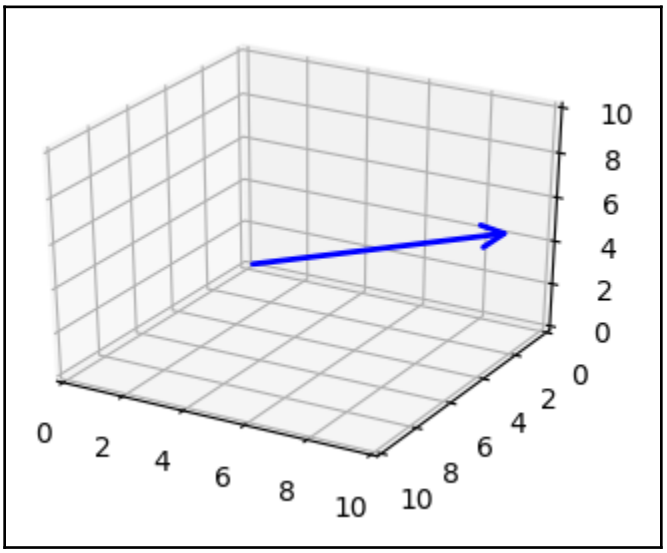
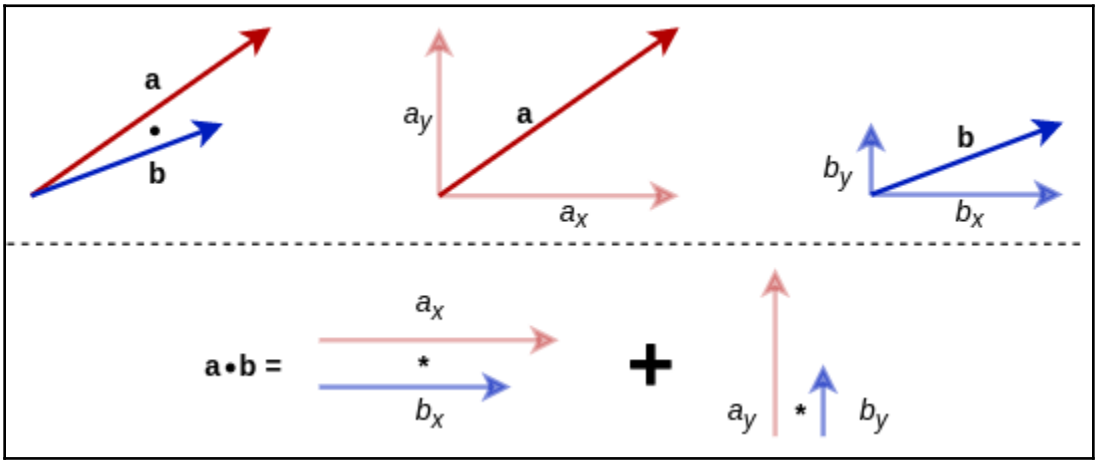
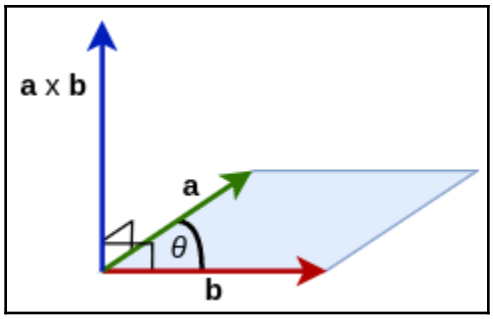
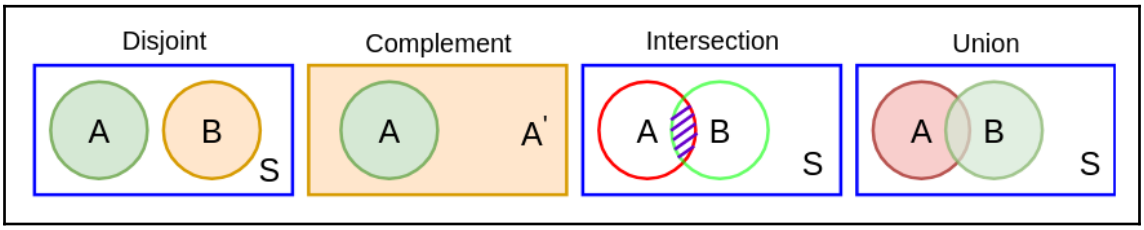
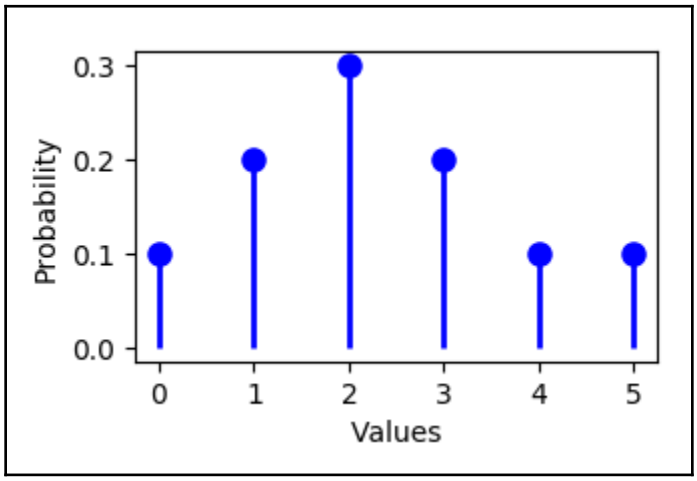
Chapter 1: The Nuts and Bolts of Neural Networks

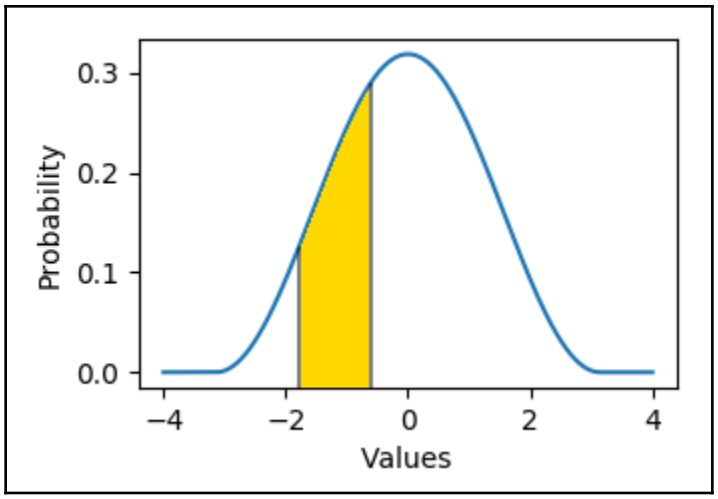


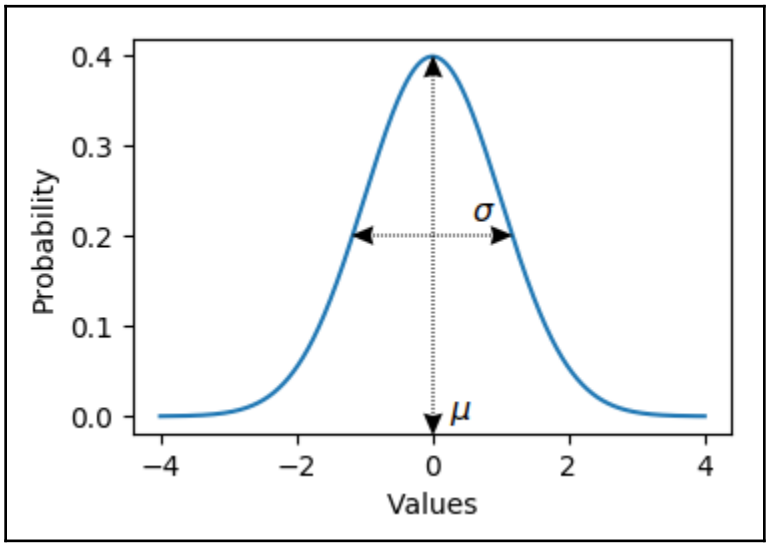


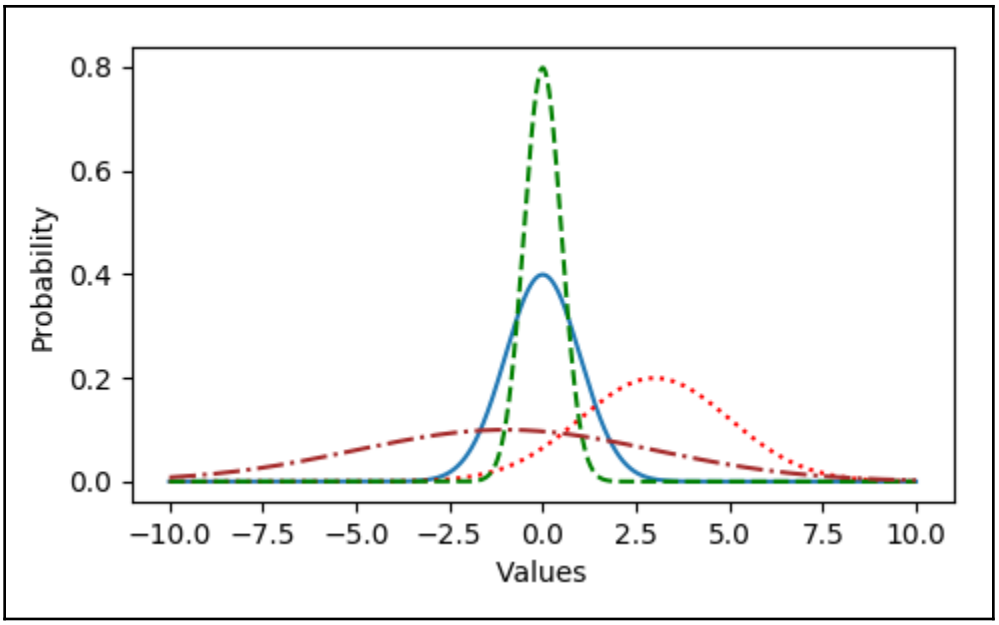

a x b
a
θ
b

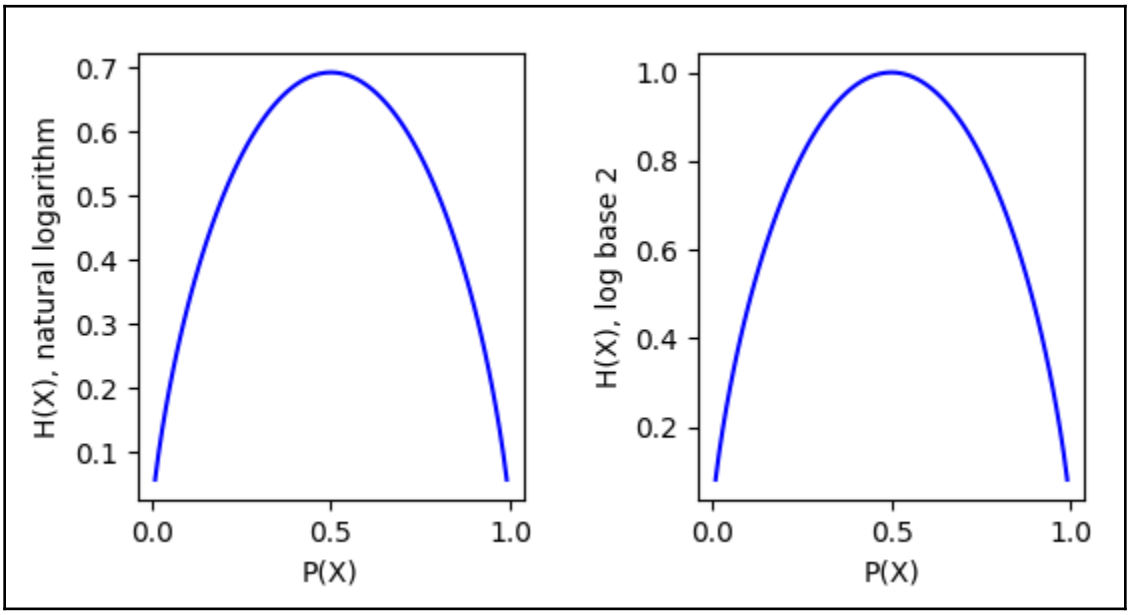

Disjoint
A
B
S
Complement
A
A'
Intersection
A
B
S
Union
A
B
S

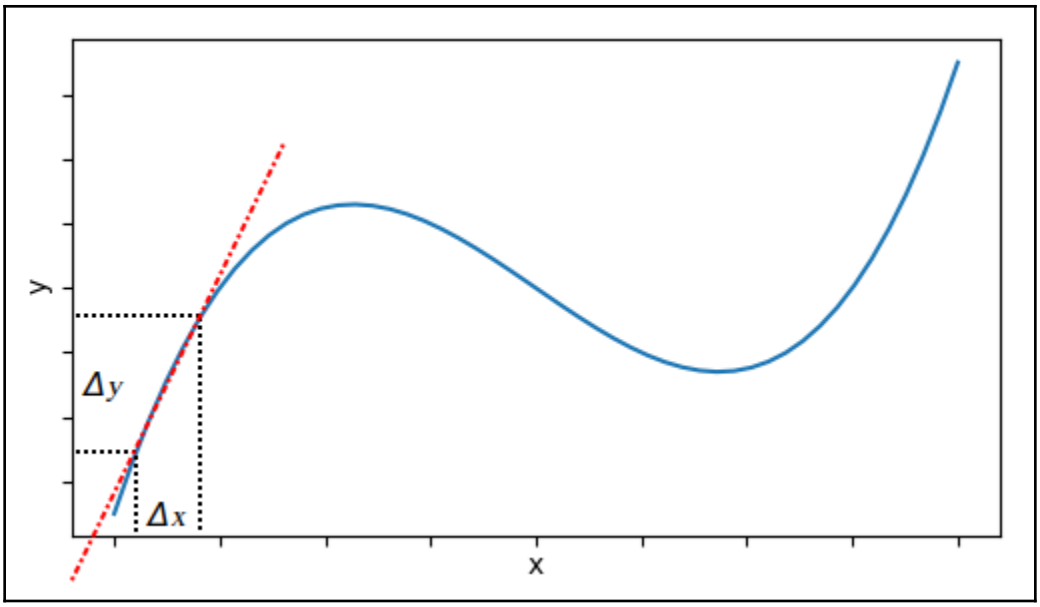

0.3
0.2
0.1
0.0
Probability
0
1
2
3
4
5
Values

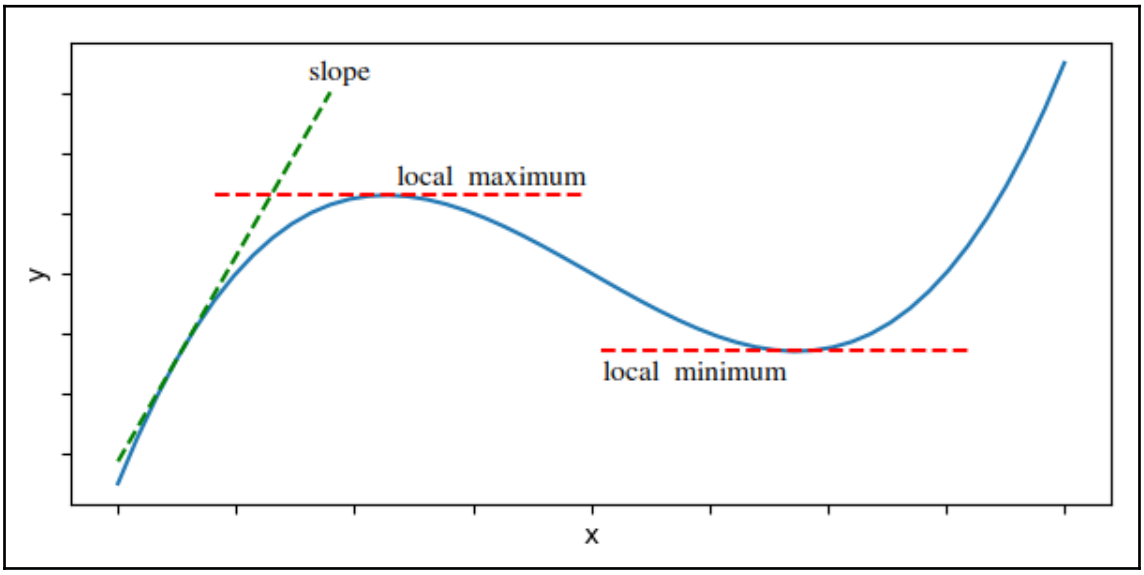

0.3
0.2
0.1
0.0
Probability
−4
−2
0
2
4
Values

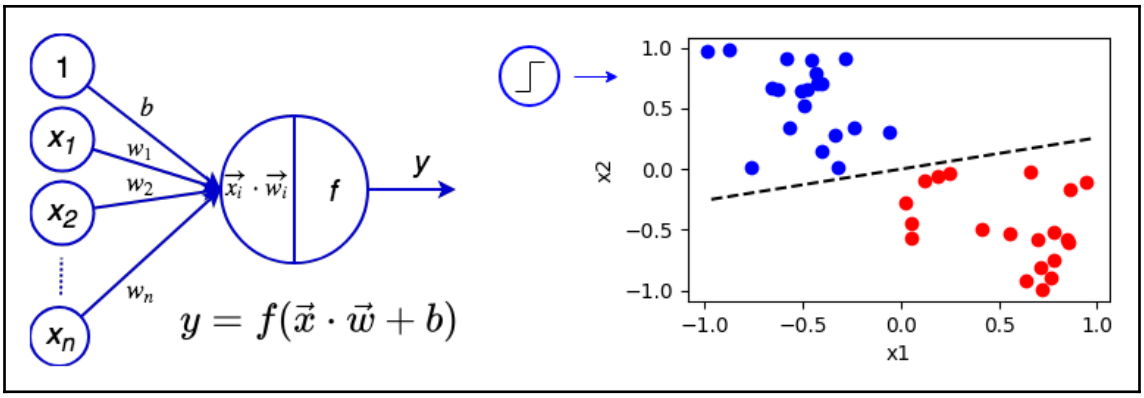

0.4
0.3
0.2
0.1
0.0
Probability
σ
μ
−4
−2
0
2
4
Values

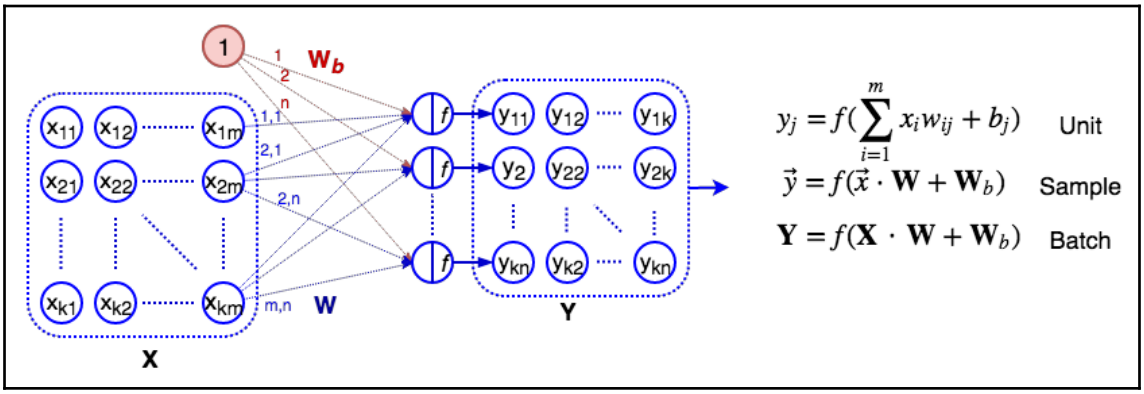

Probability
0.8
0.6
0.4
0.2
0.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
Values

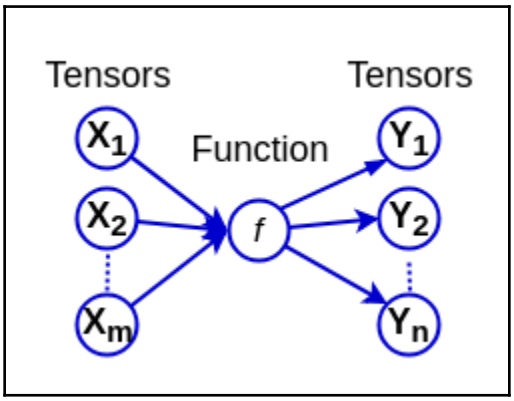

H(X), natural logarithm
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.5
1.0
P(X)
H(X), log base 2
1.0
0.8
0.6
0.4
0.2
0.0
0.5
1.0
P(X)

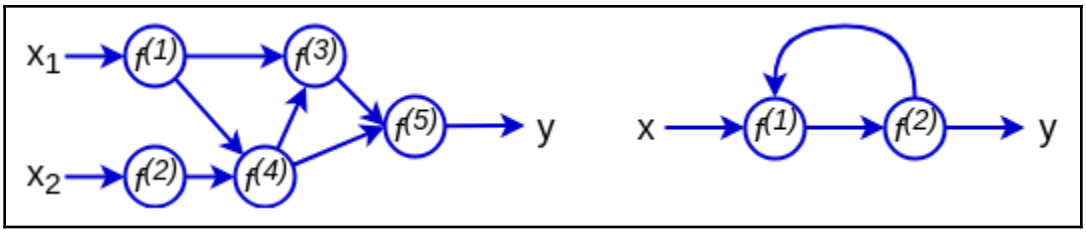

y
Δy
Δx
x

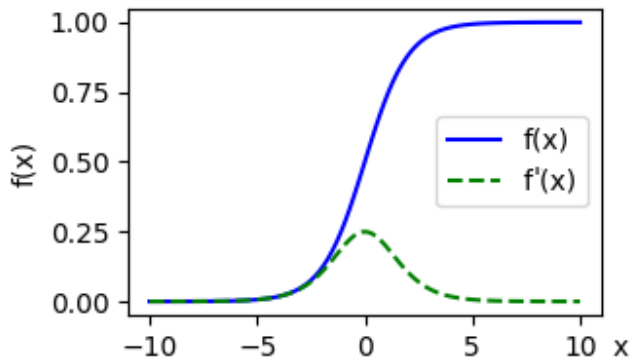

slope
local maximum
local minimum
y
x


1
b
x1
w1
x2
w2
xn
wn
f
y
$y = f(\vec{x} \cdot \vec{w} + b)$
1.0
0.5
0.0
−0.5
−1.0
x2
x1


1
Wb
X
W
Y
$y_j = f(\sum_{i=1}^{m} x_i w_{ij} + b_j)$ Unit
$\vec{y} = f(\vec{x} \cdot \mathbf{W} + \mathbf{W}_b)$ Sample
$\mathbf{Y} = f(\mathbf{X} \cdot \mathbf{W} + \mathbf{W}_b)$ Batch

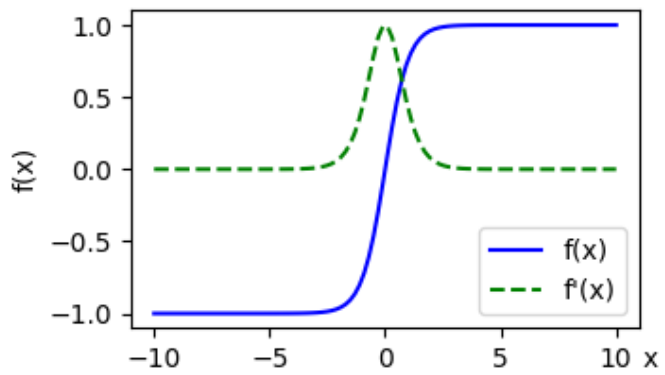

Tensors
Tensors
Function
f





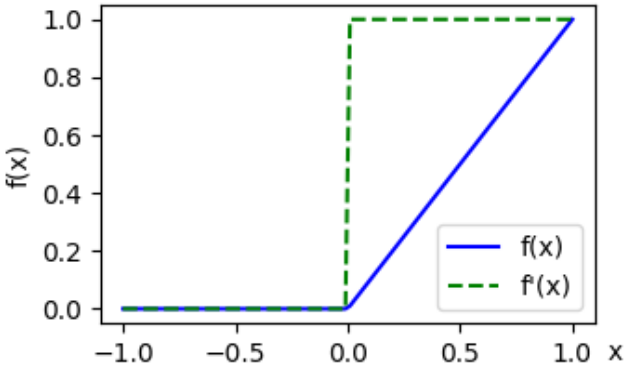
$$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$$

$$f'(x) = \sigma(x)(1-\sigma(x))$$



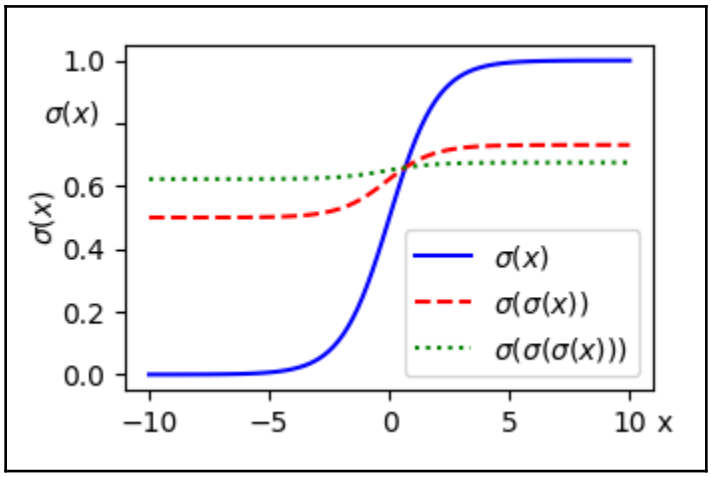
$$f(x) = tanh(x)$$

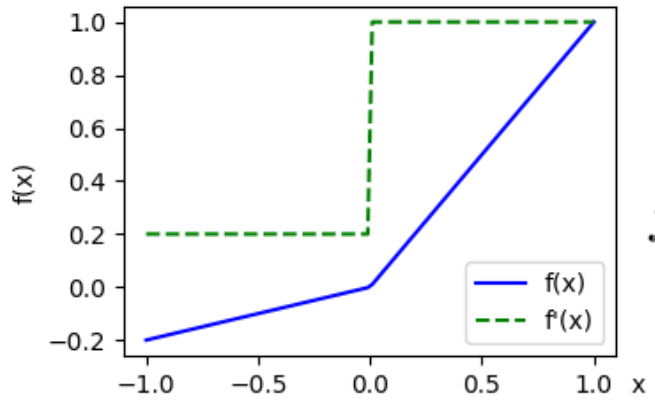
$$f'(x) = 1 - tanh(x)^2$$



$$f(x) = \begin{cases} x \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases}$$

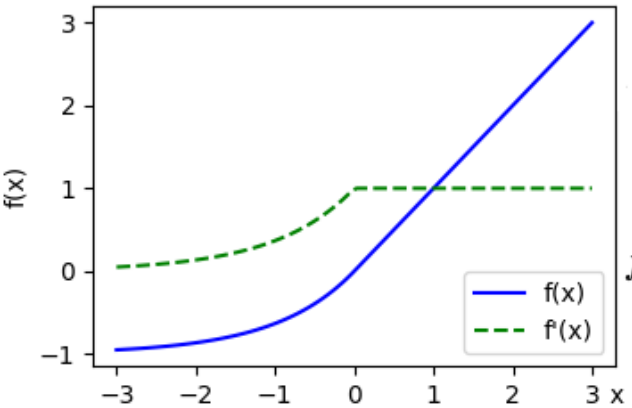
$$f'(x) = \begin{cases} 1 \text{ if } x > 0 \\ 0 \text{ if } x < 0 \end{cases}$$





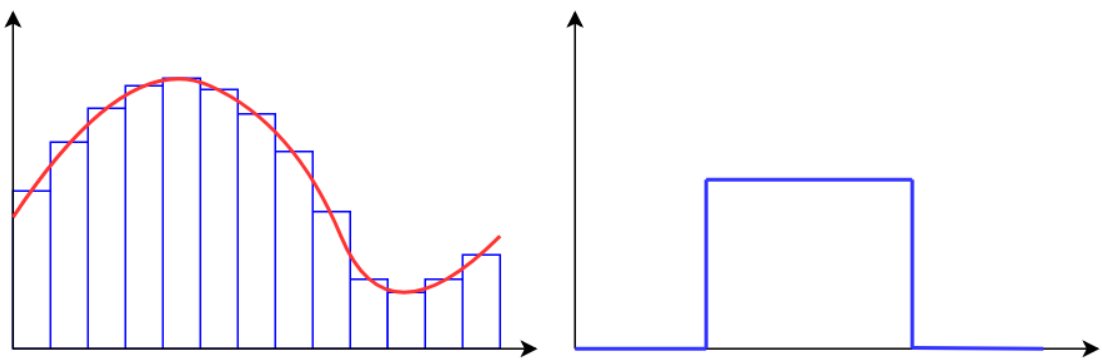
$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases}$$

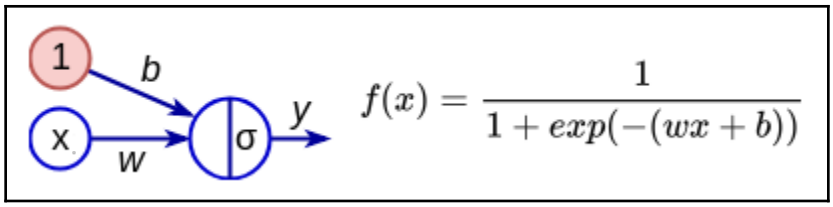
$$f'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ \alpha & \text{if } x < 0 \end{cases}$$

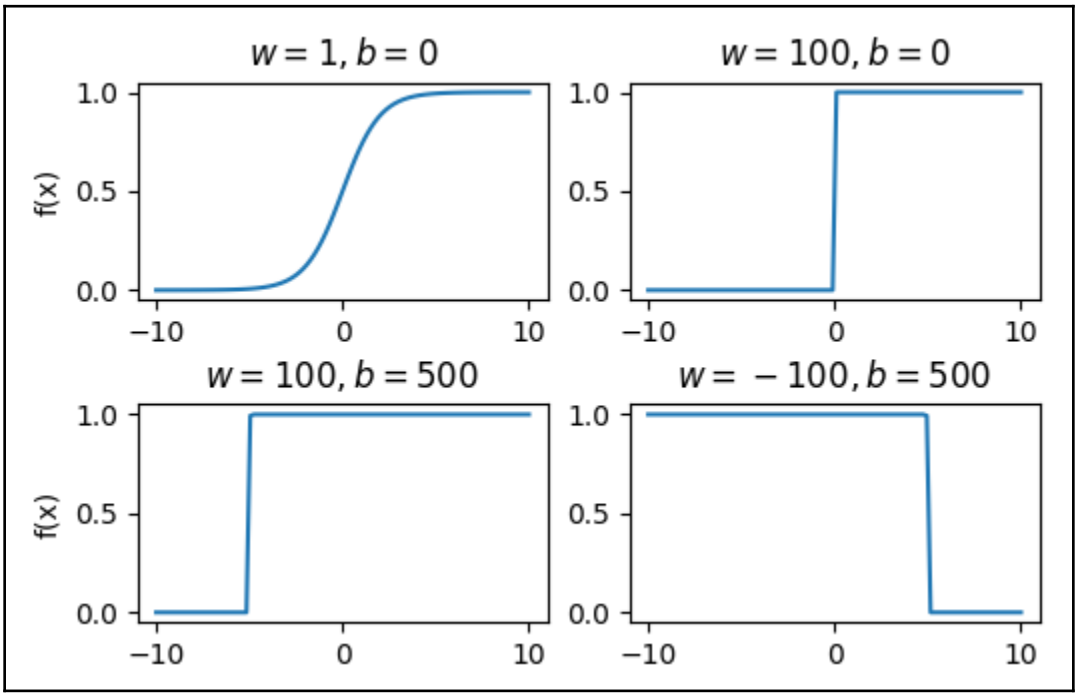


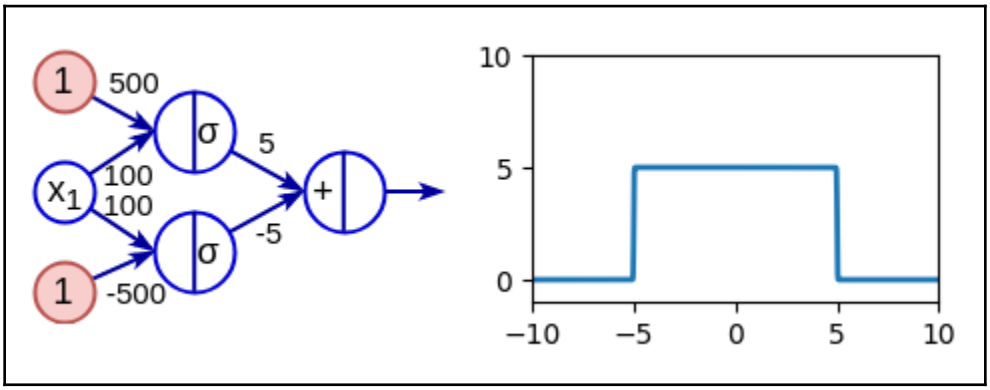
$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases}$$

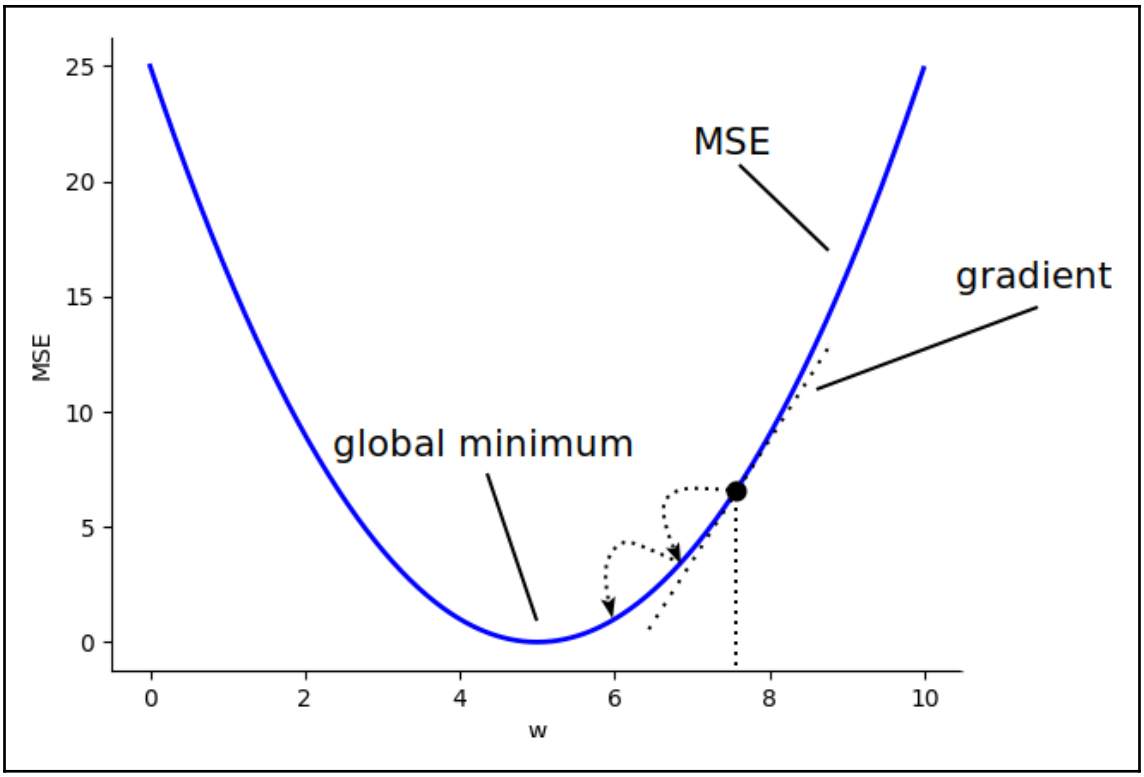
$$f'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ \alpha e^x & \text{if } x < 0 \end{cases}$$

