Autoregressive Generative Models

PixelRNN, PixelCNN and WaveNet

2019-07-05

곽민구

Contents

Contents

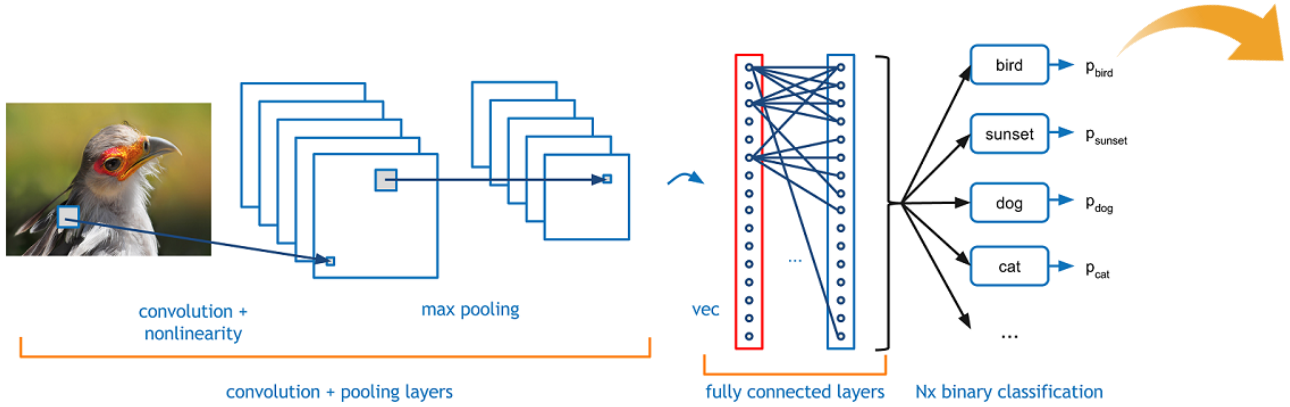
01 | Autoregressive & Generative

- 어떠한 작업을 할 것인가?
- 어떤 종류의 데이터를 사용할 것인가?
- 어떤 방법을 사용하는 것이 효과적일까?

Autoregressive + Generative

01 | Generative

- 어떤 작업을 할 것인가?
- Density modeling
  - ✓ 컴퓨터 비전에서 이미지 분류와 같은 분야에서는 많은 발전을 이루었으나, 이는 data와 label 사이의 관계를 학습한 것 (Supervised)
  - ✓ 데이터 자체로써 가지고 있는 특징을 이해하기 위한 시도 (Unsupervised)



[이미지 분류]

- Goal: 데이터 갖는 True Distribution을 학습하는 것
  → 데이터에 대한 모든 것을 학습하고 싶다
- Maximum likelihood

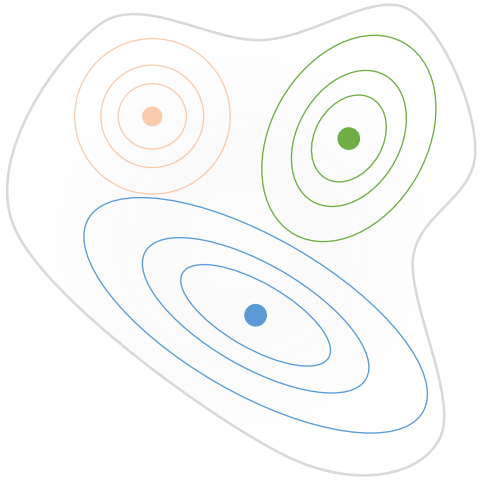
$D = \{x\}$

$L(D) = \sum_{x \in D} -\log p(x)$

[Density Modeling]

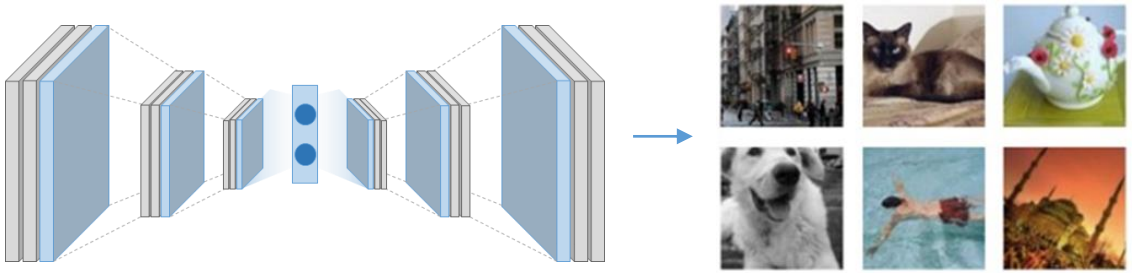
01 | Generative

- Density modeling을 통해 generative model을 얻을 수 있다
- Generative (생성) = 우리가 학습하지 않은 새로운 데이터를 만들어 내는 것
  - ✓ What I cannot create, I do not understand – Richard Feynman



- log-likelihood
- 주어진 데이터에 대해서
  얼마나 잘 학습했는지
  수치적으로 정보를 줌

[Density Modeling]

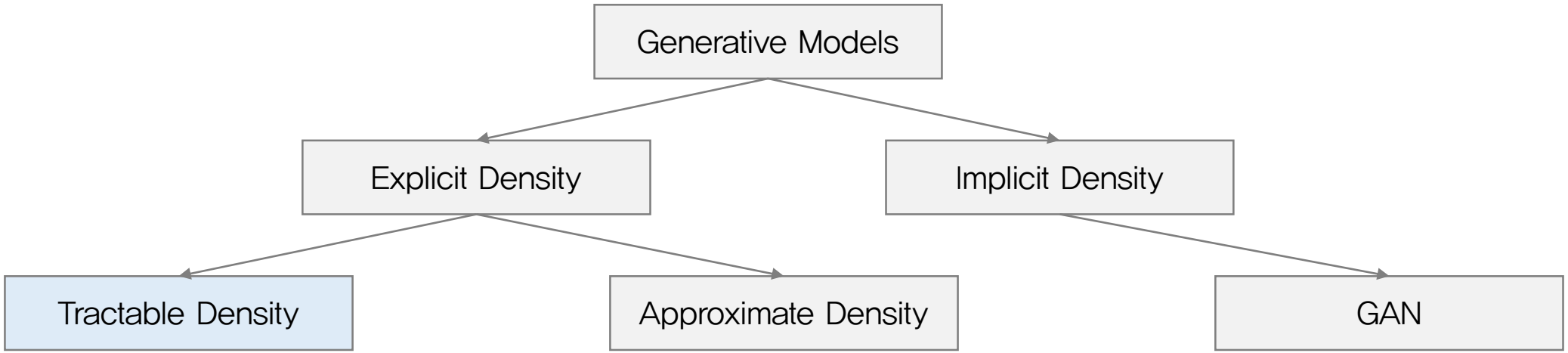


- Data generation
  → 우리가 학습한 데이터의 '패턴' & '구조'를 알 수 있음

[Generative Model]

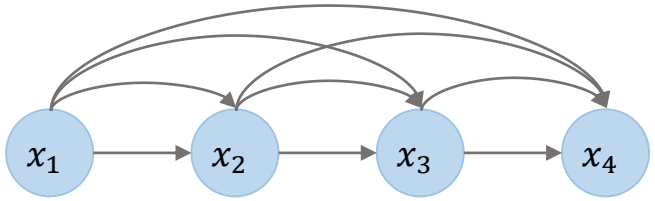
01 | Taxonomy of Generative Models

- 다양한 종류의 모델들이 존재
- 어떤 종류의 데이터를 사용할 것인가? & 어떤 방법을 사용하는 것이 효과적일까?



01 | Autoregressive

- 자기 자신을 입력으로 하여 자기 자신을 예측하는 모델이며, 이전 상태의 정보들에 기반하여 (조건부) 예측을 수행
- 텍스트, 오디오, 이미지 등의 다양한 종류의 데이터의 특성을 반영하기에 좋은 구조
- Necessary Math → The Chain Rule for Probabilities



make a new image

make	a	new	image
make	a	new	image
make	a	new	image
make	a	new	image

Joint probability를
연속된 conditional probability로
분해

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1})$$

$$p(x_1, x_2, x_3, x_4) = p(x_4 | x_1, x_2, x_3)\, p(x_1, x_2, x_3)$$

$p(x_1)$

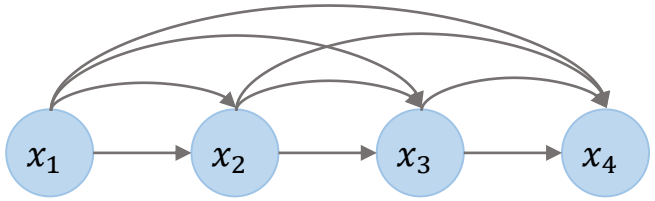
$p(x_2 | x_1)$

$p(x_3 | x_1, x_2)$

$p(x_4 | x_1, x_2, x_3)$

01 | Autoregressive

- 자기 자신을 입력으로 하여 자기 자신을 예측하는 모델이며, 이전 상태의 정보들에 기반하여 (조건부) 예측을 수행
- 텍스트, 오디오, 이미지 등의 다양한 종류의 데이터의 특성을 반영하기에 좋은 구조
- Necessary Math → The Chain Rule for Probabilities



make a new image

make a new image

make a new image

make a new image

make a new image

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1})$$

$$p(x_1, x_2, x_3, x_4) = \boldsymbol{p(x_4 | x_1, x_2, x_3)}\, p(x_1, x_2, x_3)$$

(참고) VAE
intractable density function
with latent z

$$p_\theta(\boldsymbol{x}) = \int p_\theta(z) p_\theta(x|z) dz$$

01 | Autoregressive

- Advantages and Disadvantages

장점

1. 정의하기가 쉬움
   → 데이터 시퀀스의 순서를 정의
2. Generation 과정이 쉽다 (개념적으로)
   → 매 step에서 예측된 결과물에서 샘플을 추출하여 그 다음 step에 새로운 input으로 넣어주는 것을 반복
3. 이미지, 오디오, 비디오 등에서 log-likelihood 결과가 좋음

단점

1. 순서를 정의하는 방법에 dependent
   → 학습 뿐만 아니라 generation 과정에서 예측값이 input으로 사용되기 때문
2. Generation 과정이 너무 오래 걸린다
   → 학습 과정에서는 병렬화가 가능하지만, sequential generation 특성상 너무 느리다
3. Predict one step ahead

Contents

02 | PixelRNN and PixelCNN

- Paper
- Äaron van den Oord

Pixel recurrent neural networks
A Oord, N Kalchbrenner, K Kavukcuoglu - arXiv preprint arXiv:1601.06759, 2016 - arxiv.org
... occluded completions original Figure 1. Image completions sampled from a **PixelRNN** ... The networks also incorporate residual connections (He et al., 2015) around LSTM layers; we observe that this helps with training of the **PixelRNN** for up to twelve layers of depth ...
☆ 99 666회 인용 관련 학술자료 전체 7개의 버전 »

Pixel Recurrent Neural Networks

Äaron van den Oord AVDNOORD@GOOGLE.COM
Nal Kalchbrenner NALK@GOOGLE.COM
Koray Kavukcuoglu KORAYK@GOOGLE.COM

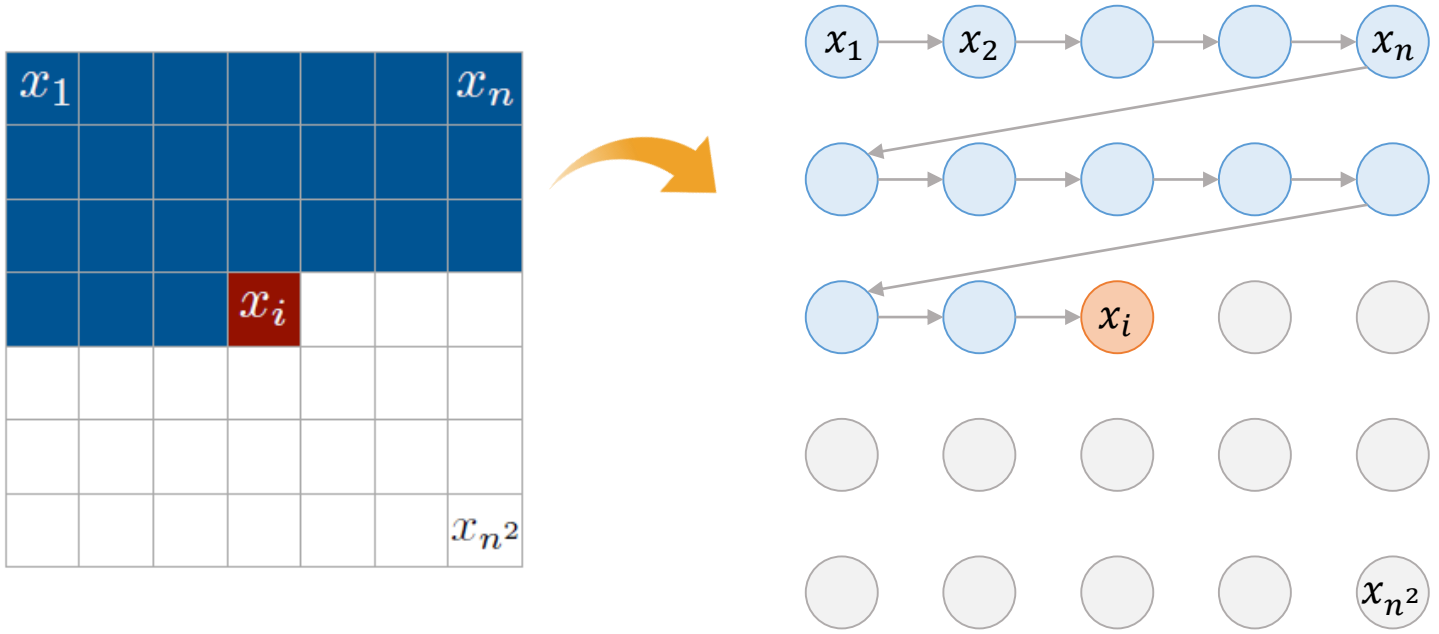
Google DeepMind

Abstract

Modeling the distribution of natural images is a landmark problem in unsupervised learning. This task requires an image model that is at once expressive, tractable and scalable. We present a deep neural network that sequentially predicts the pixels in an image along the two spatial dimensions. Our method models the discrete probability of the raw pixel values and encodes the complete set of dependencies in the image. Architectural novelties include fast two-dimensional recurrent layers and an effective use of residual connections in deep recurrent networks. We achieve log-likelihood scores on natural images that are considerably better than the previous state of the art. Our main results also provide benchmarks on the diverse ImageNet dataset. Samples generated from the model appear crisp, varied and globally coherent.

