

注意機構とトランスフォーマー

本谷 秀堅

名古屋工業大学

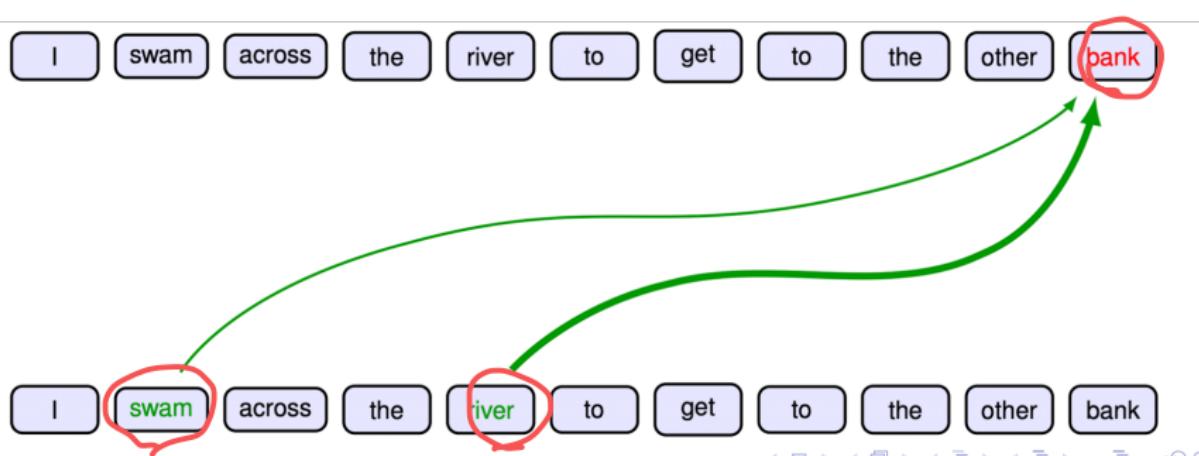
Transformer (トランスフォーマー)

注意機構 (Attention) を核とするニューラルネットワーク

言語データ（再掲）

言語データ

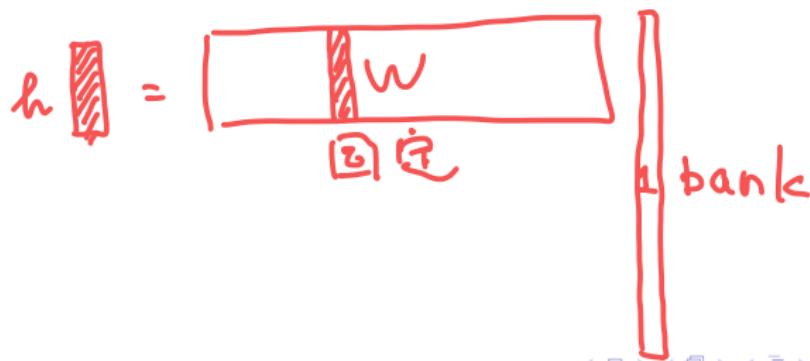
- ・シンボルの系列
- ・離れたシンボル間に強い関係
- ・シンボル間の非対称な関係（例：対義語・類義語・包含関係）



Word2Vecによる単語ベクトルの埋め込み

- 埋め込み先は文脈に依存せずに同じ
 - bank を表すベクトルは、堤のときも銀行のときも同じ

$$h = Wx$$



言語データと分布仮説（再掲）

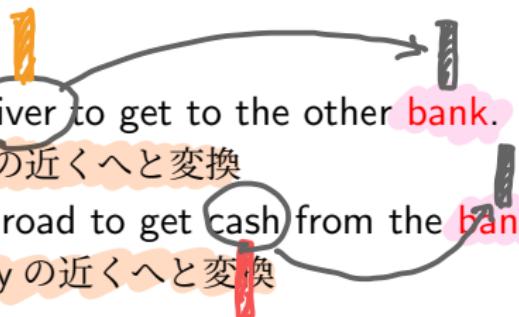
分布仮説

単語の意味はその単語の周辺に現れる単語、すなわち文脈 (context) によって決まる

各単語を更新する。

注意機構の目的

各単語の変換先を、系列中の他の単語に依存して変化させたい

- 
- The diagram illustrates word embeddings in a 2D space. Two vectors originate from the origin: one labeled 'river' pointing upwards and to the left, and another labeled 'bank' pointing downwards and to the right. A curved arrow points from the 'river' vector towards the 'bank' vector, indicating a dependency or transformation relationship between them.
- I swam across the river to get to the other bank.
 - “bank” は river の近くへと変換
 - I walked across the road to get cash from the bank.
 - “bank” は money の近くへと変換

注意機構により文脈に依存して各単語の埋め込み先を変化させる

- 特徴ベクトルのことを「トークン (token)」と呼ぶ。

方針

文脈中のすべての単語を参照して、各単語の埋め込み先を決める

Word2Vecで実現

- 入力： N 単語。それぞれ D 次元のベクトル（トークン）で表現

$$x_1, x_2, \dots, x_N \quad (x_n \in \mathbb{R}^D)$$

- 出力：各単語の変換後の D 次元トークン

$$y_1, y_2, \dots, y_N \quad (y_n \in \mathbb{R}^D)$$

文脈中の各単語の荷重平均を埋め込み先とする

$$y_n = \sum_{m=1}^N a_{nm} \overset{\odot}{=} x_m$$

- 文中の全 n の単語

→ 単語 n と

単語 m の関係の強さ

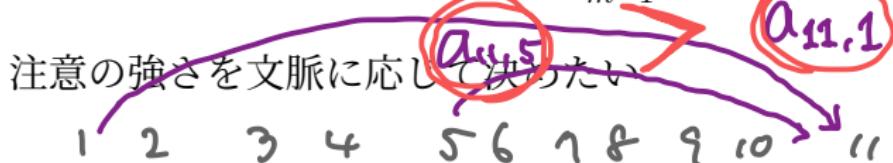
を読み取る前から何番目

注意プーリング (Attention Pooling) ⑨

注意プーリング (pooling: 集約するの意)

a_{nm} : 注意の強さ ($a_{nm} > 0$, $\sum_{m=1}^N a_{nm} = 1$)

$$y_n = \sum_{m=1}^N a_{nm} x_m$$



- I swam across the river to get to the other bank.
 - "bank" は river と似た方向へと変換
- I walked across the road to get cash from the bank.
 - "bank" は money と似た方向と変換

