Casimir effect and 3d QED from machine learning

Harold Erbin

Università di Torino & Infn (Italy)

In collaboration with: M. Chernodub (Tours), V. Goy, I. Grishmanovky, A. Molochkov (Vladivostok) [1911.07571 + to appear]

Outline: 1. Motivations

Motivations

Machine learning

Introduction to lattice QFT

Casimir effect

3d QED

Conclusion

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 $\rightarrow$ wide range of applications
- general theory unknown (black box problem)

Question

Where does it fit in theoretical physics?

$\rightarrow$ particle physics, cosmology, many-body physics, quantum information, lattice simulations, string vacua...

Lattice QFT

Ideas:

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

Applications:

- access non-perturbative effects, strong-coupling regime
- study phase transitions
- QCD phenomenology (confinement, quark-gluon plasma. . .)
- Regge / CDT approaches to quantum gravity
- supersymmetric gauge theories for AdS/CFT

Lattice QFT

Ideas:

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

Applications:

- access non-perturbative effects, strong-coupling regime
- study phase transitions
- QCD phenomenology (confinement, quark-gluon plasma...)
- Regge / CDT approaches to quantum gravity
- supersymmetric gauge theories for AdS/CFT

Limitations:

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

→ use machine learning

Machine learning for Monte Carlo

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

- compute useful quantities, predict phase
- learn field distribution
- identify important (order) parameters
- generalize to other regions of parameter space
- reduce autocorrelation times
- avoid fermion sign problem

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

Plan

1. Casimir energy for arbitrary boundaries for a $3d$ scalar field
   $\rightarrow$ speed improvement and accuracy

2. deconfinement phase transition in $3d$ compact QED
   $\rightarrow$ extrapolation to different lattice sizes

Outline: 2. Machine learning

Motivations

Machine learning

Introduction to lattice QFT

Casimir effect

3d QED

Conclusion

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

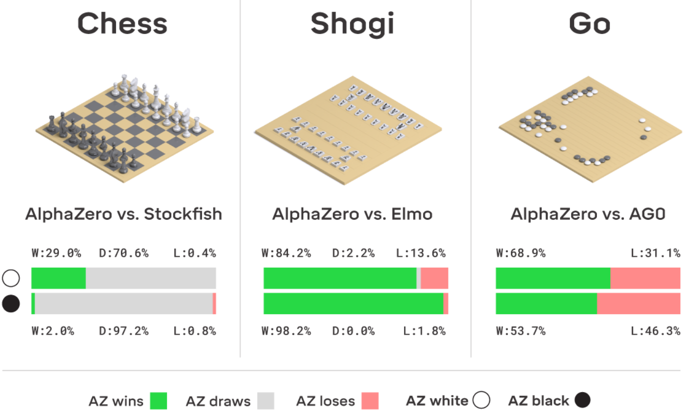
Media applications:

- 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 (+ question answering, reading comprehension, summarization, translation)
- Yolo: real-time object detection [1804.02767]
- Face2Face: real-time face reenactement
- cyclic GAN: image generation and translation

Science applications:

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

Examples (AlphaZero)



Some facts: [Silver et al. '18]

- AlphaGo (supervised): 4–1 against human champion (03/2016)
- AlphaGo Zero (reinforcement): 100–0 against AlphaGo (04/2017)
- positions/s: Stockfish $= 5.8 \times 10^7$, AlphaZero $= 6 \times 10^4$
- training time: 9h (chess), 12 hours (shogi), 13 days (Go)

Examples (GPT)

Human entry:

A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.

Machine answer:

The incident occurred on the downtown train line, which runs from Covington and Ashland stations.

In an email to Ohio news outlets, the U.S. Department of Energy said it is working with the Federal Railroad Administration to find the thief.

"The theft of this nuclear material will have significant negative consequences on public and environmental health, our workforce and the economy of our nation," said Tom Hicks, the U.S. Energy Secretary, in a statement. "Our top priority is to secure the theft and ensure it doesn't happen again."

