Georgia Tech
College of Computing

CX4240 Spring 2026

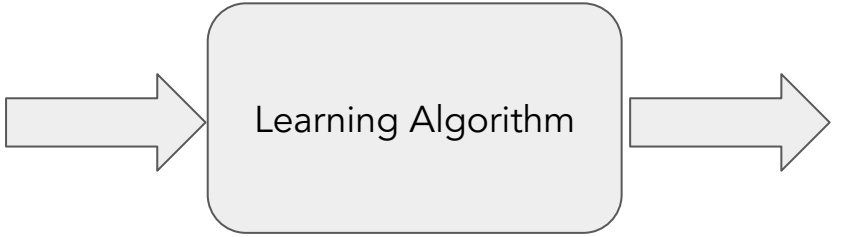
(Large) Language Model (Part I): Attention and Transformers

Bo Dai
School of CSE, Georgia Tech
bodai@cc.gatech.edu

Presentation

- Please signup the presentation date in group
  - Today is the deadline!
  - Group 7, 11, 18, and 21
- Final presentation will be online.
- Make sure to complete in time to get the bonus
  - Think this is a conference talk
- Final report is due on May 4th.

Supervised Learning vs. Unsupervised Learning



Training Data

Supervised $\{x^i, y^i\}_{i=1}^m$

Unsupervised $\{x_i\}_{i=1}^m$

Target Function

$f : X \to Y$

Density Estimation $p(x)$

Clustering $f : X \to Y$

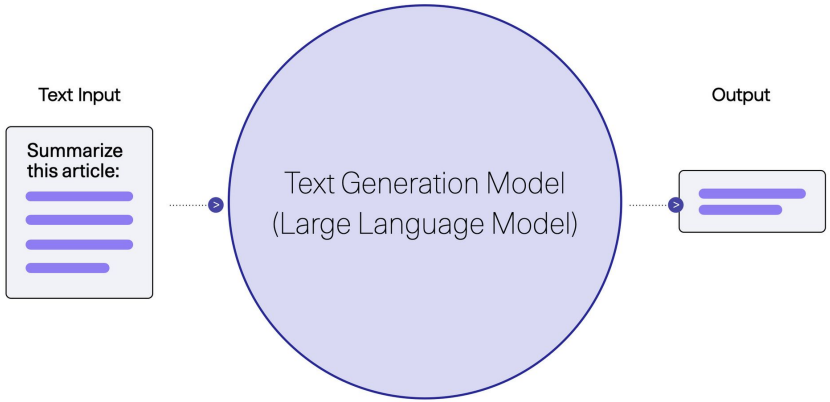
$Y \in \{0, 1, \ldots, k\}$

Dimension Reduction $f : X \to Y$

$X \in \mathbb{R}^p, \quad Y \in \mathbb{R}^d$

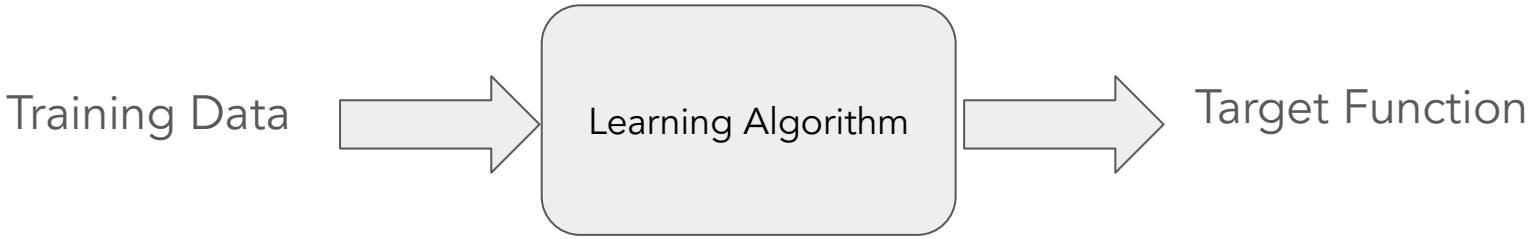
(Large) Language Models

all text data

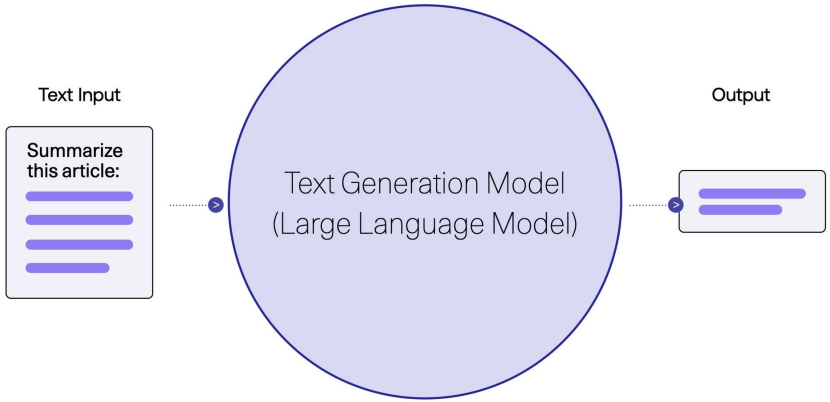


$$p(x) = \prod_{i=1}^{k} p(x_i | x_{<i})$$

(Large) Language Models



all text data

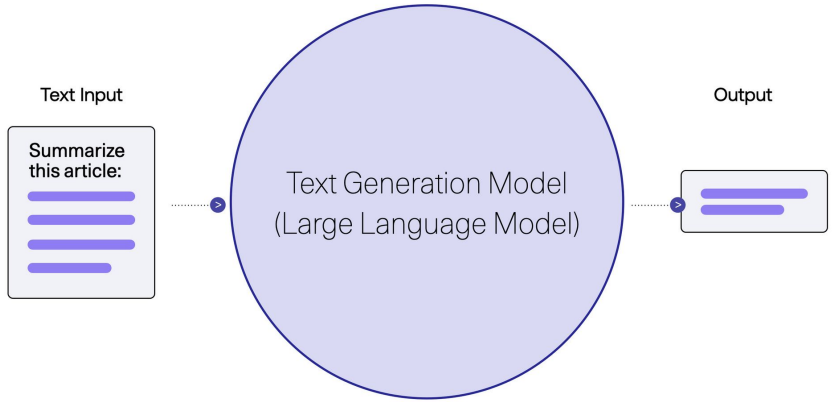


$$p(x_{l,\ldots,k}|x_{<l}) = \prod_{i=l}^{k} p(x_i|x_{<i})$$

(Large) Language Models

Training Data → Learning Algorithm → Target Function

Unlabelled Data: text sequences



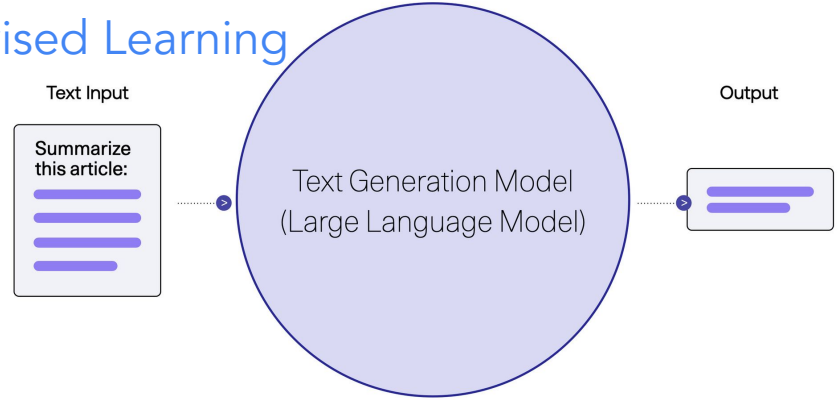
$$p(x) = \prod_{i=1}^{k} p(x_i | x_{<i})$$

(Large) Language Models

Training Data → Learning Algorithm → Target Function

Unlabelled Data: text sequences

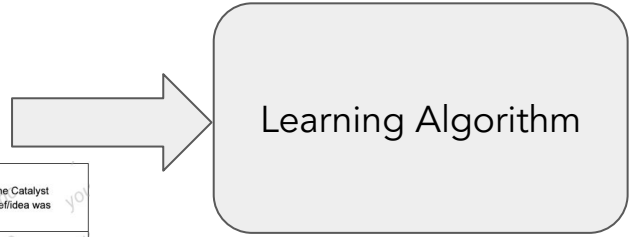
Unsupervised Learning



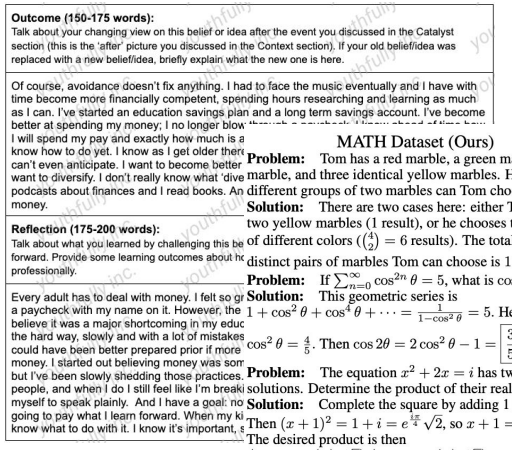
$$p(x) = \prod_{i=1}^{k} p(x_i | x_{<i})$$

(Large) Language Models

Training Data



Target Function



MATH Dataset (Ours)

Problem: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?

Solution: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2} = 6$ results). The total number of distinct pairs of marbles Tom can choose is $1 + 6 = \boxed{7}$.

Problem: If $\sum_{n=0}^{\infty} \cos^{2n} \theta = 5$, what is $\cos 2\theta$?

Solution: This geometric series is $1 + \cos^2 \theta + \cos^4 \theta + \cdots = \frac{1}{1-\cos^2 \theta} = 5$. Hence, $\cos^2 \theta = \frac{4}{5}$. Then $\cos 2\theta = 2\cos^2 \theta - 1 = \boxed{\frac{3}{5}}$.

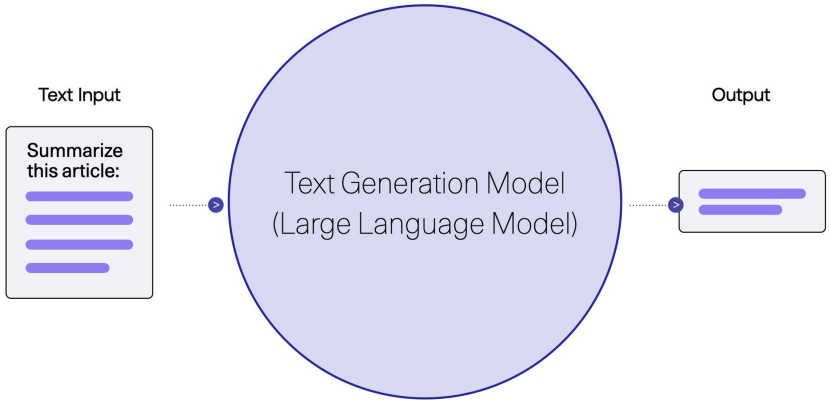
Problem: The equation $x^2 + 2x = i$ has two complex solutions. Determine the product of their real parts.

