2025 한국원자력학회 추계학술발표회 원자력 인공지능강습회

생성형 인공지능 기초 1: RNN부터 Transformer까지

전준구

Assistant Professor, NINE Lab.,

Division of Advanced Nuclear Engineering,

POSTECH

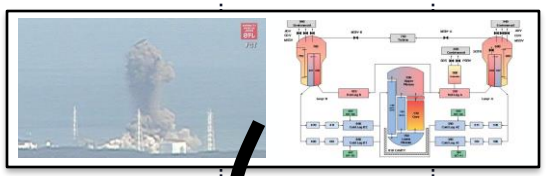
My Background

[01] J. Jeon et al., *Ann. Nucl. Energy*, 2018.
[02] J. Jeon et al., *Nucl. Eng. Technol.*, 2019.
[03] J. Jeon et al., *Energies*, 2020.
[04] J. Jeon et al., *Int. J. Heat Mass. Transf.*, 2021
[05] J. Jeon et al., *Nucl. Eng. Technol.*, 2019.
[06] J. Jeon et al., *Nucl. Eng. Technol.*, 2021.
[07] J. Jeon et al., *Int. J. Energy Res.*, 2022.
[08] J. Jeon et al., *Int. J. Heat Mass. Transf.*, 2024
[09] J. Jeon et al., *Phys. Fluids*, 2025.
[10] S. Khanal et al., *Nucl. Eng. Technol.*, 2025.
[11] S. Jo et al., *under review.*
[12] J. Shin et al., *under review.*
[13] S. Baral et al., *under review.*
[14] S. Khanal et al., *in preparation.*
[15] M. Lee et al., *in preparation.*
[16] S. Yu et al., *in preparation.*

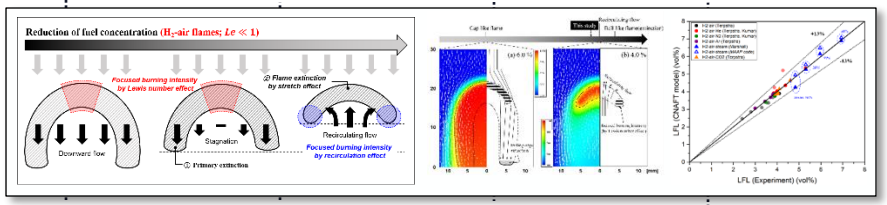
2016 — 2018 — 2020 — 2022 — 2024 — 2025

Master degree

- *Numerical analysis of* ***severe accident*** *[1,2].*



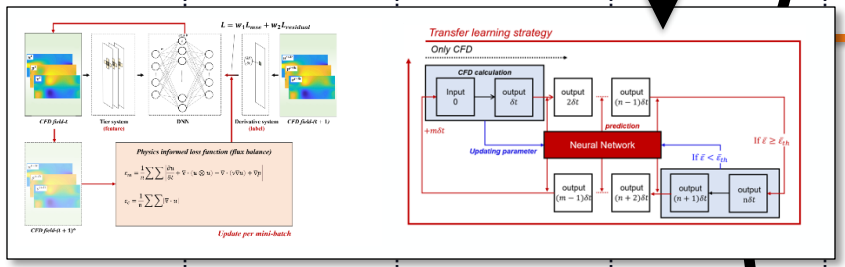
"We need hydrogen flammability model !"



- ***Hydrogen flame extinction mechanism*** *with CFD [3, 4].*
- ***Hydrogen flammability limit model*** *[5, 6].*
- ***Hydrogen combustion risk prediction code*** *(HYCOR) (flammability, flame acceleration, DDT evaluation).*

Ph.D. degree

"We need CFD acceleration using AI !"



- ***Finite volume method network (FVMN)*** *model [7].*
- ***Physics-informed transfer learning (RePIT)*** *strategy [8].*
- *RePIT open-source software optimizations [10, 13]*

"We can apply AI to SMR-Digital Twin !"

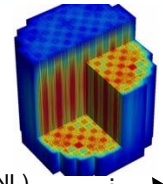
LLM applications

- *LoRA fine-tuning*

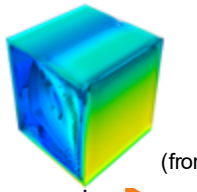
SMR-DT dependency: SciML vs CFD

	Low fidelity	High fidelity	Optimal fidelity	
CFD				
SciML	Stand-alone	Hybrid solver	Agent	
	① Real-time simulator	② CFD acceleration	③ Optimal control	

- ***DT level 1: Real-time simulator (AI surrogate model) [11, 12, 14, 15]***
- ***DT level 2: CFD acceleration (SMR applications) [13]***
- ***DT level 3: optimal control strategy (reinforcement learning) [9]***

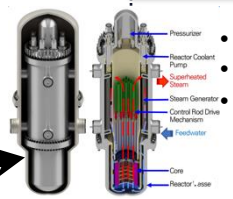


(from ORNL)



(from NINE Lab.)

"SMR innovation and licensing"



- *SMR regulatory technology*
- ***Hydrogen risk assessment for i-SMR [16]***
- *i-SMR MELCOR modeling*

"We need SMR hydrogen regulatory technology !"

My Background

[01] J. Jeon et al., *Ann. Nucl. Energy*, 2018.
[02] J. Jeon et al., *Nucl. Eng. Technol.*, 2019.
[03] J. Jeon et al., *Energies*, 2020.
[04] J. Jeon et al., *Int. J. Heat Mass. Transf.*, 2021
[05] J. Jeon et al., *Nucl. Eng. Technol.*, 2019.
[06] J. Jeon et al., *Nucl. Eng. Technol.*, 2021.
[07] J. Jeon et al., *Int. J. Energy Res.*, 2022.
[08] J. Jeon et al., *Int. J. Heat Mass. Transf.*, 2024
[09] J. Jeon et al., *Phys. Fluids*, 2025.
[10] S. Khanal et al., *Nucl. Eng. Technol.*, 2025.
[11] S. Jo et al., *under review.*
[12] J. Shin et al., *under review.*
[13] S. Baral et al., *under review.*
[14] S. Khanal et al., *in preparation.*
[15] M. Lee et al., *in preparation.*
[16] S. [illegible] et al., *in preparation.*

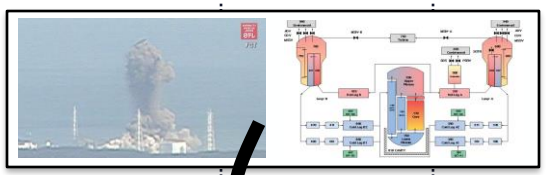
LLM applications

- *LoRA fine-tuning*

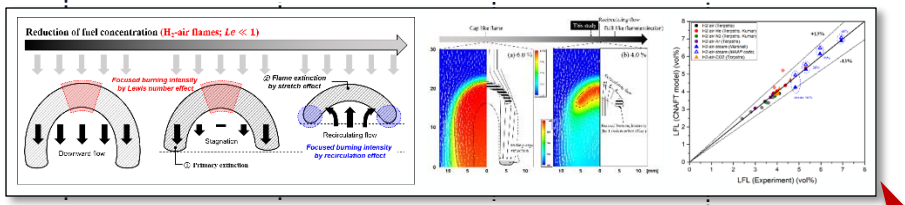
2016 — **2018** — **2020** — **2022** — **2024** — **2025**

Master degree

- *Numerical analysis of **severe accident** [1,2].*



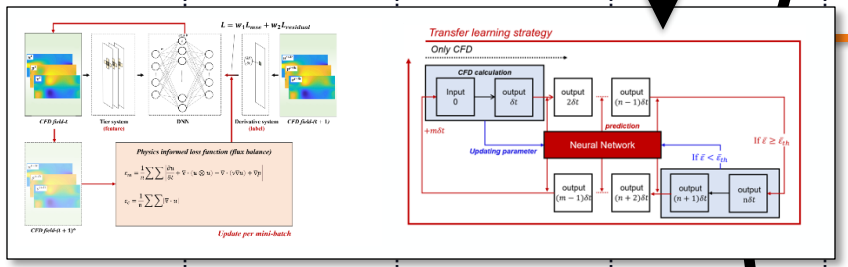
"We need hydrogen flammability model !"



- ***Hydrogen flame extinction mechanism** with CFD [3, 4].*
- ***Hydrogen flammability limit model** [5, 6].*
- ***Hydrogen combustion risk prediction code** (HYCOR) (flammability, flame acceleration, DDT evaluation).*

Ph.D. degree

"We need CFD acceleration using AI !"



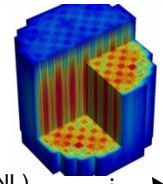
- ***Finite volume method network (FVMN)** model [7].*
- ***Physics-informed transfer learning (RePIT)** strategy [8].*
- *RePIT open-source software optimizations [10, 13]*

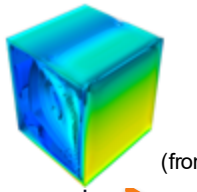
"We can apply AI to SMR-Digital Twin !"

SMR-DT dependency: SciML vs CFD

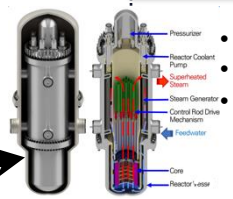
	Low fidelity	High fidelity	Optimal fidelity	
CFD				
SciML	Stand-alone	Hybrid solver	Agent	
	① Real-time simulator	② CFD acceleration	③ Optimal control	

- ***DT level 1: Real-time simulator (AI surrogate model) [11, 12, 14, 15]***
- ***DT level 2: CFD acceleration (SMR applications) [13]***
- ***DT level 3: optimal control strategy (reinforcement learning) [9]***


(from ORNL)


(from NINE Lab.)

"SMR innovation and licensing"

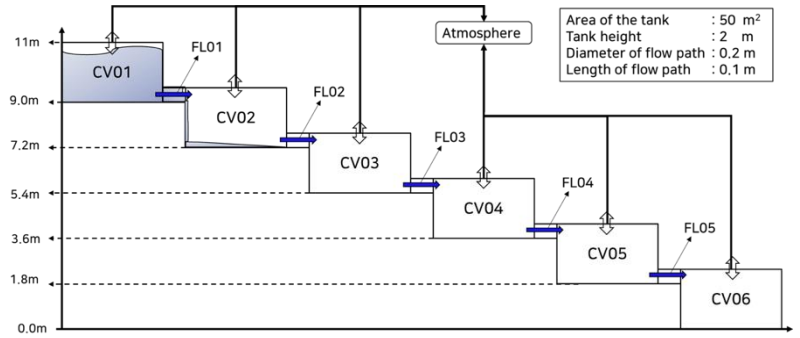


- *SMR regulatory technology*
- ***Hydrogen risk assessment for i-SMR [16]***
- *i-SMR MELCOR modeling*

"We need SMR hydrogen regulatory technology !"

The world has changed!

MELCOR



Prof. Sooyoung Lee Mr. Kiho Kim

탱크 개수, 온도, 압력 조건 → 입력문 3000개 생성

Scenario prompt

```
Scenario: Gravity-driven fluid problem.
*** MELCOR Assessment problems test***
CV01000   CV10     2        2     1
CV010A0   3
CV010A1   PVOL   1.01E+05
CV010A2   TATM   293.23
CV010A3   MLFR.4   1.0
CV010BA     0.0      0.0
CV010BB    30.0      100.0
```

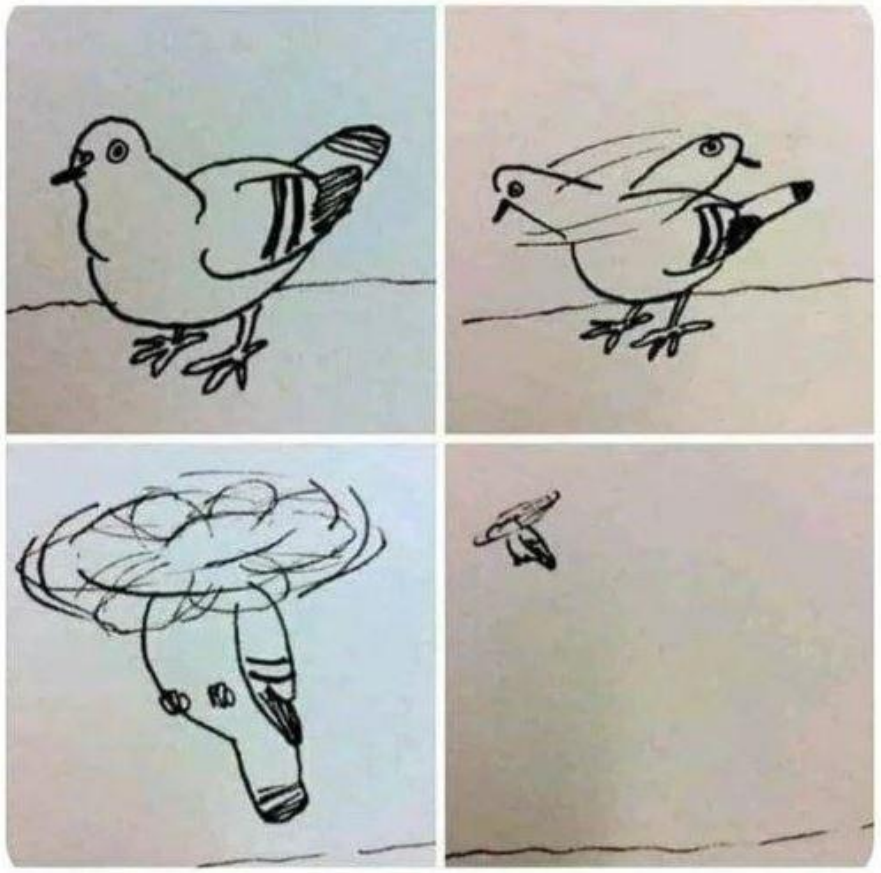
Instruction prompt

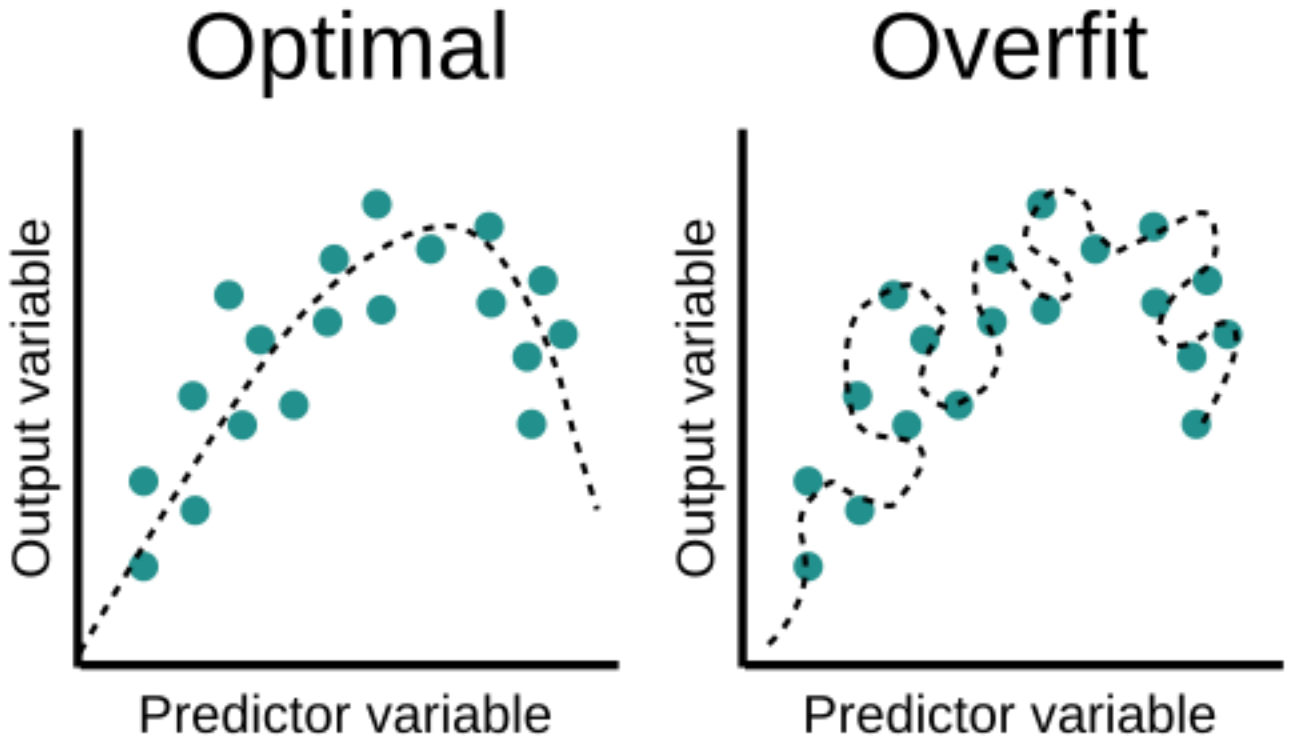
```
create MELCOR input with 7 tanks in series.
```

```
*            NAME    SWITCH    FLOW-FLAG    ICVTYP
CV00100    CV01       2          1          1
*           ITYPTH
CV001A0    3
CV001A1    TPOL    293.23
CV001A2    PVOL    1.01E+05
CV001A3    TATM    293.23
CV001A4    MLFR.4    1.0
CV001A5    VPOL     100.00
CV001BA      4.0      0.0
CV001BB      6.0      100.0
*
*            NAME    SWITCH    FLOW-FLAG    ICVTYP
CV00200    CV02       2          1          1
*           ITYPTH
CV002A0    3
CV002A2    PVOL    1.01E+05
CV002A3    TATM    293.23
CV002A4    MLFR.4    1.0
CV002BA      3.0      0.0
CV002BB      5.0      100.0
*
*            NAME    SWITCH    FLOW-FLAG    ICVTYP
CV00300    CV03       2          1          1
*           ITYPTH
CV003A0    3
CV003A2    PVOL    1.01E+05
CV003A3    TATM    293.23
CV003A4    MLFR.4    1.0
CV003BA      2.0      0.0
CV003BB      4.0      100.0
```

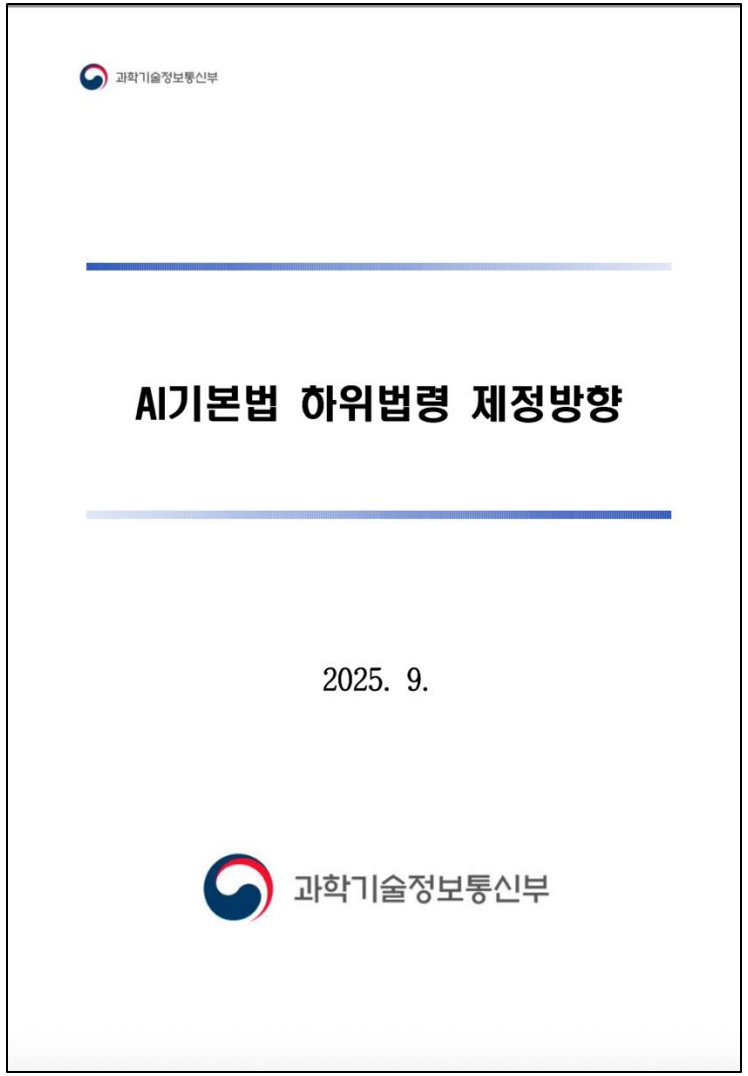
```
CF10100    'TEST'     L-GT    2    1.0    0.0
CF10111      1.0      0.0001       CVH-LIQLEV.008
CF10112      1.0      0.0       CVH-LIQLEV.007
* Connection between all the CVs and outermost volume
*
*                FLNAME KCVFM KCVTO ZFM    ZTO
FL01100      CV1-10    001    010       6.0    6.1
FL01101      100.0    0.1       1.0
FL01102      4        0    0    0
FL01103      1.0      1.0
FL01104      0.0      0.0
FL011S1      314.0      0.1    0.2
*
*                FLNAME KCVFM KCVTO ZFM    ZTO
FL01200      CV2-10    002    010       5.0    5.1
FL01201      100.0    0.1       1.0
FL01202      4        0    0    0
FL01203      1.0      1.0
FL01204      0.0      0.0
FL012S1      314.0      0.1    0.2
*
*                FLNAME KCVFM KCVTO ZFM    ZTO
FL01300      CV3-10    003    010       4.0    4.1
FL01301      100.0    0.1       1.0
FL01302      4        0    0    0
FL01303      1.0      1.0
FL01304      0.0      0.0
FL013S1      314.0      0.1    0.2
*
*                FLNAME KCVFM KCVTO ZFM    ZTO
FL01400      CV4-10    004    010       3.0    3.1
FL01401      100.0    0.1       1.0
FL01402      4        0    0    0
FL01403      1.0      1.0
FL01404      0.0      0.0
FL014S1      314.0      0.1    0.2
*
*                FLNAME KCVFM KCVTO ZFM    ZTO
FL01500      CV5-10    005    010       2.0    2.1
FL01501      100.0    0.1       1.0
FL01502      4        0    0    0
FL01503      1.0      1.0
FL01504      0.0      0.0
FL015S1      314.0      0.1    0.2
*
```

But not always…





AI 기본법 26년 1월 시행 예정



붙임 **AI사업자별 사업자 책무 대상 구분**

사업자 책무 구분		사업자 책무의 구체적 내용(안)	개발사업자 책무	이용사업자 책무
위험관리방안의 수립·운영 (제1호)		• 위험관리계획을 구축하고 관련 절차 및 수행기준을 마련	○	○
		• 위험관리조직을 구성하거나 조직 내 담당 인력 지정	○	○
		• 고영향AI 시스템으로 인한 합리적으로 예측가능한 범위의 잠재적 위험을 식별, 분석·평가	○	○
		• 식별된 AI위험에 대응하기 위해 위험을 처리하여 허용가능한 수준으로 완화	○	○
설명 방안의 수립·시행 (제2호)	AI 도출 결과 주요기준 근거마련	• 기술적으로 가능한 범위에서 투명성·설명가능성 확보를 위한 기술적 조치 적용	○	-
	학습용 데이터 정보의 관리	• AI에 활용된 학습용 형식, 수량, 크기, 정의·관리		
	설명 방안의 수립·시행	• 설명을 위한 절차, 범위		
		• 실제 이용자에게 설명방안 시행	-	○
이용자 보호방안 (제3호)		• 데이터의 적법하고 안전한 수집	○	○
		• 안전한 알고리즘 설계 및 시스템 개발 수행	○	-
		• 다양한 위험에 대비한 충분한 평가·테스트 실시	○	○
		• 운영시 발생하는 문제에 대한 실시간 모니터링 기반 구축	○	○
		• 이용자 피드백 반영 및 개선 프로세스 구축	○	○
		• 개인정보 등 이용자 권리보장 정책 수립	-	○
고영향AI에 대한 사람의 관리·감독 (제4호)		• 고영향AI에 대해 사람이 AI의 동작에 개입할 수 있는 기준 및 절차, 방법을 마련	○	○
		• 성능저하 및 오류 예방을 위한 정기적 점검 계획 수립	○	○
		• 이용사업자의 고영향AI에 대한 이해도 제고를 위한 교육 훈련 방안 마련	○	-
		• 이용자의 고영향AI 이해도 제고 및 안전한 이용을 위한 교육 훈련 방안 마련	-	○
문서의 작성·보관 (제5호)		• 사업자 책무 사항에 관하여 문서 작성 및 주기적 점검, 최신 기술·방법론을 적용하여 관리	○	○

개발, 활용, 검토하는 인공지능 기술에 대한 이론적인 이해의 중요성

Source of lecture materials

- 대한기계학회 기계인공지능연구회 2023, 2024, 2025 강습회 자료
- 가천대학교 김낭중 교수님 2024 강습회 강의자료
- 중앙대학교 이수영 교수님 2025 강습회 강의자료
- 한국전기연구원 김태완 박사님 2025 강습회 강의자료
- Andew Ng et al., CS229 Lecture Notes (Stanford University)
- Illustrations from various columns (noted separately)

Tips for Beginners

Stanford Online ‘CS229’ Andrew Ng

대한기계학회-기계인공지능연구회 ‘AI Bootcamp’

https://sites.google.com/view/aiksme/home?authuser=0

Topics	Colab	Slides	Youtube
Python Programming		pdf#00	
Introduction to Artificial Intelligence (AI)		pdf#01	iYoutube#01
End-to-End Machine Learning	iColab#02	pdf#02	iYoutube#02
Regression, Classification	iColab#03	pdf#03, pdf#04	iYoutube#03
Dimension Reduction	iColab#05	pdf#05	iYoutube#04
Clustering	iColab#06	pdf#06	iYoutube#05
Artificial Neural Networks (ANN)	iColab#07	pdf#07	iYoutube#06, iYoutube#07
Autoencoder	iColab#08	pdf#08	iYoutube#08
Convolutional Neural Networks (CNN)	iColab#09	pdf#09	iYoutube#09
Long Short-Term Memory (LSTM)	iColab#10	pdf#10	iYoutube#10
Reinforcement Learning (RL)		pdf#11	iYoutube#11
Value-based RL		pdf#12	iYoutube#12
Policy-based RL	iColab#13	pdf#13	iYoutube#13
기계공학에서 LLM 활용	pdf ChatGPT4	pdf#14	iYoutube#14
Language-based Robotics		pdf#15	iYoutube#15
ChatGPT 활용 화성 탐사선 제어		pdf#16	iYoutube#16

Table of Contents

Computational science: solve the equation!

- **general PDE form**:

$$\mathcal{F}\left(\mathbf{u}(\mathbf{x},\mathrm{t}), \nabla\mathbf{u}(\boldsymbol{x},t), \nabla^2\mathbf{u}(\boldsymbol{x},t)\right) = 0, \qquad (\mathbf{x},t) \in \Omega \times [0,T]$$

- example 1) **Navier-Stokes equation**: $\mathcal{F}(\mathbf{u}, \nabla\mathbf{u}, \nabla^2\mathbf{u}) = \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu\nabla^2\mathbf{u}$

- example 2) **Fourier's Law**: $\mathcal{F}(T, \nabla^2 T) = \dfrac{\partial T}{\partial t} - \alpha\nabla^2 T$

- example 3) **Faraday's Law**: $\mathcal{F}(B, \nabla\times E) = \dfrac{\partial B}{\partial t} + \nabla\times E$

Computational science: solve the equation!

$$\frac{d^2u(x)}{dx^2} = f(x) \quad \text{in } \Omega \text{ (1-D Poisson equation)}$$

finite difference method (FDM): $\dfrac{u(x+\Delta x) - 2u(x) + u(x-\Delta x)}{\Delta x^2} = f(x)$

finite volume method (FVM): $\displaystyle\int_V \frac{d^2u(x)}{dx^2}\,dV = \int_V f(x)\,dV \quad \longrightarrow \quad \int_S \frac{du}{dx}\cdot \mathbf{n}\,dS = \int_V f(x)\,dV$

finite element method (FEM): $\displaystyle\int_\Omega \frac{du(x)}{dx}\frac{dv(x)}{dx}\,dx = \int_\Omega f(x)v(x)\,dx$

$$u_h(x) = \sum_{i=1}^{N} u_{hi}\,\phi_i(x),\ v_h = \phi_j(x) \quad \longrightarrow \quad \underline{A_h}\,\underline{u_h} = \underline{F_h}$$

$$A_{h\,ij} = \int_\Omega \frac{d\phi_i(x)}{dx}\frac{d\phi_j(x)}{dx}\,dx$$

$$F_{h\,ij} = \int_\Omega f\phi_j(x)\,dx$$

Machine learning: approximate the equation!

- For a given target function $f(x)$, we aim to approximate it as a polynomial centered at x_0 and parametrized by the coefficients $a_0, a_1, \cdots, a_n$.

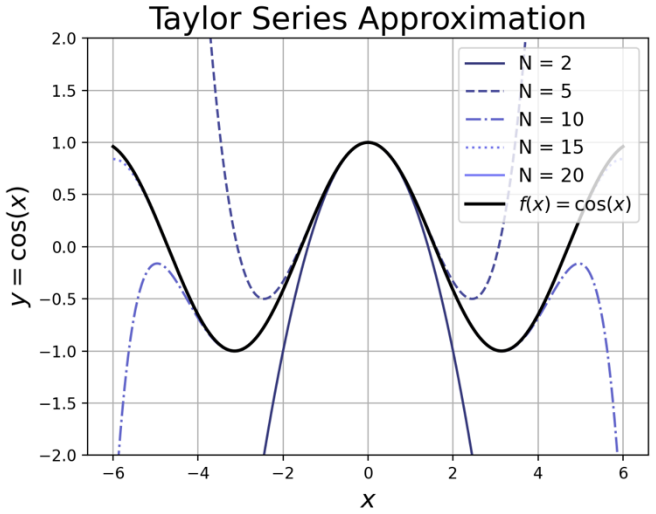
$$f(x) \approx a0 + a_1(x - x_0) + a_2(x - x_0)^2 + a_3(x - x_0)^3 + \cdots + a_n(x - x_0)^n$$

- How to determine the parameters (coefficients) which best approximate the target function?

$$a_k = \frac{f^k(x_0)}{k!}$$

- Is this a reliable approximation?

  Convergence of the Talyor Series



Machine learning: approximate the equation!

- For a given target function $f(x)$, we aim to approximate it as a trigonometric function parametrized by the coefficients $a_1, \cdots, a_n$ and $b_1, \cdots, b_n$.

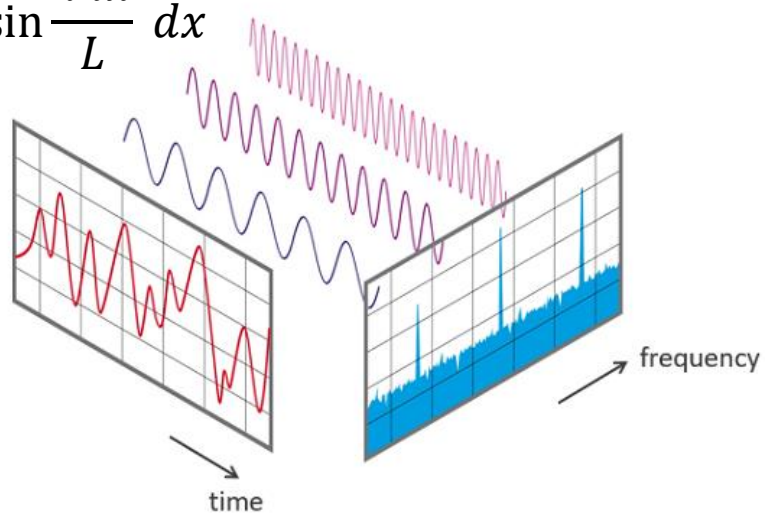
$$f(x) \approx \frac{a_0}{2} + \sum_{k=1}^{n} \left(a_k \cos\frac{k\pi x}{L} + b_k \sin\frac{k\pi x}{L}\right)$$

- How to determine the parameters (coefficients) which best approximate the target function?

$$a_k = \frac{1}{L}\int_{-L}^{L} f(x)\cos\frac{k\pi x}{L}\,dx\,, b_k = \frac{1}{L}\int_{-L}^{L} f(x)\sin\frac{k\pi x}{L}\,dx$$

- Is this a reliable approximation?

  Convergence of the Fourier Series



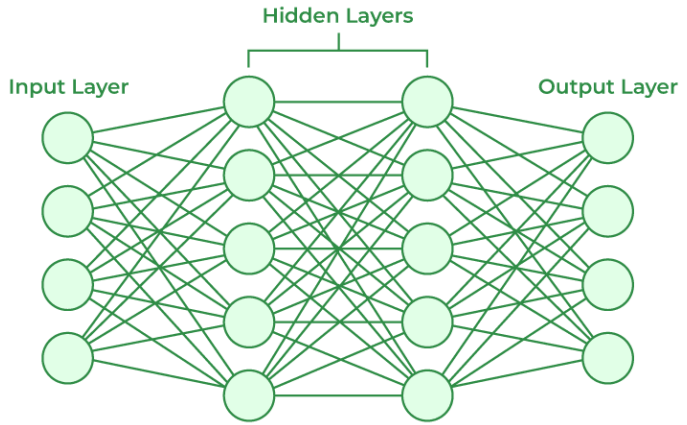
Machine learning (neural networks)

- For a given target function $f(x)$, we aim to approximate it as a an L-layer feed-forward neural network parametrized by θ (weight and bias).

$$f^1(x) = W^1 x + b^1$$
$$f^\ell(x) = W^\ell \sigma\left(f^{\ell-1}(x)\right) + b^\ell \text{for } 2 \le \ell \le L \implies f(x) \approx f^L$$

- How to determine the parameters (coefficients) which best approximate the target function?

  Set a loss function and minimize it ('learning' as well as 'approximation')



- Is this a reliable approximation?

  Universal approximation theorem

Neural networks: universal nonlinear function approximator

Universal Approximation Theorem

For a continuous function f, there exists a sequence of growing neural networks $\{f_n\}$ such that

$$\lim_{n\to\infty} \max_{x} |f_n(x) - f(x)| = 0.$$

- Arbitrary width case for two-layer neural networks ($L = 2$, $n_1 \to \infty$).