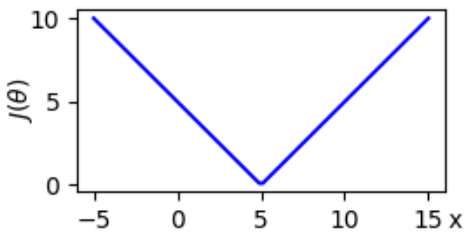



1
b
x
w
σ
y
$f(x) = \frac{1}{1 + exp(-(wx + b))}$

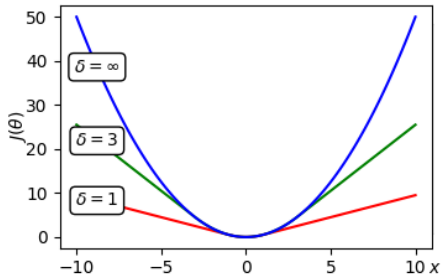

w = 1, b = 0
w = 100, b = 0
w = 100, b = 500
w = −100, b = 500
f(x)
1.0
0.5
0.0
−10
0
10


1
500
σ
5
x1
100
100
+
σ
-5
1
-500
10
5
0
−10
−5
0
5
10

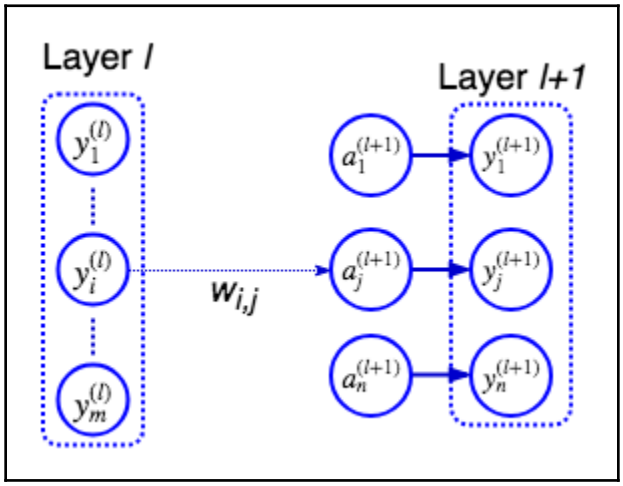




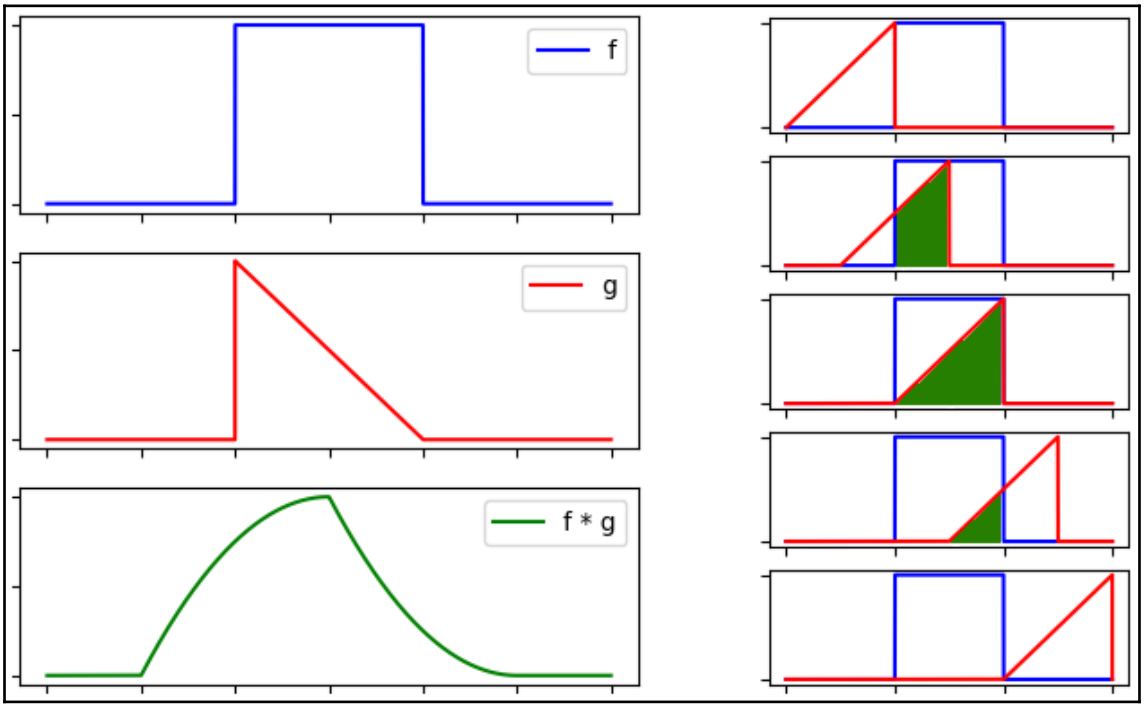
$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} |f(\vec{x}^{(i)}; \theta) - t^{(i)}|$$

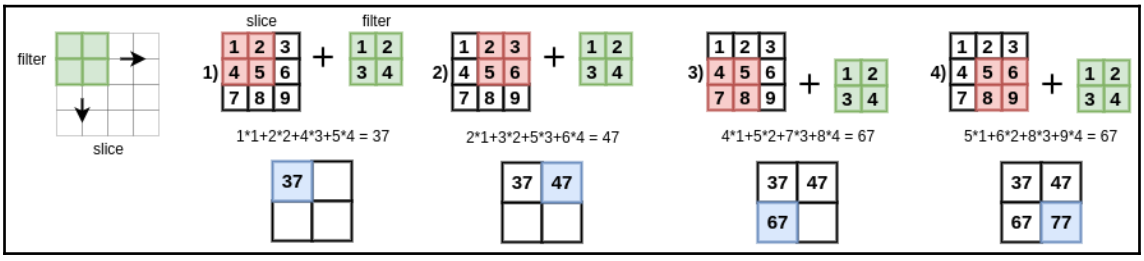


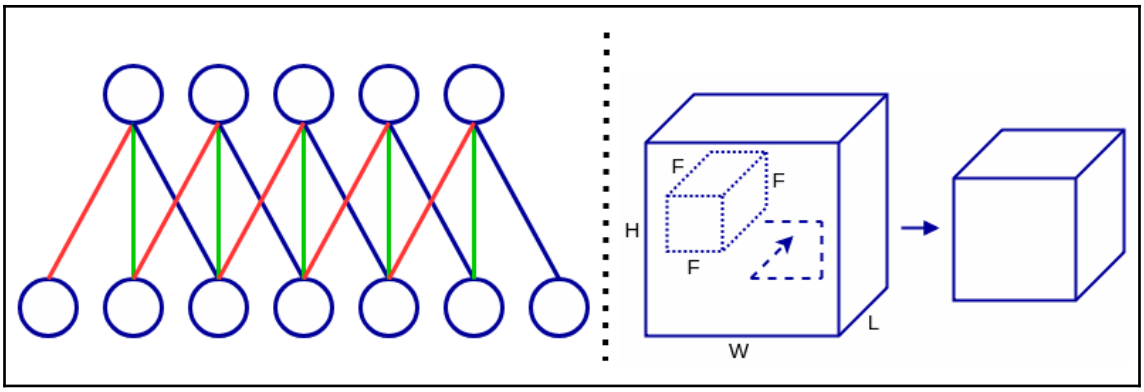
$$J(\theta) = \begin{cases} \frac{1}{2}(f(\vec{x}^{(i)};\theta) - t^{(i)})^2 & \text{if } |f(\vec{x}^{(i)};\theta) - t^{(i)}| \leq \delta \\ \delta|f(\vec{x}^{(i)};\theta) - t^{(i)}| - \frac{1}{1}\delta^2 & \text{otherwise} \end{cases}$$

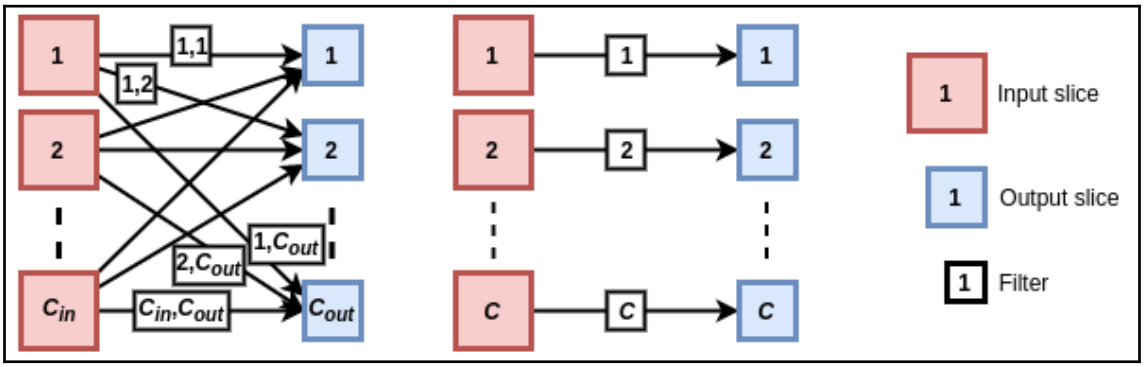


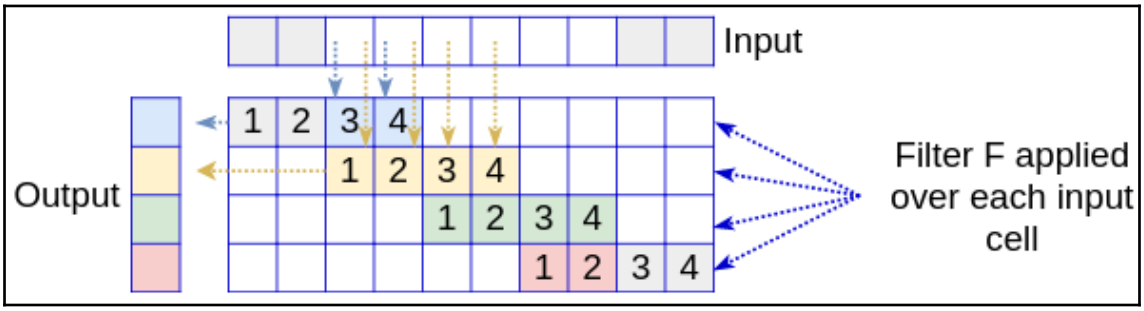
Chapter 2: Understanding Convolutional Networks

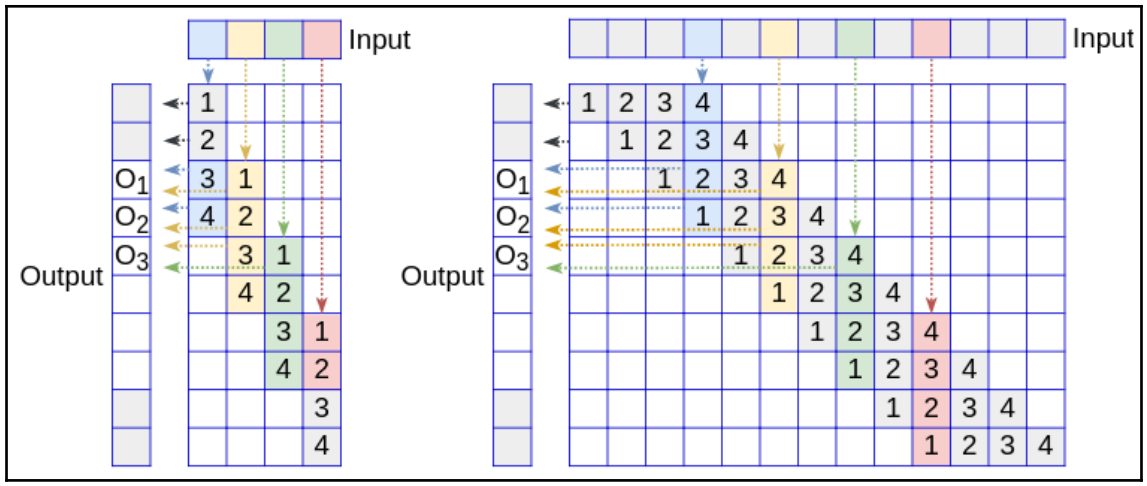


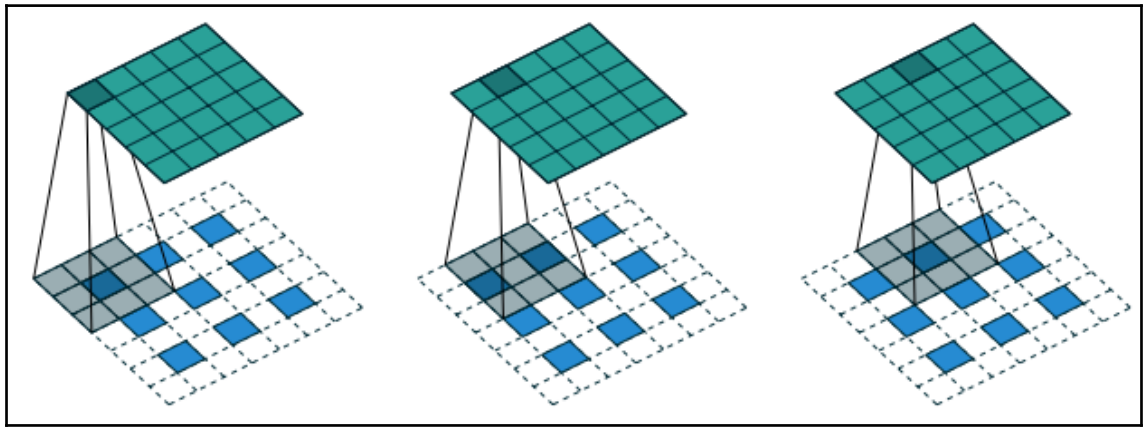


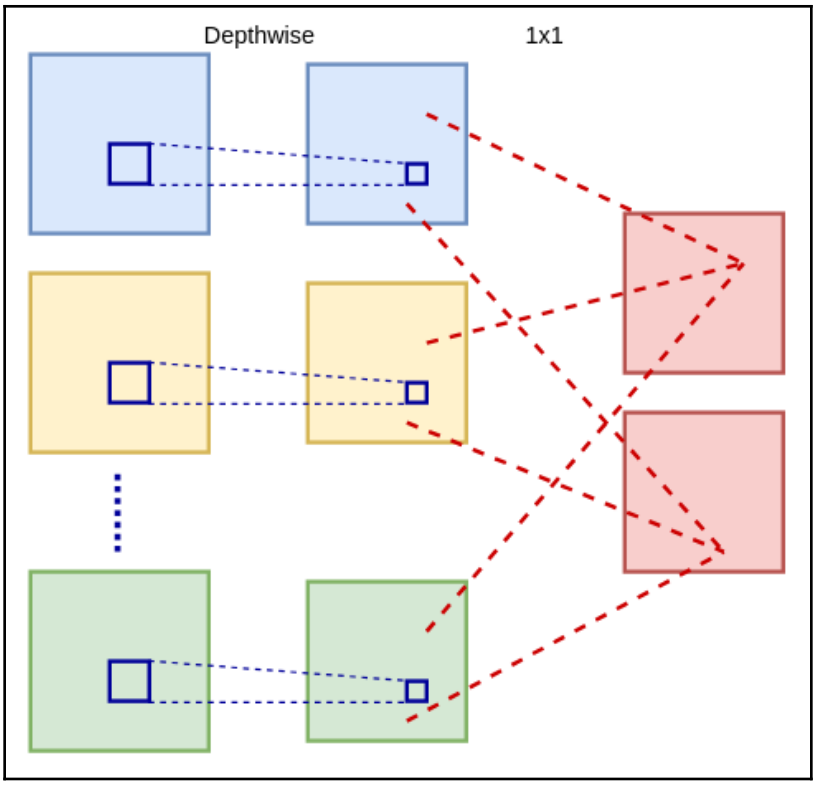

F
F
F
H
W
L

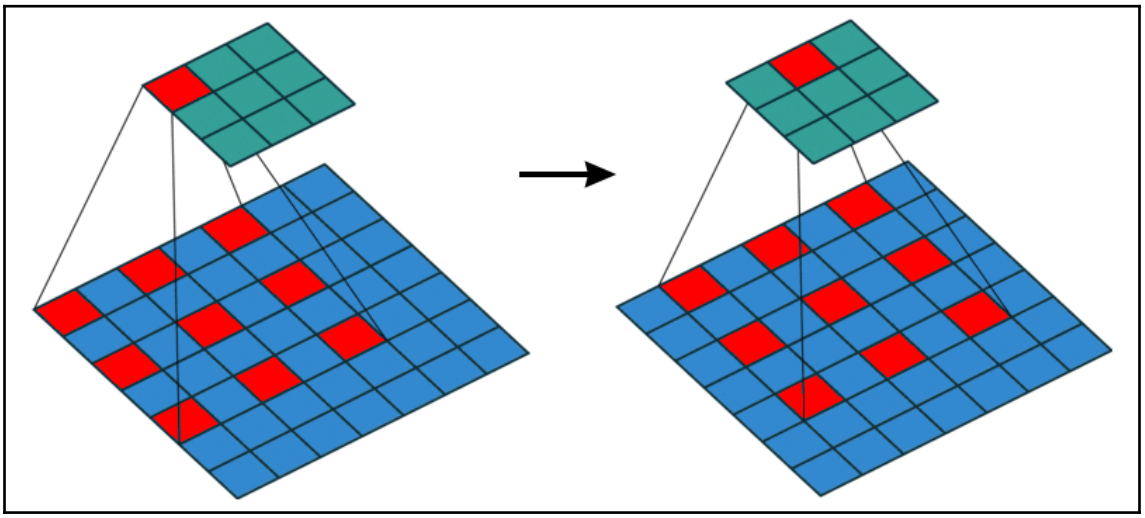

1
2
C_{in}
1,1
1,2
2,C_{out}
1,C_{out}
C_{in},C_{out}
C_{out}
1
2
C
Input slice
Output slice
Filter

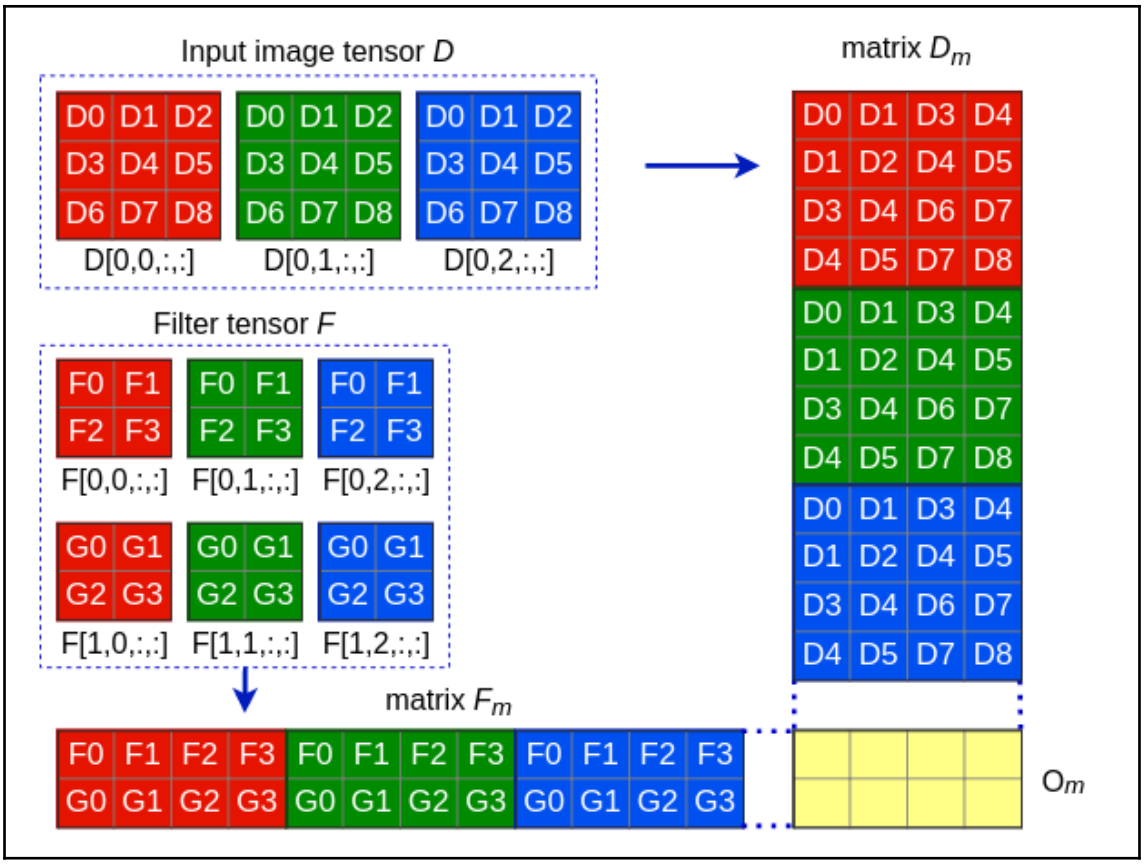

Input
Output
Filter F applied over each input cell

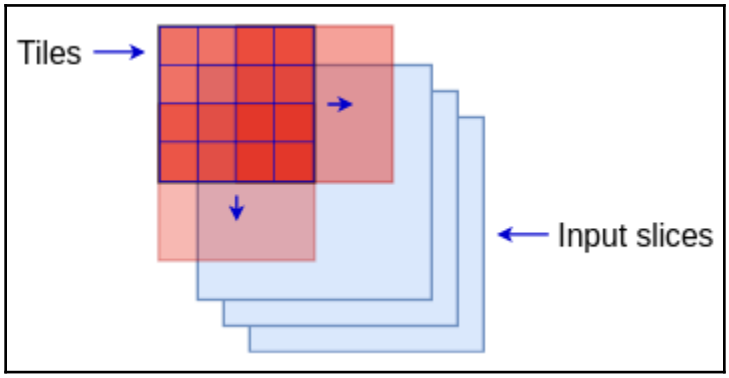

Input
Output
O1
O2
O3
Input
Output
O1
O2
O3

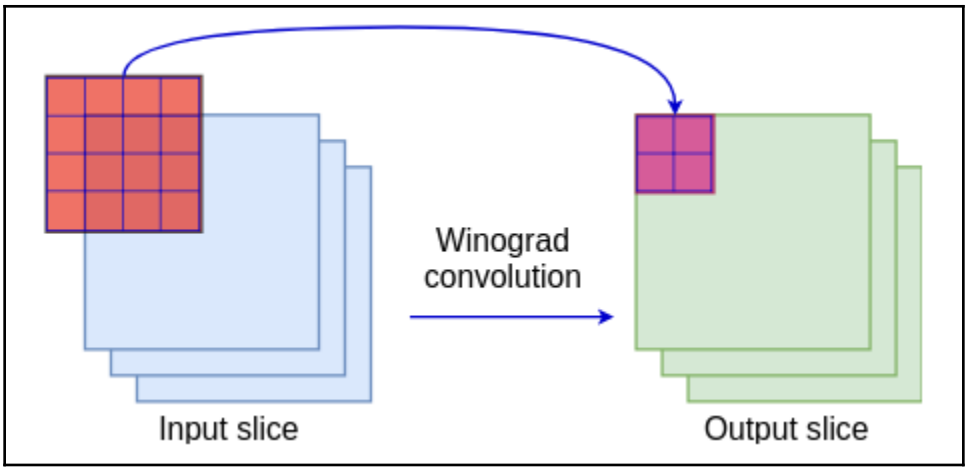


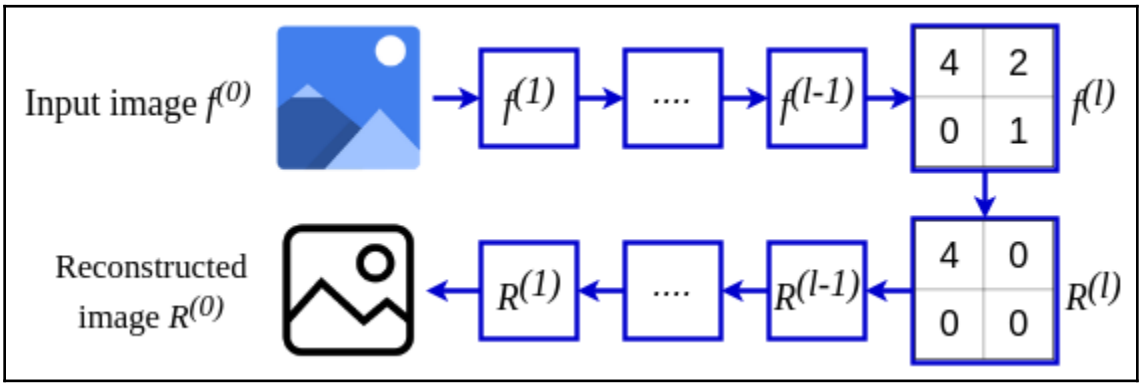

Depthwise
1x1

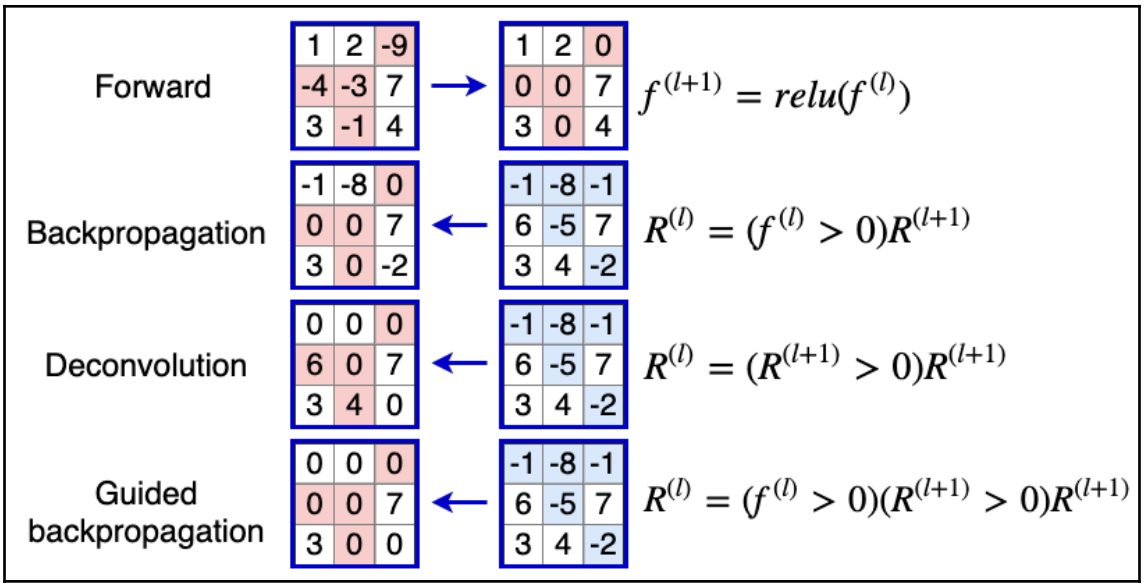


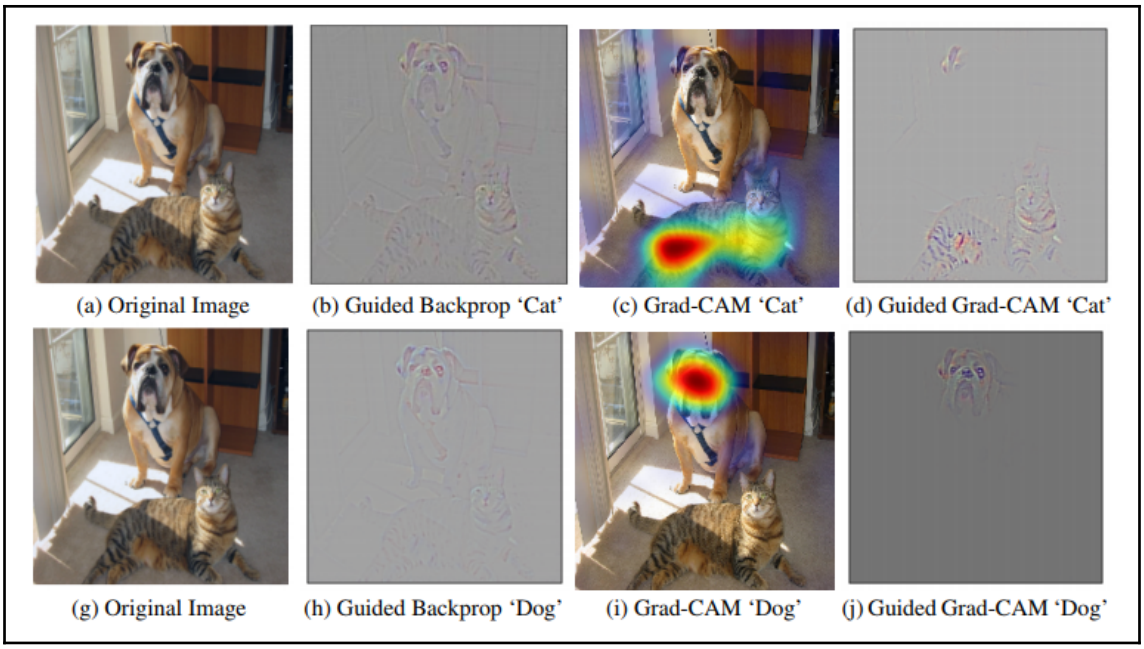

Input image tensor D
D0 D1 D2
D3 D4 D5
D6 D7 D8
D[0,0,:,:]
D[0,1,:,:]
D[0,2,:,:]
matrix D_m
D0 D1 D3 D4
D1 D2 D4 D5
D3 D4 D6 D7
D4 D5 D7 D8
Filter tensor F
F0 F1
F2 F3
F[0,0,:,:]
F[0,1,:,:]
F[0,2,:,:]
G0 G1
G2 G3
F[1,0,:,:]
F[1,1,:,:]
F[1,2,:,:]
matrix F_m
F0 F1 F2 F3 F0 F1 F2 F3 F0 F1 F2 F3
G0 G1 G2 G3 G0 G1 G2 G3 G0 G1 G2 G3
O_m


Tiles
Input slices


Winograd
convolution
Input slice
Output slice

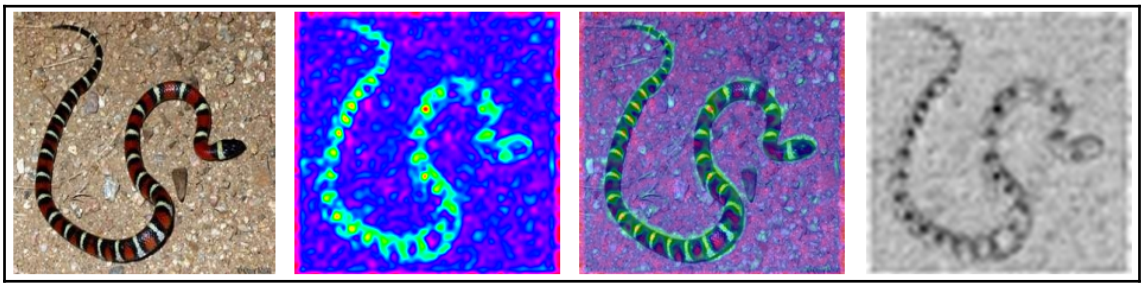

Input image $f^{(0)}$
$f^{(1)}$
....
$f^{(l-1)}$
4 2
0 1
$f^{(l)}$
Reconstructed
image $R^{(0)}$
$R^{(1)}$
....
$R^{(l-1)}$
4 0
0 0
$R^{(l)}$

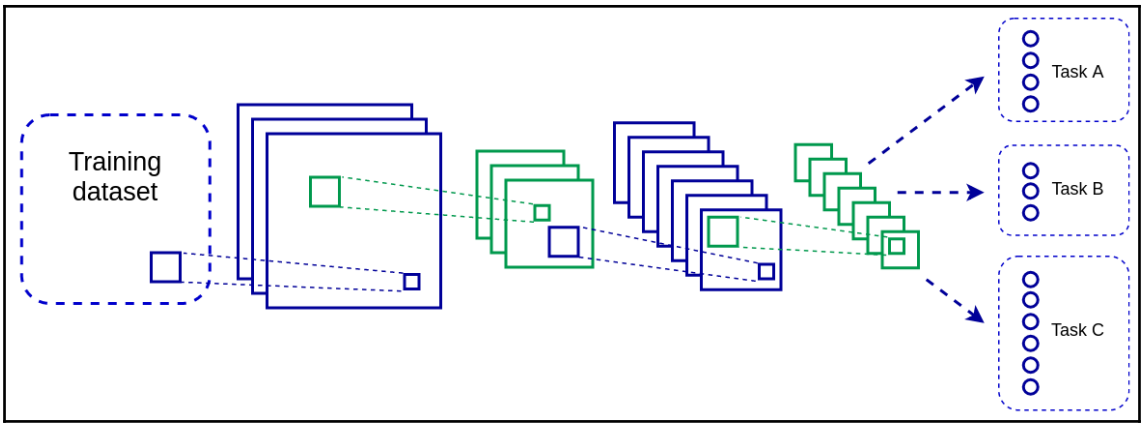

Forward
1 2 -9
-4 -3 7
3 -1 4
1 2 0
0 0 7
3 0 4
$f^{(l+1)} = relu(f^{(l)})$
Backpropagation
-1 -8 0
0 0 7
3 0 -2
-1 -8 -1
6 -5 7
3 4 -2
$R^{(l)} = (f^{(l)} > 0)R^{(l+1)}$
Deconvolution
0 0 0
6 0 7
3 4 0
-1 -8 -1
6 -5 7
3 4 -2
$R^{(l)} = (R^{(l+1)} > 0)R^{(l+1)}$
Guided backpropagation
0 0 0
0 0 7
3 0 0
-1 -8 -1
6 -5 7
3 4 -2
$R^{(l)} = (f^{(l)} > 0)(R^{(l+1)} > 0)R^{(l+1)}$

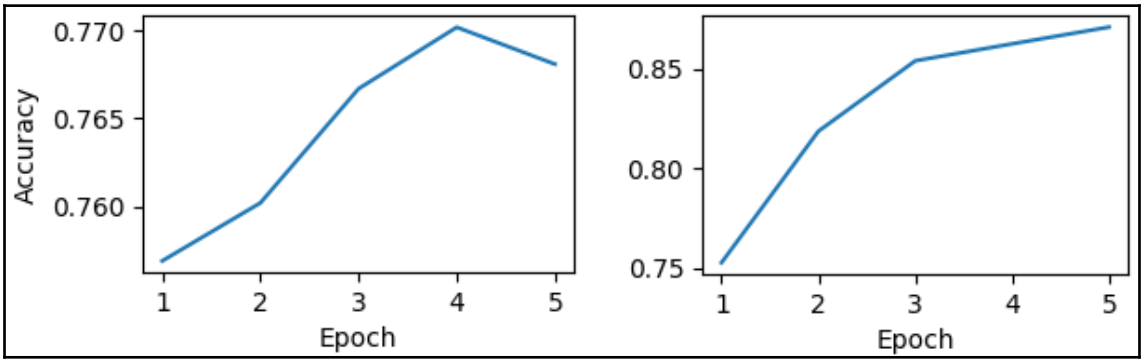


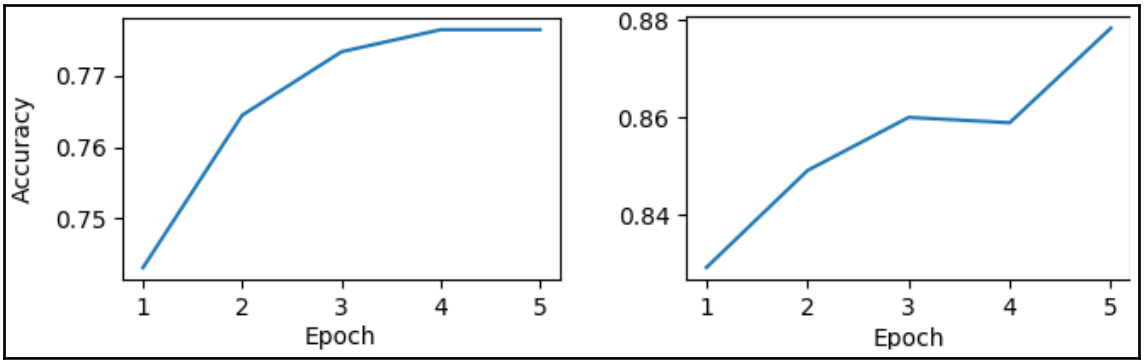
(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d) Guided Grad-CAM 'Cat'

