Video: Samples from "cooking" subset of Kinetics, Weissenborn et al.

CMP784

DEEP LEARNING

Lecture #9 – Autoencoders and Autoregressive Models

HACETTEPE UNIVERSITY COMPUTER VISION LAB

Erkut Erdem // Hacettepe University // Fall 2021

Previously on CMP784

- Content-based attention
- Location-based attention
- Soft vs. hard attention
- Case study: Show, Attend and Tell
- Self-attention
- Case study: Transformer networks

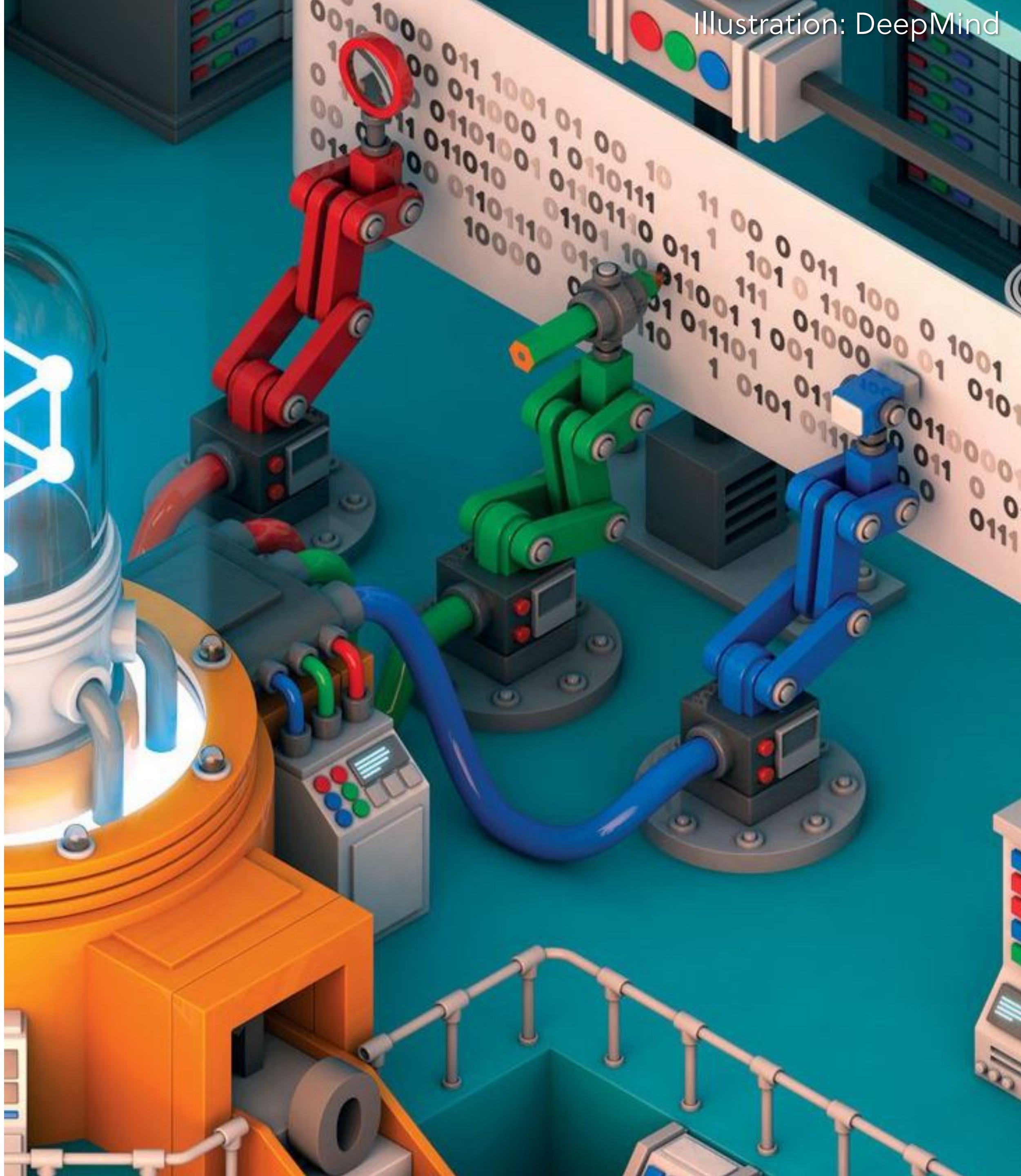


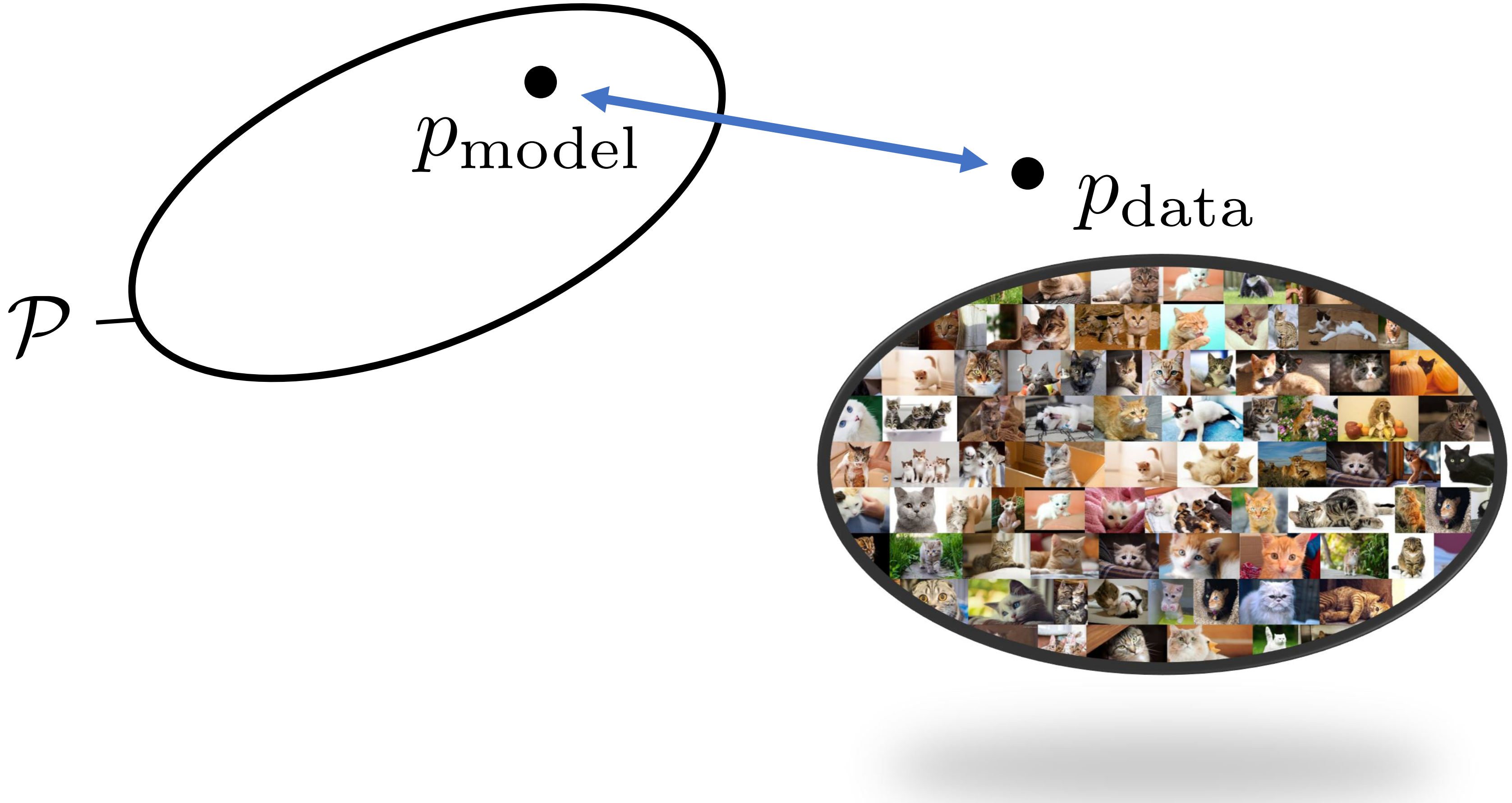
Illustration: DeepMind

Lecture overview

- Generative Modeling
- Basic Foundations
  - Sparse Coding
  - Autoencoders
- Autoregressive Generative Models

Disclaimer: Much of the material and slides for this lecture were borrowed from
—Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas' Berkeley CS294-158 class
—Ruslan Salakhutdinov's talk titled "Unsupervised Learning: Learning Deep Generative Models"
—Yoshua Bengio's IDT6266 class
—Bill Freeman, Antonio Torralba and Phillip Isola's MIT 6.869 class
—Nal Kalchbrenner's talks on "Generative Modelling as Sequence Learning" and "Generative Models of Language and Images"

Generative Modeling



- **Goal:** Learn some underlying hidden structure of the training samples to generate novel samples from same data distribution

Yann LeCun's Black Forest Cake

Pure Reinforcement Learning (cherry)
- The machine predicts a scalar reward given once in a while.
- A few bits for some samples

Supervised Learning (icing)
- The machine predicts a category or a few numbers for each input
- Predicting human-supplied data
- 10→10,000 bits per sample

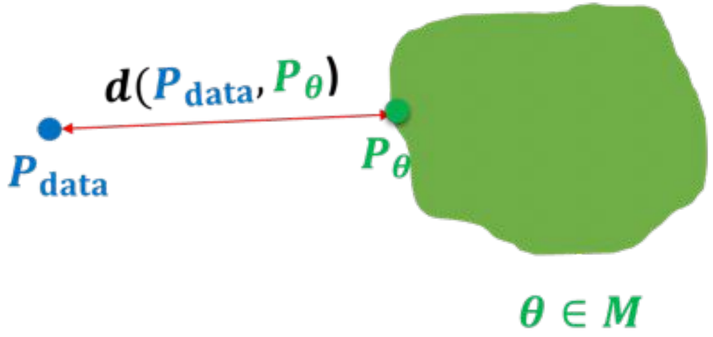
Unsupervised/ Predictive Learning (cake génoise)
- The machine predicts any part of its input for any observed part.
- Predicts future frames in videos
- **Millions of bits per sample**

"If intelligence is a cake, the bulk of the cake is unsupervised learning, the icing on the cake is supervised learning, and the cherry on the cake is reinforcement learning (RL)."

Learning a generative model

- We are given a training set of examples, e.g., images of dogs

$\mathbf{x}_i \sim P_{\text{data}}$
$i = 1, 2, \dots, n$



Model family

- We want to learn a probability distribution p(x) over images x s.t.
  - **Generation**: If we sample $x_{new} \sim p(x)$, x_{new} should look like a dog (sampling)
  - **Density estimation:** p(x) should be high if x looks like a dog, and low otherwise (anomaly detection)
  - **Unsupervised representation learning**: We should be able to learn what these images have in common, e.g., ears, tail, etc. (features)

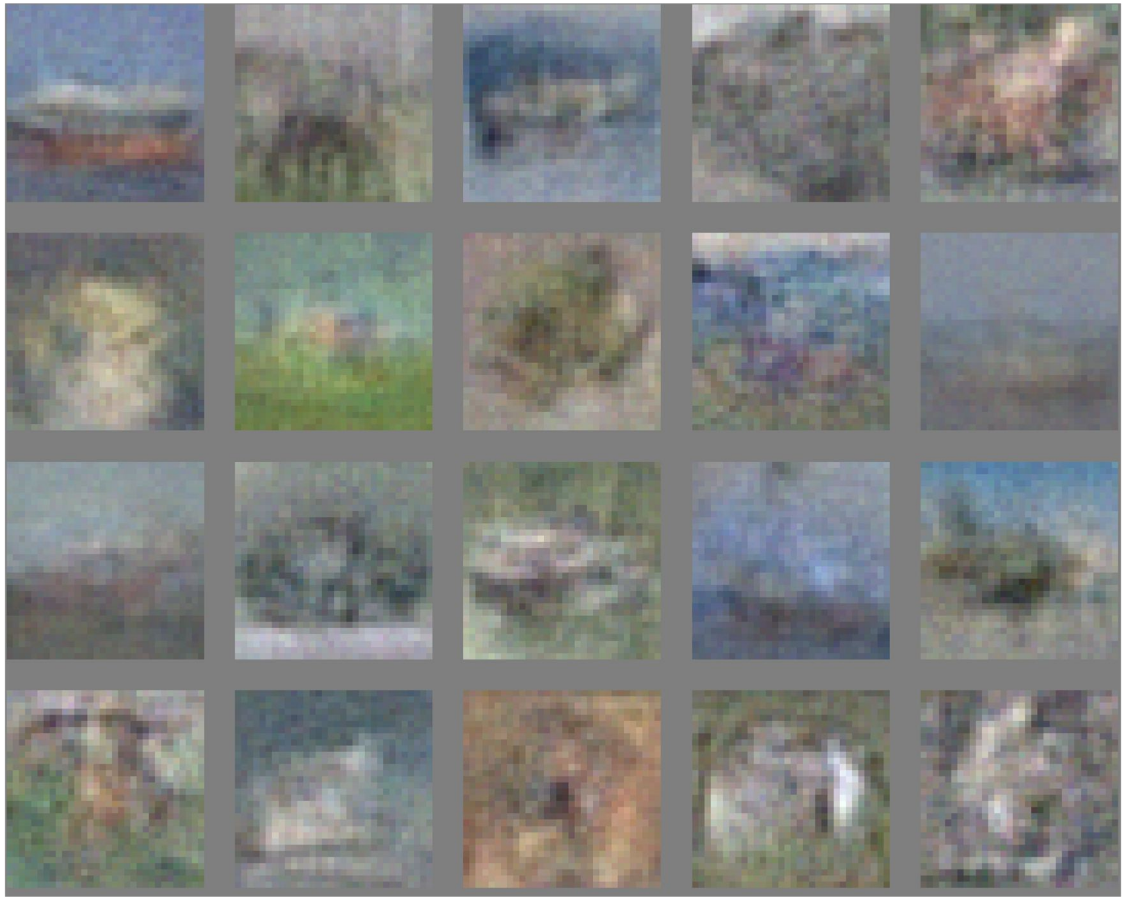
Generate Images

[Deep Belief Nets, Hinton, Osindero, Teh, 2006]

Generate Images

[VAE, Kingma and Welling, 2013]

Generate Images



[GAN, Goodfellow et al. 2014]

Generate Images

[DCGAN, Radford, Metz, Chintala 2015]

Generate Images

[DCGAN, Radford, Metz, Chintala 2015]

Generate Images

bicubic
(21.59dB/0.6423)

SRResNet
(23.53dB/0.7832)

SRGAN
(21.15dB/0.6868)

original

[Ledig, Theis, Huszar et al, 2017]

Generate Images

[CycleGAN: Zhu, Park, Isola & Efros, 2017]

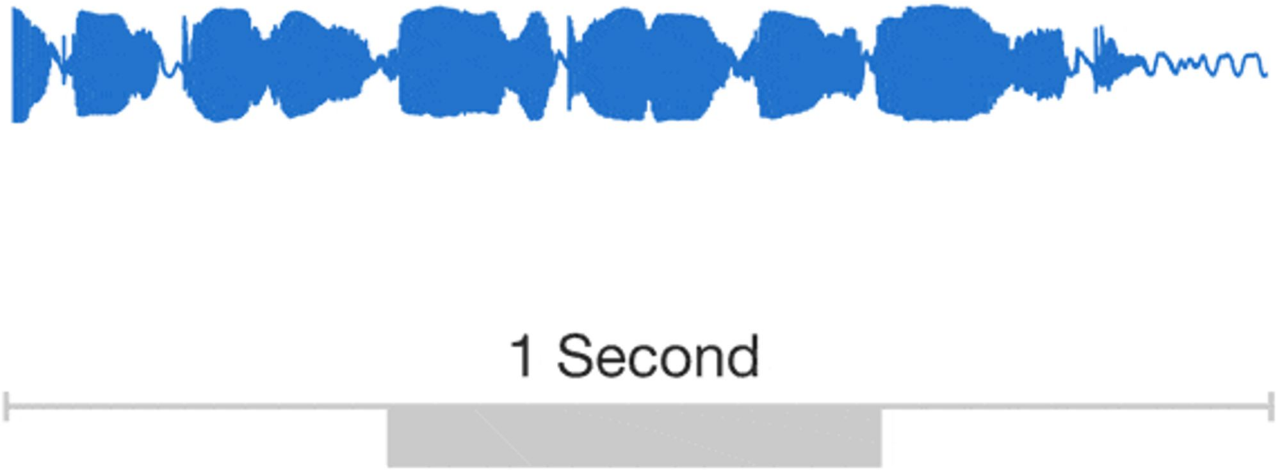
Generate Images

[BigGAN, Brock, Donahue, Simonyan, 2018]

Generate Images

[StyleGAN, Karras, Laine, Aila, 2018]

Generate Audio

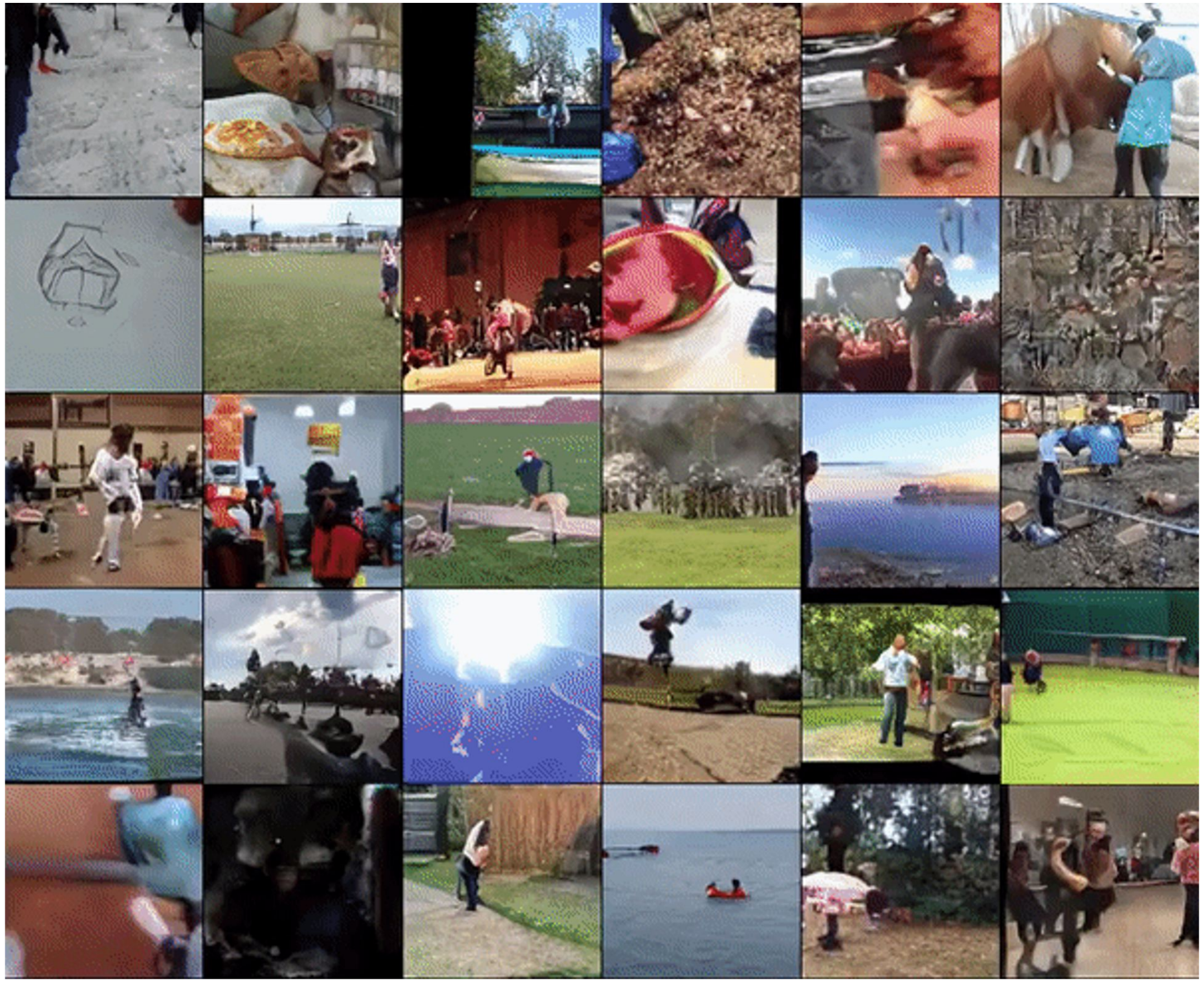


[Parametric](#)

[WaveNet](#)

[WaveNet, Oord et al., 2018]

Generate Video



DVD-GAN: Adversarial Video Generation on Complex Datasets, Clark, Donahue, Simonyan, 2019

Generate Text

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.
```

[Char-rnn, karpathy, 2015]

Generate Math

