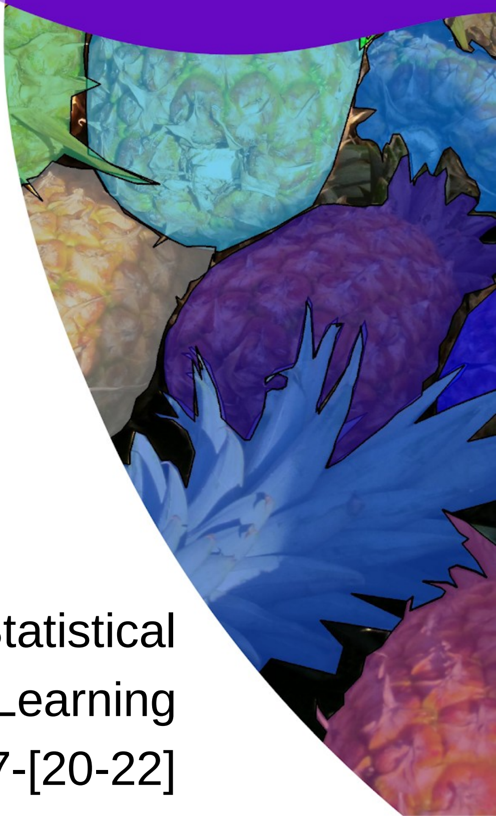
NEW YORK UNIVERSITY

From Machine Learning to Autonomous Intelligence Lecture 2

Yann LeCun
NYU - Courant Institute & Center for Data Science
Meta - Fundamental AI Research
http://yann.lecun.com

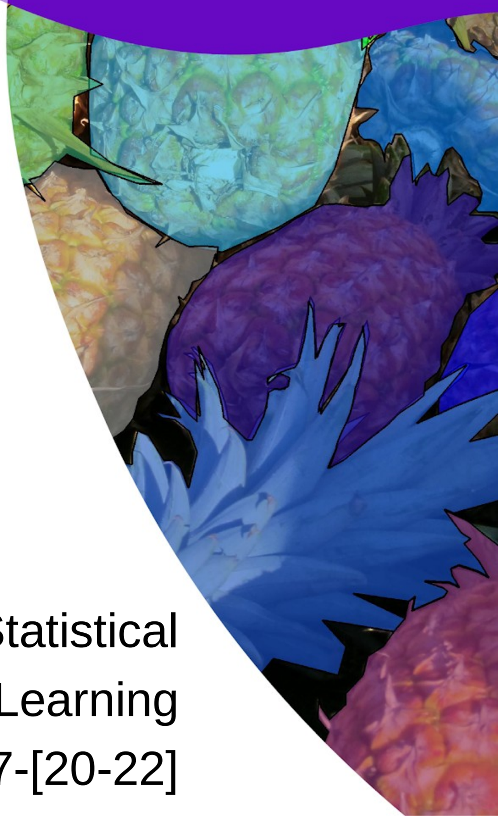


Summary School on Statistical Physics & Machine Learning
Les Houches, 2022-07-[20-22]

NEW YORK UNIVERSITY

From Machine Learning to Autonomous Intelligence

Yann LeCun
NYU - Courant Institute & Center for Data Science
Meta - Fundamental AI Research
http://yann.lecun.com



Summer School on Statistical Physics & Machine Learning
Les Houches, 2022-07-[20-22]

FACEBOOK AI

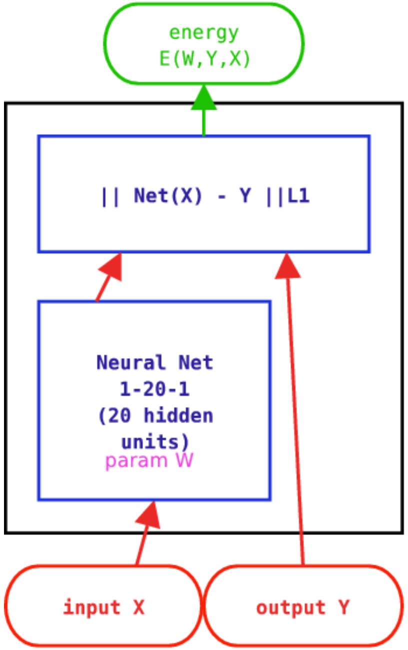
EBM Training

1. Contrastive methods
2. Regularized & Architectural methods

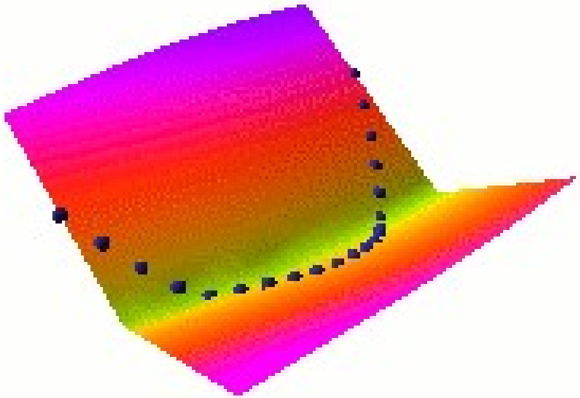
Y. LeCun

Regularized and Architectural EBM Training

- With some architecture, simply pushing down on the energy of data points will make the energy function take the right shape.

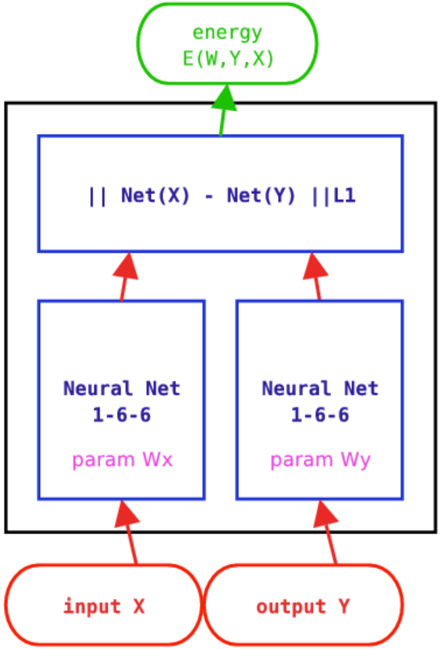


(a)

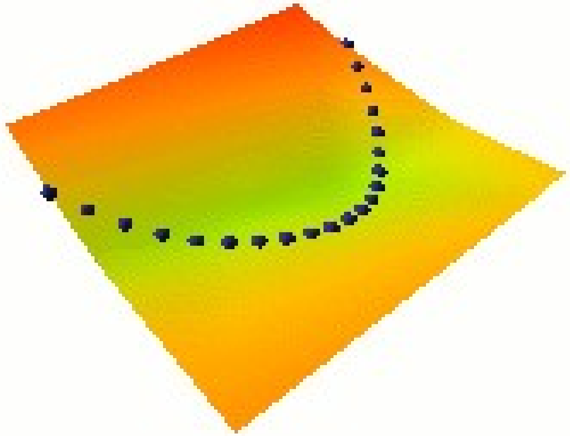


Architectures that can collapse!

- **With architectures like joint embedding, pushing down on the training sample energy can make the energy landscape flat**
  - The networks ignores the input and produce identical constant outputs

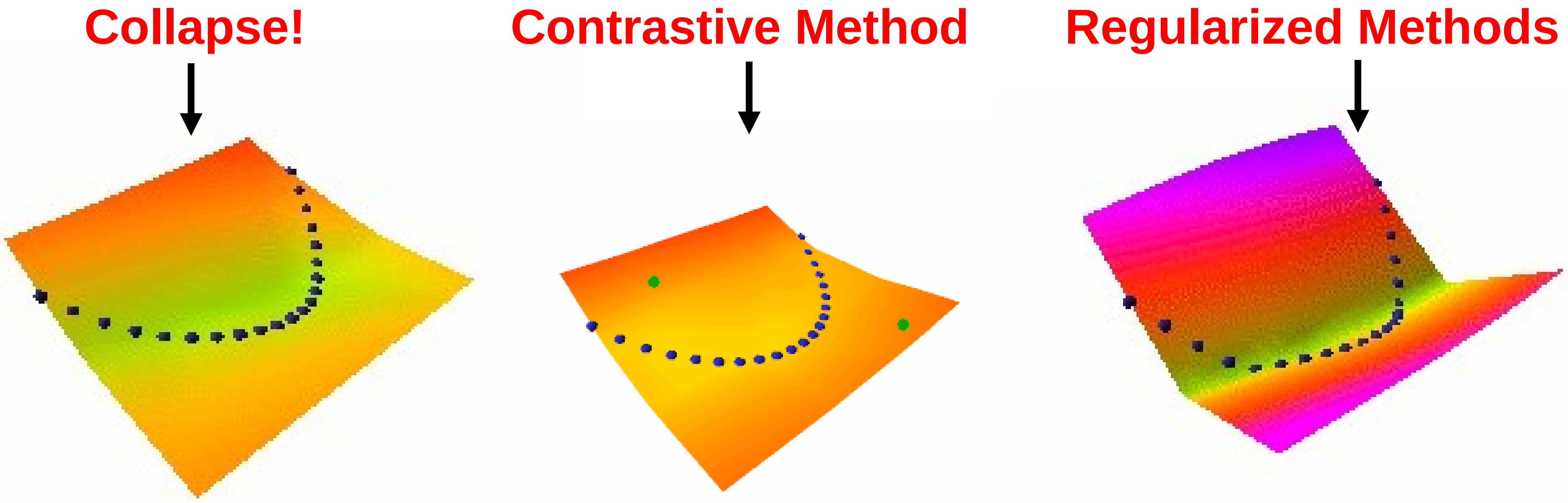


(b)



