Machine learning for QFT and string theory

Harold Erbin

Università di Torino & Infn (Italy)

In collaboration with:

- Maxim Chernodub (Institut Denis Poisson, Tours)
- Riccardo Finotello (Università di Torino)
- Vladimir Goy, Ilya Grishmanovky, Alexander Molochkov (Pacific Quantum Center, Vladivostok)

arXiv: 1911.07571, 2006.09113, 2007.13379, 2007.15706

Outline: 1. Motivations

Machine learning

Machine Learning (ML)

Set of techniques for pattern recognition / function approximation without explicit programming.

- learn to perform a task implicitly by optimizing a cost function
- flexible $\rightarrow$ wide range of applications
- general theory unknown (black box problem)

Machine learning

Machine Learning (ML)

Set of techniques for pattern recognition / function approximation without explicit programming.

- learn to perform a task implicitly by optimizing a cost function
- flexible $\rightarrow$ wide range of applications
- general theory unknown (black box problem)

Question

Where does it fit in theoretical physics?

Machine learning

Machine Learning (ML)

Set of techniques for pattern recognition / function approximation without explicit programming.

- learn to perform a task implicitly by optimizing a cost function
- flexible $\to$ wide range of applications
- general theory unknown (black box problem)

Question

Where does it fit in theoretical physics?

- particle physics
- cosmology
- many-body physics
- quantum information
- lattice theories
- string theory

[1903.10563, Carleo et al.]

Outline: 1.1. Lattice QFT

Lattice QFT

Ideas:

- discretization of action and path integral
- Monte Carlo (MC) algorithms

Applications:

- QCD phenomenology (confinement, quark-gluon plasma...)
- statistical physics
- Regge / CDT / matrix model approaches to quantum gravity
- supersymmetric gauge theories for AdS/CFT

Lattice QFT

Ideas:

- discretization of action and path integral
- Monte Carlo (MC) algorithms

Applications:

- QCD phenomenology (confinement, quark-gluon plasma. . .)
- statistical physics
- Regge / CDT / matrix model approaches to quantum gravity
- supersymmetric gauge theories for AdS/CFT

Limitations:

- computationally expensive
- convergence only for some regions of the parameter space

$\rightarrow$ use machine learning

Machine learning for Monte Carlo

Support MC with ML [1605.01735, Carrasquilla-Melko]:

- compute observables, predict phase
- generalize to other parameters
- identify order parameters
- generate configurations
- reduce autocorrelation times
- prevent sign problem for fermions

Selected references: 1608.07848, Broecker et al.; 1703.02435, Wetzel; 1705.05582, Wetzel-Scherzer; 1805.11058, Abe et al.; 1801.05784, Shanahan-Trewartha-Detmold; 1807.05971, Yoon-Bhattacharya-Gupta; 1810.12879, Zhou-Endrõdi-Pang; 1811.03533, Urban-Pawlowski; 1904.12072, Albergo-Kanwar-Shanahan; 1909.06238, Matsumoto-Kitazawa-Kohno; . . .
QCD review: 1904.09725, Joó et al.

Outline: 1.2. String theory

String phenomenology

Goal

Find “the” Standard Model from string theory

Method:

- type II / heterotic strings, M-theory, F-theory: $D = 10, 11, 12$
- vacuum choice (flux compactification):
  - typically Calabi–Yau (CY) 3- or 4-fold
  - fluxes and intersecting branes

  $\rightarrow$ reduction to $D = 4$
- check consistency (tadpole, susy. . .)
- read the $D = 4$ QFT (gauge group, spectrum. . .)

String phenomenology

Goal

Find “the” Standard Model from string theory

Method:

- type II / heterotic strings, M-theory, F-theory: $D = 10, 11, 12$
- vacuum choice (flux compactification):
  - typically Calabi–Yau (CY) 3- or 4-fold
  - fluxes and intersecting branes

  $\rightarrow$ reduction to $D = 4$
- check consistency (tadpole, susy...)
- read the $D = 4$ QFT (gauge group, spectrum...)

No vacuum selection mechanism $\Rightarrow$ string landscape

Landscape mapping

String phenomenology:

- find consistent string models
- find generic/common features
- reproduce the Standard model

Landscape mapping

String phenomenology:

- find consistent string models
- find generic/common features
- reproduce the Standard model

Typical questions:

- understand geometries: properties and equations involving many integers
- find the distribution of some parameters
- find good EFTs to describe low-energy limit
- (construct an explicit string field theory)

Number of geometries

Calabi–Yau (CY) manifolds

- CICY (complete intersection in products of projective spaces): 7890 (3-fold), 921,497 (4-fold)
- Kreuzer–Skarke (reflexive polyhedra): 473,800,776 ($d = 4$)

String models and flux vacua

- type IIA/IIB models: 10^{500}
- F-Theory: 10^{755} (geometries), $10^{272,000}$ (flux vacua)

[Lerche-Lüst-Schellekens '89; hep-th/0303194, Douglas; hep-th/0307049, Ashok-Douglas; hep-th/0409207, Douglas; 1511.03209, Taylor-Wang; 1706.02299, Halverson-Long-Sun; 1710.11235, Taylor-Wang; 1810.00444, Constantin-He-Lukas]

Challenges

- numbers are huge
- difficult maths problems (NP-complete, NP-hard, undecidable) [hep-th/0602072, Denef-Douglas; 1809.08279, Halverson-Ruehle]
- methods from algebraic topology: cumbersome, rarely closed-form formulas

Challenges

- numbers are huge
- difficult maths problems (NP-complete, NP-hard, undecidable) [hep-th/0602072, Denef-Douglas; 1809.08279, Halverson-Ruehle]
- methods from algebraic topology: cumbersome, rarely closed-form formulas

$\rightarrow$ use machine learning

Selected references: 1404.7359, Abel-Rizos; 1706.02714, He; 1706.03346, Krefl-Song; 1706.07024, Ruehle; 1707.00655, Carifio-Halverson-Krioukov-Nelson; 1804.07296, Wang-Zang; 1806.03121, Bull-He-Jejjala-Mishra; ...

Review: Ruehle '20

Plan

1. Casimir energy for arbitrary boundaries for a $3d$ scalar field
   $\rightarrow$ speed improvement and accuracy
   [2006.09113, Chernodub-HE-Goy-Grishmanovky-Molochkov]

2. deconfinement phase transition in $3d$ compact QED
   $\rightarrow$ extrapolation to different lattice sizes
   [1911.07571, Chernodub-HE-Goy-Molochkov]

3. Hodge numbers for CICY 3-fold
   $\rightarrow$ extends methods for algebraic topology
   [2007.13379, 2007.15706, HE-Finotello]

Outline: 2. Machine learning

Definition

Machine learning (Samuel)

The field of study that gives computers the ability to learn without being explicitly programmed.