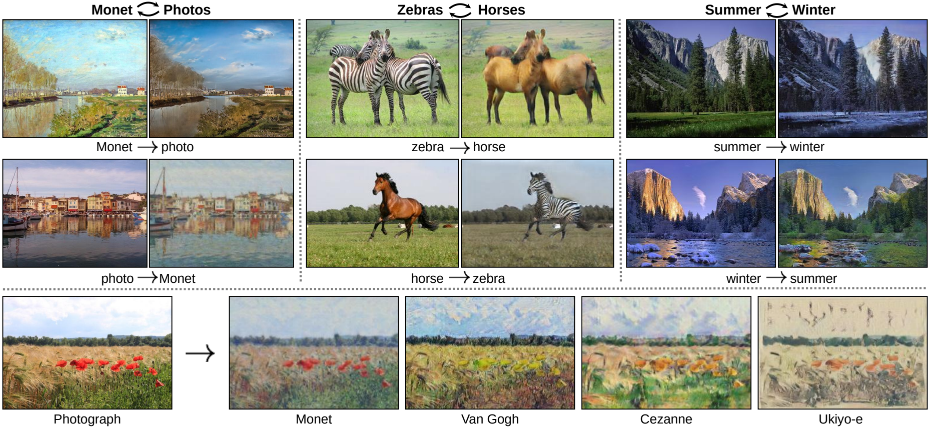
The stolen material was taken from the University of Cincinnati's Research Triangle Park nuclear research site, according to a news release from Department officials. (. . .)

[https://openai.com/blog/better-language-models]

Examples (videos)

- Yolo [https://www.youtube.com/watch?v=VOC3huqHrss]
- Deepfake [https://www.youtube.com/watch?v=ohmajJTcpNk]

Examples (cycle GAN)



[1703.10593]

Examples (protein)

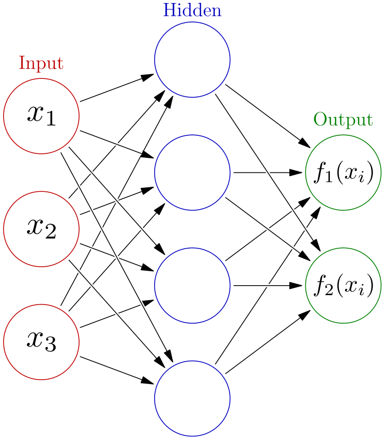
[https://deepmind.com/blog/article/alphafold]

Deep neural network

Architecture:

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

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

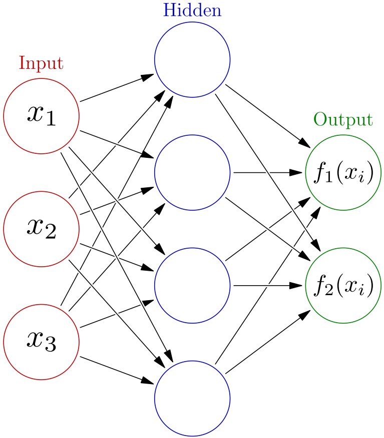


Generic method: fixed functions $g^{(n)}$, learn weights $W^{(n)}$

Deep neural network

$$x_{i_1}^{(1)} := x_{i_1}$$
$$x_{i_2}^{(2)} = g^{(2)}(W_{i_2 i_1}^{(1)} x_{i_1}^{(1)})$$
$$f_{i_3}(x_{i_1}) := x_{i_3}^{(3)} = g^{(3)}(W_{i_3 i_2}^{(2)} x_{i_2}^{(2)})$$

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



Learning method

- define a loss function L

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

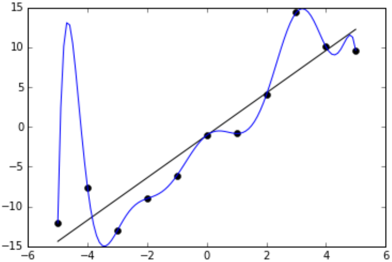
- minimize the loss function (iterated gradient descent...)

Learning method

- define a loss function L

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

- minimize the loss function (iterated gradient descent. . .)
- main risk: overfitting (= cannot generalize)
  → various solutions (regularization, dropout. . .)
  → 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

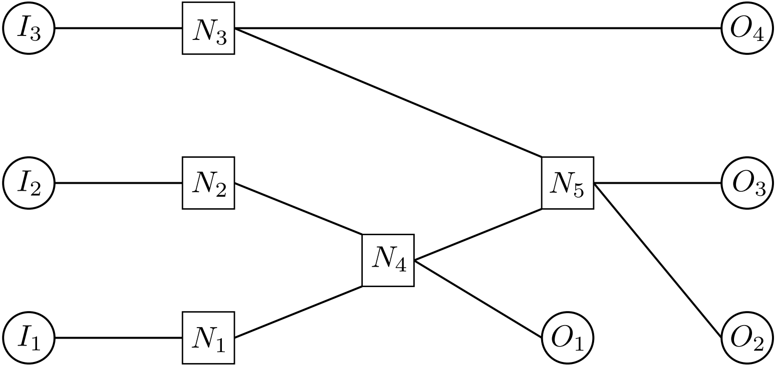
ML workflow

Real-world workflow:

