Perspectives of Deep learning techniques in Lattice 1+1d Scalar Field Theory

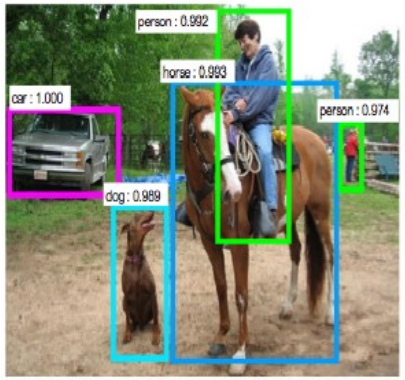
Kai Zhou 周凯 (FIAS, Frankfurt, Germany)

arXiv:1810.12879

In collaboration with :

Gergely Endrődi (ITP, Frankfurt, Germany)
Long-gang Pang (UC Berkeley, Berkeley, USA)

Introduction



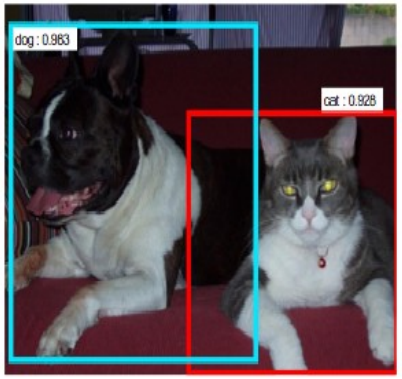
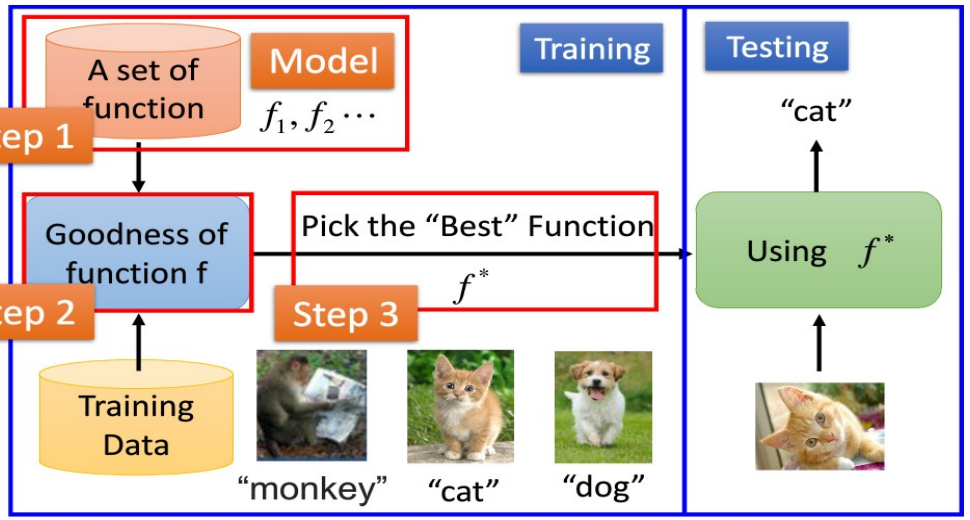


Image Recognition:

Framework

$f($  $)=$ "cat"



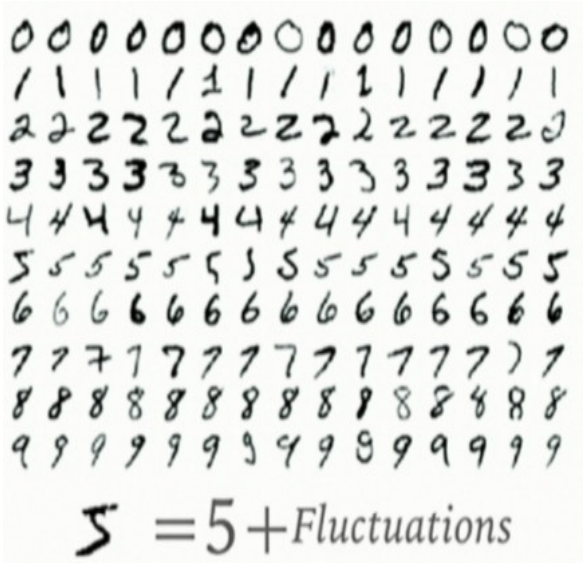
Find and **Decode** the mapping/representations into Deep Neural Network

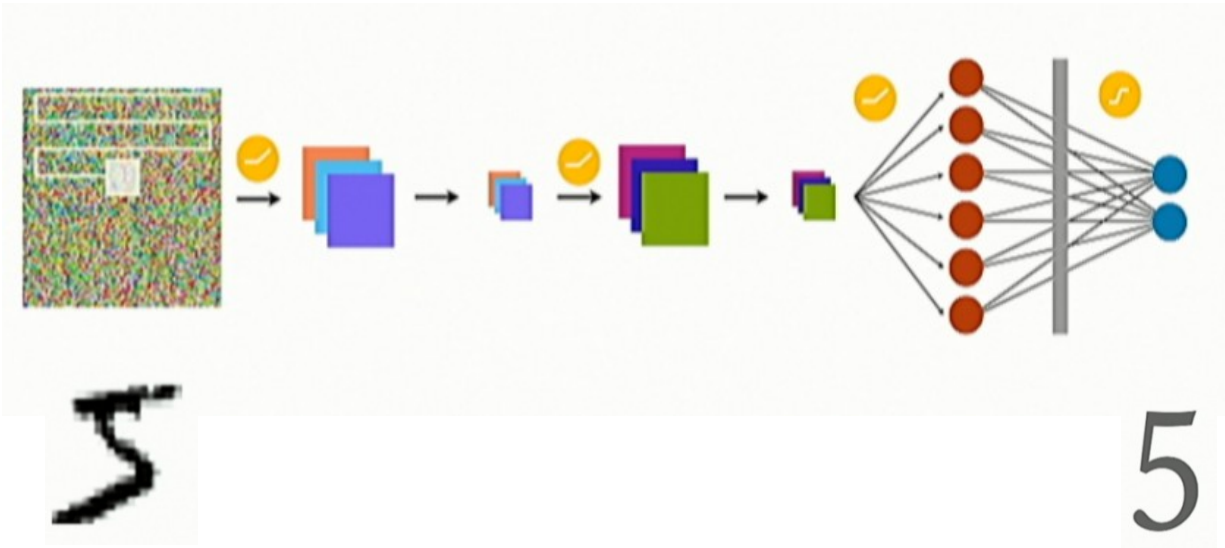
→ **Function approximator**

Universal approximator (Hastad et al 86 & 91)

Introduction

- Convolutional Neural Network has proved to be extremely powerful in Pattern Recognition, Image Classification





- Discriminative learning (**prediction**) : Classification, Regression
  Generative modelling (**generation**) : RBM, VAE, GAN

1+1d $\lambda\phi^4$ (prepare training set)

regularization of continuum Action:

$$S^{\rm lat} = \sum_x \left\{ (4+m^2)\phi^*(x)\phi(x) + \lambda[\phi^*(x)\phi(x)]^2 - \sum_{\nu=1,2} \left[e^{\mu\delta_{\nu,2}}\phi^*(x)+\hat{\nu}) + e^{-\mu\delta_{\nu,2}}\phi^*(x)\phi(x-\hat{\nu}) \right] \right\}$$

Partition sum :

$$\mathcal{Z} = \int D[\phi] exp\left(-S^{\rm lat}[\phi]\right)$$

Dualization approach :

$$\mathcal{Z} = \sum_{\{k,\ell\}} \prod_n \left\{ e^{\mu k_t(n)} \cdot W[s(n)] \cdot \delta[\nabla \cdot k(n)] \cdot \prod_\nu A[k_\nu(x), \ell_\nu(x)] \right\}$$

$$W[s(n)] = \int_0^\infty {\rm d}r\, r^{s(n)+1}\, e^{-(4+m^2)r^2-\lambda r^4}$$

$$s(n) = \sum_\nu [|k_\nu(n)| + |k_\nu(n-\hat{\nu})| + 2(\ell_\nu(n)+\ell_\nu(n-\hat{\nu})]$$

$$A[k_\nu(x), \ell_\nu(x)] = \frac{1}{(\ell_\nu(n)+|k_\nu(n)|)!\;\ell_\nu(n)!}$$

***C. Gattringer and T. Kloiber*, Nucl. Phys. B869 (2013) 56–73**

O. Orasch and C. Gattringer, Int. J.Mod.Phys.A33(2018) no.01,1850010,

Dualization approach for 1+1d $\lambda\phi^4$

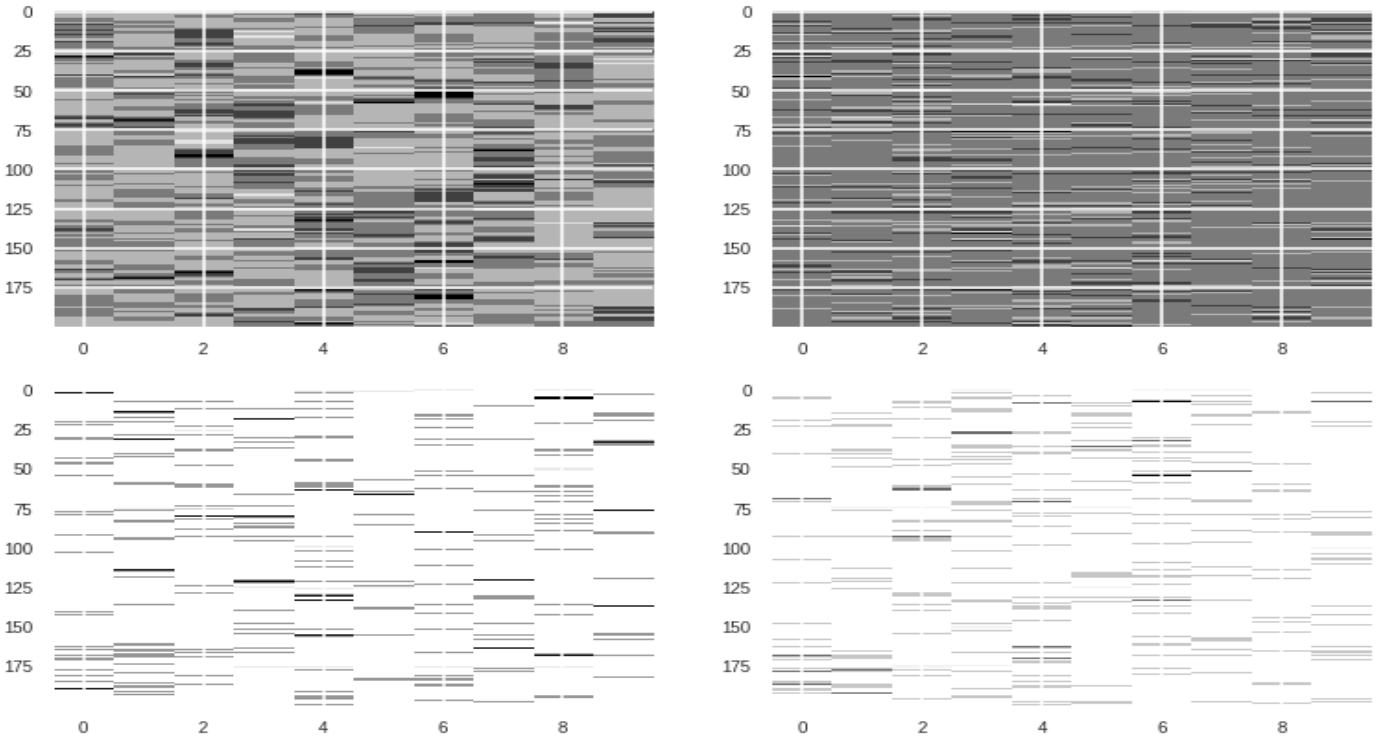
configurations – 4 integer-valued variables $k_t, \quad k_x, \quad l_t, \quad l_x$

$$m = 0.1$$

$$\lambda = 1.0$$

$$N_t = 200$$

$$N_x = 10$$



Divergence constraint : $\nabla \cdot k(n) = \sum_{\nu} [k_\nu(n) - k_\nu(n - \hat{\nu})] = 0$