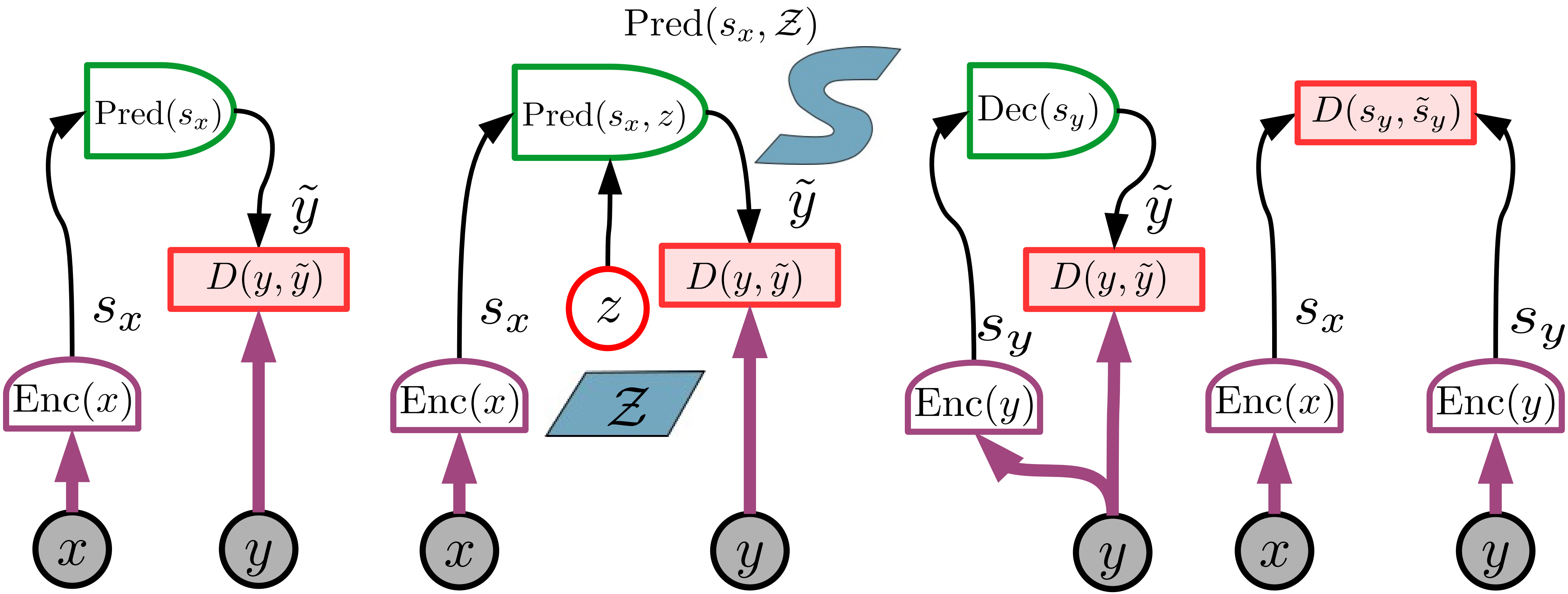
Shaping the energy surface / preventing collapse

- **A flexible energy surface can take any shape.**
- **We need a loss function that shapes the energy surface so that:**
  - Data points have low energies
  - Points outside the regions of high data density have higher energies.



EBM Architectures

- **Some architectures can lead to a collapse of the energy surface**



a) Prediction / regression
NO COLLAPSE

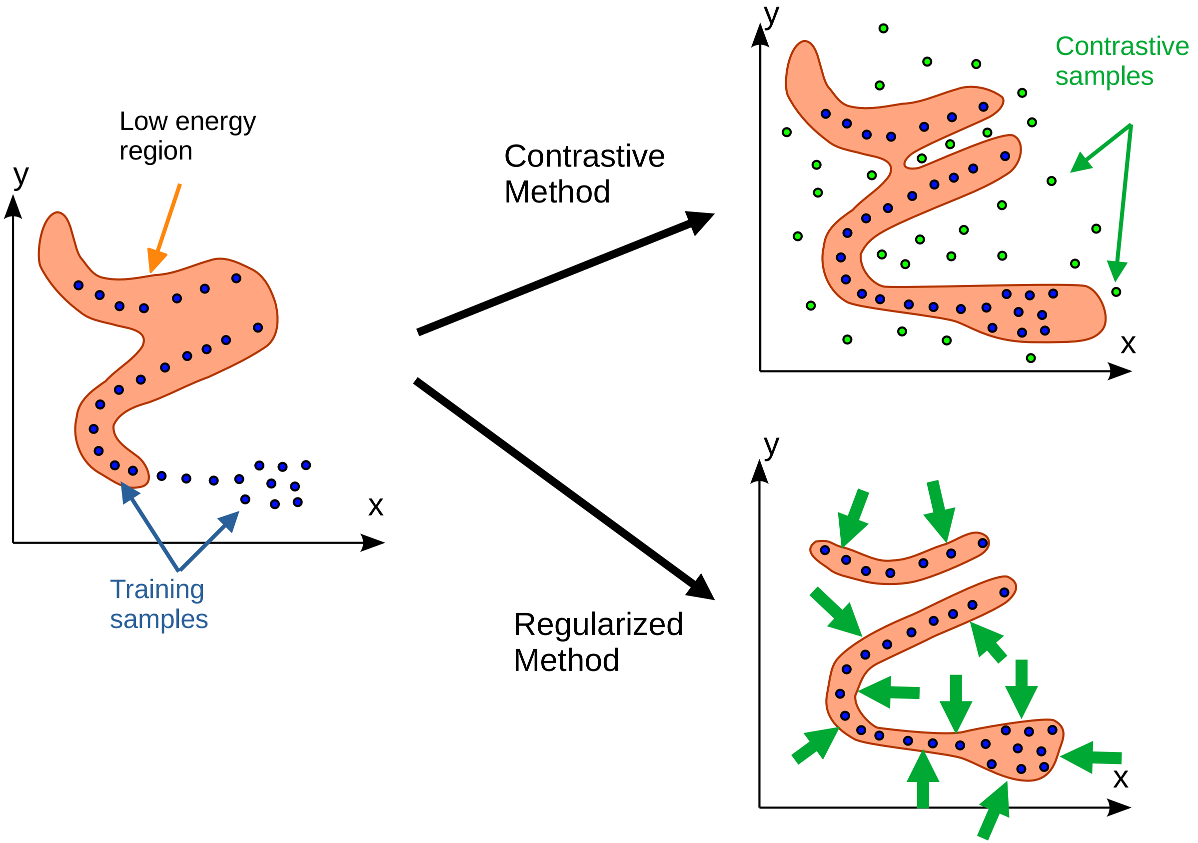
b) Generative latent-variable Architecture
CAN COLLAPSE

c) Auto-Encoder
CAN COLLAPSE

d) Joint Embedding Architecture
CAN COLLAPSE

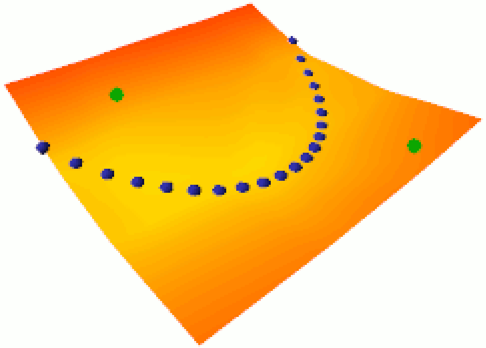
EBM Training: two categories of methods

- **Contrastive methods**
  - Push down on energy of training samples
  - Pull up on energy of suitably-generated contrastive samples
  - Scales very badly with dimension
- **Regularized Methods**
  - Regularizer minimizes the volume of space that can take low energy



Contrastive methods vs Regularized Methods

- **Contrastive methods: works with any architecture**
  - Expensive in high dimension
  - Example of contrastive loss: pick a ŷ to push up.