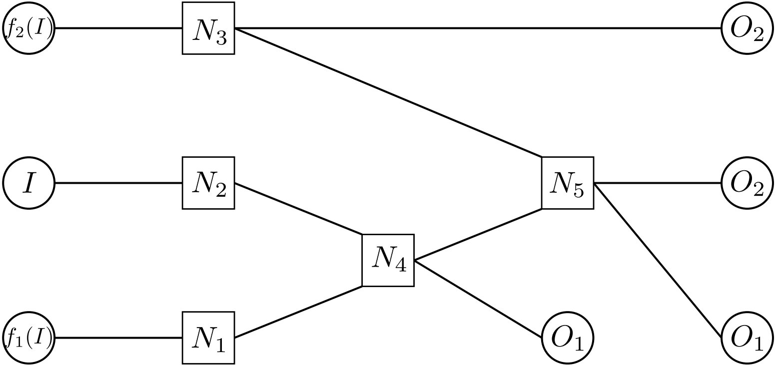
1. understand the problem
2. exploratory data analysis
   - feature engineering
   - feature selection
3. baseline model
   - full working pipeline
   - lower-bound on accuracy
4. validation strategy
5. machine learning model
6. ensembling

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

Complex neural network



Complex 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. Introduction to lattice QFT

Motivations

Machine learning

Introduction to lattice QFT

Casimir effect

3d QED

Conclusion

Discretization

- Euclidean periodic lattice Λ, spacing a

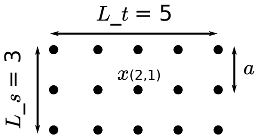
$$x^\mu/a \in \Lambda = \{0, \ldots, L_t - 1\} \times \{0, \ldots, L_s - 1\}^{d-1}$$

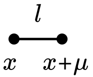
- scalar field $\in$ site: $\quad \phi(x) \longrightarrow \phi_x$
- gauge field $\rightarrow$ phase factor $\in$ link $l = (x, \mu)$

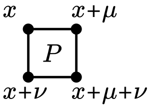
$$U_\mu(x) = P \exp\left(\mathrm{i} \int_x^{x+\hat{\mu}} \mathrm{d}x'^\nu \, A_\nu\right) \quad \rightarrow \quad U_{x,\mu} = \mathrm{e}^{\mathrm{i}aA_\mu + O(a^2)}$$

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

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







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: 4. Casimir effect

Motivations

Machine learning

Introduction to lattice QFT

Casimir effect

3d QED

Conclusion

Scalar field theory

- partition function and action ($\mu = 0, 1, 2$)

$$Z = \int \mathrm{d}\phi \, \mathrm{e}^{-S[\phi]}, \qquad S[\phi] = \frac{1}{2} \int \mathrm{d}^3 x \, \partial_\mu \phi \partial^\mu \phi$$

- Dirichlet boundary condition

$$\phi(x)|_{x \in \mathcal{S}} = 0$$

- Euclidean energy

$$T_{00} = \frac{1}{2} \left[- \left(\frac{\partial \phi}{\partial x_0} \right)^2 + \left(\frac{\partial \phi}{\partial x_1} \right)^2 + \left(\frac{\partial \phi}{\partial x_2} \right)^2 \right]$$

- Casimir energy

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

  = change in vacuum energy density due to boundaries
- modify QCD vacuum → 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

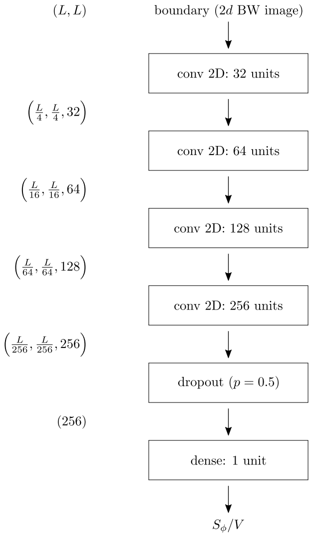
$$T_{00} = \frac{1}{4} \sum_{\mu} \eta_\mu \left[(\phi_{x+\hat{\mu}} - \phi_x)^2 + (\phi_x - \phi_{x-\hat{\mu}})^2 \right]$$

