Why Unsupervised Deep Networks Generalize

Based on Anita de Mello Koch, Ellen de Mello Koch and Robert de Mello Koch, "Why Unsupervised Deep Networks Generalize," arXiv:2012.03531[cs.LG].

Robert de Mello Koch

ECT* workshop: Machine Learning for High Energy Physics, on and off the Lattice, Sep 27 - Oct 1

Huzhou University, Zhejiang, China

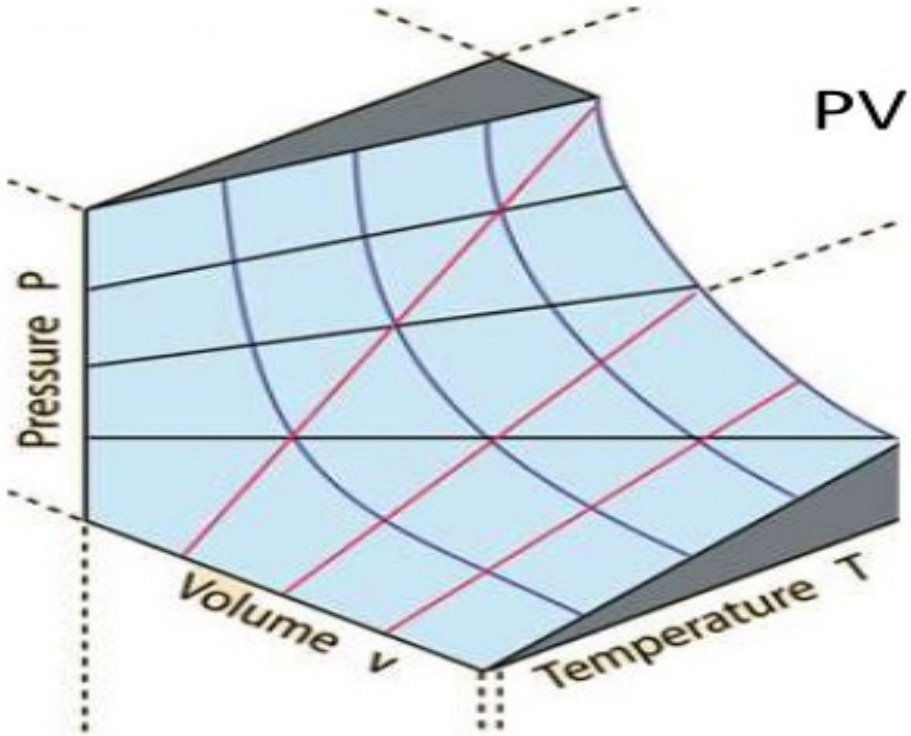
University of the Witwatersrand, Johannesburg, South Africa

October 1, 2021

Contents

What is machine learning?

Machine learning is basically learning patterns from data, i.e. curve fitting. By taking a collection of pressure and temperature measurements of gasses in different volumes we can fit a surface to the data and discover the ideal gas law.



What is machine learning? Supervised learning

Machine learning is basically learning patterns from data, i.e. curve fitting.

Data is a labeled list of numbers - the pixel values of each pixel in the image. The label for each list is Dog(=0) or Cat(=1).

Goal: train a machine to give 0/1 for an unseen image.


Dog


Dog


Dog


Cat


Cat

There is probably no elegant law for what it means to be a cat or a dog.

What is machine learning? Unsupervised learning

Machine learning is basically learning patterns from data, i.e. curve fitting.

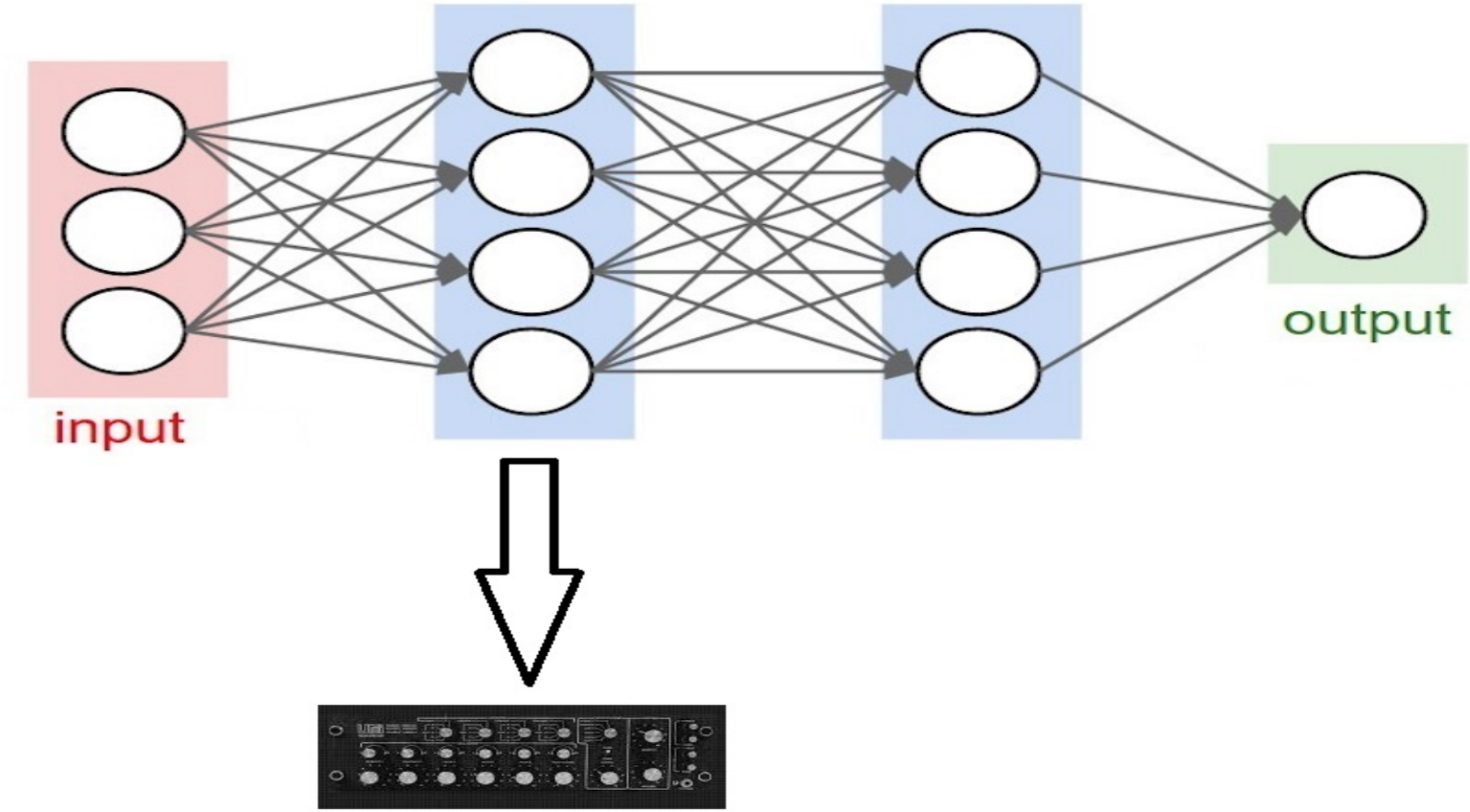
Data is an unlabeled a list of numbers - the pixel values of each pixel in an image.

Goal: train a machine to learn patterns in the data. (Construct a best guess for a probability distribution which describes the data set.)

With unsupervised learning you might try to

- Fill in missing patches of an image.
- Make new recommendations for a user, given their past history.
- Used for anomaly detection in forensic accounting (credit card fraud) or to detect hacking attempts.

Deep Learning



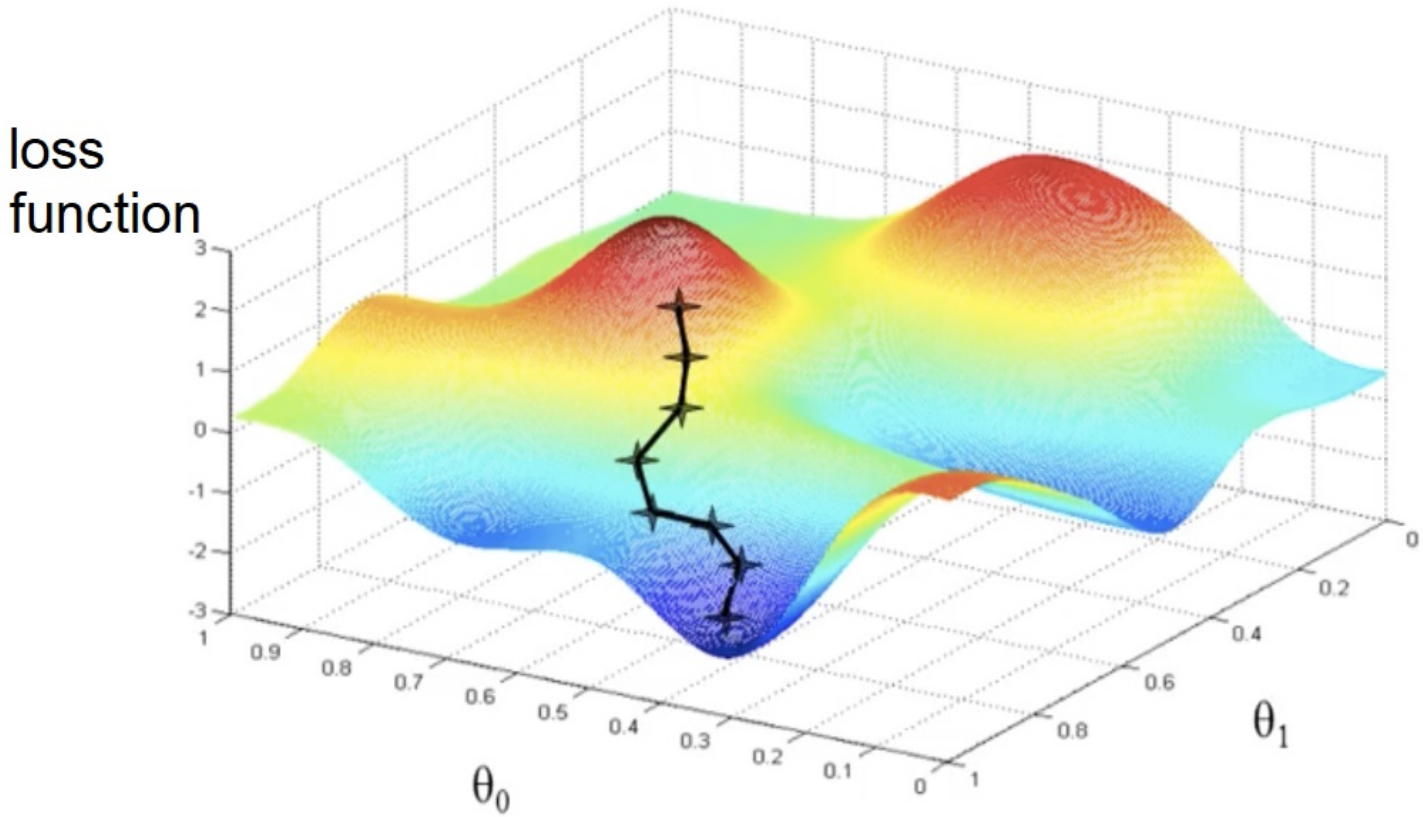
Each layer performs a transformation on its inputs to produce its outputs. Training means setting the dials so that the network classifies the data correctly. There is a "dial tuning algorithm".