02 | PixelRNN

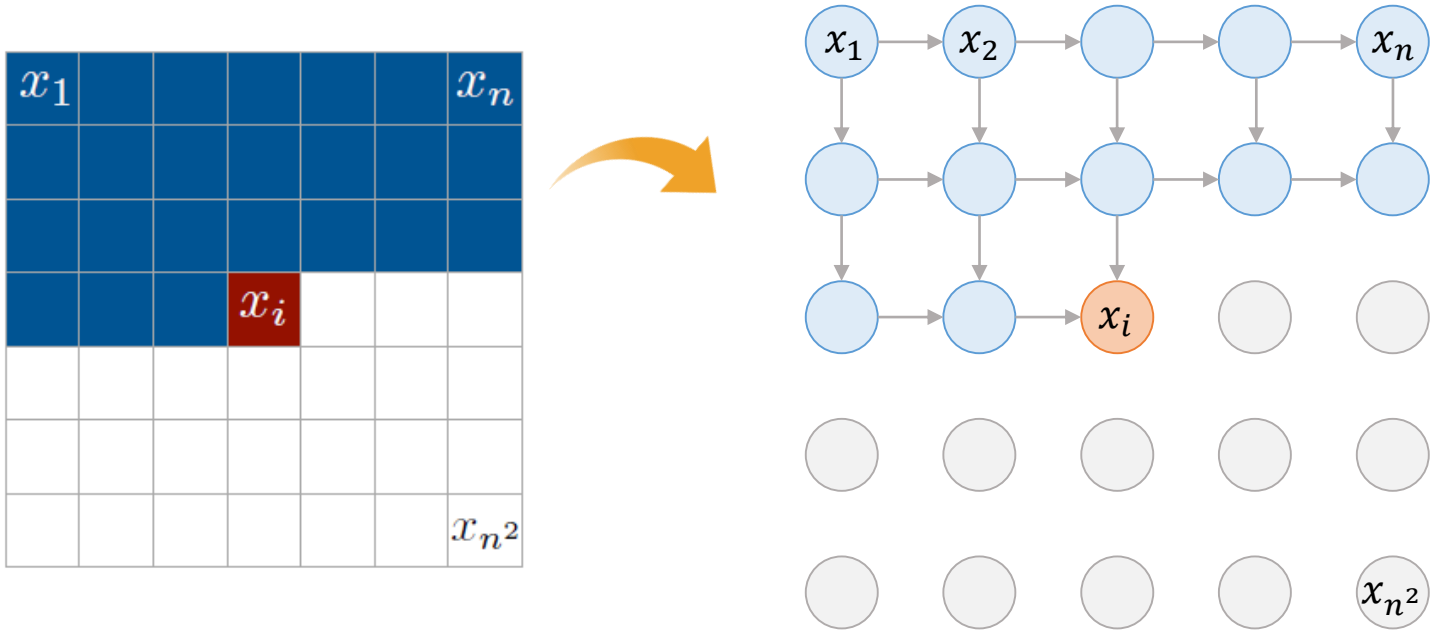
- 이미지의 픽셀 하나하나를 일련의 sequence로 생각
- Autoregressive modeling을 수행



1. 이미지에 있는 임의의 픽셀은
   이전 픽셀들로부터 영향을 받는다

02 | PixelRNN

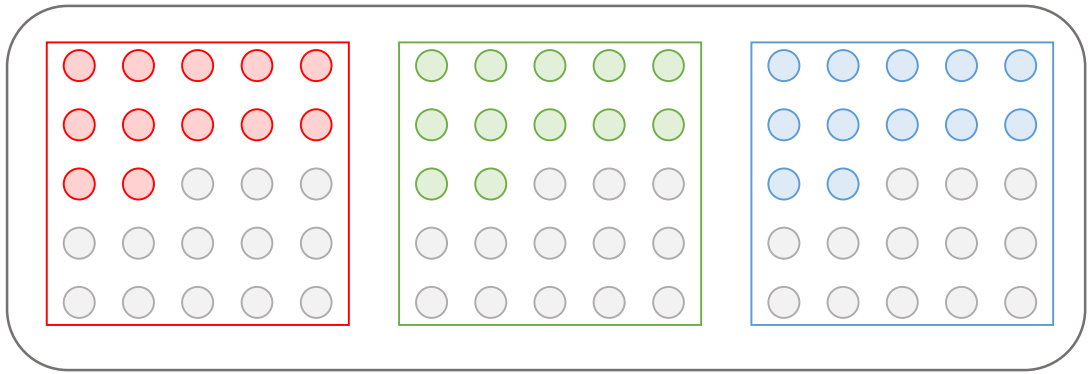
- 이미지의 픽셀 하나하나를 일련의 sequence로 생각
- Autoregressive modeling을 수행



1. 이미지에 있는 임의의 픽셀은 이전 픽셀들로부터 영향을 받는다
2. 이전 픽셀들이란, 왼쪽 위를 의미 (이미지의 특성을 반영)

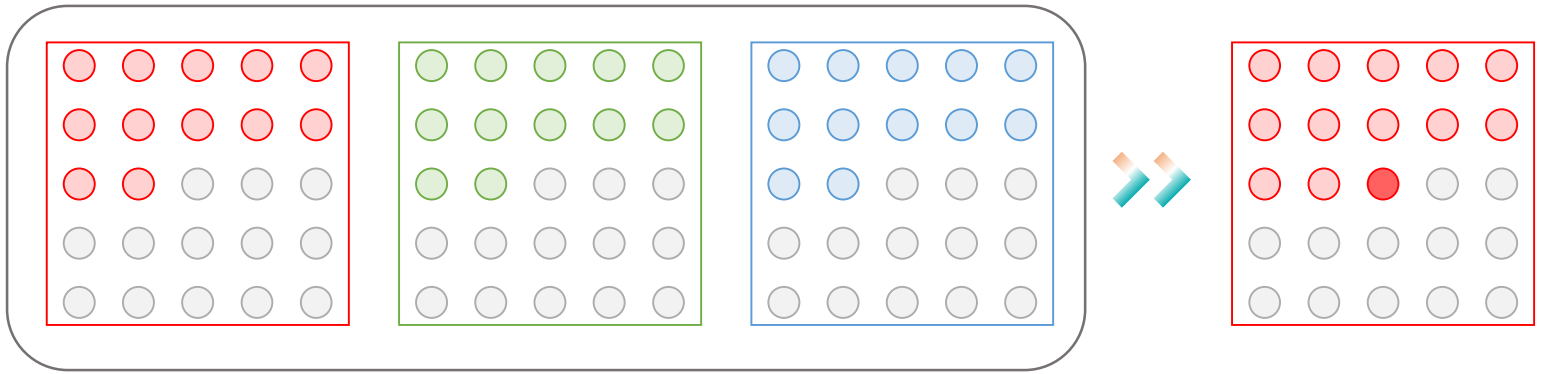
02 | Red-Green-Blue

- 이미지 채널간의 순서는 Red-Green-Blue 순서로 정의
  - ✓ $p(x_i|X_{<i}) = p(x_{i,R}|X_{<i})p(x_{i,G}|X_{<i}, x_{i,R})\ p(x_{i,B}|X_{<i}, x_{i,R}, x_{i,G})$



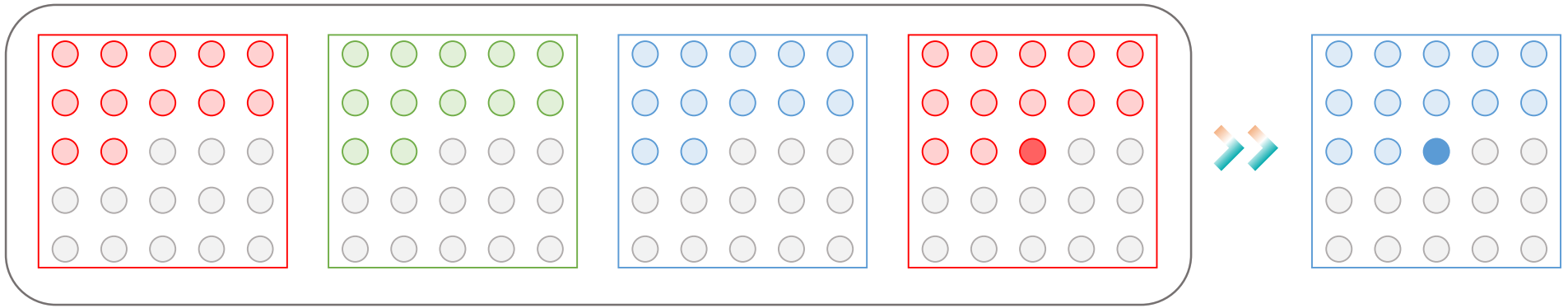
02 | Red-Green-Blue

- 이미지 채널간의 순서는 Red-Green-Blue 순서로 정의
  - ✓ $p(x_i|X_{<i}) = p(x_{i,R}|X_{<i})p(x_{i,G}|X_{<i}, x_{i,R})\ p(x_{i,B}|X_{<i}, x_{i,R}, x_{i,G})$



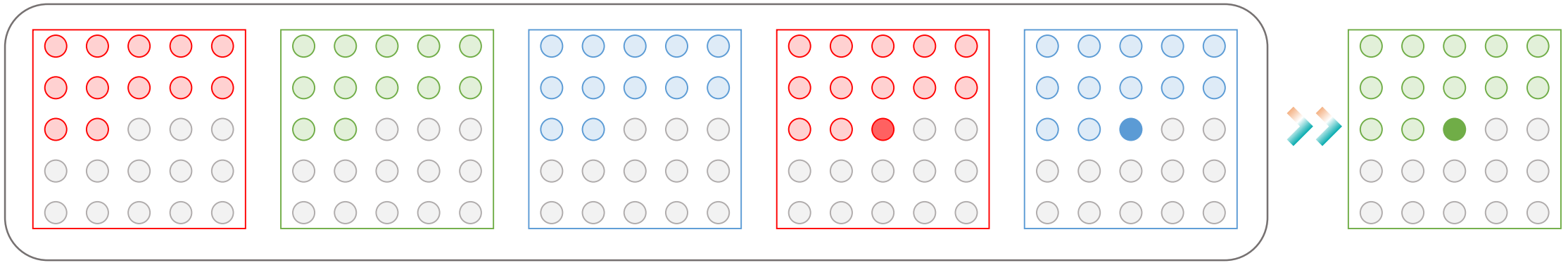
02 | Red-Green-Blue

- 이미지 채널간의 순서는 Red-Green-Blue 순서로 정의
  - ✓ $p(x_i|X_{<i}) = p(x_{i,R}|X_{<i})p(x_{i,G}|X_{<i}, x_{i,R})\ p(x_{i,B}|X_{<i}, x_{i,R}, x_{i,G})$



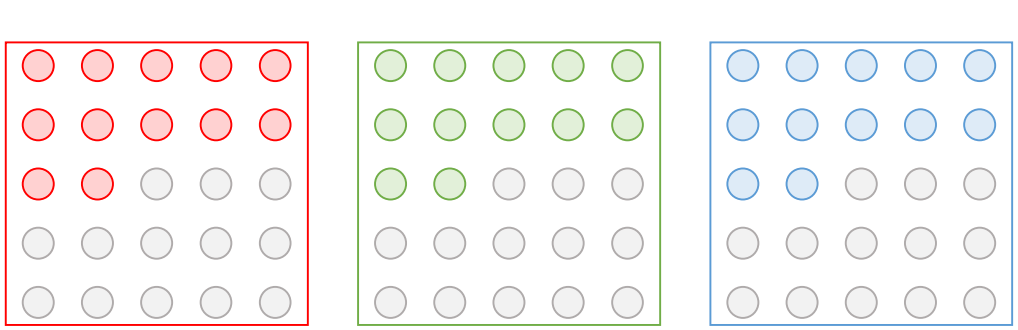
02 | Red-Green-Blue

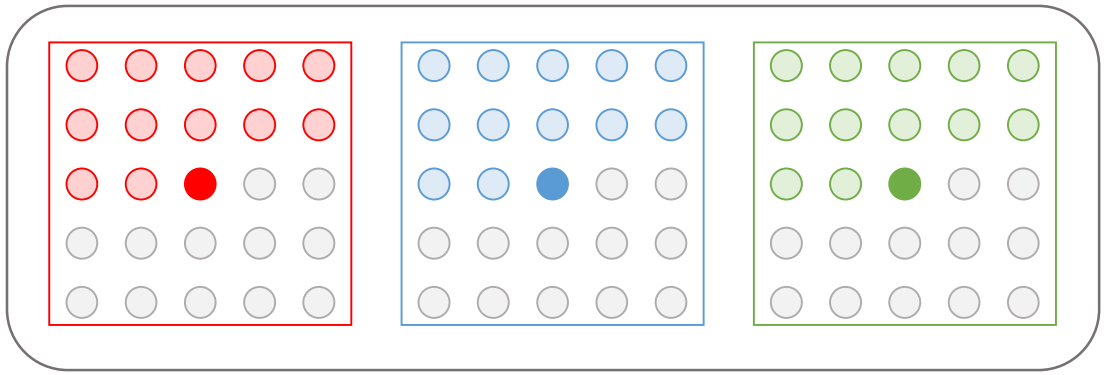
- 이미지 채널간의 순서는 Red-Green-Blue 순서로 정의
  - ✓ $p(x_i|X_{<i}) = p(x_{i,R}|X_{<i})p(x_{i,G}|X_{<i}, x_{i,R})\ p(x_{i,B}|X_{<i}, x_{i,R}, x_{i,G})$



02 | Red-Green-Blue

- 이미지 채널간의 순서는 Red-Green-Blue 순서로 정의
  - ✓ $p(x_i|X_{<i}) = p(x_{i,R}|X_{<i})p(x_{i,G}|X_{<i}, x_{i,R})\ p(x_{i,B}|X_{<i}, x_{i,R}, x_{i,G})$