Observables : n and $|\phi|^2$

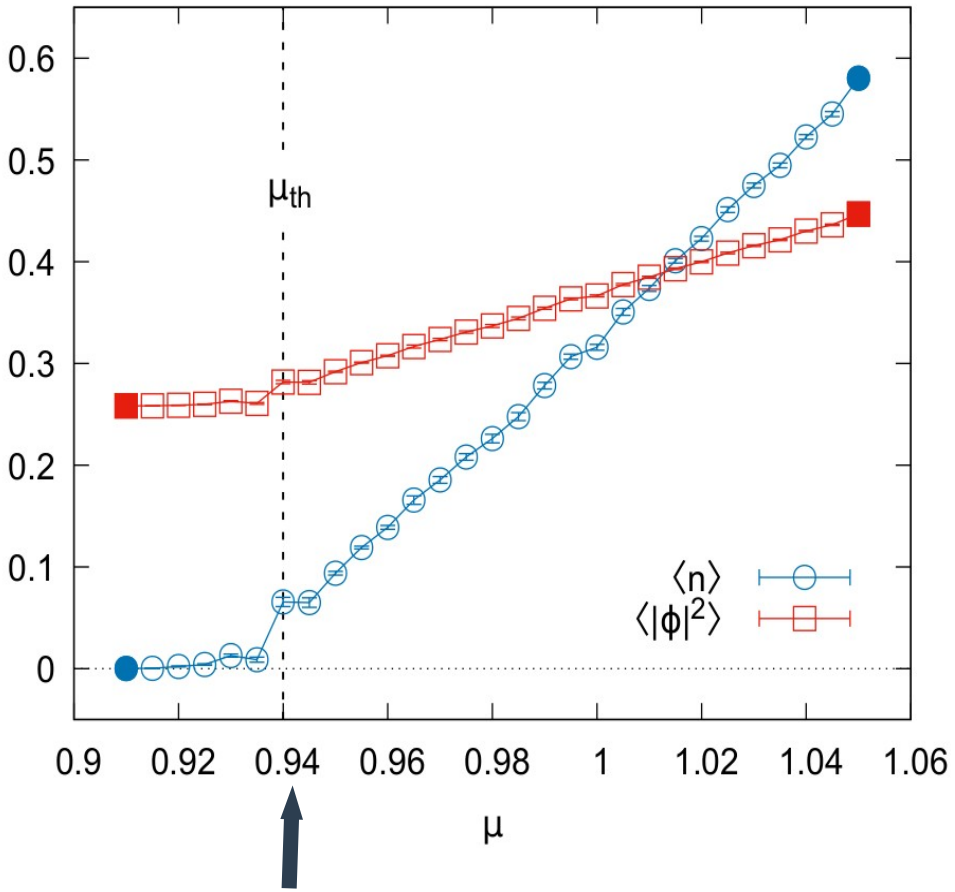
Grand canonical ensemble

$$\langle n \rangle = \frac{T}{L} \frac{\partial \log \mathcal{Z}}{\partial \mu}$$

$$n = \frac{1}{N_x N_t a} \sum_n k_t(n)$$

$$\langle |\phi|^2 \rangle = \frac{T}{L} \frac{\partial \log \mathcal{Z}}{\partial (m^2)}$$

$$|\phi|^2 = \frac{1}{N_x N_t} \sum_n \frac{W[s(n)+2]}{W[s(n)]}$$



Condensation sets in at $\mu_{th} \sim m_{phys} \sim 0.94$

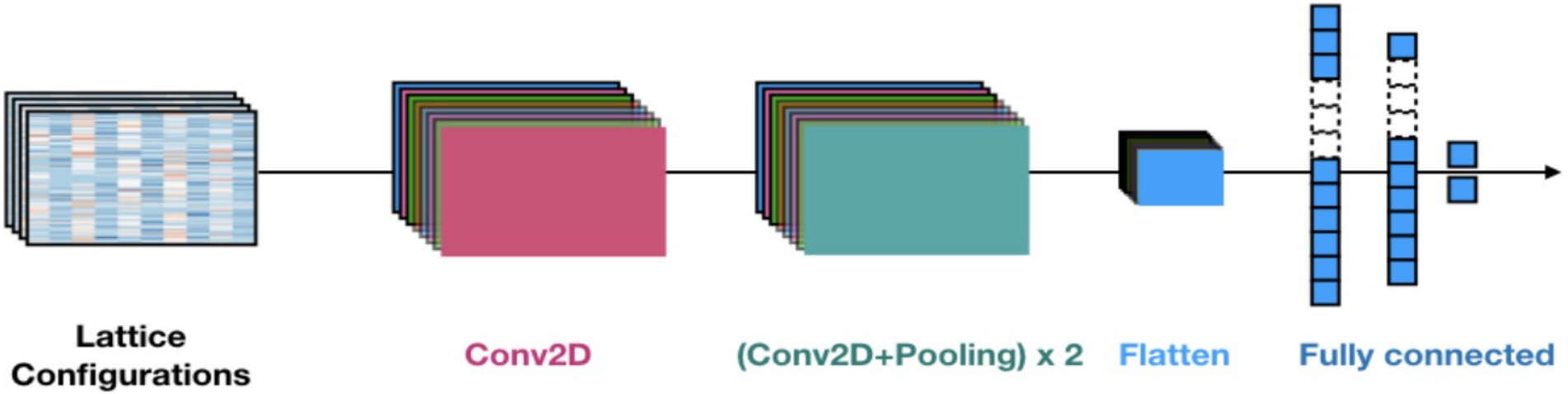
Exploring NN application here

(1) Classification : detect 'phase transition' status based on configurations
(identify order parameter)

(2) Regression : physical observables regression
(identify thermodynamics)

(3) GAN(generate) : * Learn to generate new configurations
* Generate configs with proper distribution
(identify partition function)

DCNN Architechture – Classification

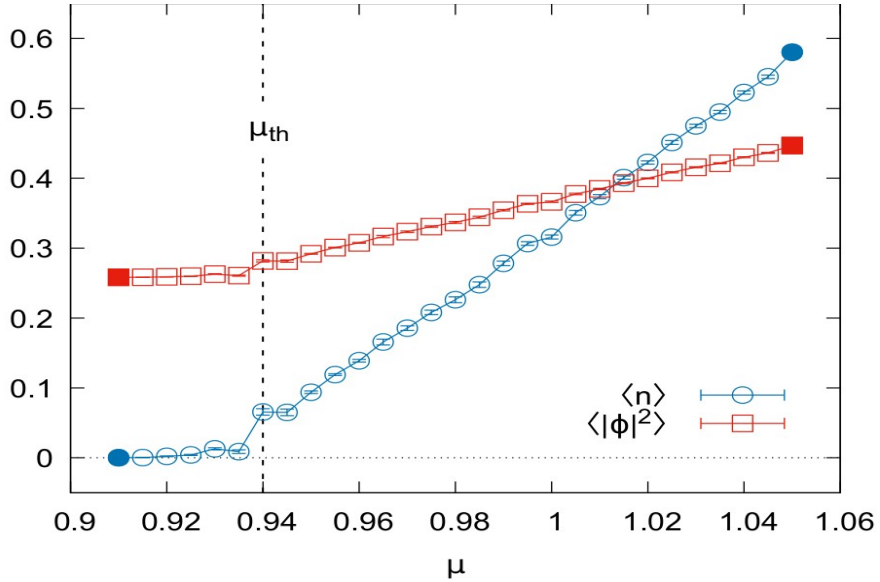


Training set consist two ensembles of configs at

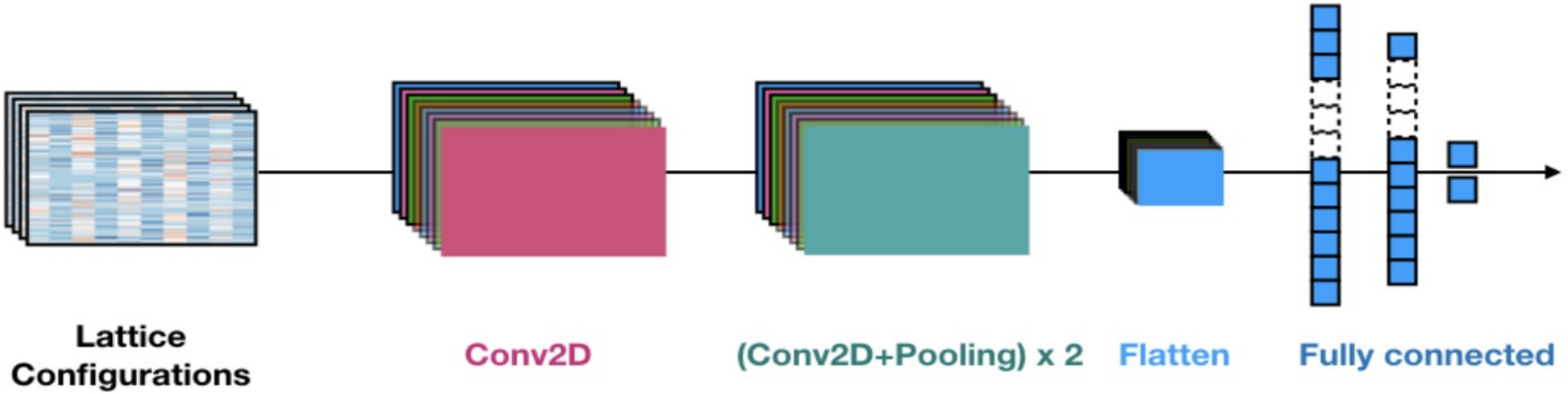
$$\mu = 0.91 \quad \text{with label} \quad y = (0, 1)$$

and

$$\mu = 1.05 \quad \text{with label} \quad y = (1, 0)$$



DCNN Architechture – Classification



Training set consist two ensembles of configs at

$$\mu = 0.91 \quad \text{with label} \quad y = (0, 1)$$

and

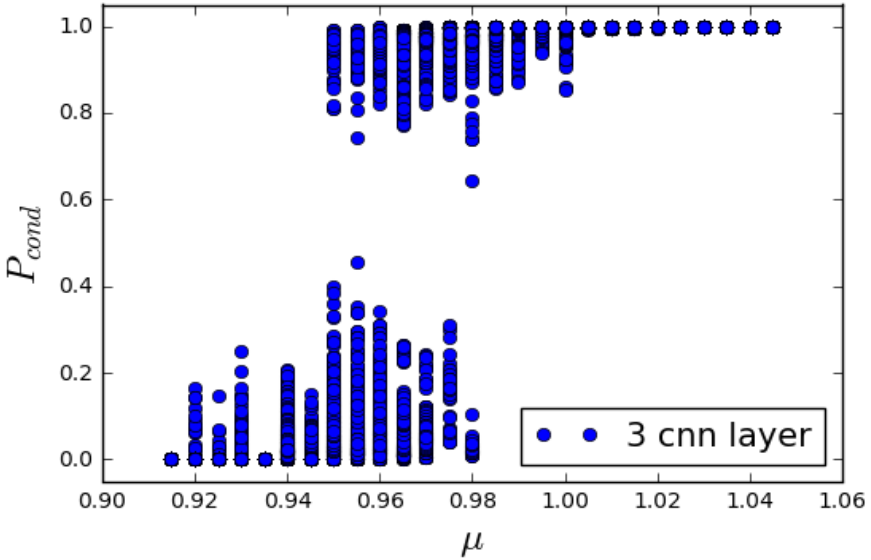
$$\mu = 1.05 \quad \text{with label} \quad y = (1, 0)$$

