Polyhedra". In: *Commun. Math. Phys.* 372.2 (2019), pp. 657–678. DOI: 10.1007/s00220-019-03331-9. arXiv: 1808.02422 [hep-th].

[113] P. Berglund and T. Hübsch. "On Calabi–Yau generalized complete intersections from Hirzebruch varieties and novel $K3$-fibrations". In: *Adv. Theor. Math. Phys.* 22 (2018), pp. 261–303. DOI: 10.4310/ATMP.2018.v22.n2.a1. arXiv: 1606.07420 [hep-th].

[114] D. R. Morrison and C. Vafa. "Compactifications of F theory on Calabi-Yau threefolds. 1". In: *Nucl. Phys. B* 473 (1996), pp. 74–92. DOI: 10.1016/0550-3213(96)00242-8. arXiv: hep-th/9602114.

[115] D. R. Morrison and C. Vafa. "Compactifications of F theory on Calabi-Yau threefolds. 2." In: *Nucl. Phys. B* 476 (1996), pp. 437–469. DOI: 10.1016/0550-3213(96)00369-0. arXiv: hep-th/9603161.

[116] D. R. Morrison and W. Taylor. "Toric bases for 6D F-theory models". In: *Fortsch. Phys.* 60 (2012), pp. 1187–1216. DOI: 10.1002/prop.201200086. arXiv: 1204.0283 [hep-th].

[117] D. R. Morrison and W. Taylor. "Classifying bases for 6D F-theory models". In: *Central Eur. J. Phys.* 10 (2012), pp. 1072–1088. DOI: 10.2478/s11534-012-0065-4. arXiv: 1201.1943 [hep-th].

[118] T. Hubsch. *Calabi-Yau manifolds: A Bestiary for physicists*. Singapore: World Scientific, 1994. ISBN: 9789810219277, 981021927X.

[119] D. D. Joyce. *Compact manifolds with special holonomy*. Oxford University Press on Demand, 2000.

[120] P. Candelas. "Lectures on complex manifolds". In: *Superstrings and grand unification*. 1988.

[121] M. Nakahara. *Geometry, topology and physics*. CRC press, 2003.

[122] R. Hartshorne. *Algebraic geometry*. Vol. 52. Springer Science & Business Media, 2013.

[123] P. Griffiths and J. Harris. *Principles of algebraic geometry*. John Wiley & Sons, 2014.

[124] L. B. Anderson. "Heterotic and M-theory Compactifications for String Phenomenology". PhD thesis. Oxford University, Aug. 2008. arXiv: 0808.3621 [hep-th].

[125] K. Uhlenbeck and S.-T. Yau. "On the existence of hermitian-yang-mills connections in stable vector bundles". In: *Communications on Pure and Applied Mathematics* 39.S1 (1986), S257–S293.

[126] S. Kobayashi. *Differential geometry of complex vector bundles*. Princeton University Press, 2014.

[127] L. B. Anderson, J. Gray, A. Lukas, and E. Palti. "Heterotic Line Bundle Standard Models". In: *JHEP* 06 (2012), p. 113. DOI: 10.1007/JHEP06(2012)113. arXiv: 1202.1757 [hep-th].

[128] C. T. C. Wall. "Classification problems in differential topology. V". In: *Inventiones mathematicae* 1.4 (1966), pp. 355–374.

[129] M. Kreuzer, E. Riegler, and D. A. Sahakyan. "Toric complete intersections and weighted projective space". In: *J. Geom. Phys.* 46 (2003), pp. 159–173. DOI: 10.1016/S0393-0440(02)00124-9. arXiv: math/0103214.

[130] A. Constantin and A. Lukas. "Formulae for Line Bundle Cohomology on Calabi-Yau Threefolds". In: *Fortsch. Phys.* 67.12 (2019), p. 1900084. DOI: 10.1002/prop.201900084. arXiv: 1808.09992 [hep-th].