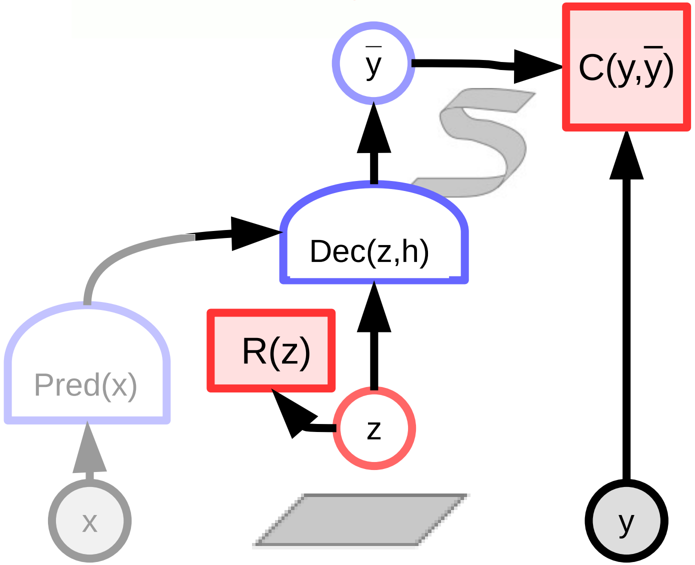


$$\mathcal{L}(x, y, \hat{y}, w) = [F_w(x, y) - F_w(x, \hat{y}) + m(y, \hat{y})]^+$$

- **Regularized methods: minimizing the volume of low-energy space**
  - E.g. by limiting the capacity of the latent

$$\mathcal{L}(x, y, w) = F_w(x, y)$$

$$F_w(x, y) = \min_z [C(\mathrm{Dec}(\mathrm{Pred}(x), z), y) + R(z)]$$



Contrastive Methods vs Regularized/Architectural Methods

- **Contrastive: [they all are different ways to pick which points to push up]**
  - C1: push down of the energy of data points, push up everywhere else: Max likelihood (needs tractable partition function or variational approximation)
  - C2: push down of the energy of data points, push up on chosen locations: max likelihood with MC/MMC/HMC, Contrastive divergence, Metric learning/Siamese nets, Ratio Matching, Noise Contrastive Estimation, Min Probability Flow, adversarial generator/GANs
  - C3: train a function that maps points off the data manifold to points on the data manifold: denoising auto-encoder, masked auto-encoder (e.g. BERT)
- **Regularized/Architectural: [Different ways to limit the information capacity of the latent representation]**
  - A1: build the machine so that the volume of low energy space is bounded: PCA, K-means, Gaussian Mixture Model, Square ICA, normalizing flows…
  - A2: use a regularization term that measures the volume of space that has low energy: Sparse coding, sparse auto-encoder, LISTA, Variational Auto-Encoders, discretization/VQ/VQVAE.
  - A3: $F(x,y) = C(y, G(x,y))$, make $G(x,y)$ as "constant" as possible with respect to y: Contracting auto-encoder, saturating auto-encoder
  - A4: minimize the gradient and maximize the curvature around data points: score matching

Contrastive Methods

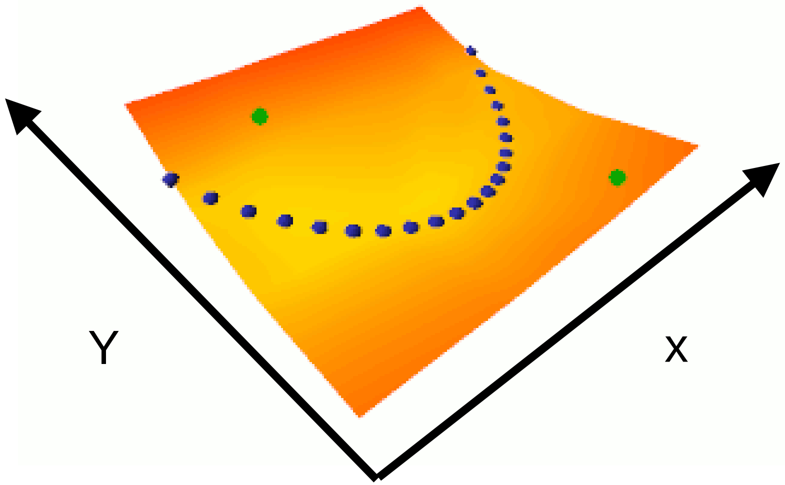
Push down on the energy of training samples
Pull up on the energy of “contrastive” samples
Lots of possible loss functions

Contrastive energy-based learning

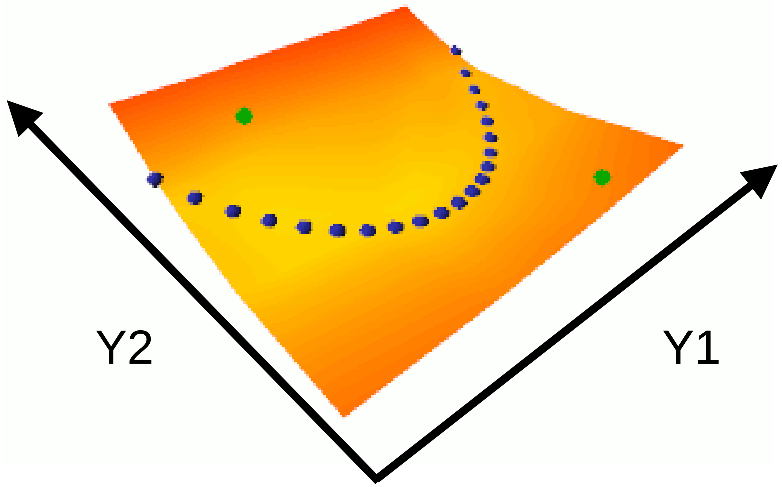
- **Push down on the energy of training samples**
- **Pull up on the energy of other “well-chosen” points**

$$\mathcal{L}(x_1 \ldots x_{p^+}, y_1 \ldots y_{p^+}, \hat{y}_1 \ldots \hat{y}_{p^-}, w) = H\left(E(x_1, y_1), \ldots E(x_{p^+}, y_{p^+}), E(x_1, \hat{y}_1), \ldots E(x_{p^+}, \hat{y_{p^+}}), M(Y_{1\ldots p^+}, \hat{Y}_{1\ldots p^-})\right)$$

conditional



unconditional



Contrastive Methods: pairwise margin losses

- **Push down on data points, push up on other points**
  - well chosen contrastive points
  - General margin loss: $$\mathcal{L}(x, y, \hat{y}, w) = H(F_w(x, y), F_w(x, \hat{y}), m(y, \hat{y}))$$
    - Where $H(F^+, F^-, m)$ is a strictly increasing function of F^+ and a strictly decreasing function of F^-, at least whenever $F^- - F^+ < m$.
  - Examples:
    - Simple [Bromley 1993]:

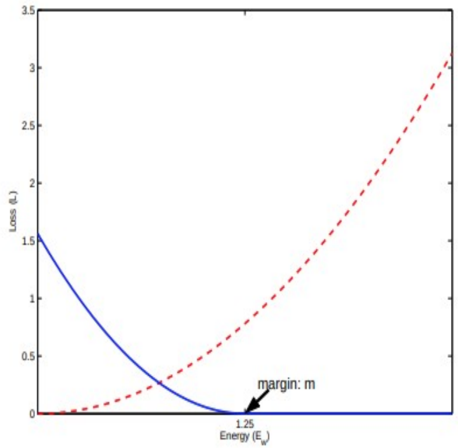
$$\mathcal{L}(x, y, \hat{y}, w) = [F_w(x, y)]^+ + [m(y, \hat{y}) - F_w(x, \hat{y})]^+$$

    - Hinge pair loss [Altun 2003], Ranking loss [Weston 2010]:

$$\mathcal{L}(x, y, \hat{y}, w) = [F_w(x, y) - F_w(x, \hat{y}) + m(y, \hat{y})]^+$$

    - Square-Square: [Chopra CVPR 2005] [Hadsell CVPR 2006]:

$$\mathcal{L}(x, y, \hat{y}, w) = \left([F_w(x, y)]^+\right)^2 + \left([m(y, \hat{y}) - F_w(x, \hat{y})]^+\right)^2$$



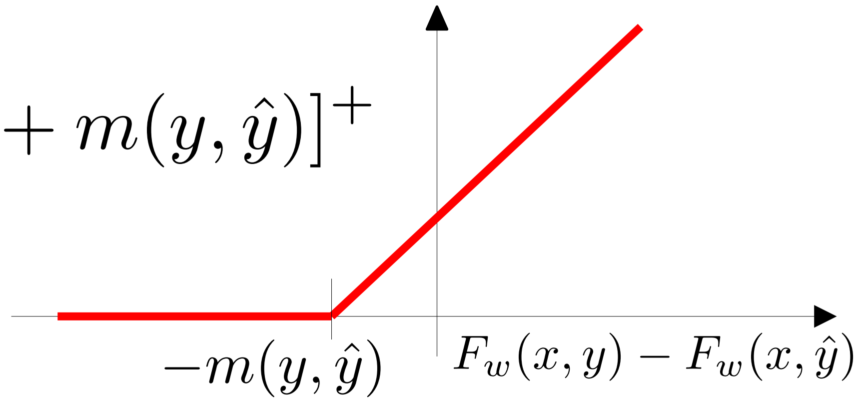
General margin loss

- Considers all possible outputs (or a well-chosen subset)

$$\mathcal{L}(x,y,w) = \sum_{\hat{y}\in\mathcal{Y}} H(F_w(x,y), F_w(x,\hat{y}), m(y,\hat{y}))$$

- Hinge loss that makes F(x,y) lower than F(x,y’) by a quantity (margin) that depends on the distance between y and y’
- Example:

$$\mathcal{L}(x,y,w) = \sum_{\hat{y}\in\mathcal{Y}} [F_w(x,y) - F_w(x,\hat{y}) + m(y,\hat{y})]^+$$