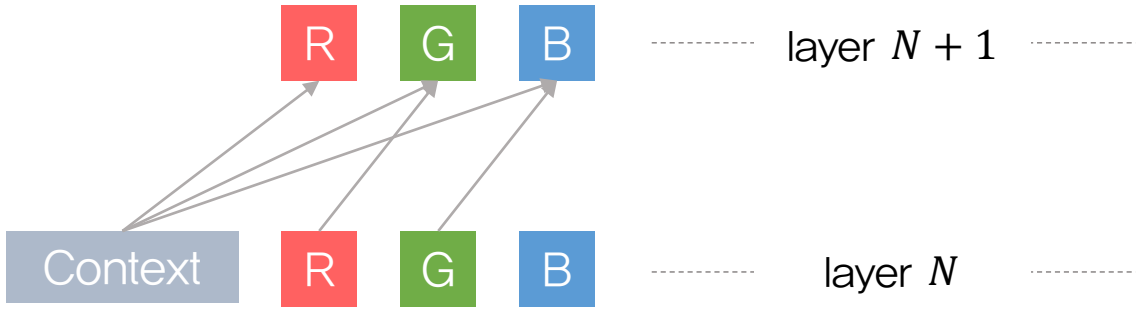




02 | Masking

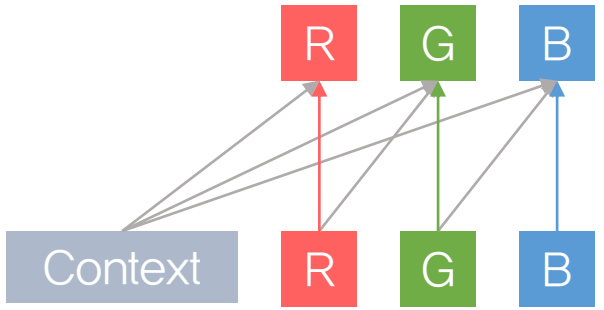
- 현재 픽셀을 생성하는데 있어서 현재를 포함한 미래의 정보를 사용하지 않아야 한다
- 어떻게 masking을 사용하는가에 따라서 모델링에 사용할 수 있는 정보의 범위가 달라짐

[Mask A]



- Context는 $p(x_i|X_{<i})$를 의미
- 현재 픽셀 정보값을 사용하지 않는 masking
  * channels are not connected to themselves
- Input 이미지에 직접적으로 사용하는 방법
- 처음에 현재 자기 자신의 정보를 사용해서 자기 자신을 infer하는 것은 불가능

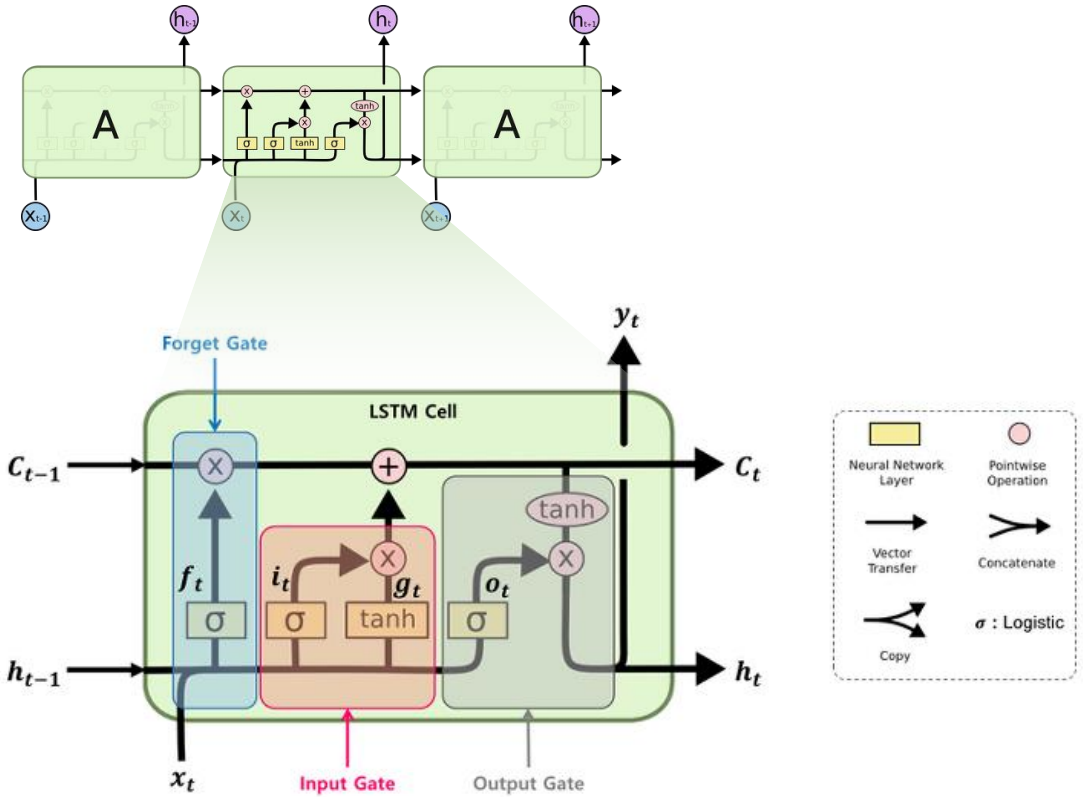
[Mask B]

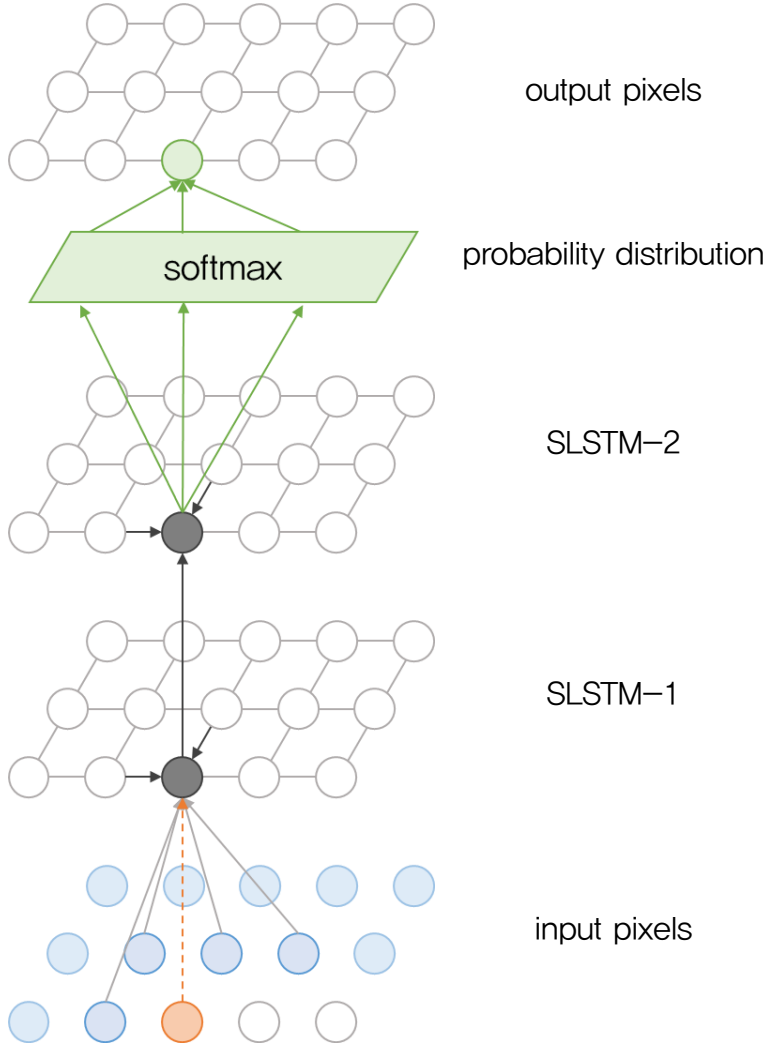


- 현재 픽셀 정보값을 사용하는 않는 masking
- Mask A를 사용한 이후에 사용
- Mask A를 통해 이미 이전 픽셀들간의 dependency를 제거했기 때문에 사용이 가능

02 | Mapping Architectures

- input-to-state & state-to-state mapping을 따로 사용
- LSTM과 같이 이전 정보를 모두 사용할 수 있도록 디자인
  - ✓ Generative Image Modeling Using Spatial LSTMs (2015)



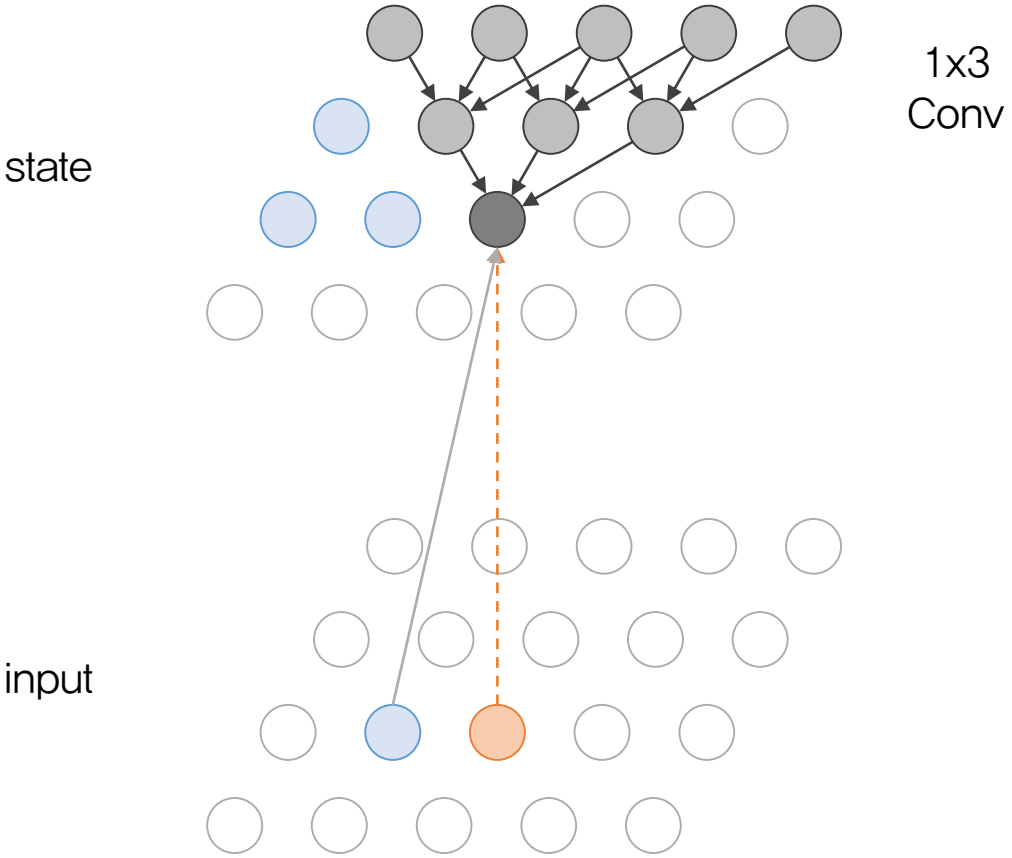


02 | Mapping Architectures

- Spatial LSTM과 같은 예전 방법들은, state-to-state 계산을 모두 수행
- Computational cost를 줄이기 위해 3가지 방법을 제안
  - ✓ Row LSTM, Diagonal BiLSTM, PixelCNN

Row LSTM

- Feature map에서 타겟 state보다 윗 행에 존재하는 state의 정보들만 반영하여 계산을 수행
- input – state – ouput은 모두 같은 크기 (zero padding을 사용)
- 계산량을 줄이기 위해, convolution을 사용

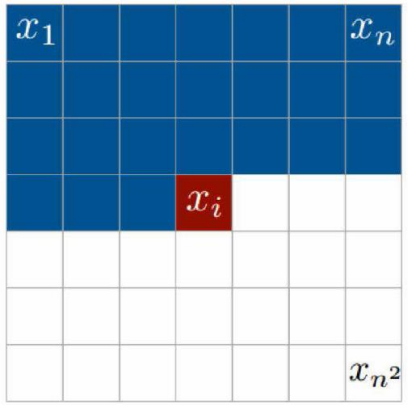


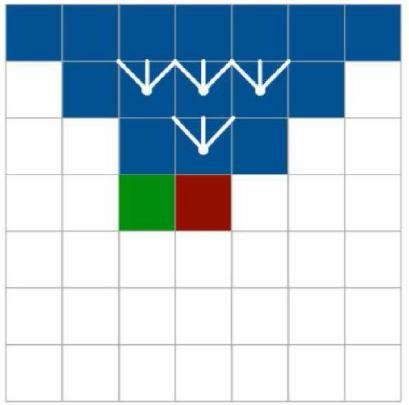
02 | Mapping Architectures

- Spatial LSTM과 같은 예전 방법들은, state-to-state 계산을 모두 수행
- Computational cost를 줄이기 위해 3가지 방법을 제안
  - ✓ Row LSTM, Diagonal BiLSTM, PixelCNN

Row LSTM

- Feature map에서 타겟 state보다 윗 행에 존재하는 state의 정보들만 반영하여 계산을 수행
- input – state – ouput은 모두 같은 크기 (zero padding을 사용)
- 계산량을 줄이기 위해, convolution을 사용





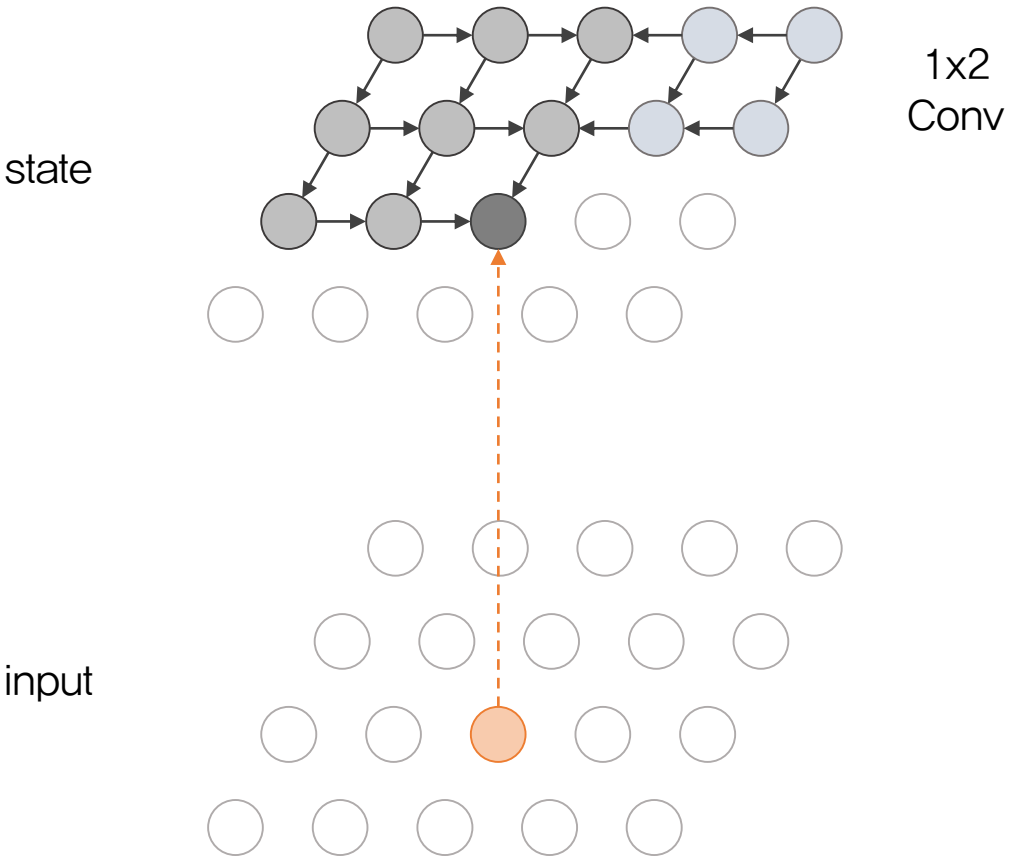
- Receptive field는 작아지지만, 계산의 효율성이 높아짐

02 | Mapping Architectures

- Spatial LSTM과 같은 예전 방법들은, state-to-state 계산을 모두 수행
- Computational cost를 줄이기 위해 3가지 방법을 제안
  - ✓ Row LSTM, Diagonal BiLSTM, PixelCNN