Machine learning (Mitchell)

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

Approaches to machine learning

Learning approaches (task: $x \longrightarrow y$):

- supervised: learn a map from a set $(x_{\text{train}}, y_{\text{train}})$, then predict y_{data} from x_{data}
- unsupervised: give x_{data} and let the machine find structure (i.e. appropriate y_{data})
- reinforcement: give x_{data}, let the machine choose output following rules, reward good and/or punish bad results, iterate

Applications

General idea = pattern recognition:

- classification / clustering
- regression (prediction)
- transcription / translation
- structuring
- anomaly detection
- denoising
- synthesis and sampling
- density estimation
- conjecture generation

Applications in industry: computer vision, language processing, medical diagnosis, fraud detection, recommendation system, autonomous driving. . .

Examples

Multimedia applications:

- MuZero, AlphaZero (DeepMind): play chess, shogi, Go
- MuZero, AlphaStar (Deepmind), OpenAI Five, etc.: play video games (Starcraft 2, Dota 2, Atari...)
- GPT-2 (OpenAI): conditional synthetic text sampling (+ general language tasks)
- DeepL: translation
- Yolo: real-time object detection [1804.02767]
- Face2Face: real-time face reenactement (deep fake)

Science applications:

- AlphaFold (DeepMind): protein folding
- (astro)particles [1806.11484, 1807.02876, darkmachines.org]
- astronomy [1904.07248]
- geometrical structures [geometricdeeplearning.com]

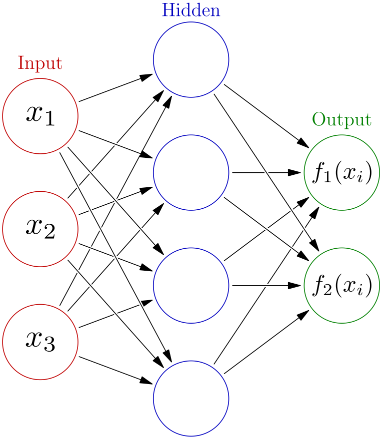
Deep neural network

$$x_{i_1}^{(1)} := x_{i_1}$$

$$x_{i_2}^{(2)} = g^{(2)}(W_{i_2 i_1}^{(1)} x_{i_1}^{(1)})$$

$$f_{i_3}(x_{i_1}) := x_{i_3}^{(3)} = g^{(3)}(W_{i_3 i_2}^{(2)} x_{i_2}^{(2)})$$

$$i_1 = 1, 2, 3;\ i_2 = 1, \ldots, 4;\ i_3 = 1, 2$$

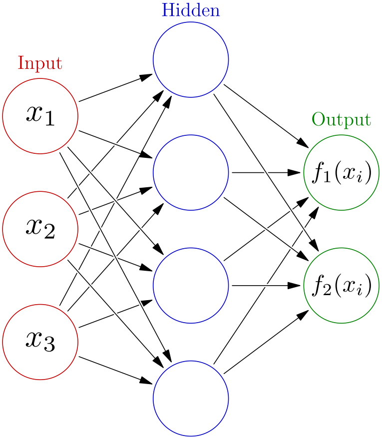


Deep neural network

Architecture:

- 1–many hidden layers, vector $x^{(n)}$
- link: weighted input, matrix $W^{(n)}$
- neuron: non-linear "activation function" $g^{(n)}$

$$x^{(n+1)} = g^{(n+1)}(W^{(n)}x^{(n)})$$



Generic method: fixed functions $g^{(n)}$, learn weights $W^{(n)}$

Learning method

- define a loss function L

$$L = \sum_{i=1}^{N_{\text{train}}} \mathrm{distance}(y_i^{(\text{train})}, y_i^{(\text{pred})})$$

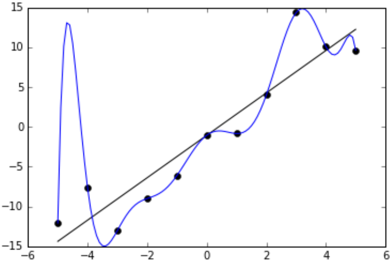
- minimize the loss function (iterated gradient descent...)

Learning method

- define a loss function L

$$L = \sum_{i=1}^{N_{\text{train}}} \text{distance}(y_i^{(\text{train})}, y_i^{(\text{pred})})$$

- minimize the loss function (iterated gradient descent. . .)
- main risk: overfitting (= cannot generalize)
  $\rightarrow$ various solutions (regularization, dropout. . .)
  $\rightarrow$ split data set in two (training and test)



ML workflow

"Naive" workflow:

1. get raw data
2. write neural network with many layers
3. feed raw data to neural network
4. get nice results (or give up)

https://xkcd.com/1838

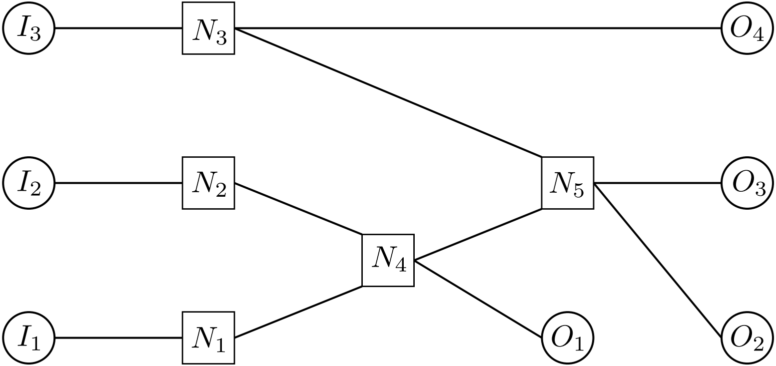
ML workflow

Real-world workflow:

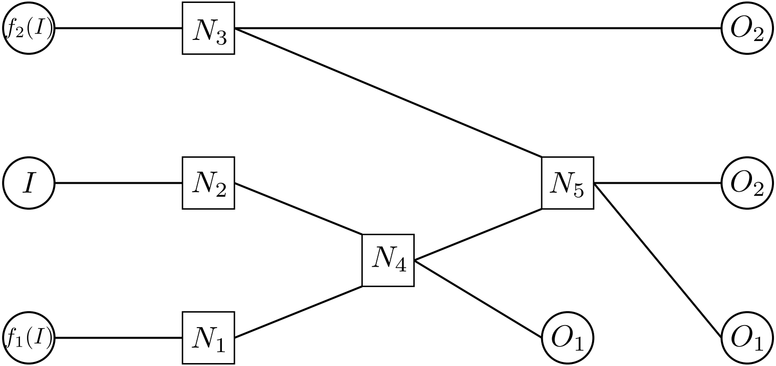
1. understand the problem
2. exploratory data analysis
   - feature engineering
   - feature selection
3. baseline model
   - full working pipeline
   - lower-bound on accuracy
4. validation strategy
5. machine learning model
6. ensembling

Pragmatic ref.: [coursera.org/learn/competitive-data-science]

Advanced neural network



Advanced neural network



Particularities:

- $f_i(I)$: engineered features
- identical outputs (stabilisation)

Some results

Universal approximation theorem

Under mild assumptions, a feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$.