The "dial tuning algorithm"

Training usually proceeds using gradient descent

$$\theta^{n+1} = \theta^n - \eta \nabla I(\theta)$$

to minimize some loss function $I(\theta) = \sum(\text{desired label - net output})^{2n}$.



Generalization

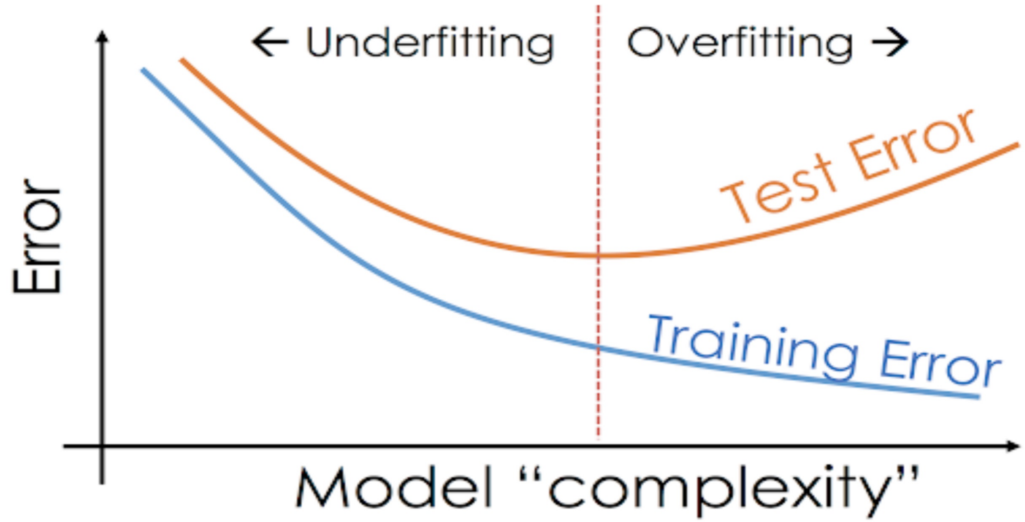
Train network on a training dataset. Training data sets may vary from thousands or tens of thousands to hundreds of thousands of data points.

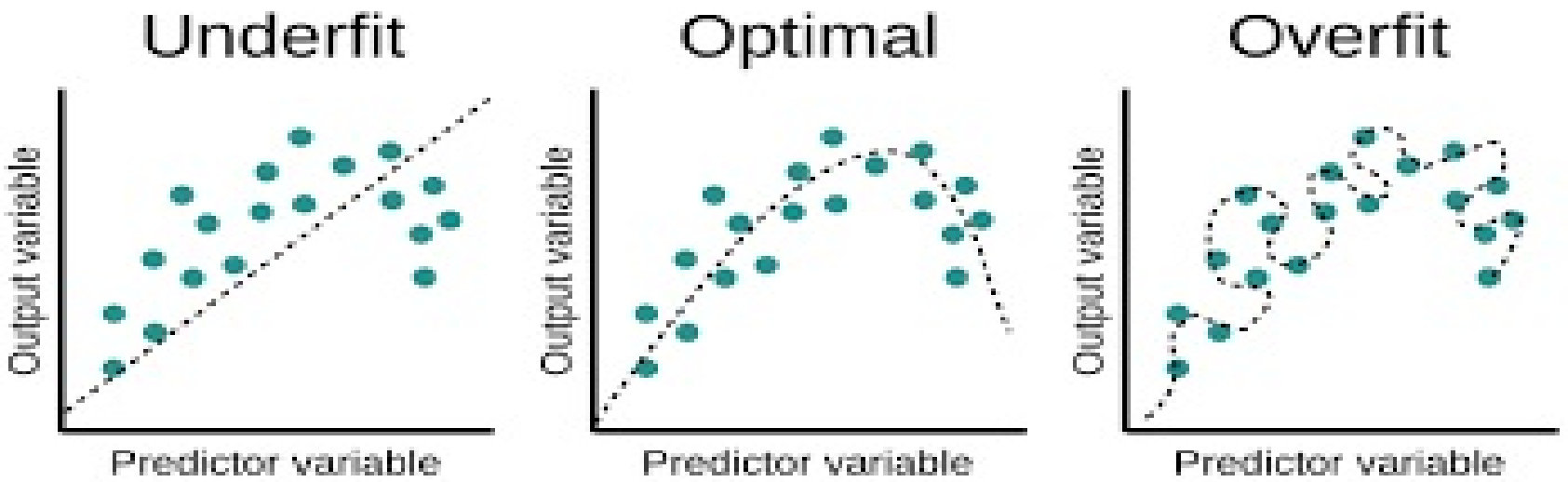
Training is performed in epochs. Train using entire data set = an epoch. Data set is reused for second and subsequent epochs. Training may take several thousand epochs. Training sets the weights in the network. Training is terminated when training error is no longer decreasing.

Important parameter specifying performance of a network is the *generalization error*, = difference in error achieved on training data and error achieved on unseen data. If generalization error is small, we say the network generalizes. An important question in machine learning is:

When do we expect networks to generalize?

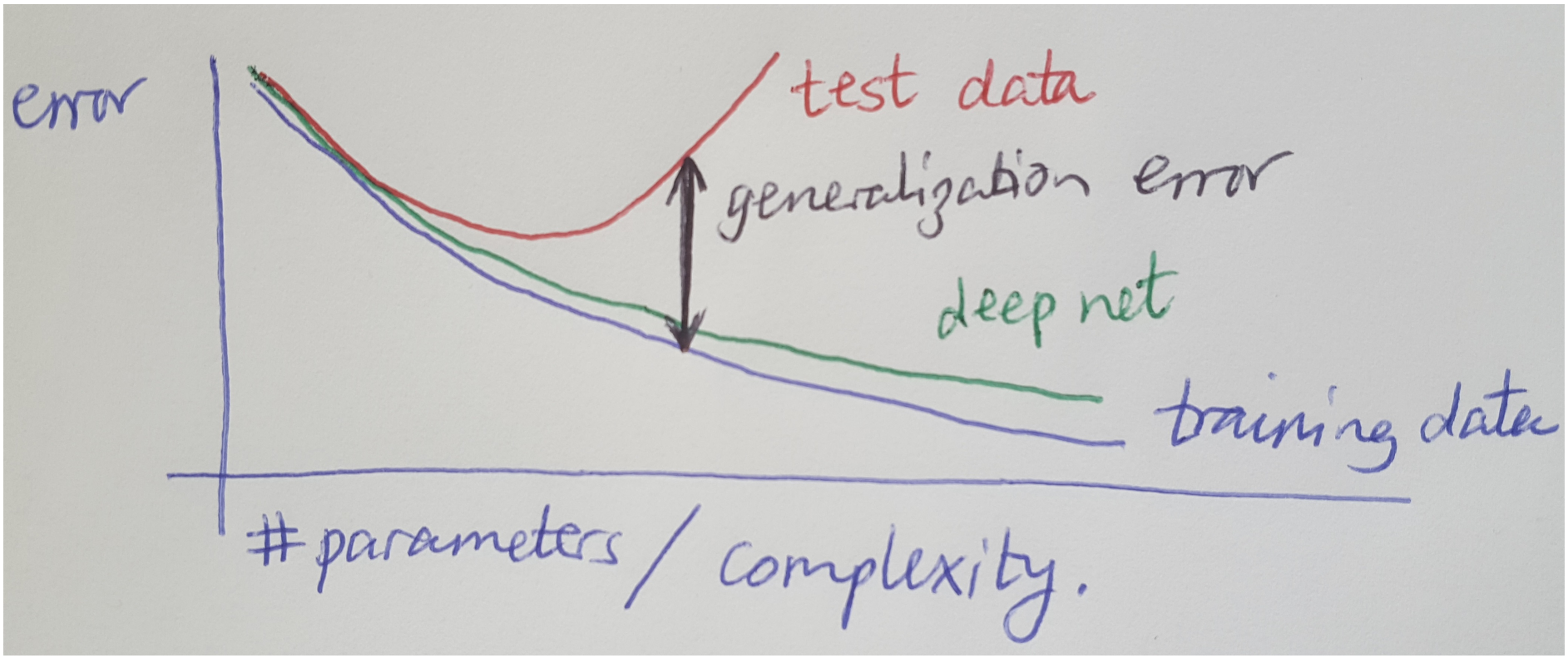
Intuition from curve fitting





The generalization puzzle: Why do deep nets generalize?

