1. Science (Backpropagation, Regularization, SGD, GD, Vanishing and Exploding gradient problem, Little bit more on regularization)
2. Alchemy-like knowledge (hidden units, how many layers, what kind of layers, why? What kind of architecture should we use for what kind of problem? Not having a general theory, but figuring out by trial and error in the community.
)

Many kinds of Layers

↳ Linear Layer + Non linear activation functions

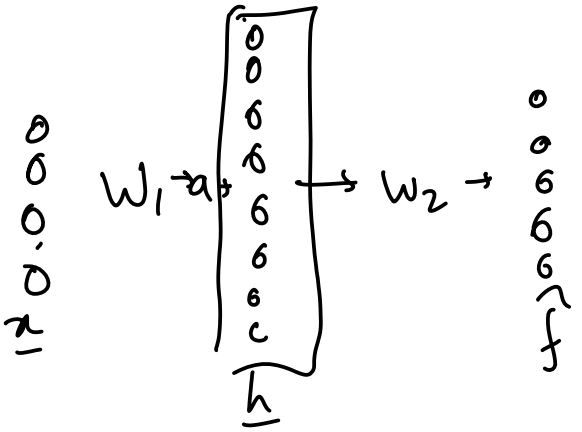
MLP: Multi-Layer Perceptron

Universal Approximation Theorem

2-layer MLP

$$\hat{f}(\underline{x}) = W_2\left(a\left(W_1\underline{x} + \underline{b}_1\right)\right) + \underline{b}_2$$

W_1: number of hidden units



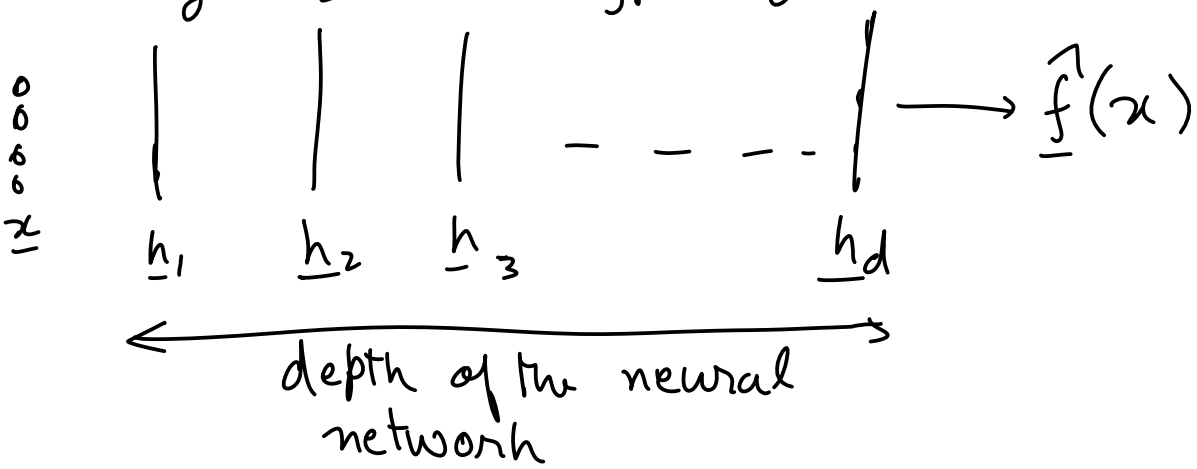
hidden units ?

100, 1000, 10000, ∞

UAT: You can approximate any continuous function using an infinitely-wide 2-layer MLP with activation functions like, ReLU, sigmoid, tanh, ----

---

UAT': ∞-layer MLP with sufficiently wide hidden unit



Fourier series, Taylor series } with ∞ expansion, you can approximate any continuous function