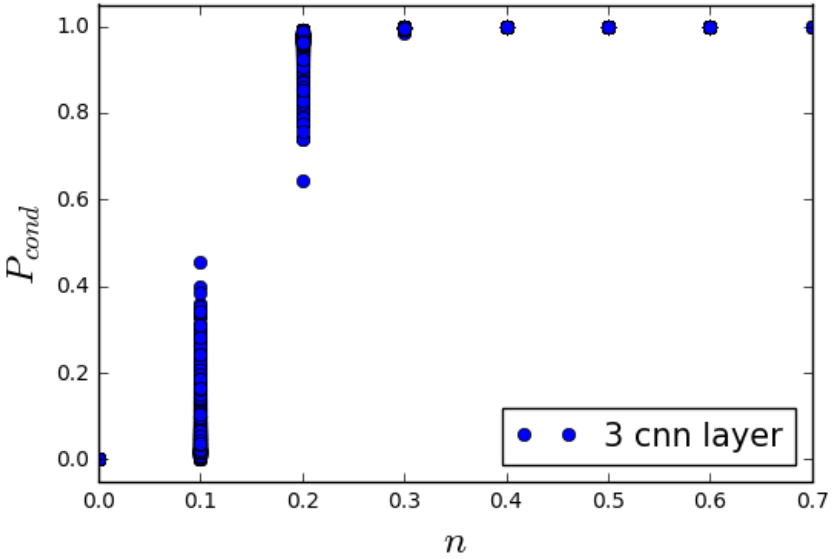
Testing set consist of

different ensembles of configurations at different chemical potential

$$0.91 < \mu < 1.05$$

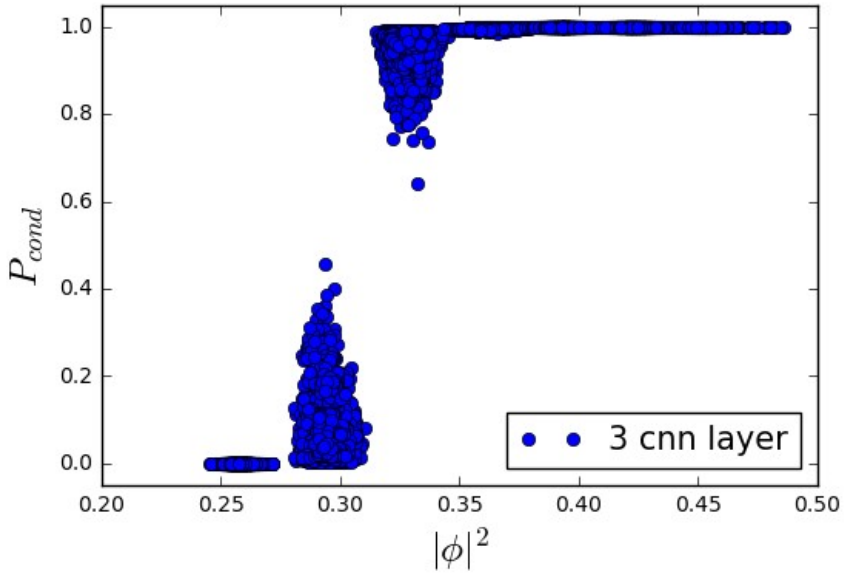
Condensation probability from DCNN





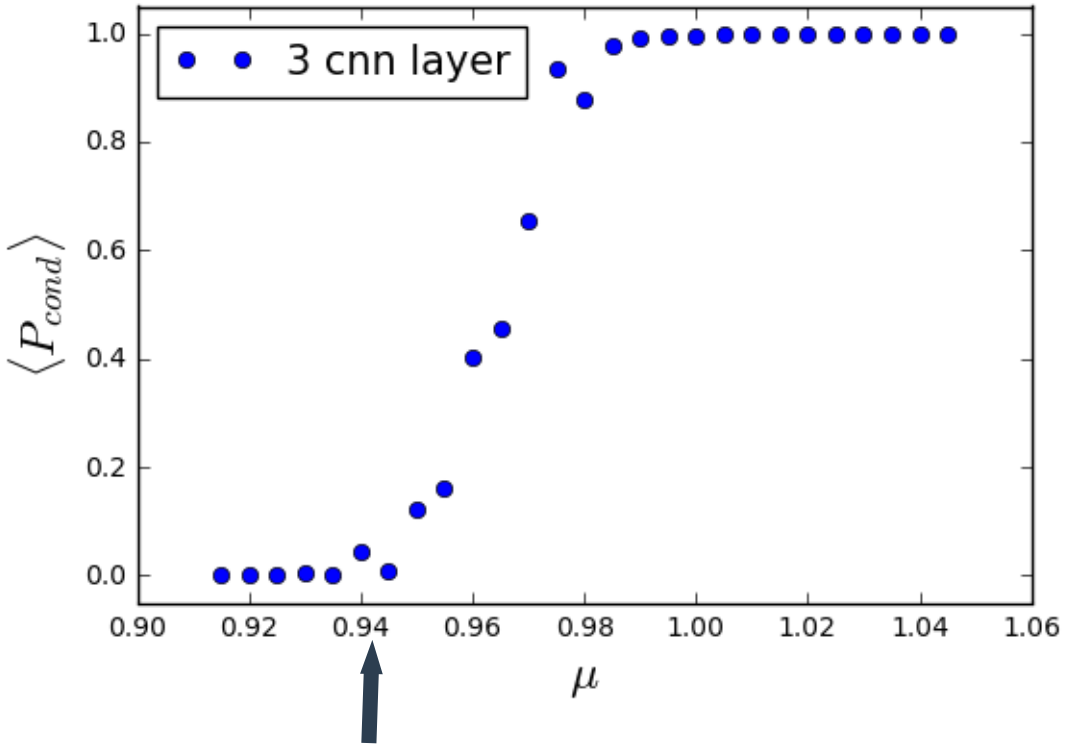
Strong correlation between
P_cond and observables :

n / squared field



Ensemble average cond-probability

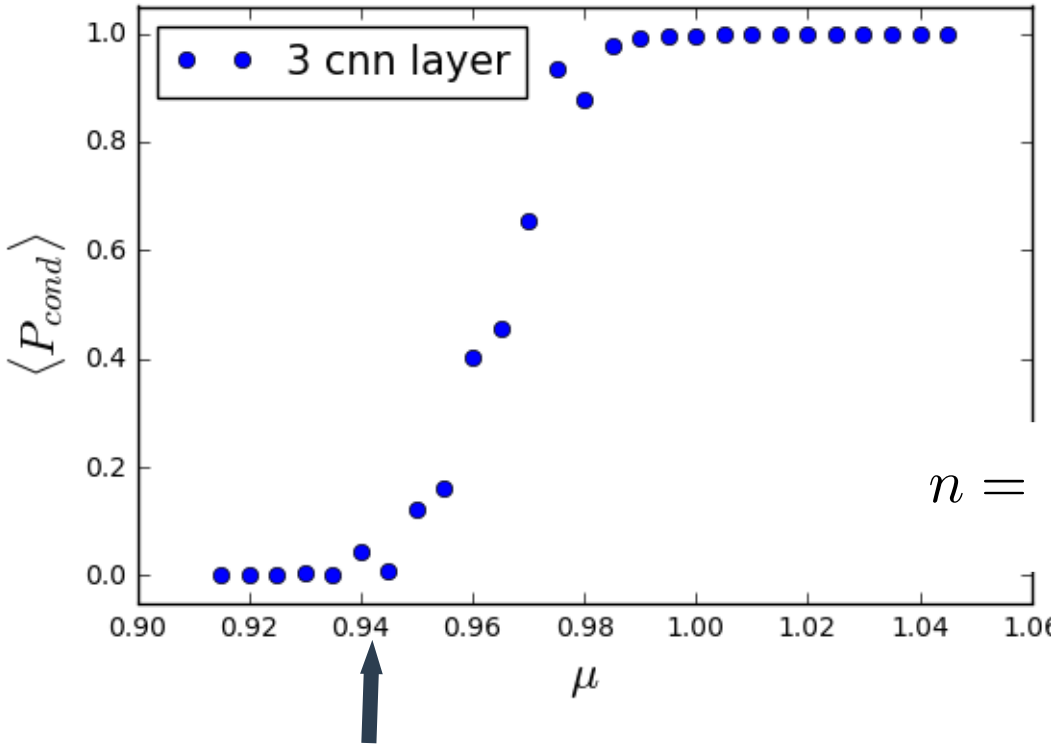
Classifier of the phases : $\langle n \rangle = 0 \quad \text{and} \quad \langle n \rangle \neq 0$



$$\mu_{th}(\langle P_{cond} \rangle > 0) \sim \mu_{th}(\langle n \rangle > 0)$$

Ensemble average cond-probability

Classifier of the phases : $\langle n \rangle = 0 \quad \text{and} \quad \langle n \rangle \neq 0$