記法

→ 違ふ書き

- N 単語
- 各単語はベクトルで表現 $x_n \in \mathbb{R}^D$
- 文脈は行列で表現 : $\mathbf{X} \in \mathbb{R}^{N \times D}$

$$\mathbf{X} = [x_1, x_2, \dots, x_n]^\top$$

- 出力も行列で表現 : $\mathbf{Y} \in \mathbb{R}^{N \times D}$

$$\mathbf{Y} = [y_1, y_2, \dots, y_n]^\top$$

500 < 511

トーンの次元

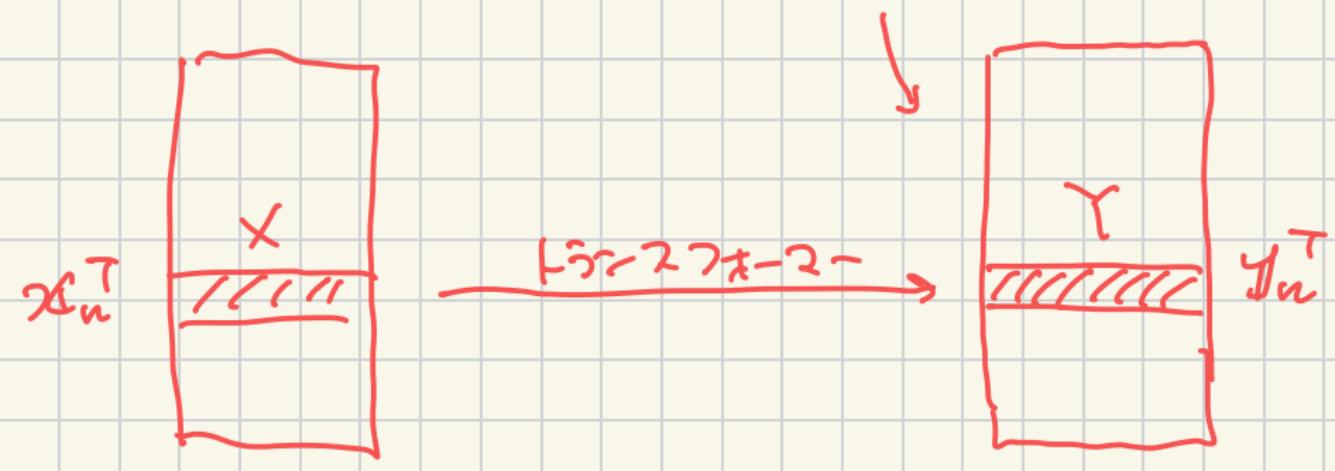
文書の次元

N (tokens)

x_n^T

\mathbf{X}

D (features)



注意の大きさを文脈で決める（案1）

もともと似たベクトルどうしを強い重みで

$$a_{nm} = \frac{\exp(\mathbf{x}_n^\top \mathbf{x}_m)}{\sum_k \exp(\mathbf{x}_n^\top \mathbf{x}_k)}$$



$$\mathbf{A} = \text{SM}(\mathbf{X} \mathbf{X}^\top) \in \mathbb{R}^{N \times N},$$

$$\mathbf{Y} = \mathbf{A} \mathbf{X} = \text{SM}(\mathbf{X} \mathbf{X}^\top) \in \mathbb{R}^{N \times N}.$$

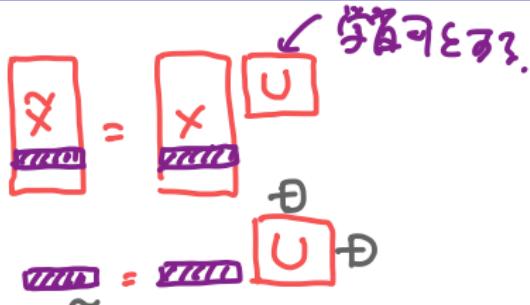
似たベクトル
似た意味

- トークン間の類似度を内積で評価する場合の式
- 行列 \mathbf{A} の (n, m) 要素が a_{nm}
- Softmax は各行で計算
- $\mathbf{X} \mathbf{X}^\top$ は対称行列
- a_{nm} に「学習」の余地無し。柔軟性に欠ける。

注意の大きさを文脈で決める（案2）

注意の大きさの決め方を学習する

$$\tilde{\mathbf{X}} = \mathbf{XU}$$

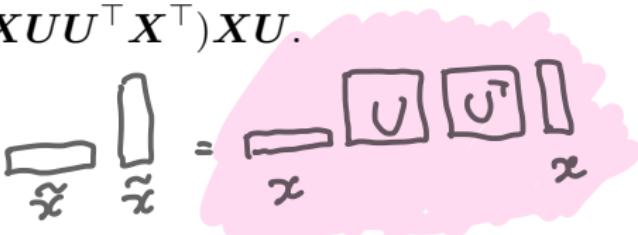


上記 $\mathbf{U} \in \mathbb{R}^{D \times D}$ を学習する

$$\mathbf{A} = \text{SM}(\mathbf{XUU}^\top \mathbf{X}^\top),$$

$$\mathbf{Y} = \text{SM}(\mathbf{XUU}^\top \mathbf{X}^\top) \mathbf{XU}.$$

- 案1より柔軟
- $\mathbf{XUU}^\top \mathbf{X}^\top$ は対称行列 (This item is circled in red)
- Softmax 関数を適用すると対称ではなくなるが大小関係は維持される
- 包含関係、対義語、類義語、指示代名詞など複雑な関係の表現には自由度が不足

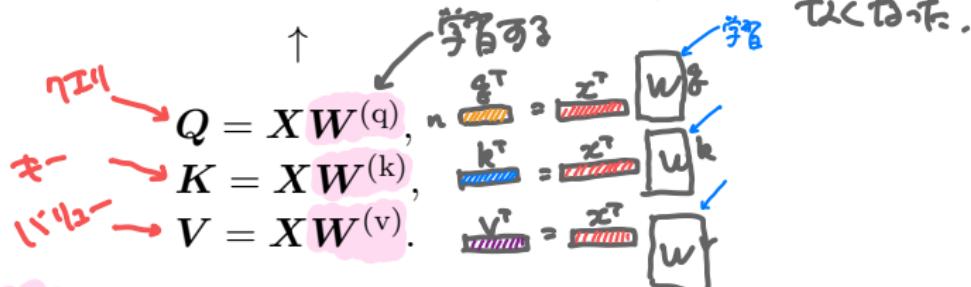




注意の大きさの決め方：クエリ・キー・バリューの考え方を採用する

$$a_{nm} = \frac{\exp(\mathbf{q}_n^\top \mathbf{k}_m)}{\sum_k \exp(\mathbf{q}_n^\top \mathbf{k}_k)}$$

nとmが入力がめると
 $\mathbf{q}_n^\top \mathbf{k}_m$ 2-進化
→対称性



行列 $\mathbf{W}^{(q)}, \mathbf{W}^{(k)}, \mathbf{W}^{(v)}$ を学習により決める。
 $\mathbf{W}^{(q)}, \mathbf{W}^{(k)} \in \mathbb{R}^{D \times D_k}, \mathbf{W}^{(v)} \in \mathbb{R}^{D \times D_v}$