Solution: Complete the square by adding 1 to each side. Then $(x+1)^2 = 1 + i = e^{\frac{i\pi}{4}}\sqrt{2}$, so $x + 1 = \pm e^{\frac{i\pi}{8}}\sqrt[4]{2}$. The desired product is then

$$\left(-1 + \cos\left(\tfrac{\pi}{8}\right)\sqrt[4]{2}\right)\left(-1 - \cos\left(\tfrac{\pi}{8}\right)\sqrt[4]{2}\right) =$$

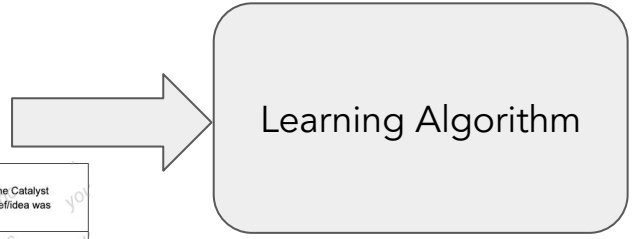
$$1 - \cos^2\left(\tfrac{\pi}{8}\right)\sqrt{2} = 1 - \frac{\left(1+\cos\left(\frac{\pi}{4}\right)\right)}{2}\sqrt{2} = \boxed{\frac{1-\sqrt{2}}{2}}.$$

Labelled Data: Prompt & Answer



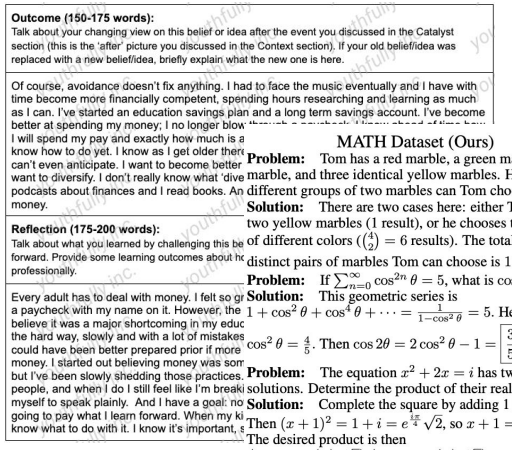
$$p(x_{l,\ldots,k}|x_{<l}) = \prod_{i=l}^{k} p(x_i|x_{<i})$$

(Large) Language Models



Training Data

Target Function



MATH Dataset (Ours)

Problem: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?

Solution: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2} = 6$ results). The total number of distinct pairs of marbles Tom can choose is $1 + 6 = \boxed{7}$.

Problem: If $\sum_{n=0}^{\infty} \cos^{2n} \theta = 5$, what is $\cos 2\theta$?

Solution: This geometric series is $1 + \cos^2 \theta + \cos^4 \theta + \cdots = \frac{1}{1-\cos^2 \theta} = 5$. Hence, $\cos^2 \theta = \frac{4}{5}$. Then $\cos 2\theta = 2\cos^2 \theta - 1 = \boxed{\frac{3}{5}}$.

Problem: The equation $x^2 + 2x = i$ has two complex solutions. Determine the product of their real parts.

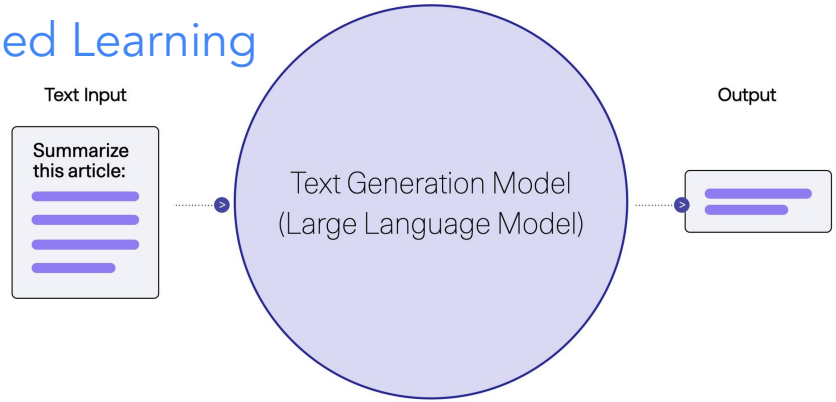
Solution: Complete the square by adding 1 to each side. Then $(x+1)^2 = 1 + i = e^{\frac{i\pi}{4}}\sqrt{2}$, so $x + 1 = \pm e^{\frac{i\pi}{8}}\sqrt[4]{2}$. The desired product is then

$$\left(-1 + \cos\left(\tfrac{\pi}{8}\right)\sqrt[4]{2}\right)\left(-1 - \cos\left(\tfrac{\pi}{8}\right)\sqrt[4]{2}\right) =$$

$$1 - \cos^2\left(\tfrac{\pi}{8}\right)\sqrt{2} = 1 - \frac{\left(1+\cos\left(\frac{\pi}{4}\right)\right)}{2}\sqrt{2} = \boxed{\frac{1-\sqrt{2}}{2}}.$$

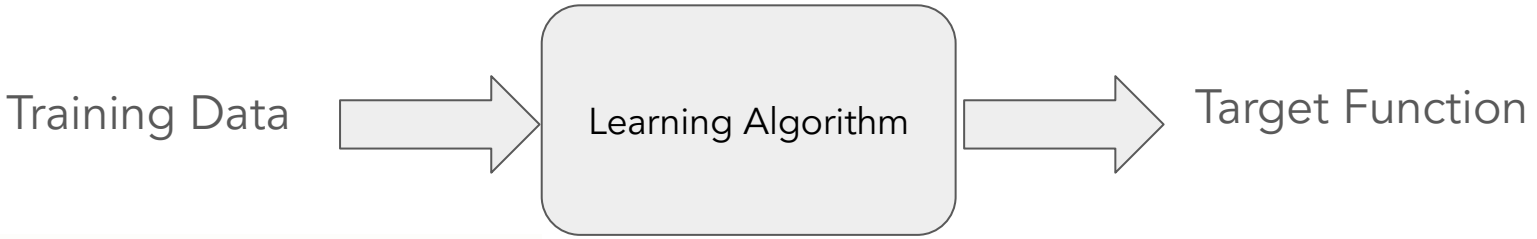
Labelled Data: Prompt & Perfect Answer

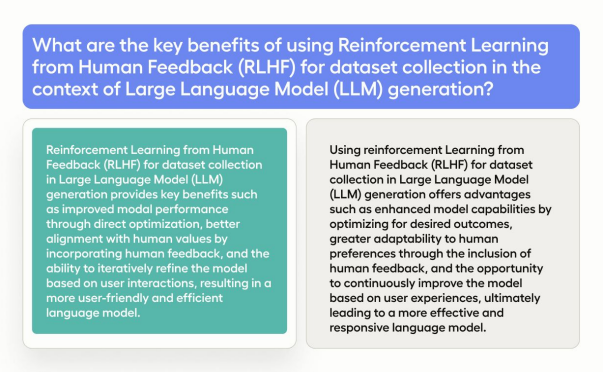
Supervised Learning

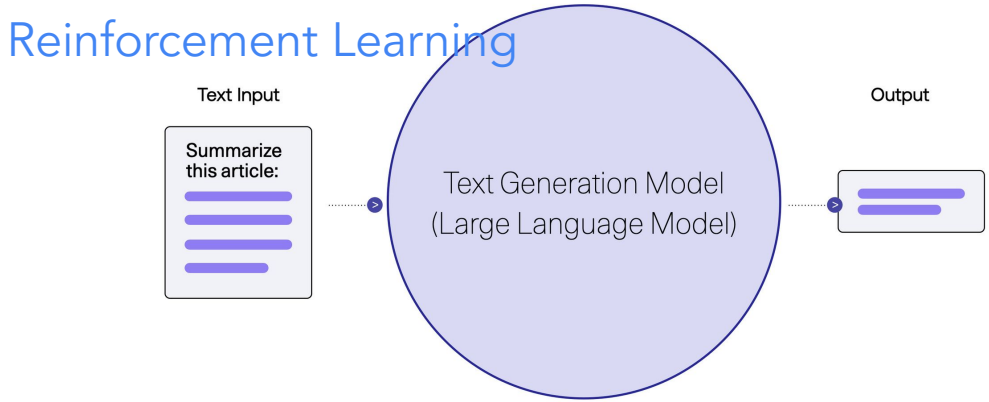


$$p(x_{l,\ldots,k}|x_{<l}) = \prod_{i=l}^{k} p(x_i|x_{<i})$$

(Large) Language Models



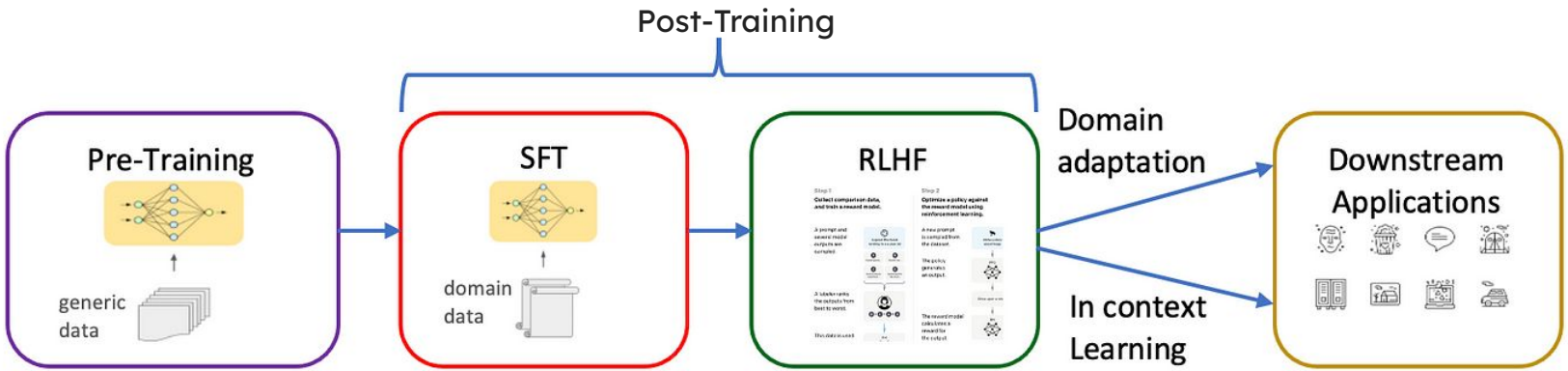




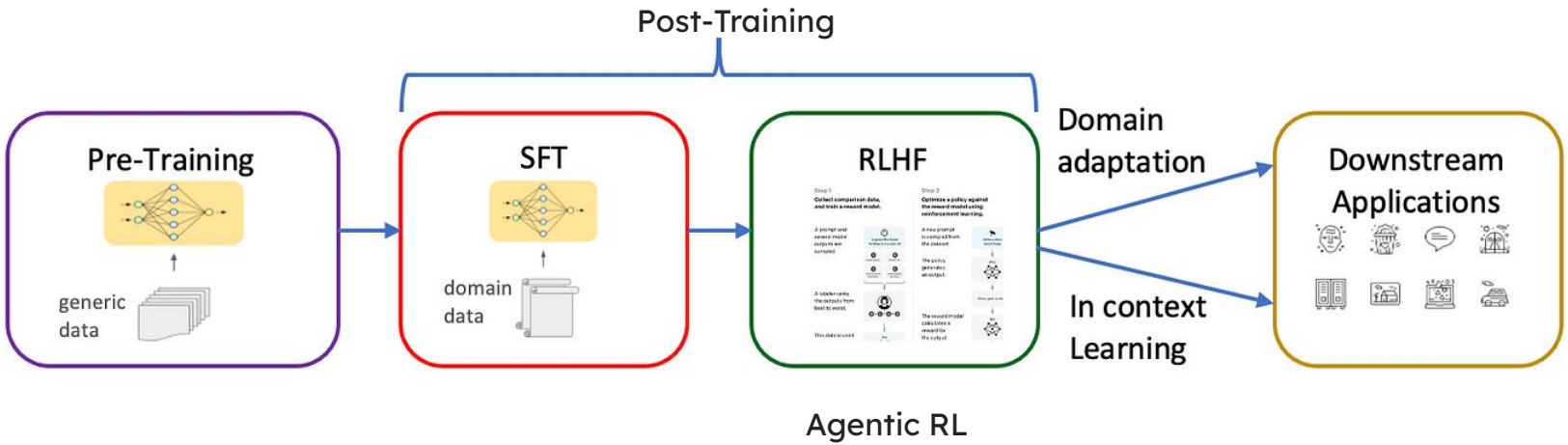
Preference Data:
Prompt & Positive/ Negative Answers

