Convolutional Neural Networks in Speech

Lecture 20

CS 753

Instructor: Preethi Jyothi

Convolutional Neural Networks (CNNs)

- Fully connected (dense) layers have no awareness of spatial information
- Key concept behind convolutional layers is that of ***kernels*** or ***filters***
- Filters slide across an input space to detect spatial patterns (translation invariance) in local regions (locality)

Fully Connected Layers

32x32x3 image -> stretch to 3072 x 1

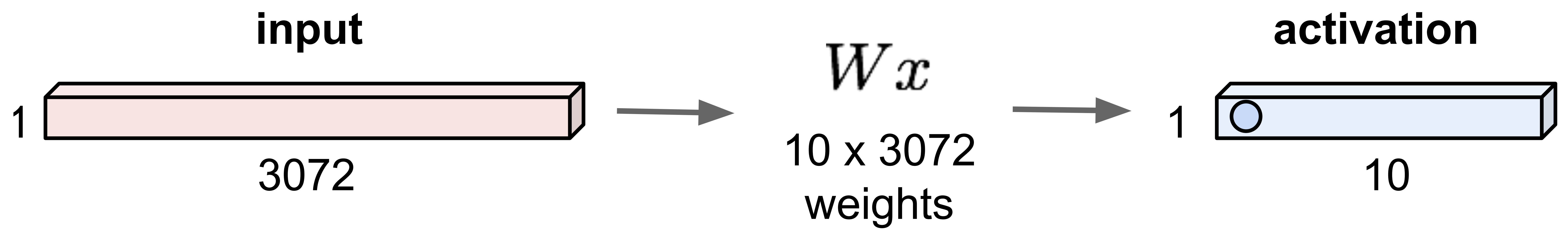
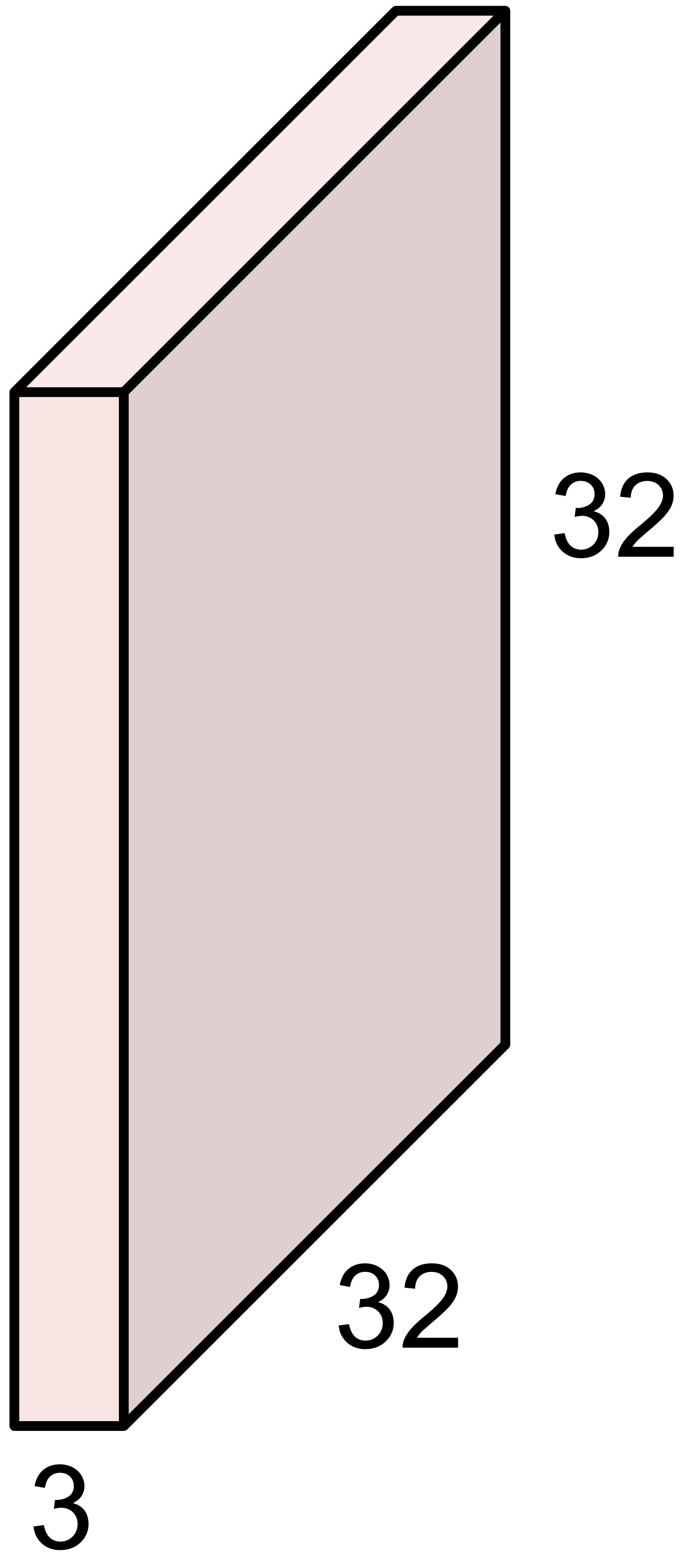


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolution Layer

32x32x3 image



5x5x3 filter

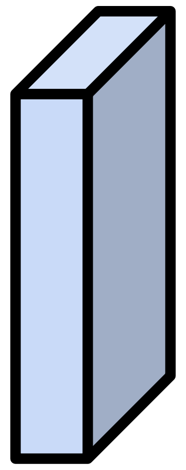


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolution Layer

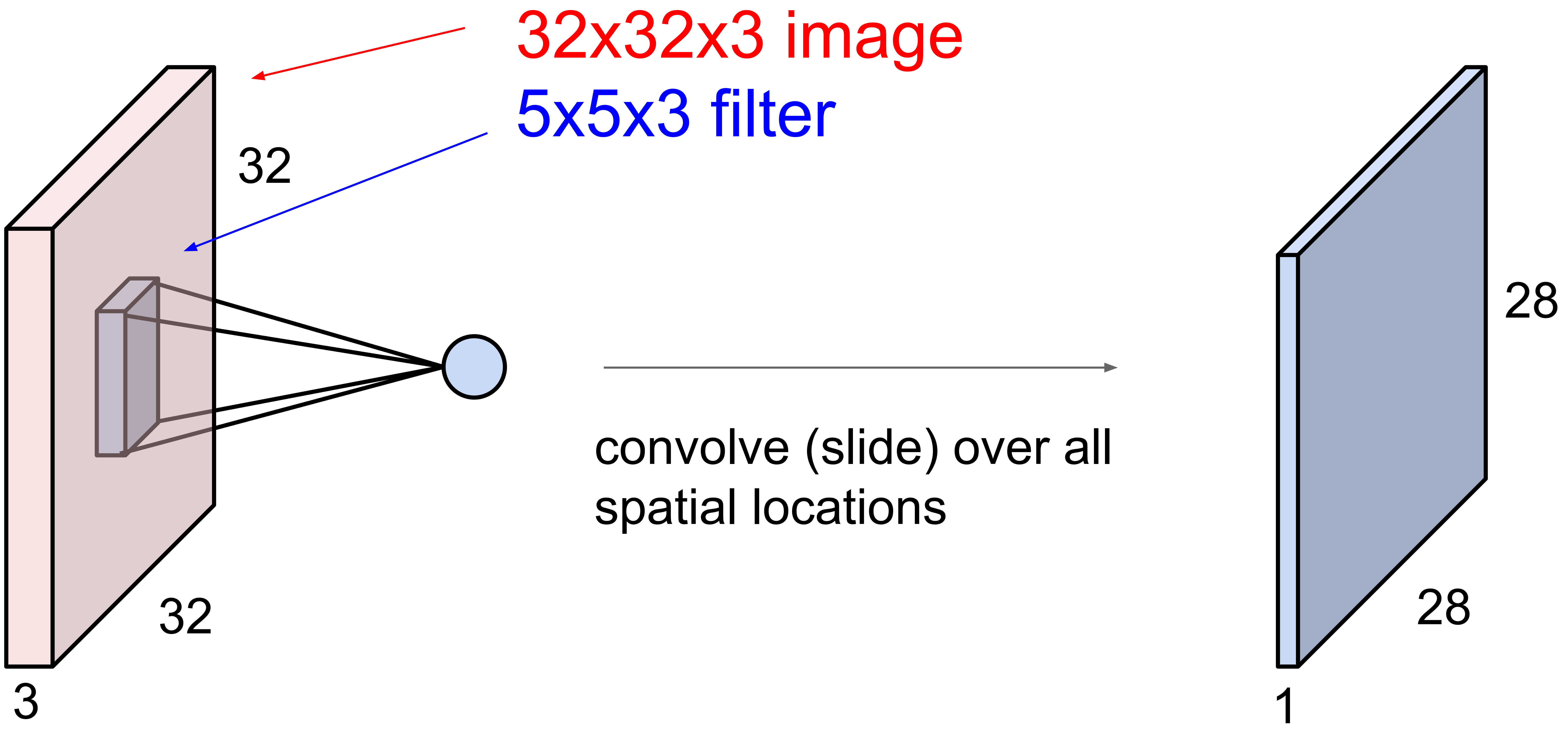


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolution Layer

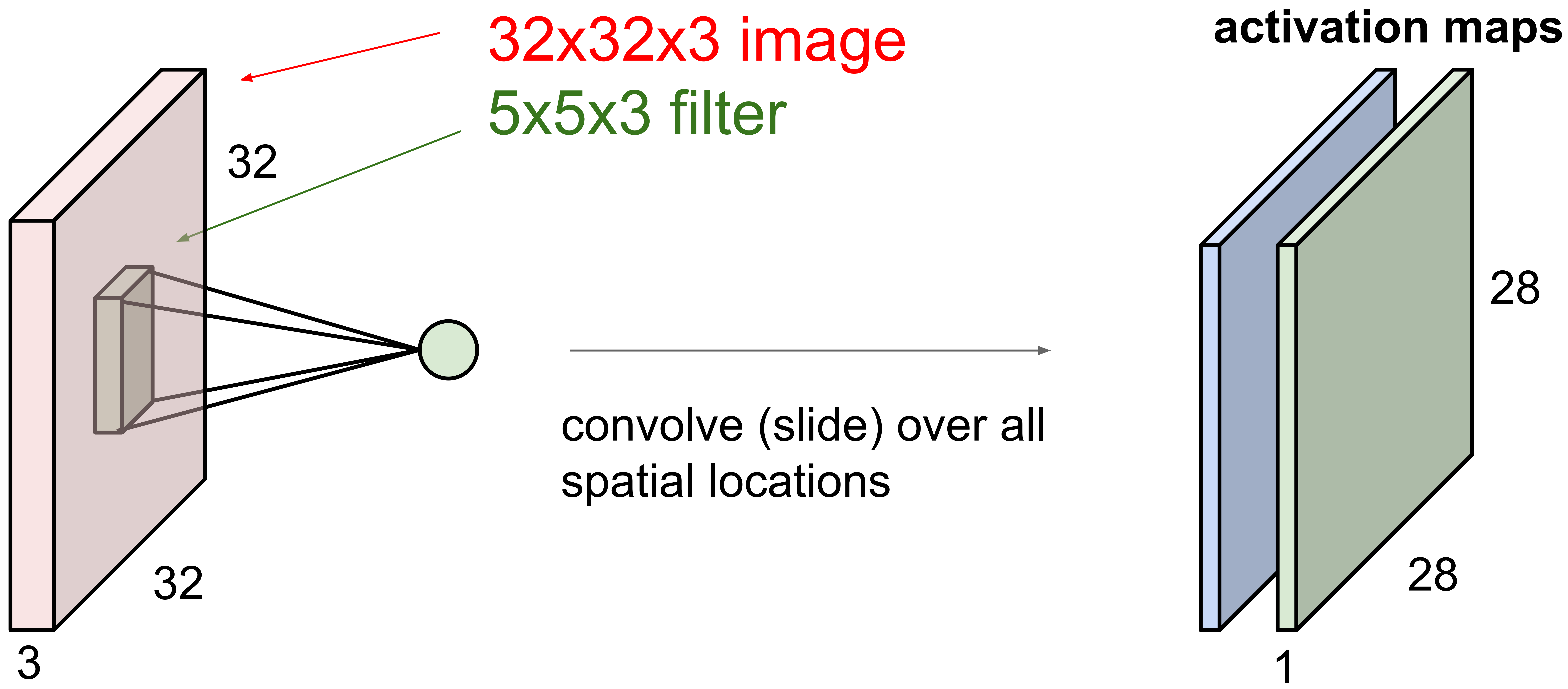


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolution Layer

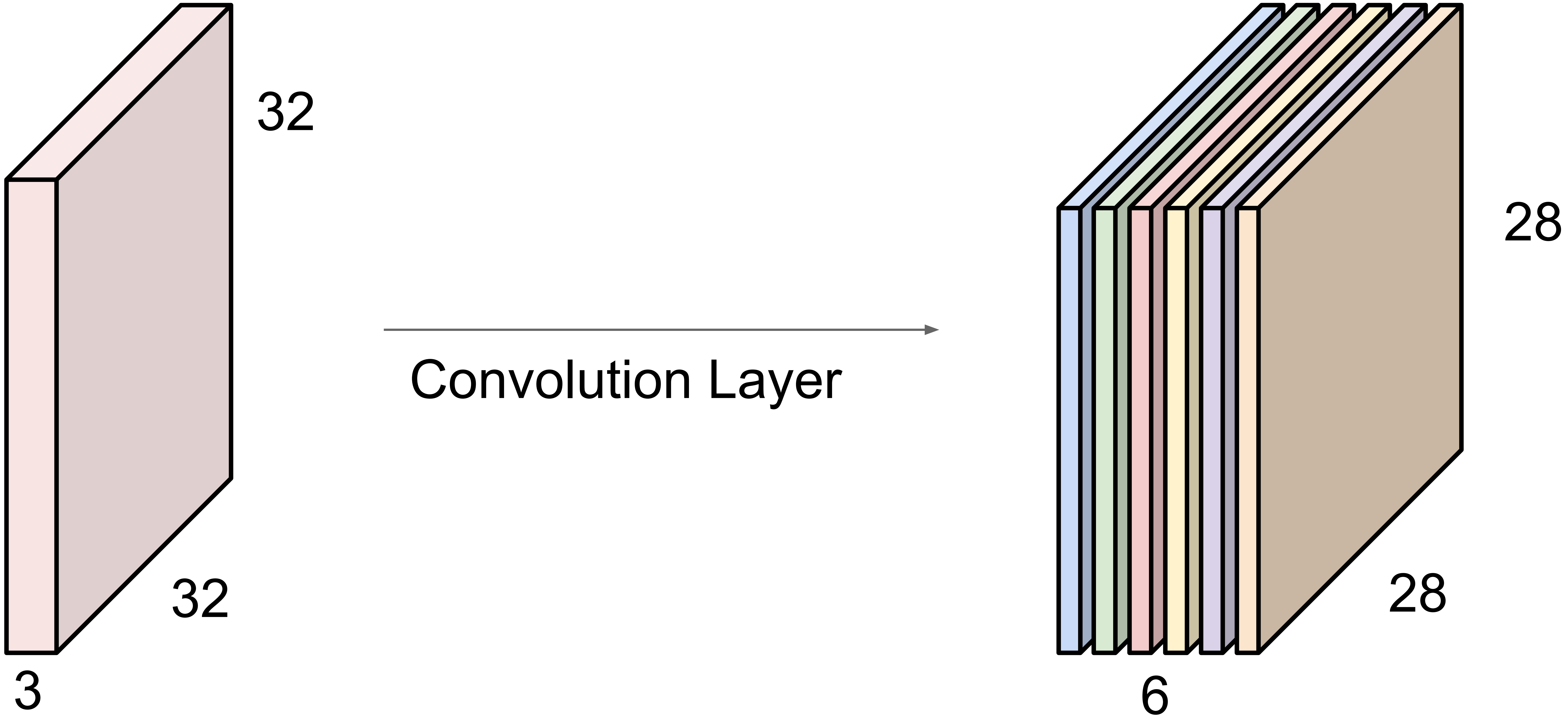


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolutional Neural Network

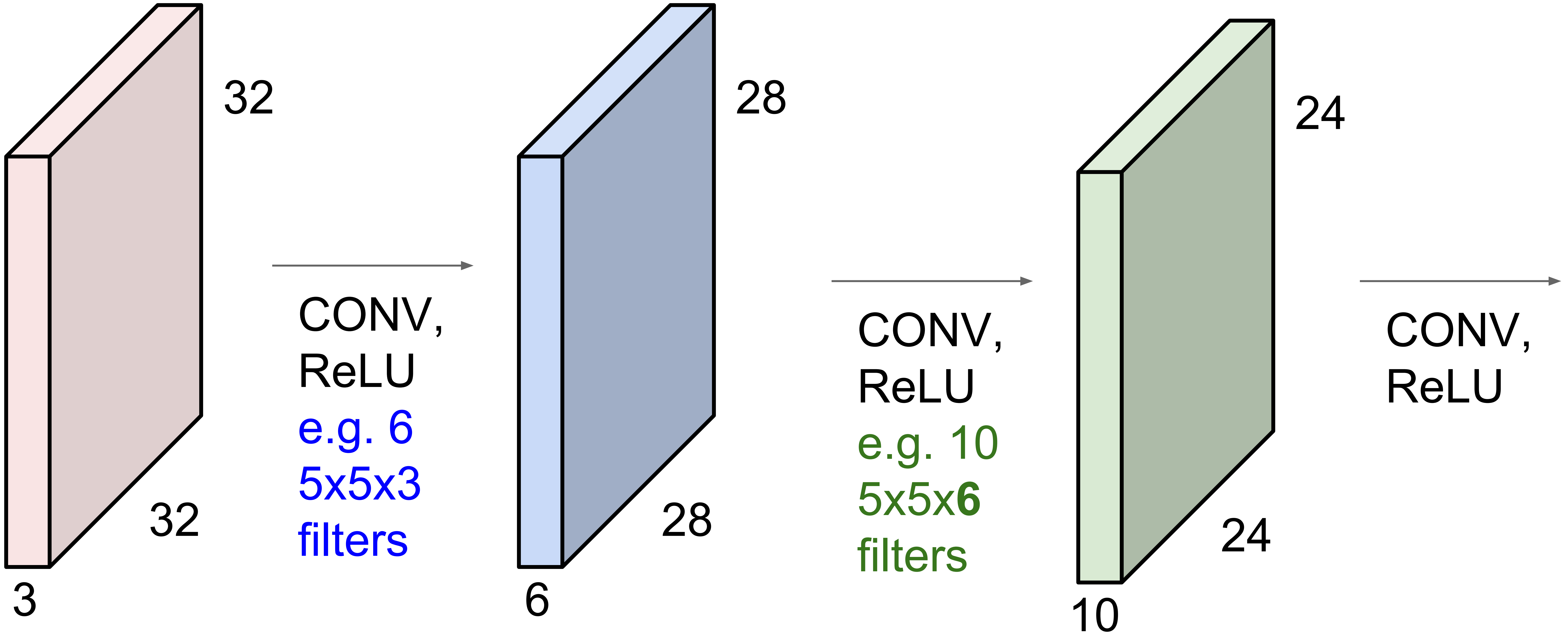


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

What do these layers learn?