Diagonal BiLSTM

- 2개의 LSTM을 합쳐서 사용
- 오른쪽에 있는 영역을 효과적으로 반영
- Convolution을 사용해서 계산량을 감소시킴
- Row LSTM에 비해 많이 느림



02 | Mapping Architectures

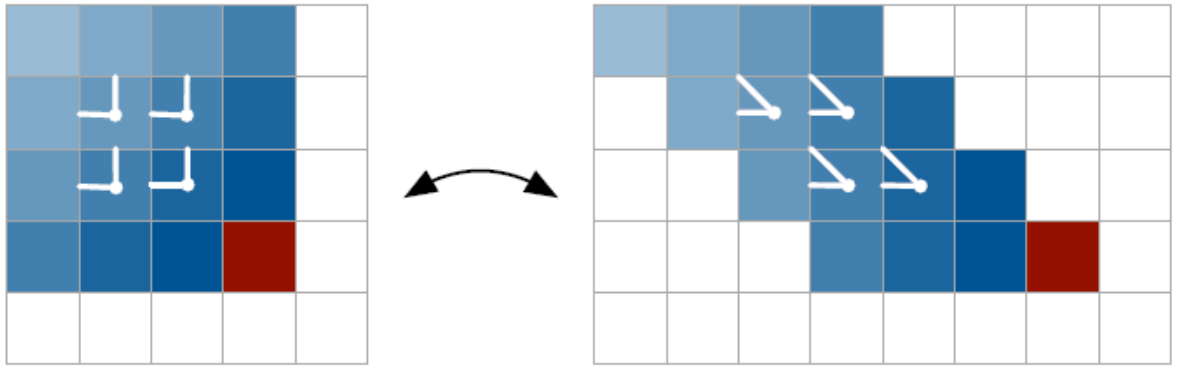
- Spatial LSTM과 같은 예전 방법들은, state-to-state 계산을 모두 수행
- Computational cost를 줄이기 위해 3가지 방법을 제안
  - ✓ Row LSTM, Diagonal BiLSTM, PixelCNN

Diagonal BiLSTM

- (Trick) 병렬적으로 계산을 수행하기 위해

  이미지를 한 칸씩 옆으로 밀어준다

  → parallelized by skew operation

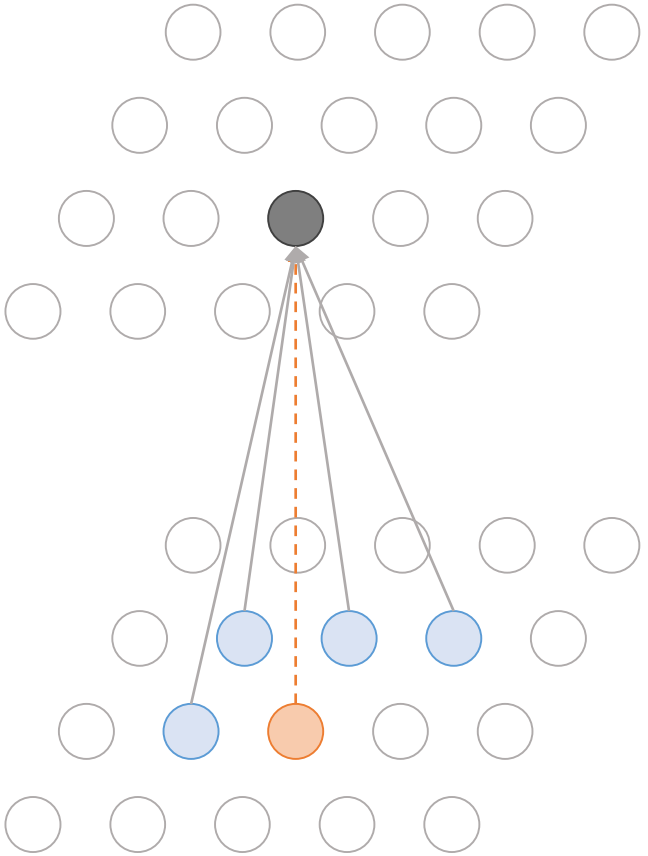


02 | Mapping Architectures

- Spatial LSTM과 같은 예전 방법들은, state-to-state 계산을 모두 수행
- Computational cost를 줄이기 위해 3가지 방법을 제안
  - ✓ Row LSTM, Diagonal BiLSTM, PixelCNN

PixelCNN

- LSTM의 연산 자체가 너무 느림
- Convolution 연산만을 사용해서 모델을 구축



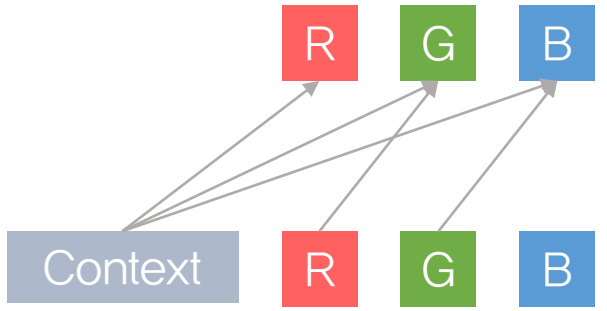
3x3
Masked
Conv

02 Mapping Architectures

- Spatial LSTM과 같은 예전 방법들은, state-to-state 계산을 모두 수행
- Computational cost를 줄이기 위해 3가지 방법을 제안
  - ✓ Row LSTM, Diagonal BiLSTM, PixelCNN

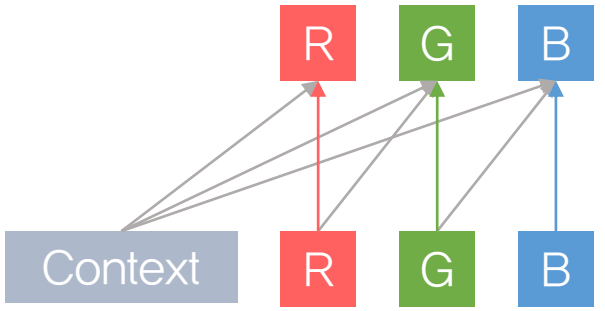
PixelCNN

- Masked Convolution을 사용
- 미래(오른쪽 아래) 정보를 반영하지 않기 위한 트릭
- Convolution 연산이 수행되기 전에 반영하지 않을 영역에 0을 곱해서 제외



w	w	w
w	0	0
0	0	0

[Mask A]



w	w	w
w	w	0
0	0	0

[Mask B]

02 | Mapping Architectures

- 계산량과 성능 사이의 trade-off가 존재
- 추가적으로, residual connections를 사용해서 많은 수의 레이어를 쌓음

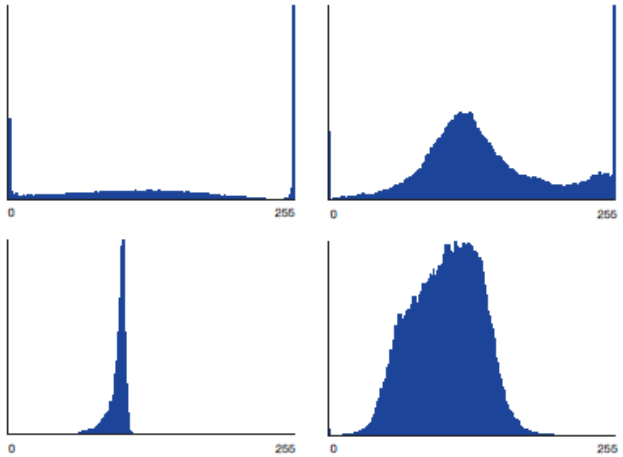
항목	PixelRNN		PixlCNN
	Row LSTM	Diagonal BiLSTM	
Receptive Field	Triangular	Full Dependency	Full Dependency
Computation	Slow	Slowest	Fastest
Log-Likelihood	Good	Best	Worst

02 | Discrete Softmax Distribution

- 이미지의 각 픽셀은 0~255에 해당하는 정수값을 갖음
- Autoregressive 모델이므로, 매 step마다 새로운 input은 정수값으로 들어가야 함
- 수치 예측을 수행하는 대신, 256개의 클래스를 갖는 softmax layer를 사용

- 클래스에 order, prior 등을 사전에 부여하지 않음
- Softmax distribution을 살펴보았을 때,

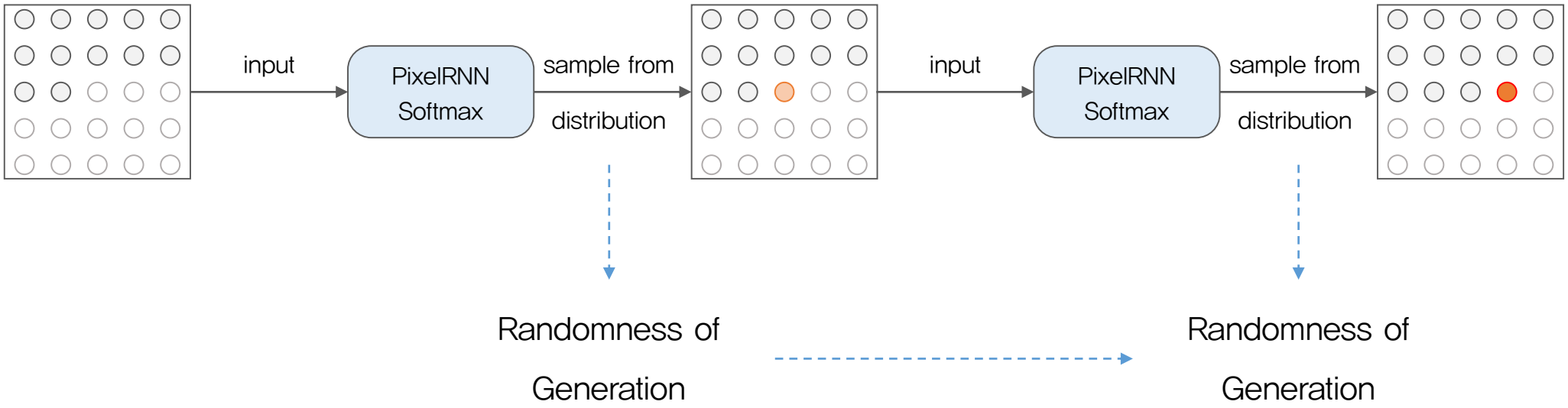
  인접한 클래스들끼리 유사한 확률값을 갖음
- 계산상으로도 이점이 있음
- Output을 0~255 사이값으로 강제하지 않아도 됨



Softmax Distributions

02 Discrete Softmax Distribution

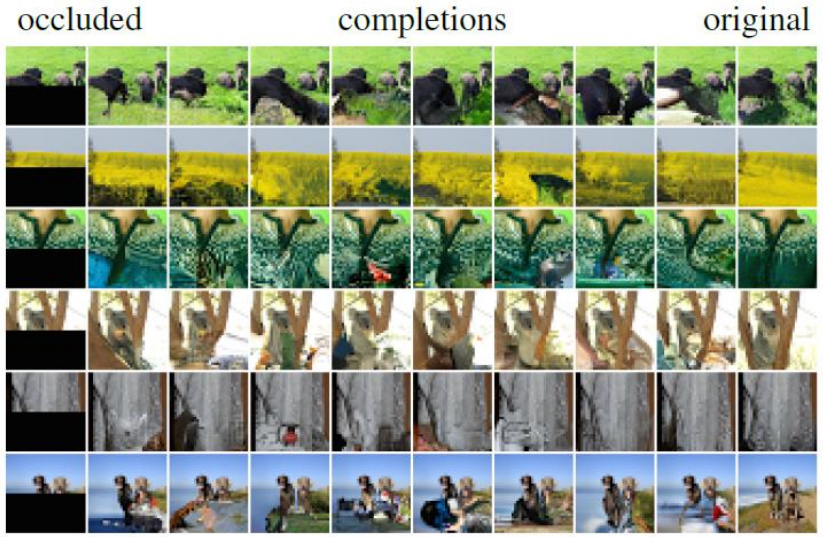
- Inference, 데이터를 생성하는 과정에서 있어서 매 step별로 softmax layer를 지나게 됨
- Softmax layer를 지날때 마다, argmax를 취하지 않고 분포에서 랜덤하게 샘플링을 수행
- 다양한 데이터가 생성될 수 있는 randomness가 발생



02 | Experiments

- 당시 연구되었던 image generation 모델에 비해 매우 좋은 성능을 보임
- 평가 방법: Negative Log-Likelihood
- Image Completion와 같은 작업에도 사용할 수 있음

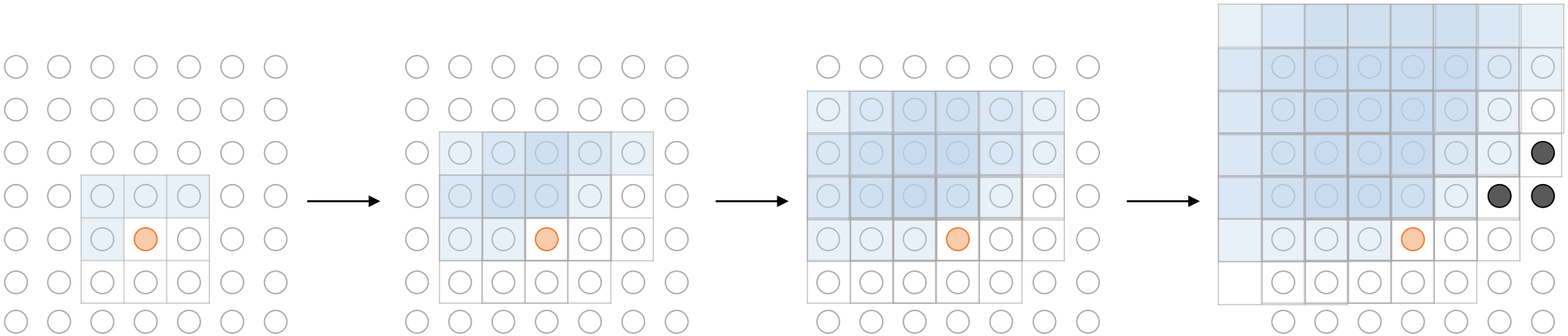
Model	NLL Test (Train)
Uniform Distribution:	8.00
Multivariate Gaussian:	4.70
NICE [1]:	4.48
Deep Diffusion [2]:	4.20
Deep GMMs [3]:	4.00
RIDE [4]:	3.47
PixelCNN:	3.14 (3.08)
Row LSTM:	3.07 (3.00)
Diagonal BiLSTM:	**3.00** (2.93)



