Trends and Frontiers in Deep Generative Models

Scott Reed

30 Jan 2018

Overview

Part I: Background

- Deep Autoregressive Networks
  - Generating sequential data (WaveNet)
  - Generating spatially-structured data (PixelCNN, ScanNet)
- Generative Adversarial Networks
  - Generating high-resolution images (Progressive GAN)

Part II: Frontiers

- Learning from limited data
- Predicting far into the future
- Generative models for Agents

Part I: Background

Autoregressive Models

Generative models - Research Landscape

- Latent variable models (VAE, DRAW)
- Implicit (GAN, GMMN, Progressive GAN)
- Transform (NICE, IAF, Real NVP)
- **Autoregressive** (NADE, MADE, RIDE, PixelCNN, WaveNet)

UAI 2017 Tutorial on Deep Generative Models.

NIPS 2016 Tutorial on Generative Adversarial Networks

Slide credit: Aaron van den Oord

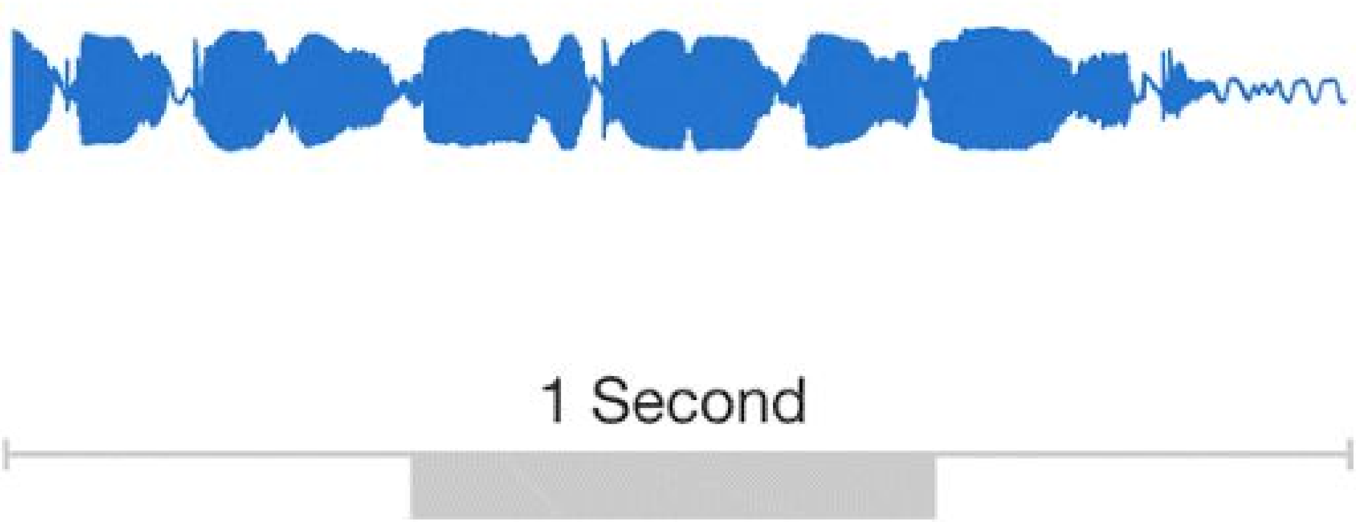
Autoregressive Models

$$P(x;\theta) = \prod_{n=1}^{N} P(x_n | x_{<n}; \theta)$$

- Each factor can be parametrized by θ, which can be shared.

- The variables can be arbitrarily ordered and grouped, as long as the ordering and grouping is consistent.

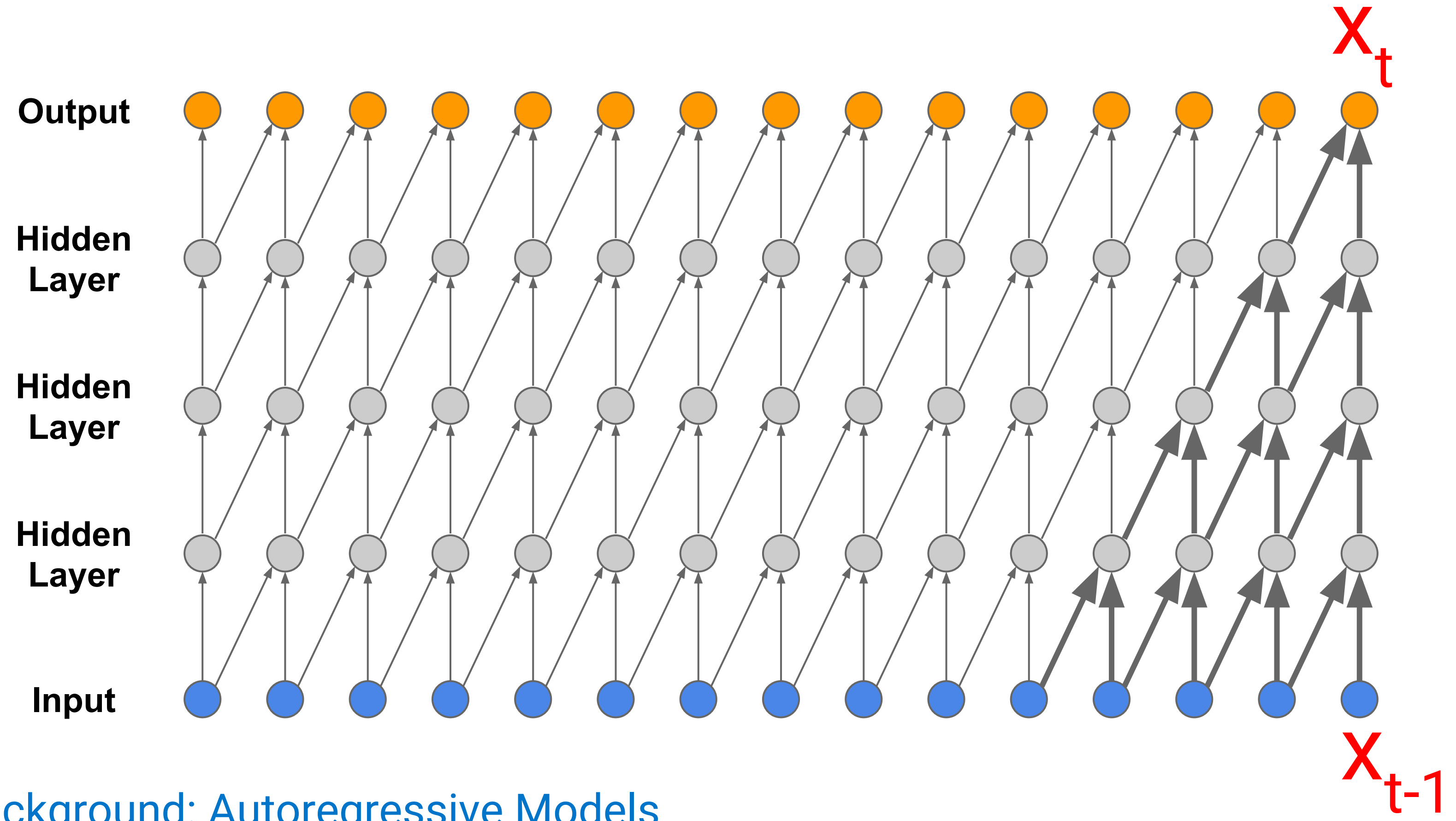
Modeling Audio



Background: Autoregressive Models

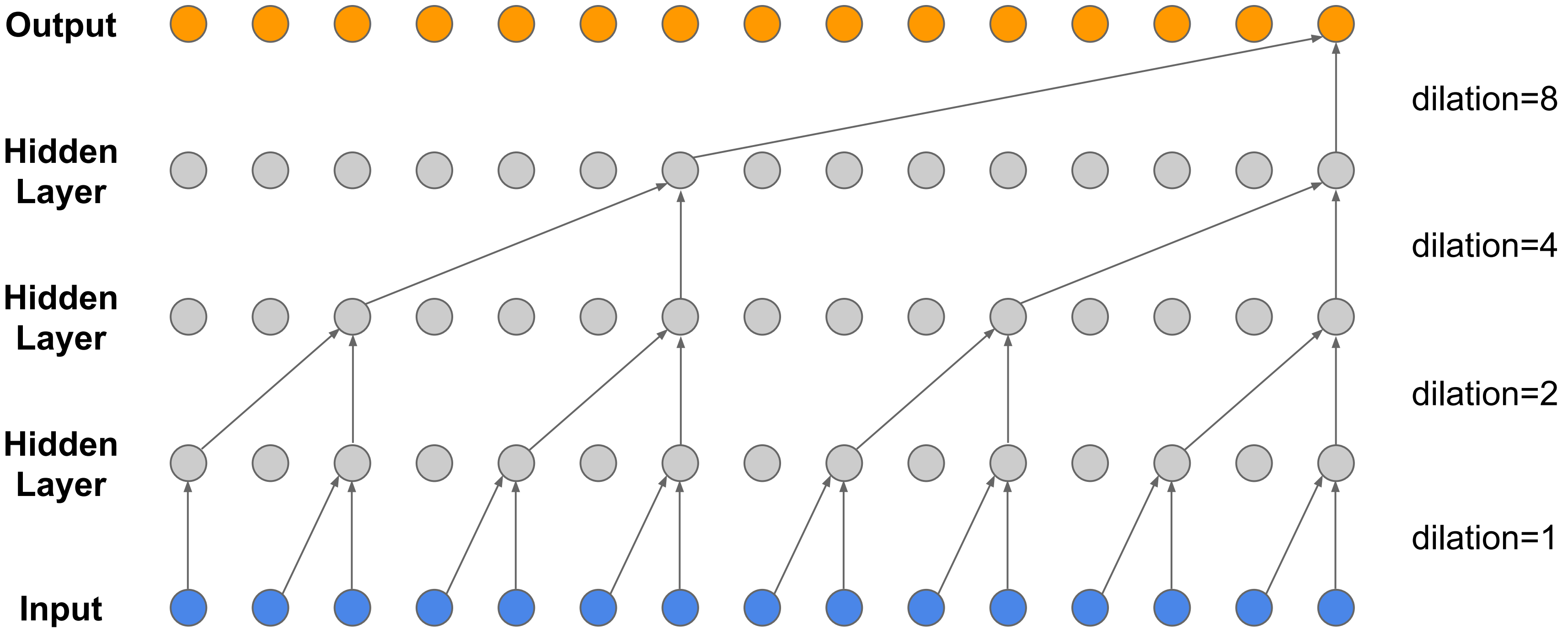
Slide credit: Aaron van den Oord

Causal Convolution



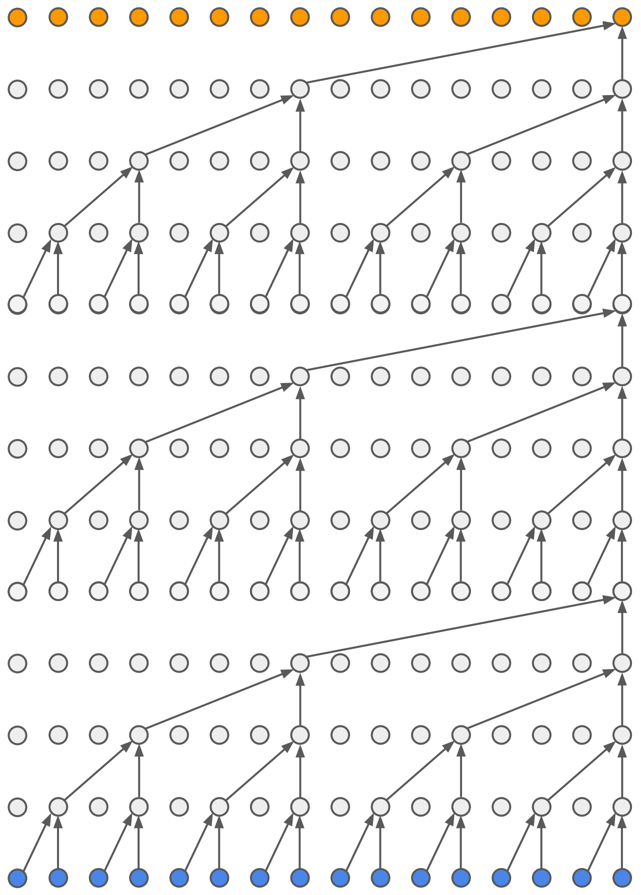
Slide credit: Aaron van den Oord

Causal Dilated Convolution



Slide credit: Aaron van den Oord

Multiple Stacks



Background: Autoregressive Models

Slide credit: Aaron van den Oord

Cross entropy loss

Given preceding observations $x_{<t}$, the network computes logits y. We can compute the softmax over possible quantized values for sampling:

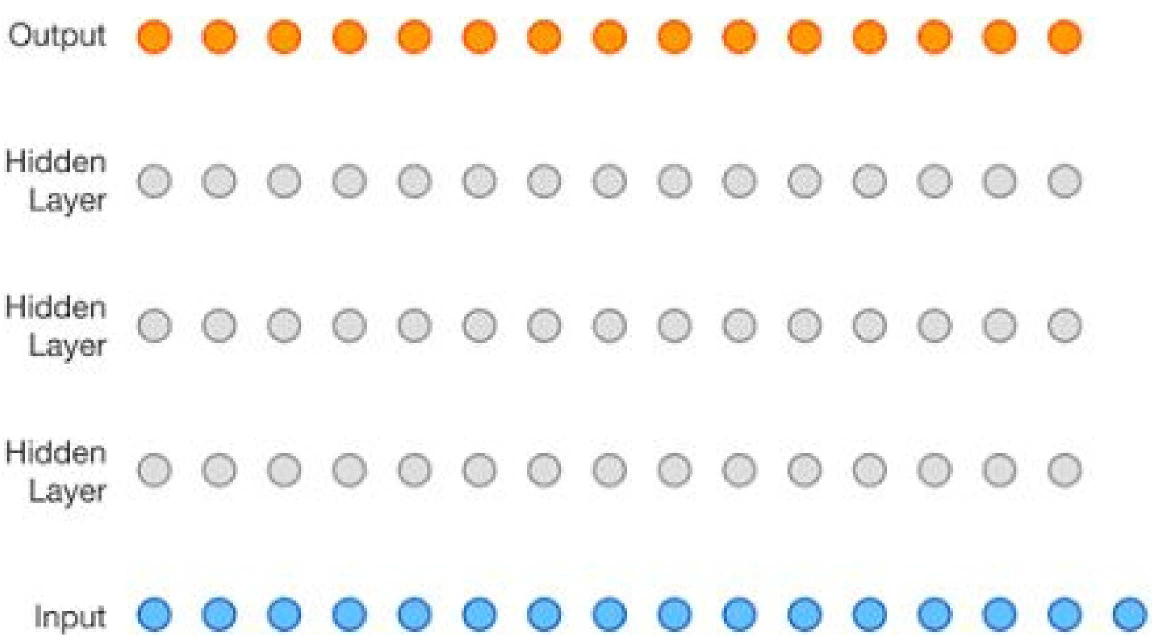
$$P(x_t = n | x_{<t}; \theta) = e^{y_n} / \Sigma_{n'=0}^{255} e^{y_{n'}}$$