Some results

Universal approximation theorem

Under mild assumptions, a feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$.

Comparisons

- results comparable and sometimes superior to human experts (cancer diagnosis, traffic sign recognition. . .)
- perform generically better than any other machine learning algorithm

Some results

Universal approximation theorem

Under mild assumptions, a feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$.

Comparisons

- results comparable and sometimes superior to human experts (cancer diagnosis, traffic sign recognition. . .)
- perform generically better than any other machine learning algorithm

Drawbacks

- black box
- magic
- numerical

(= how to extract analytical / predictable / exact results?)

Outline: 3. Lattice QFT

Outline: 3.1. Introduction to lattice QFT

Discretization

- Euclidean periodic lattice Λ, spacing a

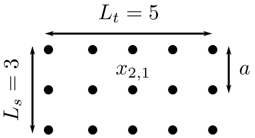
$$x^\mu / a \in \Lambda = \{0, \ldots, L_t - 1\} \times \{0, \ldots, L_s - 1\}^{d-1}$$

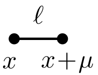
- scalar field $\in$ site: $\quad \phi(x) \longrightarrow \phi_x$
- gauge field $\to$ phase factor $\in$ link $\ell = (x, \mu)$

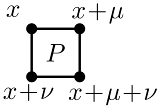
$$U_\mu(x) = P \exp\left(\mathrm{i} \int_x^{x+\hat{\mu}} \mathrm{d}x'^\nu \, A_\nu \right) \quad \to \quad U_{x,\mu} = \mathrm{e}^{\mathrm{i} a A_\mu + O(a^2)}$$

- field strength $\to$ phase factor $\in$ plaquette $P = (x, \mu, \nu)$

$$U_{\mu\nu}(x) = U_\nu(x)^\dagger U_\mu(x + \hat{\nu})^\dagger U_\nu(x + \hat{\mu}) U_\mu(x)$$
$$\to \quad U_{x,\mu\nu} = \mathrm{e}^{\mathrm{i} a^2 F_{\mu\nu} + O(a^3)}$$







Monte Carlo methods

- interpret path integral → statistical system partition function

$$\int \prod_x \mathrm{d}\phi_x \longrightarrow \sum_C \quad \text{and} \quad \langle \mathcal{O}[C] \rangle = \frac{\sum_C \mathrm{e}^{-\beta S[C]} \mathcal{O}[C]}{\sum_C \mathrm{e}^{-\beta S[C]}}$$

  $C = \{\phi_x\}_{x \in \Lambda}$ field configuration
- Monte Carlo: sample susbset $E = \{C_1, \ldots, C_N\}$ s.t.

$$\mathrm{Prob}(C_k) = Z^{-1}\, \mathrm{e}^{-\beta S[C_k]}, \qquad \langle \mathcal{O} \rangle = \frac{1}{N} \sum_{k=1}^{N} \mathcal{O}[C_k]$$

- Markov chain: built E by sequence of state
  stochastic transition $\mathrm{Prob}(C_k \to C_{k+1}) = \mathrm{Prob}(C_k, C_{k+1})$
- Metropolis algorithm: select trial configuration C', accept
  $C_{k+1} = C'$ with probability given by action difference

$$\mathrm{Prob}(C_k \to C') = \min\left(1, \mathrm{e}^{-\beta(S[C'] - S[C_k])}\right)$$

$$\mathrm{Prob}(C_k \to C_k) = 1 - \mathrm{Prob}(C_k \to C')$$

Outline: 3.2. Casimir effect

Scalar field theory

- action and Dirichlet boundary conditions ($\mu = 0, 1, 2$)

$$S[\phi] = \frac{1}{2} \int \mathrm{d}^3 x \, \partial_\mu \phi \partial^\mu \phi, \qquad \phi(x)|_{x \in \mathcal{S}} = 0$$

- static boundary $\mathcal{S}$: quasi-parallel lines or closed curve
- Casimir energy

$$\mathcal{E}_\mathcal{S} = \langle T_{00} \rangle_\mathcal{S} - \langle T_{00} \rangle_0$$

  $=$ change in vacuum energy density due to defect
- motivations:
  - 3*d* scalar: simplest model exhibiting this effect
  - interesting for technological applications (micro-electromechanical and micro-fluidic systems) [Rodriguez-Capasso-Johnson, Nature Photonics '11]
  - modify QCD vacuum $\rightarrow$ chiral symmetry breaking / confinement [1805.11887, Chernodub et al.]

Discretization

- partition function and action

$$Z = \int \prod_x \mathrm{d}\phi_x \, \mathrm{e}^{-S[\phi]}, \qquad S[\phi] = \frac{1}{2} \sum_{x,\mu} (\phi_{x+\hat{\mu}} - \phi_x)^2$$

  $\hat{\mu}$ unit vector in direction μ

- Euclidean energy

$$T_{00} = \frac{1}{4} \sum_{\mu} \eta_\mu \left[(\phi_{x+\hat{\mu}} - \phi_x)^2 + (\phi_x - \phi_{x-\hat{\mu}})^2 \right]$$

  $(\eta_0, \eta_1, \eta_2) = (-1, 1, 1)$

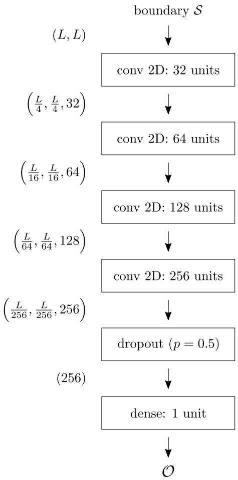
- Hybrid Monte Carlo algorithm (MC + molecular dynamics)
- boundaries: parallel lines or closed curves