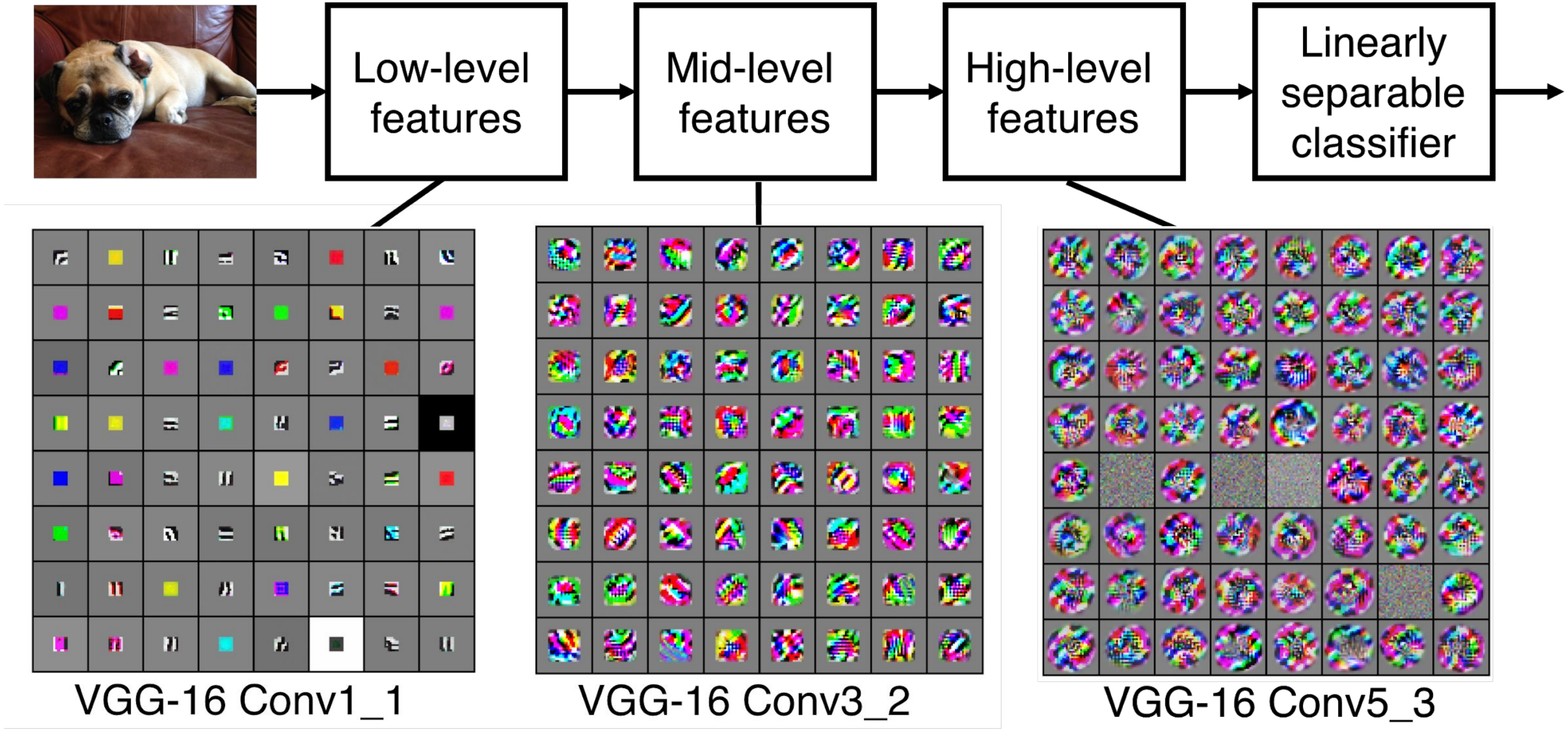
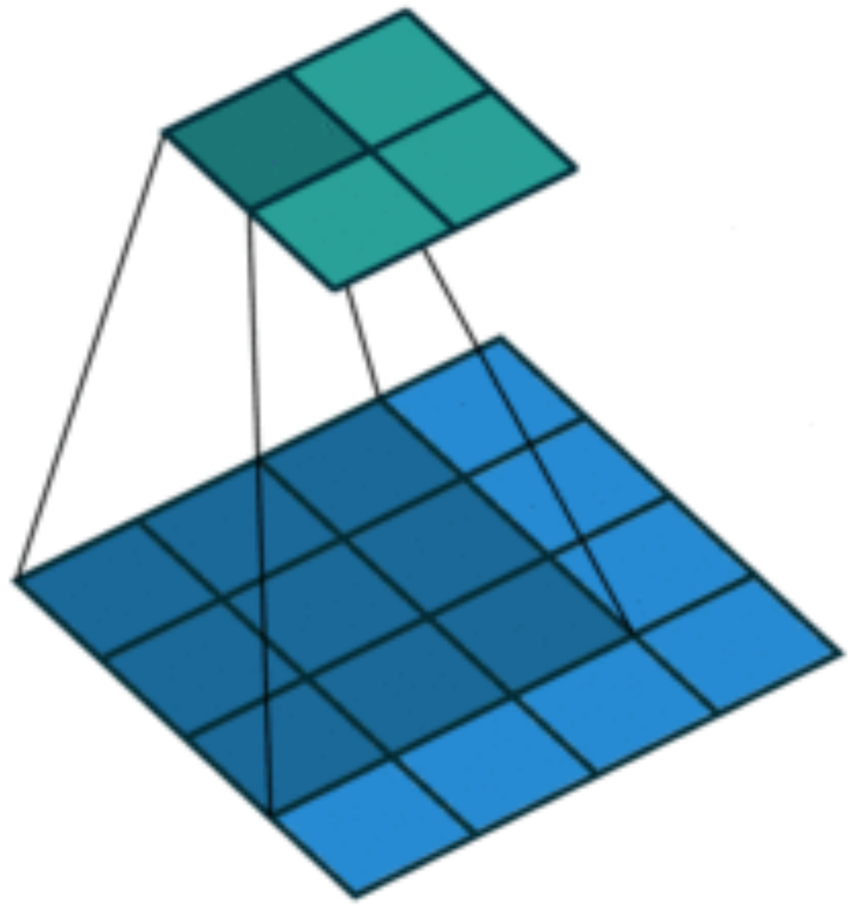


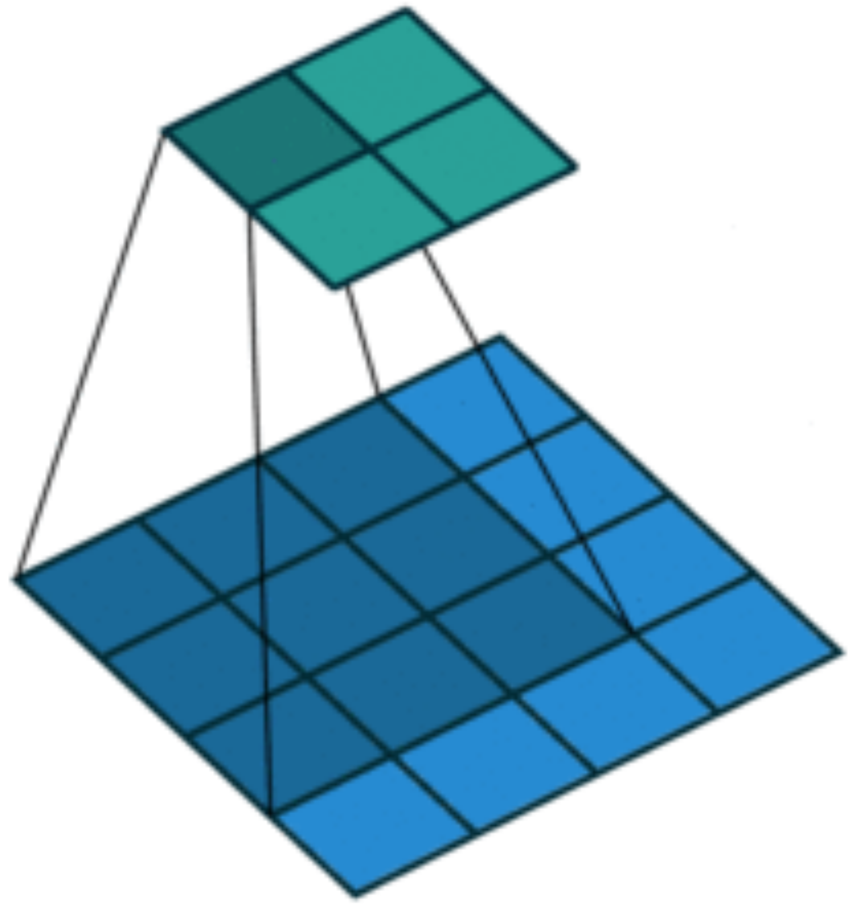
Image from: Simonyan and Zisserman, 2014

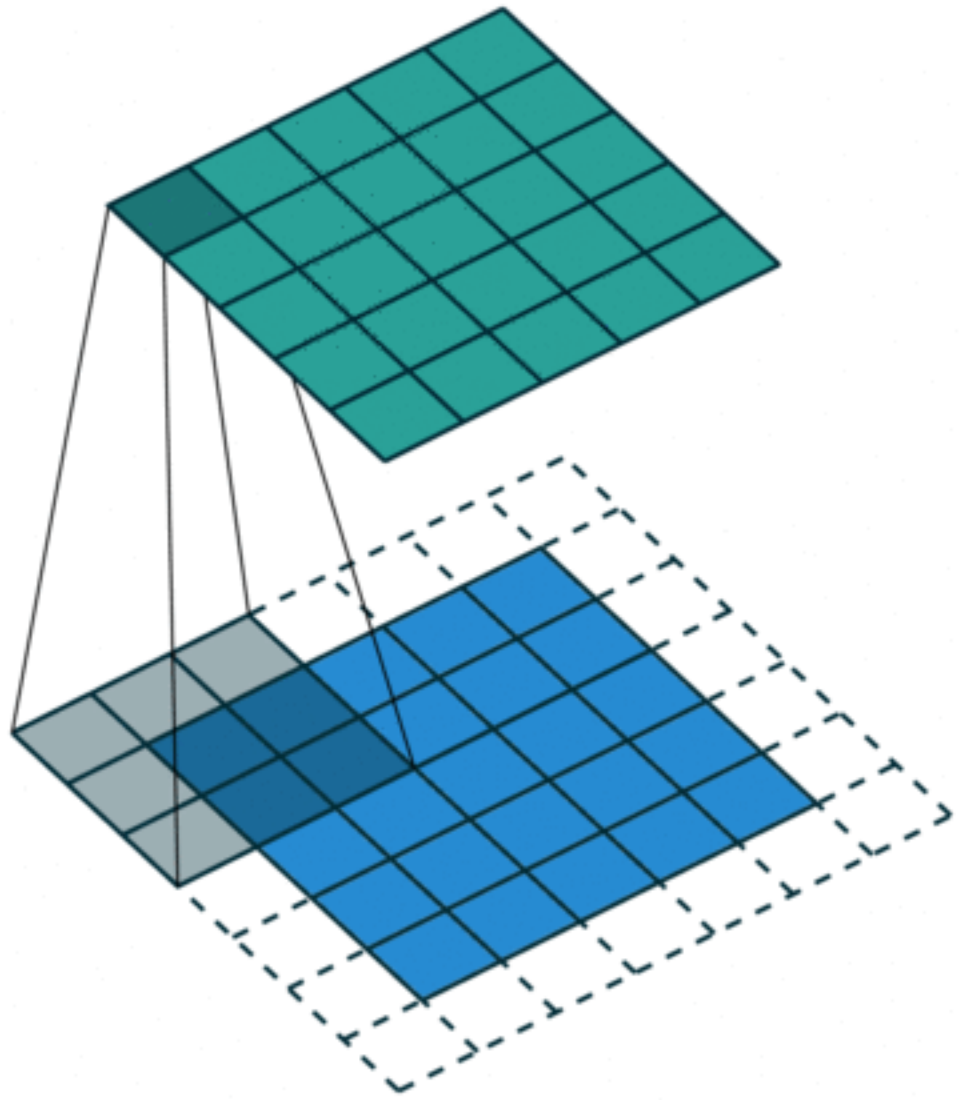
Convolutional Neural Networks (CNNs)



All animations are from: https://github.com/vdumoulin/conv_arithmetic

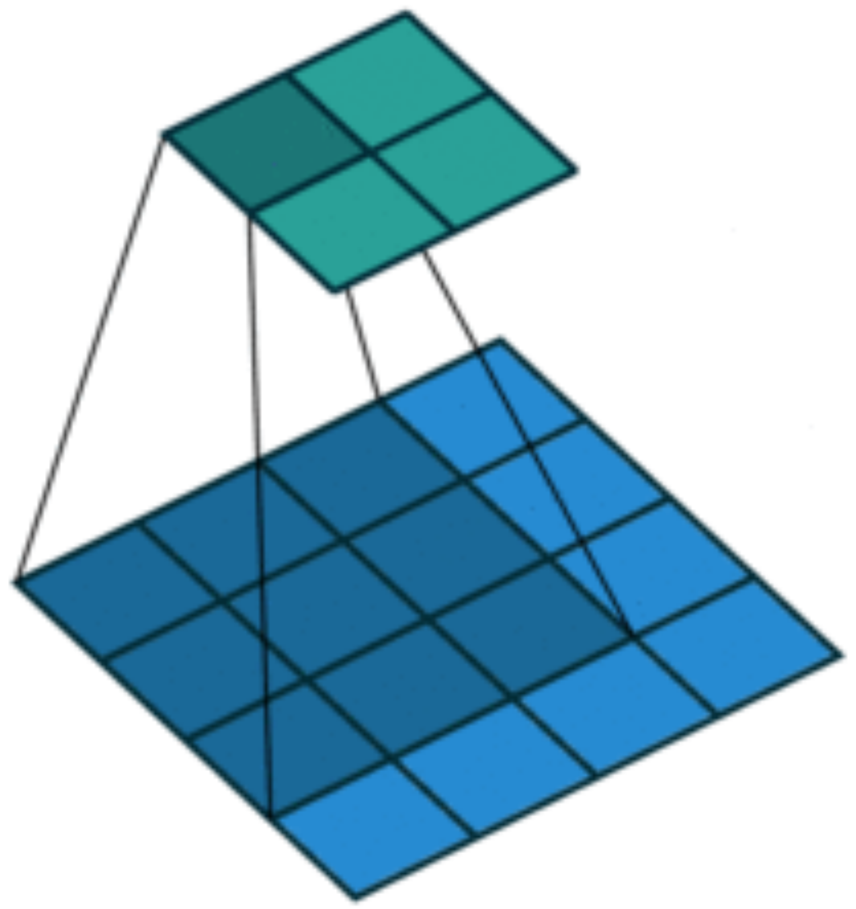
Convolutional Neural Networks (CNNs)