The objective is to minimize the negative log-likelihood:

$$\mathcal{L}(x; \theta) = -\log P(x_t | x_{<t}; \theta)$$

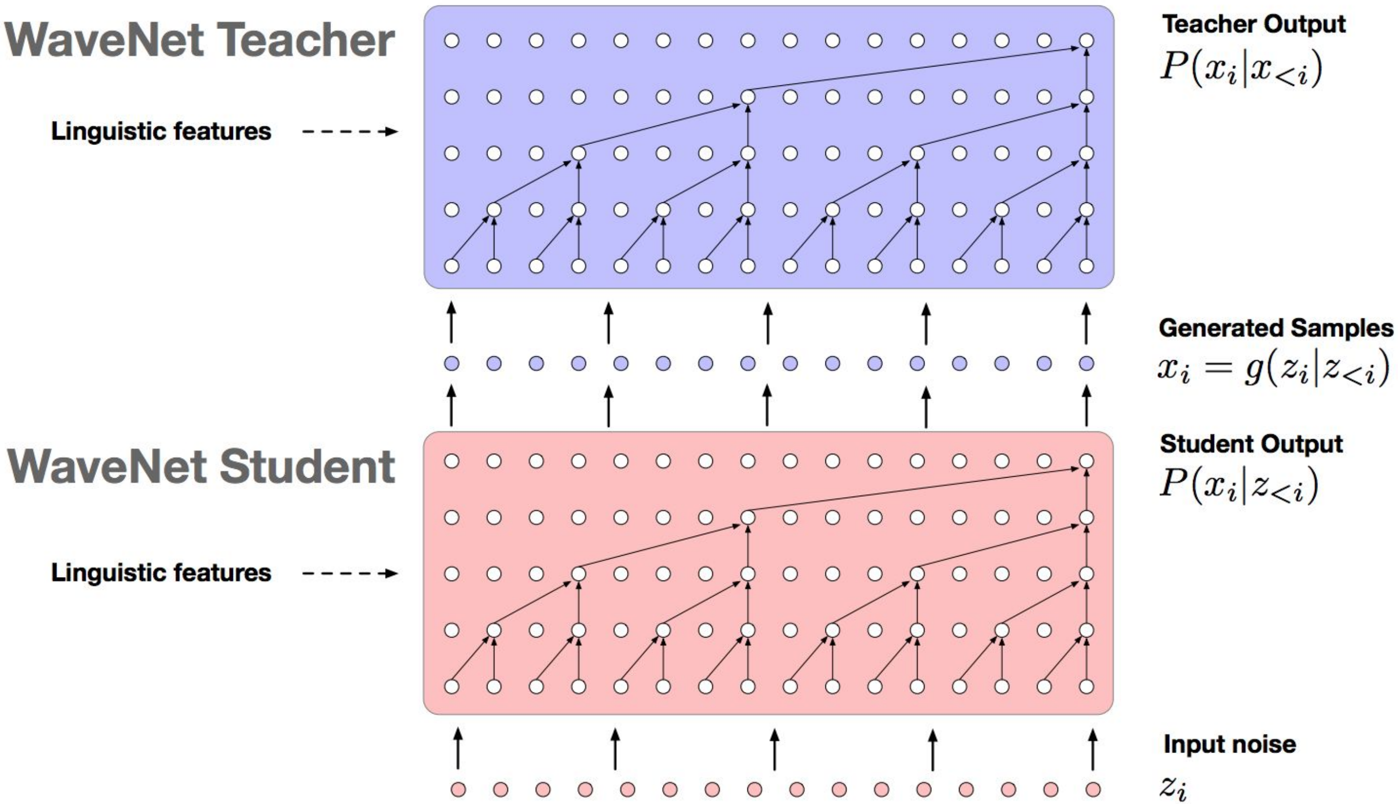
Convenient function in TF: `tf.nn.softmax_cross_entropy_with_logits`.

Sampling - Sequential, O(N) for N samples



Slide credit: Aaron van den Oord

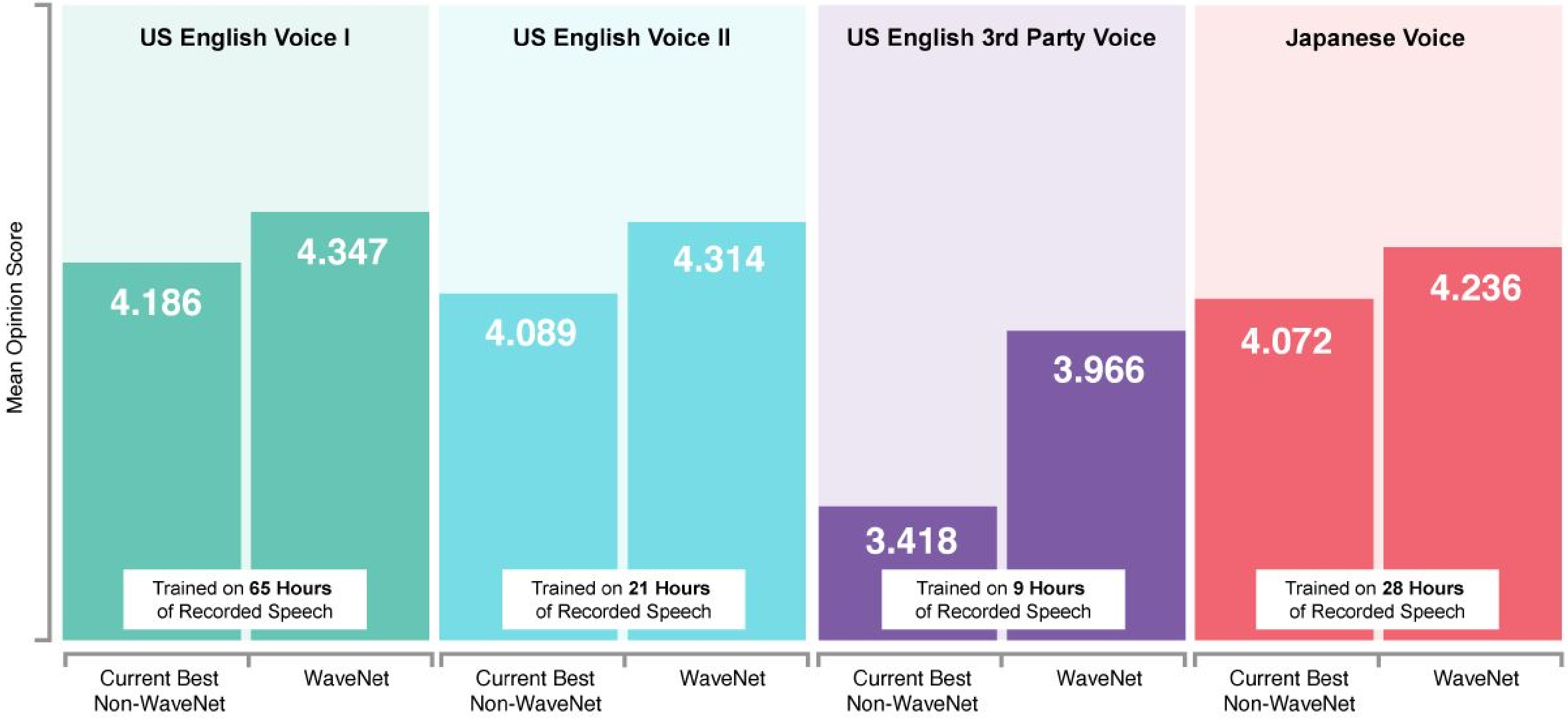
Distillation: from O(N) to O(1) sampling



1. Oord, Aaron van den, et al. "Parallel WaveNet: Fast High-Fidelity Speech Synthesis."

Distillation: from O(N) to O(1) sampling

Mean Opinion Scores



Modeling Text

The cat sat on the mat (word-level)

Shorter sequences and dependencies, semantically meaningful units, many UNK

T h e _ c a t _ s a t _ o n _ t h e _ m a t (character-level)

Long sequences and dependencies, semantically not meaningful units, no UNK

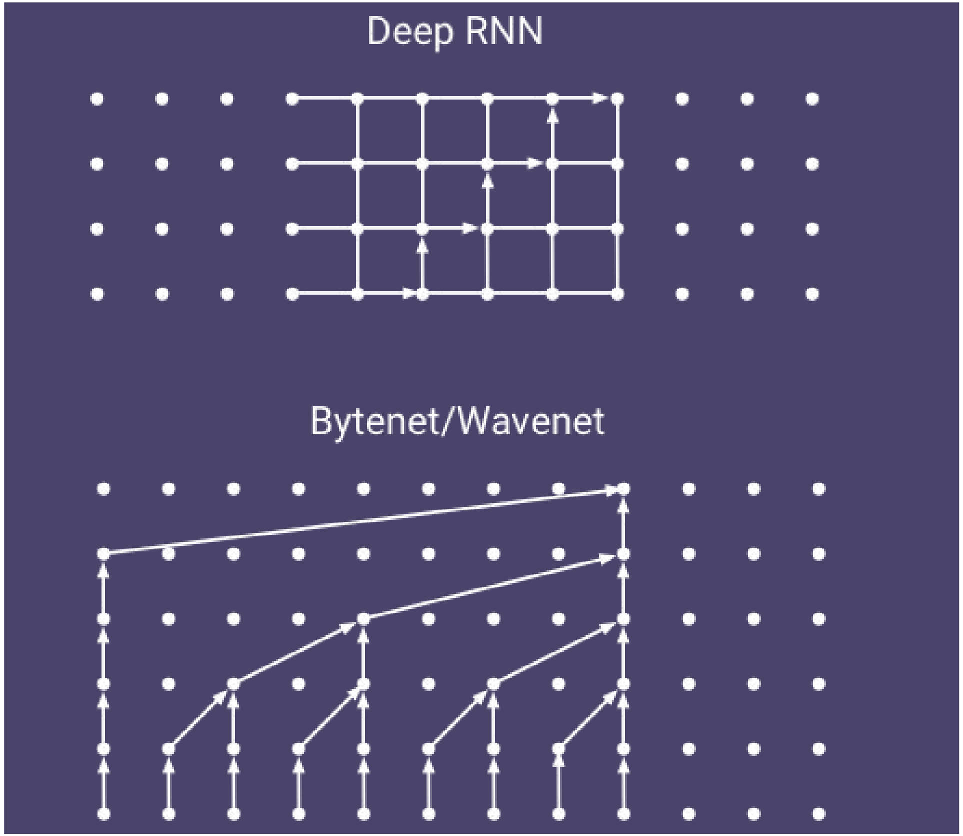
The _ cat _ s a t _ on _ the _ m a t (mixed)

(byte level)

(bit level)

Slide credit: Nal Kalchbrenner

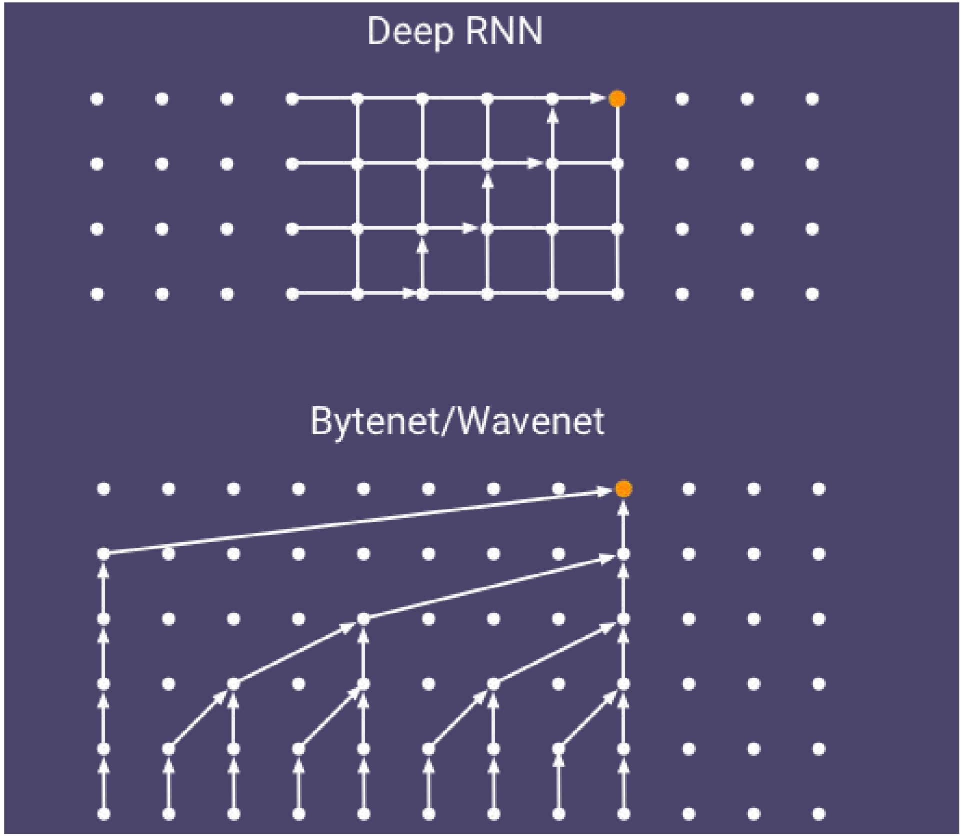
Recurrent versus Causal Convolutional Nets



- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past

Slide credit: Nal Kalchbrenner

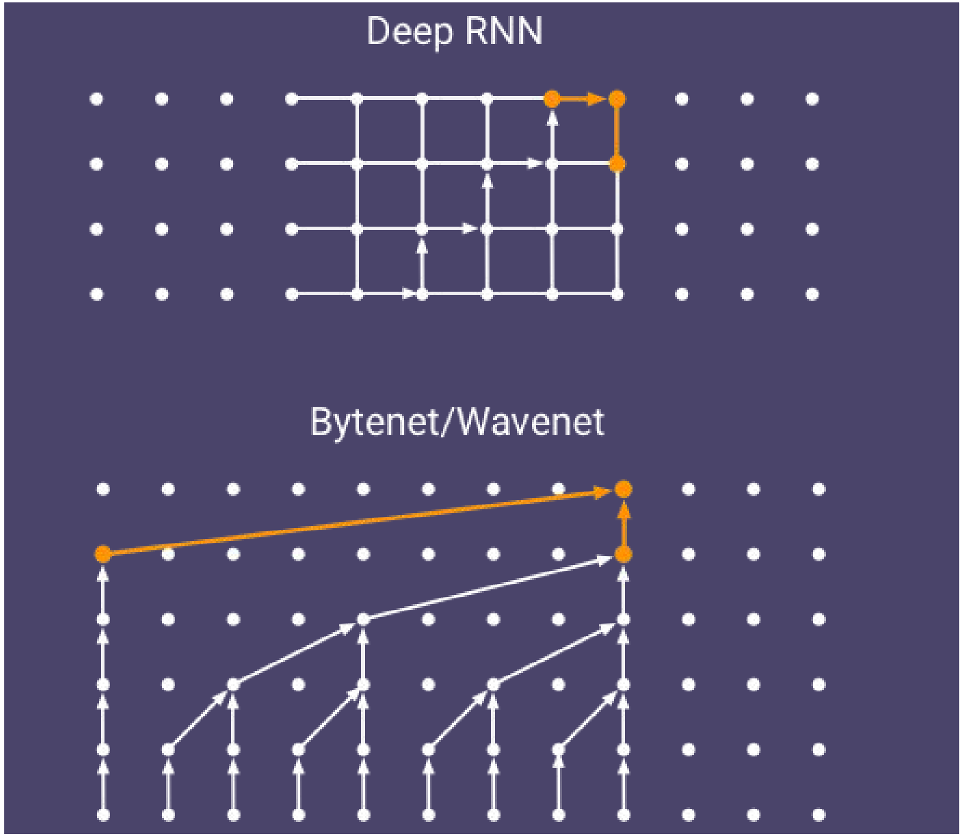
Recurrent versus Causal Convolutional Nets



- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past

Slide credit: Nal Kalchbrenner

Recurrent versus Causal Convolutional Nets

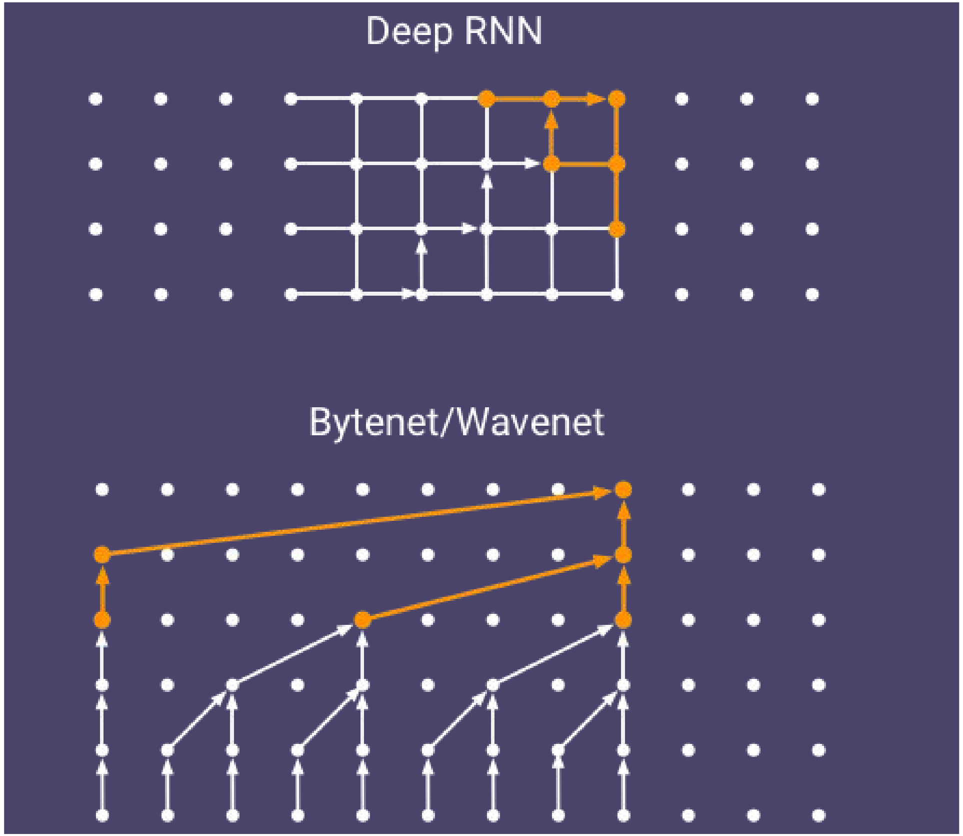


- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past

Background: Autoregressive Models

Slide credit: Nal Kalchbrenner

Recurrent versus Causal Convolutional Nets

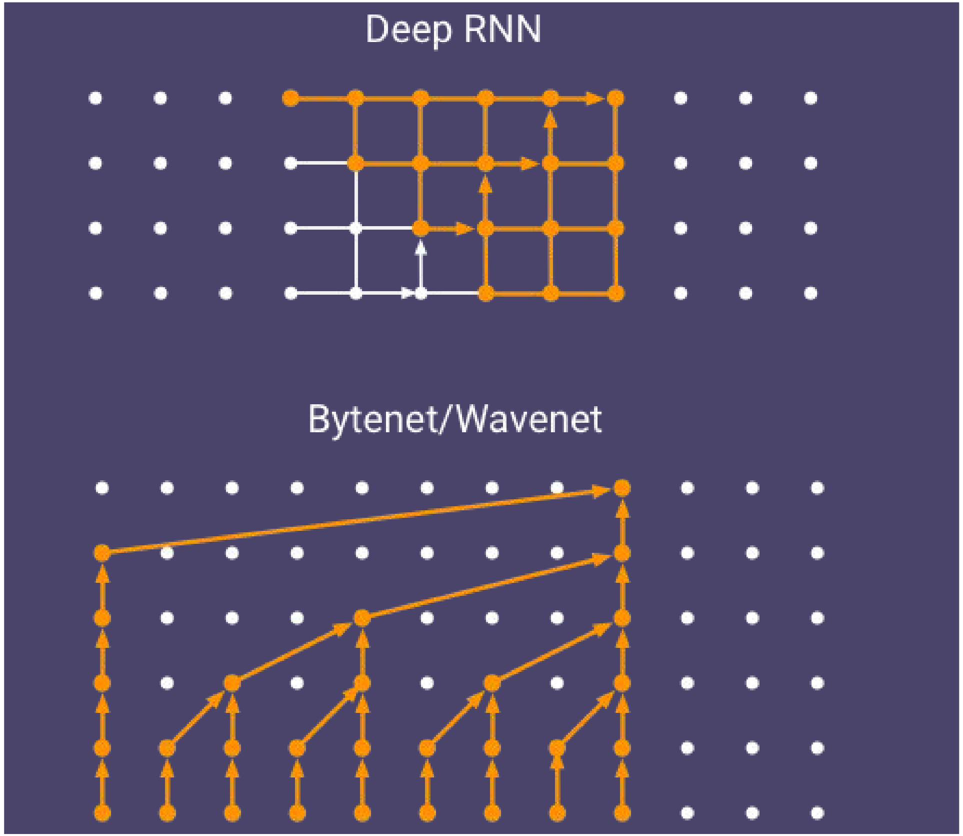


Background: Autoregressive Models

- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past

Slide credit: Nal Kalchbrenner

Recurrent versus Causal Convolutional Nets

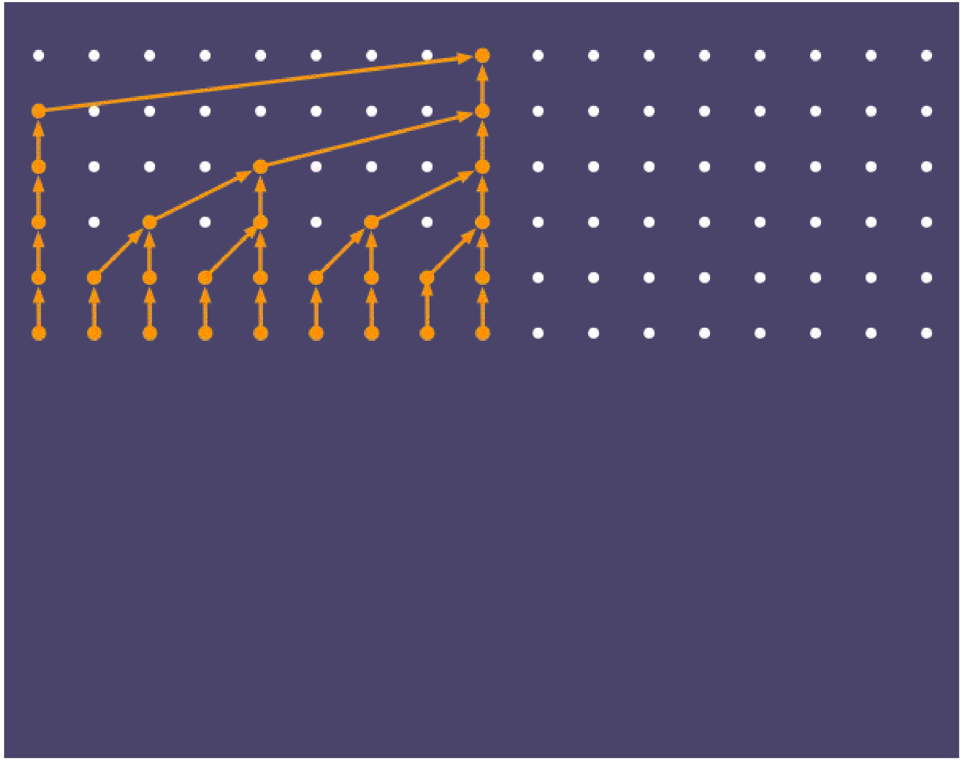


- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past

Background: Autoregressive Models

Slide credit: Nal Kalchbrenner

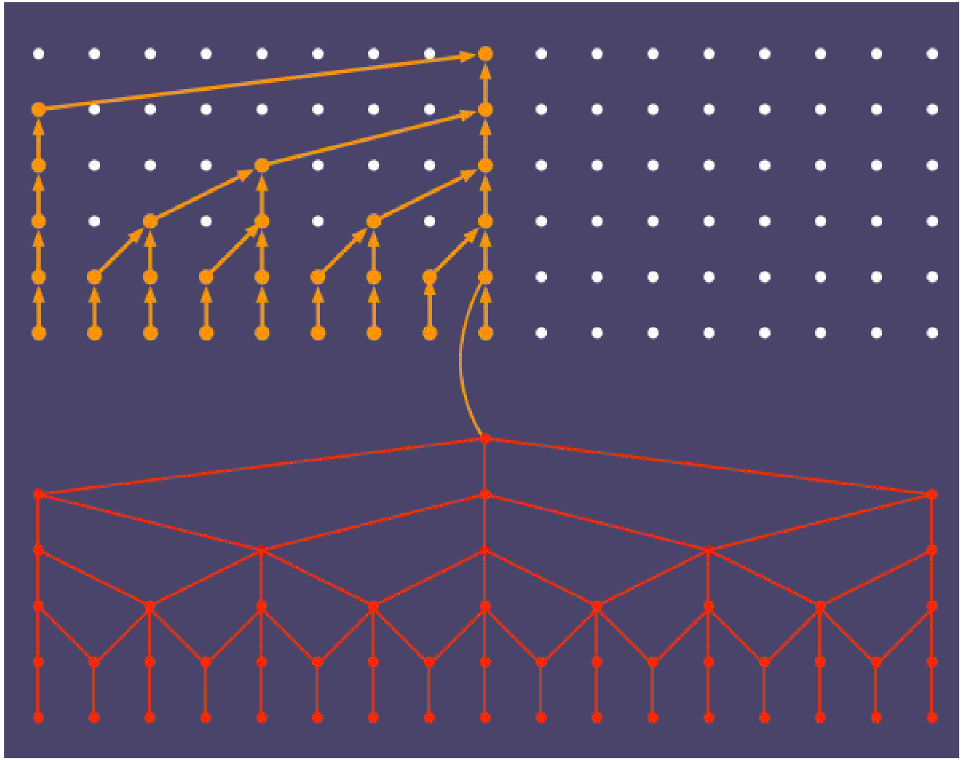
NMT with dilated causal convolutions



Background: Autoregressive Models

Slide credit: Nal Kalchbrenner

NMT with dilated causal convolutions



Stacking preserves resolution compared to seq2seq LSTM

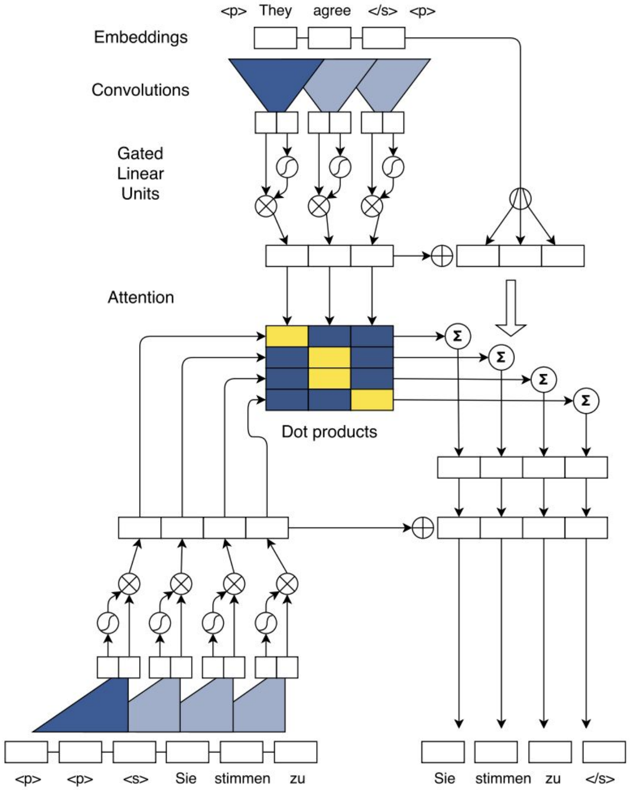
Dynamic unfolding enables variable length outputs

