Pixel RNN

Google DeepMind

Presented by Xinrui Hua

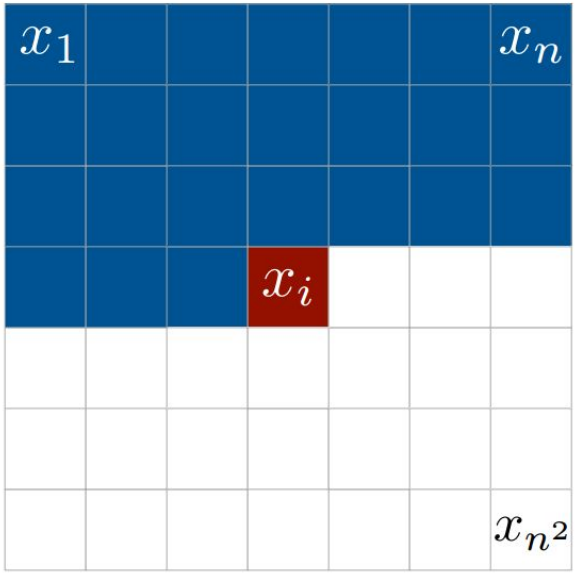
Model

Model joint distribution of image pixels tractably

Use a product of conditional distribution

$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i|x_1, ..., x_{i-1})$$

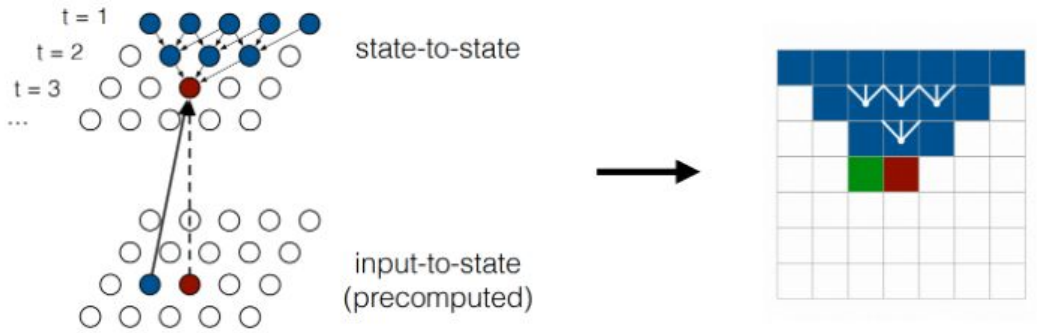
$$p(x_{i,R}|\mathbf{x}_{<i})p(x_{i,G}|\mathbf{x}_{<i}, x_{i,R})p(x_{i,B}|\mathbf{x}_{<i}, x_{i,R}, x_{i,G})$$



Row LSTM

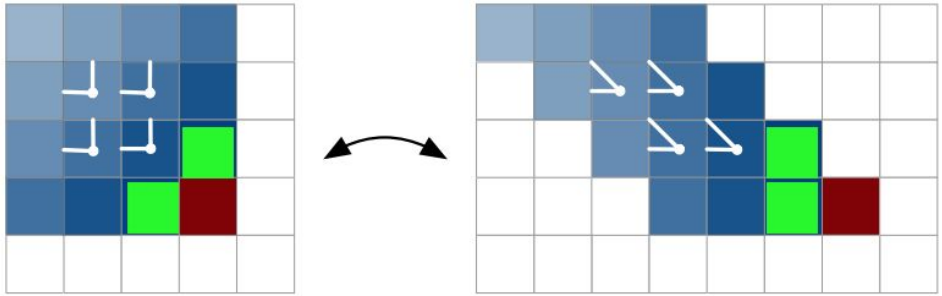
- capture a triangular shape context
- parallel compute input-to-state maps

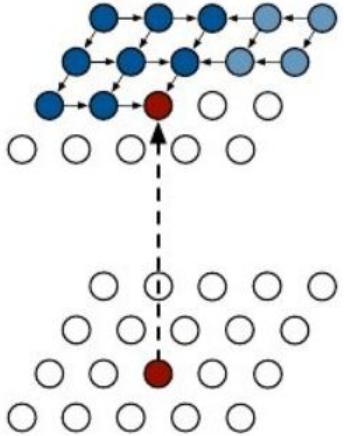
$$[\mathbf{o}_i, \mathbf{f}_i, \mathbf{i}_i, \mathbf{g}_i] = \sigma(\mathbf{K}^{ss} \circledast \mathbf{h}_{i-1} + \mathbf{K}^{is} \circledast \mathbf{x}_i)$$