(g) Original Image (h) Guided Backprop 'Dog' (i) Grad-CAM 'Dog' (j) Guided Grad-CAM 'Dog'

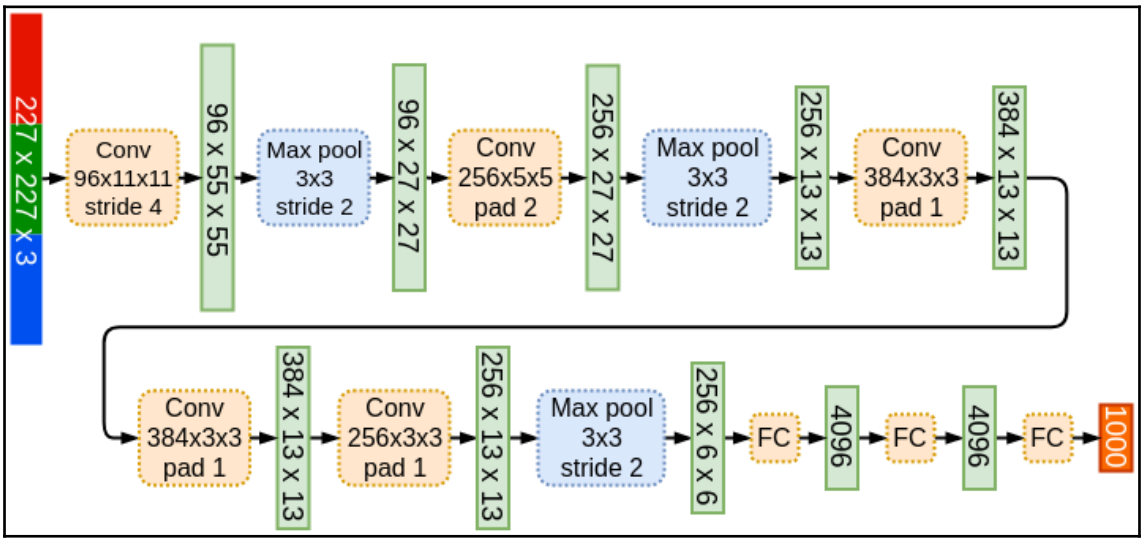



Training
dataset
Task A
Task B
Task C

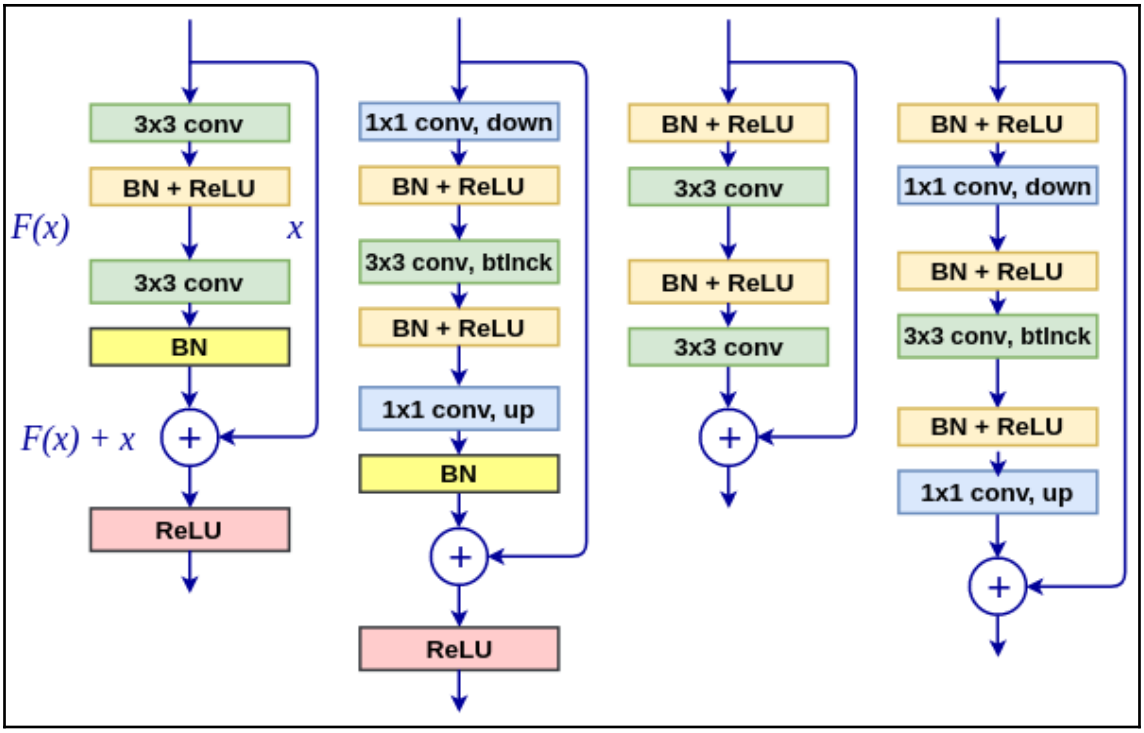

Accuracy
0.770
0.765
0.760
1
2
3
4
5
Epoch
0.85
0.80
0.75
1
2
3
4
5
Epoch


Accuracy
0.77
0.76
0.75
1
2
3
4
5
Epoch
0.88
0.86
0.84
1
2
3
4
5
Epoch

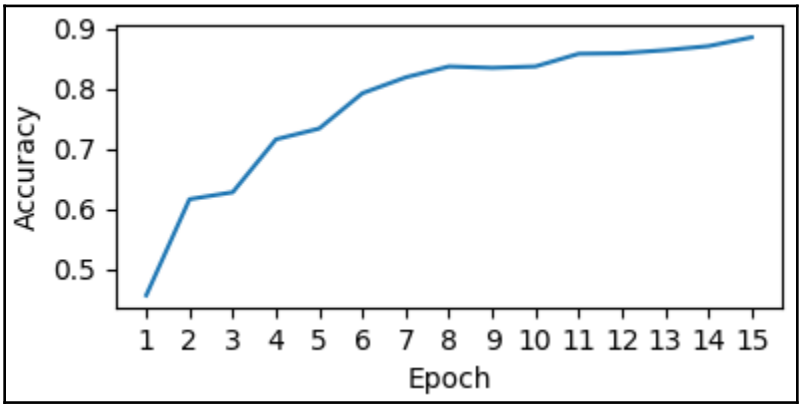
Chapter 3: Advanced Convolutional Networks

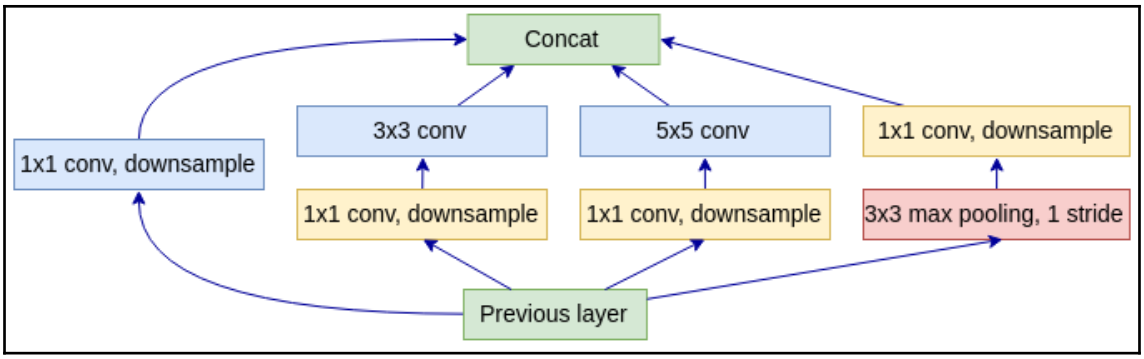


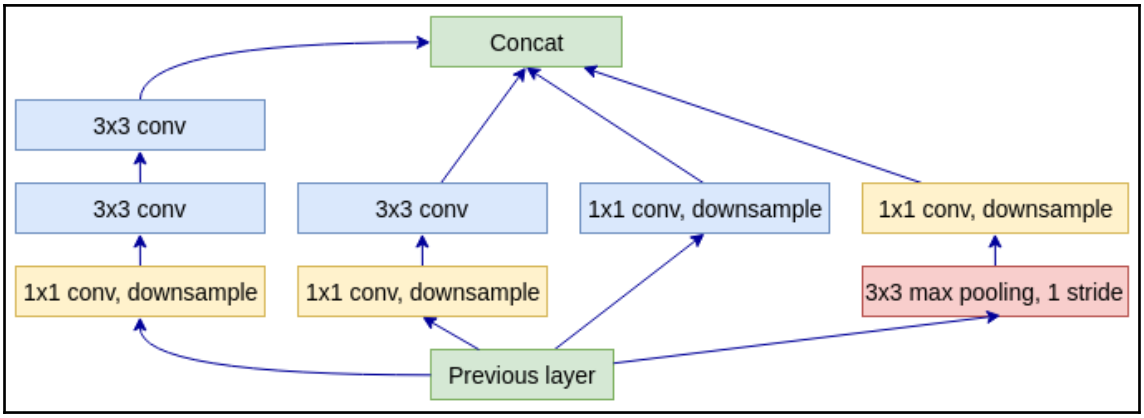
VGG16	VGG19
conv 3x3, 64	conv 3x3, 64
conv 3x3, 64	conv 3x3, 64
max pool	
conv 3x3, 128	conv 3x3, 128
conv 3x3, 128	conv 3x3, 128
max pool	
conv 3x3, 256	conv 3x3, 256
conv 3x3, 256	conv 3x3, 256
conv 3x3, 256	conv 3x3, 256
	conv 3x3, 256
max pool	
conv 3x3, 512	conv 3x3, 512
conv 3x3, 512	conv 3x3, 512
conv 3x3, 512	conv 3x3, 512
	conv 3x3, 512
max pool	
conv 3x3, 512	conv 3x3, 512
conv 3x3, 512	conv 3x3, 512
conv 3x3, 512	conv 3x3, 512
	conv 3x3, 512
max pool	
fc-4096	
fc-4096	
fc-1000	
softmax	

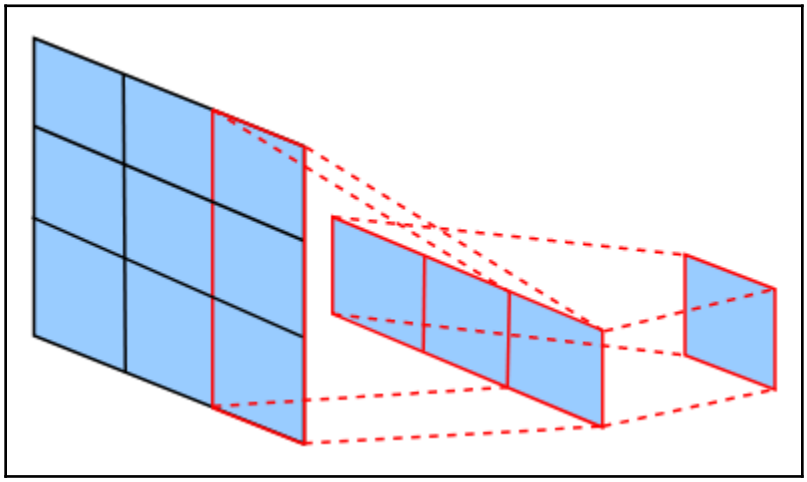

3x3 conv
BN + ReLU
F(x)
x
3x3 conv
BN
F(x) + x
ReLU
1x1 conv, down
BN + ReLU
3x3 conv, btlnck
BN + ReLU
1x1 conv, up
BN
ReLU
BN + ReLU
3x3 conv
BN + ReLU
3x3 conv
BN + ReLU
1x1 conv, down
BN + ReLU
3x3 conv, btlnck
BN + ReLU
1x1 conv, up

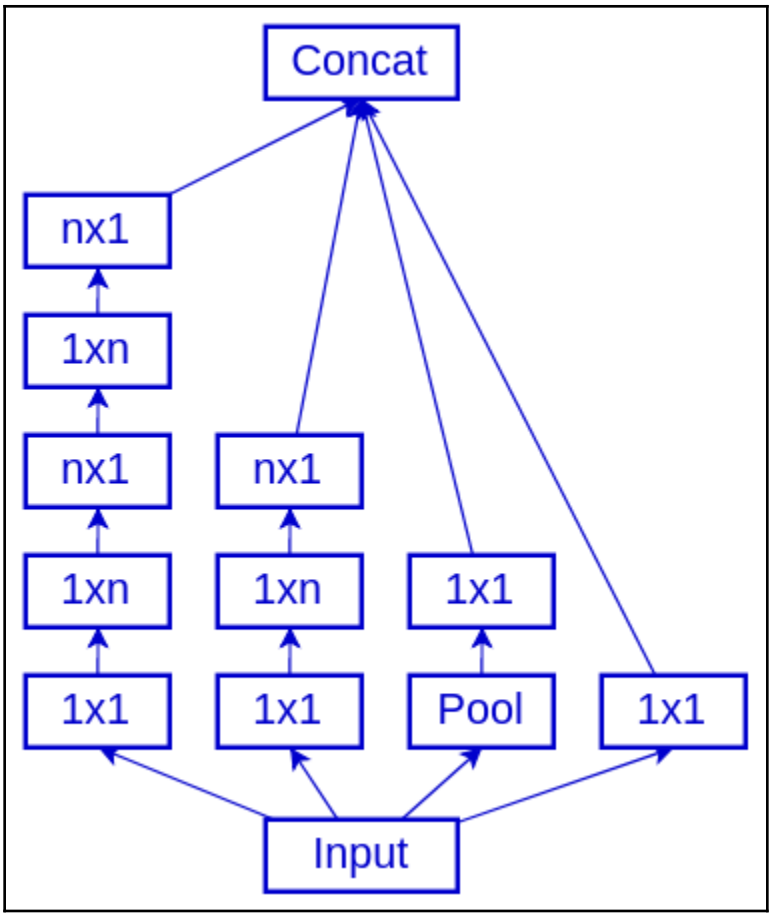
output size	18-layer	34-layer	50-layer	101-layer	152-layer
112x112	7x7 conv, stride 2				
56x56	3x3 max pool, stride 2				
	3x3, 64 3x3, 64 x2	3x3, 64 3x3, 64 x3	1x1, 64 3x3, 64 1x1, 256 x3	1x1, 64 3x3, 64 1x1, 256 x3	1x1, 64 3x3, 64 1x1, 256 x3
28x28	3x3, 128 3x3, 128 x2	3x3, 128 3x3, 128 x4	1x1, 128 3x3, 128 1x1, 512 x4	1x1, 128 3x3, 128 1x1, 512 x4	1x1, 128 3x3, 128 1x1, 512 x8
14x14	3x3, 256 3x3, 256 x2	3x3, 256 3x3, 256 x6	1x1, 256 3x3, 256 1x1, 1024 x6	1x1, 256 3x3, 256 1x1, 1024 x23	1x1, 256 3x3, 256 1x1, 1024 x36
7x7	3x3, 512 3x3, 512 x2	3x3, 512 3x3, 512 x3	1x1, 512 3x3, 512 1x1, 2048 x3	1x1, 512 3x3, 512 1x1, 2048 x3	1x1, 512 3x3, 512 1x1, 2048 x3
1x1	average pool, 1000-d fc, softmax				

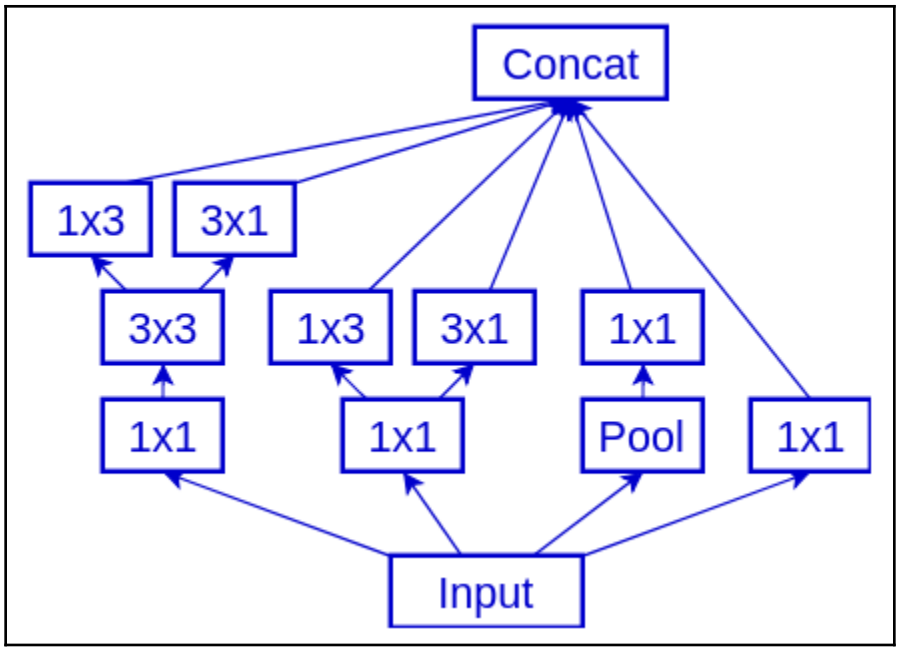


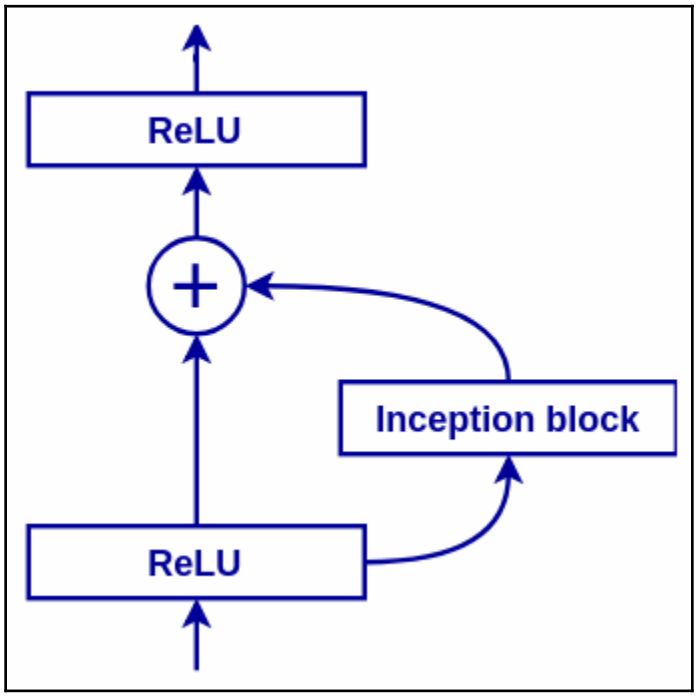

Concat
1x1 conv, downsample
3x3 conv
5x5 conv
1x1 conv, downsample
1x1 conv, downsample
1x1 conv, downsample
3x3 max pooling, 1 stride
Previous layer

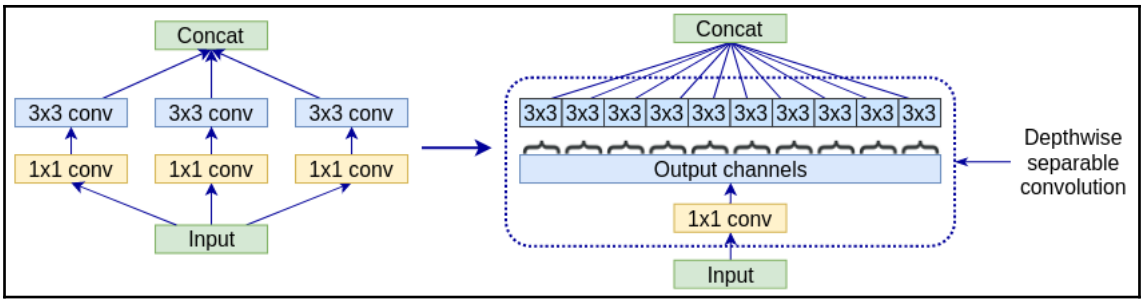

Concat
3x3 conv
3x3 conv
3x3 conv
1x1 conv, downsample
1x1 conv, downsample
1x1 conv, downsample
1x1 conv, downsample
3x3 max pooling, 1 stride
Previous layer

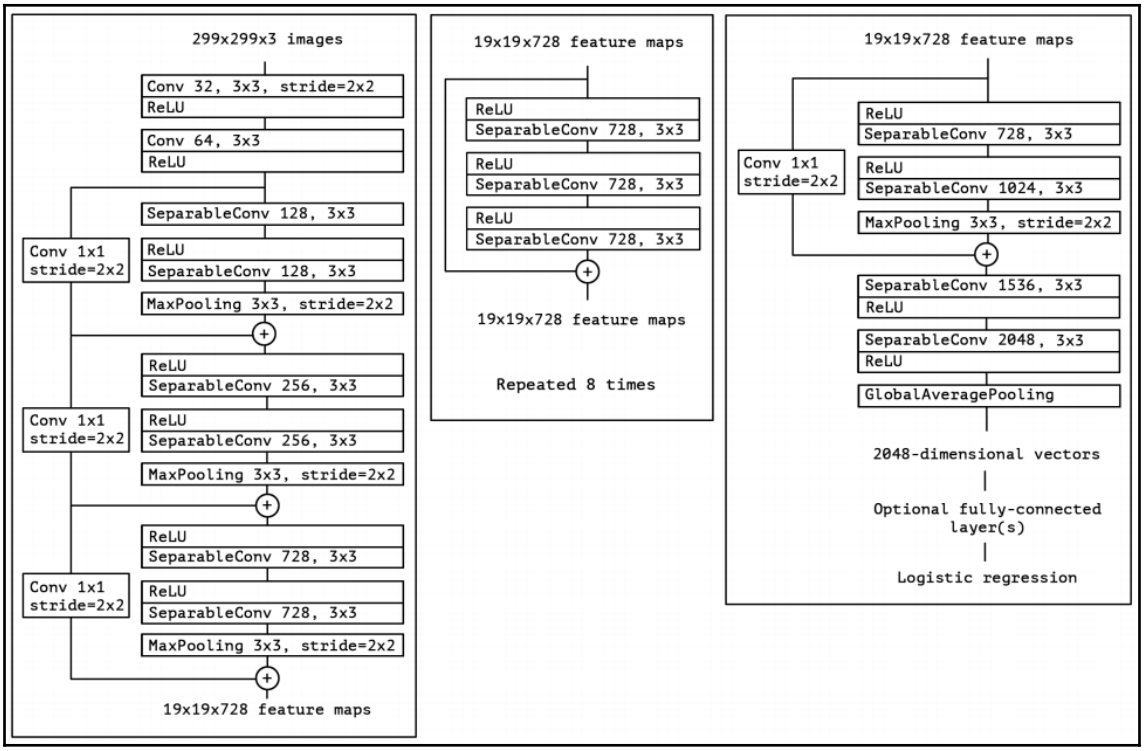


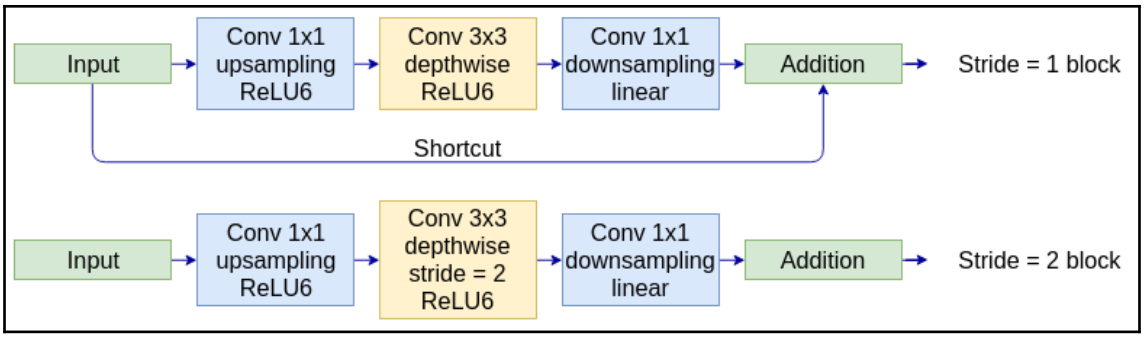

Concat
nx1
1xn
nx1
1xn
1x1
nx1
1xn
1x1
Pool
1x1
Input

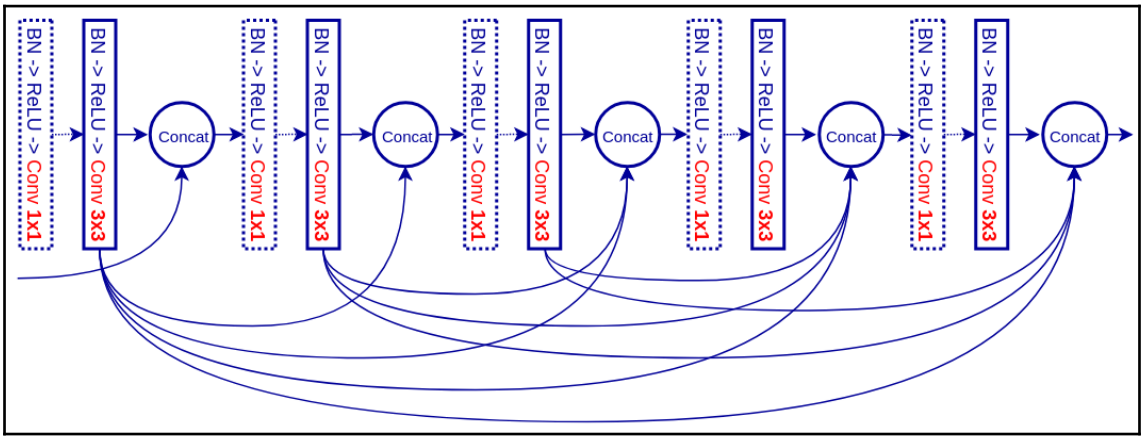

Concat
1x3
3x1
3x3
1x3
3x1
1x1
1x1
1x1
Pool
1x1
Input


ReLU
Inception block
ReLU

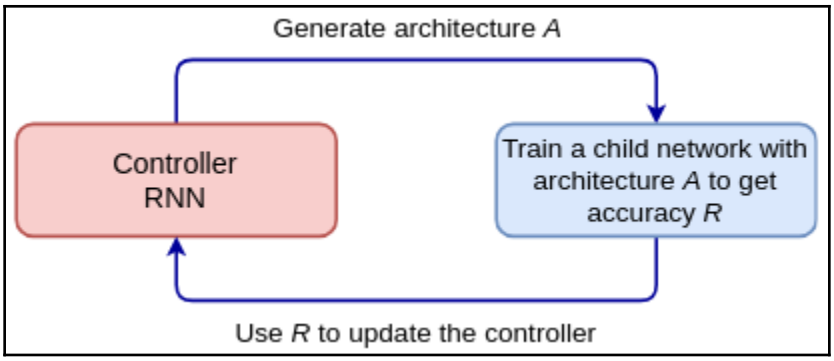

Concat
3x3 conv
3x3 conv
3x3 conv
1x1 conv
1x1 conv
1x1 conv
Input
Concat
3x3 3x3 3x3 3x3 3x3 3x3 3x3 3x3 3x3 3x3
Output channels
1x1 conv
Input
Depthwise separable convolution

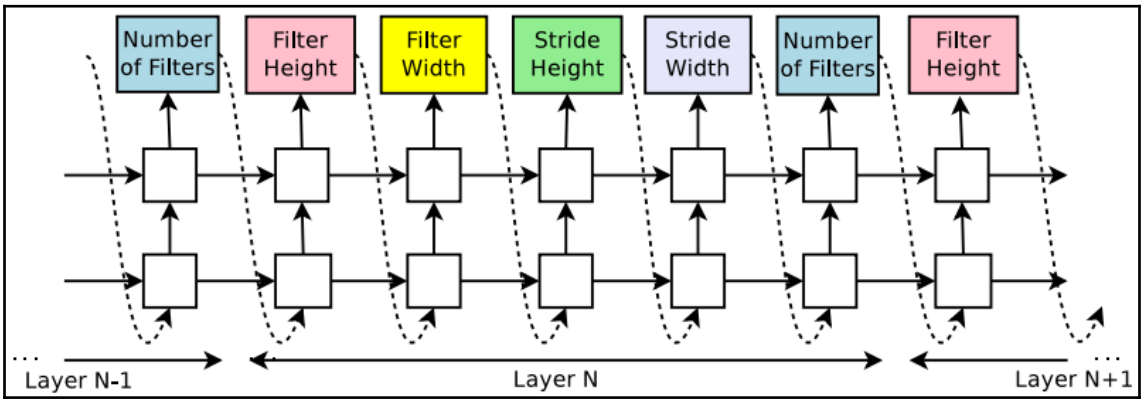

299x299x3 images
Conv 32, 3x3, stride=2x2
ReLU
Conv 64, 3x3
ReLU
SeparableConv 128, 3x3
Conv 1x1 stride=2x2
ReLU
SeparableConv 128, 3x3
MaxPooling 3x3, stride=2x2
ReLU
SeparableConv 256, 3x3
Conv 1x1 stride=2x2
ReLU
SeparableConv 256, 3x3
MaxPooling 3x3, stride=2x2
ReLU
SeparableConv 728, 3x3
Conv 1x1 stride=2x2
ReLU
SeparableConv 728, 3x3
MaxPooling 3x3, stride=2x2
19x19x728 feature maps
19x19x728 feature maps
ReLU
SeparableConv 728, 3x3
ReLU
SeparableConv 728, 3x3
ReLU
SeparableConv 728, 3x3
19x19x728 feature maps
Repeated 8 times
19x19x728 feature maps
Conv 1x1 stride=2x2
ReLU
SeparableConv 728, 3x3
ReLU
SeparableConv 1024, 3x3
MaxPooling 3x3, stride=2x2
SeparableConv 1536, 3x3
ReLU
SeparableConv 2048, 3x3
ReLU
GlobalAveragePooling
2048-dimensional vectors
Optional fully-connected layer(s)
Logistic regression

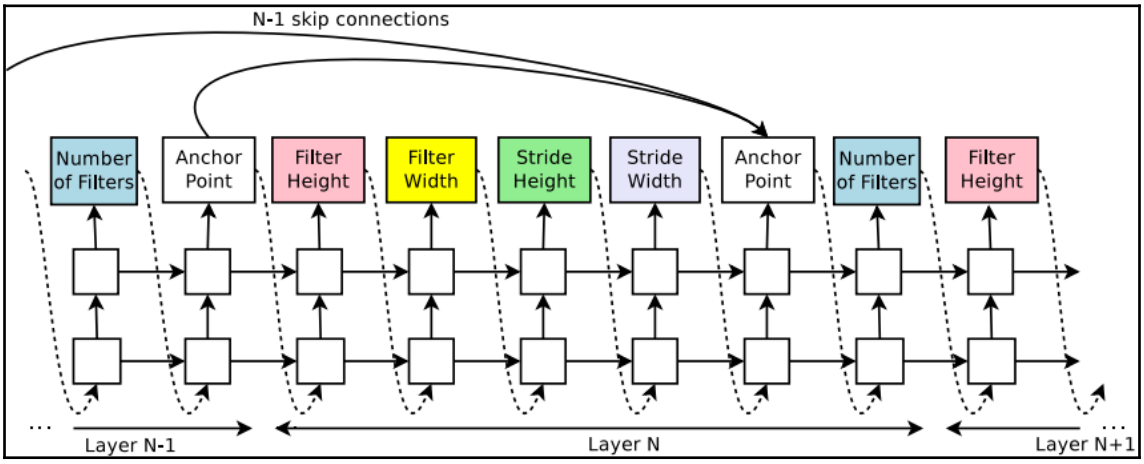


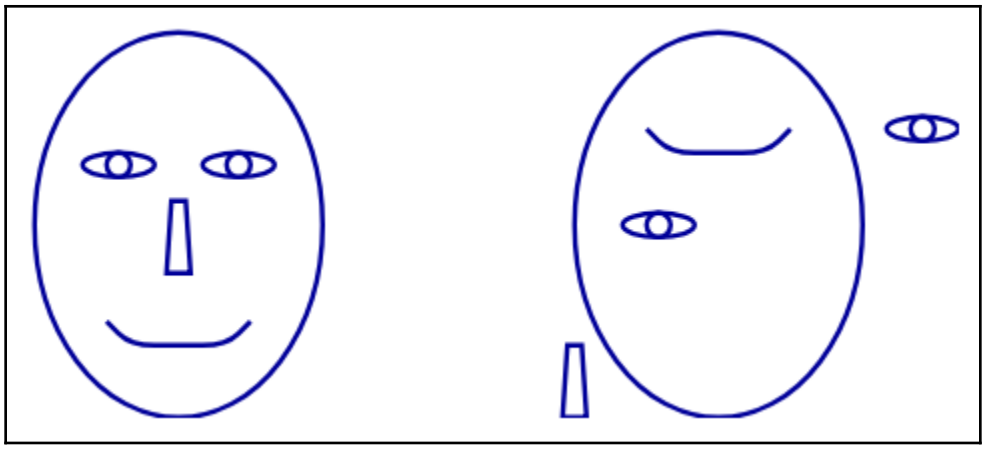


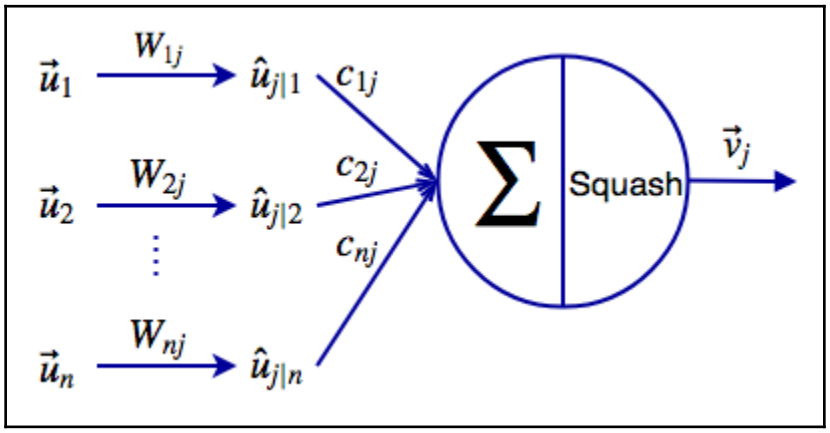
Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully connected, softmax			

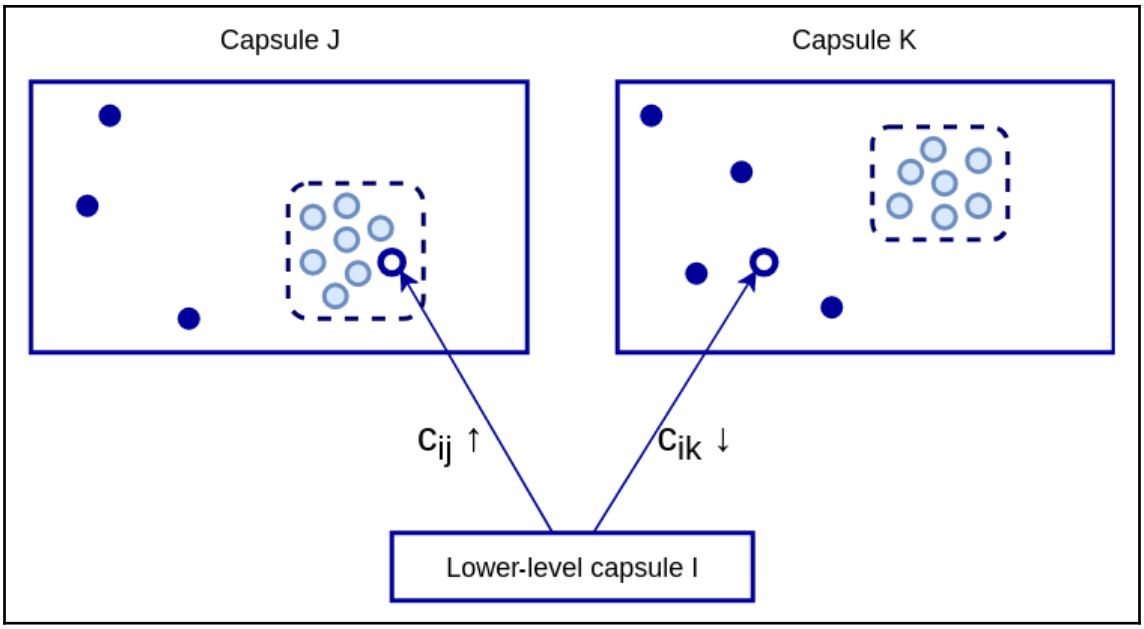

Generate architecture A
Controller RNN
Train a child network with architecture A to get accuracy R
Use R to update the controller