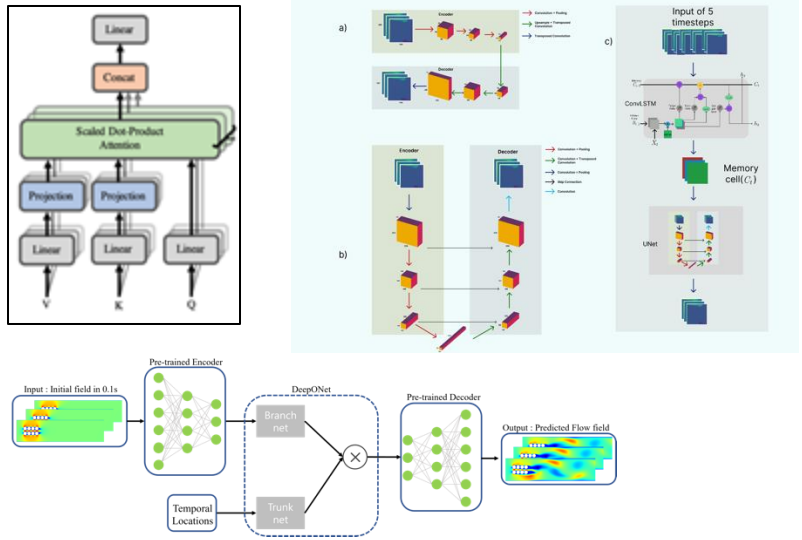
  G. Cybenko,
  Approximation by superpositions of a sigmoidal function,
  Math. Control Signals Systems, 2(4):303–314, 1989.

  K. Hornik,
  Approximation capabilities of multilayer feedforward networks,
  Neural Networks, 4(2):251–257, 1991.

- Bounded width and arbitrary depth case ($n_i < \infty$, $L \to \infty$).

  P. Kidger and T. Lyons,
  Universal Approximation with Deep Narrow Network,
  Proceedings of Machine Learning Research, 2306–2327, 2020.

Deep learning architectures



Machine learning as a computational science.

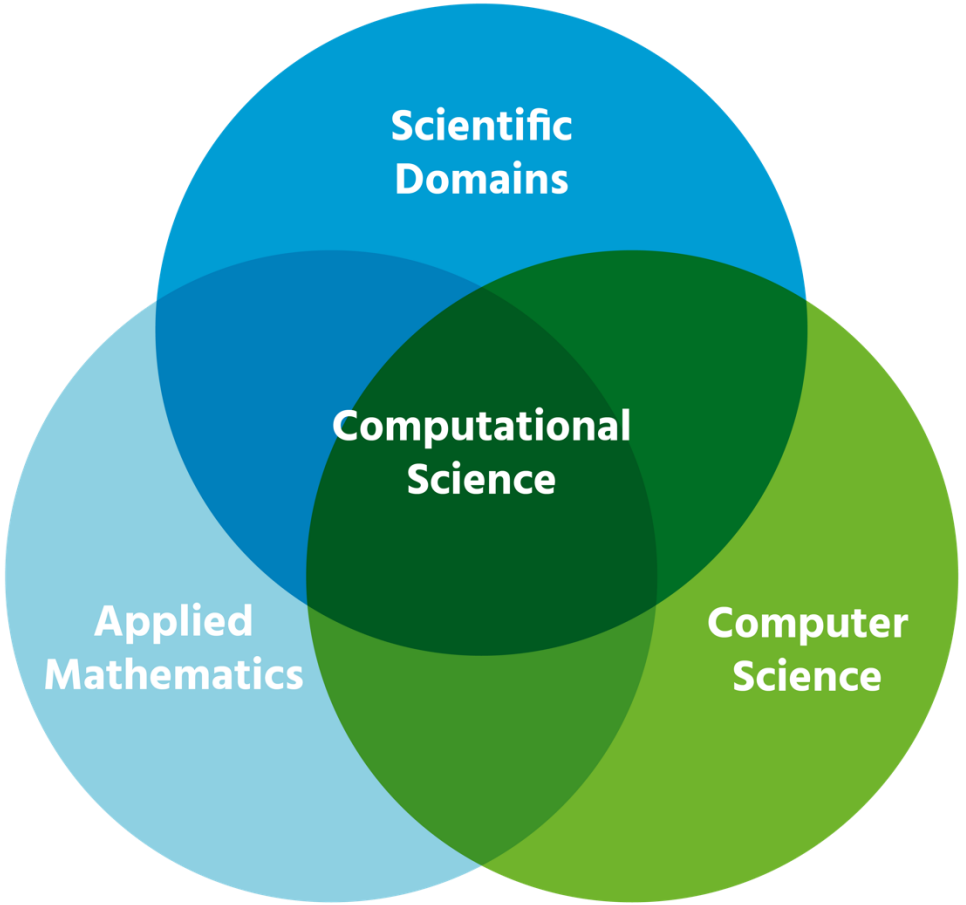
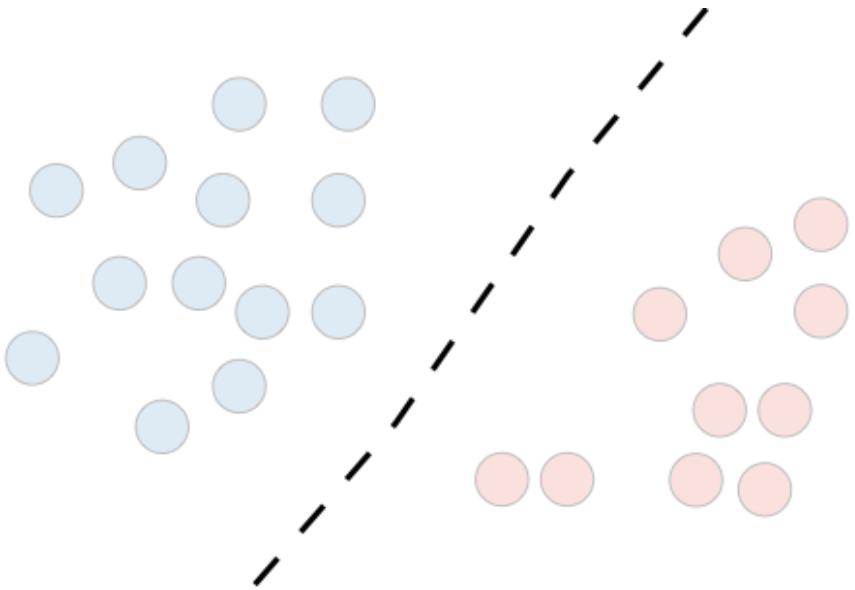
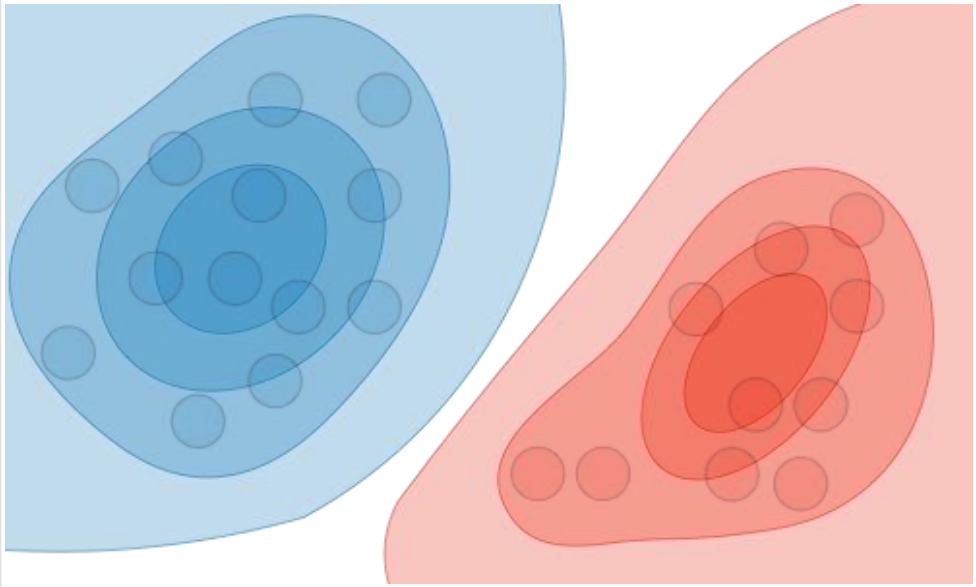


Table of Contents

Types of learning: Discriminative vs Generative

	Discriminative model	Generative model
Goal	Directly estimate $P(y\|x)$	Estimate $P(x\|y)$ to then deduce $P(y\|x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GDA, Naive Bayes

https://stanford.edu/%7Eshervine/teaching/cs-229/cheatsheet-supervised-learning

Generative Learning Algorithms

- Algorithms that try to learn $p(y|x)$ directly: Discriminative Learning Algorithms
- Algorithms that try to model $p(x|y)$: Generative Learning Algorithms

For instance:

$y(0|1) \rightarrow$ Dog | Elephant

$p(x|y) = 0 \rightarrow$ models distribution of dogs features

$p(x|y) = 1 \rightarrow$ models distribution of elephants features

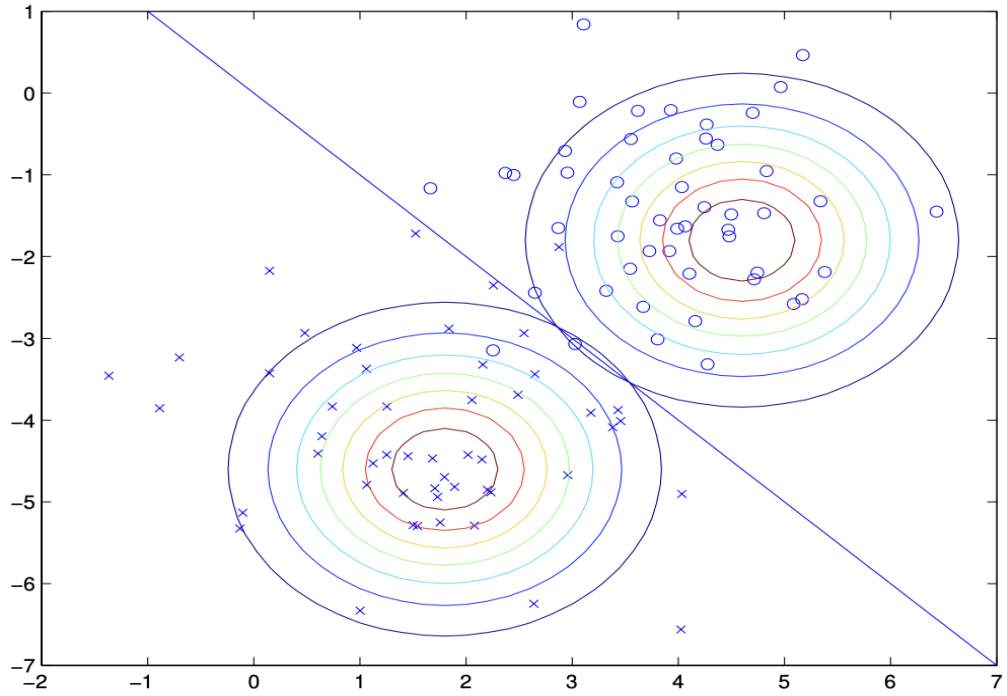
Gaussian Discriminant Analysis (GDA) model

$$p(y|x) = \frac{p(x|y)\,p(y)}{p(x)}$$

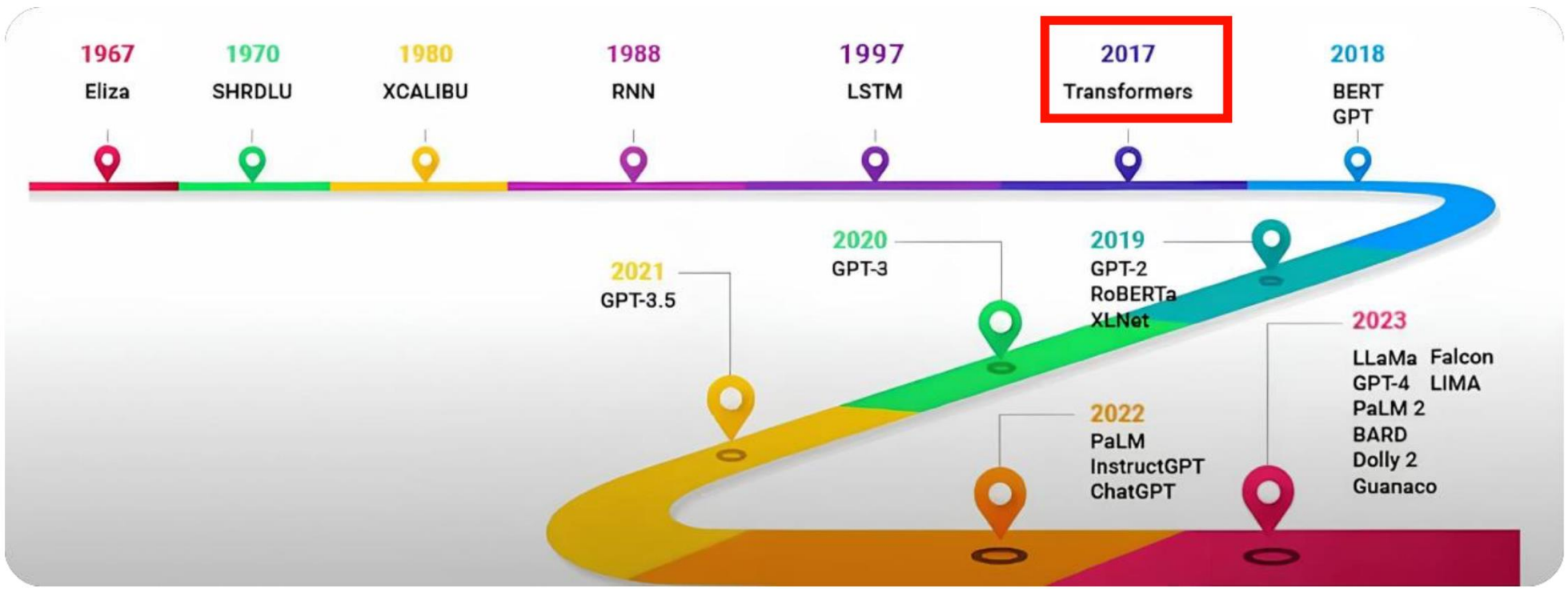
$$y \sim \text{Bernoulli}(\phi)$$

$$x|y = 0 \sim N(\mu_0, \Sigma)$$

$$x|y = 1 \sim N(\mu_1, \Sigma)$$



And now…

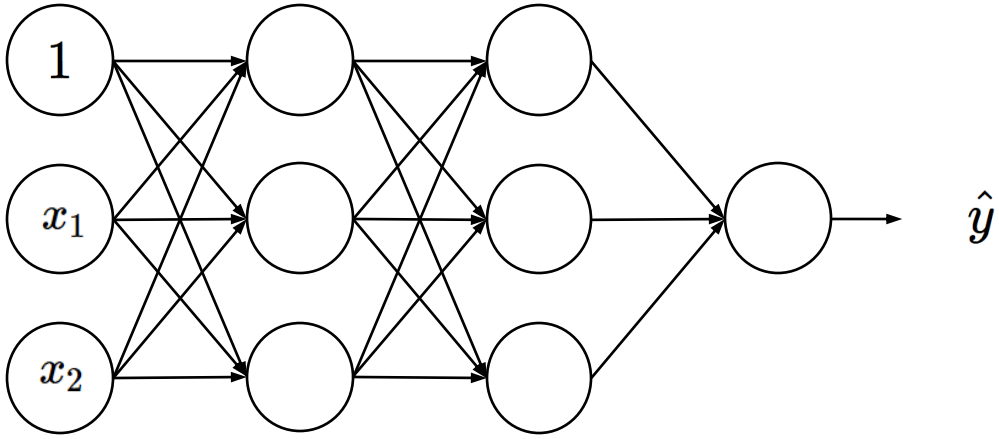


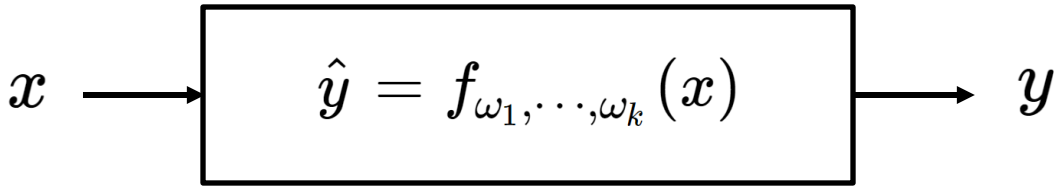
- **2017년 Transformer 공개**
  - 2017년 이후 거의 모든 언어모델은 Transformer 아키텍쳐를 기반으로 설계 됨
  - 대규모 언어모델의 시초
    - 이 전까진 상기한 문제들로 인하여 대규모 파라미터, 대규모 데이터 활용에 한계가 있었음

Table of Contents

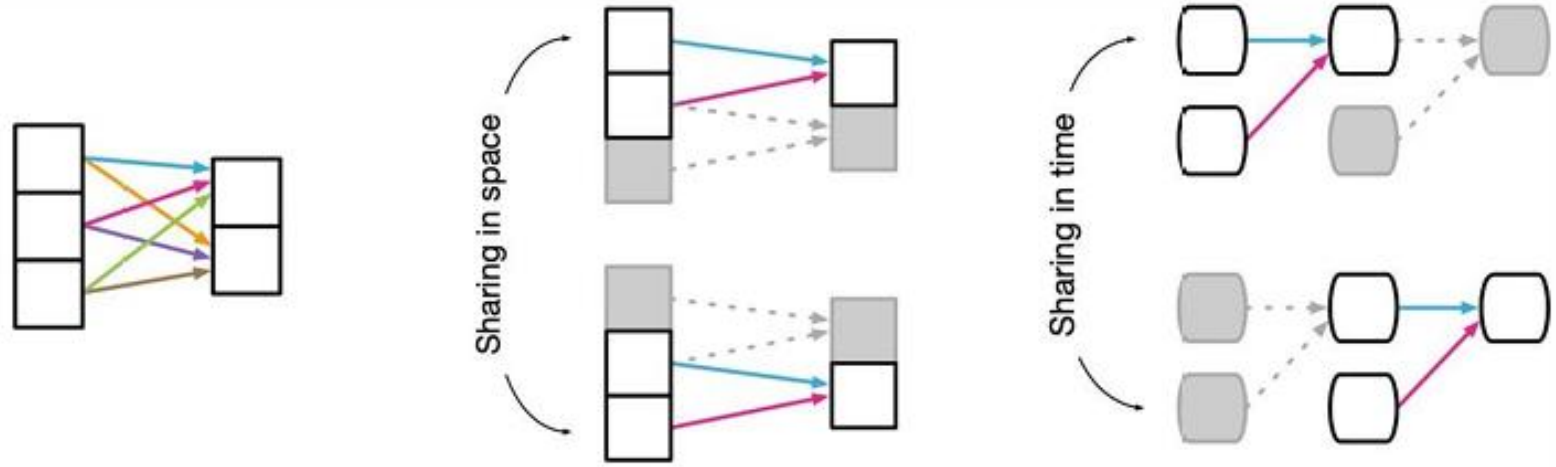
ANN – Multi Layer Perceptron

- Universal function approximator
- Nonlinear mapping can be represented by another neurons





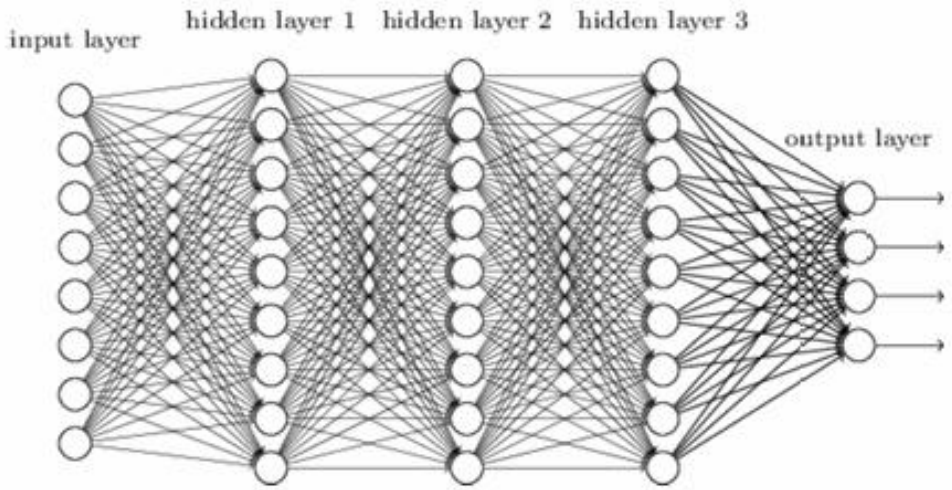
Relational inductive bias



(a) Fully connected (b) Convolutional (c) Recurrent

- Inductive bias is the set of assumptions that learning model uses to predict output for generalization.
- Relational inductive bias is the assumptions on relationships in input data.
- Relational inductive biases are one of the keys to modeling architectures such as FCN, ConvNet, RNN, GNN, or etc..

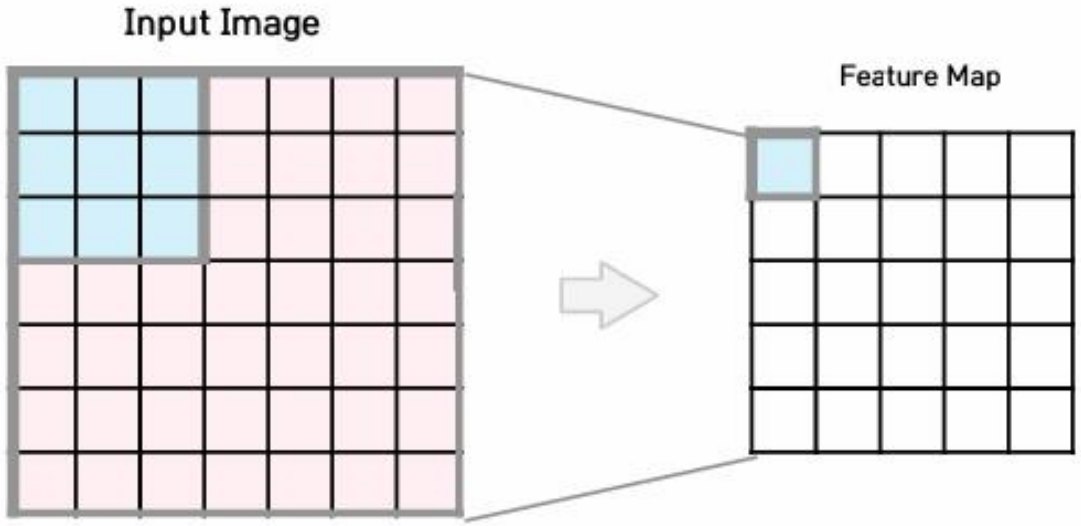
Relational inductive bias

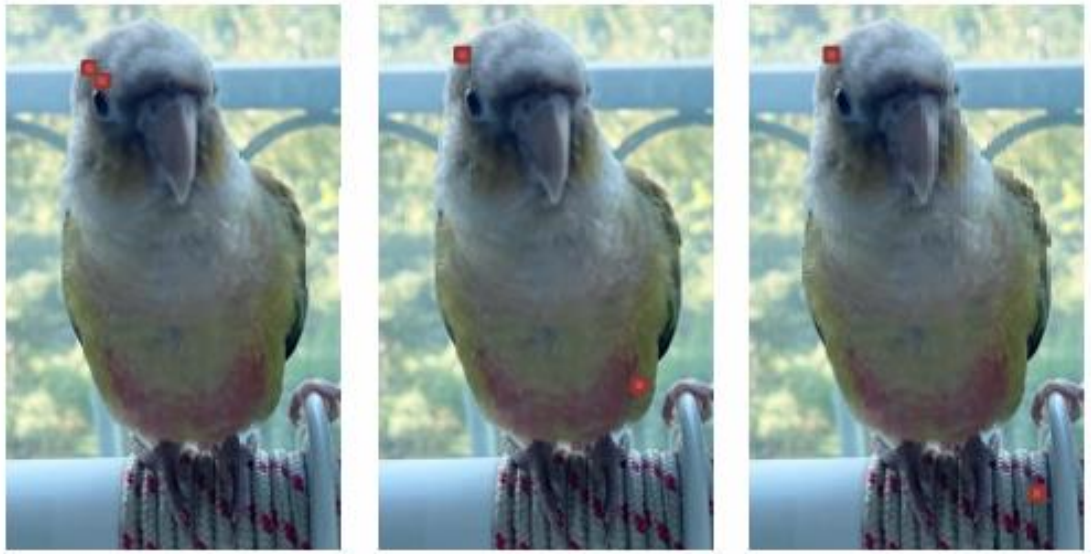


Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

Table 1: Various relational inductive biases in standard deep learning components. See also Section 2.

Relational inductive bias

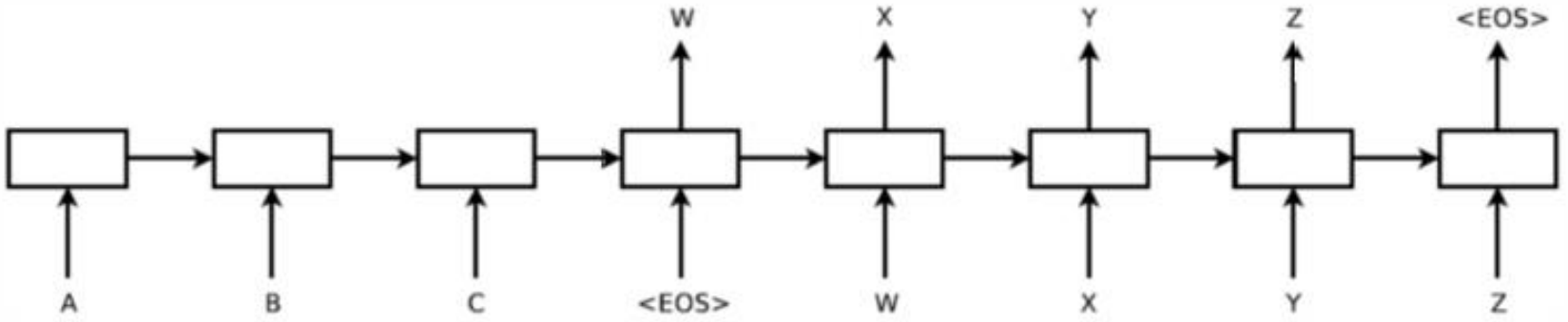




Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

Table 1: Various relational inductive biases in standard deep learning components. See also Section 2.

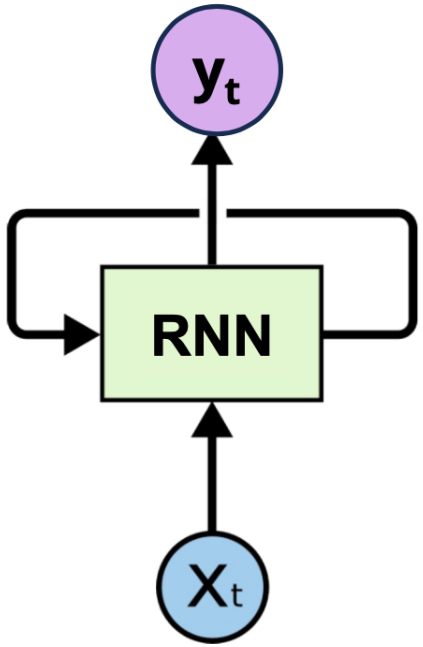
Relational inductive bias



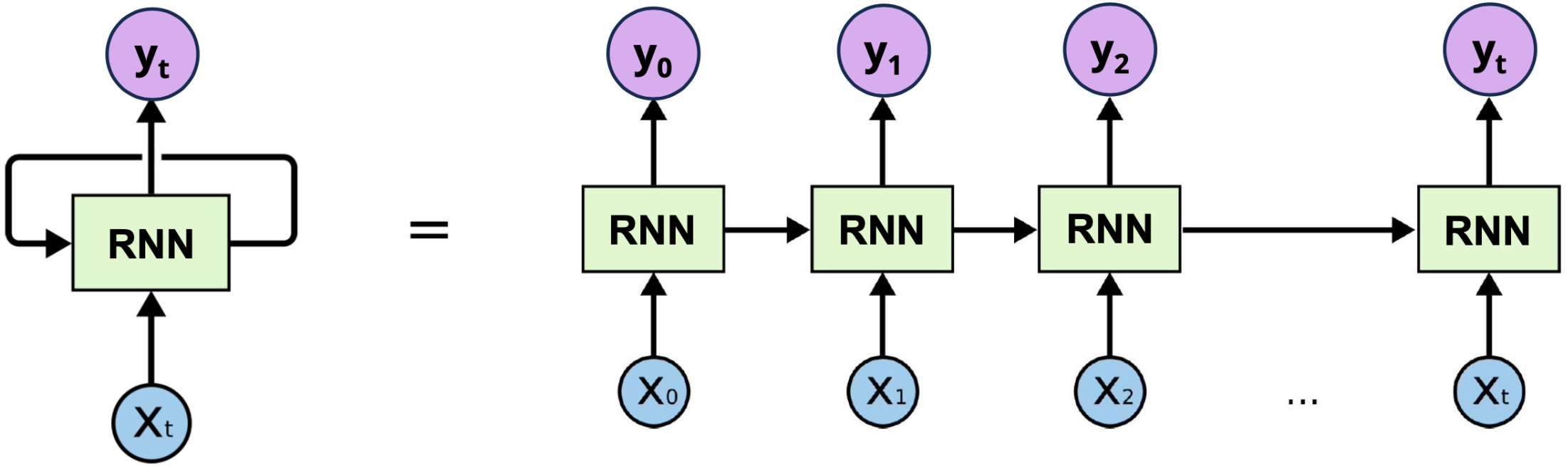
Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

Table 1: Various relational inductive biases in standard deep learning components. See also Section 2.

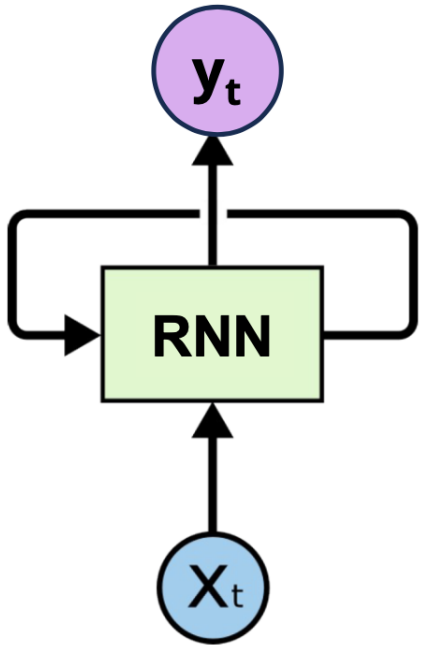
Recurrent Neural Network (RNN)



Recurrent Neural Network (RNN)



Recurrent Neural Network (RNN)



We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

$$h_t = f_W(h_{t-1}, x_t)$$

new state

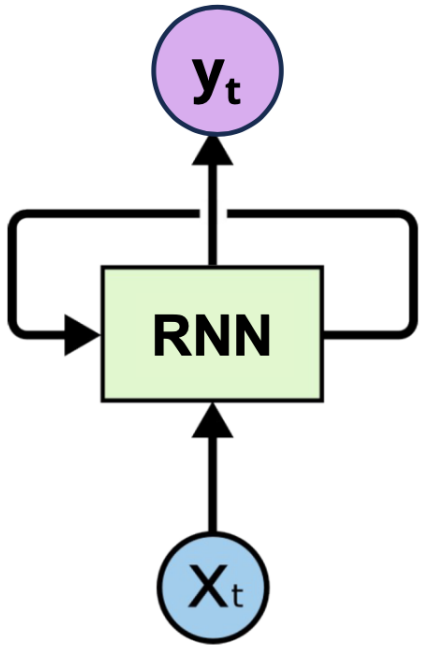
some function with parameters W

old state

input vector at some time step

Notice: the same function and the same set of parameters are used at every time step.