[131] P. S. Green, T. Hubsch, and C. A. Lutken. "All Hodge Numbers of All Complete Intersection Calabi-Yau Manifolds". In: *Class. Quant. Grav.* 6 (1989), pp. 105–124. DOI: 10.1088/0264-9381/6/2/006.

[132] C. Szegedy et al. "Going Deeper with Convolutions". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594. arXiv: 1409.4842 [cs.CV].

[133] A. Benzine, M. El Amine Seddik, and J. Desmarais. "Deep Miner: A Deep and Multi-branch Network which Mines Rich and Diverse Features for Person Re-identification". In: *arXiv e-prints* (Feb. 2021). arXiv: 2102.09321 [cs.CV].

[134] Y.-H. He, S. Lal, and M. Z. Zaz. "The World in a Grain of Sand: Condensing the String Vacuum Degeneracy". In: (Nov. 2021). arXiv: 2111.04761 [hep-th].

[135] S. Ioffe and C. Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. Vol. abs/1502.03167. PMLR. 2015, pp. 448–456. arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

[136] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the Inception Architecture for Computer Vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2818–2826. DOI: 10.1109/CVPR.2016.308. arXiv: 1512.00567.

[137] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning”. In: *31st AAAI Conference on Artificial Intelligence*. AAAI’17. AAAI Press, 2017, pp. 4278–4284. arXiv: 1602.07261.

[138] S. Santurkar, D. Tsipras, A. Ilyas, and A. Mądry. “How does batch normalization help optimization?” In: *Proceedings of the 32nd international conference on neural information processing systems*. 2018, pp. 2488–2498. arXiv: 1805.11604 [stat.ML].

[139] S. Falkner, A. Klein, and F. Hutter. “BOHB: Robust and Efficient Hyperparameter Optimization at Scale”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, July 2018, pp. 1437–1446. arXiv: 1807.01774 [cs.LG]. URL: http://proceedings.mlr.press/v80/falkner18a.html.

[140] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. “Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization”. In: *Journal of Machine Learning Research* 18.185 (2018), pp. 1–52. arXiv: 1603.06560 [cs.LG]. URL: http://jmlr.org/papers/v18/16-558.html.

[141] P. J. Huber. “Robust Estimation of a Location Parameter”. In: *The Annals of Mathematical Statistics* 35.1 (1964), pp. 73 –101. DOI: 10.1214/aoms/1177703732. URL: https://doi.org/10.1214/aoms/1177703732.

[142] E. I. Buchbinder, A. Constantin, and A. Lukas. “The Moduli Space of Heterotic Line Bundle Models: a Case Study for the Tetra-Quadric”. In: *JHEP* 03 (2014), p. 025. DOI: 10.1007/JHEP03(2014)025. arXiv: 1311.1941 [hep-th].

[143] M. Larfors and R. Schneider. *pyCICY - A python CICY toolkit*. June 2019. DOI: 10.5281/zenodo.3243914. URL: https://github.com/robin-schneider/CICY.

[144] ***cohomCalg*** *package*. Download link. High-performance line bundle cohomology computation based on [146]. 2010. URL: https://github.com/BenjaminJurke/cohomCalg.

[145] C. R. Brodie, A. Constantin, R. Deen, and A. Lukas. “Index Formulae for Line Bundle Cohomology on Complex Surfaces”. In: *Fortschritte der Physik* 68.2 (2020), p. 1900086. DOI: 10.1002/prop.201900086. arXiv: 1906.08769.

[146] T. Rahn and H. Roschy. “Cohomology of Line Bundles: Proof of the Algorithm”. In: *J. Math. Phys.* 51 (2010), p. 103520. DOI: 10.1063/1.3523318. arXiv: 1006.2392 [hep-th].

[147] A. Strominger. “Yukawa Couplings in Superstring Compactification”. In: *Phys. Rev. Lett.* 55 (1985), p. 2547. DOI: 10.1103/PhysRevLett.55.2547.

[148] V. Braun, Y.-H. He, and B. A. Ovrut. “Yukawa couplings in heterotic standard models”. In: *JHEP* 04 (2006), p. 019. DOI: 10.1088/1126-6708/2006/04/019. arXiv: hep-th/0601204.