Contents

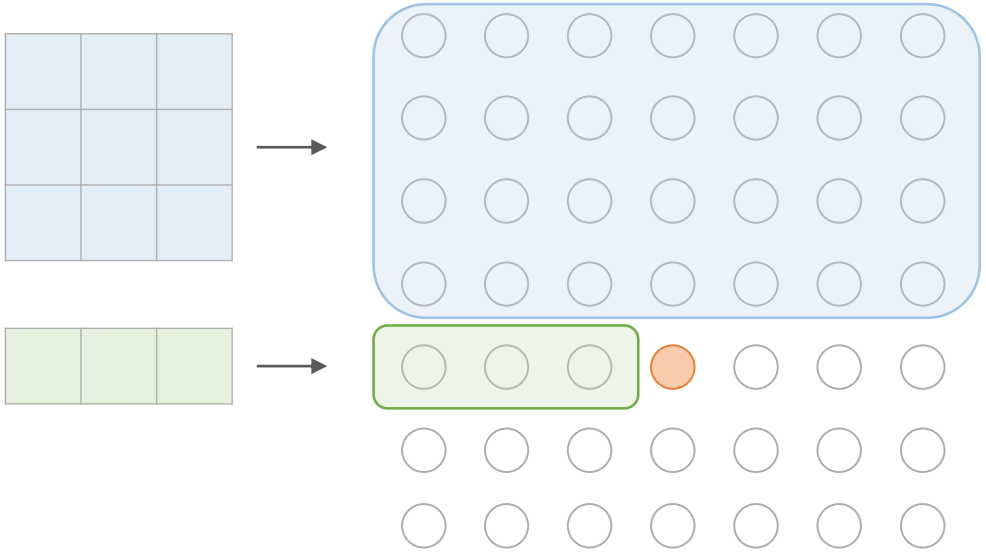
03 | Gated PixelCNN

- PixelCNN – 계산속도는 빠르지만, 성능이 PixelRNN에 비해서 낮음
  - ✓ 이미지 생성 자체는 픽셀 단위에서 sequential하게 진행
- PixelCNN의 receptive field에는 blind spot이 존재



03 | Gated PixelCNN

- Horizontal & Vertical Stack
  - ✓ Vertical – 현재 픽셀 위에 있는 모든 행에 대한 정보를 가져옴 & Horizontal Stack에 추가적인 정보로 다시 들어감
  - ✓ Horizontal – 현재 행에 대한 정보를 가져옴
  - ✓ Blind Spot을 없애고 PixelRNN과 같이 이전 모든 픽셀의 정보를 가져올 수 있음



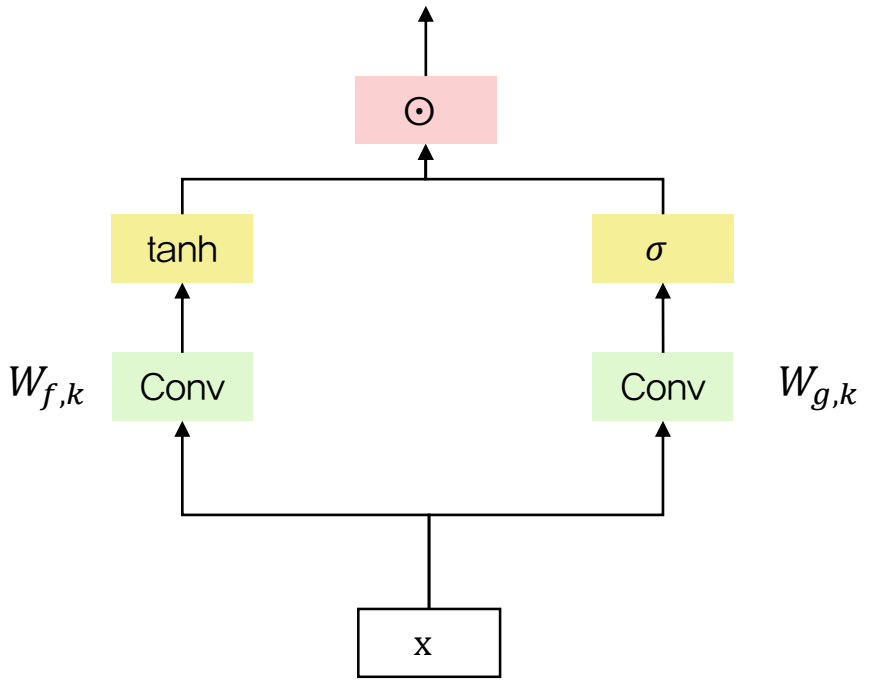
03 Gated PixelCNN

- Gated Activation
  - ✓ PixelRNN의 LSTM cell 안에 사용되는 multiplicative units → complex interaction을 모델에 반영할 수 있음
  - ✓ PixelCNN에도 Gated Activation을 사용하도록 디자인

Gated Activation

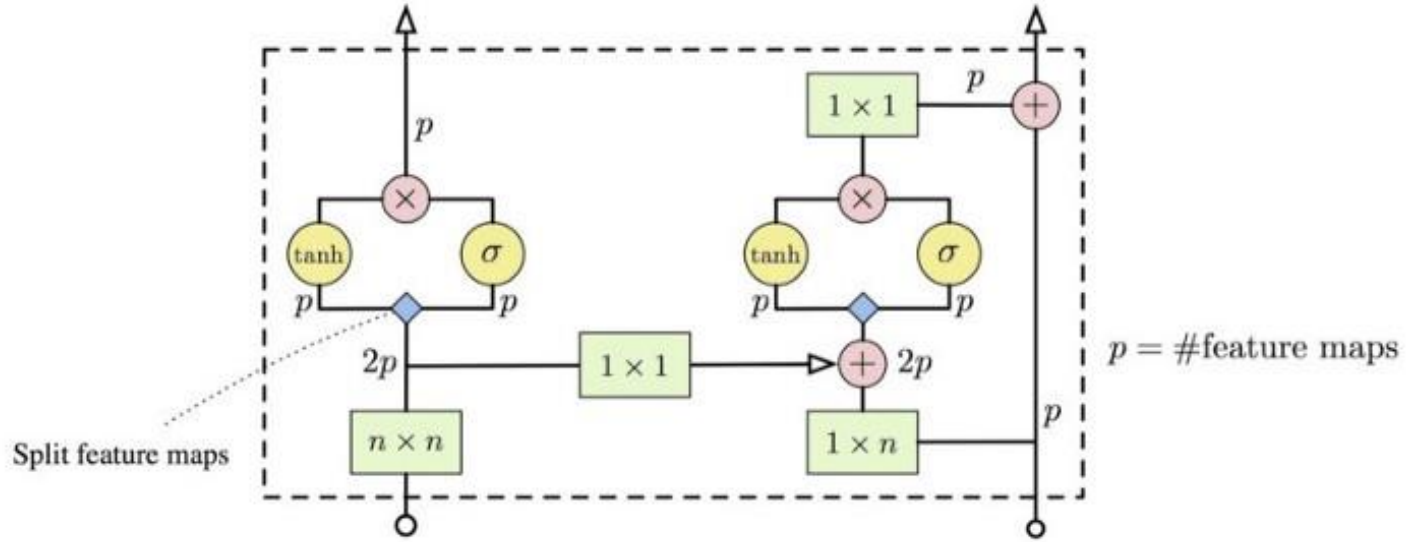
$$z = \tanh(W_{f,k} * \mathrm{x}) \odot \sigma(W_{g,k} * \mathrm{x})$$

- $*$: convolution operator
- $\odot$: element-wise multiplication operator
- k: layer index
- f, g: filter, gate → W is learnable convolution filter



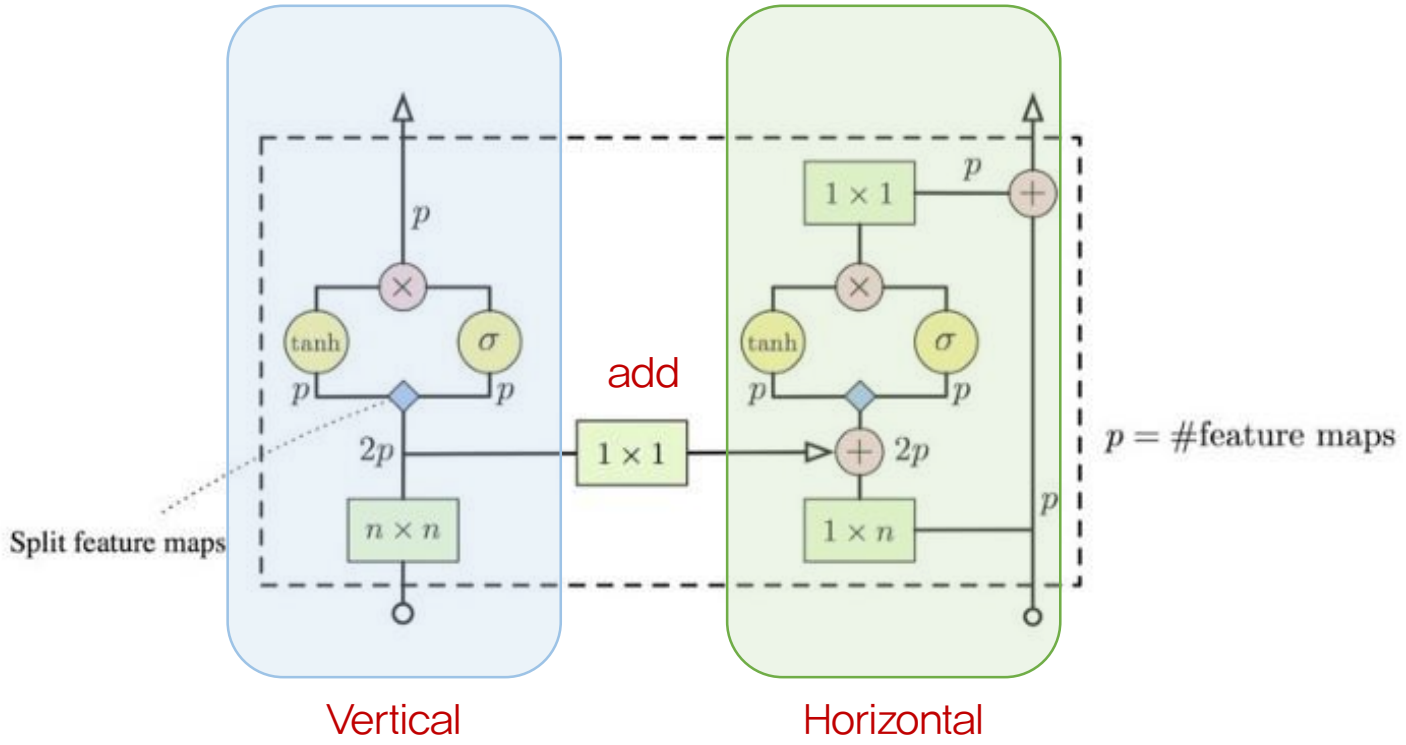
03 | Gated PixelCNN

- Gated Activation
  - ✓ PixelRNN의 LSTM cell 안에 사용되는 multiplicative units → complex interaction을 모델에 반영할 수 있음
  - ✓ PixelCNN에도 Gated Activation을 사용하도록 디자인



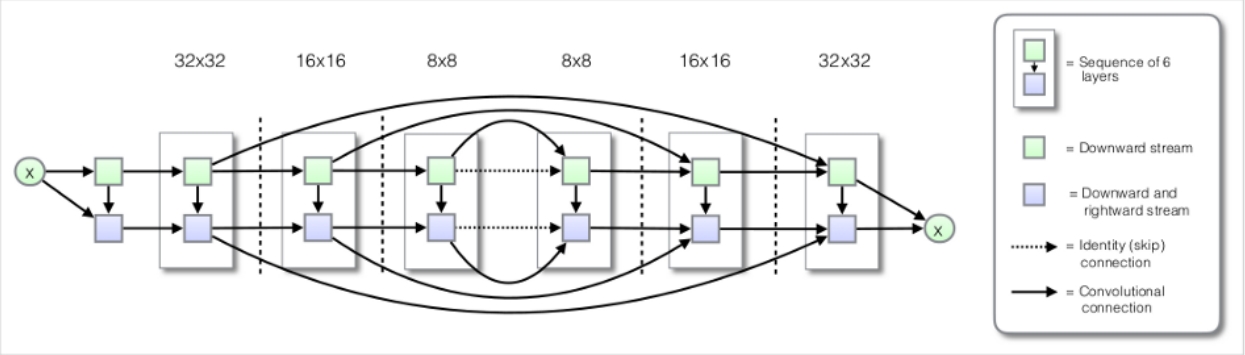
03 Gated PixelCNN

- Gated Activation
  - ✓ PixelRNN의 LSTM cell 안에 사용되는 multiplicative units → complex interaction을 모델에 반영할 수 있음
  - ✓ PixelCNN에도 Gated Activation을 사용하도록 디자인



03 | PixelCNN++

- PixelCNN에 다양한 테크닉을 적용시켜서, 모델의 성능을 향상시킴
  - ✓ Discretized logistic mixture likelihood
    - Softmax layer 대신 logistic distributions의 합으로 이루어진 latent intensity variable을 사용 (Kingma)
    - memory efficient
  - ✓ Conditioning on whole pixels
    - PixelCNN은 RGB를 따로따로 고려했으나, 실제로 color channel 사이의 관계는 그리 복잡하지 않음
    - Color channel 별로 output을 산출하지 않고, 채널의 joint distribution로 산출하도록 변경
  - ✓ Downsampling
    - Long range dependency에 취약한 부분을 보완하기 위해 stride를 적용
  - ✓ Short-cut connections
    - Encoder-Decoder 구조를 사용, symmetric 연결
  - ✓ Dropout
    - Regularization



Contents

04 | WaveNet

- Paper

[PDF] **WaveNet**: A generative model for raw audio.
A Van Den Oord, S Dieleman, H Zen, K Simonyan... - SSW, 2016 - regmedia.co.uk
This paper introduces **WaveNet**, a deep neural network for generating raw audio waveforms. The model is fully probabilistic and autoregressive, with the predictive distribution for each audio sample conditioned on all previous ones; nonetheless we show that it can be ...
☆ 99 915회 인용 관련 학술자료 전체 5개의 버전 »