```
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on
$\mathcal{C}$.
\item Given a morphism $\Delta : \mathcal{F} \to \mathcal{I}$
is an injective and let $\mathfrak q$ be an abelian sheaf on
$X$.
Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a
category.
\begin{enumerate}
\item \hyperref[setain-construction-phantom]{Lemma}
\label{lemma-characterize-quasi-finite}
Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on
$\mathcal{C}$.
Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
$\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
\item The following are equivalent
\begin{enumerate}
\item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
\end{lemma}
```

For $\bigoplus_{n=1,\dots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the comparicoly in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

is an open subset of X. Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S.

Proof. See discussion of sheaves of sets. □

The result for prove any open covering follows from the less of Example ??. It may replace S by $X_{spaces,\acute{e}tale}$ which gives an open subspace of X and T equal to S_{Zar}, see Descent, Lemma ??. Namely, by Lemma ?? we see that R is geometrically regular over S.

[Char-rnn, karpathy, 2015]

Generate Text

Prompt [human written]: A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.

GPT-2: *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.

The Nuclear Regulatory Commission did not immediately release any information. According to the release, the U.S. Department of Energy's Office of Nuclear Material Safety and Security is leading that team's investigation.

'The safety of people, the environment and the nation's nuclear stockpile is our highest priority,' Hicks said. 'We will get to the bottom of this and make no excuses.'

[OpenAI's GPT2 -- Radford, Wu, Child, Luan, Amodei, Sutskever, 2019]

Generate Text

Prompt (human written): Recycling is good for the world, no, you could not be more wrong.

GPT-2: Recycling is NOT good for the world. It is bad for the environment, it is bad for our health, and it is bad for our economy. I’m not kidding. Recycling is not good for the environment. It is destructive to the earth and it is a major contributor to global warming. Recycling is not good for our health. It contributes to obesity and diseases like heart disease and cancer. Recycling is bad for our economy. It increases the cost of a product, and in turn, the price of everything that is made with that product. Recycling is not good for our nation. We pay a tremendous price for the privilege of having the world’s most advanced and efficient recycling system. Recycling is a huge, colossal waste of time, energy, money, and resources.

[OpenAI’s GPT2 -- Radford, Wu, Child, Luan, Amodei, Sutskever, 2019]

Try it yourself

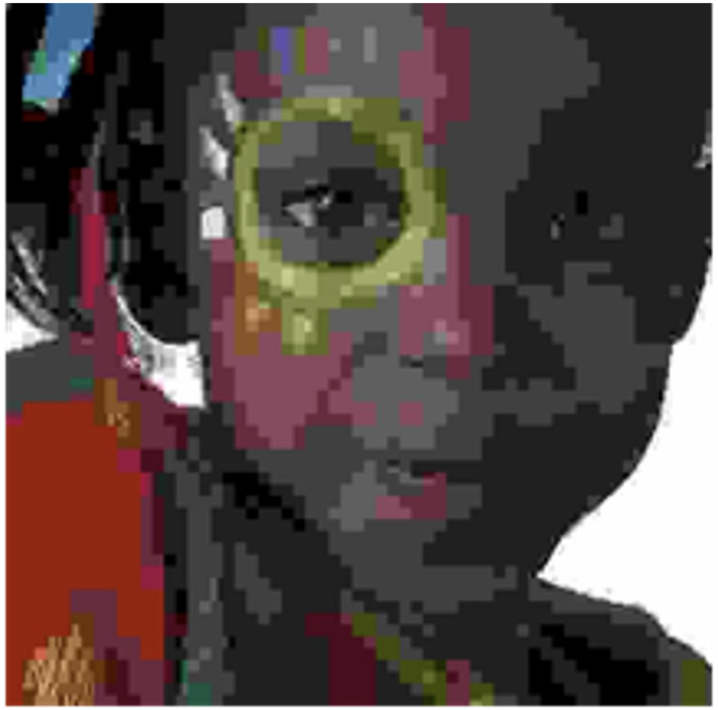
https://talktotransformer.com/

Compression - Lossless

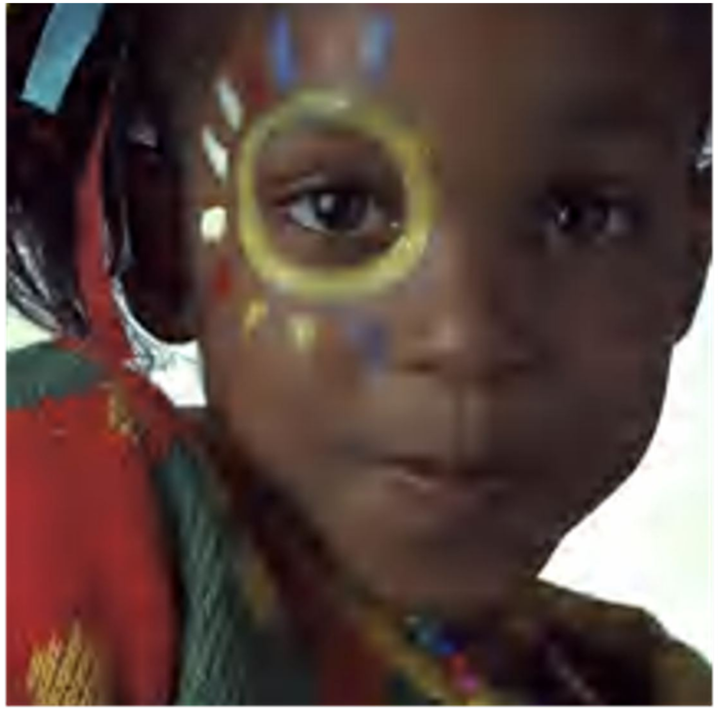
Model	Bits per byte
CIFAR-10	
PixelCNN (Oord et al., 2016)	3.03
PixelCNN++ (Salimans et al., 2017)	2.92
Image Transformer (Parmar et al., 2018)	2.90
PixelSNAIL (Chen et al., 2017)	2.85
Sparse Transformer 59M (strided)	**2.80**
Enwik8	
Deeper Self-Attention (Al-Rfou et al., 2018)	1.06
Transformer-XL 88M (Dai et al., 2018)	1.03
Transformer-XL 277M (Dai et al., 2018)	**0.99**
Sparse Transformer 95M (fixed)	**0.99**
ImageNet 64x64	
PixelCNN (Oord et al., 2016)	3.57
Parallel Multiscale (Reed et al., 2017)	3.7
Glow (Kingma & Dhariwal, 2018)	3.81
SPN 150M (Menick & Kalchbrenner, 2018)	3.52
Sparse Transformer 152M (strided)	**3.44**
Classical music, 5 seconds at 12 kHz	
Sparse Transformer 152M (strided)	**1.97**

Generative models provide better bit-rates than distribution-unaware compression methods like JPEG, etc.

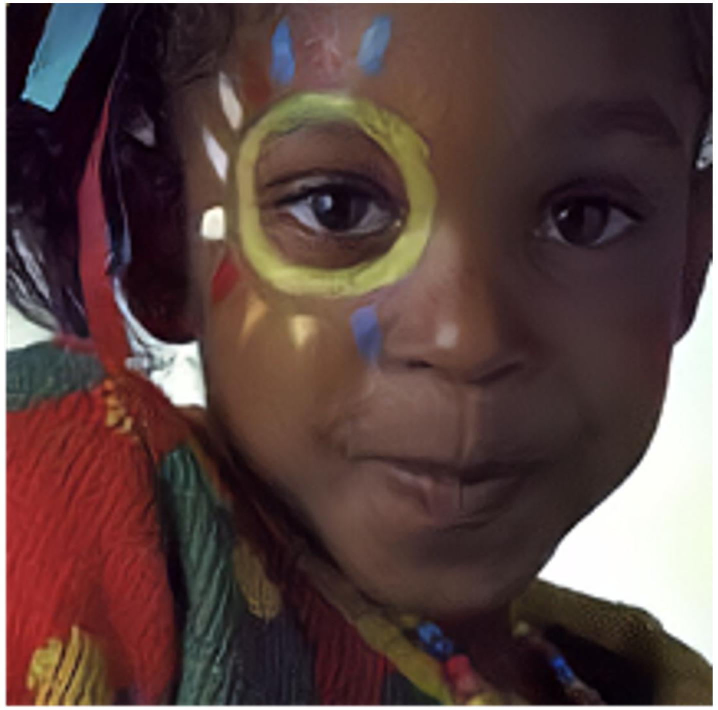
Compression - Lossy



JPEG



JPEG2000



WaveOne

[Rippel & Bourdev, 2017]

Downstream Task - Sentiment Detection