[149] S. Blesneag, E. I. Buchbinder, P. Candelas, and A. Lukas. "Holomorphic Yukawa Couplings in Heterotic String Theory". In: *JHEP* 01 (2016), p. 152. DOI: 10.1007/JHEP01(2016)152. arXiv: 1512.05322 [hep-th].

[150] S. Blesneag, E. I. Buchbinder, and A. Lukas. "Holomorphic Yukawa Couplings for Complete Intersection Calabi-Yau Manifolds". In: *JHEP* 01 (2017), p. 119. DOI: 10.1007/JHEP01(2017)119. arXiv: 1607.03461 [hep-th].

[151] L. B. Anderson, J. Gray, D. Grayson, Y.-H. He, and A. Lukas. "Yukawa Couplings in Heterotic Compactification". In: *Commun. Math. Phys.* 297 (2010), pp. 95–127. DOI: 10.1007/s00220-010-1033-8. arXiv: 0904.2186 [hep-th].

[152] M. Karkheiran. "Yukawa textures from singular spectral data". In: *JHEP* 11 (2021), p. 131. DOI: 10.1007/JHEP11(2021)131. arXiv: 2107.07830 [hep-th].

[153] S. K. Donaldson. "Some numerical results in complex differential geometry". In: *arXiv Mathematics e-prints*, math/0512625 (Dec. 2005), math/0512625. arXiv: math/0512625 [math.DG].

[154] M. R. Douglas, R. L. Karp, S. Lukic, and R. Reinbacher. "Numerical Calabi-Yau metrics". In: *J. Math. Phys.* 49 (2008), p. 032302. DOI: 10.1063/1.2888403. arXiv: hep-th/0612075.

[155] V. Braun, T. Brelidze, M. R. Douglas, and B. A. Ovrut. "Calabi-Yau Metrics for Quotients and Complete Intersections". In: *JHEP* 05 (2008), p. 080. DOI: 10.1088/1126-6708/2008/05/080. arXiv: 0712.3563 [hep-th].

[156] V. Braun, T. Brelidze, M. R. Douglas, and B. A. Ovrut. "Eigenvalues and Eigenfunctions of the Scalar Laplace Operator on Calabi-Yau Manifolds". In: *JHEP* 07 (2008), p. 120. DOI: 10.1088/1126-6708/2008/07/120. arXiv: 0805.3689 [hep-th].

[157] M. R. Douglas, R. L. Karp, S. Lukic, and R. Reinbacher. "Numerical solution to the hermitian Yang-Mills equation on the Fermat quintic". In: *JHEP* 12 (2007), p. 083. DOI: 10.1088/1126-6708/2007/12/083. arXiv: hep-th/0606261.

[158] L. B. Anderson, V. Braun, R. L. Karp, and B. A. Ovrut. "Numerical Hermitian Yang-Mills Connections and Vector Bundle Stability in Heterotic Theories". In: *JHEP* 06 (2010), p. 107. DOI: 10.1007/JHEP06(2010)107. arXiv: 1004.4399 [hep-th].

[159] L. B. Anderson, V. Braun, and B. A. Ovrut. "Numerical Hermitian Yang-Mills Connections and Kahler Cone Substructure". In: *JHEP* 01 (2012), p. 014. DOI: 10.1007/JHEP01(2012)014. arXiv: 1103.3041 [hep-th].

[160] M. Headrick and A. Nassar. "Energy functionals for Calabi-Yau metrics". In: *Adv. Theor. Math. Phys.* 17.5 (2013), pp. 867–902. DOI: 10.4310/ATMP.2013.v17.n5.a1. arXiv: 0908.2635 [hep-th].

[161] W. Cui and J. Gray. "Numerical Metrics, Curvature Expansions and Calabi-Yau Manifolds". In: *JHEP* 05 (2020), p. 044. DOI: 10.1007/JHEP05(2020)044. arXiv: 1912.11068 [hep-th].

