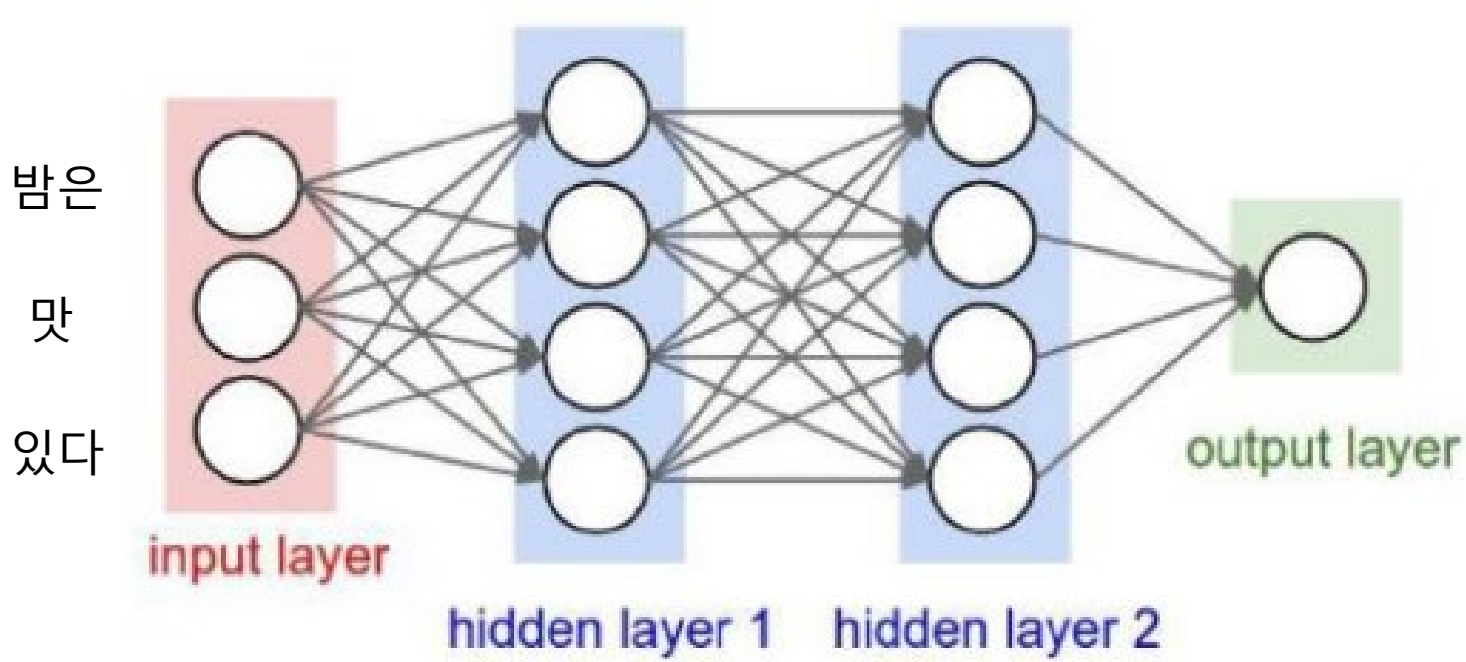
AI 9주차 RNN

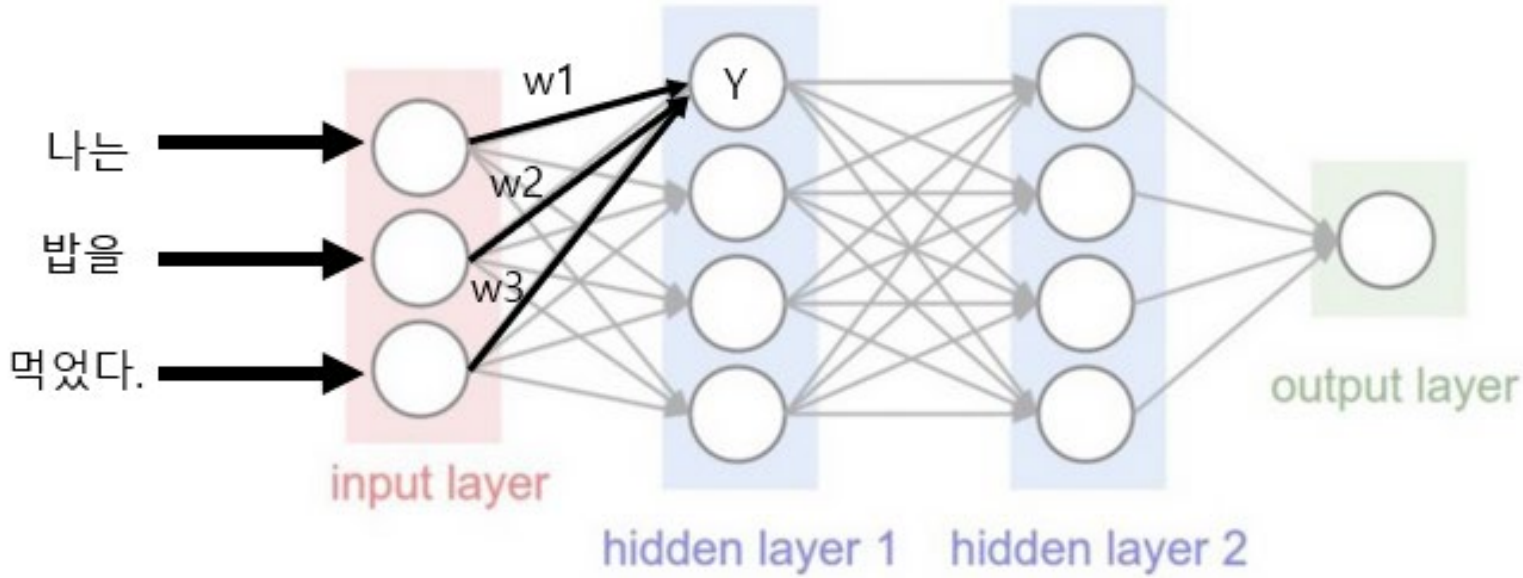
그럼 DNN으로 자연어 구현이 잘 될까?

수 많은 선이 최적의 선을 구하는데, 그럼 어려운 문제를 잘 해결할 것 같다.

DNN으로 문장 넣는다면?

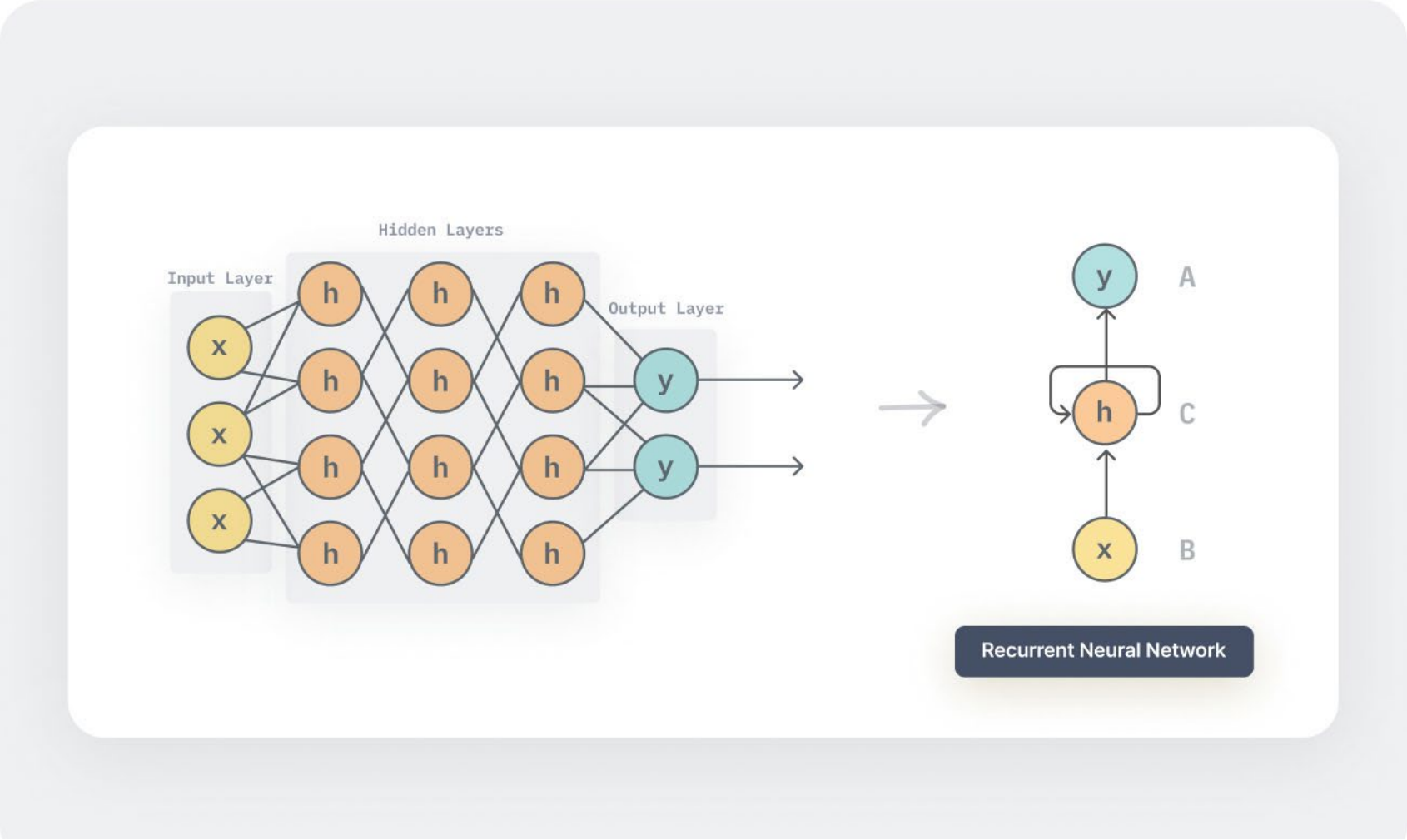


문장에 대한 흐름 파악?