$$\mathbf{Y} = \text{SM}(\mathbf{Q}\mathbf{K}^\top)\mathbf{V}$$

(復習) クエリ・キー・バリュー

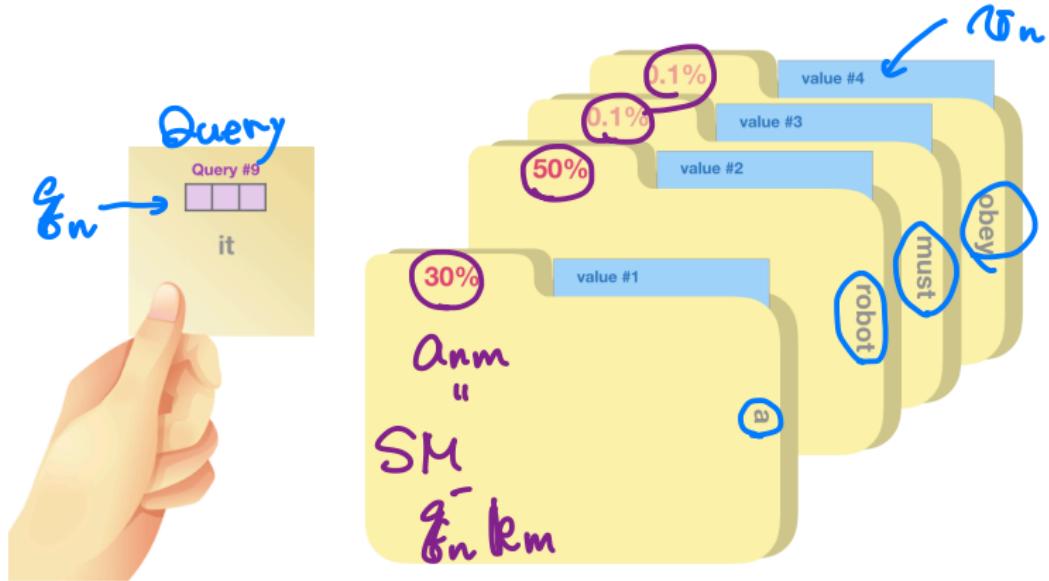
- キー (key) とバリュー (value: 値) の組のデータ集合
- クエリ (query) による検索

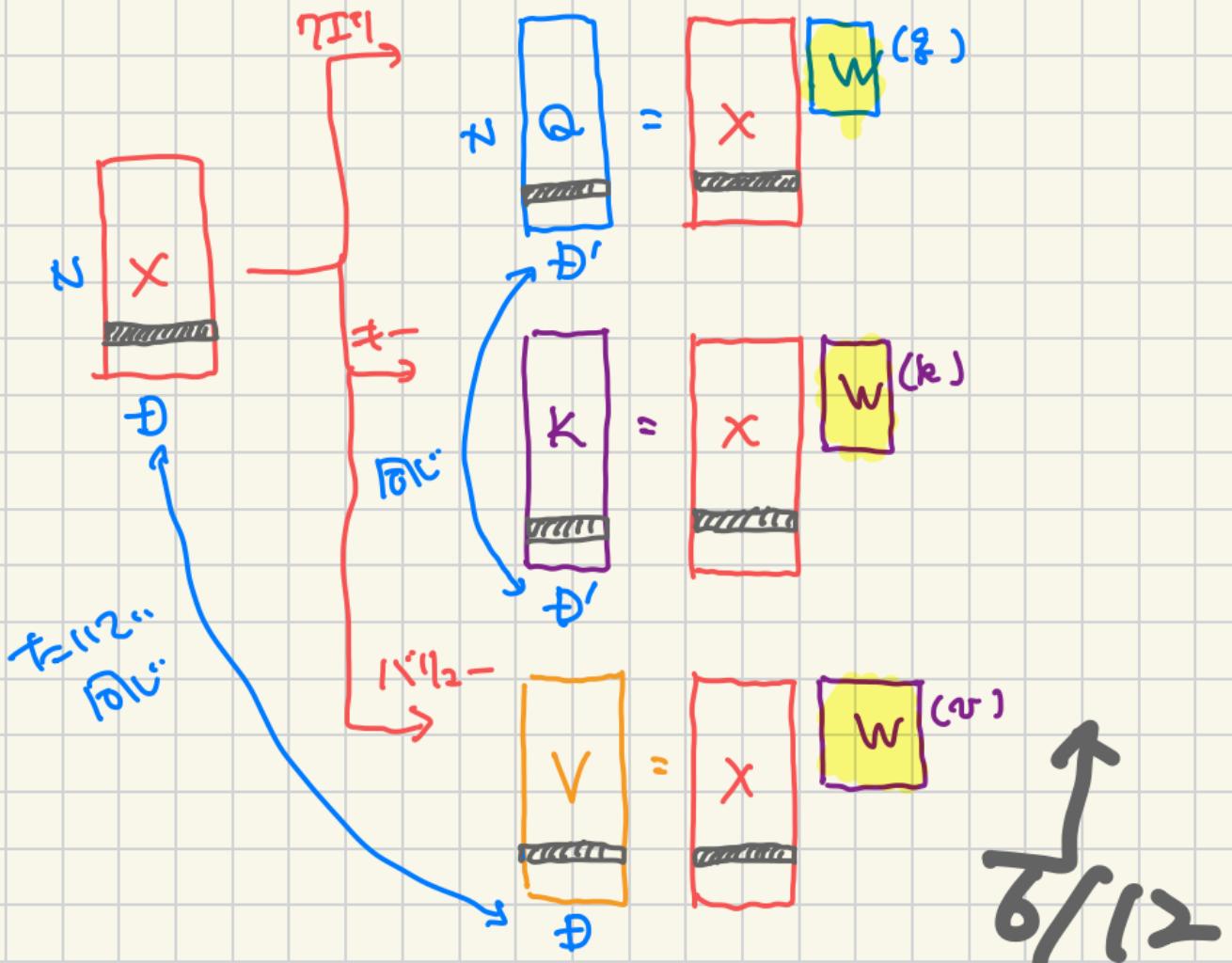
例：

- データ集合:
 $\mathcal{D} = \{(\text{夏目}, \text{漱石}), (\text{森}, \text{鴎外}), (\text{中島}, \text{敦}), (\text{松尾}, \text{芭蕉})\}$
- 検索 : query = “森”
検索結果 : value = “鴎外”

query と key が類似度 → 検索
value → 結果

(復習) クエリ・キー・バリュー





注意機構 (1/2)

$$\mathbf{Y} = \text{SM}(\mathbf{Q}\mathbf{K}^{\top})\mathbf{V}$$

① 行列 $\mathbf{W}^{(q)}, \mathbf{W}^{(k)}$ を学習する

- key-vector の次元を D_k で表すと、いずれも $\mathbb{R}^{D \times D_k}$

② $\mathbf{Q} = \mathbf{X}\mathbf{W}^{(q)}, \mathbf{K} = \mathbf{X}\mathbf{W}^{(k)}$