Recurrent Neural Network (RNN)



The state consists of a single *"hidden"* vector **h**:

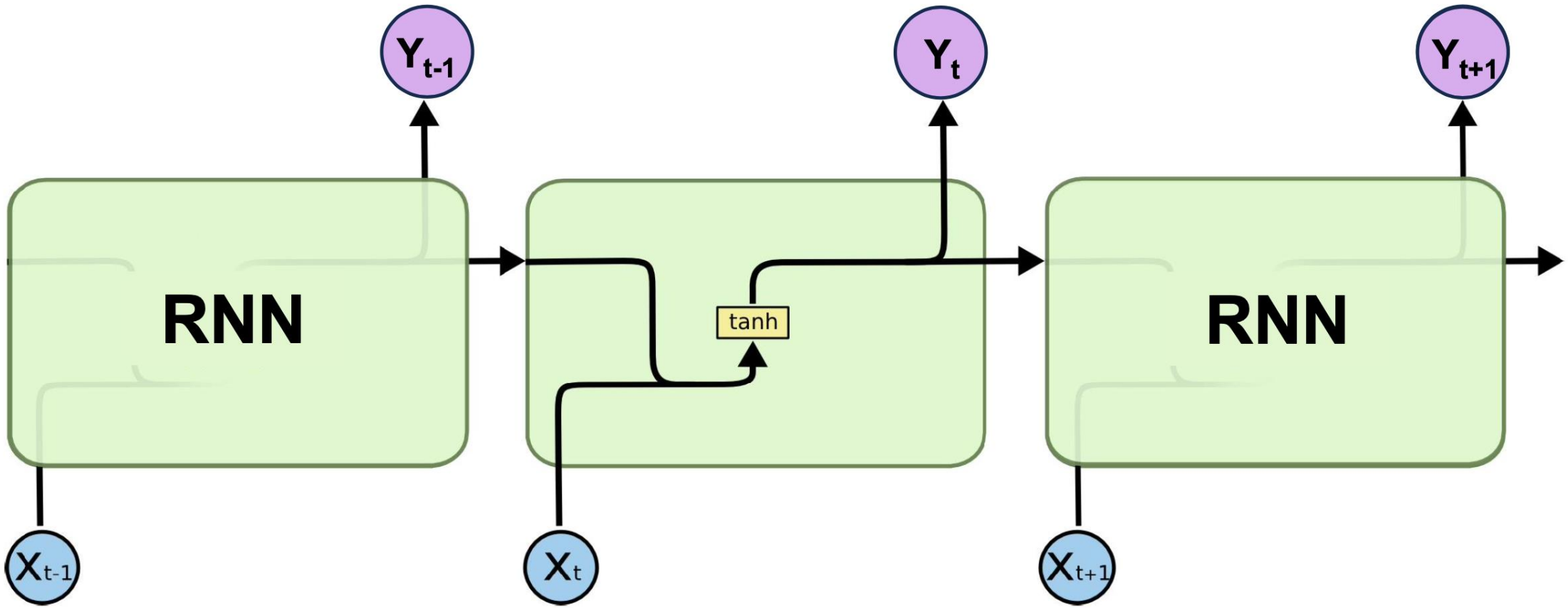
$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

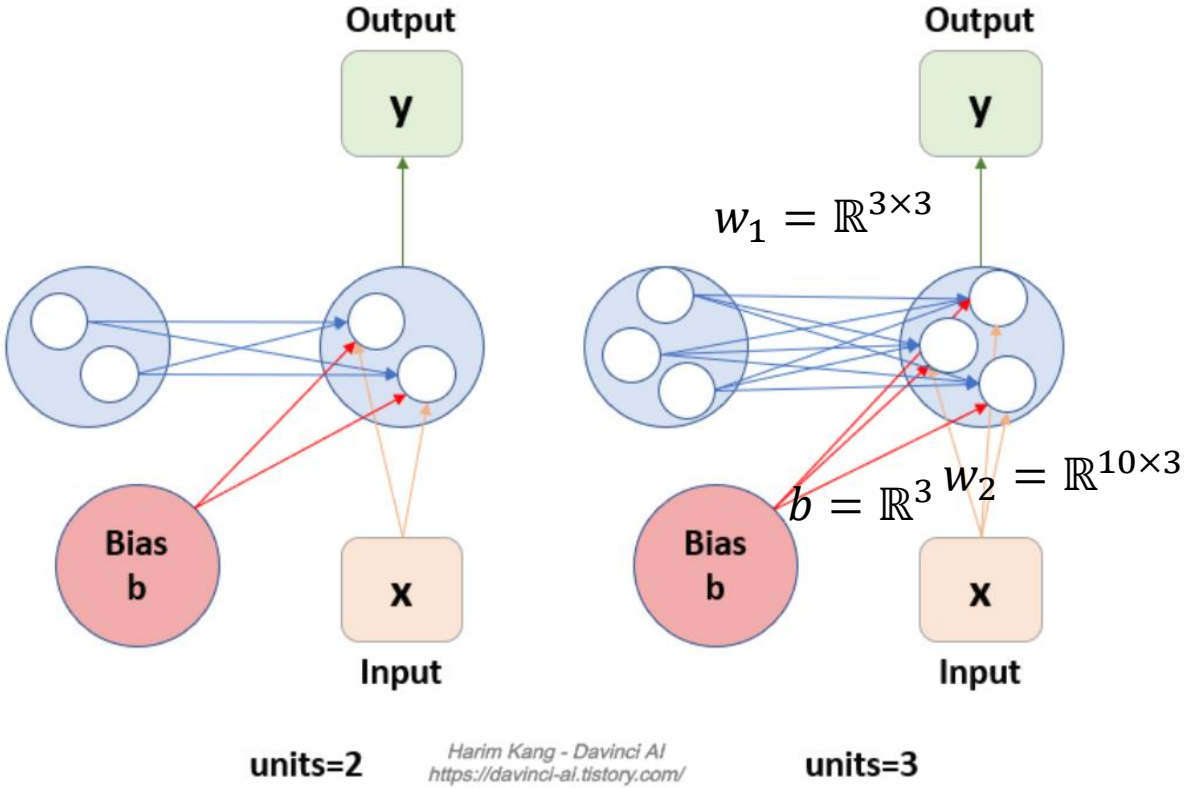
$$y_t = W_{hy}h_t + b_y$$

Sometimes called a "Vanilla RNN"

Recurrent Neural Network (RNN)



Let's calculate parameter numbers!




```
# Simple RNN 파라미터 참고
# => output_dim, ( time-length or setence-length, input_dim)
# => Dh, (t, d)

from keras.models import Sequential
from keras.layers import SimpleRNN

model = Sequential()
model.add(SimpleRNN(3, input_shape=(2,10)))
# model.add(SimpleRNN(3, input_length=2, input_dim=10))와 동일함.
model.summary()
```

