Modern autoregressive models

Figure 1. Image completions sampled from a PixelRNN.

PixelRNN

- Decompose the data likelihood of an $n \times n$ image $p(\boldsymbol{x}) = \prod_{i=1}^{n^2} p(x_i|\boldsymbol{x}_{<i})$
- Each pixel conditional corresponds to a triplet of colors
  → Further decompose per color (same as above)

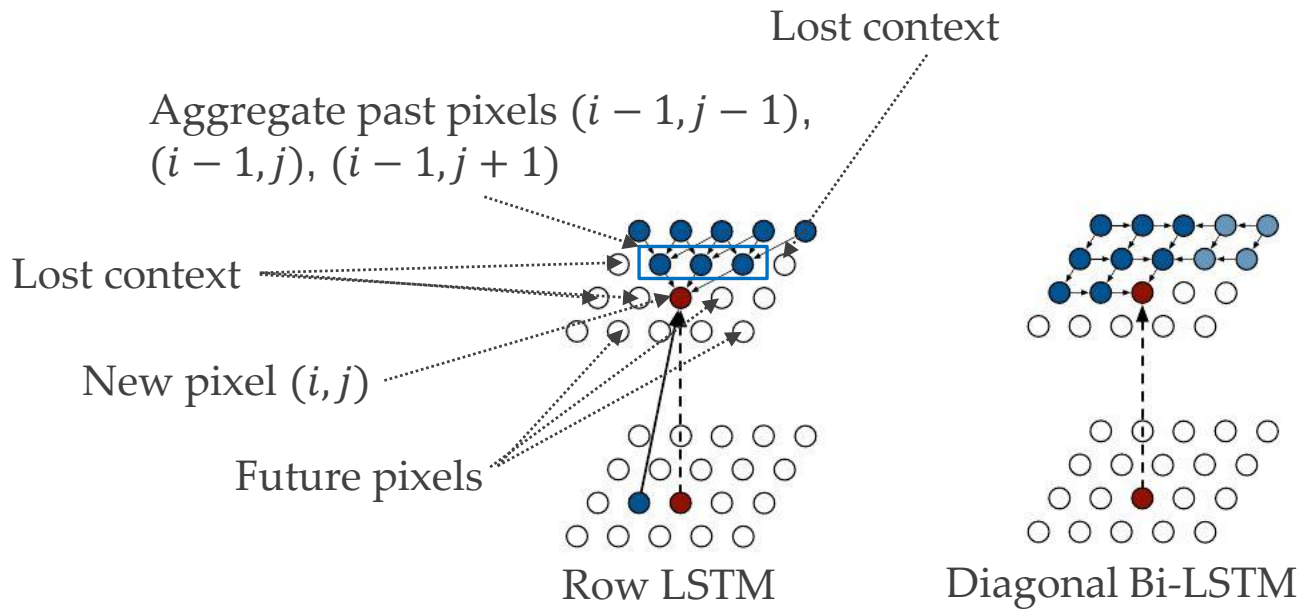
$$p(x_i|\boldsymbol{x}_{<i}) = p\left(x_{i,R}|\boldsymbol{x}_{<i}\right) \cdot p\left(x_{i,G}|\boldsymbol{x}_{<i}, x_{i,R}\right) \cdot p\left(x_{i,B}|\boldsymbol{x}_{<i}, x_{i,R}, x_{i,G}\right)$$

- Model the conditionals $p\left(x_{i,R}|x_{<i}\right)$, ... with 12-layer convolutional RNN
  - The MLP from NADE cannot easily scale and statistics are not shared
- Model the output as a categorical distribution
  - 256-way softmax