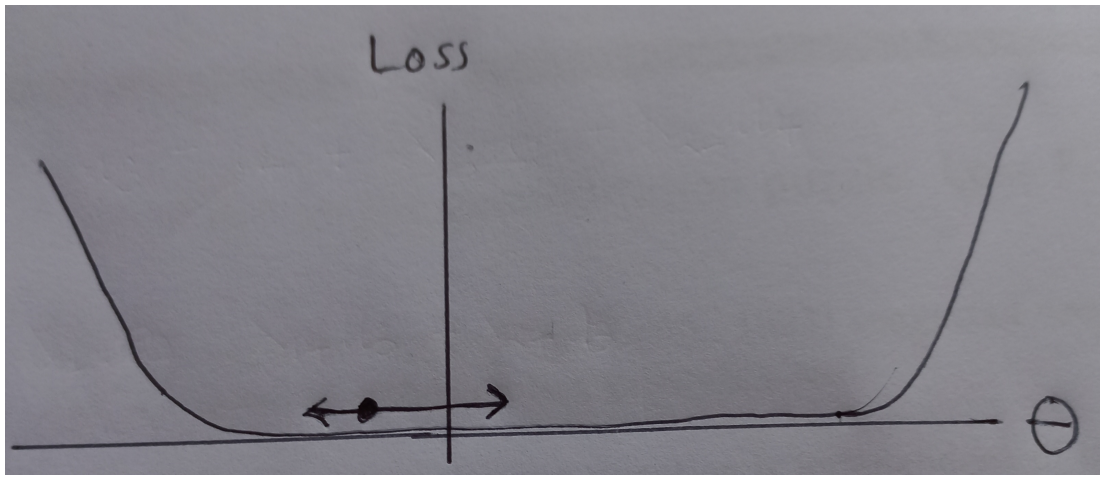
For deep nets typically $\sim$ 20 000 000 parameters and $\sim$ 50 000 samples. **Deep networks should not generalize!** But they do! Why don't we over fit the data?



Approaches to the generalization puzzle: Loss Function

Early attempts (Hinton and Van Camp, 1993) suggested deep networks with flat minima in training loss generalize well.

Intuitively: a potential flat in a direction is independent of that coordinate.



Approaches to the generalization puzzle: Loss Function

Early attempts (Hinton and Van Camp, 1993) suggested deep networks with flat minima in training loss generalize well.

Intuitively: a potential flat in a direction is independent of that coordinate.

Hope: Flat potentials may have so few effective network parameters that the generalization puzzle is solved.

Numerical studies $\Rightarrow$ sharp minima lead to higher generalization errors.

"Flatness" can be quantified by stability of network's output, under addition of noise to network's parameters (Langford and Caruana, 2002).

Resulting bounds do not resolve the puzzle.

Approaches to generalization puzzle: Weight Matrix

Another notion of noise stability: network's ability to reject noise injected at input or internal nodes of the network (Morcos et. al. 2018).

Using this (Arora et. al. 2018) proved bounds closer to solving the puzzle.

Consider fully connected layer (each input neuron is connected to each output neuron). SVD of trained weight matrix shows many singular values are small, and can be set to zero.

Large number of parameters reduces to smaller number of effective parameters. (Reduces number of parameters by a factor ~ 10.)

But this reduction is still nowhere near enough!

Our approach to generalization puzzle

The basic observation of (Arora et. al. 2018) is that a small number of singular values of the trained weight matrix are non-zero.

Our contribution: Consider the singular vectors. Using **the correct basis** these too have very few non-zero components.

Further reduction in the number of parameters by factor ~ 40. Total reduction is by a factor ~ 400 and this resolves the generalization puzzle.

Also we learn what the network is doing: it is doing one step of RG.

Some details

Our study used unsupervised networks called Restricted Bolztmann Machines (RBMs). We will review what an RBM is and how it is trained.

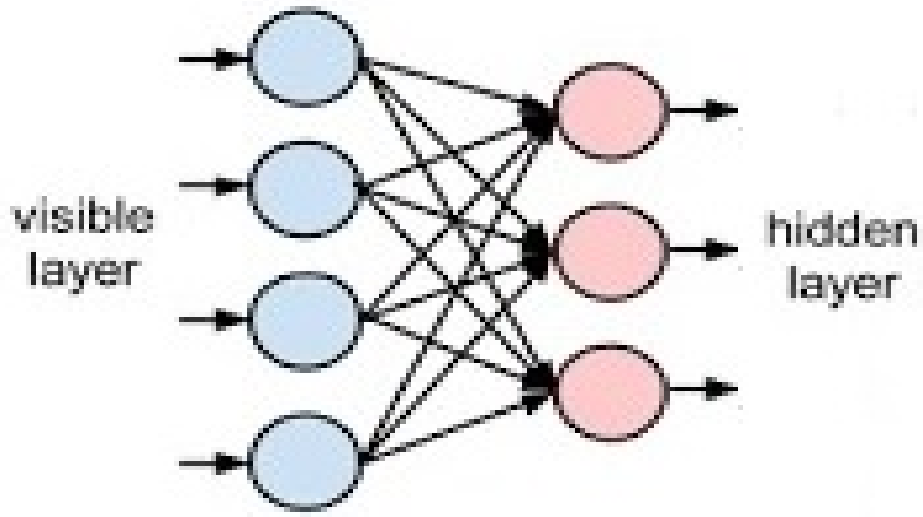
First example studied was RBM trained on Ising data. (Iso, Shiba and Yokoo, 2018; Funai and Giataganas, 2020). This exhibits (i) a close (and novel) connection to RG and (ii) a dramatic reduction in parameters.

Ising data set is special. Is connection to RG model independent or not?

Using standard image data sets from computer vision, we exhibit (i) close (and novel) connection to RG and (ii) dramatic reduction in parameters.

Provides clear resolution to generaliation puzzle for this class of networks.

Restricted Boltzman Machine



Typical application: inputs = 80×80 lattice state. outputs = 40×40 lattice state. 10 240 000 weights, 6400 input biases and 1600 output biases.

Training data set has 40 000 samples. Training takes $\sim$ 6000 epochs. (Takes about 6 hours on a GPU.)

Number of parameters = 256× number of data samples

Restricted Boltzman Machines

$$E = -\sum_a b_a^{(h)} h_a - \sum_{i,a} v_i W_{ia} h_a - \sum_i b_i^{(v)} v_i$$

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}} e^{-E} \qquad \mathcal{Z} = \sum_{\{\mathbf{v},\mathbf{h}\}} e^{-E}$$

Training minimizes relative entropy ($D_{KL}(q||p)$). Gradients used to update are: (greedy layer wise trained)

$$\frac{\partial D_{KL}(q||p)}{\partial W_{ia}} = \langle v_i h_a \rangle_{data} - \langle v_i h_a \rangle_{model}$$

$$\frac{\partial D_{KL}(q||p)}{\partial b_i^{(v)}} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model}$$

$$\frac{\partial D_{KL}(q||p)}{\partial b_a^{(h)}} = \langle h_a \rangle_{data} - \langle h_a \rangle_{model}$$

Restricted Boltzman Machines

Given visible vectors $\hat{v}$ sample the hidden vectors $\hat{h}$ using

$$p(\hat{h}_a = 1|\hat{v}) = \frac{1}{2}\left(1 + \tanh\left(\sum_i W_{ia}\hat{v}_i + b_a^{(h)}\right)\right) \tag{1}$$

Given $\hat{h}$ sample $\tilde{v}$ using

$$p(\tilde{v}_i = 1|\hat{h}) = \frac{1}{2}\left(1 + \tanh\left(\sum_a W_{ia}\hat{h}_a + b_i^{(v)}\right)\right) \tag{2}$$

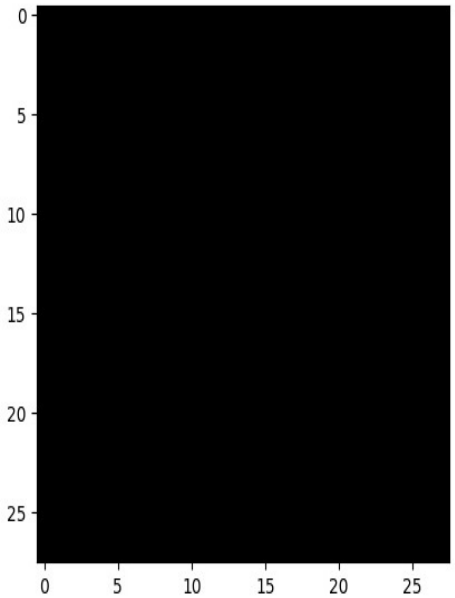
Given $\tilde{v}$ sample $\tilde{h}$ using (1). Contrastive divergence approximates

$$\langle v_i h_a\rangle_{data} = \frac{1}{N_s}\sum_{A=1}^{N_s} \hat{v}_i^{(A)}\hat{h}_a^{(A)} \tag{3}$$

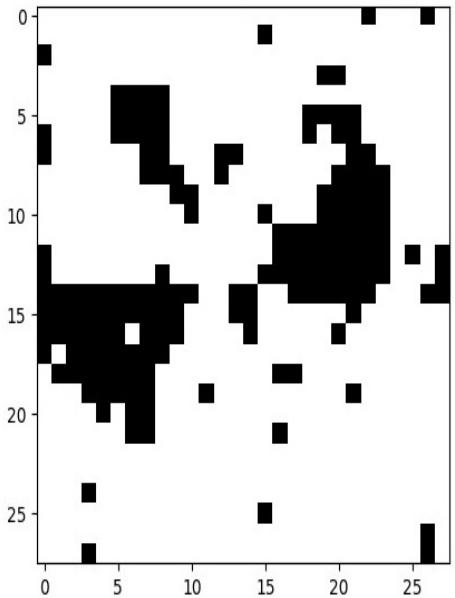
$$\langle v_i h_a\rangle_{model} = \frac{1}{N_s}\sum_{A=1}^{N_s} \tilde{v}_i^{(A)}\tilde{h}_a^{(A)} \tag{4}$$