- n 番目の単語のクエリ (\mathbf{q}_n) とキー (\mathbf{k}_n) は次のとおり

$$\mathbf{q}_n^{\top} = \mathbf{x}_n^{\top} \mathbf{W}^{(q)}, \quad \mathbf{k}_n^{\top} = \mathbf{x}_n^{\top} \mathbf{W}^{(k)}$$

③ n 番目の単語を変換するために $m = 1, 2, \dots, N$ 番目の単語を参照

$$\mathbf{q}_n^{\top} \mathbf{k}_m = (\mathbf{Q}\mathbf{K}^{\top})_{nm}$$

④ $\mathbf{Q}\mathbf{K}^{\top}$ の各行で Softmax を計算

$$a_{nm} = \frac{\exp(\mathbf{q}_n^{\top} \mathbf{k}_m)}{\sum_k \exp(\mathbf{q}_n^{\top} \mathbf{k}_k)}$$

注意機構 (2/2)

$$y_n = \sum_{m=1}^n \overbrace{a_{nm}}^{\text{attention weight}} v_m$$
$$Y = \underline{\underline{SM(QK^\top)V}}$$

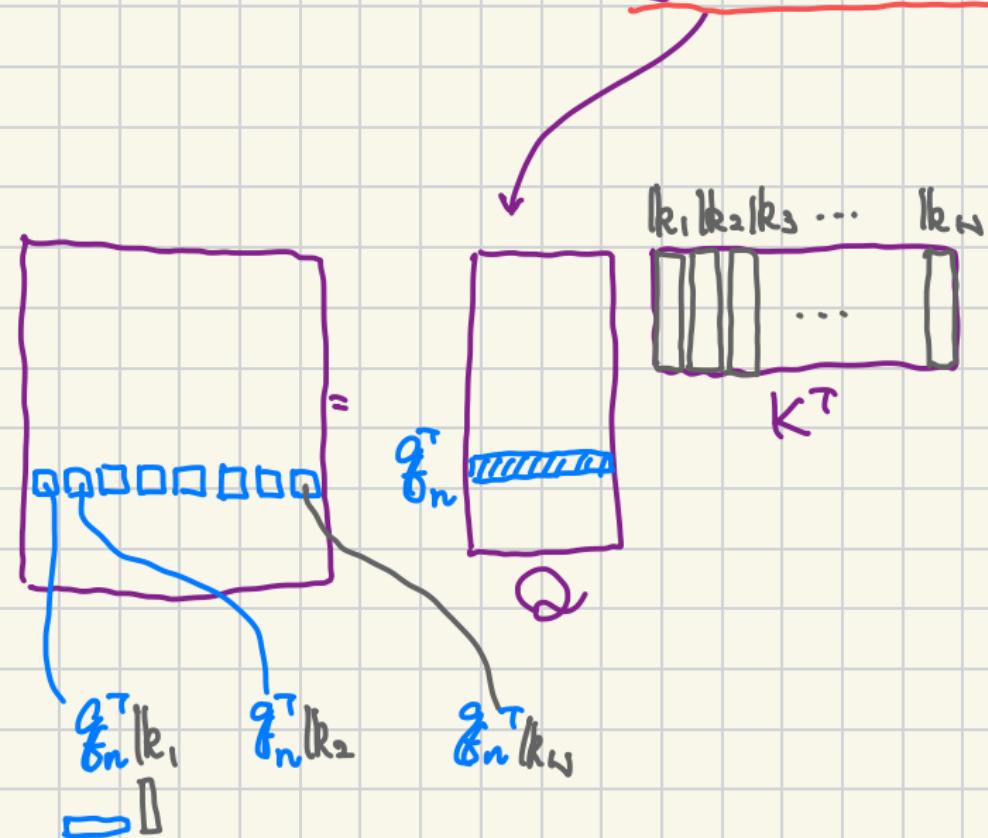
- 行列 $\mathbf{W}^{(v)} \in \mathbb{R}^{D \times D}$ を学習する
- バリュ－ (value) の計算 : $\mathbf{V} = \mathbf{X}\mathbf{W}^{(v)}$
 - n 番目の単語のバリュ－, v_n

$$v_n^\top = \mathbf{x}_n^\top \mathbf{W}^{(v)}$$

- トークンの更新
 - n 番目の単語更新用の注意 : $a_n^\top = [a_{n1}, a_{n2}, \dots, a_{nN}]$
 - n 番目の単語のトークンの更新

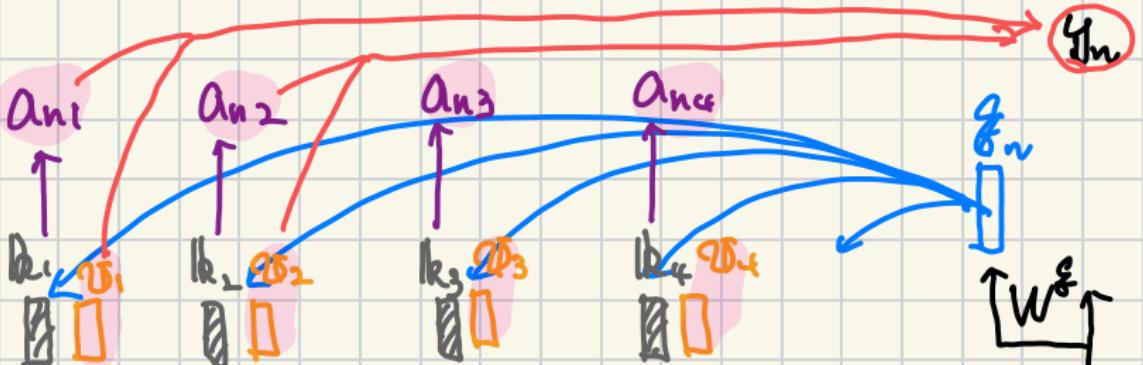
$$y_n^\top = a_n^\top \mathbf{V}$$

$SM(QK^T)$



$SM(QK^T)$

$$= SM \left(\begin{array}{c|ccccc} & & & & & N \\ & & \vdots & & & \\ & & a_{11} & a_{12} & \dots & a_{1N} \\ \hline & & a_{N1} & a_{N2} & \dots & a_{NN} \end{array} \right) \quad \begin{array}{c} V \\ \vdots \\ \varphi_N^T \\ \vdots \\ \varphi_2^T \\ \varphi_1^T \end{array}$$
$$\frac{\exp(u_n)}{\sum_{n'} \exp(u_{n'})} \quad \begin{array}{l} \text{OKT} \\ \text{EGLI} \end{array}$$
$$\boxed{y_n^T} = \sum a_{nm} \boxed{w_m^T}$$



I swam across the river to get to...