This is one of Crichton's best books. The characters of Karen Ross, Peter Elliot, Munro, and Amy are beautifully developed and their interactions are exciting, complex, and fast-paced throughout this impressive novel. And about 99.8 percent of that got lost in the film. Seriously, the screenplay AND the directing were horrendous and clearly done by people who could not fathom what was good about the novel. I can't fault the actors because frankly, they never had a chance to make this turkey live up to Crichton's original work. I know good novels, especially those with a science fiction edge, are hard to bring to the screen in a way that lives up to the original. But this may be the absolute worst disparity in quality between novel and screen adaptation ever. The book is really, really good. The movie is just dreadful.

[Radford et al., 2017]

Downstream Tasks - NLP (BERT Revolution)

Rank	Name	Model	URL	Score	BoolQ	CB	COPA	MultiRC	ReCoRD	RTE	WiC	WSC	AX-b	AX-g
1	ERNIE Team - Baidu	ERNIE 3.0		90.6	91.0	98.6/99.2	97.4	88.6/63.2	94.7/94.2	92.6	77.4	97.3	68.6	92.7/94.7
2	Zirui Wang	T5 + UDG, Single Model (Google Brain)		90.4	91.4	95.8/97.6	98.0	88.3/63.0	94.2/93.5	93.0	77.9	96.6	69.1	92.7/91.9
3	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4		90.3	90.4	95.7/97.6	98.4	88.2/63.7	94.5/94.1	93.2	77.5	95.9	66.7	93.3/93.8
4	SuperGLUE Human Baselines	SuperGLUE Human Baselines		89.8	89.0	95.8/98.9	100.0	81.8/51.9	91.7/91.3	93.6	80.0	100.0	76.6	99.3/99.7
5	T5 Team - Google	T5		89.3	91.2	93.9/96.8	94.8	88.1/63.3	94.1/93.4	92.5	76.9	93.8	65.6	92.7/91.9
6	Huawei Noah's Ark Lab	NEZHA-Plus		86.7	87.8	94.4/96.0	93.6	84.6/55.1	90.1/89.6	89.1	74.6	93.2	58.0	87.1/74.4
7	Alibaba PAI&ICBU	PAI Albert		86.1	88.1	92.4/96.4	91.8	84.6/54.7	89.0/88.3	88.8	74.1	93.2	75.6	98.3/99.2
8	Infosys : DAWN : AI Research	RoBERTa-iCETS		86.0	88.5	93.2/95.2	91.2	86.4/58.2	89.9/89.3	89.9	72.9	89.0	61.8	88.8/81.5
9	Tencent Jarvis Lab	RoBERTa (ensemble)		85.9	88.2	92.5/95.6	90.8	84.4/53.4	91.5/91.0	87.9	74.1	91.8	57.6	89.3/75.6
10	Zhuiyi Technology	RoBERTa-mtl-adv		85.7	87.1	92.4/95.6	91.2	85.1/54.3	91.7/91.3	88.1	72.1	91.8	58.5	91.0/78.1

[https://super.gluebenchmark.com/leaderboard]

Downstream Tasks - Vision (Contrastive)

Method	Architecture	mAP
Transfer from labeled data:		
Supervised baseline	ResNet-152	74.7
Transfer from unlabeled data:		
Exemplar [17] by [13]	ResNet-101	60.9
Motion Segmentation [47] by [13]	ResNet-101	61.1
Colorization [64] by [13]	ResNet-101	65.5
Relative Position [14] by [13]	ResNet-101	66.8
Multi-task [13]	ResNet-101	70.5
Instance Discrimination [60]	ResNet-50	65.4
Deep Cluster [7]	VGG-16	65.9
Deeper Cluster [8]	VGG-16	67.8
Local Aggregation [66]	ResNet-50	69.1
Momentum Contrast [25]	ResNet-50	74.9
Faster-RCNN trained on CPC v2	ResNet-161	**76.6**

"If, by the first day of autumn (Sept 23) of 2015, a method will exist that can match or beat the performance of R-CNN on Pascal VOC detection, without the use of any extra, human annotations (e.g. ImageNet) as pre-training, Mr. Malik promises to buy Mr. Efros one (1) gelato (2 scoops: one chocolate, one vanilla)."

Table: Data-Efficient Image Recognition using CPC

[Henaff, Srinivas, et al.]

Why Unsupervised Learning?

- Given high-dimensional data $X = (x_1, \ldots, x_n)$ we want to find a low-dimensional model characterizing the population.

- Recent progress mostly in supervised DL

- Real challenges for unsupervised DL

- Potential benefits:
  - **Exploit tons of unlabeled data**
  - Answer new questions about the variables observed
  - Regularizer – transfer learning – domain adaptation
  - Easier optimization (divide and conquer)
  - Joint (structured) outputs

Why Latent Factors & Unsupervised Representation Learning? Because of Causality.

- If Ys of interest are among the causal factors of X, then

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

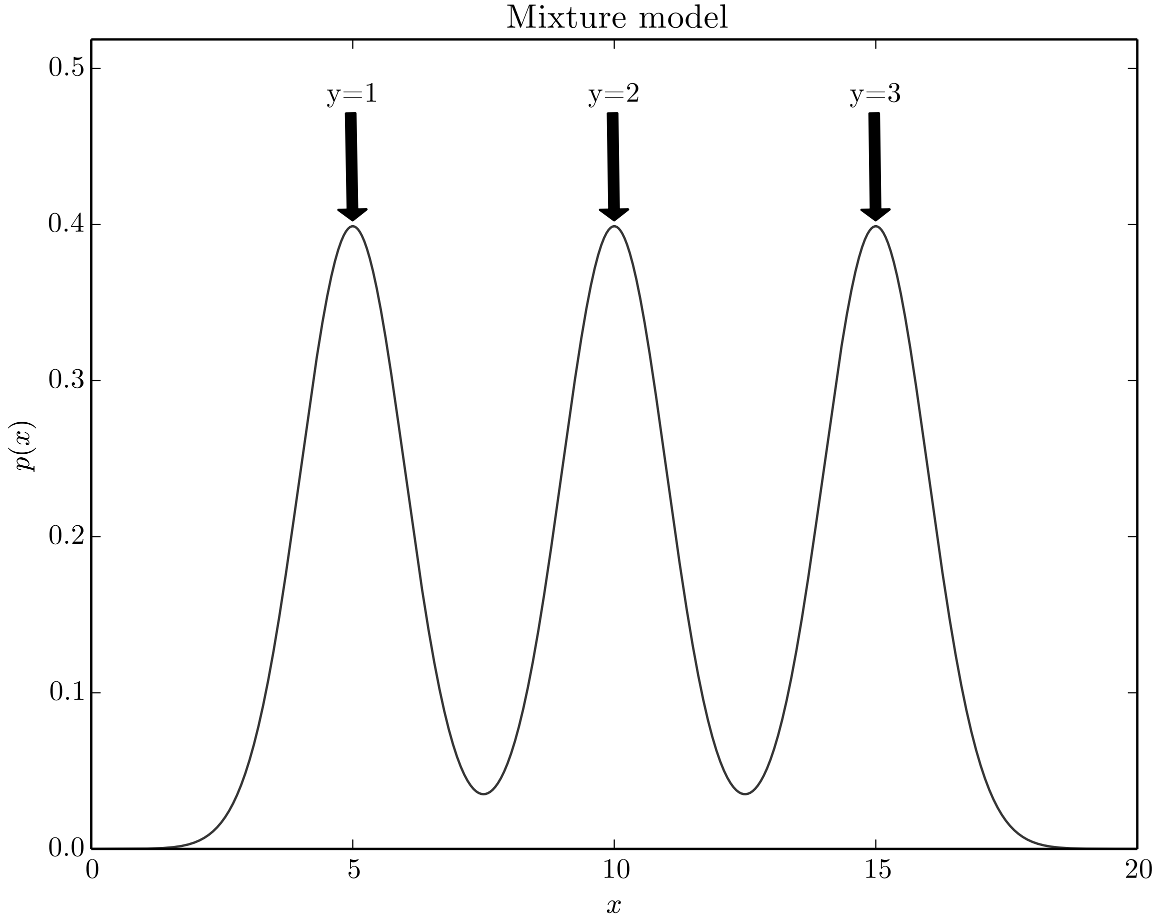
is tied to P(X) and P(X|Y), and P(X) is defined in terms of P(X|Y), i.e.

- The best possible model of X (unsupervised learning) MUST involve Y as a latent factor, implicitly or explicitly.
- Representation learning SEEKS the latent variables H that explain the variations of X, making it likely to also uncover Y.

On causal and anticausal learning, Janzing et al. ICML 2012

If Y is a Cause of X, Semi-Supervised Learning Works

- Just observing the x-density reveals the causes y (cluster ID)
- After learning p(x) as a mixture, a single labeled example per class suffices to learn p(y|x)



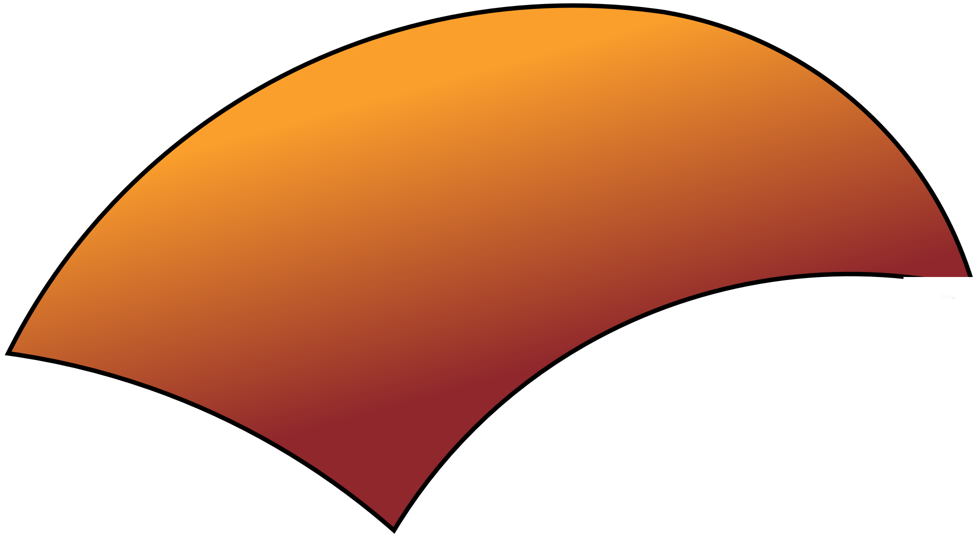
Invariance & Disentangling Underlying Factors