van den Oord, Kalchbrenner and Kavukcuoglu, Pixel Recurrent Neural Networks

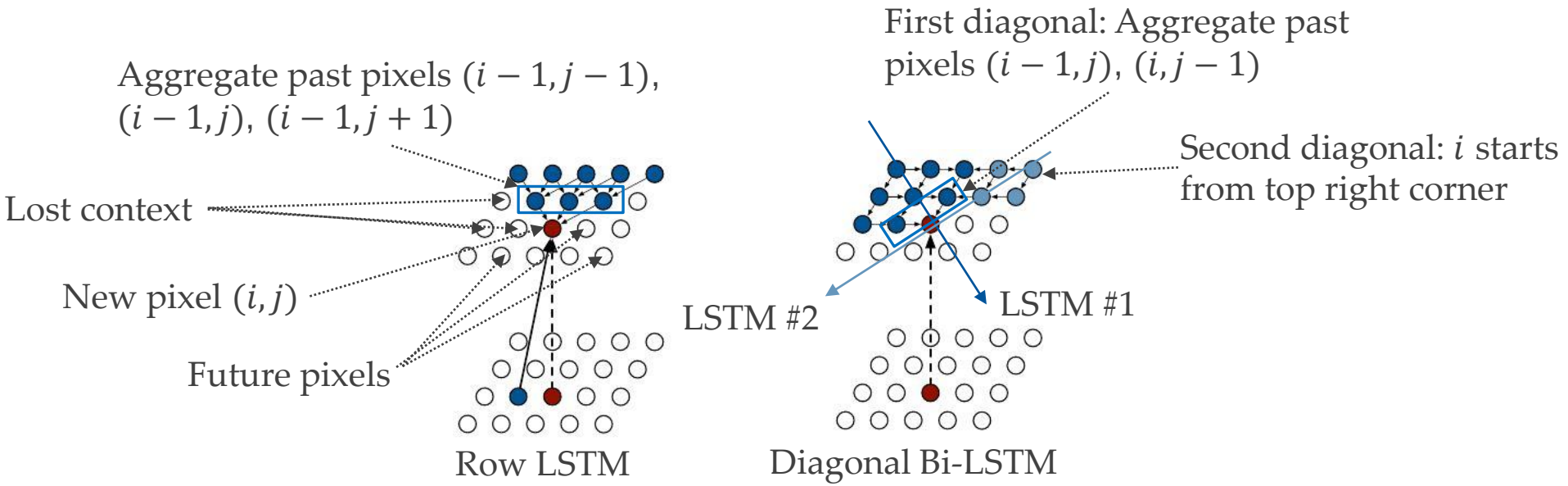
Row LSTM

- Row LSTM with 'causal' triangular receptive field
  - Per new pixel (row i) use 1-d conv (size 3) to aggregate pixels above $(i-1)$
  - The effective receptive field spans a triangle
  - Convolution only on 'past' pixels $(i-1)$, not 'future pixels' → causal
  - Loses some context



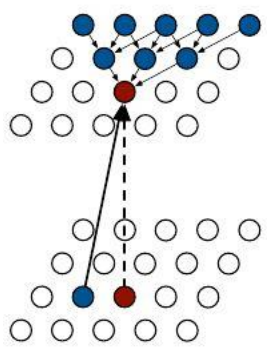
Diagonal BiLSTM

- Have two LSTMs moving on oppose diagonals
  - First diagonal: the convolution past is $(i-1, j)$, $(i, j-1)$
- Combine the two LSTMs
  - recursively the entirety of past context is captured

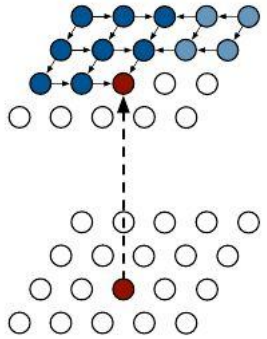


Why not a regular LSTM?

- It would require sequential, pixel-wise computations
  - Less parallelization
  - Slower training
- With Row LSTM and Diagonal BiLSTM we process one row at a time
  - Parallelization possible



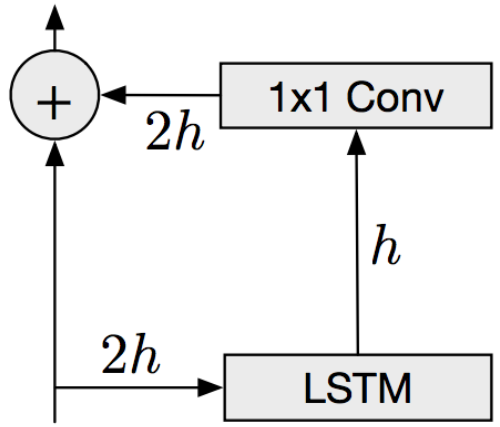
Row LSTM



Diagonal Bi-LSTM

Deep LSTMs with Residual connections

- Use 12 layers of LSTMs
- Add residual connections to speed up learning
- Although good modelling of $p(x) \rightarrow$ nice image generation
- Slow training because of LSTM, slow generation

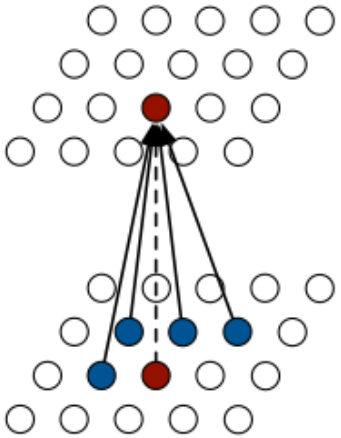


PixelRNN - Generations

Figure 1. Image completions sampled from a PixelRNN.