[162] N. Afkhami-Jeddi, A. Ashmore, and C. Cordova. “Calabi-Yau CFTs and Random Matrices”. In: (July 2021). arXiv: 2107.11461 [hep-th].

[163] M. Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[164] W. K. Hastings. “Monte Carlo sampling methods using Markov chains and their applications”. In: *Biometrika* 57.1 (Apr. 1970), pp. 97–109. ISSN: 0006-3444. DOI: 10.1093/biomet/57.1.97. eprint: https://academic.oup.com/biomet/article-pdf/57/1/97/23940249/57-1-97.pdf. URL: https://doi.org/10.1093/biomet/57.1.97.

[165] M. D. Homan and A. Gelman. “The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo”. In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), 1593–1623. ISSN: 1532-4435.

[166] B. Shiffman and S. Zelditch. “Distribution of Zeros of Random and Quantum Chaotic Sections of Positive Line Bundles”. In: *Communications in Mathematical Physics* 200.3 (Jan. 1999), pp. 661–683. DOI: 10.1007/s002200050544. arXiv: math/9803052 [math.CV].

[167] M. Larfors, A. Lukas, F. Ruehle, and R. Schneider. *Half a billion Calabi-Yau metrics*. To appear in 2022.

[168] M. Raissi, P. Perdikaris, and G. E. Karniadakis. *Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations*. 2017. arXiv: 1711.10561 [cs.AI].

[169] M. Raissi, P. Perdikaris, and G. E. Karniadakis. *Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations*. 2017. arXiv: 1711.10566 [cs.AI].

[170] M. Raissi, P. Perdikaris, and G. E. Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (2019), pp. 686–707.

[171] D. Masters and C. Luschi. *Revisiting Small Batch Training for Deep Neural Networks*. 2018. arXiv: 1804.07612 [cs.LG].

[172] A. Dosovitskiy and J. Djolonga. “You Only Train Once: Loss-Conditional Training of Deep Networks”. In: *International Conference on Learning Representations*. 2019.

[173] J. Hestness et al. *Deep Learning Scaling is Predictable, Empirically*. 2017. arXiv: 1712.00409 [cs.LG].

[174] J. Kaplan et al. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG].

[175] J. S. Rosenfeld, A. Rosenfeld, Y. Belinkov, and N. Shavit. *A Constructive Prediction of the Generalization Error Across Scales*. 2019. arXiv: 1909.12673 [cs.LG].

[176] T. Henighan et al. *Scaling Laws for Autoregressive Generative Modeling*. 2020. arXiv: 2010.14701 [cs.LG].

[177] Y. Bahri, E. Dyer, J. Kaplan, J. Lee, and U. Sharma. *Explaining Neural Scaling Laws*. 2021. arXiv: 2102.06701 [cs.LG].

[178] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. 2021. DOI: 10.48550/ARXIV.2104.13478. URL: https://arxiv.org/abs/2104.13478.

[179] S.-M. Udrescu and M. Tegmark. "AI Feynman: A physics-inspired method for symbolic regression". In: *Science Advances* 6.16 (2020), eaay2631.

[180] P. Candelas, G. T. Horowitz, A. Strominger, and E. Witten. "Vacuum Configurations for Superstrings". In: *Nucl. Phys. B* 258 (1985), pp. 46–74. DOI: 10.1016/0550-3213(85)90602-9.

[181] B. R. Greene, K. H. Kirklin, P. J. Miron, and G. G. Ross. "A Superstring Inspired Standard Model". In: *Phys. Lett. B* 180 (1986). Ed. by S. C. Loken, p. 69. DOI: 10.1016/0370-2693(86)90137-1.

[182] B. R. Greene, K. H. Kirklin, P. J. Miron, and G. G. Ross. "A Three Generation Superstring Model. 1. Compactification and Discrete Symmetries". In: *Nucl. Phys. B* 278 (1986), pp. 667–693. DOI: 10.1016/0550-3213(86)90057-X.