Number of Filters
Filter Height
Filter Width
Stride Height
Stride Width
Number of Filters
Filter Height
Layer N-1
Layer N
Layer N+1

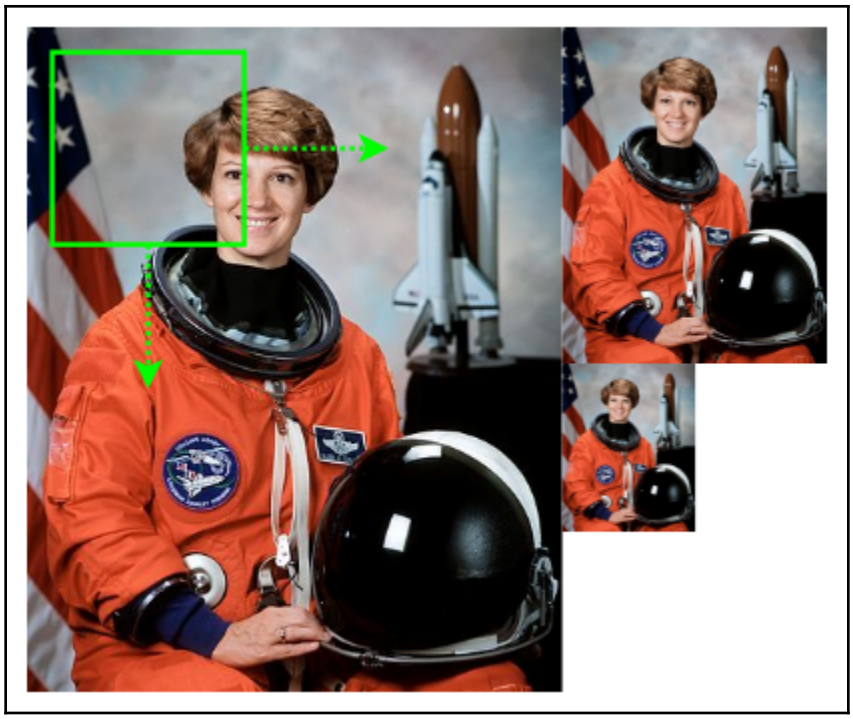

N-1 skip connections
Number of Filters
Anchor Point
Filter Height
Filter Width
Stride Height
Stride Width
Anchor Point
Number of Filters
Filter Height
Layer N-1
Layer N
Layer N+1



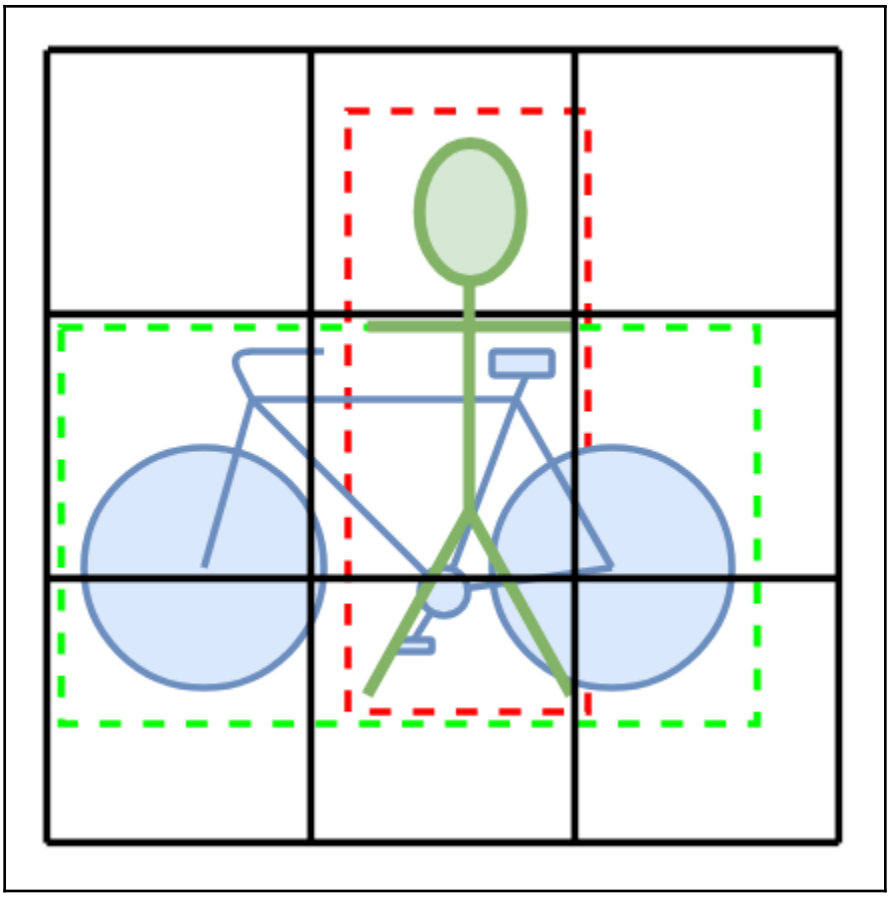

$\vec{u}_1$
W_{1j}
$\hat{u}_{j|1}$
c_{1j}
$\vec{u}_2$
W_{2j}
$\hat{u}_{j|2}$
c_{2j}
c_{nj}
$\vec{u}_n$
W_{nj}
$\hat{u}_{j|n}$
$\sum$
Squash
$\vec{v}_j$

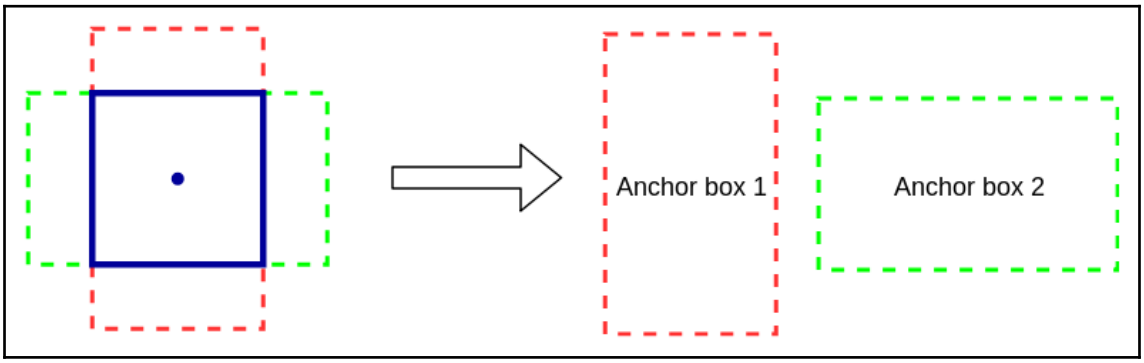

Capsule J
Capsule K
c_{ij} ↑
c_{ik} ↓
Lower-level capsule I

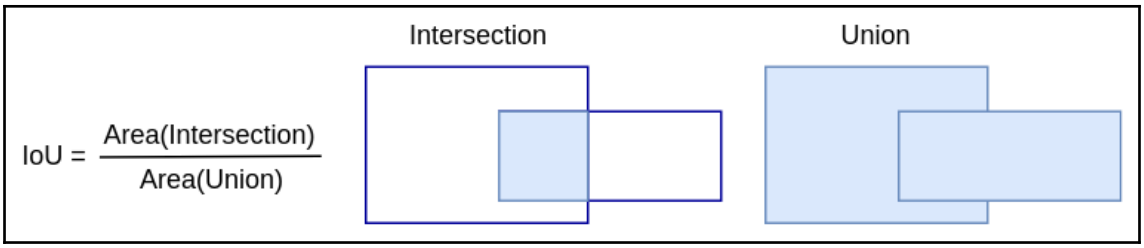
Chapter 4: Object Detection and Image Segmentation

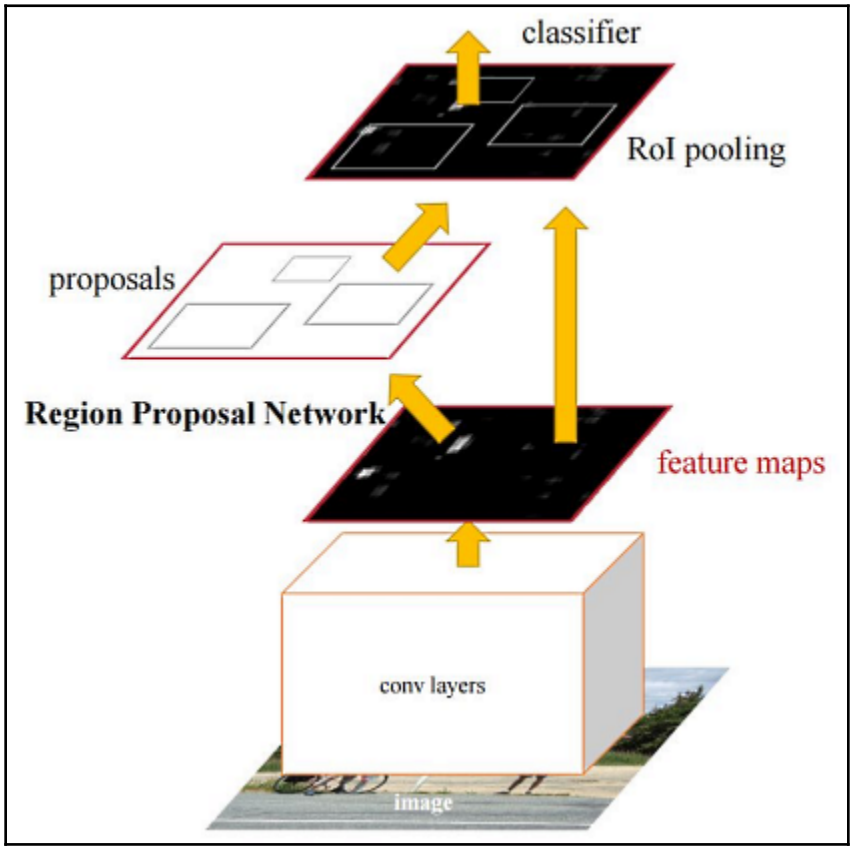


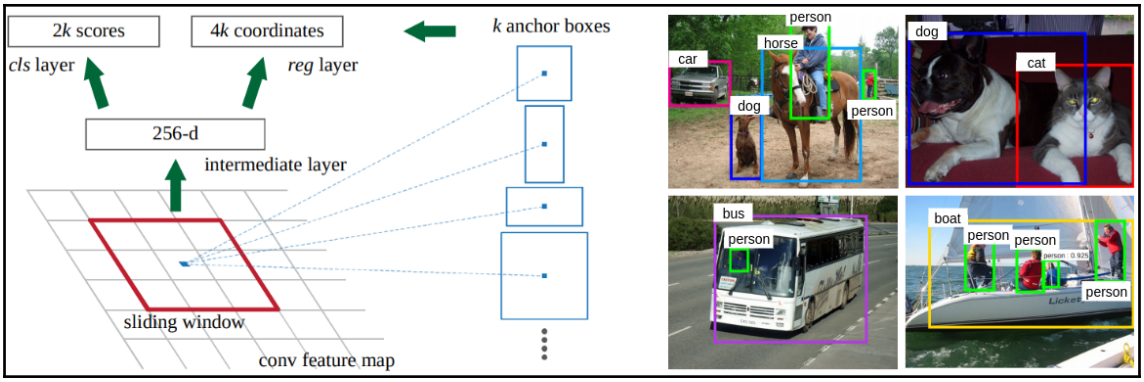
	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

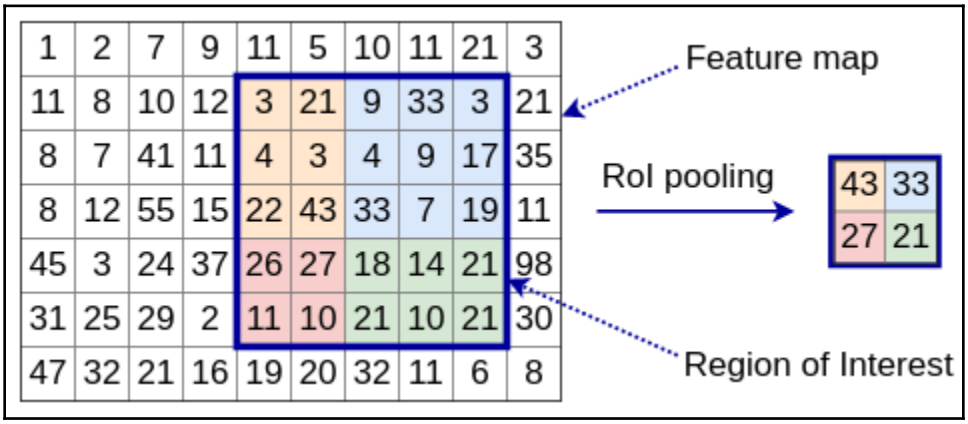



Anchor box 1
Anchor box 2

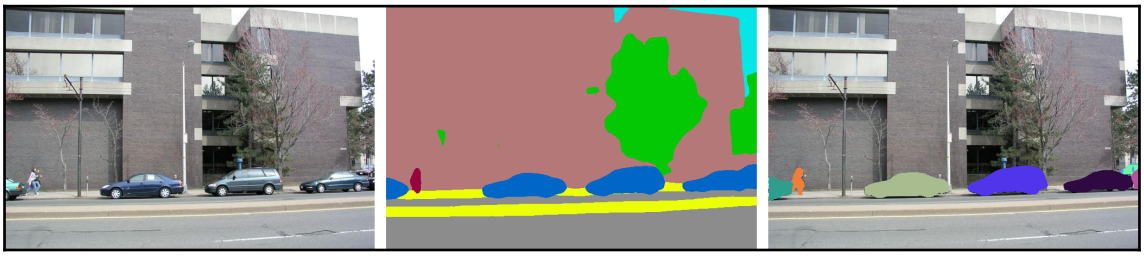

IoU = Area(Intersection) / Area(Union)
Intersection
Union

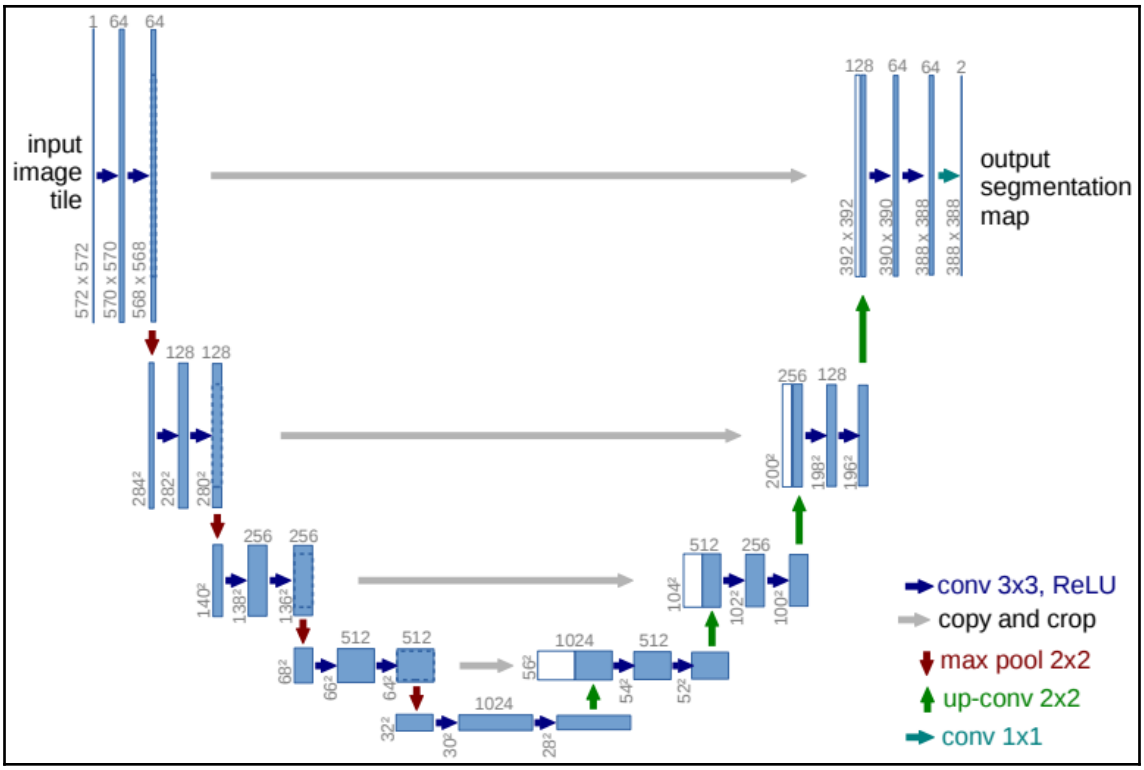

classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers
image

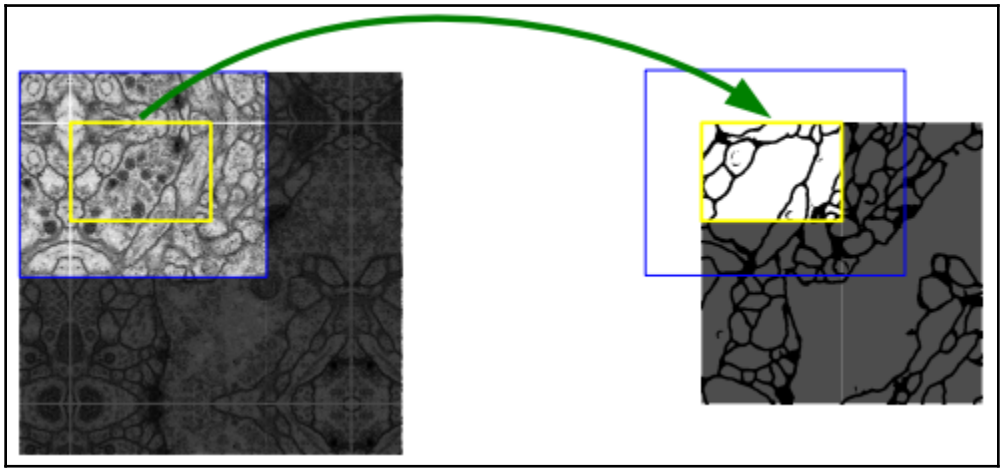

2k scores
4k coordinates
k anchor boxes
cls layer
reg layer
256-d
intermediate layer
sliding window
conv feature map
person
horse
car
dog
person
dog
cat
bus
person
boat
person
person
person

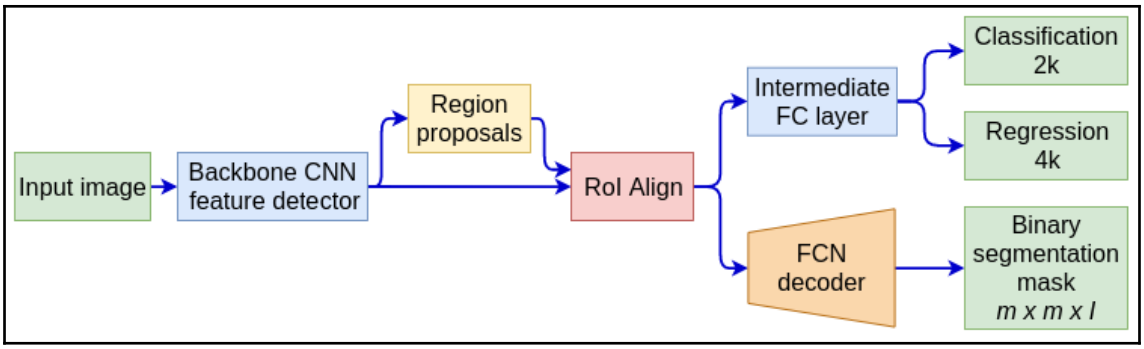

1 2 7 9 11 5 10 11 21 3
11 8 10 12 3 21 9 33 3 21
8 7 41 11 4 3 4 9 17 35
8 12 55 15 22 43 33 7 19 11
45 3 24 37 26 27 18 14 21 98
31 25 29 2 11 10 21 10 21 30
47 32 21 16 19 20 32 11 6 8
Feature map
RoI pooling
43 33
27 21
Region of Interest

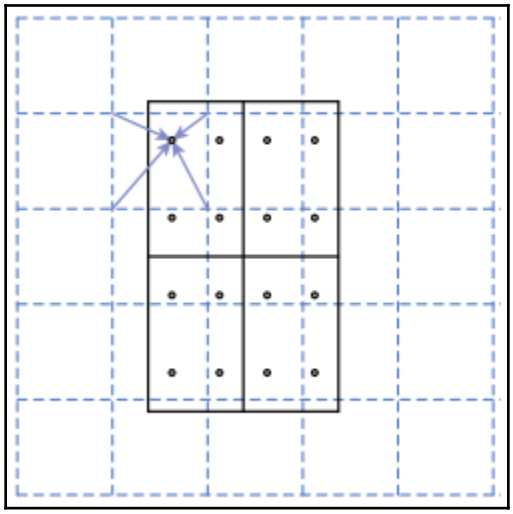
bus
traffic light
person
person
car
parking meter
rnv
3284

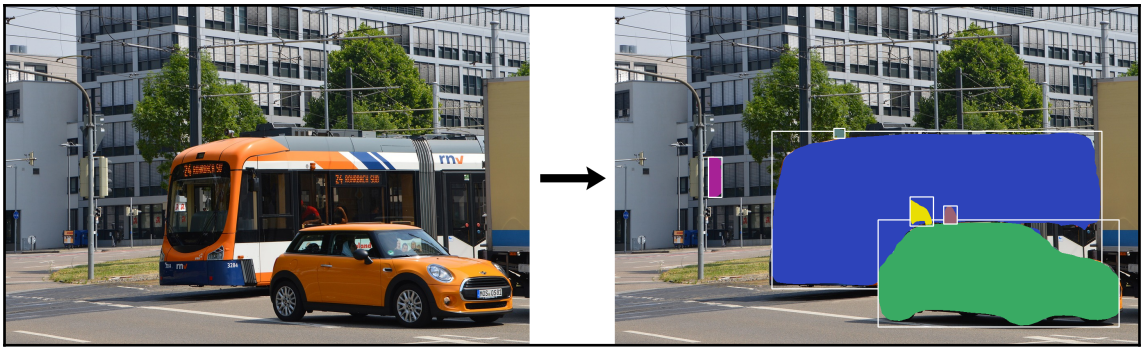



input image tile
572 x 572
570 x 570
568 x 568
1 64 64
128 128
256 256
512 512
1024
output segmentation map
392 x 392
390 x 390
388 x 388
128 64 64 2
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

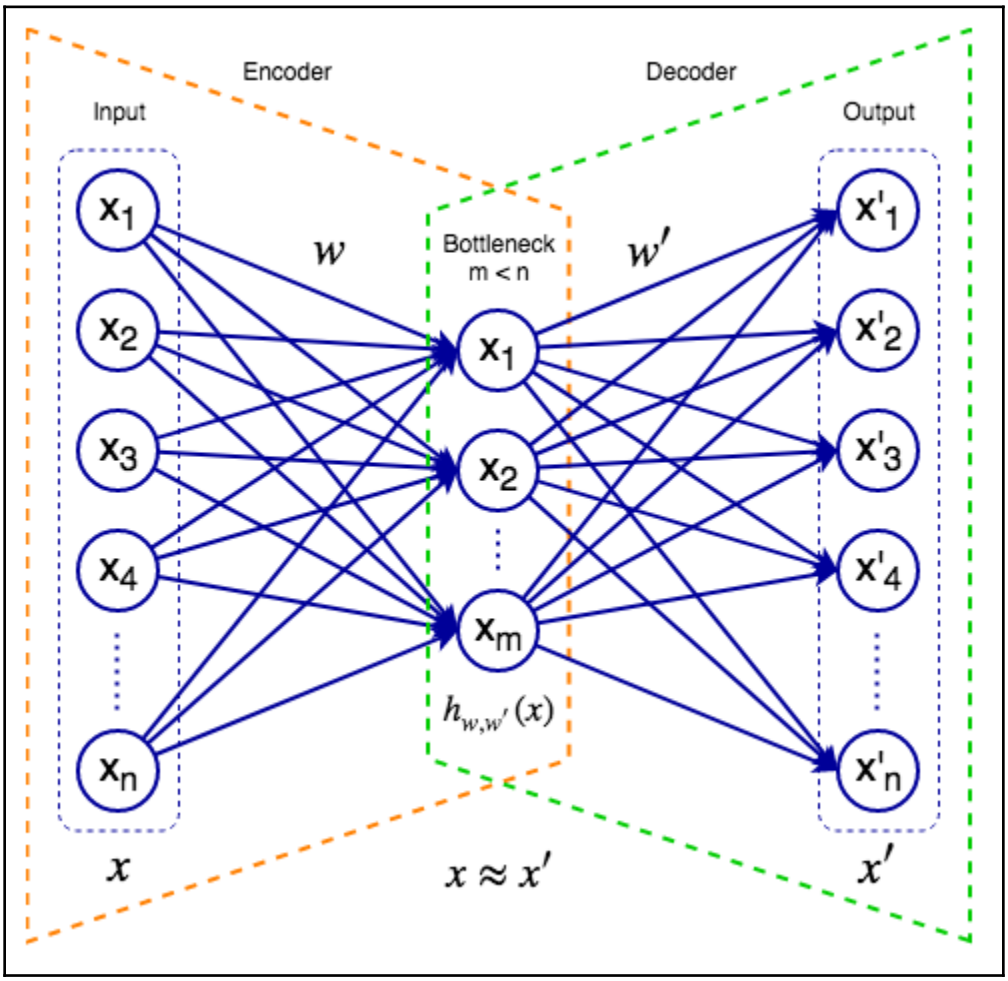


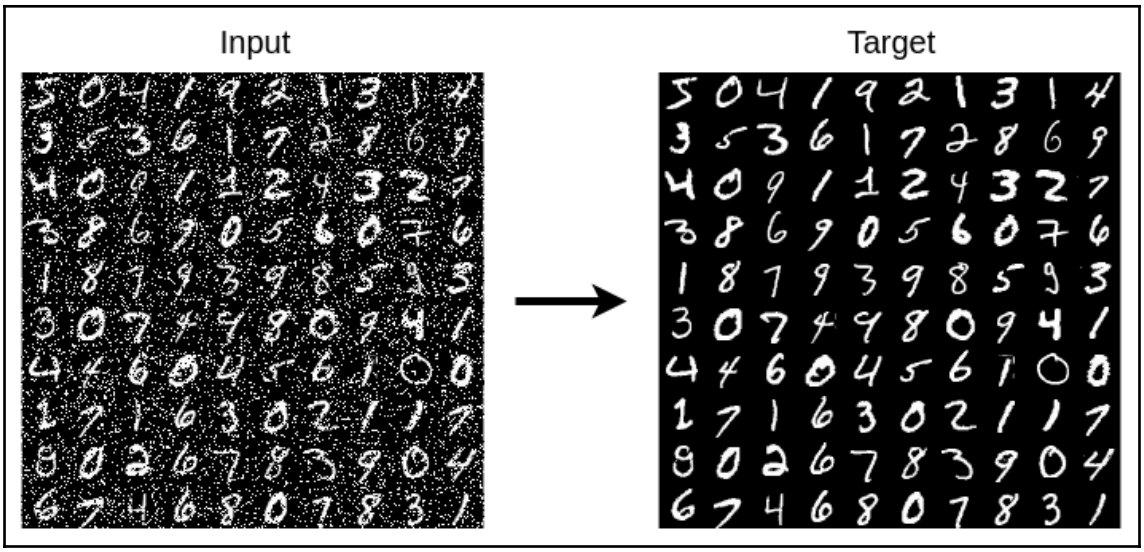

Input image
Backbone CNN feature detector
Region proposals
RoI Align
Intermediate FC layer
Classification 2k
Regression 4k
FCN decoder
Binary segmentation mask
m x m x l

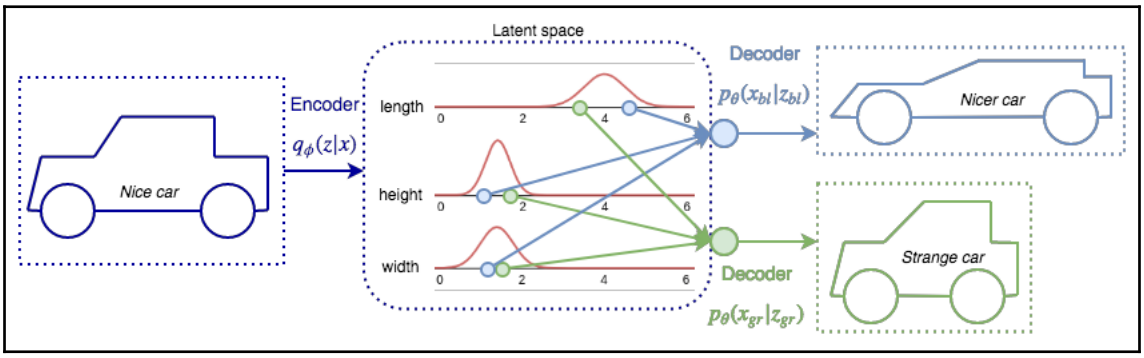


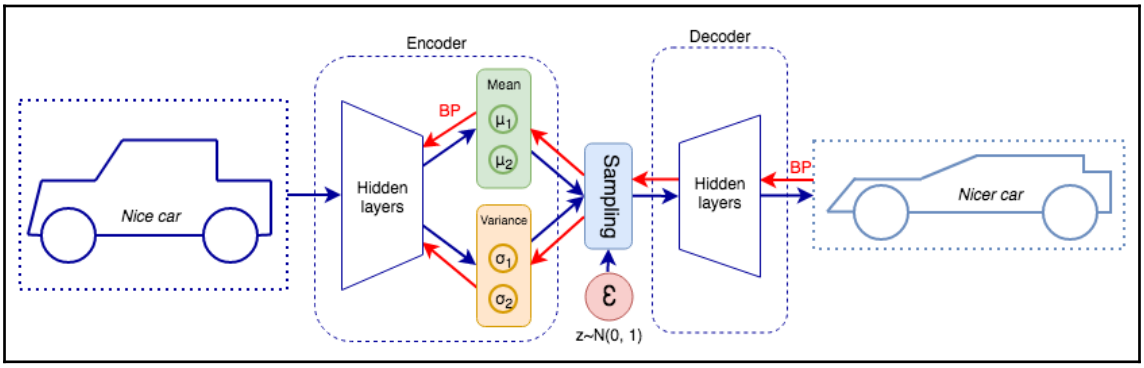


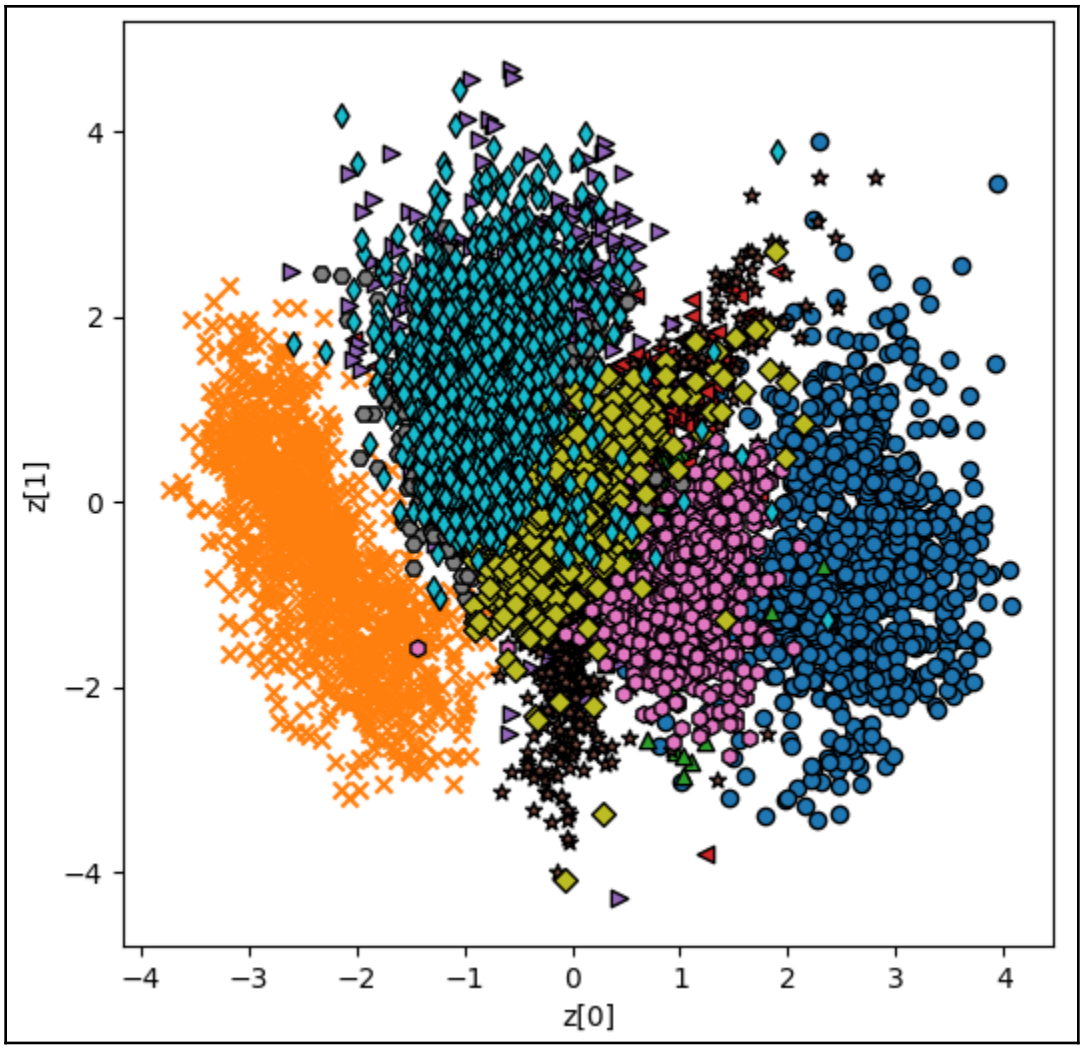
Chapter 5: Generative Models

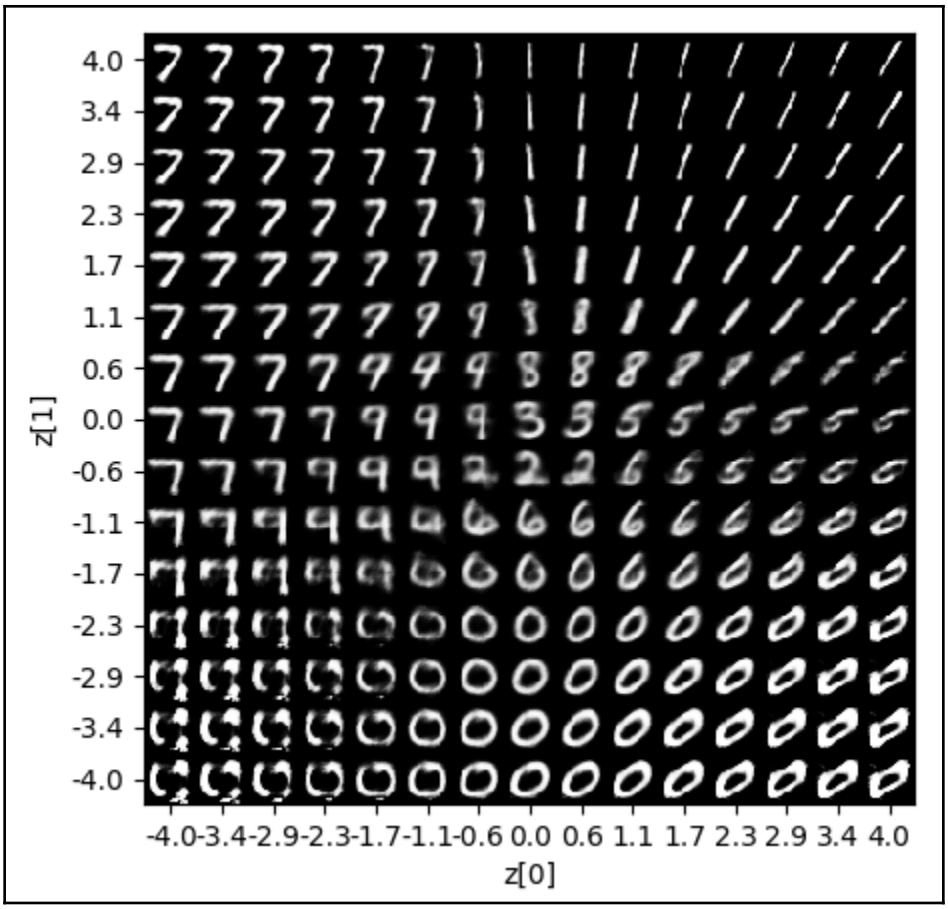


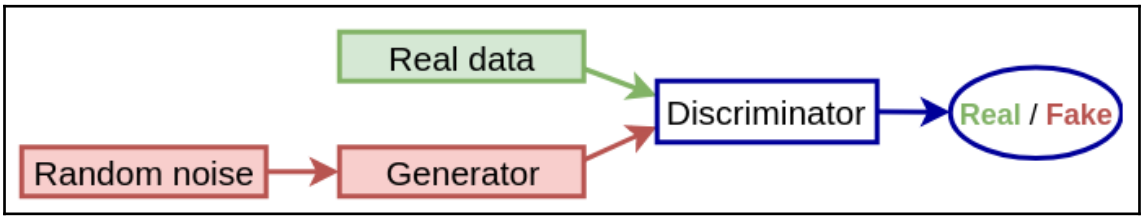

Input
Target

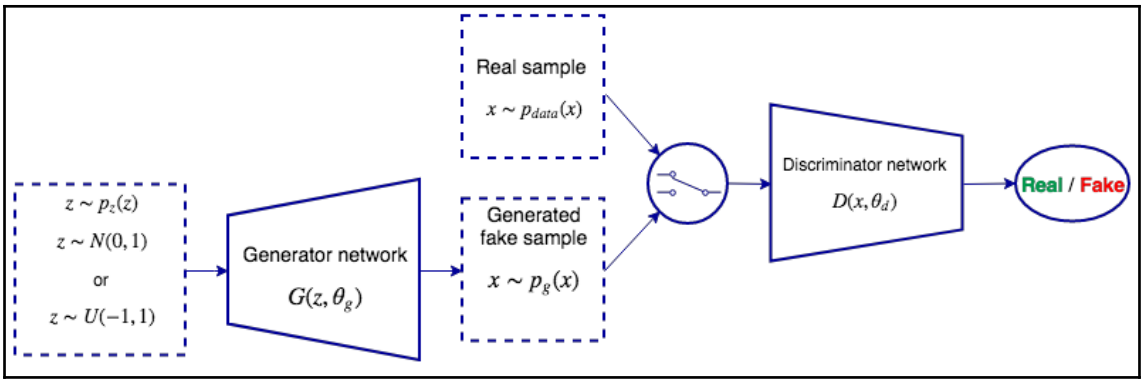

Latent space
Encoder
$q_\phi(z|x)$
Nice car
length
height
width
0
2
4
6
Decoder
$p_\theta(x_{bl}|z_{bl})$
Nicer car
Decoder
$p_\theta(x_{gr}|z_{gr})$
Strange car