Ising Model

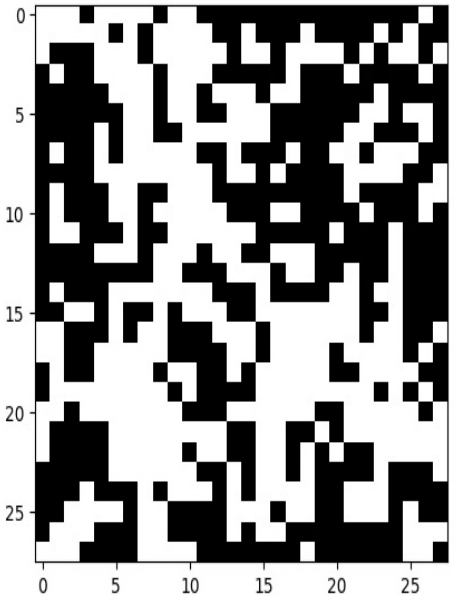
$$\rho(\{v_i\}) = e^{-\beta H_{\text{Ising}}} \qquad \beta = \frac{1}{T} \qquad H_{\text{Ising}} = -\sum_{<i,j>} \sigma_i \sigma_j$$



T<Tc

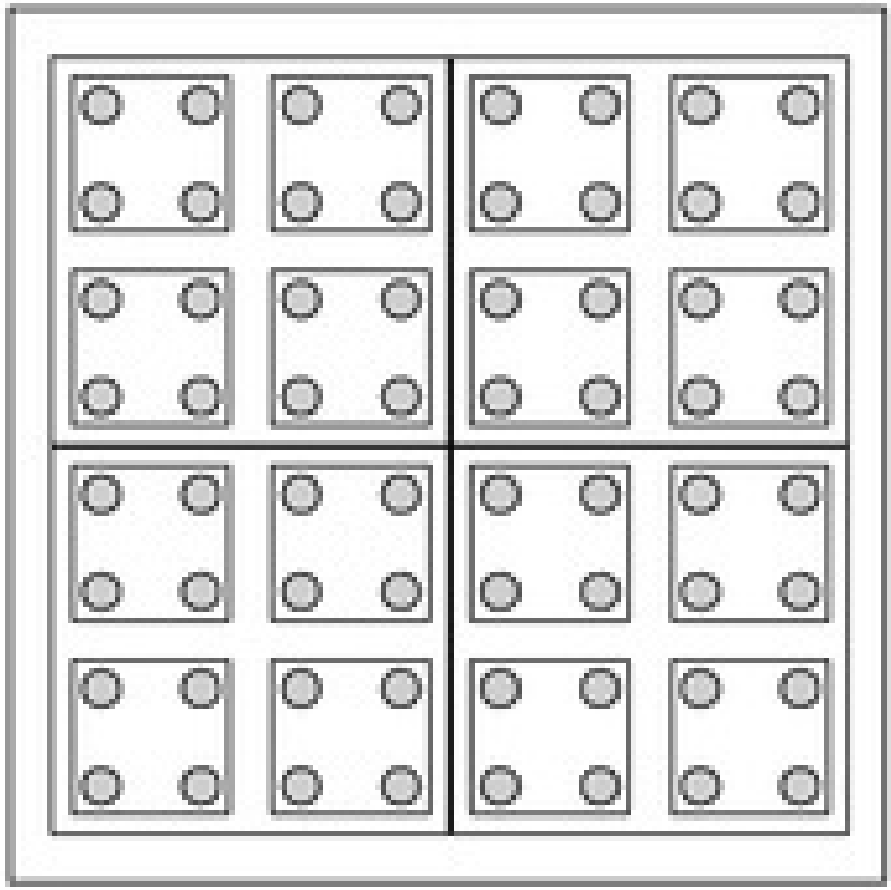


Tc

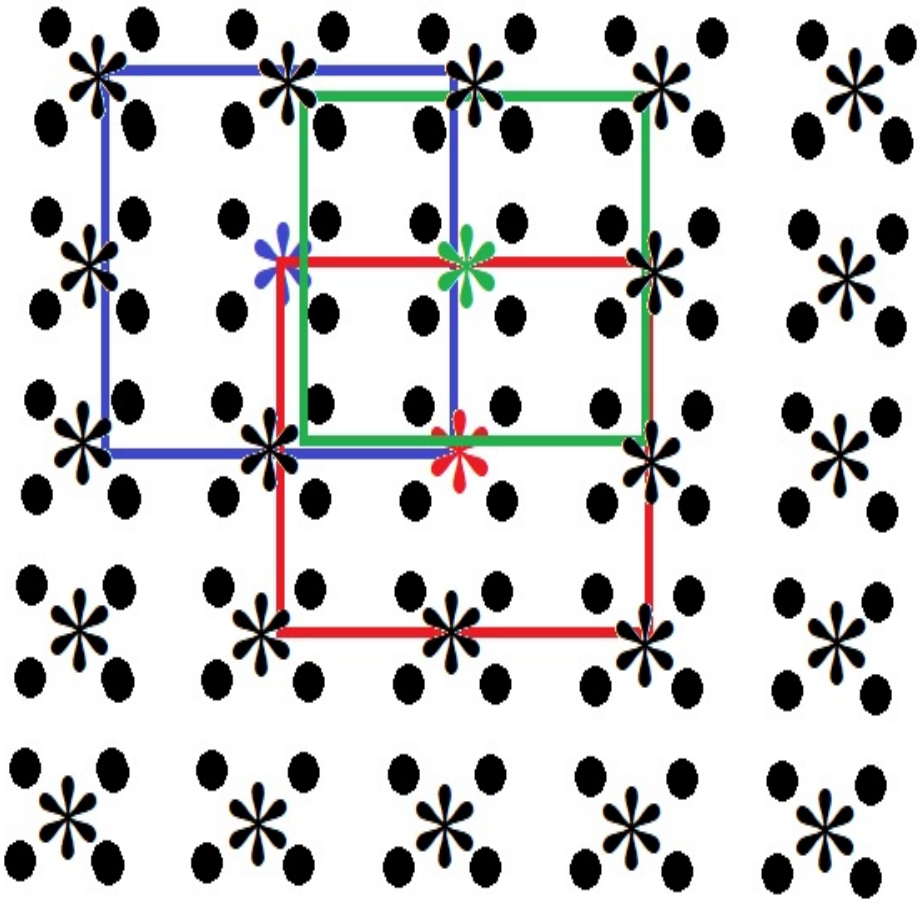


T>Tc

RG for the Ising Model



Overlapping RG for the Ising Model



Comparing RG and RBM

$$
\begin{aligned}
E &= -\sum_a b_a^{(h)} h_a - \sum_{i,a} v_i W_{ia} h_a - \sum_i b_i^{(v)} v_i \\
p(\mathbf{v},\mathbf{h}) &= \frac{1}{\mathcal{Z}} e^{-E} \qquad \mathcal{Z} = \sum_{\{\mathbf{v},\mathbf{h}\}} e^{-E} \\
p(h_a = 1|v) &= \frac{1}{2}\left(1 + \tanh\left(\sum_i W_{ia} v_i + b_a^{(h)}\right)\right)
\end{aligned}
$$

The trained bias is small $|b_a^{(h)}| \approx 0.01$ and trained weights large $|W_{ia}| \sim 10$, $\Rightarrow p(h_a = 1|v)$ always close to 0 or 1. Problem is essentially deterministic.

Ignoring normalization h_a is essentially $\sum_i W_{ia} v_i$.

Comparing RG and RBM

$$p(h_a = 1|v) \quad = \quad \frac{1}{2}\left(1 + \tanh\left(\sum_i W_{ia} v_i + b_a^{(h)}\right)\right)$$

$|b_a^{(h)}| \approx 0.01$ and $|W_{ia}| \sim 10$, $\Rightarrow p(h_a = 1|v)$ always close to 0 or 1.
Ignoring normalization $h_a = \sum_i W_{ia} v_i$.