[183] J. A. Casas and C. Munoz. "Three Generation SU(3) x SU(2) x U(1)-Y x U(1) Orbifold Models Through Fayet-Iliopoulos Terms". In: *Phys. Lett. B* 209 (1988), pp. 214–220. DOI: 10.1016/0370-2693(88)90935-5.

[184] V. Braun, Y.-H. He, B. A. Ovrut, and T. Pantev. "A Heterotic standard model". In: *Phys. Lett. B* 618 (2005), pp. 252–258. DOI: 10.1016/j.physletb.2005.05.007. arXiv: hep-th/0501070.

[185] V. Bouchard and R. Donagi. "An SU(5) heterotic standard model". In: *Phys. Lett. B* 633 (2006), pp. 783–791. DOI: 10.1016/j.physletb.2005.12.042. arXiv: hep-th/0512149.

[186] R. Blumenhagen, S. Moster, and T. Weigand. "Heterotic GUT and standard model vacua from simply connected Calabi-Yau manifolds". In: *Nucl. Phys. B* 751 (2006), pp. 186–221. DOI: 10.1016/j.nuclphysb.2006.06.005. arXiv: hep-th/0603015.

[187] V. Braun. "On Free Quotients of Complete Intersection Calabi-Yau Manifolds". In: *JHEP* 04 (2011), p. 005. DOI: 10.1007/JHEP04(2011)005. arXiv: 1003.3235 [hep-th].

[188] J. Gray and J. Wang. "Free Quotients of Favorable Calabi-Yau Manifolds". In: (Dec. 2021). arXiv: 2112.12683 [hep-th].

[189] O. Lebedev, H. P. Nilles, S. Raby, S. Ramos-Sanchez, M. Ratz, P. K. S. Vaudrevange, and A. Wingerter. "A Mini-landscape of exact MSSM spectra in heterotic orbifolds". In: *Phys. Lett. B* 645 (2007), pp. 88–94. DOI: 10.1016/j.physletb.2006.12.012. arXiv: hep-th/0611095.

[190] O. Lebedev, H. P. Nilles, S. Ramos-Sanchez, M. Ratz, and P. K. S. Vaudrevange. "Heterotic mini-landscape. (II). Completing the search for MSSM vacua in a Z(6) orbifold". In: *Phys. Lett. B* 668 (2008), pp. 331–335. DOI: 10.1016/j.physletb.2008.08.054. arXiv: 0807.4384 [hep-th].

[191] L. B. Anderson, J. Gray, Y.-H. He, and A. Lukas. "Exploring Positive Monad Bundles And A New Heterotic Standard Model". In: *JHEP* 02 (2010), p. 054. DOI: 10.1007/JHEP02(2010)054. arXiv: 0911.1569 [hep-th].

[192] A. Constantin, Y.-H. He, and A. Lukas. "Counting String Theory Standard Models". In: *Phys. Lett. B* 792 (2019), pp. 258–262. DOI: 10.1016/j.physletb.2019.03.048. arXiv: 1810.00444 [hep-th].

[193] L. B. Anderson, J. Gray, A. Lukas, and E. Palti. "Two Hundred Heterotic Standard Models on Smooth Calabi-Yau Threefolds". In: *Phys. Rev. D* 84 (2011), p. 106005. DOI: 10.1103/PhysRevD.84.106005. arXiv: 1106.4804 [hep-th].

[194] L. B. Anderson, A. Constantin, J. Gray, A. Lukas, and E. Palti. "A Comprehensive Scan for Heterotic SU(5) GUT models". In: *JHEP* 01 (2014), p. 047. DOI: 10.1007/JHEP01(2014)047. arXiv: 1307.4787 [hep-th].

[195] Y.-H. He, S.-J. Lee, A. Lukas, and C. Sun. "Heterotic Model Building: 16 Special Manifolds". In: *JHEP* 06 (2014), p. 077. DOI: 10.1007/JHEP06(2014)077. arXiv: 1309.0223 [hep-th].