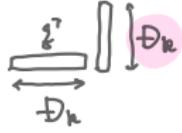
Xn

語順無限性。

注意のスケールの調整

\exp の中の絶対値が大きくなることを防ぐ。(絶対値が大きいと勾配がゼロに近づき学習に失敗する。)

$$a_{nm} = \frac{\exp(\mathbf{q}_n^\top \mathbf{k}_m)}{\sum_k \exp(\mathbf{q}_n^\top \mathbf{k}_k)}$$

\downarrow 

$$a_{nm} = \frac{\exp\left(\frac{\mathbf{q}_n^\top \mathbf{k}_m}{\sqrt{D_k}}\right)}{\sum_k \exp\left(\frac{\mathbf{q}_n^\top \mathbf{k}_k}{\sqrt{D_k}}\right)}$$

D_k は \mathbf{q}, \mathbf{k} の次元

まとめ：注意機構 ⑩

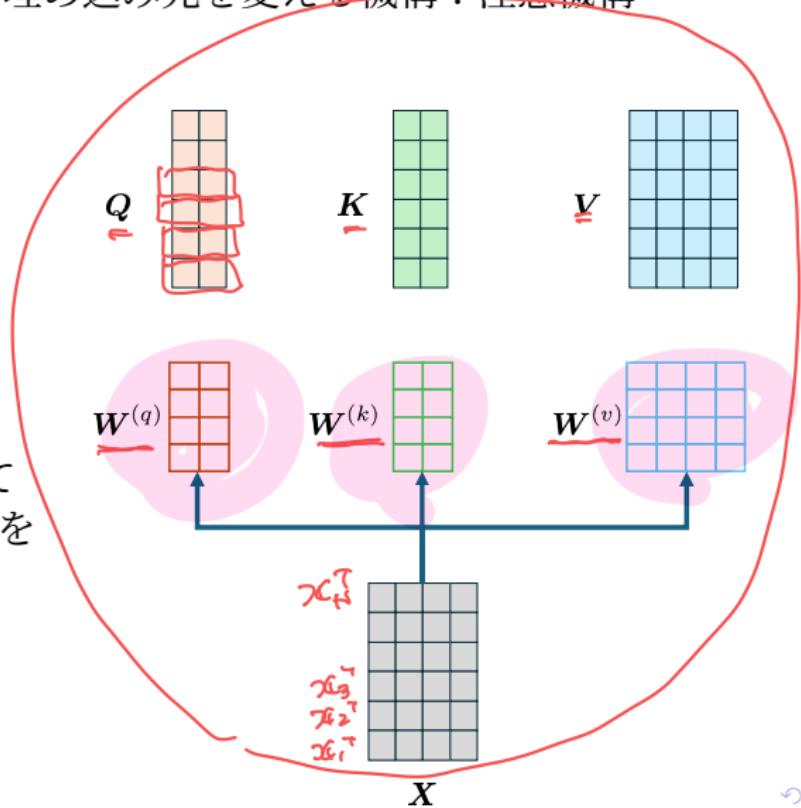
$$\mathbf{Y} = \text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{SM}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_k}}\right)\mathbf{V}$$