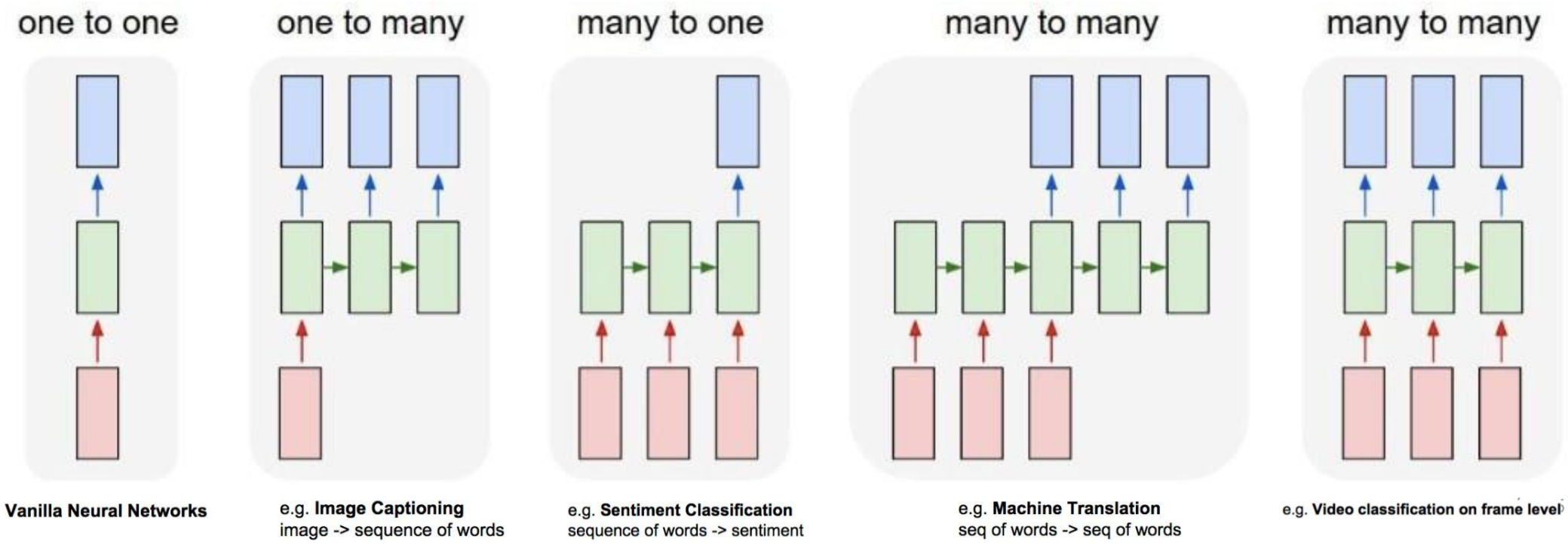
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_1 (SimpleRNN)     (None, 3)                 42
=================================================================
Total params: 42
Trainable params: 42
Non-trainable params: 0
_________________________________________________________________
```

$3 \times 3 + 10 \times 3 + 3 = 42$

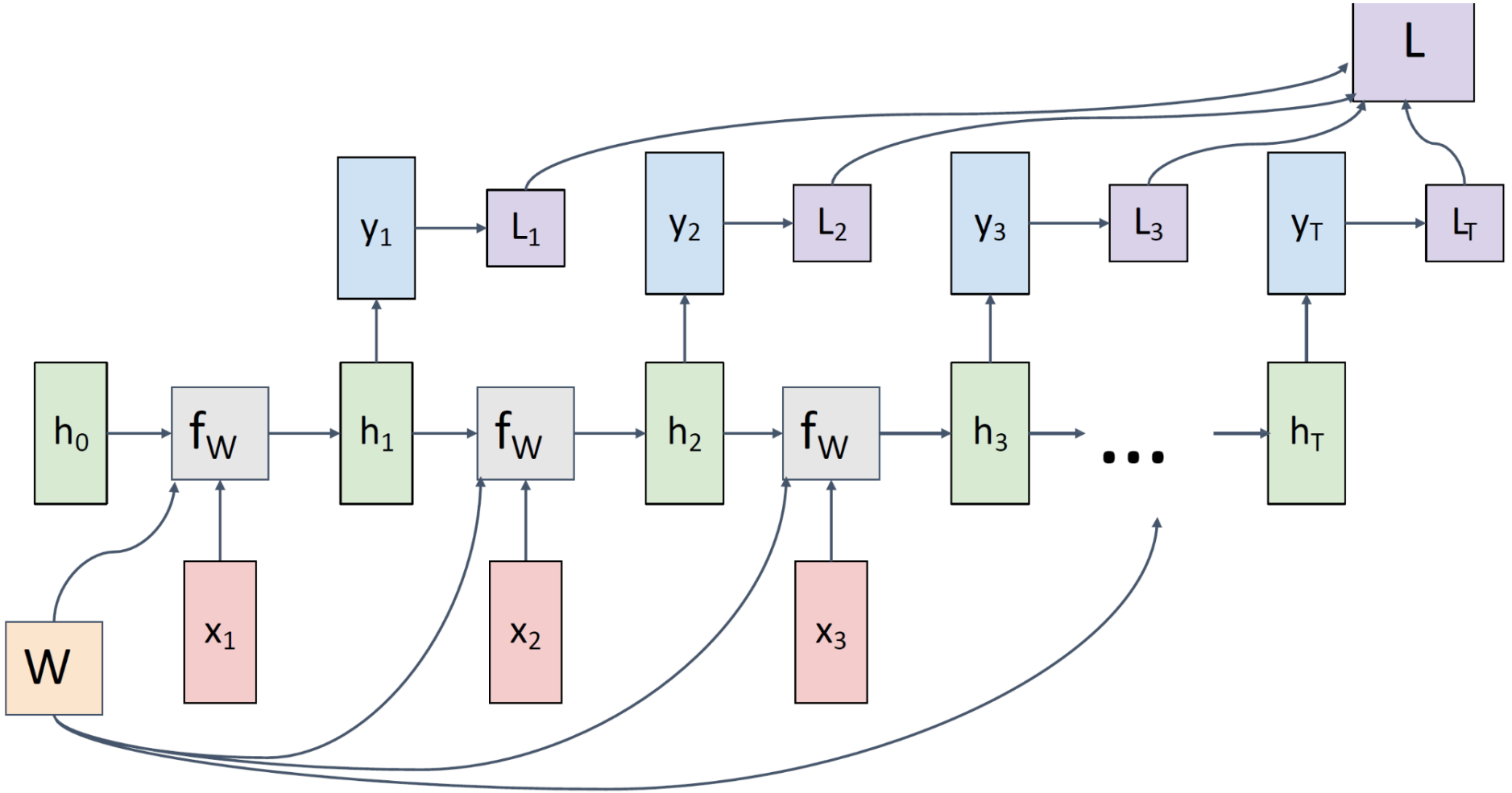
※ input 시계열 수와 상관없이 parameter 공유

Recurrent Neural Network (RNN)

- Variable length of sequence can be input in RNN, because of shared weights at each time step. RNN has various structure according to task types

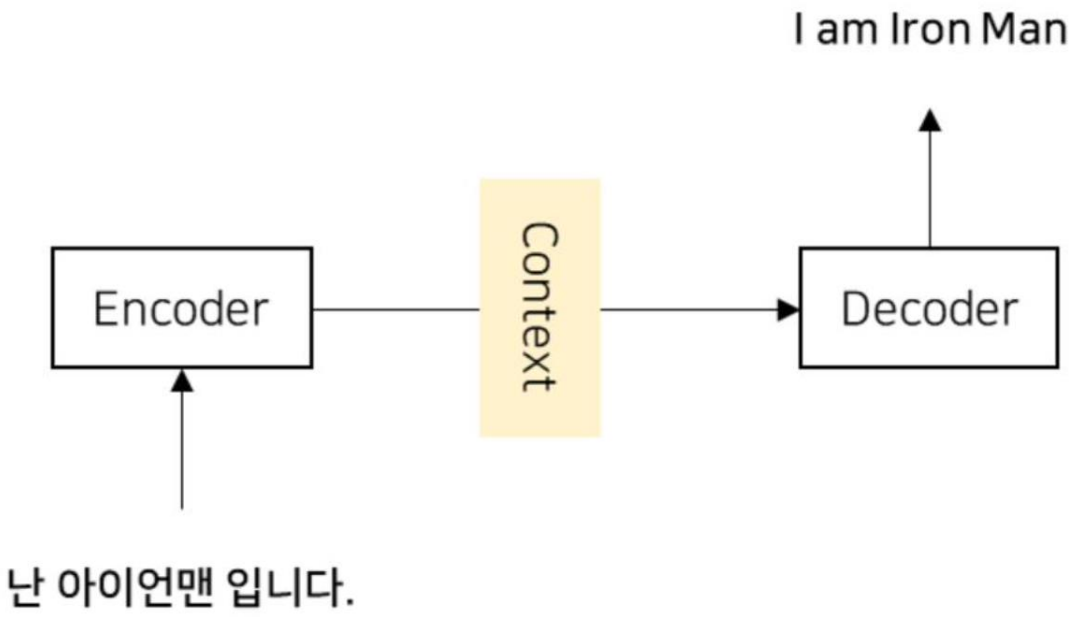


RNN Computational Graph (Many to Many)



Sequence to Sequence (seq2seq): Many to One + One to Many

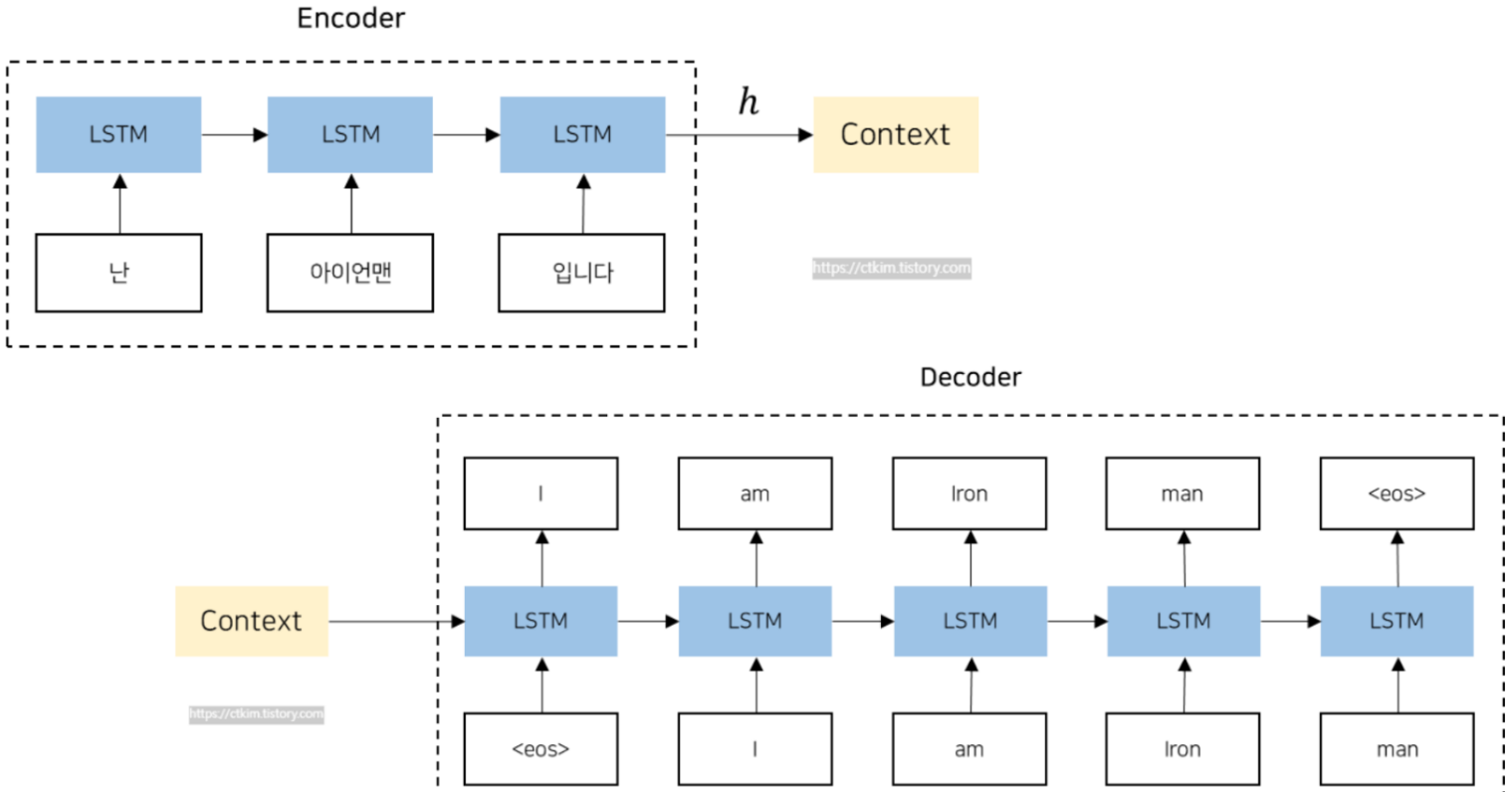
- Seq2seq is a model for training to convert sequences from one domain (e.g. sentences in English) to sequences in another domain (e.g. the same sentences translated to French).



https://ctkim.tistory.com/entry/RNN-seq2seq란 -무엇인가

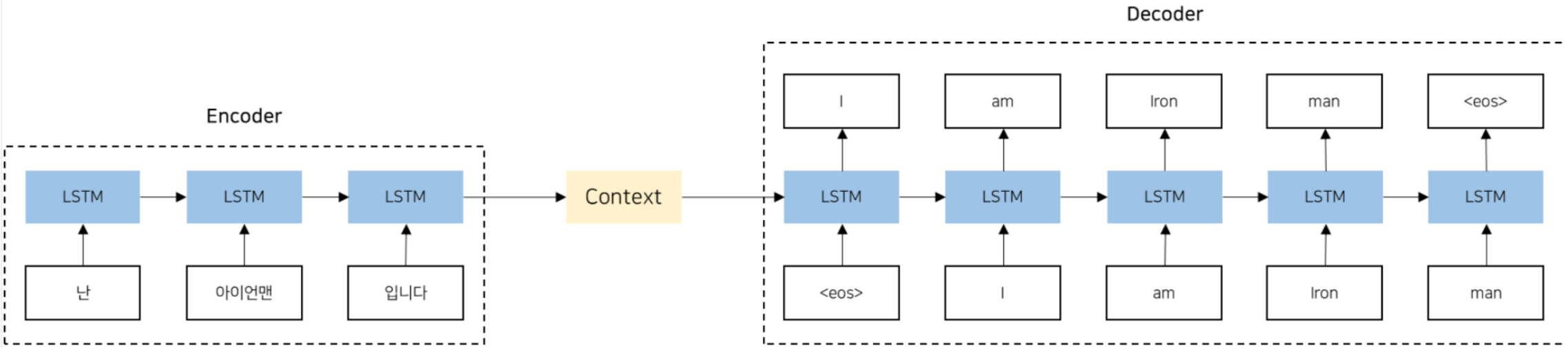
Sequence to Sequence (seq2seq): Many to One + One to Many

- Seq2seq is a model for training to convert sequences from one domain (e.g. sentences in English) to sequences in another domain (e.g. the same sentences translated to French).



Sequence to Sequence (seq2seq): Many to One + One to Many

- Seq2seq is a model for training to convert sequences from one domain (e.g. sentences in English) to sequences in another domain (e.g. the same sentences translated to French).



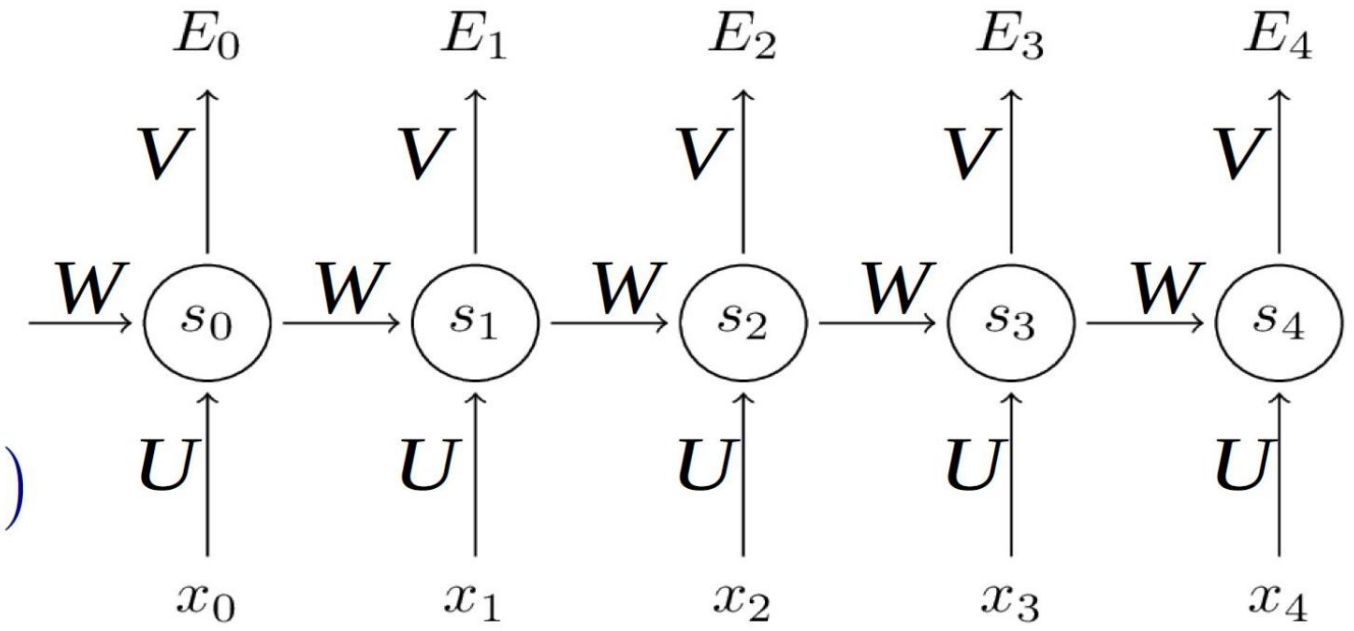
Problem 1) fixed size of context vector
Problem 2) gradient vanishing

Backpropagation Through Time

$$s_t = \tanh(Ux_t + Ws_{t-1})$$

$$\hat{y}_t = softmax(Vs_t)$$

$$E(y, \hat{y}) = -\sum_t E_t(y_t, \hat{y}_t)$$



Backpropagation Through Time

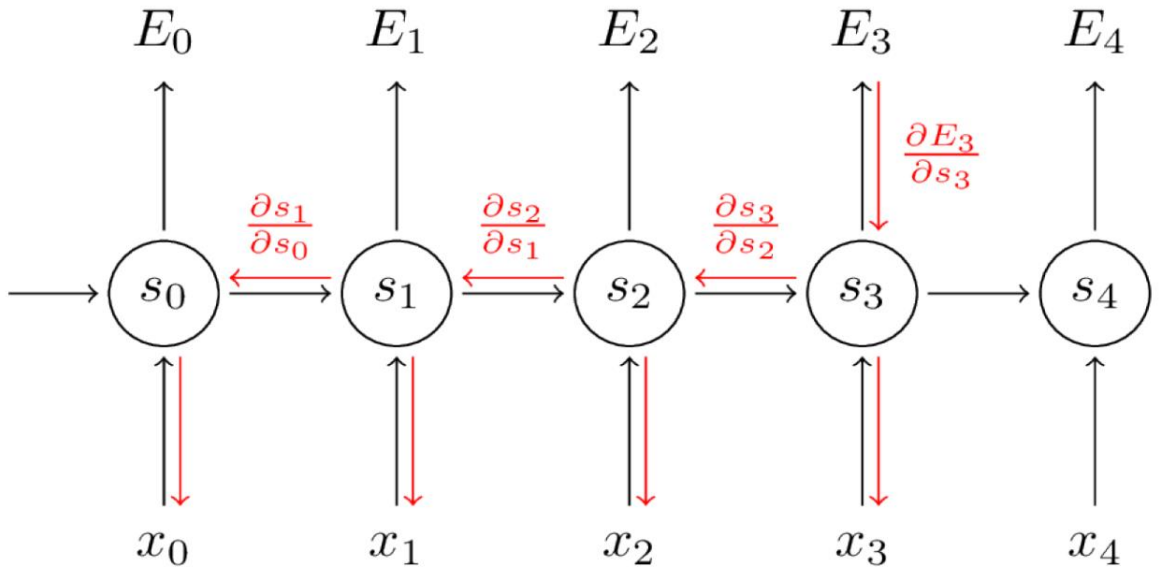
$$\frac{\partial E}{\partial W} = \sum_t \frac{\partial E_t}{\partial W}$$

$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial W}$$

But $s_3 = \tanh(Ux_t + Ws_2)$

S_3 depends on s_2, which depends on W and s_1, and so on.

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial s_k}\frac{\partial s_k}{\partial W}$$



Backpropagation Through Time

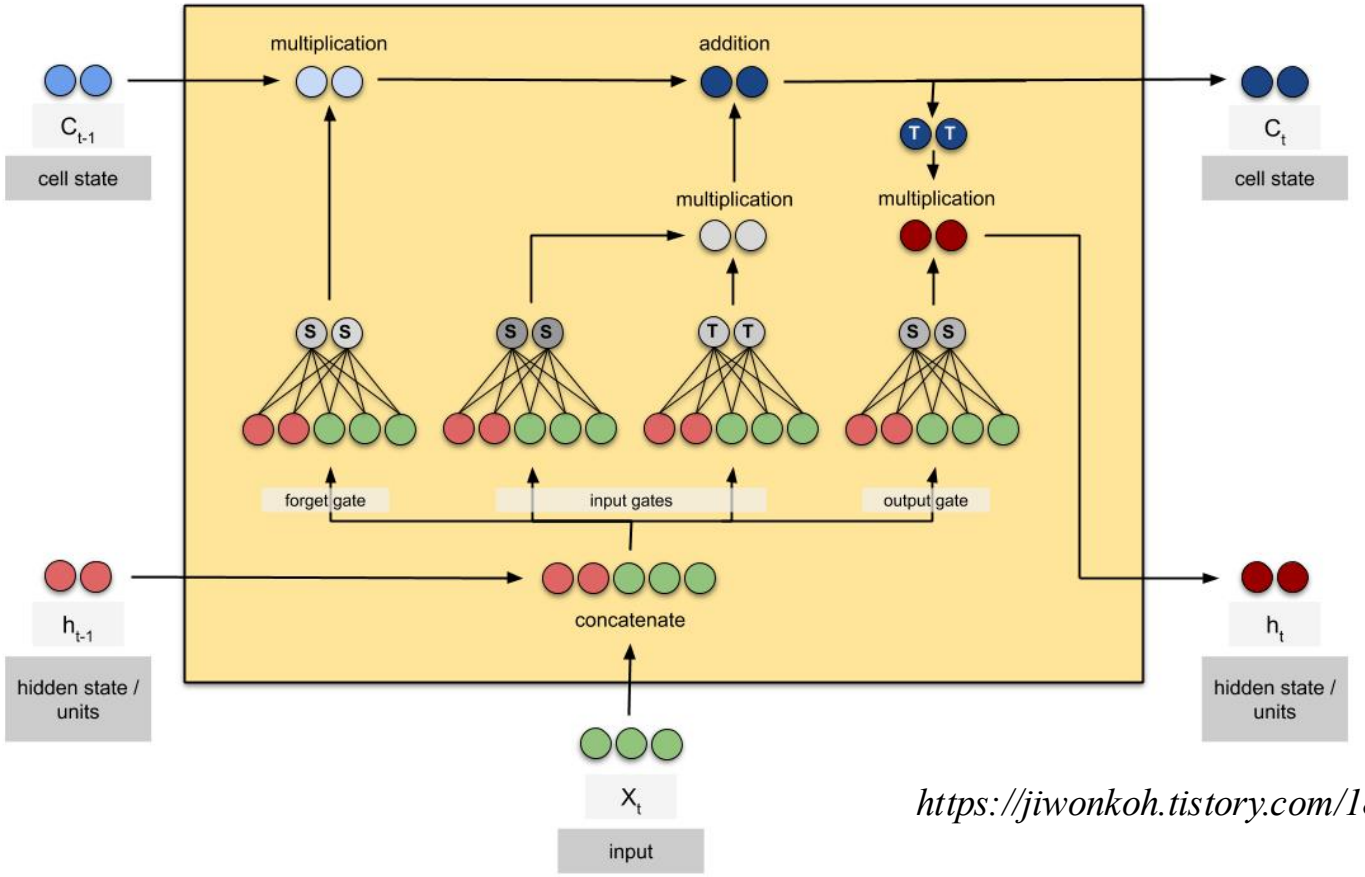
$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial s_3} \frac{\partial s_3}{\partial s_k} \frac{\partial s_k}{\partial W}$$

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial s_3} \left(\prod_{j=k+1}^{3} \frac{\partial s_j}{\partial s_{j-1}} \right) \frac{\partial s_k}{\partial W}$$

- **tanh** maps all values into a range between -1 and 1, and the **derivative is bounded by 1**.
- With multiple matrix multiplication, gradient values **shrink exponentially**.
- Gradient contribution from “far away” steps become zero.
- Depending on activation functions and network parameters, gradients could **explode** instead of vanishing.

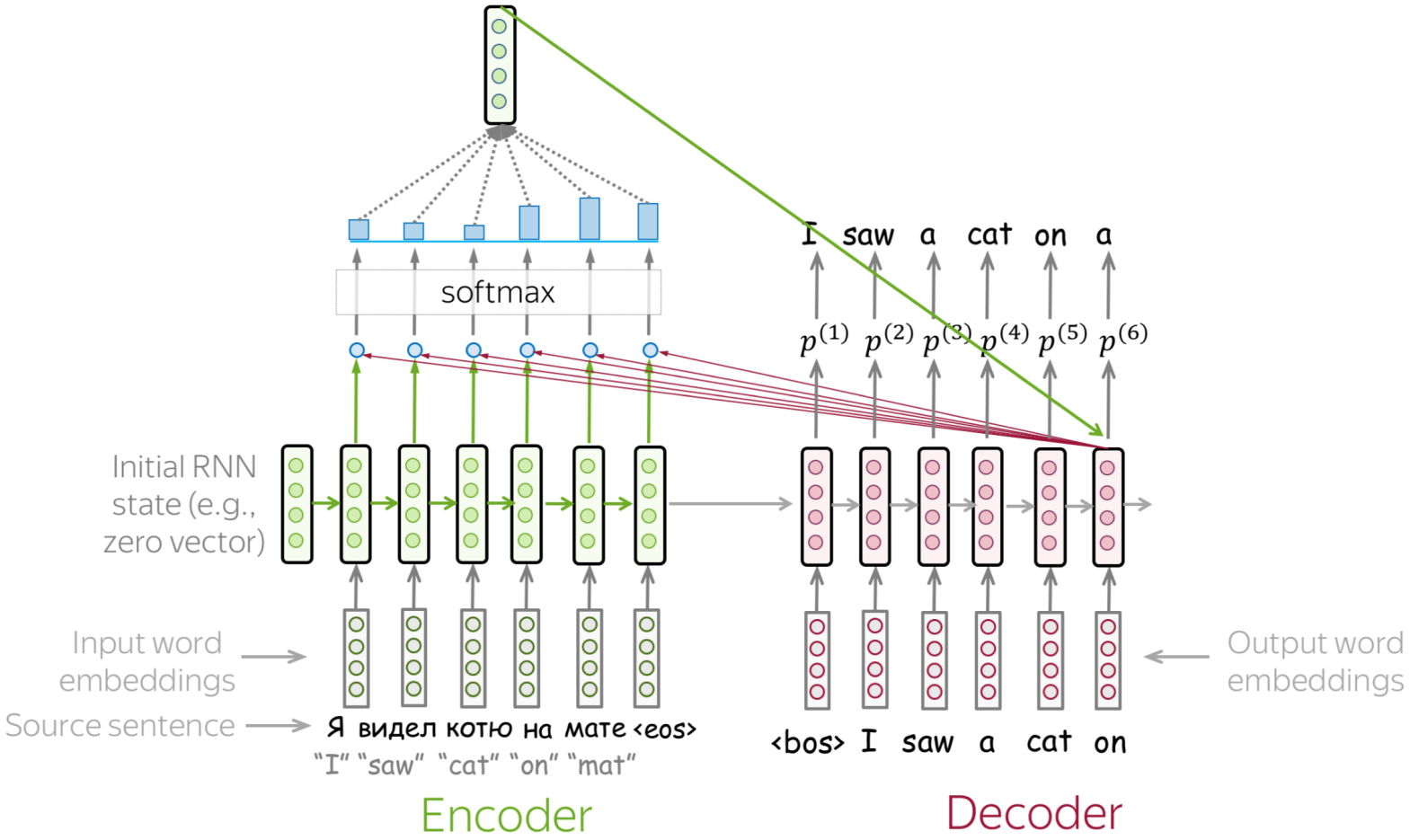
Long Short Term Memory (LSTM)

- In addition to the hidden state vector, $\boldsymbol{h_t}$ **(short-term),** LSTMs also maintain a memory vector $\boldsymbol{c_t}$ **(long-term)**



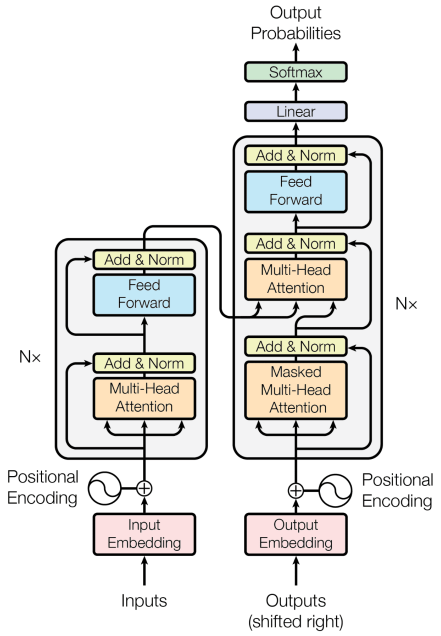
https://jiwonkoh.tistory.com/188

Seq2Seq with Attention Mechanism



Attention Mechanism: Attention is All You Need

- The fundamental concept that underpins a transformer is *attention*.
  - Originally developed as an enhancement to RNNs for machine translation.
  - Later showed significantly improved performance by eliminating the recurrence structure and focusing exclusively on the attention mechanism.



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Vaswani *et al.* (2017)

Table of Contents