WAVENET: A GENERATIVE MODEL FOR RAW AUDIO

Aäron van den Oord **Sander Dieleman** **Heiga Zen**†

Karen Simonyan **Oriol Vinyals** **Alex Graves**

Nal Kalchbrenner **Andrew Senior** **Koray Kavukcuoglu**

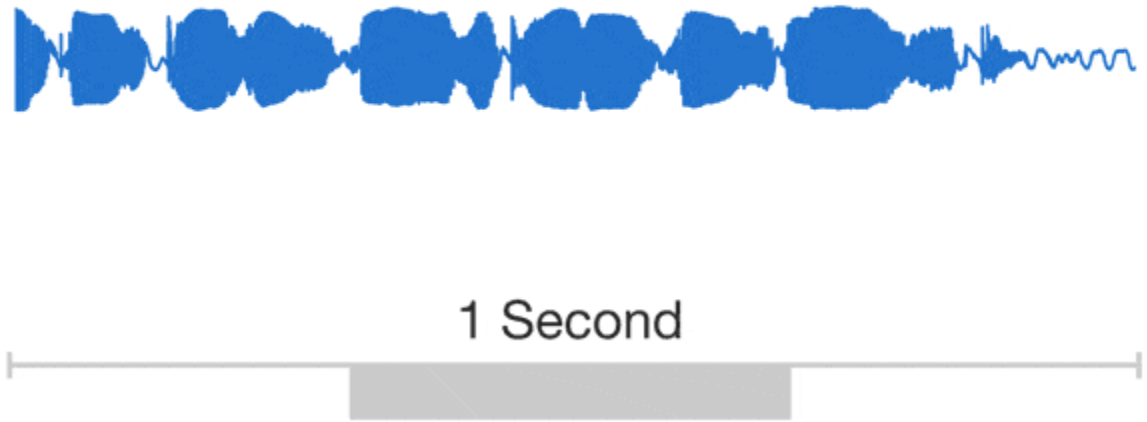
Google DeepMind, London, UK
† Google, London, UK

ABSTRACT

This paper introduces WaveNet, a deep neural network for generating raw audio waveforms. The model is fully probabilistic and autoregressive, with the predictive distribution for each audio sample conditioned on all previous ones; nonetheless we show that it can be efficiently trained on data with tens of thousands of samples per second of audio. When applied to text-to-speech, it yields state-of-the-art performance, with human listeners rating it as significantly more natural sounding than the best parametric and concatenative systems for both English and Chinese. A single WaveNet can capture the characteristics of many different speakers with equal fidelity, and can switch between them by conditioning on the speaker identity. When trained to model music, we find that it generates novel and often highly realistic musical fragments. We also show that it can be employed as a discriminative model, returning promising results for phoneme recognition.

04 | Audio Data

- Raw audio – 시간에 따라 자잘한 변동이 너무 심하다 (ticks)
  - ✓ 일반적으로 1초에 16,000개의 샘플이 추출됨 (16kHz)
  - ✓ 이전 모든 데이터에 대해 dependency가 존재
- 2차원 이미지 데이터에 적용한 PixelRNN, PixelCNN을 1차원 오디오 데이터에 적용하고자 하는 시도

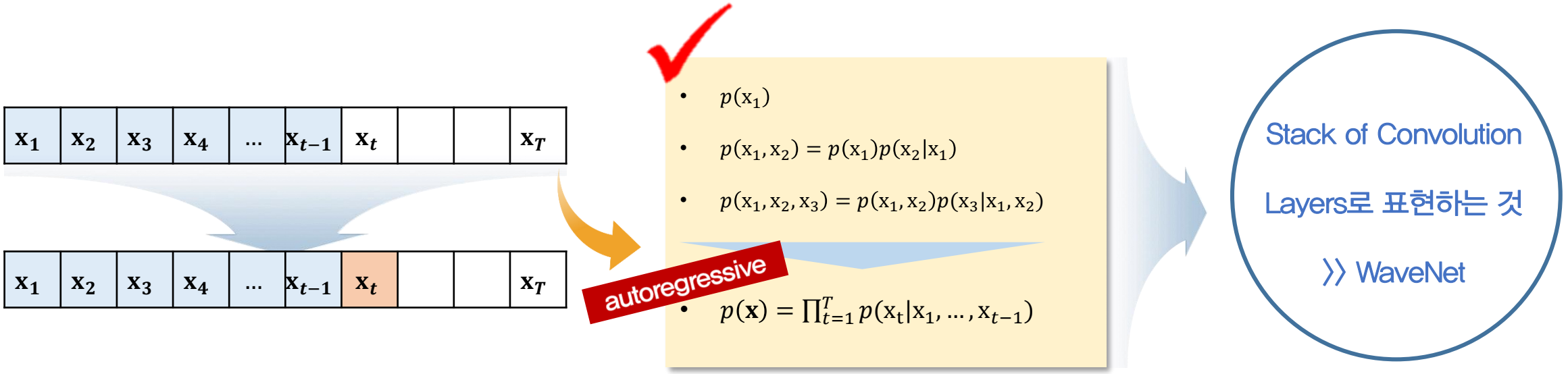


04 | Key Ideas

- Dilated causal convolutions
  - ✓ Long-term dependecy
- Companding transformation
  - ✓ From regression to classification
- Gated convolution + residual + skip-connections
  - ✓ Powerful non-linearity

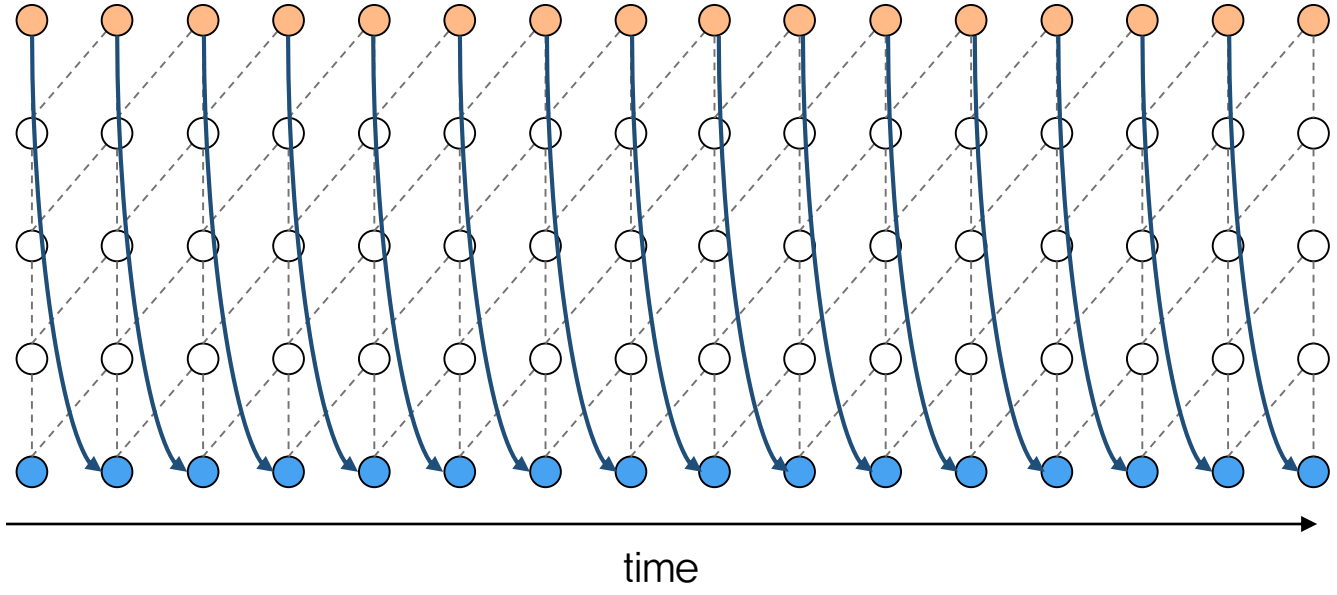
04 | Causal Convolutions

- Remember chain rule
- Joint probability → product of conditional probabilities → stack of convolution layers



04 | Causal Convolutions

- Autoregressive model with convolution

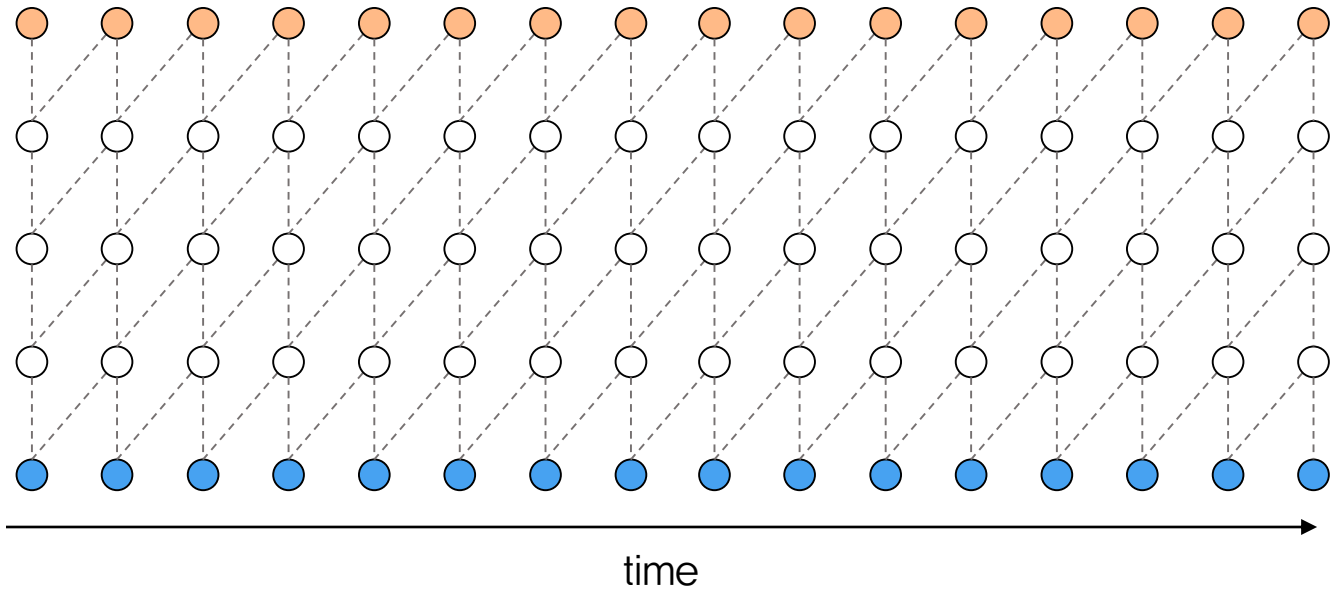


- Autoregressive model:
  $t-1$ 시점에서 얻은 output은 t 시점의 input으로 사용이 된다
- Input, hidden, output size = 16
  →pooling layer를 사용하지 않고,
  zero-padding을 통해 size를 유지함

- 왜 recurrent 대신 convolution을 사용하는가?
  → 예를 들어, 16kHz에서 100ms는 1,600 time steps이다. RNN, LSTM을 사용하여 특징을 학습하기에는 너무 길다.

04 Causal Convolutions

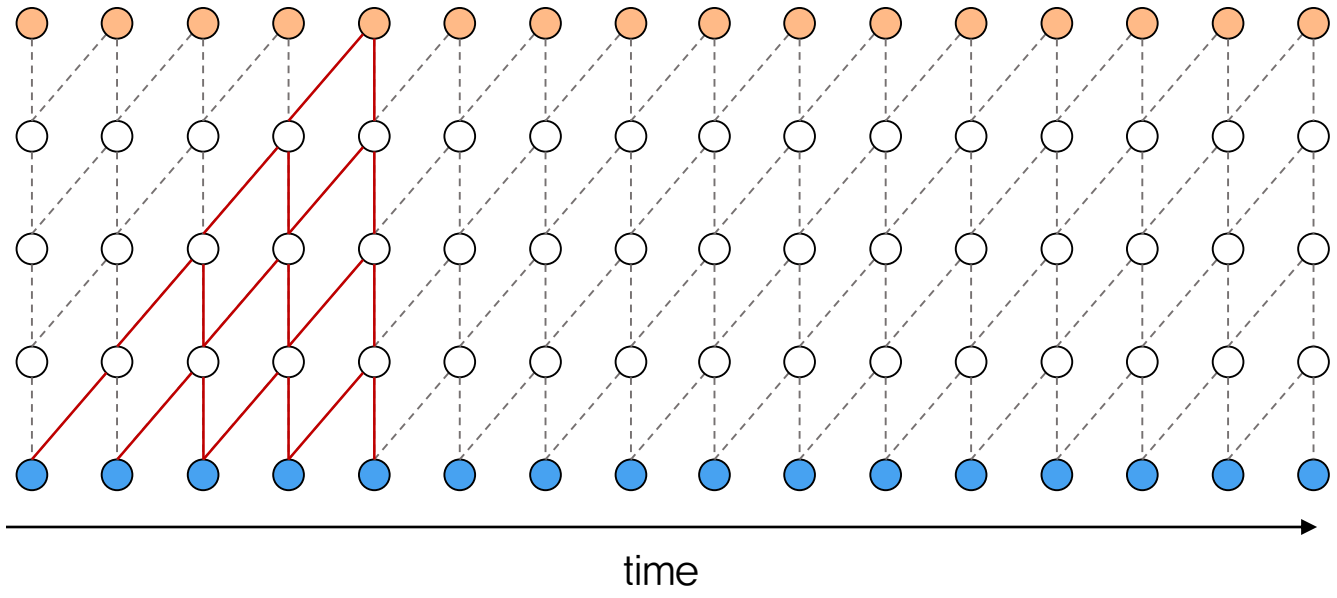
- Causal convolution layers



- # hidden layers = 3
- filter size = (2, 1)