Linear time computation

Background: Autoregressive Models

Slide credit: Nal Kalchbrenner

Convolutional MT models with attention

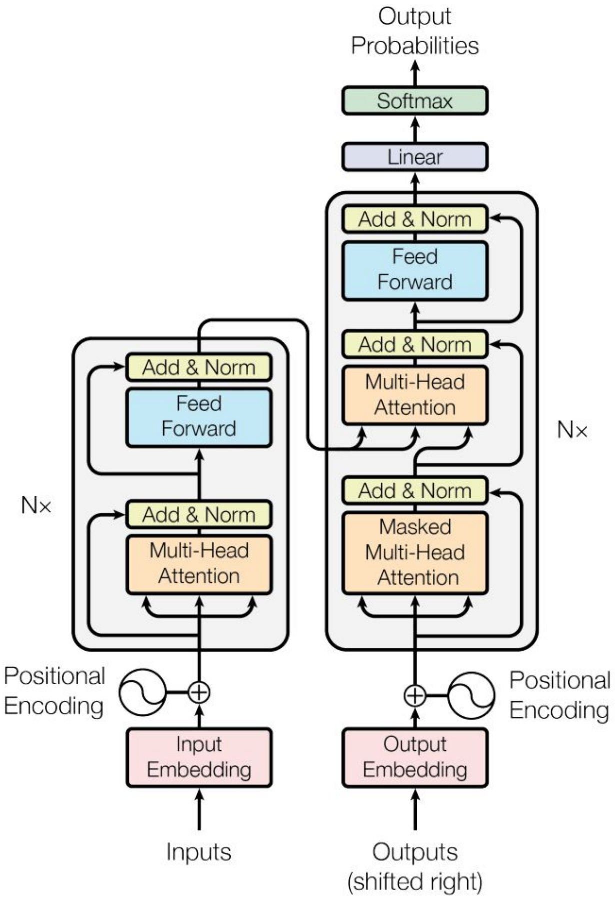


1. Gehring, Jonas, et al. "Convolutional Sequence to Sequence Learning." In *ICML*, 2017.

Attention-only (!) autoregressive models

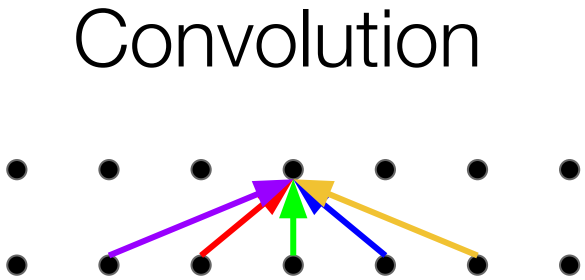
The Transformer

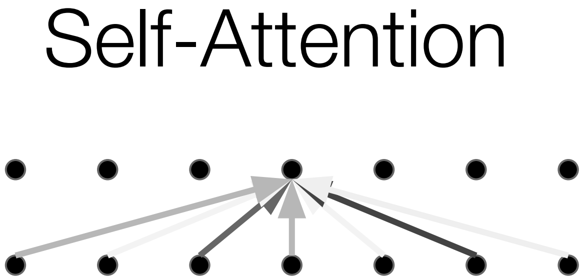
- Not Recurrent
- Not Convolutional
- Dot-product attention over inputs is **masked** to preserve causal structure.



1. Vaswani, Ashish, et al. "Attention is all you need". In *NIPS*, 2017

Self-Attention

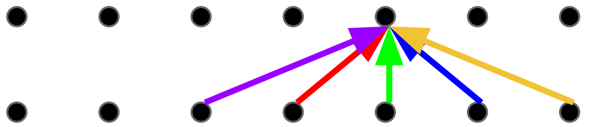




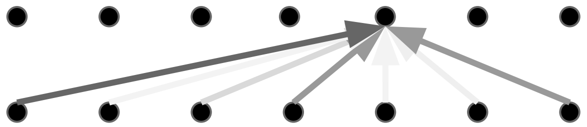
Att is all you need,
Vaswani, et al, 2017

Self-Attention

Convolution

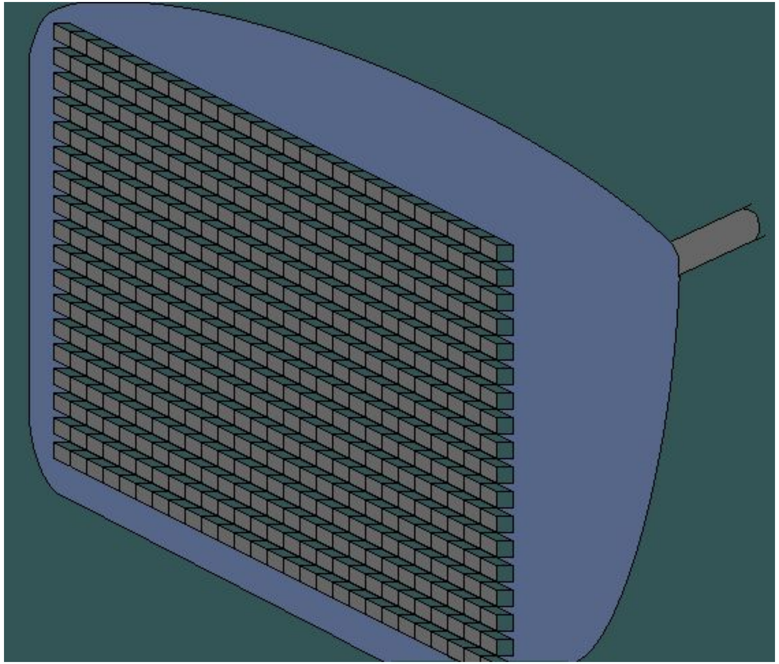


Self-Attention



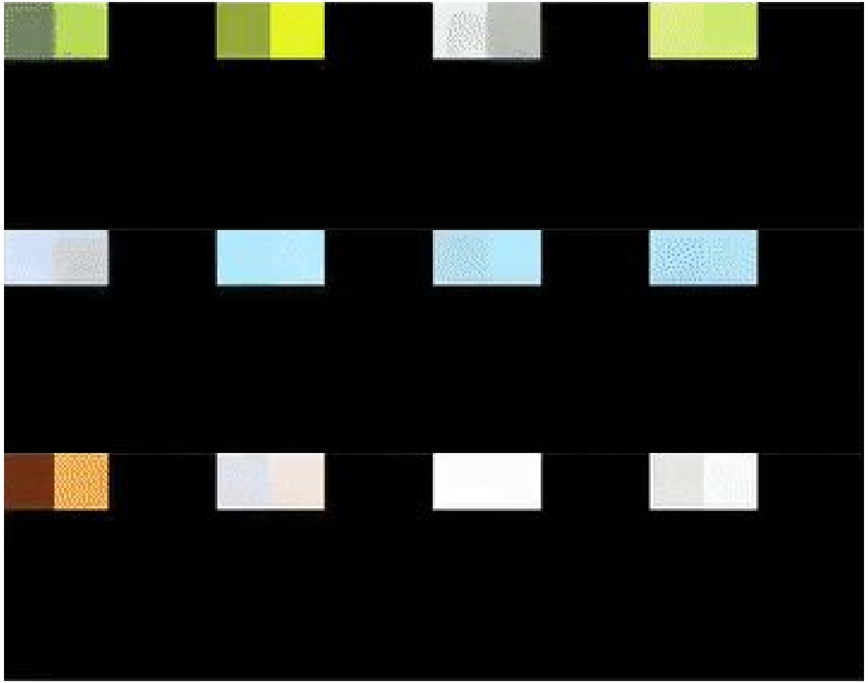
Att is all you need,
Vaswani, et al, 2017

Modeling Images



Pixel-by-pixel

https://giphy.com/gifs/televsion-13epOe3Z06gHba

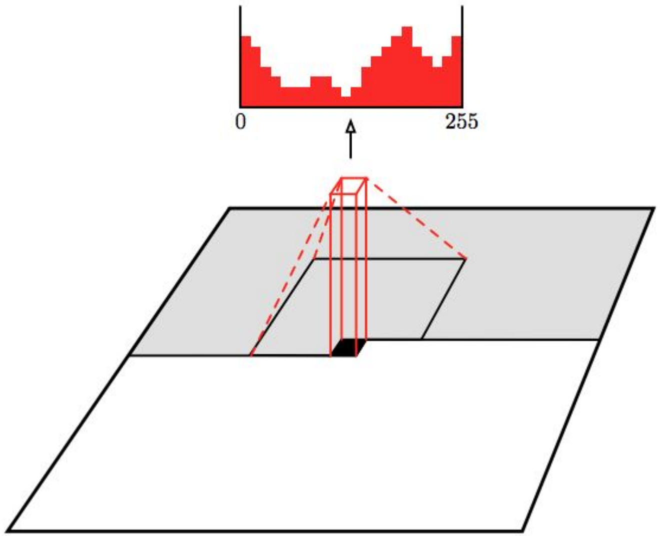


Group-by-group

Reed et al. "Parallel Multiscale Autoregressive Density Estimation."

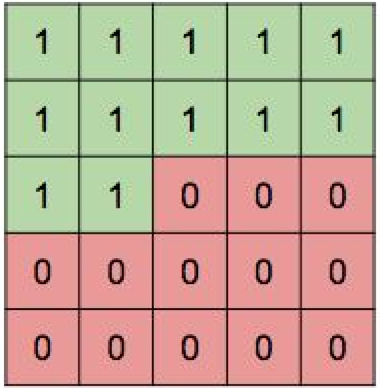
Modeling Images pixel-by-pixel

$$P(x;\theta) = \prod_{n=1}^{N} P(x_n | x_{<n};\theta)$$

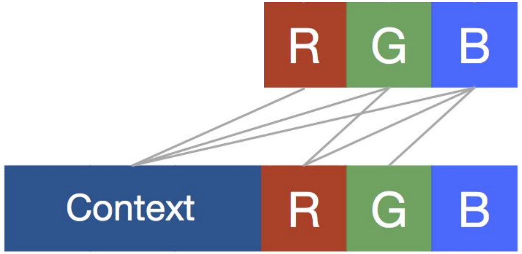


Each factor can be modeled by a shared network (e.g. PixelCNN).