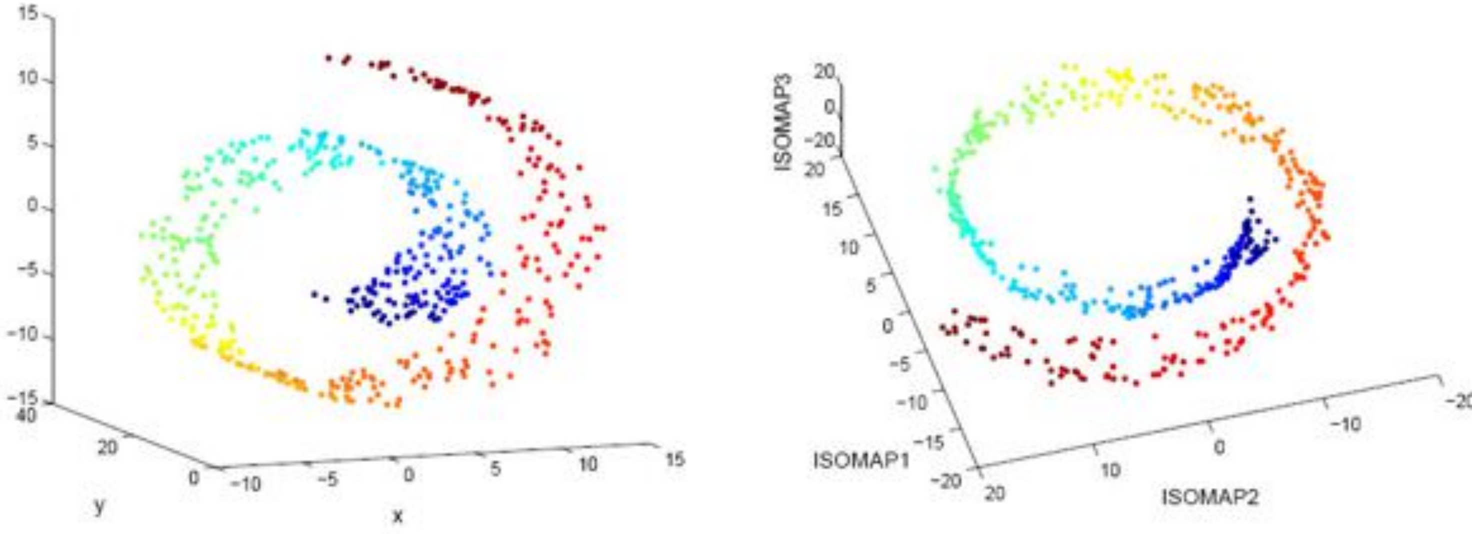
- Invariant features
- Which invariances?
- Alternative: learning to disentangle factors, i.e. keep all the explanatory factors in the representation
- Good disentangling ➞ avoid the curse of dimensionality
- Emerges from representation learning

(Goodfellow et al. 2009, Glorot et al. 2011)

Curse of Dimensionality

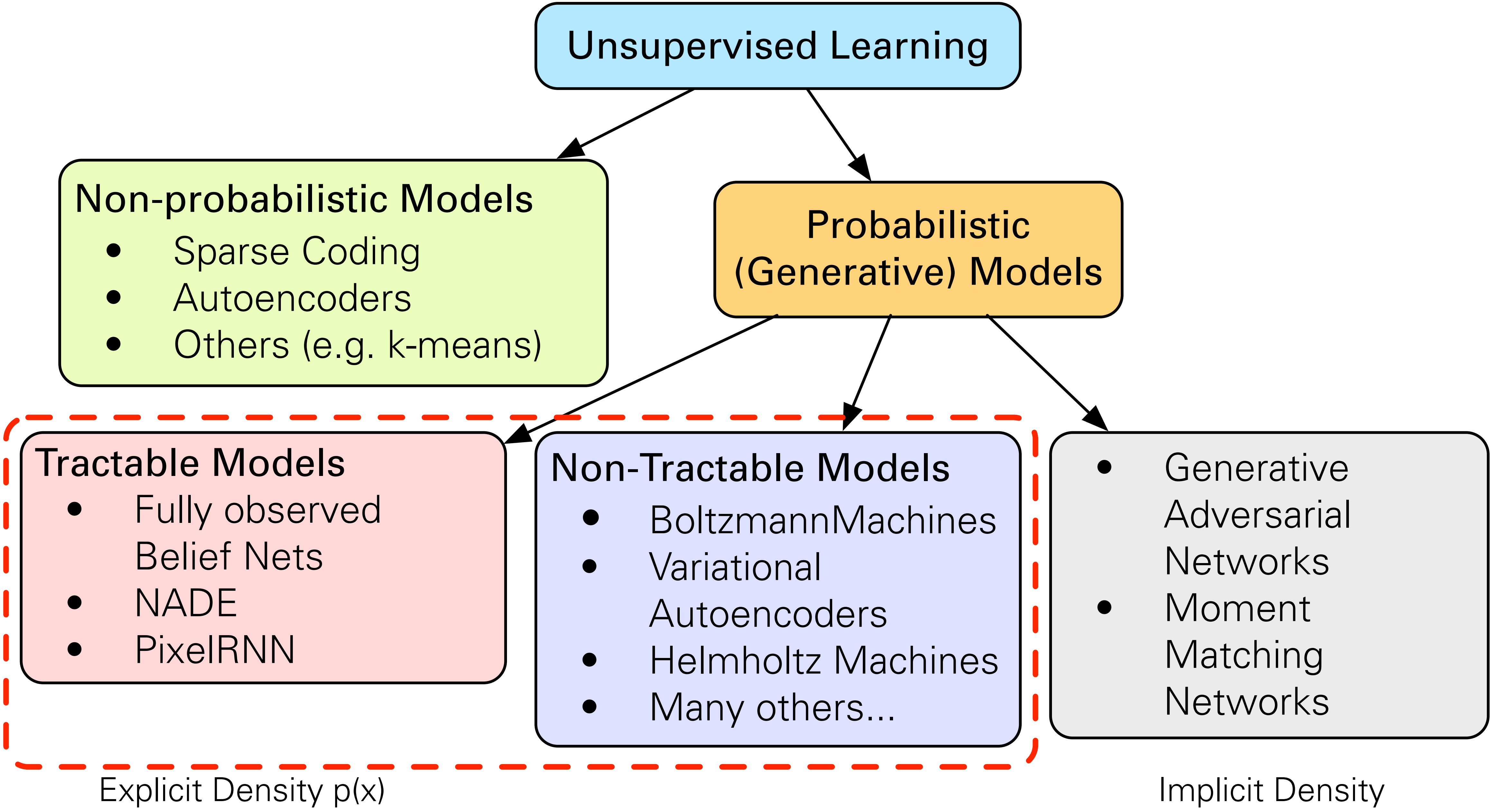
- Challenge: How to model $p(x)$, $x \in \mathbb{R}^N$ (or $x \in \Omega^N$) for large N?
- An existing hypothesis is that, although the ambient dimensionality is high, the intrinsic dimensionality of x is low.





(a) Swiss Roll (b) Isomap embedding

Slide credit: Joan Bruna


Unsupervised Learning
Non-probabilistic Models
Sparse Coding
Autoencoders
Others (e.g. k-means)
Probabilistic (Generative) Models
Tractable Models
Fully observed Belief Nets
NADE
PixelRNN
Non-Tractable Models
BoltzmannMachines
Variational Autoencoders
Helmholtz Machines
Many others...
Generative Adversarial Networks
Moment Matching Networks
Explicit Density p(x)
Implicit Density

Unsupervised Learning

- Basic Building Blocks:
  - Sparse Coding
  - Autoencoders
- Autoregressive Generative Models
- Generative Adversarial Networks
- Variational Autoencoders
- Normalizing Flow Models

Sparse Coding

- Sparse coding (Olshausen & Field, 1996). Originally developed to explain early visual processing in the brain (edge detection).

- **Objective:** Given a set of input data vectors $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, learn a dictionary of bases, such that:

$$\mathbf{x}_n = \sum_{k=1}^{K} a_{nk} \phi_k$$

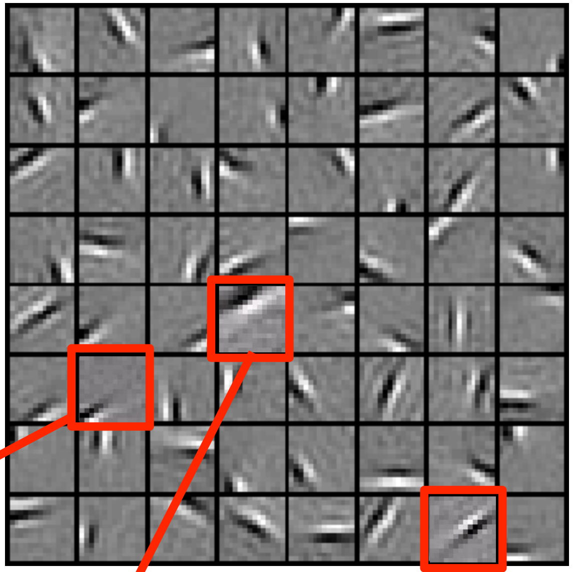
Sparse: mostly zeros

- Each data vector is represented as a sparse linear combination of bases.

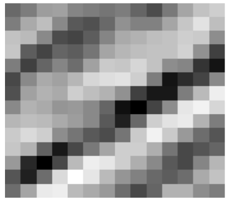
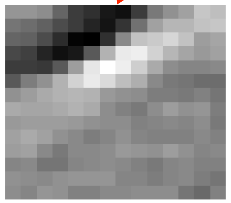
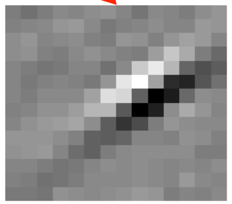
Sparse Coding

Natural Images

Learned bases: "Edges"



New example

 $= 0.8 *$  $+ 0.3 *$  $+ 0.5 *$ 

$$x = 0.8 * \phi_{36} + 0.3 * \phi_{42} + 0.5 * \phi_{65}$$

[0.0, 0.0, ... **0.8**, ..., **0.3**, ..., **0.5**, ...] = coefficients (feature representation)

Slide Credit: Honglak Lee

Sparse Coding: Training

- Input image patches: $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N \in \mathbb{R}^D$
- Learn dictionary of bases: $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_K \in \mathbb{R}^D$

$$\min_{\mathbf{a},\boldsymbol{\phi}} \underbrace{\sum_{n=1}^{N} \left\| \mathbf{x}_n - \sum_{k=1}^{K} a_{nk} \boldsymbol{\phi}_k \right\|_2^2}_{\text{Reconstruction error}} + \lambda \underbrace{\sum_{n=1}^{N} \sum_{k=1}^{K} |a_{nk}|}_{\text{Sparsity penalty}}$$

- Alternating Optimization:
  1. Fix dictionary of bases and solve for activations **a** (a standard Lasso problem).
  2. Fix activations **a**, optimize the dictionary of bases (convex QP problem).

Sparse Coding: Testing Time

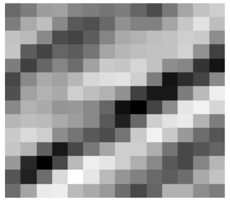
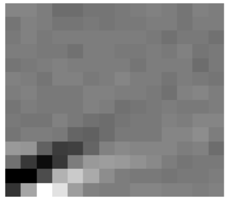
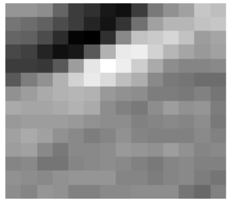
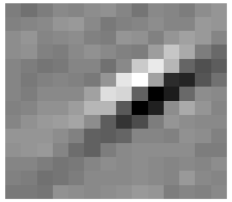
- **Input:** a new image patch x* , and K learned bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_K$
- **Output:** sparse representation **a** of an image patch x*.

$$\min_{\mathbf{a}} \left\| \mathbf{x}^* - \sum_{k=1}^{K} a_k \boldsymbol{\phi}_k \right\|_2^2 + \lambda \sum_{k=1}^{K} |a_k|$$

Sparse Coding: Testing Time

- **Input:** a new image patch x* , and K learned bases $\phi_1, \phi_2, ..., \phi_K$
- **Output:** sparse representation **a** of an image patch x*.

$$\min_{\mathbf{a}} \left\| \mathbf{x}^* - \sum_{k=1}^{K} a_k \phi_k \right\|_2^2 + \lambda \sum_{k=1}^{K} |a_k|$$

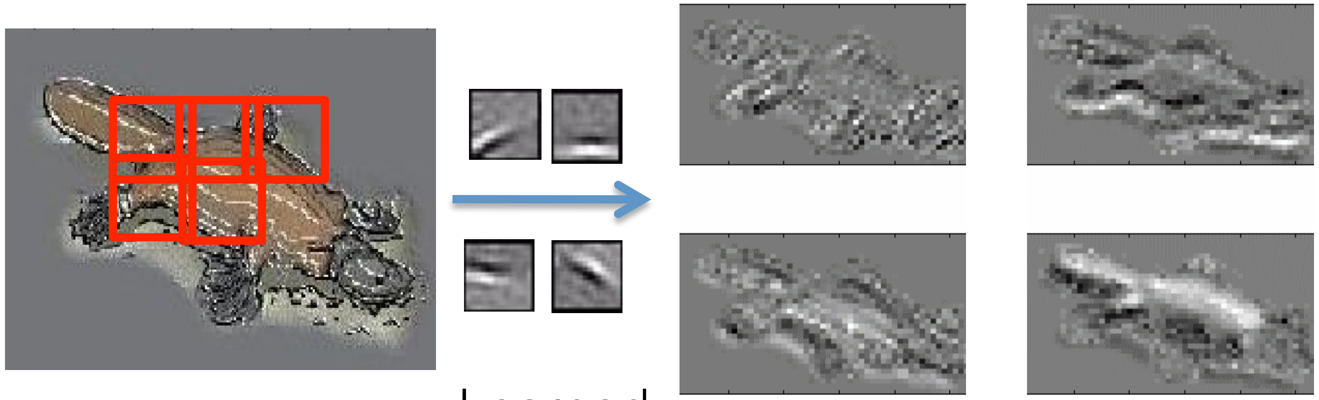
 $= 0.8 \, *$  $+ \, 0.3 \, *$  $+ \, 0.5 \, *$ 

$x^* \quad = 0.8 * \phi_{36} \quad + \; 0.3 * \phi_{42} \quad + 0.5 * \phi_{65}$

[0.0, 0.0, ... **0.8**, ..., **0.3**, ..., **0.5**, ...] = coefficients (feature representation)

Image Classification

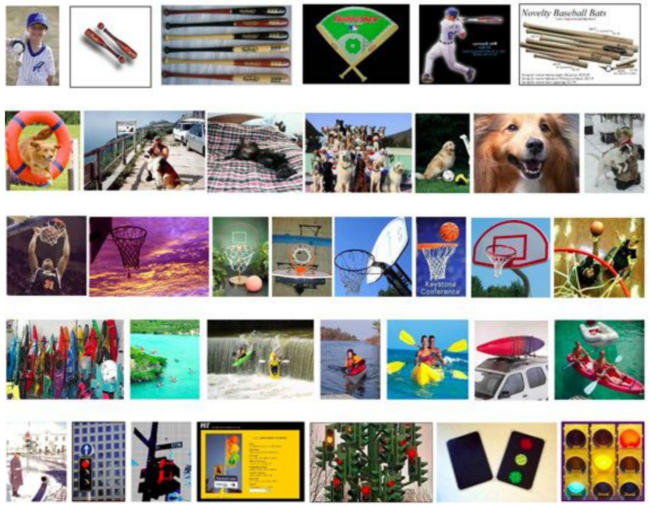
- Evaluated on Caltech101 object category dataset.



Input Image | Learned bases | Features (coefficients)

→ Classification Algorithm (SVM)

9K images, 101 classes



Algorithm	Accuracy
Baseline (Fei-Fei et al., 2004)	16%
PCA	37%
Sparse Coding	**47%**