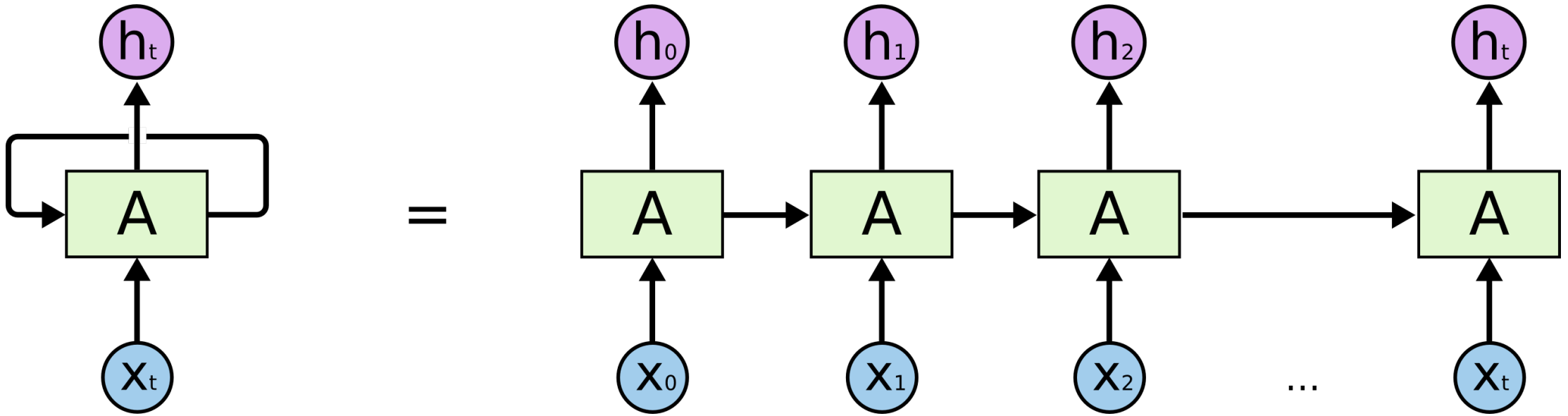


Y = 나는 * w1 + 밥을 * w2 + 먹었다. * w3

순서를 고려한 모델?


Hidden Layers
Input Layer
Output Layer
x
x
x
h
y
y
A
C
B
Recurrent Neural Network

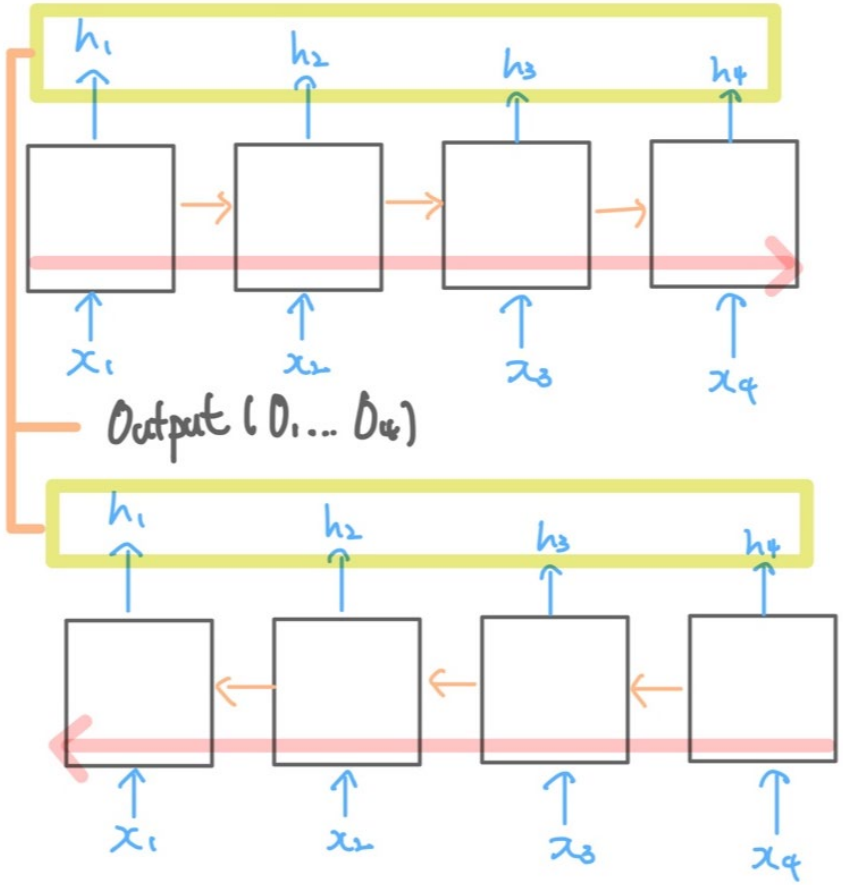
RNN(순환 신경망)?



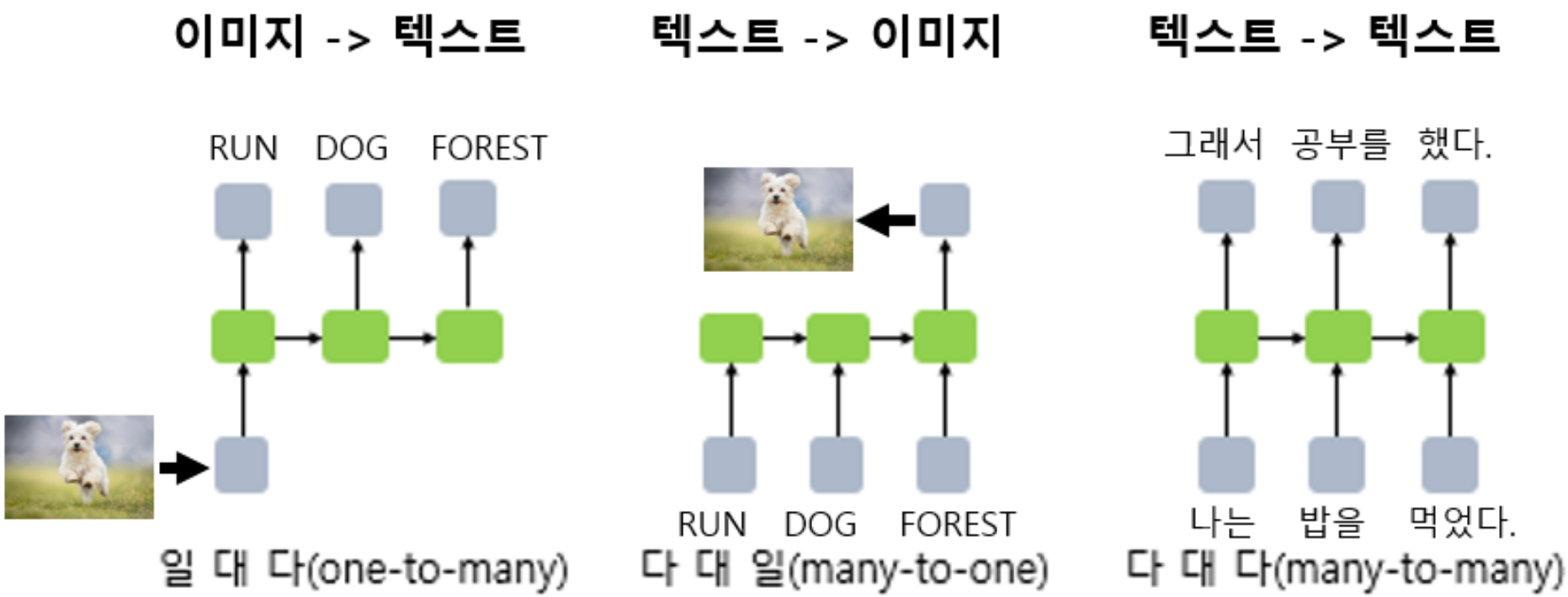
한방향 아닌

양방향도 가능한가?