Causal Convolutions

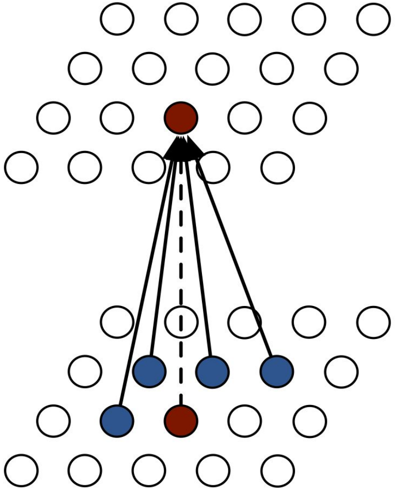


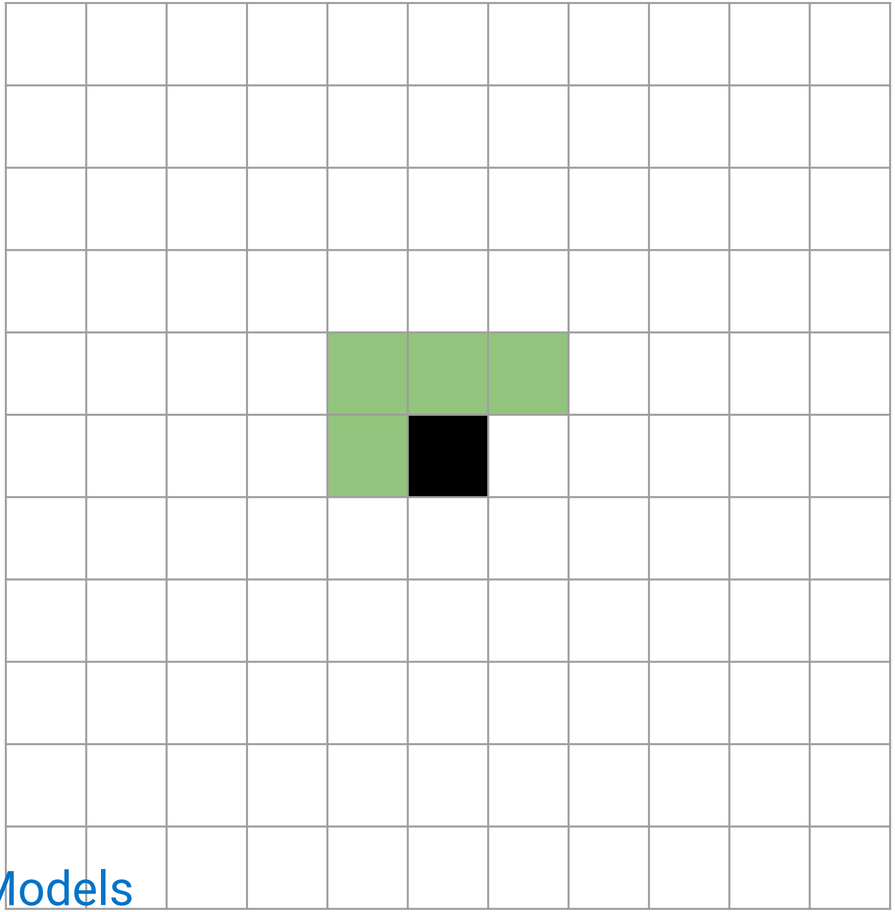
Spatially



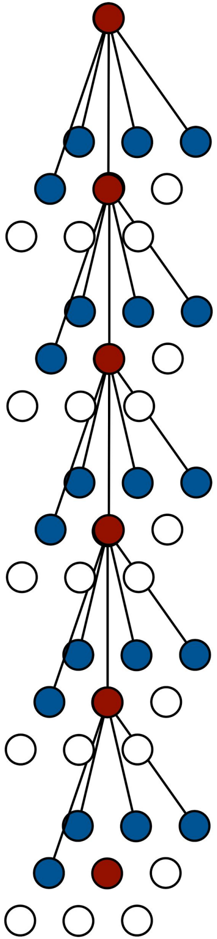
Colors

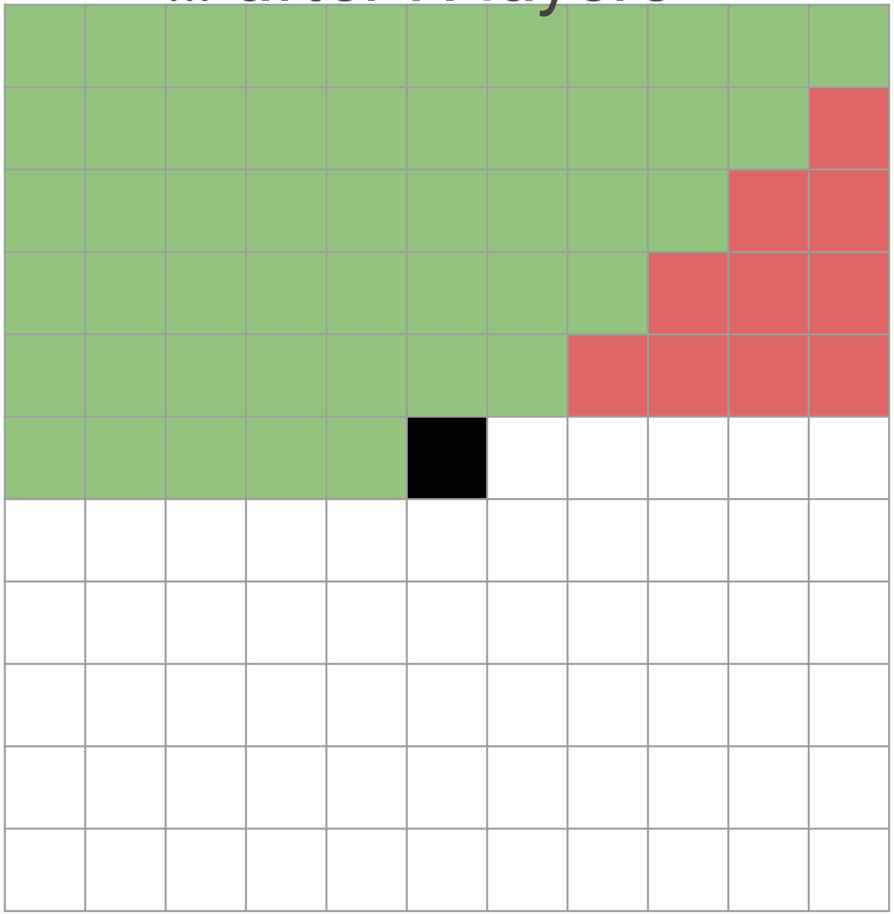
Pixel receptive field after 1 causal layer





… after N layers





Background: Autoregressive Models

Modeling images group-by-group

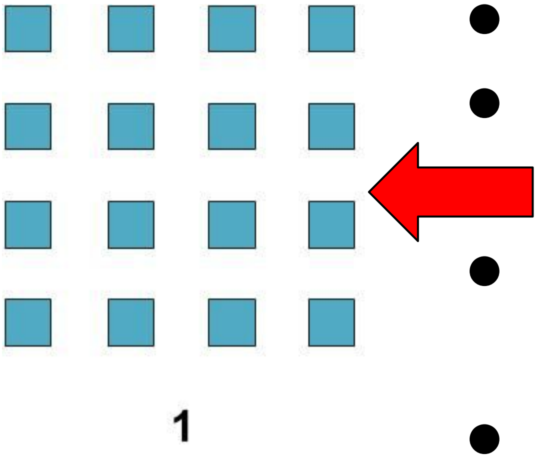
$$P(x;\theta) = \prod_{g=1}^{G} P(\mathbf{x}^g | \mathbf{x}^{<g}; \theta)$$

All pixels in group *g*

All pixels in all preceding groups

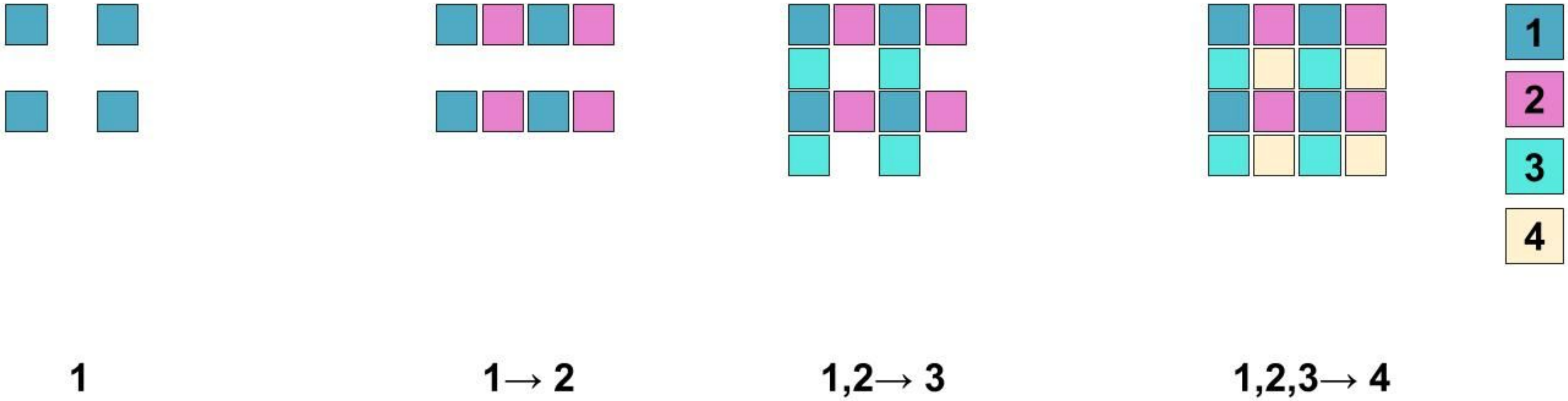
- Group structure encodes conditional independence assumptions.
- If G << N, sampling is cheaper than in pixel-by-pixel.

Parallel Autoregressive models in 2D



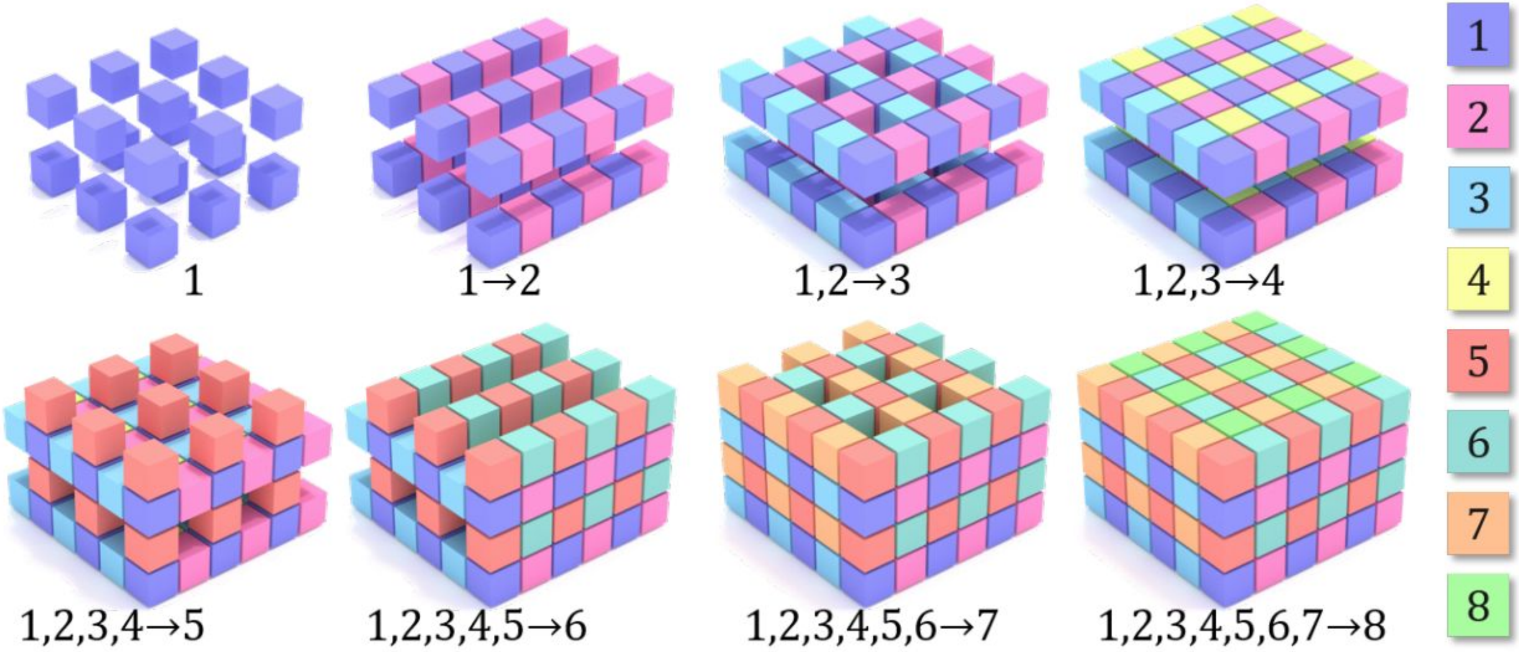
- Went from O(N) factors to O(1)...
- Wait! Where did these group 1 pixels come from?
- If we have enough context to model them as independent, generate in parallel.
- Otherwise, recurse.

Parallel Autoregressive models in 2D



- In total then, there will be O(log N) factors.

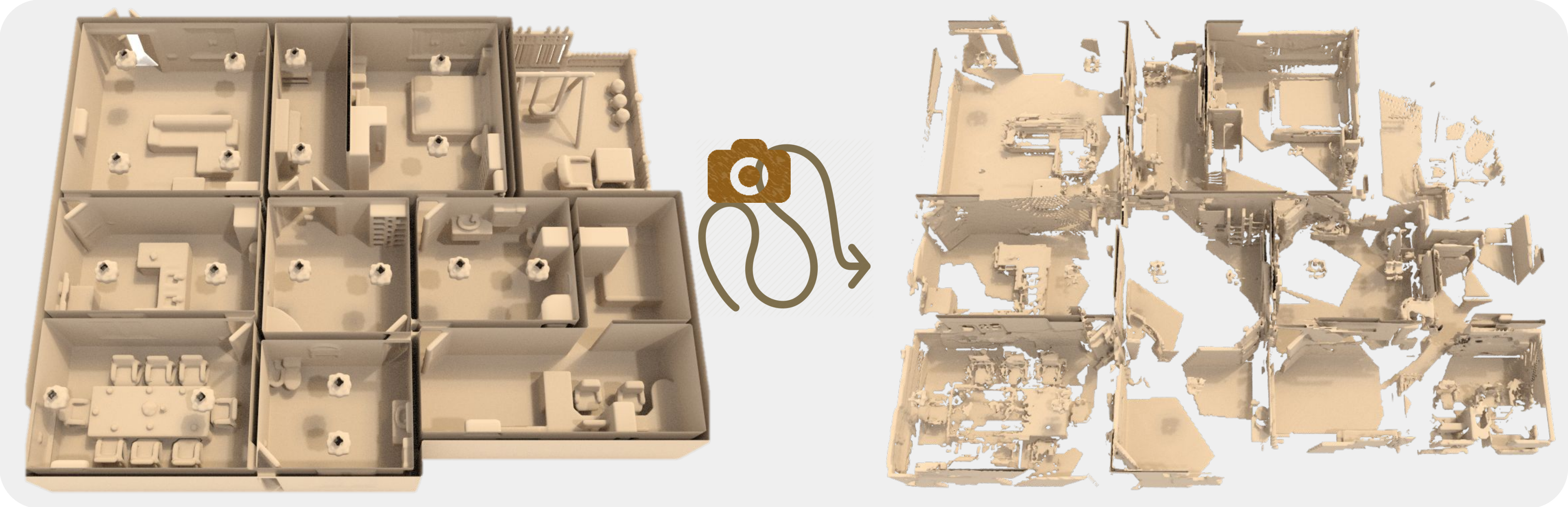
Parallel Autoregressive models in 3D



1. Angela Dai et al. "ScanComplete: Large-Scale Scene Completion and Semantic Segmentation for 3D Scans".

Application: Learning to Complete 3D Scans

Virtually scan synthetic data



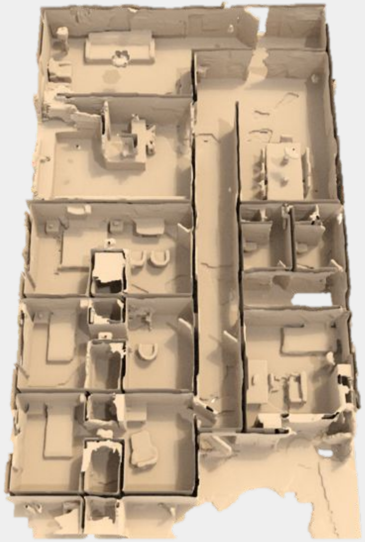
Scenes from SUNCG [Song et al. 17]