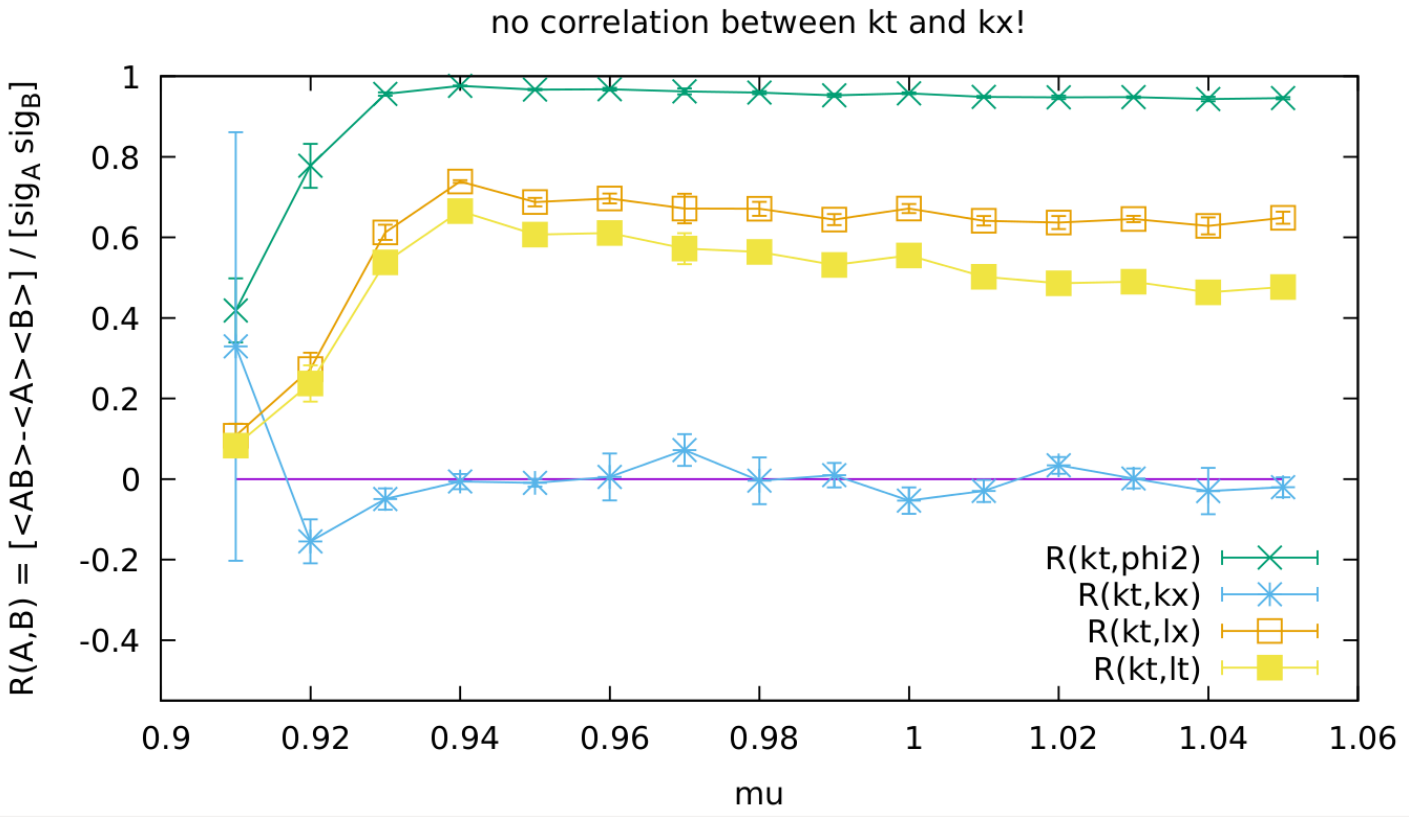


$$n = \frac{1}{N_x N_t a} \sum_n k_t(n)$$

$$\mu_{th}(\langle P_{cond} \rangle > 0) \sim \mu_{th}(\langle n \rangle > 0)$$

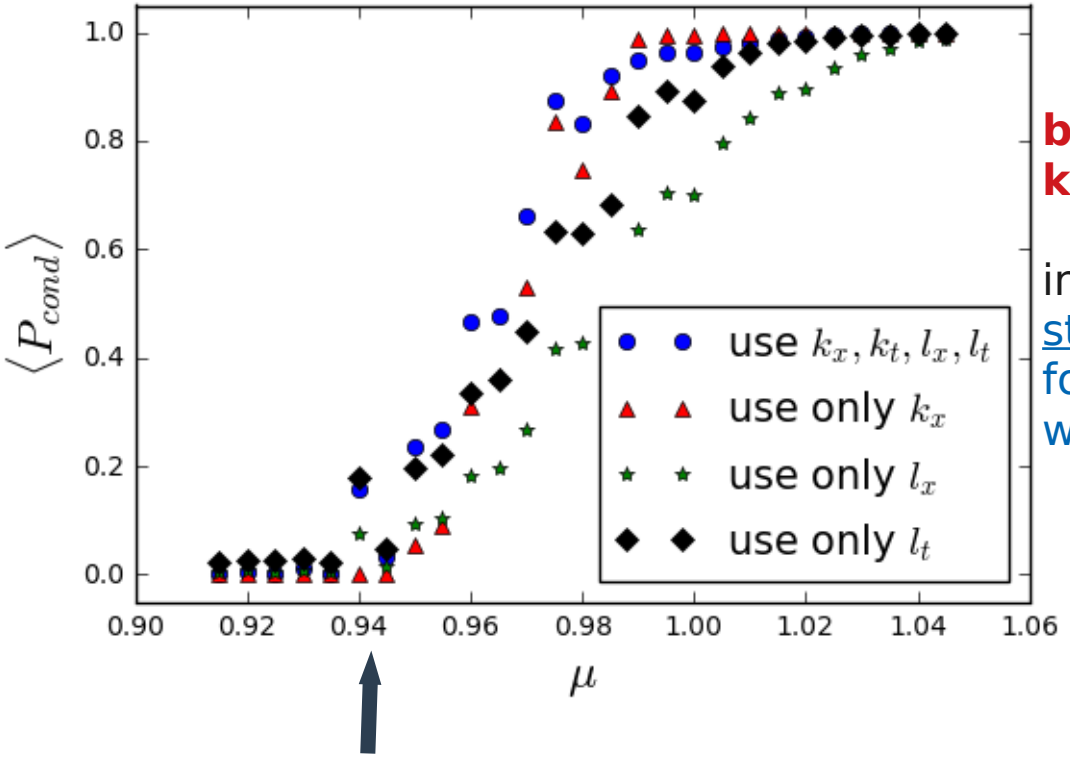
Discard kt information

There's finite correlation between kt and l variable, But, no correlation bet. kt and kx



Discard kt information

The same transition point, even use only kx!



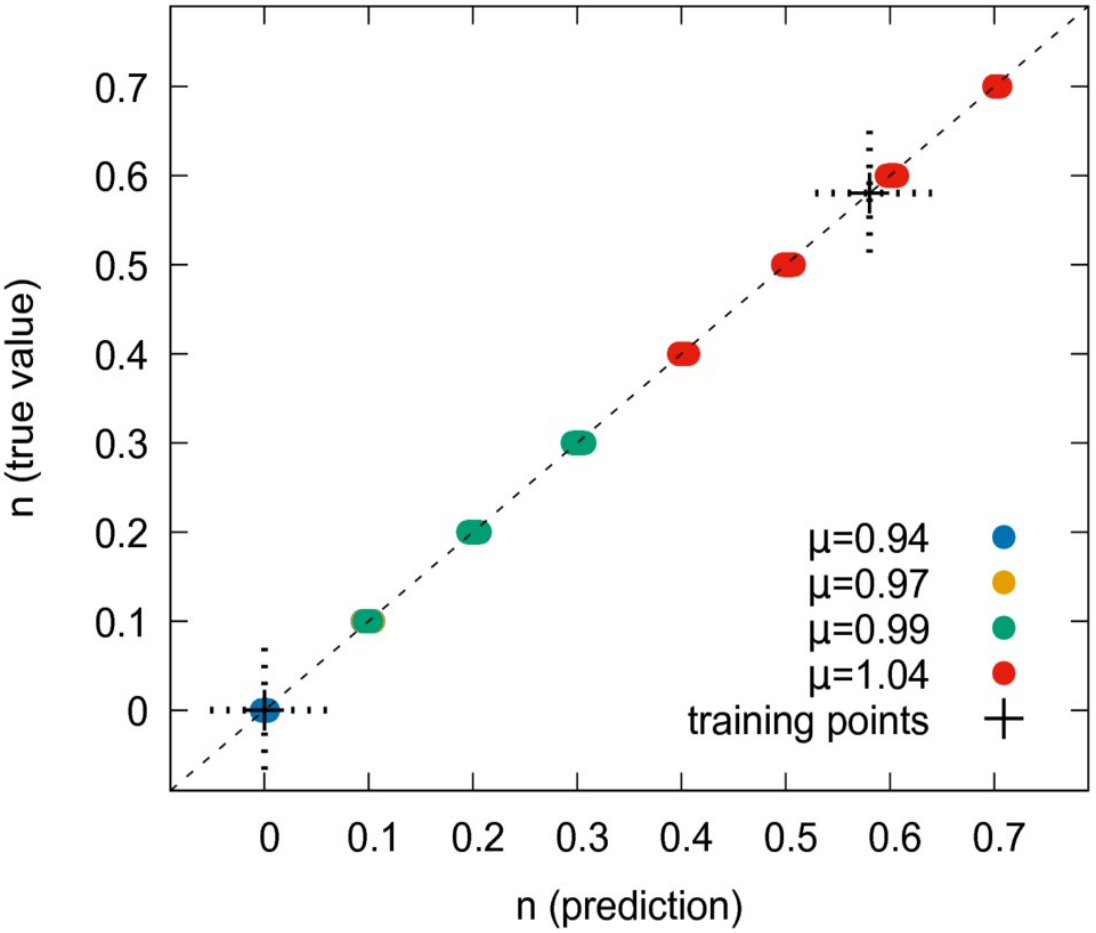
beyond conventional knowledge :

indicating hidden structures in kx variables for number density n, which is not only in kt.

$$\mu_{th}(\langle P_{cond} \rangle > 0) \sim \mu_{th}(\langle n \rangle > 0)$$

regression for particle density n

Note, for training, only used $\mu = 0.91$ and $\mu = 1.05$

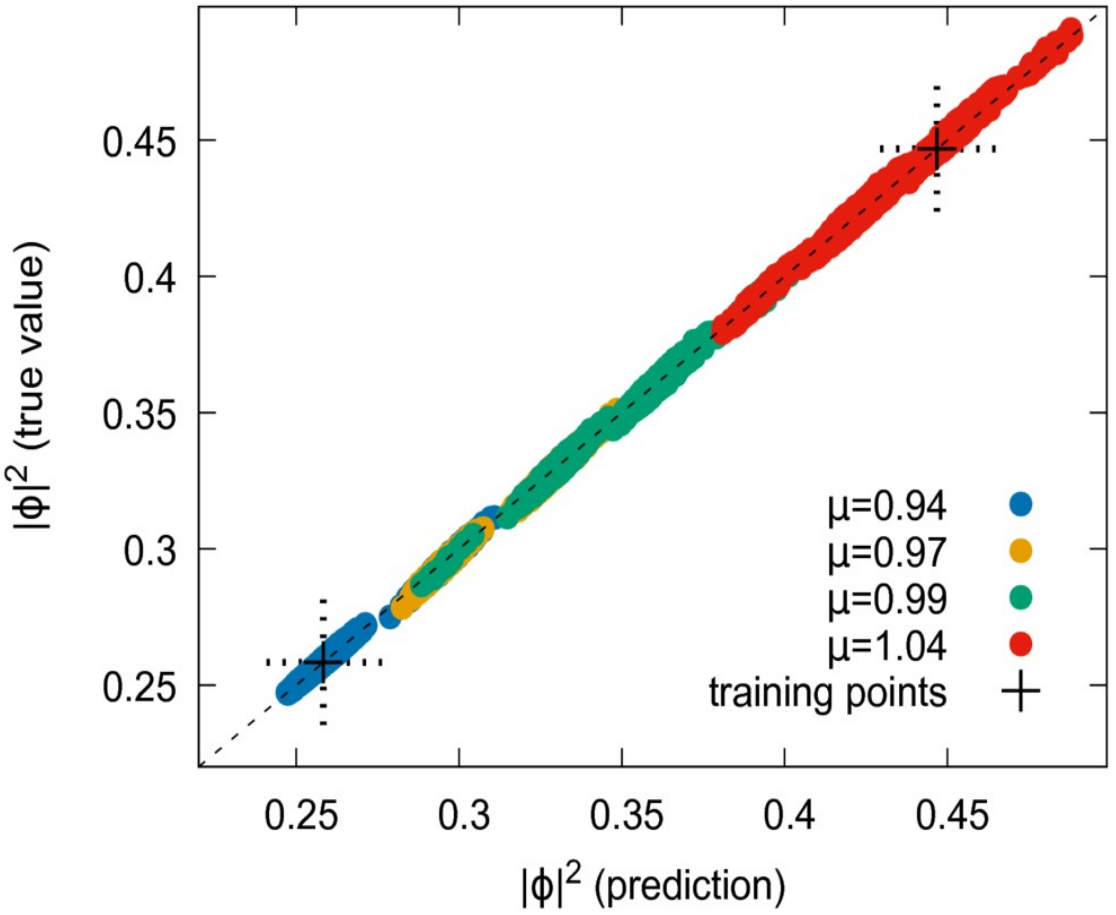


$$n = \frac{1}{N_x N_t a} \sum_n k_t(n)$$

$$RMSE < 0.003$$

regression for squared field $|\phi|^2$

Note, for training, only used $\mu = 0.91$ and $\mu = 1.05$



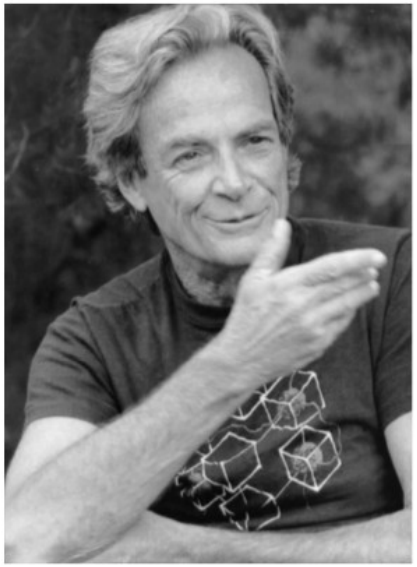
$$|\phi|^2 = \frac{1}{N_x N_t} \sum_n \frac{W[s(n)+2]}{W[s(n)]}$$