---

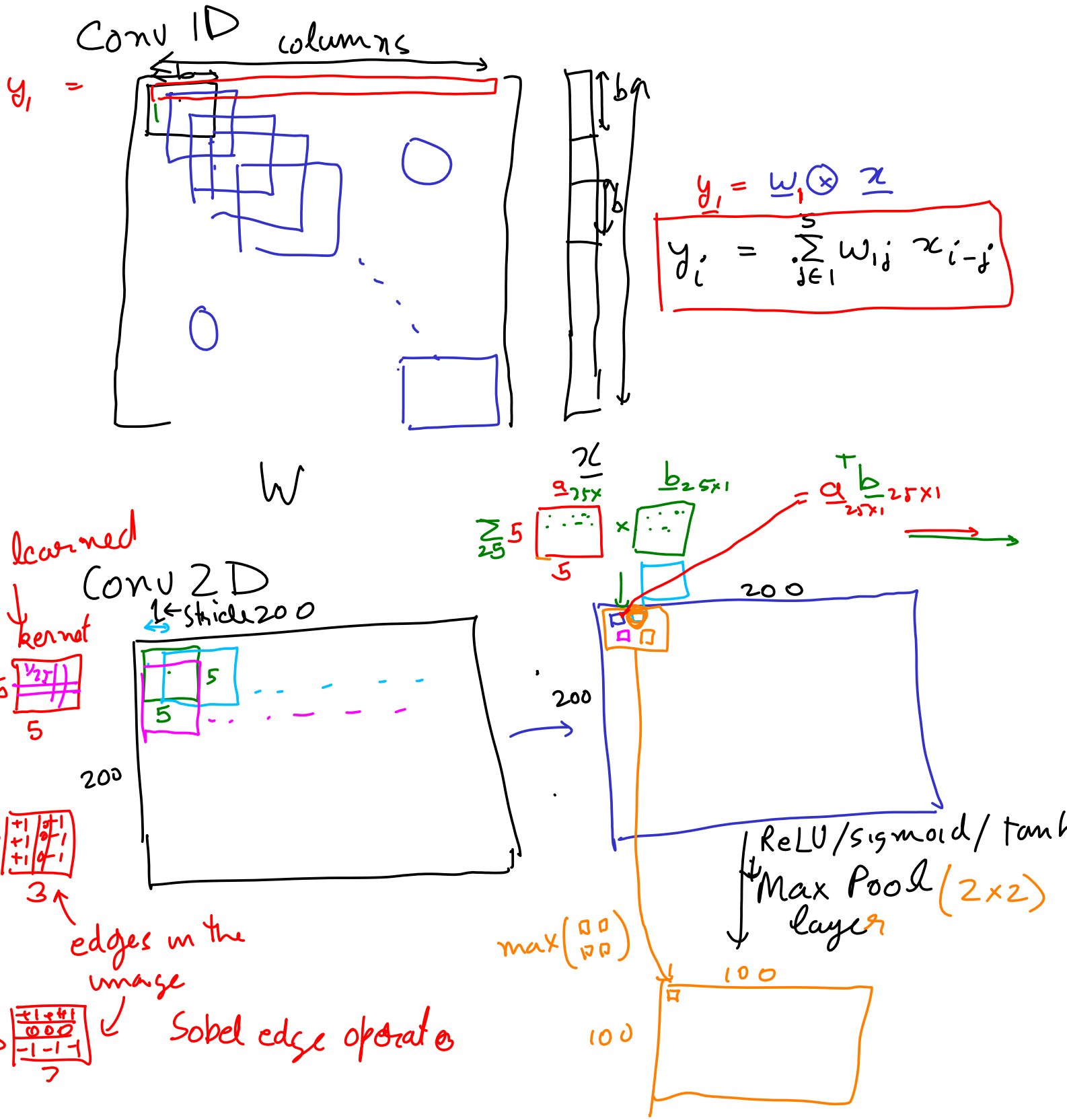
① MLP layers
② Convolutional architecture, ③ Transformer architecture