1. Angela Dai et al. “ScanComplete: Large-Scale Scene Completion and Semantic Segmentation for 3D Scans”.

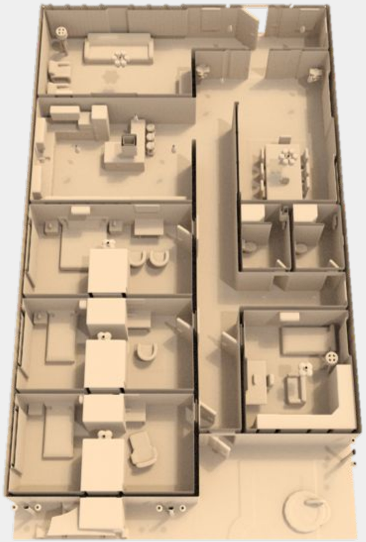
Application: Learning to Complete 3D Scans

Input

Completion



Ground Truth



Scenes from SUNCG [Song et al. 2017]

1. Angela Dai et al. "ScanComplete: Large-Scale Scene Completion and Semantic Segmentation for 3D Scans".

Application: Learning to Complete 3D Scans

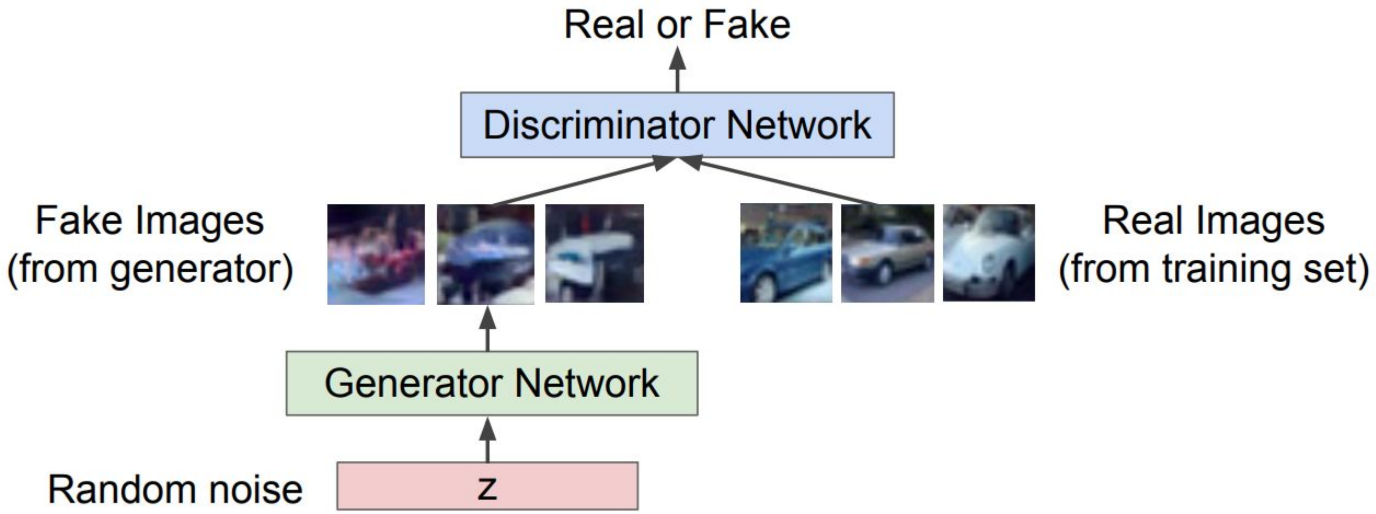
Part I: Background

Generative Adversarial Networks

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

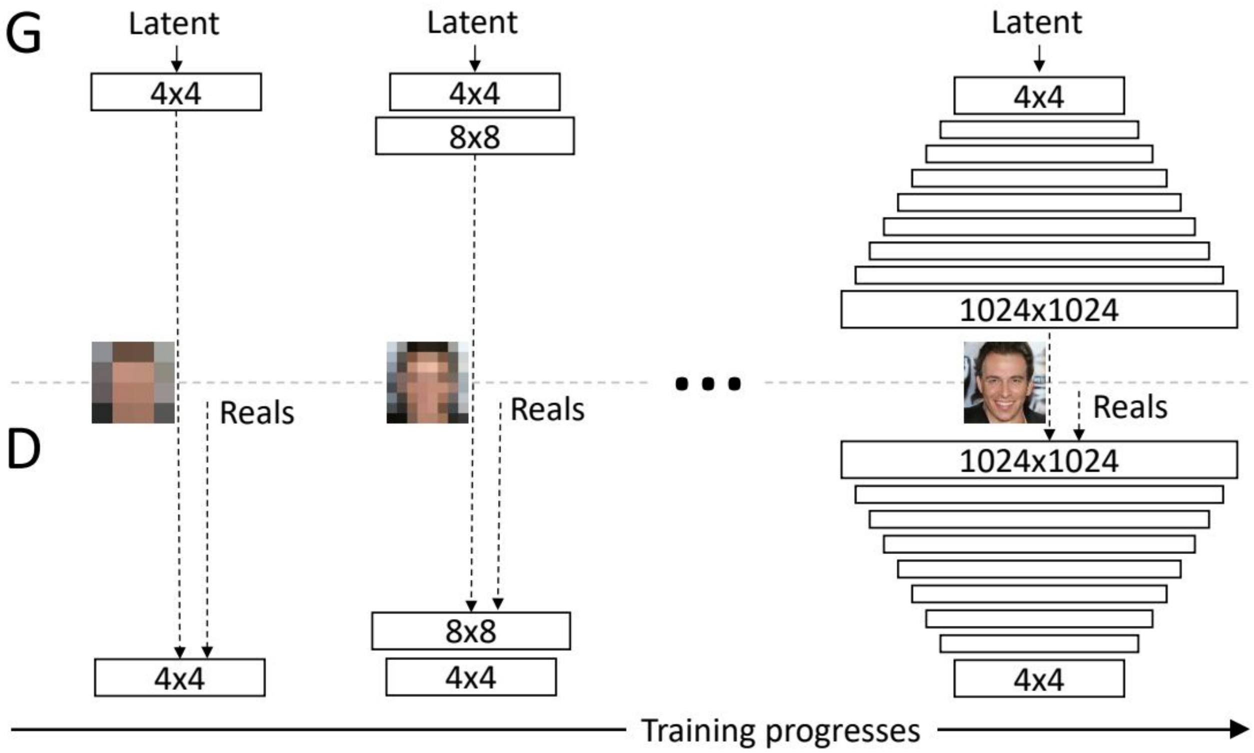
Minimax objective function:

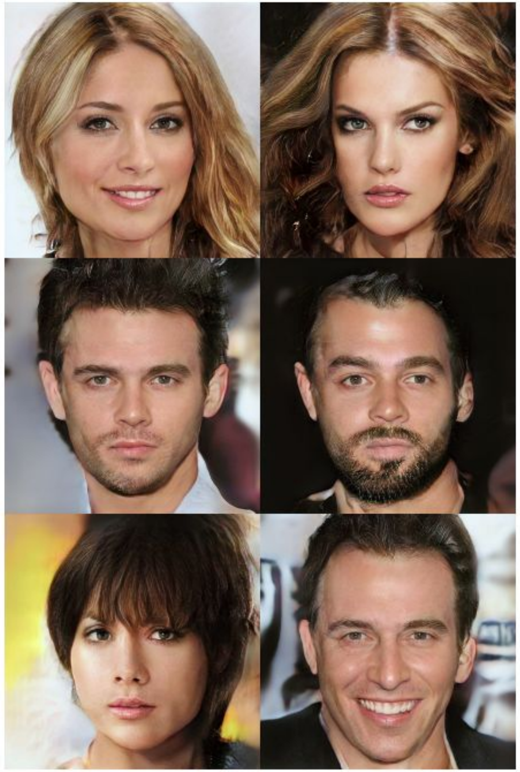
Discriminator outputs likelihood in (0,1) of real image

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{Discriminator output for real data x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}) \right]$$

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Recent breakthrough: Progressive Training





1. Karras, Tero, et al. "Progressive growing of gans for improved quality, stability, and variation." *ICLR, 2018.*

Part II: Frontiers

Learning from Limited Data

Learning from Limited Data - Attention PixelCNN

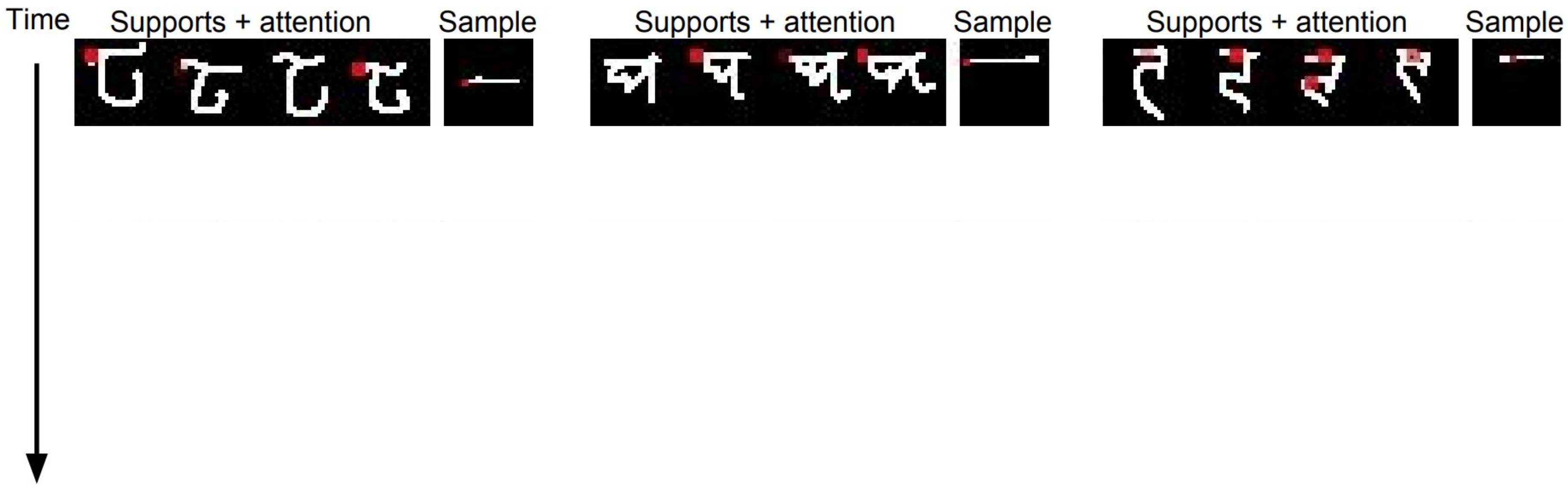
$$P(\mathbf{x}|s;\theta) = \prod_{t=1}^{N} P(x_t|x_{<t}, f(s);\theta)$$

At each pixel *t*, attend to *s* using latest context.

Let *s* be a small training set.

1. Reed, Scott, et al. "Few-shot Autoregressive Density Estimation: Towards Learning to Learn Distributions." *ICLR, 2018.*

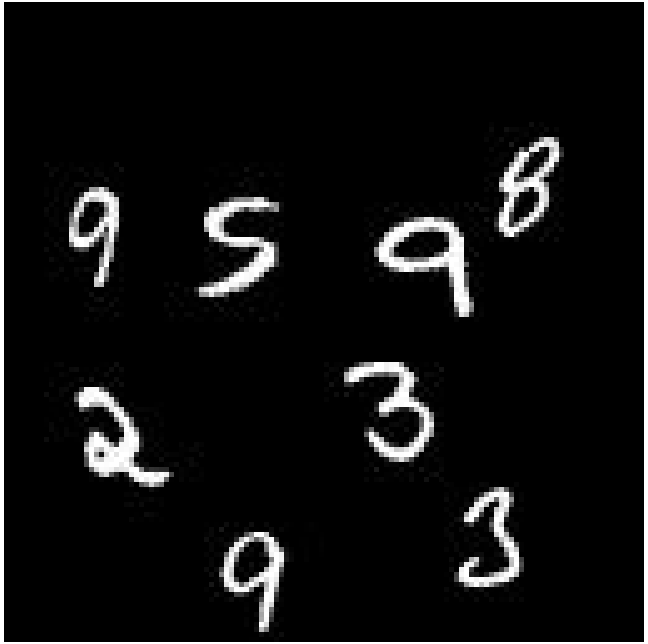
Learning from Limited Data - Attention PixelCNN