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

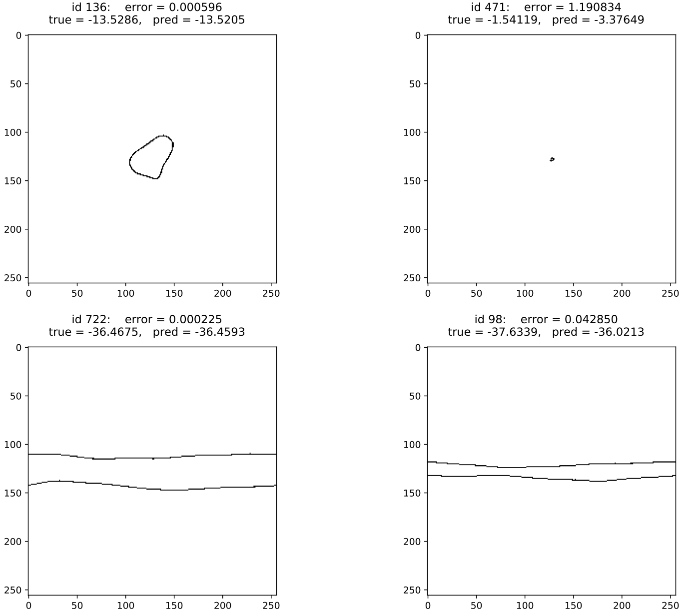
- Hybrid Monte Carlo algorithm (MC + molecular dynamics)
- boundaries: parallel lines or closed curves

ML analysis

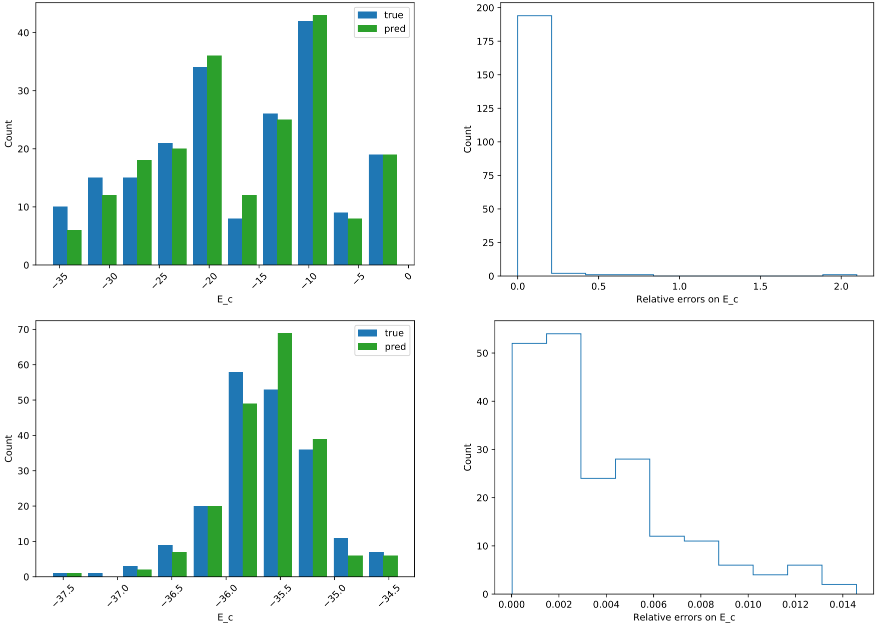
- input: $2d$ boundary condition (= BW image), $L_s = 255$
- output: Casimir energy $\in \mathbb{R}$
- network: 4 convolution layers, 390k parameters
- data: 80% train, 10% validate, 10% test
- time comparison:
  - training = 5 min / 800 samples
  - prediction = 5 ms / 100 samples
  - MC: 3.1 hours / sample



Examples

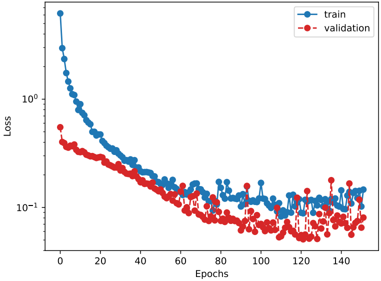

id 136: error = 0.000596
true = -13.5286, pred = -13.5205
id 471: error = 1.190834
true = -1.54119, pred = -3.37649
id 722: error = 0.000225
true = -36.4675, pred = -36.4593
id 98: error = 0.042850
true = -37.6339, pred = -36.0213
0
50
100
150
200
250