$$p(x_{l,\ldots,k}|x_{<l}) = \prod_{i=l}^{k} p(x_i|x_{<i})$$

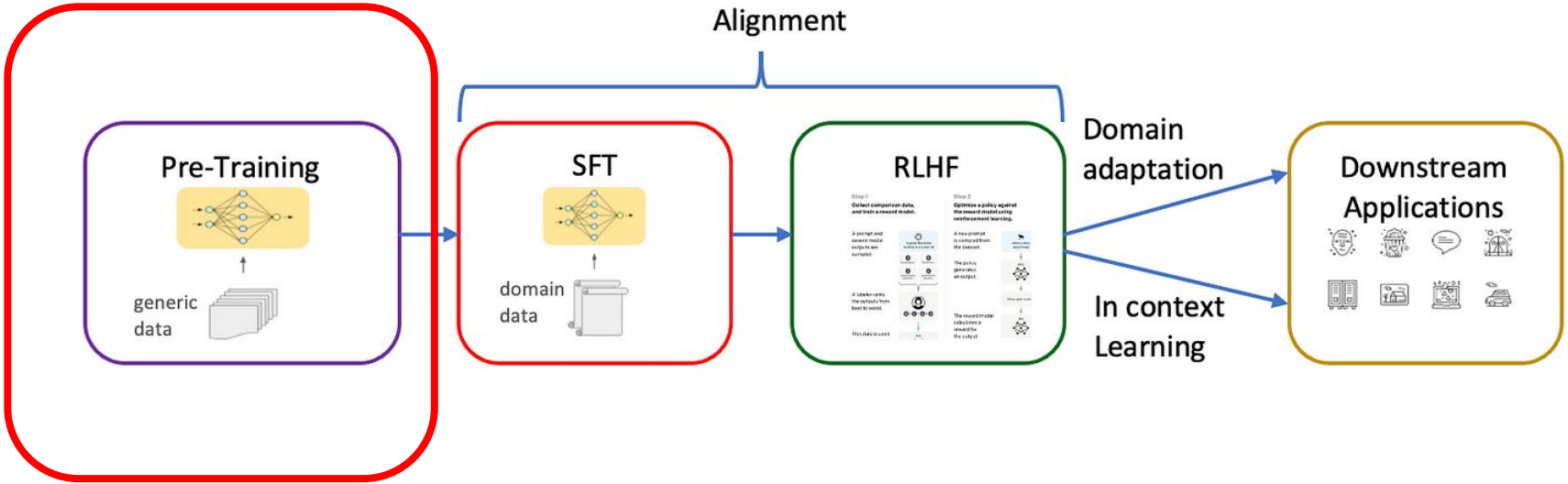
LLM Training



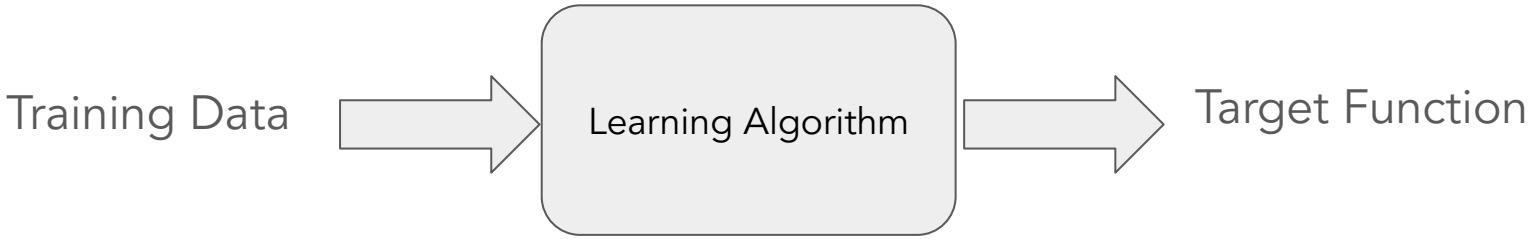
LLM Training



LLM Training



(Large) Language Models

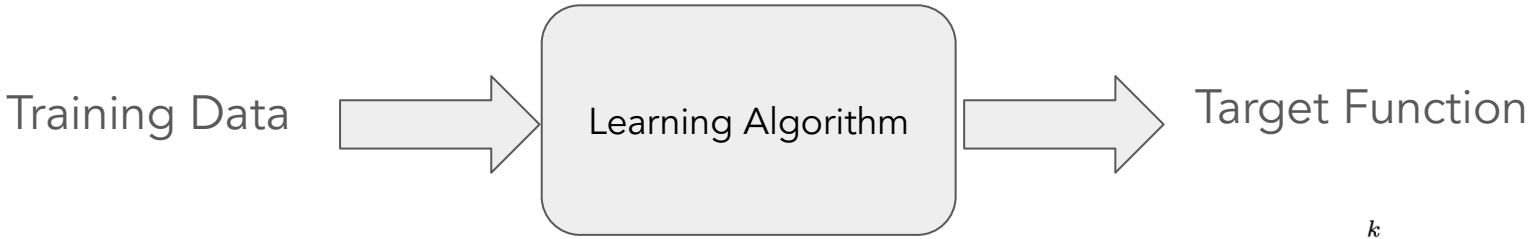


For a movie that gets no respect there sure are a lot of memorable quotes listed for this gem. Imagine a movie where Joe Piscopo is actually funny! Maureen Stapleton is a scene stealer. The Moroni character is an absolute scream. Watch for Alan "The Skipper" Hale jr. as a police Sgt.

$$p(X)$$

$$D = \{X^i\}_{i=1}^n, \quad X^i = \{x_j^i\}_{j=1}^t$$

(Large) Language Models



$$D = \{X^i\}_{i=1}^n, \quad X^i = \{x_j^i\}_{j=1}^t$$

$$p(x) = \prod_{i=1}^{k} p(x_i | x_{<i})$$

1. Build probabilistic models
   Categorical Distribution + Autoregressive + RNN->Transformer
2. Derive loss function (by MLE or MAP….)
3. Select optimizer

Combating Combinatorial Complexity: Autoregressive Model

$$p(X) = p\left(\{x_j\}_{j=1}^{t}\right) \qquad O(|V|^T)$$

Combating Combinatorial Complexity: Autoregressive Model

$$p(X) = p\left(\{x_j\}_{j=1}^{t}\right) \qquad O(|V|^T)$$
$$= \prod_{j=1}^{t} p(x_j | x_{<j})$$

Categorical Distribution

$$
\begin{aligned}
p(X) &= p\left(\{x_j\}_{j=1}^{t}\right) && O(|V|^T) \\
&= \prod_{j=1}^{t} p(x_j | x_{<j}) \\
&= \prod_{j=1}^{t} \frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}
\end{aligned}
$$

Recursive Neural Network

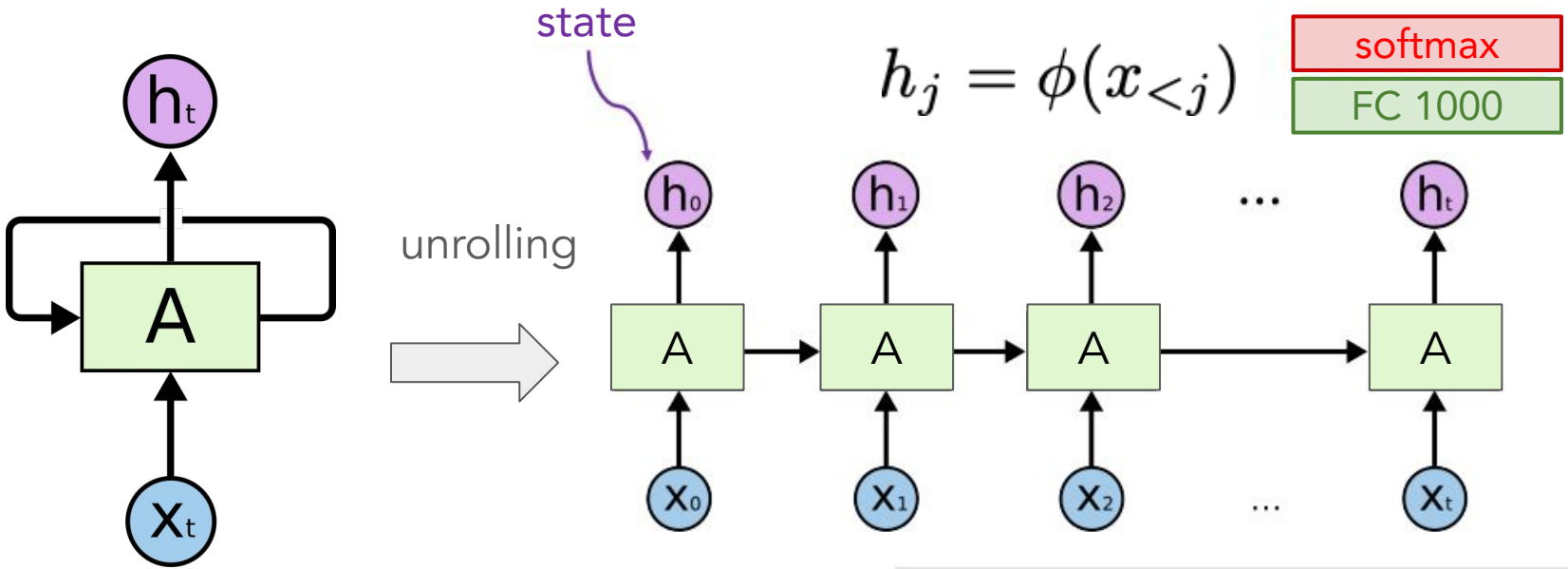
$$p(X) = p\left(\{x_j\}_{j=1}^{t}\right) \qquad O(|V|^T)$$

$$= \prod_{j=1}^{t} p(x_j | x_{<j})$$

$$= \prod_{j=1}^{t} \frac{\exp(W_{x_j} \phi(x_{<j}))}{\sum_{l=1}^{V} \exp(W_l \phi(x_{<j}))} \qquad \text{RNN}$$

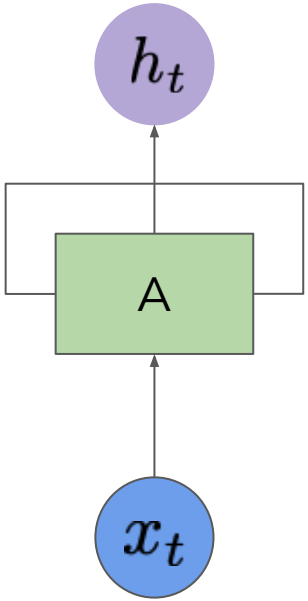
Recurrent Neural Network

$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$

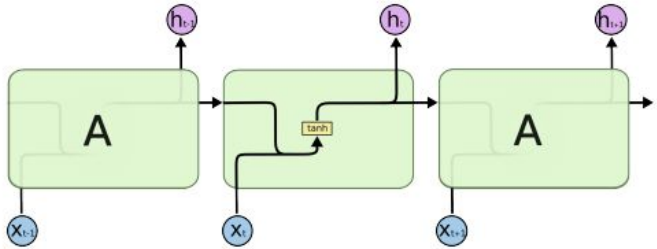


For a movie that gets no respect there sure are a lot of memorable quotes listed for this gem. Imagine a movie where Joe Piscopo is actually funny! Maureen Stapleton is a scene stealer. The Moroni character is an absolute scream. Watch for Alan "The Skipper" Hale jr. as a police Sgt.