Recurrent Neural Network (RNN)

RNN

LSTM

Transformer

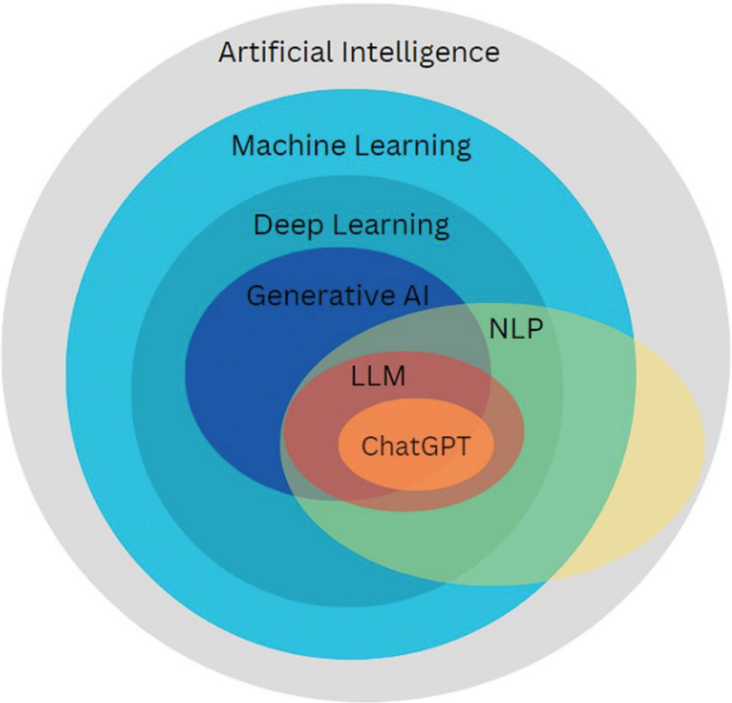
Large Language Model (LLM)

▪ LLM 정의?

- 대규모 (Large): 수십억에서 수천억 개의 매개변수를 갖는 거대한 신경망 모델
  - 소형 언어모델: 수백만~ 수업억 개의 매개변수 수준
- 언어 (Language): 방대한 양의 언어적 표현을 담고 있는 텍스트 데이터로 학습
- 모델(Model): 언어적 표현의 의미, 맥락, 지식을 확률적으로 학습 하여 내재화

▪ Nature Language Processing 의 최종 진화형

- 지금까지의 모든 인공지능 기술들을 종합
- 막대한 규모의 데이터와 초거대 모델 파라미터를 활용
- 인간의 자연어를 다룬다는 점에서 범용성이 매우 뛰어남

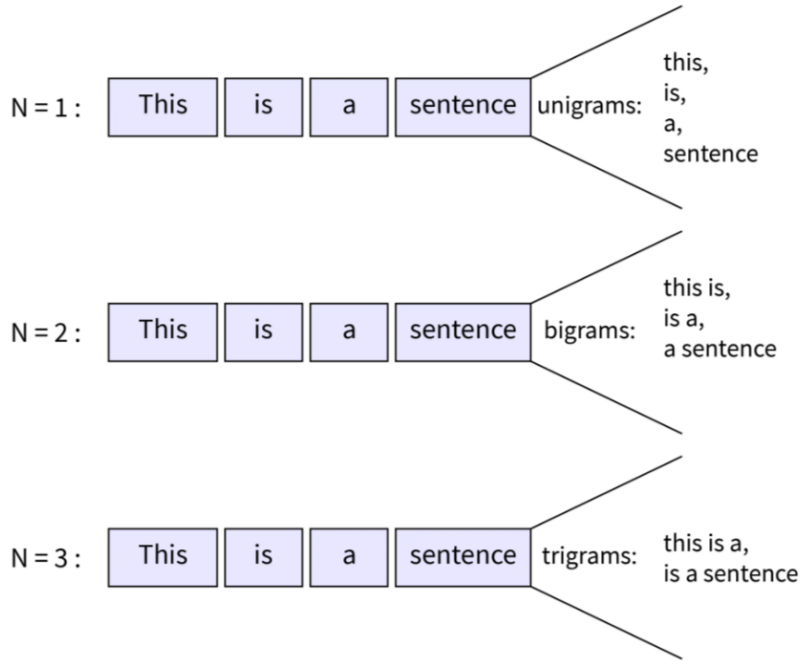


https://www.researchgate.net/figure/Evolution-of-LLM-Illustrating-the-interplay-and-overlap_fig2_381943056

Natural Language Processing (NLP)의 진화: 통계적 언어 모델

▪ 통계적 언어모델

- 특정 단어 시퀀스의 등장의 자연스러움 정도를 확률로 계산
- 과거의 통계적 데이터 빈도에 의존
- 조건부 확률의 연쇄 법칙을 따라 계산 됨
  - $P(w_1, w_2, \ldots, w_T) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2) \times \cdots \times P(w_T|w_1, \ldots, w_{T-1})$
- 대표적인 통계적 언어모델: N-gram 모델
  - 마크로브 가정을 통해 바로 직전의 N-1 개에 대한 의존성만을 고려
  - $P(w_T|w_1, \ldots, w_{T-1}) \approx P(w_T|w_{T-(N-1)}, \ldots, w_{T-1})$
  - 확률 계산 식: $P(w_n|w_{n-1}) = \frac{Count(w_{n-1}, w_n)}{Count(w_{n-1})}$



▪ 한계점

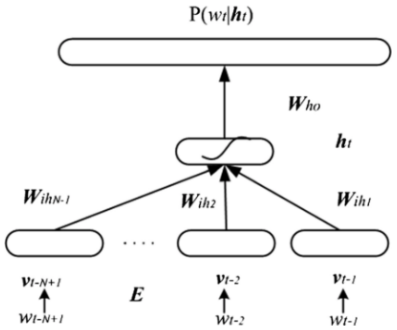
- 희소성 문제 ➔ 말뭉치에 한번도 나타나지 않을 수 있음
- 문맥 길이의 한계 ➔ 고려할 수 있는 길이에 한계
- 의미 표현의 한계 ➔ 단어의 유사성 고려 한계

https://www.scaler.com/topics/nlp/nlp-textblob/

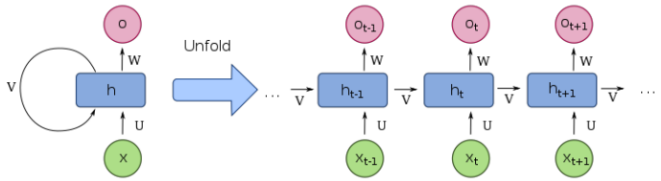
Natural Language Processing (NLP)의 진화: 신경망 언어 모델

▪ 신경망 언어 모델

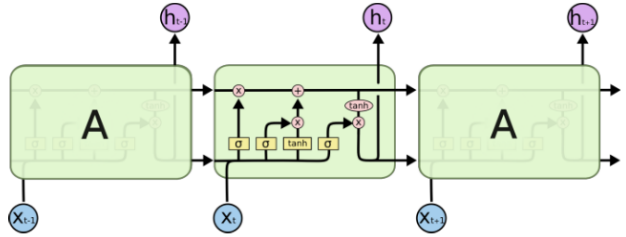
- 단어의 의미를 벡터 공간에 표현(임베딩) 하여 단어간의 유사도를 학습
- 신경망을 통해 단어를 저차원의 **Dense Vector** 로 임베딩
  - 한번도 본 적이 없는 단어 시퀀스라도 유사한 의미를 갖는 단어 시퀀스를 참고하여 예측할 수 있음
- 대표적인 신경망 언어모델: Feed-Forward NNLM, RNN, LSTM



Feed-Forward NNLM



Recurrent Neural Network



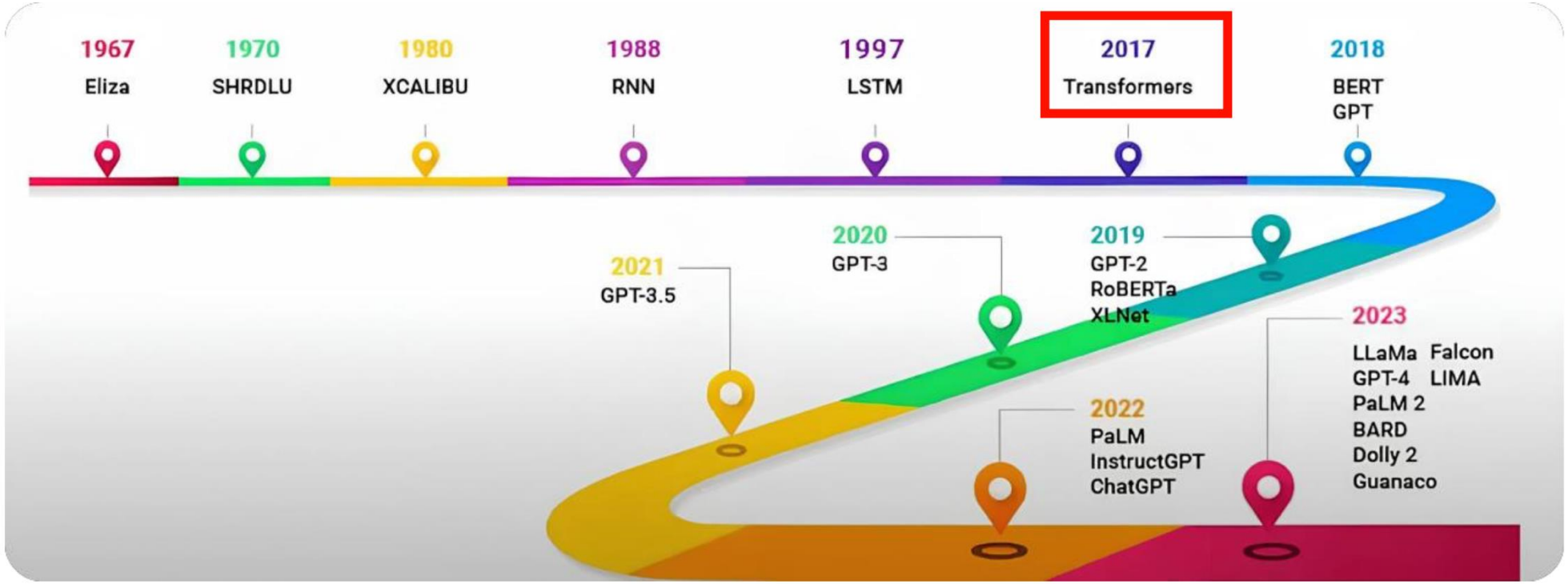
Long Short-Term Memory

▪ 한계점

- 고정 된 문맥 고려 (NNLM) ➔ RNN 으로 극복
- 초기 정보 소실 문제 ➔ LSTM 으로 일부 극복
- 순차적 처리 ➔ 병렬 처리 불가능, 학습 속도 느림

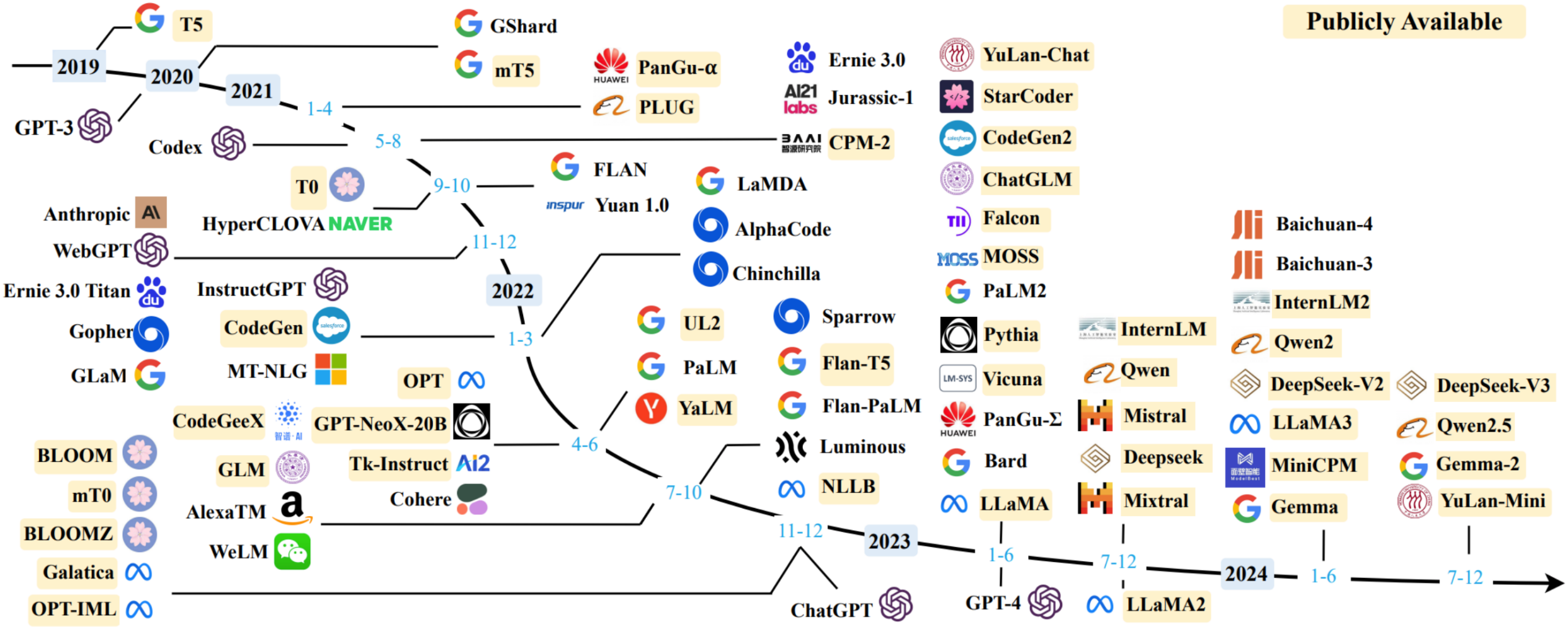
https://www.researchgate.net/figure/Str
forward-neural-network-language-model
https://colah.github.io/posts/2015-08-U

NLP의 발전 흐름: Transformer의 등장



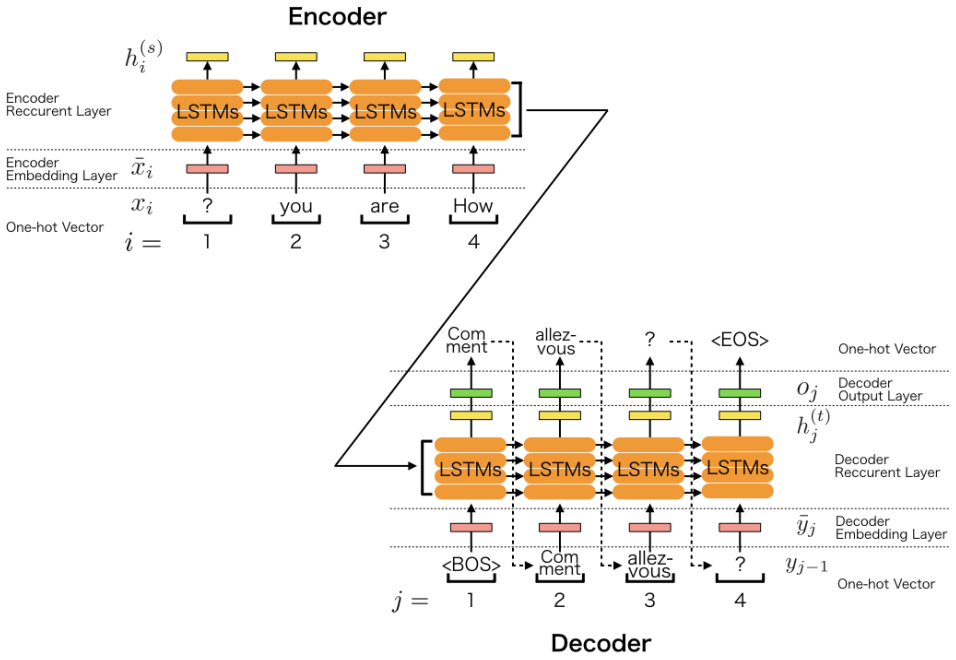
- **2017년 Transformer 공개**
  - 2017년 이후 거의 모든 언어모델은 Transformer 아키텍쳐를 기반으로 설계 됨
  - 대규모 언어모델의 시초
    - 이 전까진 상기한 문제들로 인하여 대규모 파라미터, 대규모 데이터 활용에 한계가 있었음

Transformer 이후 언어모델의 발전흐름

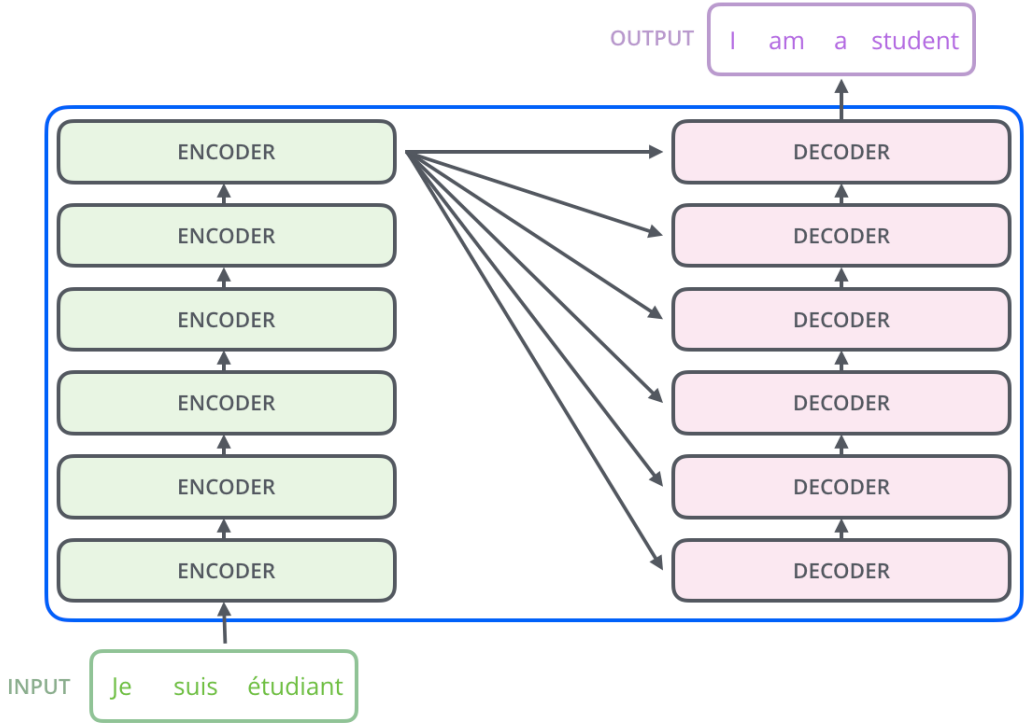


- 이 모든 것들이 다 Transformer 기반 언어모델

Transformer Overview

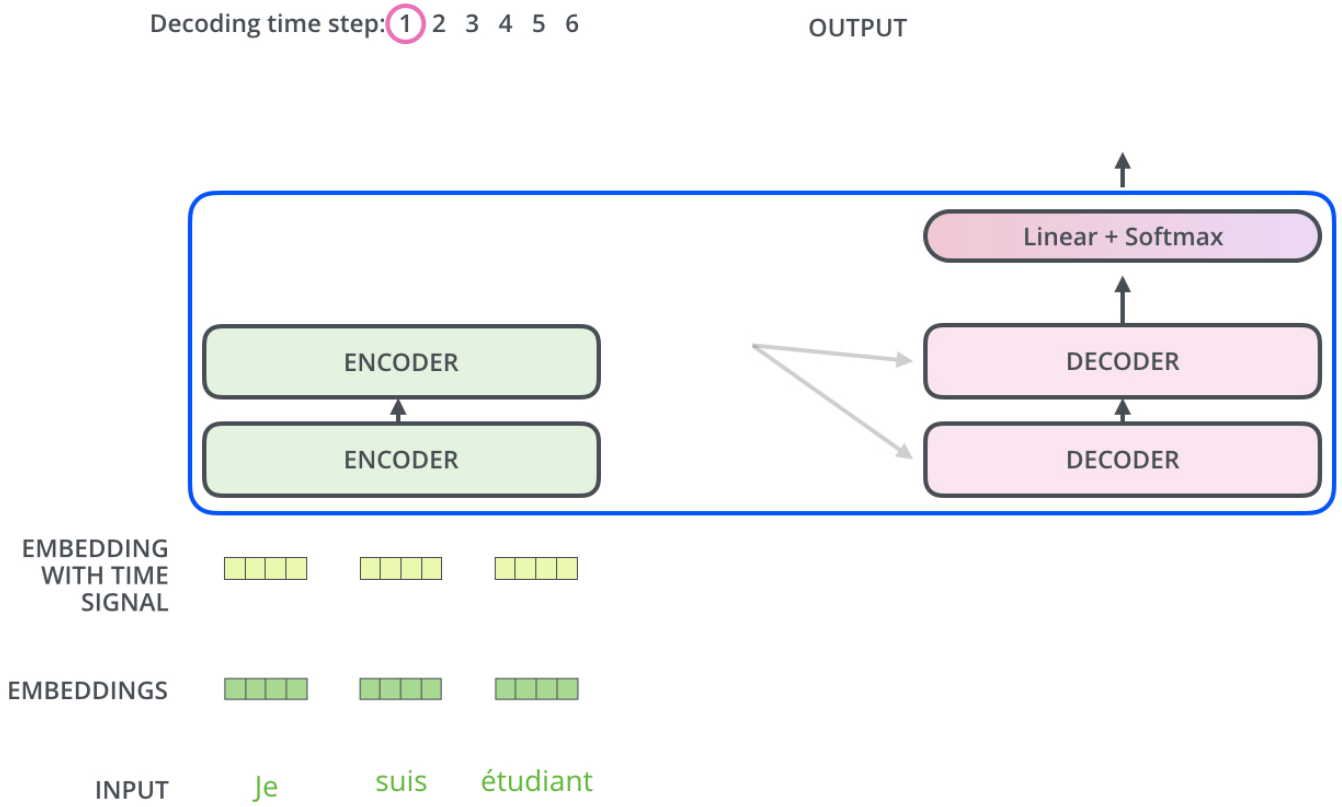


RNN architecture



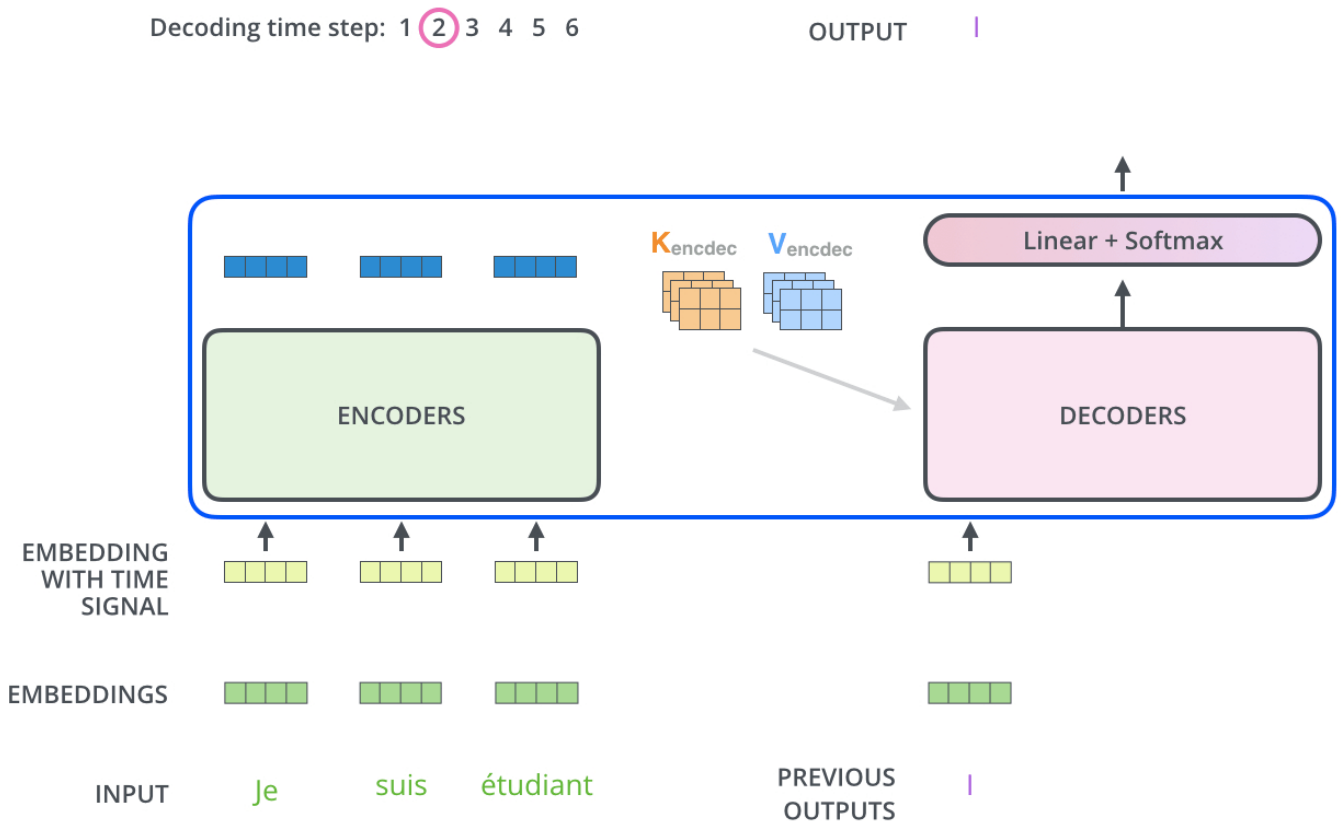
Transformer architecture

Transformer Overview



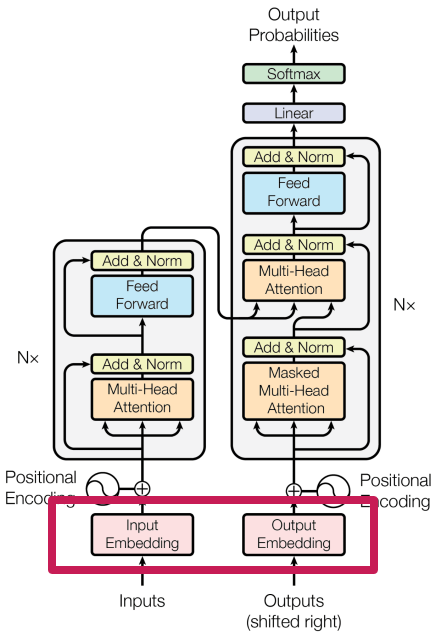
https://jalammar.github.io/illustrated-transformer/

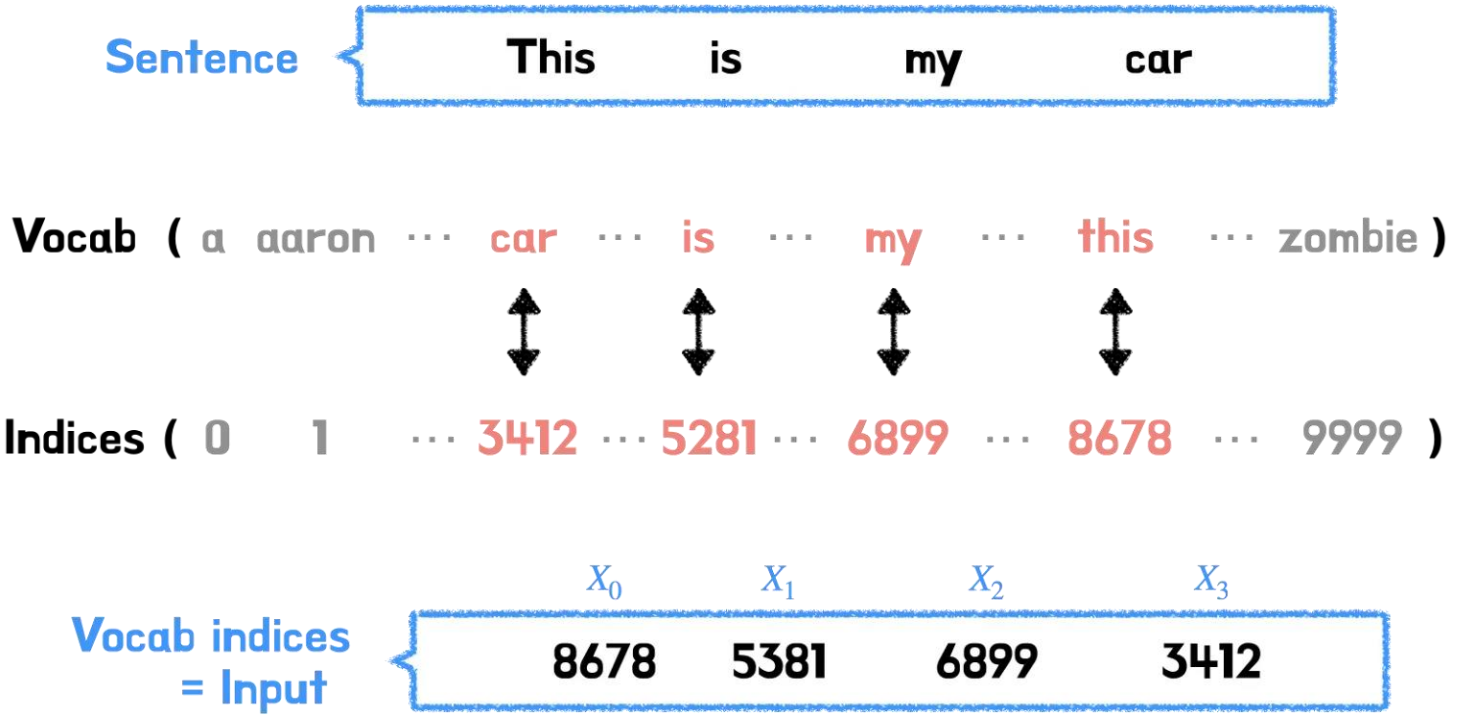
Transformer Overview



https://jalammar.github.io/illustrated-transformer/

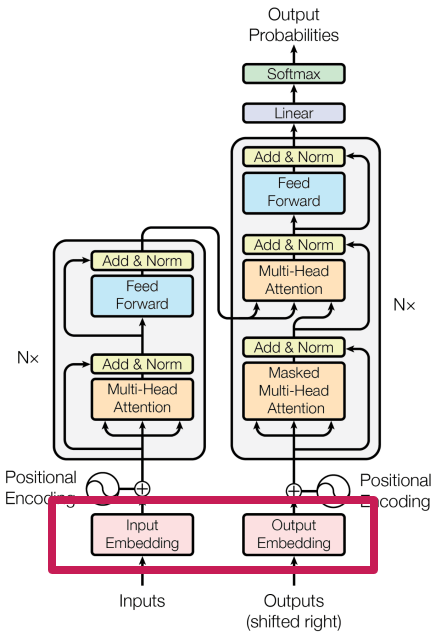
Input Embedding (Word Embedding)

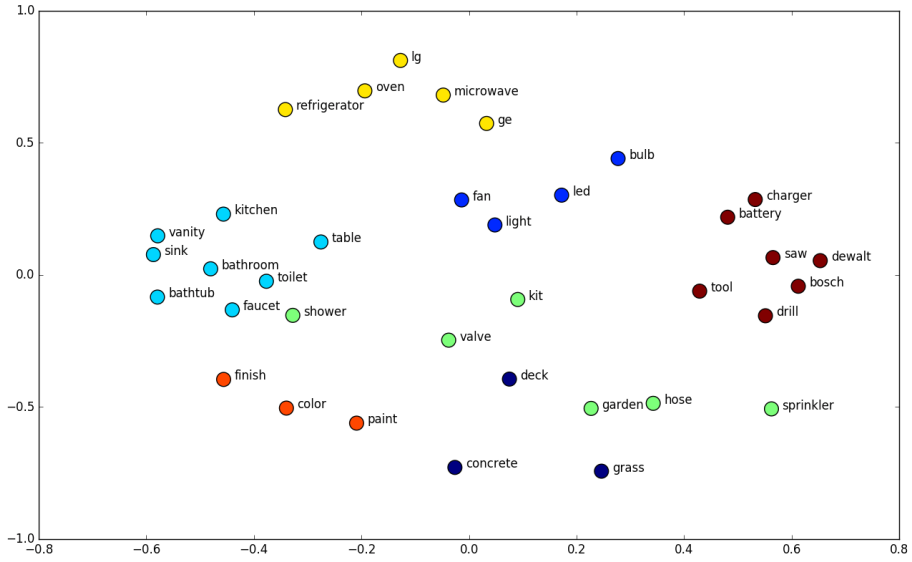




https://www.blossominkyung.com/deeplearning/transfomer-positional-encoding

Input Embedding (Word Embedding)

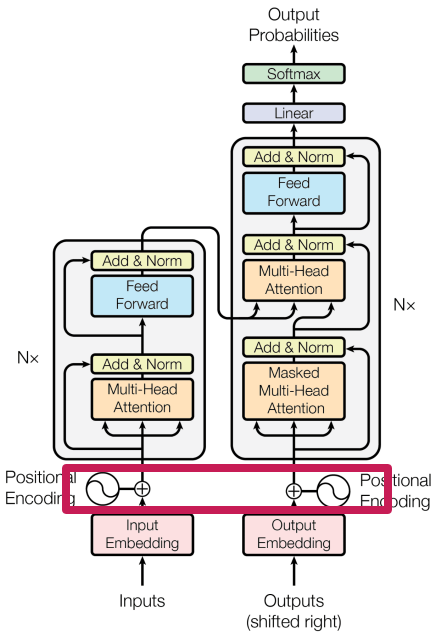


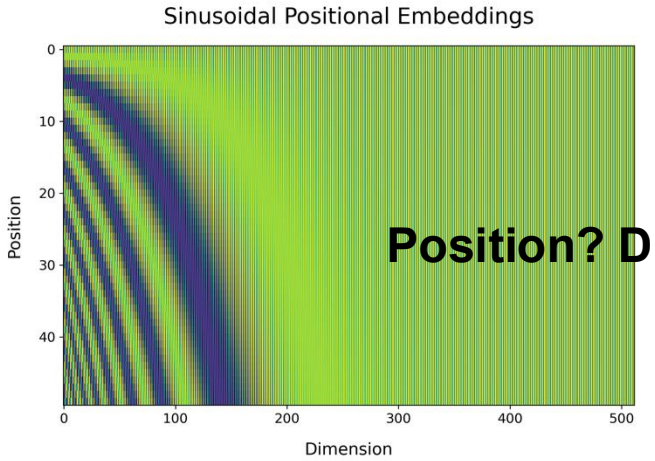


Note: Recent tokenizers are more complex.

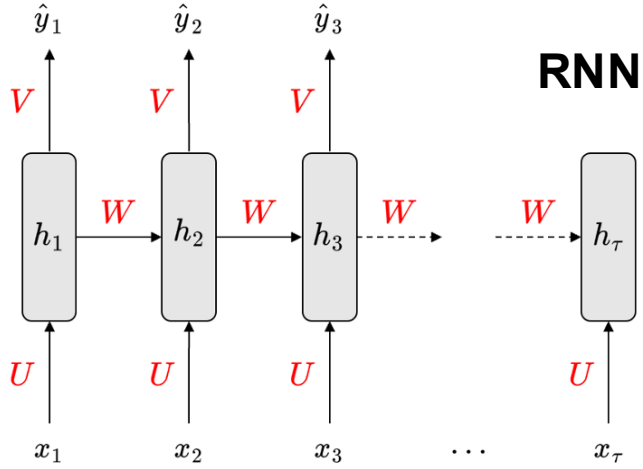
https://www.blossominkyung.com/deeplearning/transfomer-positional-encoding

Positional Encoding



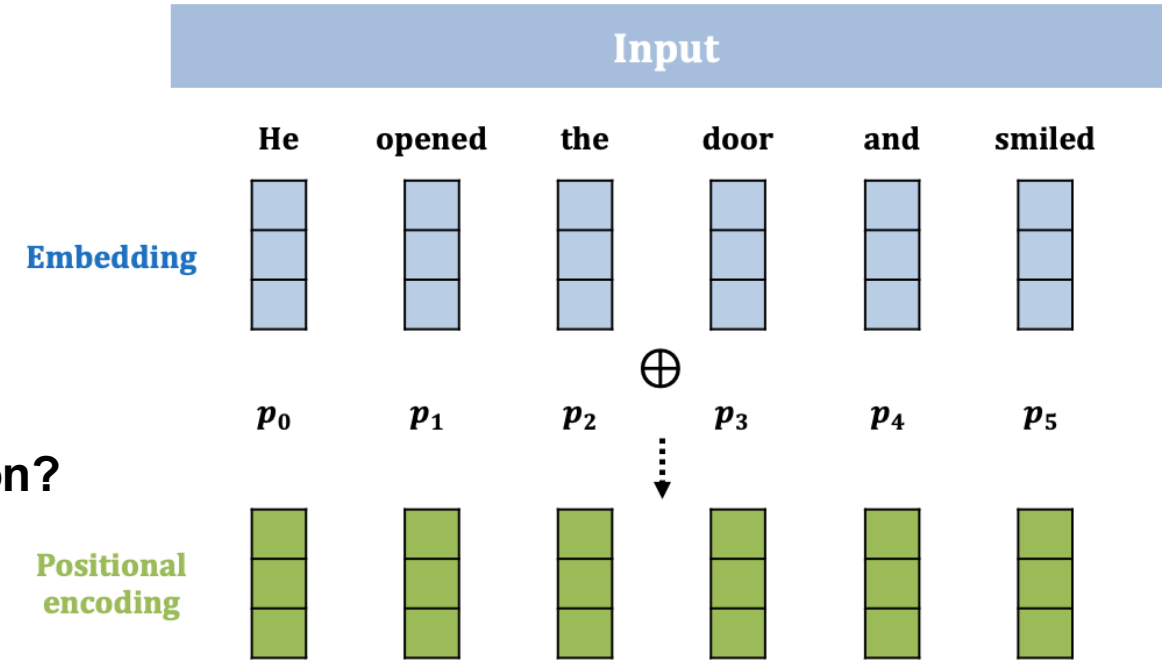


Position? Dimension?

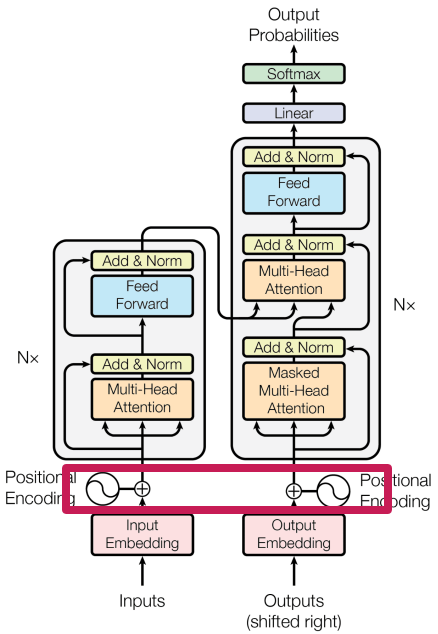


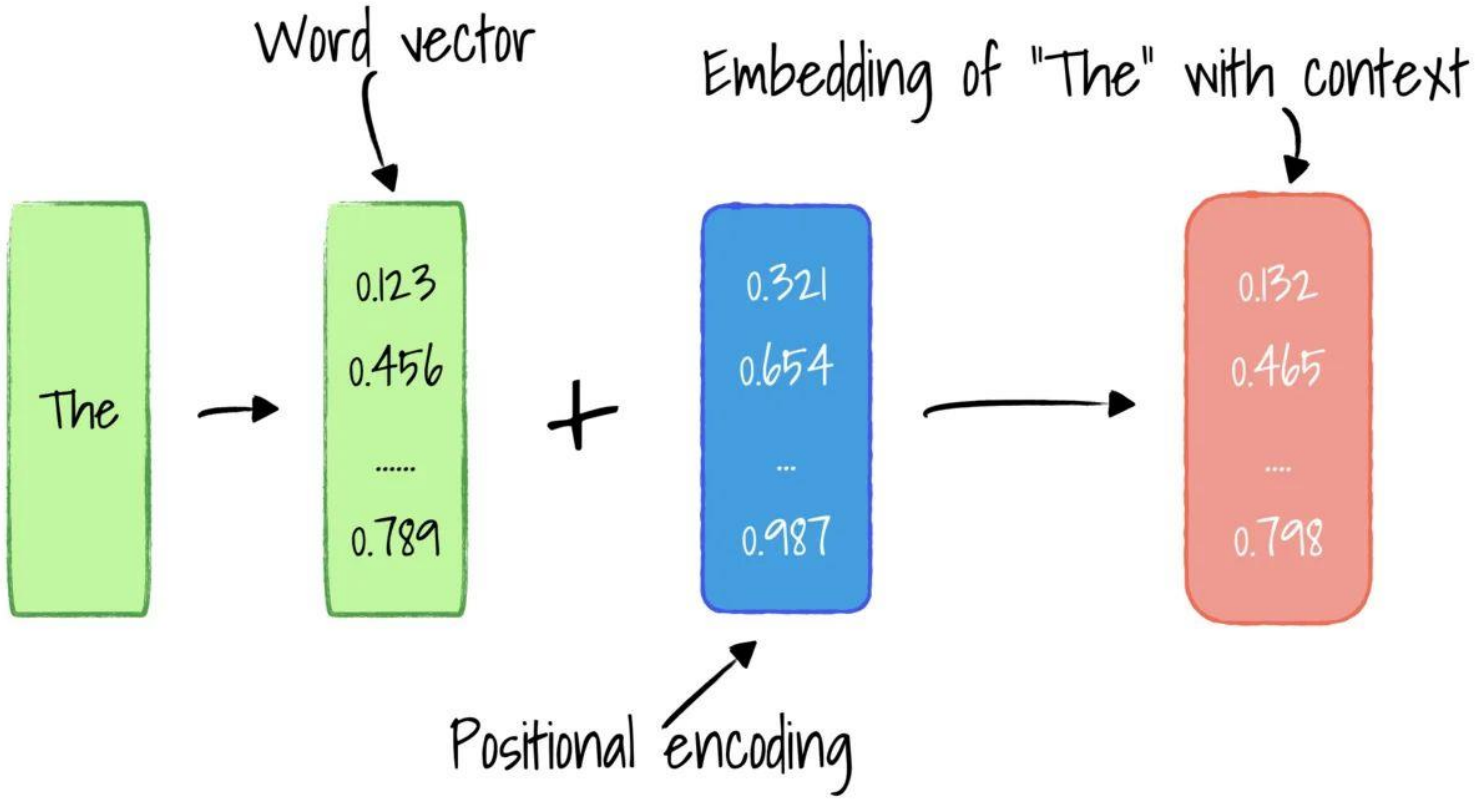
RNN: sequential procedure

Transformer: no-sequential information



Positional Encoding

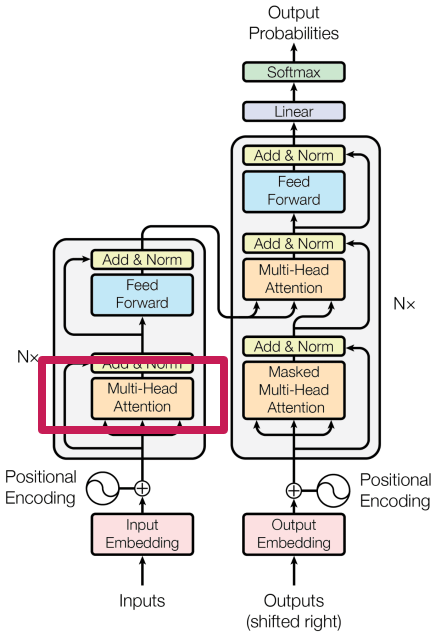


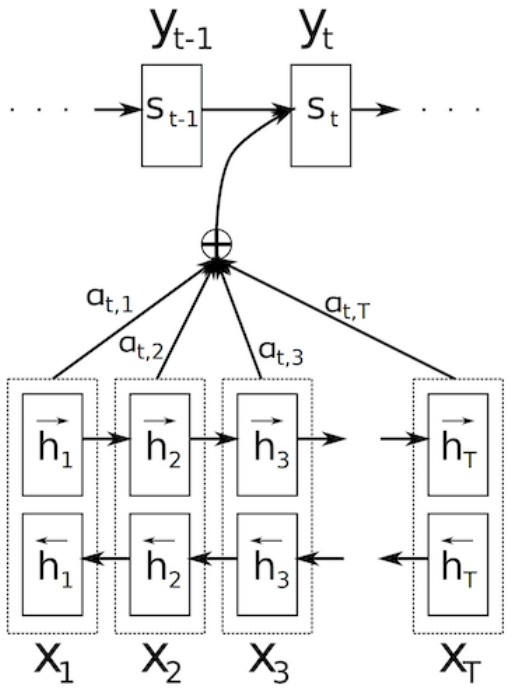


https://jinglescode.github.io/2020/05/27/illustrated-guide-transformer/

Attention Mechanism: Attention is All You Need

- The fundamental concept that underpins a transformer is *attention*.