Slide Credit: Honglak Lee

(Lee, Battle, Raina, Ng, NIPS 2007)

Modeling Image Patches

- Natural image patches:
  - small image regions extracted from an image of nature (forest, grass, …)

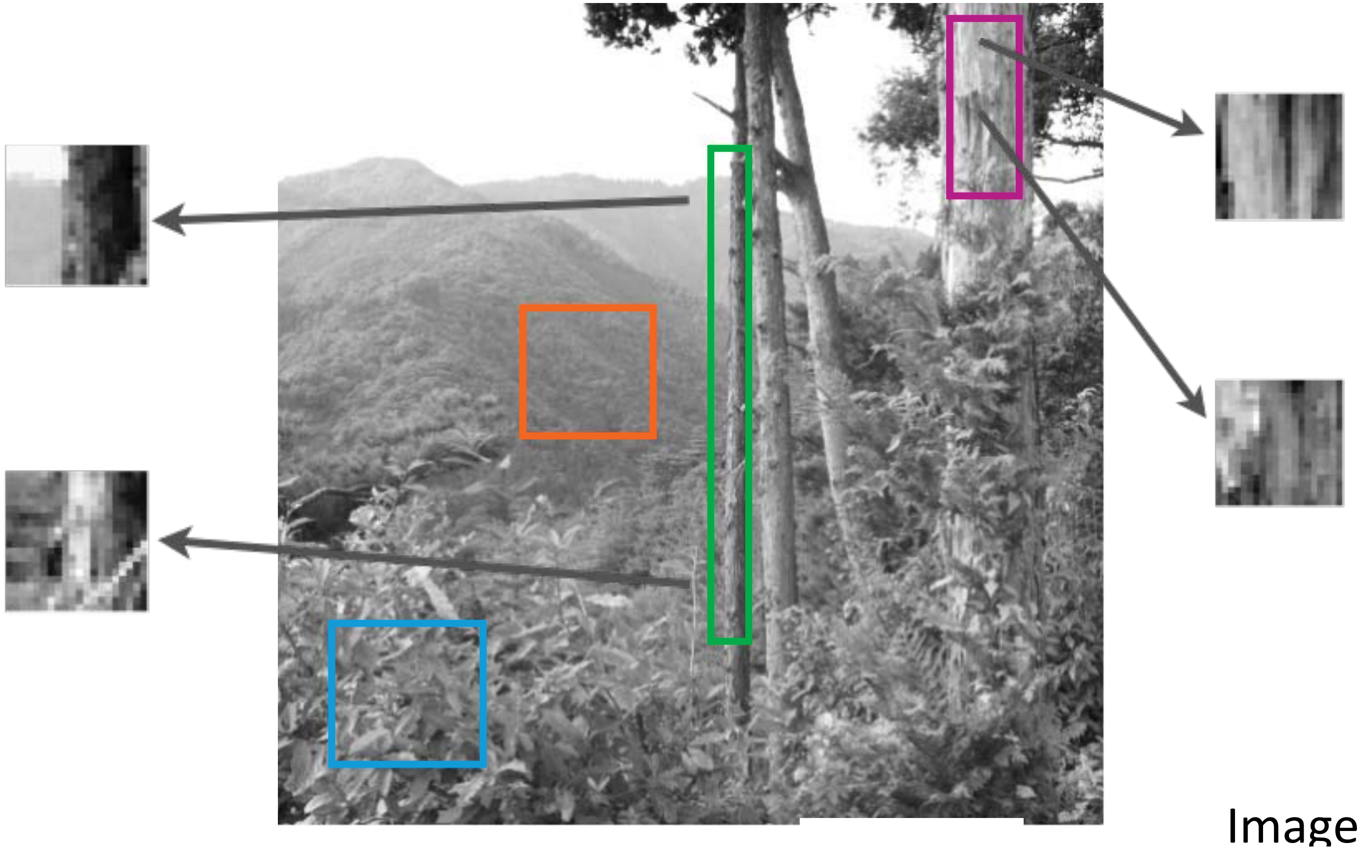
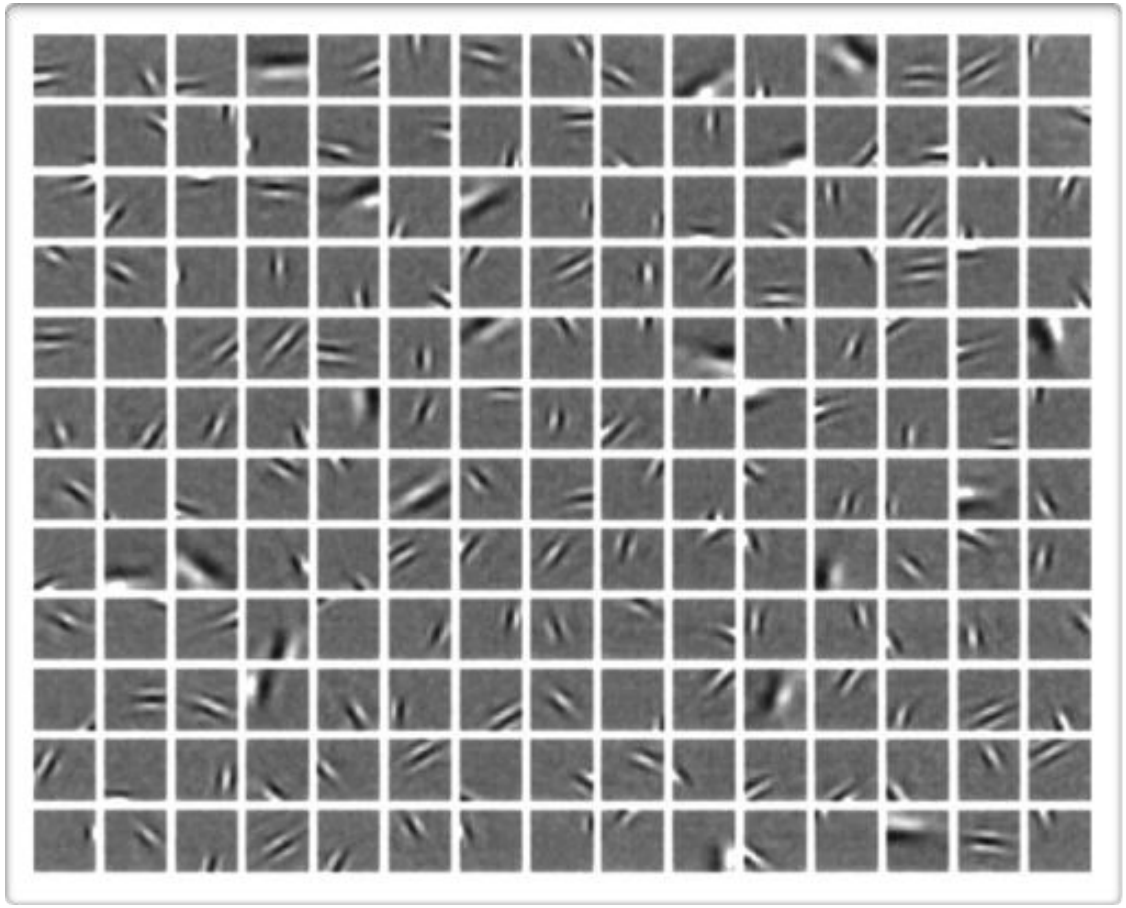


Image taken from: Emergence of complex cell properties by learning to generalize in natural scenes. Karklin and Lewicki, 2009

Relationship to V1

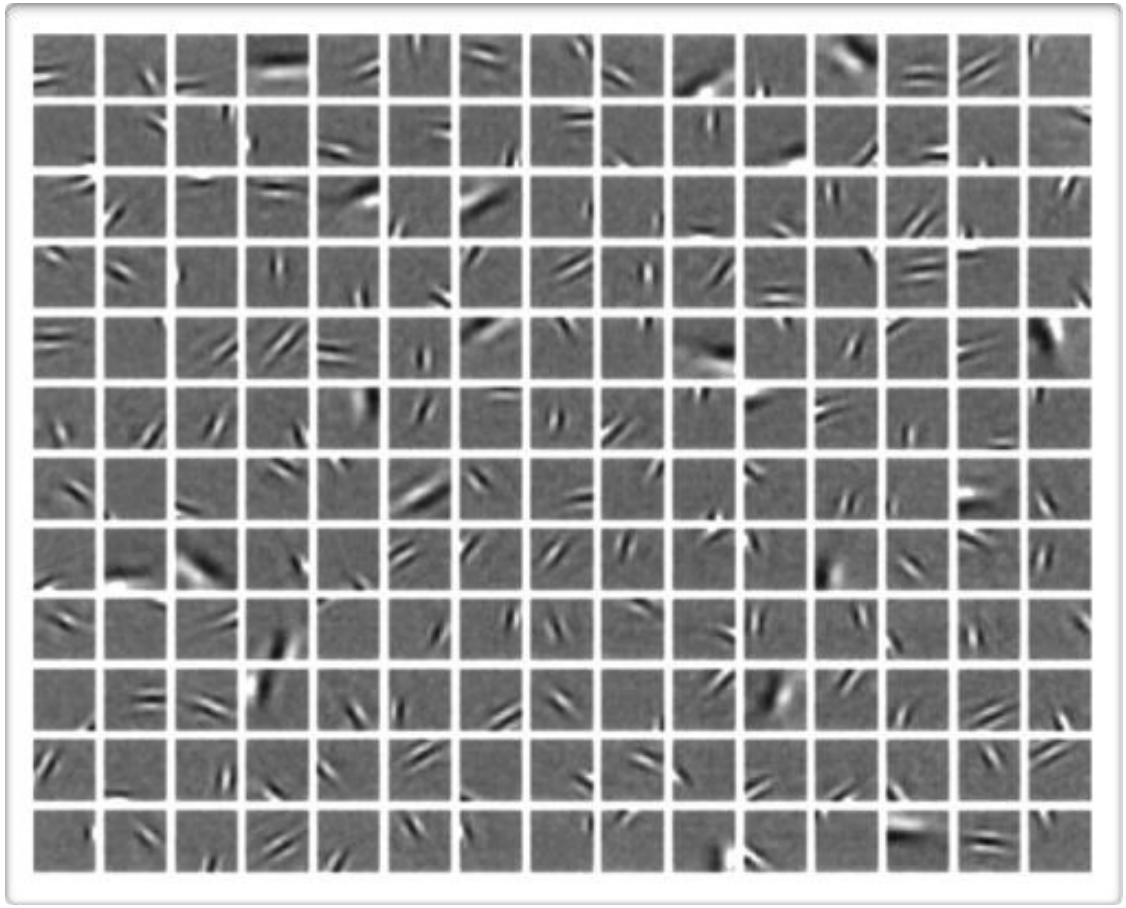
- When trained on natural image patches
  - the dictionary columns (''atoms'') look like edge detectors
  - each atom is tuned to a particular position, orientation and spatial frequency
  - V1 neurons in the mammalian brain have a similar behavior



Emergence of simple-cell receptive field properties by learning a sparse code of natural images. Olshausen and Field, 1996

Relationship to V1

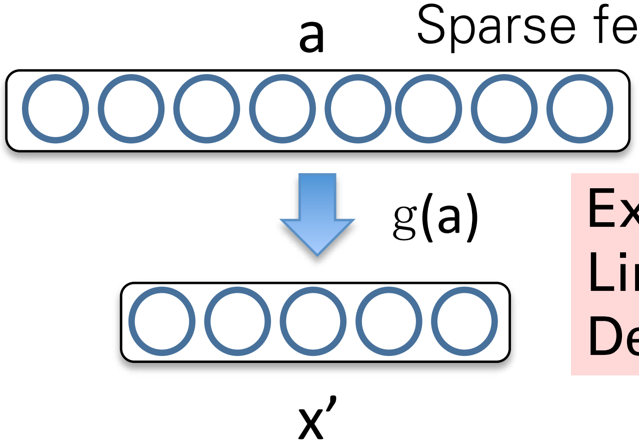
- Suggests that the brain might be learning a sparse code of visual stimulus
  - Since then, many other models have been shown to learn similar features
  - they usually all incorporate a notion of sparsity



Emergence of simple-cell receptive field properties by learning a sparse code of natural images. Olshausen and Field, 1996

Interpreting Sparse Coding

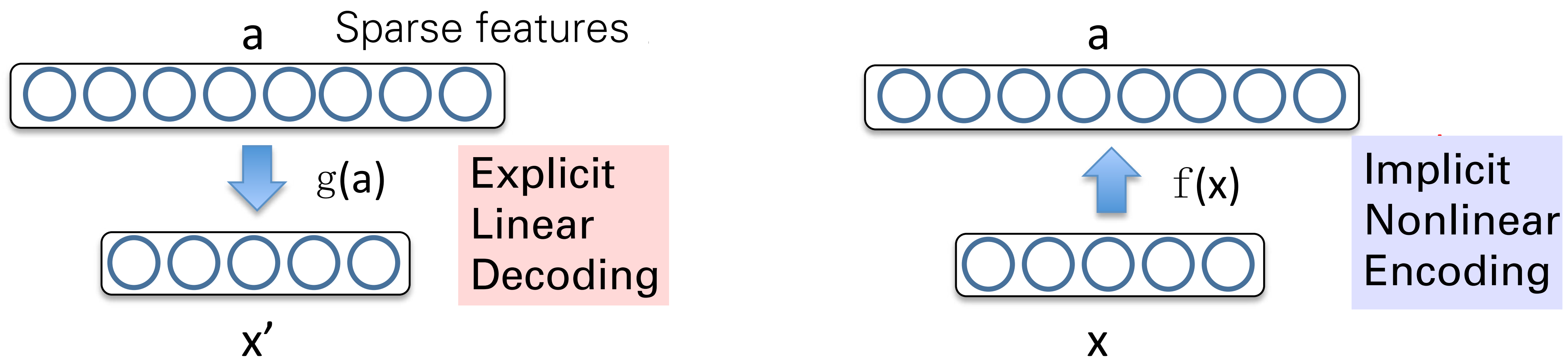
$$\min_{\mathbf{a},\phi} \sum_{n=1}^{N} \left\| \mathbf{x}_n - \sum_{k=1}^{K} a_{nk}\phi_k \right\|_2^2 + \lambda \sum_{n=1}^{N}\sum_{k=1}^{K} |a_{nk}|$$



Explicit
Linear
Decoding

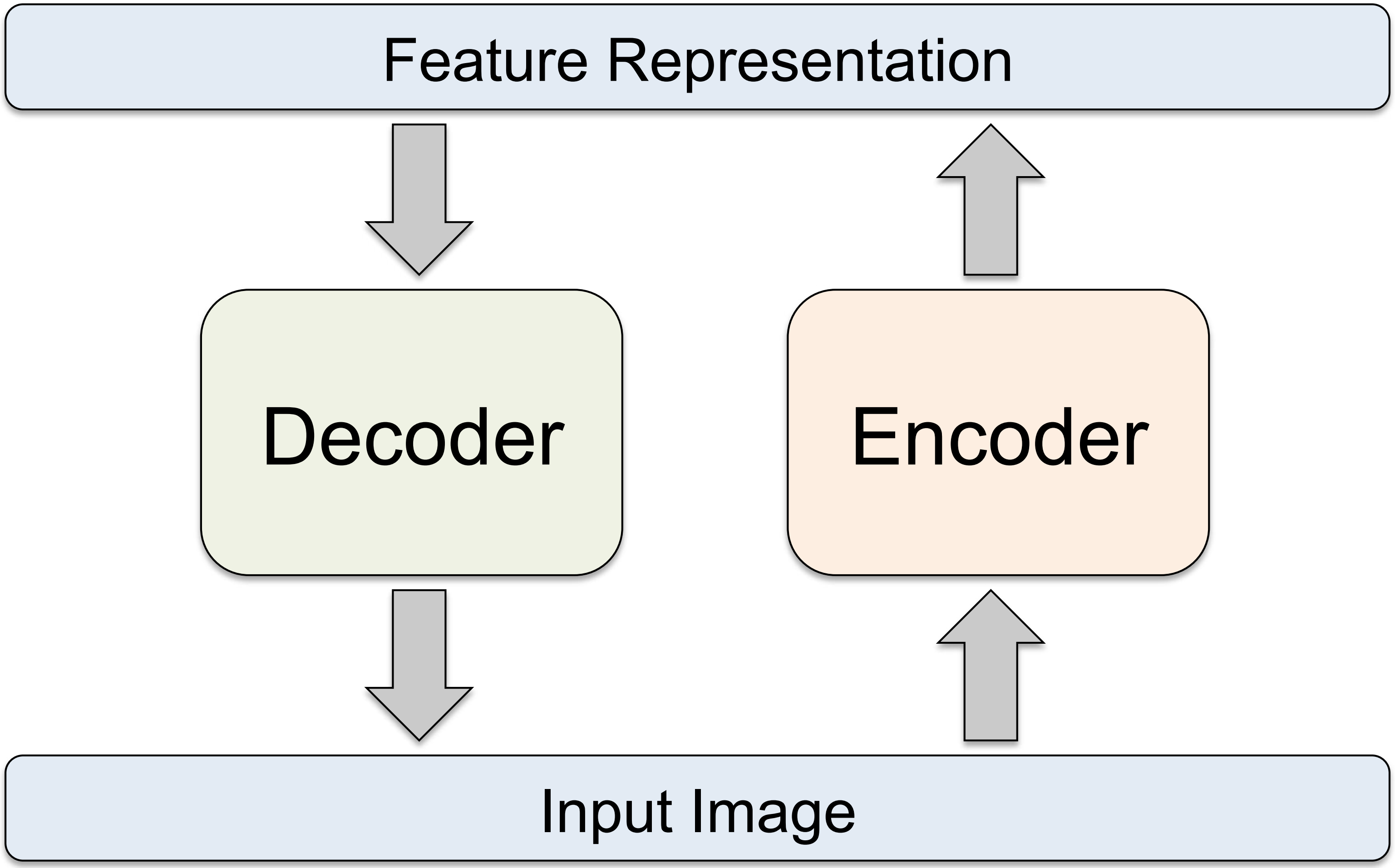
Interpreting Sparse Coding

$$\min_{\mathbf{a},\phi} \sum_{n=1}^{N} \left\| \mathbf{x}_n - \sum_{k=1}^{K} a_{nk}\phi_k \right\|_2^2 + \lambda \sum_{n=1}^{N}\sum_{k=1}^{K} |a_{nk}|$$

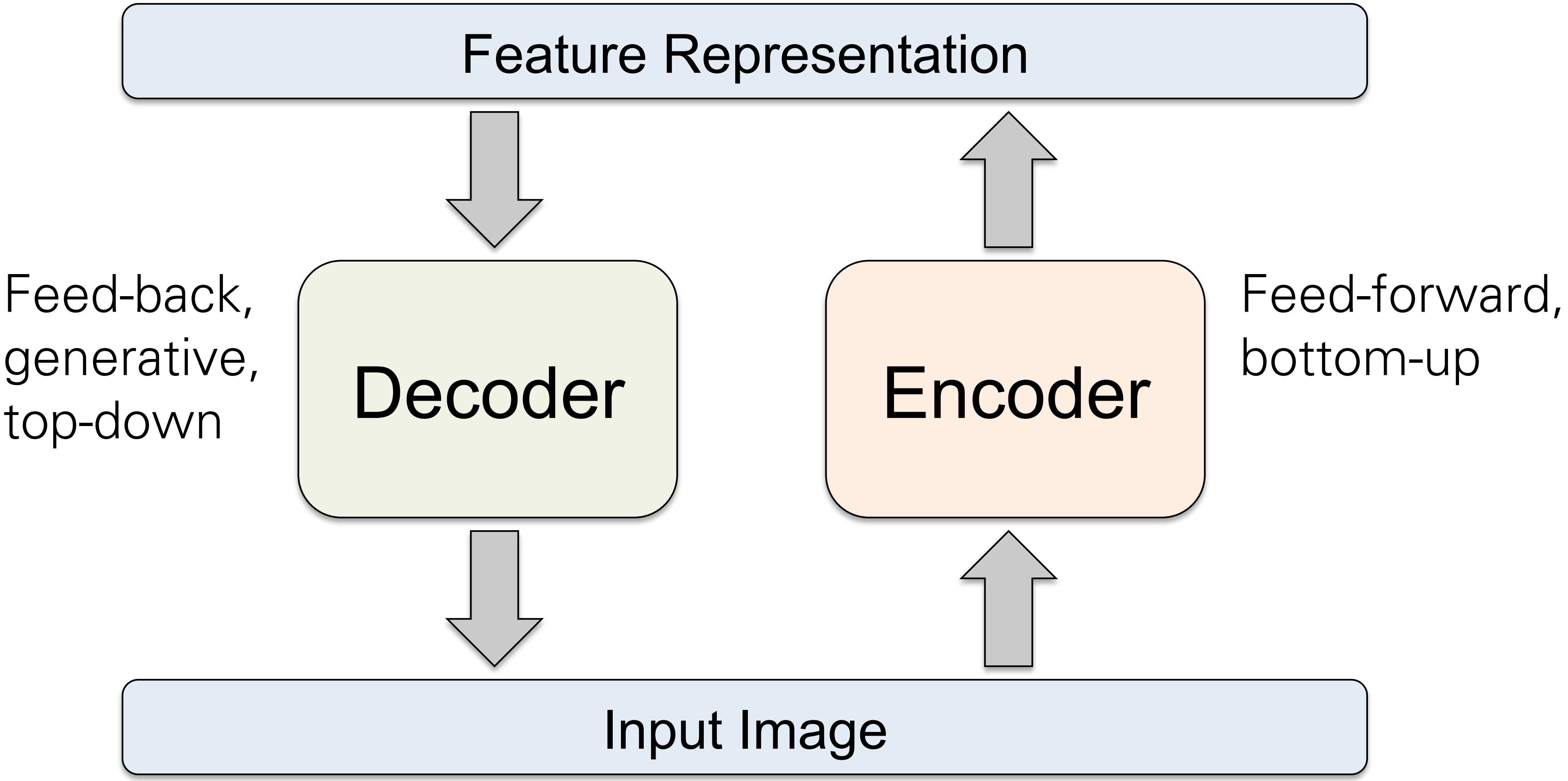


- Sparse, over-complete representation **a**.
- Encoding **a** = f(**x**) is implicit and nonlinear function of **x**.
- Reconstruction (or decoding) **x'** = g(**a**) is linear and explicit.

Autoencoder

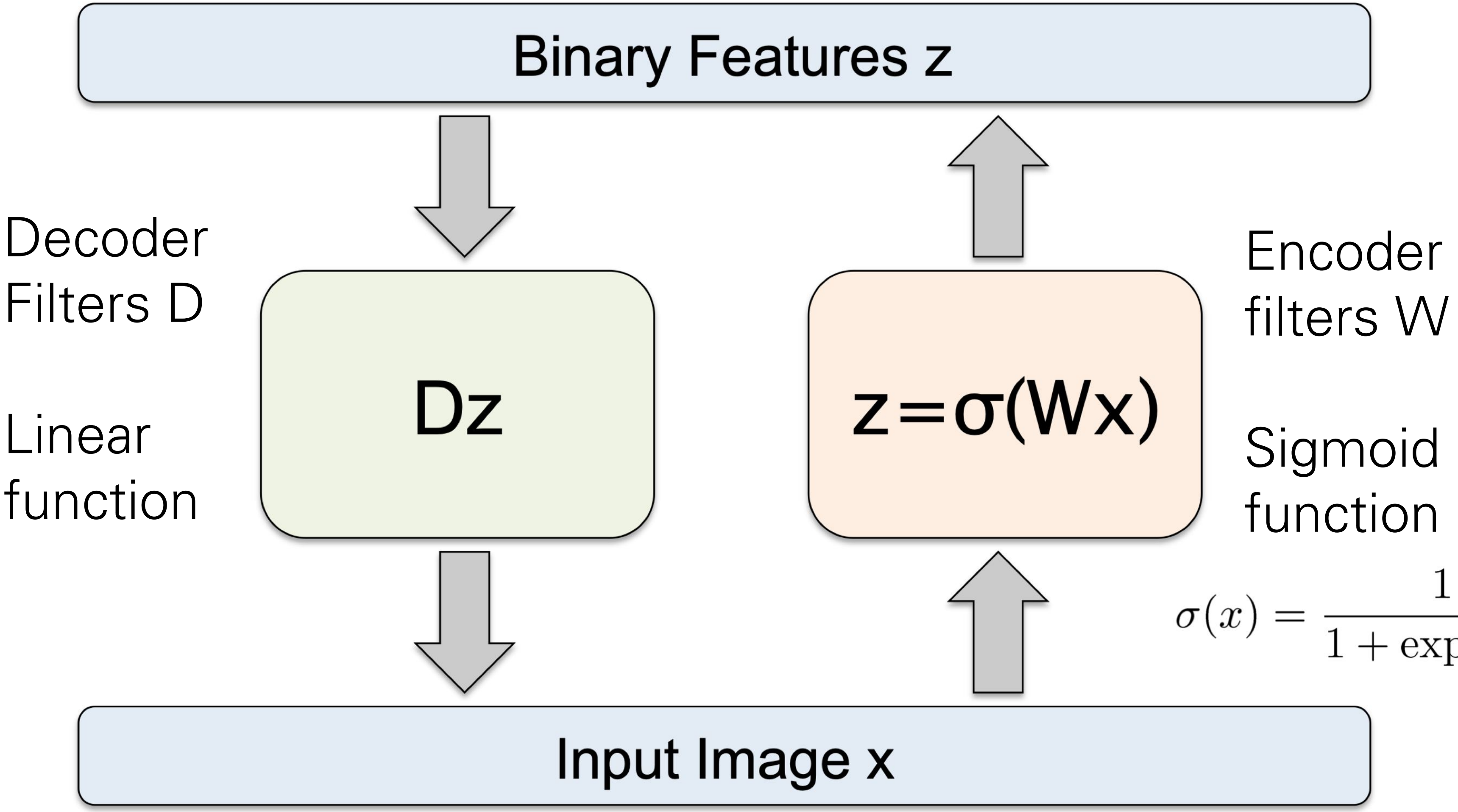


Autoencoder



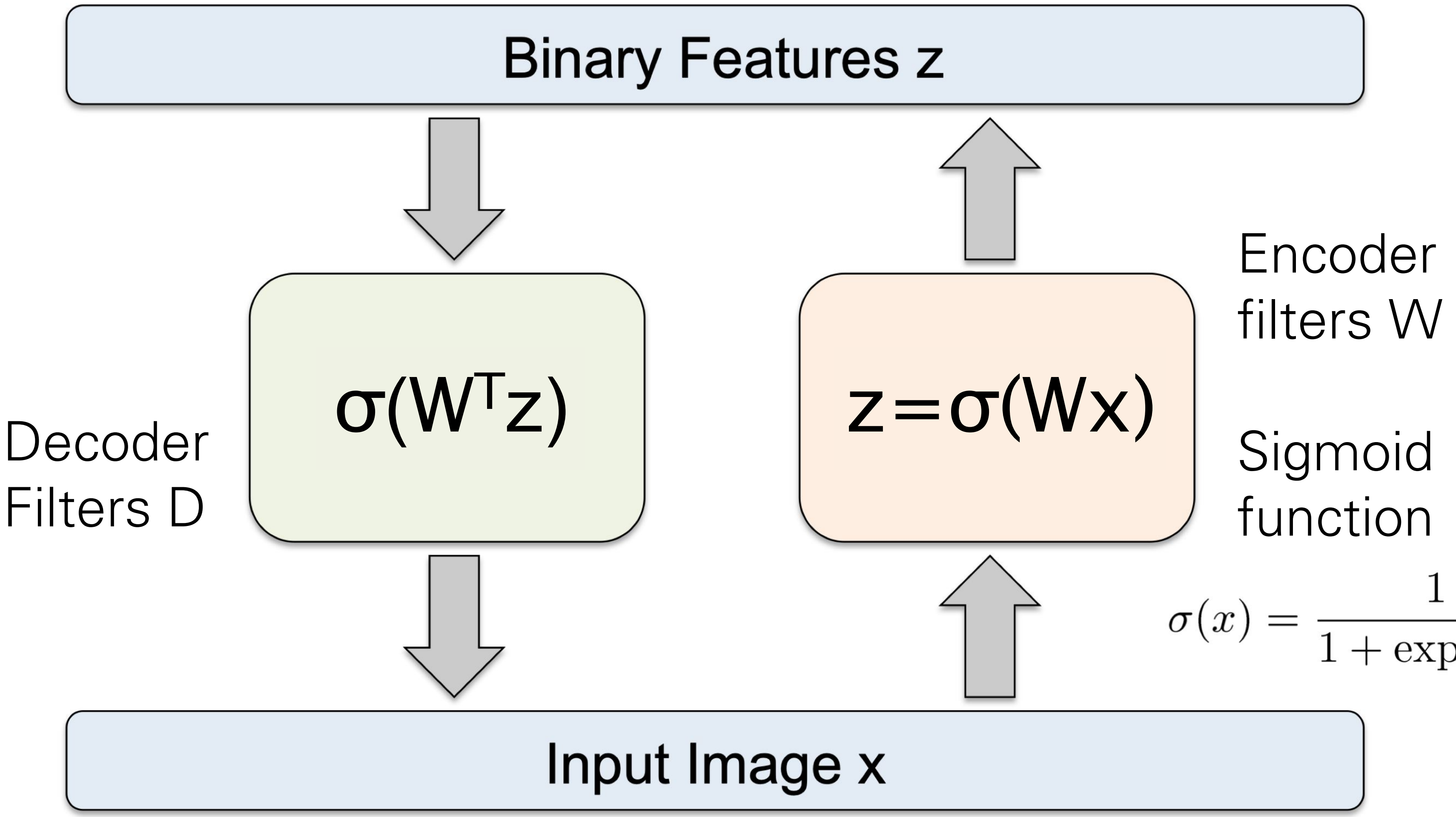
- Details of what goes insider the encoder and decoder matter!
- Need constraints to avoid learning an identity.

Autoencoder



$$\sigma(x) = \frac{1}{1+\exp(-x)}$$

Autoencoder

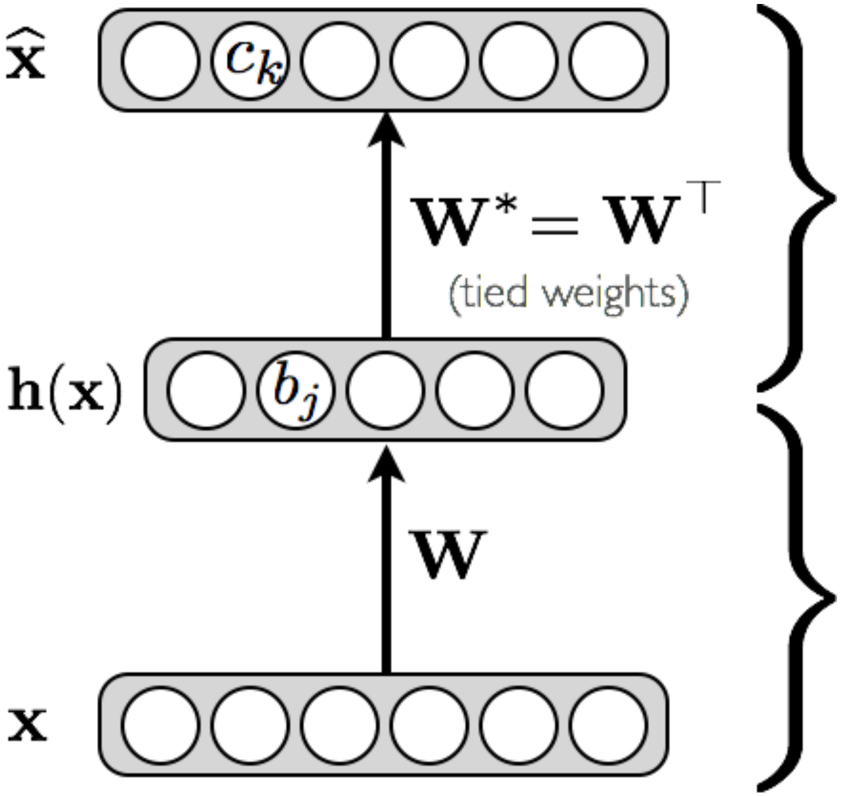


$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

- Need additional constraints to avoid learning an identity.
- Relates to Restricted Boltzmann Machines (later).

Autoencoder

- Feed-forward neural network trained to reproduce its input at the output layer



Decoder

$$\begin{aligned}\widehat{\mathbf{x}} &= o(\widehat{\mathbf{a}}(\mathbf{x})) \\ &= \underbrace{\mathrm{sigm}}_{\text{for binary units}}(\mathbf{c} + \mathbf{W}^*\mathbf{h}(\mathbf{x}))\end{aligned}$$

Encoder

$$\begin{aligned}\mathbf{h}(\mathbf{x}) &= g(\mathbf{a}(\mathbf{x})) \\ &= \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x})\end{aligned}$$

Loss Function

- Loss function for binary inputs

$$l(f(\mathbf{x})) = -\sum_k \left(x_k \log(\widehat{x}_k) + (1 - x_k)\log(1 - \widehat{x}_k)\right)$$

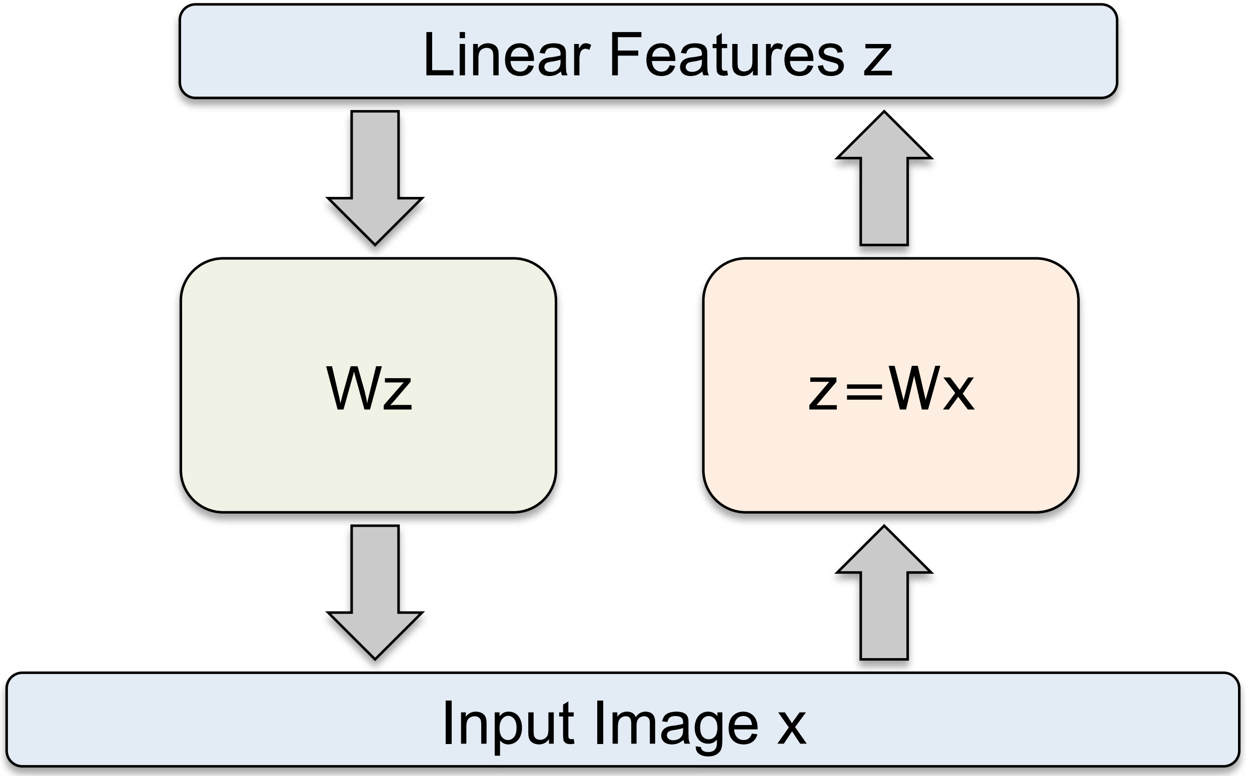
  - Cross-entropy error function (reconstruction loss) $f(\mathbf{x}) \equiv \widehat{\mathbf{x}}$

- Loss function for real-valued inputs

$$l(f(\mathbf{x})) = \frac{1}{2}\sum_k (\widehat{x}_k - x_k)^2$$

  - sum of squared differences (reconstruction loss)
  - we use a linear activation function at the output

Autoencoder

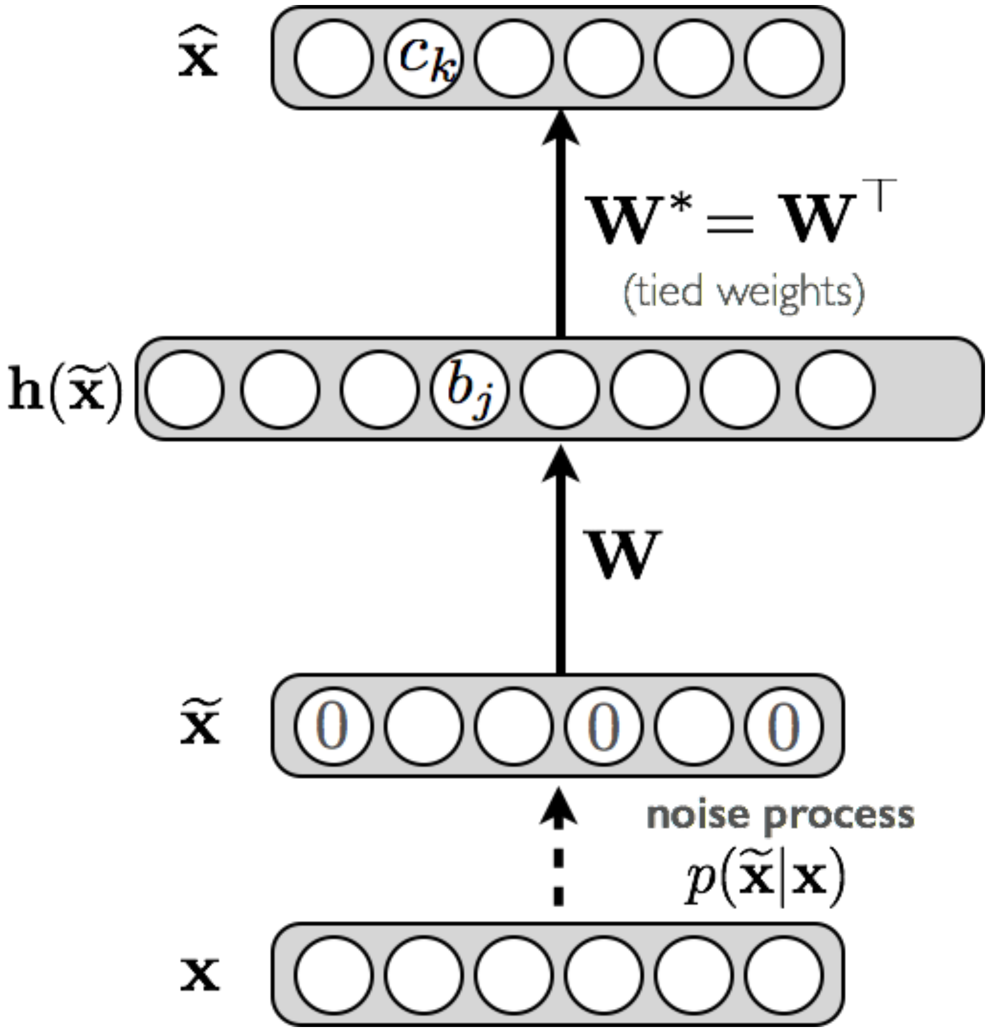


- If the hidden and output layers are linear, it will learn hidden units that are a linear function of the data and minimize the squared error.
- The K hidden units will span the same space as the first k principal components. The weight vectors may not be orthogonal.

- With nonlinear hidden units, we have a nonlinear generalization of PCA.

Denoising Autoencoder

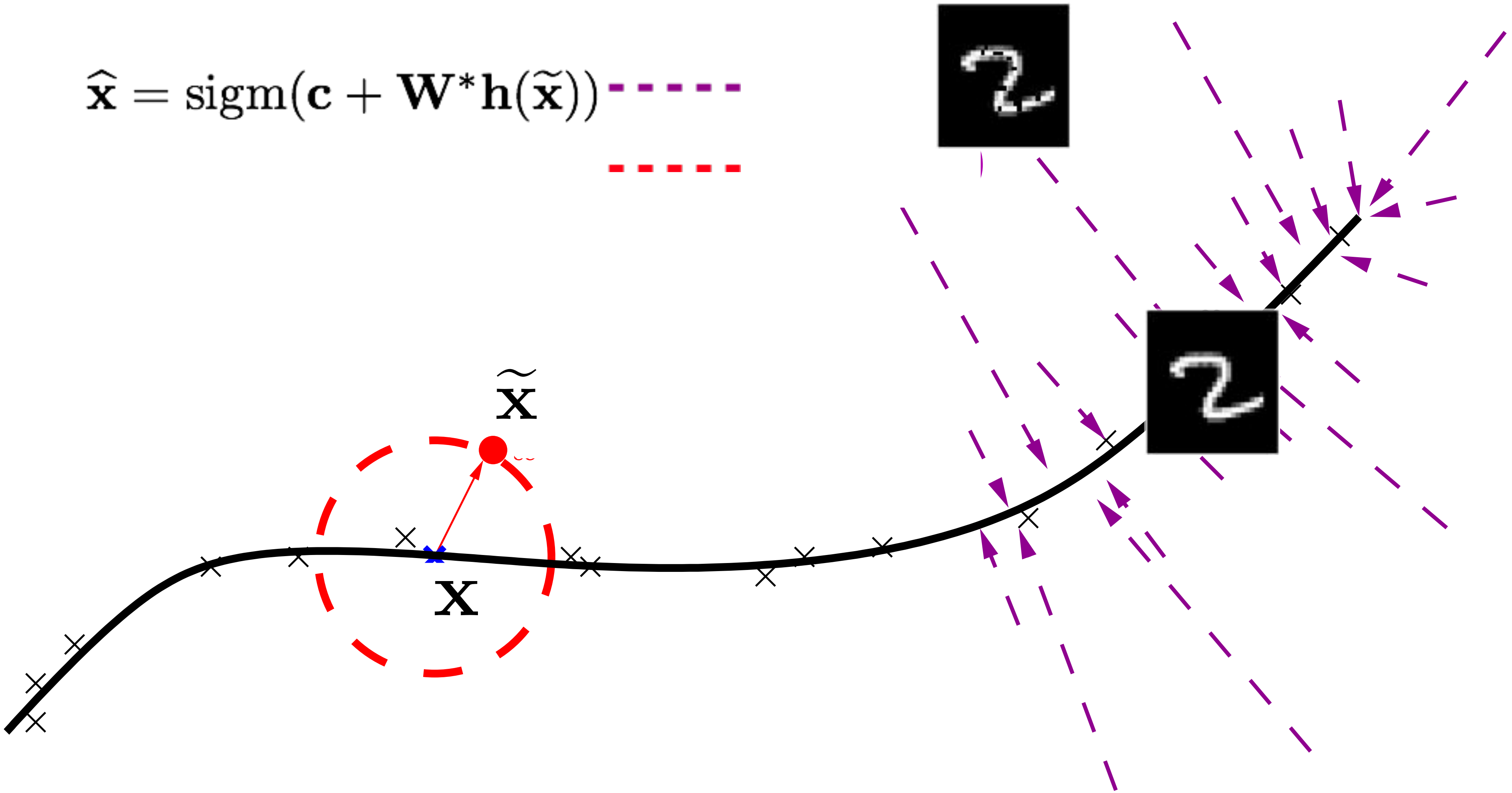
- **Idea:** Representation should be robust to introduction of noise:
  - random assignment of subset of inputs to 0, with probability ν
  - Similar to dropouts on the input layer
  - Gaussian additive noise



- **Reconstruction** $\widehat{\mathbf{x}}$ computed from the corrupted input $\widetilde{\mathbf{x}}$