RNN Cell



$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$

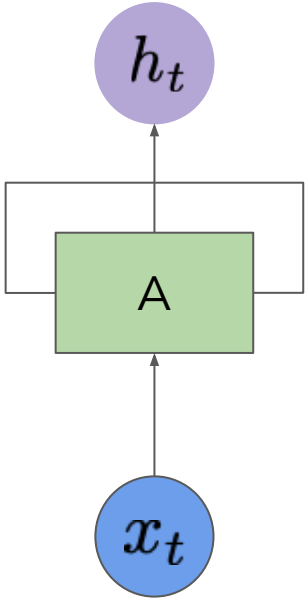


Simple RNN

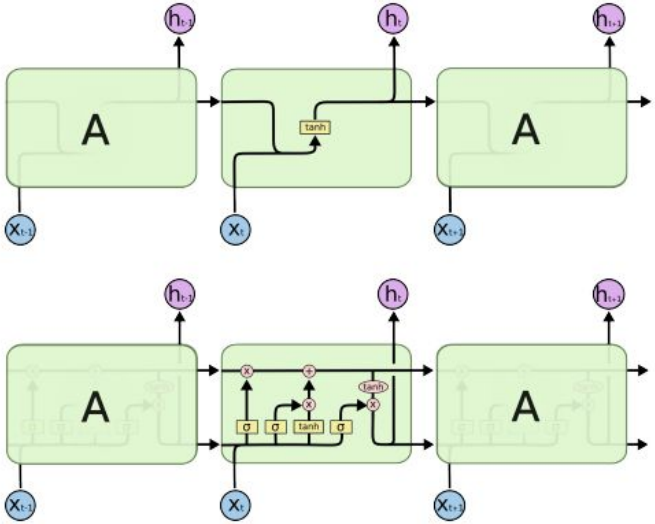
$\mathbf{h_{100}}$ is almost irrelevant to $\mathbf{x_1}$: $\frac{\partial \mathbf{h_{100}}}{\partial \mathbf{x_1}}$ is near zero.

Gradient Vanishing

RNN Cell



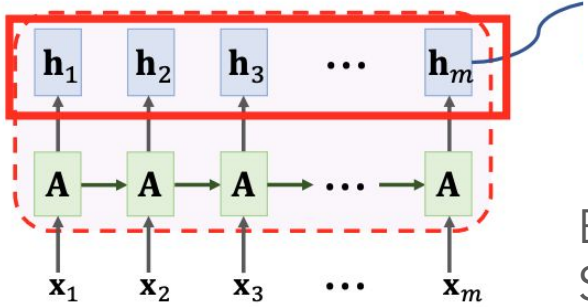
$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$



Simple RNN

LSTM

Bottleneck in RNN

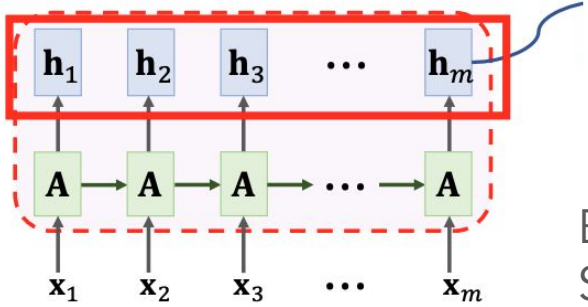


$$h_j = \phi(x_{<j})$$

$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$

Bottleneck:
Sequences bottlenecked through a fixed-sized vector.

Bottleneck in RNN



$$h_j = \phi(x_{<j})$$

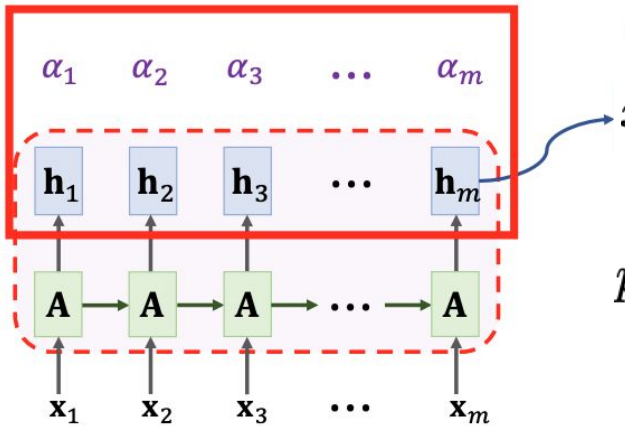
$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$

Bottleneck:
Sequences bottlenecked through a fixed-sized vector.

Fixed size -> Length dependent

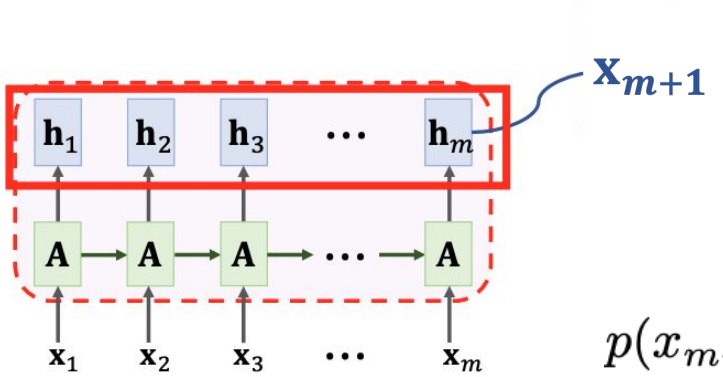
Attention: Flatten RNN Computation

Context vector: $\mathbf{c}_0 = \alpha_1 \mathbf{h}_1 + \cdots + \alpha_m \mathbf{h}_m$.



$$p(x_{m+1}|x_{<m+1}) = \frac{\exp(W_{x_j} c_0)}{\sum_{l=1}^{V} \exp(W_l c_0)}$$

Attention: Adding Output Dependency

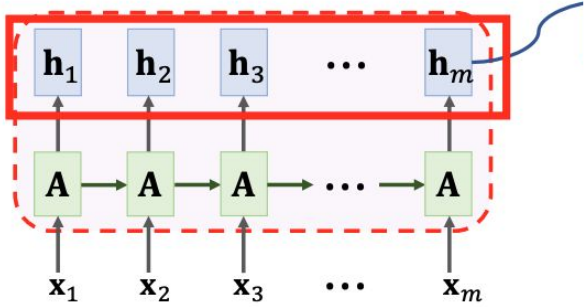


1. Linear maps:
   - $\mathbf{k}_i = \mathbf{W}_K \cdot \mathbf{h}_i$, for $i = 1$ to m.
   - $\mathbf{q}_0 = \mathbf{W}_Q \cdot \mathbf{x}_{m+1}$
2. Inner product:
   - $\tilde{\alpha}_i = \mathbf{k}_i^T \mathbf{q}_0$, for $i = 1$ to m.
3. Normalization:
   - $[\alpha_1, \cdots, \alpha_m] = \text{Softmax}([\tilde{\alpha}_1, \cdots, \tilde{\alpha}_m])$.