[196] A. P. Braun, C. R. Brodie, and A. Lukas. "Heterotic Line Bundle Models on Elliptically Fibered Calabi-Yau Three-folds". In: *JHEP* 04 (2018), p. 087. DOI: 10.1007/JHEP04(2018)087. arXiv: 1706.07688 [hep-th].

[197] R. Blumenhagen, F. Gmeiner, G. Honecker, D. Lust, and T. Weigand. "The Statistics of supersymmetric D-brane models". In: *Nucl. Phys. B* 713 (2005), pp. 83–135. DOI: 10.1016/j.nuclphysb.2005.02.005. arXiv: hep-th/0411173.

[198] F. Gmeiner, R. Blumenhagen, G. Honecker, D. Lust, and T. Weigand. "One in a billion: MSSM-like D-brane statistics". In: *JHEP* 01 (2006), p. 004. DOI: 10.1088/1126-6708/2006/01/004. arXiv: hep-th/0510170.

[199] M. Cvetič, J. Halverson, L. Lin, M. Liu, and J. Tian. "Quadrillion F-Theory Compactifications with the Exact Chiral Spectrum of the Standard Model". In: *Phys. Rev. Lett.* 123.10 (2019), p. 101601. DOI: 10.1103/PhysRevLett.123.101601. arXiv: 1903.00009 [hep-th].

[200] F. A. Bogomolov. "Holomorphic tensors and vector bundles on projective varieties". In: *Mathematics of the USSR-Izvestiya* 13.3 (1979), p. 499.

[201] A. Lukas, B. A. Ovrut, and D. Waldram. "Nonstandard embedding and five-branes in heterotic M theory". In: *Phys. Rev. D* 59 (1999), p. 106005. DOI: 10.1103/PhysRevD.59.106005. arXiv: hep-th/9808101.

[202] S. K. Donaldson. "Anti Self-Dual Yang-Mills Connections Over Complex Algebraic Surfaces and Stable Vector Bundles". In: *Proceedings of the London Mathematical Society* 3.1 (1985), pp. 1–26.

[203] L. B. Anderson, J. Gray, A. Lukas, and B. Ovrut. "Stabilizing the Complex Structure in Heterotic Calabi-Yau Vacua". In: *JHEP* 02 (2011), p. 088. DOI: 10.1007/JHEP02(2011)088. arXiv: 1010.0255 [hep-th].

[204] L. B. Anderson, J. Gray, A. Lukas, and B. Ovrut. "Stabilizing All Geometric Moduli in Heterotic Calabi-Yau Vacua". In: *Phys. Rev. D* 83 (2011), p. 106011. DOI: 10.1103/PhysRevD.83.106011. arXiv: 1102.0011 [hep-th].

[205] J. Gray and J. Wang. “Jumping Spectra and Vanishing Couplings in Heterotic Line Bundle Standard Models”. In: *JHEP* 11 (2019), p. 073. DOI: 10.1007/JHEP11(2019)073. arXiv: 1906.09373 [hep-th].

[206] N. Cabo Bizet, C. Damian, O. Loaiza-Brito, D. K. M. Peña, and J. A. Montañez Barrera. “Testing Swampland Conjectures with Machine Learning”. In: *Eur. Phys. J. C* 80.8 (2020), p. 766. DOI: 10.1140/epjc/s10052-020-8332-9. arXiv: 2006.07290 [hep-th].

[207] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. *OpenAI Gym*. 2016. arXiv: 1606.01540 [cs.LG].

[208] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba. “Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation”. In: *Advances in neural information processing systems* 30 (2017), pp. 5279–5288. arXiv: 1708.05144 [cs.LG].

[209] P. Candelas and R. Davies. “New Calabi-Yau Manifolds with Small Hodge Numbers”. In: *Fortsch. Phys.* 58 (2010), pp. 383–466. DOI: 10.1002/prop.200900105. arXiv: 0809.4681 [hep-th].

[210] A. Constantin. “Heterotic String Models on Smooth Calabi-Yau Threefolds”. PhD thesis. Oxford U., 2018. arXiv: 1808.09993 [hep-th].