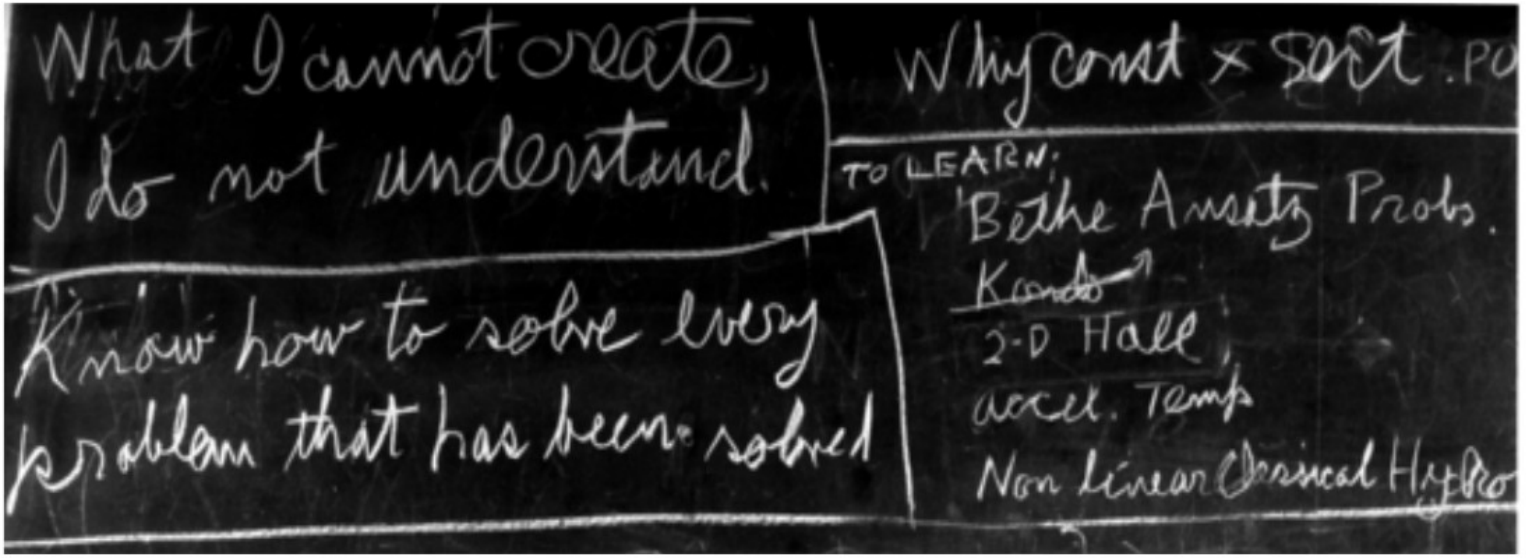
$$W[s(n)] = \int_0^\infty \mathrm{d}r\, r^{s(n)+1}\, e^{-(4+m^2)r^2-\lambda r^4}$$

$$s(n) = \sum_\nu [|k_\nu(n)| + |k_\nu(n-\hat{\nu})| + 2(\ell_\nu(n) + \ell_\nu(n-\hat{\nu})$$

$$RMSE < 0.005$$

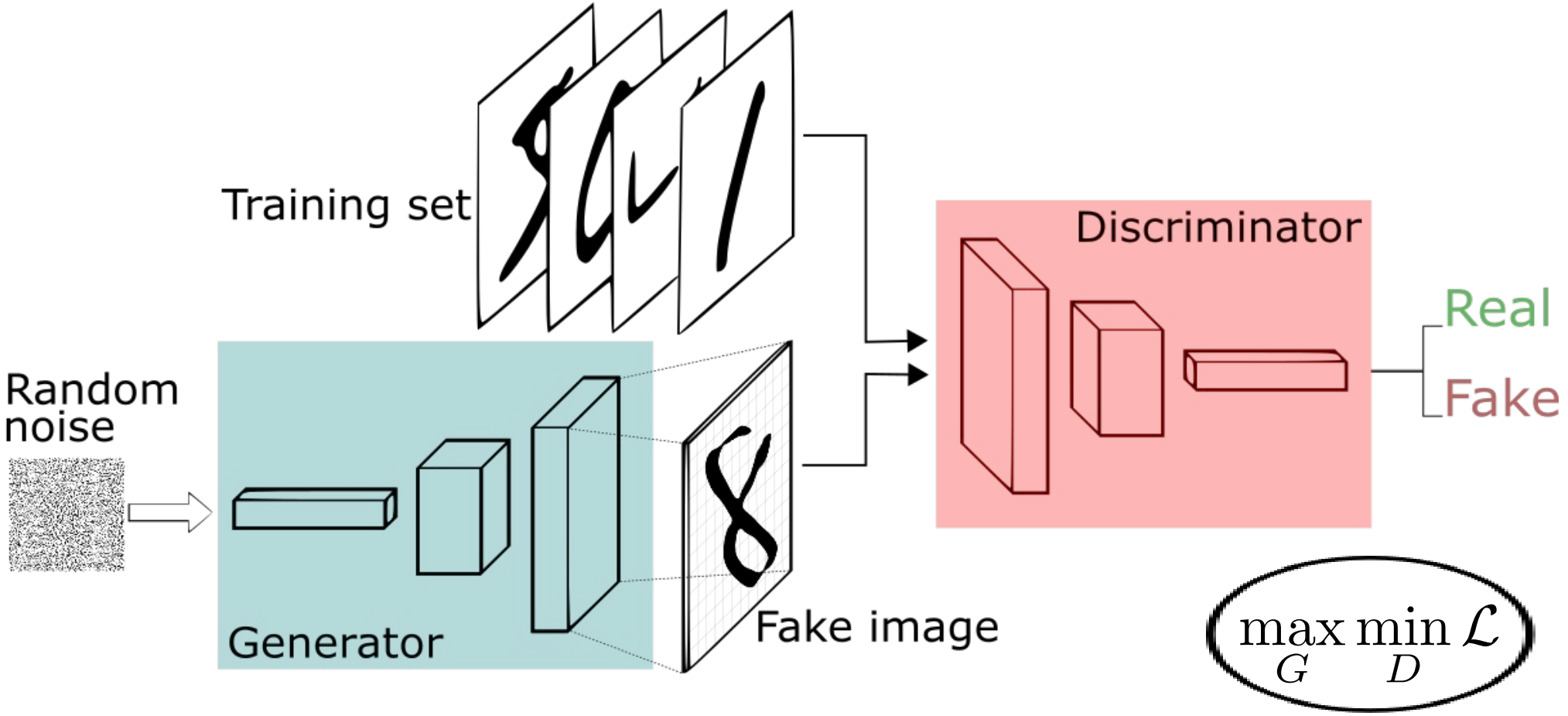
Generate new ones





"What I can not create, I do not understand"

Gnerative Adversarial Network



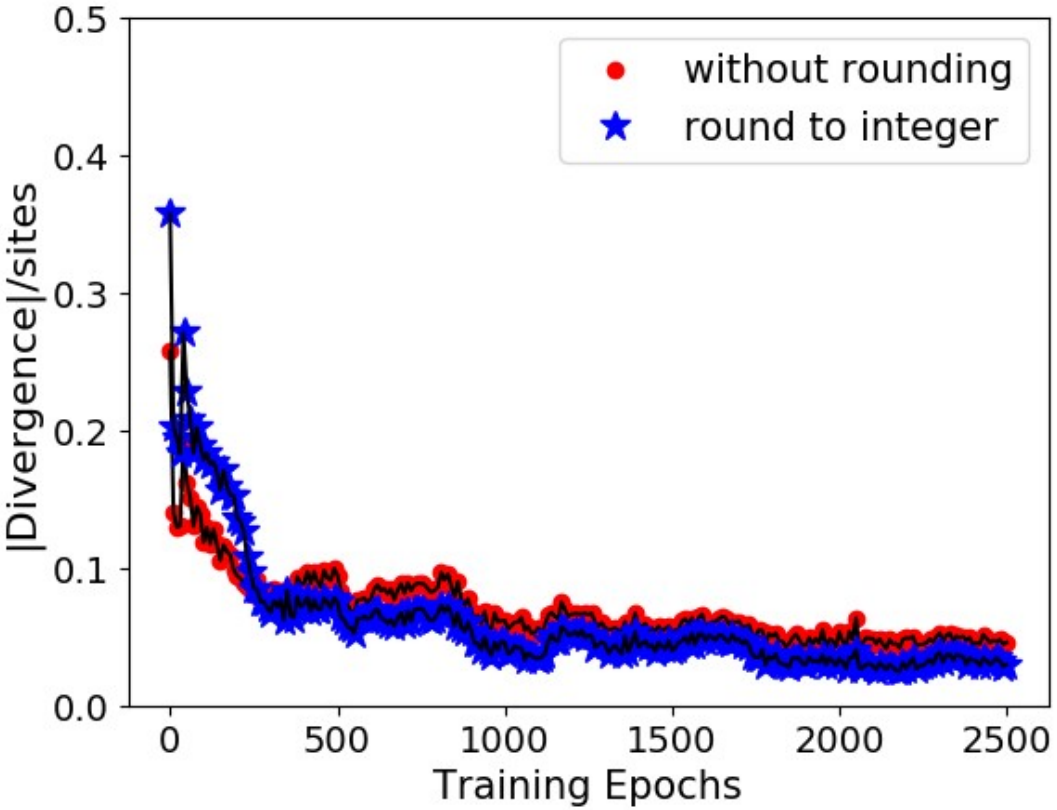
$$\mathcal{L} = -\mathbb{E}_{\hat{x}\sim p_r(\hat{x})}[\log(D(\hat{x}))] - \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]$$

GAN – generate proper configurations

The divergence condition get learned automatically:

'Physical' configs can be generated

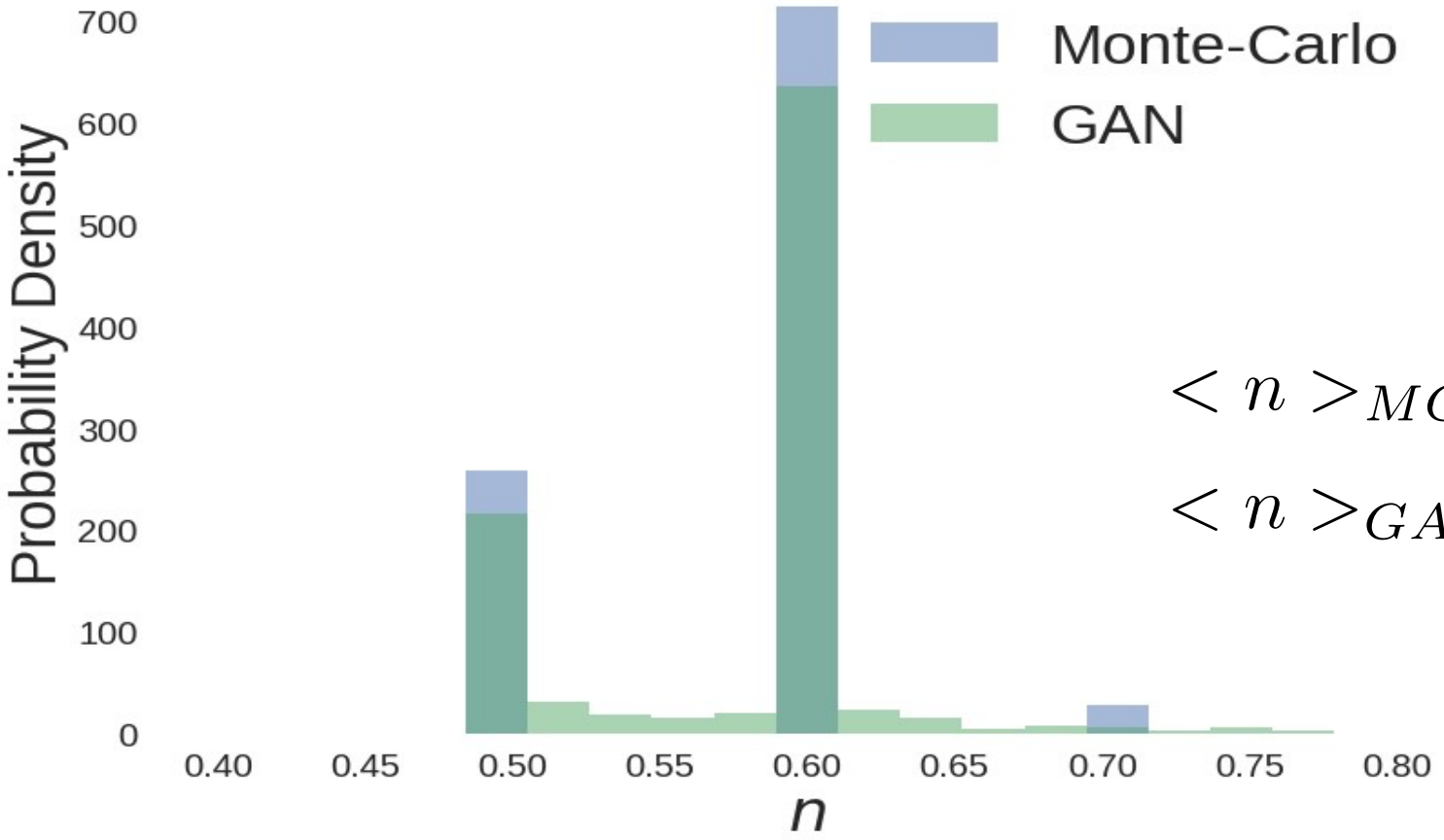
$$\nabla \cdot k(n) = \sum_{\nu} [k_\nu(n) - k_\nu(n - \hat{\nu})] = 0$$



Automatically capture the implicit physical constraint!

Fine tune GAN – distribution for n

Number density n distribution with 1k configs (after 6k trainig epochs):

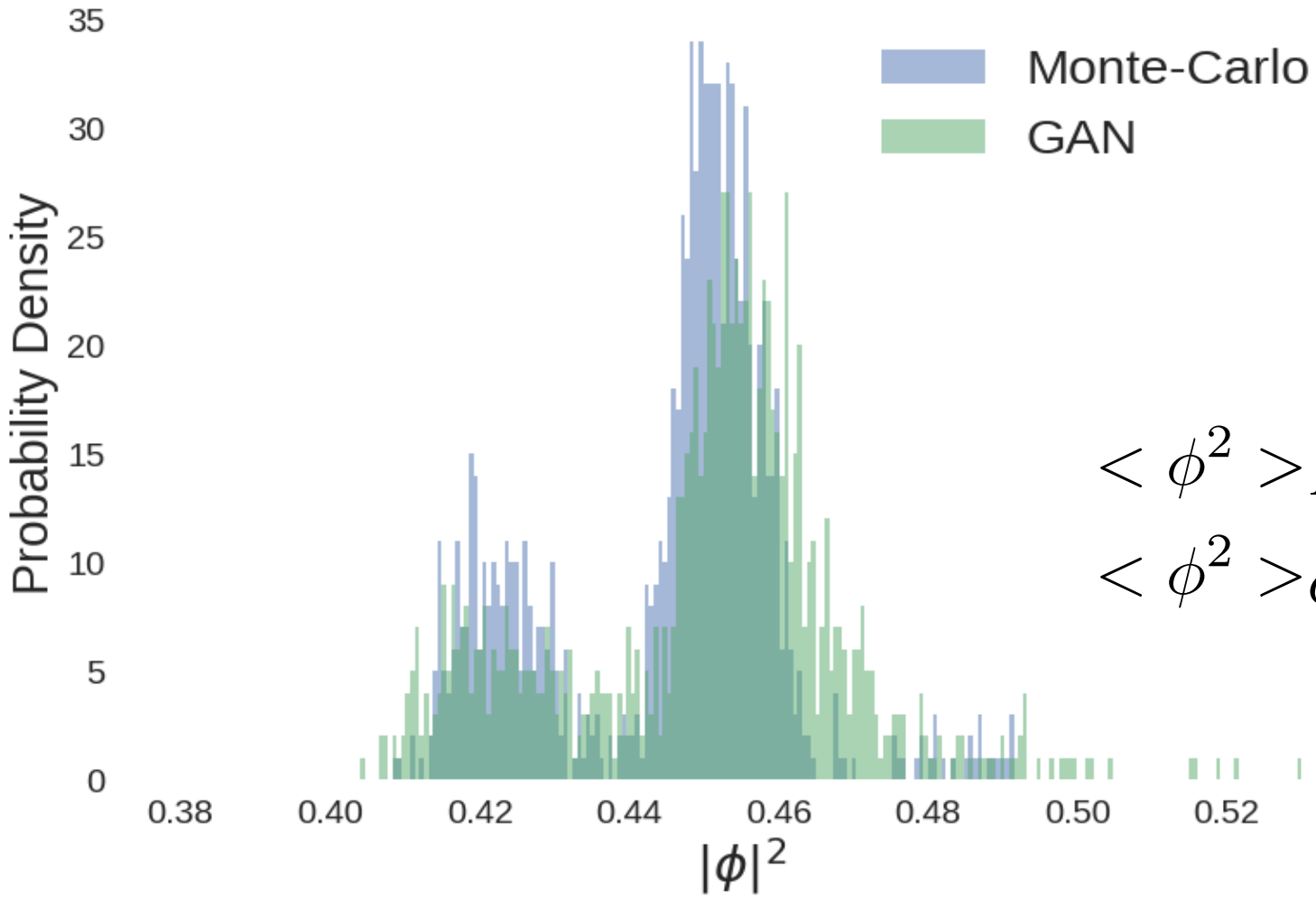


$$< n >_{MC} = 0.580$$

$$< n >_{GAN} = 0.578$$

Fine tune GAN – distribution for field $|\phi|^2$

Squared field distribution with 1k configs (after 6k training epochs):



$$< \phi^2 >_{MC} = 0.447$$

$$< \phi^2 >_{GAN} = 0.449$$

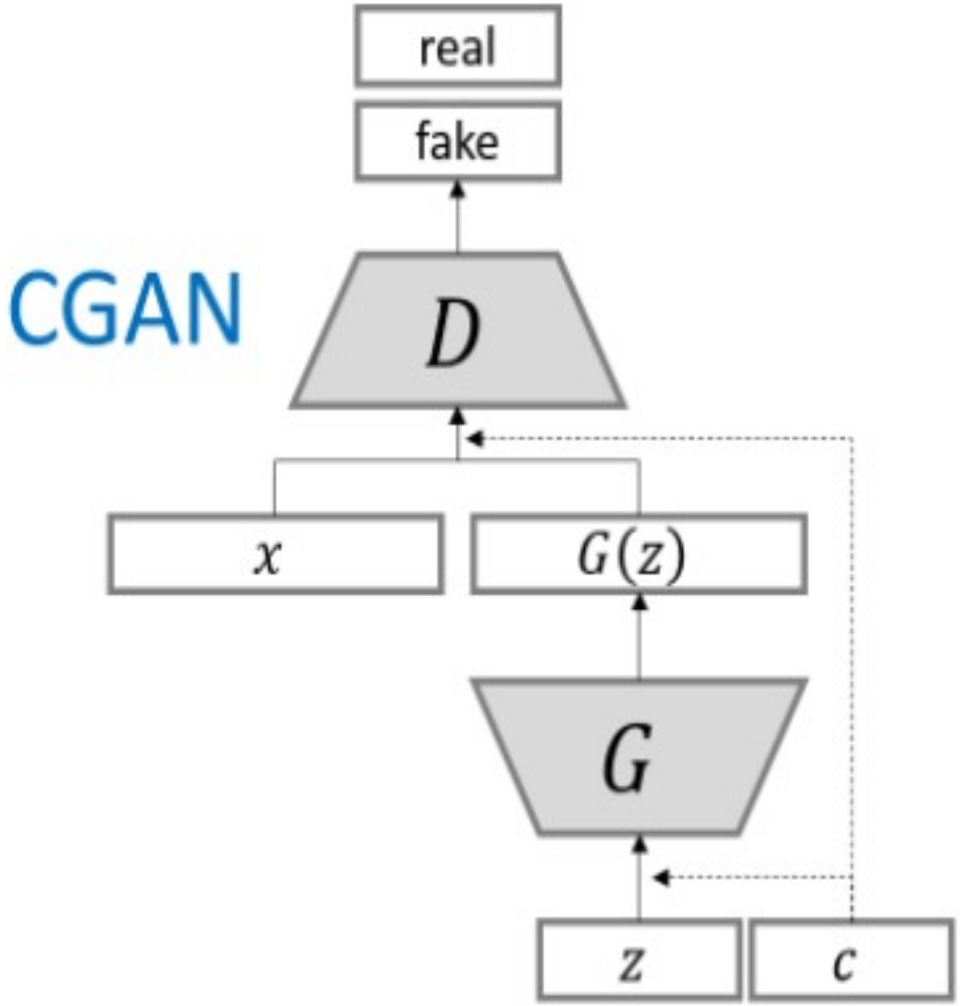
add conditional information of n

make GAN conditional on particle density n :

We train GAN using one esemble with $\mu = 1.05$ labeled as well with n (including n=0.4, 0.5, 0.6, 0.7),