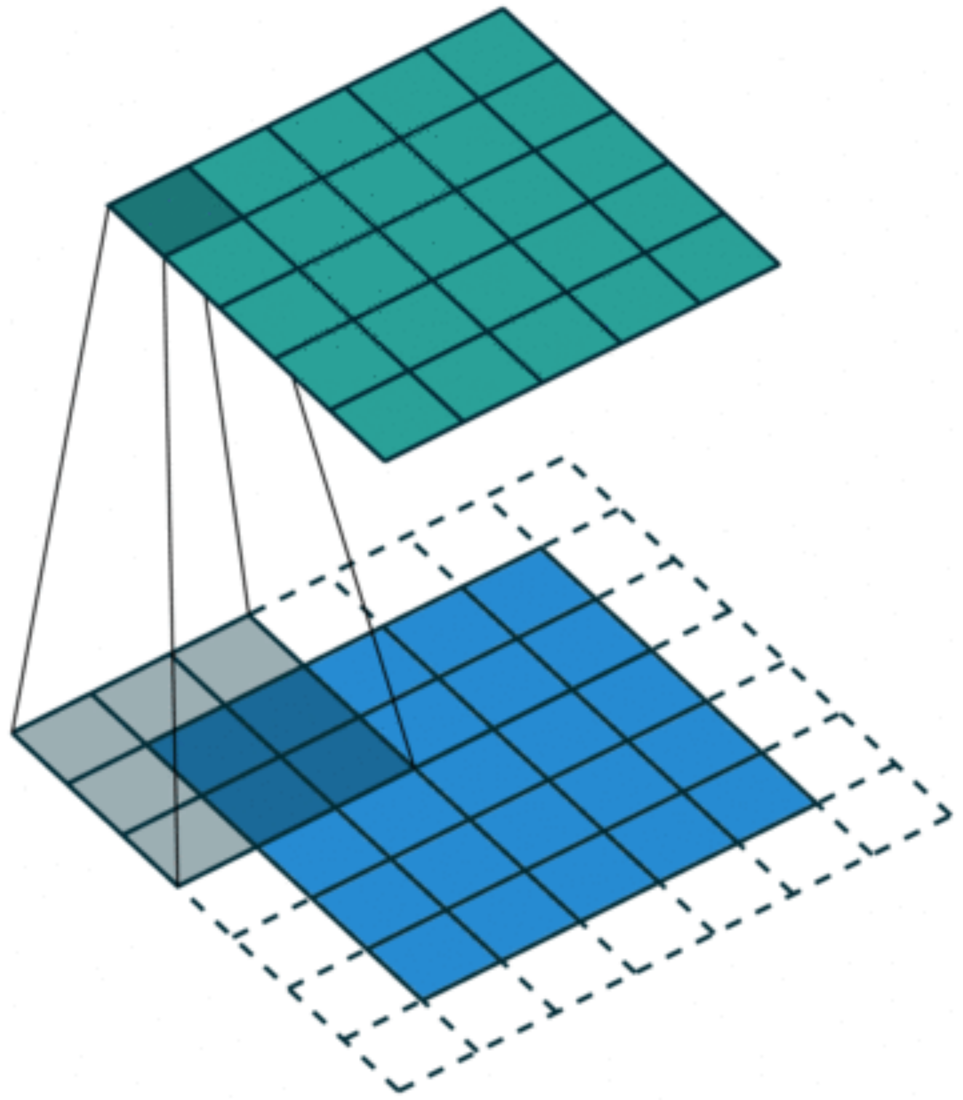


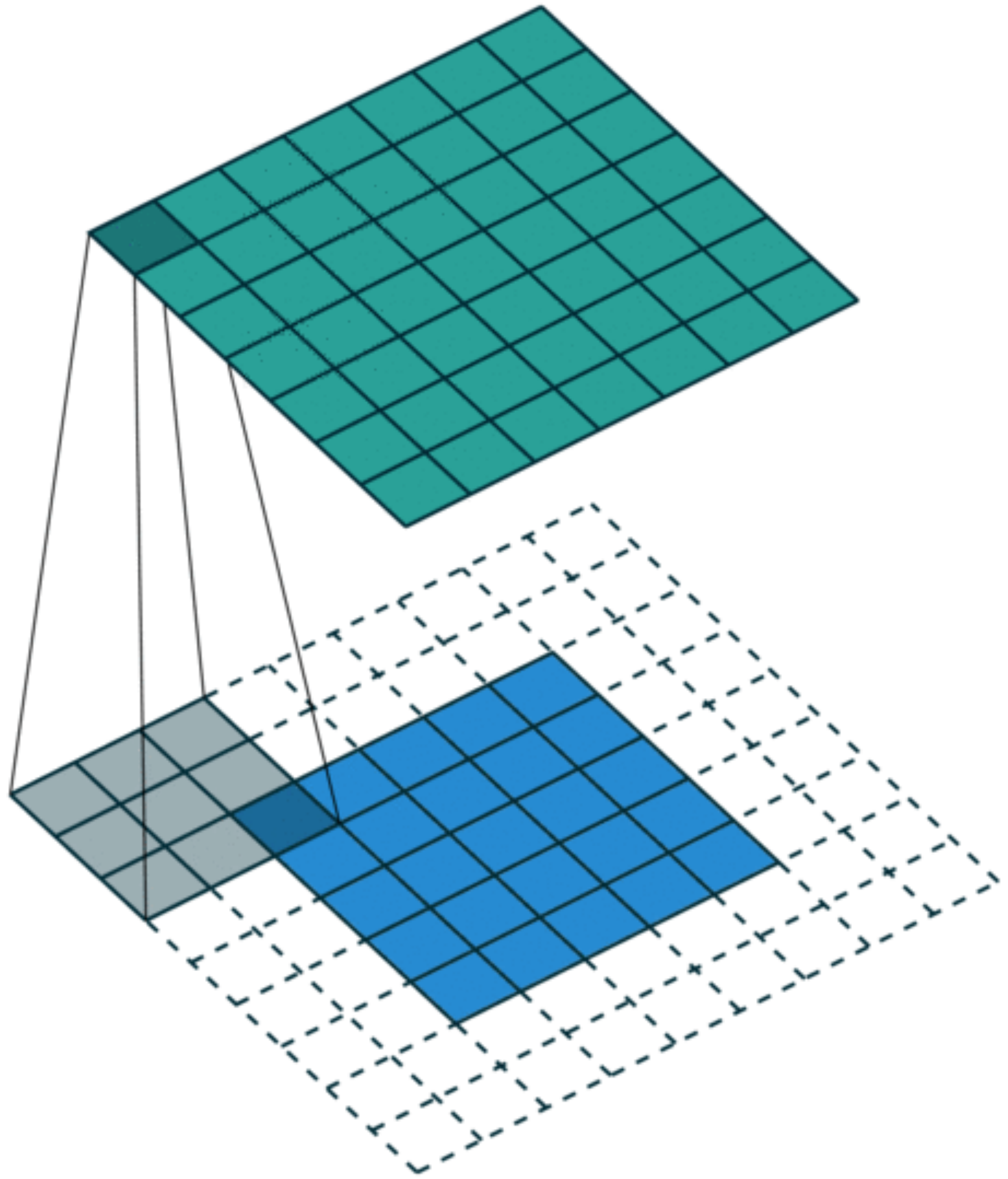


All animations are from: https://github.com/vdumoulin/conv_arithmetic

Convolutional Neural Networks (CNNs)

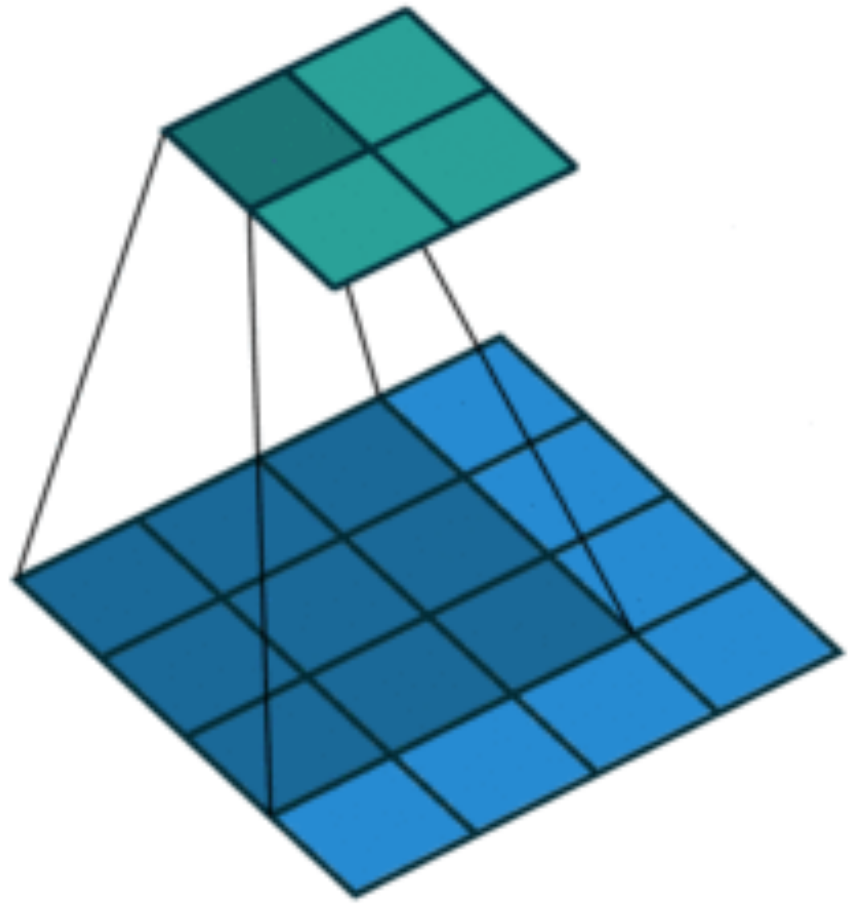


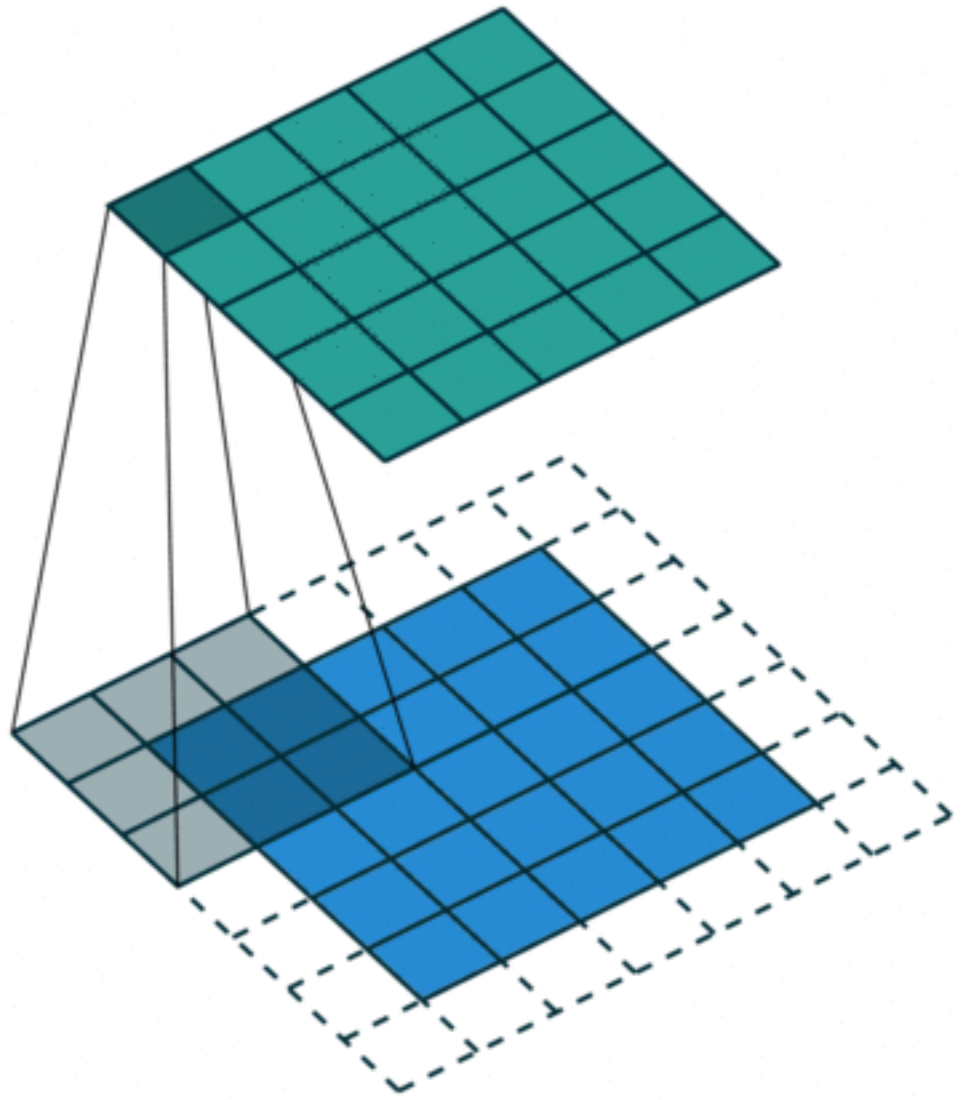


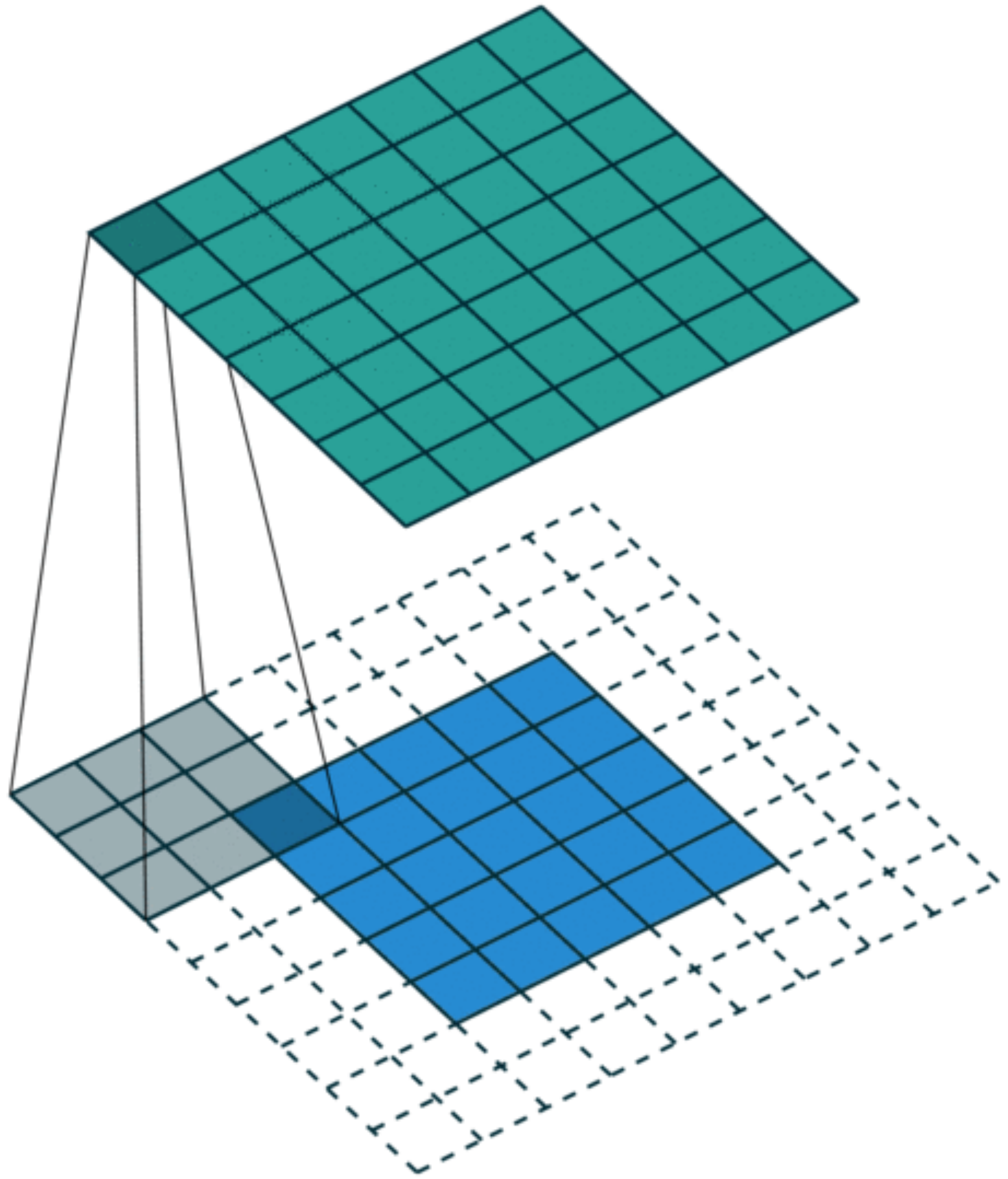


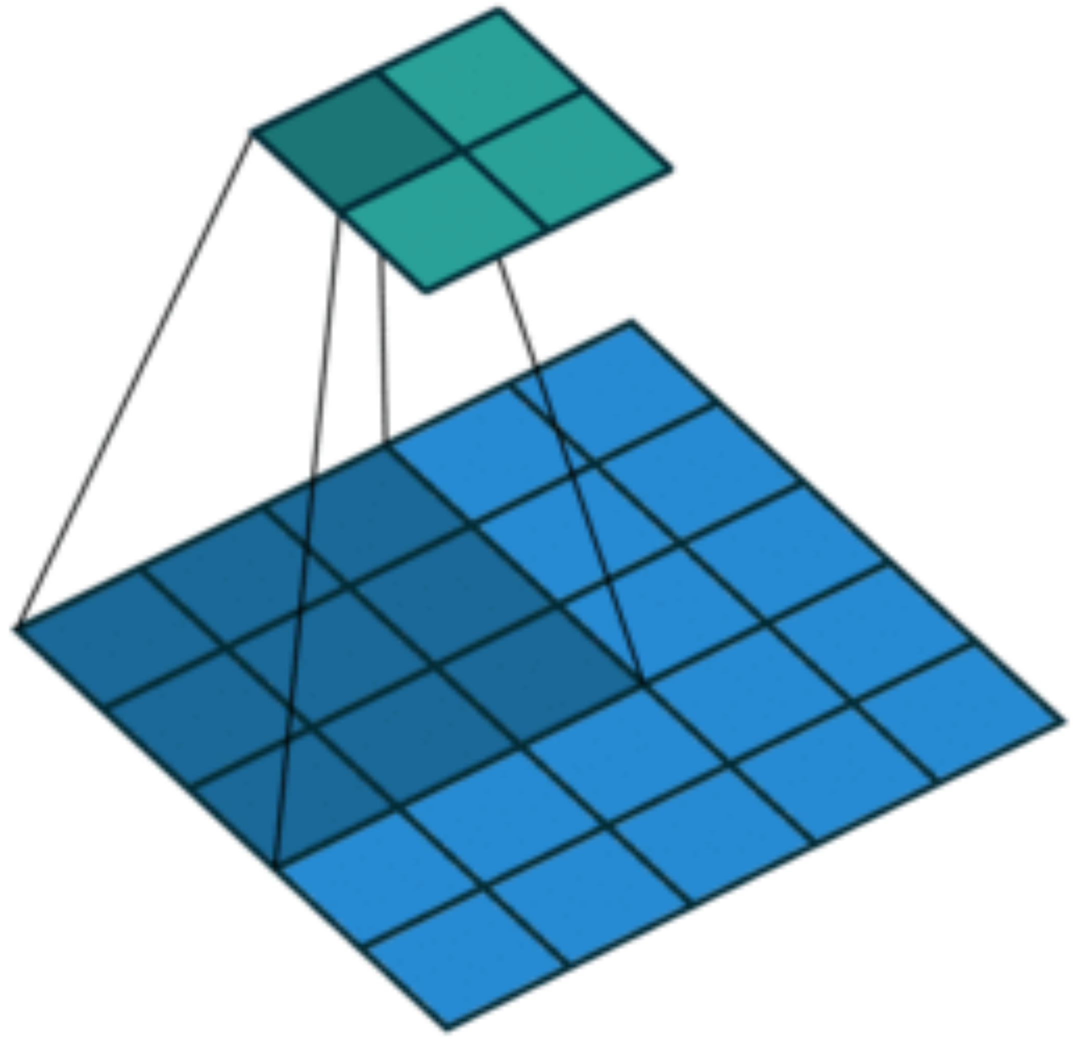
All animations are from: https://github.com/vdumoulin/conv_arithmetic

Convolutional Neural Networks (CNNs)