Coarse graining of RG can be written as (this is usual block rule)

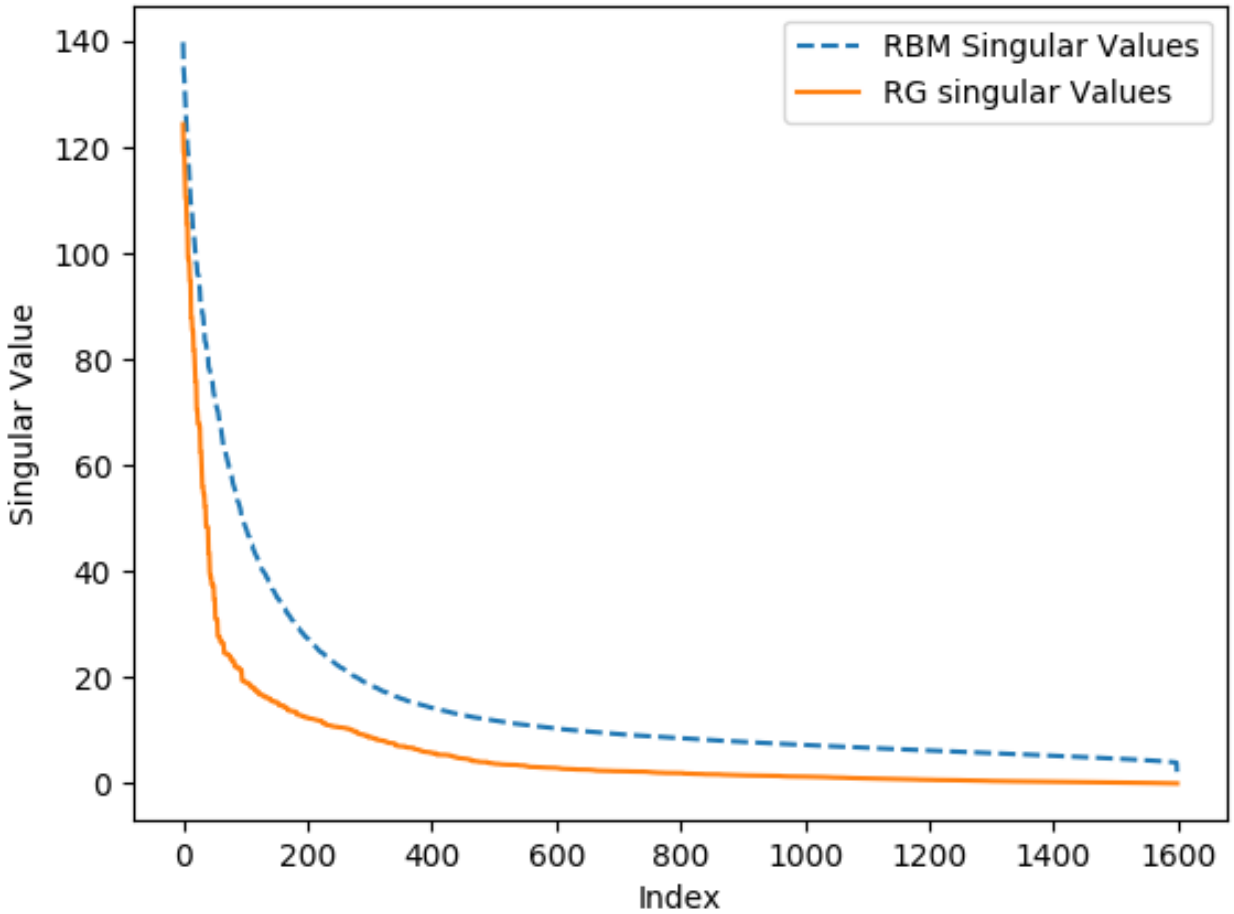
$$h_{x+\frac{1}{2},y+\frac{1}{2}} = \frac{v_{x+1,y+1} + v_{x,y} + v_{x+1,y} + v_{x,y+1}}{4}$$

i.e. $h_a = \sum_i A_{ia} v_i$.

Idea: Compare W_{ia} and A_{ia}.

SVD of the trained weight matrix vs RG: Singular Values

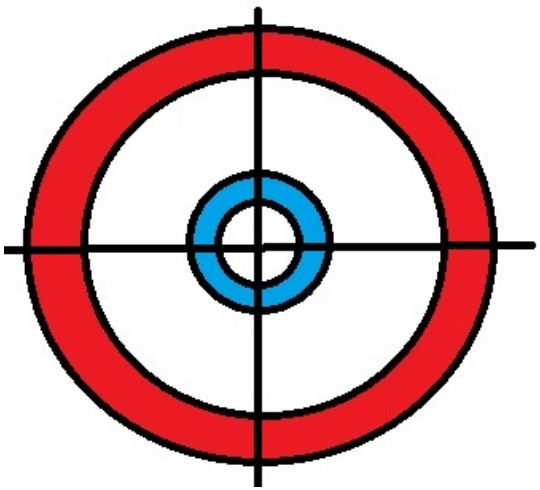
Block spins averaging $16 \times 16 = 256$ blocks, on the 80×80 lattice.

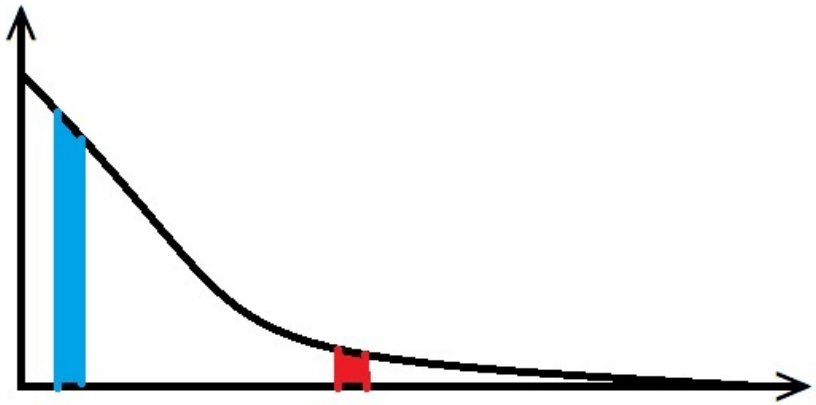


Close connection between RBM and RG

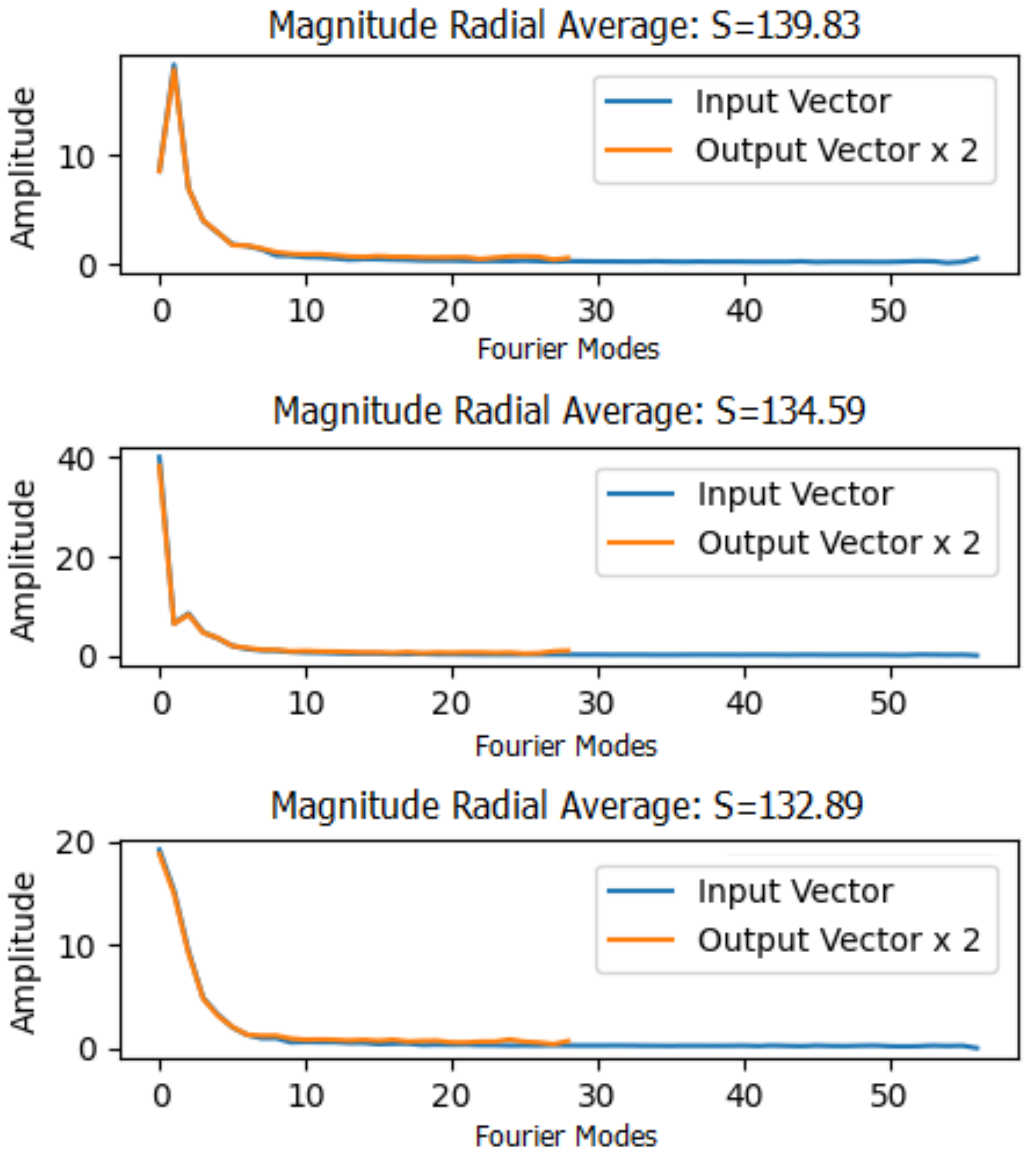
Close similarity between singular values of W_{ia} from RBM and A_{ia} from RG suggests we compare their singular vectors. Right singular vectors are 6400 dimensional vectors; left singular vectors are 1600 dimensional vectors.

RG suggests comparing in Fourier space. Long scale features (= large wavelengths) are kept, short scale features (=small wavelengths) are discarded.