Y. LeCun

Plenty of Contrastive Loss Functions to Choose From

Good and bad loss functions: good ones have non-zero margin

Loss	Formula	Margin
energy loss	$F(x,y)$	0
perceptron	$F(x,y) - \min_{\check{y}\in} F(x,\check{y})$	0
hinge	$\max\left(0, m + F(x,y) - F(x,\hat{y})\right)$	m
log	$\log\left(1 + e^{F(x,y)-F(x,\hat{y})}\right)$	∞
LVQ2	$\min\left(M, \max(0, F(x,y) - F(x,\hat{y})\right)$	0
MCE	$\left(1 + e^{-(F(x,y)-F(x,\hat{y}))}\right)^{-1}$	∞
square-square	$F(x,y)^2 - \left(\max(0, m(y,\hat{y}) - F(x,\hat{y}))\right)^2$	m
square-exp	$F(x,y)^2 + \beta e^{-F(x,\hat{y})}$	∞
NLL/MMI	$F(x,y) + \frac{1}{\beta}\log\int_{\hat{y}\in} e^{-\beta F(x,\hat{y})}$	∞
MEE	$1 - e^{-\beta F(x,y)} / \int_{\hat{y}\in} e^{-\beta F(x,\hat{y})}$	∞

Y. LeCun

Loss function zoo for contrastive EBM training

	Method	Energy	$\hat{y}$ Generation	Loss
1	Max Likelihood	discrete y	exhaustive	$F_w(x,y) + \log \sum_{y' \in \mathcal{Y}} \exp(-F_w(x,y'))$
2	Max Likelihood	tractable	exhaustive	$F_w(x,y) + \log \int_{y' \in \mathcal{Y}} \exp(-F_w(x,y'))$
3	Max likelihood	any	MC or MCMC	$F_w(x,y) - F_w(x,\hat{y})$
4	Contr. Divergence	any	trunc'd MCMC	$F_w(x,y) - F_w(x,\hat{y})$
5	Pairwise Hinge	any	most offending	$[F_w(x,y) - F_w(x,\hat{y}) + m(y,\hat{y})]^+$
6	Min-Hinge	positive	most offending	$F_w(x,y) + [m(y,\hat{y}) - F_w(x,\hat{y})]^+$
6	Square-Hinge	divergence	most offending	$F_w(x,y)^2 + \left([m(y,\hat{y}) - F_w(x,\hat{y})]^+\right)^2$
7	Square-Exp	any	most offending	$F_w(x,y)^2 + \exp(-\beta F_w(x,\hat{y}))$
8	Logistic	any	most offending	$\log(1 + \exp(F_w(x,y) - F_w(x,\hat{y})))$
9	GAN	any	$\hat{y} = g_u(z)$	$H(F_w(x,y), F_w(x,\hat{y}), m(y,\hat{y}))$
10	Denoising AE	$D(y, g_w(y))$	$\hat{y} = N(y)$	$D(y, g_w(\hat{y}))$

Contrastive Methods: group losses

- **Push down on a group of data points, push up on a group of contrastive points**
  - General group loss on p^+ data points and p^- contrastive points:

$$\mathcal{L}(x_1 \dots x_{p^+}, y_1 \dots y_{p^+}, \hat{y}_1 \dots \hat{y}_{p^-}, w) = H\left(F(x_1, y_1), \dots F(x_{p^+}, y_{p^+}), F(x_1, \hat{y}_1), \dots F(x_{p^+}, \hat{y_{p^+}}), M(Y_{1\dots p^+}, \hat{Y}_{1\dots p^-})\right)$$

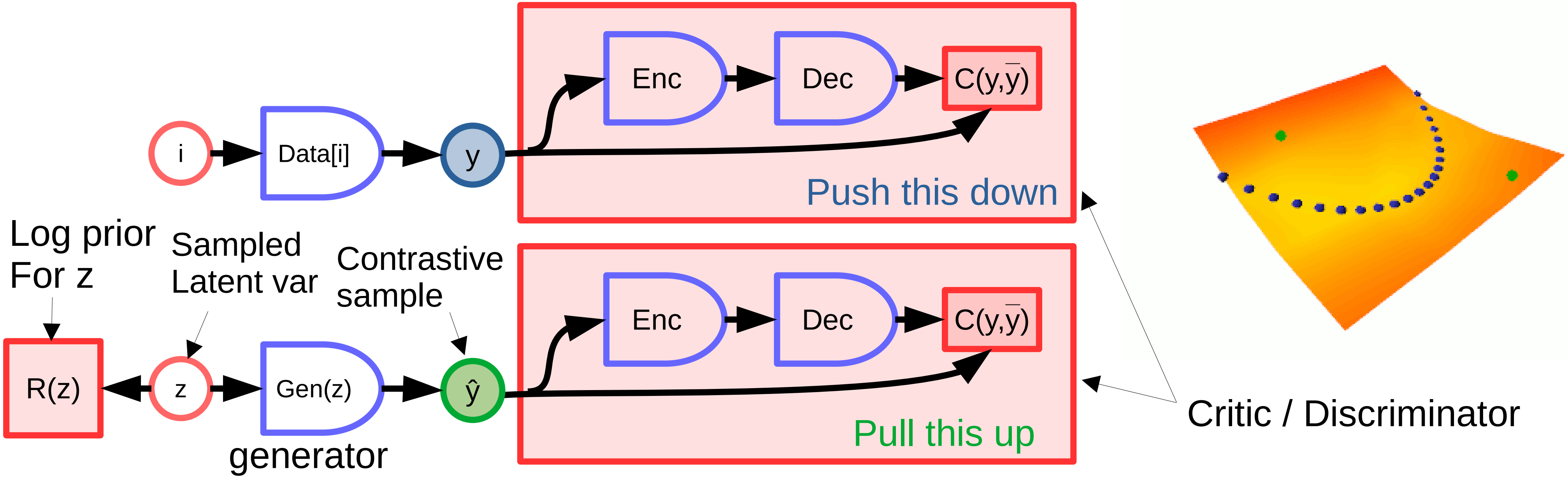
    - Where H must be an increasing fn of the data energies and decreasing fn of the contrastive point energies within the margin.
    - M is a margin matrix for all pairs of y and ŷ in the group.
  - Example: Neighborhood Component Analysis, Noise Contrastive Estimation, InfoNCE (implicit infinite margin) [Goldberger 2005] [Gutmann 2010]... [Misra 2019] [Chen 2020]

$$\mathcal{L}(x, y, \hat{y_1}, \dots \hat{y}_{p^-}, w) = -\log \frac{e^{-F_w(x,y)}}{e^{-F_w(x,y)} + \sum_{i=1}^{p^-} e^{-F_w(x,\hat{y}_i,w)}}$$

GAN is secretly a contrastive method for EBM

- **Energy-Based GAN** [Zhao 2016], **Wasserstein GAN** [Arjovsky 2017],...
  - The generator in a GAN generates contrastive samples for the critic.
  - But GANs have not been successful for learning representations of images

$$\mathcal{L}(x, y, \hat{y}, w) = H(F_w(x, y), F_w(x, \hat{y}), m(y, \hat{y}))$$



FACEBOOK AI

Maximum Likelihood as a special case of Contrastive Method

Push down on the energy of training samples
Pull up the energy of everything else to infinity

Refresher on turning energies to probabilities

- **Gibbs distribution (a.k.a. softmax, should be called softargmax)**

- **Discrete / Continuous**

$$P_w(y) = \frac{e^{-\beta F_w(y)}}{\sum_{y'} e^{-\beta F_w(y')}} \qquad P_w(y) = \frac{e^{-\beta F_w(y)}}{\int_{y'} e^{-\beta F_w(y')}}$$

Partition function

Inverse temperature

- **Joint distribution**

$$P_w(y, z) = \frac{e^{-\beta E_w(y,z)}}{\int_{y'} \int_{z'} e^{-\beta E_w(y',z')}}$$

- **Conditional distribution**

$$P_w(y, z|x) = \frac{e^{-\beta E_w(x,y,z)}}{\int_{y'} \int_{z'} e^{-\beta E_w(x,y',z')}}$$

- **Marginal distribution**

$$P_w(y|x) = \int_{z'} P_w(y, z'|x) = \frac{\int_{z'} e^{-\beta E_w(x,y,z')}}{\int_{y'} \int_{z'} e^{-\beta E_w(x,y',z')}}$$

Refresher on turning energies to probabilities

- **Joint distribution**

$$P_w(y,z) = \frac{e^{-\beta E_w(y,z)}}{\int_{y'}\int_{z'} e^{-\beta E_w(y',z')}}$$

- **Conditional distribution**

$$P_w(y|z) = \frac{e^{-\beta E_w(y,z)}}{\int_{y'} e^{-\beta E_w(y',z)}}$$

- **Marginal distribution**

$$P_w(z) = \frac{\int_{y'} e^{-\beta E_w(y',z)}}{\int_{z'}\int_{y'} e^{-\beta E_w(y',z')}}$$

- **Bayes rules!**

$$P_w(y,z) = P_w(y|z)P_w(z) = P_w(z|y)P_w(y)$$

Negative log-likelihood loss

$$L(x,y,w) = -\frac{1}{\beta}\log P_w(y|x) = F_w(x,y) + \frac{1}{\beta}\log\left[\int_{y'} e^{-\beta F_w(x,y')}\right]$$

Minus log partition function.
Like a free energy over y.