↓

Mostly applied to images, data where there is repetition of patterns

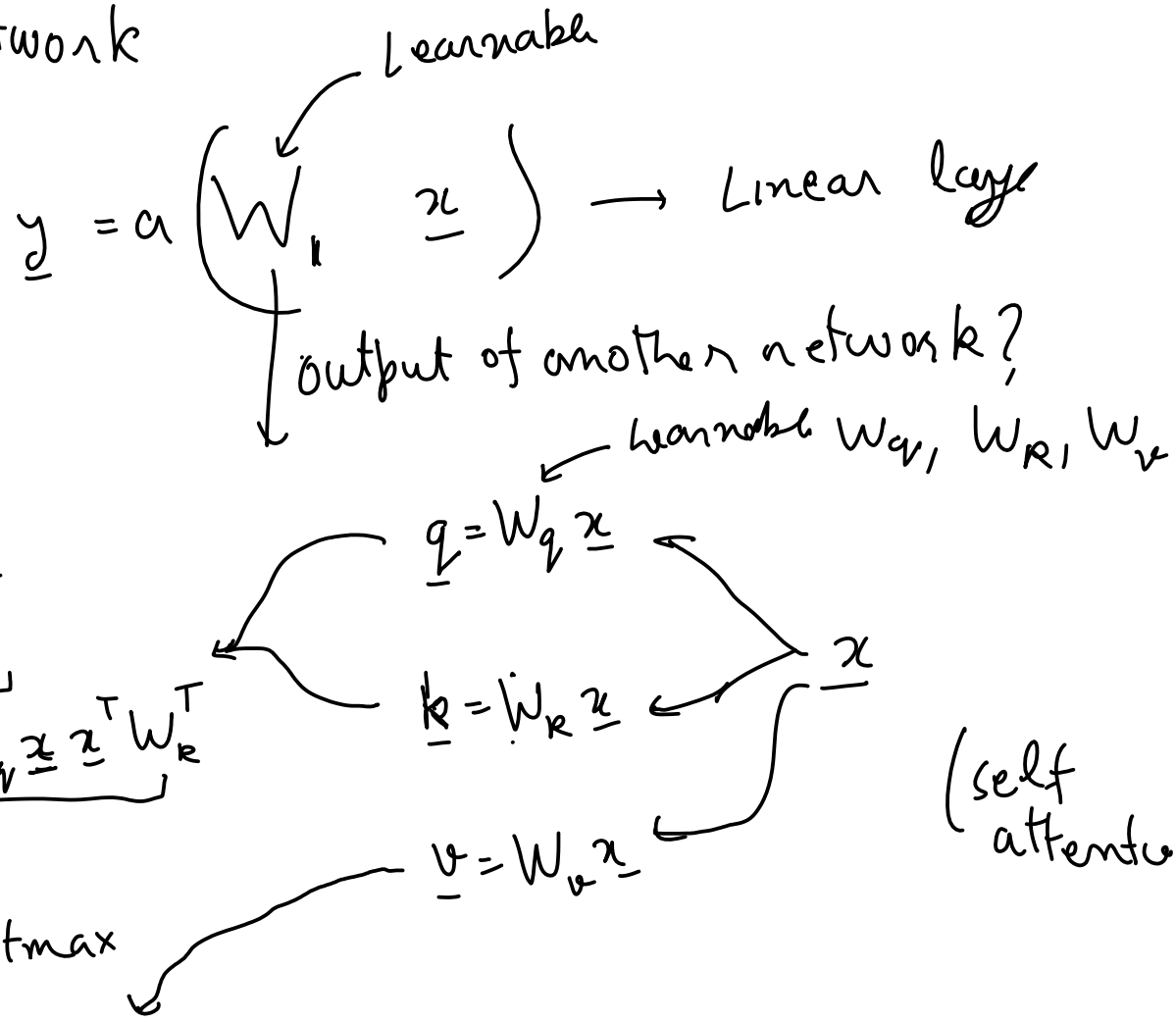
Conv 1D is a particular kind of Linear layer

Conv2D


Conv 1D
columns
y_1 =
b
b
W
$y_1 = w_1 \otimes x$
$y_i = \sum_{j=1}^{5} w_{ij} x_{i-j}$
x
$a_{25\times}$
$b_{25\times1}$
$\sum_{25} 5$
5
$= a^T_{25\times1} b_{25\times1}$
200
learned
Conv 2D
kernel
stride 200
5
5
5
200
200
3
3
edges in the image
Sobel edge operator
ReLU/sigmoid/tanh
Max Pool (2×2) layer
max
100
100
feature extractor
Pooling
(4) Pooling
Max Pool 2D
Avg Pool 2D

(3) Transformer / Self-attention / cross-attention

Hypernetwork



$$\text{softmax}(\underline{x})_i = \frac{\exp(x_i)}{\sum \exp(\underline{x})}$$

$$\underline{y} = \text{softmax}\left(\underline{q}\,\underline{k}^T\right)\underline{v}$$