양방향 RNN

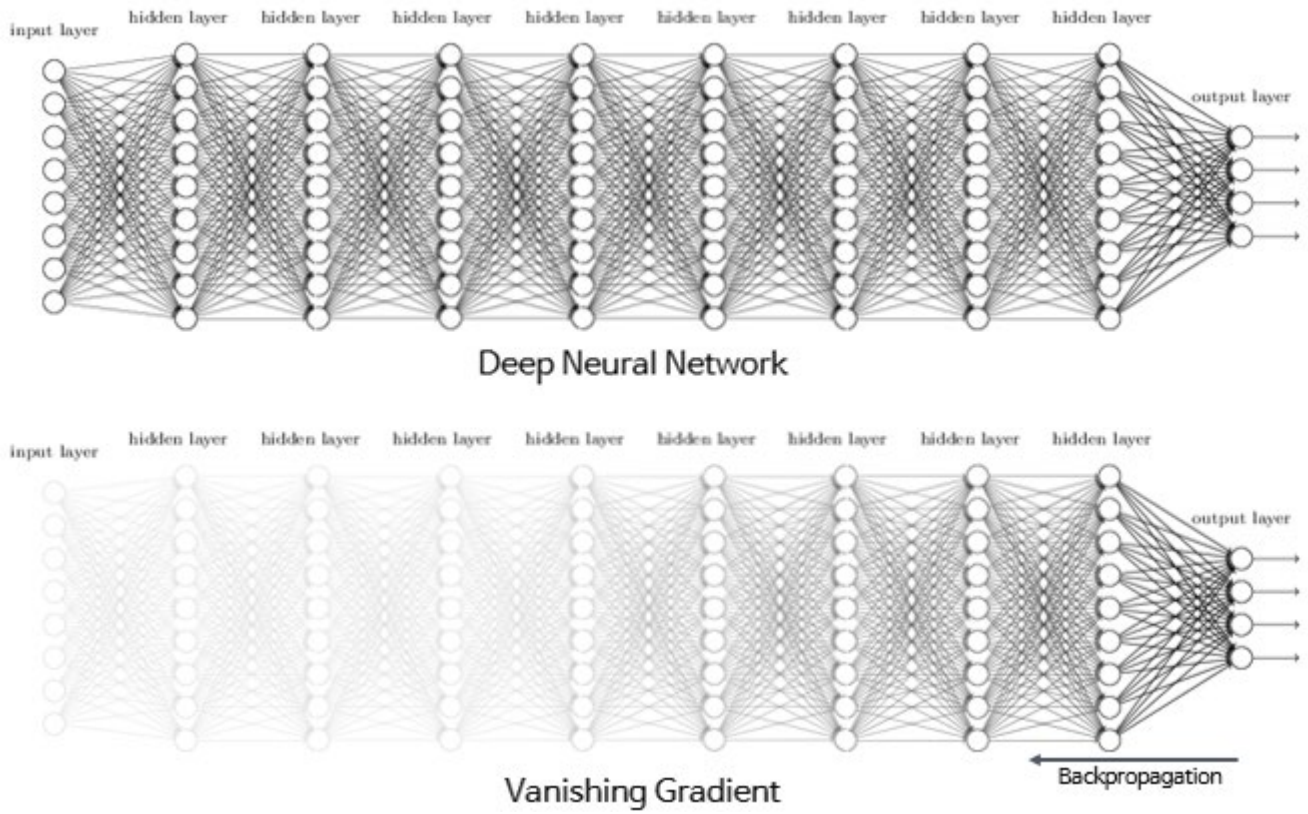


입력/출력층에 따른 종류

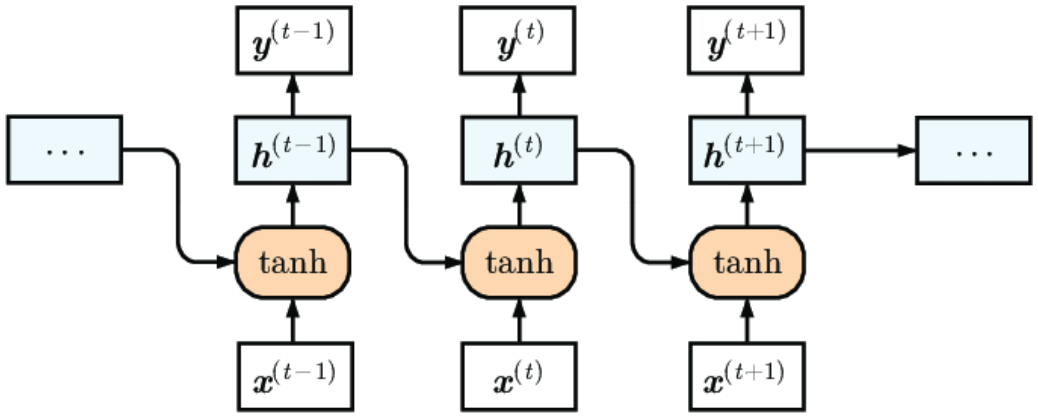

이미지 -> 텍스트
RUN DOG FOREST
일 대 다(one-to-many)
텍스트 -> 이미지
RUN DOG FOREST
다 대 일(many-to-one)
텍스트 -> 텍스트
그래서 공부를 했다.
나는 밥을 먹었다.
다 대 다(many-to-many)

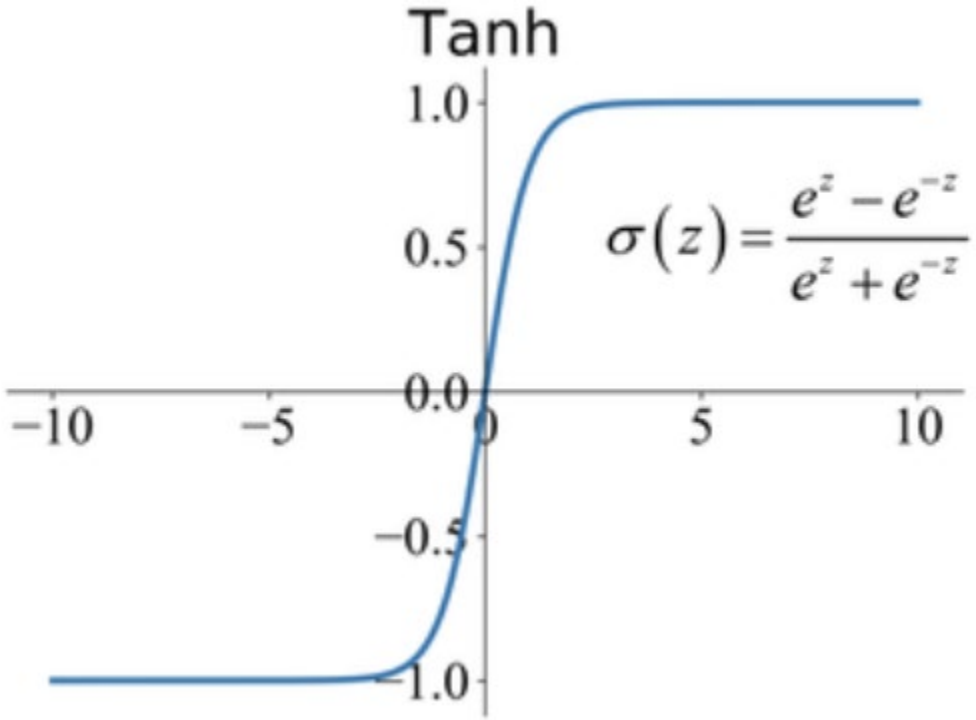
RNN에서도 약점이 있다?