ML analysis

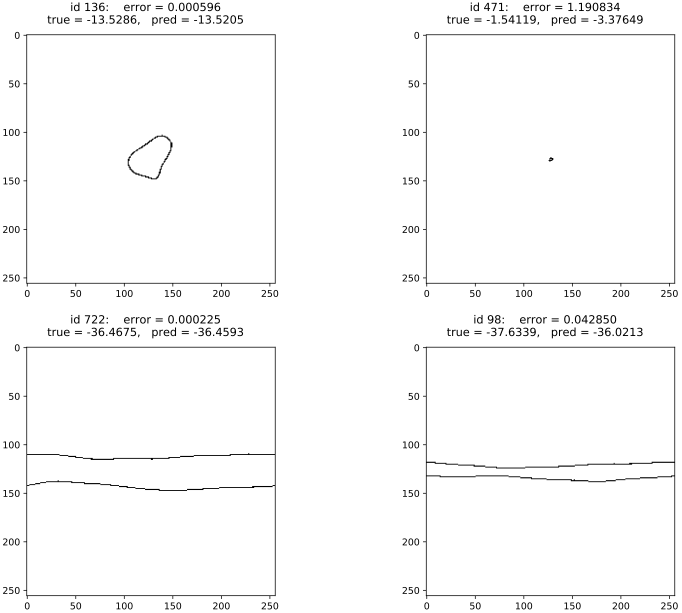
- input: $2d$ boundary condition (= BW image), $L_s = 255$
- output: Casimir energy $\in \mathbb{R}$
- network: 4 convolution layers, $\sim$ 400k param.
- data: 80% train, 10% validate, 10% test
- time comparison:
  - MC = 3.1 hours / sample
  - training = 5 min / 800 samples
  - prediction = 5 ms / sample

$\rightarrow$ 10^6 times faster

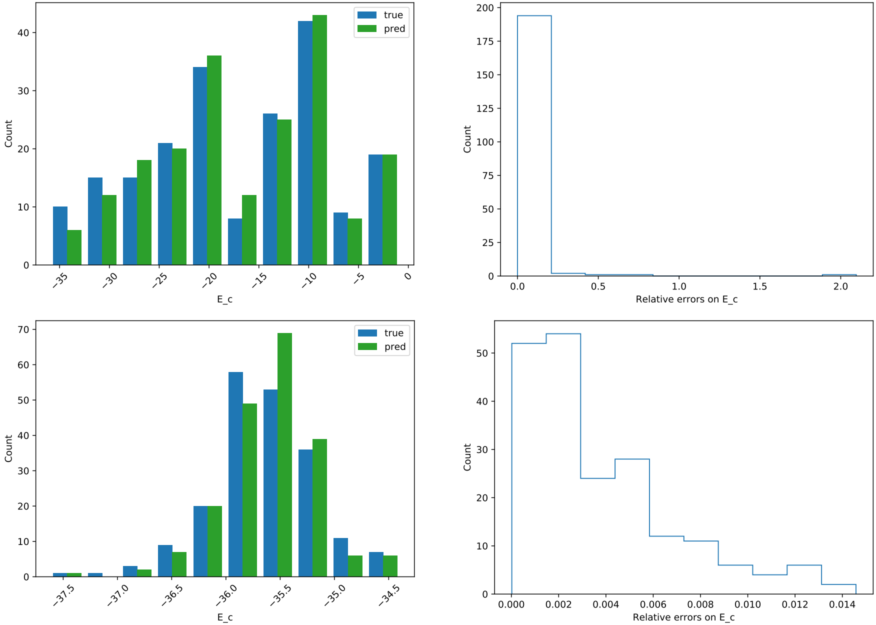
[1911.07571]



Examples

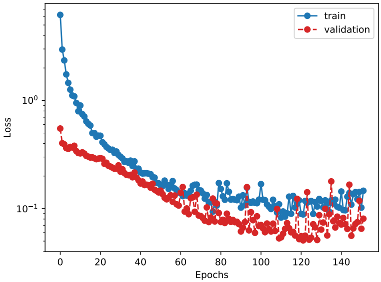

id 136: error = 0.000596
true = -13.5286, pred = -13.5205
id 471: error = 1.190834
true = -1.54119, pred = -3.37649
id 722: error = 0.000225
true = -36.4675, pred = -36.4593
id 98: error = 0.042850
true = -37.6339, pred = -36.0213
0
50
100
150
200
250

Predictions

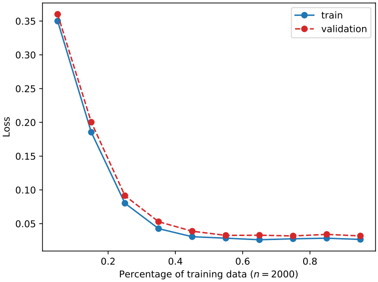


Training and learning curves

Training curve



Learning curve



Relative errors and RMSE

errors (relative)	closed curves	parallel lines
mean	0.064	0.0037
min	0.000087	0.000019
75%	0.069	0.0051
max	2.1	0.016
RMSE	0.97	0.18

$$\text{rel. error} = \left| \frac{ML - MC}{MC} \right|$$

Comparison MC and ML

Best and worst in terms of absolute error (closed curves):

	MC		ML	
	$\mathcal{E}$	$\mathrm{err}_{\mathcal{E}}$	$\mathcal{E}$	$\mathrm{err}_{\mathcal{E}}$
best	-22.62	0.13	-22.60	0.014
	-20.34	0.12	-20.34	0.0018
	-12.22	0.09	-12.23	0.011
	-9.57	0.16	-9.57	0.0028
	-9.57	0.13	-9.56	0.011
worst	-0.82	0.12	-2.54	1.72
	-1.63	0.10	-2.67	1.04
	-1.48	0.09	-2.30	0.82

Outline: 3.3. 3d QED

Compact QED: properties

Model: $3d$ compact QED at finite temperature [Polyakov '76]

- well understood [hep-lat/0106021, Chernodub-Ilgenfritz-Schiller]
- good toy model for QCD (linear confinement, mass gap generation, temperature phase transition)
- topological defects (monopoles): drive phase transition

Confinement-deconfinement phase transition:

- low temperature: confinement caused by Coulomb monopole-antimonopole gas
- high temperature: deconfinement, rare monopoles bound into neutral monopole-antimonopole pairs

Compact QED: lattice

- angle $\theta_{x,\mu} = a\, A_\mu(x) \in [-\pi, \pi)$ lattice gauge field
- elementary plaquette angle

$$\theta_{P_{x,\mu\nu}} = \theta_{x,\mu} + \theta_{x+\hat{\mu},\nu} - \theta_{x+\hat{\nu},\mu} - \theta_{x,\nu} = a^2 F_{x,\mu\nu} + O(a^4)$$

- lattice action: continuum coupling g, temperature T

$$S[\theta] = \beta \sum_x \sum_{\mu<\nu} \left(1 - \cos\theta_{P_{x,\mu\nu}}\right), \qquad \beta = \frac{1}{ag^2} = \frac{L_t T}{g^2}$$