Part II: Frontiers

Predicting Far into the Future

The Problem: Cascading Errors

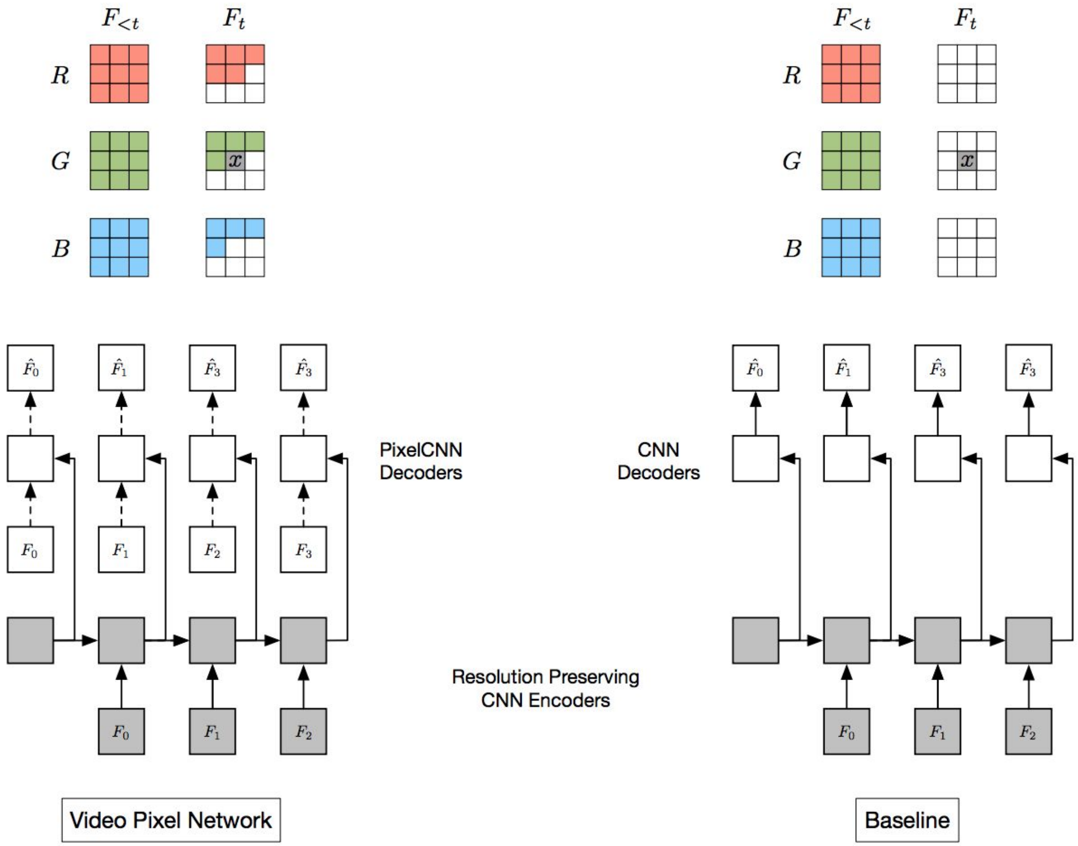


Using Convolutional LSTM
No within-frame dependencies

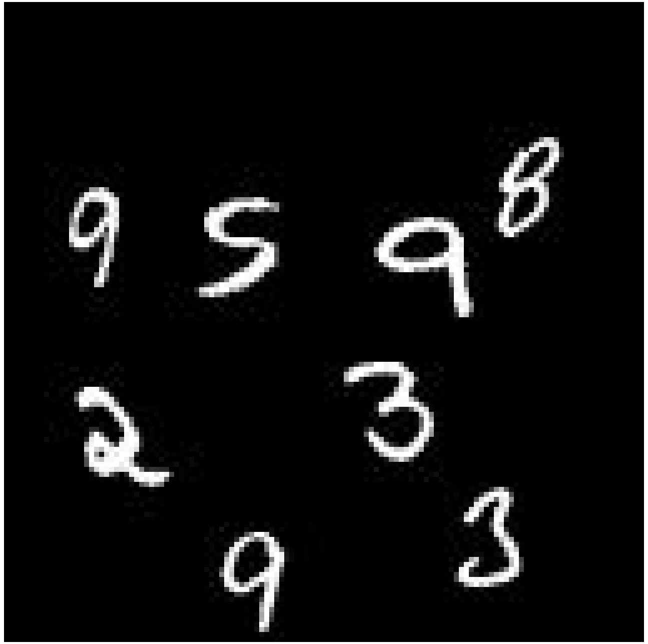
1. Kalchbrenner, Nal, et al. "Video Pixel Networks." *ICML*, 2017.

Solution #1: Train a really good model.

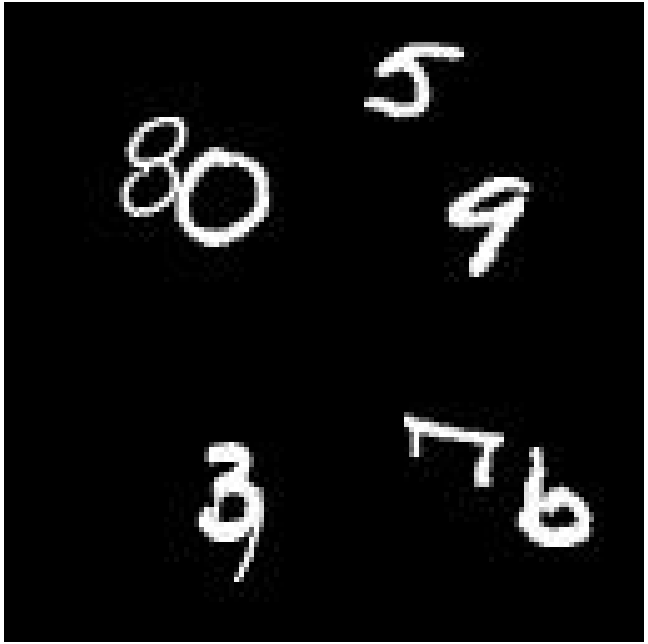
Take into account within-frame dependencies.



Bouncing MNIST



Using Convolutional LSTM
No within-frame dependencies



Using a very well-trained
Autoregressive model

1. Kalchbrenner, Nal, et al. "Video Pixel Networks." *ICML*, 2017.

Robot pushing dataset

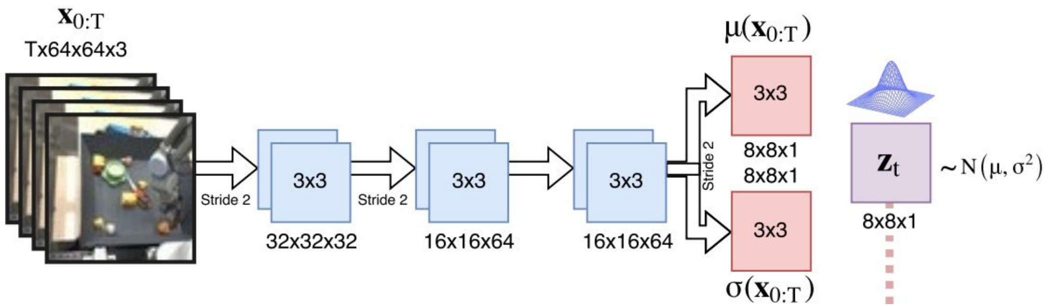
Using Convolutional LSTM
No within-frame dependencies

Using a very well-trained
Autoregressive model

1. Kalchbrenner, Nal, et al. "Video Pixel Networks." *ICML*, 2017.

… But, eventually it blows up too, sometime after 20 frames.

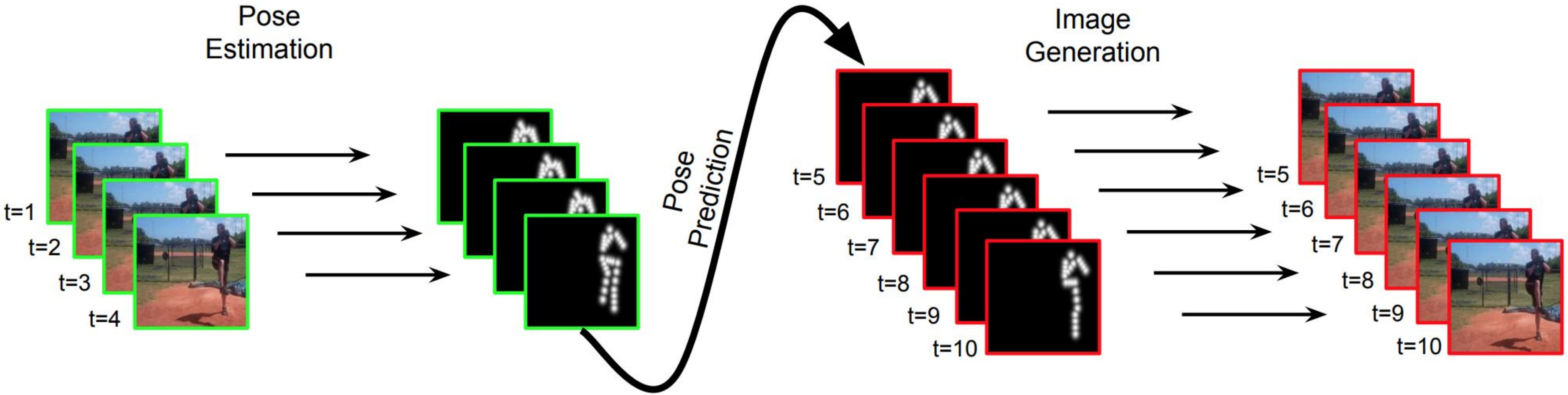
Solution #2: Model global structure using VAE



1. Babaeizadeh, Mohammad, et al. "Stochastic Variational Video Prediction." *ICLR, 2018*


5
10
18
28
28 (zoom)
Ground Truth

Solution #3: Generate video hierarchically



1. Villegas, Ruben, et al. "Learning to Generate Long-term Future via Hierarchical Prediction." *ICML*, 2017.


Input frames

Part II: Frontiers

Generative Models for Agents

Exploration: Montezuma's Revenge

1. Bellemare, Marc, et al. "Unifying count-based exploration and intrinsic motivation." *NIPS*, 2016.

Using density models to improve exploration

Prediction Gain (PG) at time step *n*:

$$\mathrm{PG}_n(x) = \underbrace{\log \rho'_n(x)}_{\text{Log-likelihood } \textbf{after} \text{ seeing x.}} - \underbrace{\log \rho_n(x)}_{\text{Log-likelihood } \textbf{before} \text{ seeing x.}}$$

1. Ostrovski, Georg, et al. "Count-Based Exploration with Neural Density Models." *ICML,* 2017.

Pseudo-Counts

Desired property: a single observation of *x* should lead to a unit increase in pseudo-count:

$$\rho_n(x) = \frac{\hat{\mathrm{N}}_n(x)}{\hat{n}}, \quad \rho'_n(x) = \frac{\hat{\mathrm{N}}_n(x) + 1}{\hat{n} + 1}$$

With some algebra, we can define $N_n(x)$ only using the density model:

$$\hat{\mathrm{N}}_n(x) = \frac{\rho_n(x)(1 - \rho'_n(x))}{\rho'_n(x) - \rho_n(x)}$$

1. Ostrovski, Georg, et al. "Count-Based Exploration with Neural Density Models." *ICML,* 2017.

Reward bonus

$N_n(x)$ can be estimated using the prediction gain:

$$\hat{\mathrm{N}}_n(x) \approx \left(e^{\mathrm{PG}_n(x)} - 1\right)^{-1}$$

Use $N_n(x)$ to provide dense rewards as a “reward bonus”.

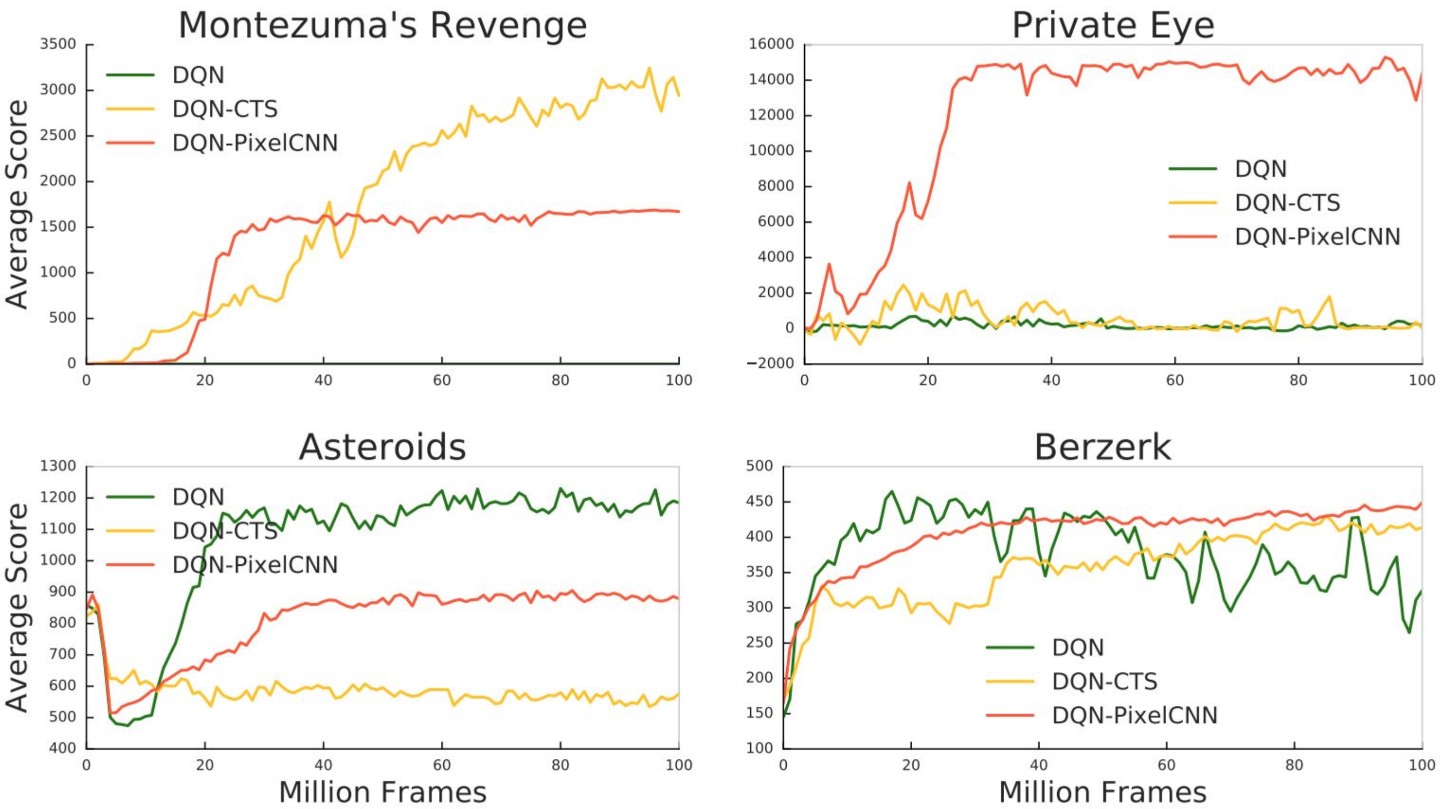
$$r^+(x) := (\hat{\mathrm{N}}_n(x))^{-1/2}$$