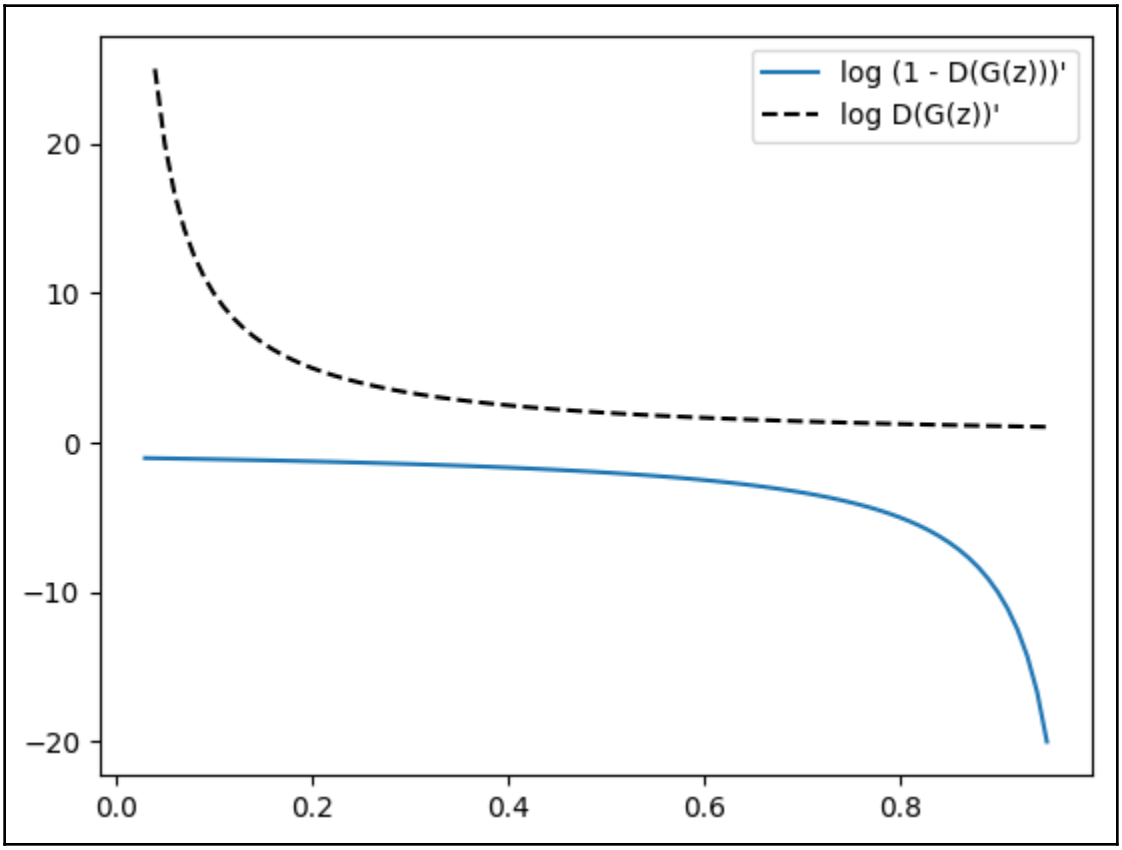

Encoder
Decoder
Nice car
Hidden layers
BP
Mean
μ_1
μ_2
Variance
σ_1
σ_2
Sampling
ε
z~N(0, 1)
Hidden layers
BP
Nicer car

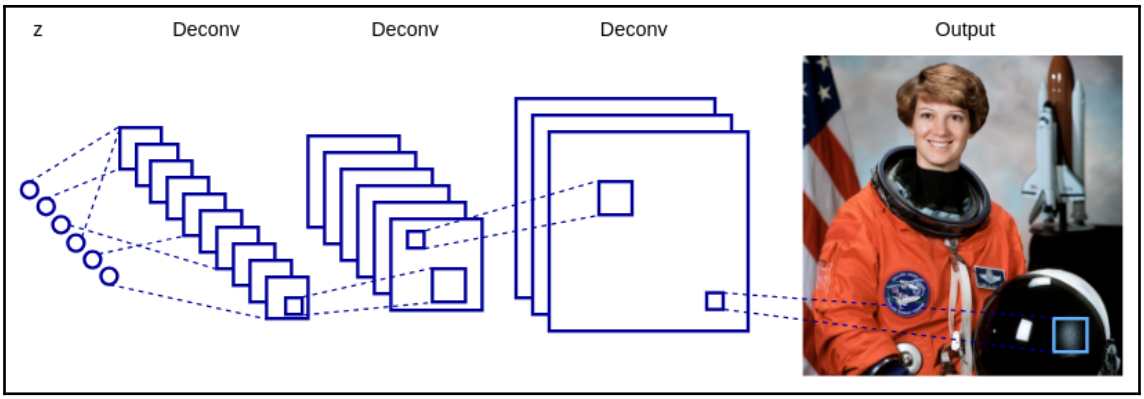

4
2
0
−2
−4
z[1]
−4
−3
−2
−1
0
1
2
3
4
z[0]

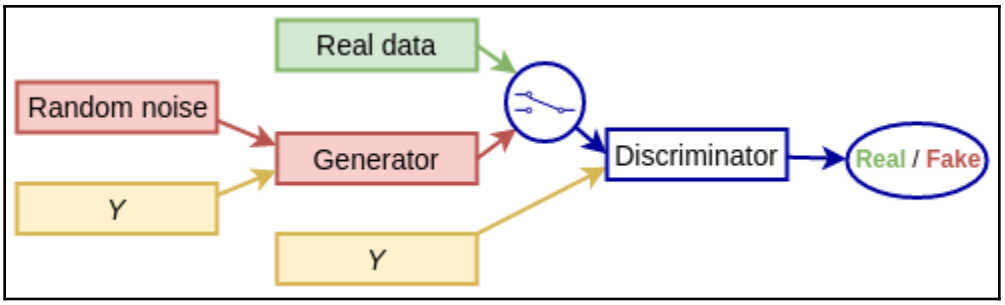

4.0
3.4
2.9
2.3
1.7
1.1
0.6
0.0
-0.6
-1.1
-1.7
-2.3
-2.9
-3.4
-4.0
z[1]
-4.0 -3.4 -2.9 -2.3 -1.7 -1.1 -0.6 0.0 0.6 1.1 1.7 2.3 2.9 3.4 4.0
z[0]

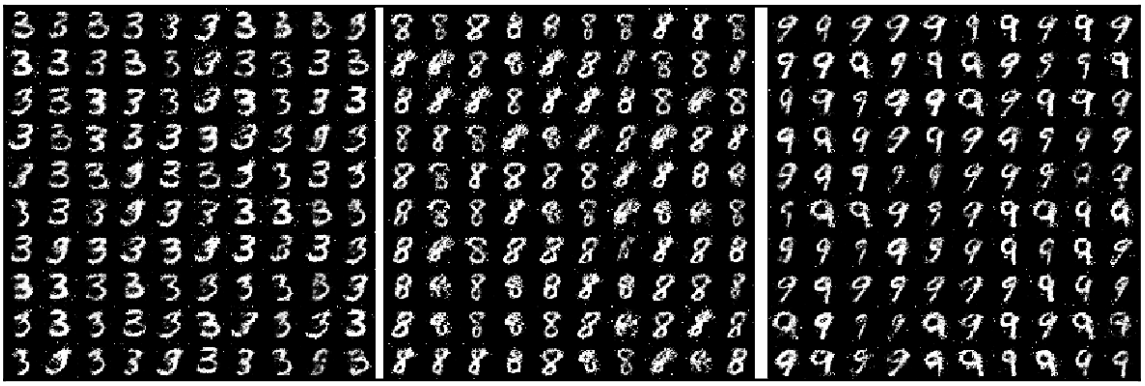

Real data
Discriminator
Real / Fake
Random noise
Generator

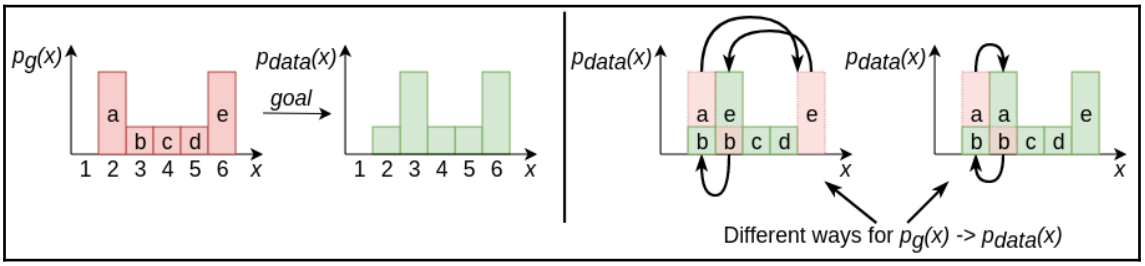

Real sample
$x \sim p_{data}(x)$
$z \sim p_z(z)$
$z \sim N(0,1)$
or
$z \sim U(-1,1)$
Generator network
$G(z, \theta_g)$
Generated fake sample
$x \sim p_g(x)$
Discriminator network
$D(x, \theta_d)$
Real / Fake

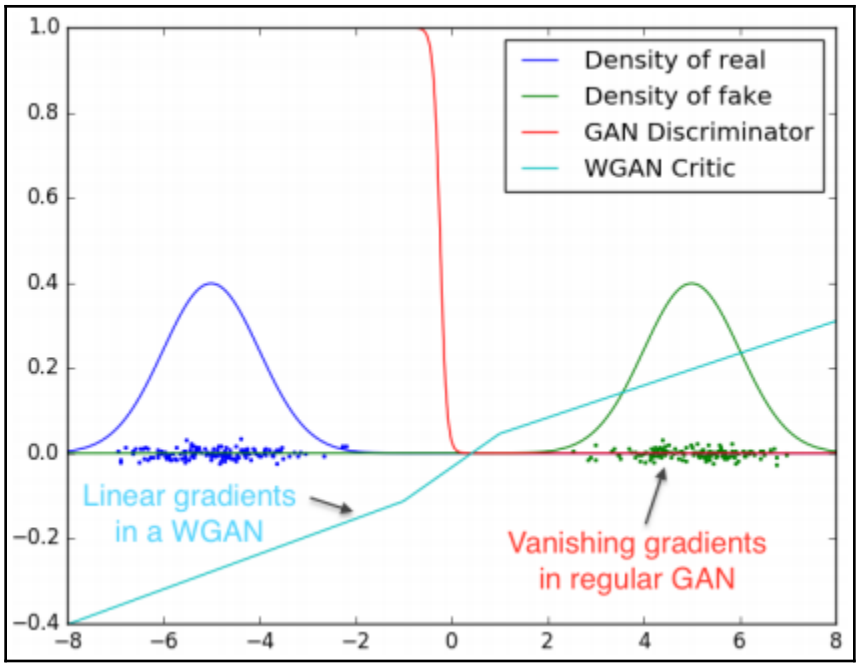

log (1 - D(G(z)))'
log D(G(z))'
20
10
0
−10
−20
0.0
0.2
0.4
0.6
0.8


z
Deconv
Deconv
Deconv
Output


Real data
Random noise
Generator
Discriminator
Real / Fake
Y
Y




pg(x)
pdata(x)
goal
a
b c d
e
1 2 3 4 5 6
x
pdata(x)
pdata(x)
a e
b b c d
e
a a
b b c d
e
x
Different ways for pg(x) -> pdata(x)

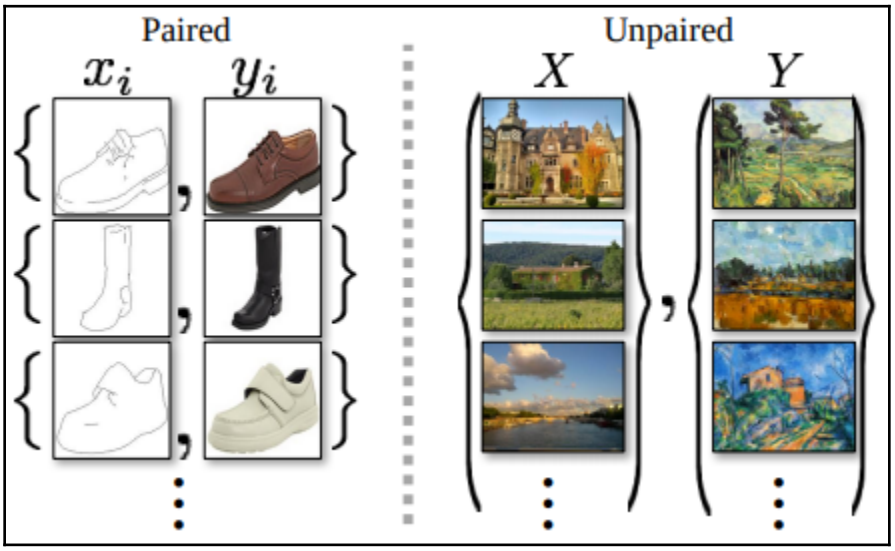


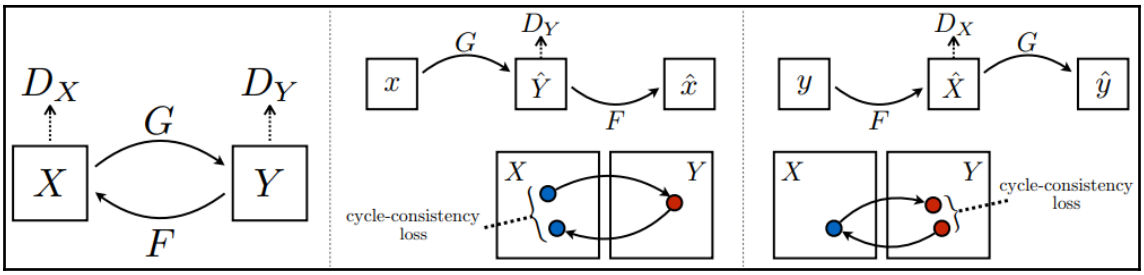
Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

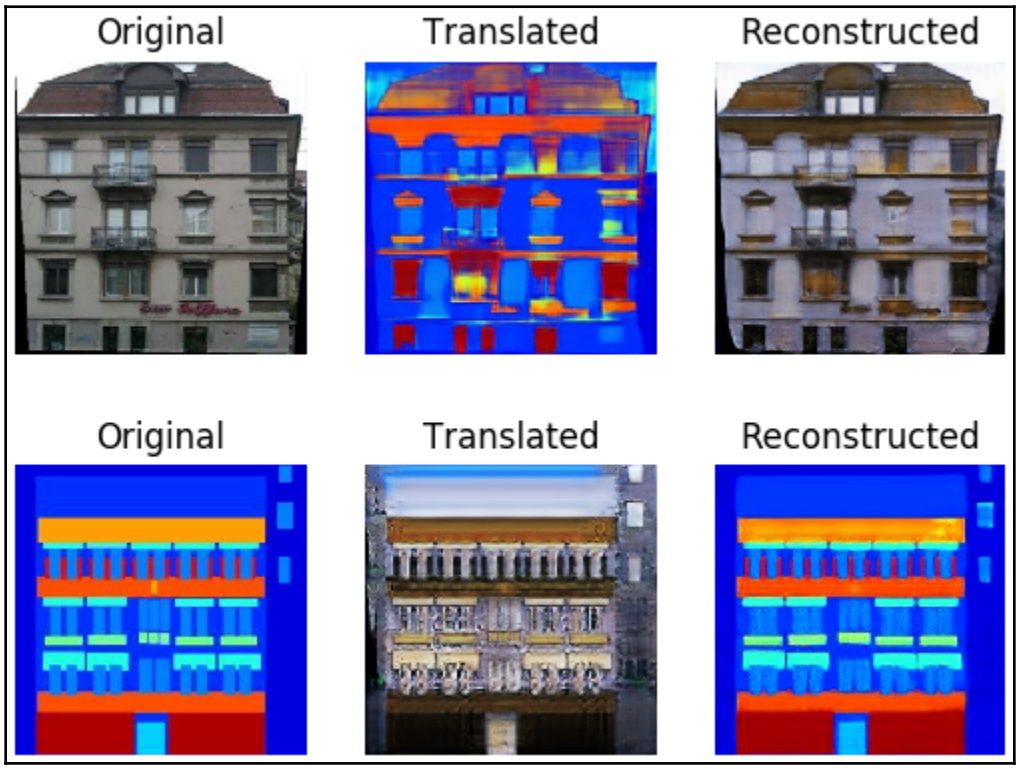
Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.

Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

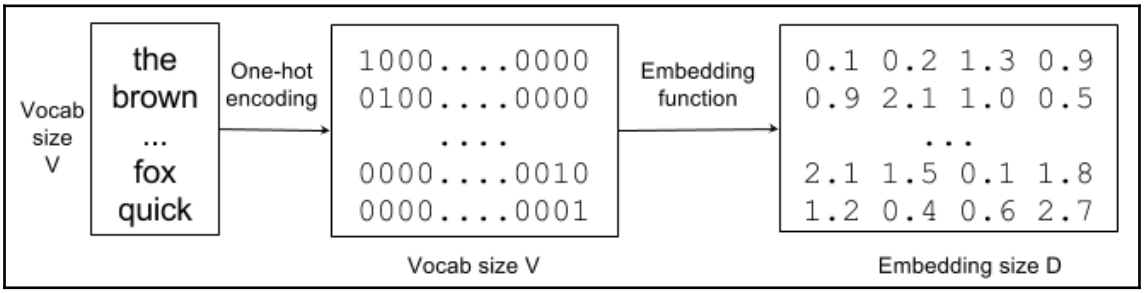

Paired
Unpaired
x_i
y_i
X
Y

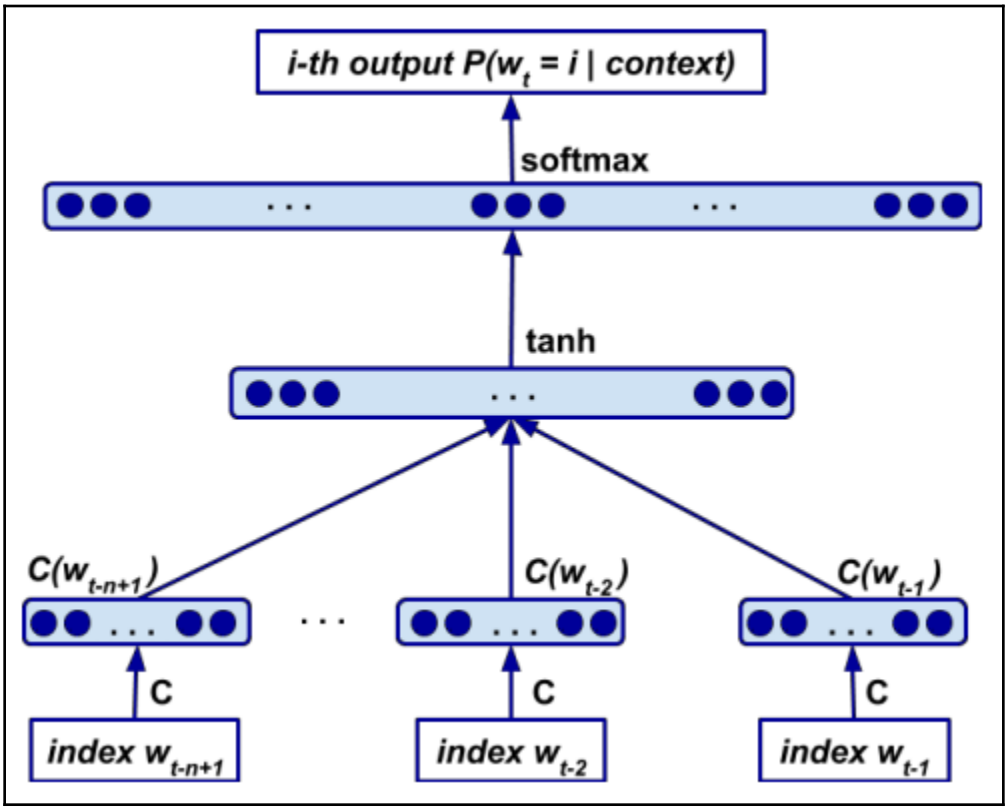

D_X
D_Y
G
X
Y
F
x
$\hat{Y}$
$\hat{x}$
cycle-consistency loss
y
$\hat{X}$
$\hat{y}$

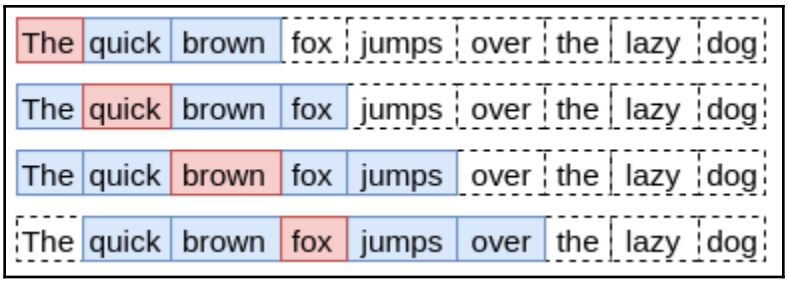

Original
Translated
Reconstructed
Original
Translated
Reconstructed

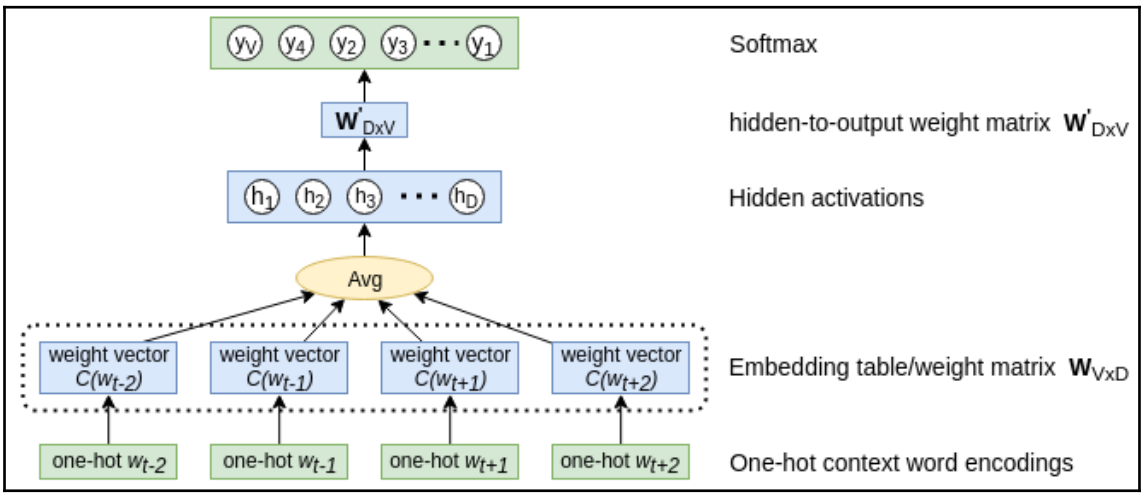
Content
Style
Generated Image

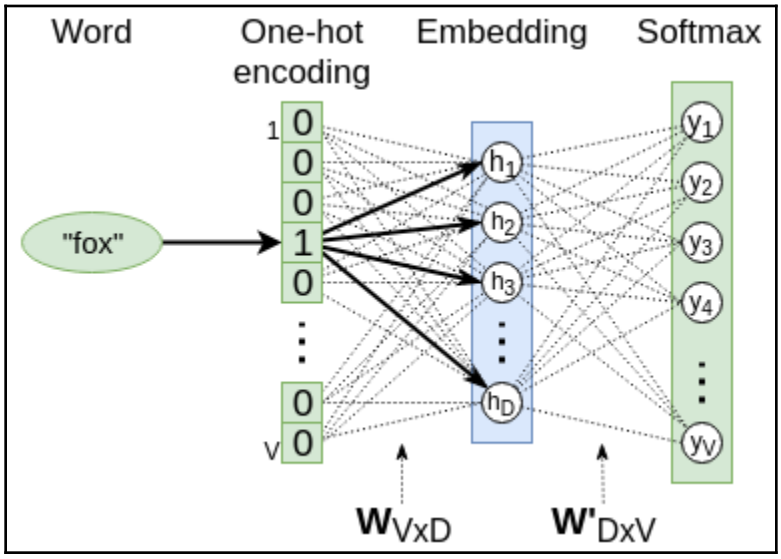
Chapter 6: Language Modeling

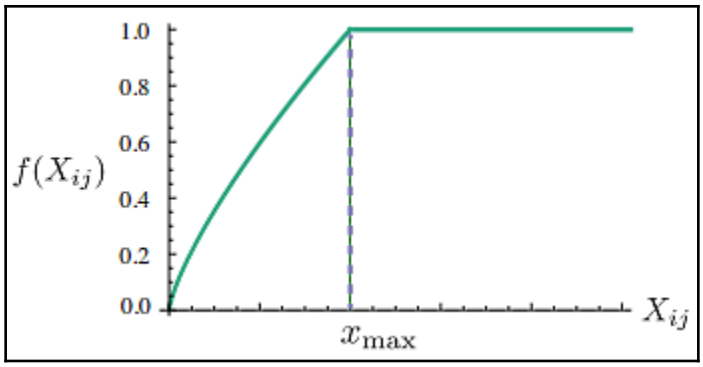


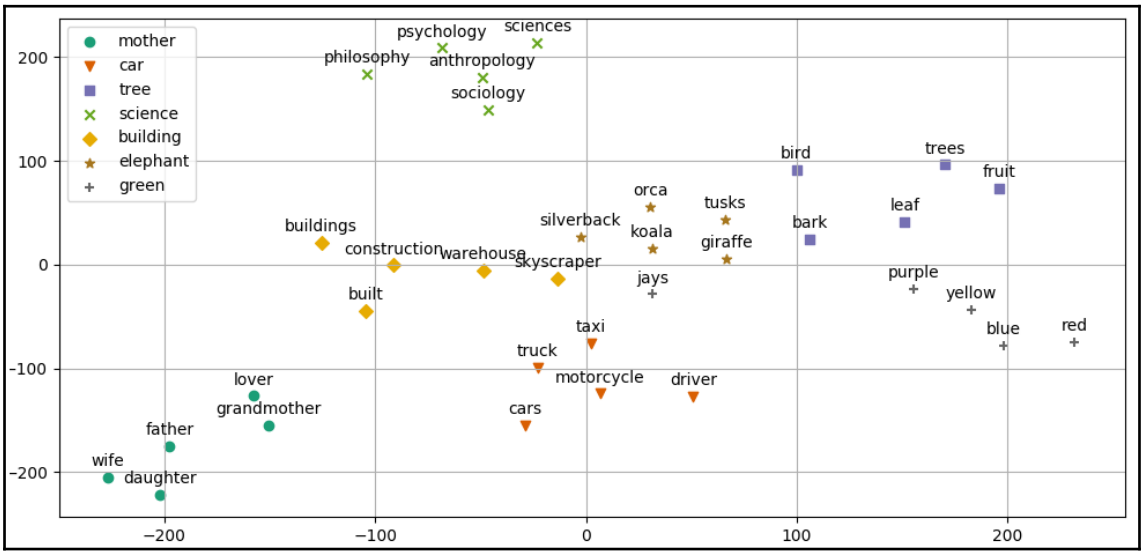



The quick brown fox jumps over the lazy dog
The quick brown fox jumps over the lazy dog
The quick brown fox jumps over the lazy dog
The quick brown fox jumps over the lazy dog

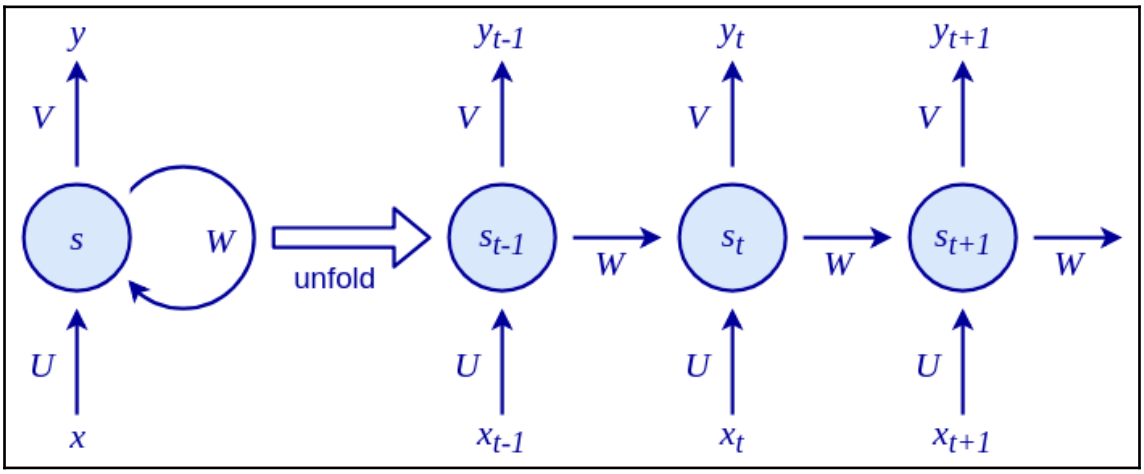

Softmax
hidden-to-output weight matrix W'DxV
Hidden activations
Avg
weight vector C(wt-2)
weight vector C(wt-1)
weight vector C(wt+1)
weight vector C(wt+2)
Embedding table/weight matrix WVxD
one-hot wt-2
one-hot wt-1
one-hot wt+1
one-hot wt+2
One-hot context word encodings

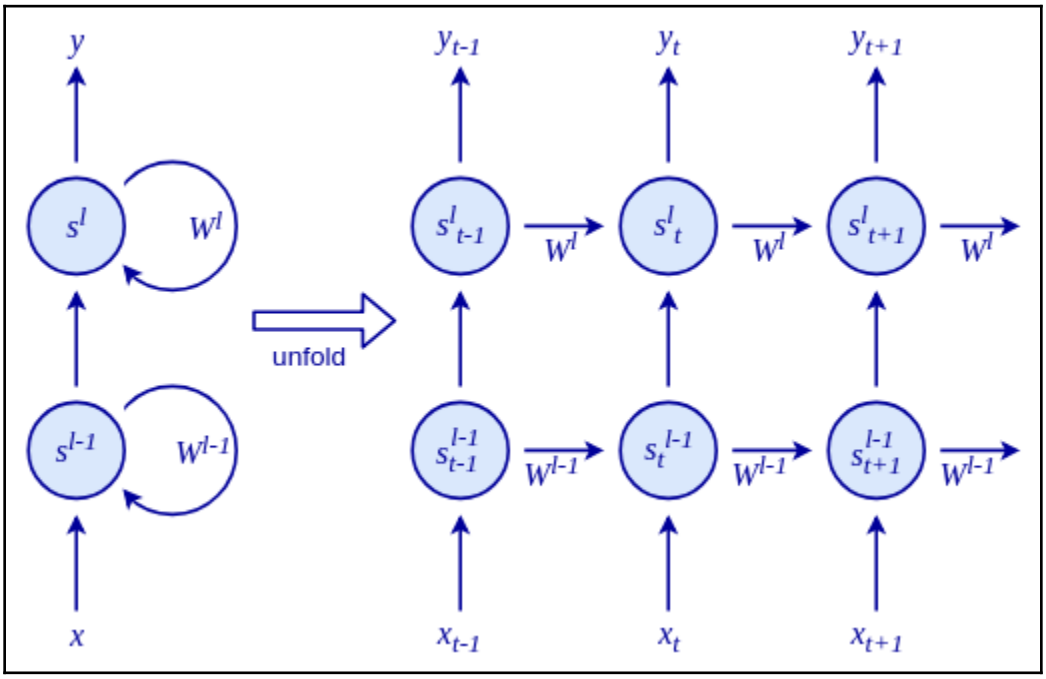

Word
One-hot encoding
Embedding
Softmax
"fox"
1
V
WVxD
W'DxV

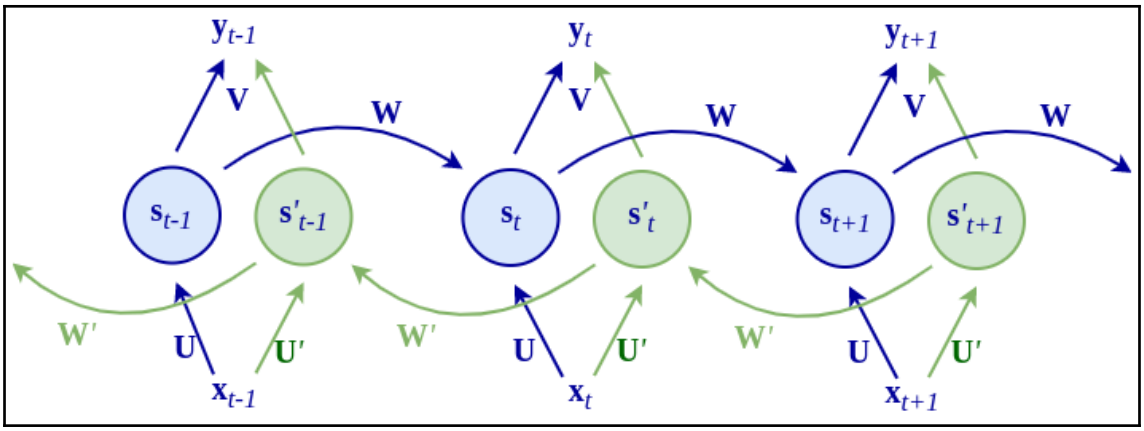

1.0
0.8
0.6
0.4
0.2
0.0
$f(X_{ij})$
$x_{\max}$
X_{ij}

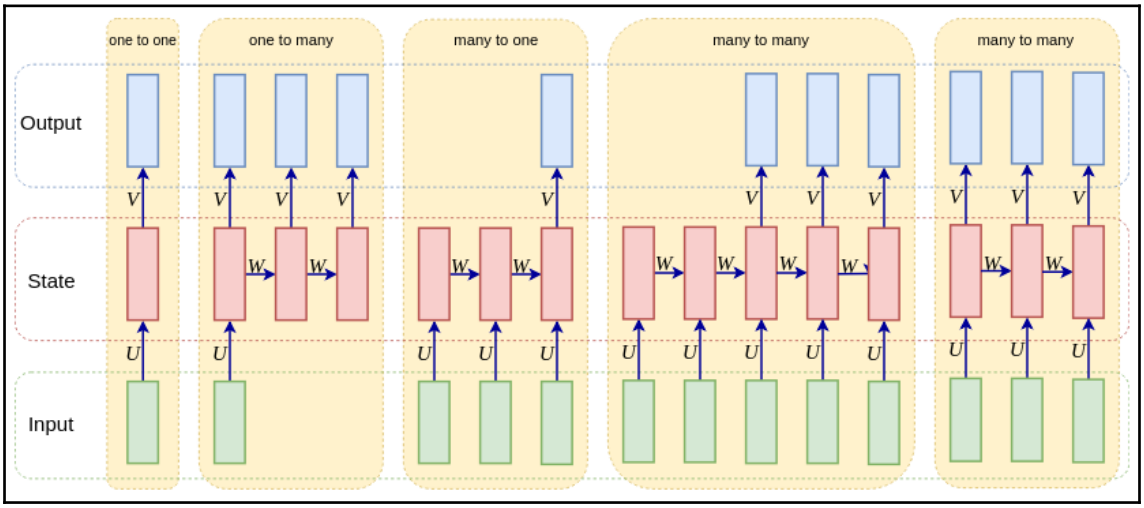

mother
car
tree
science
building
elephant
green
psychology
sciences
philosophy
anthropology
sociology
bird
trees
fruit
orca
tusks
leaf
silverback
bark
buildings
koala
giraffe
construction
warehouse
skyscraper
jays
purple
yellow
built
taxi
blue
red
truck
lover
motorcycle
driver
grandmother
cars
father
wife
daughter
200
100
0
−100
−200
−200
−100
0
100
200