- 일반적인 DNN에서도 문제 이슈

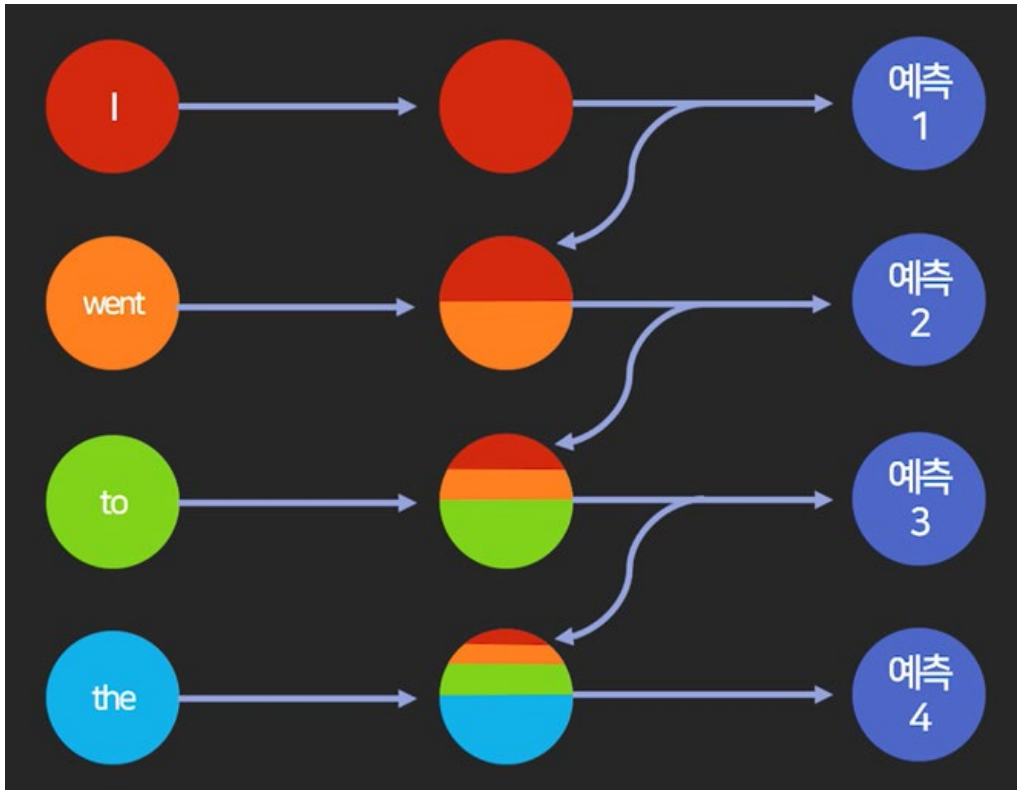


RNN 구조 상세

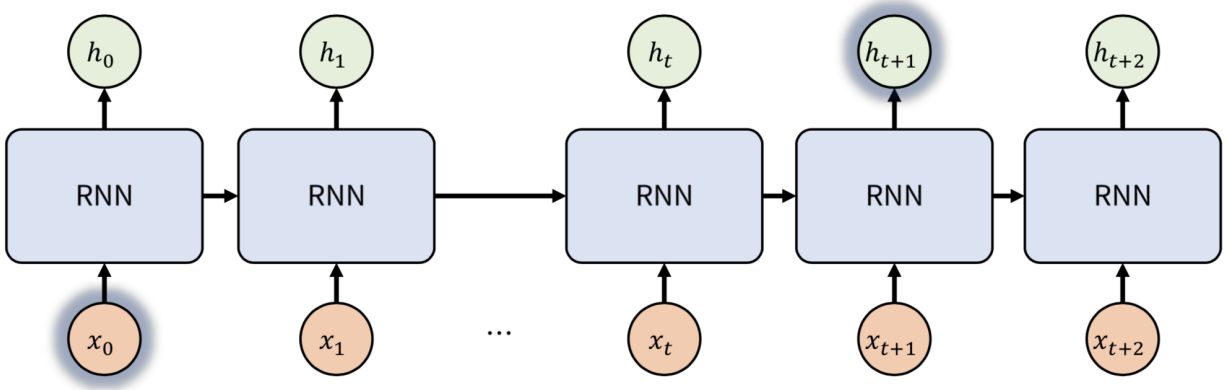

$y^{(t-1)}$
$y^{(t)}$
$y^{(t+1)}$
...
$h^{(t-1)}$
$h^{(t)}$
$h^{(t+1)}$
...
tanh
tanh
tanh
$x^{(t-1)}$
$x^{(t)}$
$x^{(t+1)}$


Tanh
1.0
0.5
0.0
−0.5
−1.0
−10
−5
0
5
10
$\sigma(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$

기울기 소실 문제 (RNN)

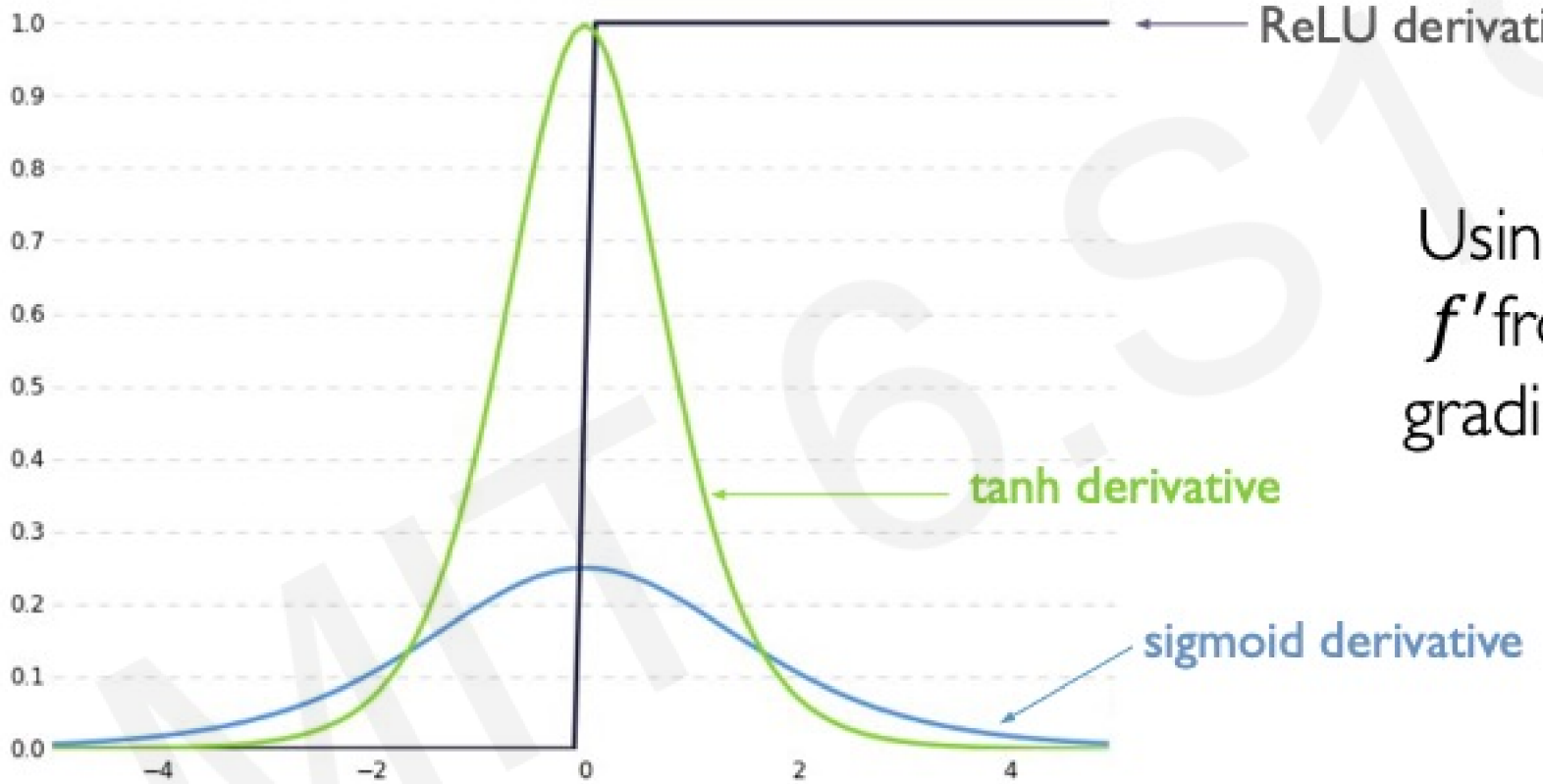


기울기 소실 문제



어떤 입력의 정보가 사용되는 시점이 차이가 많이 날 경우, 학습 능력이 저하된다.

Trick #1: Activation Functions



Using ReLU prevents f' from shrinking the gradients when $x > 0$

Trick #2: Parameter Initialization

Initialize **weights** to identity matrix

Initialize **biases** to zero

$$I_n = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

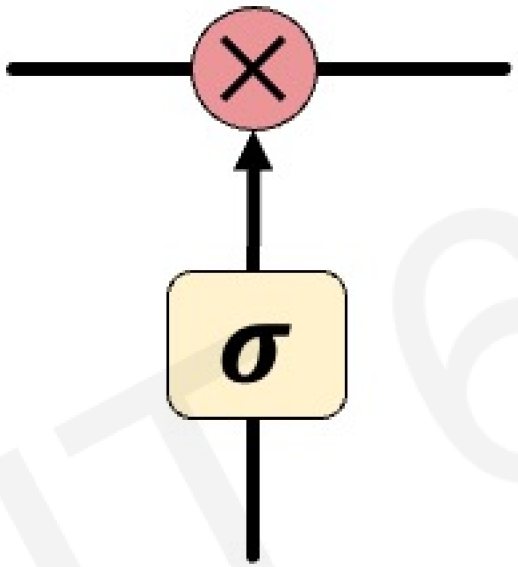
This helps prevent the weights from shrinking to zero.