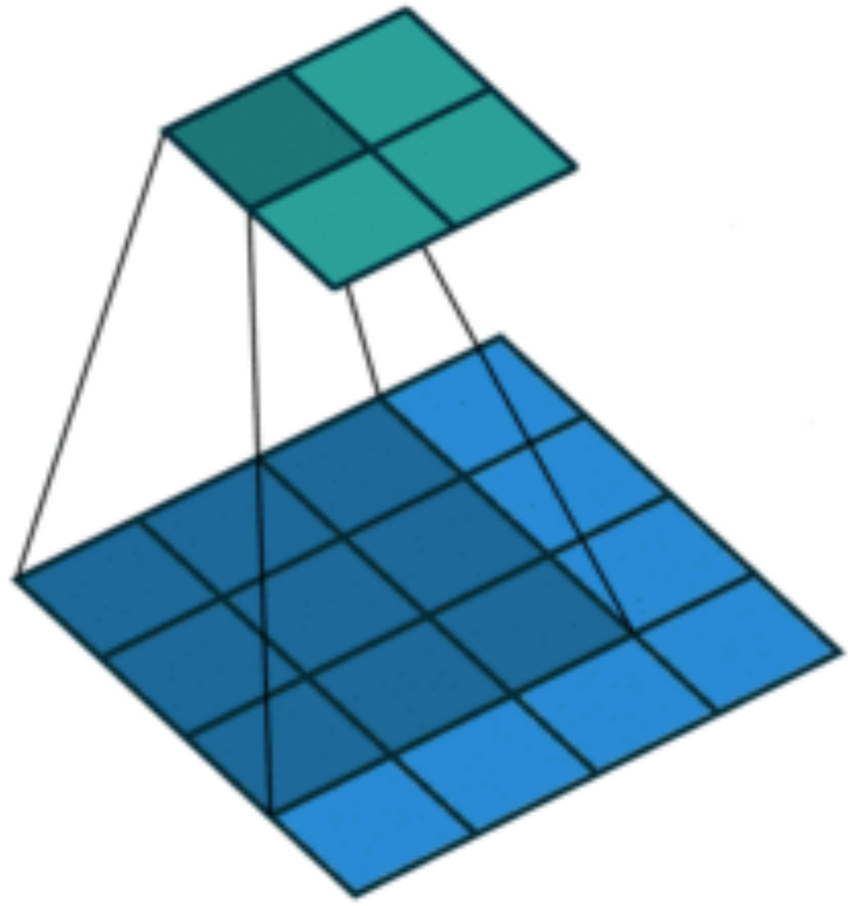


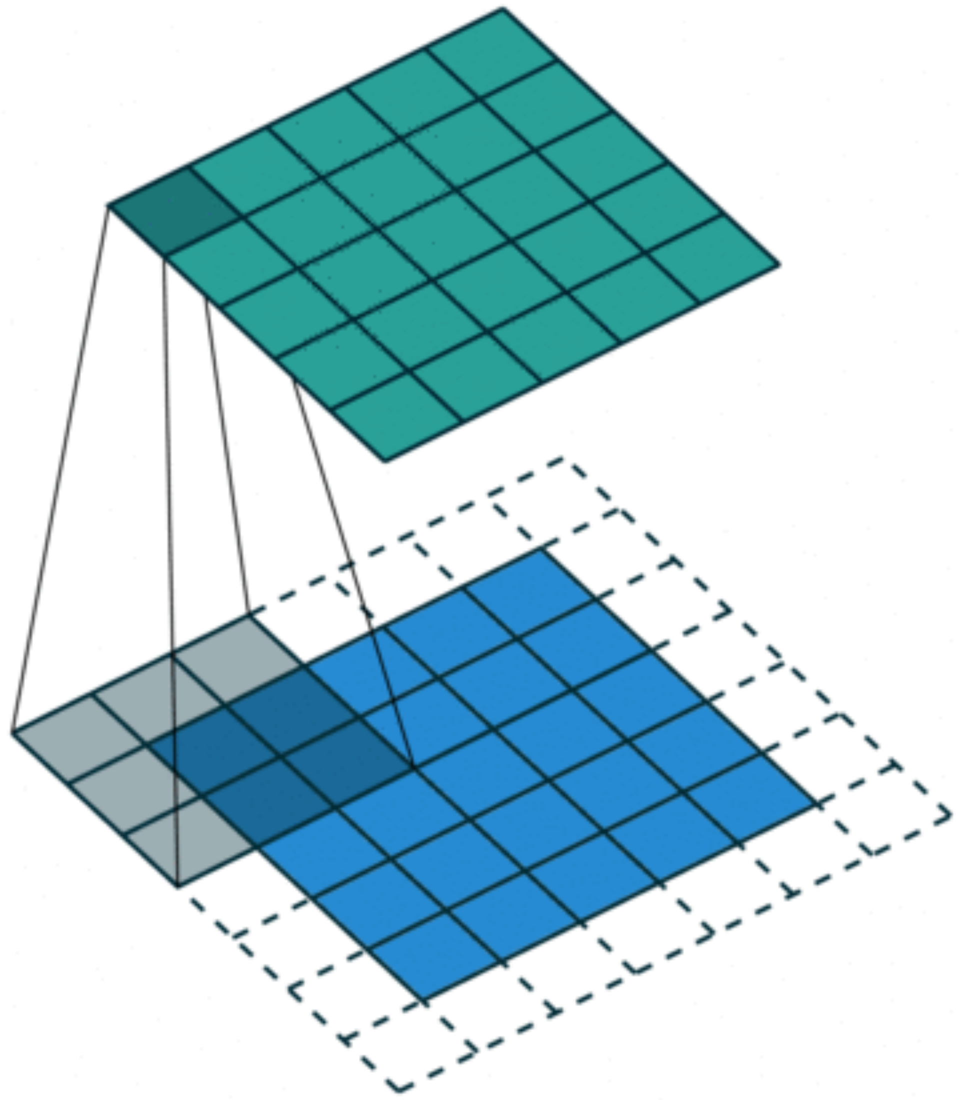


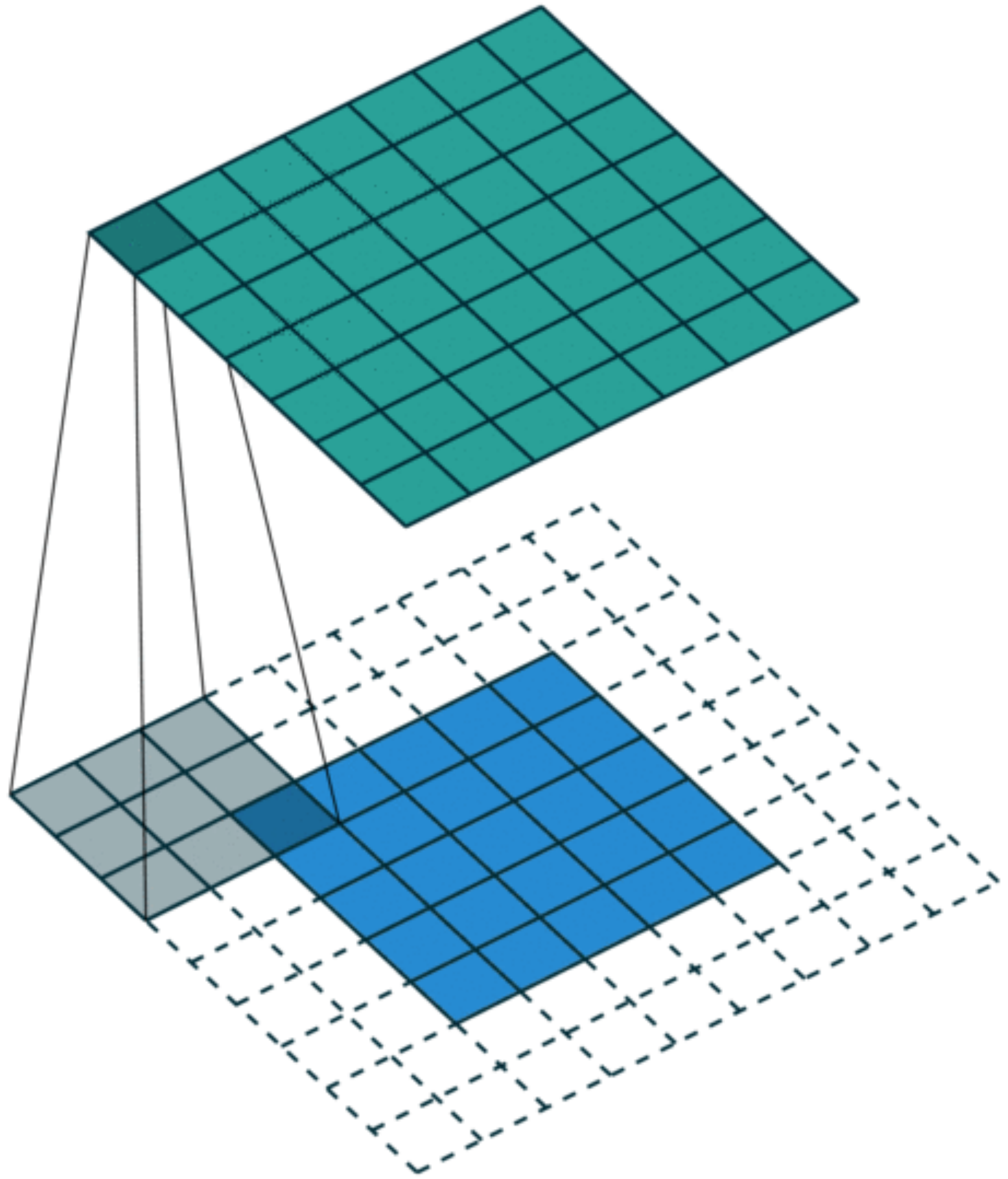


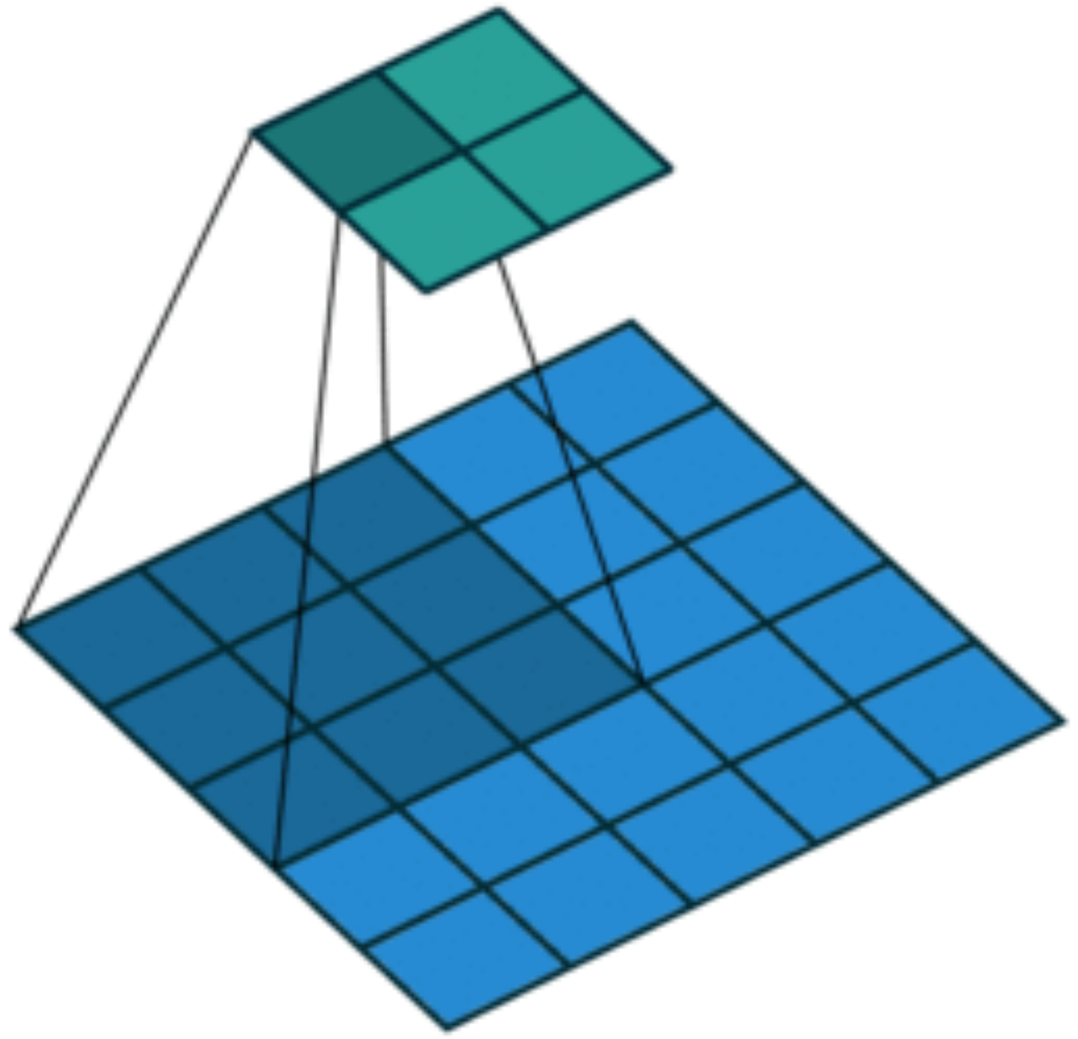
All animations are from: https://github.com/vdumoulin/conv_arithmetic

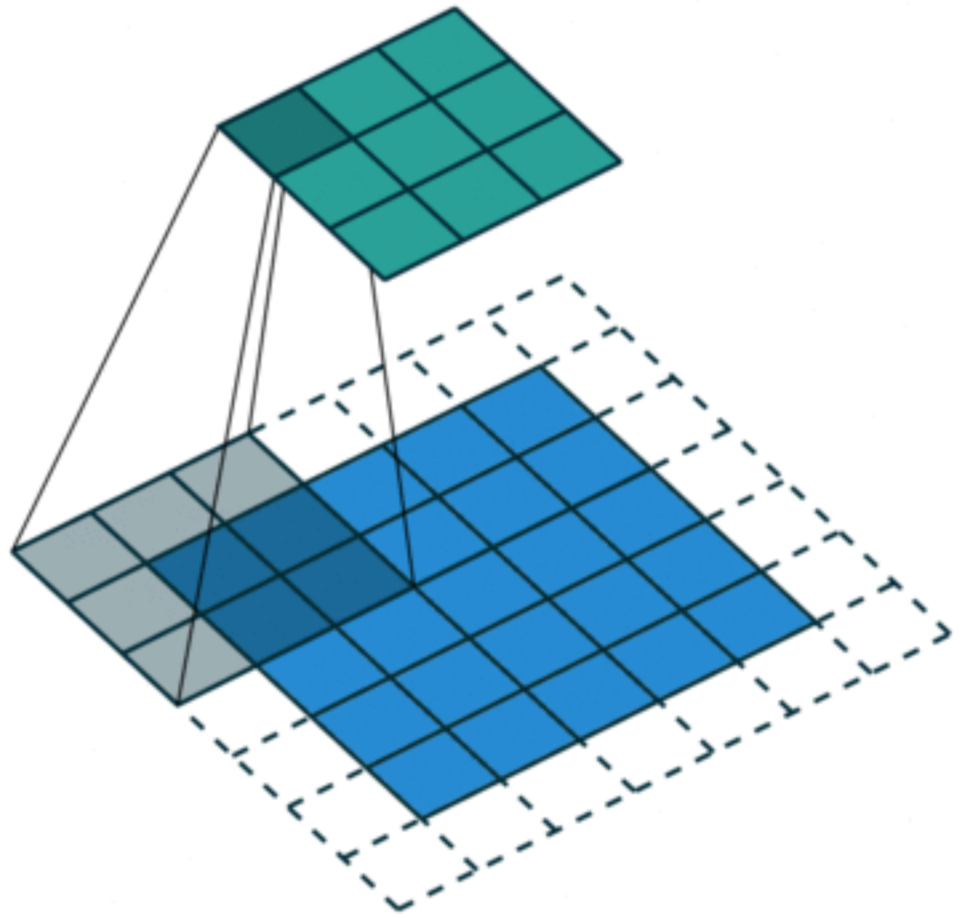
Convolutional Neural Networks (CNNs)











All animations are from: https://github.com/vdumoulin/conv_arithmetic

Convolution Layers: Summary

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
    - Number of filters K,
    - their spatial extent F,
    - the stride S,
    - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
    - $W_2 = (W_1 - F + 2P)/S + 1$
    - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
    - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.