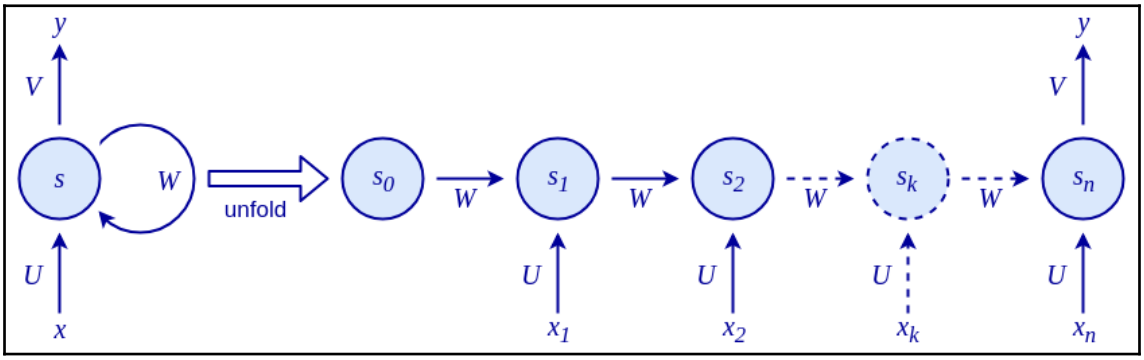
Chapter 7: Understanding Recurrent Networks

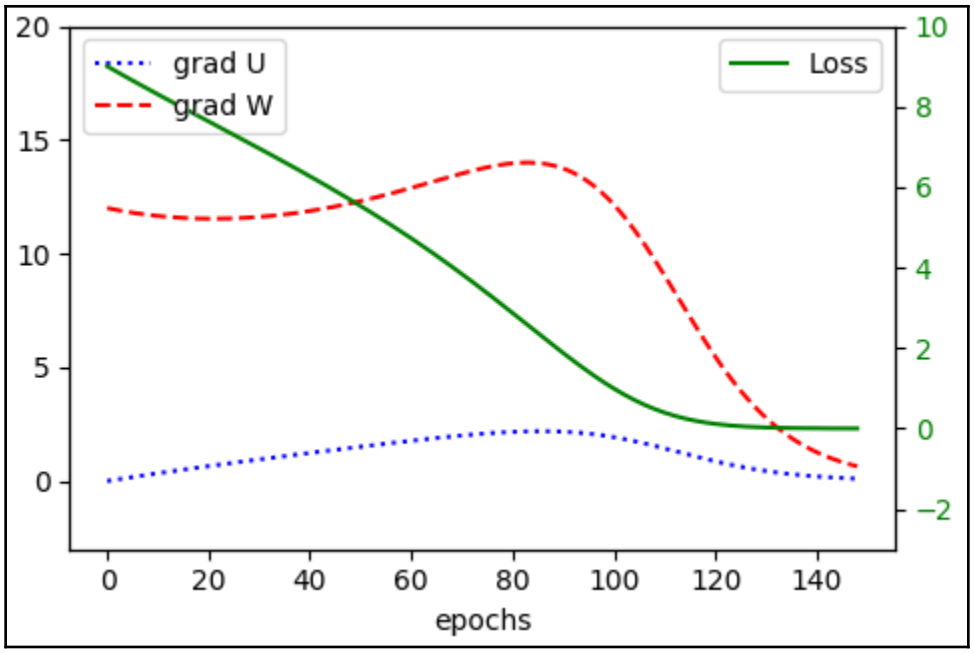


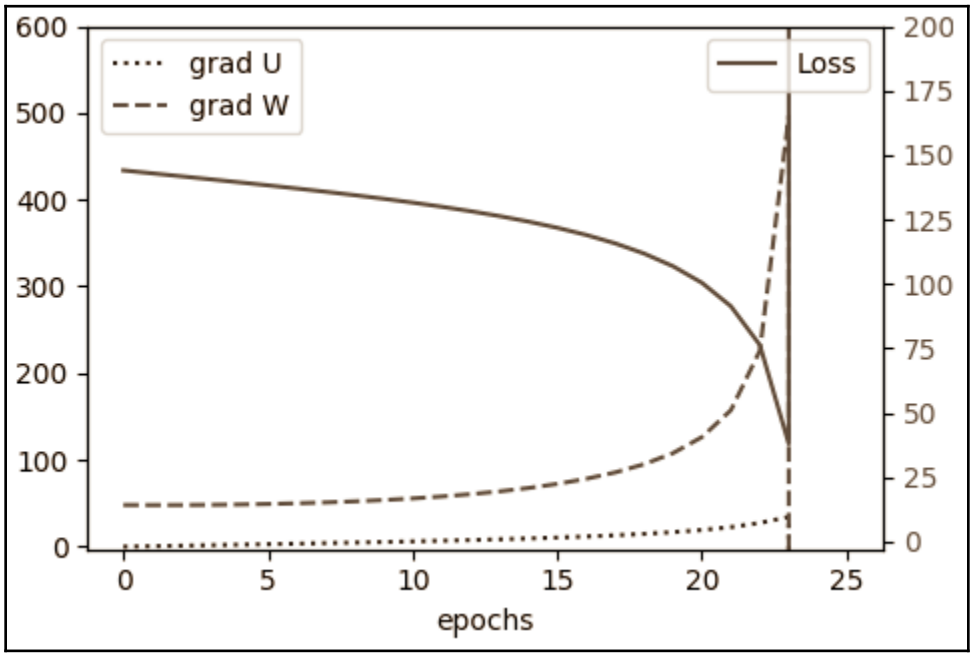


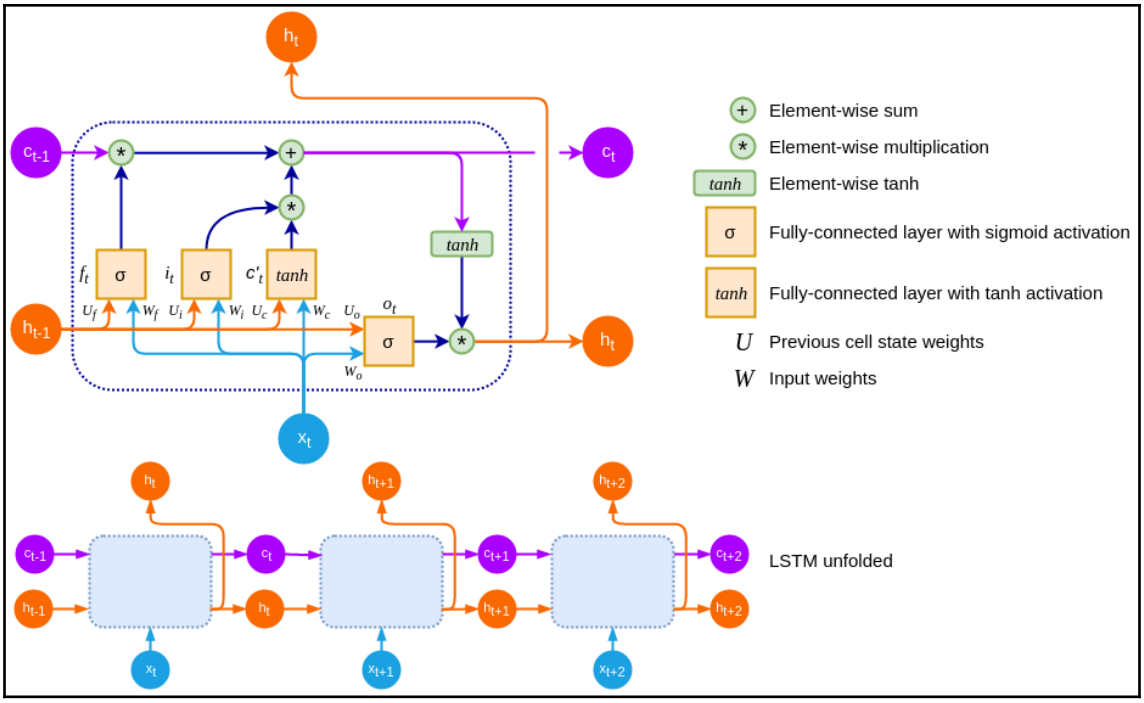

y_{t-1}
y_t
y_{t+1}
V
W
s_{t-1}
s'_{t-1}
s_t
s'_t
s_{t+1}
s'_{t+1}
W′
U
U′
x_{t-1}
x_t
x_{t+1}

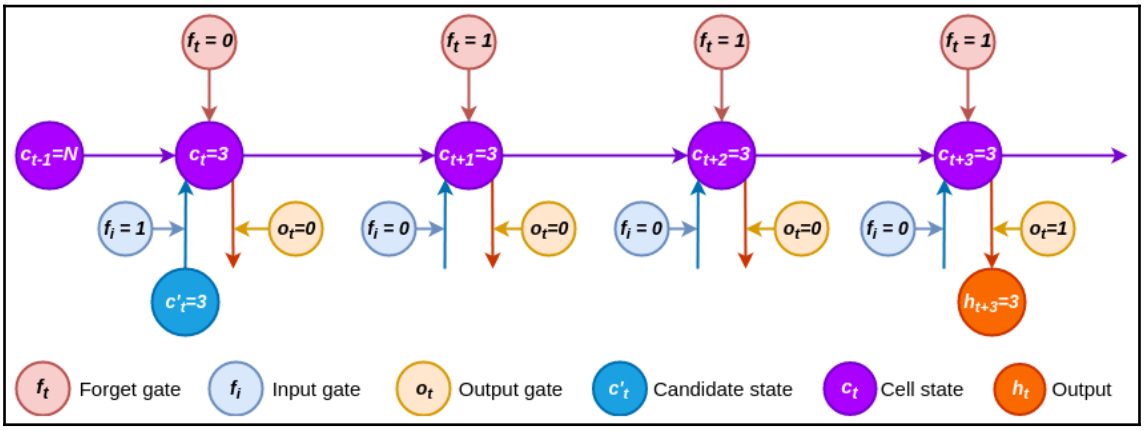

one to one
one to many
many to one
many to many
many to many
Output
State
Input
V
W
U

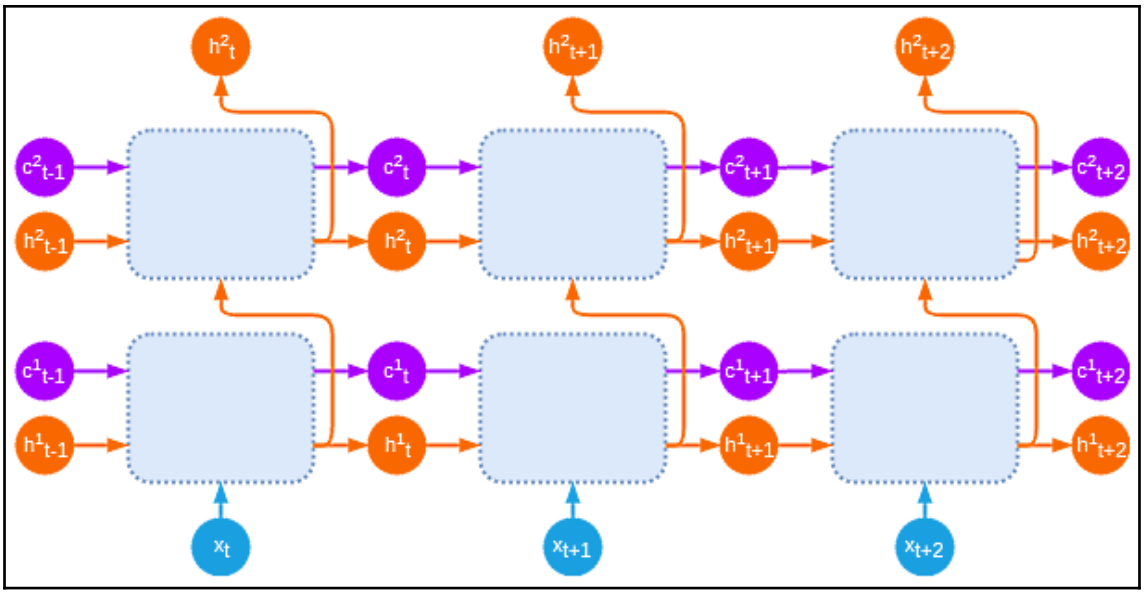

y
V
s
W
unfold
s_0
s_1
s_2
s_k
s_n
U
x
x_1
x_2
x_k
x_n

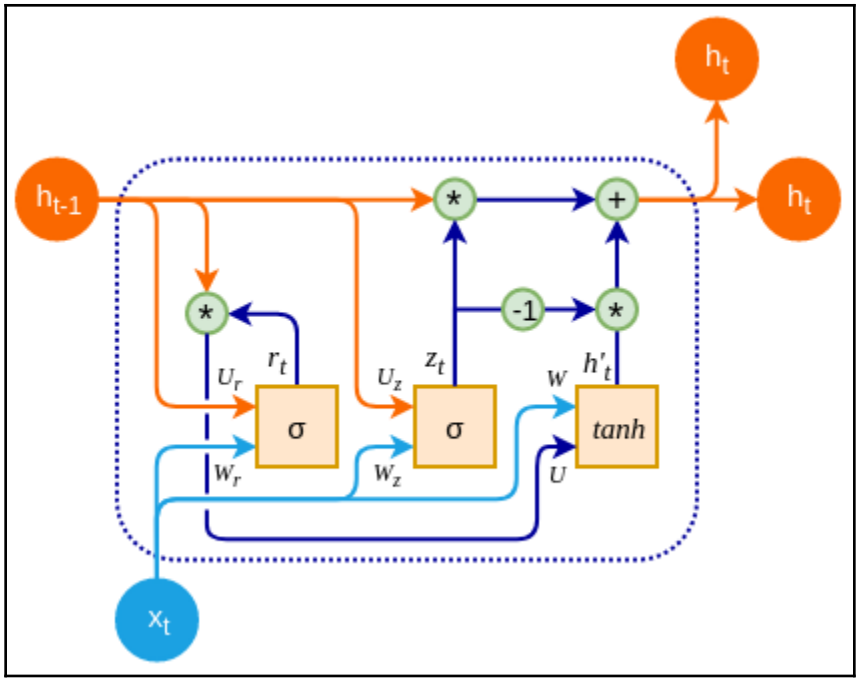

grad U
grad W
Loss
epochs

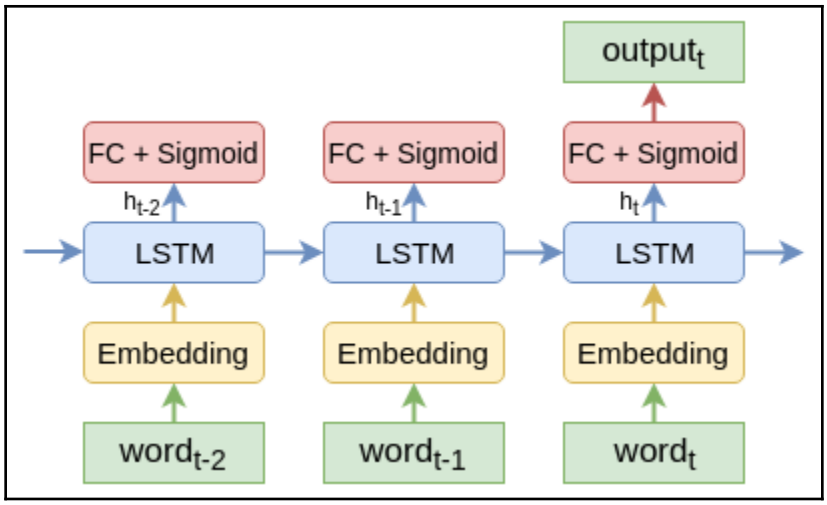

grad U
grad W
Loss
epochs


Element-wise sum
Element-wise multiplication
Element-wise tanh
Fully-connected layer with sigmoid activation
Fully-connected layer with tanh activation
Previous cell state weights
Input weights
LSTM unfolded

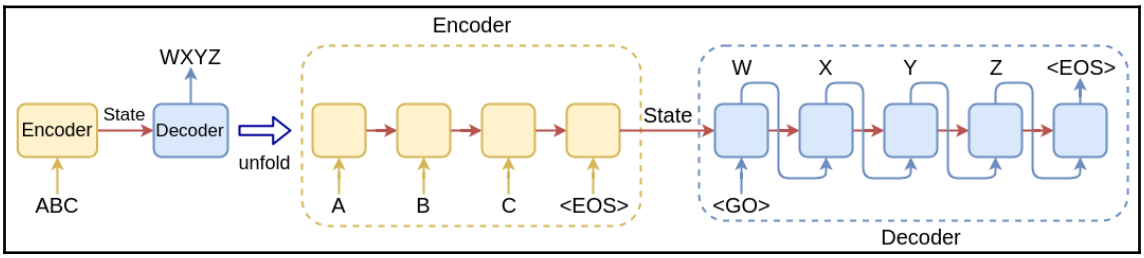

Forget gate
Input gate
Output gate
Candidate state
Cell state
Output

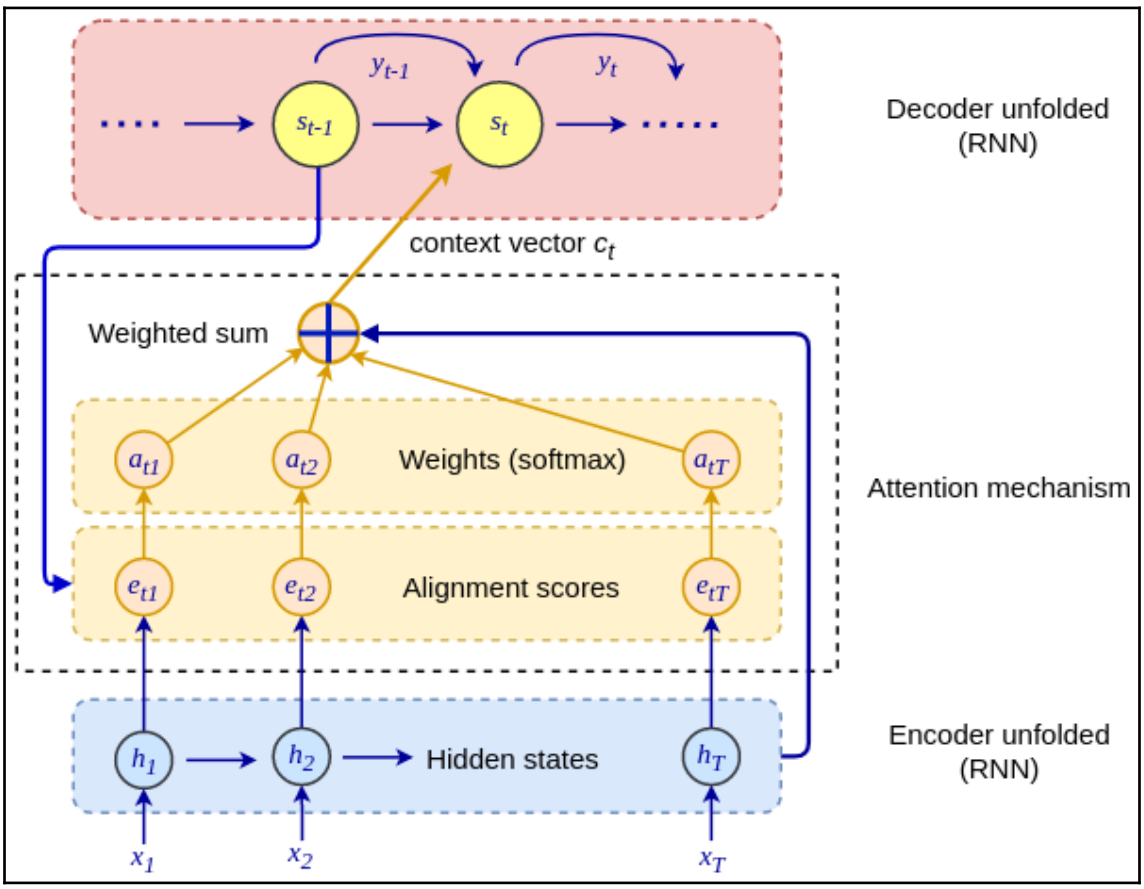

h^2_t
h^2_{t+1}
h^2_{t+2}
c^2_{t-1}
c^2_t
c^2_{t+1}
c^2_{t+2}
h^2_{t-1}
h^2_t
h^2_{t+1}
h^2_{t+2}
c^1_{t-1}
c^1_t
c^1_{t+1}
c^1_{t+2}
h^1_{t-1}
h^1_t
h^1_{t+1}
h^1_{t+2}
x_t
x_{t+1}
x_{t+2}

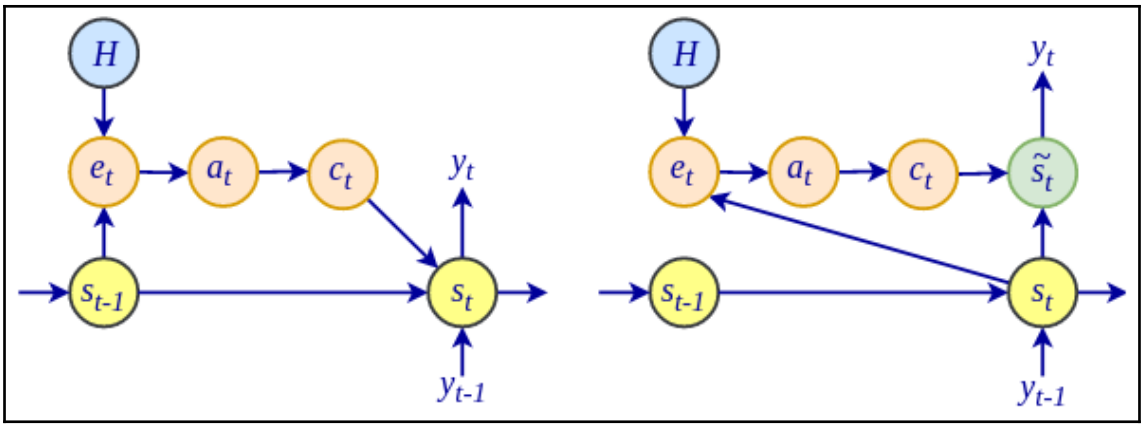

h_t
h_{t-1}
h_t
*
+
*
-1
*
r_t
z_t
h'_t
U_r
U_z
W
σ
σ
tanh
W_r
W_z
U
x_t

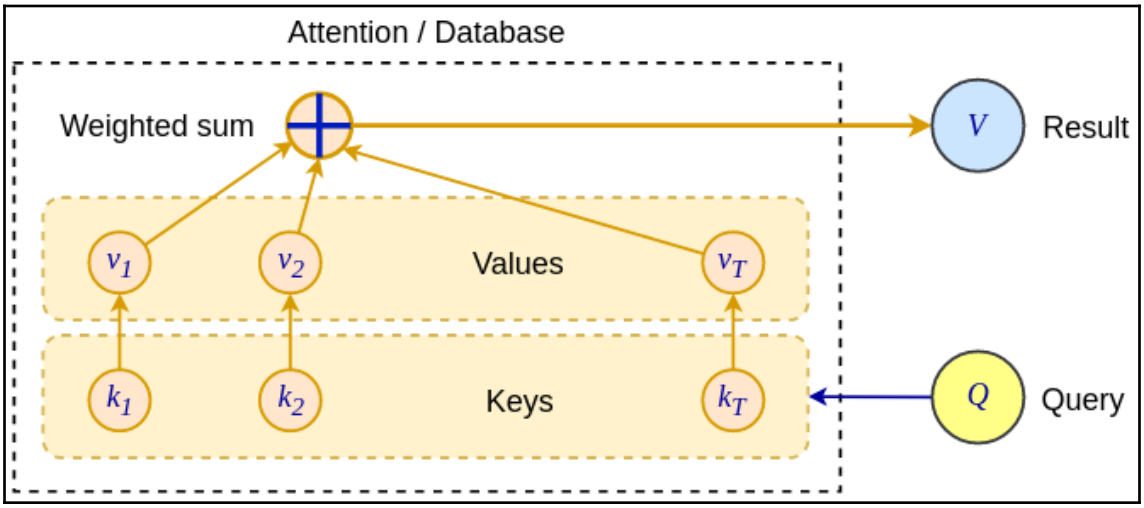

output$_t$
FC + Sigmoid
FC + Sigmoid
FC + Sigmoid
h_{t-2}
h_{t-1}
h_t
LSTM
LSTM
LSTM
Embedding
Embedding
Embedding
word$_{t-2}$
word$_{t-1}$
word$_t$

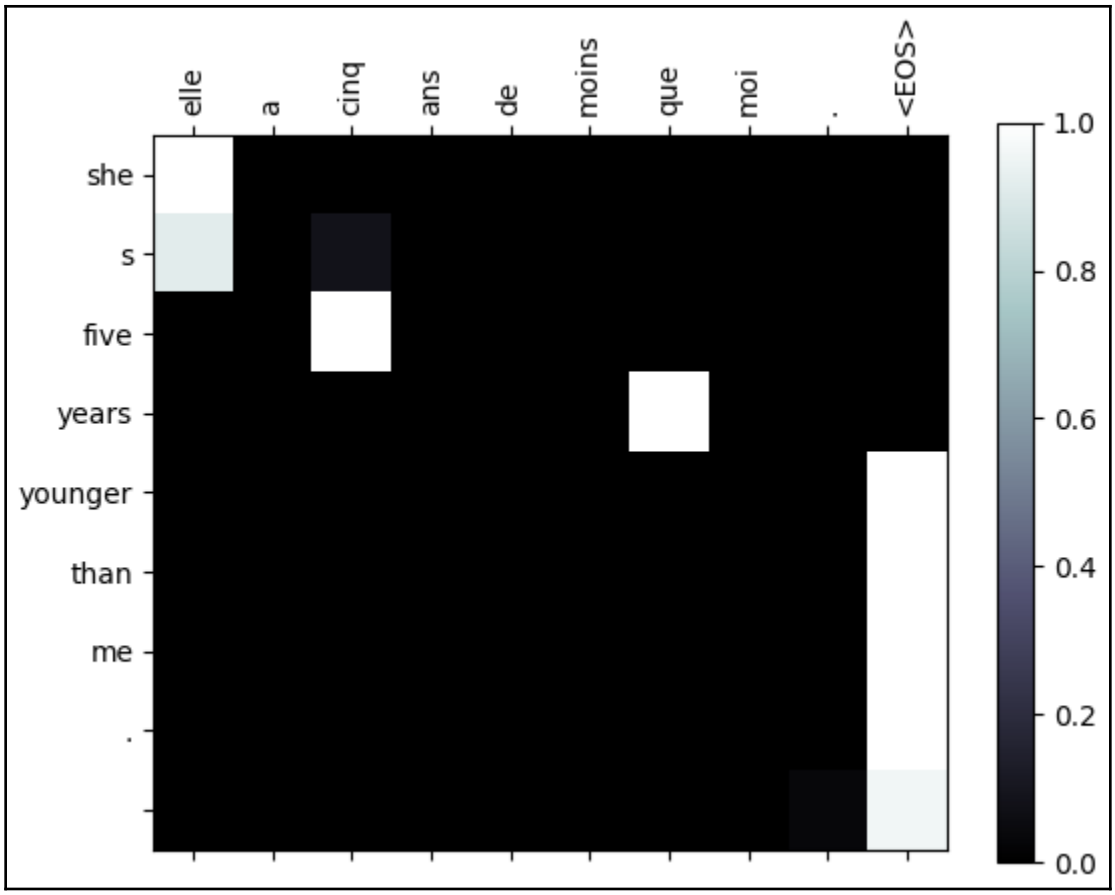
Chapter 8: Sequence-to-Sequence Models and Attention

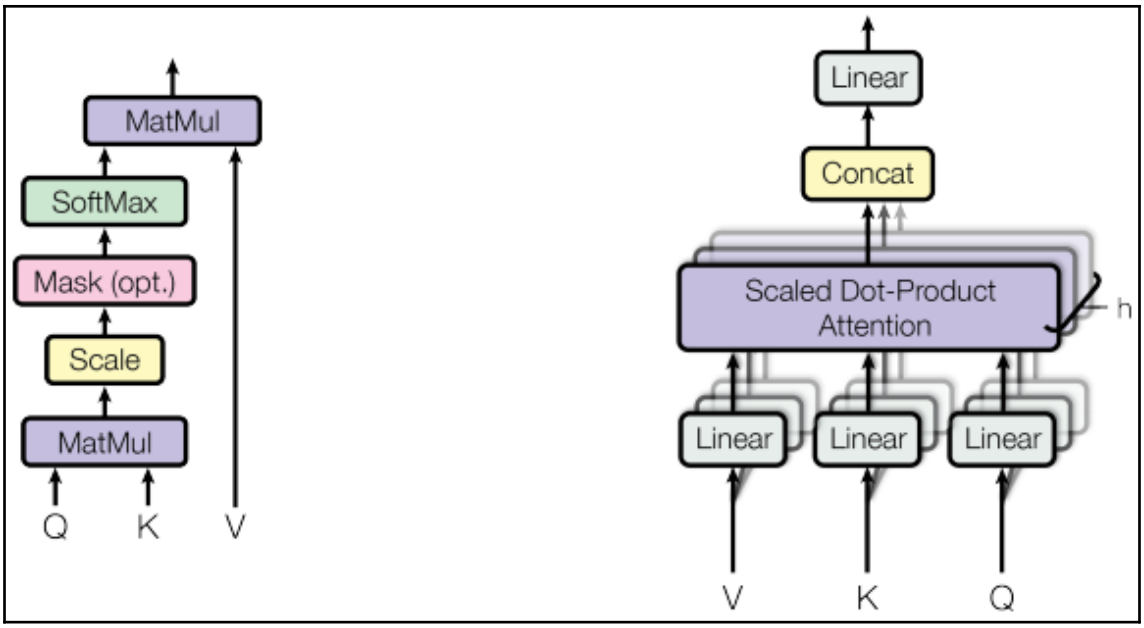


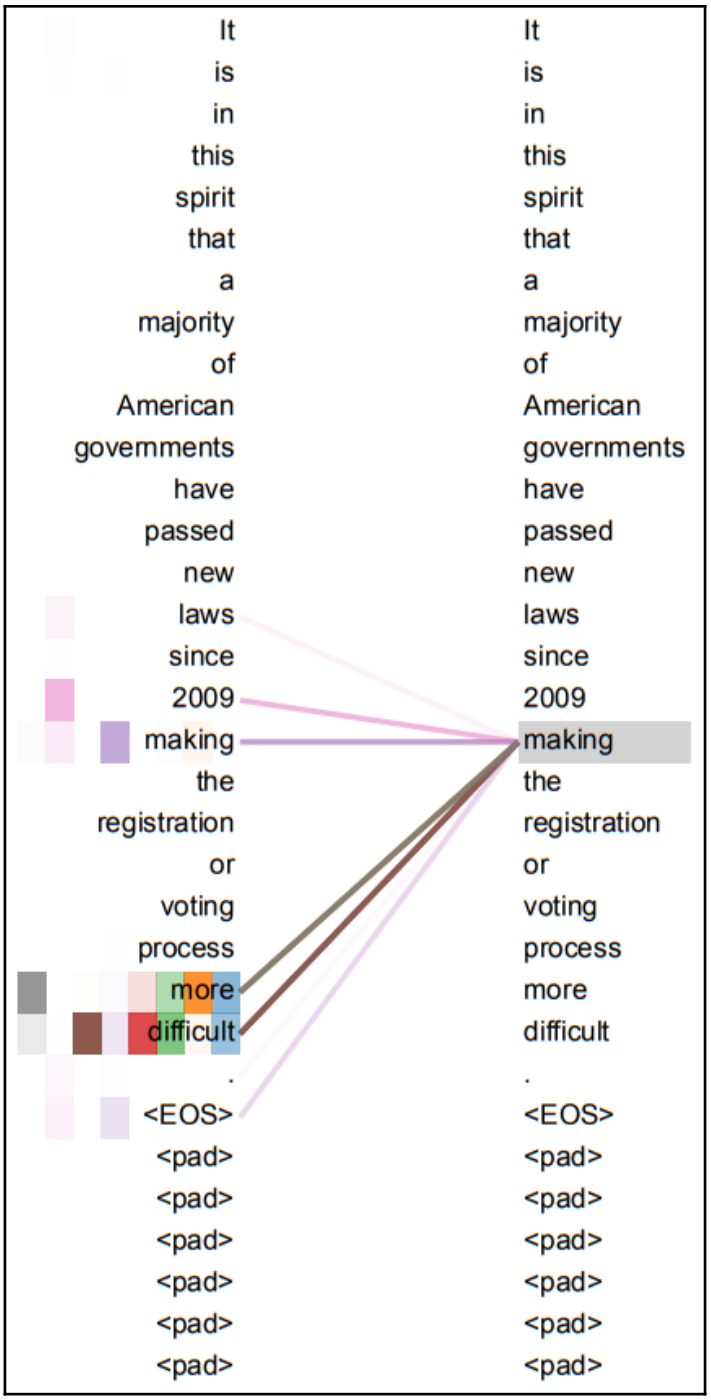


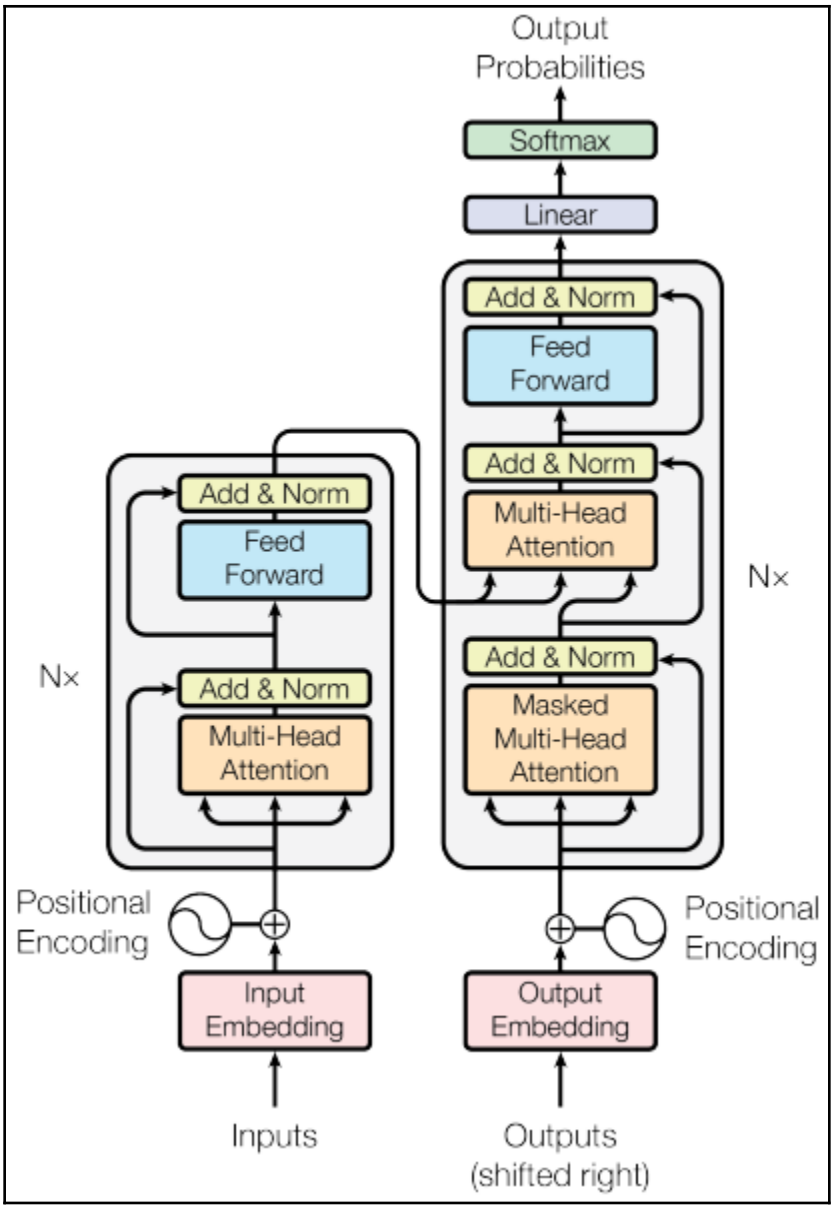

H
e_t
a_t
c_t
y_t
s_{t-1}
s_t
y_{t-1}
H
y_t
e_t
a_t
c_t
$\tilde{s}_t$
s_{t-1}
s_t
y_{t-1}