  - Originally developed as an enhancement to RNNs for machine translation.
  - Later showed significantly improved performance by eliminating the recurrence structure and focusing exclusively on the attention mechanism.





Bahdanau, Cho, and Bengio (2014)

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

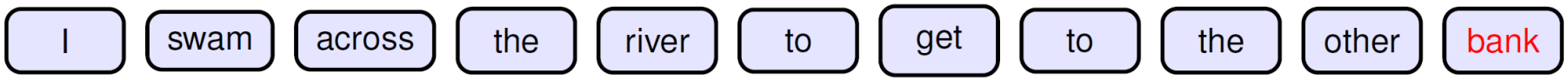
Vaswani *et al.* (2017)

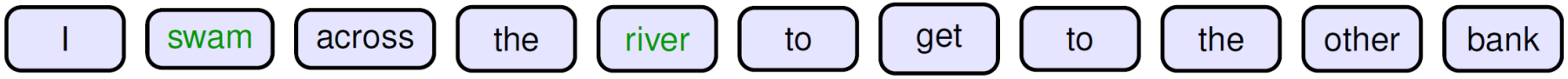
Two Different Sentences

*“I swam across the river to get to the other **bank**.”*

*“I walked across the road to get cash from the **bank**.”*

Key Concepts in Attention





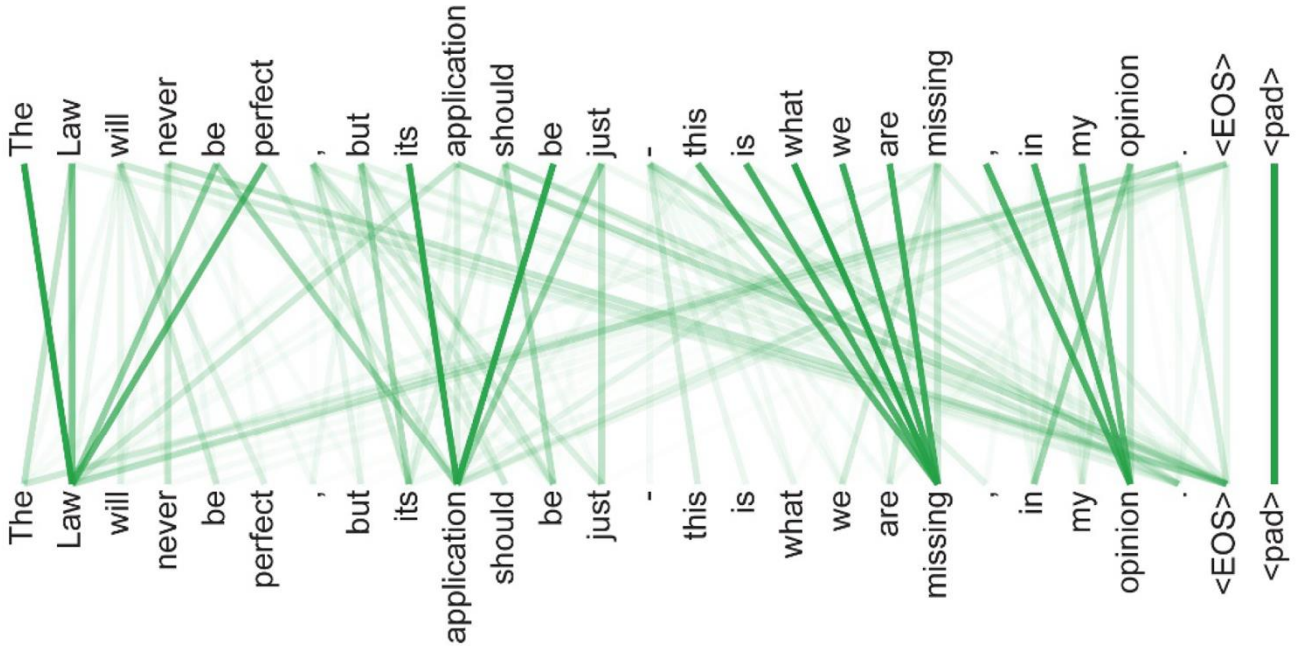
Two Different Sentences

- Particular locations that should receive more attention depend on the input sequence itself

*"I swam across the river to get to the other **bank**."*

*"I walked across the road to get cash from the **bank**."*

  - Once the network is trained, the weights for their associated inputs are generally fixed.
  - By contrast, attention uses weighting factors whose values depend on the specific input data.



Attention Coefficients

- Suppose that we want to map input vectors $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ to another embedding space $(\mathbf{y}_1, \ldots, \mathbf{y}_N)$.
- The value of $\mathbf{y}_n$ should depend stronger for particularly important vector, e.g. $\mathbf{x}_m$.
- A simple way to achieve this is to define each output vector $\mathbf{y}_n$ to be a linear combination of the input vectors with weighting coefficients a_{nm}.

$$\mathbf{y}_n = \sum_{m=1}^{N} a_{nm}\mathbf{x}_m$$

$$\begin{cases} a_{nm} \geqslant 0 \\ \sum_{m=1}^{N} a_{nm} = 1 \end{cases}$$

Institution Behind Self-Attention

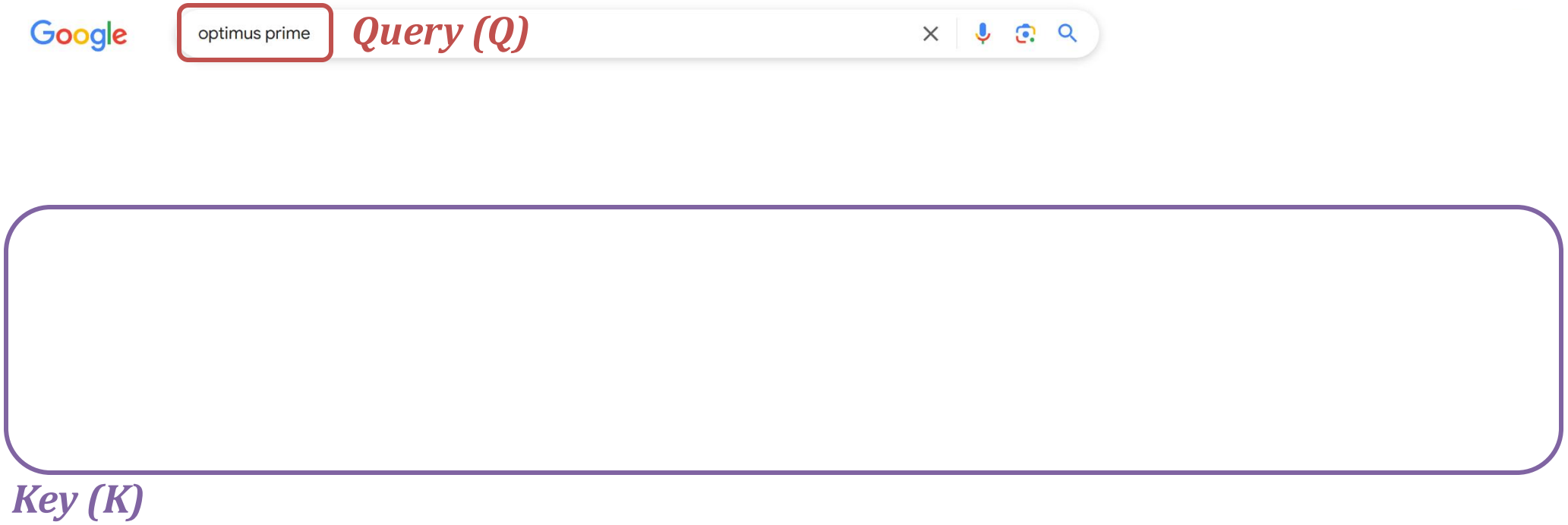
- Let's attend to the most important parts of an input

Institution Behind Self-Attention

- Our brain behaves like
  1. Identify which parts to attend to
  2. Extract the features with high attention

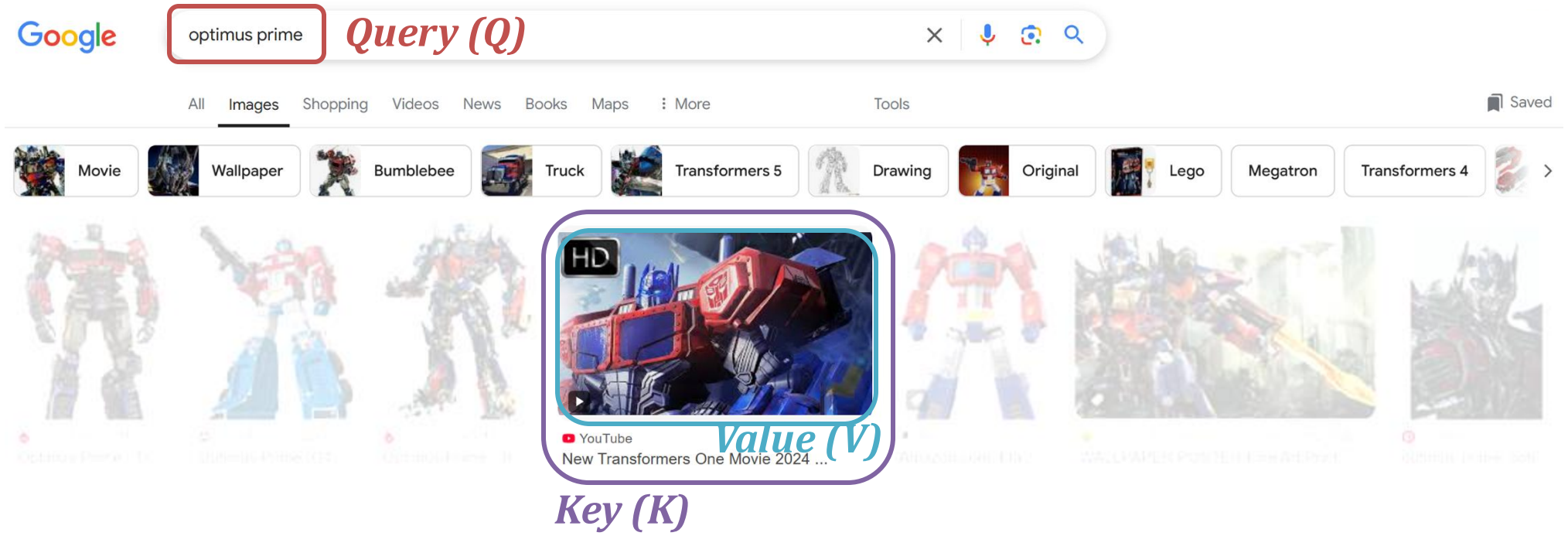
Understanding Self-Attention with Search

- Compute attention masks
  - How similar is each key to the desired query?

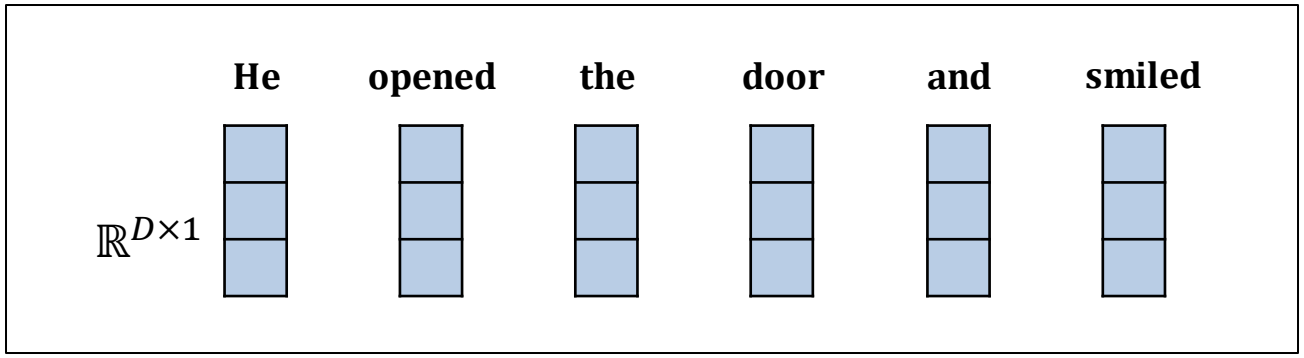


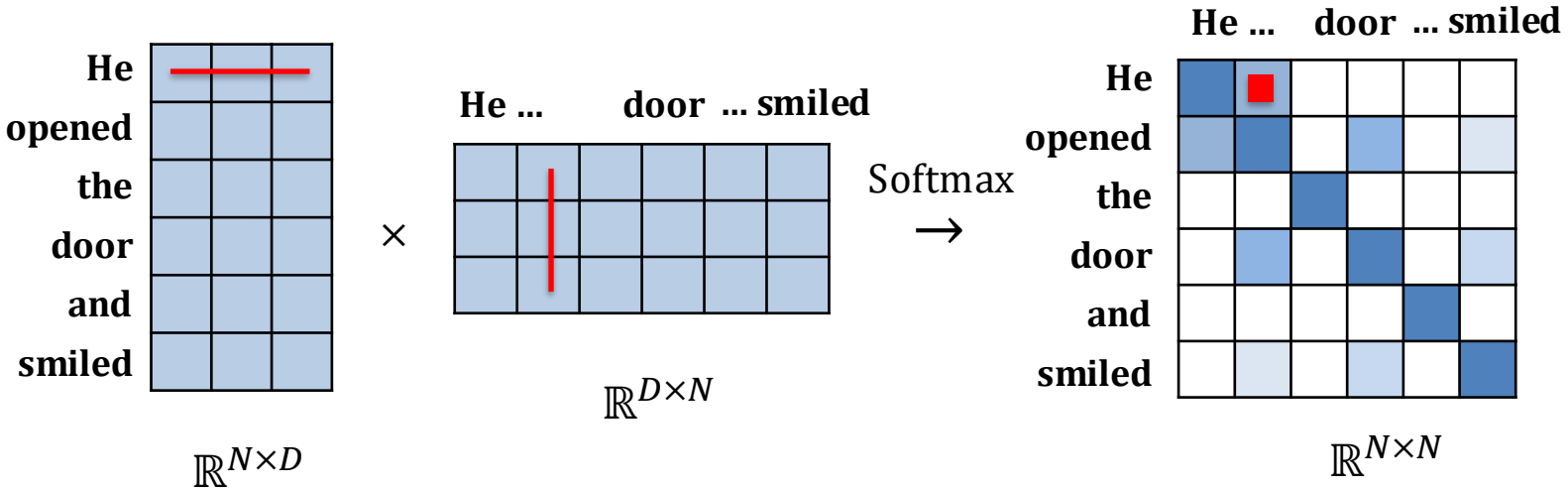
Understanding Self-Attention with Search

- Extract values based on attention
  - Return the values highest attention
  - C.f.) Hard attention vs. Soft attention

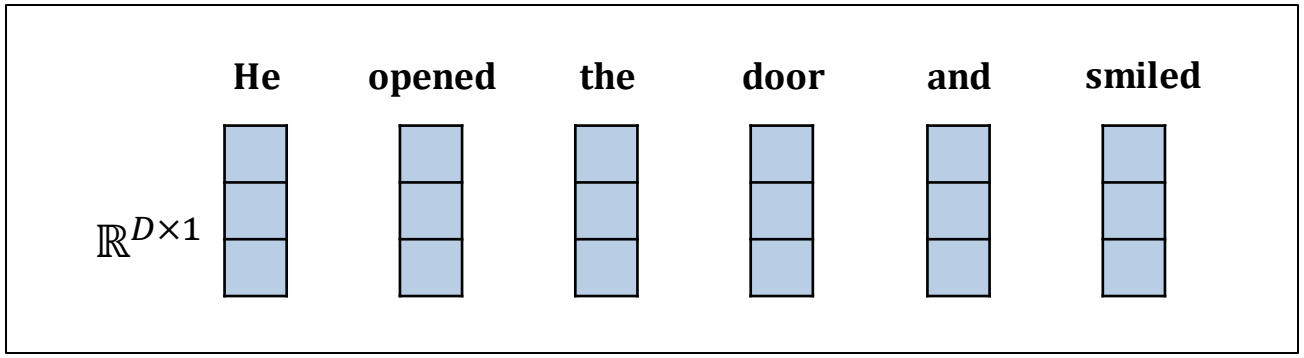


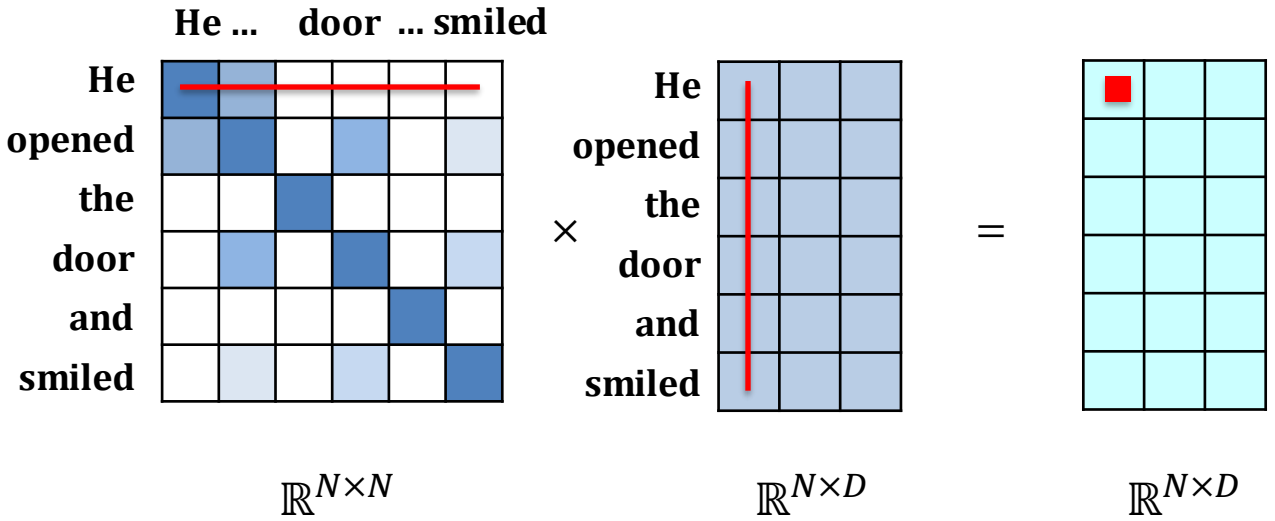
Learning Self-Attention with Neural Networks



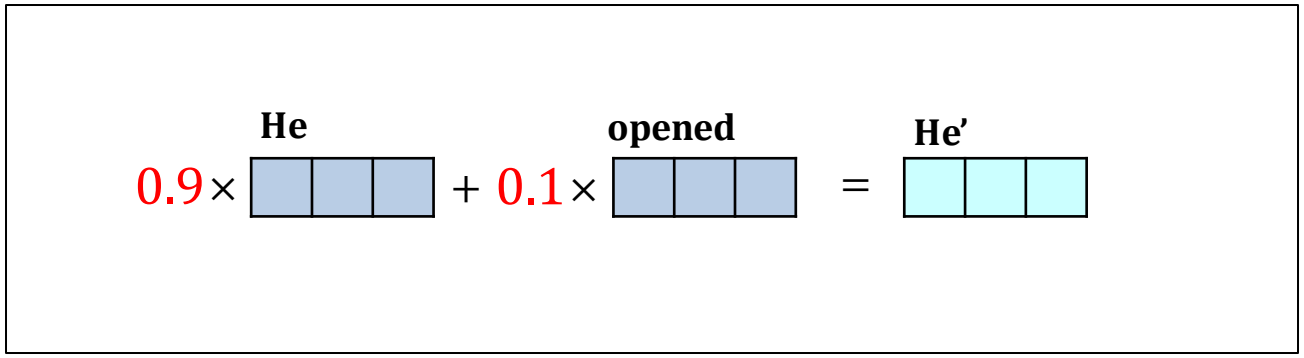


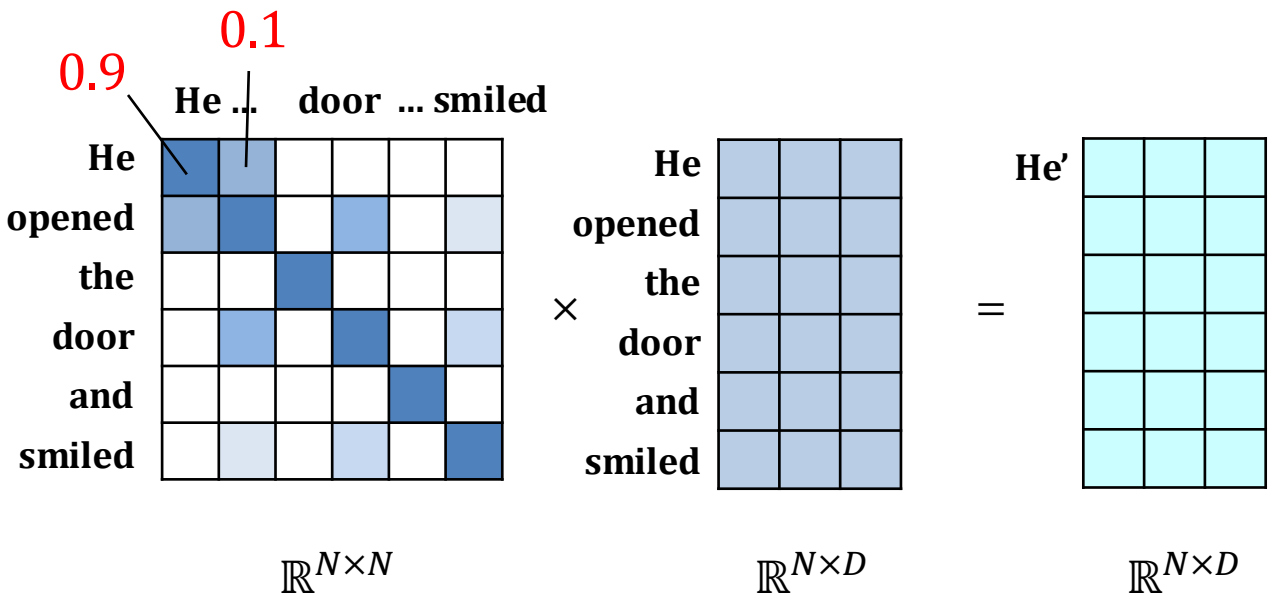
Learning Self-Attention with Neural Networks



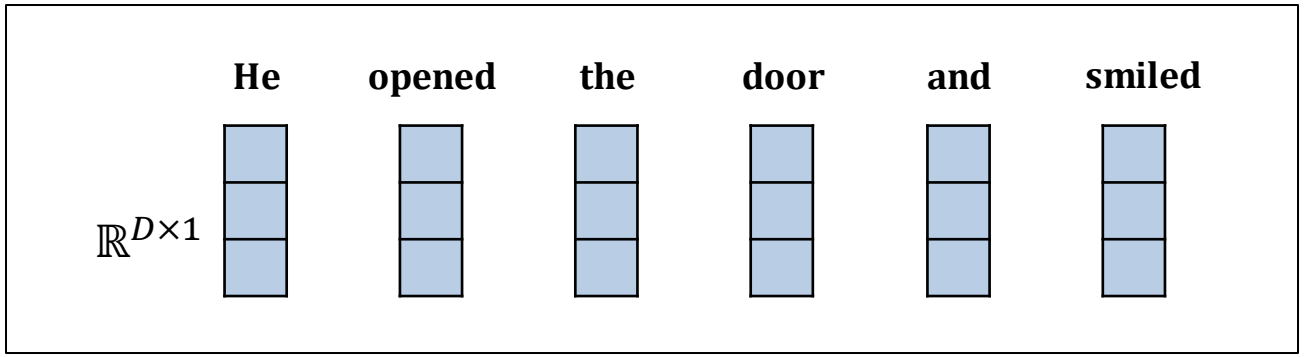


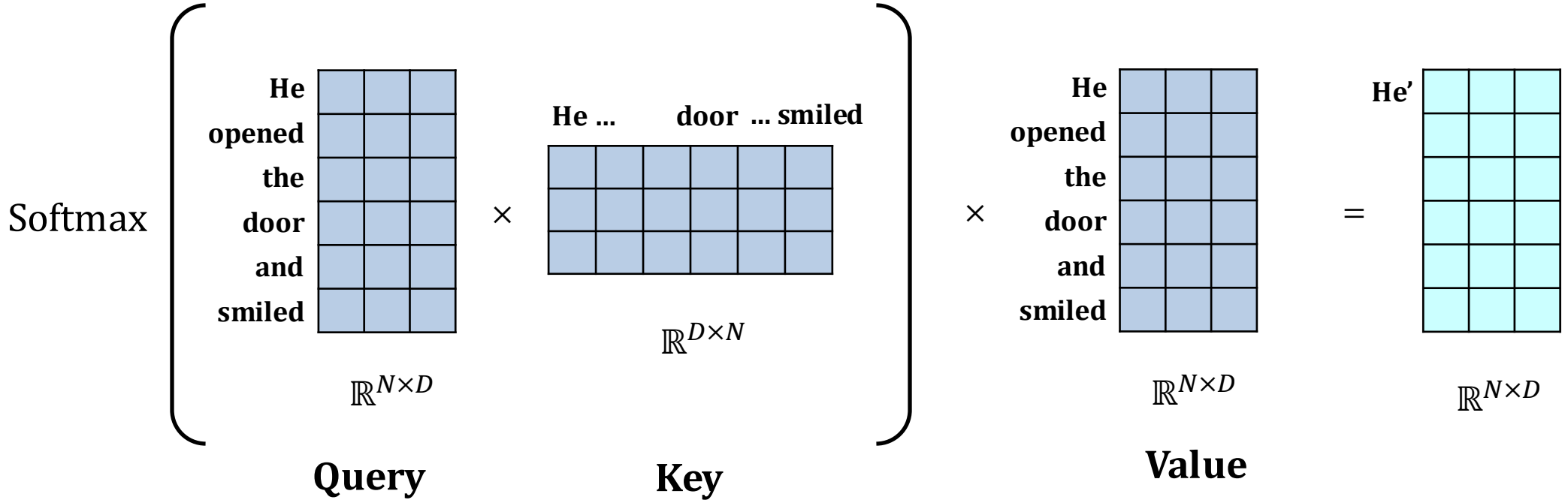
Learning Self-Attention with Neural Networks





Learning Self-Attention with Neural Networks

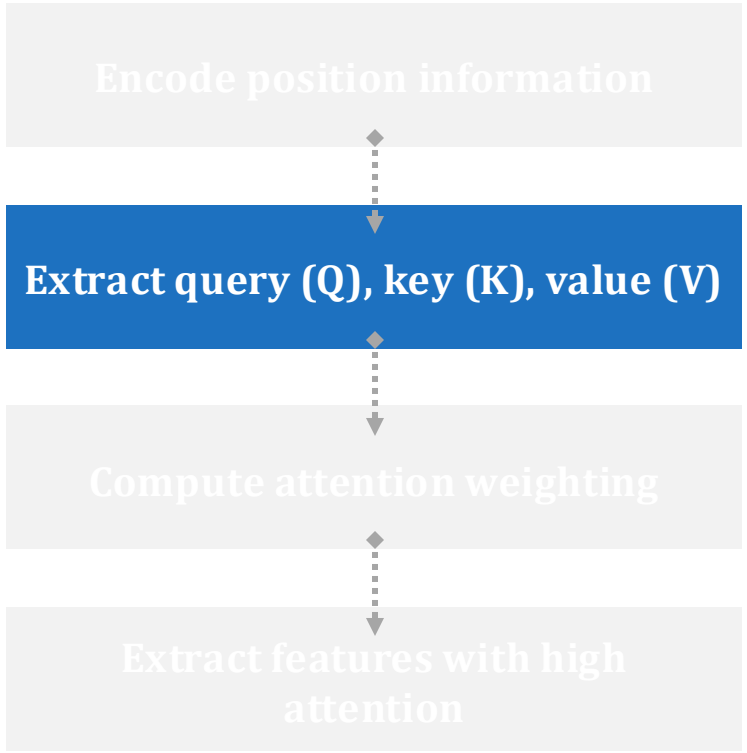


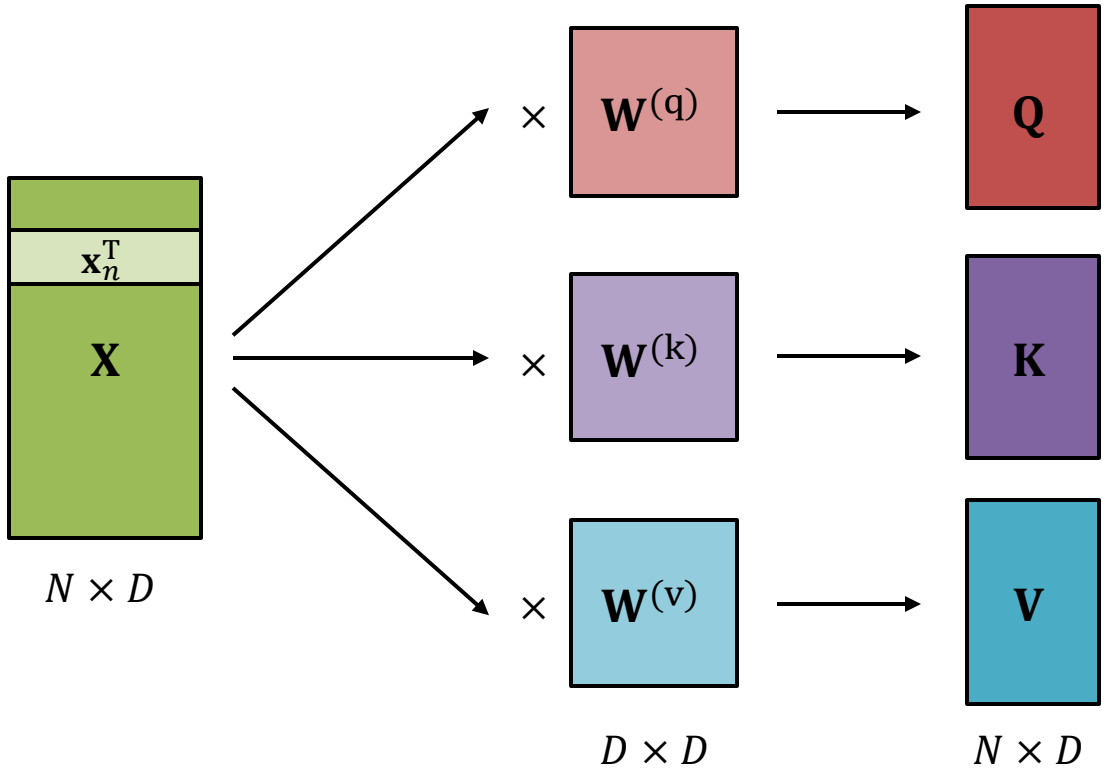


Learning Self-Attention with Neural Networks

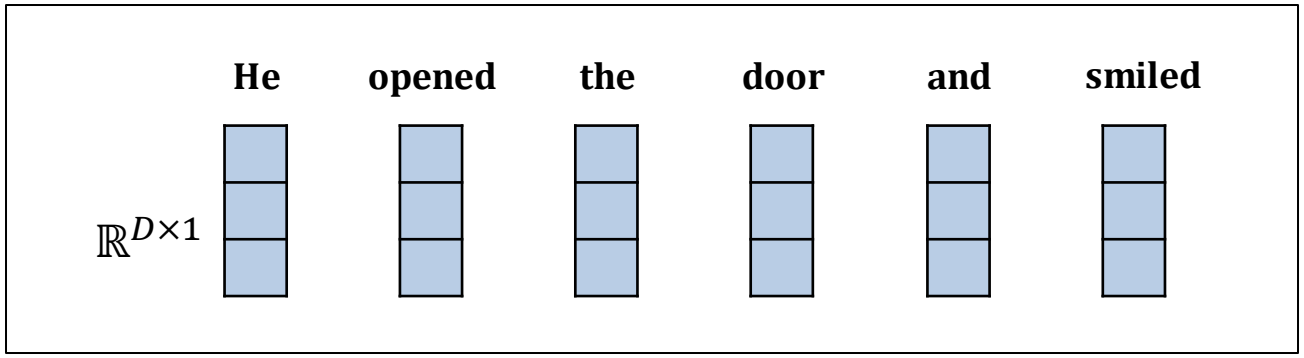
Learning Self-Attention with Neural Networks

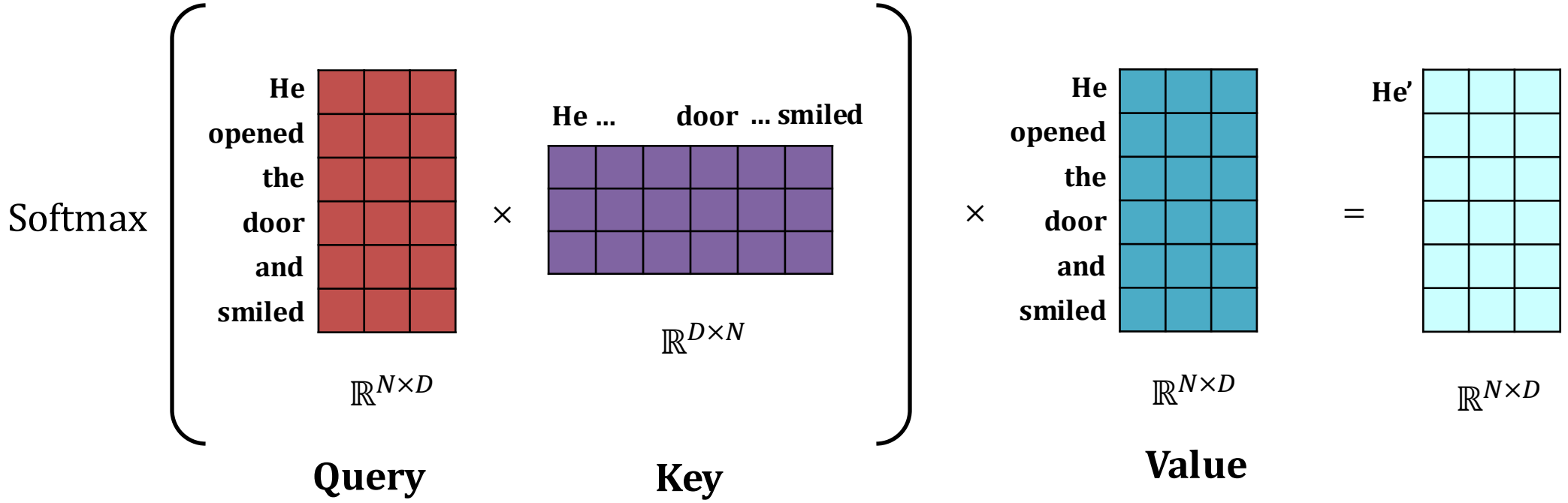
- Goal is to identify and attend to most important features in input data.



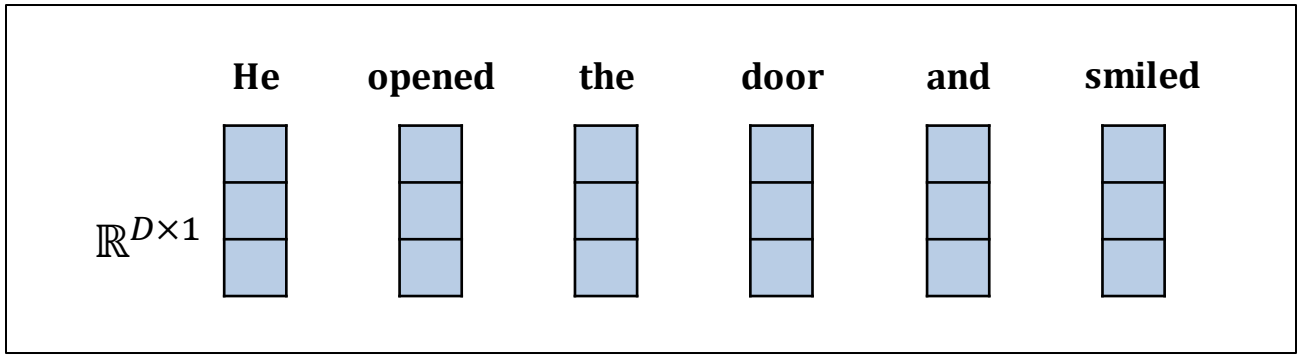


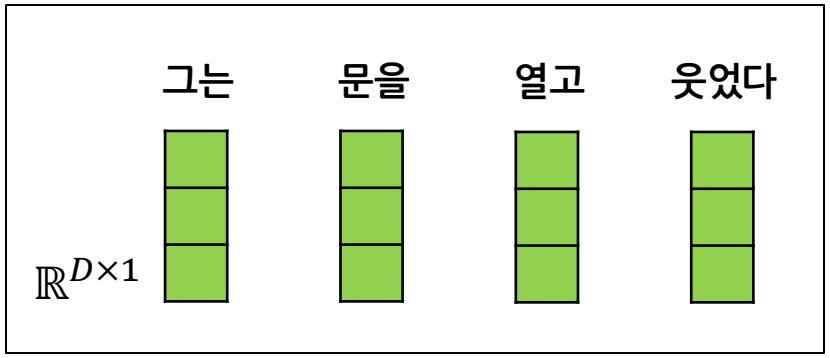
Learning Self-Attention with Neural Networks

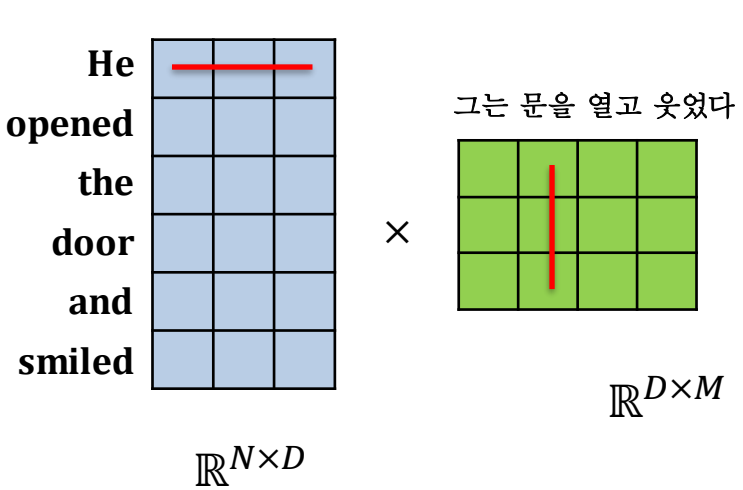




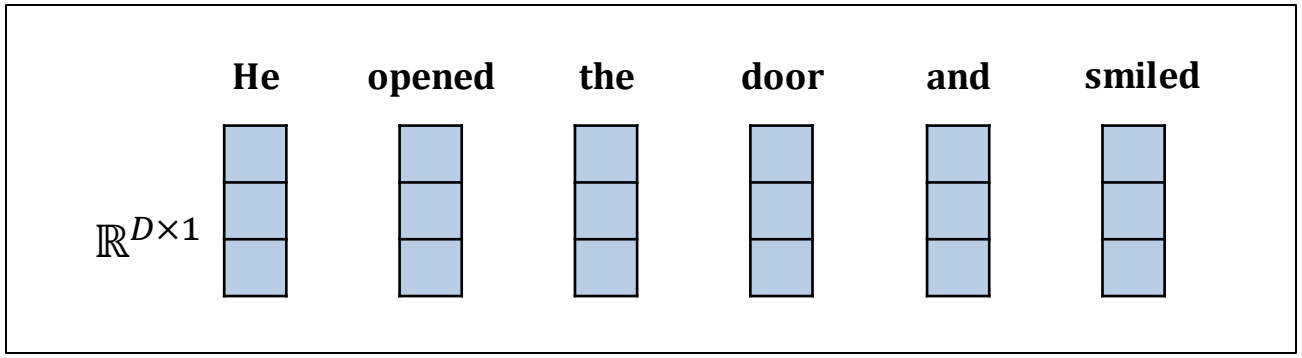
Learning Self-Attention with Neural Networks

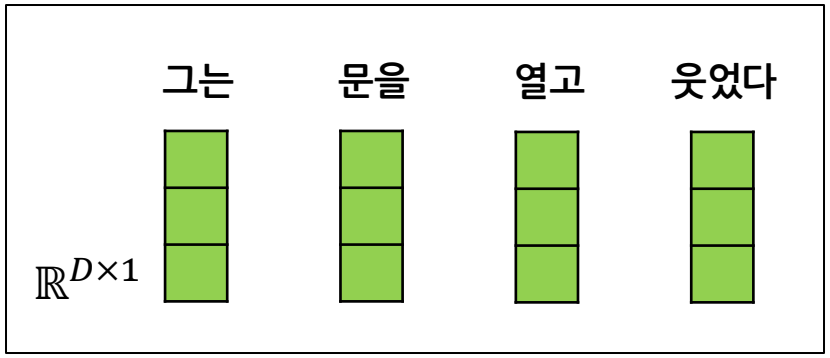




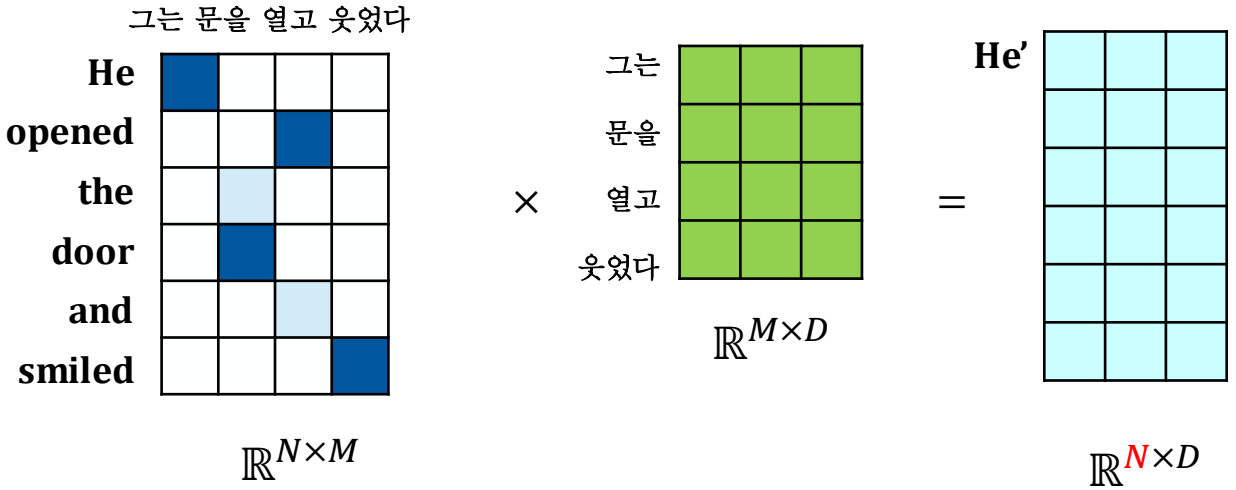


Learning Self-Attention with Neural Networks





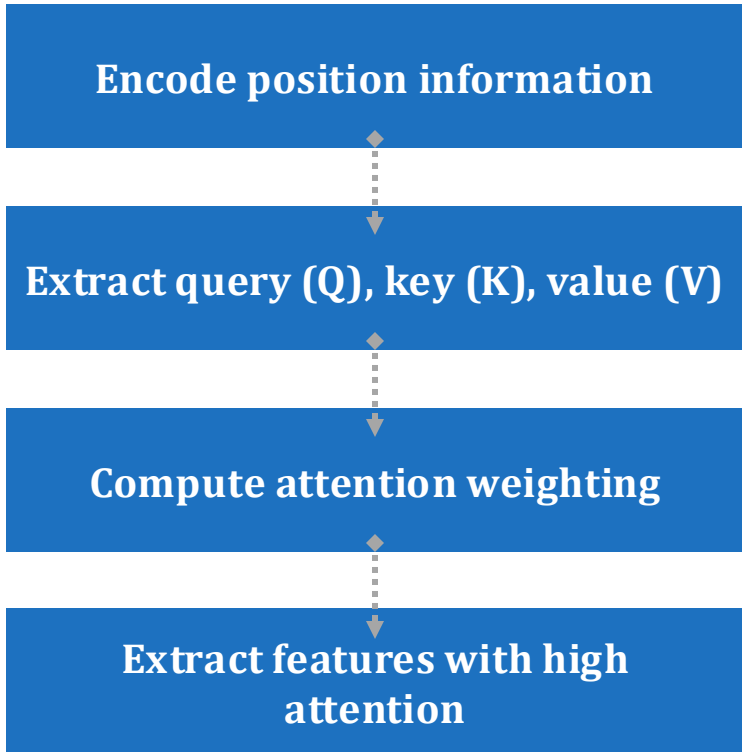
2. Key, Value



1. Query

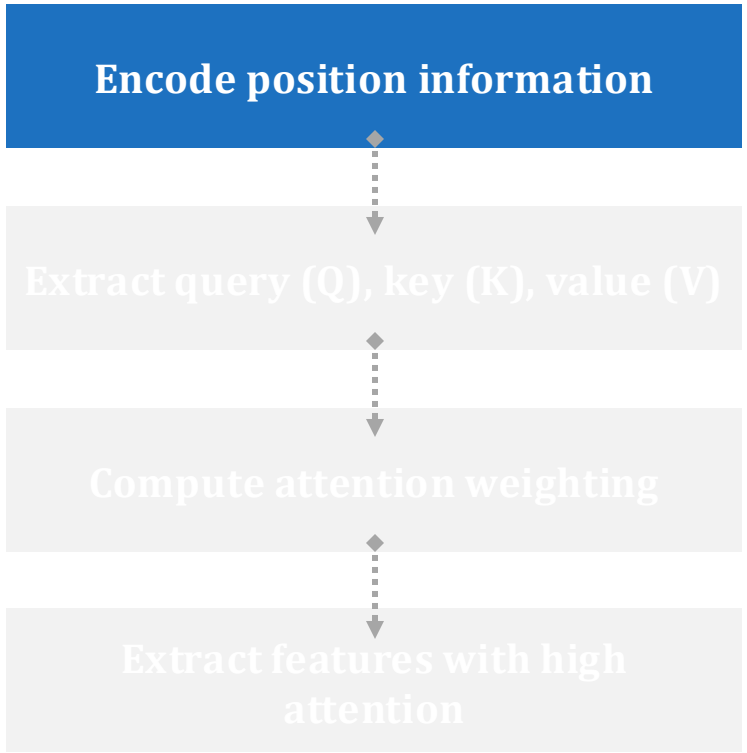
Learning Self-Attention with Neural Networks

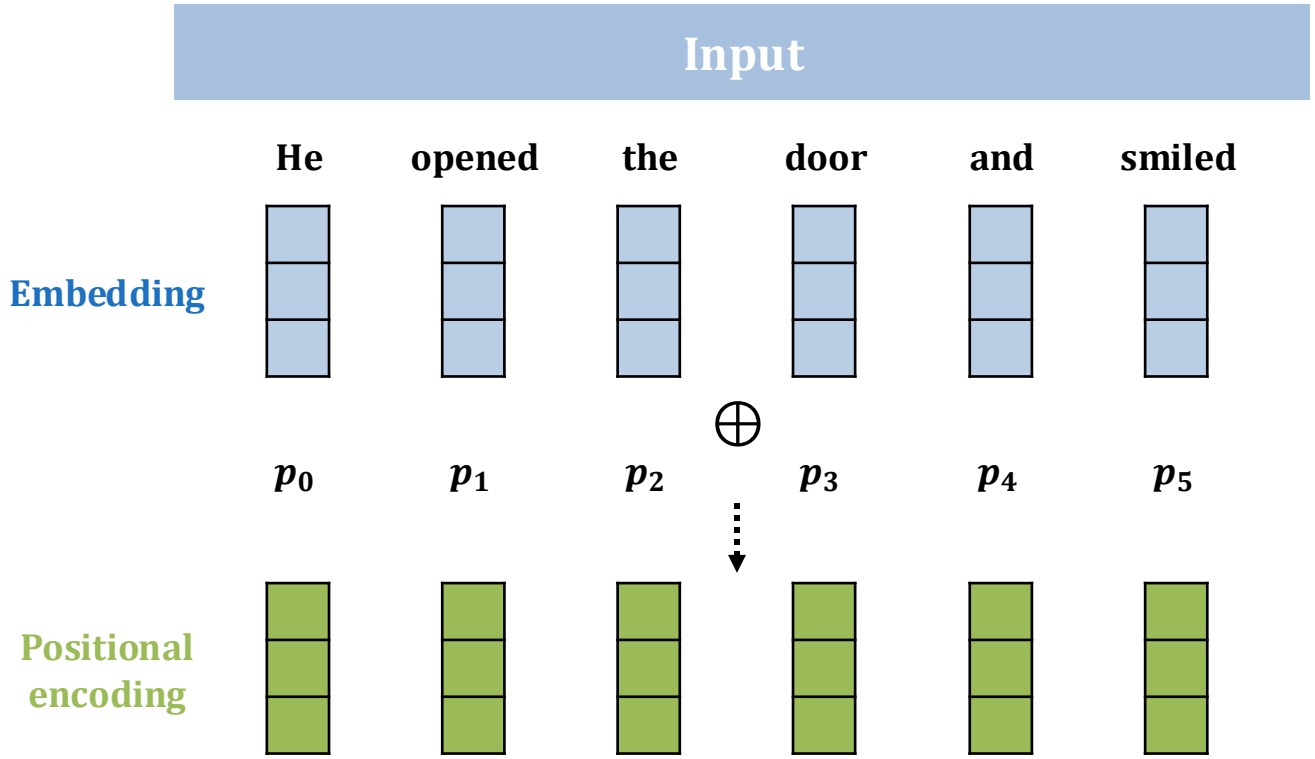