Summary from: http://cs231n.github.io/convolutional-networks/

Pooling Layer

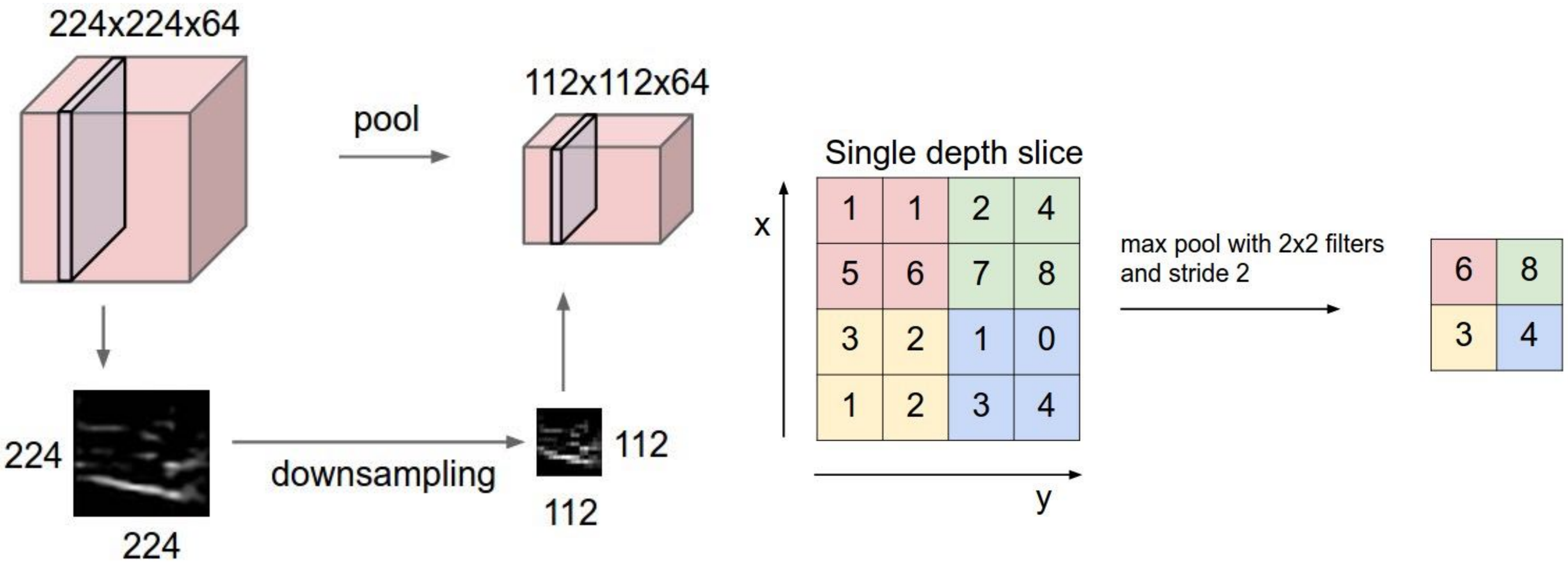


Image from: http://cs231n.github.io/convolutional-networks/

Pooling Layer

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires two hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F)/S + 1$
  - $H_2 = (H_1 - F)/S + 1$
  - $D_2 = D_1$

Summary from: http://cs231n.github.io/convolutional-networks/

CNNs for Speech

Speech features to be fed to a CNN

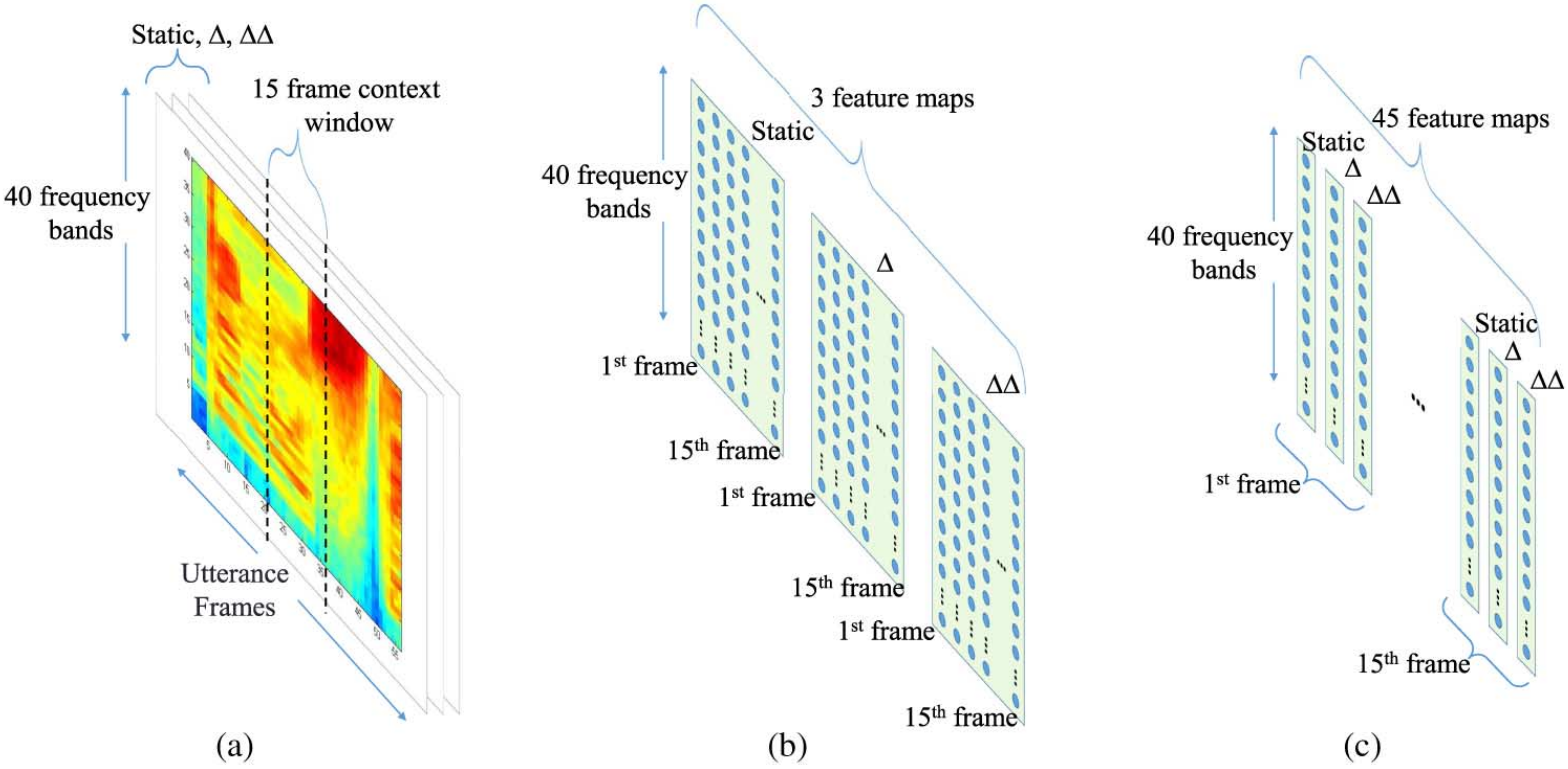
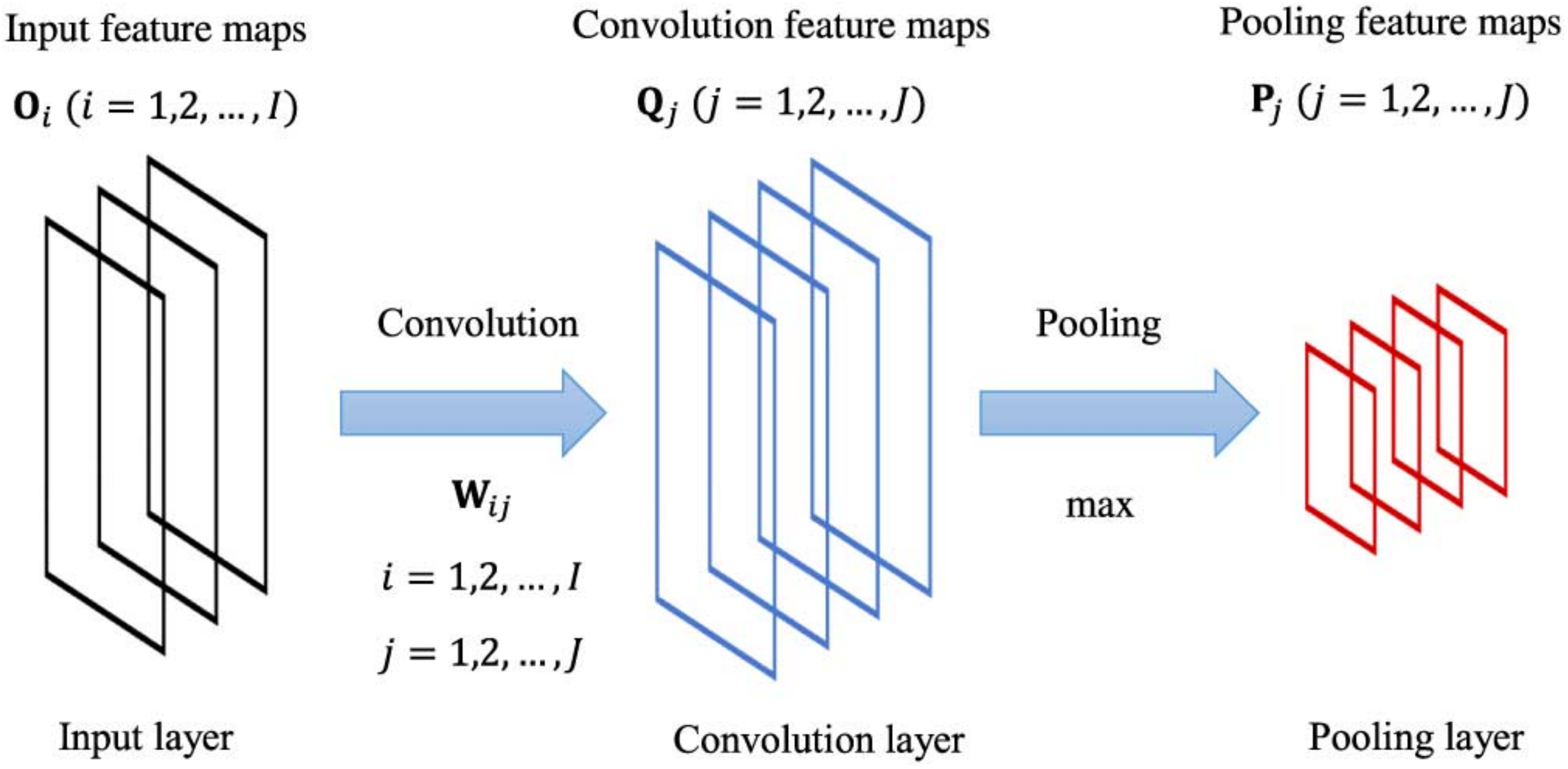


Image from Abdel-Hamid et al., "Convolutional Neural Networks for Speech Recognition", TASLP 2014

Illustrating a CNN layer



$$Q_j = \sigma\left(\sum_{i=1}^{I} O_i * \mathbf{w}_{i,j}\right) \quad (j = 1, \ldots, J)$$

Convolution Layer

$$p_{i,m} = \max_{n=1}^{G} \quad q_{i,(m-1)\times s+n}$$

Pooling Layer

Image from Abdel-Hamid et al., "Convolutional Neural Networks for Speech Recognition", TASLP 2014

Convolution operations involve a large sparse matrix

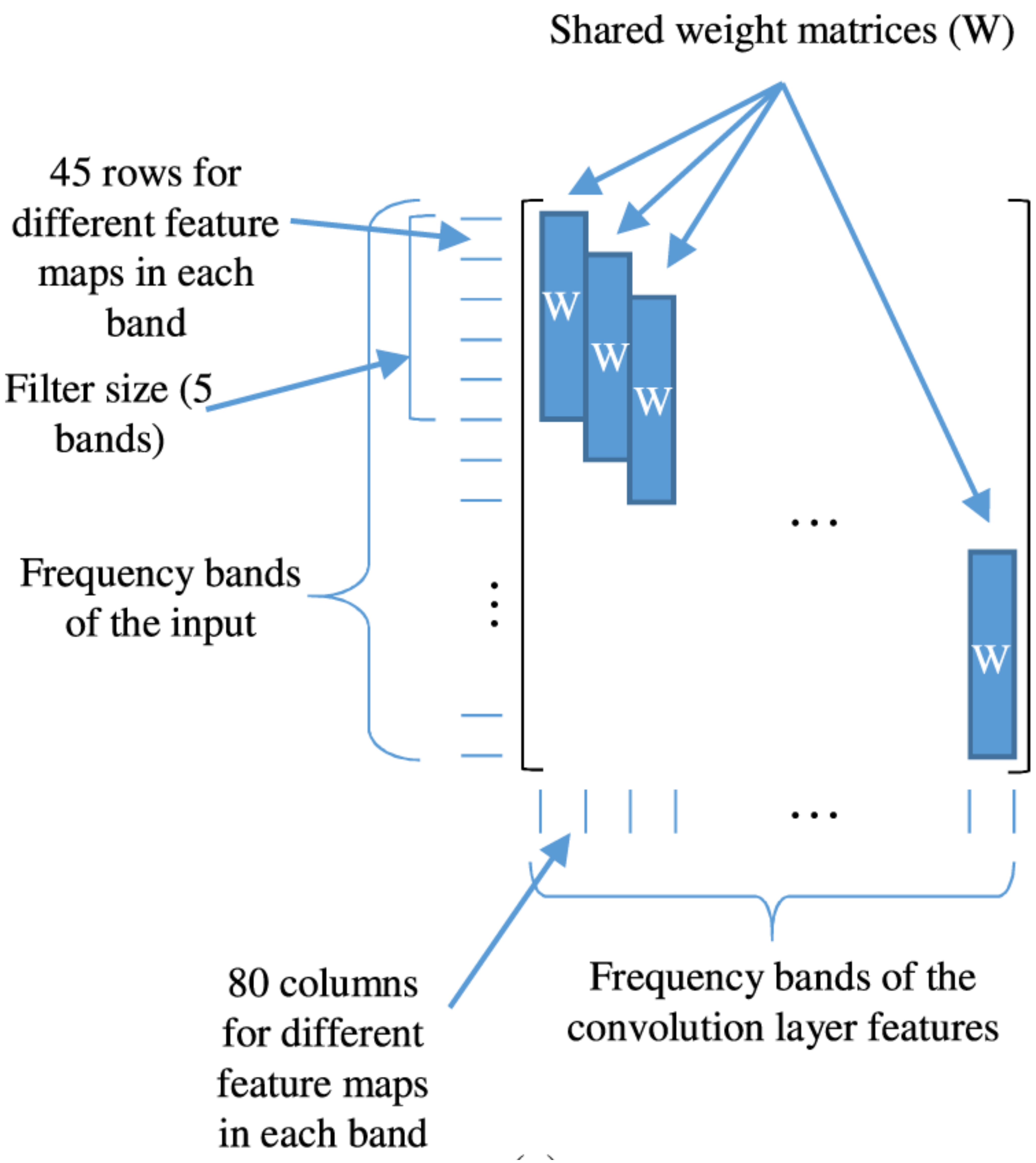


Image from Abdel-Hamid et al., “Convolutional Neural Networks for Speech Recognition”, TASLP 2014

CNN Architecture used in a hybrid ASR system

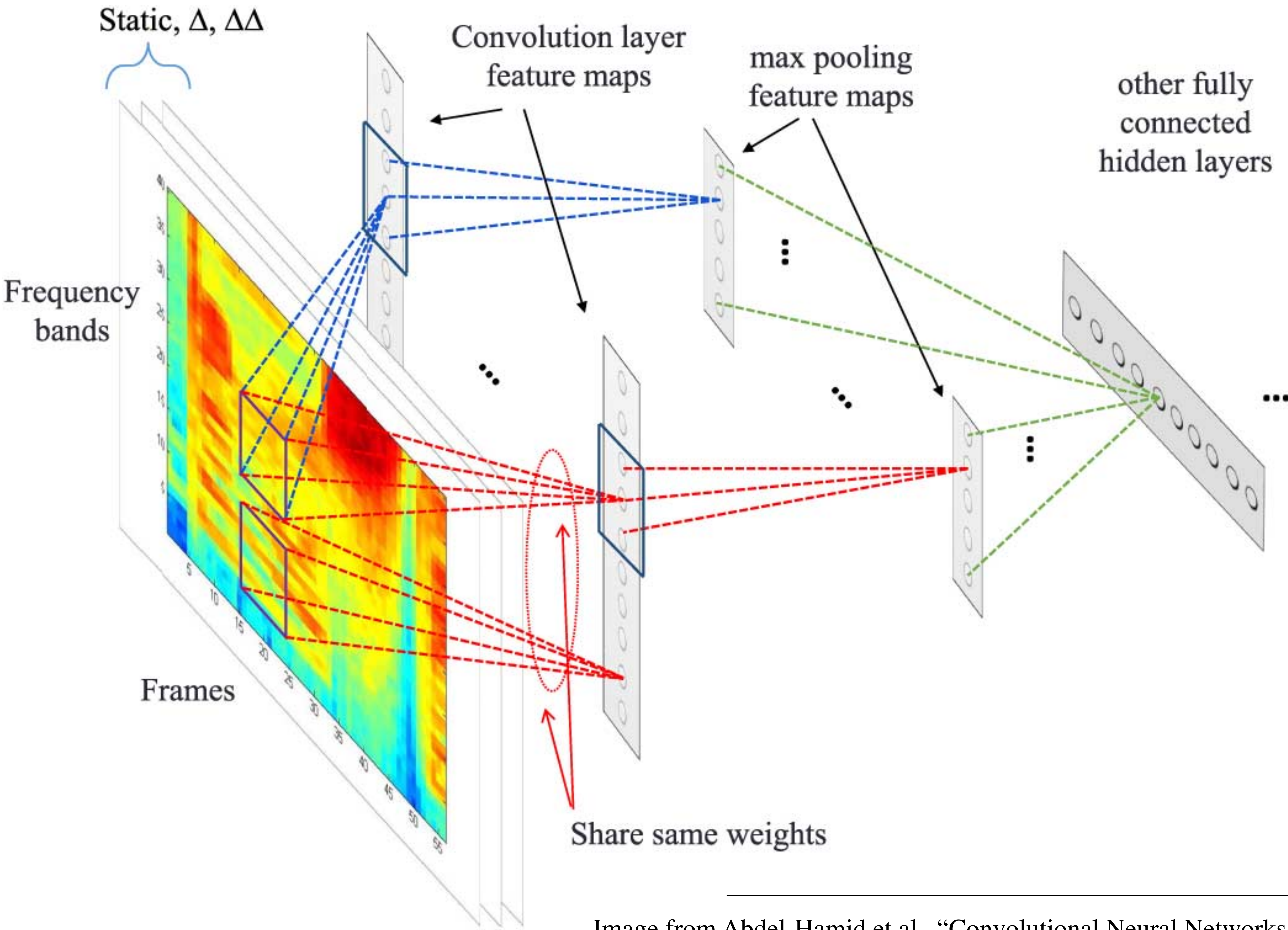
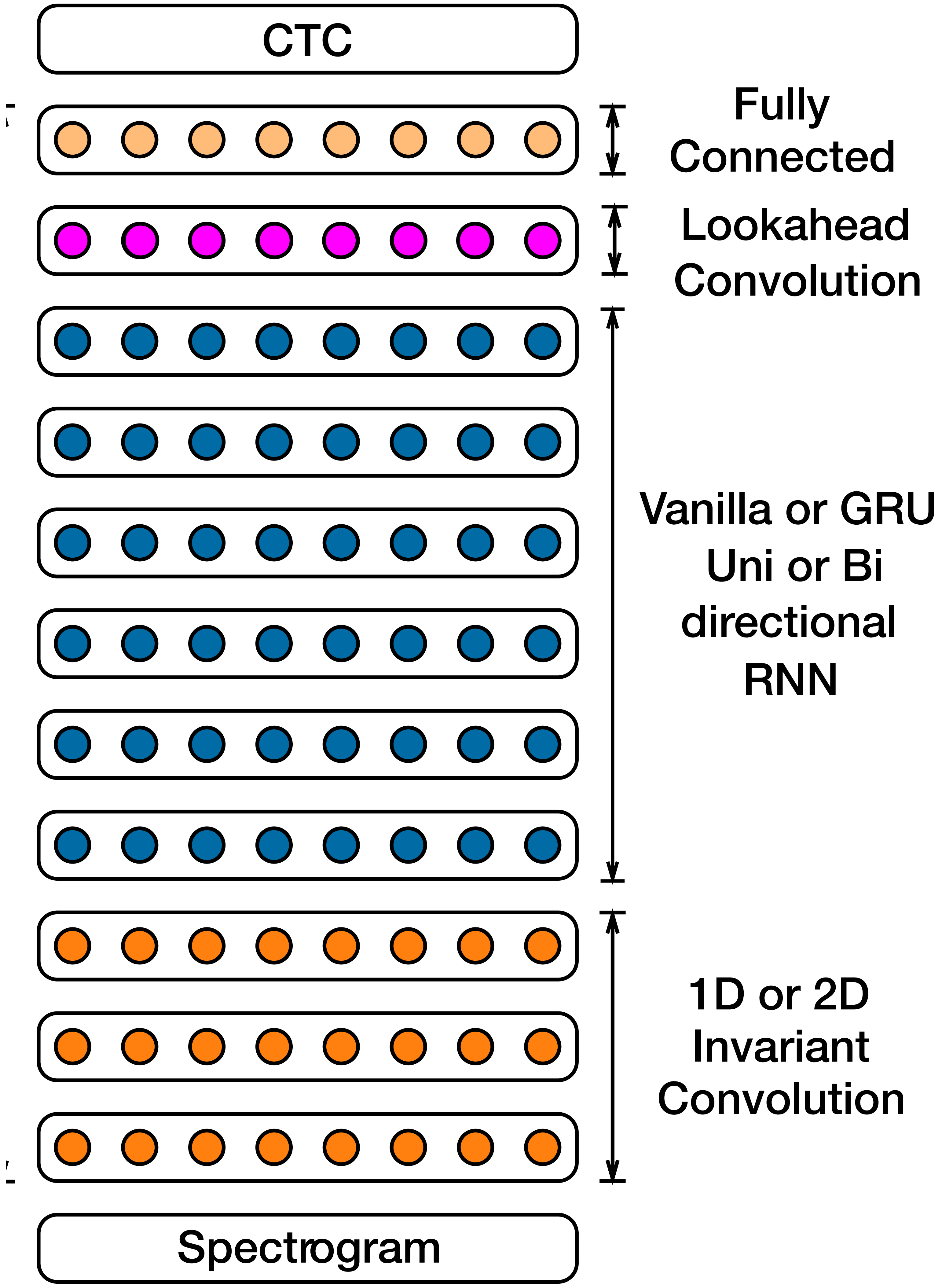