$$p(x_{m+1}|x_{<m+1}) = \frac{\exp(W_{x_j} c_0)}{\sum_{l=1}^{V} \exp(W_l c_0)}$$

Weight: $\alpha_i = \text{align}(\mathbf{h}_i, \mathbf{x}_{m+1})$

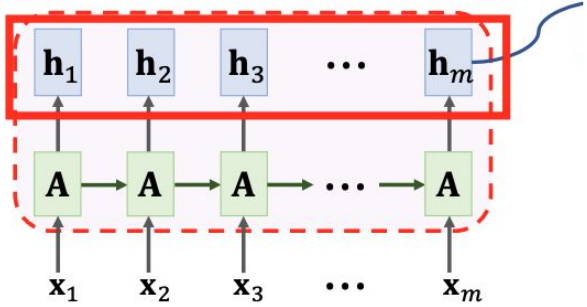
AutoRecursive Attention



$$x_{m+1}\ x_{m+2}, \ldots, x_{m+n}$$

$$p(x_{m+2}|x_{<m+2})$$

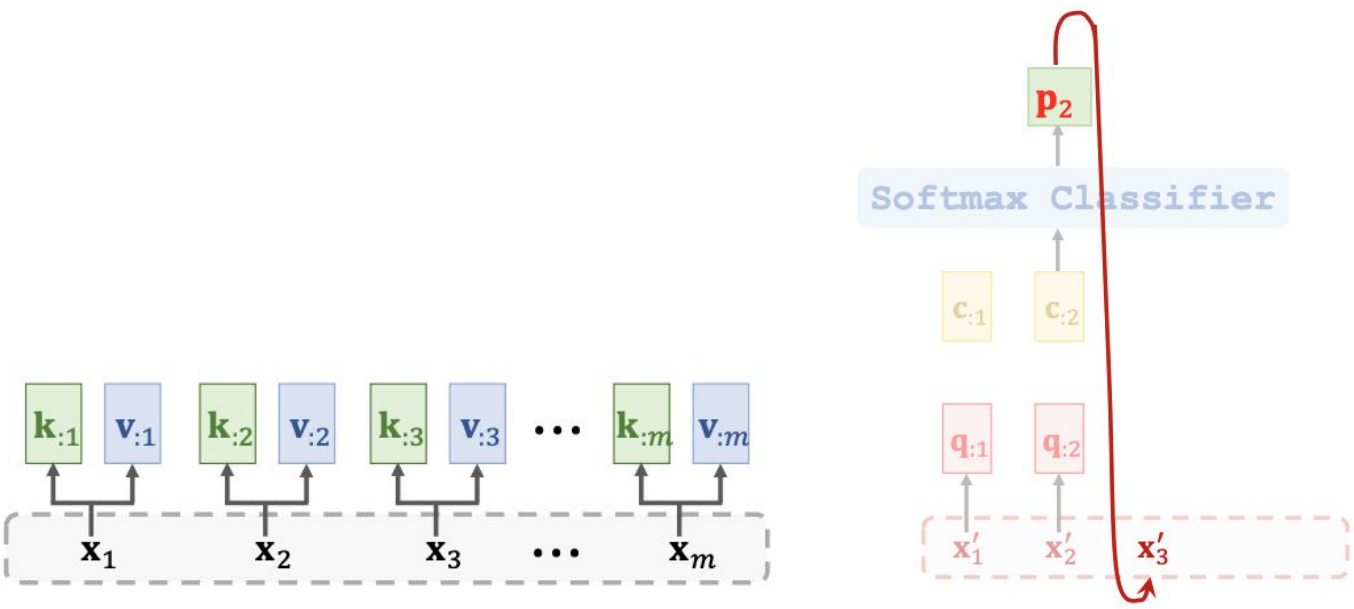
AutoRecursive Attention



$$x_{m+1}\ x_{m+2}, \ldots, x_{m+n}$$

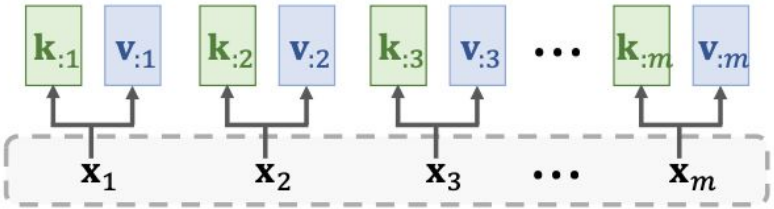
$$p(x_{m+n} | x_{<m+n})$$

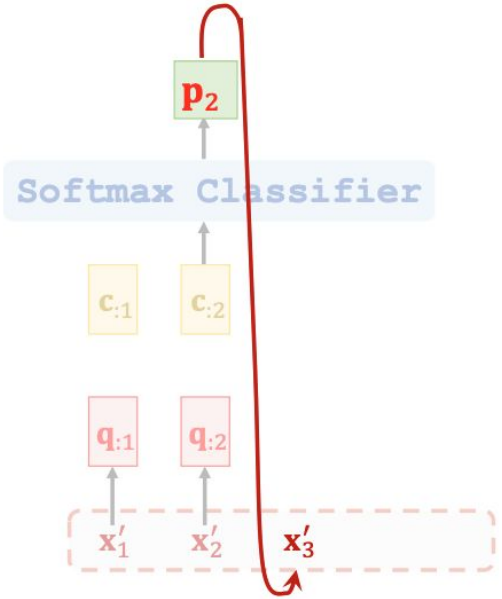
AutoRecursive Attention for Inference



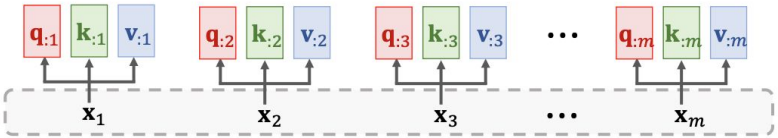
AutoRecursive Attention for Inference

Not flexible enough



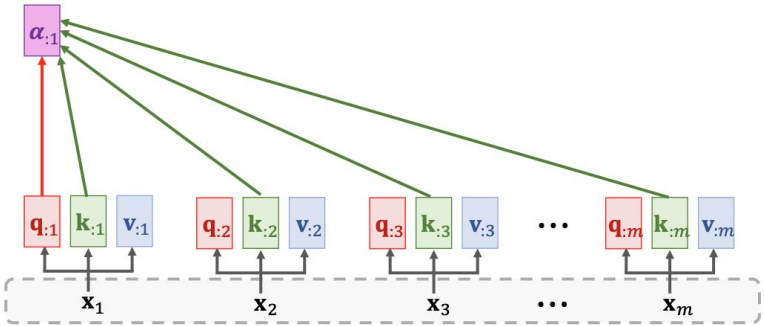


Transformer: Multi-Layer Multi-Headed Self-Attention!



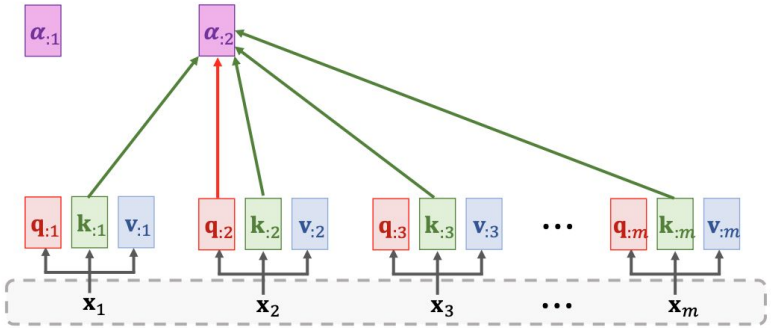
Self-Attention

Weights: $\boldsymbol{\alpha}_{:j} = \text{Softmax}\left(\mathbf{K}^T \mathbf{q}_{:j}\right) \in \mathbb{R}^m.$



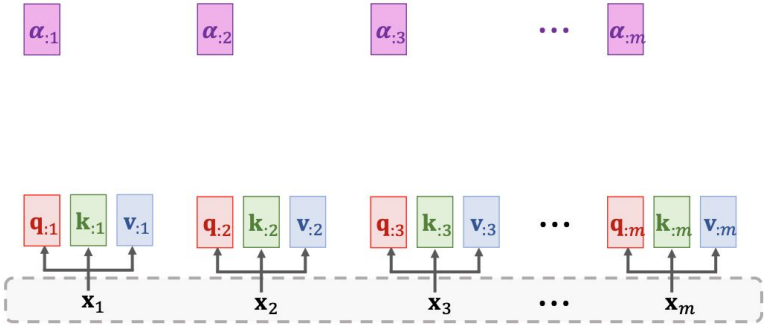
Self-Attention

Weights: $\boldsymbol{\alpha}_{:j} = \text{Softmax}\left(\mathbf{K}^T \mathbf{q}_{:j}\right) \in \mathbb{R}^m.$



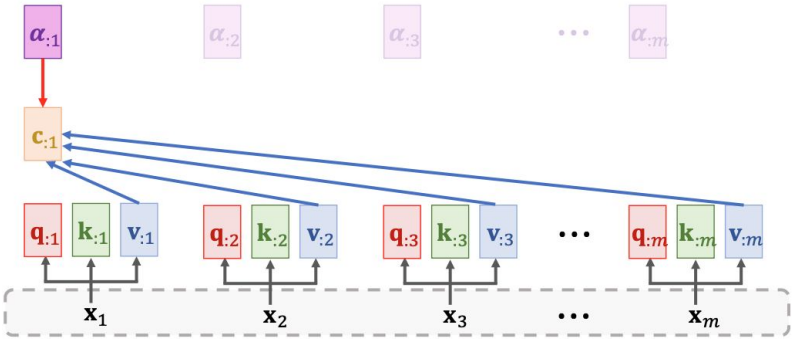
Self-Attention

Weights: $\boldsymbol{\alpha}_{:j} = \text{Softmax}\left(\mathbf{K}^T \mathbf{q}_{:j}\right) \in \mathbb{R}^m.$



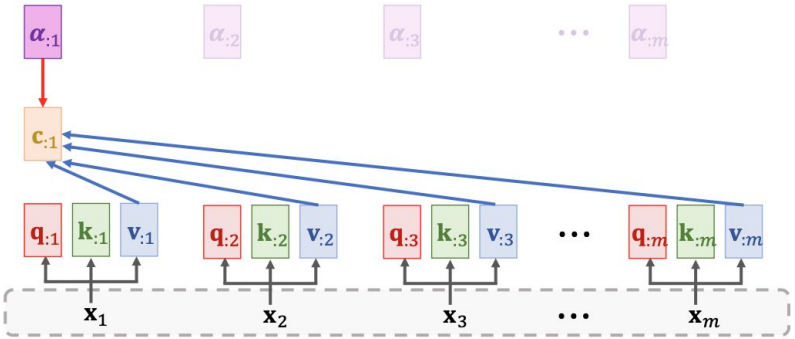
Self-Attention

Context vector: $\mathbf{c}_{:1} = \alpha_{11}\mathbf{v}_{:1} + \cdots + \alpha_{m1}\mathbf{v}_{:m} = \mathbf{V}\boldsymbol{\alpha}_{:1}.$



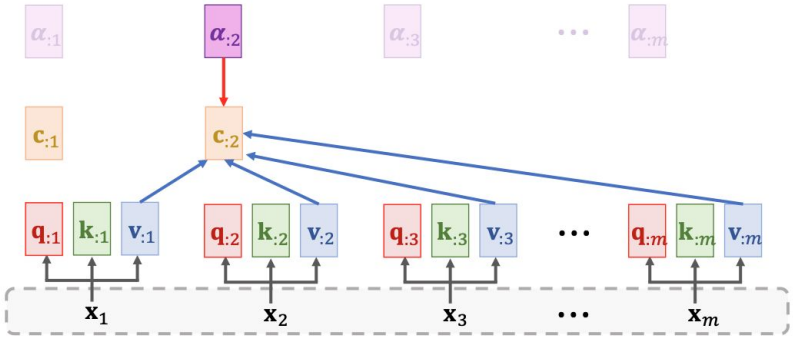
Self-Attention

Context vector: $\mathbf{c}_{:1} = \alpha_{11}\mathbf{v}_{:1} + \cdots + \alpha_{m1}\mathbf{v}_{:m} = \mathbf{V}\boldsymbol{\alpha}_{:1}.$



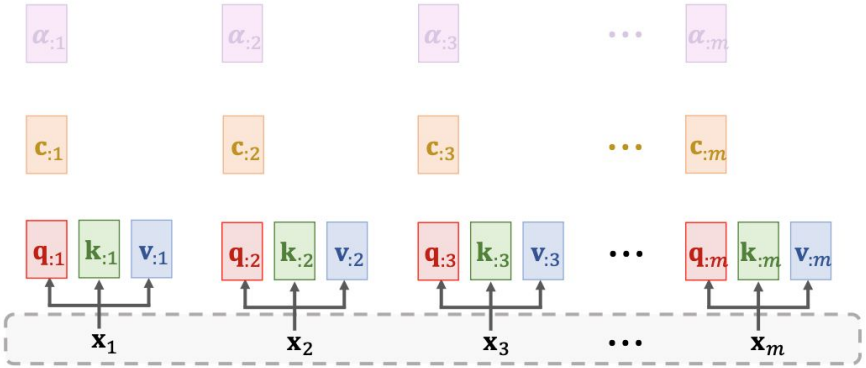
Self-Attention

Context vector: $\mathbf{c}_{:1} = \alpha_{11}\mathbf{v}_{:1} + \cdots + \alpha_{m1}\mathbf{v}_{:m} = \mathbf{V}\boldsymbol{\alpha}_{:1}.$