Attention Head

文脈に依存して埋め込み先を変える機構：注意機構

Attention Head とも呼ぶ

- 入力文全体を見渡して
「どこに注目するか」を
判断するモジュール

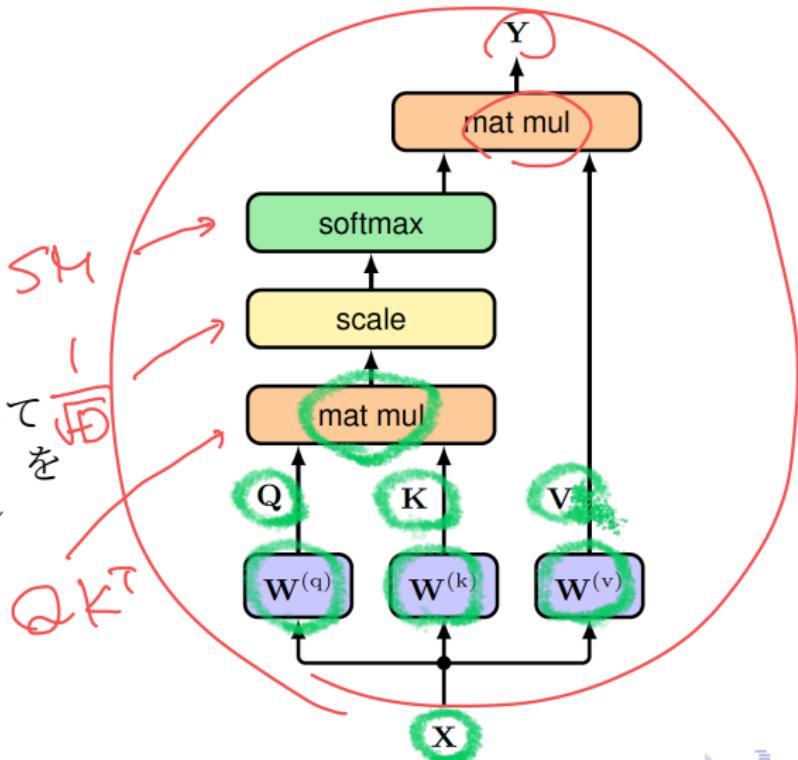


Attention Head

文脈に依存して埋め込み先を変える機構：注意機構

Attention Head とも呼ぶ

- 入力文全体を見渡して
「どこに注目するか」を
判断するモジュール

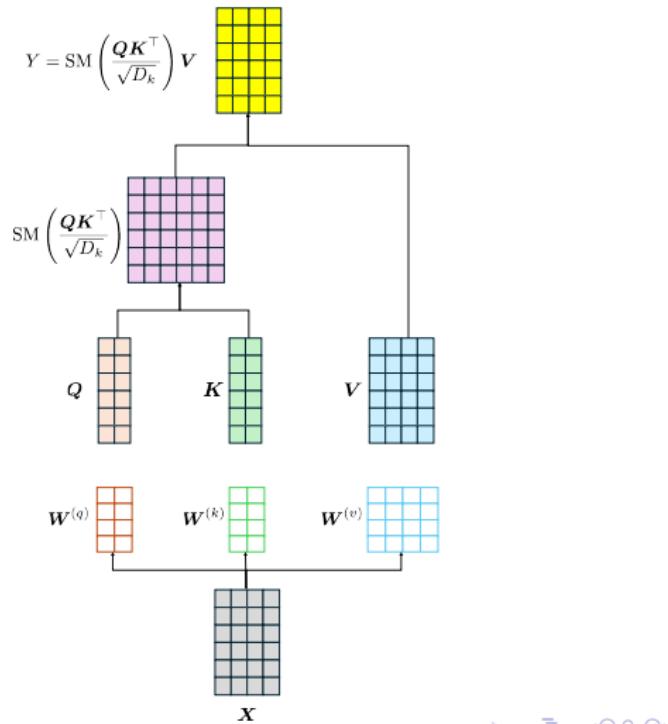


Attention Head

文脈に依存して埋め込み先を変える機構：注意機構

Attention Head とも呼ぶ

- 入力文全体を見渡して
「どこに注目するか」を
判断するモジュール



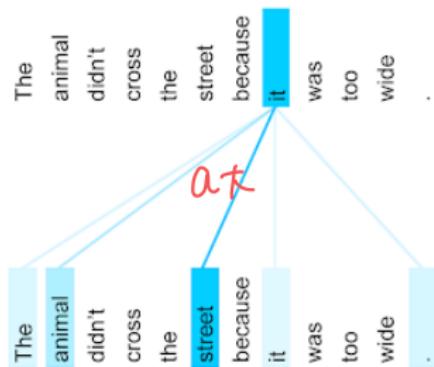
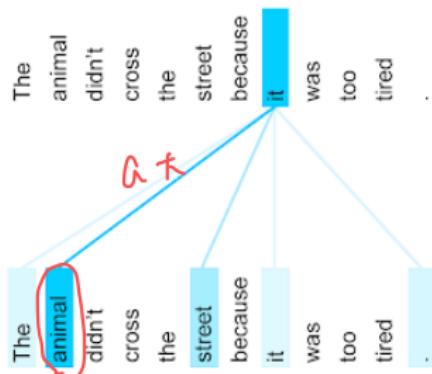
Attention Head で実現できること

- 文脈に依存して単語の埋め込み先を変える
- 注意の強弱を文脈に依存して決定

例

The animal did not cross the street because **it** was too tired.

The animal did not cross the street because **it** was too wide.



注意機構の性質

重要： $\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, $\mathbf{W}^{(v)}$ が注意の規準を定める (学習後は規準固定)

- 出力 y_n は入力 v_n の線形和

$$y_n = \sum_{m=1}^N a_{nm} v_m$$

Xに比べて大きい

- v_n は x_1, x_2, \dots, x_N の線形変換

$$\mathbf{V} = \mathbf{X}\mathbf{W}^{(o)}$$

- y_n は入力 x_1, x_2, \dots, x_N の線形変換

- 非線形性は注意の強度, a_{nm} , 経由で導入 (x_n に依存・Softmax)

注意機構の性質

重要： $\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, $\mathbf{W}^{(v)}$ が注意の規準を定める（学習後は規準固定）

- 出力 y_n は入力 v_n の線形和

$$y_n = \sum_{m=1}^N a_{nm} \underline{\underline{v}_m}$$

- v_n は、 $(\mathbf{W}^{(o)})^\top$ の列ベクトルの線形和

$$\underline{\underline{v}_n^\top} = \underline{\underline{x}_n^\top} \mathbf{W}^{(o)} \rightarrow \underline{\underline{v}_n} = (\mathbf{W}^{(o)})^\top \underline{\underline{x}_n}$$

$\mathbf{W}^{(o)}$ の列ベクトル
が「既に空き」
 $\mathbf{W}^{(o)}$ の値域が
限定される

つまり

$$v_n = \sum_{i=1}^D x_{ni} w_i^{(o)}$$

$$\underline{\underline{v}_n} = \begin{matrix} | \\ | \\ | \\ | \end{matrix} = \begin{matrix} | & | & | & | \\ w_1 & w_2 & \cdots & w_D \end{matrix} \underline{\underline{x}_n}$$

$$= \sum_m x_{nm} \underline{\underline{w}_m}$$

ただし、 $x_n = [x_{n1}, x_{n2}, \dots, x_{nD}]^\top$ 、

$$(\mathbf{W}^{(o)})^\top = [w_1^{(o)}, w_2^{(o)}, \dots, w_D^{(o)}].$$

注意機構の性質

重要： $\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, $\mathbf{W}^{(v)}$ が注意の規準を定める (学習後は規準固定)

- 出力 \mathbf{y}_n は入力 $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_D$ が張る部分空間に限定される

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

- 出力 \mathbf{y}_n は入力 \mathbf{x}_n の線形変換
- 出力 \mathbf{y}_n は学習で得られる $\mathbf{w}_1^{(o)}, \mathbf{w}_2^{(o)}, \dots, \mathbf{w}_D^{(o)}$ が張る部分空間に限定される。
- 非線形性は注意の強度, a_{nm} , 経由で導入 (\mathbf{x}_n に依存・Softmax)

注意機構の性質

重要： $\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, $\mathbf{W}^{(v)}$ が注意の規準を定める（学習後は規準固定）

- a_{nm} ($m = 1, 2, \dots, N$) は、スケール $\sqrt{D_k}$ で調整したあとでも、ひとつだけ 1 に近い値となり、その他はゼロに近い値となることが多い。
- ソフトマックス関数の \exp が主要因
- 注意機構は「特定の規準のみ」で注意の大小を決める傾向あり

Attention Head の注意の規準

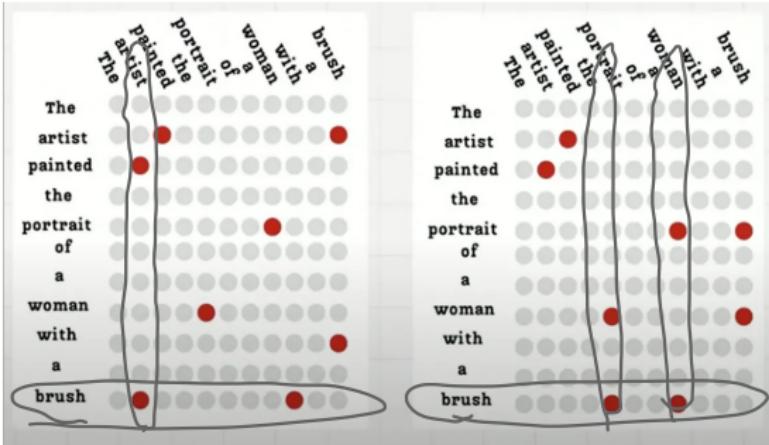
重要： $\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, $\mathbf{W}^{(v)}$ が注意の規準を定める（学習後は規準固定）

例：The artist painted the portrait of a woman with a brush.