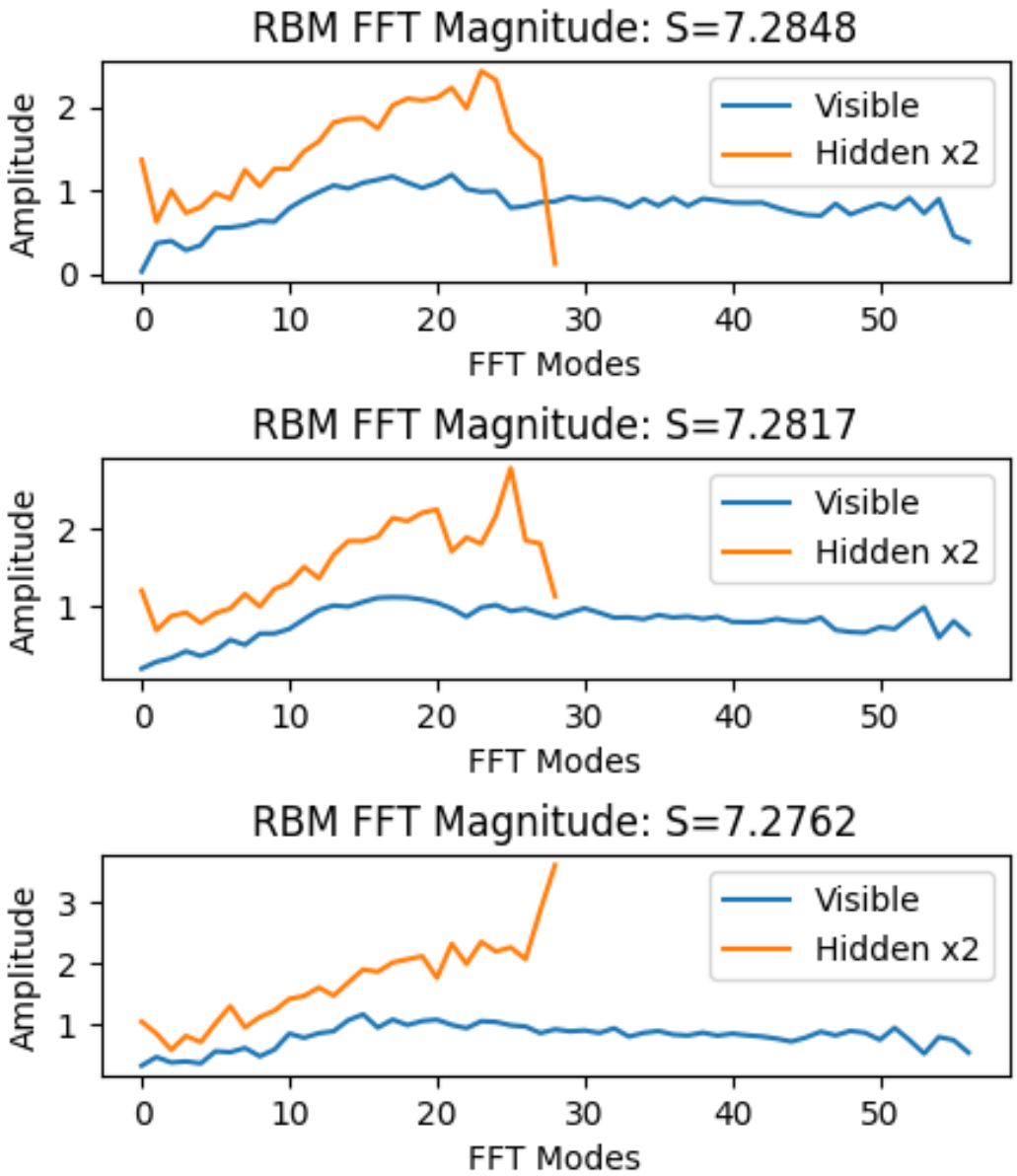




SVD of the trained weight matrix vs RG: Singular Vectors



SVD of the trained weight matrix vs RG: Singular Vectors



RG

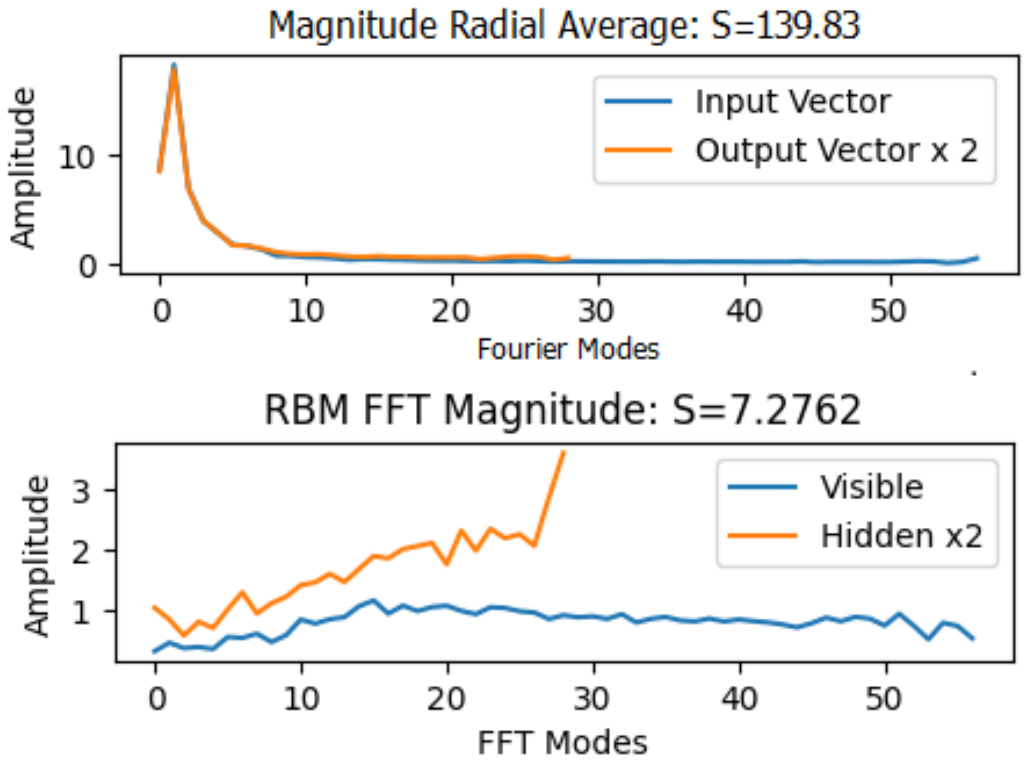
Momentum space RG follows three steps:

1. Split the field into high and low momentum modes.
2. Integrate over the high momentum modes.
3. Rescale the fields, as well as momenta and position co-ordinates.

Close to the free field fixed point, the second step of RG can be simplified as follows:

2'. Discard the high momentum modes.

RG



The weight matrix naturally projects out the high momentum components:

$$W = \sum_I |Output\rangle \lambda_I \langle Input|$$

Generalization analysis

Number of parameters = 10 240 000; number of data samples = 40 000

Keep first 230 of 1600 singular values. Reduces number of effective parameters by $\sim \frac{1}{7}$.

Keep first 9 of 56 components of radial Fourier transform of singular vectors. Reduces number of effective parameters by $\sim \frac{1}{36}$.

Number of effective parameters = $\frac{1}{7} \times \frac{1}{36} \times 10240000 = 40\ 635$.

$\Rightarrow$ Number of effective parameters $\sim$ number of data samples, explaining why the network generalizes.

Comments

Number of parameters kept depends on "noise" we tolerate. h_a = output of network; h'_a = output after some parameters set to zero.

$$\frac{\sqrt{\sum_a (h_a - h'_a)^2}}{\sqrt{\sum_b h_b^2}} < \eta$$

Number of parameters dropped fixed by η. We choose $\eta = 0.1$. Changing η changes number of effective parameters, but we're in ball park: number of effective parameters $\sim$ number of data samples.

Factor for singular values $\sim \frac{1}{7}$. Factor for singular vectors $\sim \frac{1}{36}$.

But: maybe this is all special to the RBM trained on Ising data?

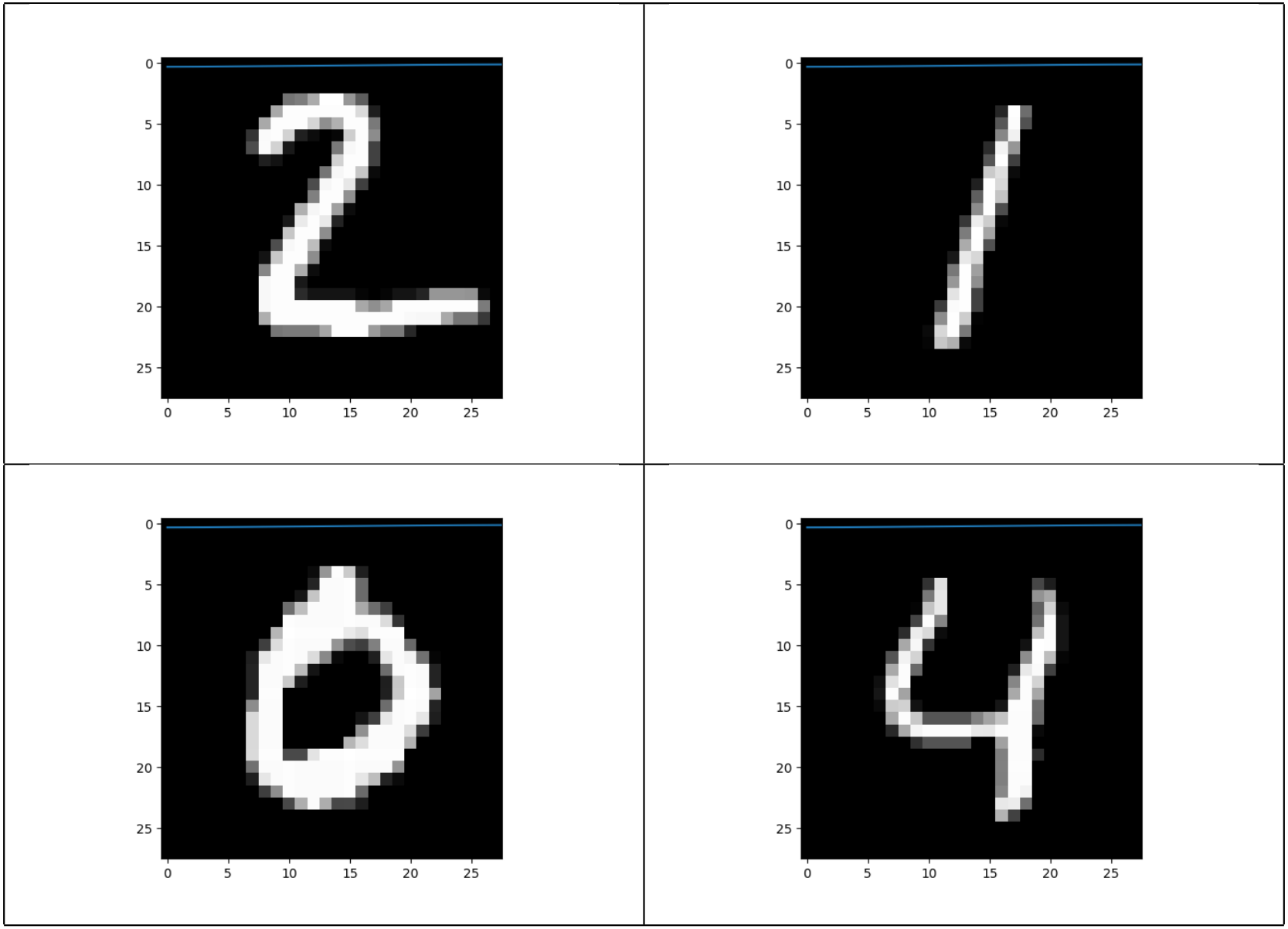
MNIST dataset

A greyscale dataset of handwritten images of size 28×28 pixels.