- **Loss function** compares $\widehat{\mathbf{x}}$ reconstruction with the noiseless input $\mathbf{x}$

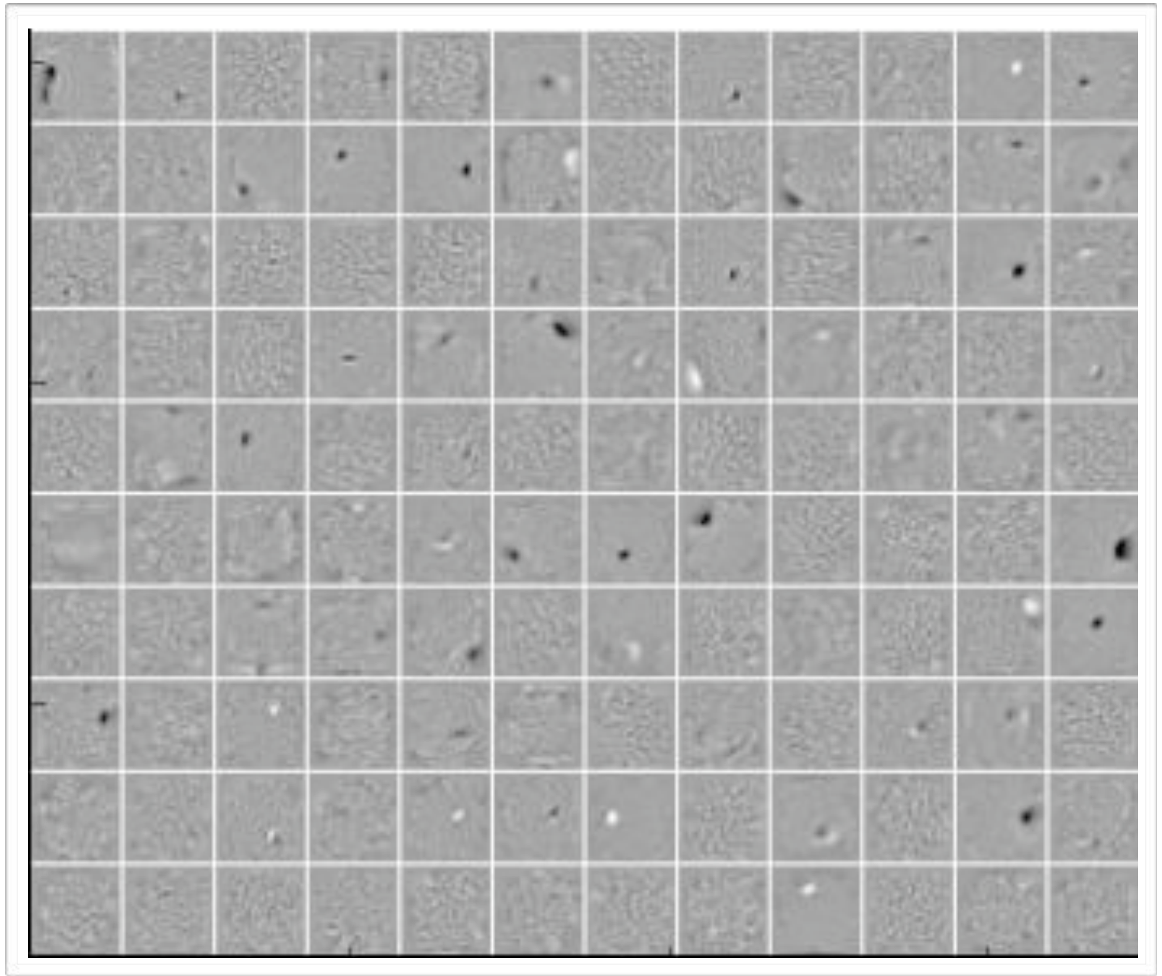
(Vincent et al., ICML 2008)

Denoising Autoencoder

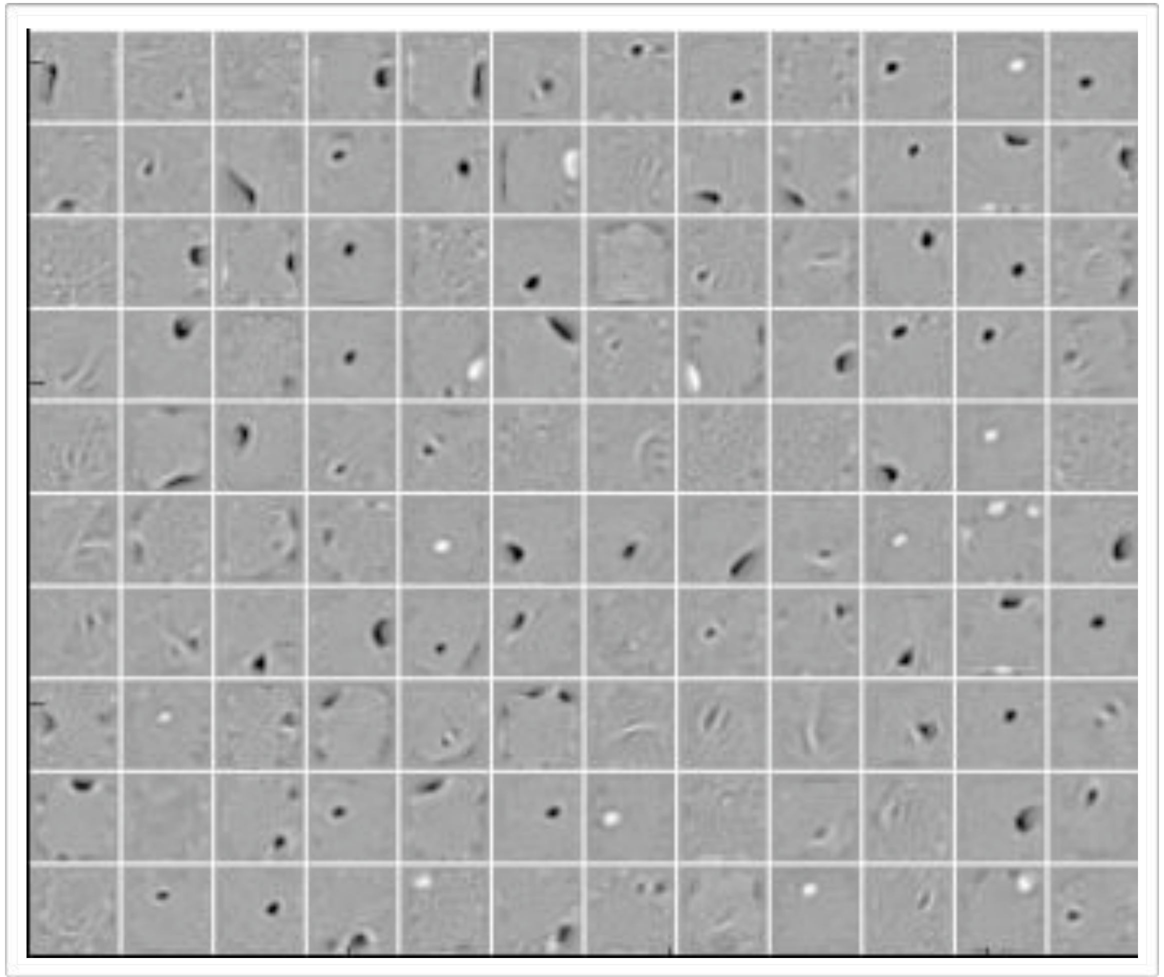


Learned Filters

Non-corrupted



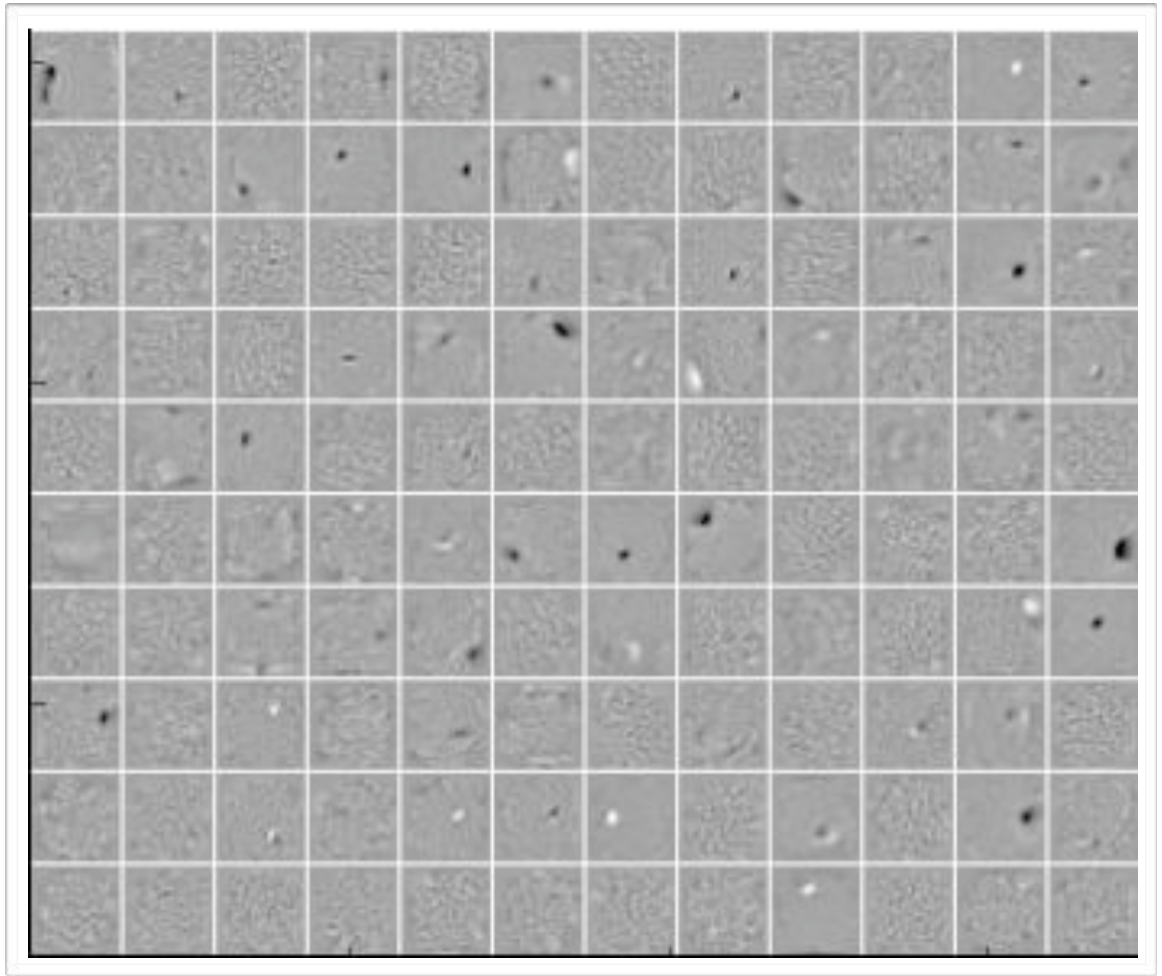
25% corrupted input



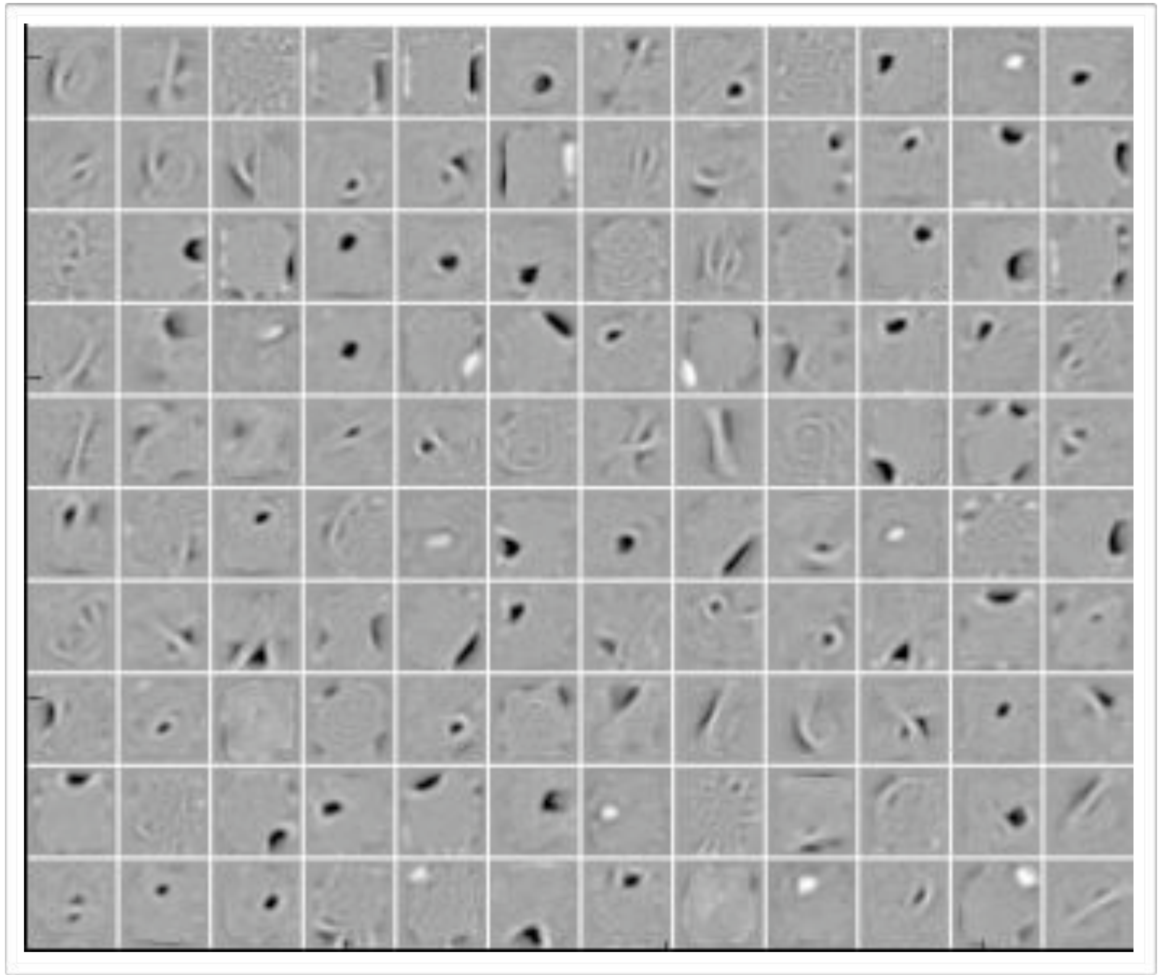
(Vincent et al., ICML 2008)

Learned Filters

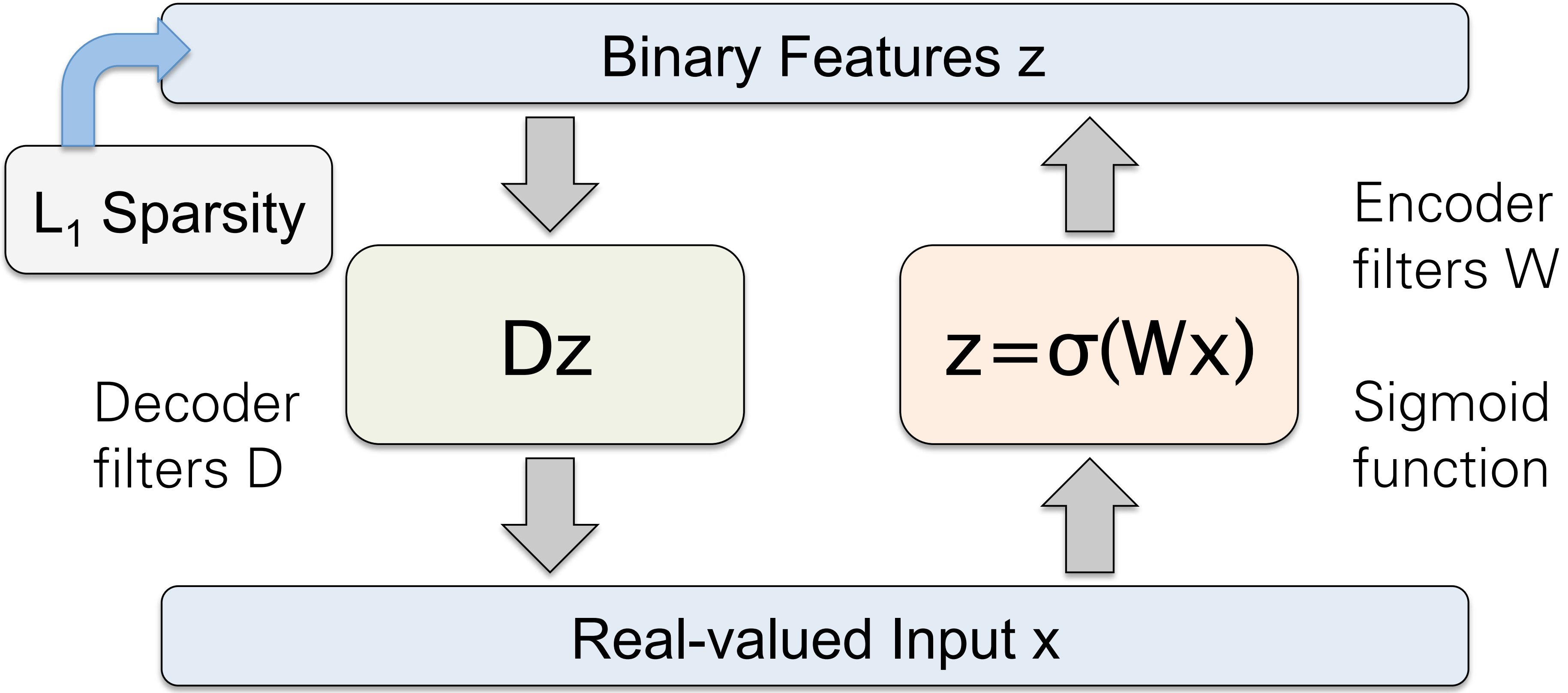
Non-corrupted



50% corrupted input

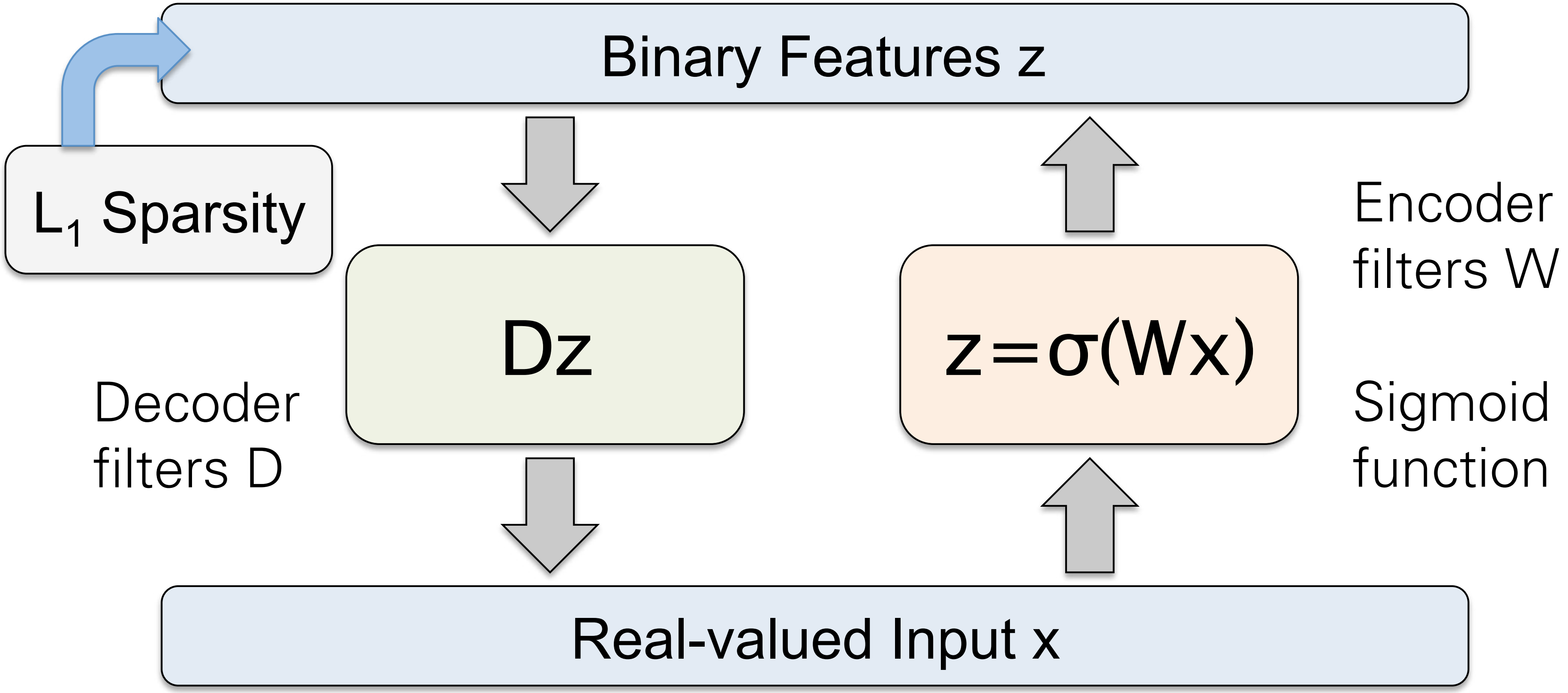


Predictive Sparse Decomposition



(Kavukcuoglu, Ranzato, Fergus, LeCun, 2009)

Predictive Sparse Decomposition

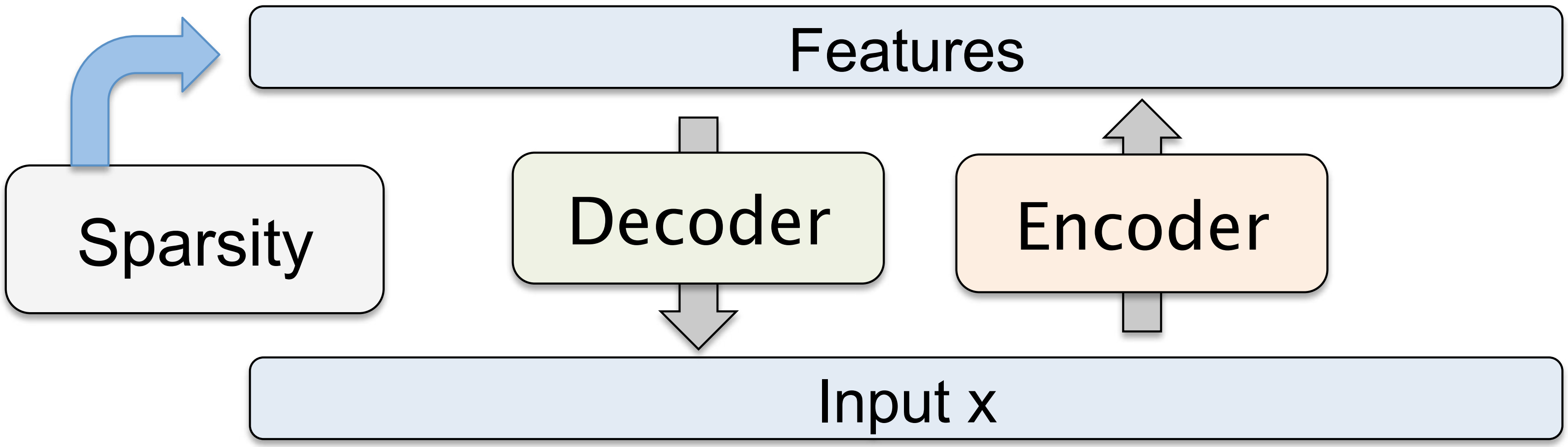


At training time

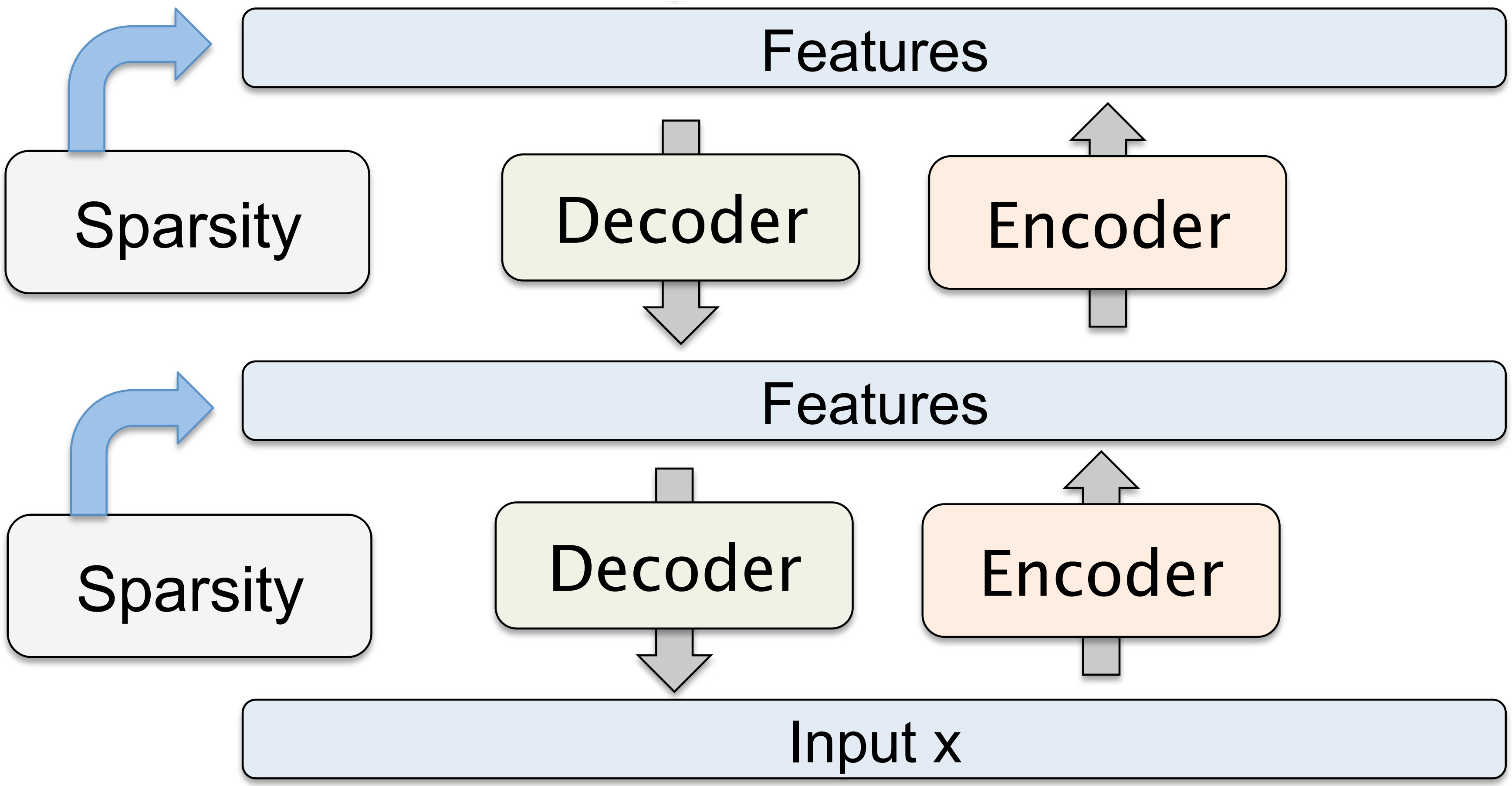
$$\min_{D,W,\mathbf{z}} \underbrace{||D\mathbf{z}-\mathbf{x}||_2^2 + \lambda|\mathbf{z}|_1}_{\text{Decoder}} + \underbrace{||\sigma(W\mathbf{x})-\mathbf{z}||_2^2}_{\text{Encoder}}$$

(Kavukcuoglu, Ranzato, Fergus, LeCun, 2009)

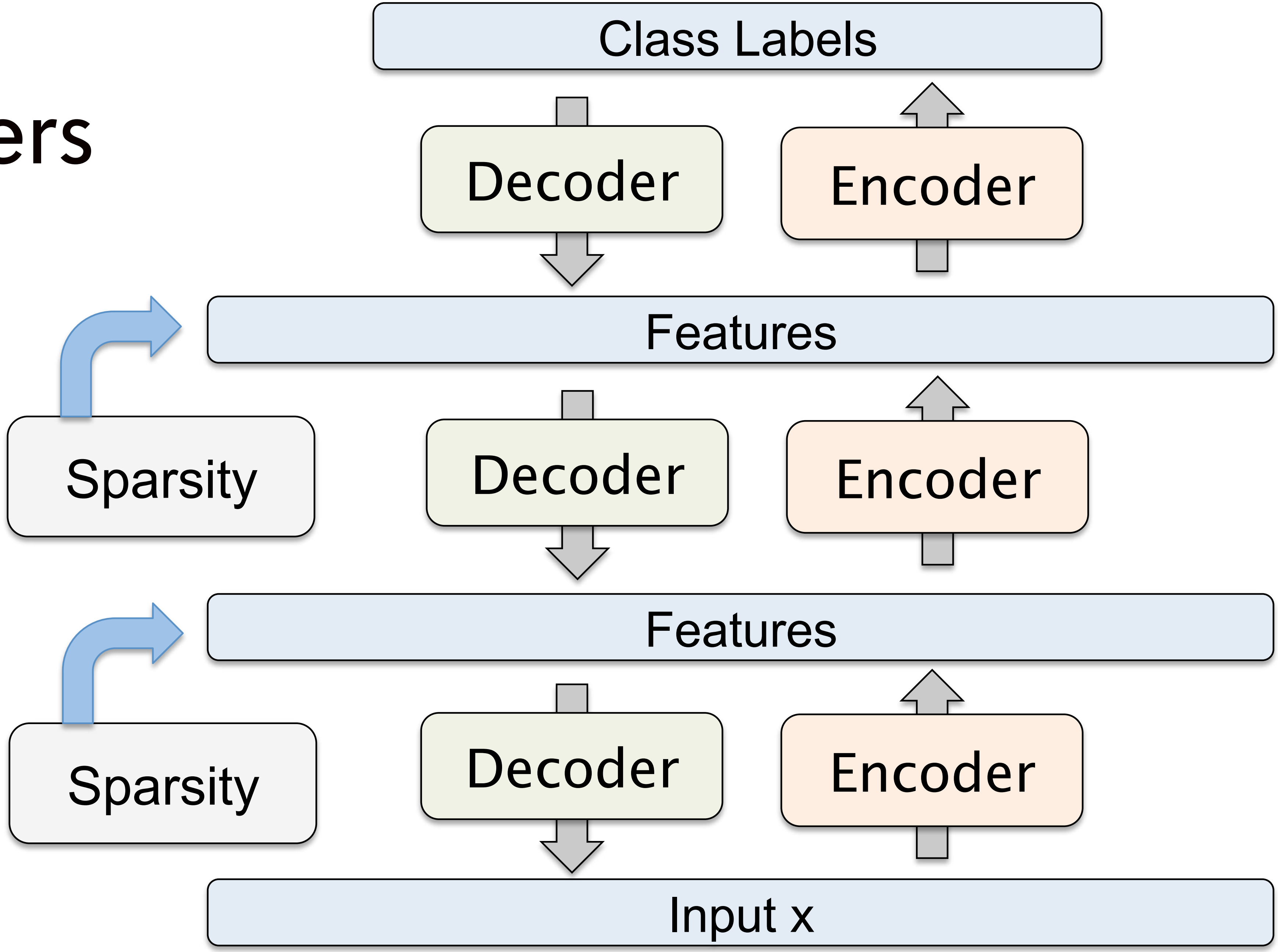
Stacked Autoencoders



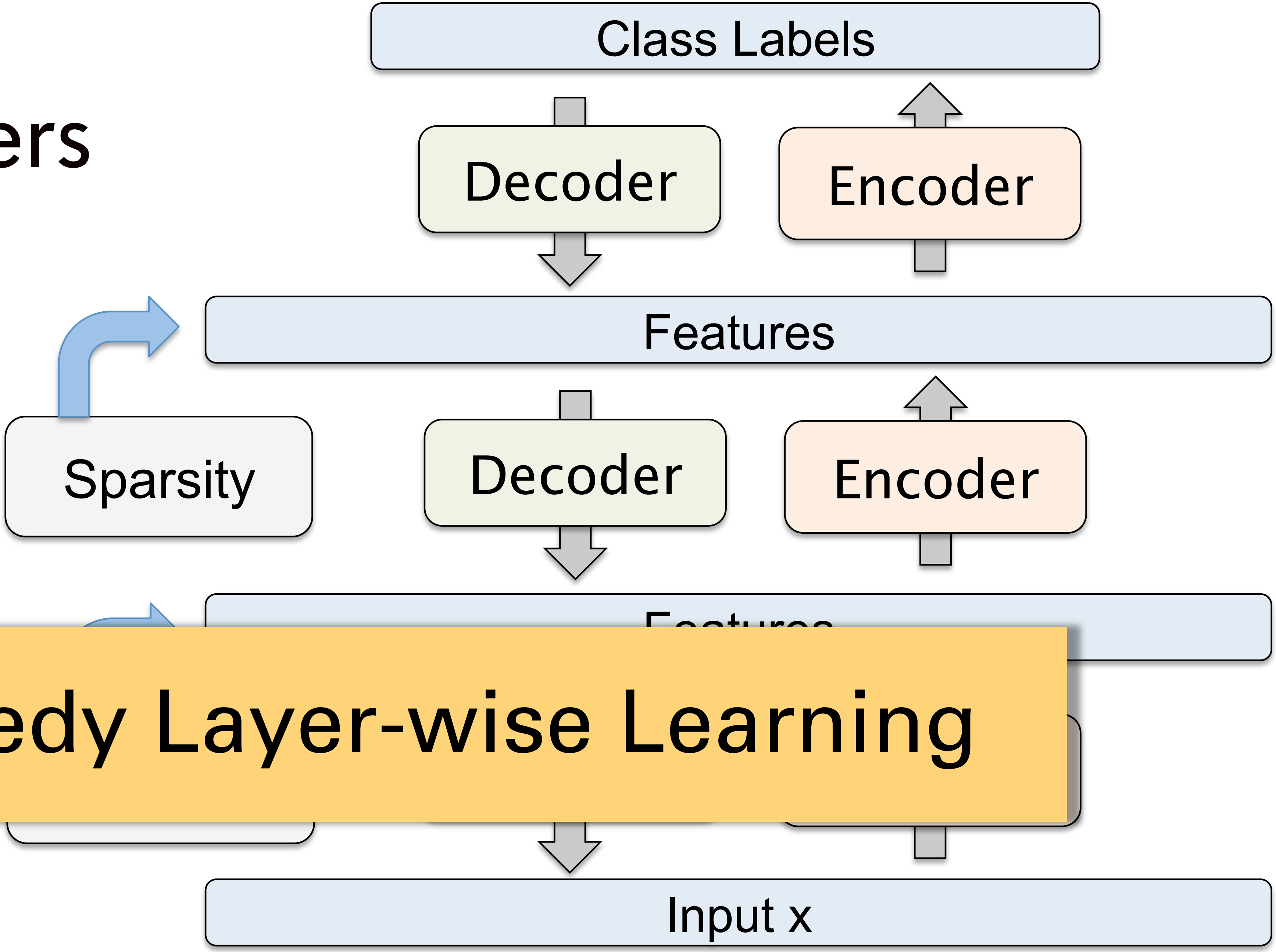
Stacked Autoencoders



Stacked Autoencoders



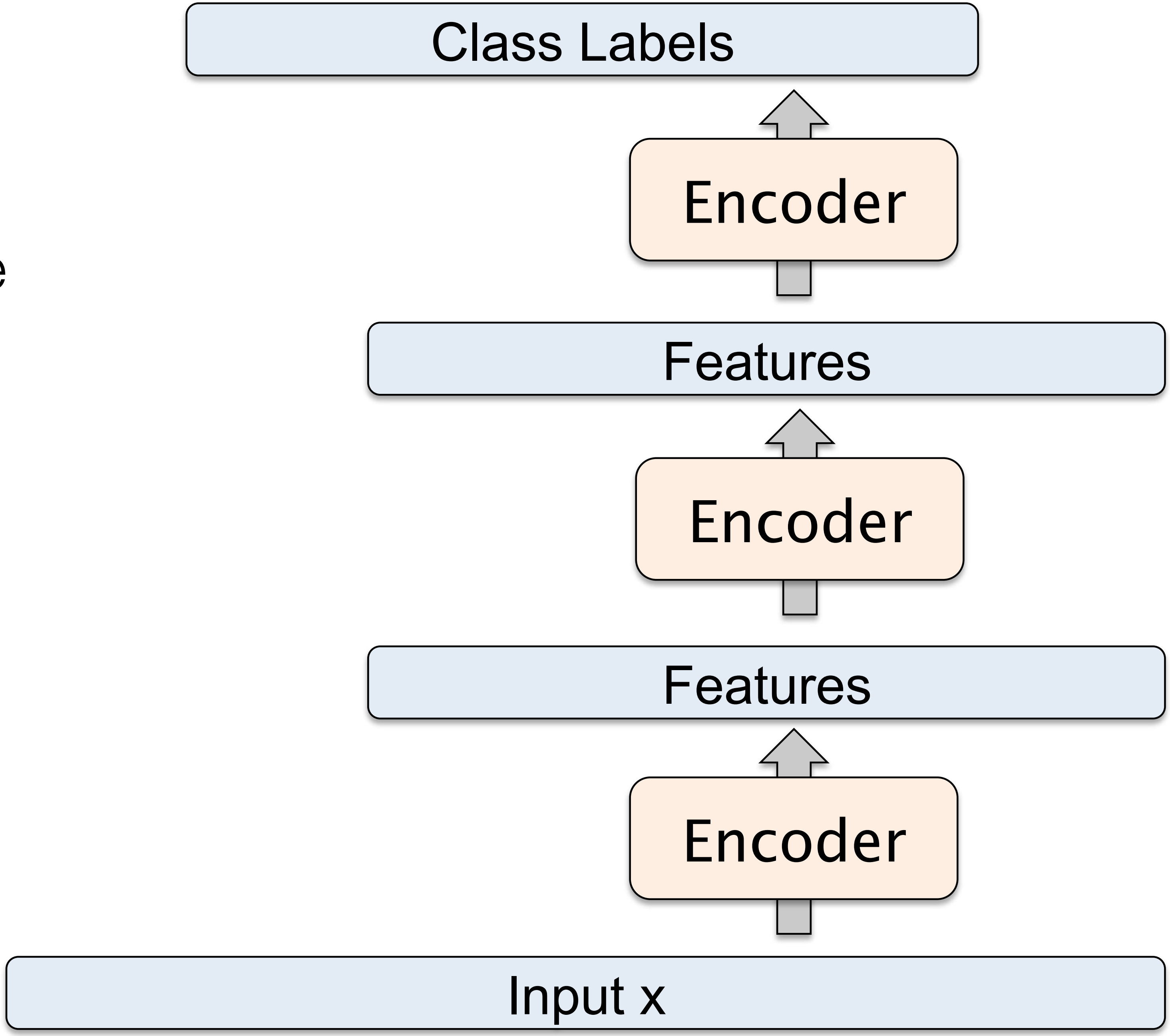
Stacked Autoencoders



Greedy Layer-wise Learning

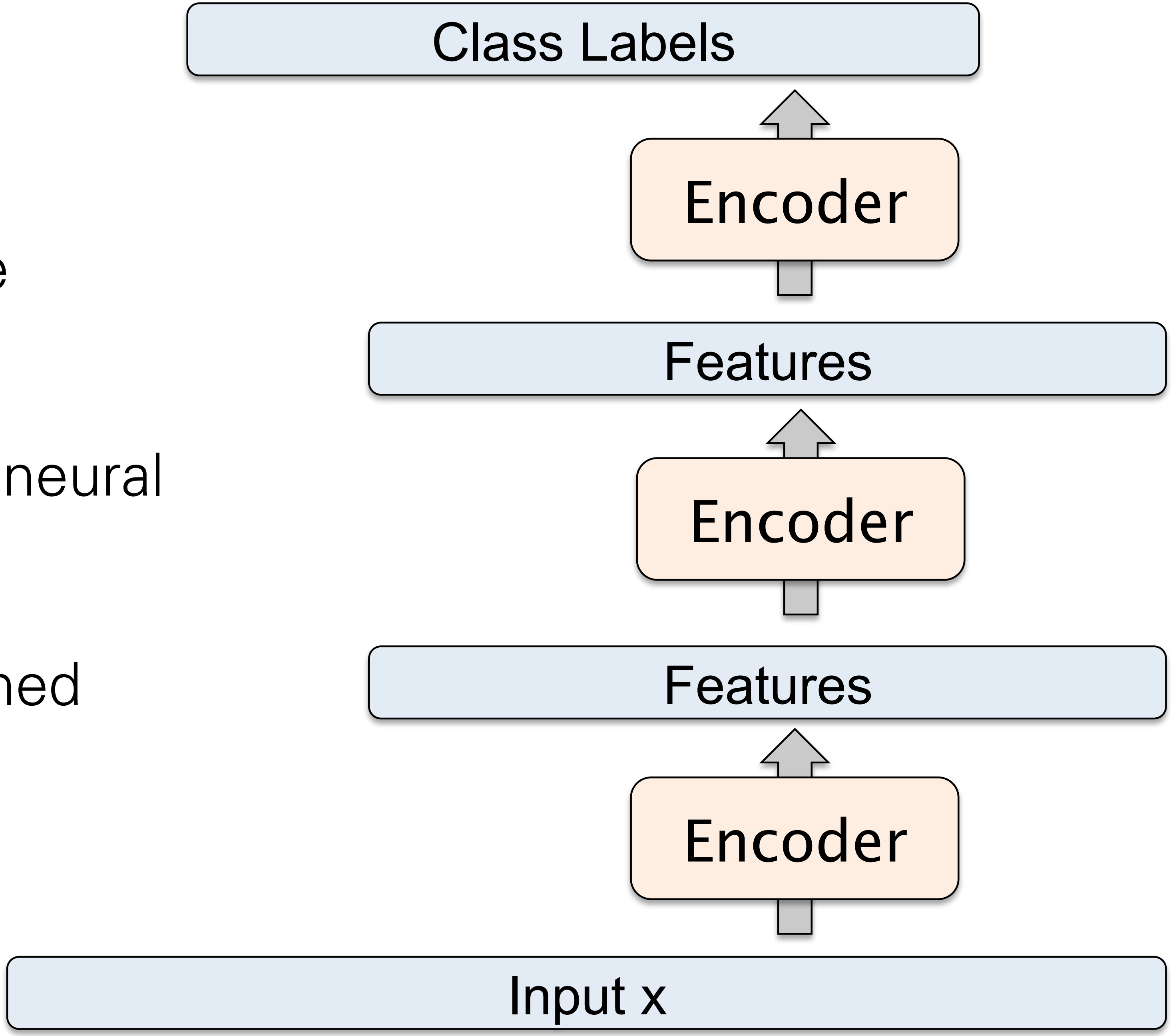
Stacked Autoencoders

- Remove decoders and use feed-forward part.

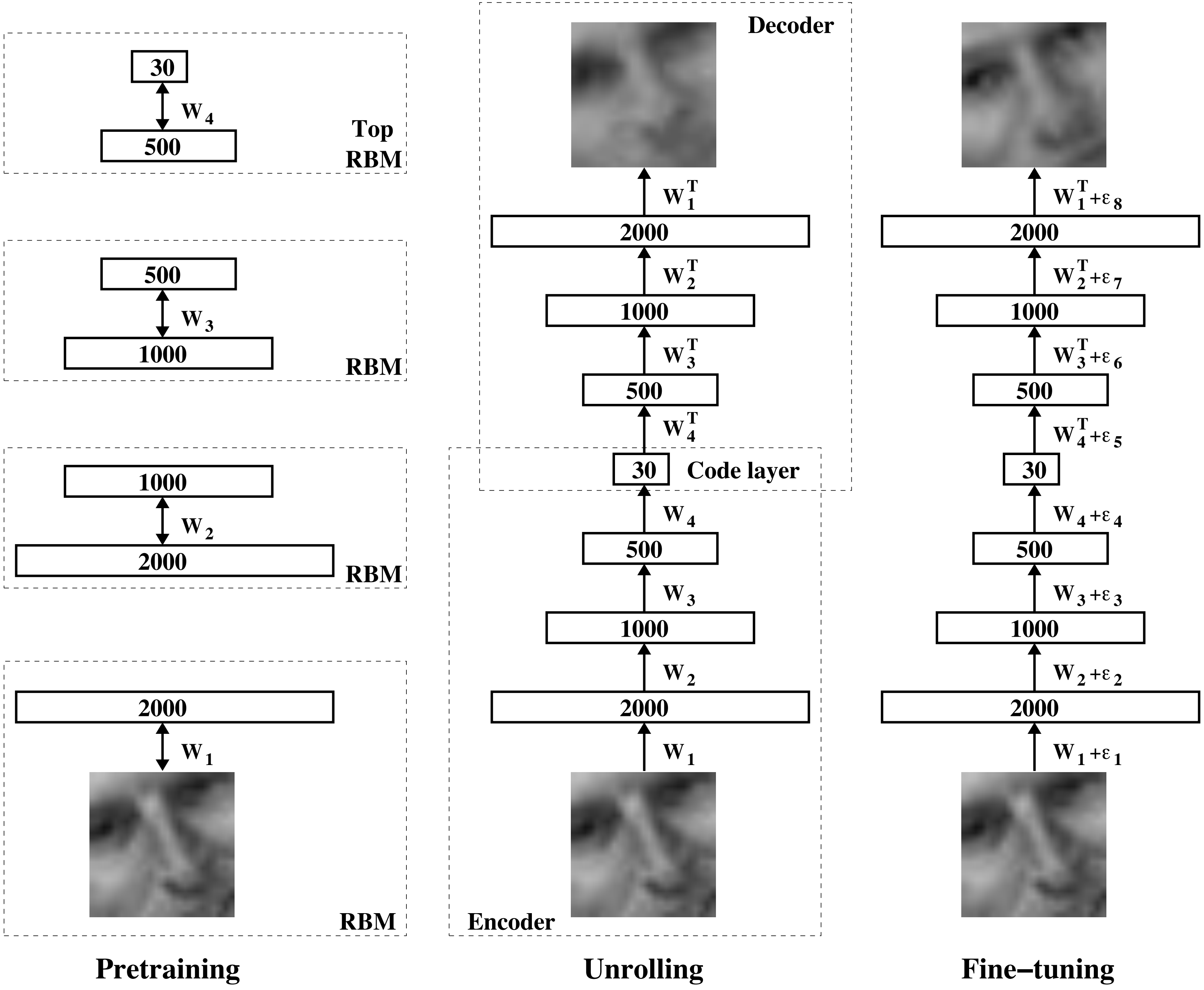


Stacked Autoencoders

- Remove decoders and use feed-forward part.
- Standard, or convolutional neural network architecture.
- Parameters can be fine-tuned using backpropagation.

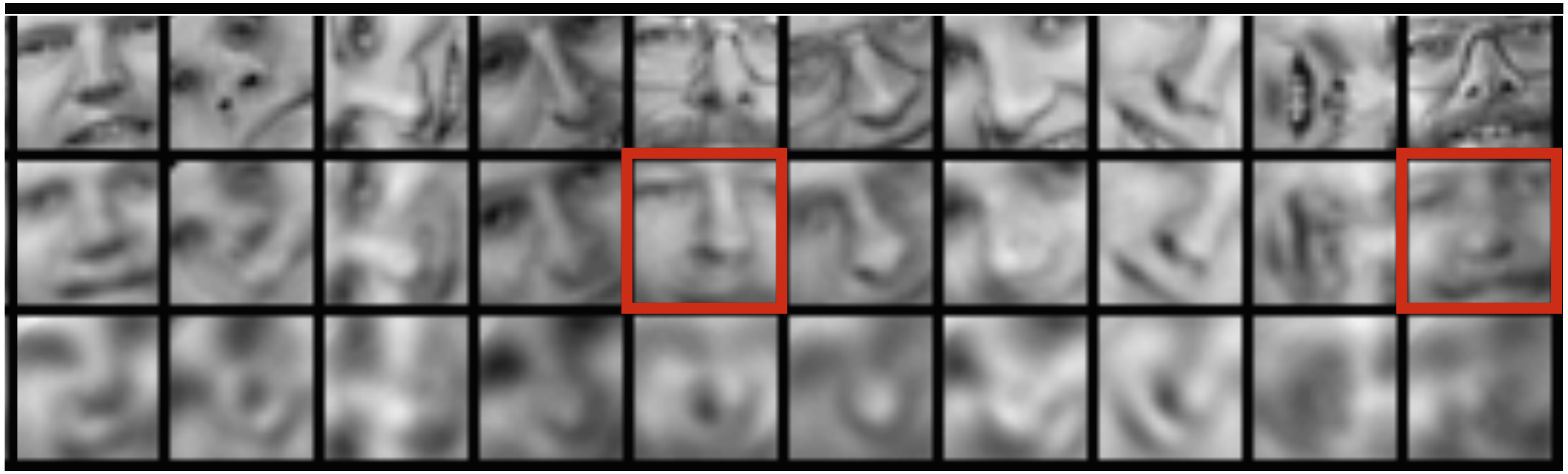


Deep Autoencoder



Deep Autoencoders

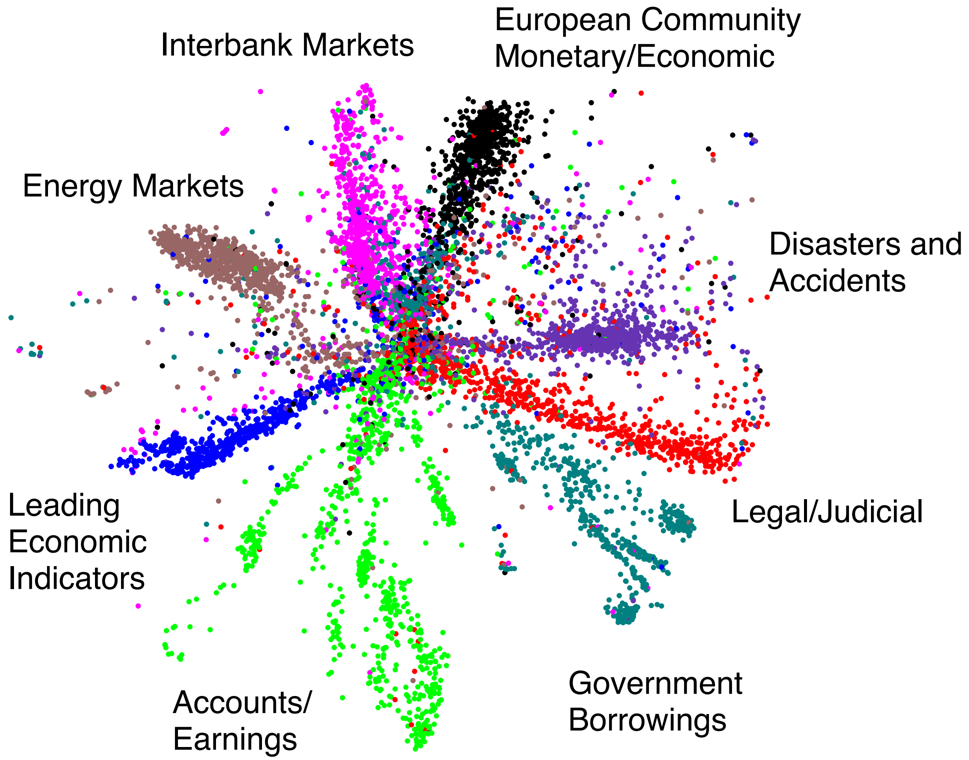
- 25x25 – 2000 – 1000 – 500 – 30 autoencoder to extract 30-D real-valued codes for Oliver face patches.

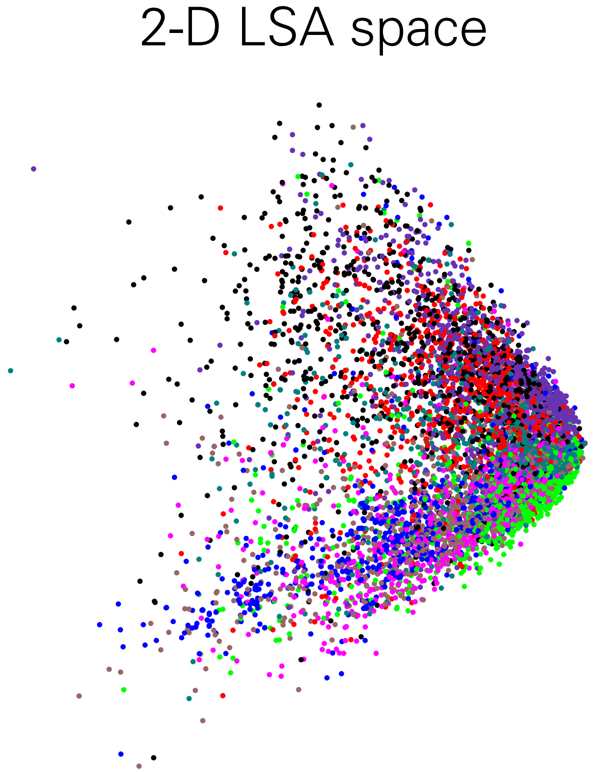


- **Top:** Random samples from the test dataset.
- **Middle:** Reconstructions by the 30-dimensional deep autoencoder.
- **Bottom:** Reconstructions by the 30-dimensional PCA.

(Hinton and Salakhutdinov, Science 2006)

Information Retrieval

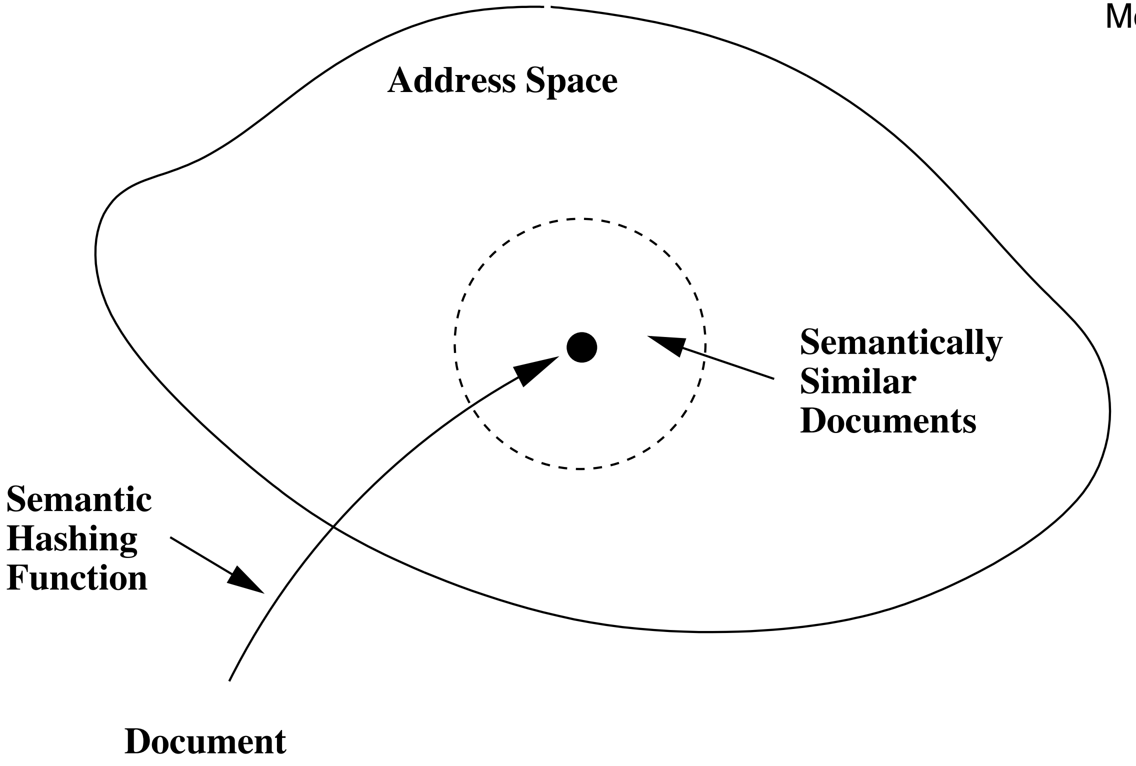


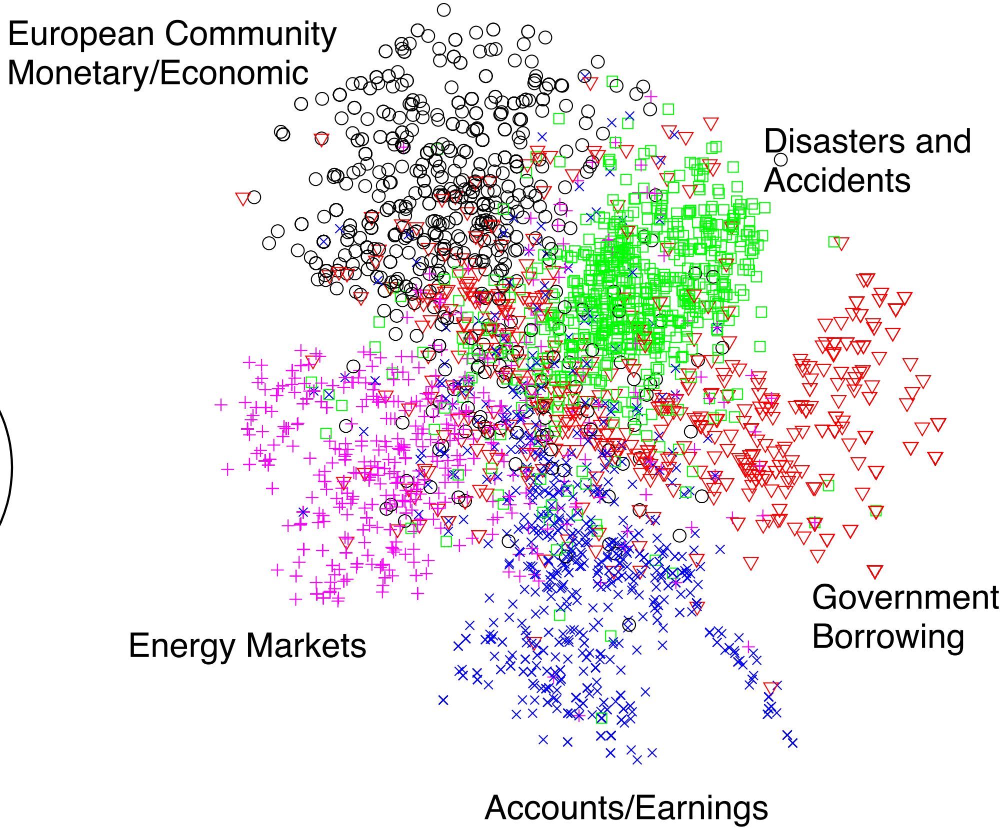


- The Reuters Corpus Volume II contains 804,414 newswire stories (randomly split into **402,207 training** and **402,207 test**).
- "Bag-of-words" representation: each article is represented as a vector containing the counts of the most frequently used 2000 words in the training set.

(Hinton and Salakhutdinov, Science 2006)

Semantic Hashing



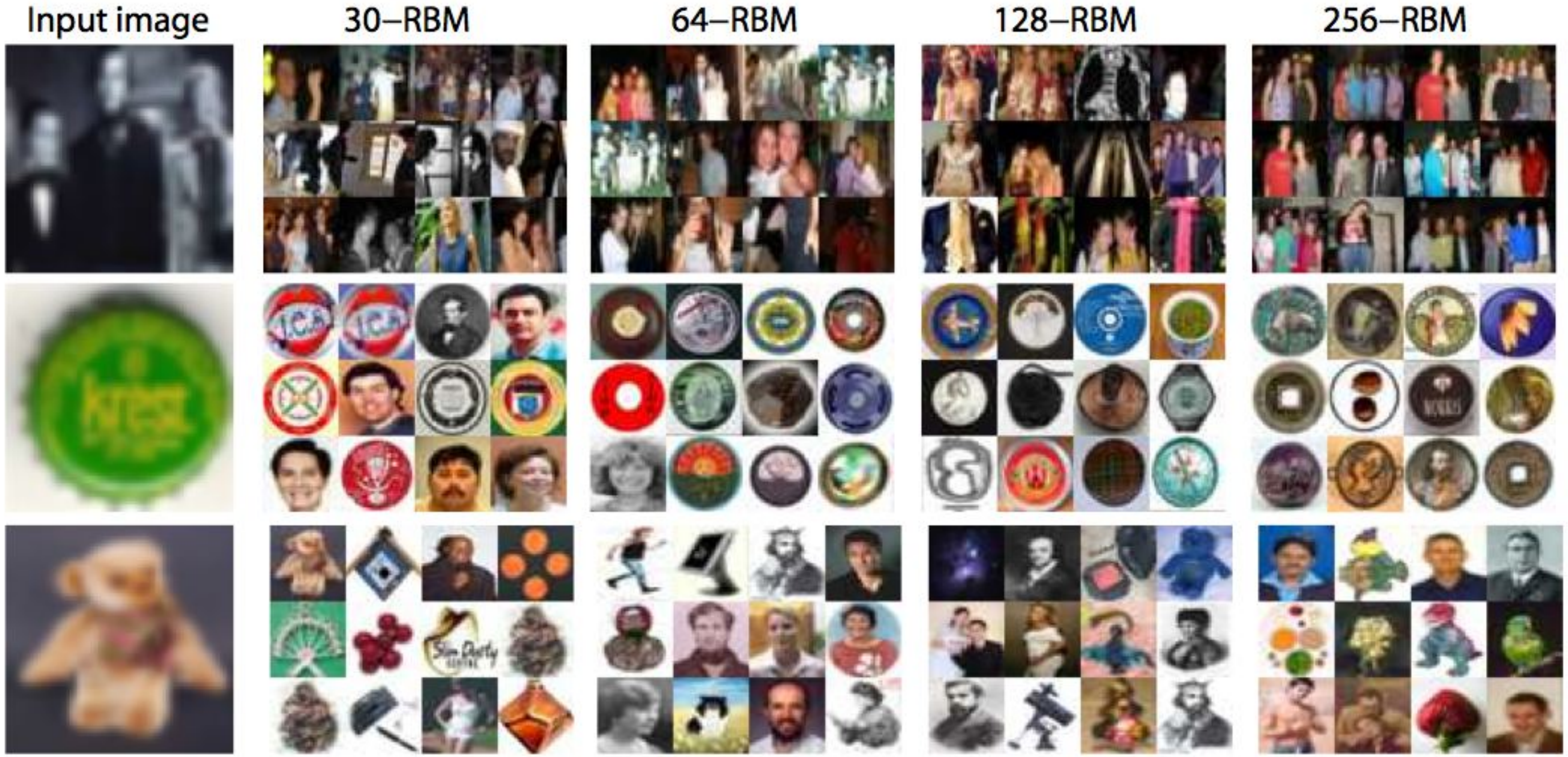


- Learn to map documents into **semantic 20-D binary codes.**
- Retrieve similar documents stored at the nearby addresses **with no search at all.**

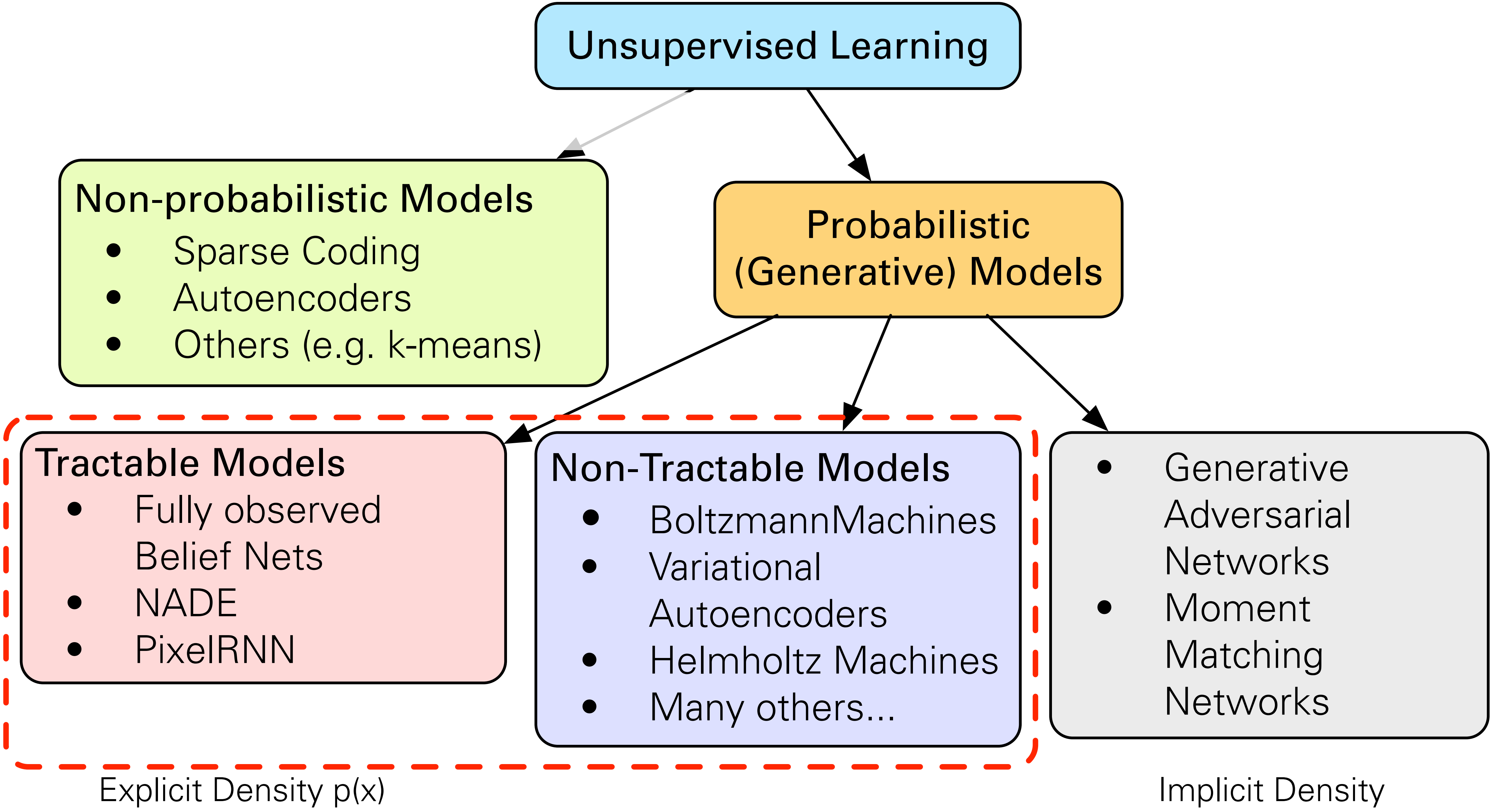
(Hinton and Salakhutdinov, Science 2006)

Searching Large Image Database using Binary Codes

- Map images into binary codes for fast retrieval.

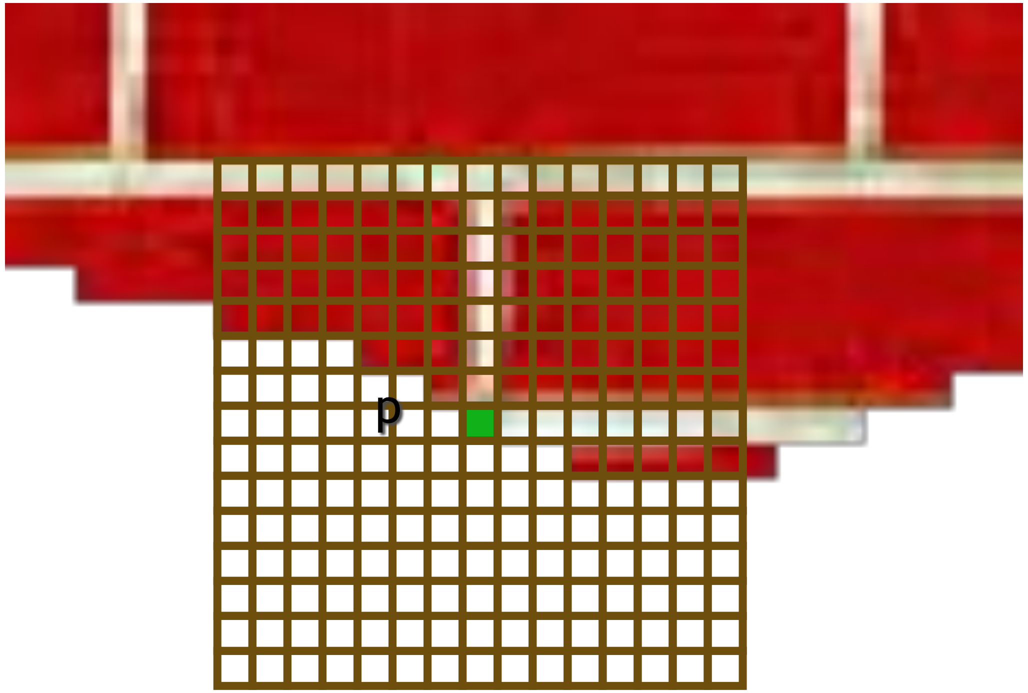


- Small Codes, Torralba, Fergus, Weiss, CVPR 2008
- Spectral Hashing, Y. Weiss, A. Torralba, R. Fergus, NIPS 2008
- Kulis and Darrell, NIPS 2009, Gong and Lazebnik, CVPR 2011
- Norouzi and Fleet, ICML 2011


Unsupervised Learning
Non-probabilistic Models
Sparse Coding
Autoencoders
Others (e.g. k-means)
Probabilistic (Generative) Models
Tractable Models
Fully observed Belief Nets
NADE
PixelRNN
Non-Tractable Models
BoltzmannMachines
Variational Autoencoders
Helmholtz Machines
Many others...
Generative Adversarial Networks
Moment Matching Networks
Explicit Density p(x)
Implicit Density

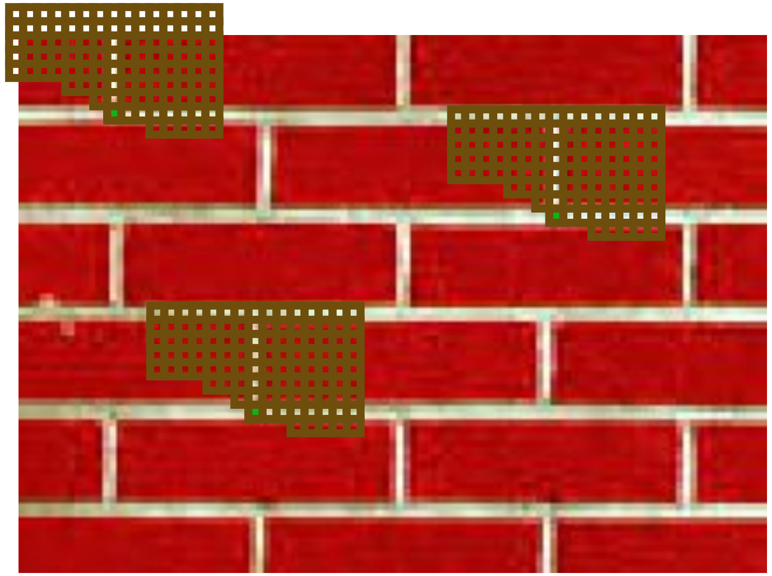
Autoregressive Generative Models

Texture synthesis by non-parametric sampling



Synthesizing a pixel

non-parametric sampling

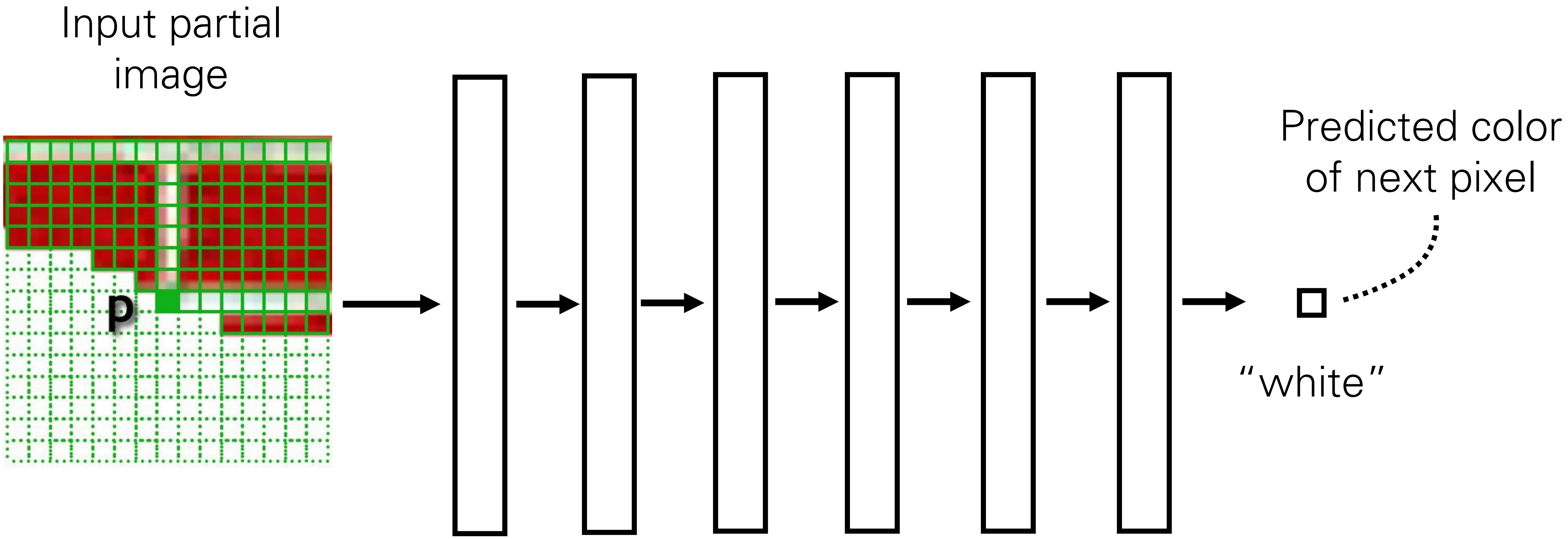


Input image

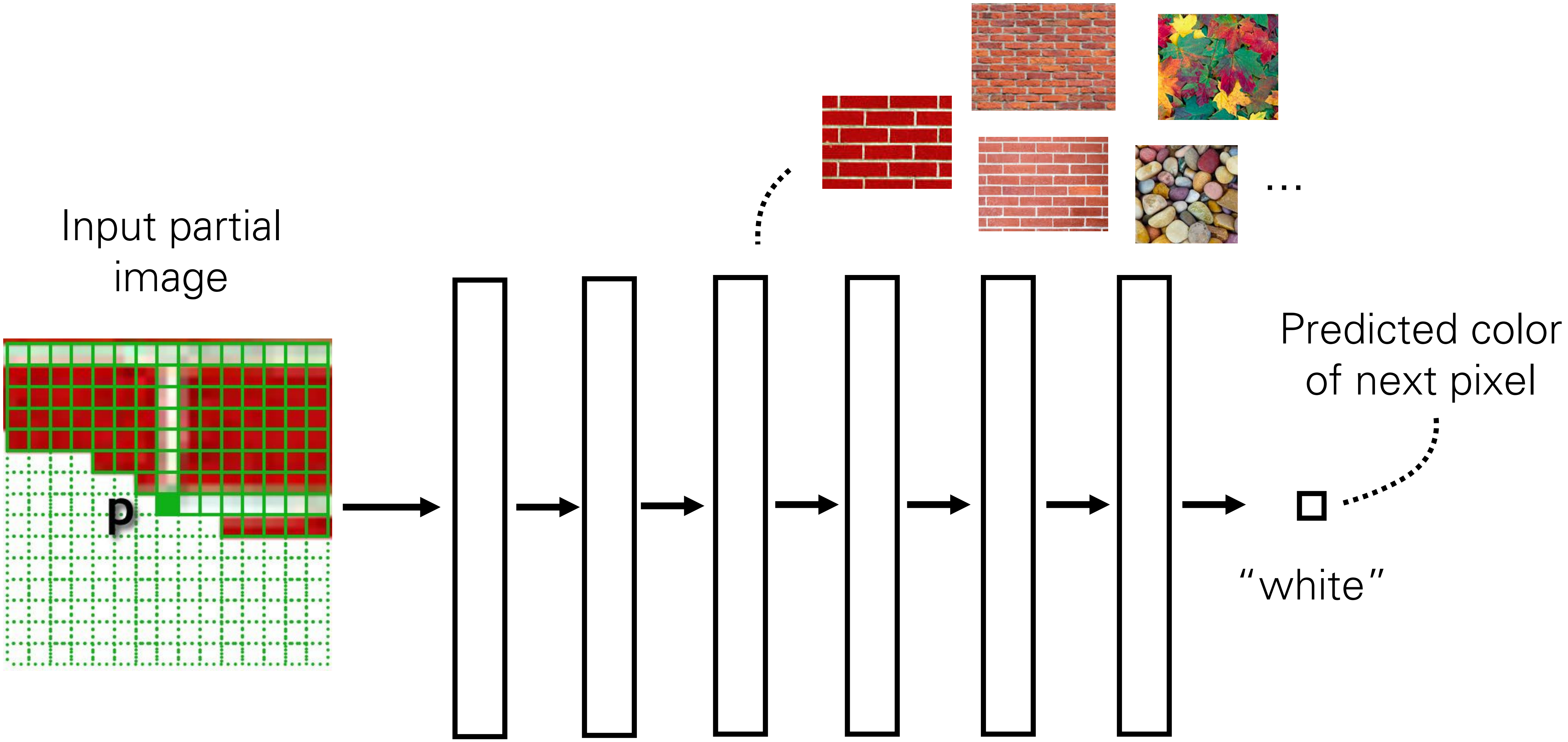
Models $P(p|N(p))$

[Efros & Leung 1999]

Texture synthesis with a deep net



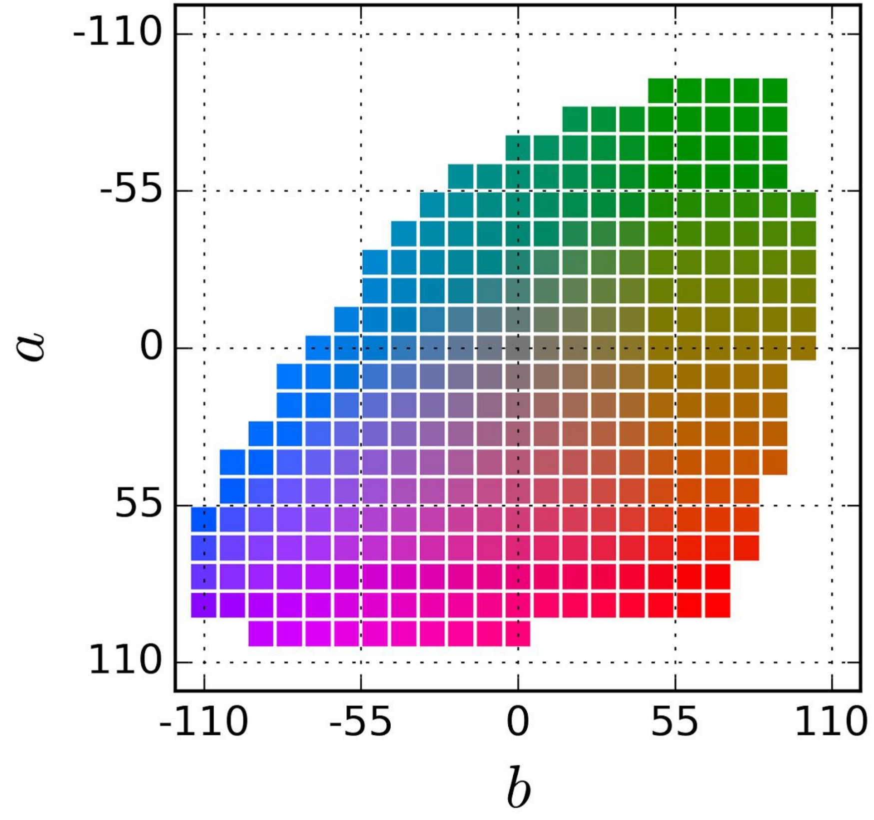
[PixelRNN, PixelCNN, van der Oord et al. 2016]

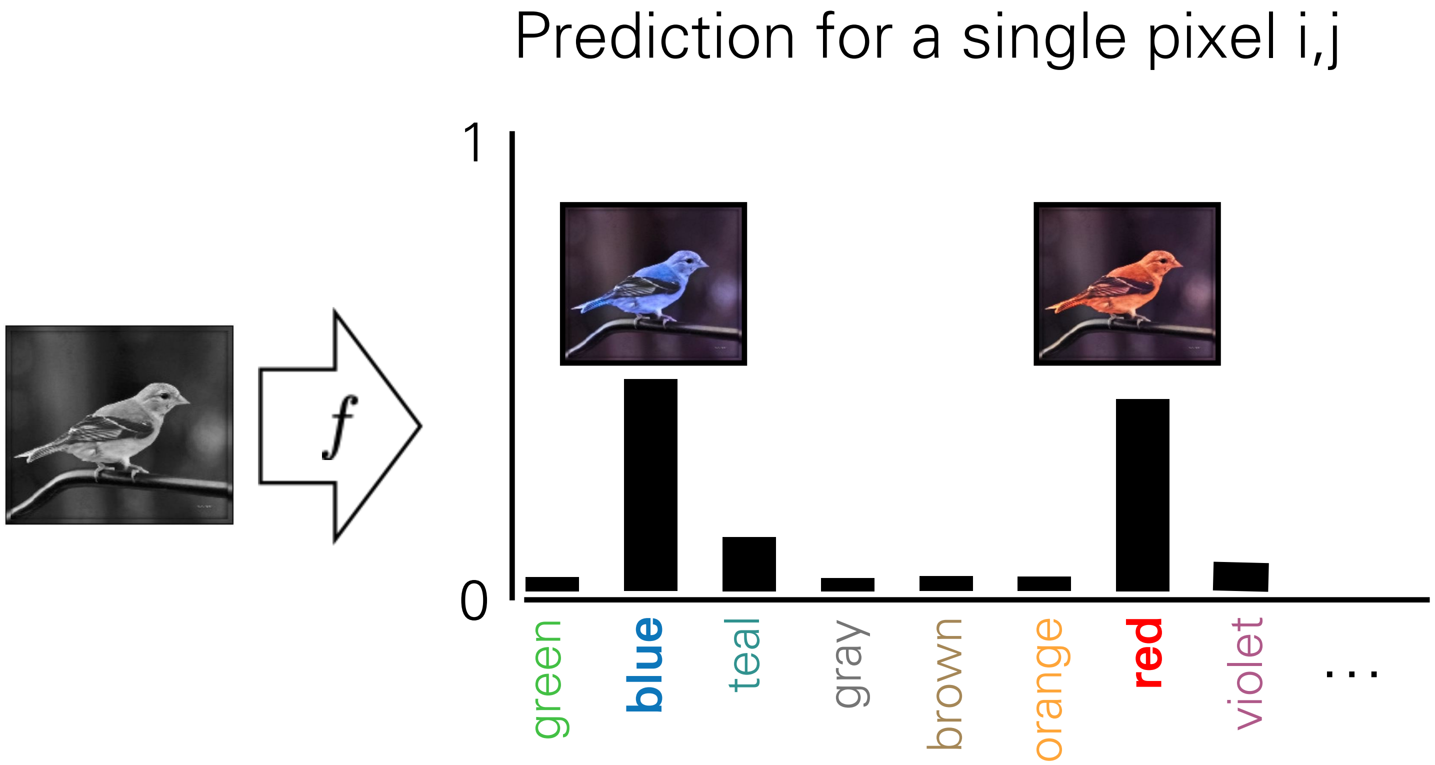


[PixelRNN, PixelCNN, van der Oord et al. 2016]

Idea: We can represent colors as discrete classes

$$\mathbf{y} \in \mathbb{R}^{H \times W \times K}$$





$$\mathcal{L}(\mathbf{y}, f_\theta(\mathbf{x})) = H(\mathbf{y}, \mathtt{softmax}(f_\theta(\mathbf{x})))$$