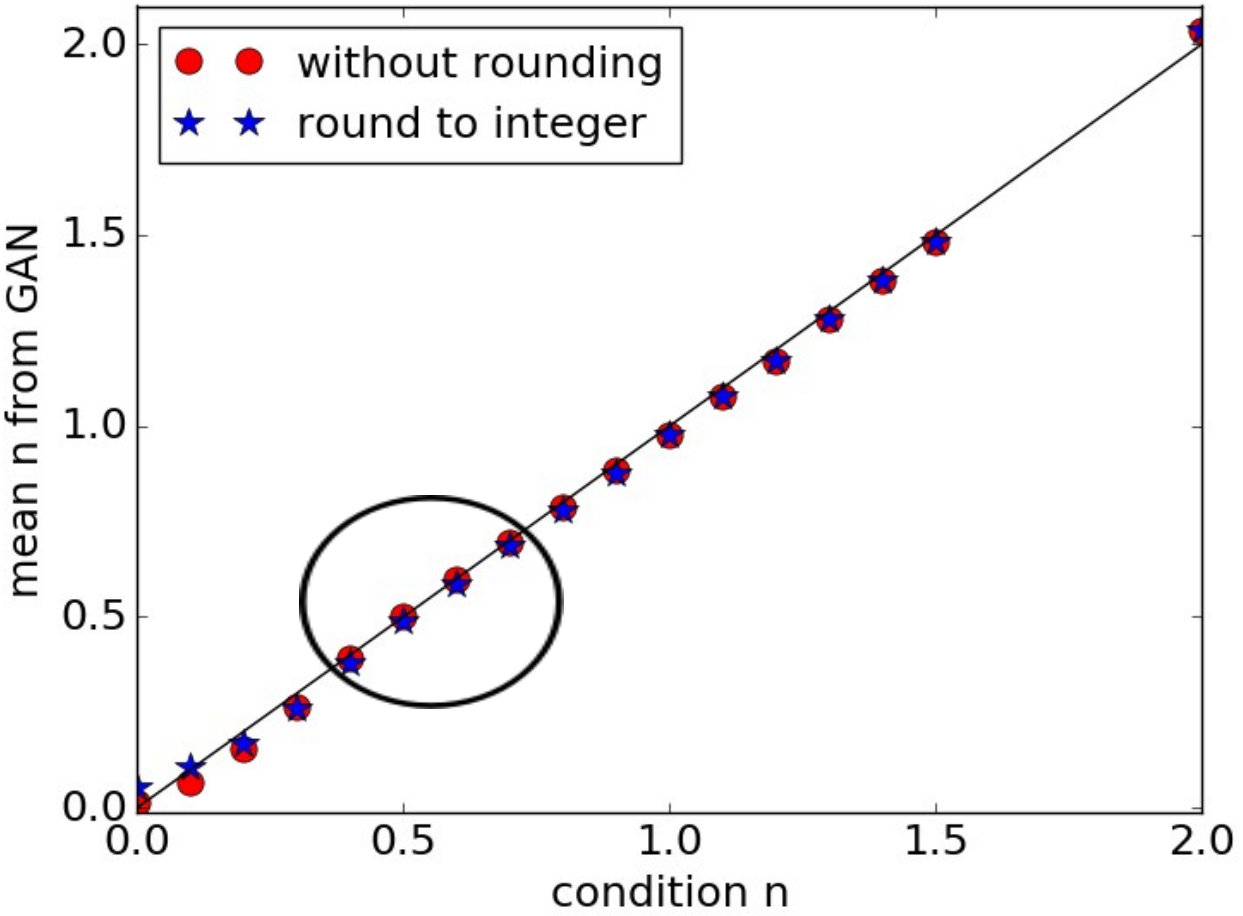
Once trained, in generating stage,

We specify different n values.



Conditional GAN – control n

mean value for n and squared field of generated ensemble is controlled by condition in c-GAN.



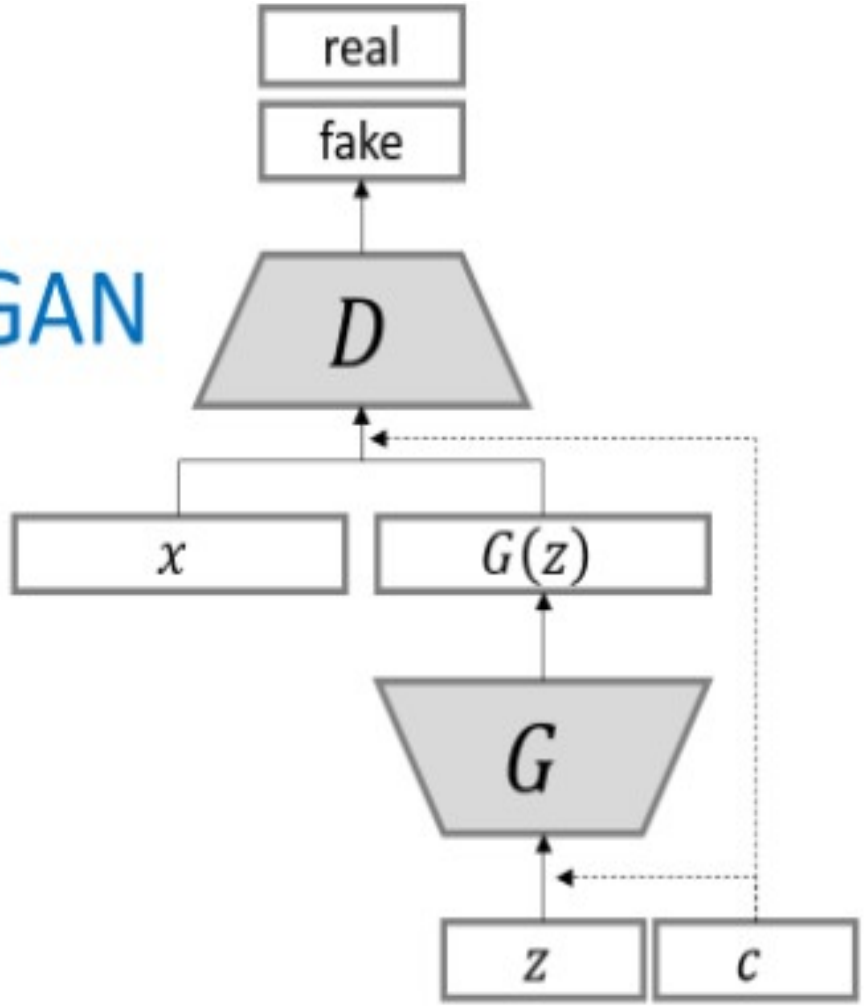
add conditional information of mu

make GAN conditional on chemical potential mu :

We train GAN using 3 ensembles with $\mu = 0.91, 0.98, 1.05$ labeled with corresponding mu for each configuration

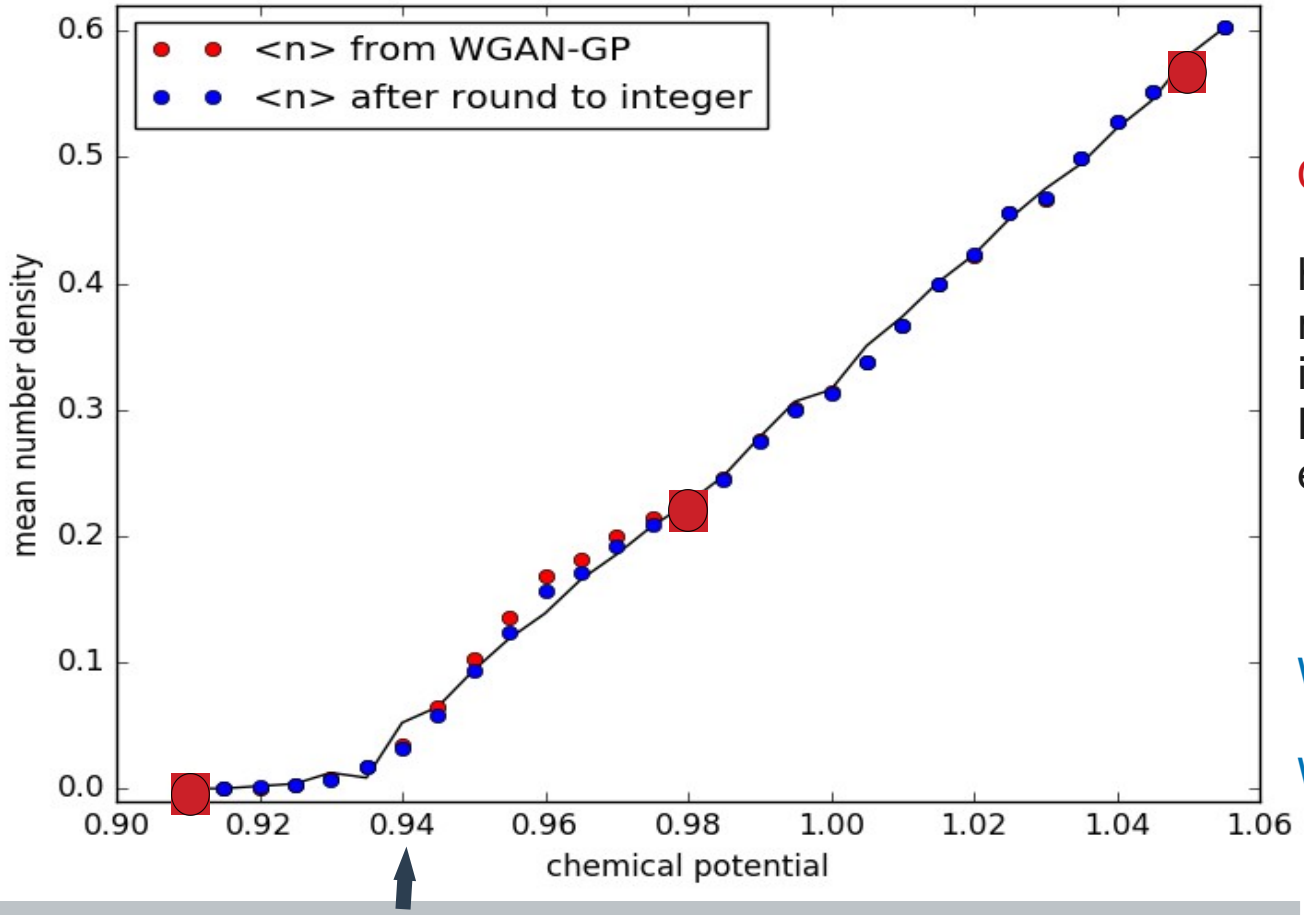
Once trained, in generating stage,

We specify different chemical potentials : generalize to different parameter range!



Conditional GAN – interpolate mu

phase diagram generated by c-GAN on mu with limited ensemble of training set:



Got the partition sum:

how much can we reconstruct the partition information from several Monte-Carlo generated ensembles of configs ?

Well phase diagram

Well transition point

Summary

(1) Classification : pin down phase 'transition' point

(identify order parameter)

(2) Regression : learn physical observable (**non-linear regression**)

(identify thermodynamics)

(3) GAN(generative): use **GAN** to **generate** new physical configuration

Capture/reproduce data distribution **(use for storage)**

limited grand canonical –-> canonical ensemble

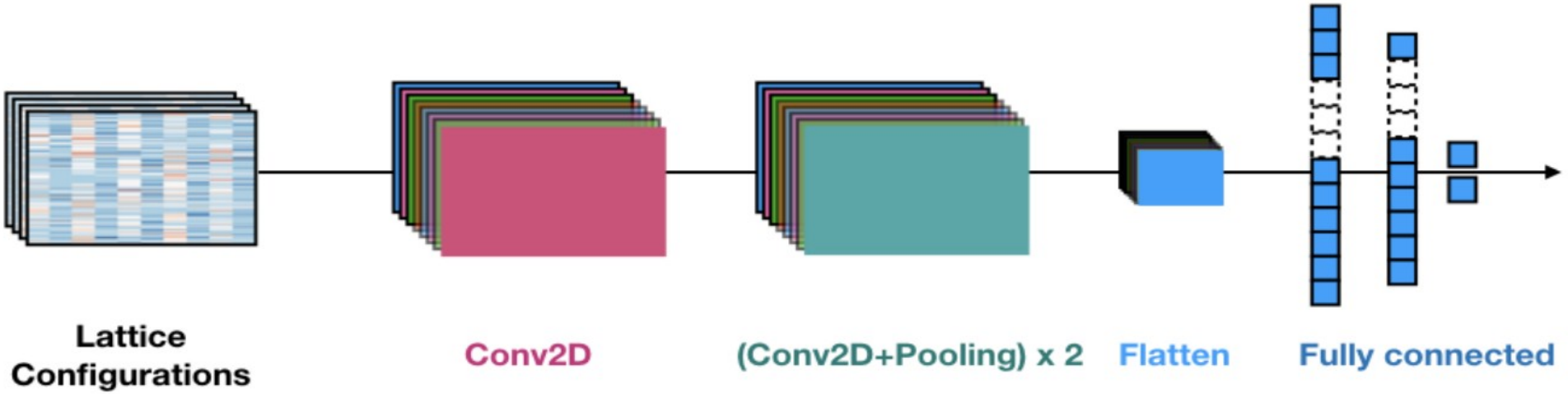
limited configs. –-> explore phase diagram

(identify partition function)

Needs more exploration !

Thanks!