- Polyakov loop $\rightarrow$ order parameter for confinement

$$L(\boldsymbol{x}) = \mathrm{e}^{\mathrm{i} \sum_{t=0}^{L_t-1} \theta_0(t,\boldsymbol{x})}, \qquad \langle L(\boldsymbol{R})\rangle = \mathrm{e}^{-F/T}$$

  infinitely heavy charged test particle, free energy F
- confining potential (σ string tension)

$$\langle L(\mathbf{0}) L(\boldsymbol{R})\rangle \propto \mathrm{e}^{-L_t V(\boldsymbol{R})}, \qquad V(\boldsymbol{R}) \sim_{T\sim 0} \sigma |\boldsymbol{R}|$$

Monte Carlo computations

- MC simulations for different temperatures β:
  1. gauge field configurations
  2. monopole configurations
  3. extract properties

- useful quantities:
  - spatially averaged Polyakov loop L
  - plaquettes U (spatial and temporal)
  - monopole density ρ

- study phase transition from $|L|$:
  - critical temperature β_c
  - phase $\phi = 0$ (confined) or $\phi = 1$ (deconfined)

[2006.09113]

ML analysis

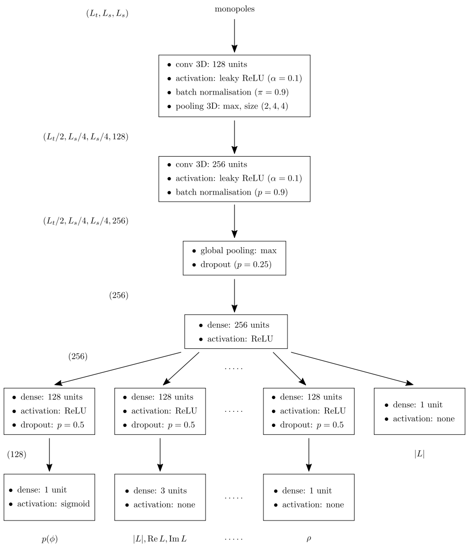
Objective:

1. train for $(L_t, L_s) = (4, 16)$
2. predict phase $\mathrm{Prob}(\phi)$, Polyakov loop $|L|$ for $(L_t, L_s) \neq (4, 16)$ ($L_t = 4, 6, 8$, $L_s = 16, 32$)
3. compute the critical temperature

Characteristics:

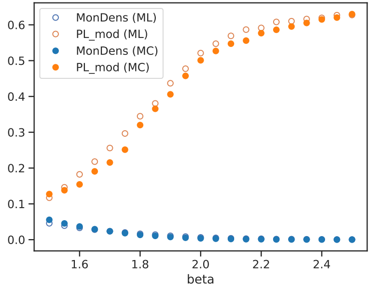
- input: $3d$ monopole configuration ($= 3d$ BW image)
- main output: $|L|$, $\mathrm{Prob}(\phi)$
- auxiliary output: L, U, ρ, β
- network: convolution + dense layers, 1.28M parameters
- data:
  - train 1: 2000 samples for each $\beta \in [1.5, 3]$, $\Delta\beta = 0.05$
  - train 2: 100 samples for each $\beta \in [0.1, 2.2]$, $\Delta\beta = 0.1$
  - validation/test: 200 samples for each $\beta \in [1.5, 2.5]$, $\Delta\beta = 0.05$

Neural network

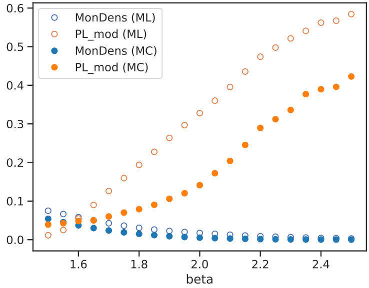


Predictions (temperature, density)

$$(L_t, L_s) = (4, 16)$$

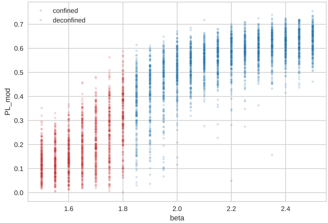


$$(L_t, L_s) = (6, 32)$$

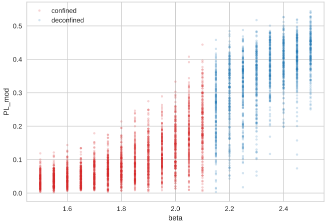


Predictions (phase)

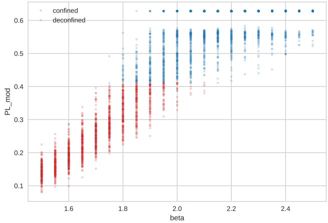
$(L_t, L_s) = (4, 16)$, MC



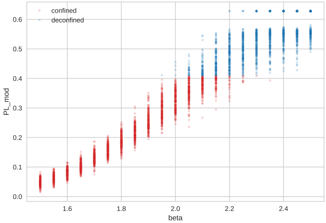
$(L_t, L_s) = (6, 32)$, MC



ML



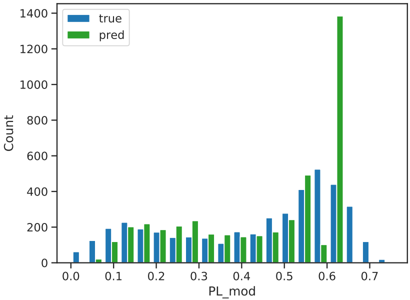
ML

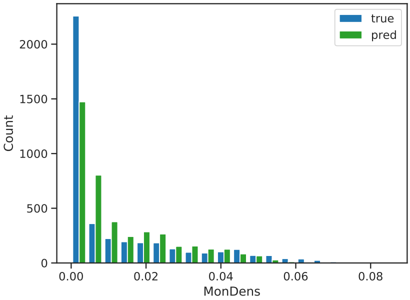


Predictions (errors)

	RMSE
$\lvert L \rvert$	0.089
ρ	0.0027
β	0.19
U	0.016

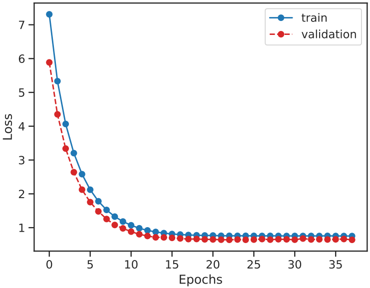
ϕ	score
accuracy	94.5%
precision	95.8%
recall	96.0%
F_1	0.96



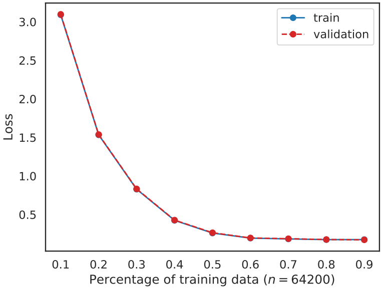


Training and learning curves

Training curve



Learning curve



Critical temperature: estimations

- maximum slope of Polyakov loop:

$$\beta_c = \operatorname*{argmax}_{\beta} \partial_\beta \langle |L| \rangle_\beta$$