And we can interpret the learner as modeling P(next pixel | previous pixels):

Softmax regression (a.k.a. multinomial logistic regression)

$$\hat{\mathbf{y}} \equiv [P_\theta(Y = 1|X = \mathbf{x}), \dots, P_\theta(Y = K|X = \mathbf{x})]$$

← predicted probability of each class given input **x**

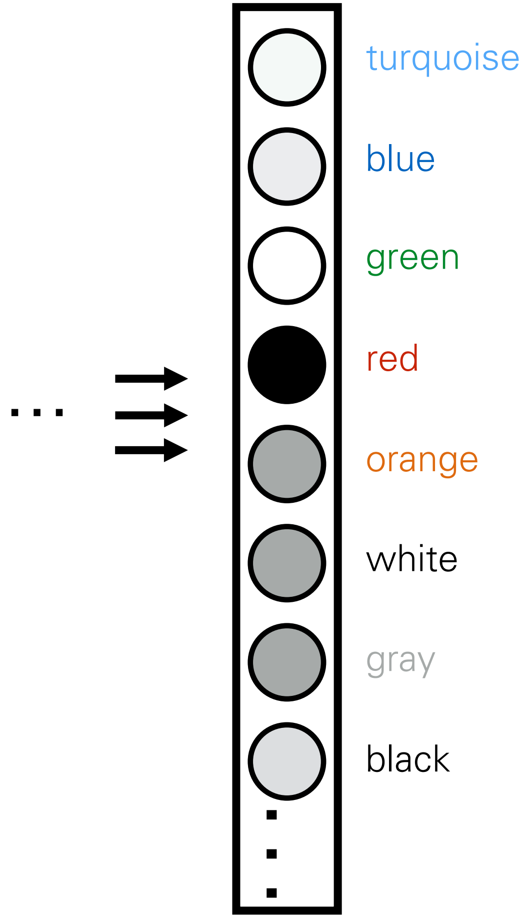
$$H(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$$

← picks out the -log likelihood of the ground truth class $\mathbf{y}$ under the model prediction $\hat{\mathbf{y}}$

$$f^* = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} H(\mathbf{y}_i, \hat{\mathbf{y}}_i)$$

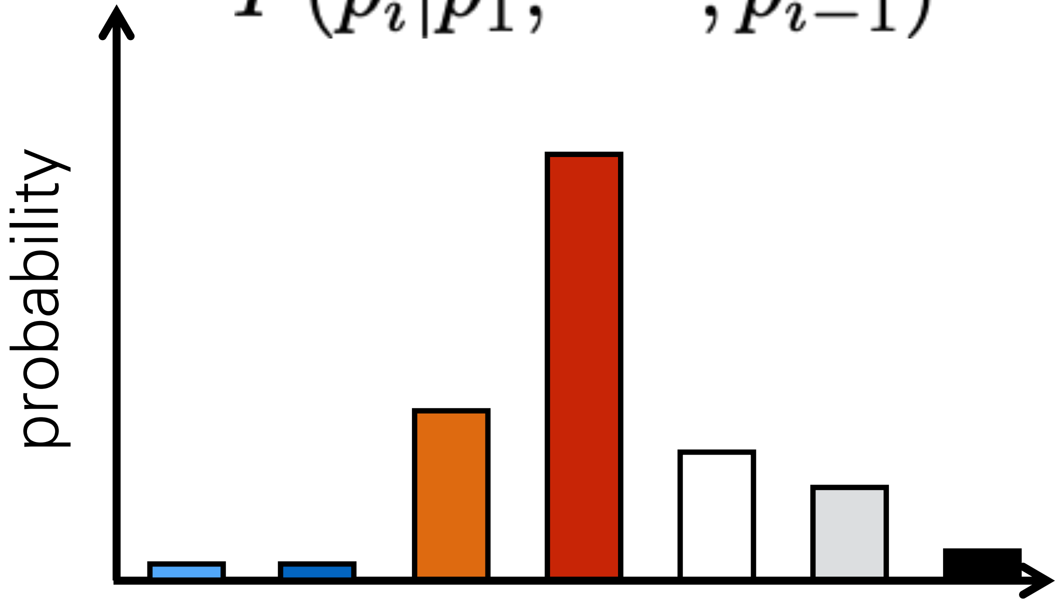
← max likelihood learner!

Network output

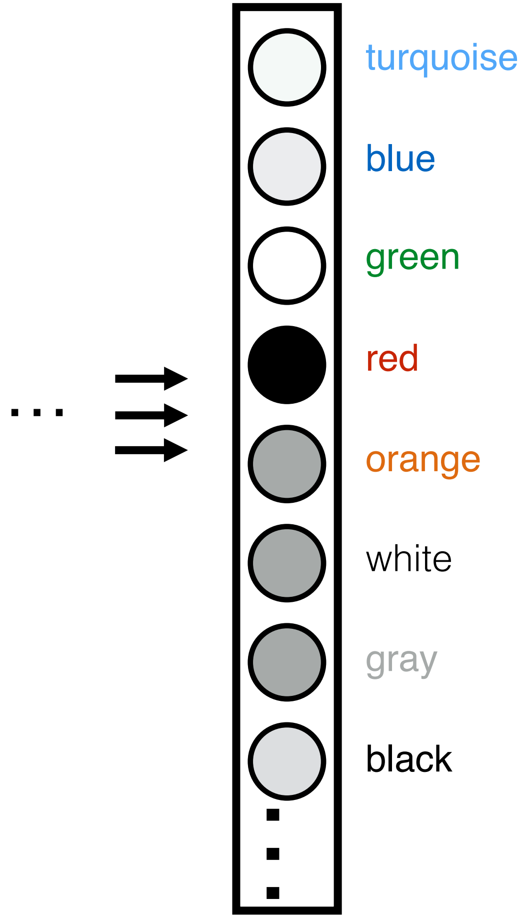


P(next pixel | previous pixels)

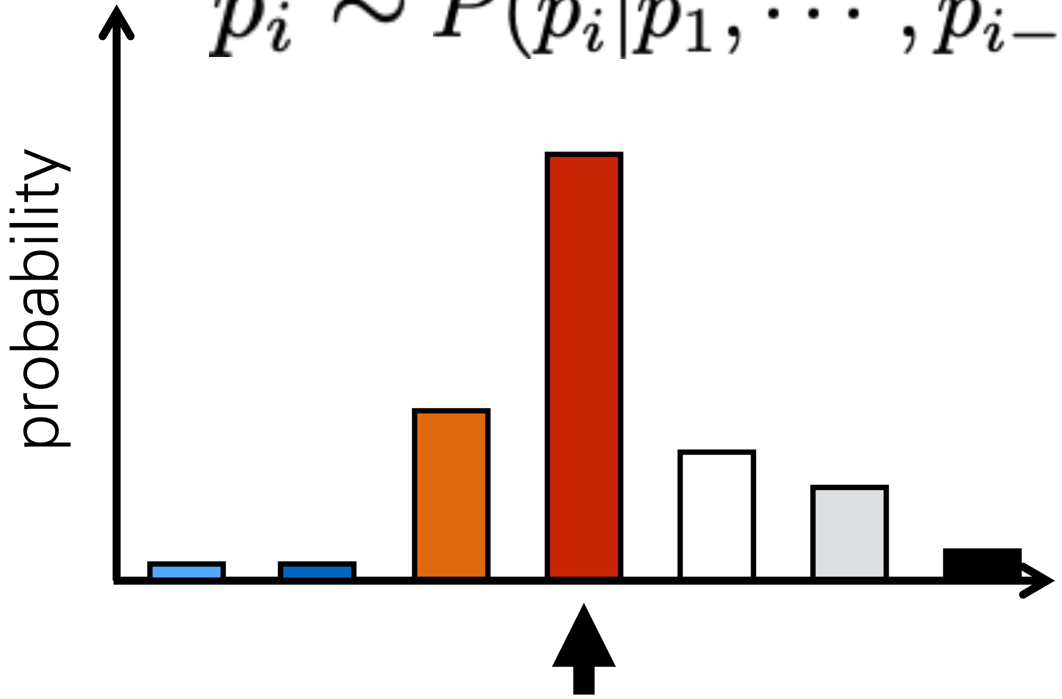
$$P(p_i | p_1, \cdots, p_{i-1})$$



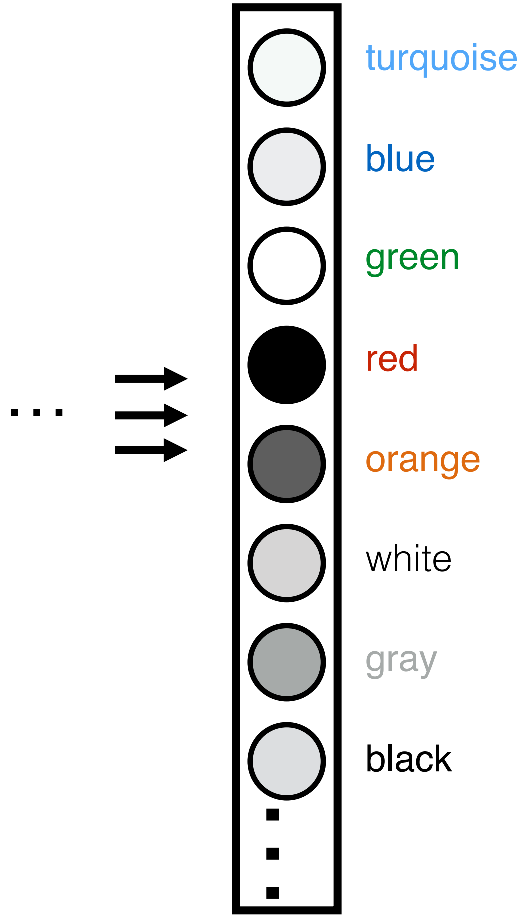
Network output



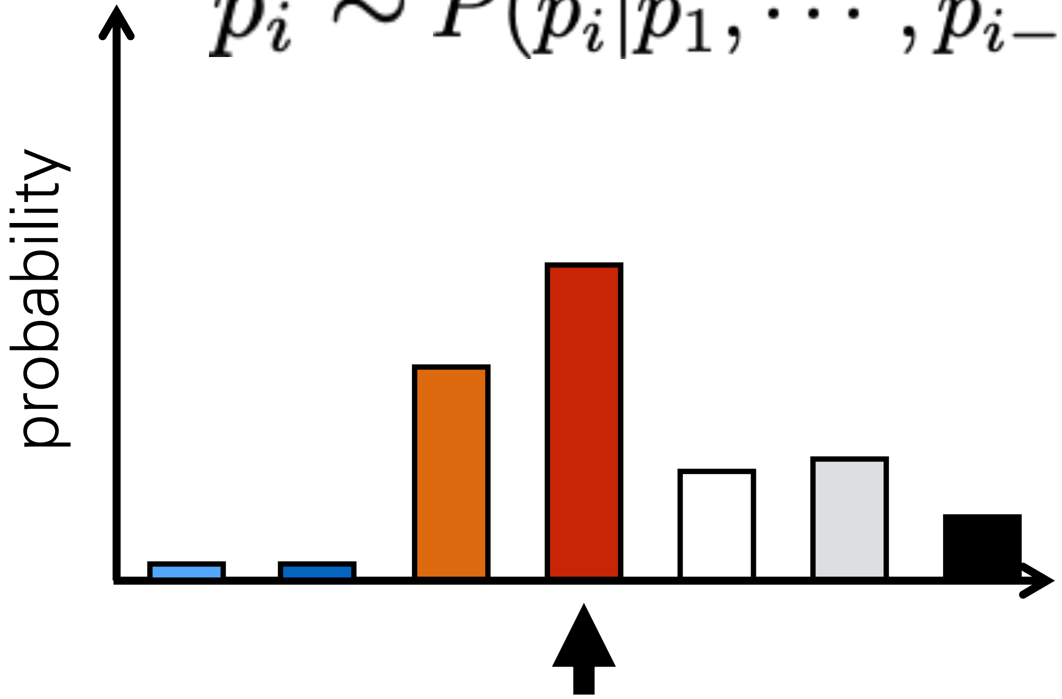
$$p_i \sim P(p_i | p_1, \cdots, p_{i-1})$$



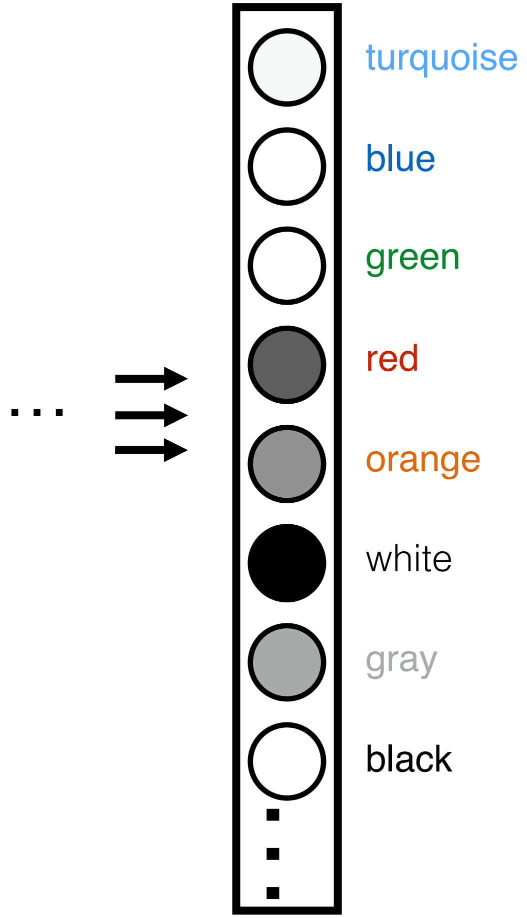
Network output



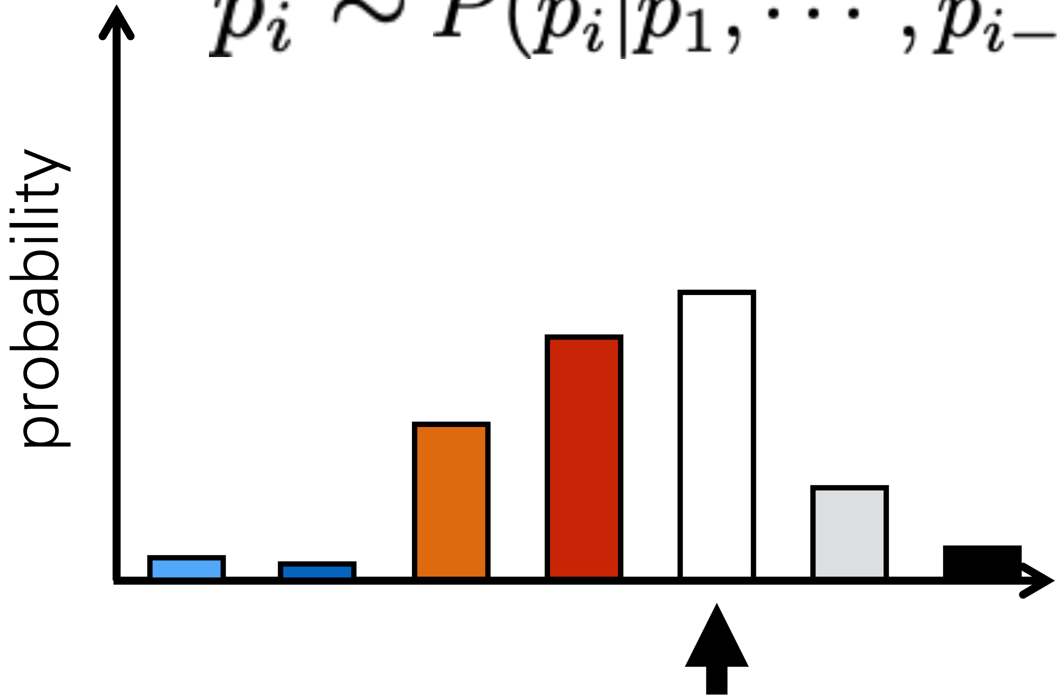
$$p_i \sim P(p_i | p_1, \cdots, p_{i-1})$$

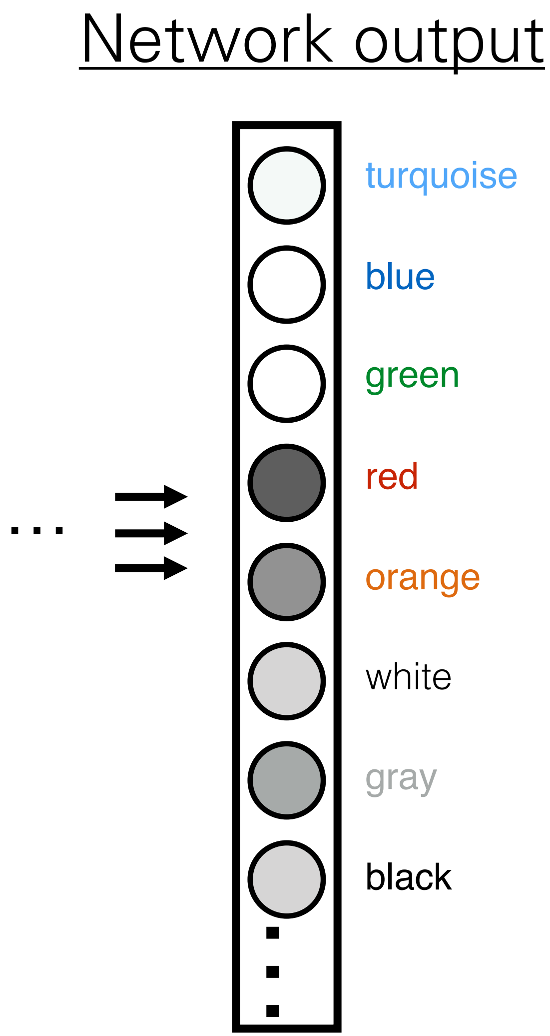


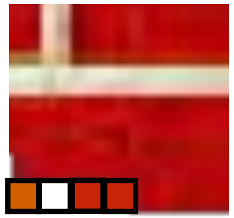
Network output



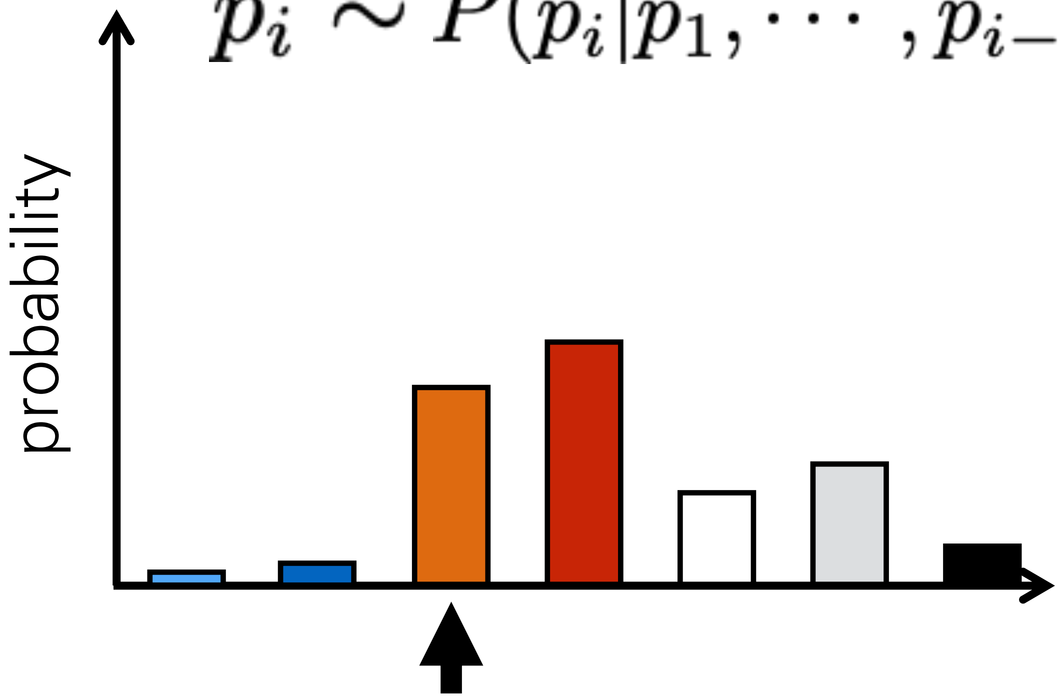
$$p_i \sim P(p_i | p_1, \cdots, p_{i-1})$$

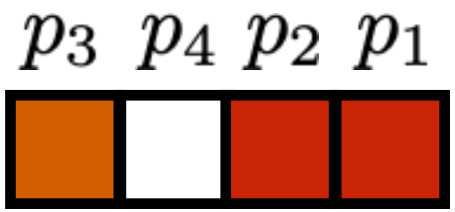






$$p_i \sim P(p_i | p_1, \cdots, p_{i-1})$$





$$p_1 \sim P(p_1)$$

$$p_2 \sim P(p_2|p_1)$$

$$p_3 \sim P(p_3|p_1, p_2)$$

$$p_4 \sim P(p_4|p_1, p_2, p_3)$$

$$\{p_1, p_2, p_3, p_4\} \sim P(p_4|p_1, p_2, p_3)P(p_3|p_1, p_2)P(p_2|p_1)P(p_1)$$

$$p_i \sim P(p_i|p_1, \ldots, p_{i-1})$$

$$\boxed{\mathbf{p} \sim \prod_{i=1}^{N} P(p_i|p_1, \ldots, p_{i-1})}$$

Autoregressive probability model

$$\mathbf{p} \sim \prod_{i=1}^{N} P(p_i | p_1, \ldots, p_{i-1})$$

$$P(\mathbf{p}) = \prod_{i=1}^{N} P(p_i | p_1, \ldots, p_{i-1}) \quad \longleftarrow \quad \text{General product rule}$$

The sampling procedure we defined above takes exact samples from the learned probability distribution (pmf).

Multiplying all conditionals evaluates the probability of a full joint configuration of pixels.

Learning the Distribution of Natural Data

$$p(\mathbf{x}) = \prod_i p(x_i|\mathbf{x}_<)$$

for 1D sequences such as text or sound

Learning the Distribution of Natural Data

$$p(\mathbf{x}) = \prod_{j} \prod_{i} p(x_{i,j} | \mathbf{x}_{<})$$

Autoregressive model for 2D tensors such as images

Learning the Distribution of Natural Data

$$p(\mathbf{x}) = \prod_k \prod_j \prod_i p(x_{i,j,k} | \mathbf{x}_<)$$

And for 3D tensors such as videos

Learning the Distribution of Natural Data

$$p(\mathbf{x}) = \prod_{k} \prod_{j} \prod_{i} p(x_{i,j,k} | \mathbf{x}_{<})$$

PixelRNN/PixelCNN (Images) [van den Oord, Kalchbrenner, Kavukcuoglu, 2016]

[van den Oord, Kalchbrenner, Vinyals, et al, 2016]

Video Pixel Nets (Videos) [Kalchbrenner, van den Oord, Simonyan, et al, 2016]

ByteNet (Language/seq2seq) [Kalchbrenner, Espeholt, Simonyan, et al, 2016]

WaveNet (Audio) [van den Oord, Dieleman, Zen, et al, 2016]

Learning the Distribution of Natural Data

$$p(\mathbf{x}) = \prod_{k} \prod_{j} \prod_{i} p(x_{i,j,k} | \mathbf{x}_{<})$$

Prior work:

Autoregressive image models: [Larochelle, Murray, 2011] [Theis, Bethge, 2015] [Uria, et al 2016]

Dilated convolutions: [Chen et al, 2015] [Yu, Koltun, 2016] [Holschneider, et al, 1989]

RNN and language/translation modelling: [Hochreiter, Schmidhuber, 1997] [Mikolov et al, 2010] [Kalchbrenner, Blunsom 2013] [Sutskever et al, 2014] [Stollenga et al, 2015] [Kaiser and Bengio, 2016]

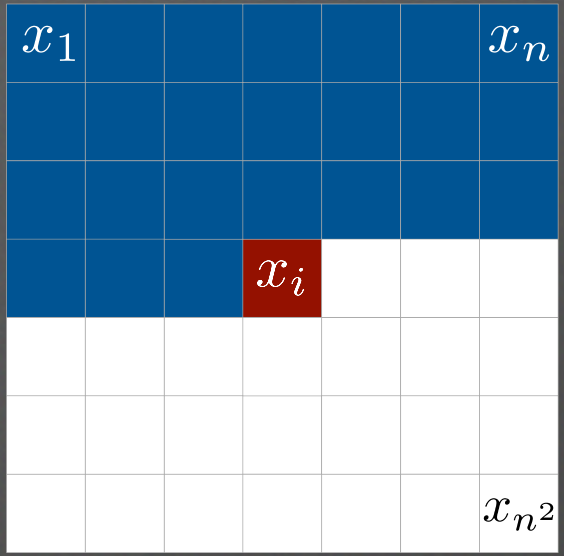
Pixel RNN

$P($  $)$

Pixel RNN

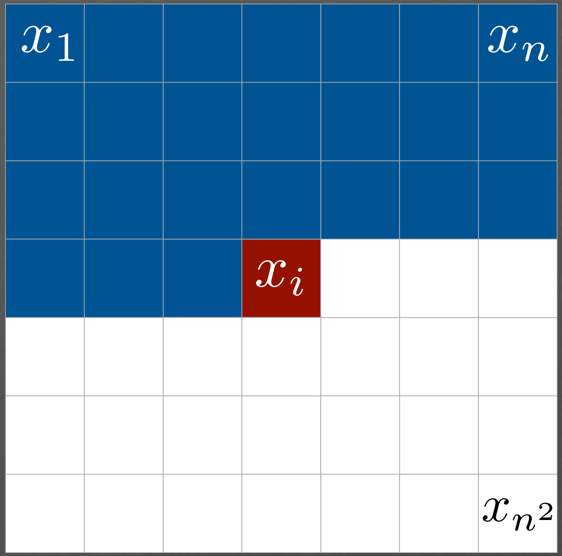
P(

)



Pixel RNN

P()

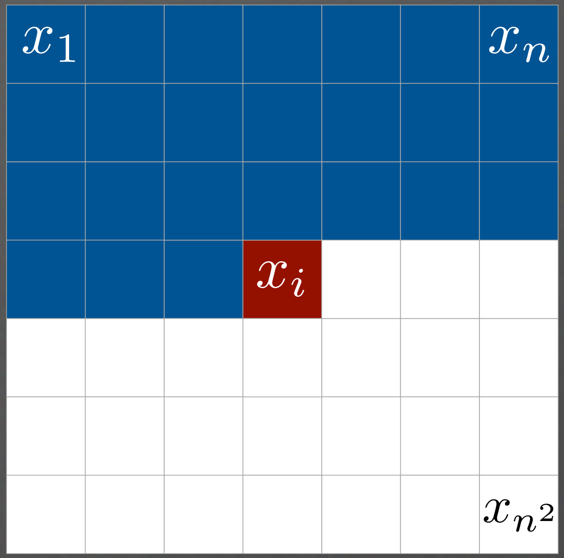


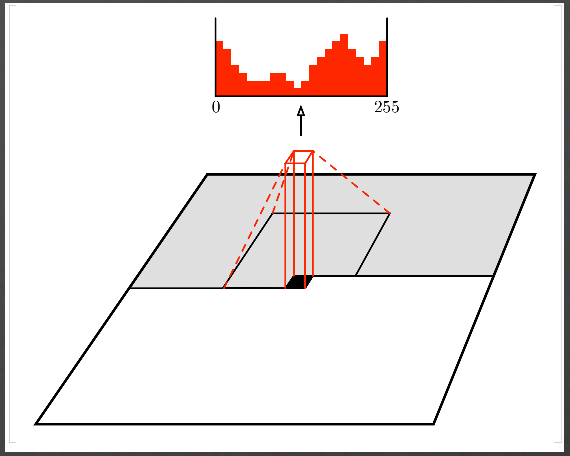
By chain rule and using pixels as variables,

$$P(X) = P(x_1)\, P(x_2|x_1)\, P(x_3|x_1,x_2)$$

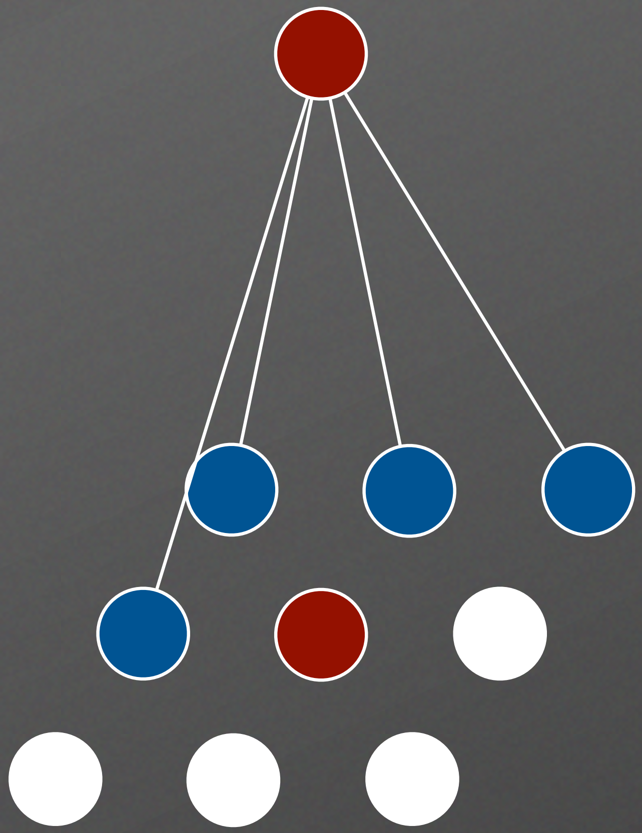
Pixel RNN

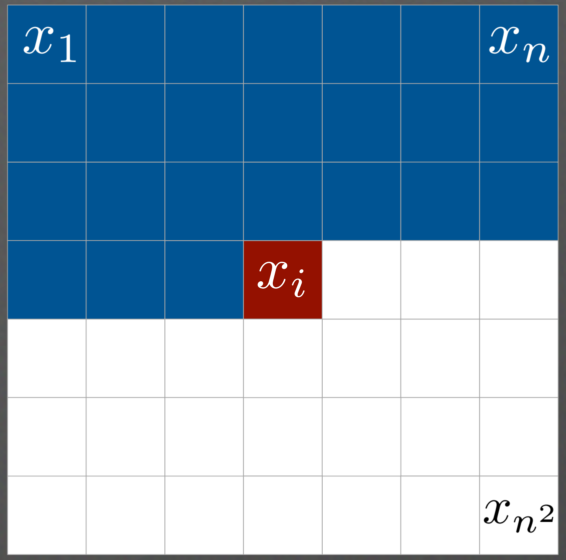
P()





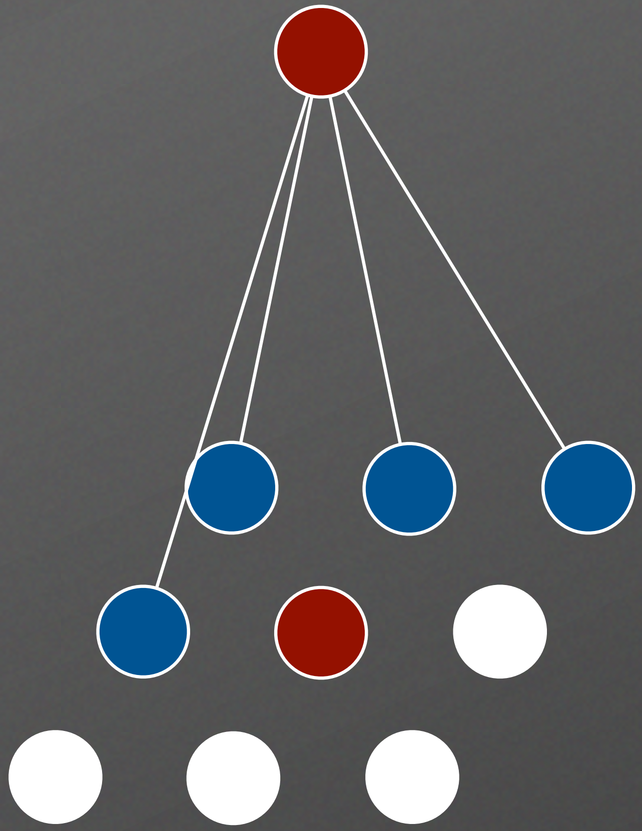
Masked Convolutions

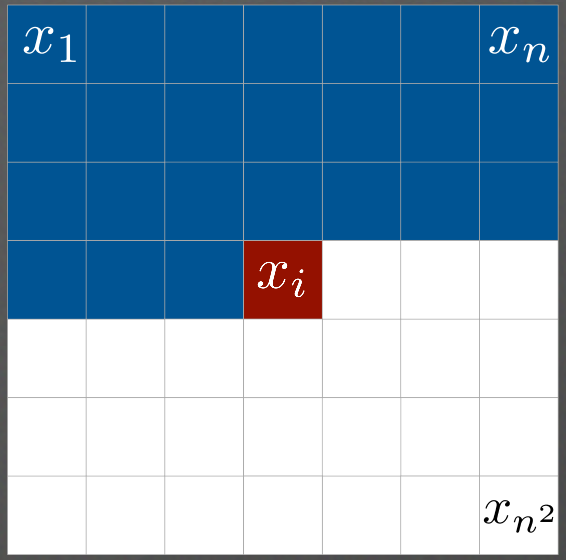


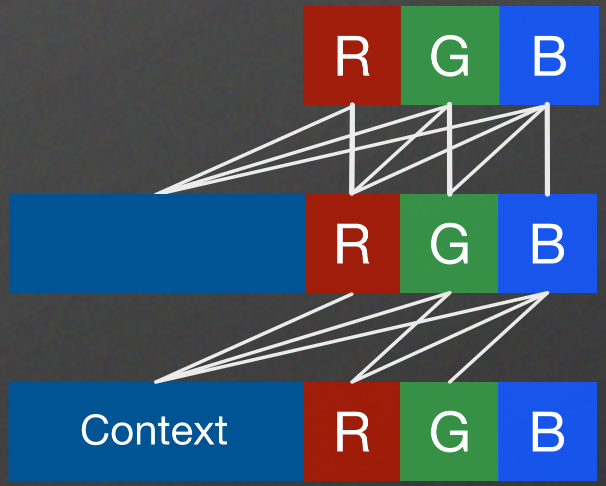


1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

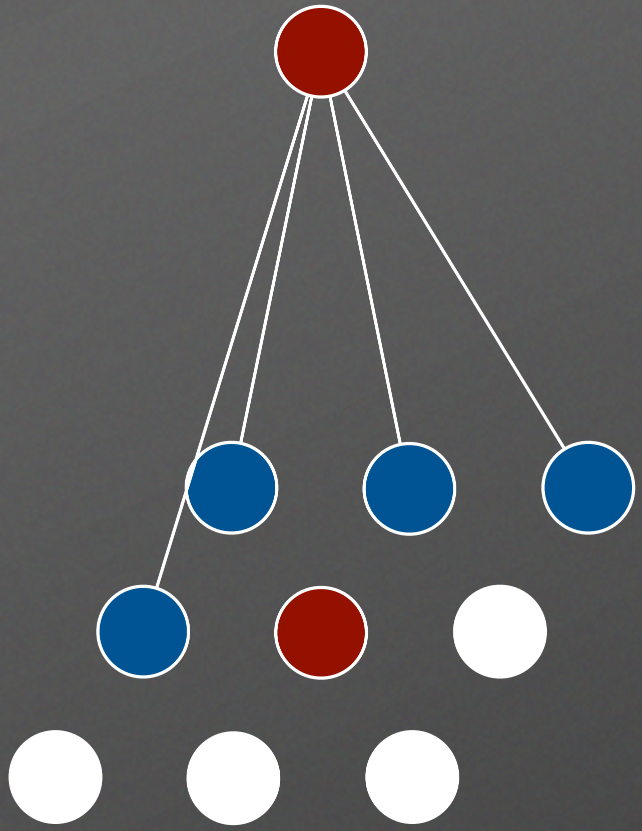
Masked Convolutions