Trick #3: Gated Cells

Idea: use **gates** to selectively **add** or **remove** information within **each recurrent unit with**

Pointwise multiplication

Sigmoid neural net layer



gated cell

LSTM, GRU, etc.

Gates optionally let information through the cell

Long Short Term Memory (LSTMs) networks rely on a gated cell to track information throughout many time steps.

RNN 개선 방법

단어들을 오랫동안

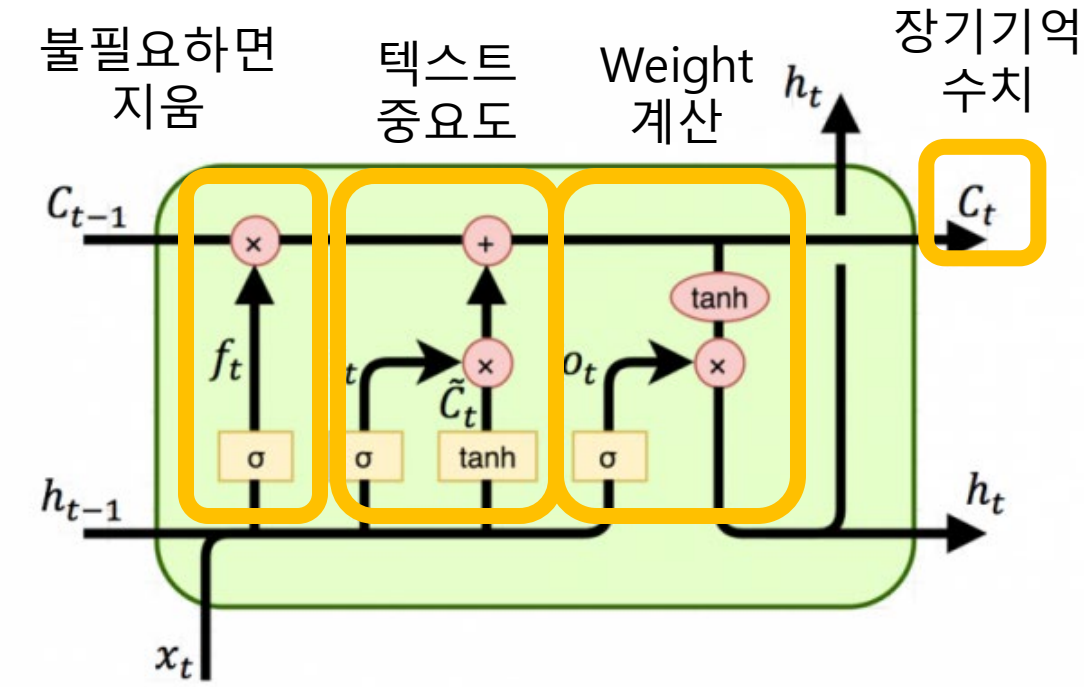
기억하면 좋을텐데

LSTM

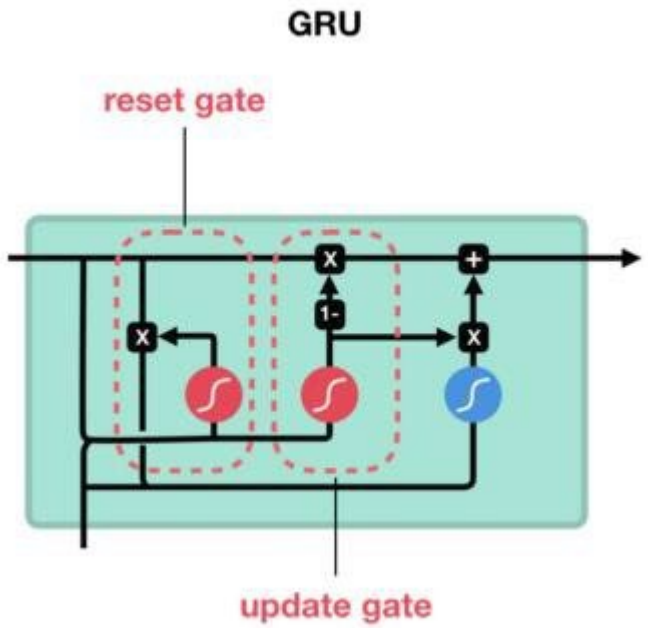
GRU

RNN 개선판

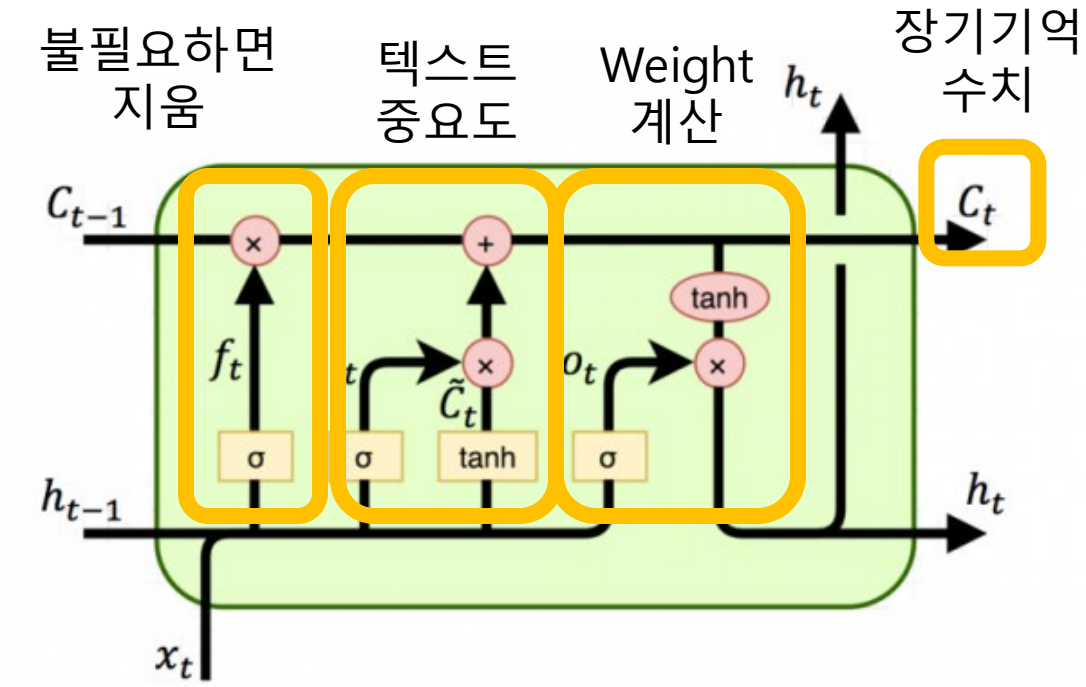
LSTM (처리할게 많음)



GRU (LSTM 보단 간단함, 성능 무난)



LSTM



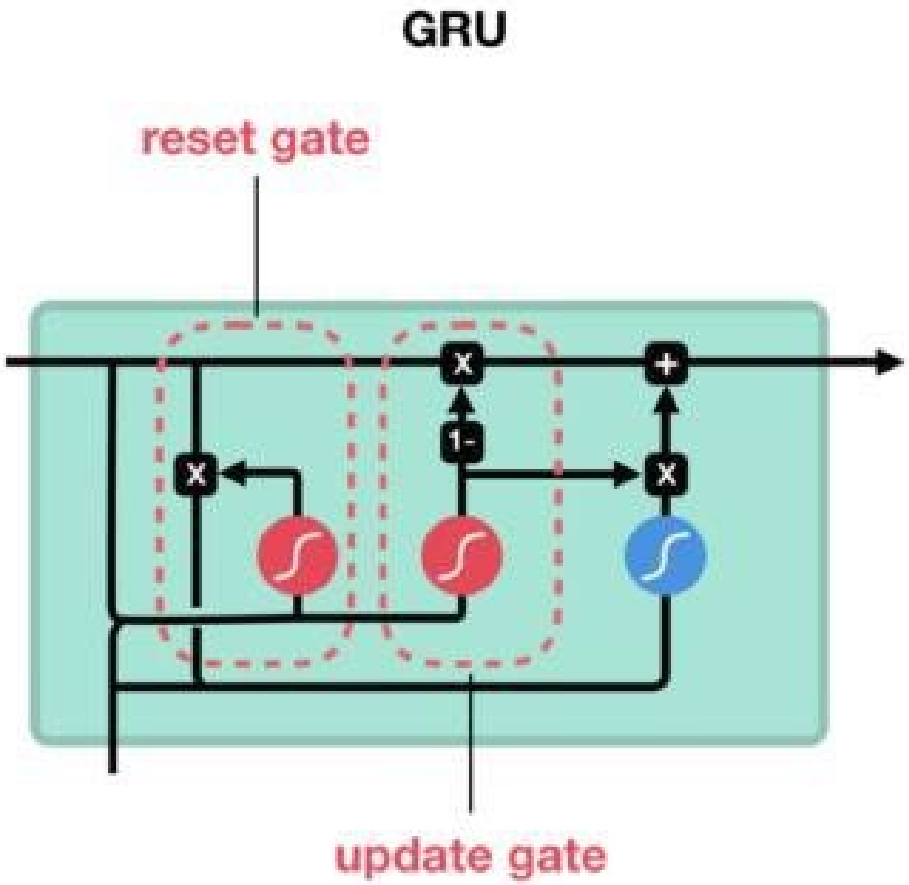
1.Weight 계산 (입력 게이트)