Predictions

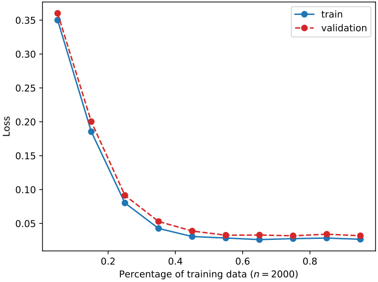


Training and learning curves

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: 5. 3d QED

Motivations

Machine learning

Introduction to lattice QFT

Casimir effect

3d QED

Conclusion

Compact QED: properties

Model: compact QED in $d = 2 + 1$ at finite temperature

- 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 $\to$ 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)

ML analysis

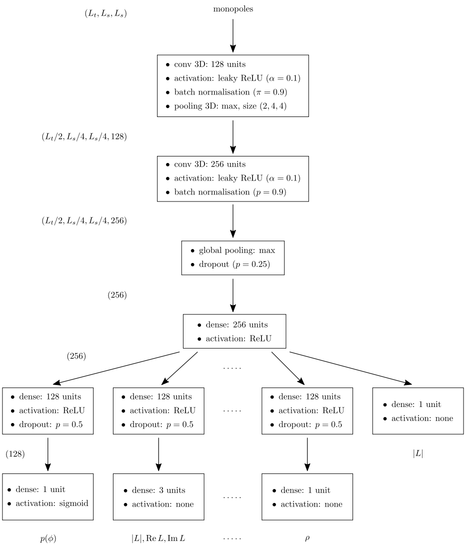
Objective:

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:

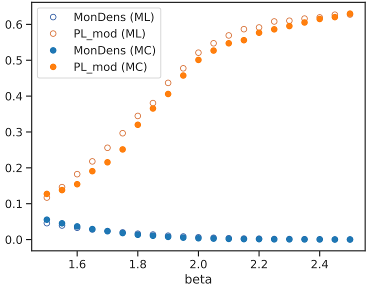
- input: $3d$ monopole configuration (= $3d$ BW image)
- main output: $|L|$, $\mathrm{Prob}(\phi)$
- auxiliary output: L, U, ρ, β
- network: convolution + dense layers, 1.28M parameters
- data:
  - train 1: 2000 samples for each $\beta \in [1.5, 3]$, $\Delta\beta = 0.05$
  - train 2: 100 samples for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

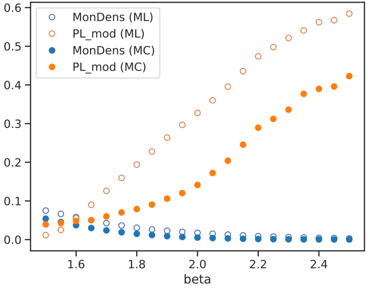


Predictions (temperature, density)

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

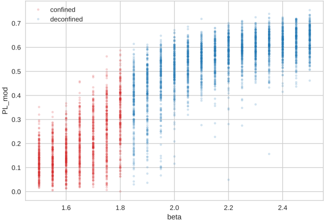


$(L_t, L_s) = (6, 32)$

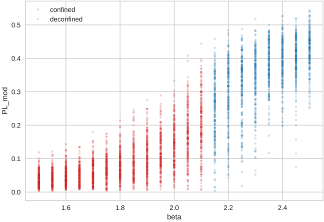


Predictions (phase)

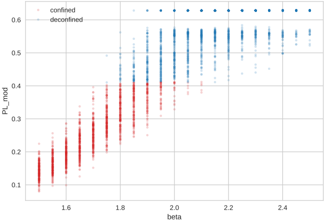
$(L_t, L_s) = (4, 16)$, MC



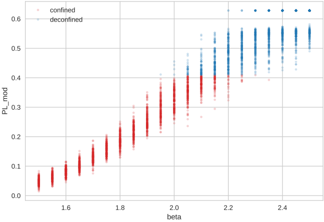
$(L_t, L_s) = (6, 32)$, MC



ML



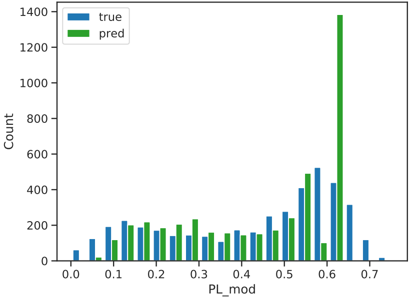
ML

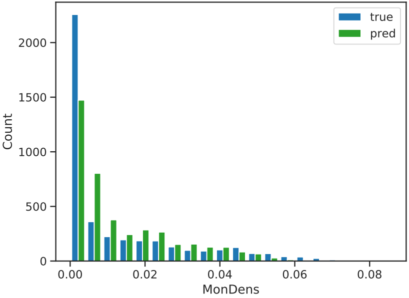


Predictions (errors)