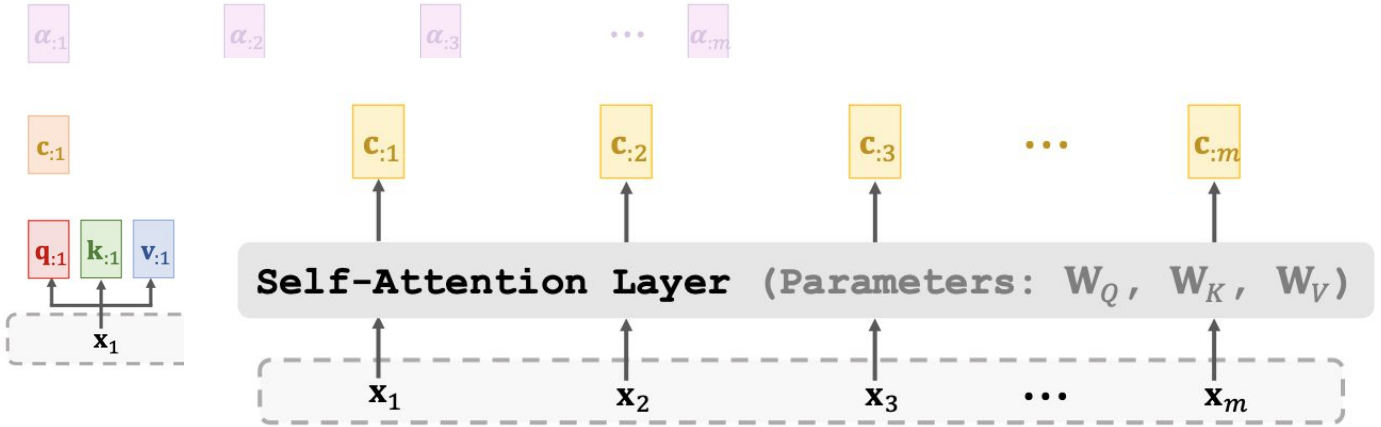
Self-Attention

Context vector: $\mathbf{c}_{:1} = \alpha_{11}\mathbf{v}_{:1} + \cdots + \alpha_{m1}\mathbf{v}_{:m} = \mathbf{V}\boldsymbol{\alpha}_{:1}.$

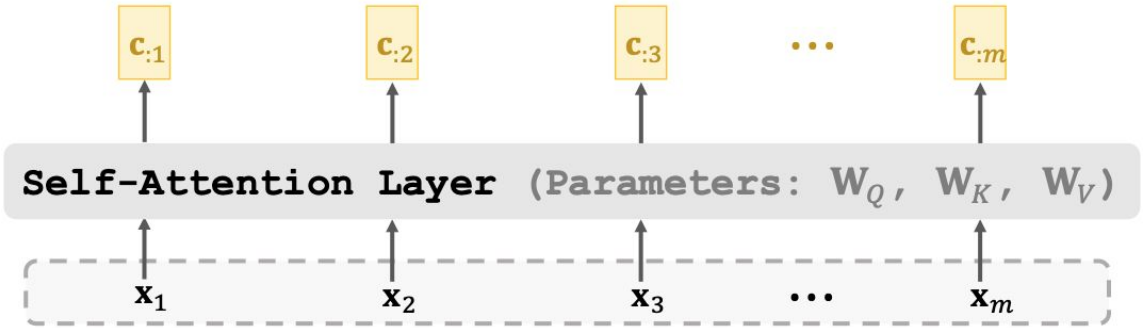


Self-Attention

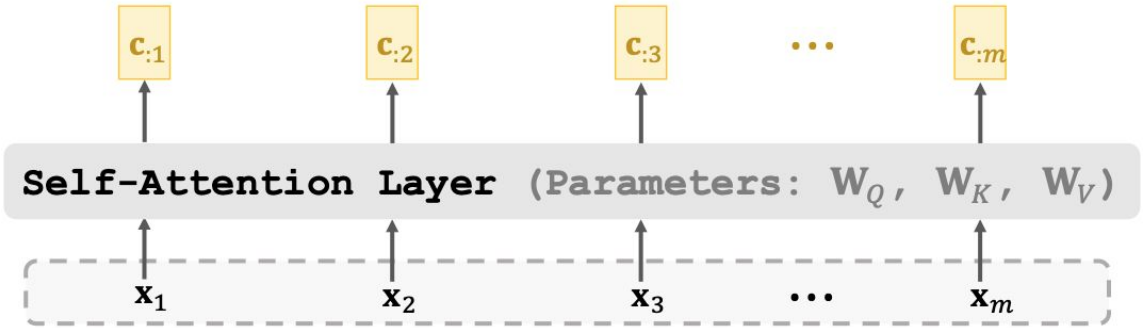
Context vector: $\mathbf{c}_{:1} = \alpha_{11}\mathbf{v}_{:1} + \cdots + \alpha_{m1}\mathbf{v}_{:m} = \mathbf{V}\boldsymbol{\alpha}_{:1}.$



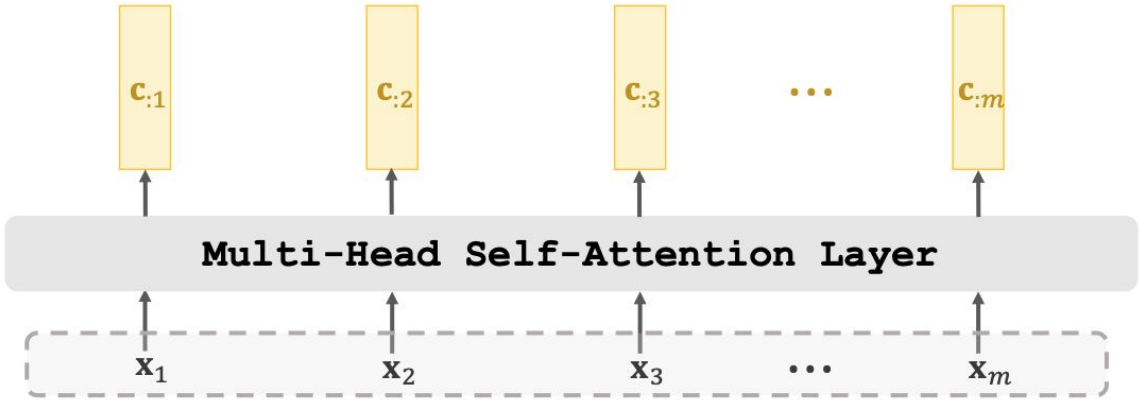
Transformer: Multi-Layer Multi-Headed Self-Attention!