- **Gradient of log partition function**

$$\frac{\partial\left[-\frac{1}{\beta}\log\left[\int_{y'} e^{-\beta F_w(x,y')}\right]\right]}{\partial w} = \int_{y'} P_w(y'|x)\frac{\partial F_w(x,y')}{\partial w}$$

- **Monte Carlo methods: sample y from P(y|x)**
  - The integral is an expectation of the gradient over the distribution of y
  - Sample y from the distribution and average the corresponding gradients.

Max Likelihood is (generally) a (bad) Contrastive Method

- **Push down on data points,**
- **Push up on all points**
- **Max likelihood / probabilistic models**

$$P_w(y|x) = \frac{e^{-\beta F_w(x,y)}}{\int_{y'} e^{-\beta F_w(x,y')}}$$

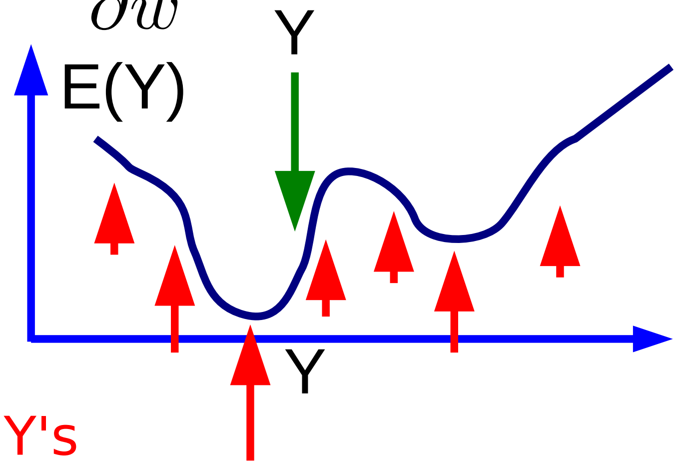
- Loss: $$\mathcal{L}(x,y,w) = F_w(x,y) + \frac{1}{\beta} \log \int_{y'} e^{-\beta F_w(x,y')}$$
- Gradient: $$\frac{\partial \mathcal{L}(x,y,w)}{\partial w} = \frac{\partial F_w(x,y)}{\partial w} - \int_{y'} P_w(y'|x) \frac{\partial F_w(x,y')}{\partial w}$$
- 2nd term is intractable: MC/MCMC/HMC/CD: $\hat{y}$ sampled from $P_w(y|x)$

$$\frac{\partial \mathcal{L}(x,y,w)}{\partial w} = \frac{\partial F_w(x,y)}{\partial w} - \frac{\partial F_w(x,\hat{y})}{\partial w}$$

push down of the energy of data points, push up everywhere else

Gradient of the negative log-likelihood loss for one sample Y:

$$\frac{\partial \mathcal{L}(x,y,w)}{\partial w} = \frac{\partial F_w(x,y)}{\partial w} - \int_{y'} P_w(y'|x) \frac{\partial F_w(x,y')}{\partial w}$$



Gradient descent:

$$w \leftarrow w - \eta \frac{\partial \mathcal{L}(x,y,w)}{\partial w}$$

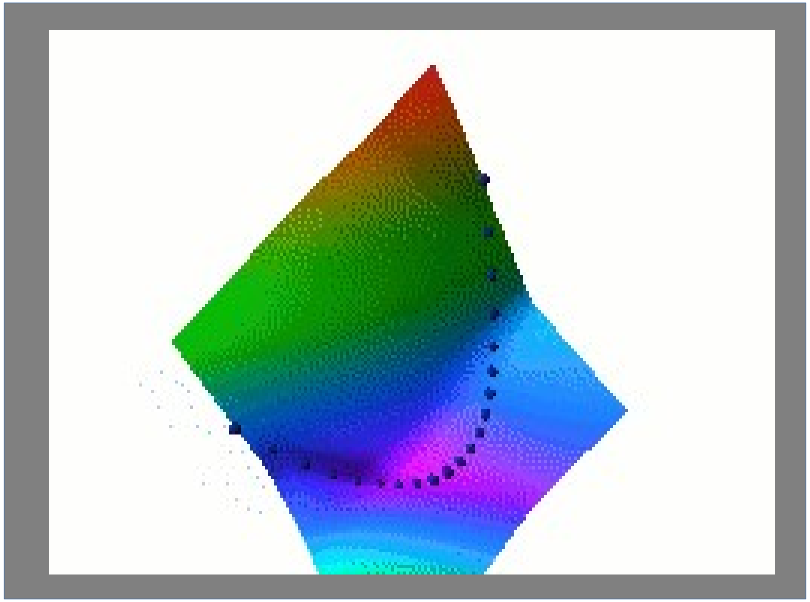
Pushes down on the energy of the samples

Pulls up on the energy of low-energy Y's

$$w \leftarrow w - \eta \frac{\partial F_w(x,y)}{\partial w} + \eta \int_{y'} P_w(y'|x) \frac{\partial F_w(x,y')}{\partial w}$$

Problem with Max Likelihood / Probabilistic Methods

- **It wants to make the difference between the energy on the data manifold and the energy just outside of it infinitely large!**
- **It wants to make the data manifold an infinitely deep and infinitely narrow canyon.**
- **The loss must be regularized to keep the energy smooth**
  - e.g. with Bayesian prior or by limiting weight sizes à la Wasserstein GAN.
  - So that gradient-based inference works
  - Equivalent to a Bayesian prior
  - But then, why use a probabilistic model?



Latent Variable Energy-Based Models

Minimize or marginalize the energy with respect to the latent variable

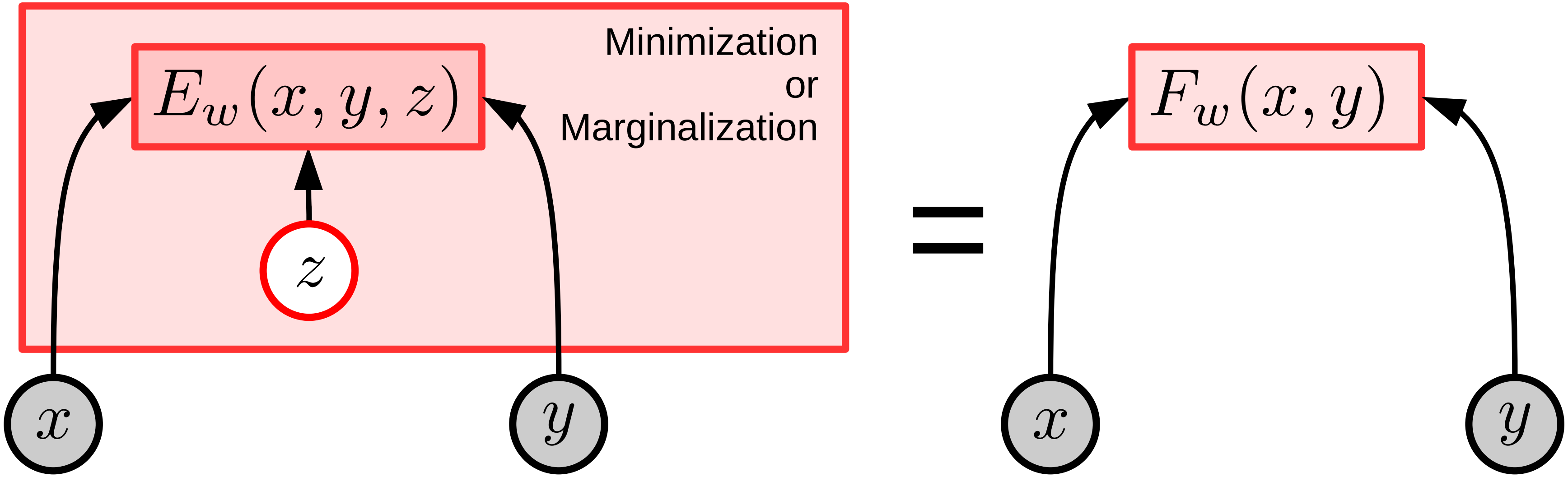
Latent-Variable Energy-Based Model

- **Zero temperature limit**

$$\check{z} = \text{argmin}_{z \in \mathcal{Z}} E_w(x, y, z) \qquad F_w(x, y) = E_w(x, y, \check{z})$$

- **Marginalization**

$$F_w(x, y) = -\frac{1}{\beta} \log \left[\int_{z'} e^{-\beta E_w(x, y, z')} \right]$$



Marginalization for Latent-Variable EBM

- **Gibbs distribution (a.k.a. softmax, should be called softargmax)**

- **Marginalization over latent variable z**

$$P_w(y|x) = \frac{\int_{z'} e^{-\beta E_w(x,y,z')}}{\int_{y'} \int_{z'} e^{-\beta E_w(x,y',z')}}$$

- **Definition of free energy over latent variable z**

$$F_w(x,y) = -\frac{1}{\beta} \log \left[\int_{z'} e^{-\beta E_w(x,y,z')} \right]$$