04 | Causal Convolutions

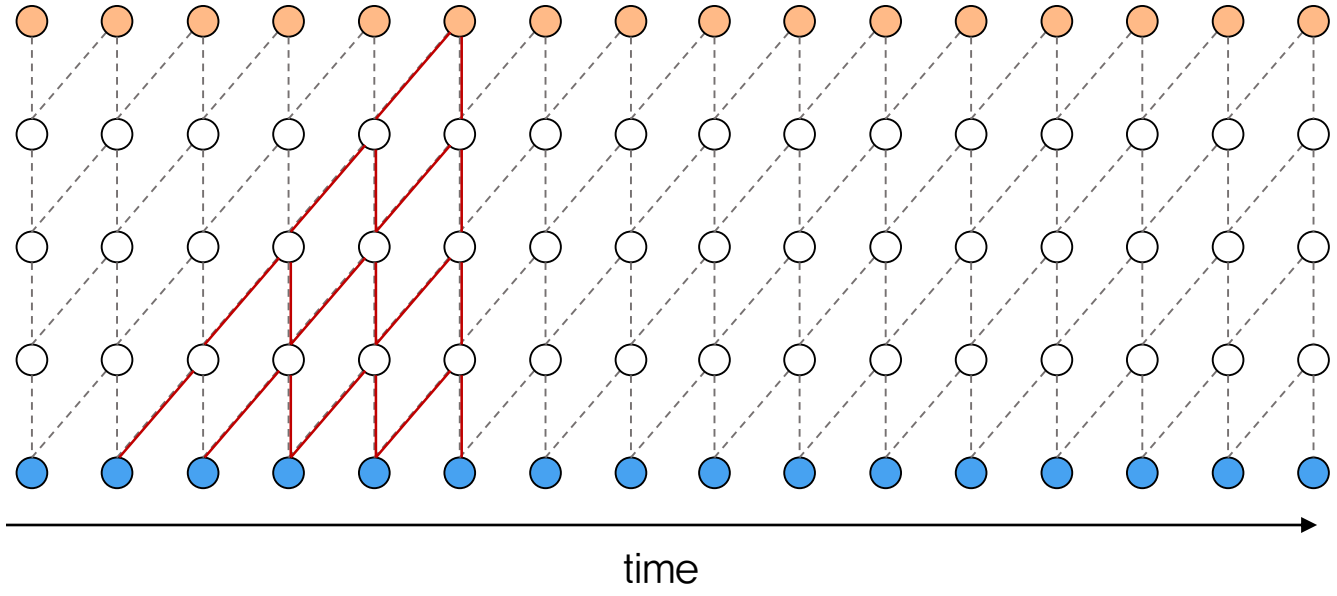
- Causal convolution layers



- 일반적인 convolution을 학습할 때,
  각 filter는 모든 input space를 지나간다

04 | Causal Convolutions

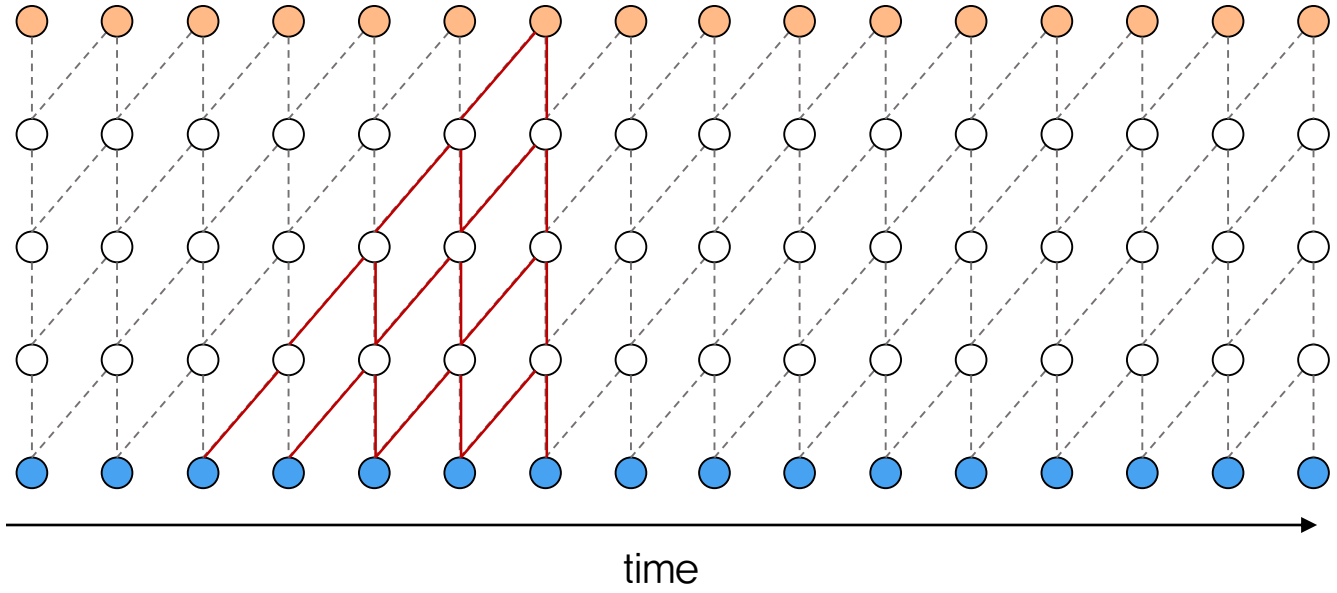
- Causal convolution layers



- 일반적인 convolution을 학습할 때,
  각 filter는 모든 input space를 지나간다

04 | Causal Convolutions

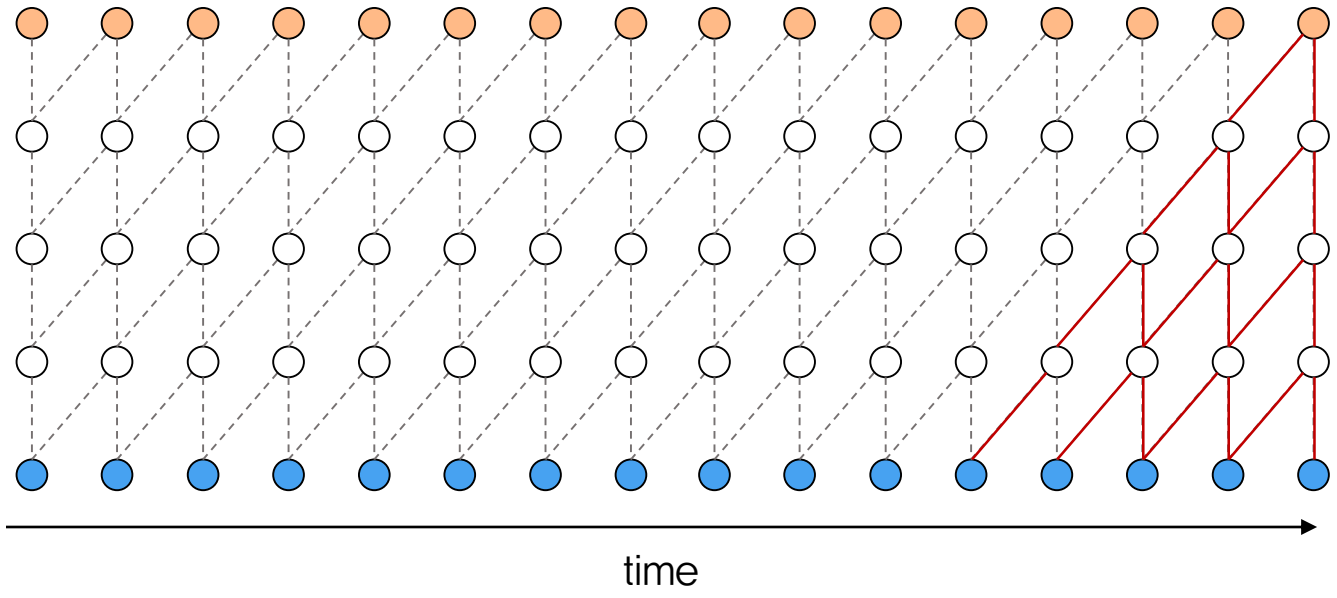
- Causal convolution layers



- 일반적인 convolution을 학습할 때,
  각 filter는 모든 input space를 지나간다

04 | Causal Convolutions

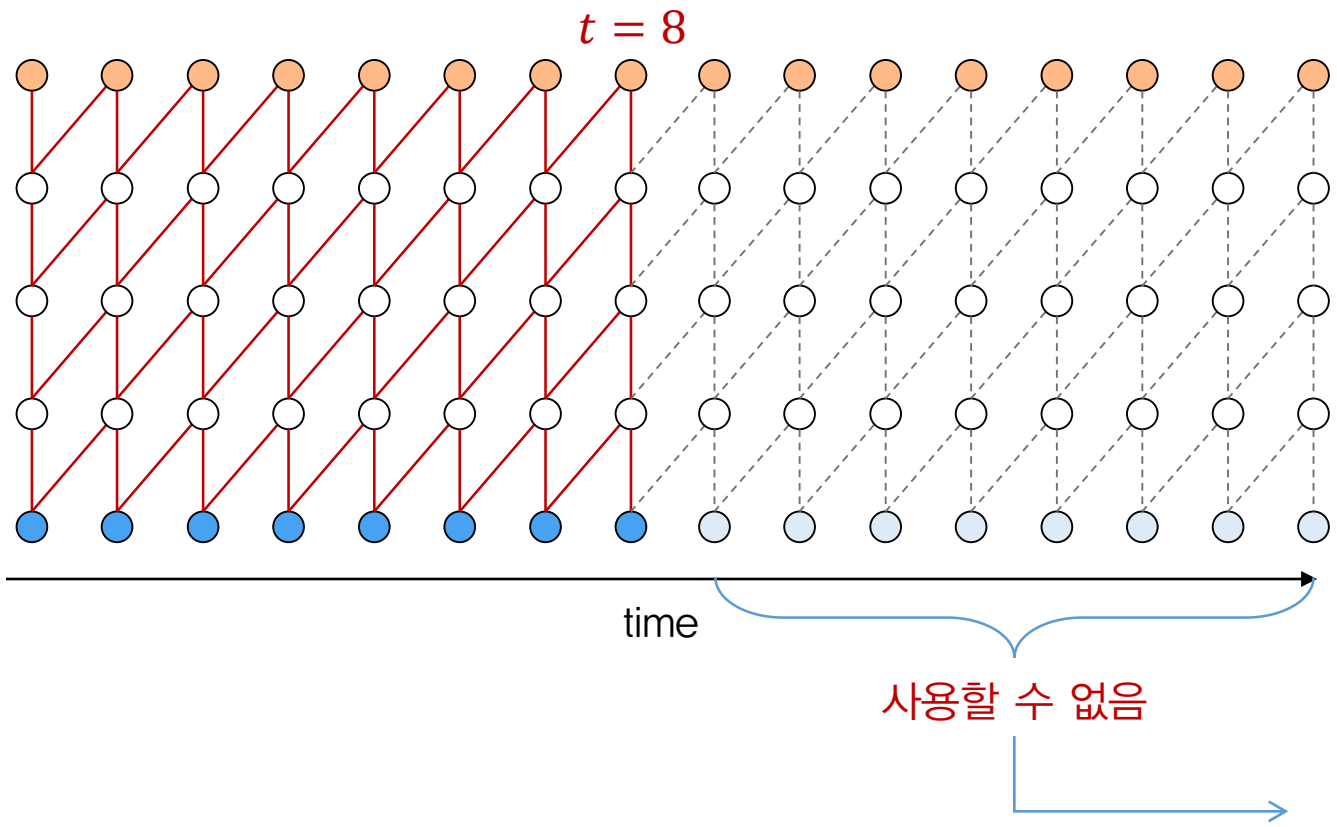
- Causal convolution layers



- 일반적인 convolution을 학습할 때,
  각 filter는 모든 input space를 지나간다

04 | Causal Convolutions

- Causal convolution layers

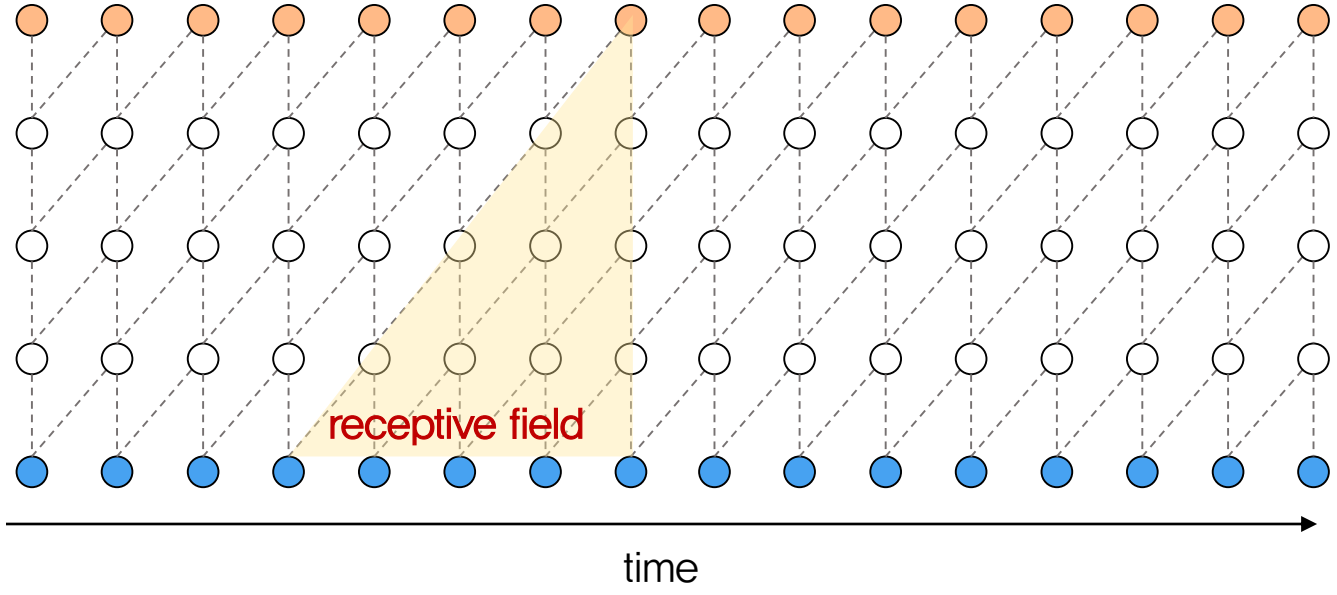


- 일반적인 convolution을 학습할 때,
  각 filter는 모든 input space를 지나간다
- 하지만, 미래의 값을 알 수 없기 때문에
  t 시점 이후의 값들은 사용할 수 없다
  (t 시점의 input은 $t-1$ 시점의 output)

time-based masking: causal filter

04 | Dilated Causal Convolutions

- Dilated → increase receptive field



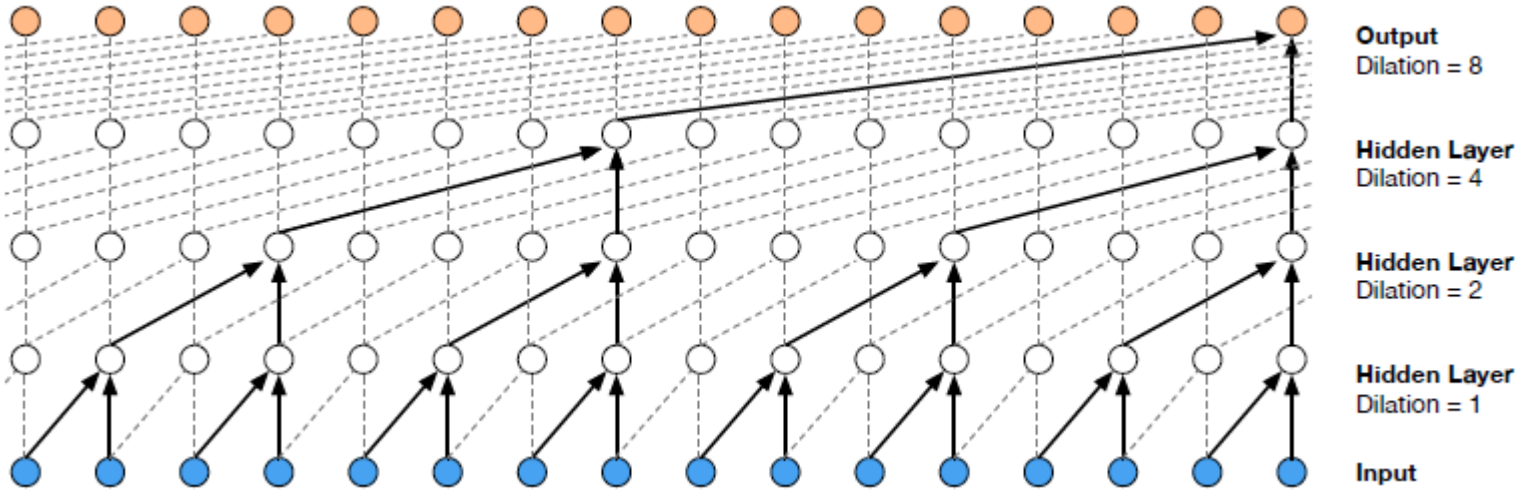
- 문제점: 작은 receptive field
  (참조하게 되는 input space의 크기)
- RNN은 이전 모든 시점들의 값을 고려하기 때문에, 이와 유사한 효과를 가지게 아키텍쳐를 구성하는 것이 필요함