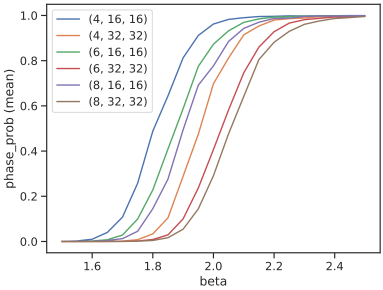
- maximum probability variance:

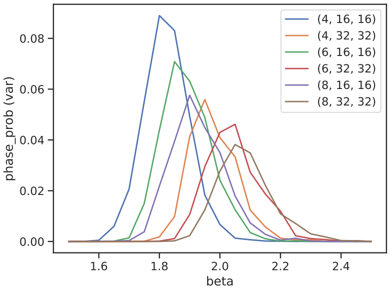
$$\beta_c = \operatorname*{argmax}_{\beta} \mathrm{Var}_\beta\big(p(\phi)\big)$$

- maximum probability uncertainty:

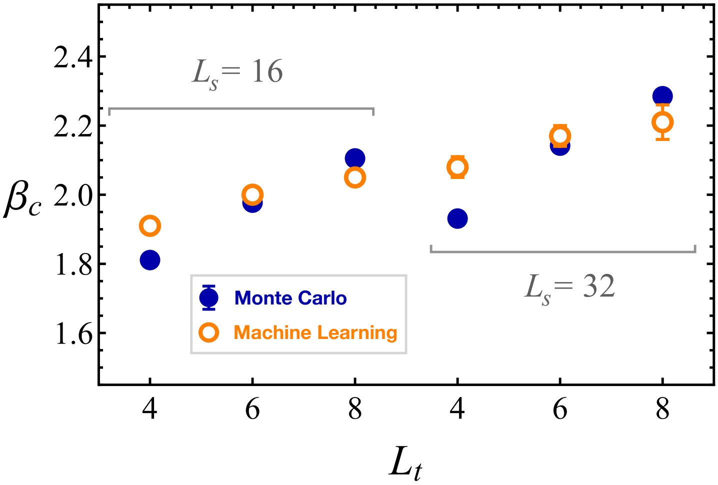
$$\langle p(\phi) \rangle_\beta |_{\beta_c} = 0.5$$

Phase probability distribution





Critical temperature: predictions



Error correction

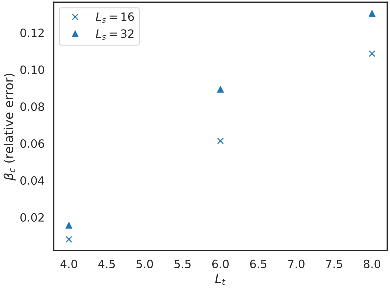
β_c prediction could be improved to $< 5\%$ error:

- modify decision function

$$\phi = \begin{cases} 0 & p(\phi) < p_c \\ 1 & p(\phi) \geq p_c \end{cases}$$

  tune p_c, predict β_c from $\langle\phi\rangle$, $\mathrm{Var}\,\phi$

- error linear in L_t
  $\rightarrow$ apply correction



Notes

- form of boosting/hyperparameter tuning using several lattices
- useful if considering many more lattices

Outline: 4. Calabi–Yau 3-folds

Outline: 4.1. Calabi–Yau 3-folds

Calabi-Yau

Complete intersection Calabi–Yau (CICY) 3-fold:

- CY: complex manifold with vanishing first Chern class
- complete intersection: non-degenerate hypersurface in products of projective spaces
- hypersurface = solution to system of homogeneous polynomial equations

Calabi-Yau

Complete intersection Calabi–Yau (CICY) 3-fold:

- CY: complex manifold with vanishing first Chern class
- complete intersection: non-degenerate hypersurface in products of projective spaces
- hypersurface = solution to system of homogeneous polynomial equations
- described by configuration matrix $m \times k$

$$X = \left[\begin{array}{c|ccc} \mathbb{P}^{n_1} & a_1^1 & \cdots & a_k^1 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{P}^{n_m} & a_1^m & \cdots & a_k^m \end{array} \right]$$

$$\dim_{\mathbb{C}} X = \sum_{r=1}^{m} n_r - k = 3, \qquad n_r + 1 = \sum_{\alpha=1}^{k} a_\alpha^r$$

- a_α^r power of coordinates on $\mathbb{P}^{n_r}$ in αth equation

Configuration matrix

Examples

- quintic ($a = 0, \ldots, 4$)

$$\left[\begin{array}{c|c} \mathbb{P}^4_x & 5 \end{array}\right] \implies \sum_a (X^a)^5 = 0$$

- 2 projective spaces, 3 equations ($a, \alpha = 0, \ldots, 3$)

$$\left[\begin{array}{c|ccc} \mathbb{P}^3_x & 3 & 0 & 1 \\ \mathbb{P}^3_y & 0 & 3 & 1 \end{array}\right] \implies \begin{cases} f_{abc}\, X^a X^b X^c = 0 \\ g_{\alpha\beta\gamma}\, Y^\alpha Y^\beta Y^\gamma = 0 \\ h_{a\alpha}\, X^a Y^\alpha = 0 \end{cases}$$

Configuration matrix

Examples

- quintic $(a = 0, \dots, 4)$

$$\left[\ \mathbb{P}^4_x \ \middle|\ 5\ \right] \implies \sum_a (X^a)^5 = 0$$

- 2 projective spaces, 3 equations $(a, \alpha = 0, \dots, 3)$

$$\left[\begin{array}{c|ccc} \mathbb{P}^3_x & 3 & 0 & 1 \\ \mathbb{P}^3_y & 0 & 3 & 1 \end{array}\right] \implies \begin{cases} f_{abc}\, X^a X^b X^c = 0 \\ g_{\alpha\beta\gamma}\, Y^\alpha Y^\beta Y^\gamma = 0 \\ h_{a\alpha}\, X^a Y^\alpha = 0 \end{cases}$$

Classification

- invariances ($\to$ huge redundancy)
  - permutation of lines and columns
  - identities between subspaces
- but:
  - constraints $\Rightarrow$ bound on matrix size
  - $\exists$ "favourable" configuration

Topology

Why topology?

- no metric known for compact CY (cannot perform KK reduction explicitly)
- topological numbers $\rightarrow$ 4d properties (number of fields, representations, gauge symmetry...)

Topology

Why topology?

- no metric known for compact CY (cannot perform KK reduction explicitly)
- topological numbers → 4d properties (number of fields, representations, gauge symmetry. . .)

Topological properties

- Hodge numbers $h^{p,q}$ (number of harmonic (p,q)-forms) here: $h^{1,1}$, $h^{2,1}$
- Euler number $\chi = 2(h^{1,1} - h^{2,1})$
- Chern classes
- triple intersection numbers
- line bundle cohomologies

Topology

Why topology?