- **Marginal distribution = Gibbs formula with free energy**

$$P_w(y|x) = \frac{e^{-\beta F_w(x,y)}}{\int_{y'} e^{-\beta F_w(x,y')}}$$

Marginalizing over a latent variable

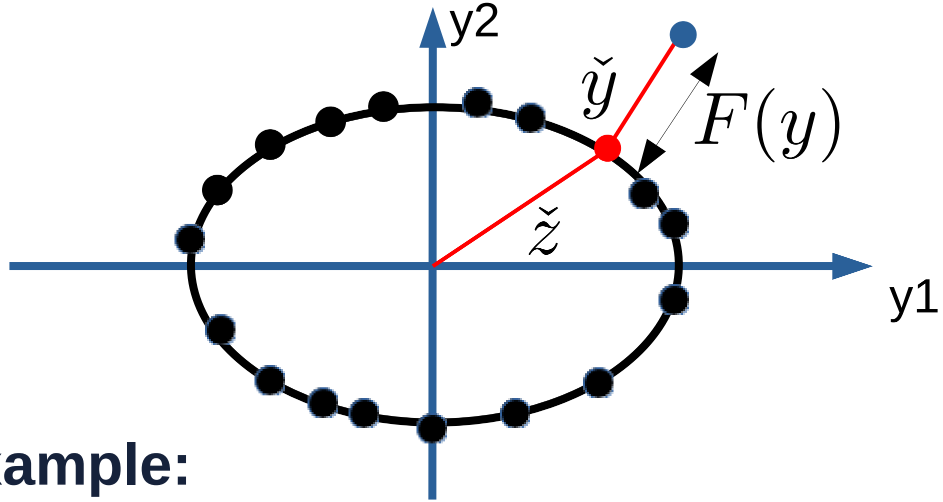
$$P(y,z|x) = \frac{e^{-\beta E(x,y,z)}}{\int_y \int_z e^{-\beta E(x,y,z)}} \qquad P(y|x) = \int_z P(y,z|x)$$

$$P(y|x) = \frac{\int_z e^{-\beta E(x,y,z)}}{\int_y \int_z e^{-\beta E(x,y,z)}} = \frac{e^{-\beta\left[-\frac{1}{\beta}\log\int_z e^{-\beta E(x,y,z)}\right]}}{\int_y e^{-\beta\left[-\frac{1}{\beta}\log\int_z e^{-\beta E(x,y,z)}\right]}} = \frac{e^{-\beta F_\beta(x,y)}}{\int_y e^{\beta F_\beta(x,y)}}$$

- **Free energy F(x,y)** $\quad F_\beta(x,y) = -\frac{1}{\beta}\log\int_z e^{-\beta E(x,y,z)}$

Inference with Latent Variable EBMs

- **The latent variable parameterizes the data manifold(s).**
- **The energy computes a distance to the learned manifold(s).**
- **The gradient of the energy points to the closest point on the data manifold(s).**



- **Example:**
  - learned manifold = ellipse
  - Latent variable = angle
  - Energy = distance of data point to ellipse
- **Model:**

$$E(y,z) = (y_1 - r_1 \sin(z))^2 + (y_2 - r_2 cos(z))^2$$

$$F(y) = \min_z (y_1 - r_1 \sin(z))^2 + (y_2 - r_2 cos(z))^2$$

FACEBOOK AI

Hopfield Nets and Boltzmann Machines

Simple concepts of historical relevance

Hopfield Nets (Hopfield 1982)

- **Energy-based model**
- **Fully-connected recurrent network with symmetric connections**
- **Binary activations**

$$E(y) = -\sum_{ij} y_i w_{ij} y_j$$

- **Inference: update neuron states with:**

$$y_i \leftarrow \mathrm{sign}\Big(\sum_j w_{ij} y_j\Big)$$

  - This makes the energy go down
- **Learning rule: minimize energy of training samples**
  - Dig holes around training samples.
  - no contrastive term! which is why it doesn't work very well

$$L(y, w) = E(y) \qquad \frac{\partial L(y,w)}{\partial w_{ij}} = -y_i y_j \qquad w_{ij} \leftarrow w_{ij} + y_i y_j$$

Boltzmann Machine [Hinton & Sejnowski 1983]

- **A Hopfield net with hidden units**

$$E(y,z) = -\sum_{ij} y_i w_{ij}^{yy} y_j - \sum_{ij} z_i w_{ij}^{zz} z_j - \sum_{ij} y_i w_{ij}^{yz} z_j$$

- **Free energy: marginalizes over z** $F(y) = -\log \sum_z \exp(-E(y,z))$

- **Loss: Negative log-likelihood (with MCMC contrastive samples)**

$$L(y,w) = F_w(y) + \log \sum_{y'} \exp(F_w(y'))$$

$$\frac{\partial L(y,w)}{\partial w_{ij}} = \frac{\partial F_w(y)}{\partial w} - \sum_{y'} P(y') \frac{\partial F_w(y')}{\partial w}$$ with $$P(y) = \frac{\exp(-F_w(y))}{\sum_{y'} \exp(-E_w(y'))}$$

Boltzmann Machine [Hinton & Sejnowski 1983]

- **But how do we marginalize on z? → MCMC sampling**
  - MCMC sampling: on z for first term; on z and y for second term

$$L(y,w) = F_w(y) + \log \sum_{y'} \exp(F_w(y')) \qquad F(y) = -\log \sum_z \exp(-E(y,z))$$

$$\frac{\partial L(y,w)}{\partial w_{ij}} = \sum_{y',z'} P(z'|y) \frac{\partial E_w(y,z')}{\partial w} - \sum_{y',z'} P(y',z') \frac{\partial E_w(y',z)}{\partial w}$$

with

$$P(z|y) = \frac{\exp(-E_w(y,z))}{\sum_{z'} \exp(-E_w(y,z'))} \qquad P(y,z) = \frac{\exp(-E_w(y,z))}{\sum_{y',z'} \exp(-E_w(y',z'))}$$

$\check{z}$ sampled from $P(z|y)$ $\qquad$ $\hat{y}, \hat{z}$ sampled from $P(y,z)$

$$\frac{\partial L(y,w)}{\partial w_{ij}^{yz}} \approx -y_i \check{z}_j + \hat{y}_i \hat{z}_j \qquad w_{ij} \leftarrow w_{ij} + \eta(y_i z_j - \hat{y}_i \hat{z}_j)$$

Boltzmann Machine [Hinton & Sejnowski 1983]

- **But how do we get MCMC samples of z (and y)?**

$\check{z}$ sampled from $P(z|y)$

$$P(z|y) = \frac{\exp(-E_w(y,z))}{\sum_{z'} \exp(-E_w(y,z'))}$$

$\hat{y}, \hat{z}$ sampled from $P(y,z)$

$$P(y,z) = \frac{\exp(-E_w(y,z))}{\sum_{y',z'} \exp(-E_w(y',z'))}$$

$$E(y,z) = -\sum_{ij} y_i w_{ij}^{yy} y_j - \sum_{ij} z_i w_{ij}^{zz} z_j - \sum_{ij} z_i w_{ij}^{yz} y_j$$

$$E(y,z)_{z_i=1} - E(y,z)_{z_i=0} = -\sum_j 1 w_{ij}^{yz} y_j + \sum_j 0 w_{ij}^{yz} y_j = \sum_j w_{ij}^{yz} y_j$$

$$\check{z}_i \text{ sampled from } P(z_i|y, z_{i \neq i}) = \frac{1}{1 + \exp(-\sum_j w_{ij}^{yz} y_i)}$$

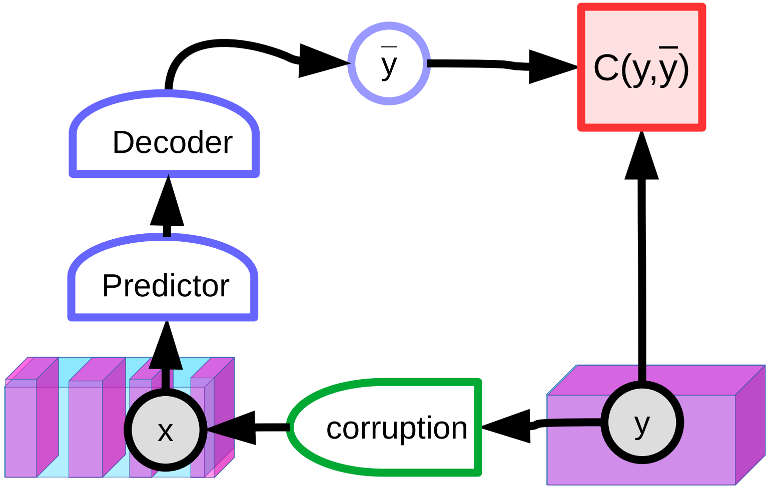
FACEBOOK AI

Denoising Auto-Encoders
Masked Auto-Encoders

Many practical applications

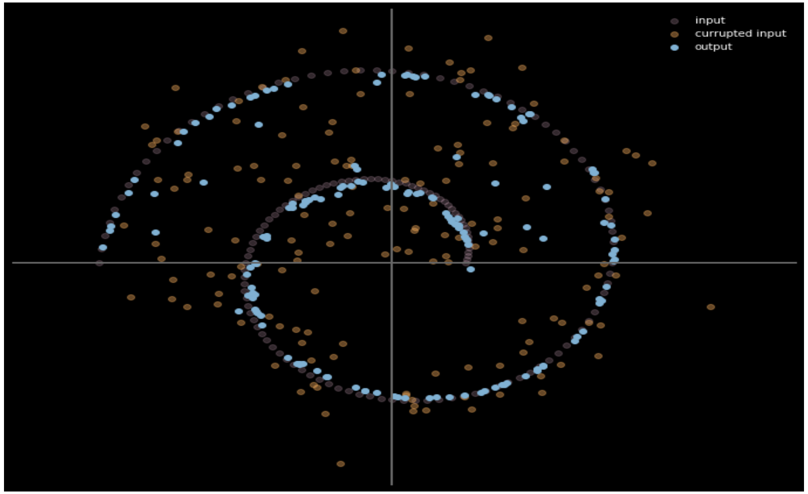
Contrastive Methods in NLP / Denoising AE / Masked AE