1. Ostrovski, Georg, et al. "Count-Based Exploration with Neural Density Models." *ICML,* 2017.

Results

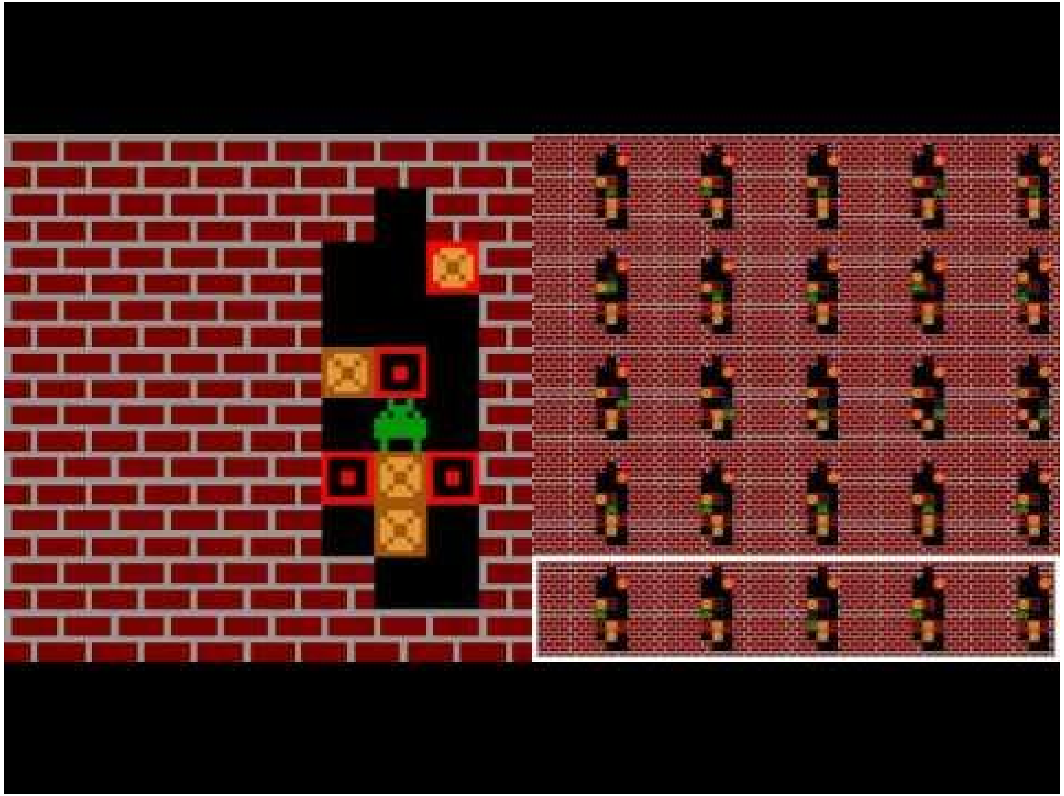
Hard exploration

Easier exploration



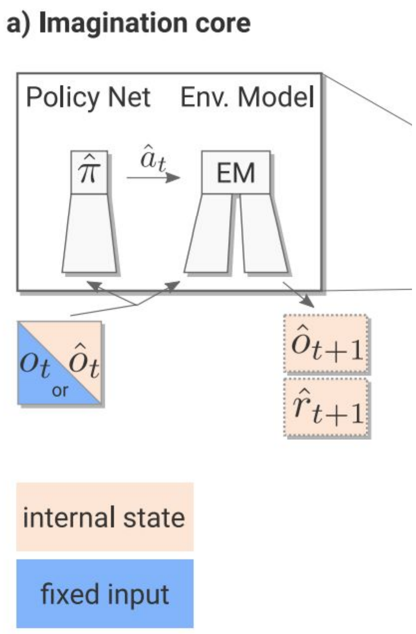
Trends and Frontiers in Deep Generative Models
— **Scott Reed**

Using generative models for planning in Sokoban



1. Racanière, Sébastien, et al. "Imagination-Augmented Agents for Deep Reinforcement Learning." *NIPS*. 2017.

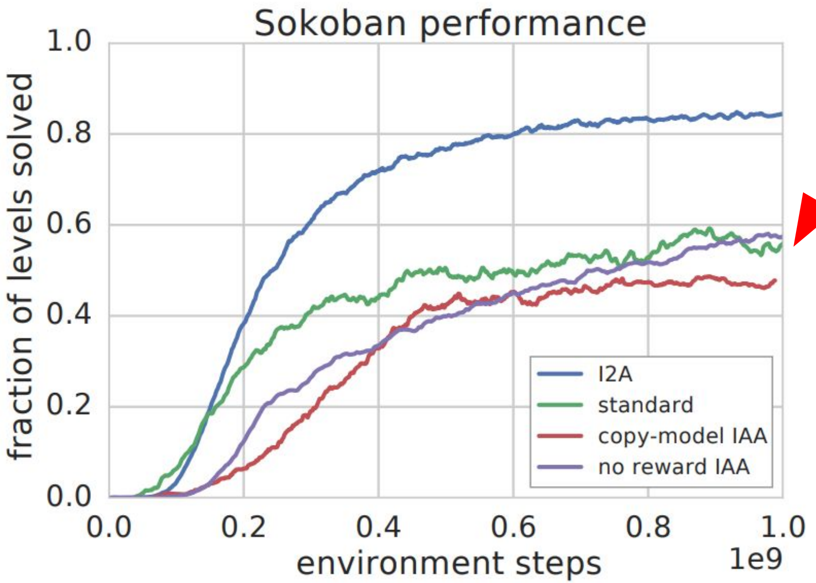
Using generative models for planning in Sokoban

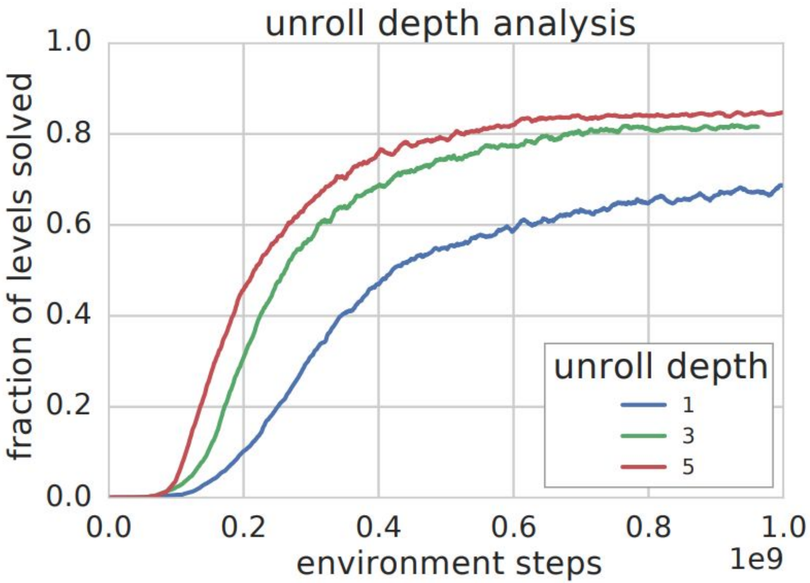


1. Racanière, Sébastien, et al. "Imagination-Augmented Agents for Deep Reinforcement Learning." *NIPS*. 2017.

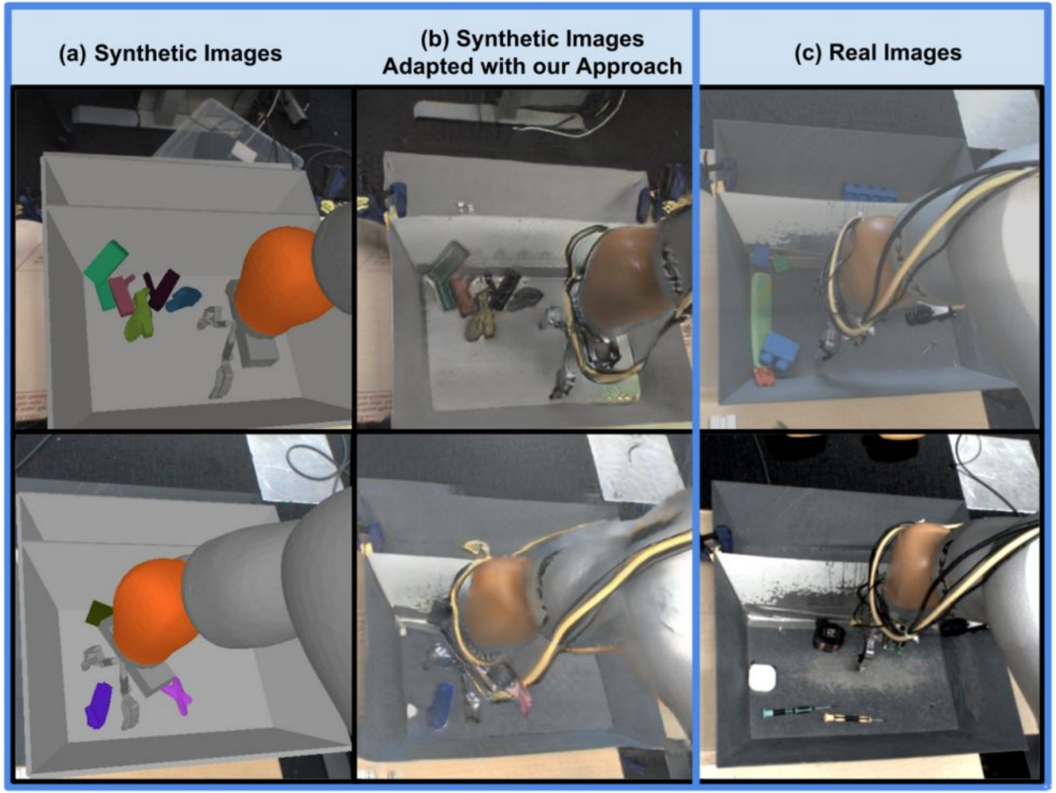
Learning curves

Model-free baselines



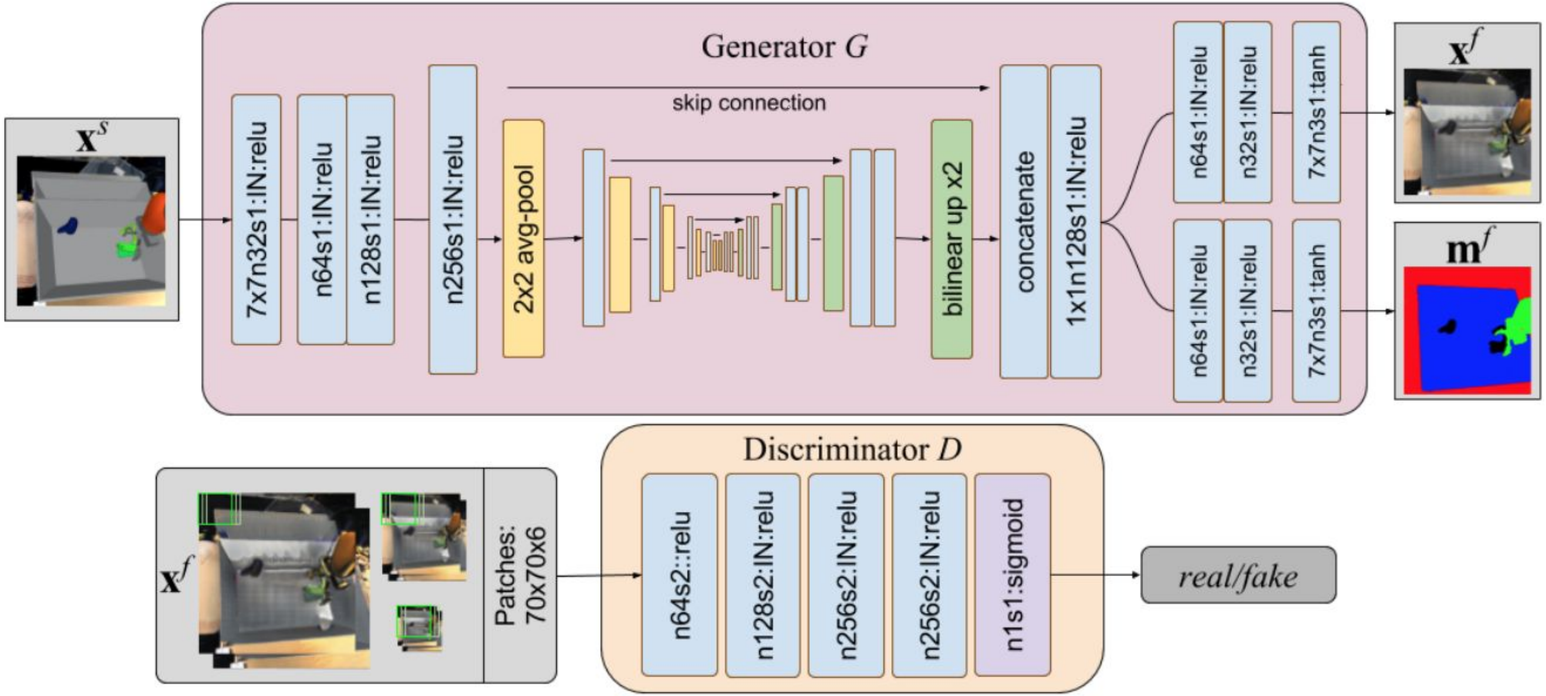


Bridging the simulation to reality gap



1. Bousmalis et al. “Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping”.

GraspGAN



Look realistic

Preserve relevant content

1. Bousmalis et al. “Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping”.

Conclusions

- Deep generative models are already ubiquitous in consumer applications, using autoregressive models:
  - Android text-to-speech
  - Neural machine translation
- Generating high-res natural images is starting to work, in narrow domains (e.g. faces).
- Generative models begin to be useful for agents on simple tasks (Atari, grasping).

Thank You!

Questions?