2.불필요하면 지움(삭제 게이트)

3.장기 기억 수치(텍스트 중요도)

4.입력 게이트 → 출력 게이트

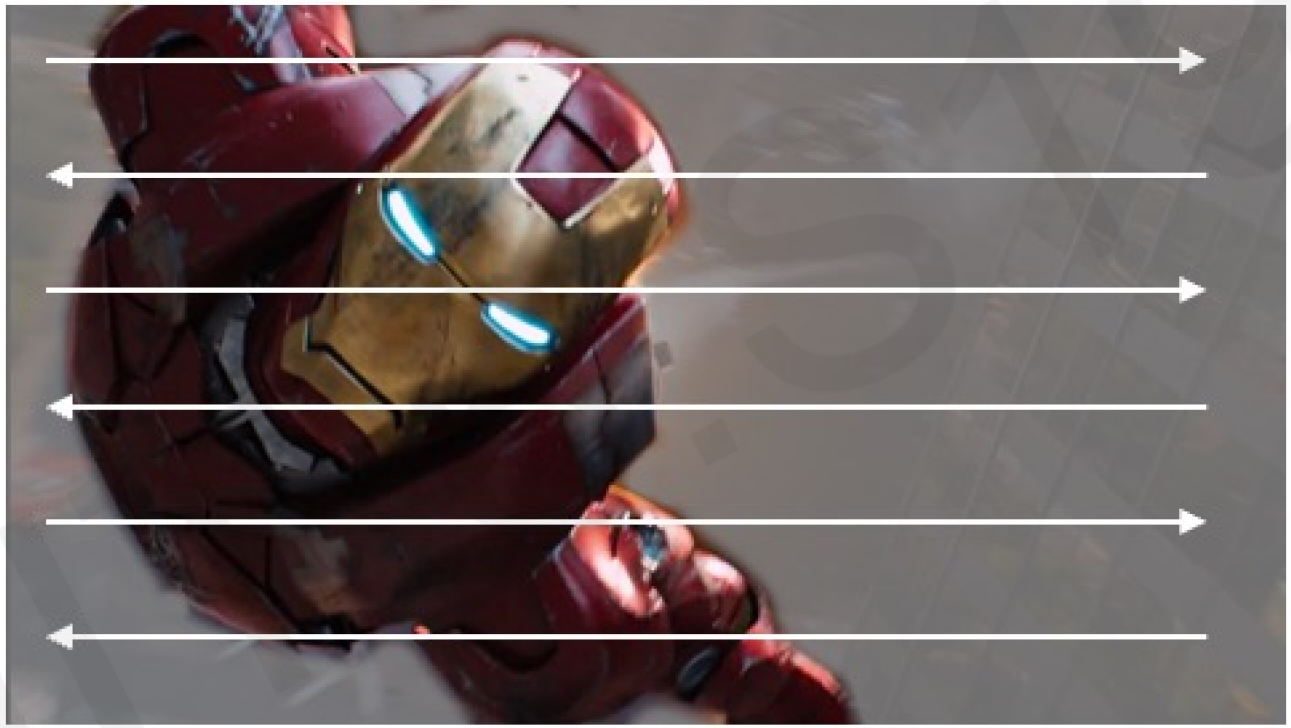
GRU



- Reset gate
  - 이전과 현재 얼마만큼?
- Update gate
  - 이전과 현재 가중치곱

Intuition Behind Self-Attention

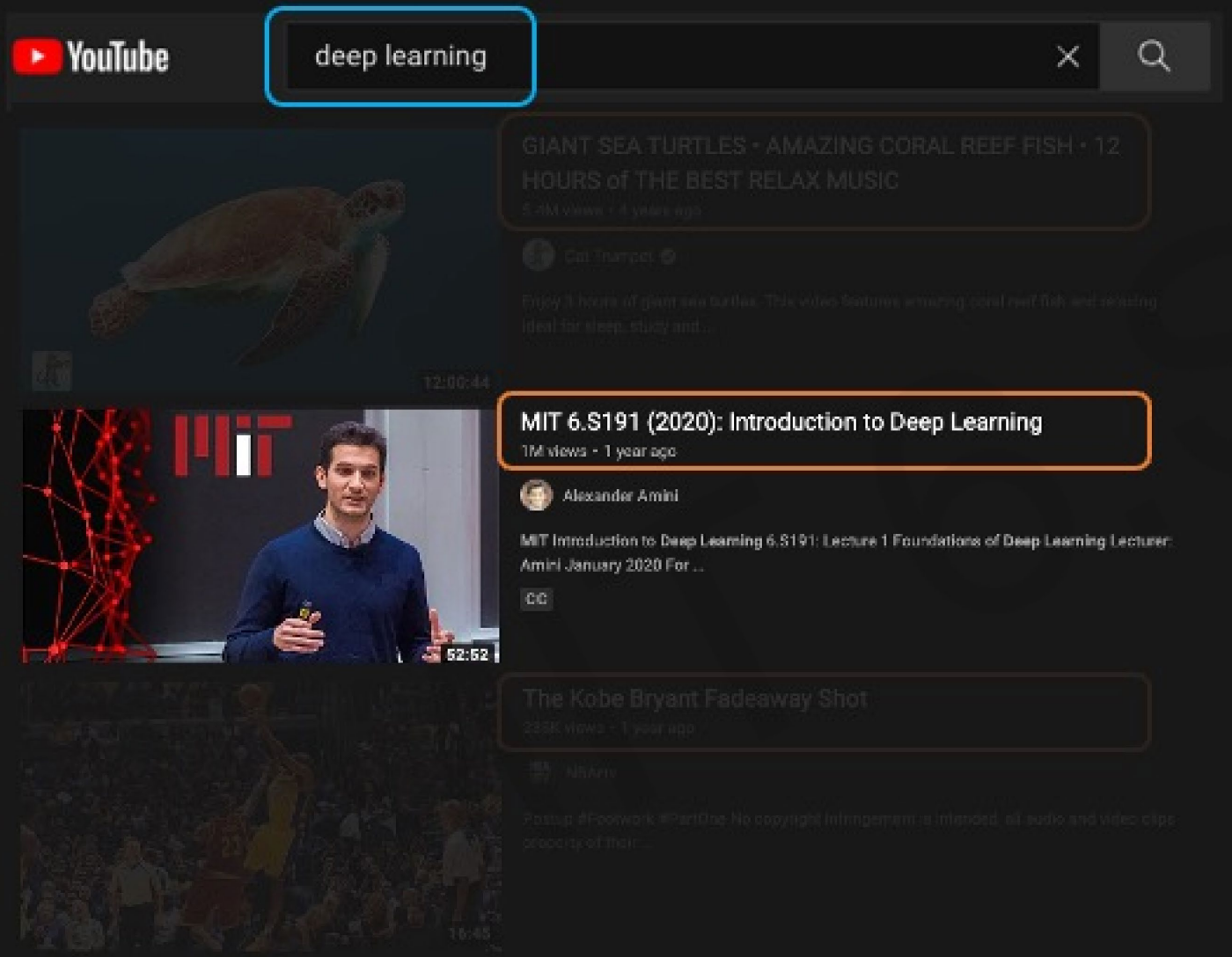
Attending to the most important parts of an input.



1. Identify which parts to attend to
2. Extract the features with high attention

Similar to a search problem!

Understanding Attention with Search



Query (Q)

Key (K_1)

Key (K_2)

Key (K_3)

How similar is the key to the query?

1. **Compute attention mask:** how similar is each key to the desired query?

Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

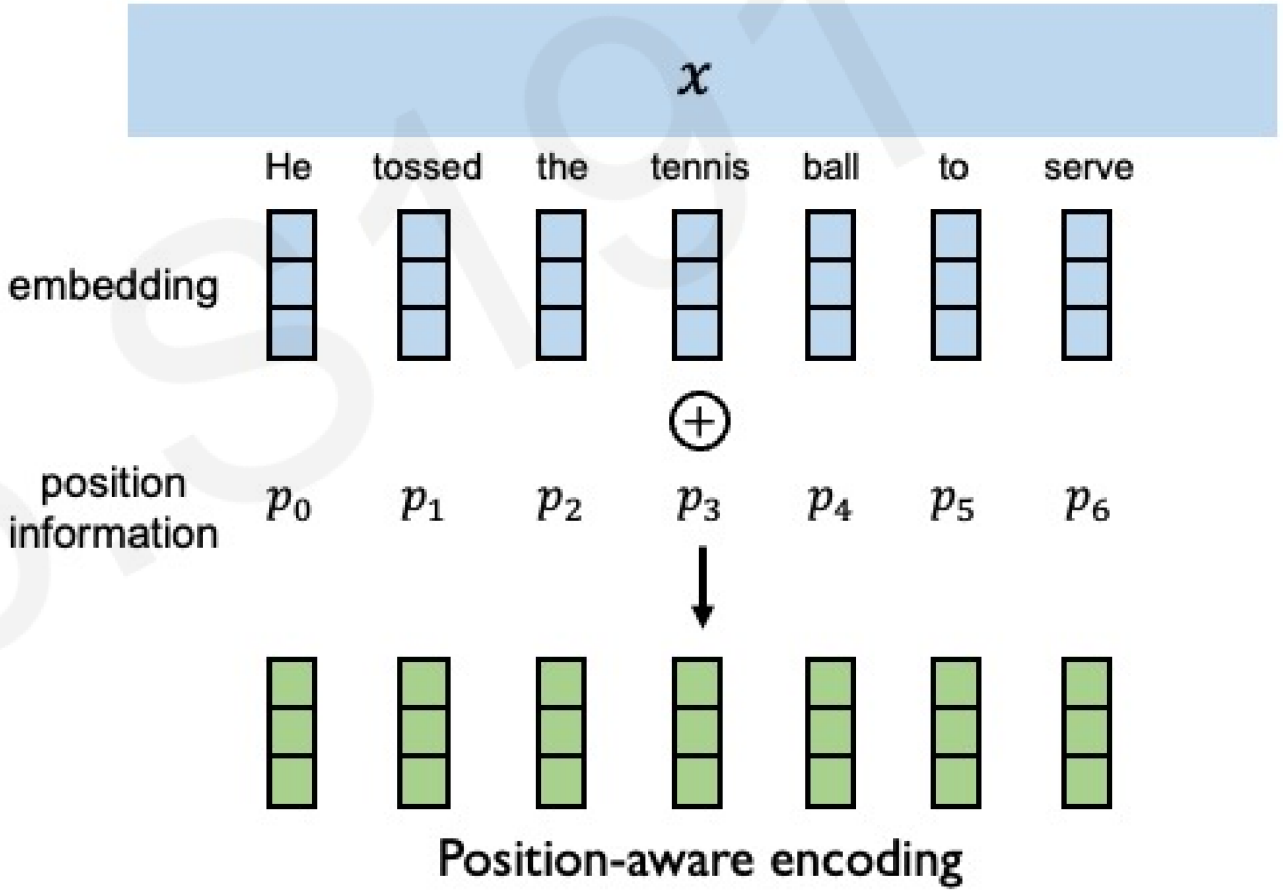
1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract **features with high attention**

Data is fed in all at once! Need to encode position information to understand order.

Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract **features with high attention**

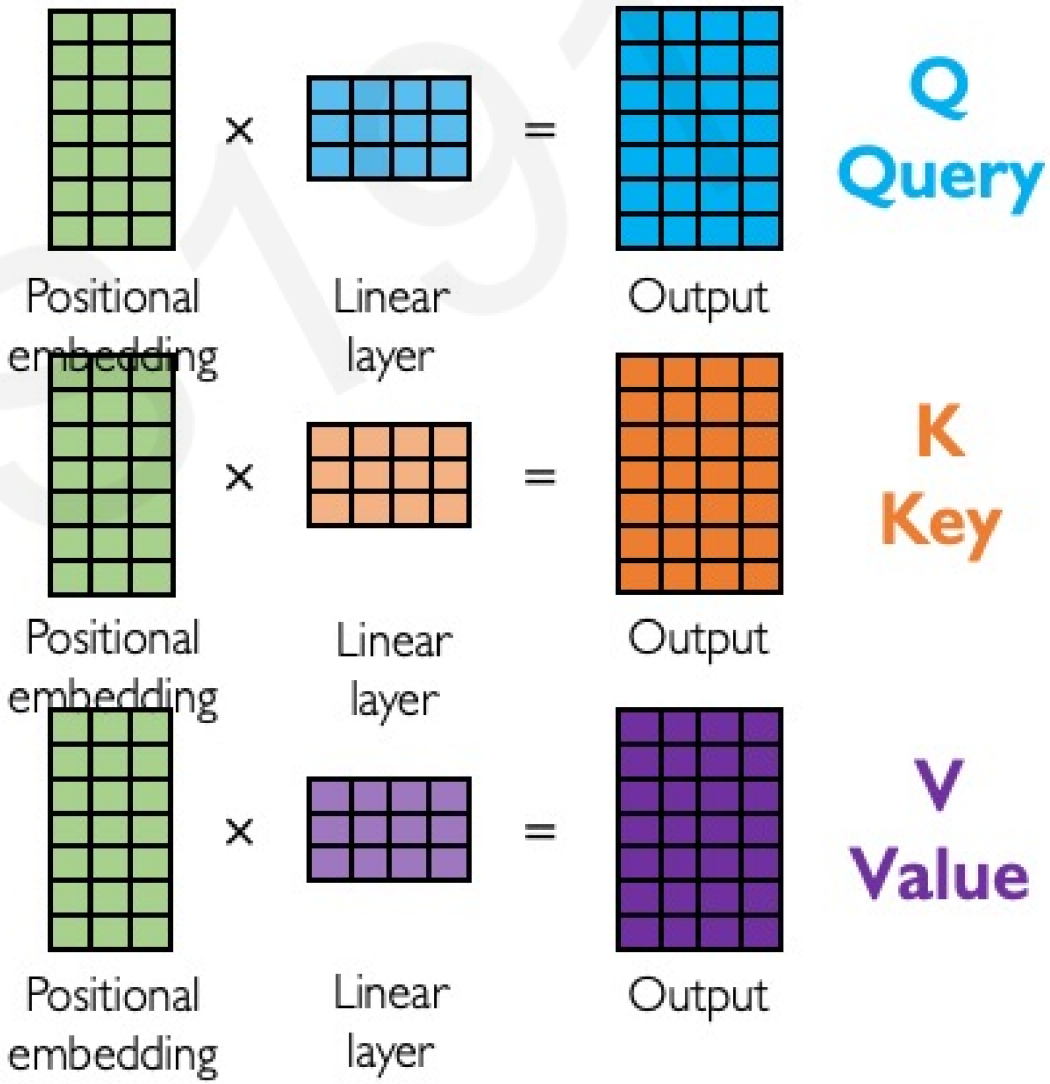


Data is fed in all at once! Need to encode position information to understand order.

Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute attention weighting
4. Extract features with high attention



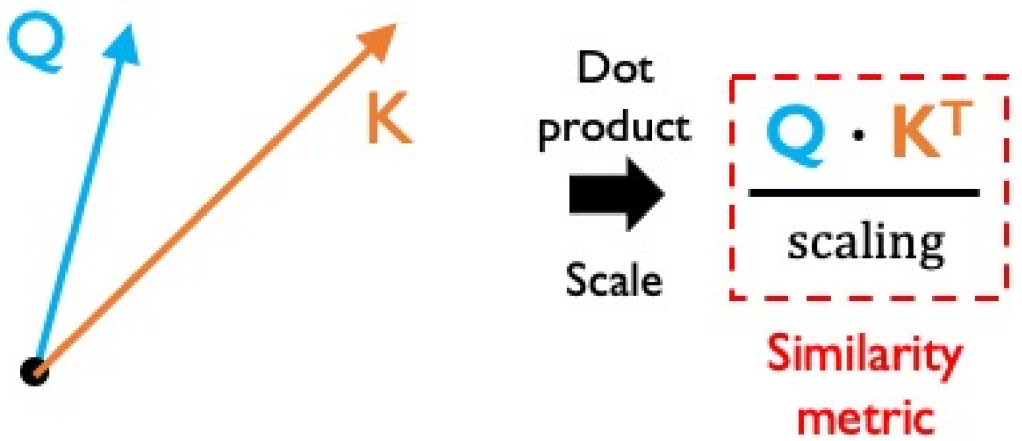
Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract features with high attention

Attention score: compute pairwise similarity between each query and key

How to compute similarity between two sets of features?