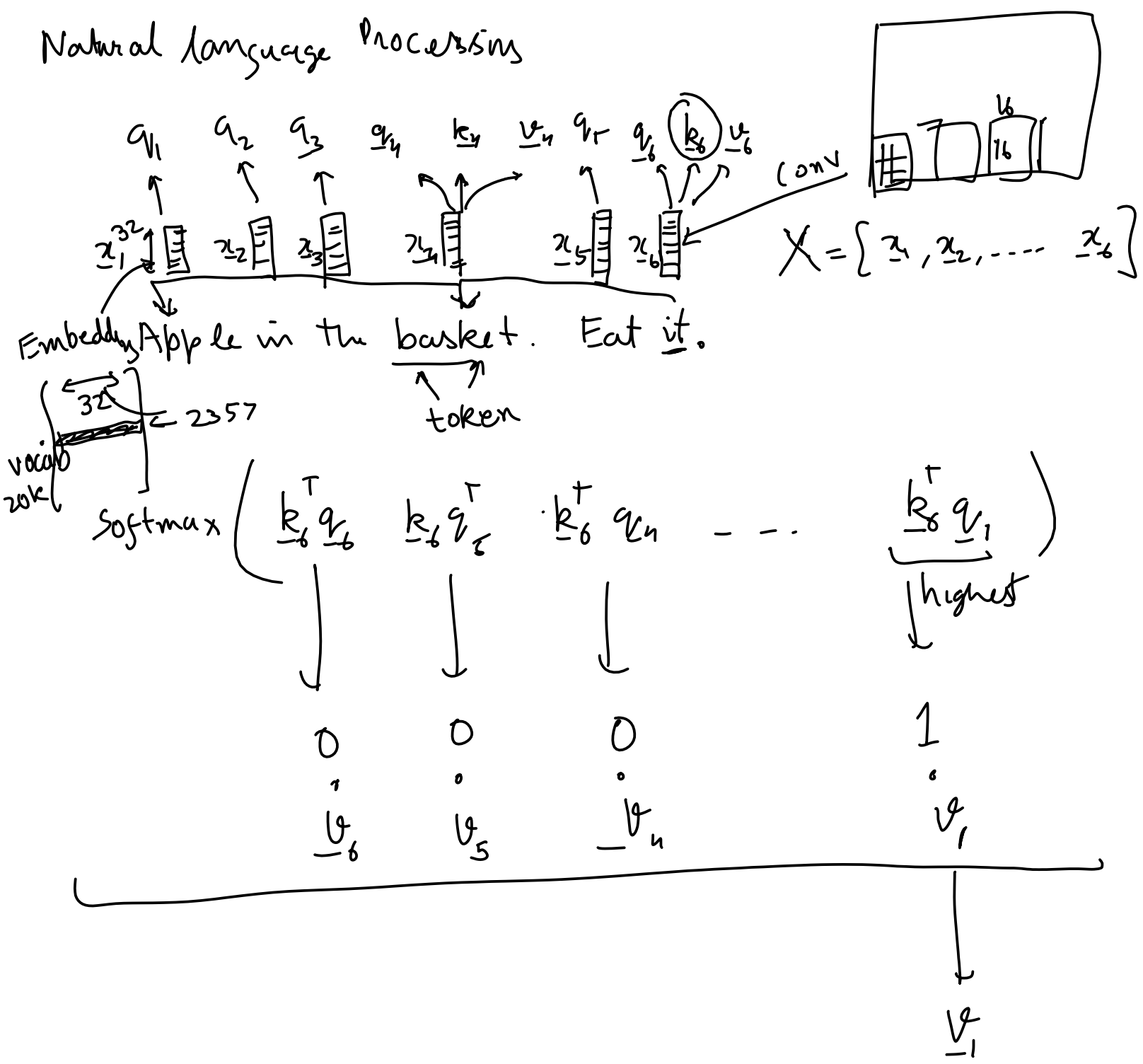
$$= \underbrace{\text{softmax}\left(q\,k^T\right) W_v}_{W}\,\underline{x}$$

Self-attention layer

$$\underline{y} = \text{softmax}\left(QK^T\right)\underbrace{W_v X}_{V}$$

$$Q = W_q X$$
$$K = W_k X$$
$$V = W_v X$$

Natural language Processing



Self attn 1
X → Self attn 2 } Multi head attn

① MLP
② Conv
③ Pooling
④ Transformer / self attn

cross attn layer

when q, k come from diff data than v

(5) Embedding Layer (indexing into a big matrix)