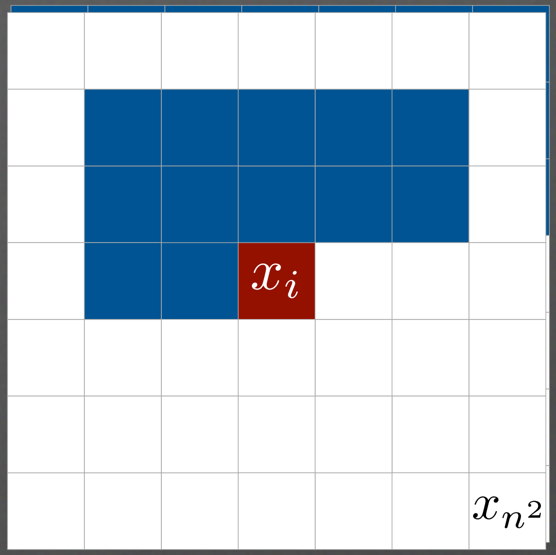


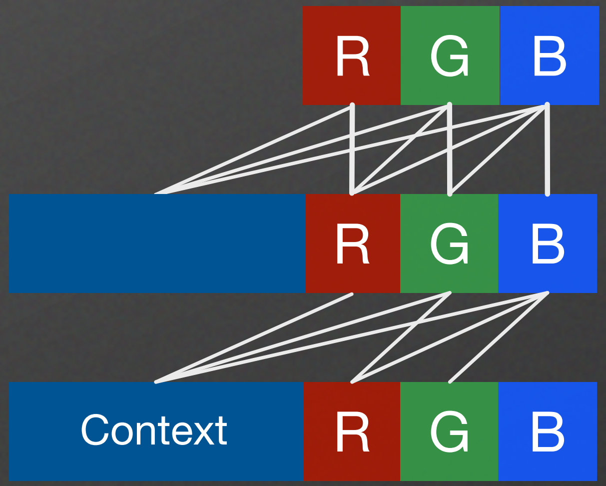




Pixel CNN

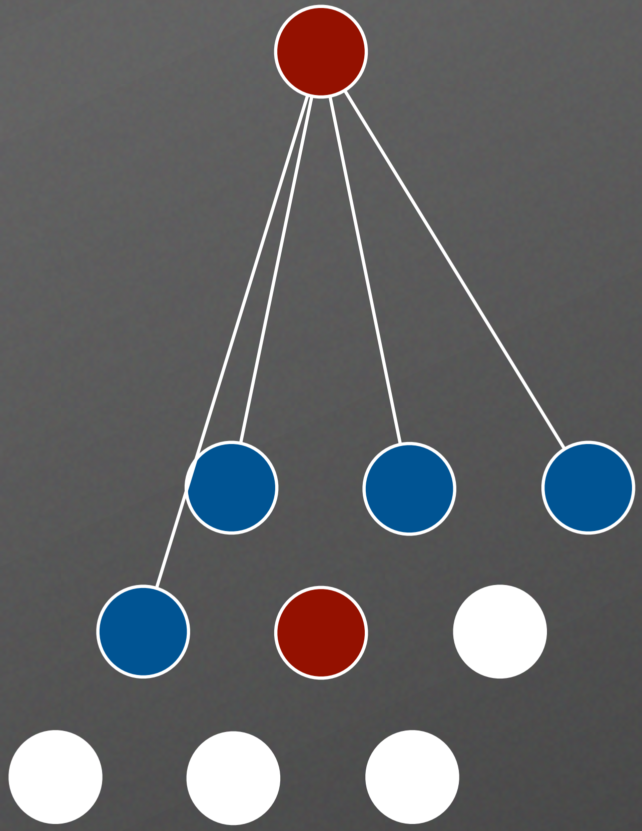


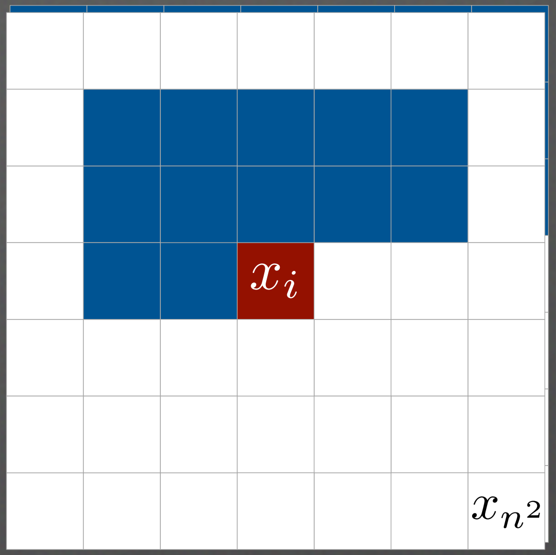


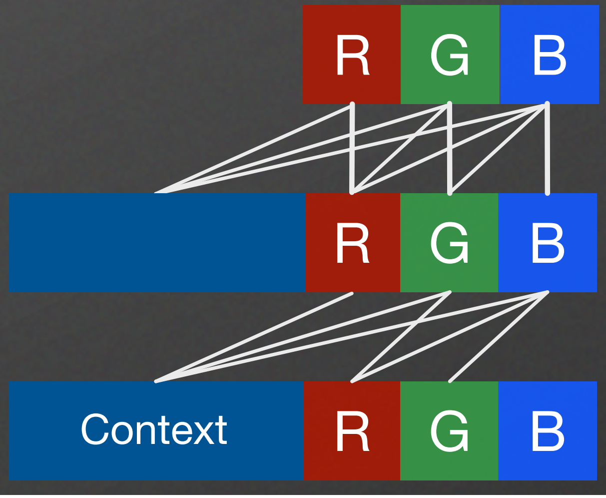


Pixel CNN

Multiple layers of masked convolutions

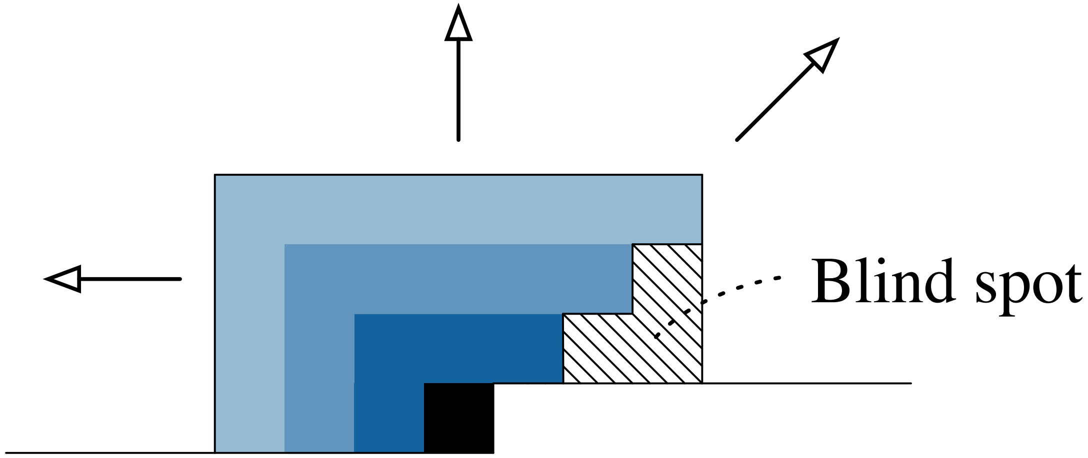




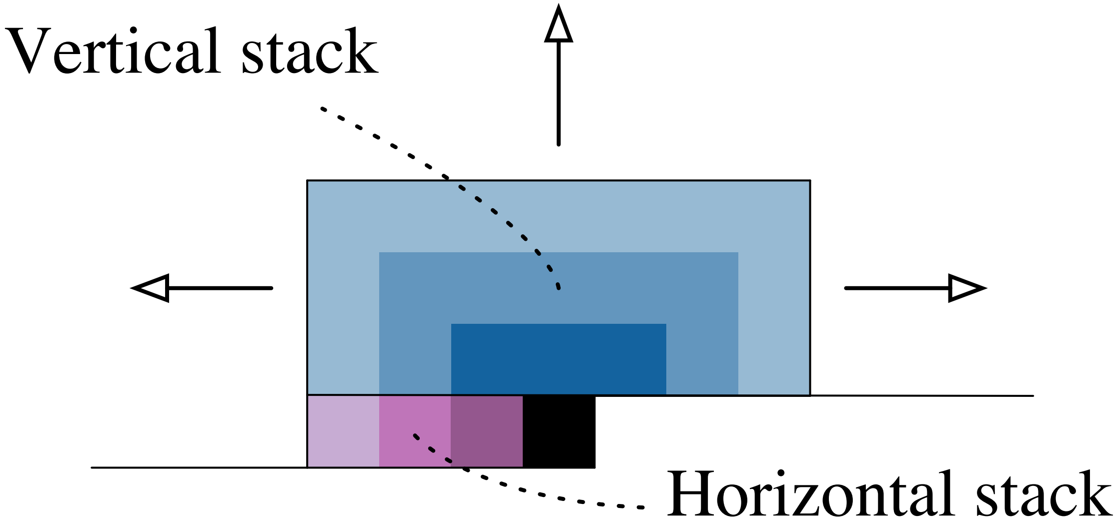


$$p(x_i \mid \mathbf{x}_{<i}) = p(x_{i,R} \mid \mathbf{x}_{<i})p(x_{i,G} \mid x_{i,R}, \mathbf{x}_{<i})p(x_{i,B} \mid x_{i,R}, x_{i,G}, \mathbf{x}_{<i})$$

Improving PixelCNN



Stacking layers of masked convolution creates a blindspot

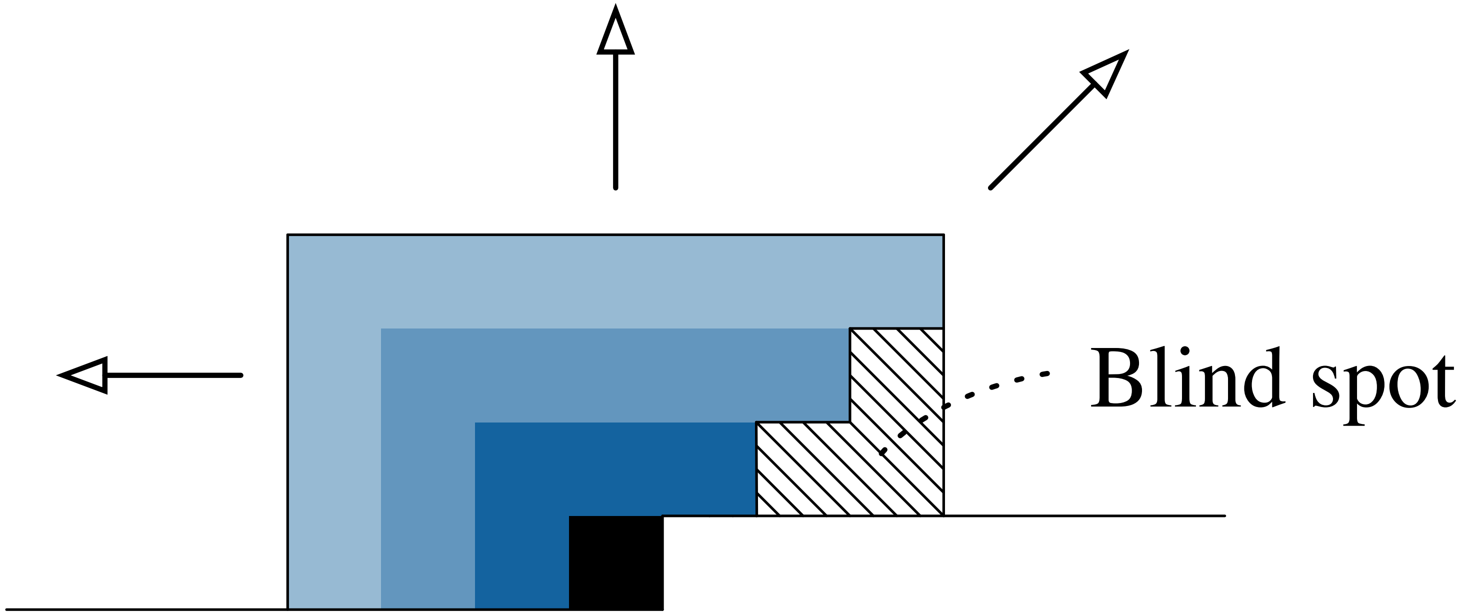


Solution: use two stacks of convolution, a vertical stack and a horizontal stack

Improving PixelCNN I

There is a problem with this form of masked convolution.

1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

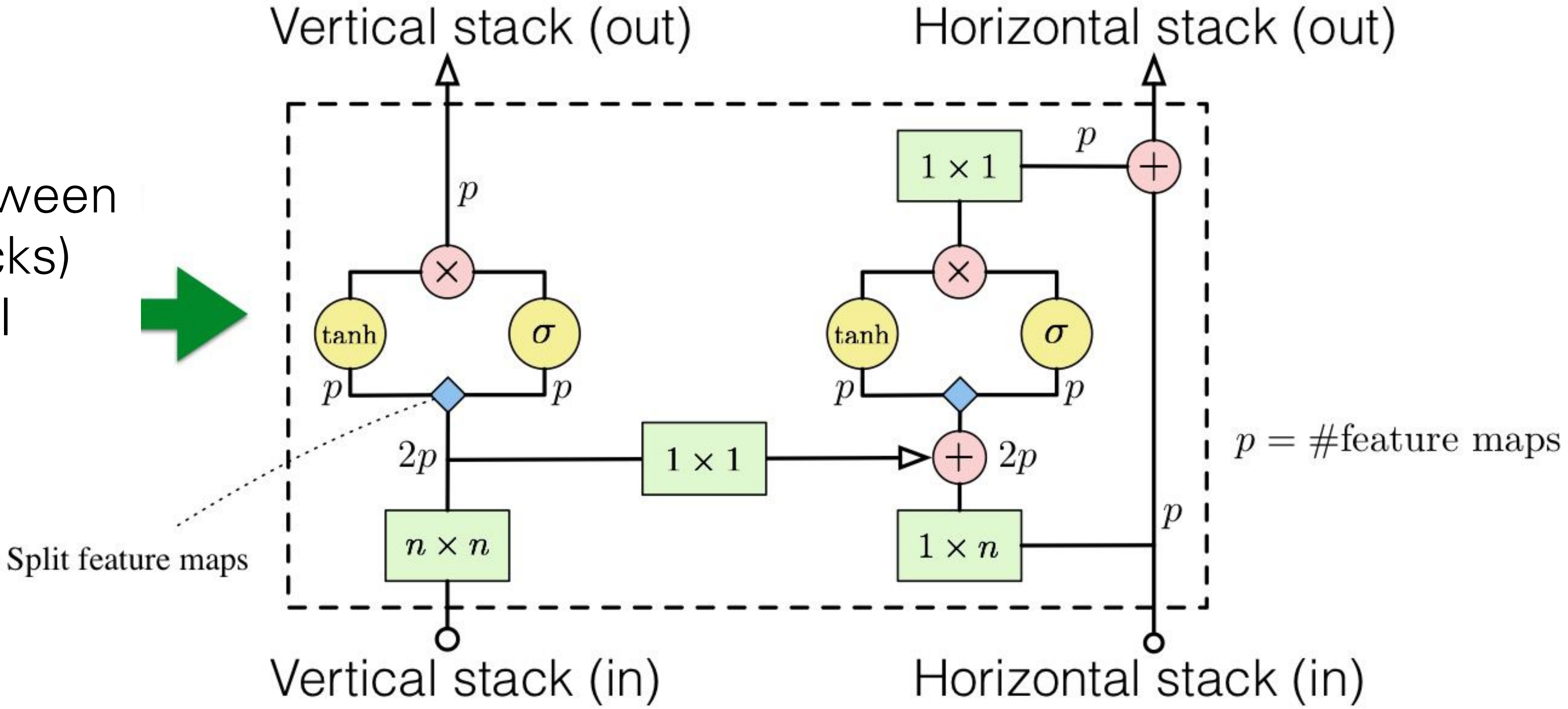


Stacking layers of masked convolution creates a blindspot

Improving PixelCNN II

Use more expressive nonlinearity: $\mathbf{h}_{k+1} = \tanh(W_{k,f} * \mathbf{h}_k) \odot \sigma(W_{k,g} * \mathbf{h}_k)$

This information flow (between vertical and horizontal stacks) preserves the correct pixel dependencies



Samples from PixelCNN

Topics: CIFAR-10

- Samples from a class-conditioned PixelCNN

Coral Reef

Samples from PixelCNN

Topics: CIFAR-10

- Samples from a class-conditioned PixelCNN

Sorrel horse

Samples from PixelCNN

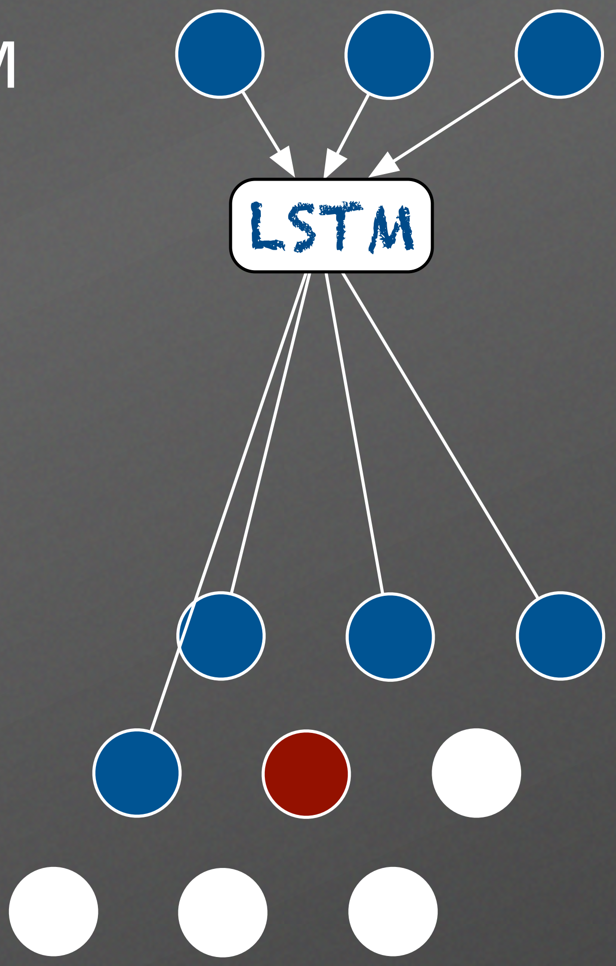
Topics: CIFAR-10

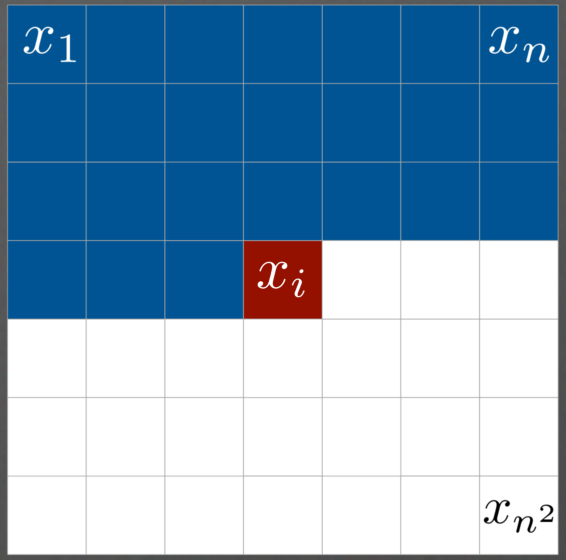
- Samples from a class-conditioned PixelCNN

Sandbar

Convolutional Long Short-Term Memory

Row LSTM

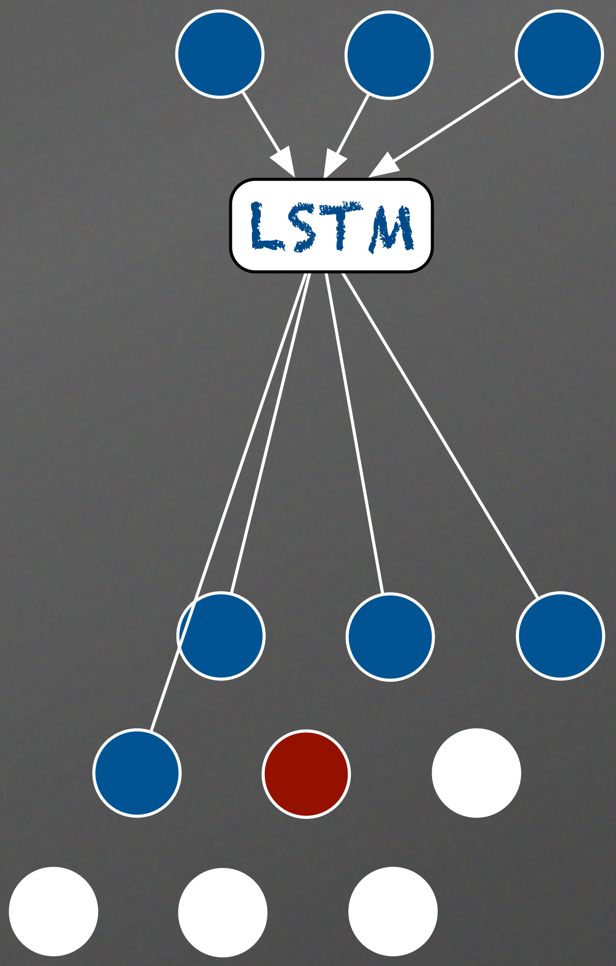


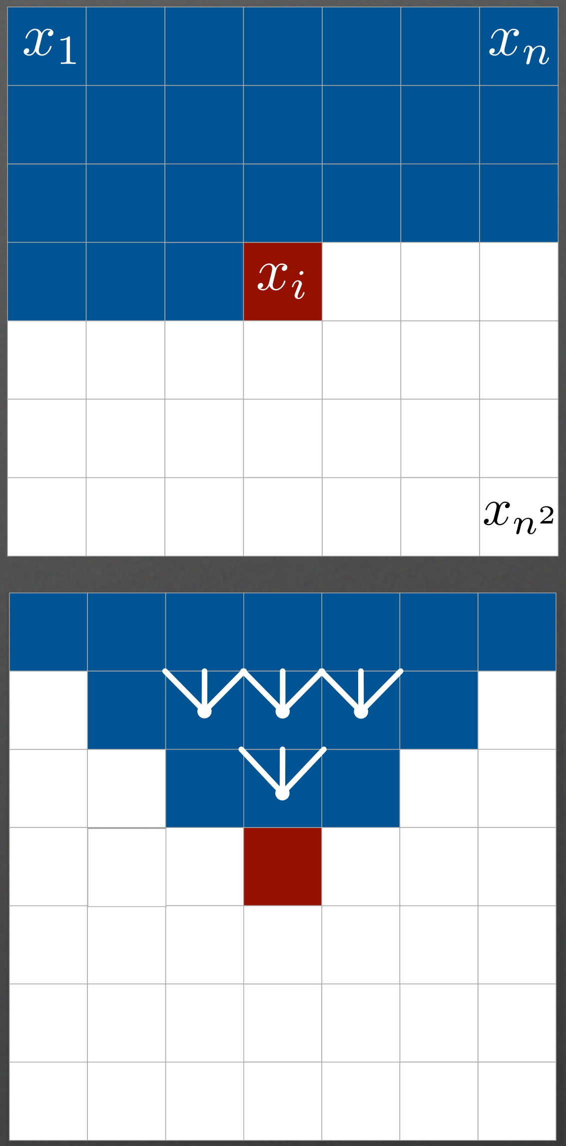


Stollenga et al, 2015
Oord, Kalchbrenner, Kavukcuoglu, 2016

Pixel RNN

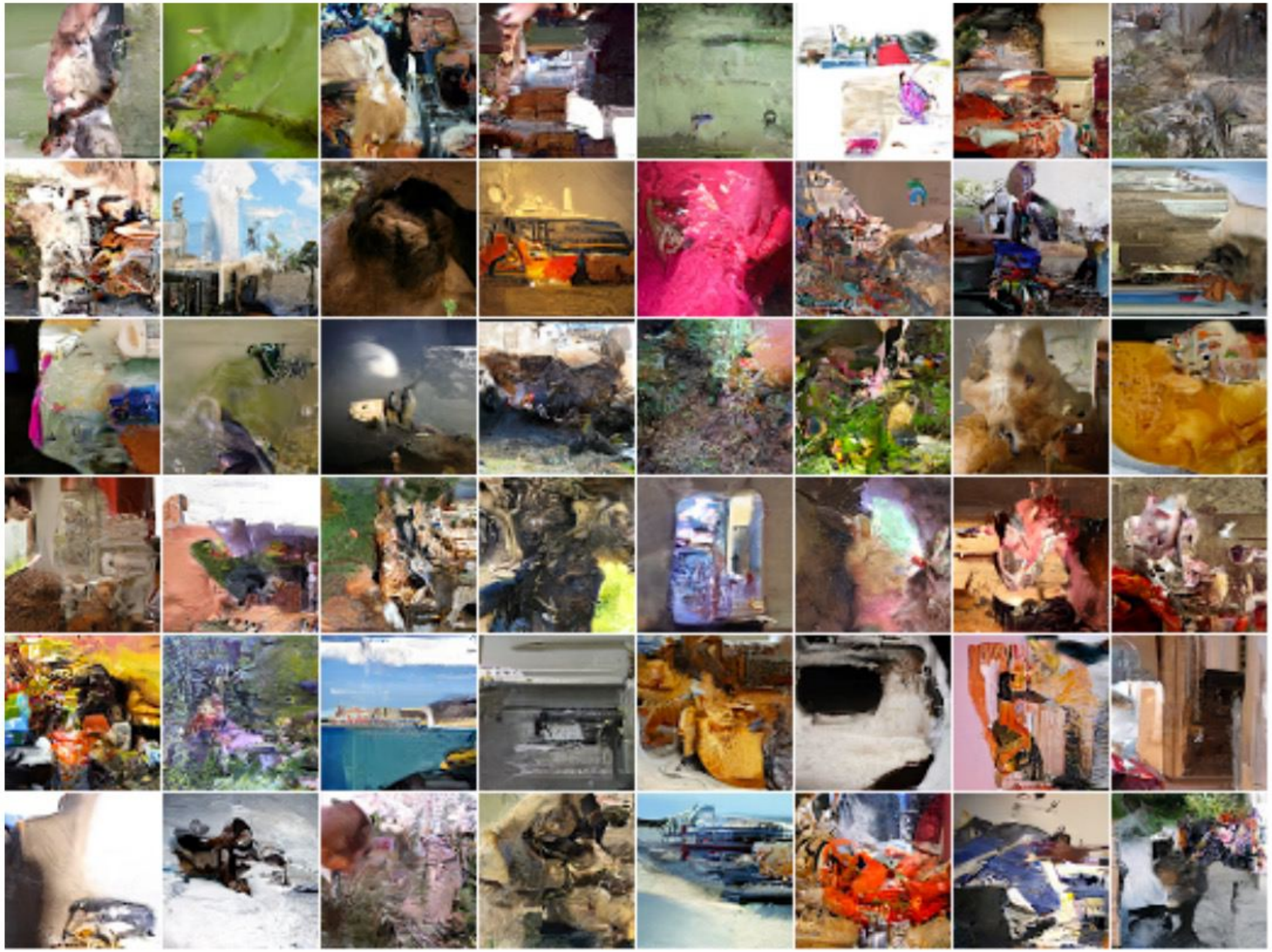
Multiple layers of convolutional LSTM





Samples from PixelRNN

Samples from PixelRNN

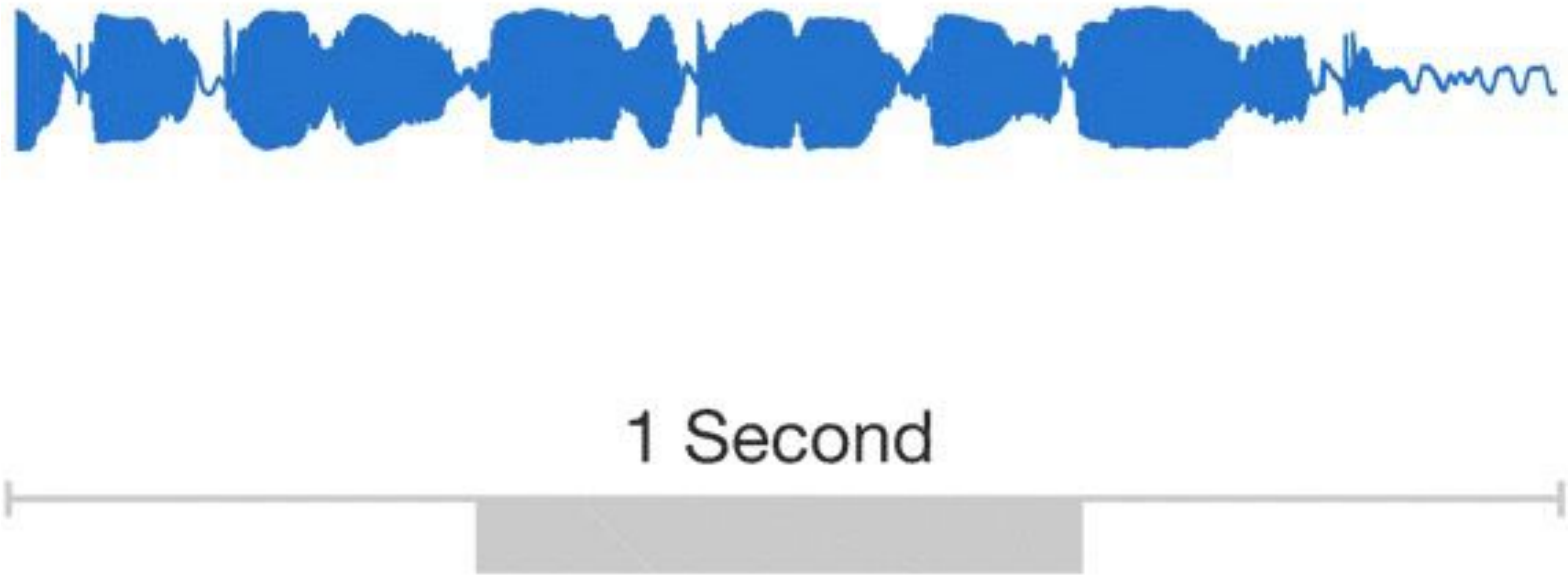


[PixelRNN, van der Oord et al. 2016]

Image completions (conditional samples) from PixelRNN

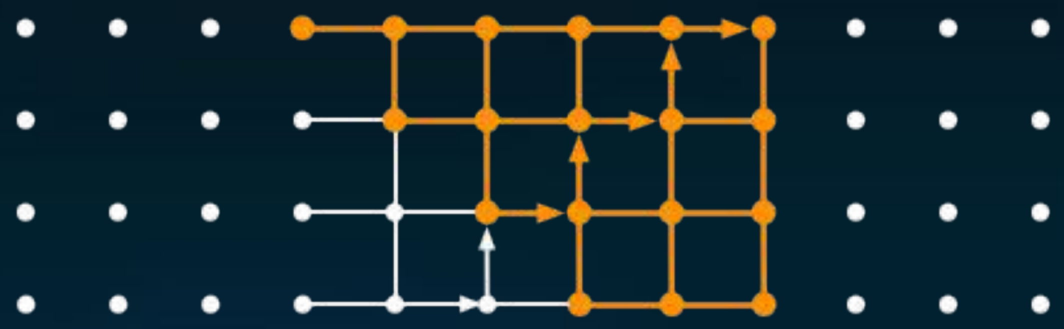
[PixelRNN, van der Oord et al. 2016]

Modeling Audio



Architecture for 1D sequences (Bytenet / Wavenet)

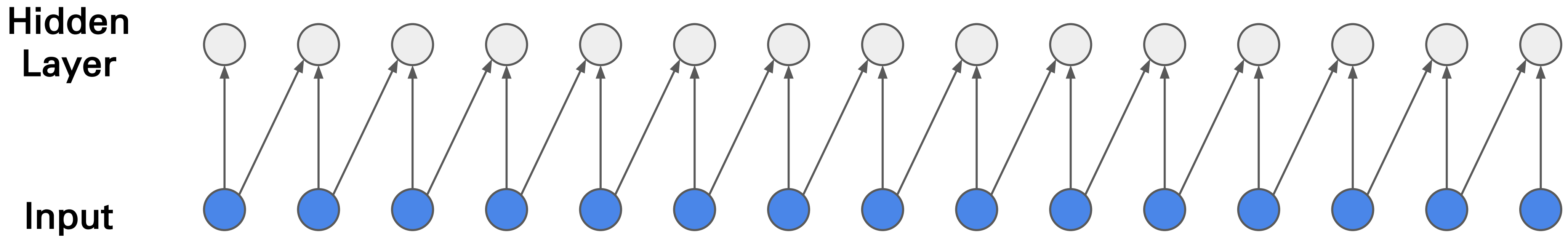
Deep RNN



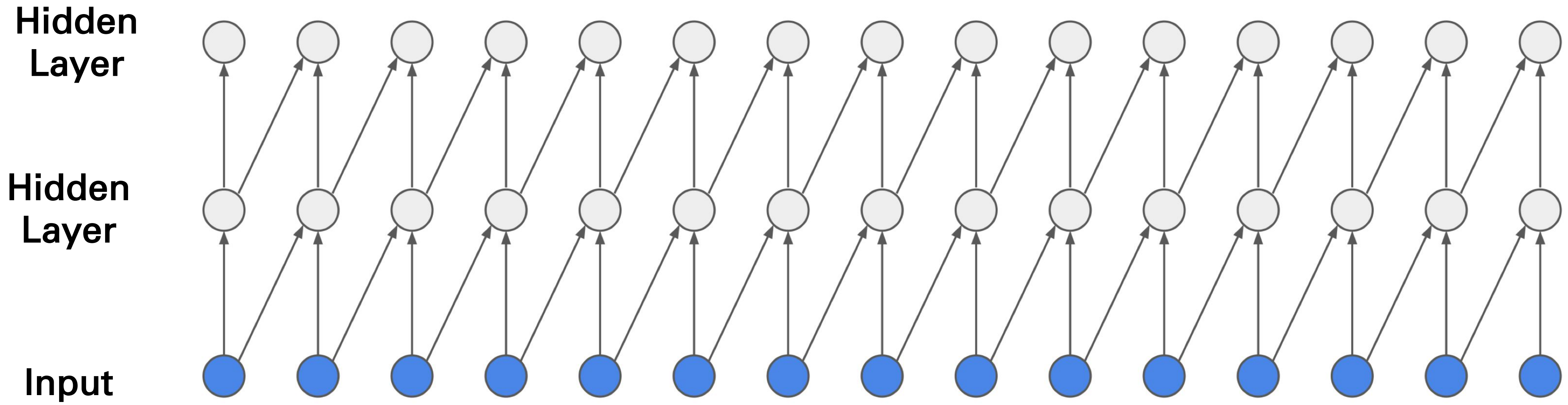
Bytenet decoder

- Stack of dilated, masked 1-D convolutions in the decoder
- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past

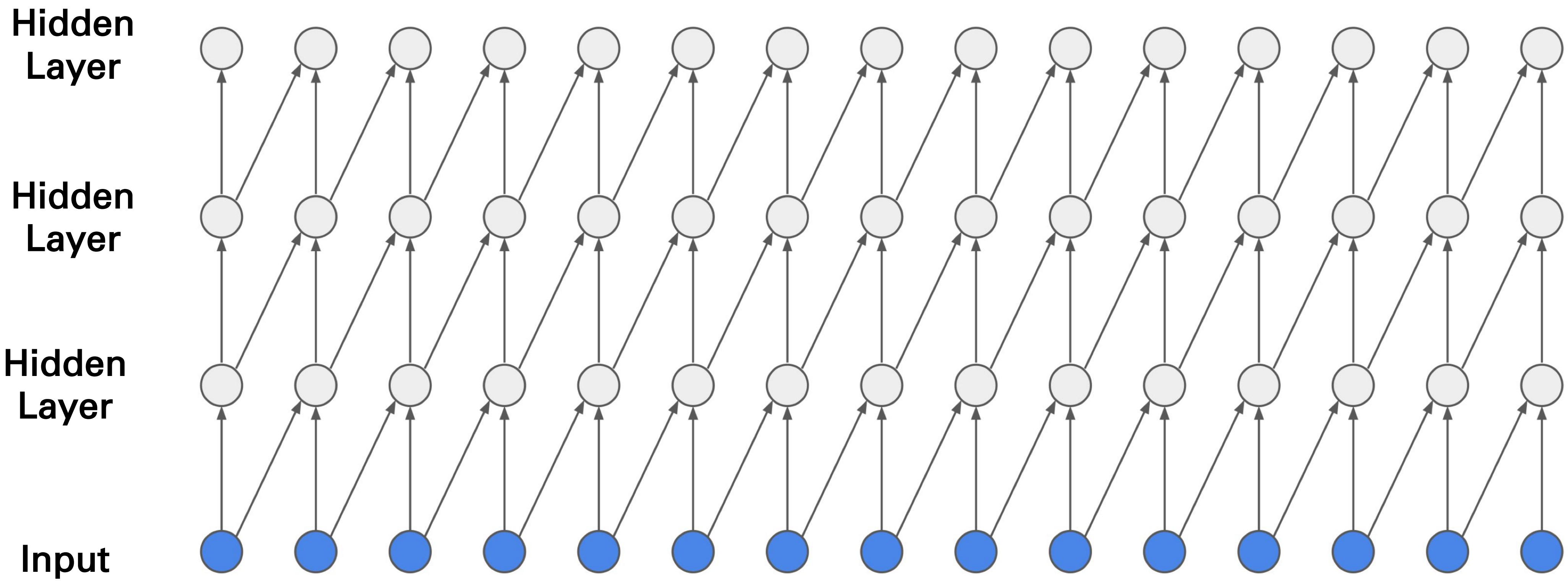
Causal Convolution



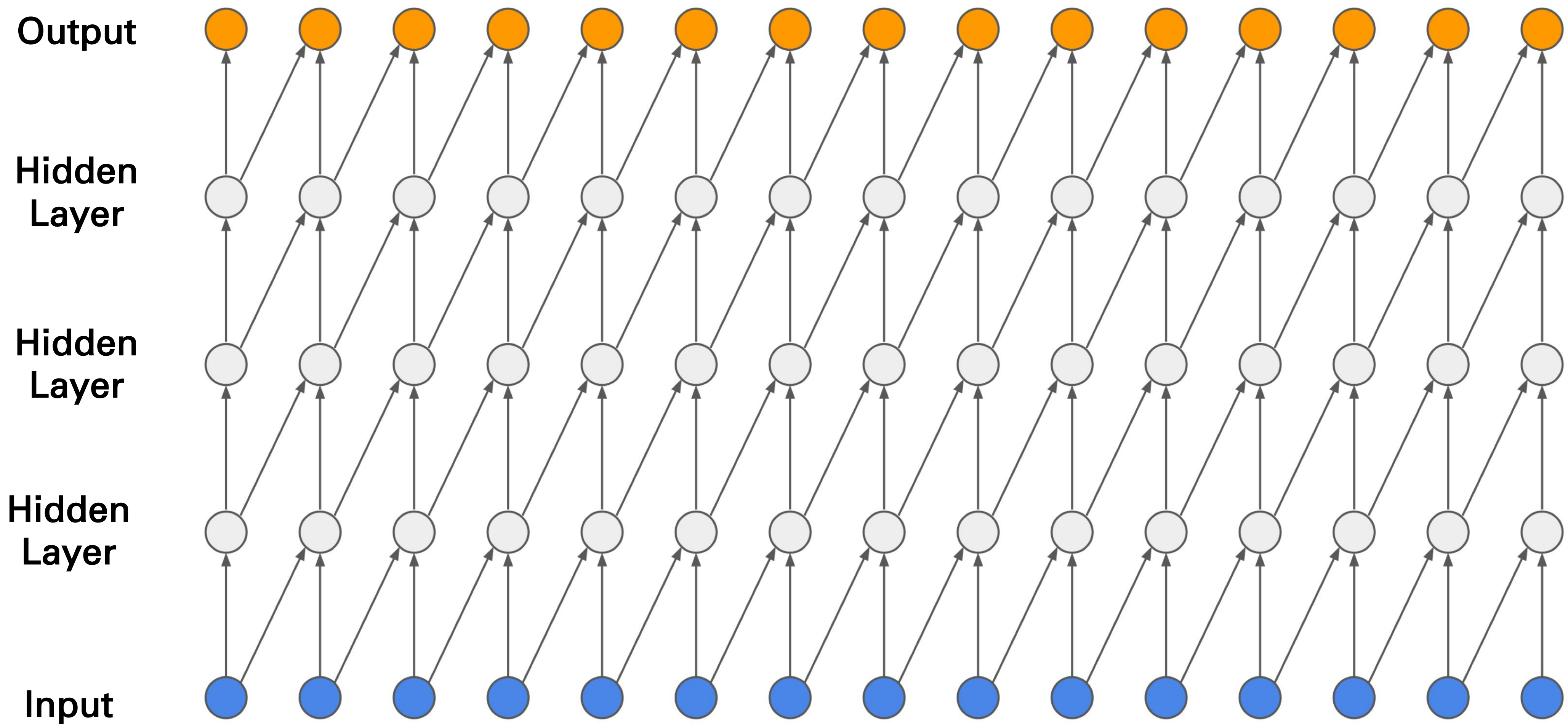
Causal Convolution



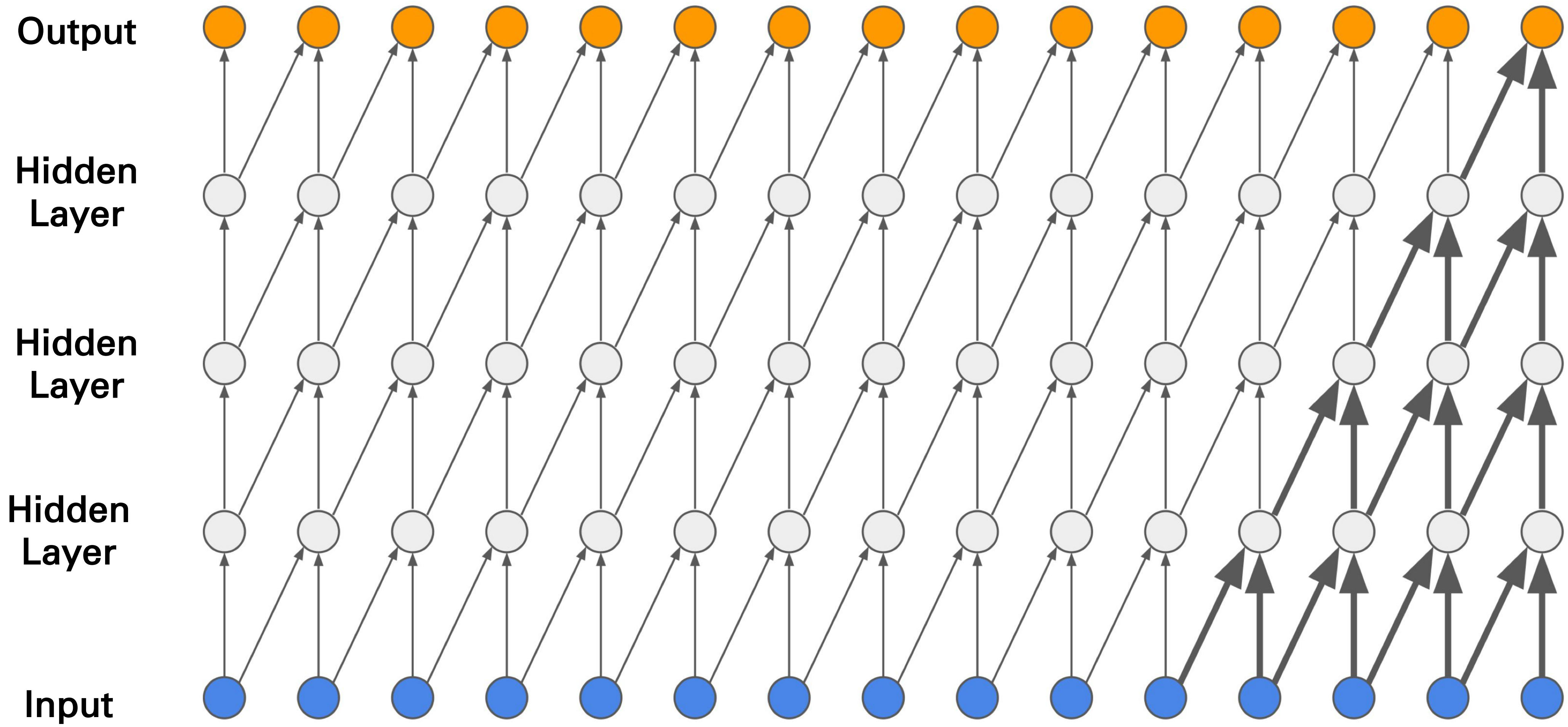
Causal Convolution



Causal Convolution



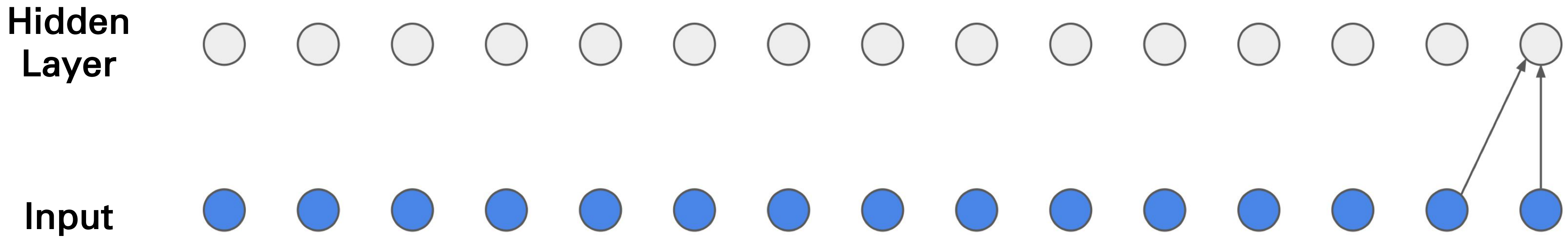
Causal Convolution



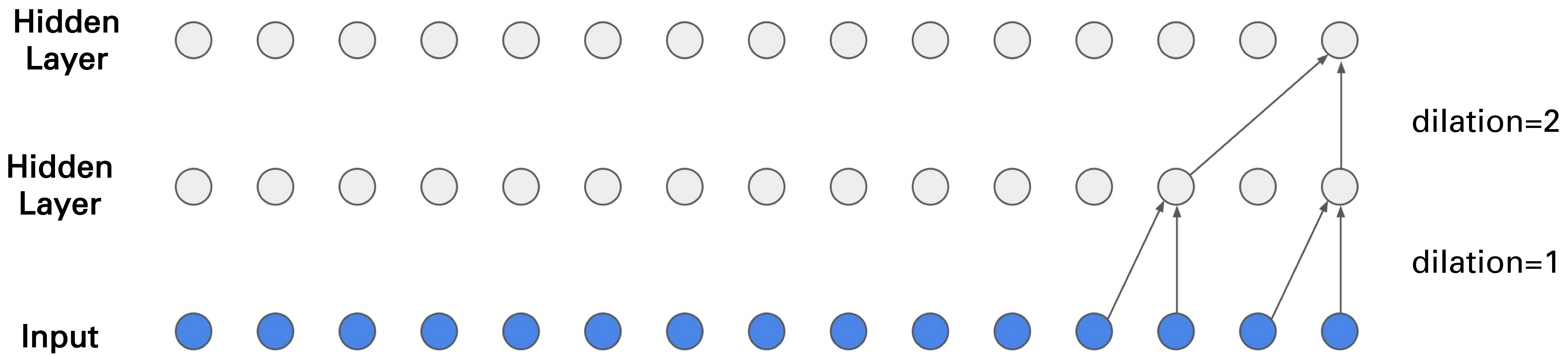
Causal Dilated Convolution

Input

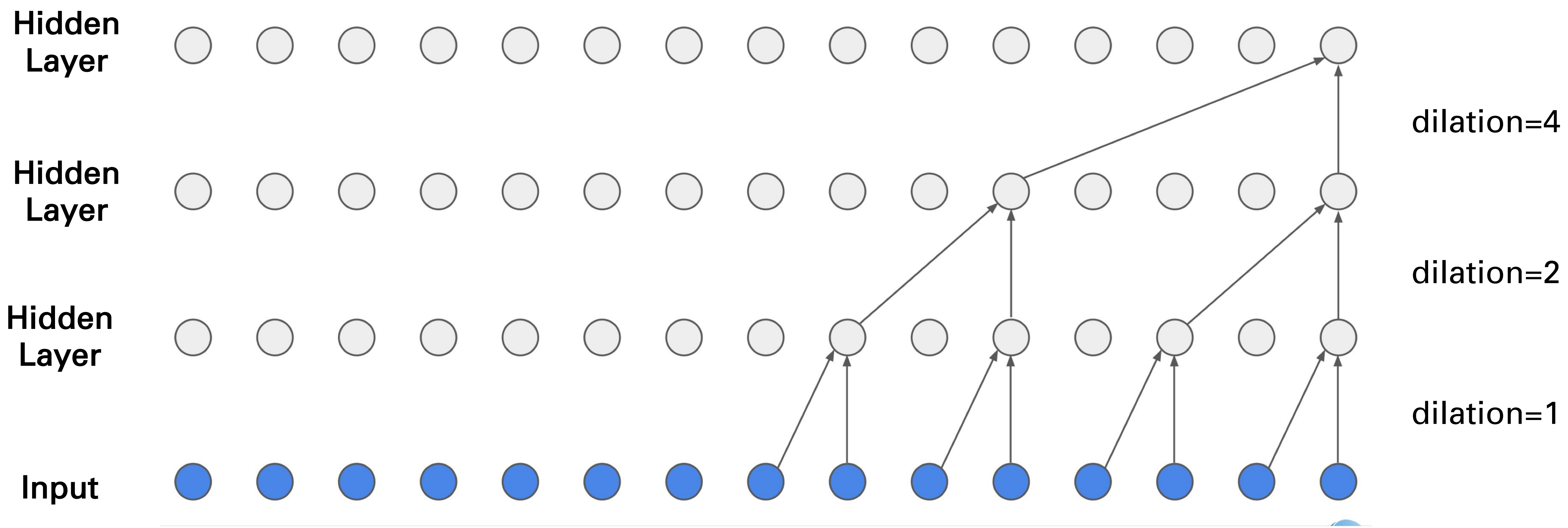
Causal Dilated Convolution



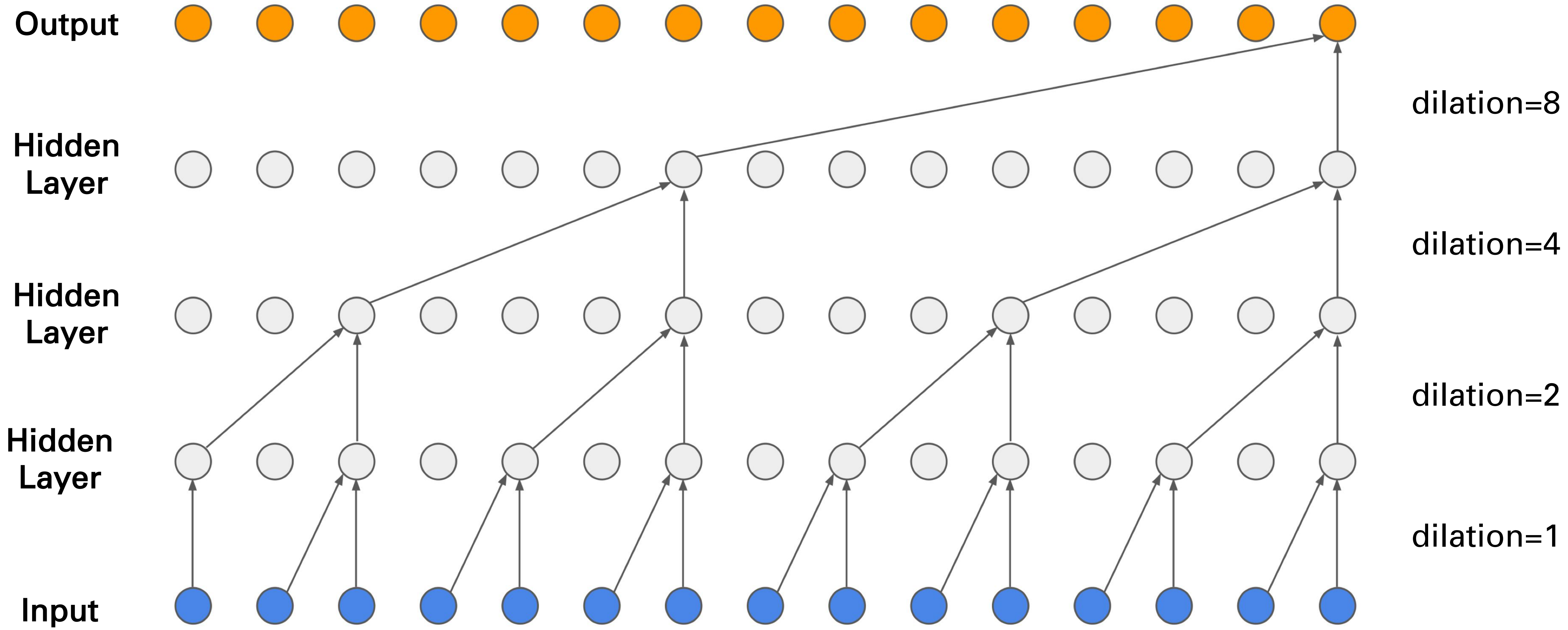
Causal Dilated Convolution



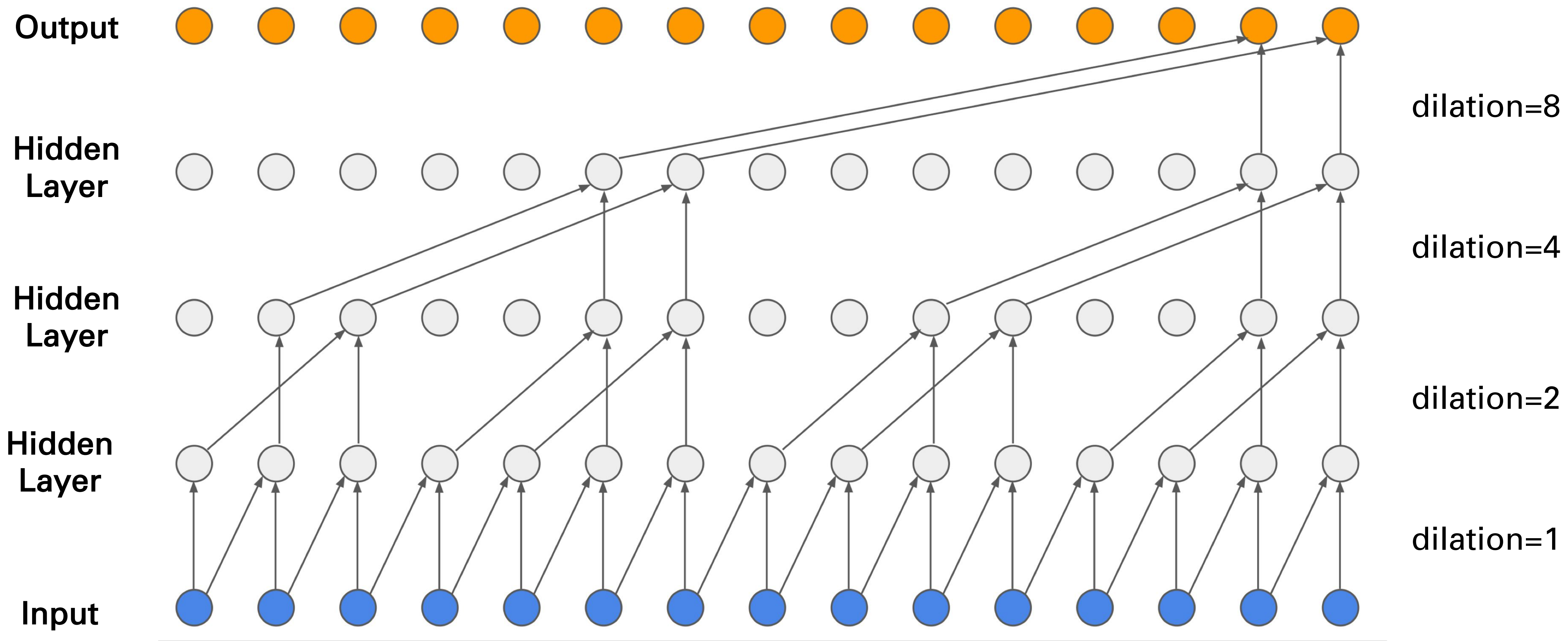
Causal Dilated Convolution



Causal Dilated Convolution

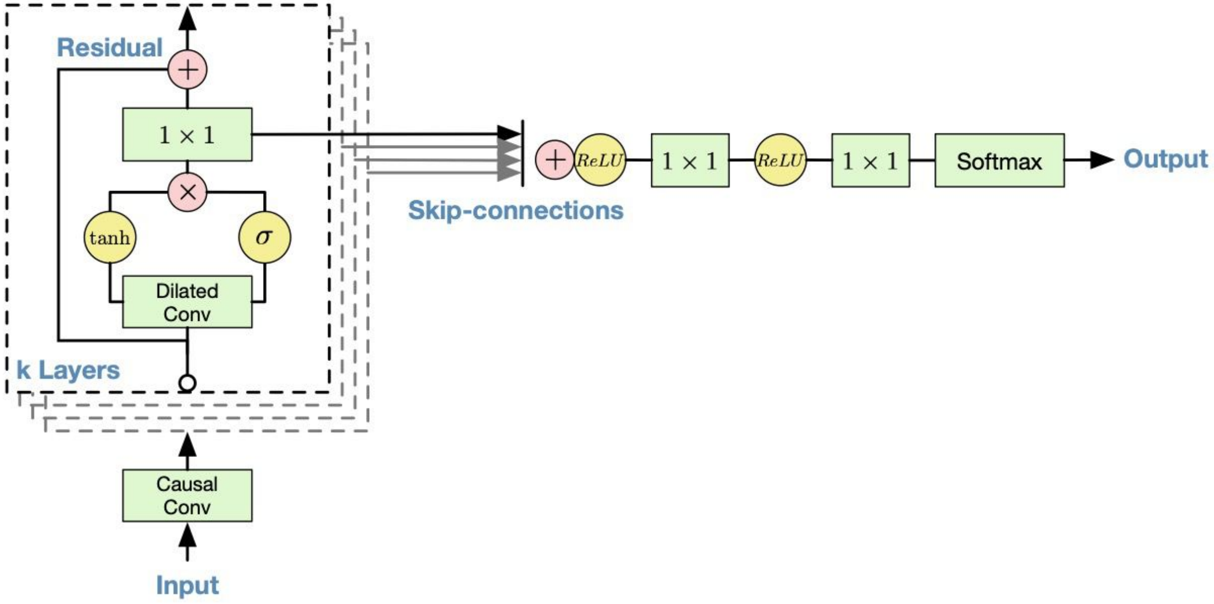


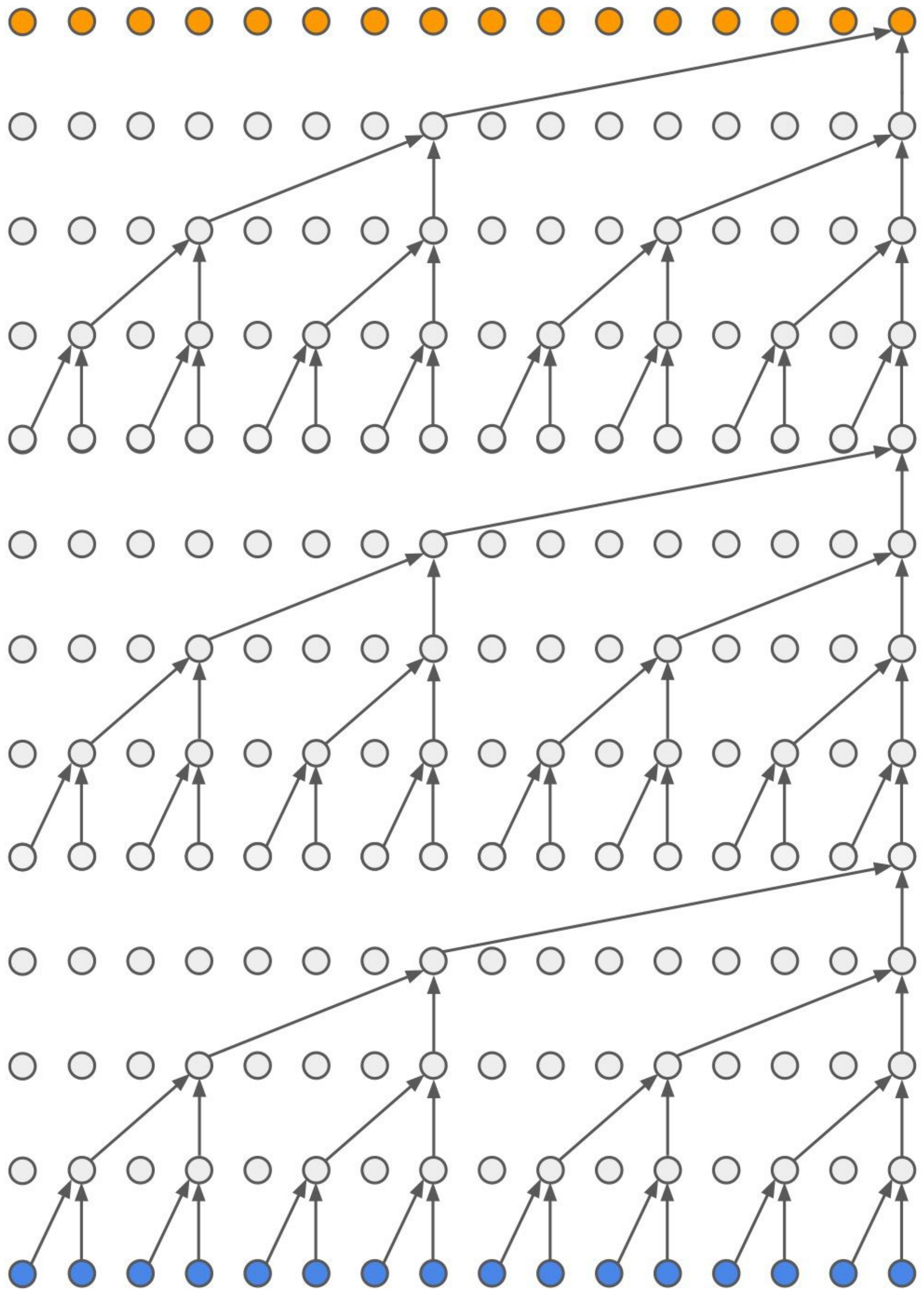
Causal Dilated Convolution



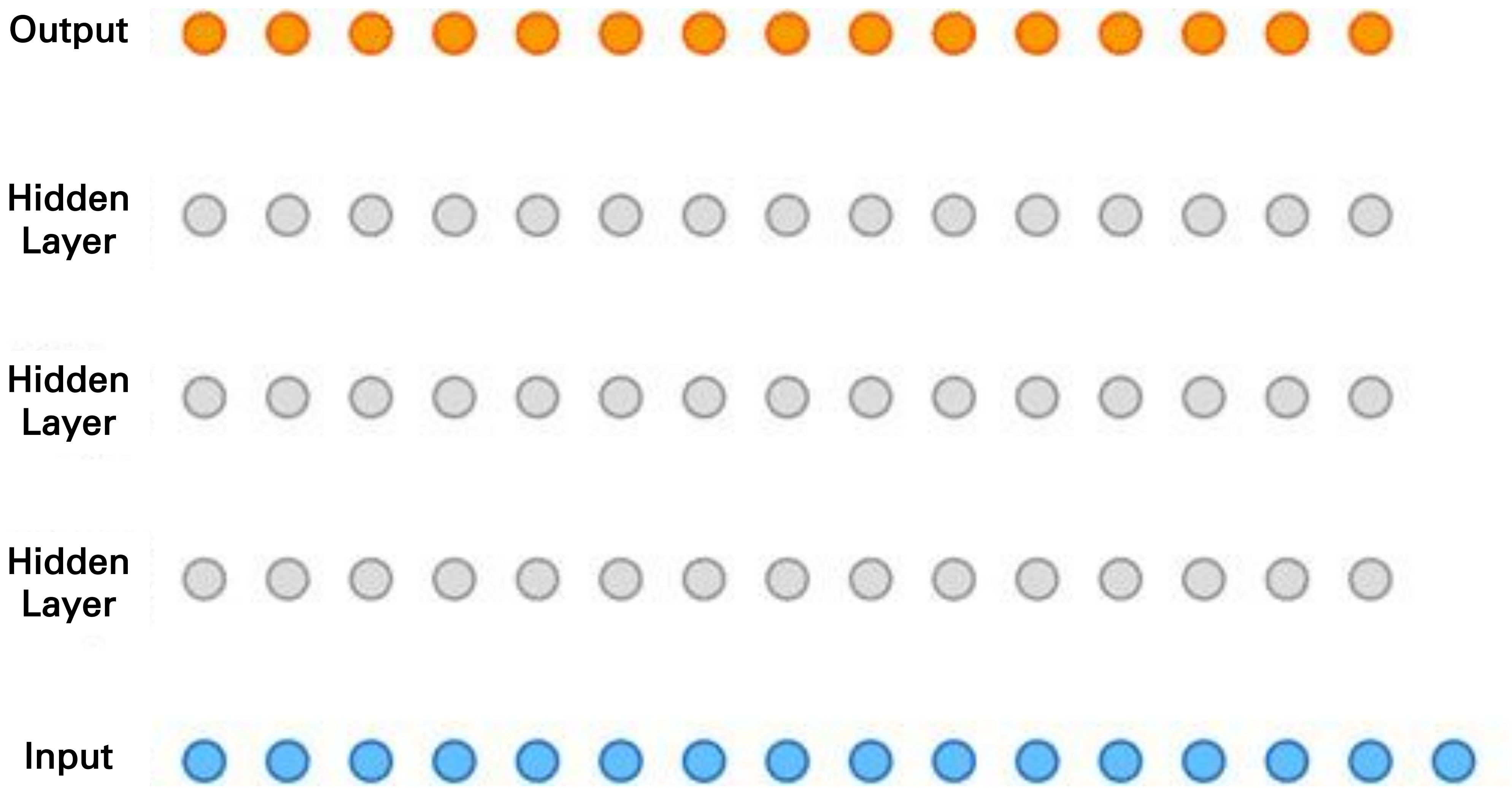
Multiple Stacks

- Improved receptive field with dilated convolutions
- Gated Residual block with skip connections





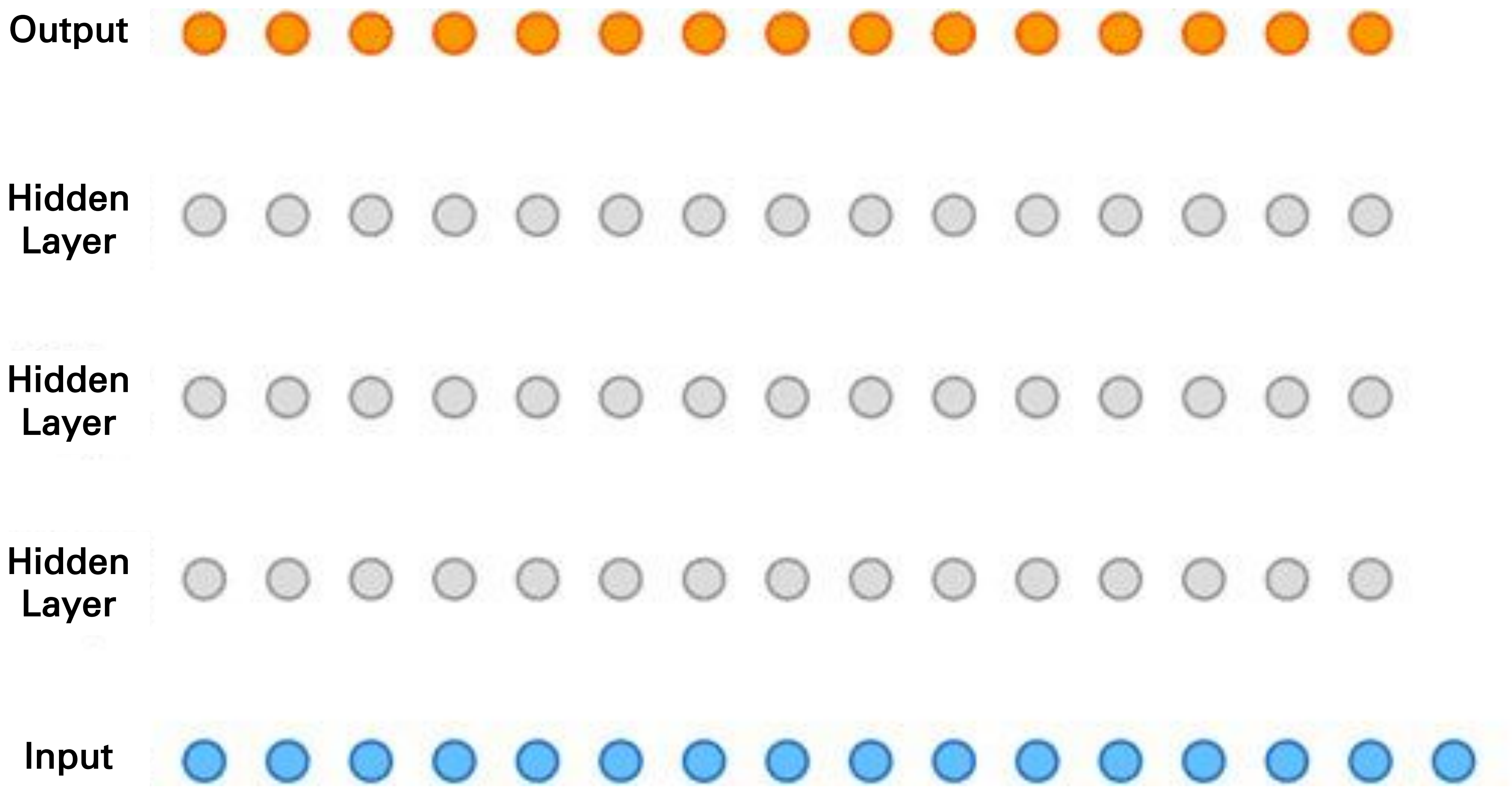
Sampling



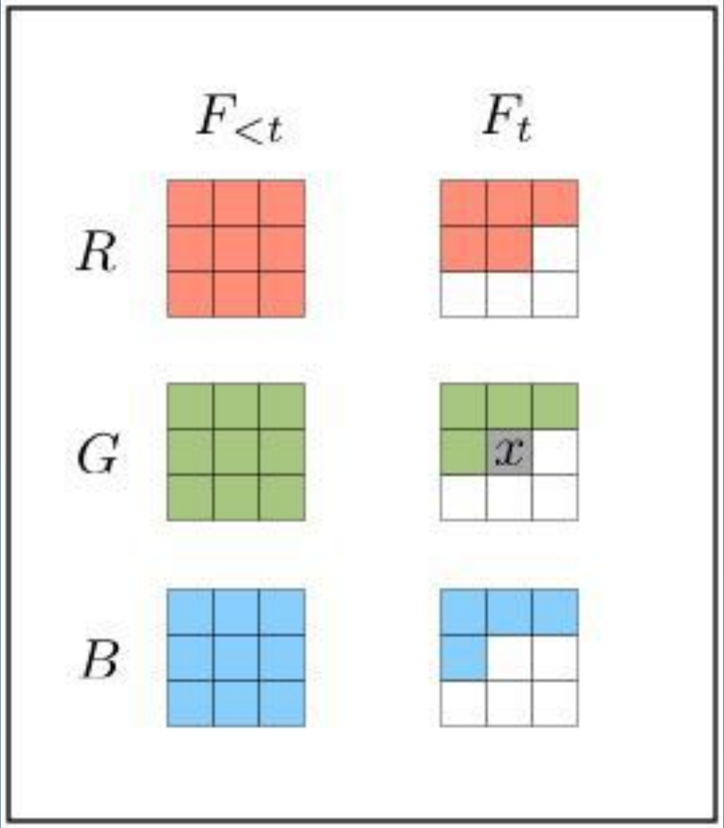
Sampling

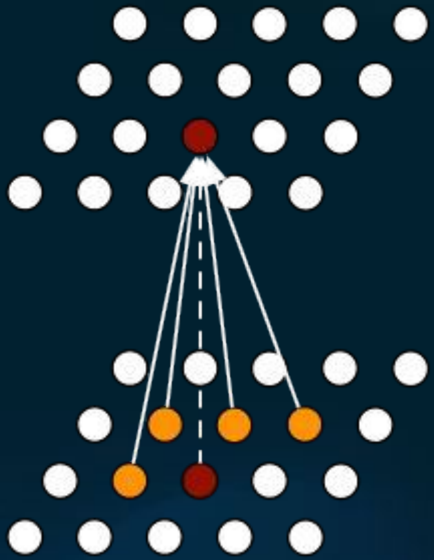
sample speech

sample music



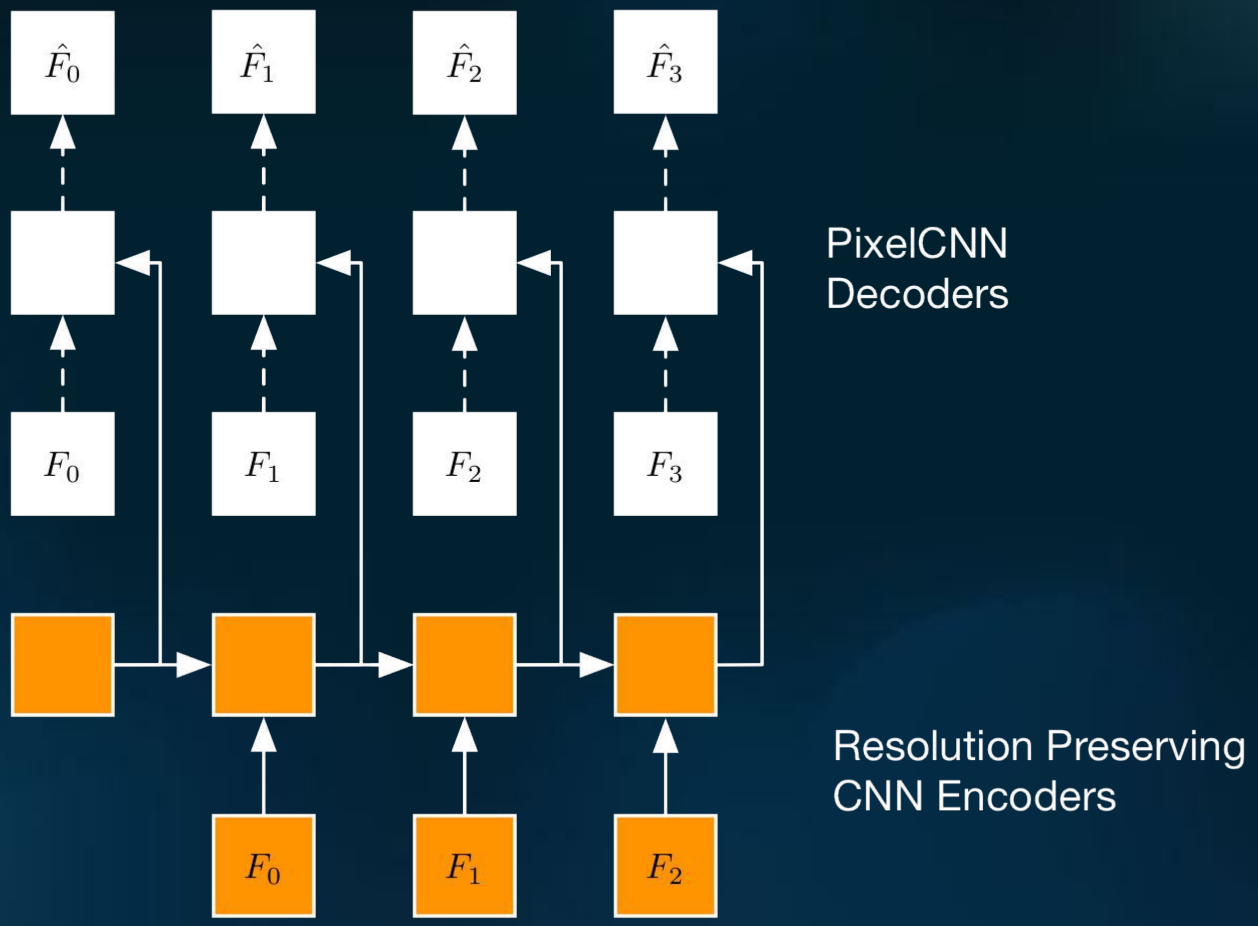
Video Pixel Net





Masked convolution

Video Pixel Net



VPN Samples for Moving MNIST

No frame dependencies

VPN

Videos on [nal.ai/vpn](nal.ai/vpn)

VPN Samples for Robotic Pushing

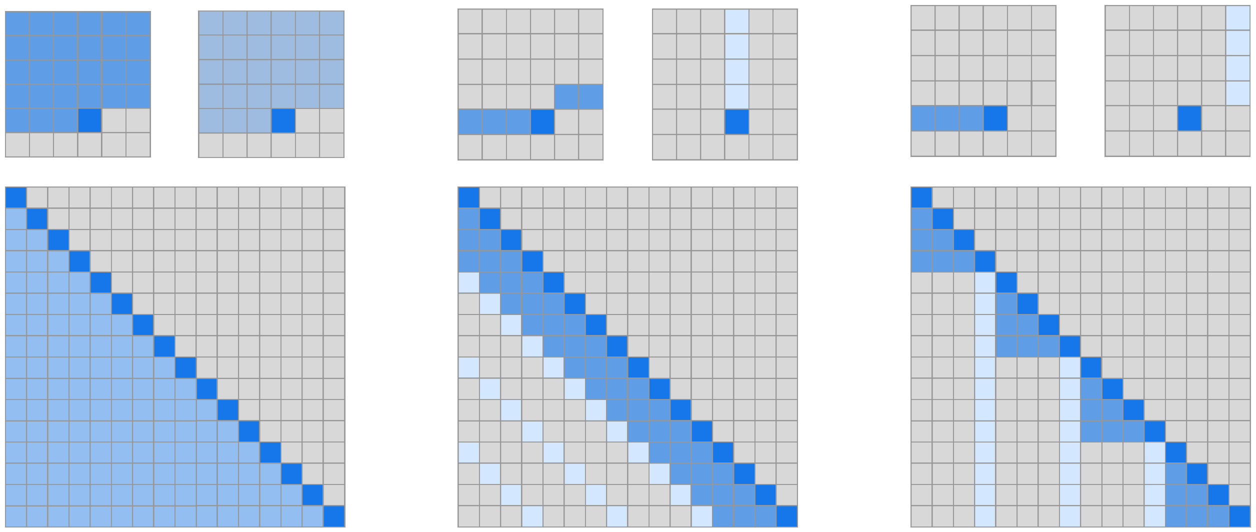
No frame dependencies

VPN

Videos on nal.ai/vpn

VPN Samples for Robotic Pushing

Sparse Transformers



Normal Transformer

Sparse Transformer (strided)

Sparse Transformer (fixed)

- Strided attention is roughly equivalent to each position attending to its row and its column
- Fixed attention attends to a fixed column and the elements after the latest column element (especially used for text).

[Child, Gray, Radford, Sutskever, 2019]

Next Lecture:
Generative Adversarial Networks and Flow Models