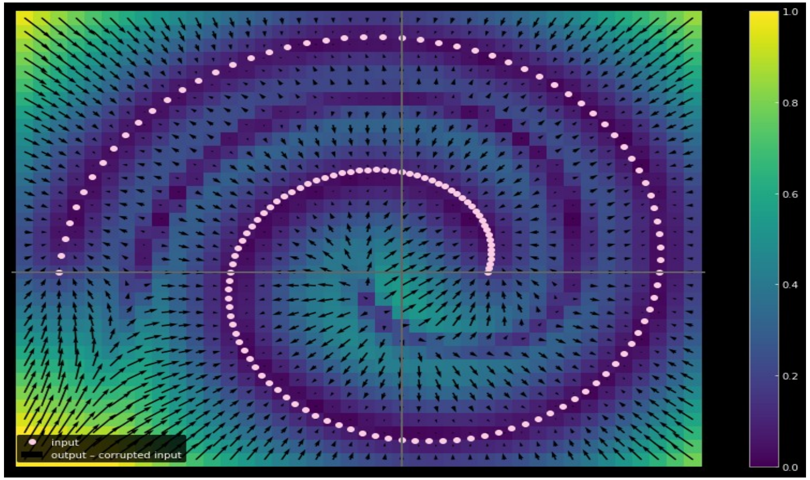
- **Contrastive method for NLP**
  - [Collobert-Weston 2011]
- **Denoising AE** [Vincent 2008]
- **Masked AE: Learning text representations**
  - BERT [Devlin 2018], RoBERTa [Ott 2019]



This is a [...] of text extracted [...] a large set of [...] articles

This is a piece of text extracted from a large set of news articles





Figures: Alfredo Canziani

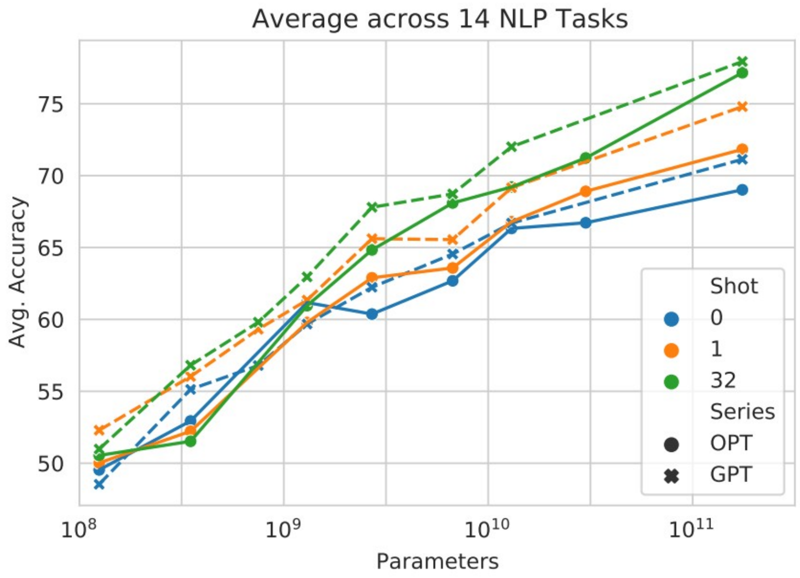
OPT-175B: language model with 175 billion parameters

- **Open source large language model from FAIR**
  - Paper: [Zhang, Roller, Goyal et al. ArXiv:2205.01068, 2022]
  - Github: https://github.com/facebookresearch/metaseq
  - Pre-trained on 180 billion tokens, 1000 GPUs (Nvidia A100)
  - Pre-trained models: 125M to 30B parameters (175B requires application)

Setup	Davinci	OPT-175B
Zero-shot	.628	**.667**
One-shot	.616	**.713**
Few-shot (binary)	.354	**.759**
Few-shot (multiclass)	.672	**.812**

Hate speech (F1)
NLP tasks
Dialog

Model	Eval	Perplexity (↓) C2	WW	ED	BST	WoI	Unigram F1 (↑) C2	WW	ED	BST	WoI
Reddit 2.7B	Unsup.	18.9	21.0	11.6	17.4	18.0	.126	.133	.135	.133	.124
BlenderBot 1	Sup.	**10.2**	12.5	**9.0**	11.9	14.7	.183	.189	.192	.178	.154
R2C2 BlenderBot	Sup.	10.5	**12.4**	9.1	**11.7**	14.6	**.205**	**.198**	**.197**	**.186**	**.160**
OPT-175B	Unsup.	10.8	13.3	10.3	12.1	**12.0**	.185	.152	.149	.162	.147



Text generation eith OPT-175B

Human-provided Prompt (bold)

Generated text

1. Introduction

In recent years, deep neural networks have led to a series of breakthroughs in a variety of domains, such as image classification and natural language understanding. In many of these works, network depth and increased model capacity seem to be critical in pushing state-of-the-art forward. In this paper, we attempt to understand what it means for a deep neural network to have a high capacity, and how to quantify it.

We introduce the notion of network capacity as the upper bound on the complexity of a neural network. We define the complexity of a neural network as the number of parameters and the number of connections, and show that the complexity of a neural network is proportional to the number of parameters, and the network capacity. We define the network capacity of a neural network as the maximum possible number of parameters that the network can have and still be able to accurately reproduce the training data. We then introduce a new measure of network capacity called the capacity-to-data (C2D) ratio, which is the ratio between the maximum number of parameters that the network can have and still be able to accurately reproduce the training data and the number of parameters of the network. We show that the C2D ratio is a good measure of network capacity, and that it is useful for comparing different neural networks. We introduce a new network compression technique called sparsity-promoting compression, which reduces the number of parameters of a neural network, while preserving its accuracy. We apply the sparsity-promoting compression technique to several datasets, and show that it can reduce the number of parameters of a neural network by up to 70%. We also show that sparsity-promoting compression can significantly improve the C2D ratio of the neural network. We then apply the sparsity-promoting compression technique to several image classification datasets, and show that it can significantly improve the accuracy of the neural network. We finally show that the sparsity-promoting compression technique can significantly reduce the memory consumption of a neural network, and that it can be used to reduce the memory consumption of deep neural networks.

Example of dialog

Human-provided Prompt (bold)

Generated text

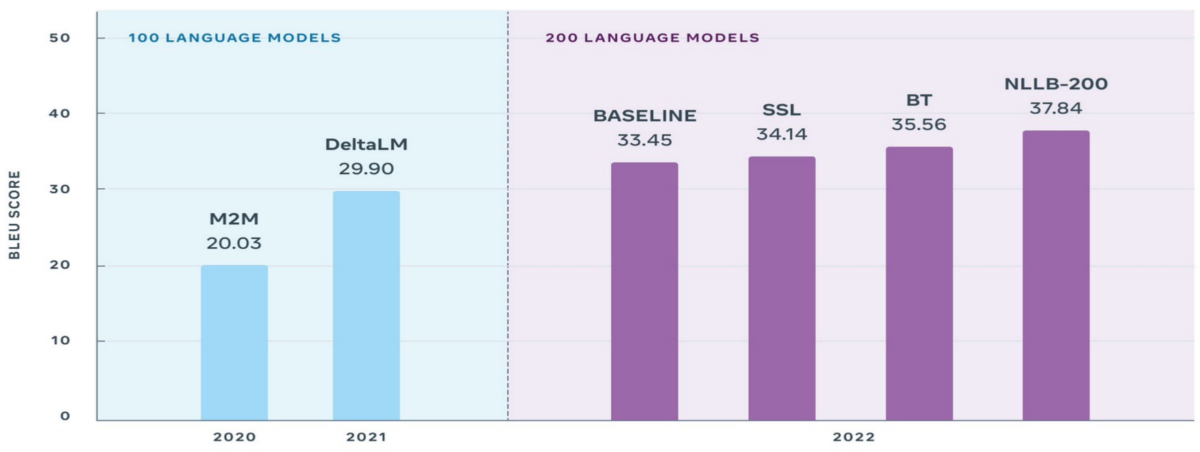
A chat between a curious human and the Statue of Liberty.

Human: What is your name?
Statue: I am the Statue of Liberty.
Human: Where do you live?
Statue: New York City.
Human: How long have you lived there?
Statue: I have been there since 1886.
Human: What is your job?
Statue: I was built to welcome immigrants to the United States.
Human: What is your favorite food?
Statue: I like ice cream and apples.
Human: What is your favorite color?
Statue: I like red, white, and blue.
Human: What is your favorite animal?
Statue: I like dogs.
Human: What is your favorite song?