Image from Abdel-Hamid et al., “Convolutional Neural Networks for Speech Recognition”, TASLP 2014

Performance on TIMIT of different CNN architectures (Comparison with DNNs)

ID	Network structure	Average PER	min-max PER	# param's	# op's
1	DNN $\{2000 + 2\times1000\}$	22.02%	21.86-22.11%	6.9M	6.9M
2	DNN $\{2000 + 4\times1000\}$	21.87%	21.68-21.98%	8.9M	8.9M
3	CNN $\{\text{LWS(m:150 p:6 s:2 f:8)} + 2\times1000\}$	20.17%	19.92-20.41%	5.4M	10.7M
4	CNN $\{\text{FWS(m:360 p:6 s:2 f:8)} + 2\times1000\}$	20.31%	20.16-20.58%	8.5M	13.6M

More recent ASR system: Deep Speech 2



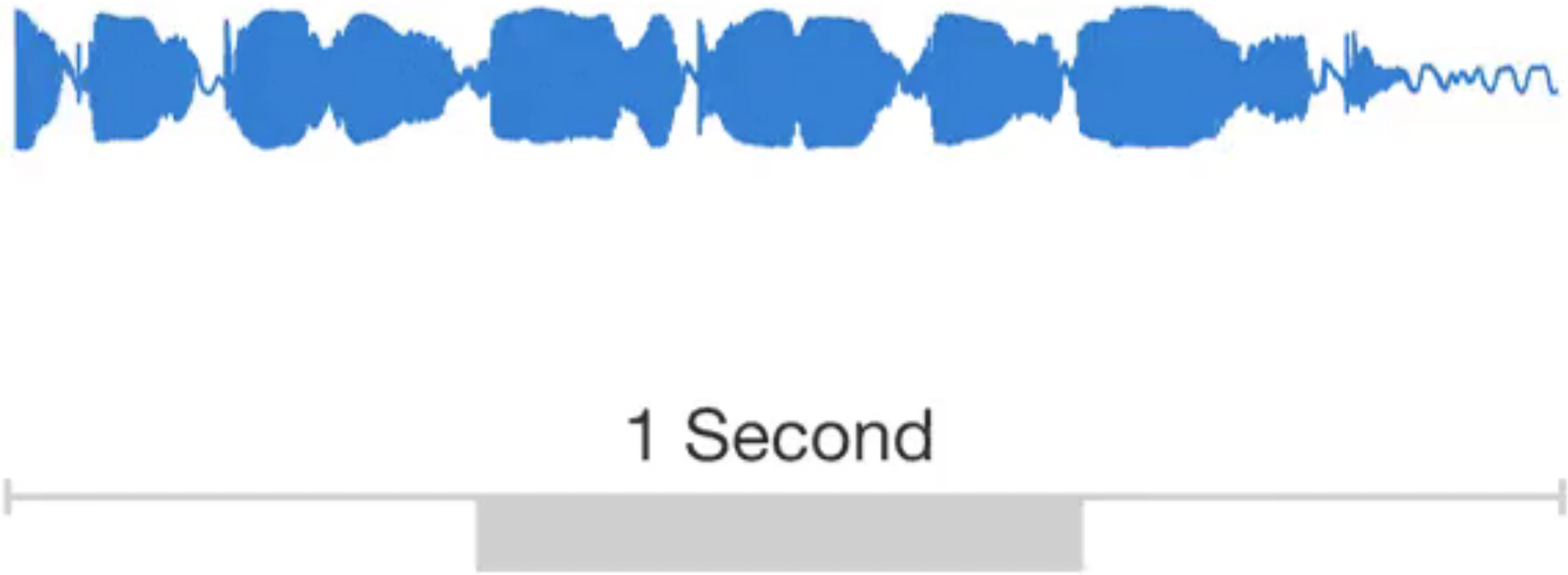
Architecture	Channels	Filter dimension	Stride	Regular Dev	Noisy Dev
1-layer 1D	1280	11	2	9.52	19.36
2-layer 1D	640, 640	5, 5	1, 2	9.67	19.21
3-layer 1D	512, 512, 512	5, 5, 5	1, 1, 2	9.20	20.22
1-layer 2D	32	41x11	2x2	8.94	16.22
2-layer 2D	32, 32	41x11, 21x11	2x2, 2x1	9.06	15.71
3-layer 2D	32, 32, 96	41x11, 21x11, 21x11	2x2, 2x1, 2x1	8.61	14.74

	Test set	Ours	Human
Read	WSJ eval'92	3.10	5.03
	WSJ eval'93	4.42	8.08
	LibriSpeech test-clean	5.15	5.83
	LibriSpeech test-other	12.73	12.69
Accented	VoxForge American-Canadian	7.94	4.85
	VoxForge Commonwealth	14.85	8.15
	VoxForge European	18.44	12.76
	VoxForge Indian	22.89	22.15
Noisy	CHiME eval real	21.59	11.84
	CHiME eval sim	42.55	31.33

Image from Amodei et al., "Deep speech 2: End-to-end speech recognition in English and Mandarin", ICML 2016

TTS: Wavenet

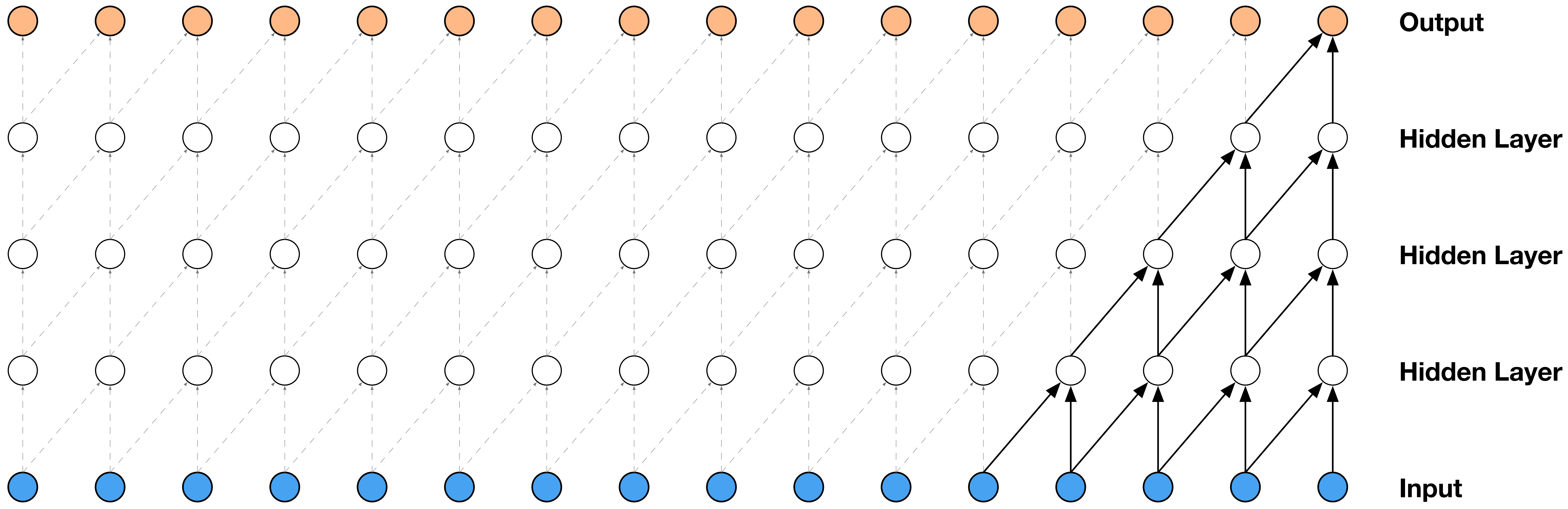
- Speech synthesis using an auto-regressive generative model
- Generates waveform sample-by-sample:16kHz sampling rate



Gif from https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Causal Convolutions

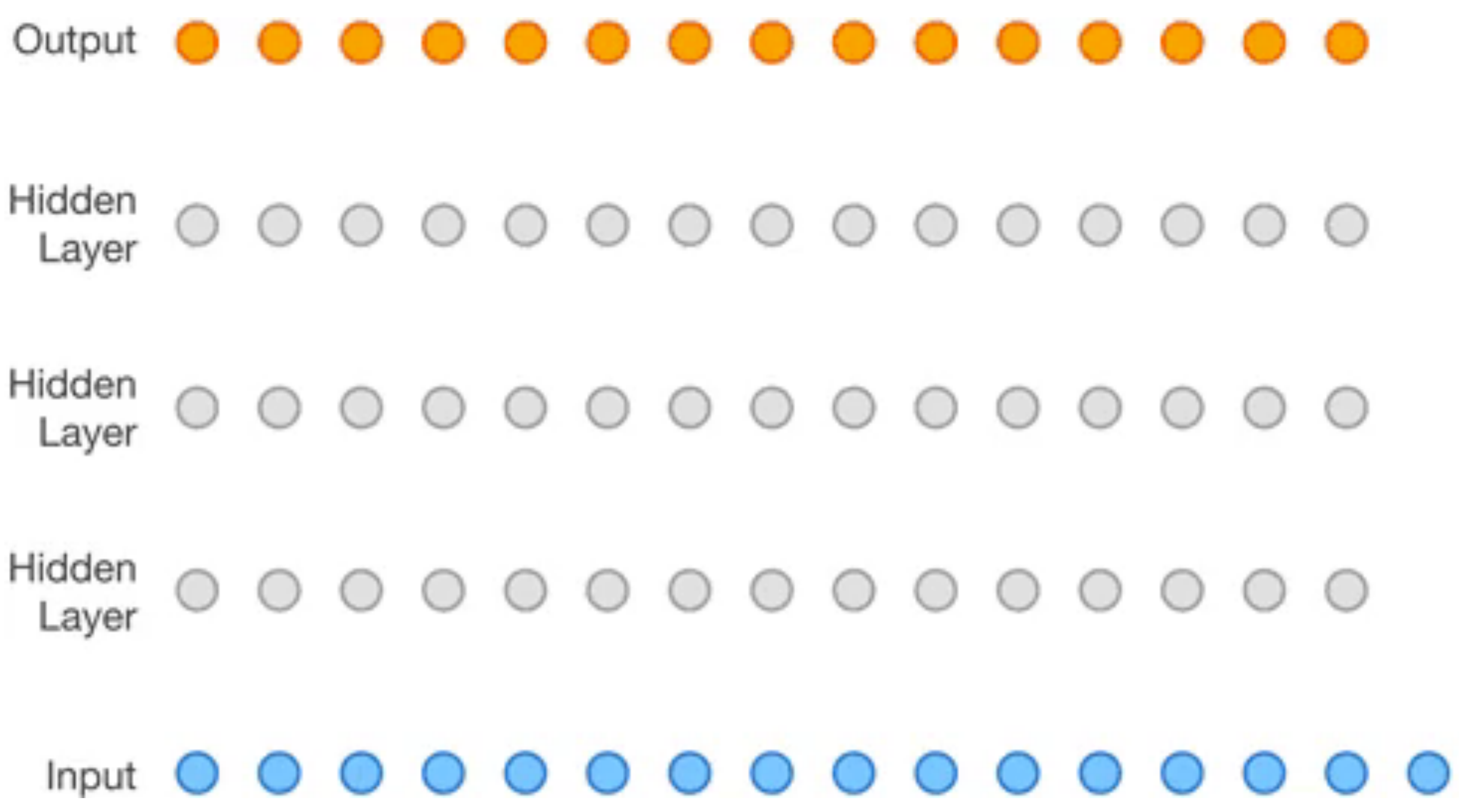
- Fully convolutional
- Prediction at timestep t cannot depend on any future timesteps



Gif from https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Dilated Convolutions

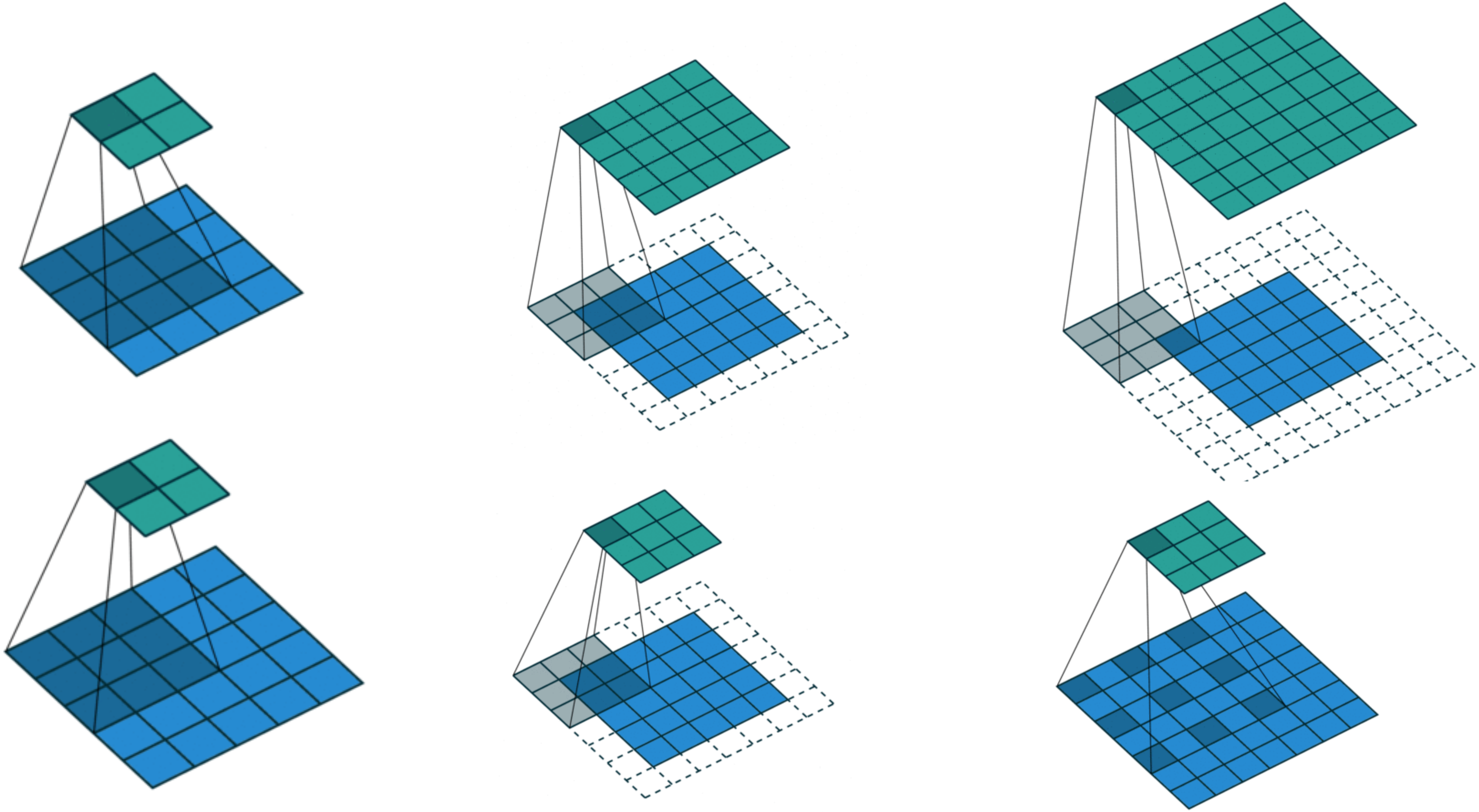
- Wavenet uses “dilated convolutions”
- Enables the network to have very large receptive fields



Gif from https://deepmind.com/blog/wavenet-generative-model-raw-audio/

[1]https://techcrunch.com/2017/10/04/googles-wavenet-machine-learning-based-speech-synthesis-comes-to-assistant/