- no metric known for compact CY (cannot perform KK reduction explicitly)
- topological numbers $\to$ 4d properties (number of fields, representations, gauge symmetry. . .)

Topological properties

- Hodge numbers $h^{p,q}$ (number of harmonic (p,q)-forms) here: $h^{1,1}$, $h^{2,1}$
- Euler number $\chi = 2(h^{1,1} - h^{2,1})$
- Chern classes
- triple intersection numbers
- line bundle cohomologies

Datasets

CICY have been classified

- 7890 configurations (but $\exists$ redundancies)
- number of product spaces: 22
- $h^{1,1} \in [0, 19]$, $h^{2,1} \in [0, 101]$
- 266 combinations $(h^{1,1}, h^{2,1})$
- $a^r_\alpha \in [0, 5]$

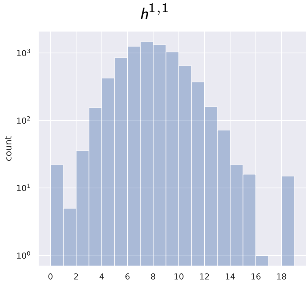
Original [Candelas-Dale-Lutken-Schimmrigk '88][Green-Hubsch-Lutken '89]

- maximal size: 12×15
- number of favourable matrices: 4874

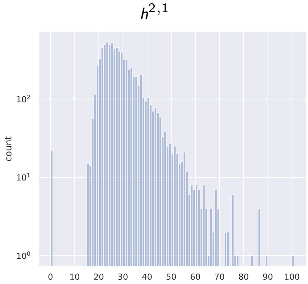
Favourable [1708.07907, Anderson-Gao-Gray-Lee]

- maximal size: 15×18
- number of favourable matrices: 7820

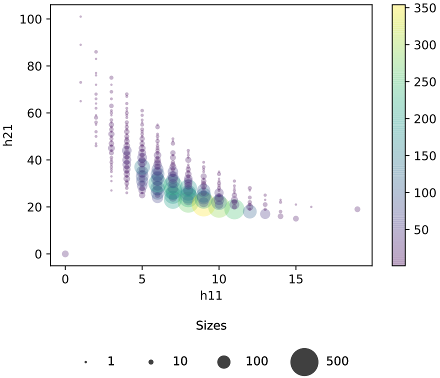
Data



$h^{1,1}$
count
10^0
10^1
10^2
10^3
0 2 4 6 8 10 12 14 16 18



$h^{2,1}$
count
10^0
10^1
10^2
0 10 20 30 40 50 60 70 80 90 100



h21
h11
0 20 40 60 80 100
0 5 10 15
50 100 150 200 250 300 350
Sizes
1
10
100
500

Goal and methodology

Philosophy

Start with the original dataset, derive everything else from configuration matrix and machine learning only.

Current goal

Input: configuration matrix $\longrightarrow$ Output: $h^{1,1}$, $h^{2,1}$

1. CICY: well studied, all topological quantities known
   $\rightarrow$ use as a sandbox
2. improve over [1706.02714, He; 1806.03121, Bull-He-Jejjala-Mishra]
3. both original and favourable datasets

[2007.13379, 2007.15706]

Outline: 4.2. ML analysis

Strategy

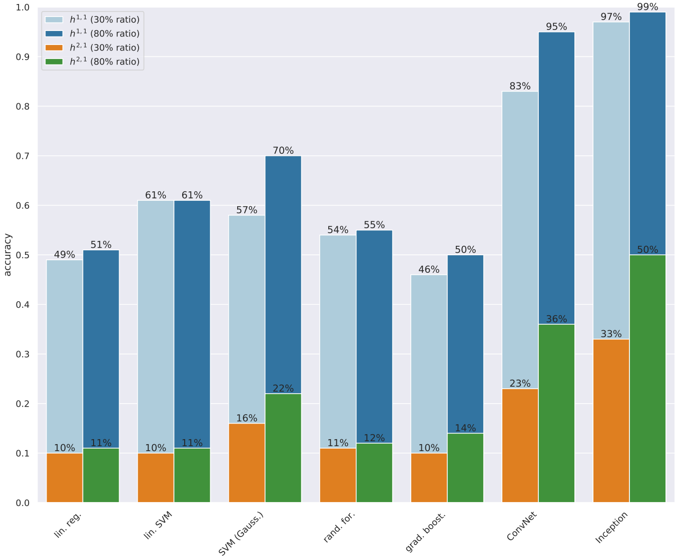
Questions:

- classification or regression?
- feature engineering?
- data diminution: remove outliers (0.49%)?
- data augmentation: generate more inputs using invariances?
- single- or multi-tasking?

Classification vs regression:

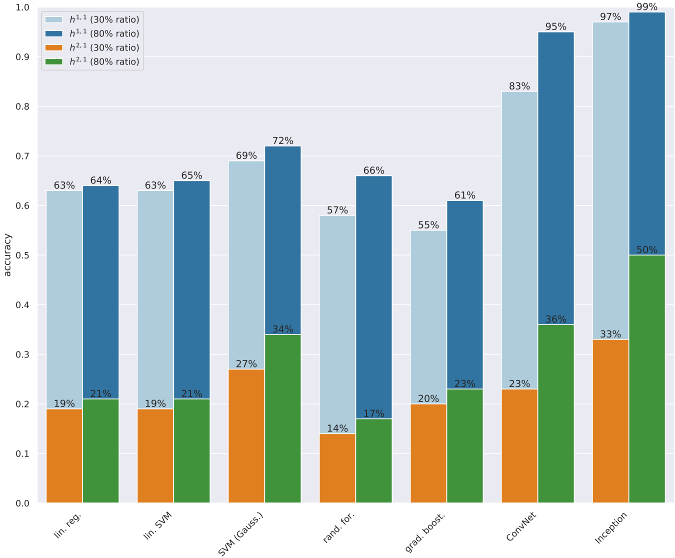
- classification: assume knowledge of boundaries
- regression: better for generalization
  different scales $\rightarrow$ normalize data $\approx$ use continuous variable

Algorithms



$h^{1,1}$ (30% ratio)
$h^{1,1}$ (80% ratio)
$h^{2,1}$ (30% ratio)
$h^{2,1}$ (80% ratio)
accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
49%
51%
10%
11%
61%
61%
10%
11%
57%
70%
16%
22%
54%
55%
11%
12%
46%
50%
10%
14%
83%
95%
23%
36%
97%
99%
33%
50%
lin. reg.
lin. SVM
SVM (Gauss.)
rand. for.
grad. boost.
ConvNet
Inception