- Goal is to identify and attend to most important features in input data.



Learning Self-Attention with Neural Networks

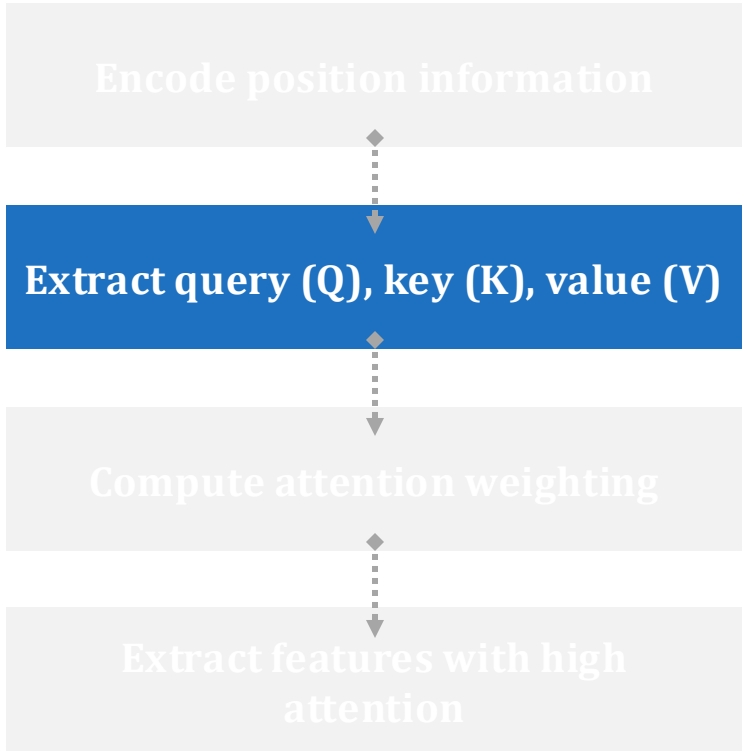
- Goal is to identify and attend to most important features in input data.

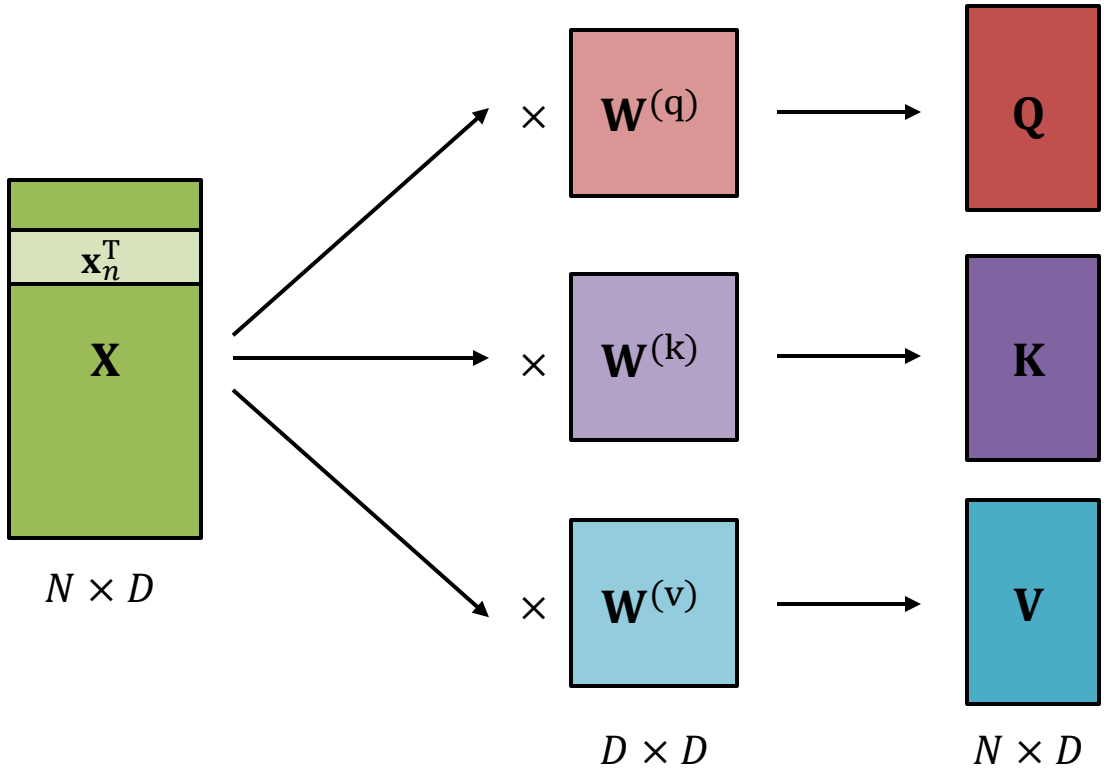




Learning Self-Attention with Neural Networks

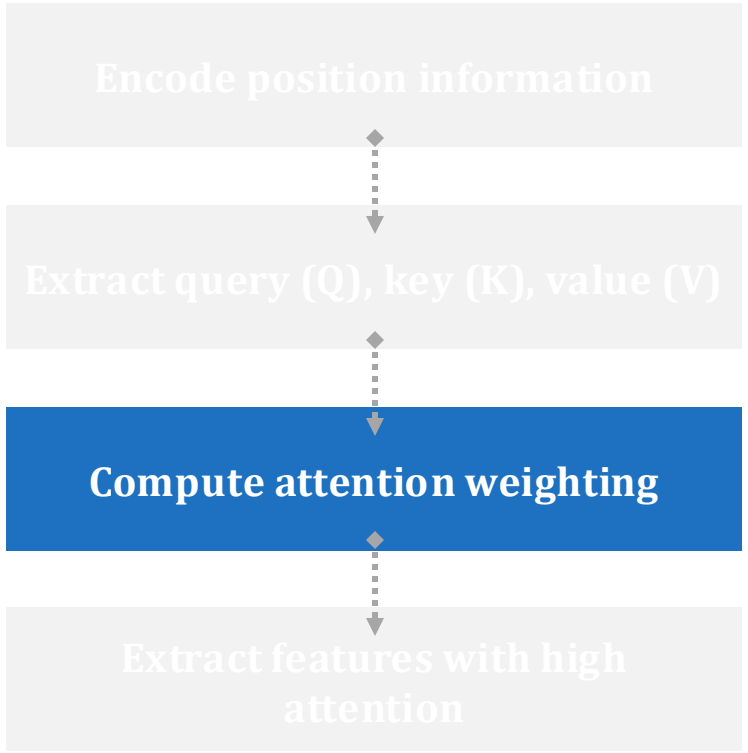
- Goal is to identify and attend to most important features in input data.

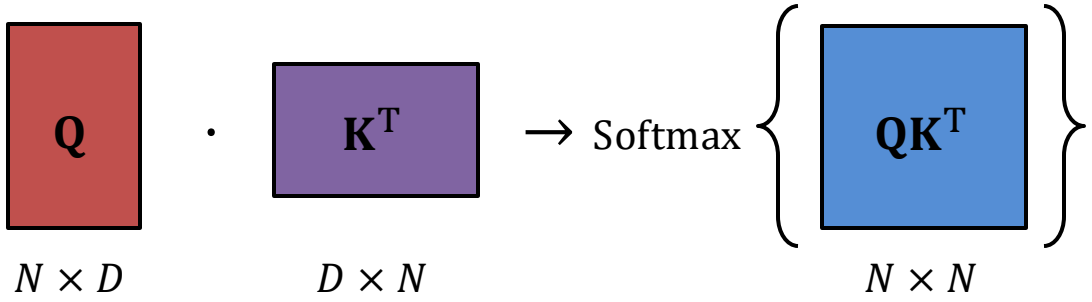




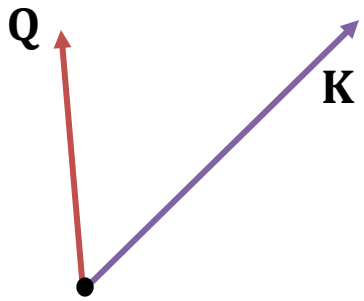
Learning Self-Attention with Neural Networks

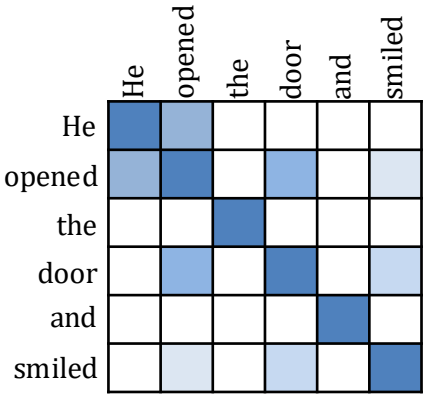
- Goal is to identify and attend to most important features in input data.





Attention score by computing similarity between Q and K





Learning Self-Attention with Neural Networks

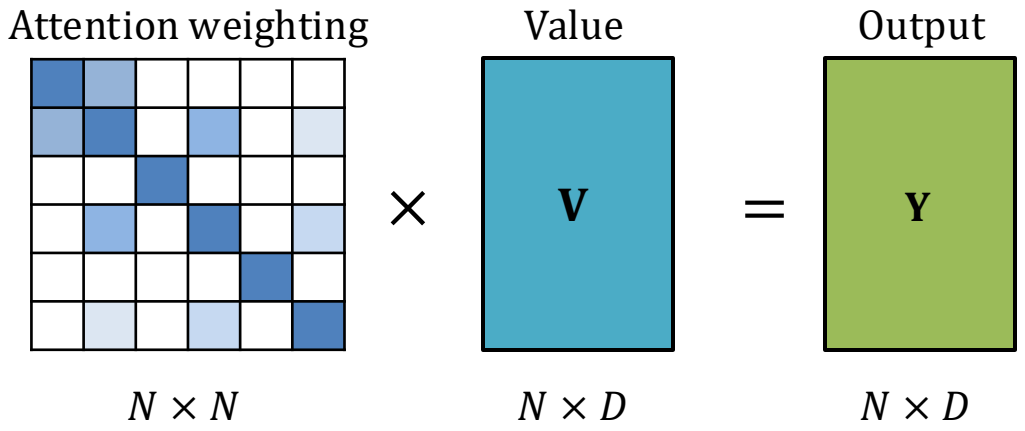
- Goal is to identify and attend to most important features in input data.

Encode position information

Extract query (Q), key (K), value (V)

Compute attention weighting

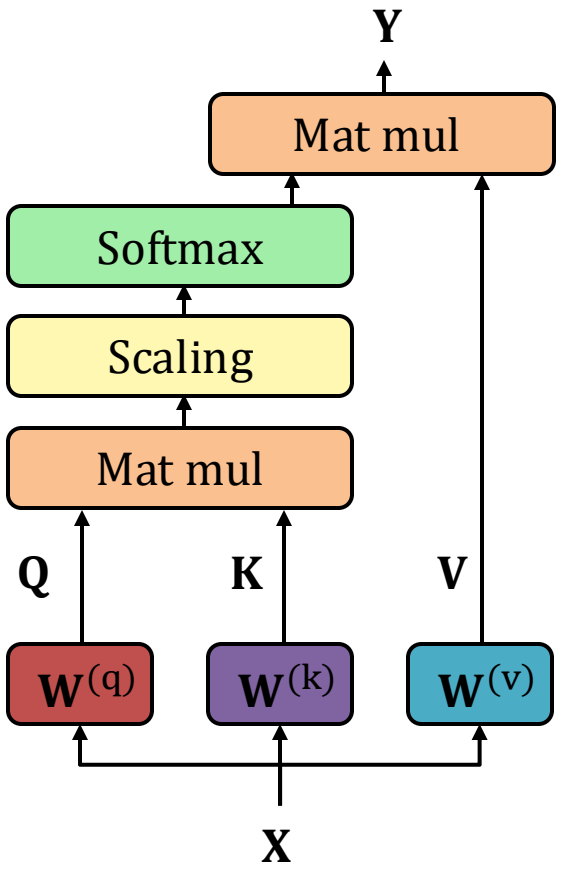
Extract features with high attention



$$\mathbf{Y} = \text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \equiv \text{Softmax}\left[\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{D_{\mathrm{k}}}}\right]\mathbf{V}$$

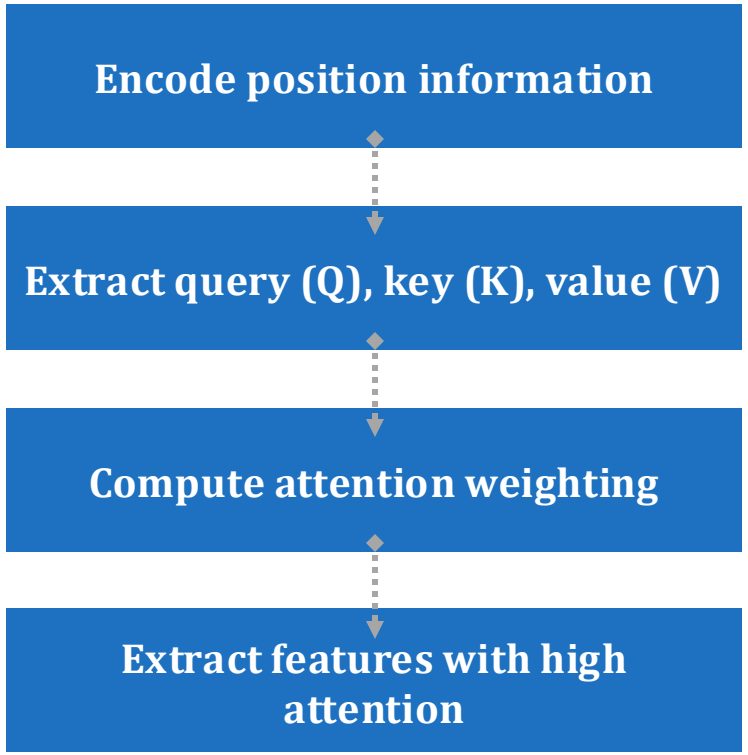
Learning Self-Attention with Neural Networks

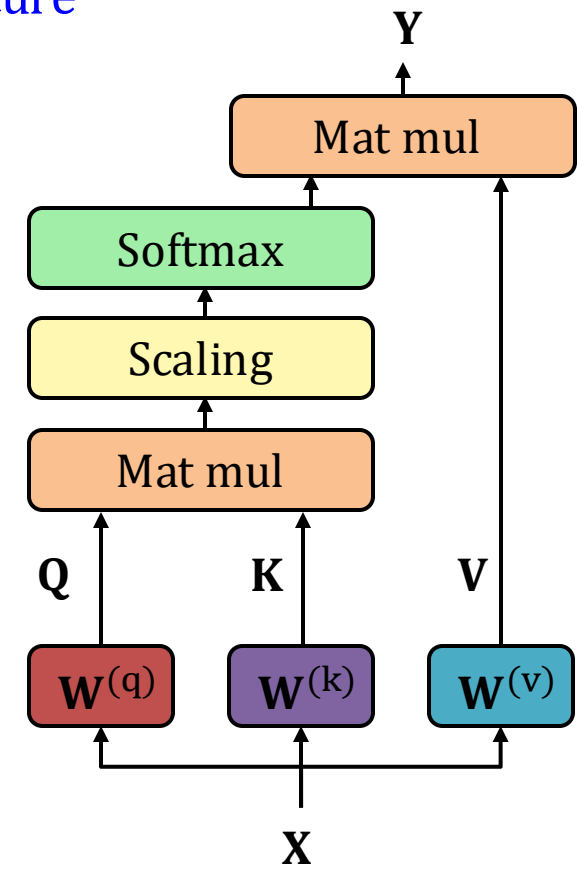
- A *self-attention head* that can plug into a larger network



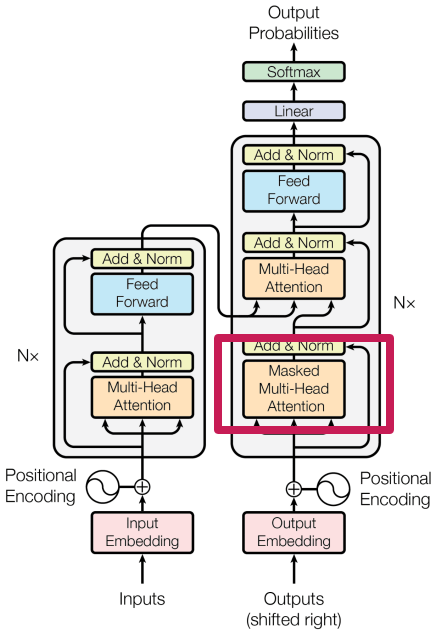
Learning Self-Attention with Neural Networks

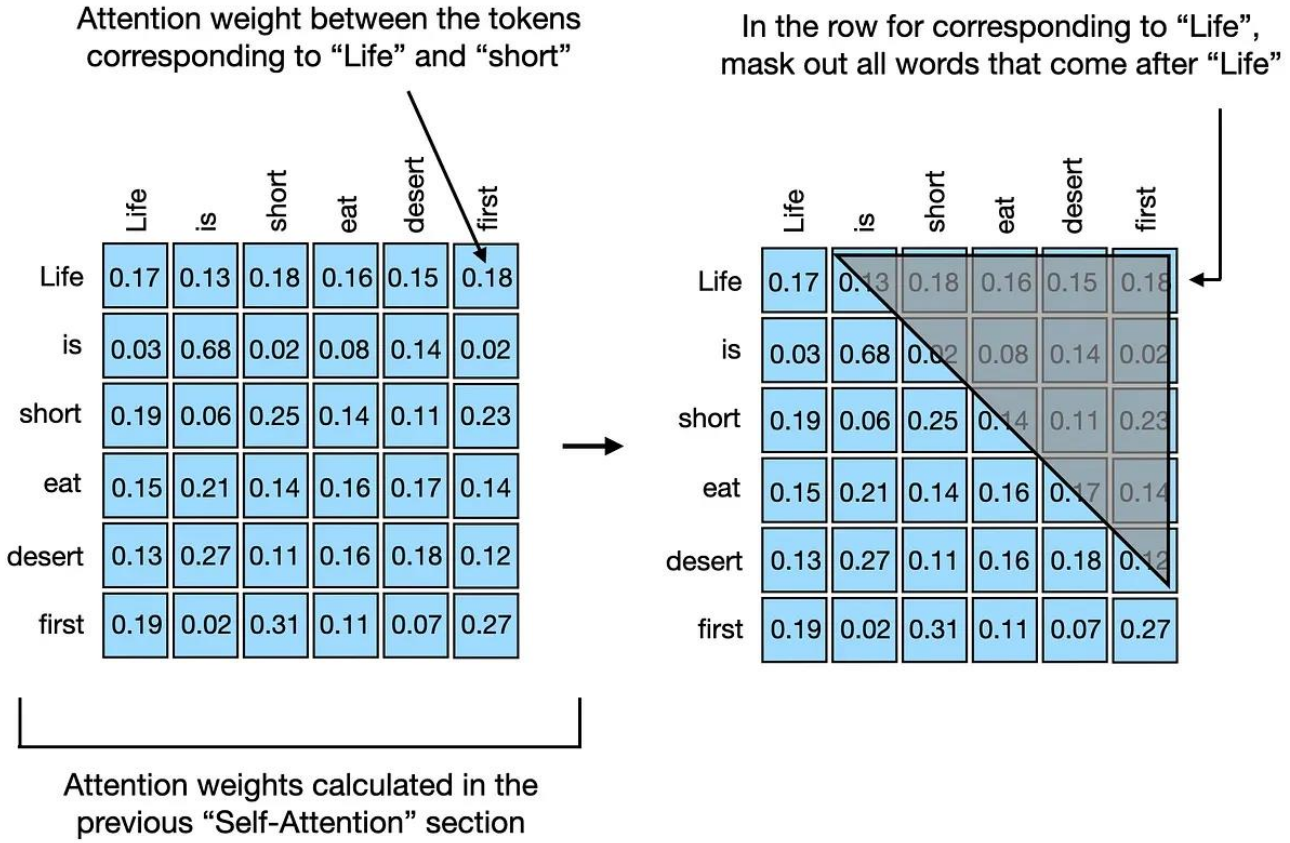
- A *self-attention head* that can plug into a larger network
- Foundational building block of the Transformer architecture





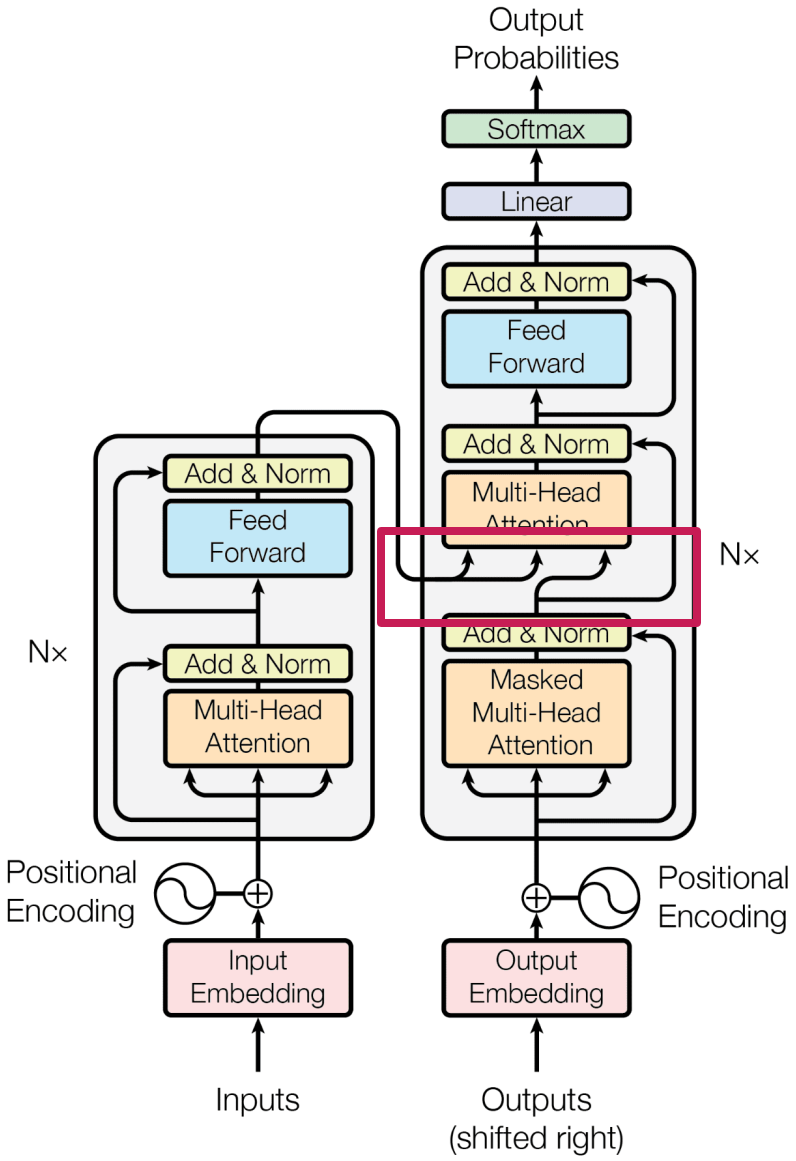
Attention Mechanism: Masked Decoder Self-Attention





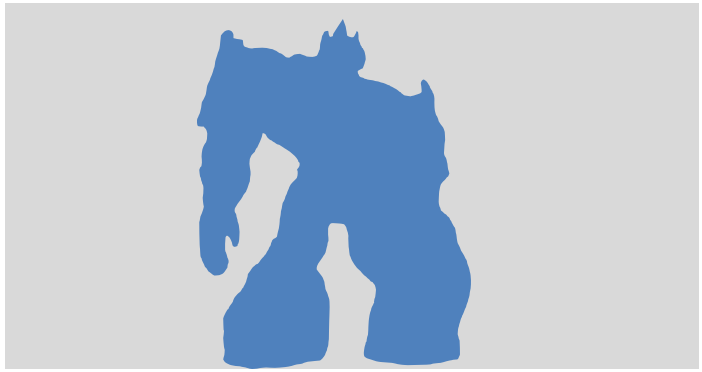
https://www.kaggle.com/code/aisuko/causal-self-attention

Attention Mechanism: Encoder-Decoder Attention



Multi-Head Attention

- To capture multiple patterns of attention that are relevant at the same time



Attention weighting

×

Value

=

Output

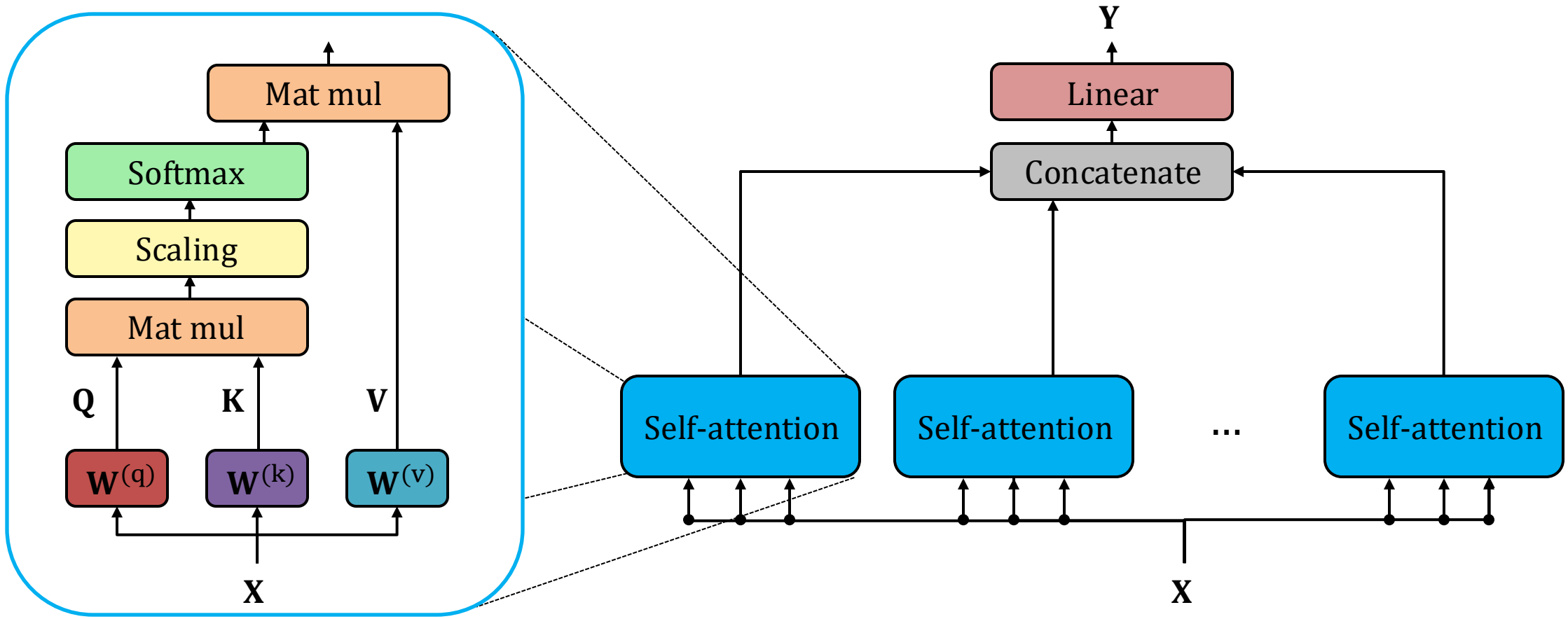
Output of attention head (1)

Output of attention head (2)

Output of attention head (3)

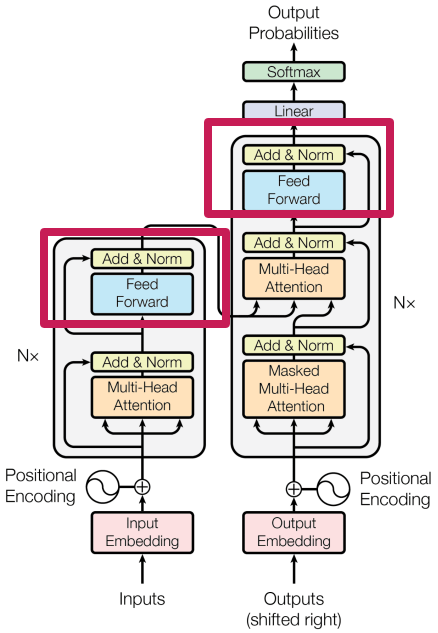
Multi-Head Attention

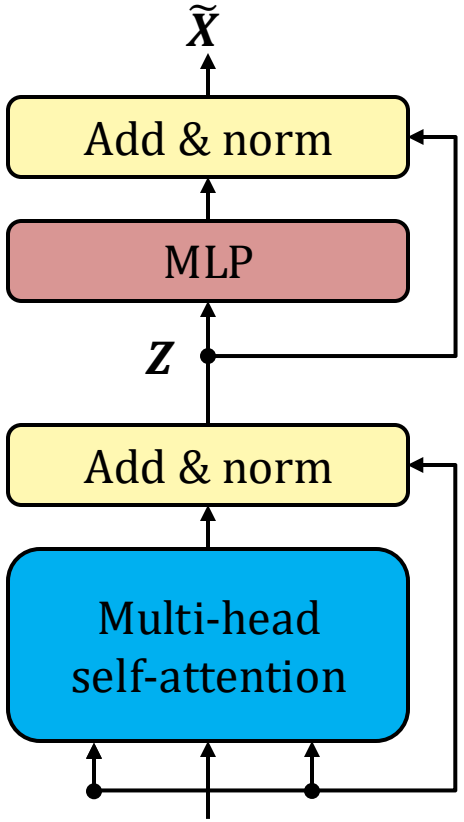
- To capture multiple patterns of attention that are relevant at the same time



Transformer Layers

- Residual connections that bypass the multi-head structure
- Layer normalization to improve training efficiency





$$\tilde{X} = WZ + b$$

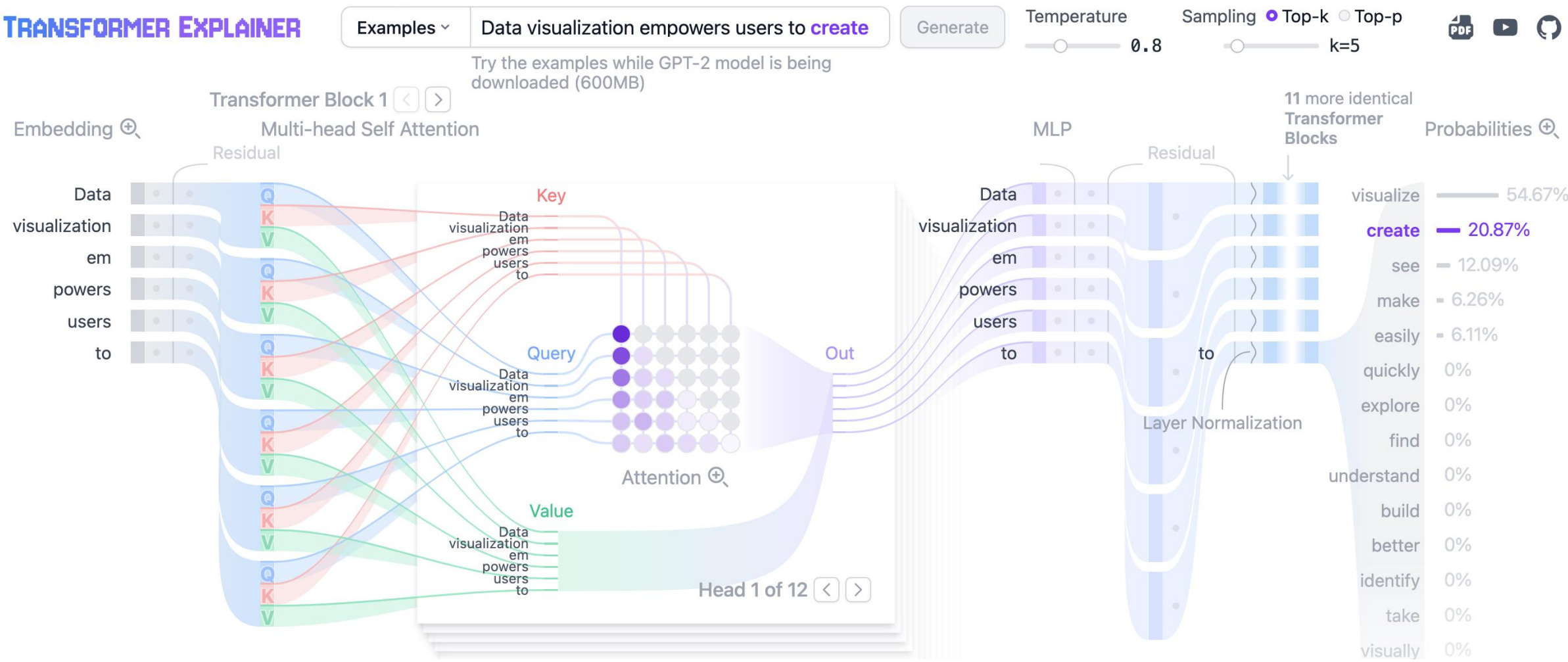
$Z \in \mathbb{R}^d$

$W \in \mathbb{R}^{V \times d}$

$\tilde{X} \in \mathbb{R}^V$

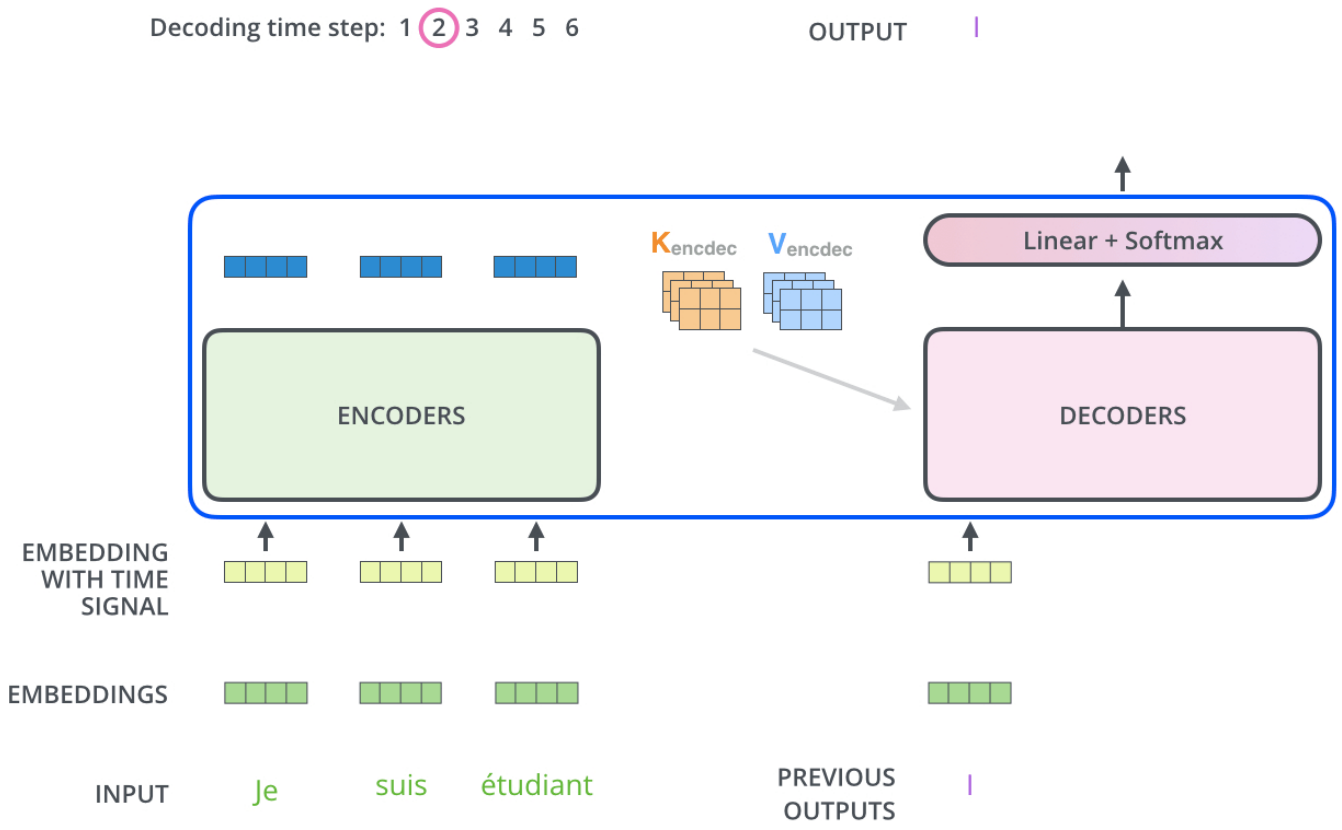
d: model size
V: number of vocabularies

Softmax and Output Probabilities



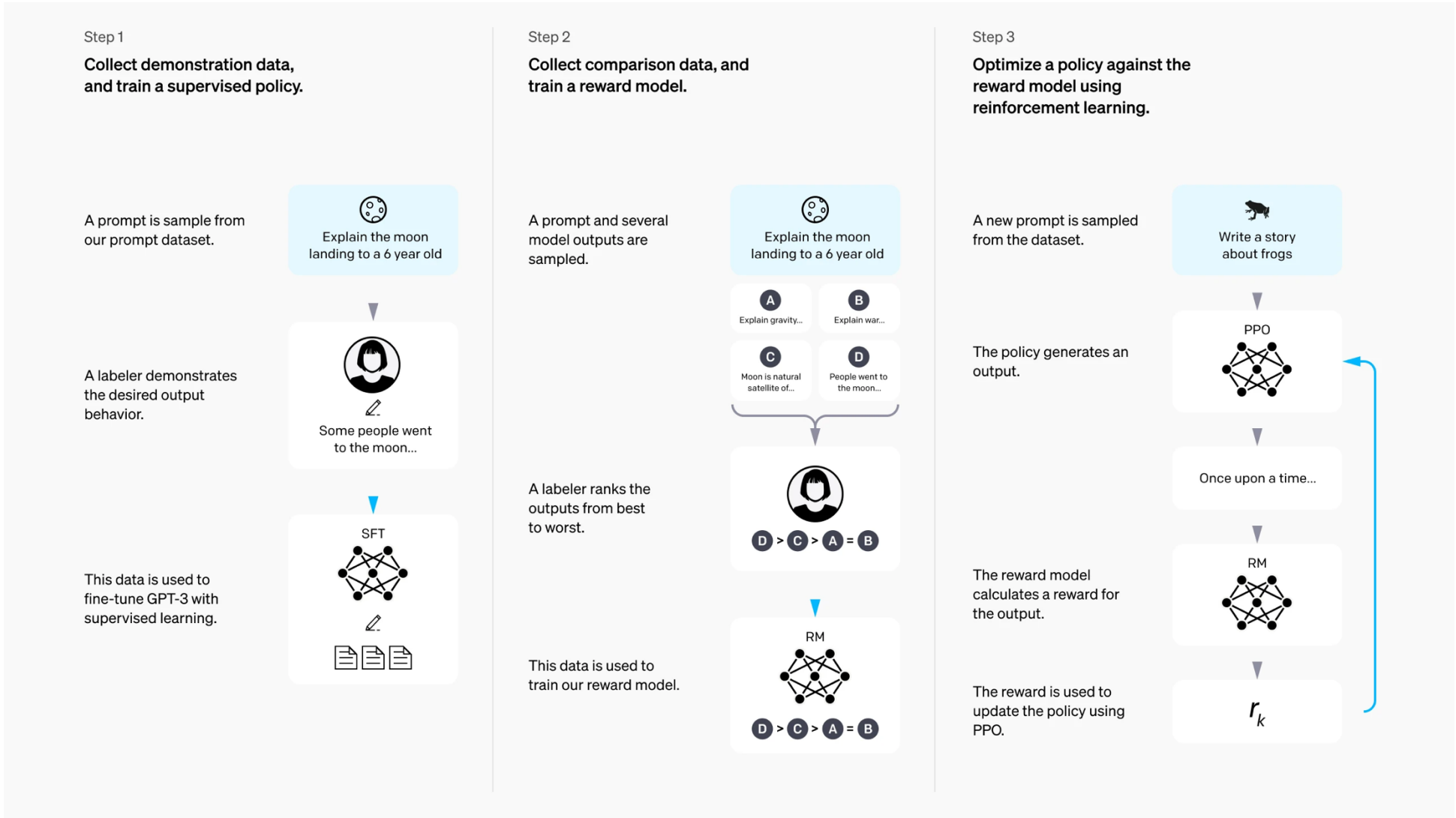
https://poloclub.github.io/transformer-explainer/

Transformer Overview



https://jalammar.github.io/illustrated-transformer/

RLHF

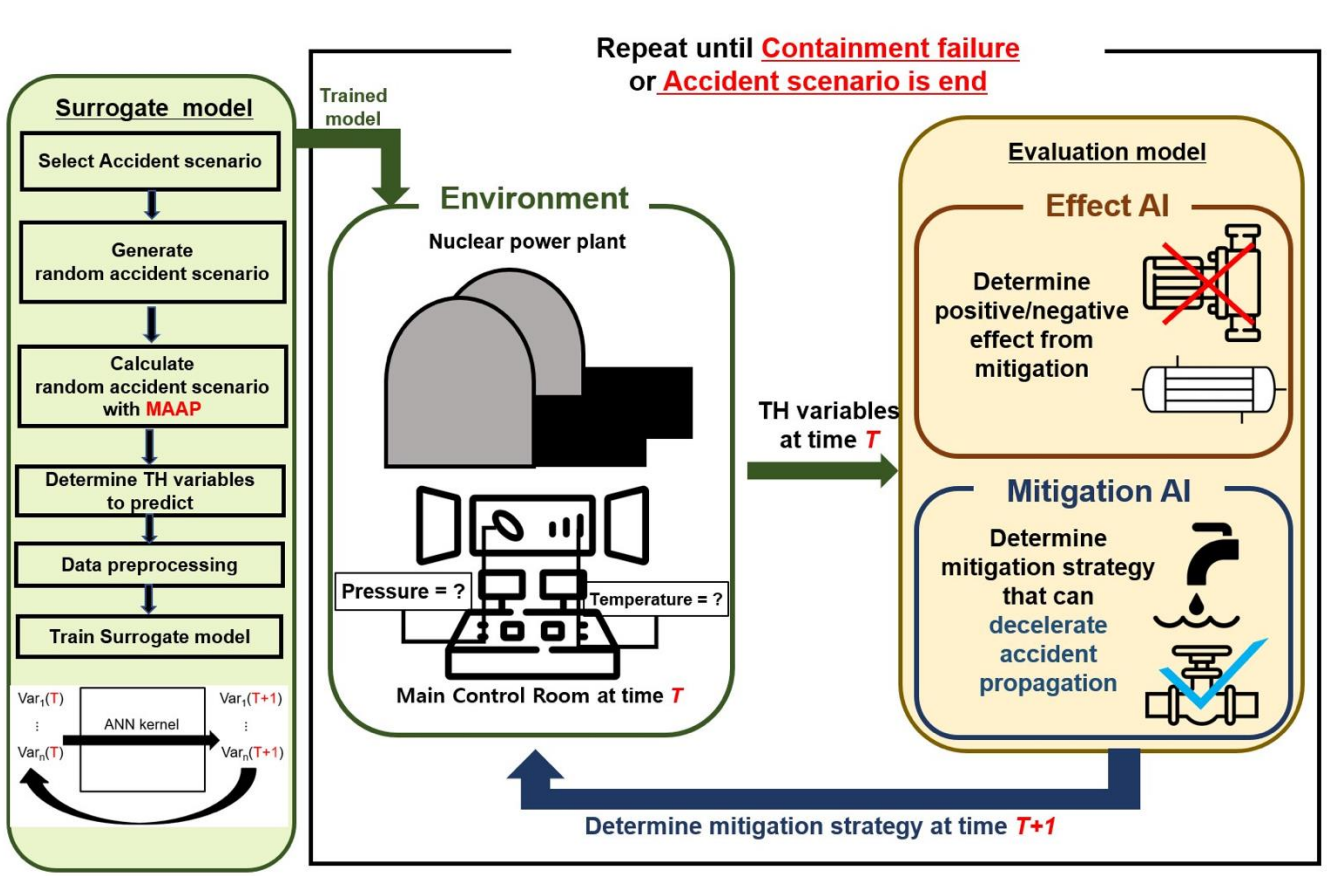


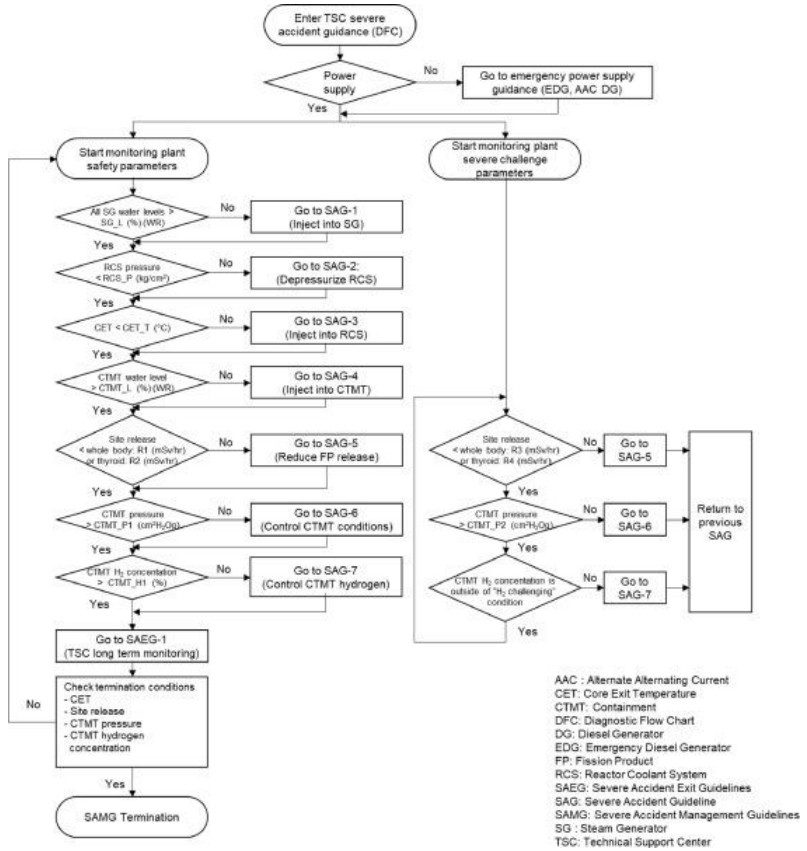
https://openai.com/ko-KR/index/instruction-following/

Table of Contents

Acceleration of Severe Accident Code

- KNHP aims to develop an AI agent to support nuclear plant operators.
- **To train the AI agent, a super-fast surrogate model that understands SAMG is required.**





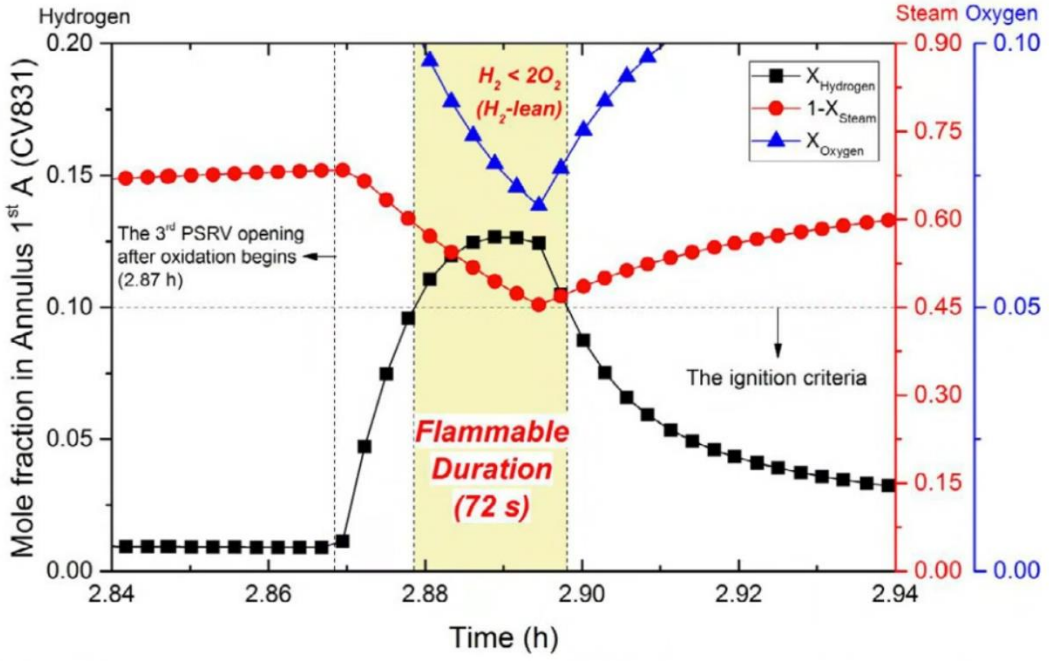
*SAMG: severe accident management guideline

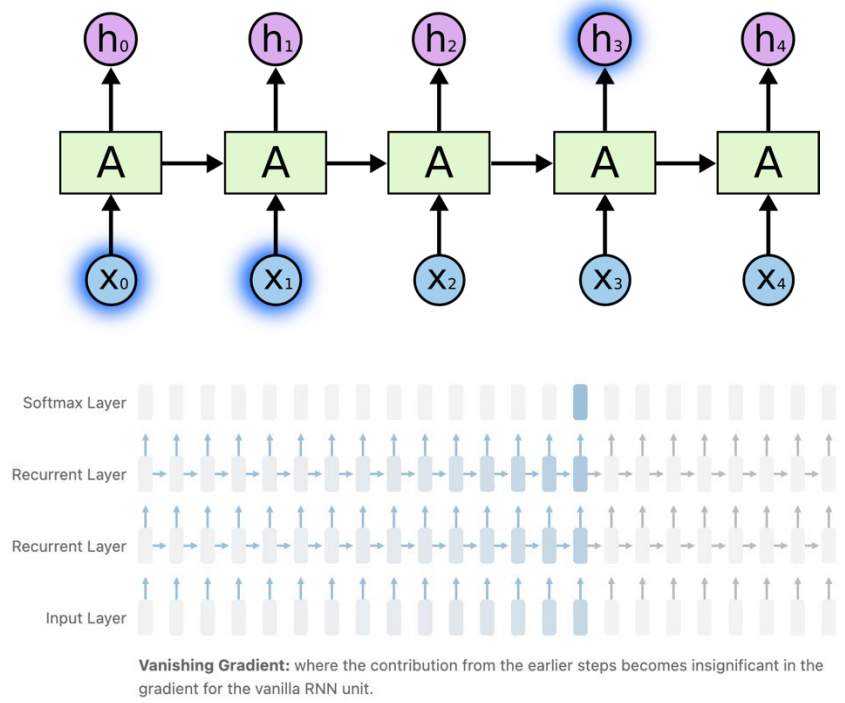
Acceleration of Severe Accident Code

- Time-series forecasting with AI model.

MLP (f_θ: $\mathbb{R}^1 \rightarrow \mathbb{R}^1$): $[x_t] \rightarrow [x_{t+1}]$ ⇨ RNN (f_θ: $\mathbb{R}^5 \rightarrow \mathbb{R}^1$): $[x_{t-4,\ldots,}x_t] \rightarrow [x_{t+1}]$ ⇨ Transformer (f_θ: $\mathbb{R}^{512} \rightarrow \mathbb{R}^1$): $[x_{t-511,\ldots,}x_t] \rightarrow [x_{t+1}]$