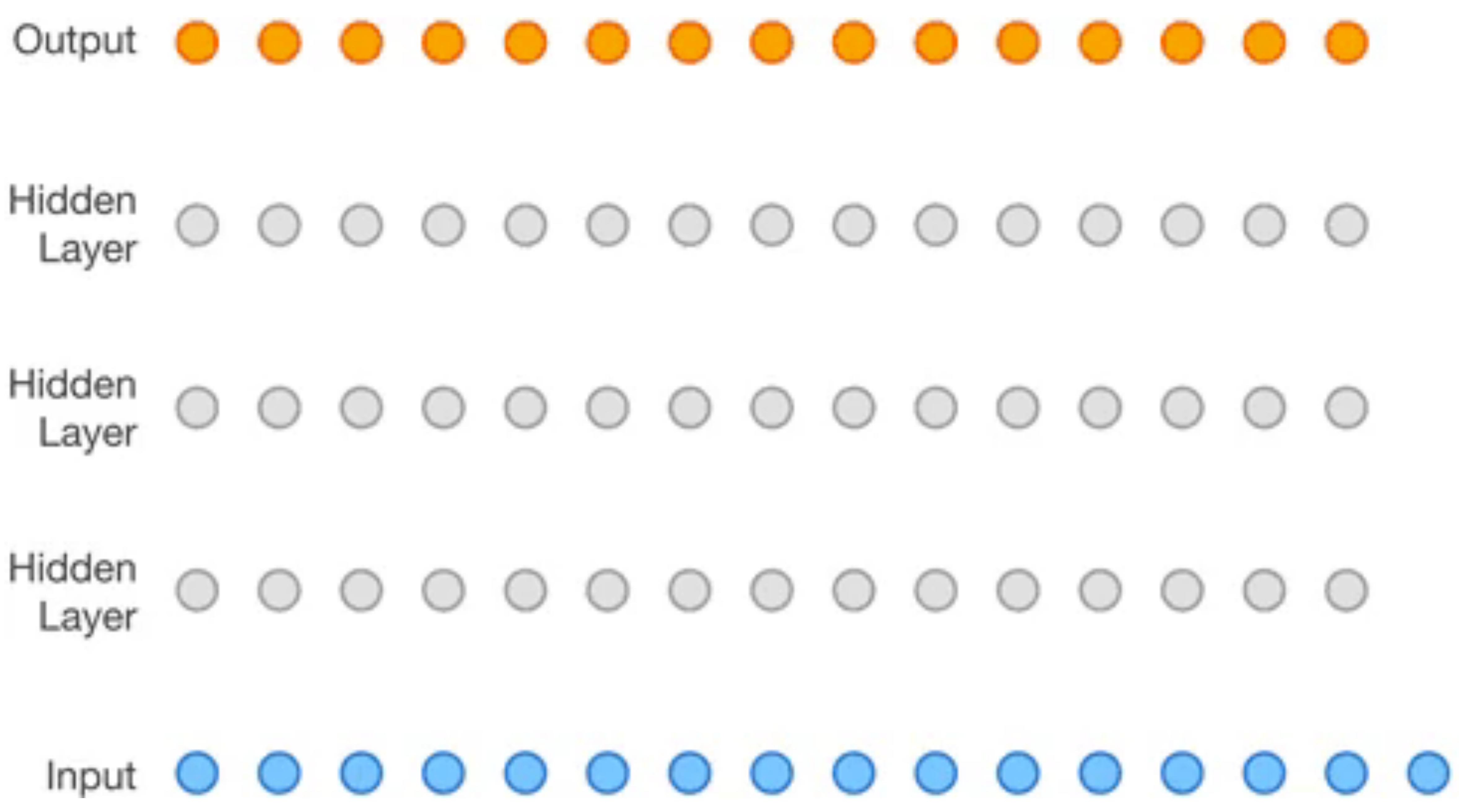
Convolutional Neural Networks (CNNs)



All animations are from: https://github.com/vdumoulin/conv_arithmetic

Conditional Wavenet

- Condition the model on input variables to generate audio with the required characteristics
- Global (same representation used to influence all timesteps)
- Local (use a second timeseries for conditioning)



Gif from https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Tacotron

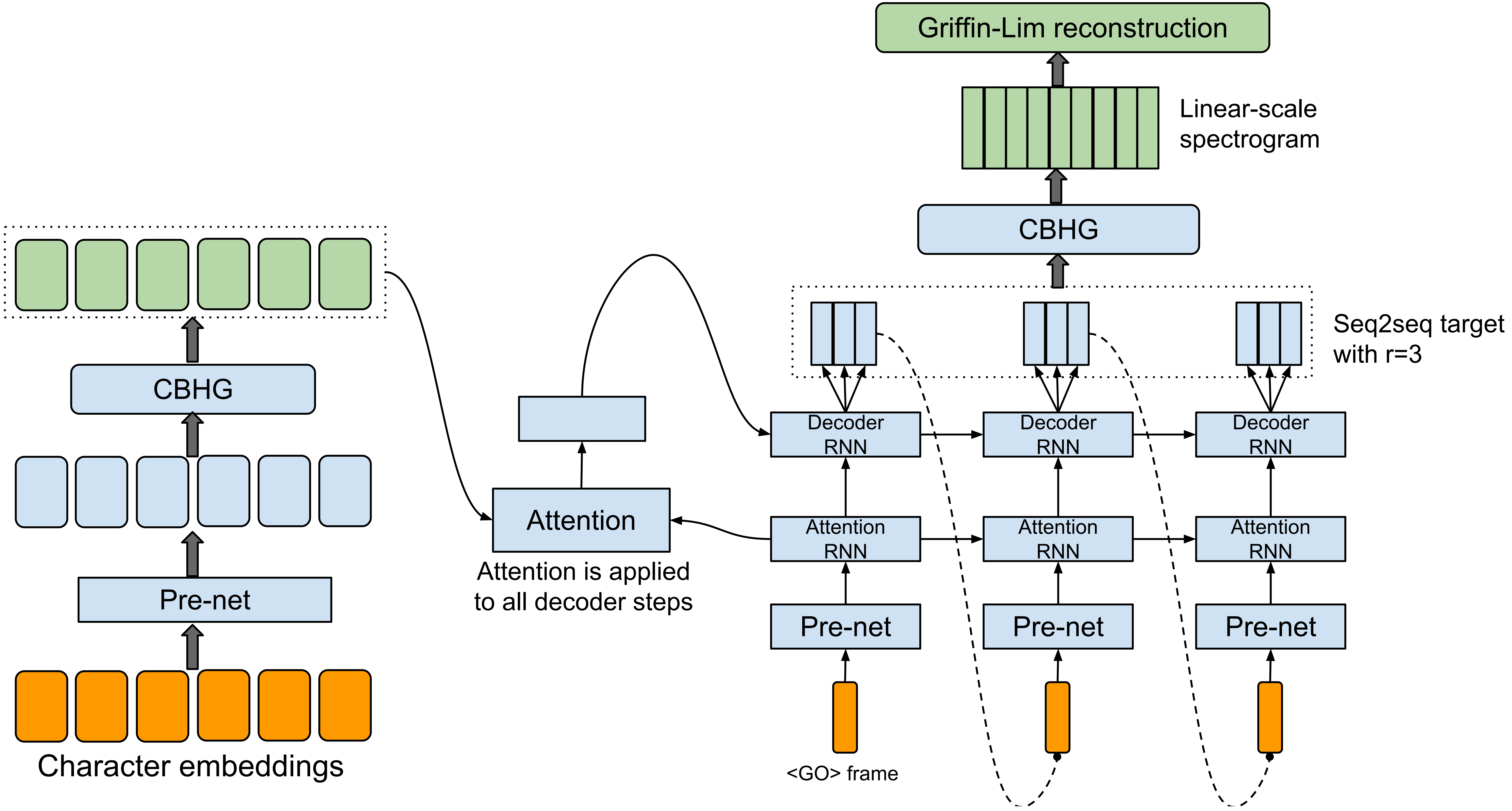


Image from Wang et al., “Tacotron: Towards end-to-end speech synthesis”, 2017. “https://arxiv.org/pdf/1703.10135.pdf

Tacotron: CBHG Module

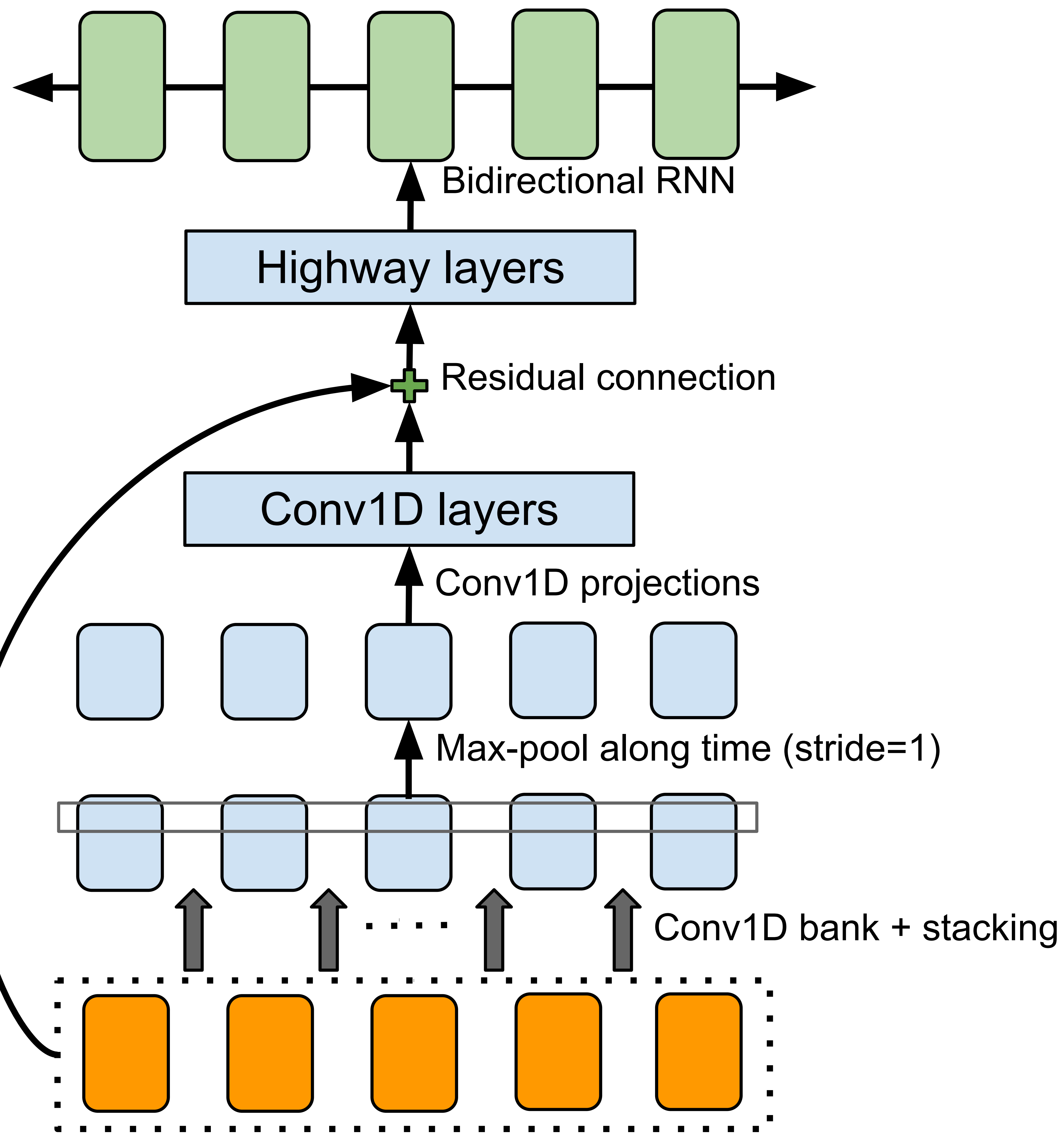


Image from Wang et al., "Tacotron: Towards end-to-end speech synthesis", 2017. "https://arxiv.org/pdf/1703.10135.pdf

Grapheme to phoneme (G2P) conversion

Grapheme to phoneme (G2P) conversion

- Produce a pronunciation (phoneme sequence) given a written word (grapheme sequence)
- Learn G2P mappings from a pronunciation dictionary
- Useful for:
    - ASR systems in languages with no pre-built lexicons
    - Speech synthesis systems
    - Deriving pronunciations for out-of-vocabulary (OOV) words

G2P conversion (I)

- One popular paradigm: Joint sequence models [BN12]
  - Grapheme and phoneme sequences are first aligned using EM-based algorithm
  - Results in a sequence of graphones (joint G-P tokens)
  - Ngram models trained on these graphone sequences
- WFST-based implementation of such a joint graphone model [Phonetisaurus]

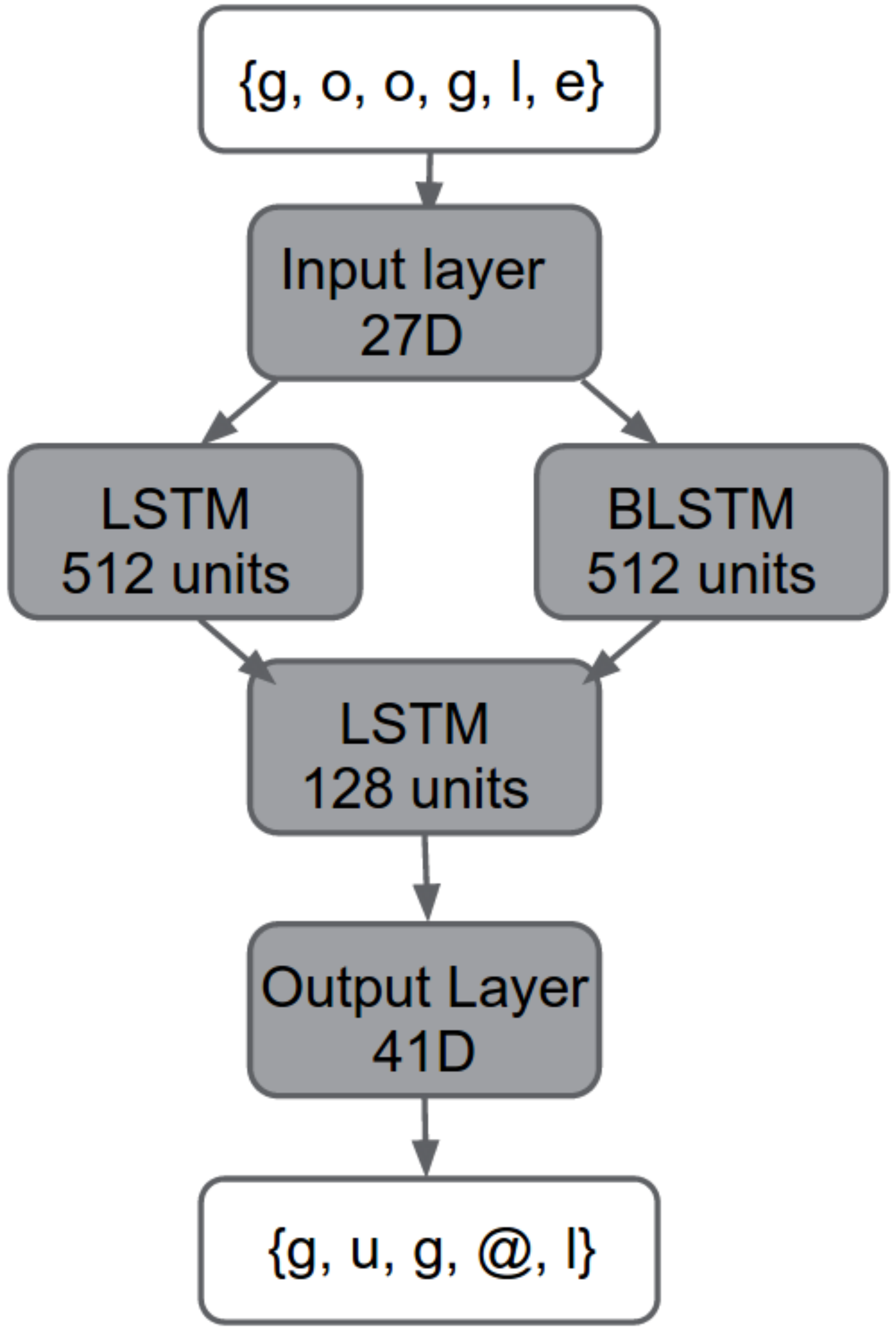
[BN12]:Bisani & Ney , “Joint sequence models for grapheme-to-phoneme conversion”,Specom 2012

[Phonetisaurus] J. Novak, Phonetisaurus Toolkit

G2P conversion (II)

- Neural network based methods are the new state-of-the-art for G2P
  - Bidirectional LSTM-based networks using a CTC output layer [Rao15]. Comparable to Ngram models.
  - Incorporate alignment information [Yao15]. Beats Ngram models.
  - No alignment. Encoder-decoder with attention. Beats the above systems [Toshniwal16].