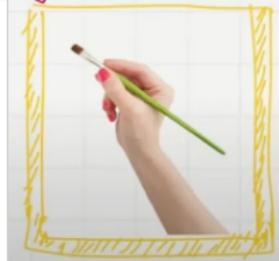
上記の文の意味はどっち？

- “a woman with a brush” を描いた
- “a woman” をブラシ（筆）を使って描いた

Attention Head の注意の規準



case 1 : Painting a woman with a "brush"



case 2: Painting of a "woman with a brush"

<https://medium.com/@shravankoninti/a-visual-explanation-of-multi-head-attention-6399d86fe51c>

Attention Head を複数

- 注意の強弱を決める基準はひとつでは足りない
- 時制 / 語彙 / 語順 / ...
- Multi-Head Attention: 複数の基準 ($\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, $\mathbf{W}^{(v)}$) を用意

マルチヘッド注意機構

$$Q_1$$

$$K_1$$

$$V_1$$

$$Q_2$$

$$K_2$$

$$V_2$$

$$W_1^{(q)}$$

$$W_1^{(k)}$$

$$W_1^{(v)}$$

$$W_2^{(q)}$$

$$W_2^{(k)}$$

$$W_2^{(v)}$$

X

マルチヘッド注意機構

H 個の注意機構を用いることにする。 h 番目 ($h = 1, 2, \dots, H$) の注意機構は次式でトークンを変換する。

$$\mathbf{H}_h = \text{Attention}(\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h)$$

クエリ・キー・バリューは次式で計算する

$$\mathbf{Q}_h = \mathbf{X} \mathbf{W}_h^{(q)},$$

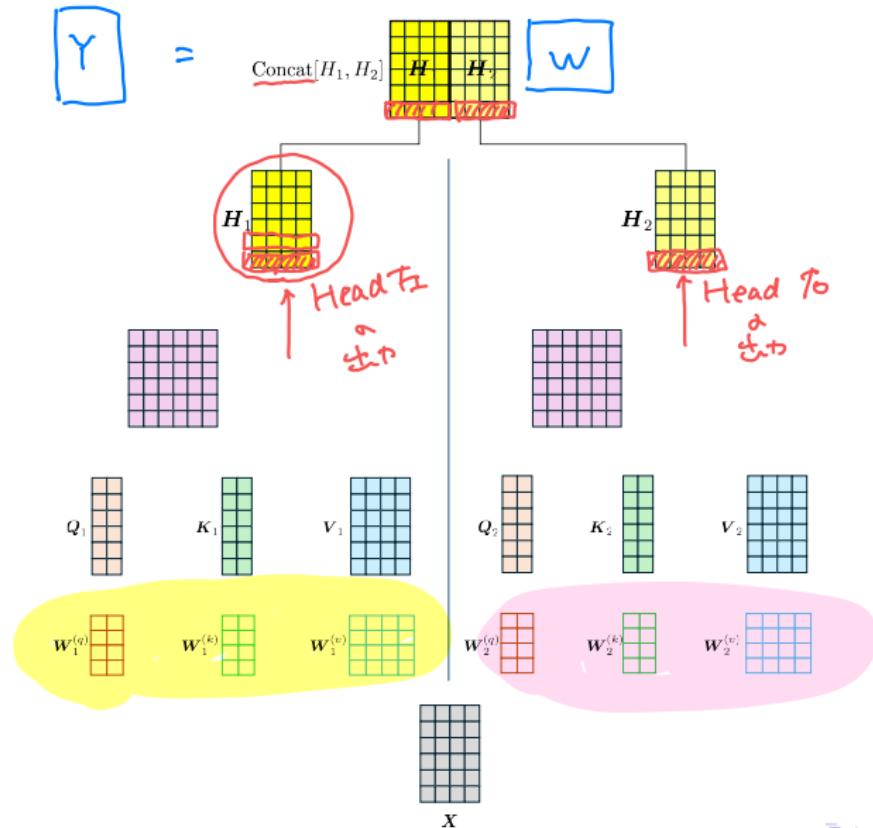
$$\mathbf{K}_h = \mathbf{X} \mathbf{W}_h^{(k)},$$

$$\mathbf{V}_h = \mathbf{X} \mathbf{W}_h^{(v)}.$$

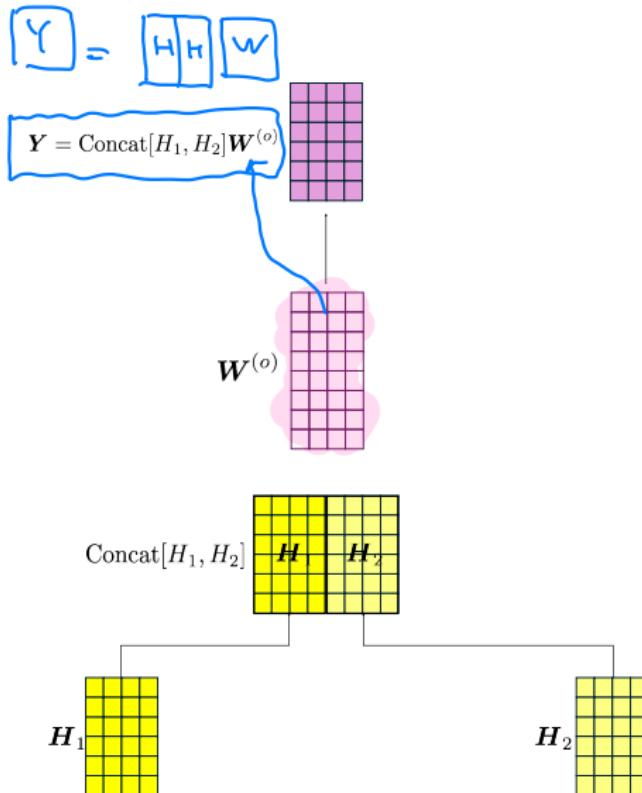
$$\underline{\mathbf{Y}(\mathbf{X}) = \text{Concat}[\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_H] \mathbf{W}^{(o)}}$$

ただし、 $\mathbf{H}_h \in \mathbb{R}^{N \times D_v}$, $\text{Concat}[\mathbf{H}_1, \dots, \mathbf{H}_H] \in \mathbb{R}^{N \times HD_v}$, $\mathbf{W}^{(o)} \in \mathbb{R}^{HD_v \times D}$, $\mathbf{Y}(\mathbf{X}) \in \mathbb{N} \times \mathbb{D}$ 。多くの場合、 $D_v = D/H$