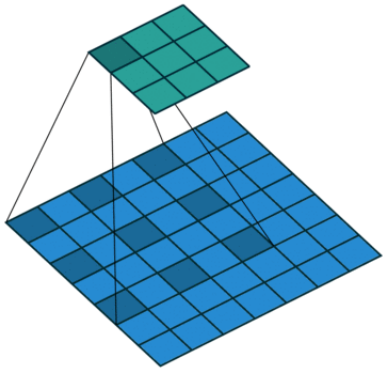
Convolution의 receptive field를 증가시키기 위해서는
→ layer를 많이 쌓거나, filter size를 키워야 함

Computational cost가 매우 높아 비효율적임

04 | Dilated Causal Convolutions

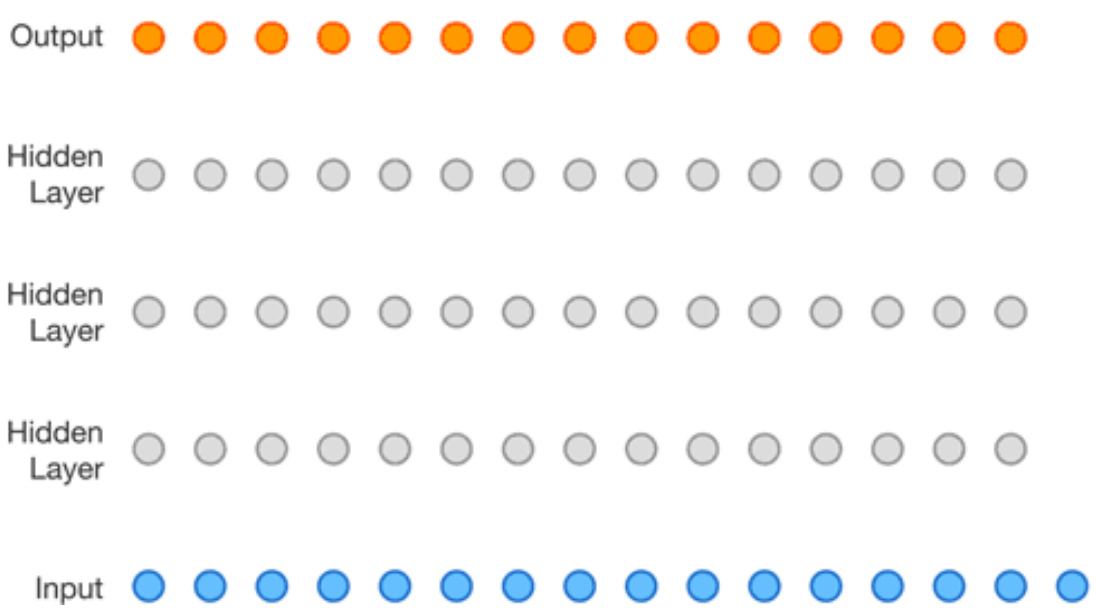
- Dilated → increase receptive field
  - ✓ Dilated convolution을 이용해, 적은 계산량으로 큰 receptive field를 얻음
  - ✓ Dilation을 1, 2, 4, ⋯, 512, 1, 2, 4, ⋯, 512, 1, 2, 4, ⋯, 512 총 30개의 layer를 구축: receptive field = 1024 (1x1024 convolution보다 훨씬 효율적)

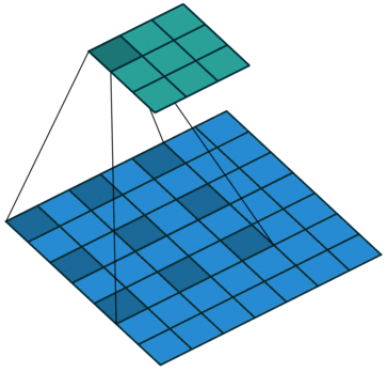




04 | Dilated Causal Convolutions

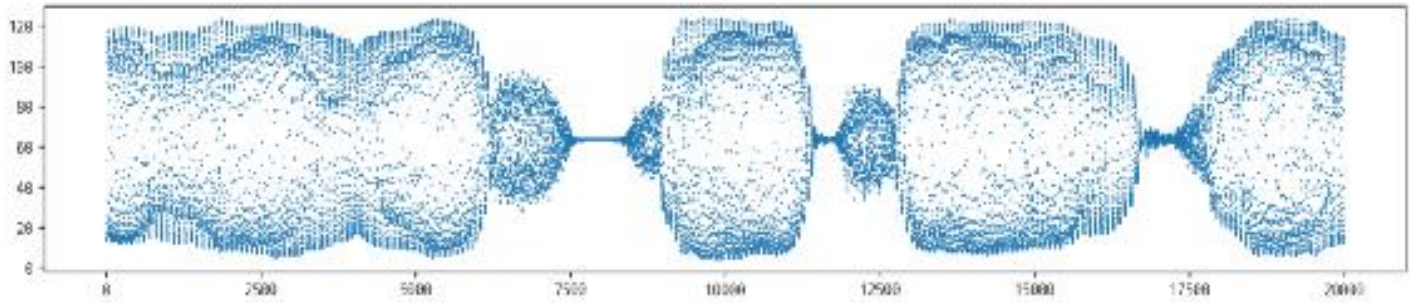
- 예시





04 | Companding Transformation

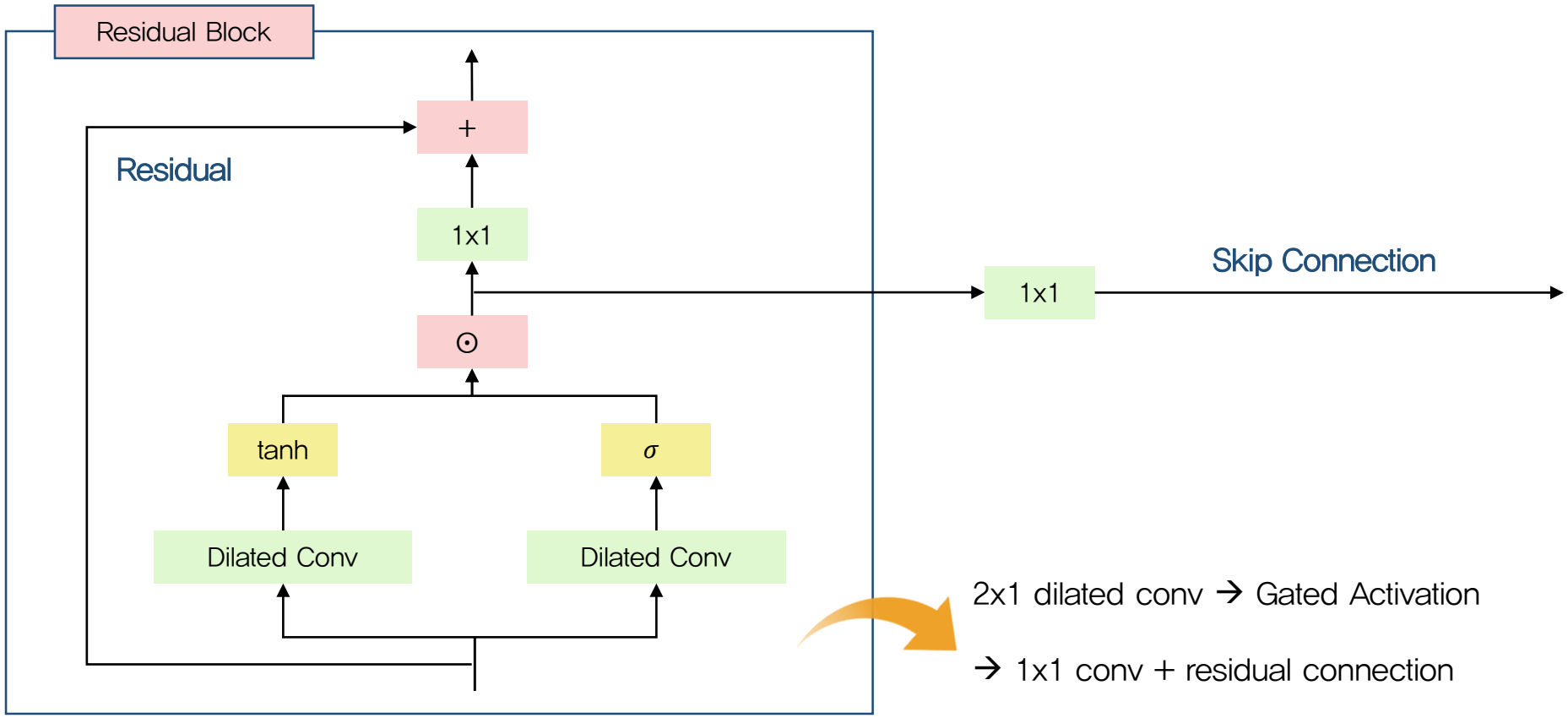
- PixelRNN, PixelCNN과 유사하게, 각 amplitude 값을 하나의 class로 가정하여 softmax distribution을 사용 (discrete)
  - ✓ 일반적으로 audio의 distribution을 표현하기 위해, Gaussian Mixture Model 등을 사용 (continuous)
  - ✓ 음성 데이터를 256개 클래스로 묶어 regression 문제를 multi-class classification 문제로 변환 ($\mu - \text{law}$)



$$f(x_t) = sign(x_t)\frac{\ln(1+\mu|x_t|)}{\ln(1+\mu)} \quad \text{, where } -1 < x_t < 1, \mu = 256$$

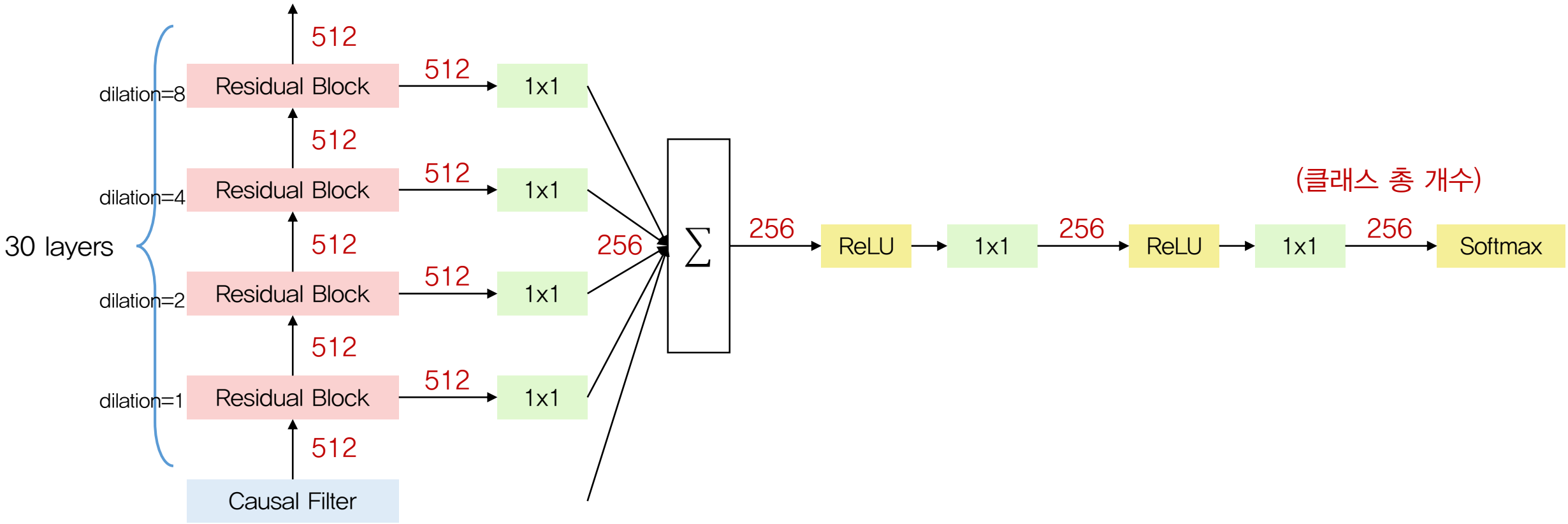
04 | Gated Convolution + Residual + Skip-Connections

- 모델을 깊게 쌓아서 non-linearity를 충분히 반영할 수 있도록 구축
  - ✓ 1x1 convolution을 통해 skip-connections도 학습을 할 수 있도록 설계



04 | Gated Convolution + Residual + Skip-Connections

- WaveNet을 펼쳤을 때 단일 timestep에서의 구조



05 | Conclusions

- Sequential한 구조를 반영하여 데이터의 분포를 학습하려는 시도
  - ✓ 이미지 – PixelRNN, PixelCNN
  - ✓ 오디오 – WaveNet

- Key – 데이터간 존재하는 dependency를 연결하는 방법

- 그 이후의 모델들…
  - ✓ Many GANs
  - ✓ Parallel WaveNet – 오디오 생성 속도를 증가시킨 모델
  - ✓ Tacotron – Encoder-Decoder-Attention End-to-End for Text-to-Speech
  - ✓ Parrotron – End-to-End model for Speech-to-Speech conversion
  - ✓ Translatotron – End-to-End model for Speech-to-Speech translation
  - ✓ Attention is All You Need


Thank
you