Also known as the "cosine similarity"

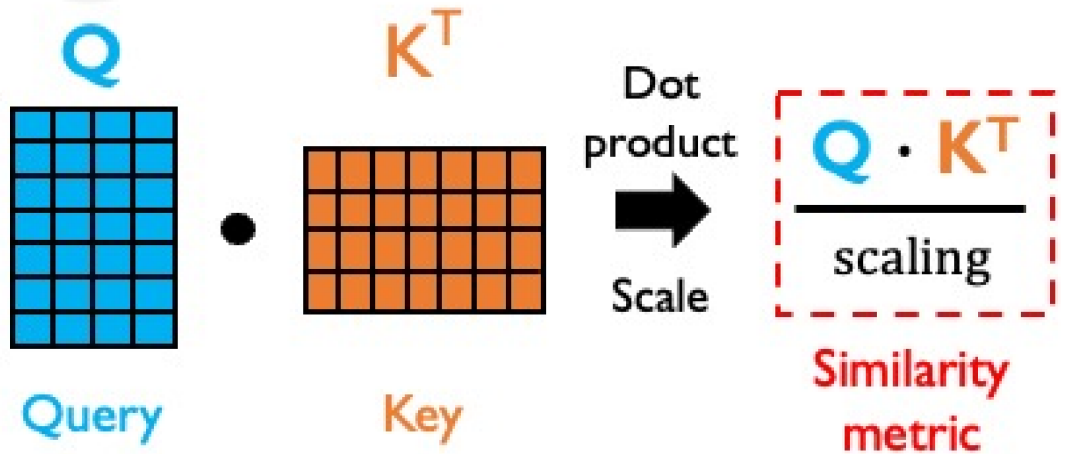
Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract features with high attention

Attention score: compute pairwise similarity between each query and key

How to compute similarity between two sets of features?



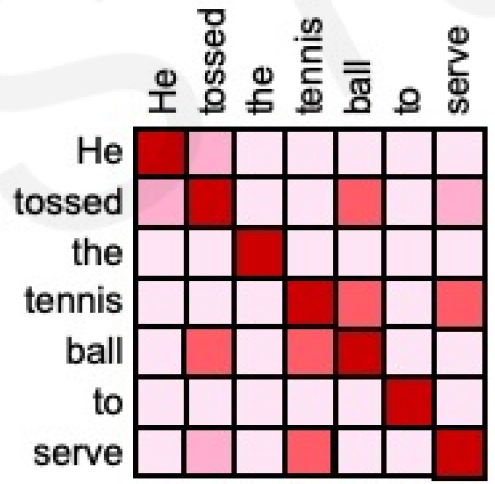
Also known as the "cosine similarity"

Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract features with high attention

Attention weighting: where to attend to! How similar is the key to the query?



$$softmax\left(\frac{Q \cdot K^T}{scaling}\right)$$

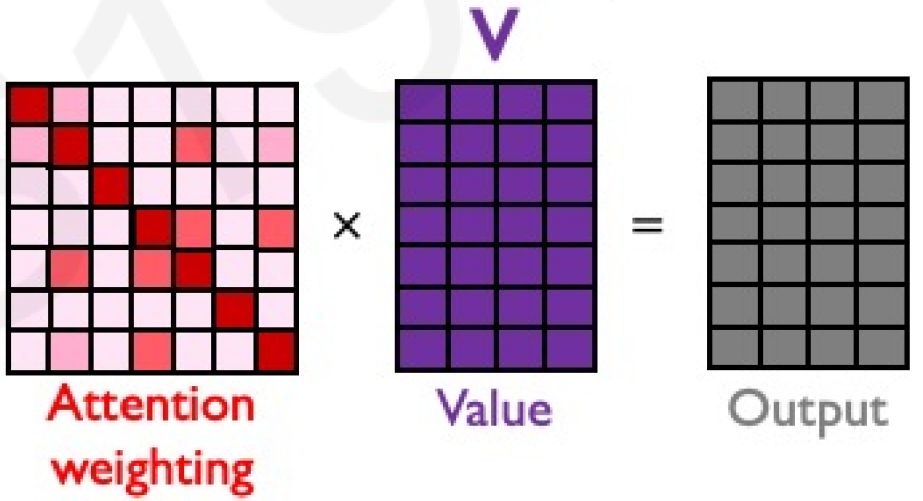
Attention weighting

Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract **features with high attention**

Last step: self-attend to extract features



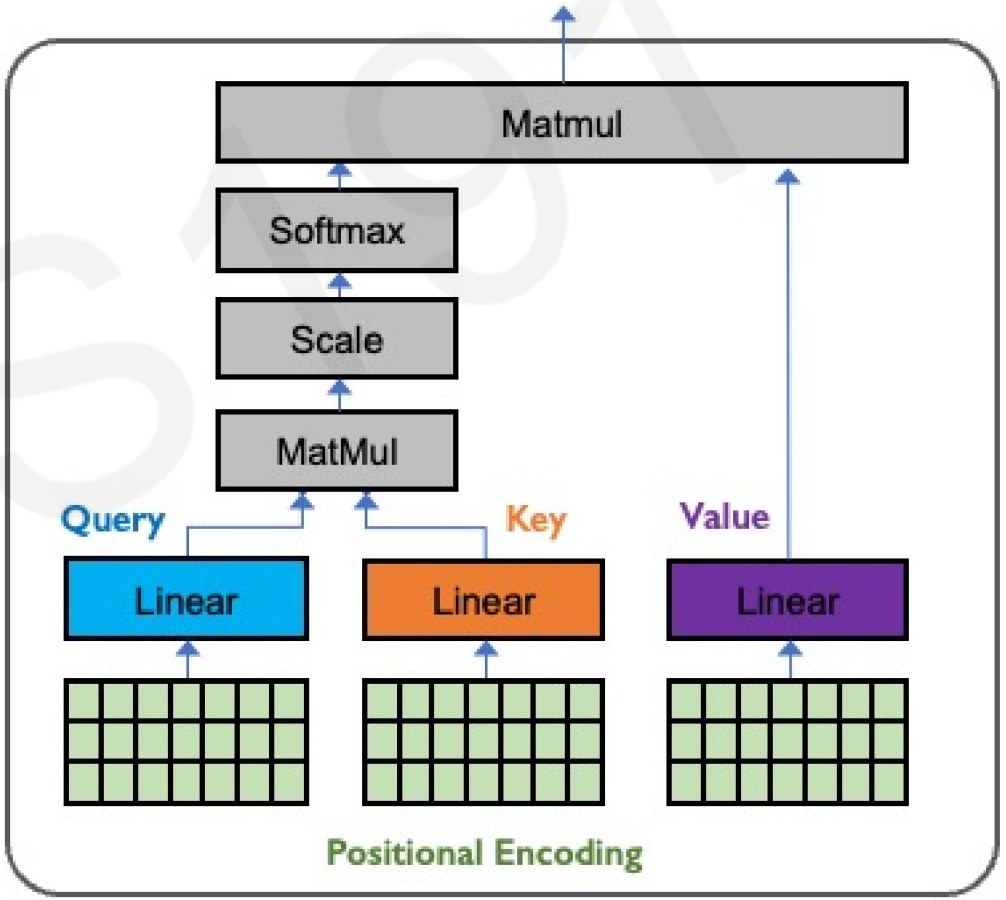
$$softmax\left(\frac{Q \cdot K^T}{scaling}\right) \cdot V = A(Q, K, V)$$

Learning Self-Attention with Neural Networks

Goal: identify and attend to most important features in input.

1. Encode **position** information
2. Extract **query, key, value** for search
3. Compute **attention weighting**
4. Extract **features with high attention**

These operations form a self-attention head that can plug into a larger network. Each head attends to a different part of input.



$$softmax\left(\frac{Q \cdot K^T}{scaling}\right) \cdot V$$

Applying Multiple Self-Attention Heads

Attention weighting

×

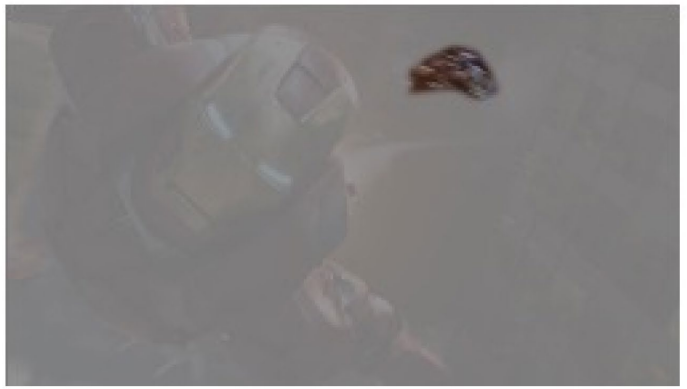
Value

=

Output

Output of attention head 1

Output of attention head 2



Output of attention head 3