No Language Left Behind (NLLB)

- **Language translation between 202 languages**
  - in any of the 40602 directions
  - Training set: 18 billion pairs of sentences for 2440 language directions
  - Most pairs have less than 1 million sentences
  - https://ai.facebook.com/research/no-language-left-behind/
- **A single neural net with 54 billion parameters**
- **Performance gets better as more languages are added**
- **Relies on Self-Supervised Learning and back-translation.**

Comparison of NLLB-200 with existing SOTA



No Language Left Behind (NLLB)

Acehnese
Acehnese
Mesopotamian Arabic
Ta'izzi-Adeni Arabic
Tunisian Arabic
Afrikaans
South Levantine Arabic
Akan
Amharic
North Levantine Arabic
Modern Standard Arabic
Modern Standard Arabic
Najdi Arabic
Moroccan Arabic
Egyptian Arabic
Assamese
Asturian
Awadhi
Central Aymara
South Azerbaijani
North Azerbaijani
Bashkir
Bambara
Balinese
Belarusian
Bemba
Bengali
Bhojpuri
Banjar
Banjar
Standard Tibetan

Bosnian
Buginese
Bulgarian
Catalan
Cebuano
Czech
Chokwe
Central Kurdish
Crimean Tatar
Welsh
Danish
German
Southwestern Dinka
Dyula
Dzongkha
Greek
English
Esperanto
Estonian
Basque
Ewe
Faroese
Fijian
Finnish
Fon
French
Friulian
Nigerian Fulfulde
Scottish Gaelic

Irish
Galician
Guarani
Gujarati
Haitian Creole
Hausa
Hebrew
Hindi
Chhattisgarhi
Croatian
Hungarian
Armenian
Igbo
Ilocano
Indonesian
Icelandic
Italian
Javanese
Japanese
Kabyle
Jingpho
Kamba
Kannada
Kashmiri
Kashmiri
Georgian
Central Kanuri
Central Kanuri
Kazakh
Kabiyè
Kabuverdianu

Khmer
Kikuyu
Kinyarwanda
Kyrgyz
Kimbundu
Northern Kurdish
Kikongo
Korean
Lao
Ligurian
Limburgish
Lingala
Lithuanian
Lombard
Latgalian
Luxembourgish
Luba-Kasai
Ganda
Luo
Mizo
Standard Latvian
Magahi
Maithili
Malayalam
Marathi
Minangkabau
Minangkabau
Macedonian
Plateau Malagasy
Maltese

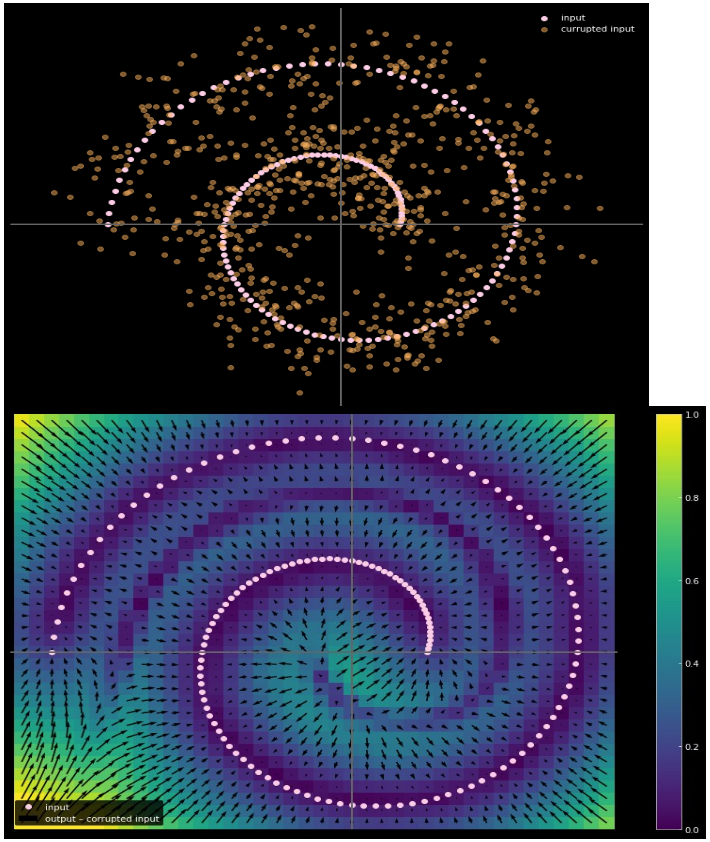
Meitei
Halh Mongolian
Mossi
Maori
Burmese
Dutch
Norwegian Nynorsk
Norwegian Bokmål
Nepali
Northern Sotho
Nuer
Nyanja
Occitan
West Central Oromo
Odia
Pangasinan
Eastern Panjabi
Papiamento
Western Persian
Polish
Portuguese
Dari
Southern Pashto
Ayacucho Quechua
Romanian
Rundi
Russian
Sango
Sanskrit
Santali
Sicilian
Shan
Sinhala
Slovak

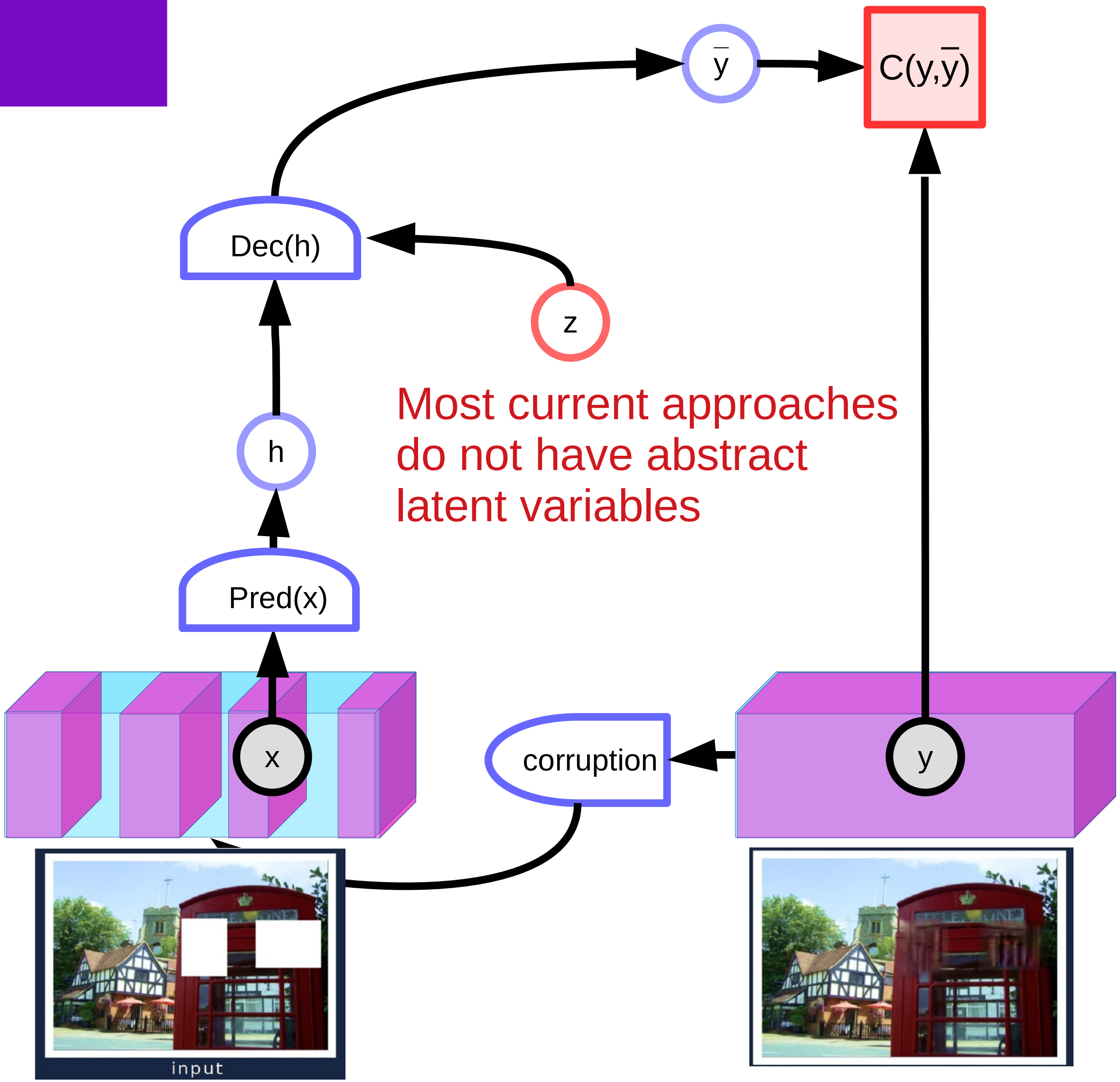
Slovenian
Samoan
Shona
Sindhi
Somali
Southern Sotho
Spanish
Tosk Albanian
Sardinian
Serbian
Swati
Sundanese
Swedish
Swahili
Silesian
Tamil
Tatar
Telugu
Tajik
Tagalog
Thai
Tigrinya
Tamasheq
Tamasheq
Tok Pisin
Tswana
Tsonga

Turkmen
Tumbuka
Turkish
Twi
Central Atlas Tamazight
Uyghur
Ukrainian
Umbundu
Urdu
Northern Uzbek
Venetian
Vietnamese
Waray
Wolof
Xhosa
Eastern Yiddish
Yoruba
Yue Chinese
Chinese
Chinese
Standard Malay
Zulu

Denoising AE in continuous domains?

- **Image inpainting** [Pathak 17]
- Latent variables? GAN?





FACEBOOK AI

Thank You!