- Why do we need long input sequence length?

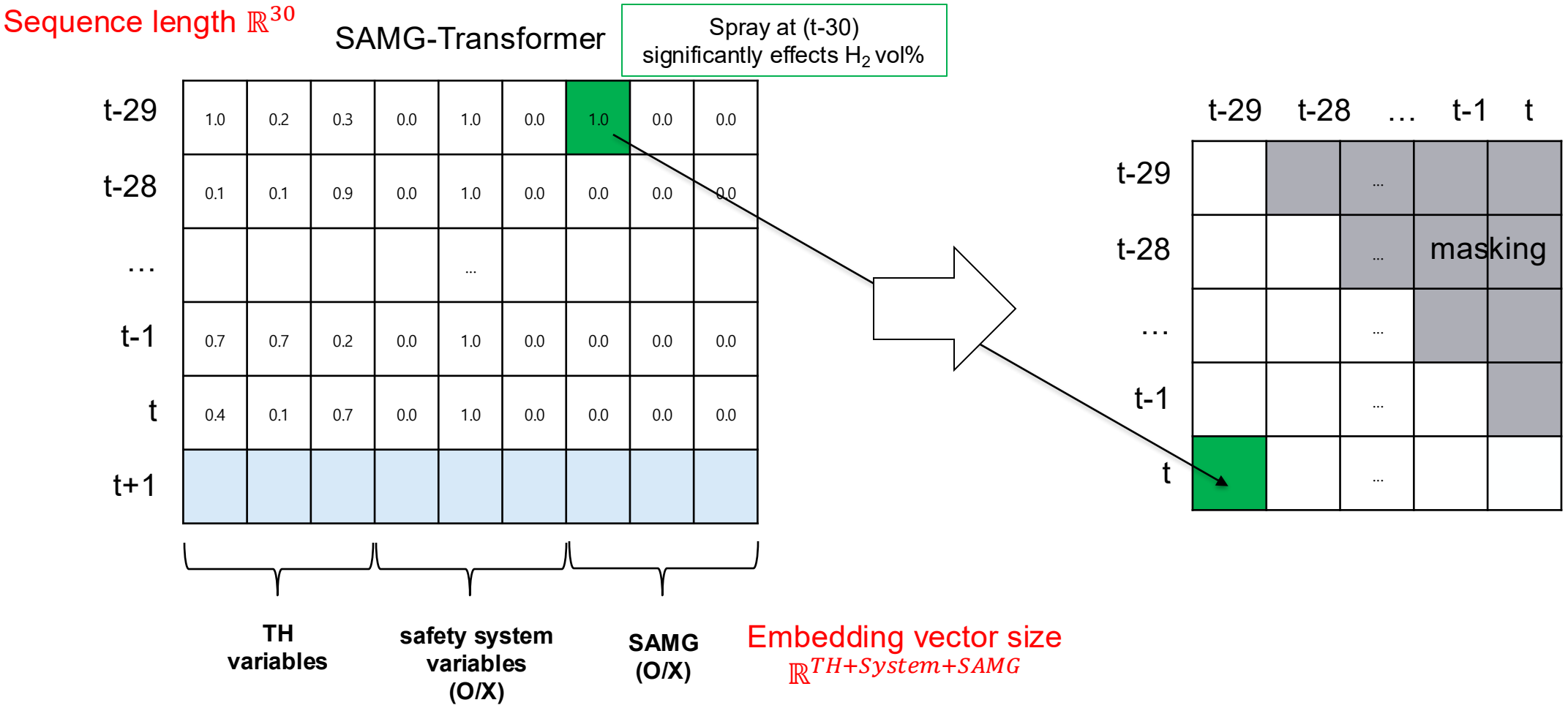




- Only transformer (attention mechanism) can analyze long input sequence length!

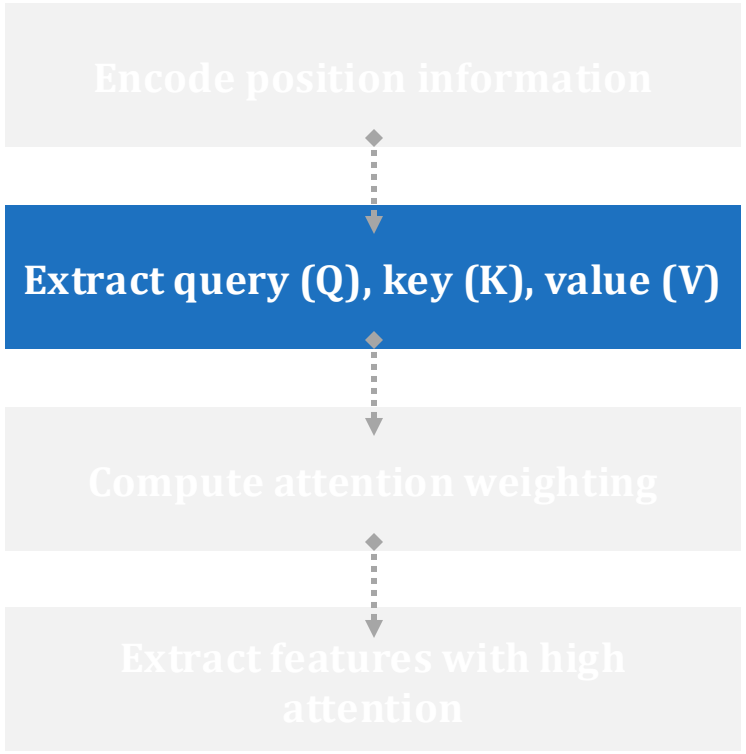
Acceleration of Severe Accident Code

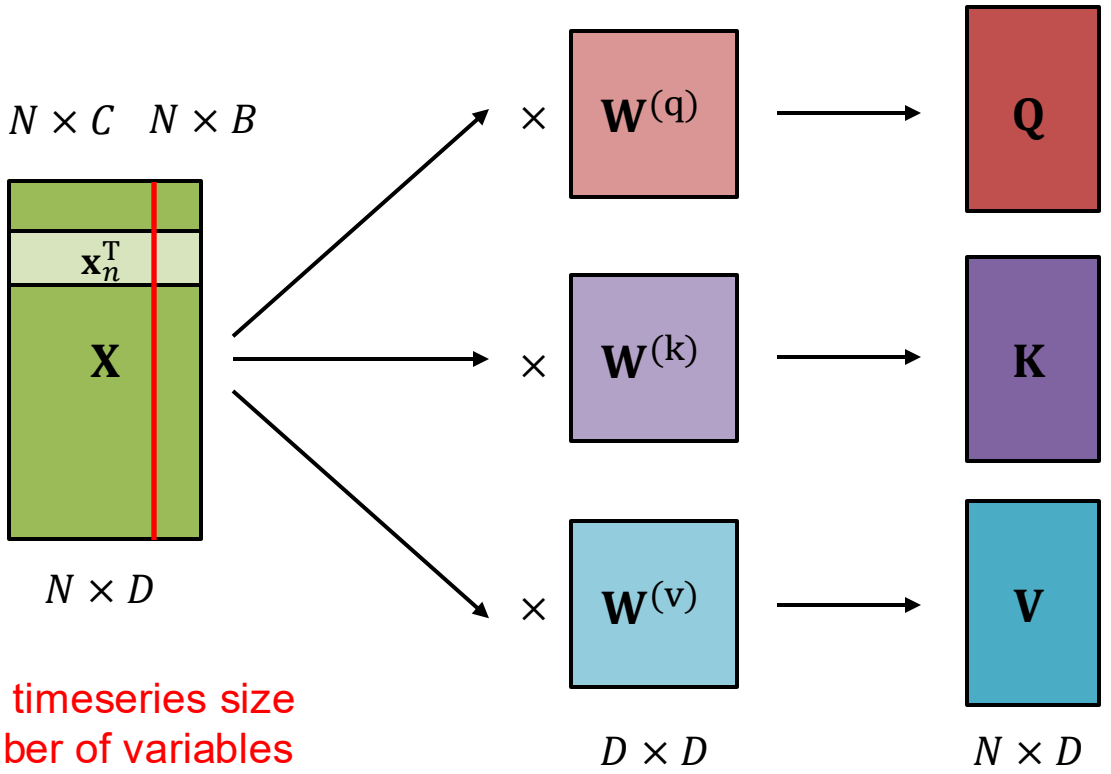
- Develop a **transformer** model that utilizes SAMG variables as embedding vectors.
- With attention mechanism**, we can develop severe accident time scale – surrogate model.**



Acceleration of Severe Accident Code

- **ABC-Transformer**: Accident binary-continuous Transformer

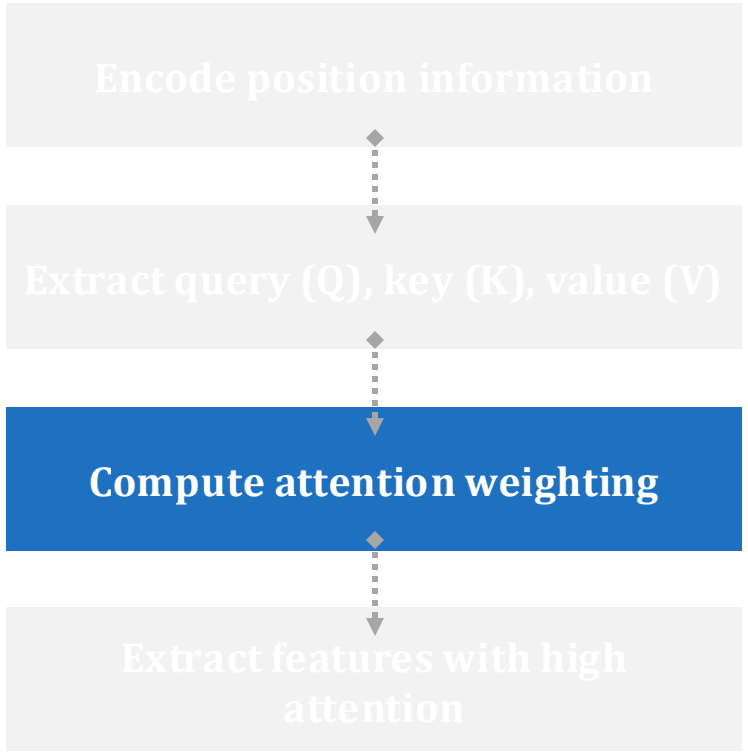




N: past timeseries size
D: number of variables
C: number of continuous variables
B: number of binary variables

Acceleration of Severe Accident Code

- **ABC-Transformer**: Accident binary-continuous Transformer



N: past timeseries size
D: number of variables

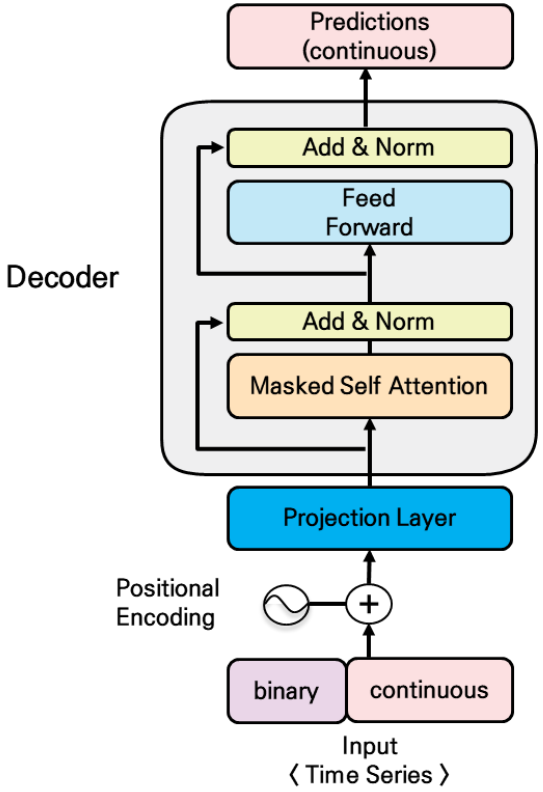
$$\mathbf{Q} \cdot \mathbf{K}^{\mathrm{T}} \rightarrow \mathrm{Softmax}\left\{\mathbf{Q}\mathbf{K}^{\mathrm{T}}\right\}$$

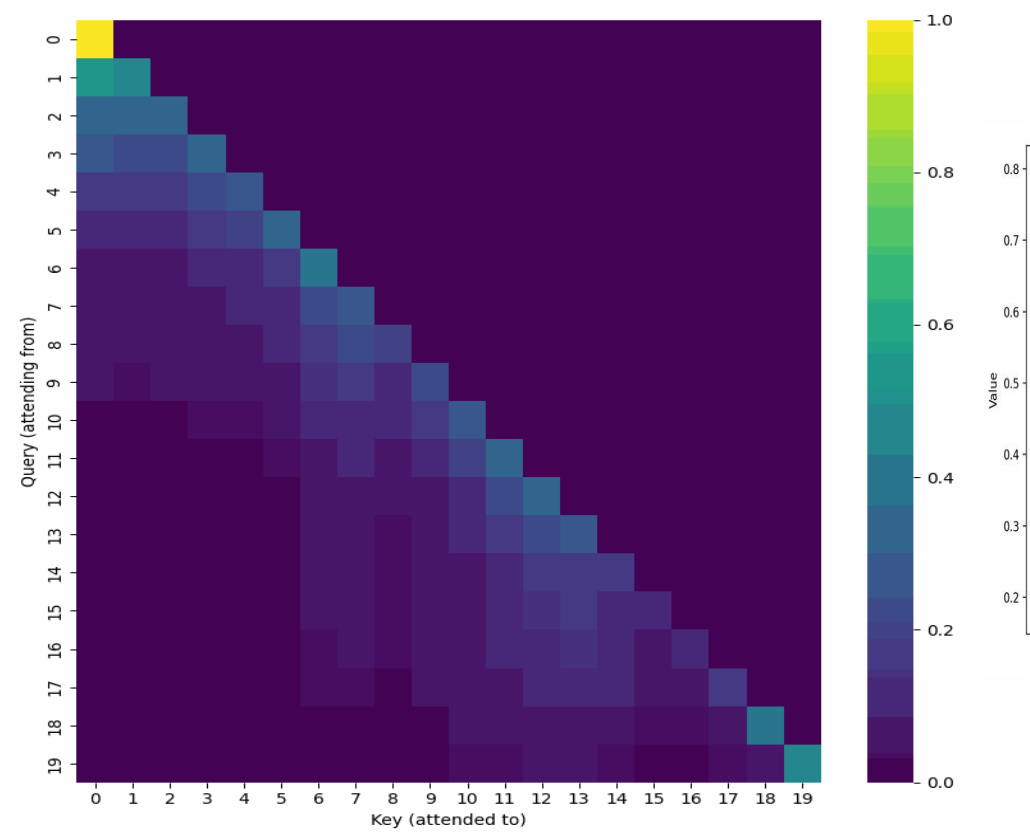
$N \times D$ $\quad$ $D \times N$ $\quad$ $N \times N$

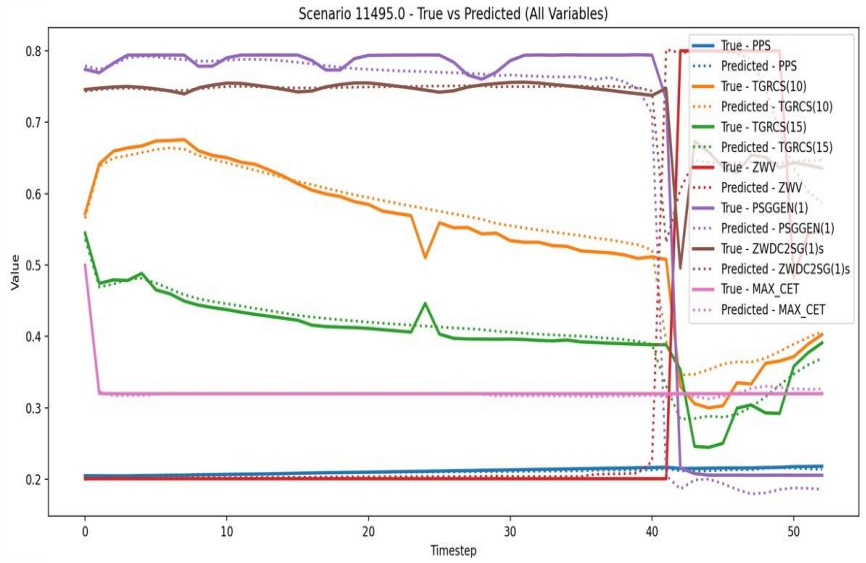
Attention score by computing similarity between **Q** and **K**

Acceleration of Severe Accident Code

- Develop the **ABC-Transformer** that utilizes SAMG variables as embedding vectors.
- Training examples:

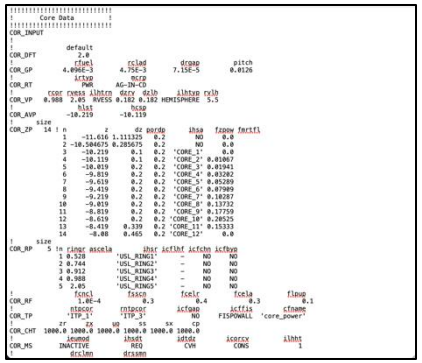


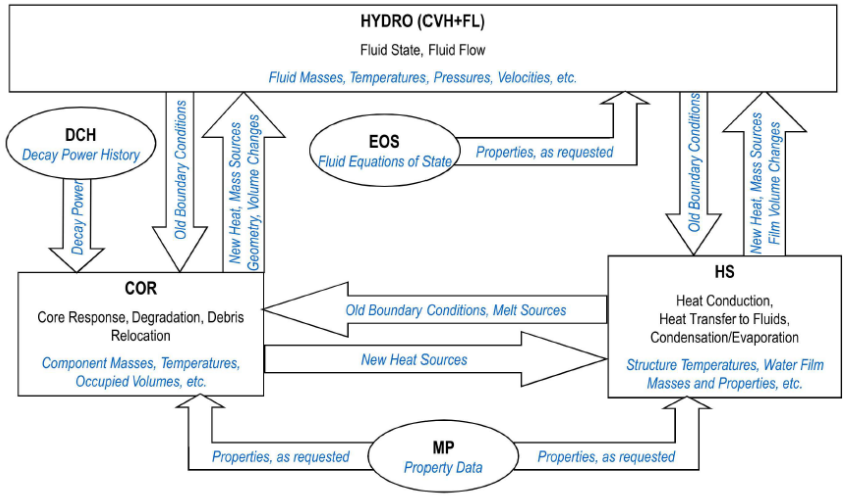


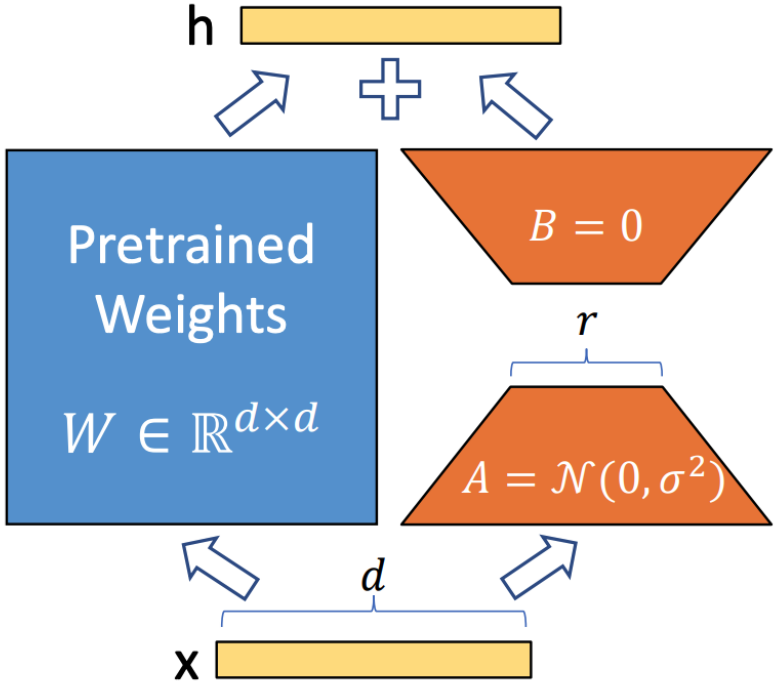


AI Agent for Automating Input Generation

- **Flamenco Project: F**ine-tuned **la**rge language model for **ME**LCOR **no-co**ding
- **Fine-tuning of open-source LLMs**





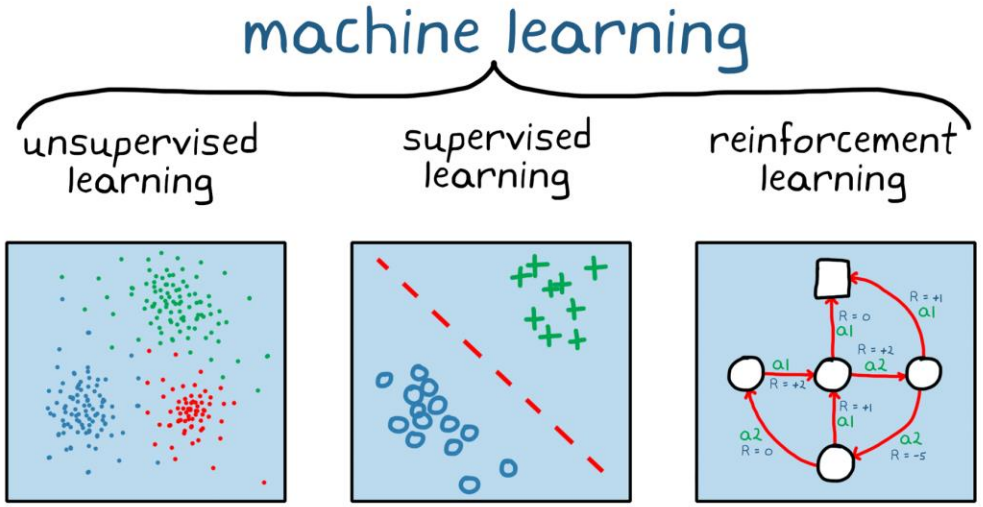


Prof. Sooyoung Lee (CAU)

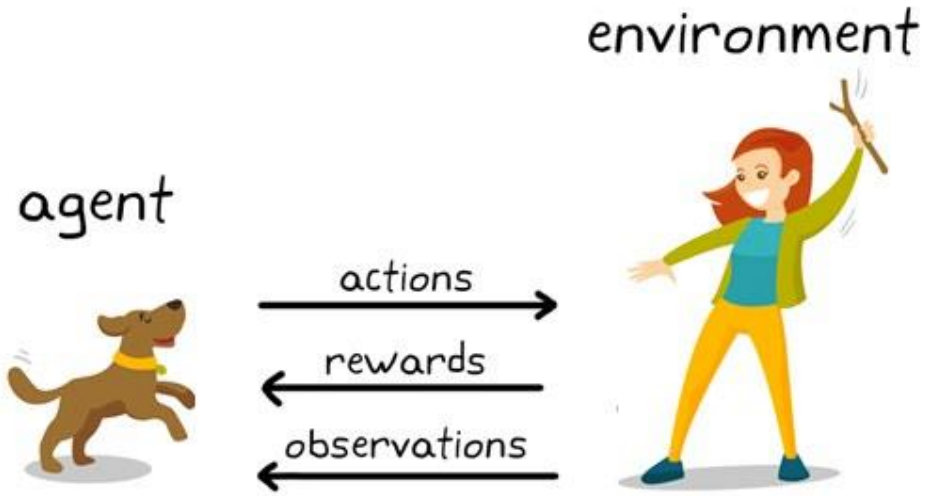
Thank You.
Any Questions?

jgjeon41@postech.ac.kr

Types of learning: SL vs UL vs RL



(MathWorks)



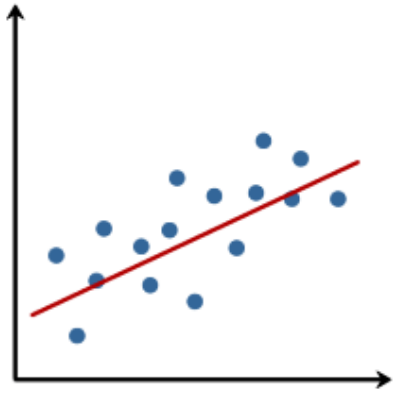
- **Supervised learning**
  - **Training data:** $\{(x_i, y_i); i = 1, \dots, N\}$

    x_i: input (feature)

    y_i: output (label)
  - Regression: cardinal values
  - Classification: categorical values
- **Unsupervised learning**
  - **Training data:** $\{x_i; i = 1, \dots, N\}$ **(no labels)**
  - Clustering, dimension reduction, Generative AI
- **Reinforcement learning**
  - **No (prior) data**
  - Learning from experience (interaction with environment)
  - like our life!

Types of learning: Regression vs Classification

Linear



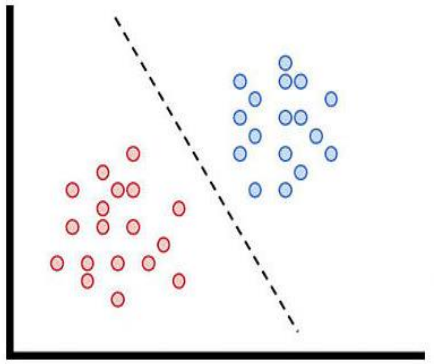
Regression

Continuous

Time / Weight / Height

VS

Logistic / Softmax



Classification

Discrete

What is Binary (Multi-class) Classification?
Variable is either 0 or 1 (0:positive / 1:negative)

- Exam : Pass or Fail
- Spam : Not Spam or Spam
- Face : Real or Fake
- Tumor : Not Malignant or Malignant

E.g., Encode variables to [0,1], or [0,1,2,…]

Seq2Seq with Attention Mechanism

- Notice a few things in the Figure:

1. The Encoder is RNN
2. Based on the hidden states, context vector(c_i) is generated $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$
3. α is parameterized as a feedforward neural network.

- The probability α_{ij} reflects the importance of the annotation h_j w.r.t. previous hidden state(s_{i-1}) in deciding the next state(s_i).

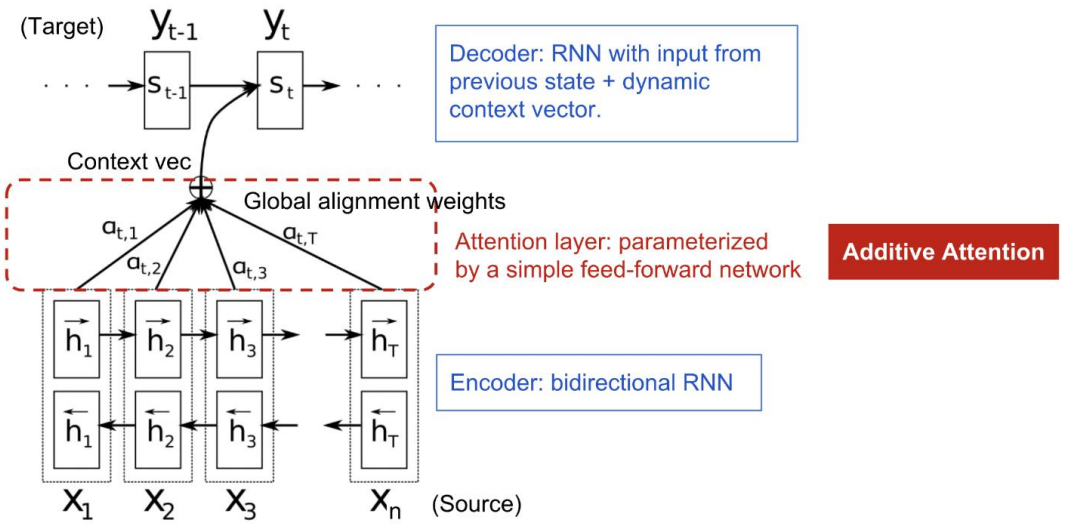


Fig. 4. The encoder-decoder model with additive attention mechanism in Bahdanau et al., 2015.

- ***So, the decoder decides which part of the sentence to pay attention to.***
- ***we don't need to encode the information into a fixed size context vector.***