[211] A. Hill et al. *Stable Baselines*. https://github.com/hill-a/stable-baselines. 2018.

[212] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas. *Sample Efficient Actor-Critic with Experience Replay*. 2017. arXiv: 1611.01224 [cs.LG]. URL: http://arxiv.org/abs/1611.01224.

[213] A. Irpan. *Deep Reinforcement Learning Doesn't Work Yet*. https://www.alexirpan.com/2018/02/14/rl-hard.html. 2018.

[214] R. Islam, P. Henderson, M. Gomrokchi, and D. Precup. “Reproducibility of Benchmarked Deep Reinforcement Learning Tasks for Continuous Control”. In: arXiv:1708.04133 (2017). arXiv: 1708.04133 [cs.LG].

[215] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. “Deep Reinforcement Learning that Matters”. In: (2017). arXiv: 1709.06560 [cs.LG].

[216] M. Larfors, A. Lukas, and F. Ruehle. “Calabi-Yau Manifolds and SU(3) Structure”. In: *JHEP* 01 (2019), p. 171. DOI: 10.1007/JHEP01(2019)171. arXiv: 1805.08499 [hep-th].

[217] C. Vafa. “The String landscape and the swampland”. In: (Sept. 2005). arXiv: hep-th/0509212.

[218] E. Palti. “The Swampland: Introduction and Review”. In: *Fortsch. Phys.* 67.6 (2019), p. 1900037. DOI: 10.1002/prop.201900037. arXiv: 1903.06239 [hep-th].

[219] A. Strominger, S.-T. Yau, and E. Zaslow. “Mirror symmetry is T duality”. In: *Nucl. Phys. B* 479 (1996), pp. 243–259. DOI: 10.1016/0550-3213(96)00434-8. arXiv: hep-th/9606040.

[220] M. Demirtas, C. Long, L. McAllister, and M. Stillman. “The Kreuzer-Skarke Axiverse”. In: *JHEP* 04 (2020), p. 138. DOI: 10.1007/JHEP04(2020)138. arXiv: 1808.01282 [hep-th].

[221] T. pandas development team. *pandas-dev/pandas: Pandas*. Version latest. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[222] Wes McKinney. “Data Structures for Statistical Computing in Python”. In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 56 –61. DOI: 10.25080/Majora-92bf1922-00a.

[223] C. R. Harris et al. “Array programming with NumPy”. In: *Nature* 585.7825 (2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[224] J. D. Hunter. “Matplotlib: A 2D graphics environment”. In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[225] M. L. Waskom. “seaborn: statistical data visualization”. In: *Journal of Open Source Software* 6.60 (2021), p. 3021. DOI: 10.21105/joss.03021. URL: https://doi.org/10.21105/joss.03021.

[226] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[227] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.4.0)*. https://www.sagemath.org. 2021.

[228] J. Rambau. “TOPCOM: Triangulations of Point Configurations and Oriented Matroids”. In: *Proceedings of the International Congress of Mathematical Software*. available https://www.zib.de/PaperWeb/abstracts/ZR-02-17. 2002. URL: http://www.zib.de/PaperWeb/abstracts/ZR-02-17.

[229] A. Meurer et al. “SymPy: symbolic computing in Python”. In: *PeerJ Computer Science* 3 (Jan. 2017), e103. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.103. URL: https://doi.org/10.7717/peerj-cs.103.

[230] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR *4-2-1 — A computer algebra system for polynomial computations*. http://www.singular.uni-kl.de. 2021.

Acta Universitatis Upsaliensis

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 2138

Editor: The Dean of the Faculty of Science and Technology

A doctoral dissertation from the Faculty of Science and Technology, Uppsala University, is usually a summary of a number of papers. A few copies of the complete dissertation are kept at major Swedish research libraries, while the summary alone is distributed internationally through the series Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology. (Prior to January, 2005, the series was published under the title "Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology".)

Distribution: publications.uu.se
urn:nbn:se:uu:diva-471719

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2022