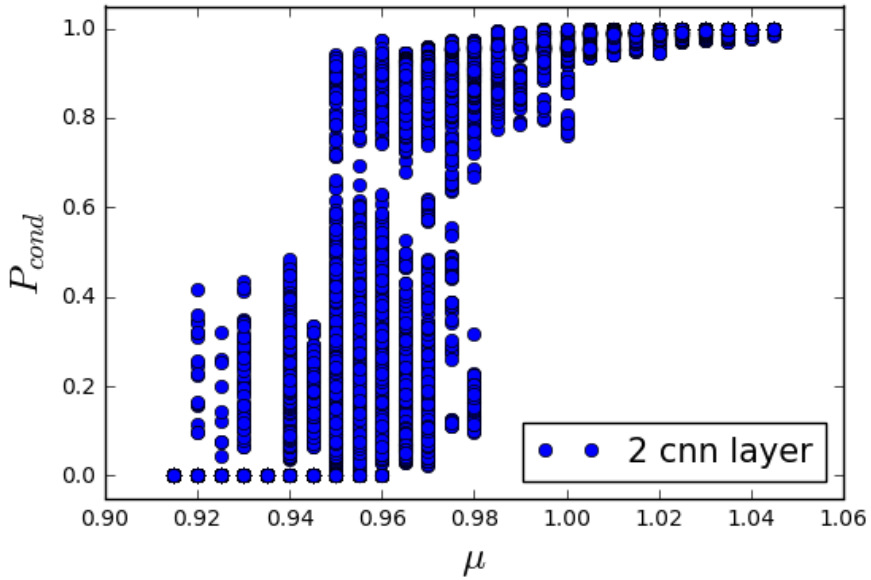
DCNN Architechture – Classification

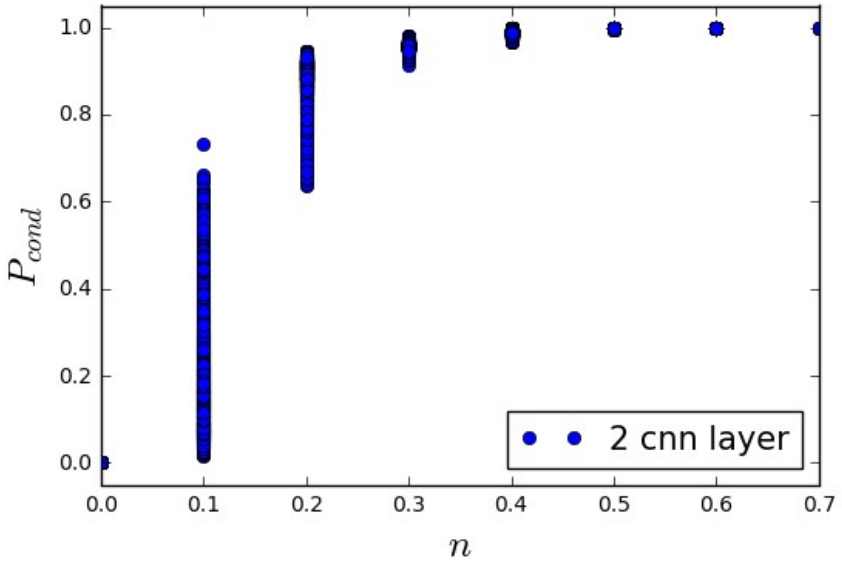


$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] + \lambda\|\theta\|_2^2$$

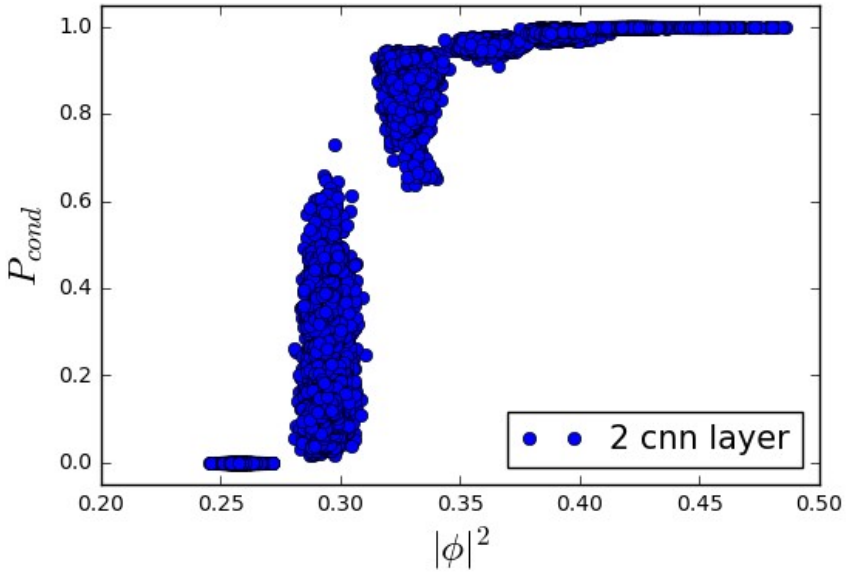
$\alpha_{lr} = 0.0001$ $AdaMax$ **optimization scheme**

Condensation probability from DCNN

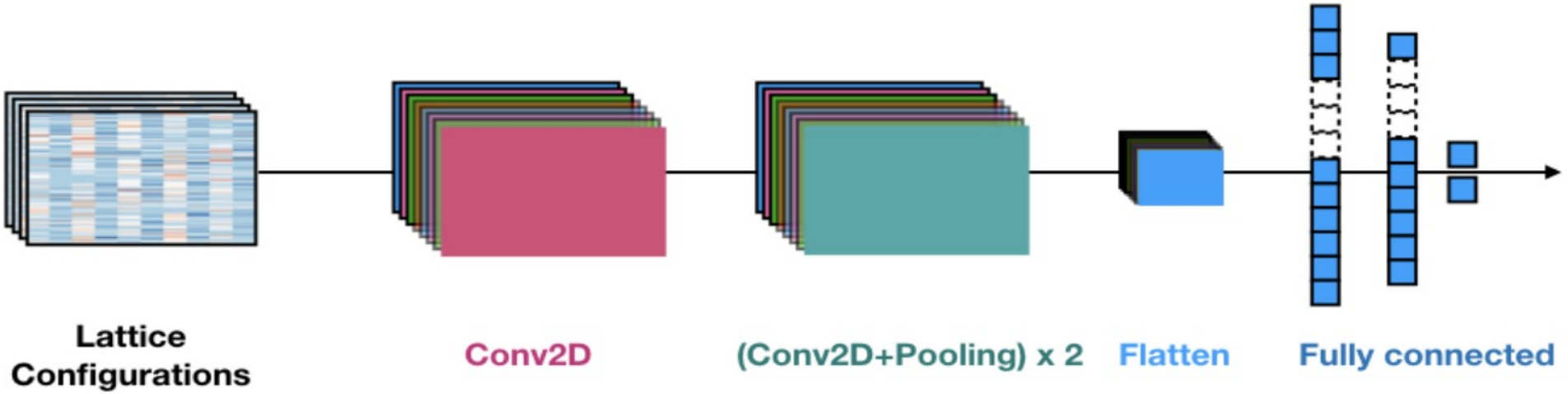




Deleting one CNN layer off gives *slightly worse* distinguish ability



DCNN Architechture – Regression



$$\mathcal{L} = -\frac{1}{2N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \lambda\|\theta\|_2^2$$

$$\text{Training set}: \quad \mu = 0.91 \quad \text{and} \quad \mu = 1.05$$

GAN – distribution

Zero-sum game – Nash equilibrium

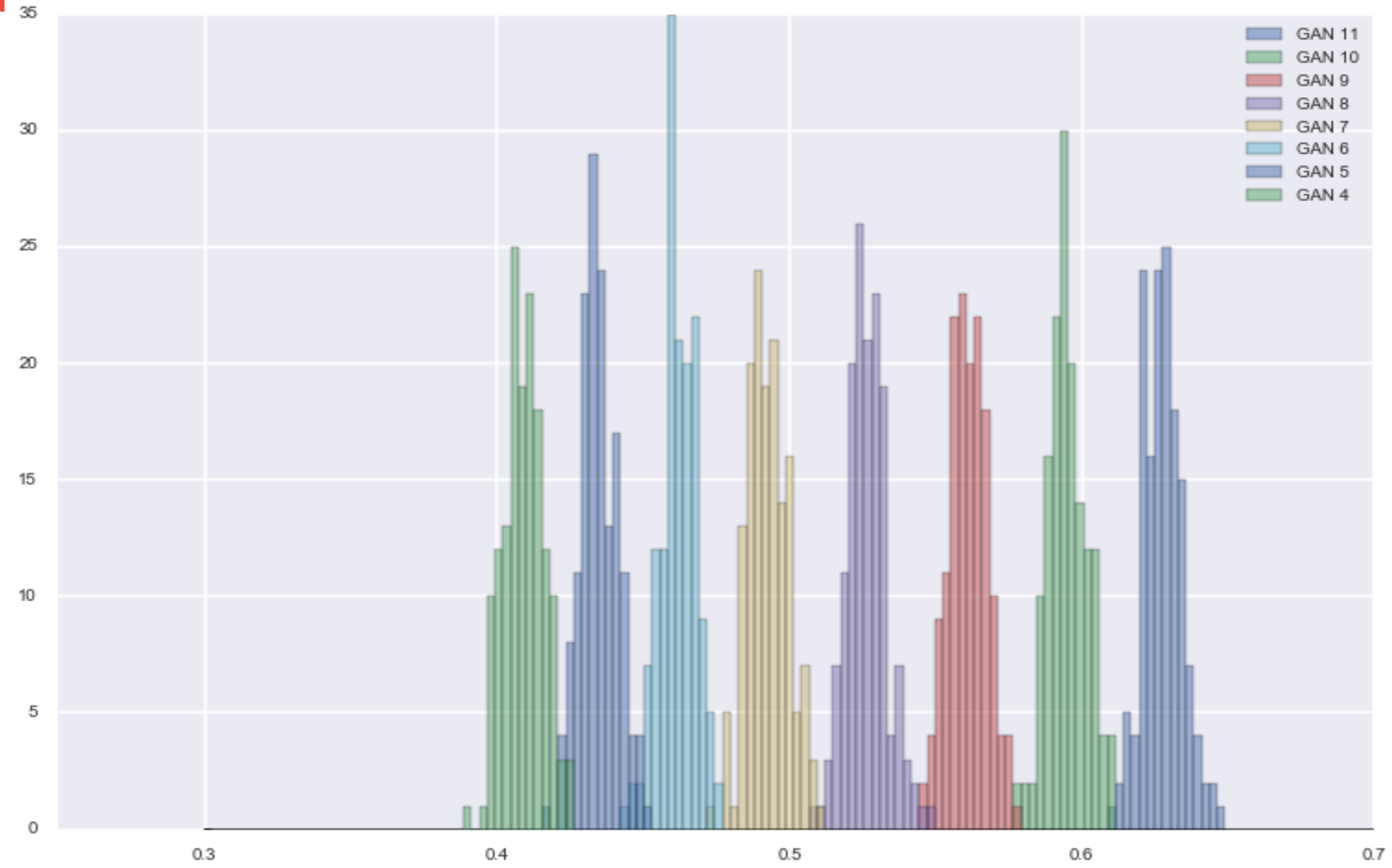
$$G^* = \arg\min_G \max_D (-\mathcal{L}_D(G, D))$$

$$\mathcal{L}_D = -\mathbb{E}_{\hat{x}\sim p_r(\hat{x})}[\log(D(\hat{x}))] - \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]$$

$$\mathcal{L}_G = \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]$$

$$D^*(\hat{x}) = \frac{p_r(\hat{x})}{p_r(\hat{x}) + p_g(\hat{x})}$$

GAN – distribution



GAN – distribution