Diagonal BiLSTM

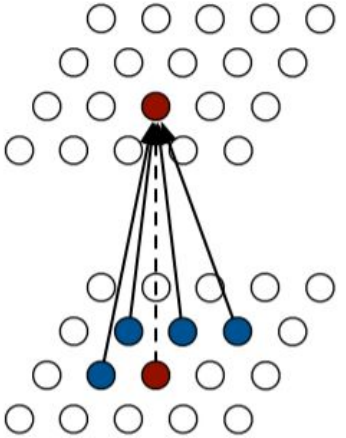
- scan the image in diagonal
- capture the entire avilable context
- parallelized by skew operation





PixelCNN

- Use bounded receptive field to replace unbounded dependency
- Turn the problem into pixel level classification problem
- Faster during training and testing step

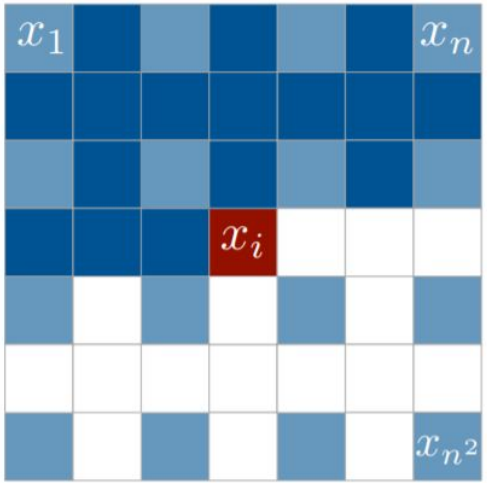


Detailed architectures

PixelCNN	Row LSTM	Diagonal BiLSTM
7×7 conv mask A		
Multiple residual blocks: (see fig 5)		
Conv 3×3 mask B	Row LSTM i-s: 3×1 mask B s-s: 3×1 no mask	Diagonal BiLSTM i-s: 1×1 mask B s-s: 1×2 no mask
ReLU followed by 1×1 conv, mask B (2 layers)		
256-way Softmax for each RGB color (Natural images) or Sigmoid (MNIST)		

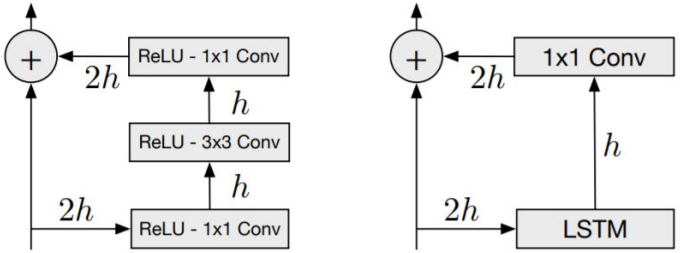
Multi-Scale PixelRNN

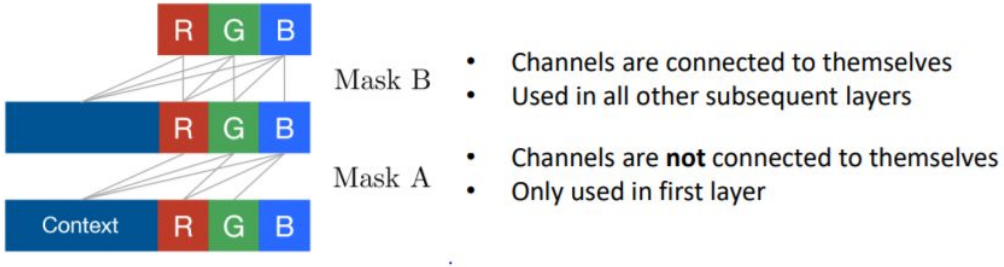
- Unconditional PixelRNN + conditional PixelRNN
- First generate low resolution images
- Take low resolution images as prior knowledge



Optimizations:

- Pixels as discrete variables
- Residual connections
- Masked convolution





Experiment Results

Model	NLL Test (Train)
Uniform Distribution:	8.00
Multivariate Gaussian:	4.70
NICE [1]:	4.48
Deep Diffusion [2]:	4.20
Deep GMMs [3]:	4.00
RIDE [4]:	3.47
PixelCNN:	3.14 (3.08)
Row LSTM:	3.07 (3.00)
Diagonal BiLSTM:	**3.00** (2.93)

Model	NLL Test
DBM 2hl [1]:	≈ 84.62
DBN 2hl [2]:	≈ 84.55
NADE [3]:	88.33
EoNADE 2hl (128 orderings) [3]:	85.10
EoNADE-5 2hl (128 orderings) [4]:	84.68
DLGM [5]:	≈ 86.60
DLGM 8 leapfrog steps [6]:	≈ 85.51
DARN 1hl [7]:	≈ 84.13
MADE 2hl (32 masks) [8]:	86.64
DRAW [9]:	≤ 80.97
PixelCNN:	81.30
Row LSTM:	80.54
Diagonal BiLSTM (1 layer, $h = 32$):	**80.75**
Diagonal BiLSTM (7 layers, $h = 16$):	**79.20**

Pros

Discrete Softmax Distribution

- Intuitive and easy to implement
- Regression probelm to classification
- All are inside [0, 255]
- Multimodal, skewed, peaked or long tailed
- Better results:
    - Row LSTM: 3.06 bits/dim
    - MCGSM: 3.22 bits/dim

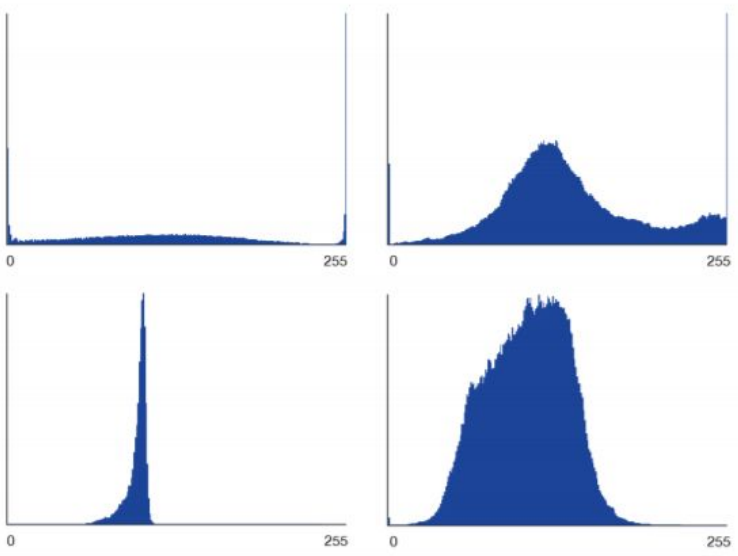
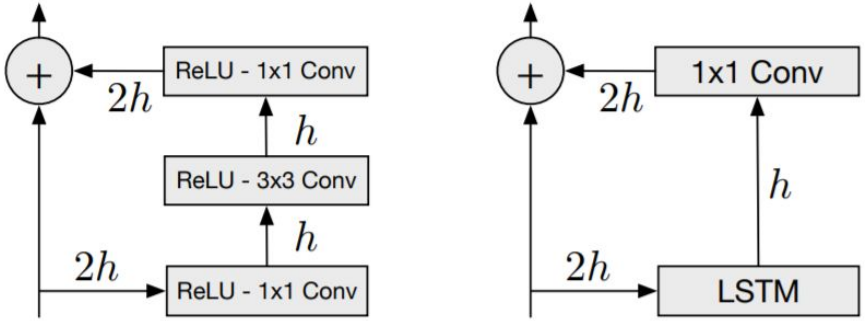


Figure: Example softmax outputs in the final layer, representing probability distribution over 256 classes.

Residual Connections

- Increase convergence speed
- Deep network
  - PixelRNN 12 layers
  - PixelCNN 15 layers
- Not require additional gates
- Better results



	No skip	Skip
No residual:	3.22	3.09
Residual:	3.07	3.06

Multiple Models and Experiments

- PixelCNN
- Row LSTM
- Disgonal BiLSTM
- Multi-Scale PixelRNN

- MNIST
- CIFAR-10
- ImageNet

Compared with GAN

- Easy and stable to train
- Provides a way to calculate exact likelihood
- Works for discrete data

- Much slower
- Image quality is lower