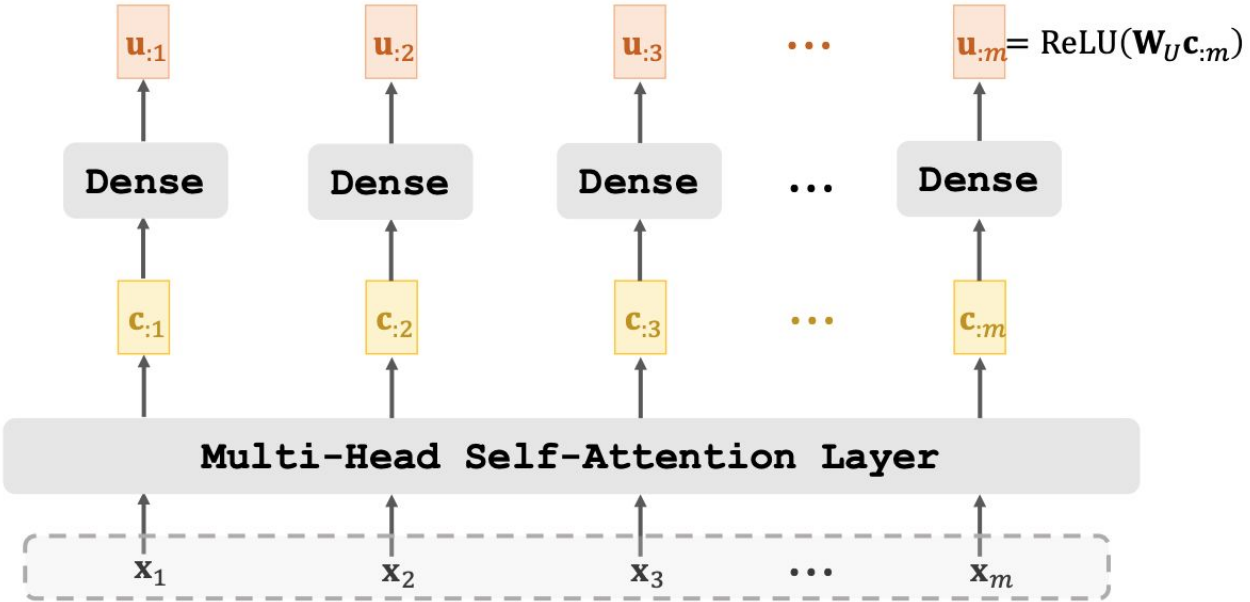
Multi-Headed Self-Attention



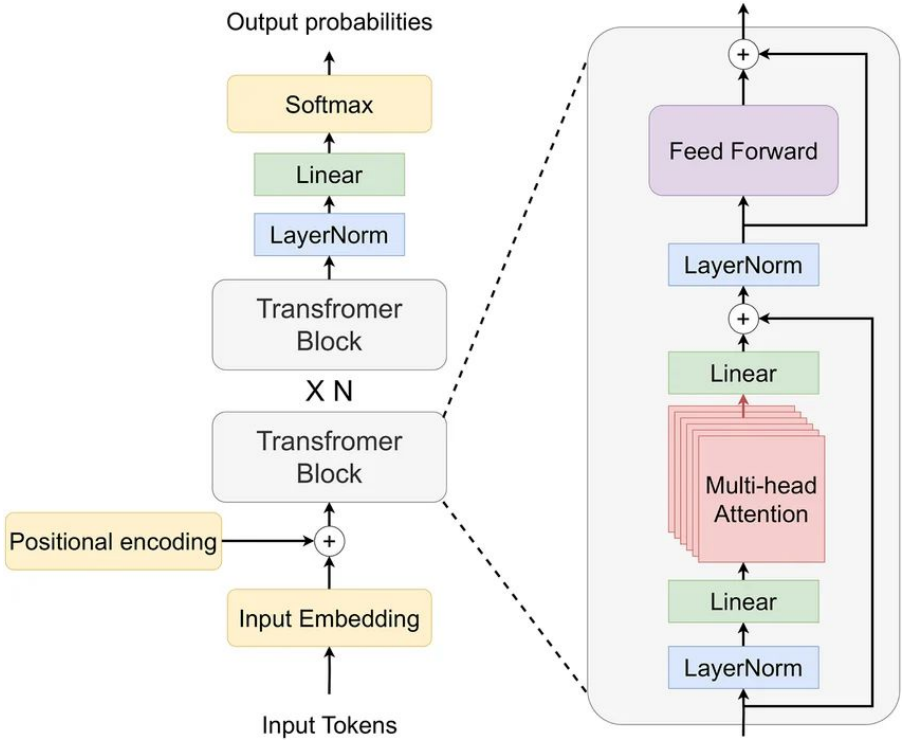
Multi-Headed Self-Attention



Multi-Headed Self-Attention



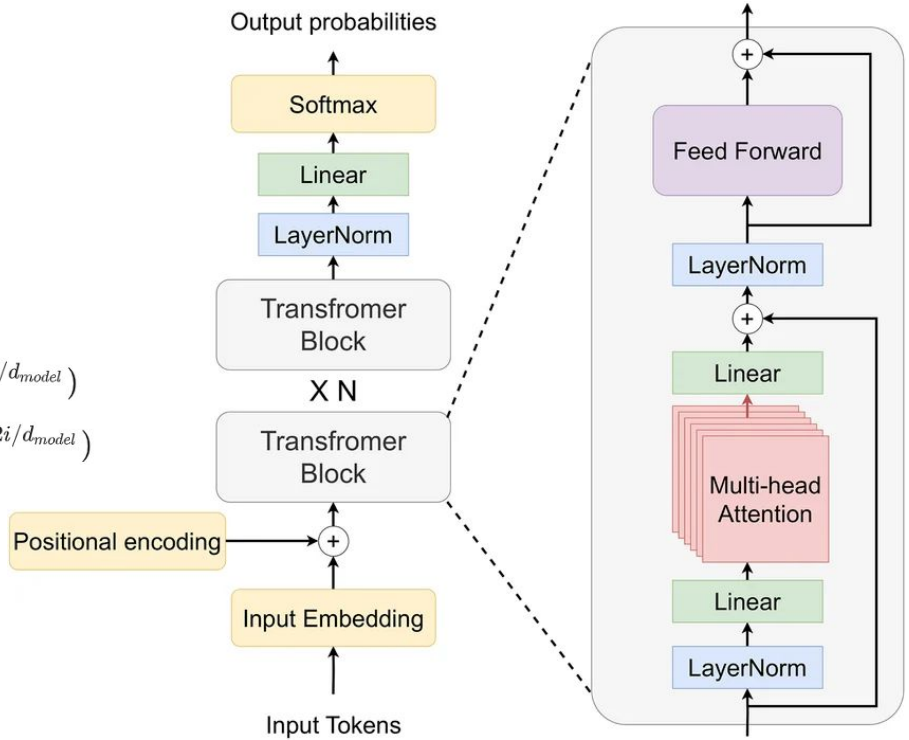
Transformer: Multi-Layer Multi-Headed Self-Attention!



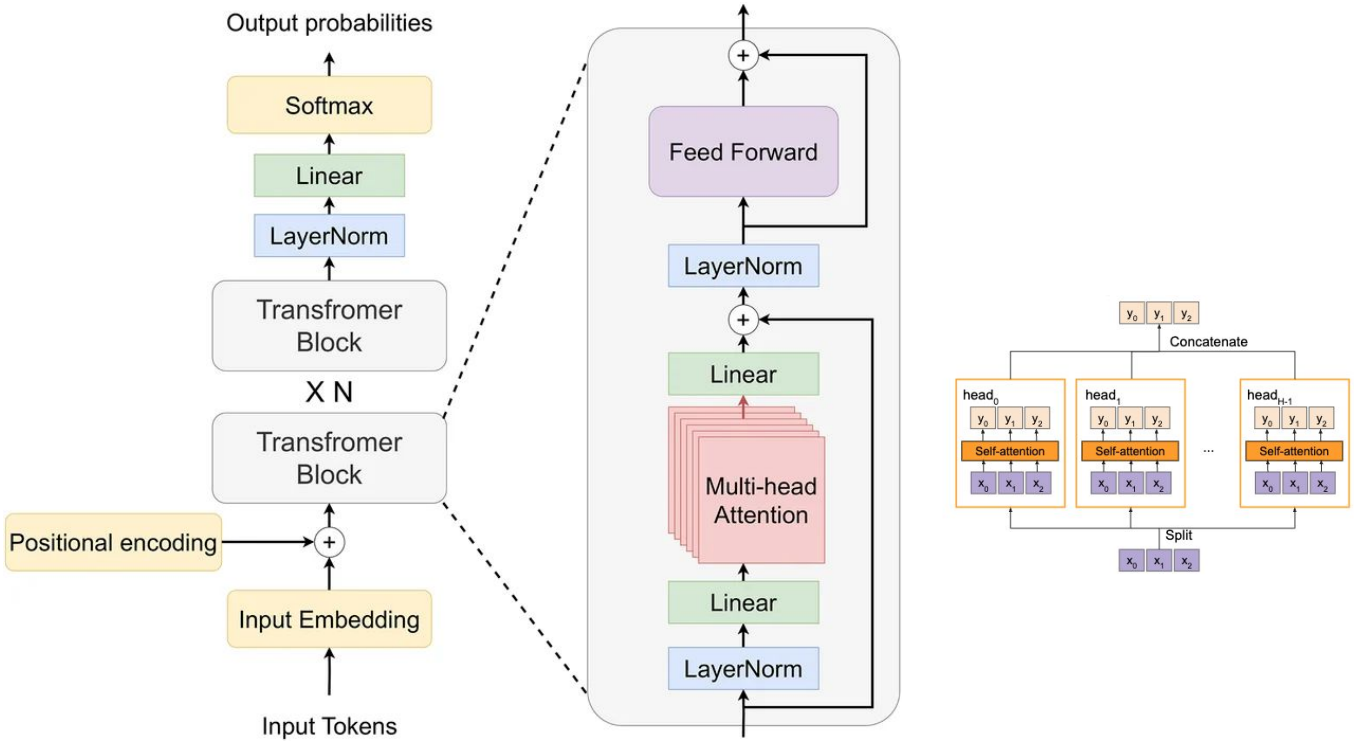
Transformer: Multi-Layer Multi-Headed Self-Attention!

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

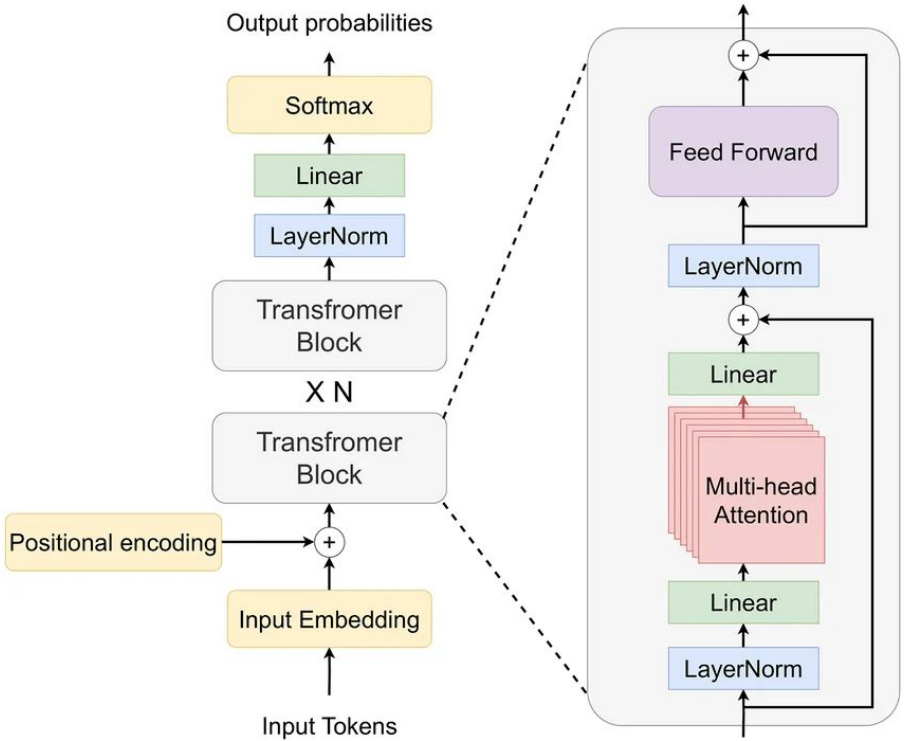
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$



Transformer: Multi-Layer Multi-Headed Self-Attention!



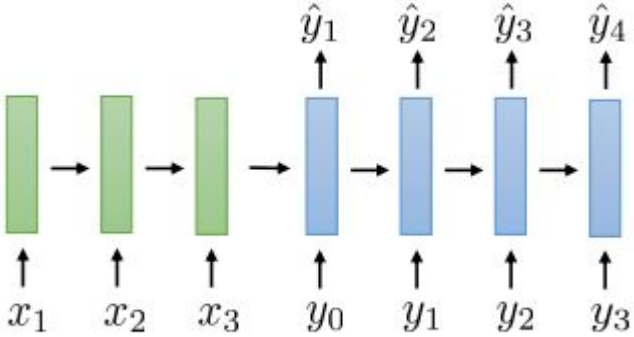
Transformer: Multi-Layer Multi-Headed Self-Attention!