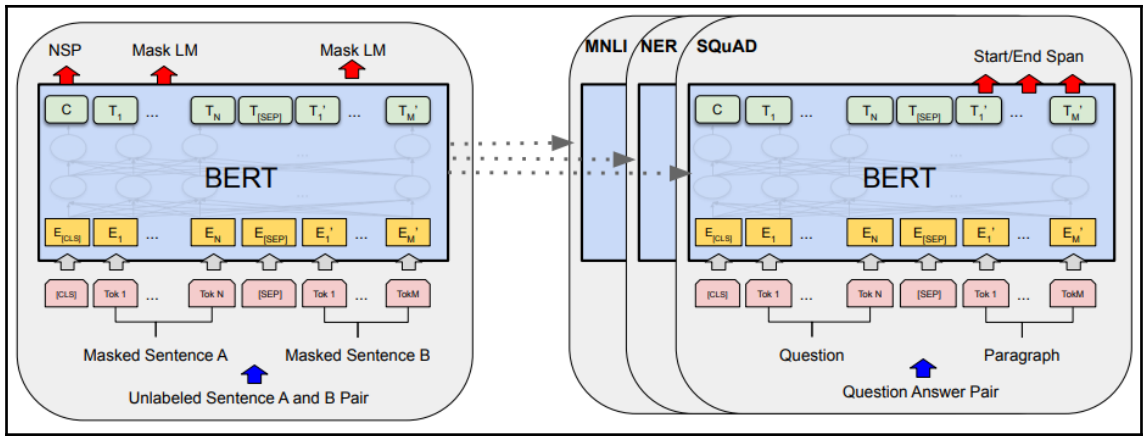

Attention / Database
Weighted sum
V
Result
v_1
v_2
Values
v_T
k_1
k_2
Keys
k_T
Q
Query

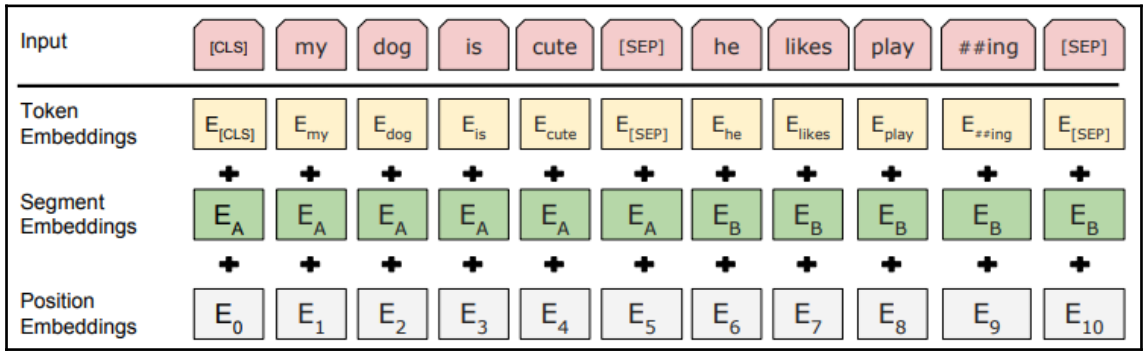

elle
a
cinq
ans
de
moins
que
moi
.
<EOS>
she
s
five
years
younger
than
me
.
1.0
0.8
0.6
0.4
0.2
0.0

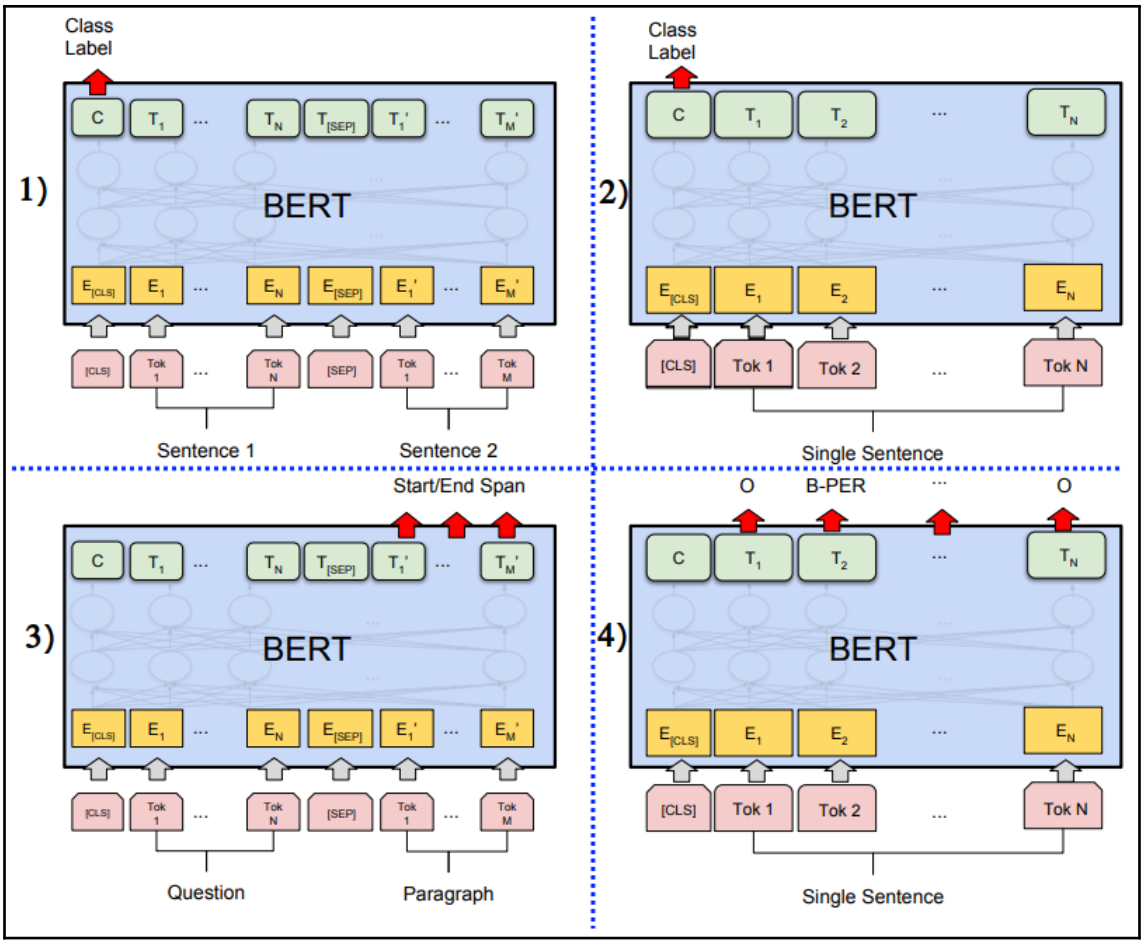

MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q

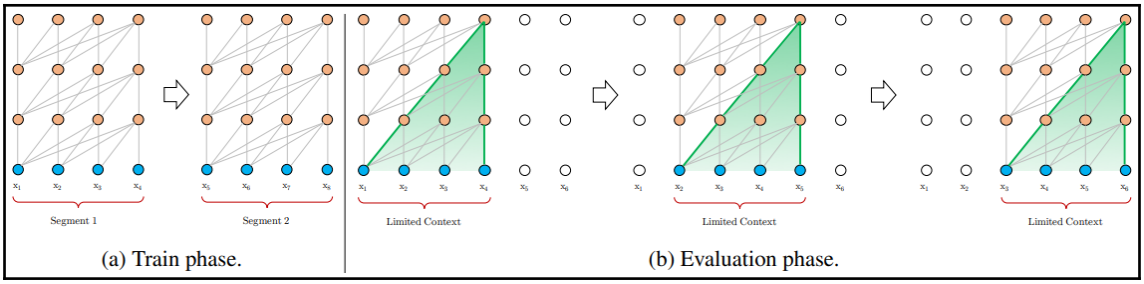

It
is
in
this
spirit
that
a
majority
of
American
governments
have
passed
new
laws
since
2009
making
the
registration
or
voting
process
more
difficult
.
<EOS>
<pad>
<pad>
<pad>
<pad>
<pad>
<pad>
It
is
in
this
spirit
that
a
majority
of
American
governments
have
passed
new
laws
since
2009
making
the
registration
or
voting
process
more
difficult
.
<EOS>
<pad>
<pad>
<pad>
<pad>
<pad>
<pad>
<pad>


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

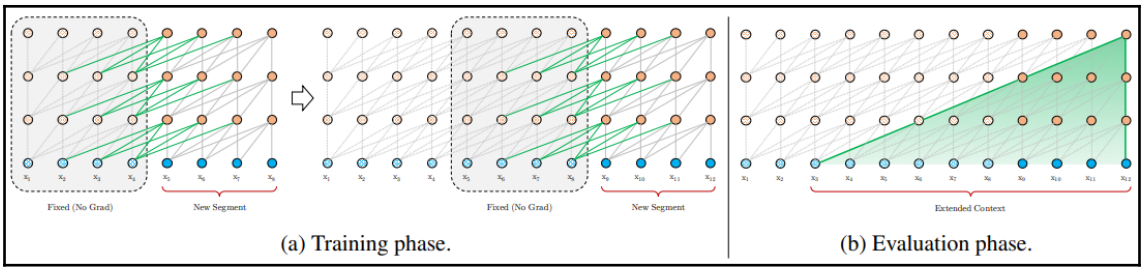

NSP
Mask LM
Mask LM
BERT
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
MNLI
NER
SQuAD
Start/End Span
BERT
Question
Paragraph
Question Answer Pair


Input
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
Segment Embeddings
Position Embeddings

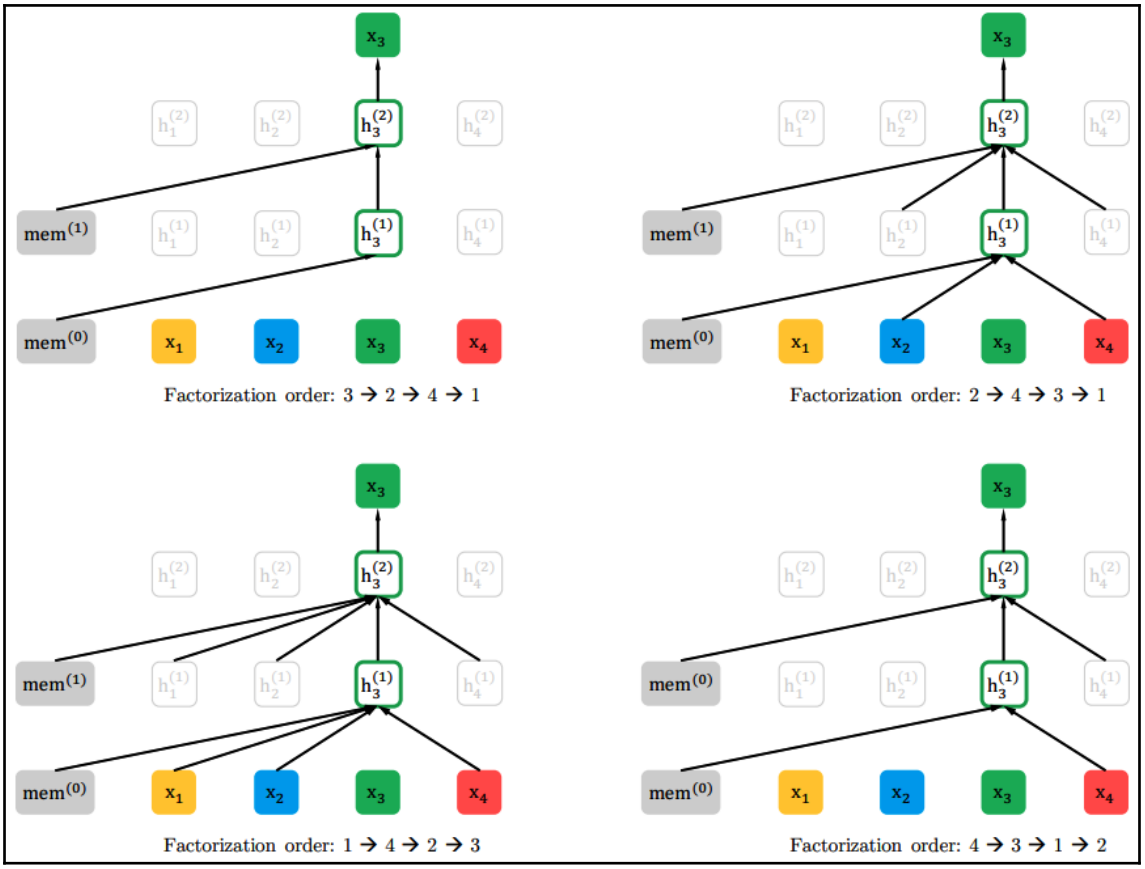




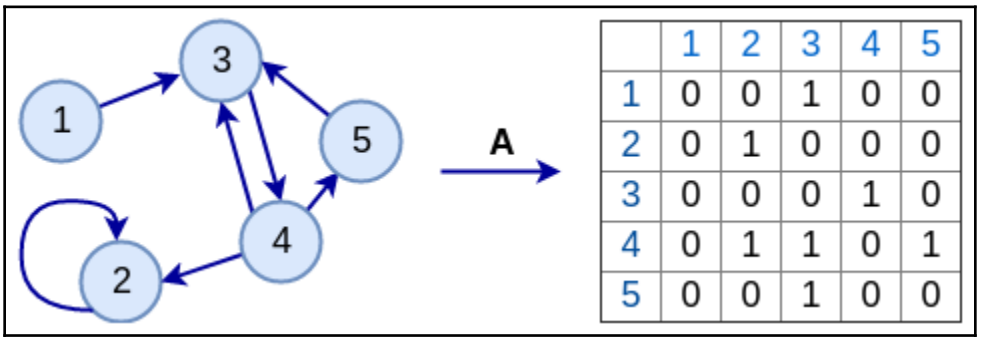
(a) Train phase. (b) Evaluation phase.



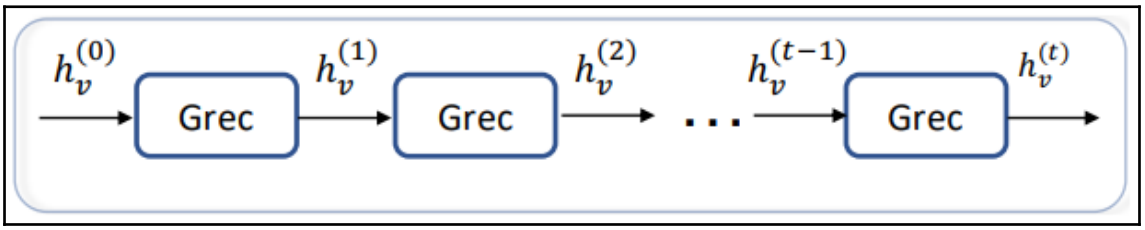
(a) Training phase. (b) Evaluation phase.

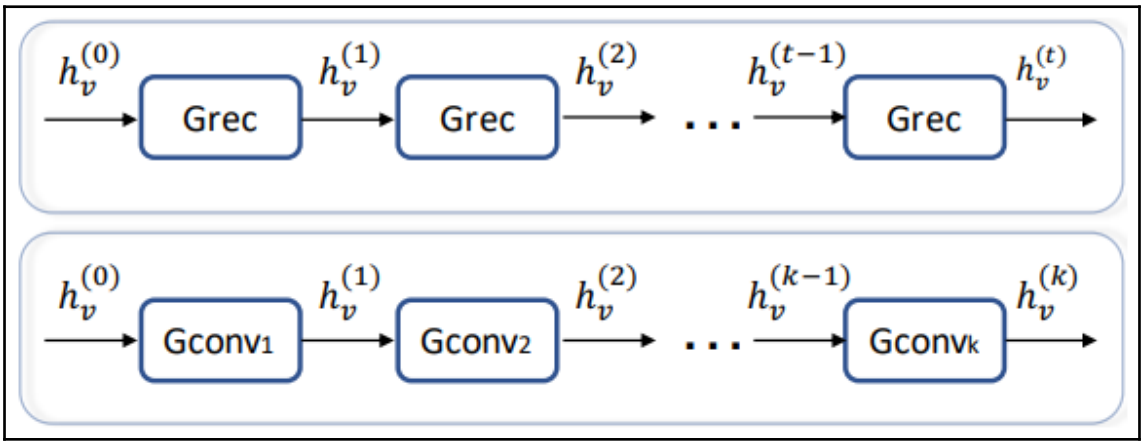


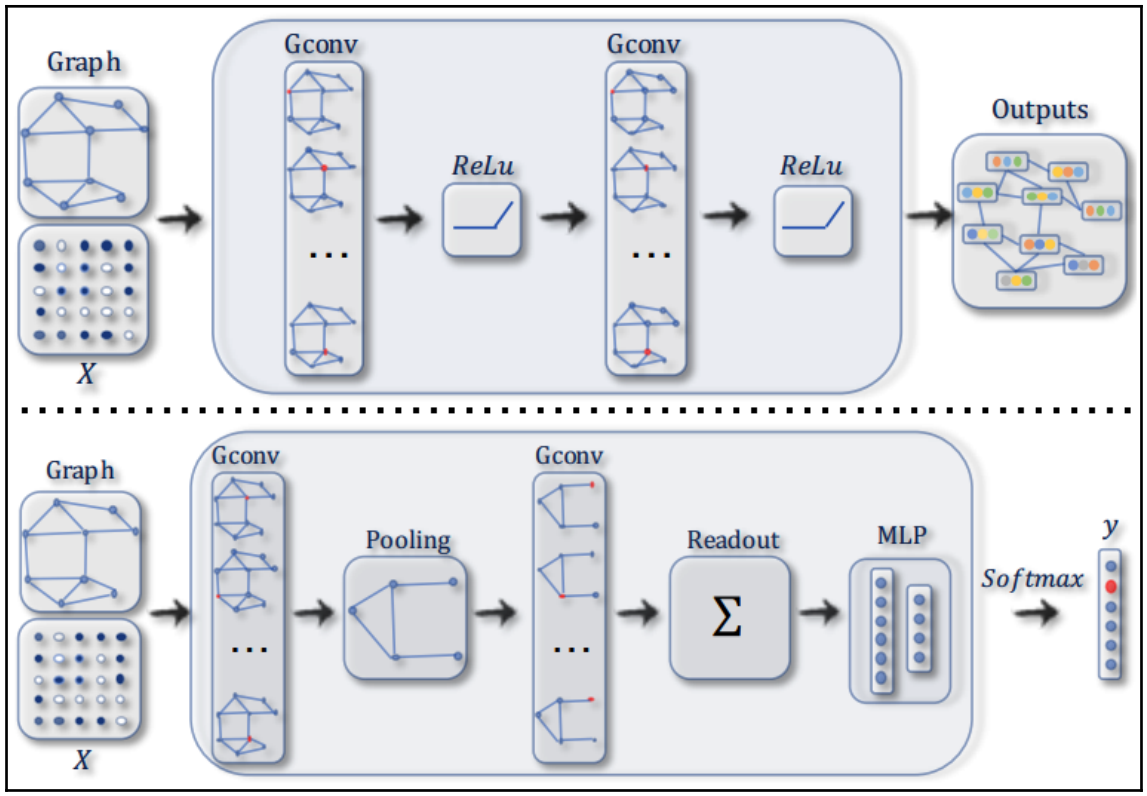
Chapter 9: Emerging Neural Network Designs

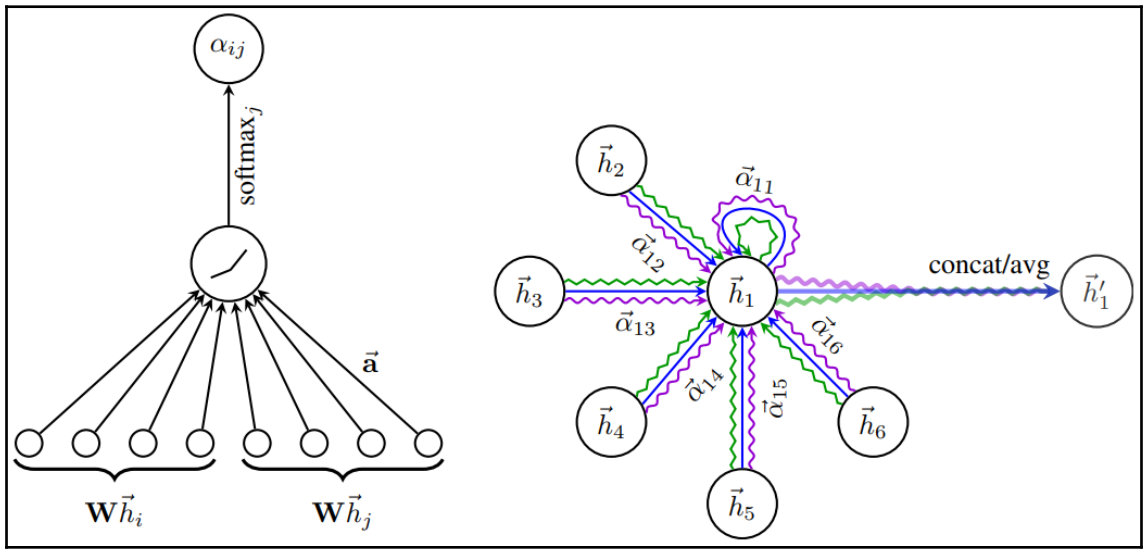


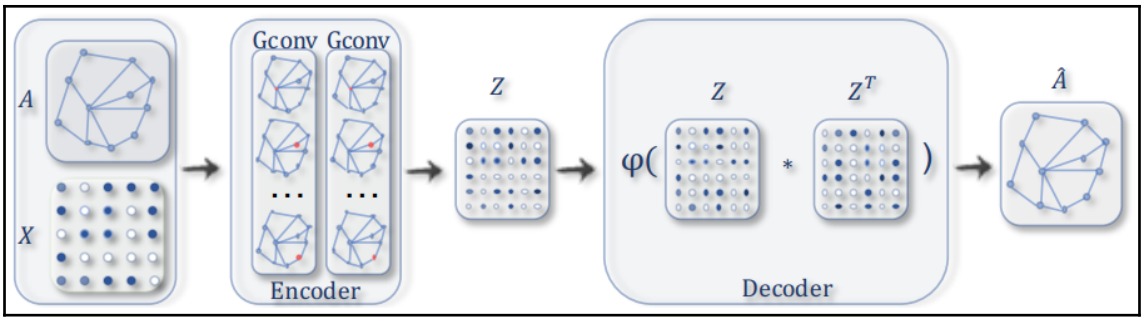
	1	2	3	4	5
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	0	1	0
4	0	1	1	0	1
5	0	0	1	0	0

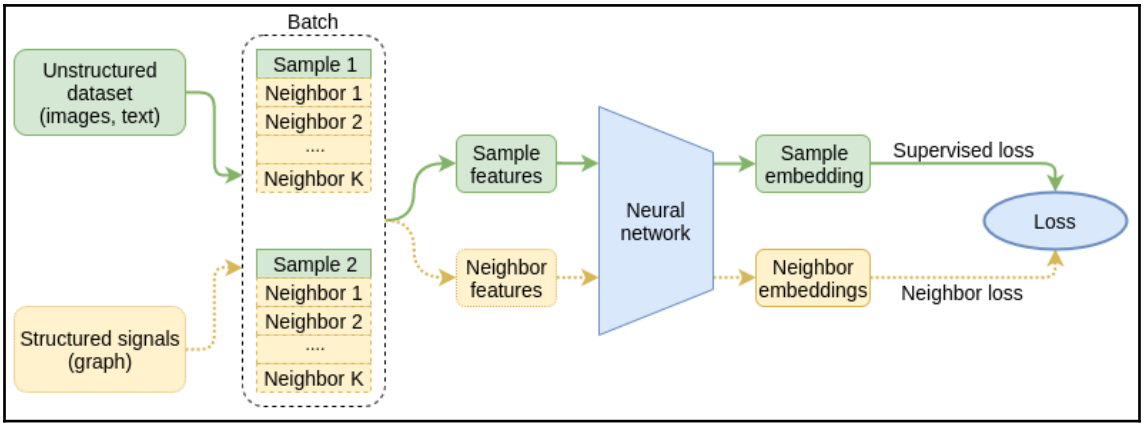


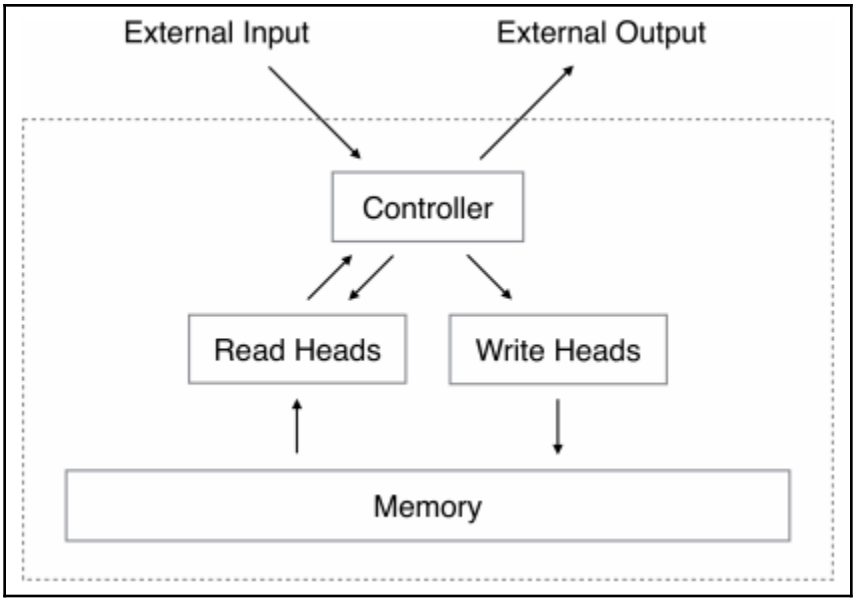


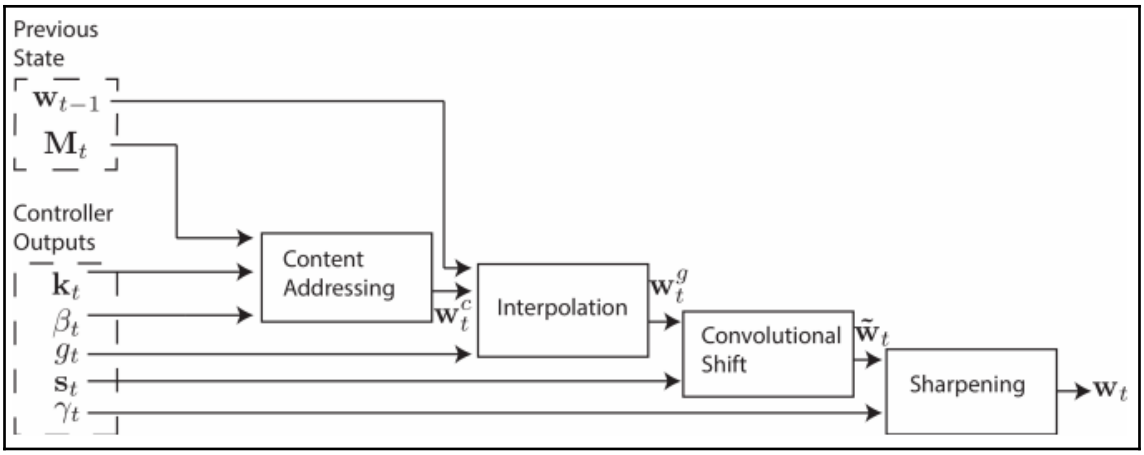

Graph
X
Gconv
ReLu
Gconv
ReLu
Outputs
Graph
X
Gconv
Pooling
Gconv
Readout
Σ
MLP
Softmax
y

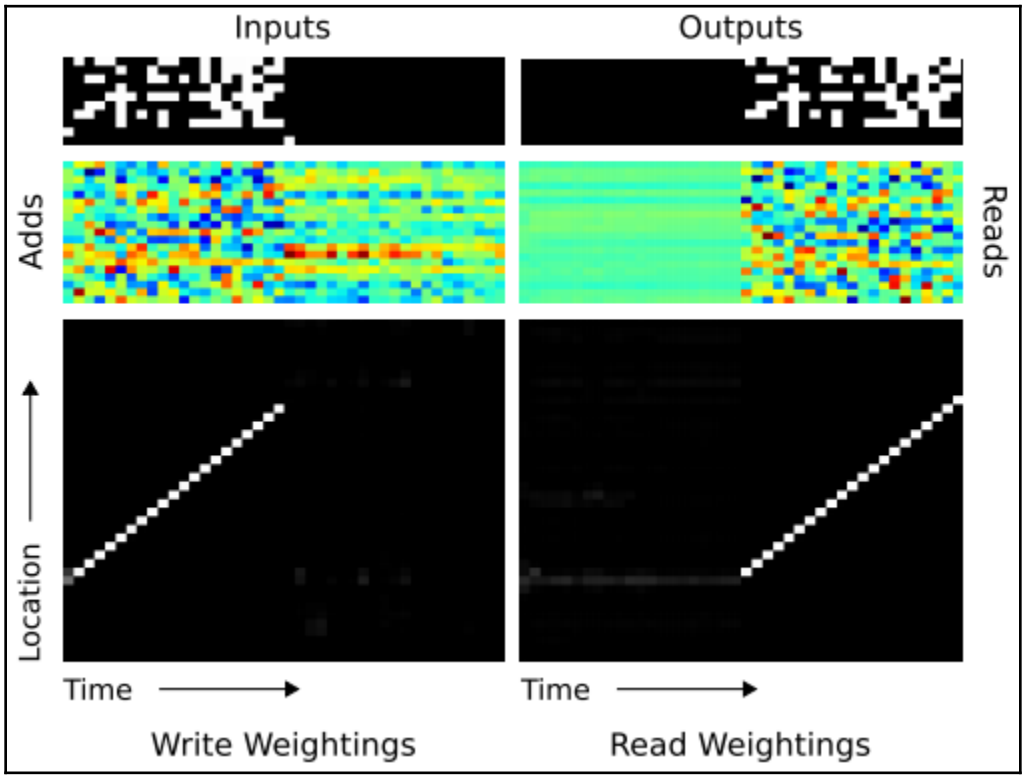

α_{ij}
softmax$_j$
$\vec{\mathbf{a}}$
$\mathbf{W}\vec{h}_i$
$\mathbf{W}\vec{h}_j$
$\vec{h}_1$
$\vec{h}_2$
$\vec{h}_3$
$\vec{h}_4$
$\vec{h}_5$
$\vec{h}_6$
$\vec{\alpha}_{11}$
$\vec{\alpha}_{12}$
$\vec{\alpha}_{13}$
$\vec{\alpha}_{14}$
$\vec{\alpha}_{15}$
$\vec{\alpha}_{16}$
concat/avg
$\vec{h}'_1$


A
X
Gconv
Gconv
Encoder
Z
Z
Z^T
φ(
*
)
Decoder
Â

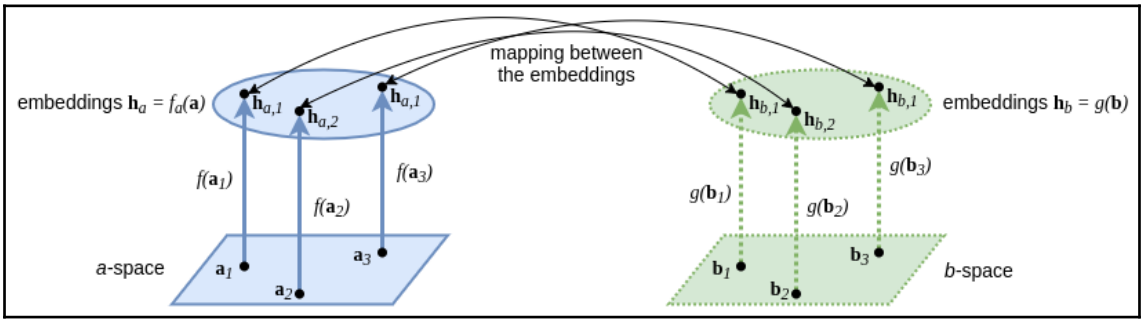

Batch
Unstructured dataset (images, text)
Structured signals (graph)
Sample 1
Neighbor 1
Neighbor 2
....
Neighbor K
Sample 2
Neighbor 1
Neighbor 2
....
Neighbor K
Sample features
Neighbor features
Neural network
Sample embedding
Neighbor embeddings
Supervised loss
Loss
Neighbor loss