LSTM + CTC for G2P conversion [Rao15]



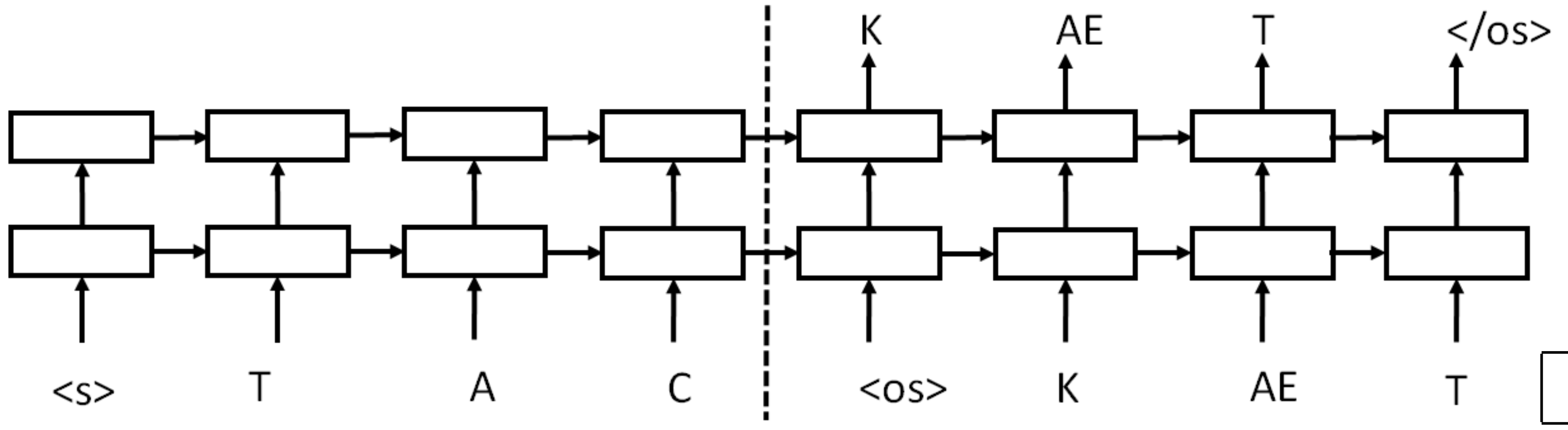
Model	Word Error Rate (%)
Galescu and Allen [4]	28.5
Chen [7]	24.7
Bisani and Ney [2]	24.5
Novak et al. [6]	24.4
Wu et al. [12]	23.4
5-gram FST	27.2
8-gram FST	26.5
Unidirectional LSTM with Full-delay	30.1
DBLSTM-CTC 128 Units	27.9
DBLSTM-CTC 512 Units	25.8
DBLSTM-CTC 512 + 5-gram FST	21.3

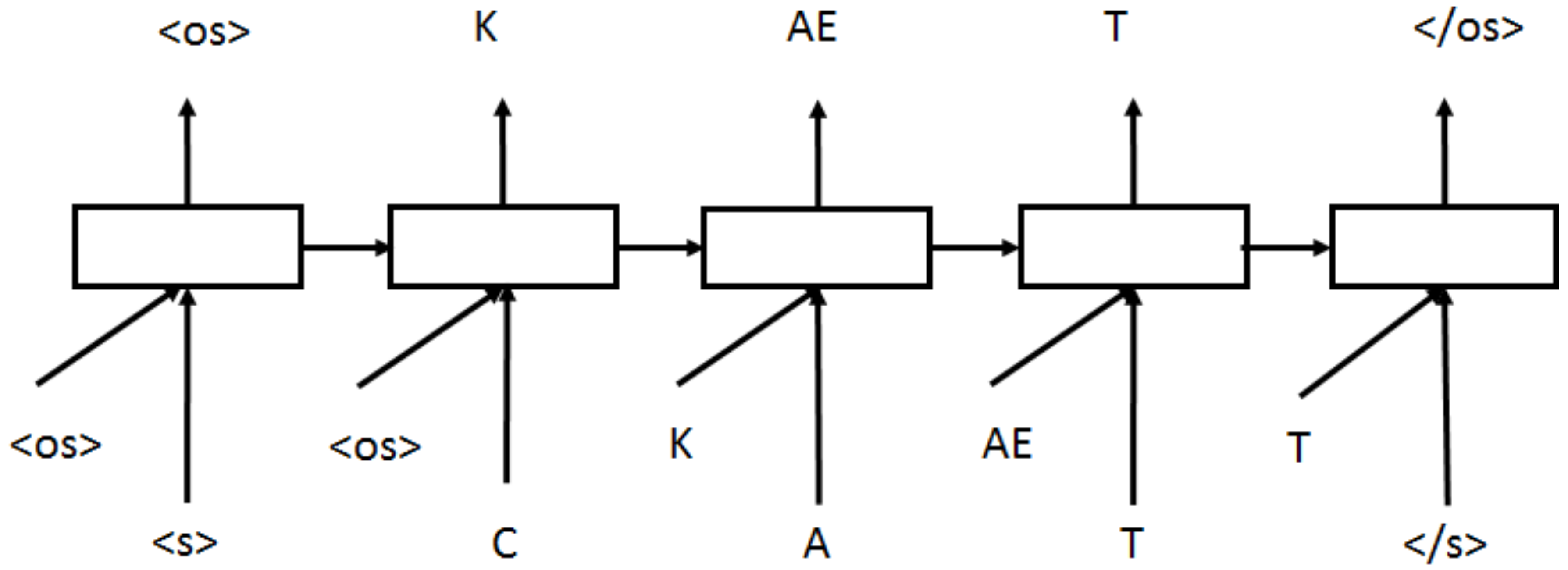
[Rao15] Grapheme-to-phoneme conversion using LSTM RNNs, ICASSP 2015

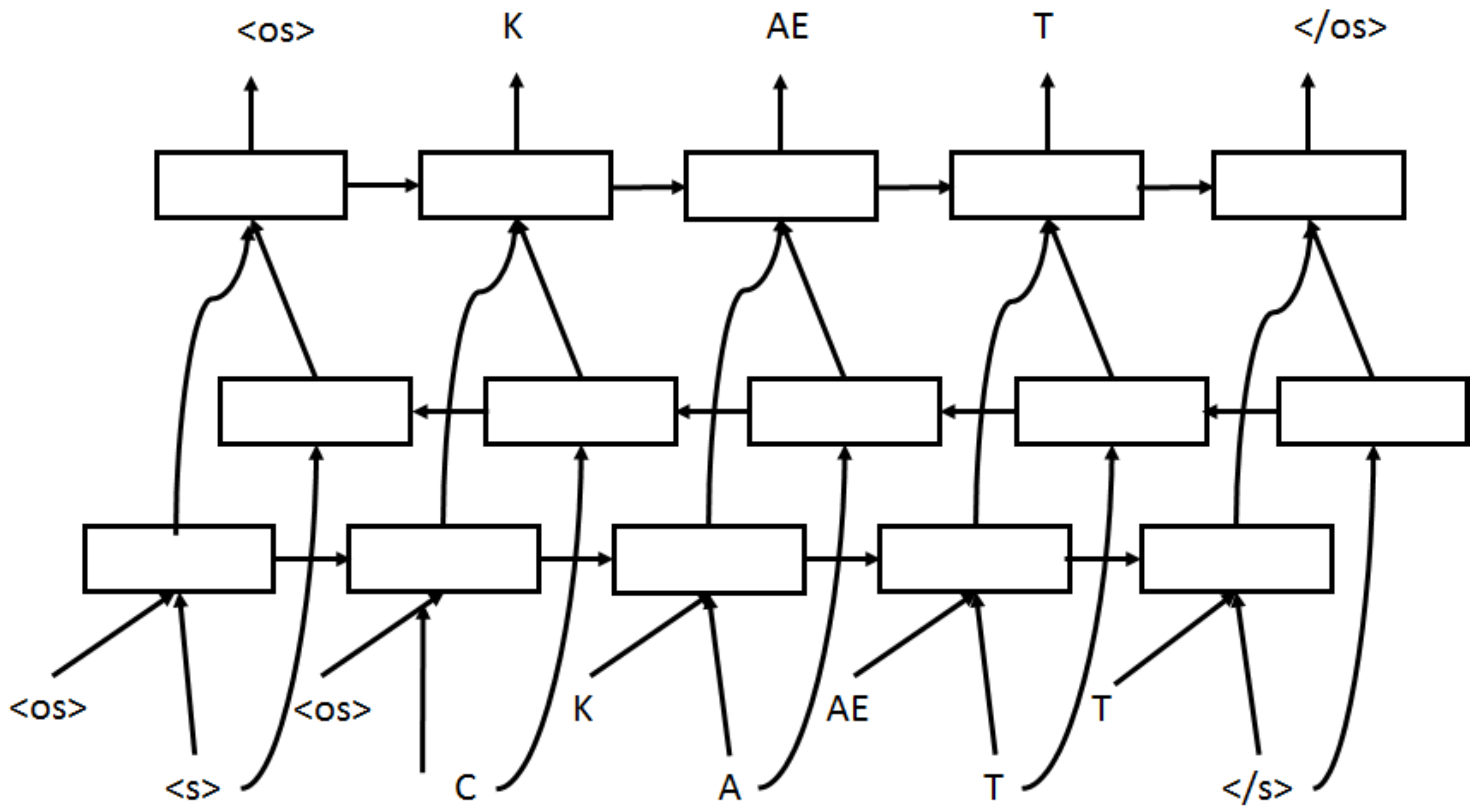
G2P conversion (II)

- Neural network based methods are the new state-of-the-art for G2P
    - Bidirectional LSTM-based networks using a CTC output layer [Rao15]. Comparable to Ngram models.
    - Incorporate alignment information [Yao15]. Beats Ngram models.
    - No alignment. Encoder-decoder with attention. Beats the above systems [Toshniwal16].

Seq2seq models
(with alignment information [Yao15])







Method	PER (%)	WER (%)
encoder-decoder LSTM	7.53	29.21
encoder-decoder LSTM (2 layers)	7.63	28.61
uni-directional LSTM	8.22	32.64
uni-directional LSTM (window size 6)	6.58	28.56
bi-directional LSTM	5.98	25.72
bi-directional LSTM (2 layers)	5.84	25.02
bi-directional LSTM (3 layers)	5.45	23.55

Data	Method	PER (%)	WER (%)
CMUDict	past results [20]	5.88	24.53
	bi-directional LSTM	5.45	23.55
NetTalk	past results [20]	8.26	33.67
	bi-directional LSTM	7.38	30.77
Pronlex	past results [20, 21]	6.78	27.33
	bi-directional LSTM	6.51	26.69

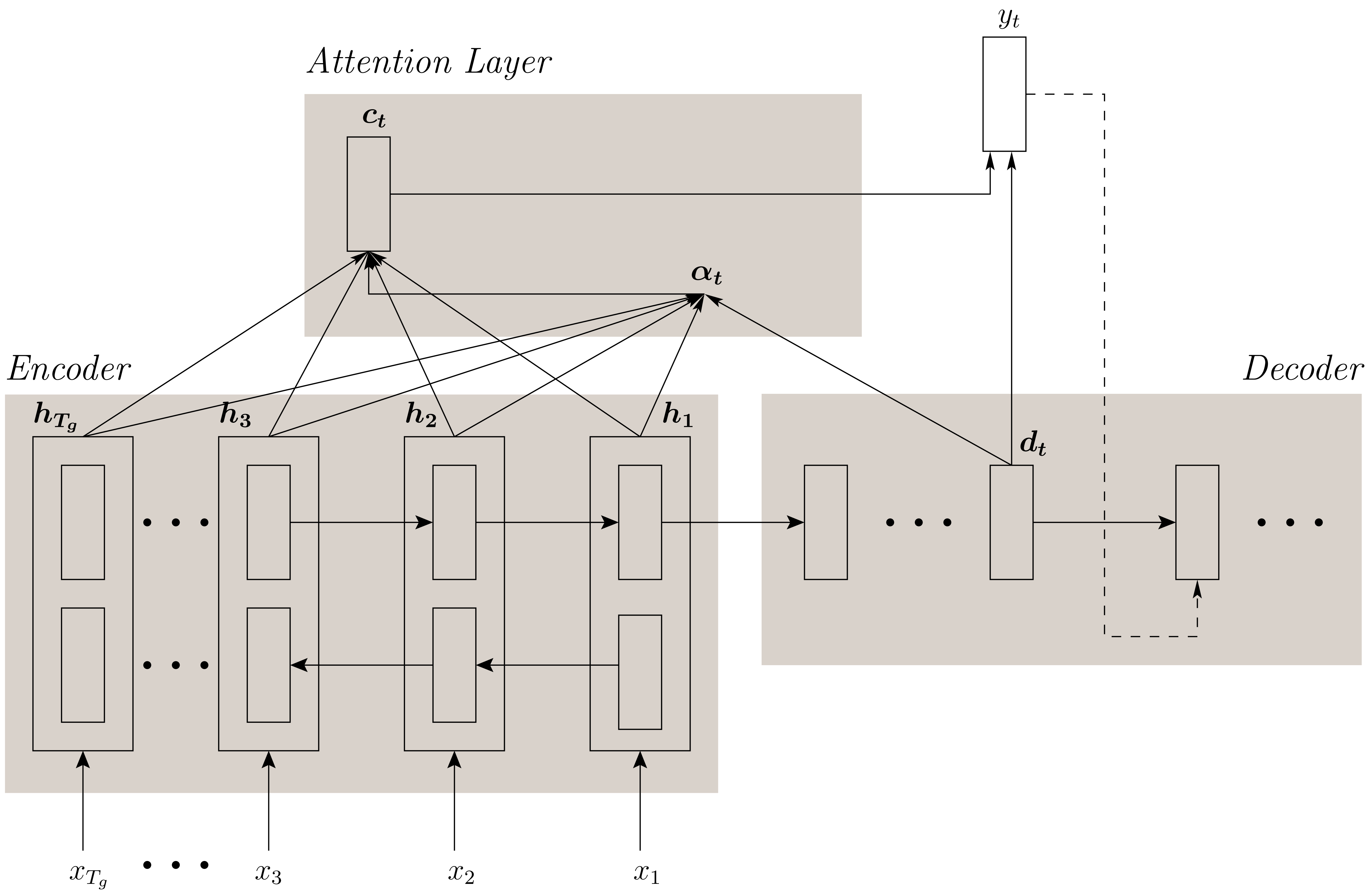
[Yao15] Sequence-to-sequence neural net models for G2P conversion, Interspeech 2015

G2P conversion (II)

- Neural network based methods are the new state-of-the-art for G2P
  - Bidirectional LSTM-based networks using a CTC output layer [Rao15]. Comparable to Ngram models.
  - Incorporate alignment information [Yao15]. Beats Ngram models.
  - No alignment. Encoder-decoder with attention. Beats the above systems [Toshniwal16].

[Rao15] Grapheme-to-phoneme conversion using LSTM RNNs, ICASSP 2015
[Yao15] Sequence-to-sequence neural net models for G2P conversion, Interspeech 2015
[Toshniwal16] Jointly learning to align and convert graphemes to phonemes with neural attention models, SLT 2016.

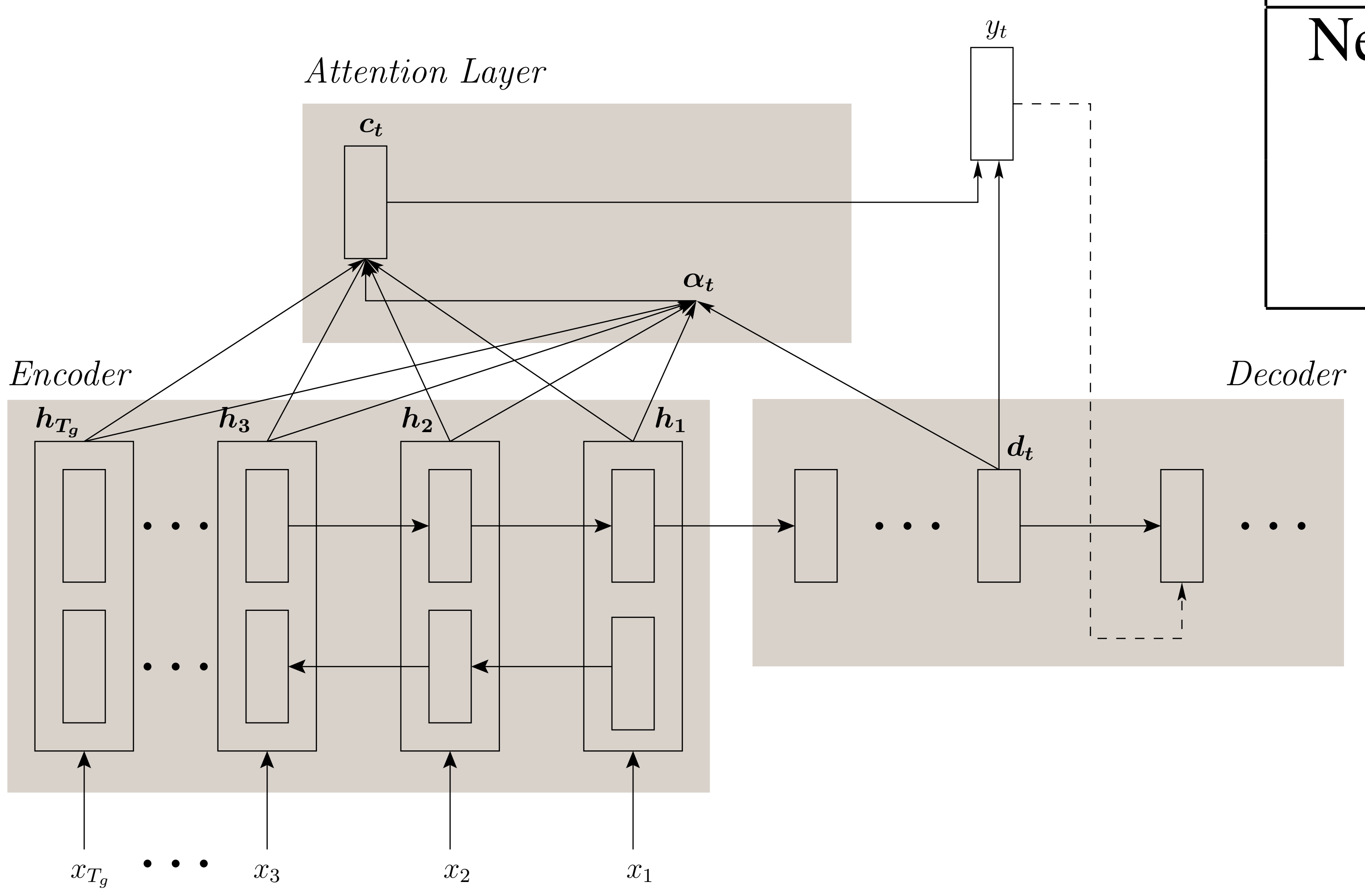
Encoder-decoder + attention for G2P [Toshniwal16]



[Toshniwal16] Jointly learning to align and convert graphemes to phonemes with neural attention models, SLT 2016.

Encoder-decoder + attention for G2P [Toshniwal16]

Data	Method	PER (%)
CMUDict	BiDir LSTM + Alignment [6]	5.45
	DBLSTM-CTC [5]	-
	DBLSTM-CTC + 5-gram model [5]	-
	Encoder-decoder + global attn	5.04 ± 0.03
	Encoder-decoder + local-m attn	5.11 ± 0.03
	Encoder-decoder + local-p attn	5.39 ± 0.04
	Ensemble of 5 [Encoder-decoder + global attn] models	$\mathbf{4.69}$
Pronlex	BiDir LSTM + Alignment [6]	6.51
	Encoder-decoder + global attn	6.24 ± 0.1
	Encoder-decoder + local-m attn	$\mathbf{5.99 \pm 0.11}$
	Encoder-decoder + local-p attn	6.49 ± 0.06
NetTalk	BiDir LSTM + Alignment [6]	7.38
	Encoder-decoder + global attn	$\mathbf{7.14 \pm 0.72}$
	Encoder-decoder + local-m attn	7.13 ± 0.11
	Encoder-decoder + local-p attn	8.41 ± 0.19



[Toshniwal16] Jointly learning to align and convert graphemes to phonemes with neural attention models, SLT 2016.