PixelCNN

- Replace LSTMs with fully convolutional networks
  - 15 layers
  - No pooling layers to preserve spatial resolution
- Use masks to mask out future pixels in convolutions
  - Otherwise 'access to future' → no 'autoregressiveness'
- Faster training as no recurrent steps required
  → Better parallelization
  - Pixel generation still sequential and thus slow



PixelCNN

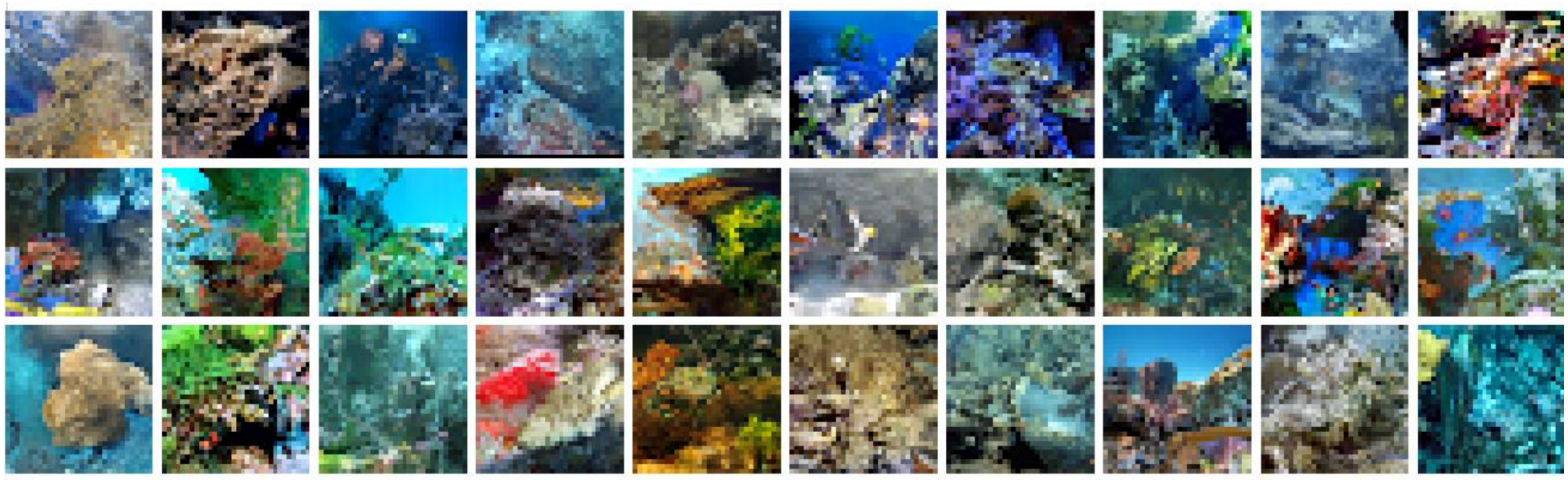
1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

Masking convolutions

van den Oord, Kalchbrenner and Kavukcuoglu, Pixel Recurrent Neural Networks

PixelCNN - Generations

Coral reef



PixelCNN - Generation

Sorrel horse

PixelCNN - Generation

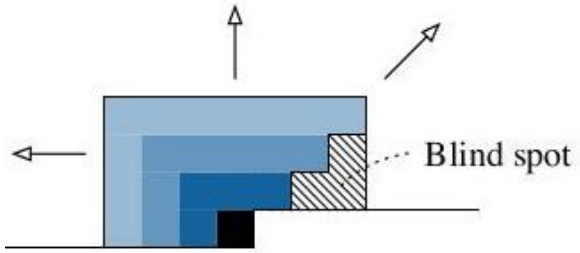
Sandbar

PixelCNN - Generation

Lhasa Apso

PixelCNN: Pros and Cons

- Faster training
- Performance is worse than PixelRNN as context is discarded
- The cascaded convolutions create a 'blind spot'
  - Use Gated PixelCNN to fix



- No latest space
- PixelCNN++ improves PixelCNN by (Salimans et al.)
  - Model output by discretized logistic mixture likelihood ← Softmax requires to many parameters and yields very sparse gradients
  - Condition on whole pixels, not colors
  - Architectural innovations

Autoregressive models: pros and cons

- Top density estimation
- They take into account complex co-dependencies
  - Potentially, better generations and more accurate likelihoods
- Autoregressive models are not necessarily latent variable models
  - They neither have necessarily an encoder nor learn representations
- Slow in learning, inference and generation
  - Computations are sequential (one at a time) → limited parallelism
  - E.g., to generate the next word we must generate past words first
- They may introduce artificial bias when assumed order is imposed