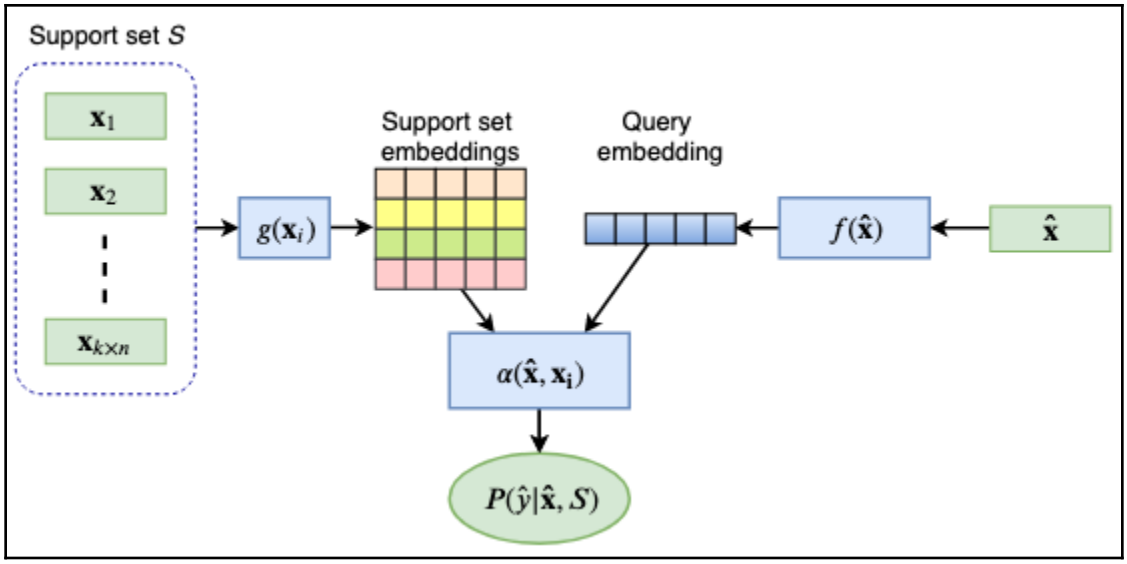

External Input
External Output
Controller
Read Heads
Write Heads
Memory

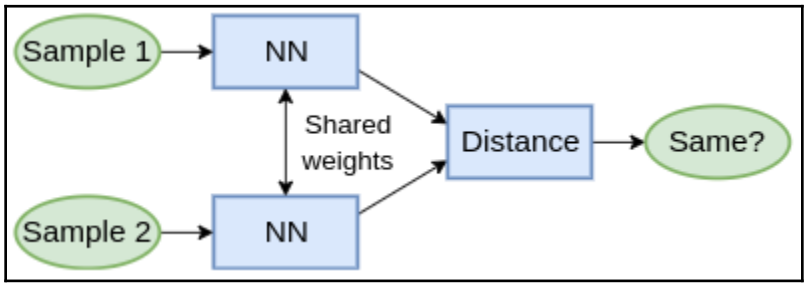

Previous State
$\mathbf{w}_{t-1}$
$\mathbf{M}_t$
Controller Outputs
$\mathbf{k}_t$
β_t
g_t
$\mathbf{s}_t$
γ_t
Content Addressing
$\mathbf{w}_t^c$
Interpolation
$\mathbf{w}_t^g$
Convolutional Shift
$\tilde{\mathbf{w}}_t$
Sharpening
$\mathbf{w}_t$

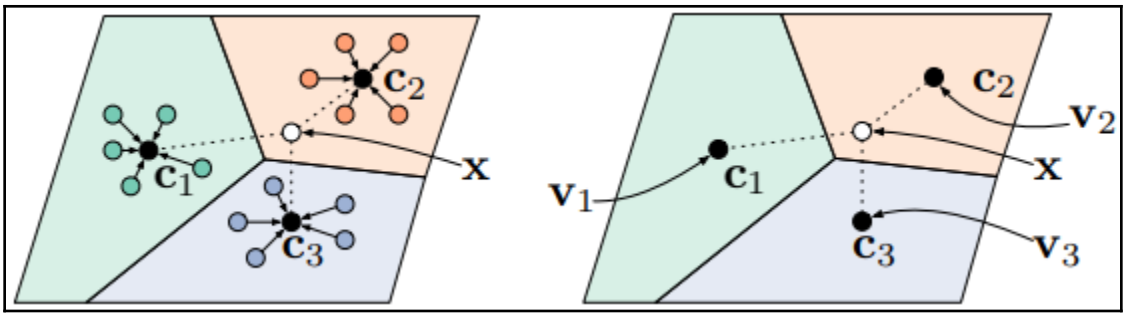

Inputs
Outputs
Adds
Reads
Location
Time
Time
Write Weightings
Read Weightings

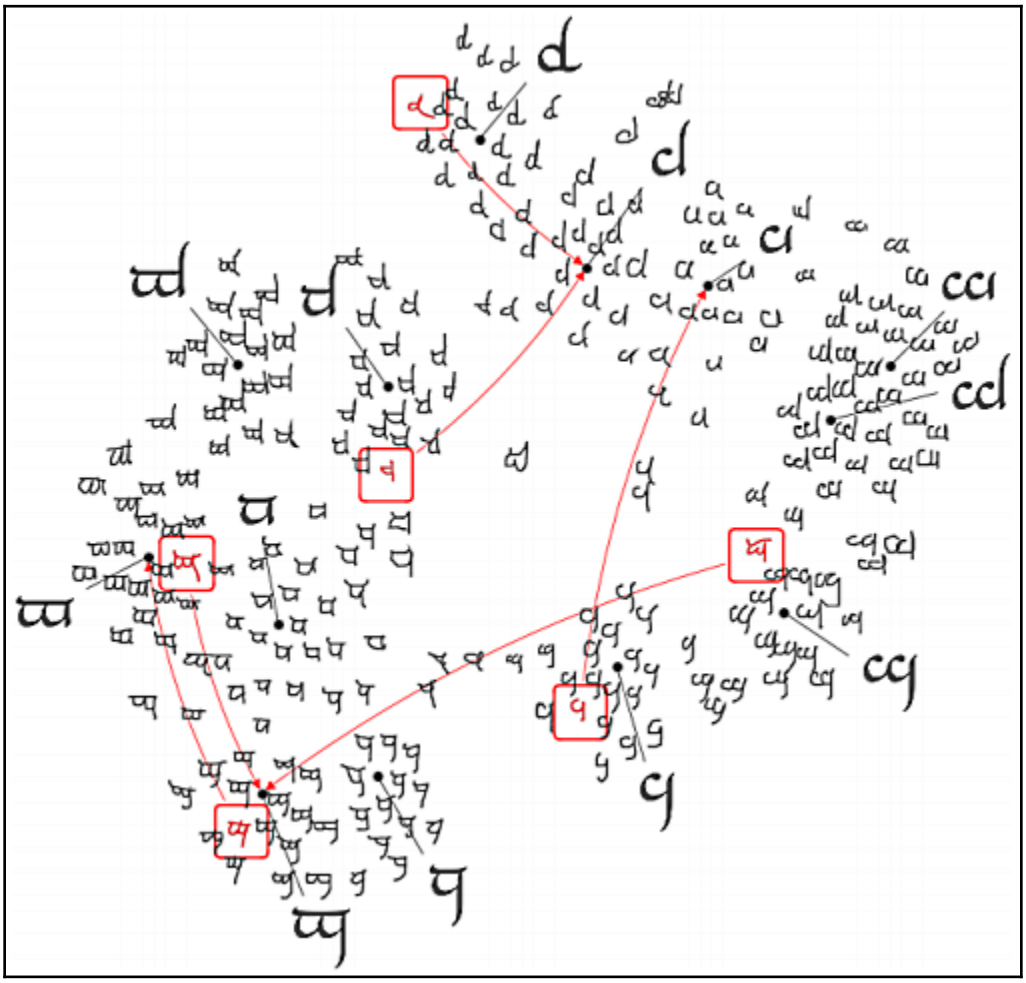
Chapter 10: Meta Learning

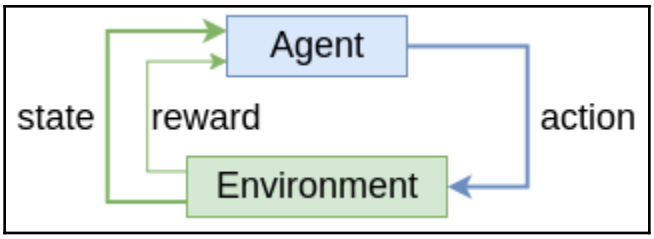


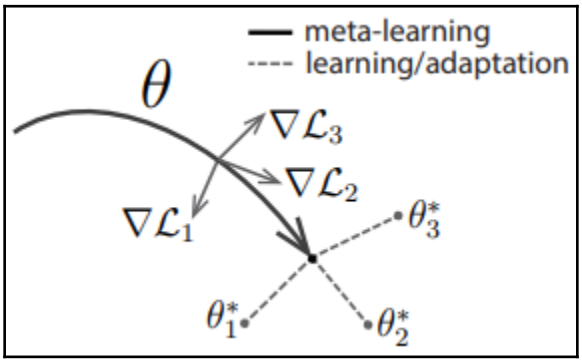




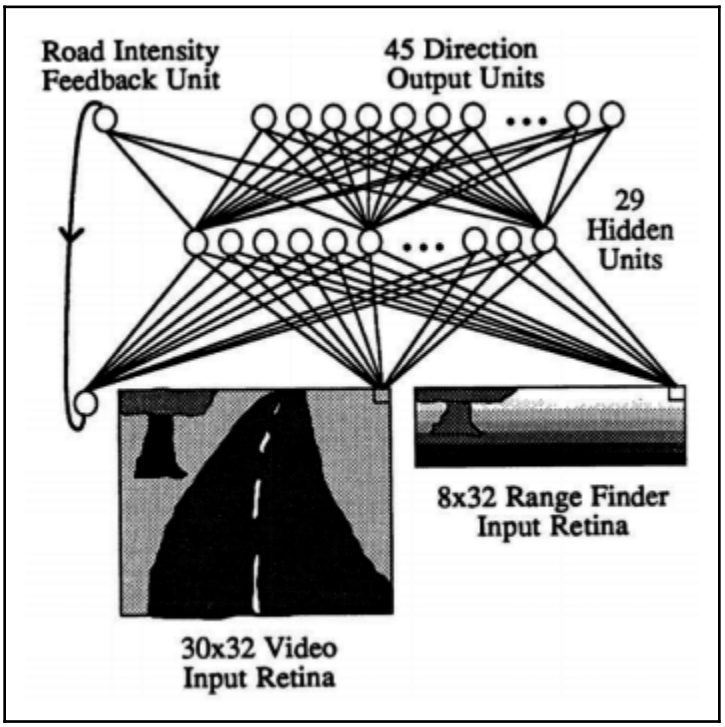

c1
c2
c3
x
v1
v2
v3

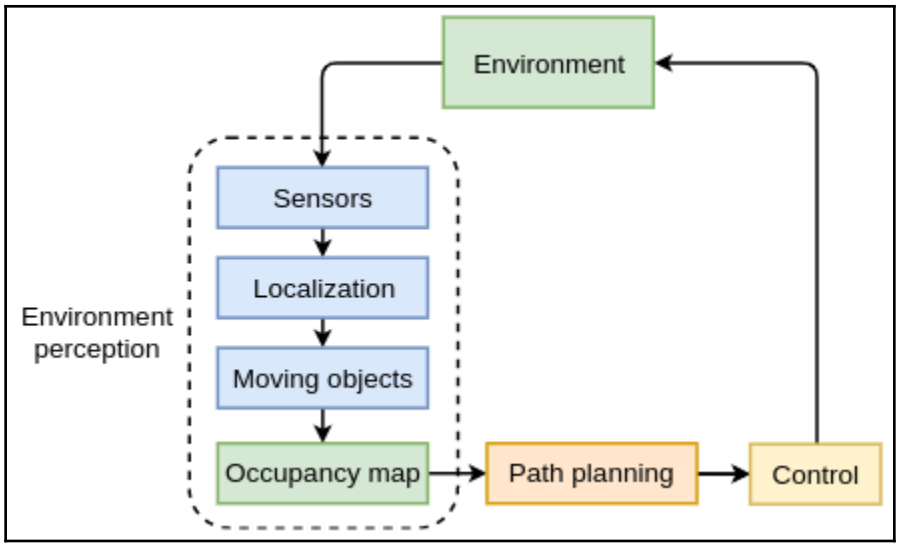


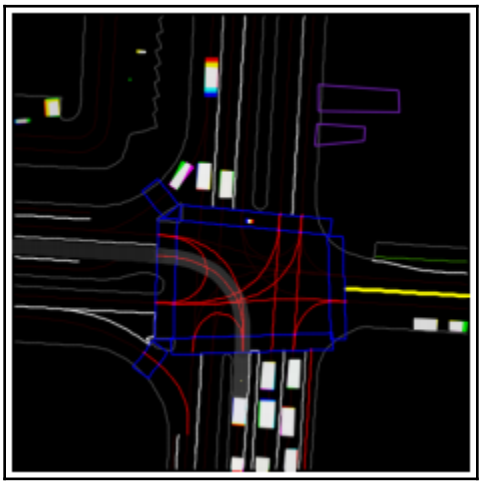

Agent
state
reward
action
Environment

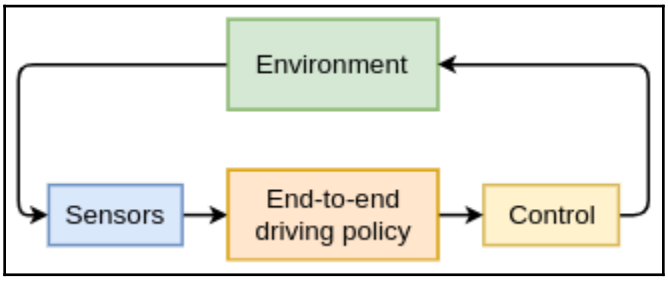

meta-learning
learning/adaptation
θ
$\nabla\mathcal{L}_3$
$\nabla\mathcal{L}_2$
$\nabla\mathcal{L}_1$
θ_3^*
θ_1^*
θ_2^*

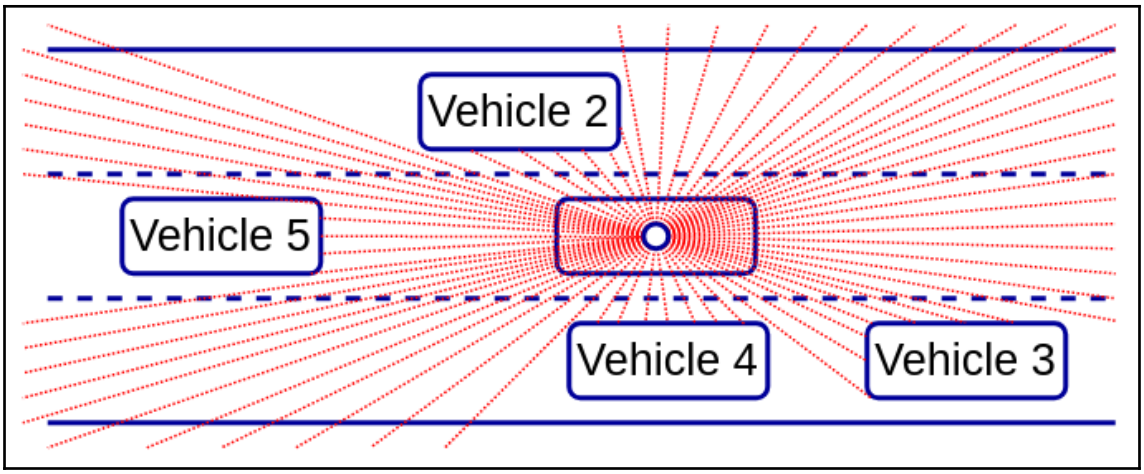
Chapter 11: Deep Learning for Autonomous Vehicles

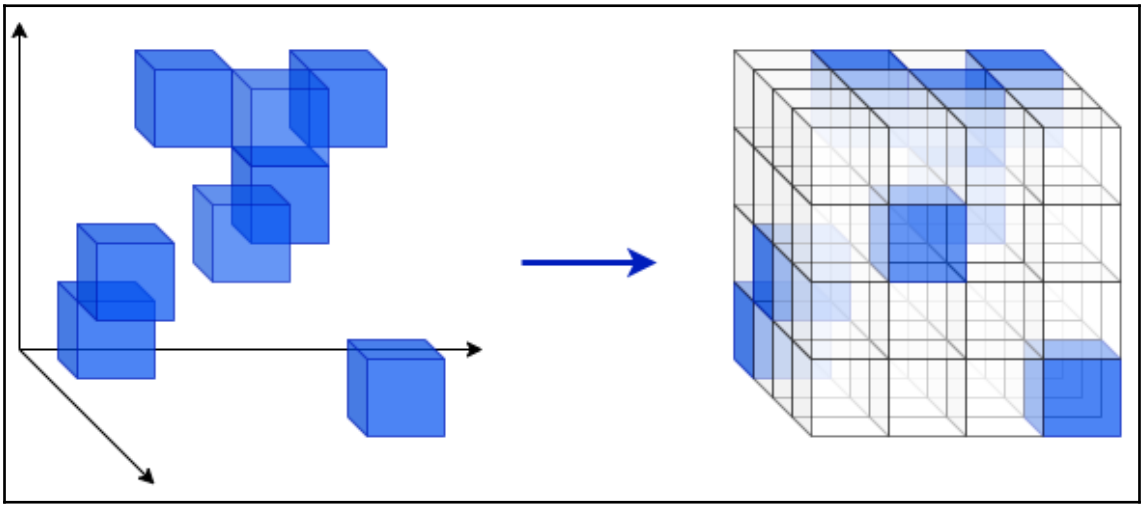


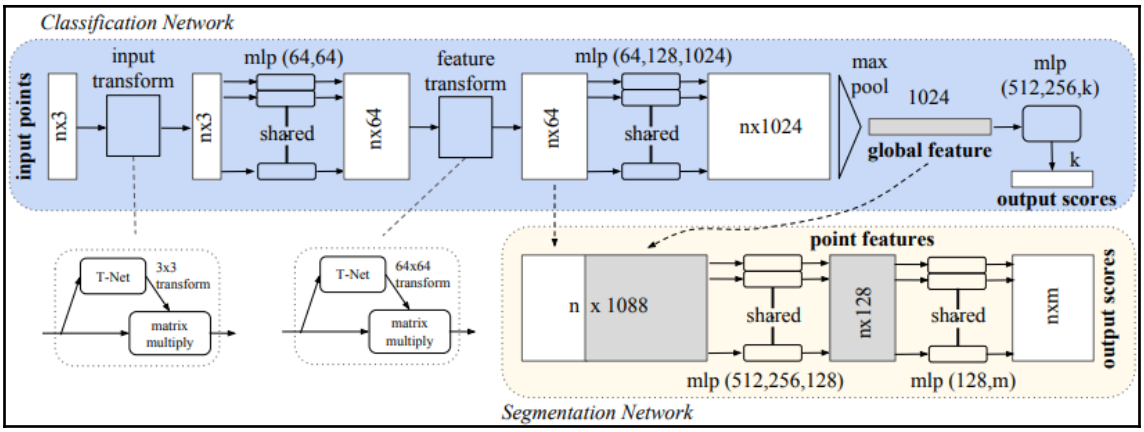

Environment
Sensors
Localization
Moving objects
Environment perception
Occupancy map
Path planning
Control

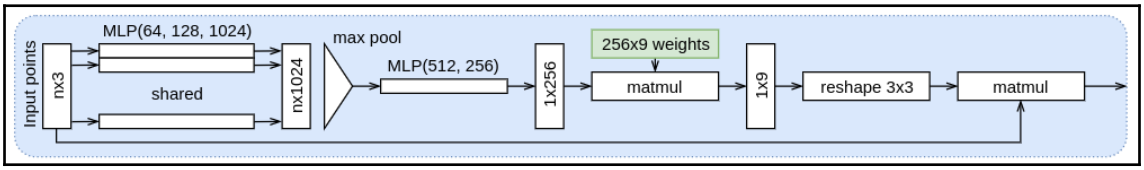


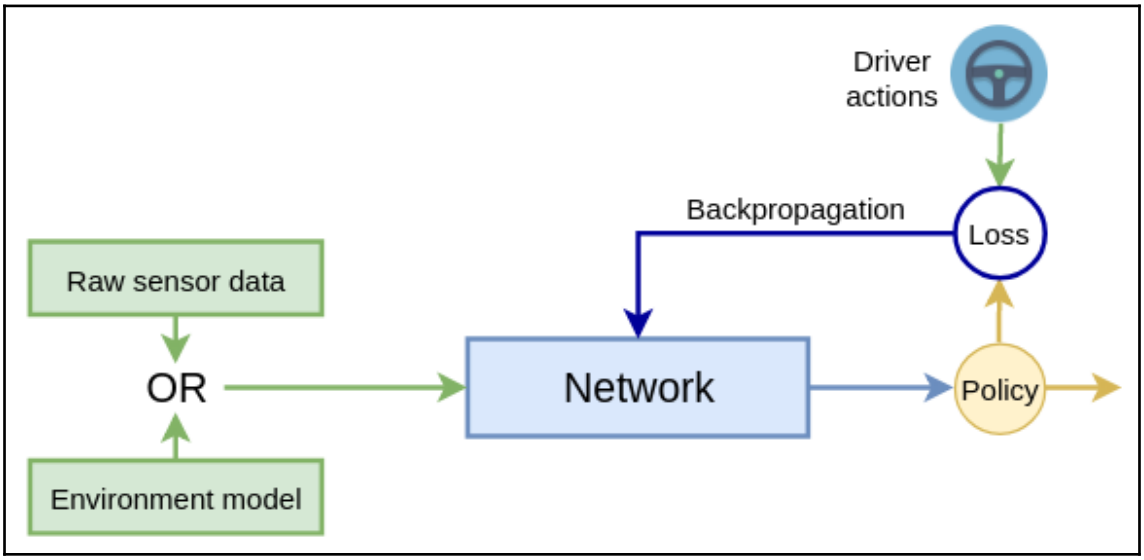

Environment
Sensors
End-to-end driving policy
Control

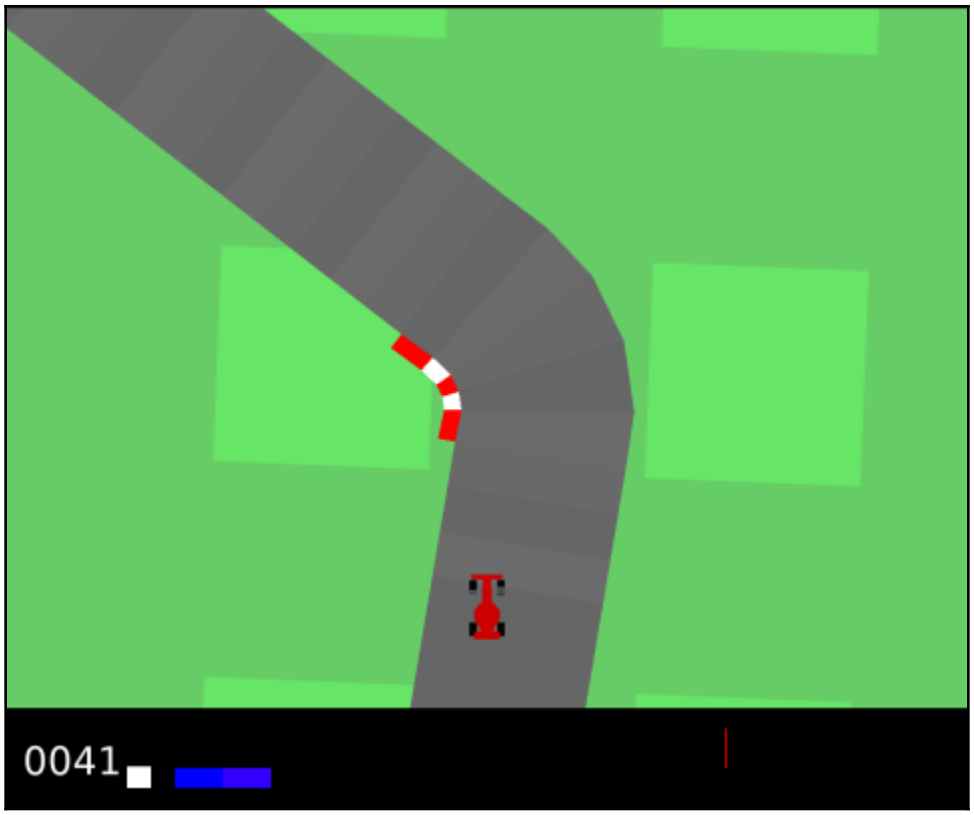

Vehicle 2
Vehicle 5
Vehicle 4
Vehicle 3

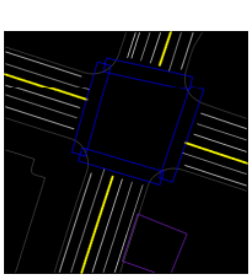



Classification Network
input points
nx3
input transform
nx3
mlp (64,64)
shared
nx64
feature transform
nx64
mlp (64,128,1024)
shared
nx1024
max pool
1024
global feature
mlp (512,256,k)
k
output scores
T-Net
3x3 transform
matrix multiply
T-Net
64x64 transform
matrix multiply
point features
n x 1088
shared
nx128
shared
nxm
output scores
mlp (512,256,128)
mlp (128,m)
Segmentation Network

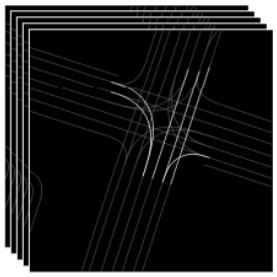

Input points
nx3
MLP(64, 128, 1024)
shared
nx1024
max pool
MLP(512, 256)
1x256
256x9 weights
matmul
1x9
reshape 3x3
matmul


Driver actions
Backpropagation
Loss
Raw sensor data
OR
Environment model
Network
Policy

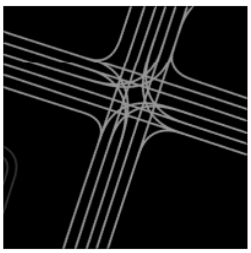

0041



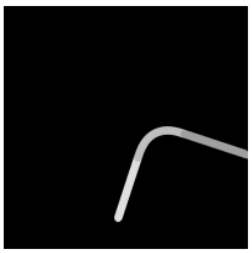
(a) Roadmap



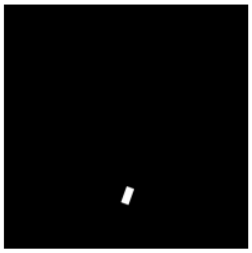
(b) Traffic Lights



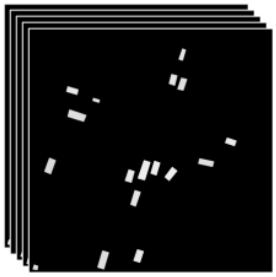
(c) Speed Limit



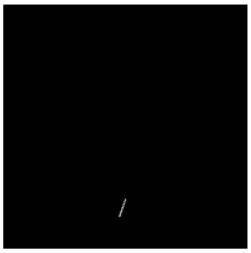
(d) Route



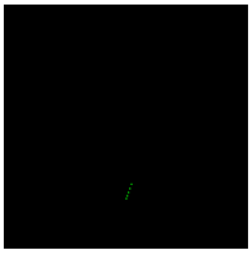
(e) Current Agent Box



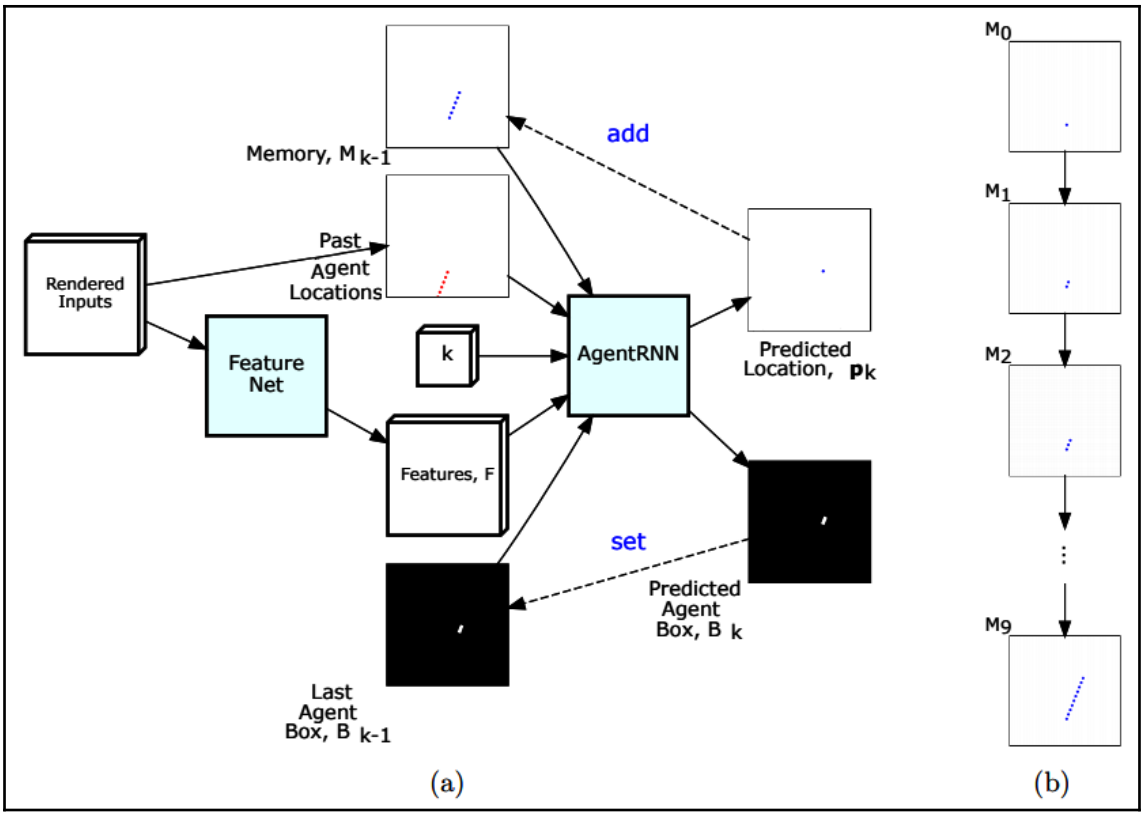
(f) Dynamic Boxes

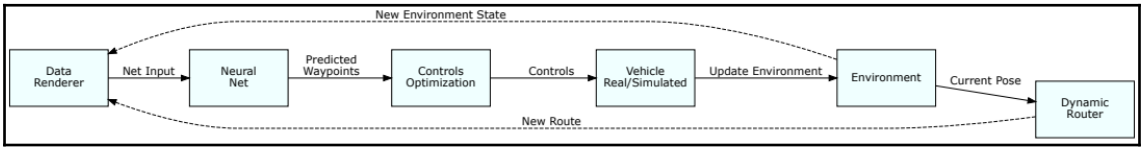


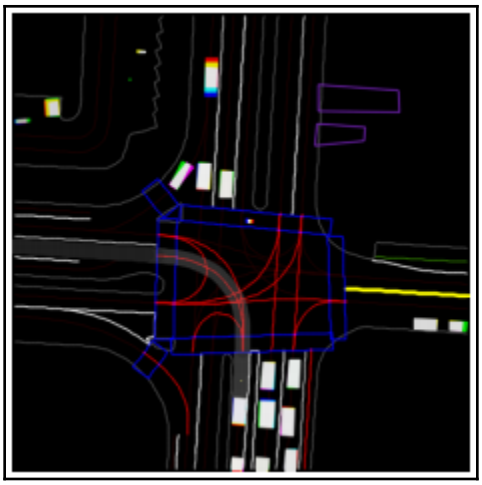
(g) Past Agent Poses

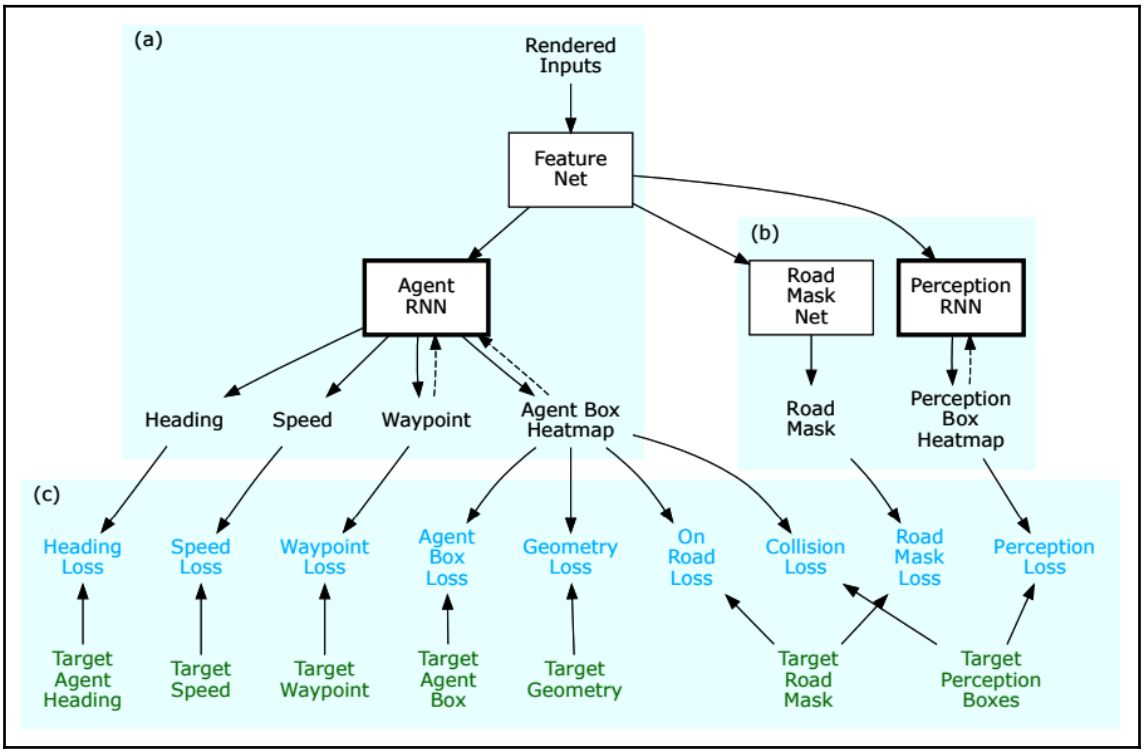
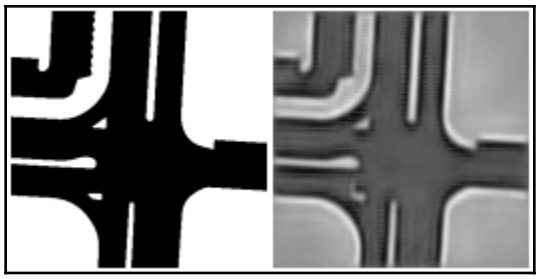
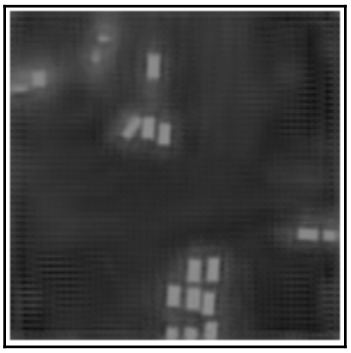


(h) Future Agent Poses


Memory, M k-1
add
Past
Agent
Locations
Rendered
Inputs
Feature
Net
k
AgentRNN
Predicted
Location, pk
Features, F
set
Predicted
Agent
Box, B k
Last
Agent
Box, B k-1
(a)
M0
M1
M2
M9
(b)


New Environment State
Data
Renderer
Net Input
Neural
Net
Predicted
Waypoints
Controls
Optimization
Controls
Vehicle
Real/Simulated
Update Environment
Environment
Current Pose
Dynamic
Router
New Route




(a)
Rendered Inputs
Feature Net
Agent RNN
Heading
Speed
Waypoint
Agent Box Heatmap
(b)
Road Mask Net
Perception RNN
Road Mask
Perception Box Heatmap
(c)
Heading Loss
Speed Loss
Waypoint Loss
Agent Box Loss
Geometry Loss
On Road Loss
Collision Loss
Road Mask Loss
Perception Loss
Target Agent Heading
Target Speed
Target Waypoint
Target Agent Box
Target Geometry
Target Road Mask
Target Perception Boxes