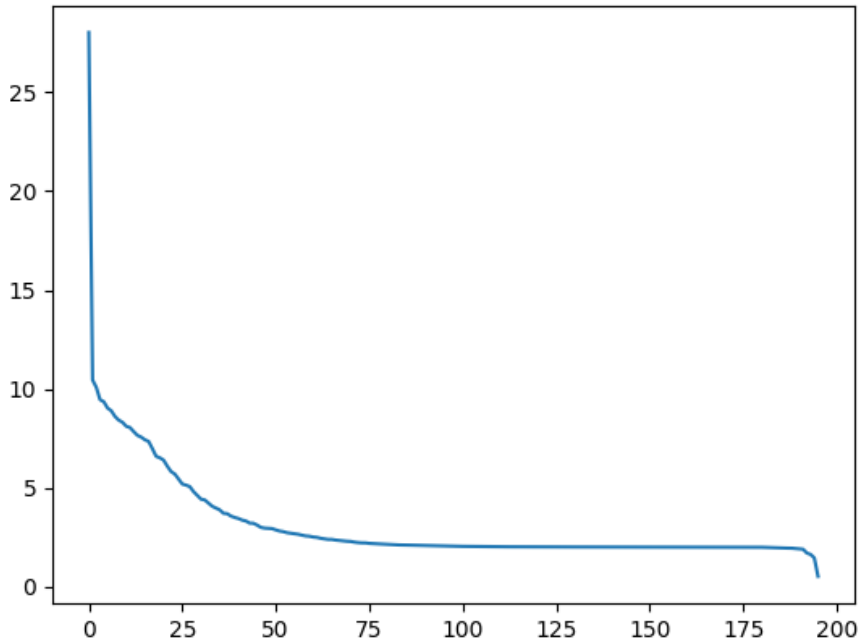
Train RBM with $784 = 28 \times 28$ inputs and $196 = 14 \times 14$ outputs, with training set of 12 000 images. Training completed in 30 epochs.

Weight matrix initialized with the block spin initial condition.

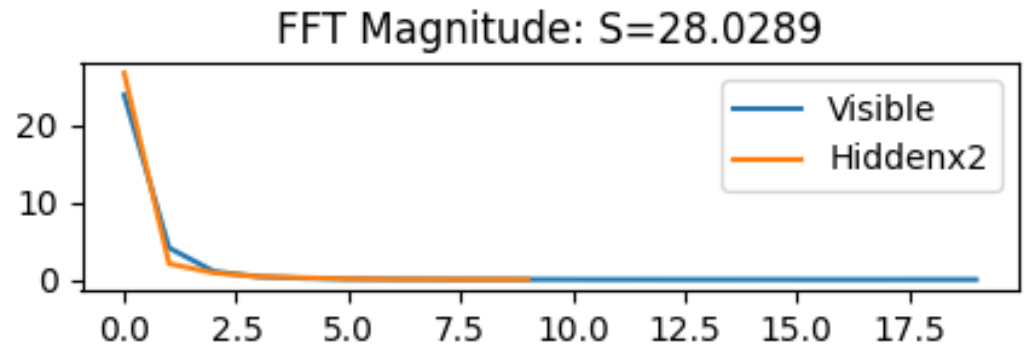
MNIST dataset

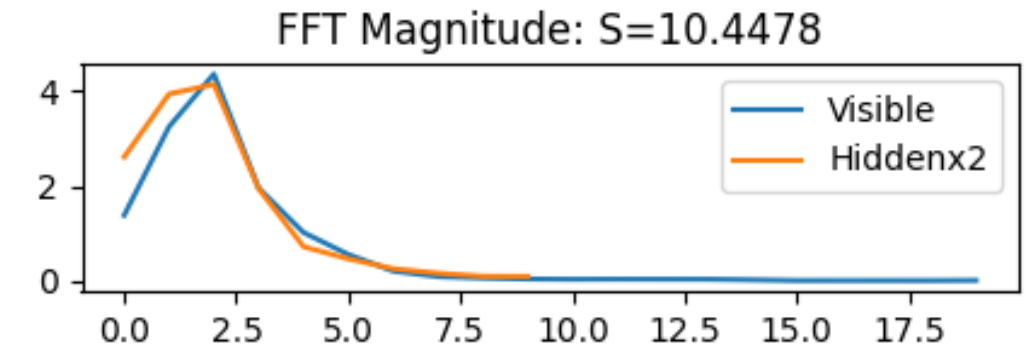


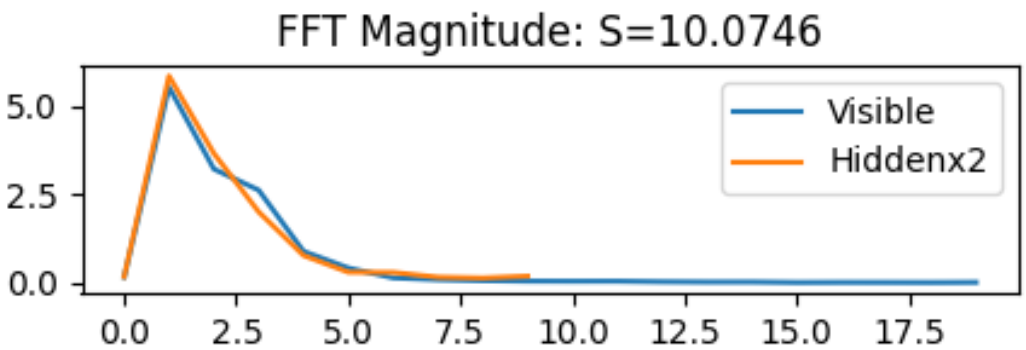
MNIST dataset: Singular Values



MNIST dataset: Singular Vectors







MNIST dataset: Generalization Analysis

The network is trained using 12 000 images. The network has 153 664 parameters.

A fraction of $\frac{75}{196}$ singular values are not zero.

A fraction of $\frac{64}{324}$ of the Fourier modes of the singular vectors are not zero.

Thus, the number of effective parameters is 11 615, so that this network will generalize.