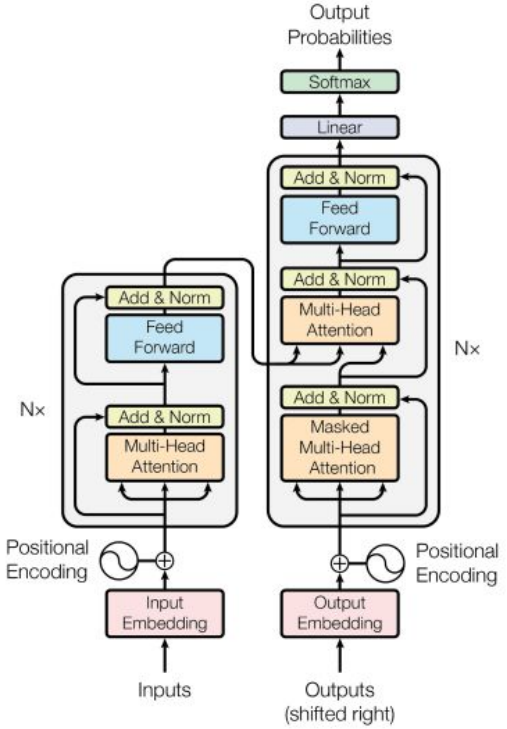


residual connect

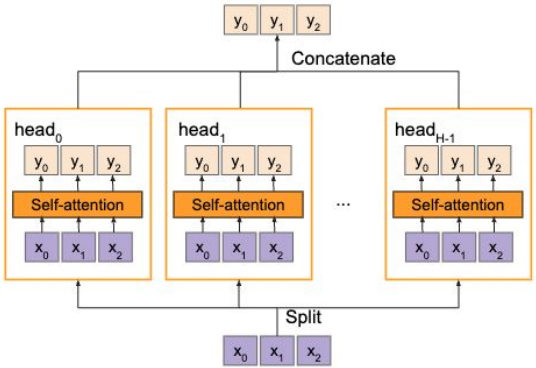
residual connect

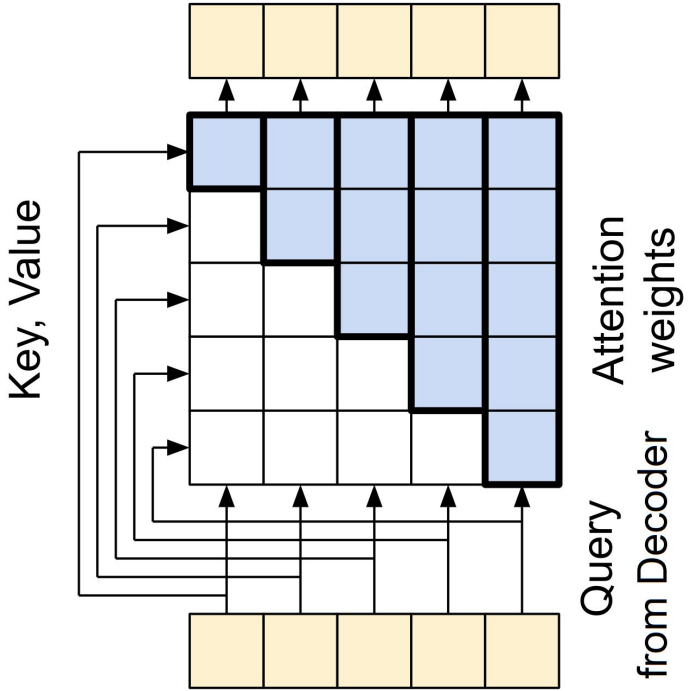
Original Transformer



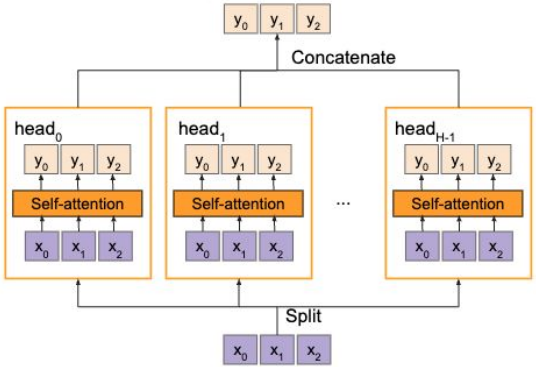


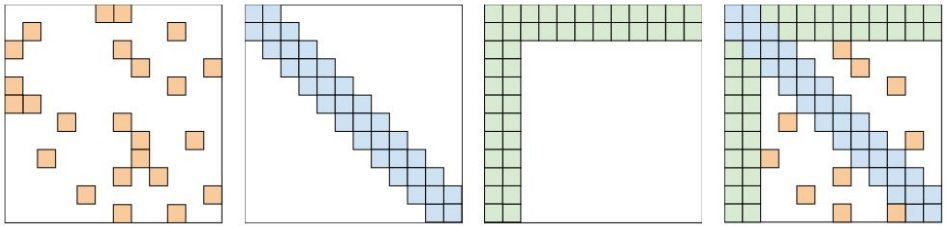
Limitations? Expensive Computation

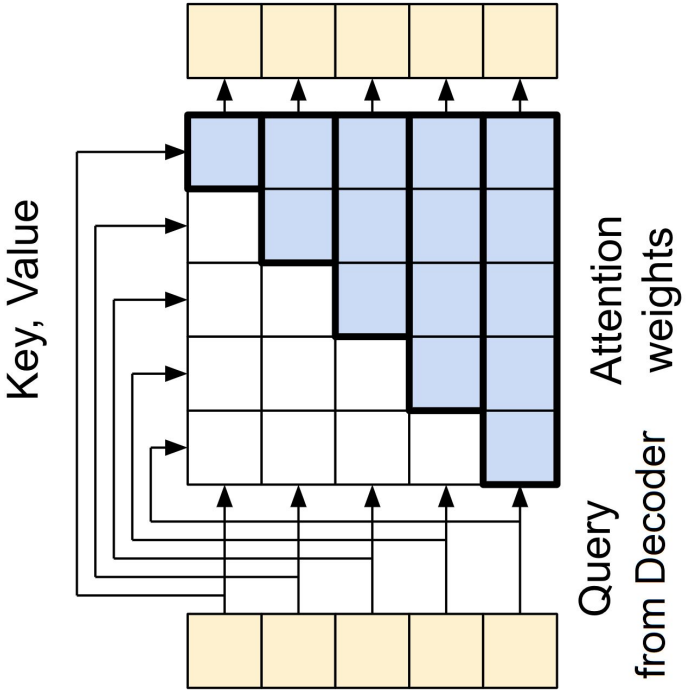




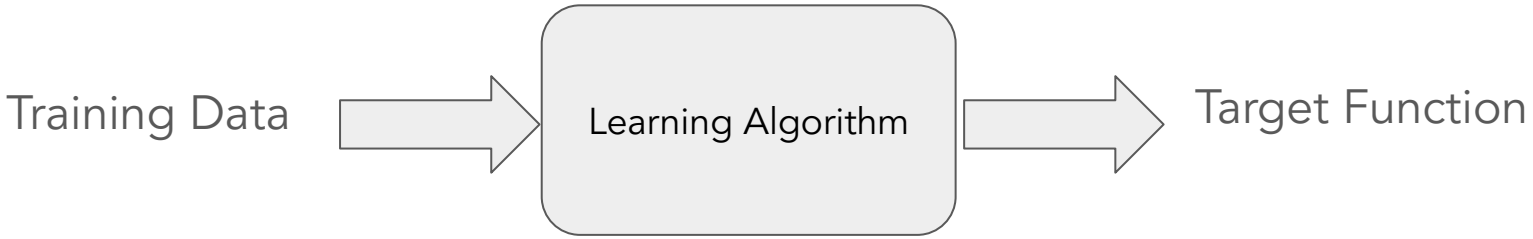
Limitations?




(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD



(Large) Language Models



$$D = \{X^i\}_{i=1}^n, \quad X^i = \{x_j^i\}_{j=1}^t$$

$$p(X)$$

1. Build probabilistic models
2. Derive loss function (by MLE or MAP….)
   MLE
3. Select optimizer

MLE

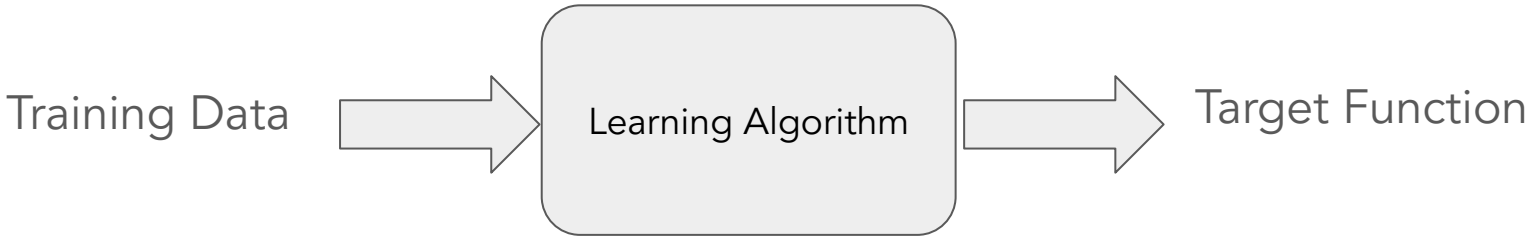
- Given all input data $D = \{X^i\}_{i=1}^n, \quad X^i = \{x_j^i\}_{j=1}^t$

$$p(X^i) = \prod_{j=1}^{t} \frac{\exp\left(W_{x_j^i}\phi\left(x_{<j}^i\right)\right)}{\sum_{l=1}^{V}\exp\left(W_l\phi\left(x_{<j}^i\right)\right)}$$

- Log-likelihood

$$\begin{aligned}
\ell(\phi) &= \sum_{i=1}^{n} p(X^i;\phi) = \sum_{i=1}^{n}\sum_{j=1}^{t}\log\left(x_j^i \mid x_{<j}^i;\phi\right) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{t}\log\frac{\exp\left(W_{x_j^i}\phi\left(x_{<j}^i\right)\right)}{\sum_{l=1}^{V}\exp\left(W_l\phi\left(x_{<j}^i\right)\right)} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{t} W_{x_j^i}\phi\left(x_{<j}^i\right) - \sum_{i=1}^{n}\sum_{j=1}^{t}\log\sum_{l=1}^{V}\exp\left(W_l\phi\left(x_{<j}^i\right)\right)
\end{aligned}$$

(Large) Language Models



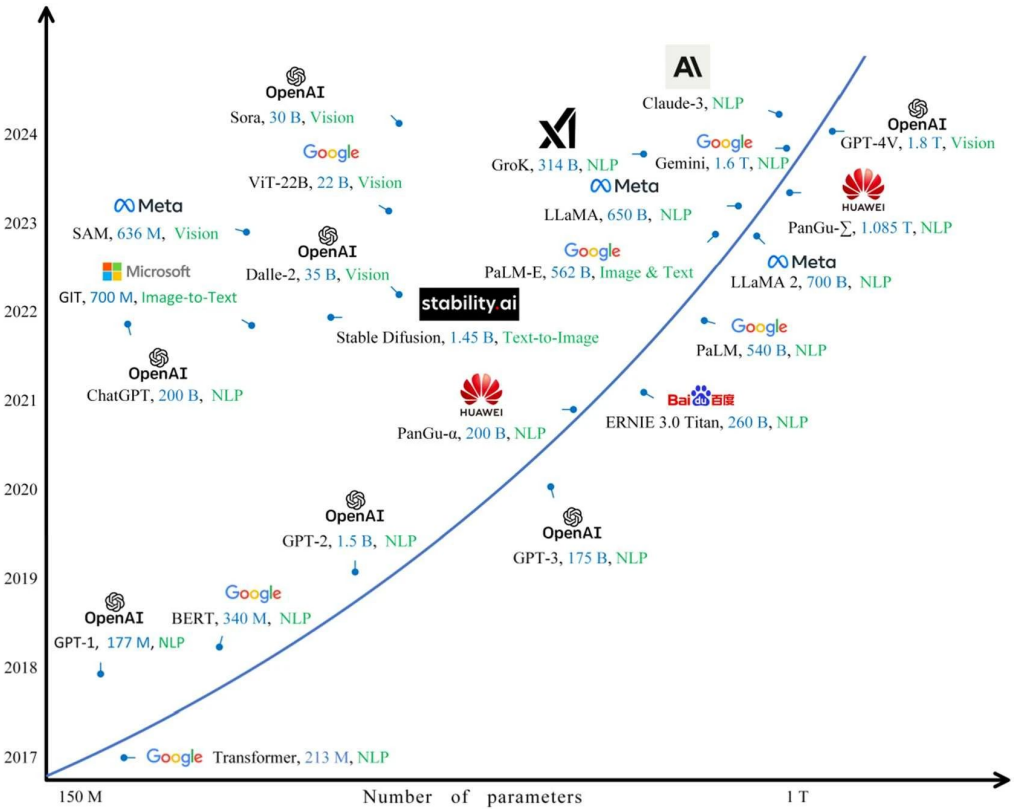
$$D = \{X^i\}_{i=1}^n, \quad X^i = \{x_j^i\}_{j=1}^t$$

$$p(X)$$

1. Build probabilistic models
2. Derive loss function (by MLE or MAP….)
3. Select optimizer

   Stochastic Gradient Descent

Large and Larger LMs



Tu, Xiaoguang, et al. "An overview of large AI models and their applications." *Visual Intelligence* 2.1 (2024): 1-22.

Summary

- Unification is the path to AGI
  - One model for every task

Summary

- Unification is the path to AGI
  - One model for every task
- Pretraining of LLM: MLE for unsupervised learning
- RNNs have bottleneck, restricting the flexibility
- Attention is actually flattening RNN
- Transformer is deep attention

Q&A

Presentation Sign-up deadline today!

Group 7, 11, 18, and 21