Algorithms

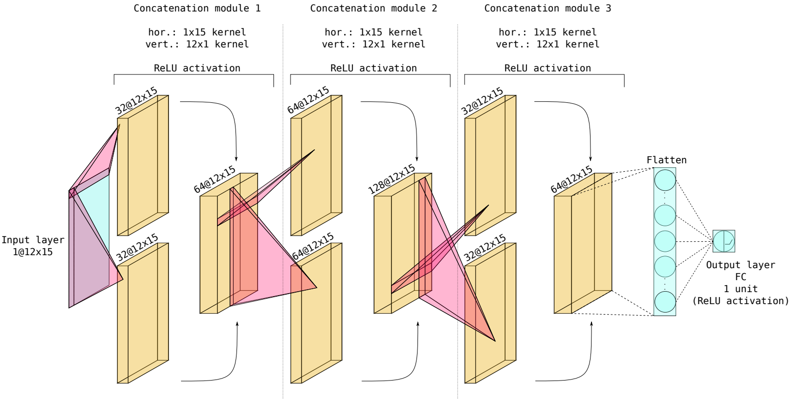


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
accuracy
$h^{1,1}$ (30% ratio)
$h^{1,1}$ (80% ratio)
$h^{2,1}$ (30% ratio)
$h^{2,1}$ (80% ratio)
63%
64%
19%
21%
63%
65%
19%
21%
69%
72%
27%
34%
57%
66%
14%
17%
55%
61%
20%
23%
83%
95%
23%
36%
97%
99%
33%
50%
lin. reg.
lin. SVM
SVM (Gauss.)
rand. for.
grad. boost.
ConvNet
Inception

Inception neural network (1)

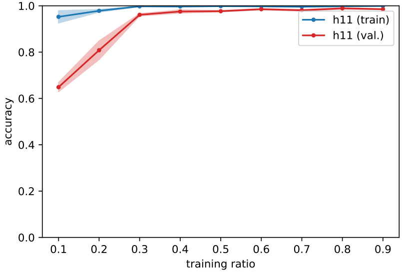
- designed by Google for computer vision
  $\rightarrow$ breakthrough in image classification
  [Szegedy et al., 1409.4842, 1512.00567, 1602.07261]
- sequence of inception modules
  $\rightarrow$ parallel convolutions with kernels of $\neq$ sizes
- learns different combinations of features at different scales
- motivations:
  - $1d$ parallel kernels of maximal sizes: look at all $\mathbb{P}^n$/equations for each equation/$\mathbb{P}^n$ at the same time
  - weight sharing (convolution): same operations for each $\mathbb{P}^n$ and equation
- neural network performs badly on $h_{2,1}$
  (but still much better than any other approach)
  $\rightarrow$ focus on $h^{1,1}$

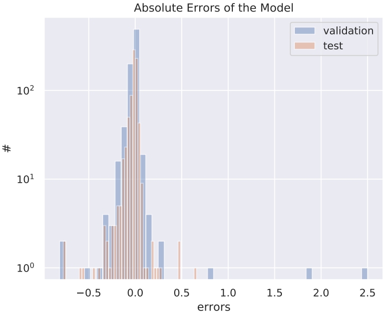
Inception neural network (2)

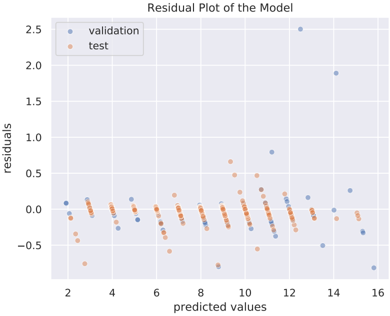


234,000 parameters: 7× less than [Bull et al.]

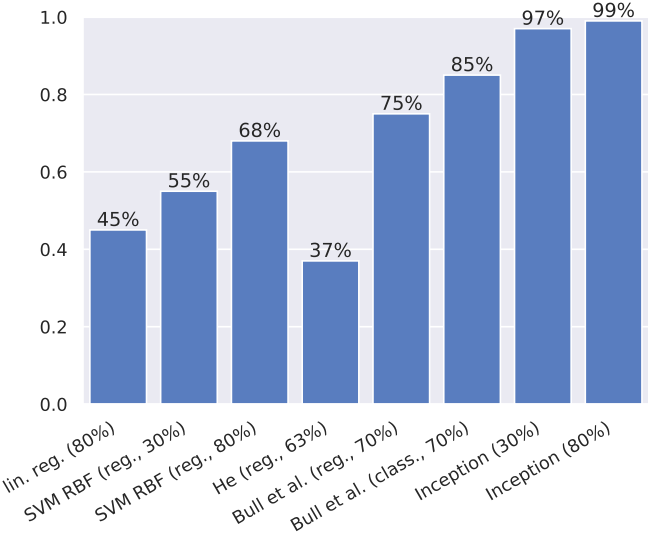
Learning curve and errors



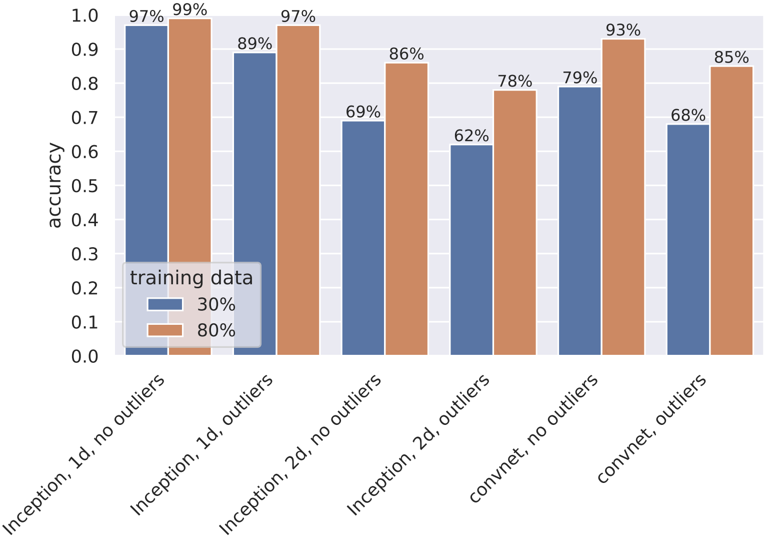




Comparing architectures



Ablation study



Outline: 5. Conclusion

Conclusion

Results:

- machine learning = promising tool for different fields of theoretical physics
- wide range of applications for lattice QFT and string theory

Outlook:

- lattice QFT: dynamics of topological objects in non-Abelian theories, generate boundaries from Casimir energy...
- string theory: extend to other geometries (4-folds, Kreuzer–Skarke polytopes...)
- dissect neural network data to understand what it learns
- symbolic computations with graph representation and reinforcement learning
- explore the space of effective field theories