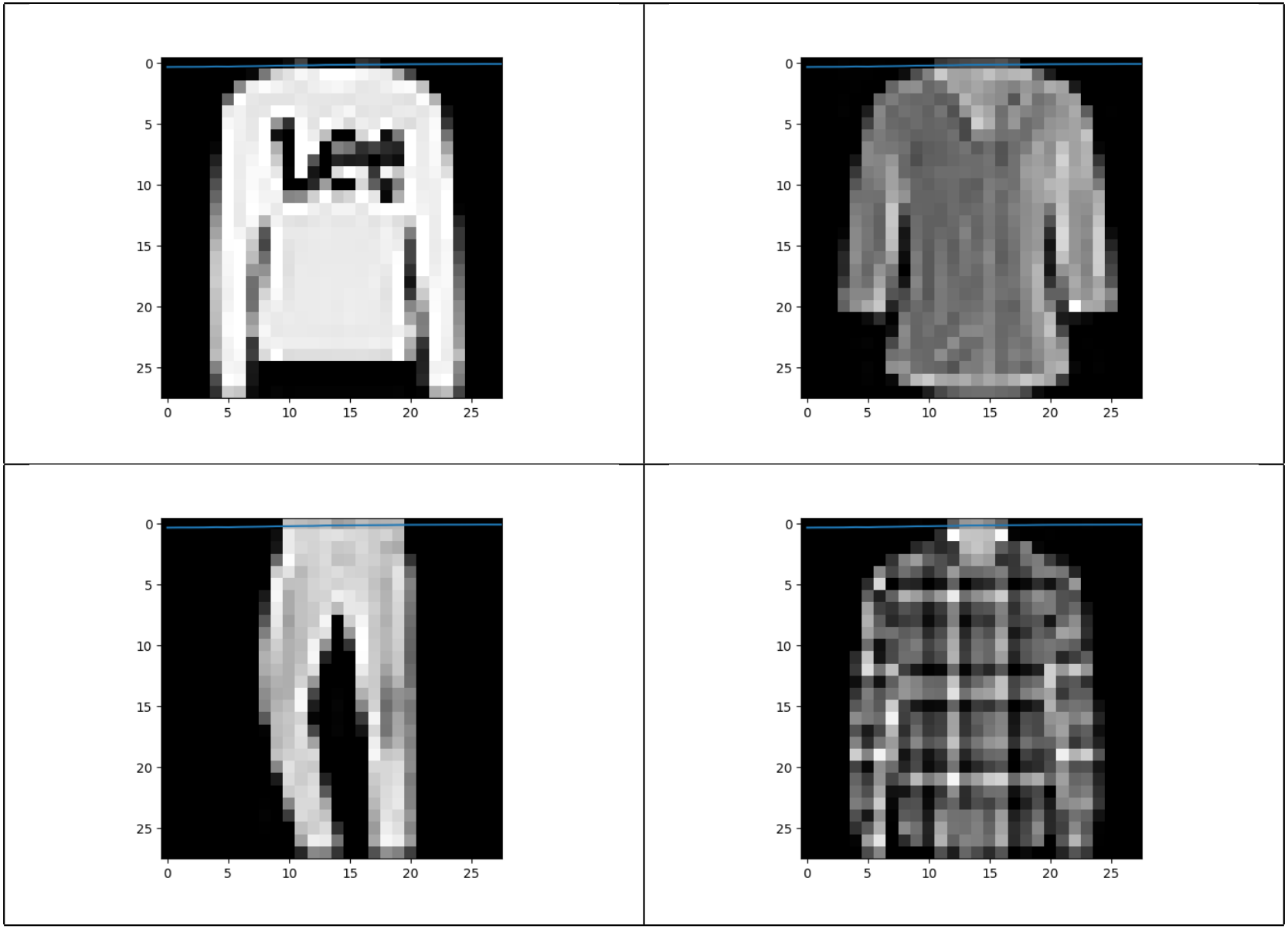
MNIST fashion dataset

Greyscale dataset consisting of images of size 28×28 pixels. Images are items of clothing forming several categories.

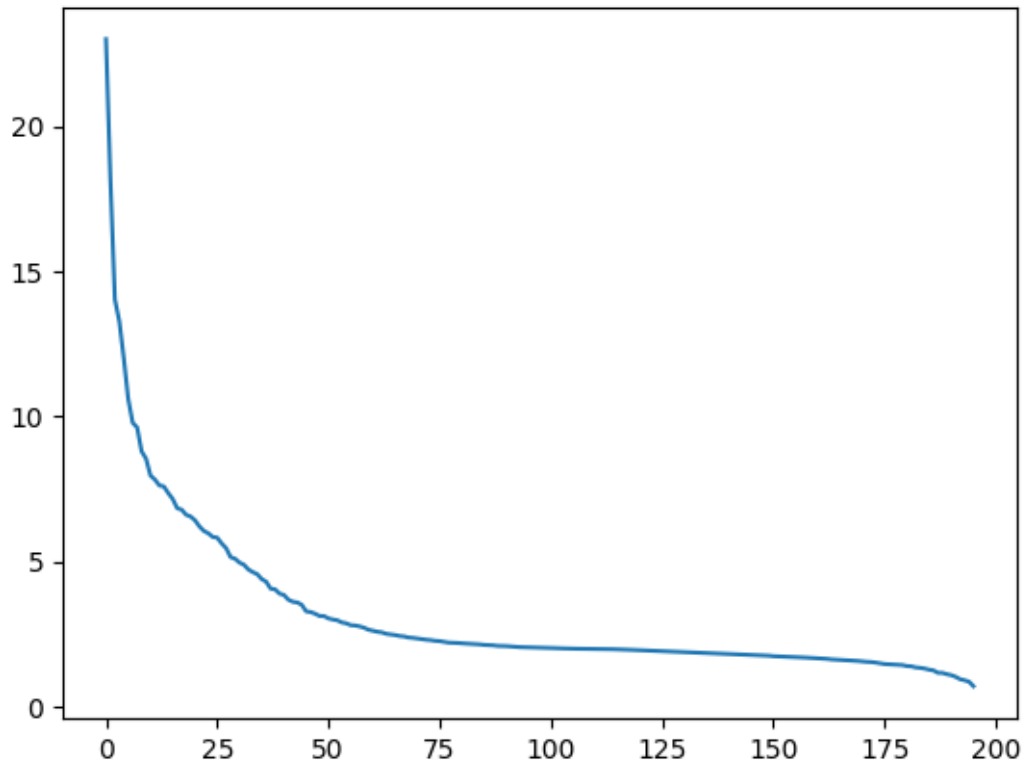
Train RBM with $784 = 28 \times 28$ inputs and $196 = 14 \times 14$ outputs, using training set of 5000 images. Training completed in 300 epochs.

Weight matrix is initialized with the block spin initial condition.

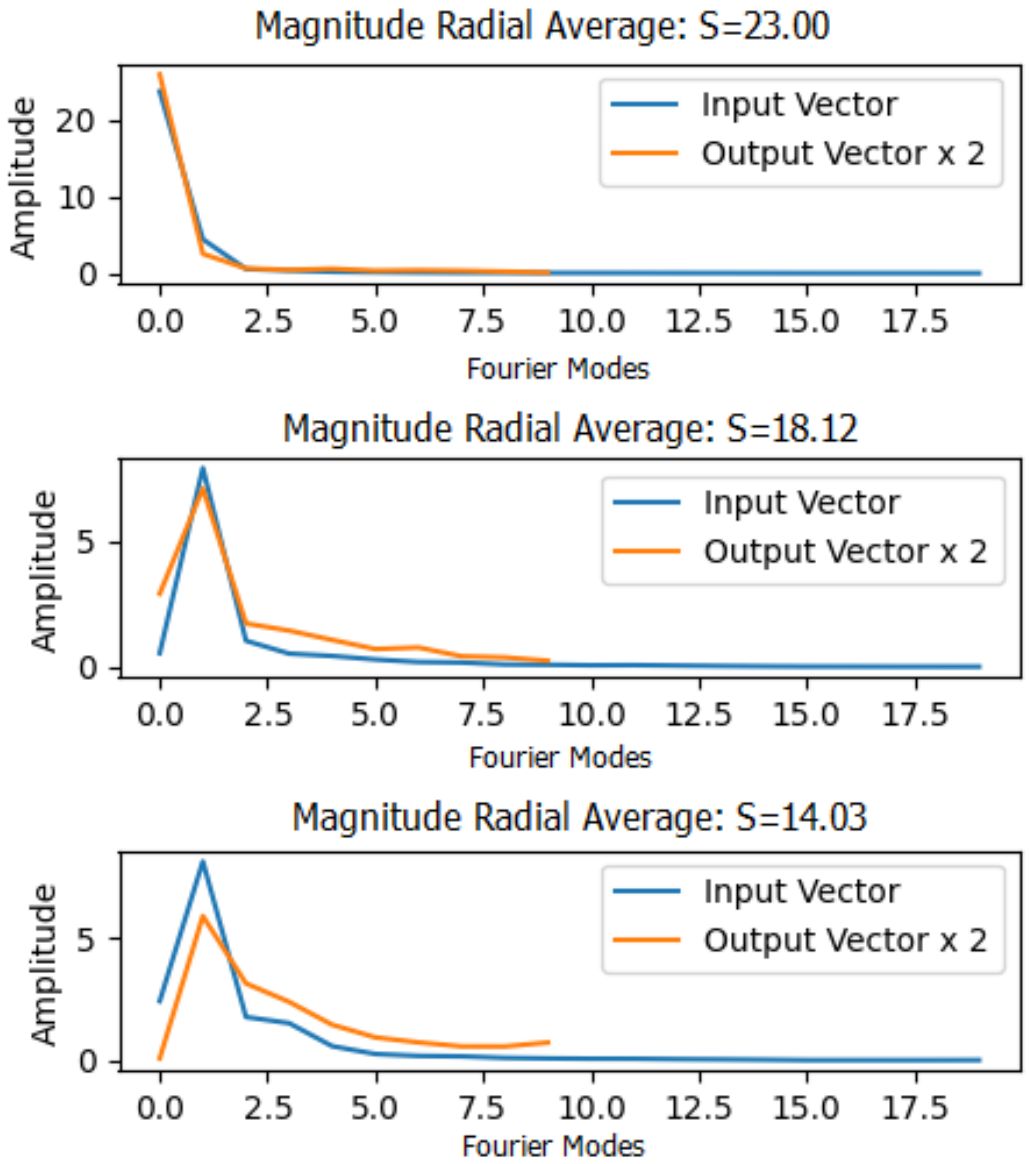
MNIST Fashion dataset



MNIST Fashion dataset: Singular Values



MNIST Fashion dataset: Singular Vectors



MNIST Fashion dataset: Generalization Analysis

The network is trained using 5 000 images. The network has 153 664 parameters.

A fraction of $\frac{60}{196}$ singular values are not zero.

A fraction of $\frac{64}{324}$ of the Fourier modes of the singular vectors are not zero.

Thus, the number of effective parameters is 5 226, explaining why this network generalizes.

Conclusions

Deep learning (RBMs) perform a coarse graining: precisely the coarse graining of momentum space RG near FFFP.

Connection to coarse graining resolves generalization puzzle $\Rightarrow$ dramatic reduction in number of parameters.

Another facet of puzzle: break data set into two disjoint subsets, one for training, one for testing. Nothing suggests results are independent of split. This independence (present in successful deep networks) is mysterious.

Optimization puzzle: training finds minimum of a function of millions of parameters. Generically such a function has many local minima. Any simple learning rule will fail to reach global minimum.

Conclusions

Tacit assumption: deep learning is curve fitting. We suggest: deep learning is coarse graining.

System is a container of water. The state of the water specified by positions and velocities of the water molecules. Microscopic description uses masses of the water molecules and complicated interaction potentials.

Coarse grained description uses few parameters: density of the fluid, its viscosity, speed of sound in the fluid. State specified by much smaller list of quantities: pressure and temperature of the water.

For water at equilibrium, any subregion of the water (training data) will fix parameters of coarse grained description (trained network). Since the water is at equilibrium, our results are independent of how training data is chosen.

Thanks for your attention!