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

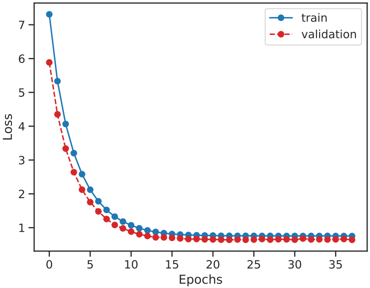
ϕ	score
accuracy	94.5%
precision	95.8%
recall	96.0%
F_1	0.96



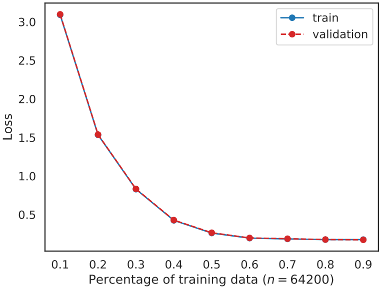


Training and learning curves

Training curve



Learning curve



Critical temperature: estimations

- maximum slope of Polyakov loop:

$$\beta_c = \underset{\beta}{\operatorname{argmax}}\, \partial_\beta \langle |L| \rangle_\beta$$

- maximum probability variance:

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

- maximum probability uncertainty:

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

Critical temperature: predictions

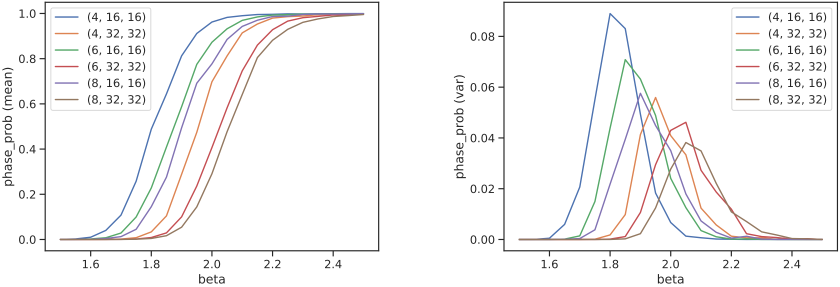
Critical temperatures:

(L_t, L_s)	$(4, 16)$	$(4, 32)$	$(6, 16)$	$(6, 32)$	$(8, 16)$	$(8, 32)$
$\lvert L\rvert$ slope	1.85	2.02	1.90	2.12	1.96	2.06
$\langle p(\phi)\rangle$	1.85	1.99	1.91	2.06	1.94	2.10
Var $p(\phi)$	1.83	1.96	1.88	2.04	1.91	2.07
MC	1.81	1.93	1.98	2.14	2.10	2.29

Errors:

(L_t, L_s)	$(4, 16)$	$(4, 32)$	$(6, 16)$	$(6, 32)$	$(8, 16)$	$(8, 32)$
$\lvert L\rvert$ slope	2.2%	4.7%	4.0%	1.6%	6.7%	10.1%
$\langle p(\phi)\rangle$	2.5%	3.1%	3.3%	3.7%	7.6%	8.5%
Var $p(\phi)$	1.4%	1.8%	5.1%	4.9%	8.8%	9.6%

Phase probability distribution



Error correction

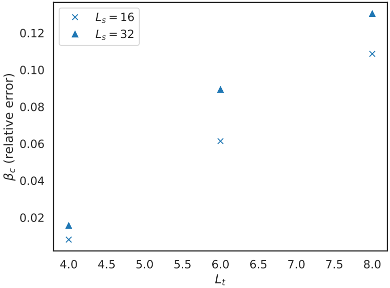
β_c prediction could be improved to $< 5\%$ error:

- 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: 6. Conclusion

Motivations

Machine learning

Introduction to lattice QFT

Casimir effect

3d QED

Conclusion

Outlooks

- Casimir effect
  1. generate boundaries associated to given Casimir energy
  2. compute local action $\rightarrow$ force on probe particle
- $3d$ QED
  1. compute monopoles from gauge field configurations
  2. extend to non-Abelian gauge theories
- applications to supersymmetric field theories