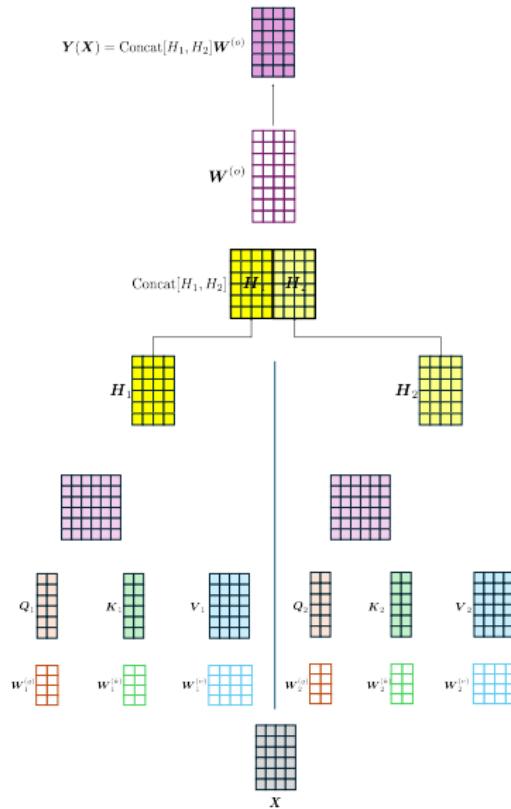
マルチヘッド注意機構



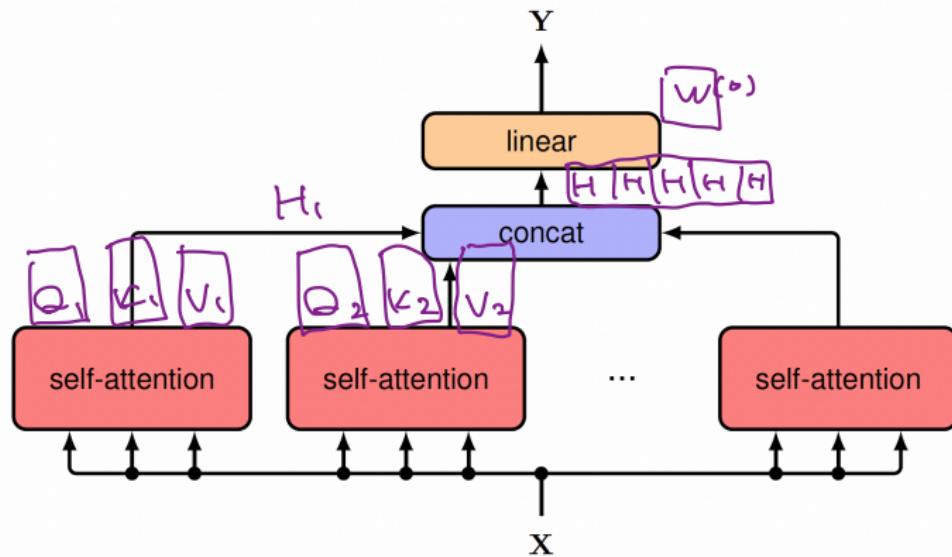
マルチヘッド注意機構



マルチヘッド注意機構



マルチヘッド注意機構



準備：レイヤ正規化 (Layer Normalization) (S)

- 誤差逆伝播の計算を安定化させる効果あり

それぞれの特徴ベクトル $\mathbf{y}_n \in \mathbb{R}^D$ を「正規化」する。

$$\mathbf{y}_n = [y_{n1}, y_{n2}, \dots, y_{nD}]^\top$$

準備：平均と分散

$$\mu_n = \frac{1}{D} \sum_{i=1}^D y_{ni}$$

$$\sigma_n^2 = \frac{1}{D} \sum_{i=1}^D (y_{ni} - \mu_n)^2$$

正規化

$$\hat{y}_{ni} = \frac{y_{ni} - \mu_n}{\sqrt{\sigma_n + \delta}}$$

記法

$$\text{LayerNorm}[\mathbf{y}_n] = [\hat{y}_{n1}, \hat{y}_{n2}, \dots, \hat{y}_{nD}]^\top$$

後処理 (1/3)

学習効率を上げる工夫: レイヤ正規化と、**残差表現**

マルチヘッドの出力: $\mathbf{Y}(\mathbf{X})$

$$\mathbf{Y}(\mathbf{X}) = \text{Concat}[\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_H] \mathbf{W}^{(o)}$$

元の入力を足して正規化。 $\mathbf{Y}(\mathbf{X})$ は元との差分のみ表現すれば良くなつた。

$$\mathbf{Z} = \text{LayerNorm}[\mathbf{Y}(\mathbf{X}) + \mathbf{X}]$$

↑
差分表現
↓
元の入力

後処理 (2/3)

注意機構の出力は v が張る線形部分空間に限定。柔軟性を改善するため
に Multi-Layer Perceptron を導入。例えば、全結合層を 2 つ追加する。

$$\mathbf{Z}' = \text{MLP}[\mathbf{Z}]$$

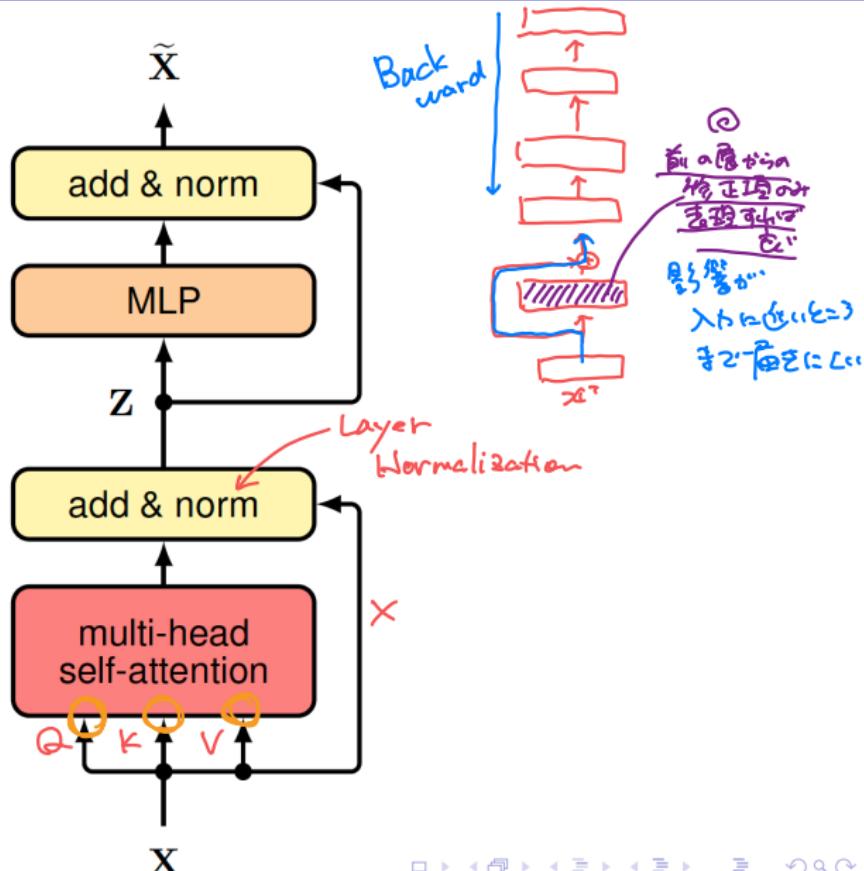
- 同じ MLP を \mathbf{Z} の各行に適用する。つまり、各単語のトークンに同
じ MLP を適用する。

後処理 (3/3)

学習効率を上げる工夫：レイヤ正規化と、残差表現

$$\tilde{\mathbf{X}} = \text{LayerNorm}[\mathbf{Z}' + \mathbf{Z}] = \text{LayerNorm}[\text{MLP}[\mathbf{Z}] + \mathbf{Z}]$$

トランスフォーマーのアーキテクチャ (符号器・1層分)



位置の表現

トークンの入力順序が変わっても出力は変わらない。このままでは、位置・順序の情報が結果に反映されない。

(再掲: 系列の先頭から n 番目のトークンの変換)

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

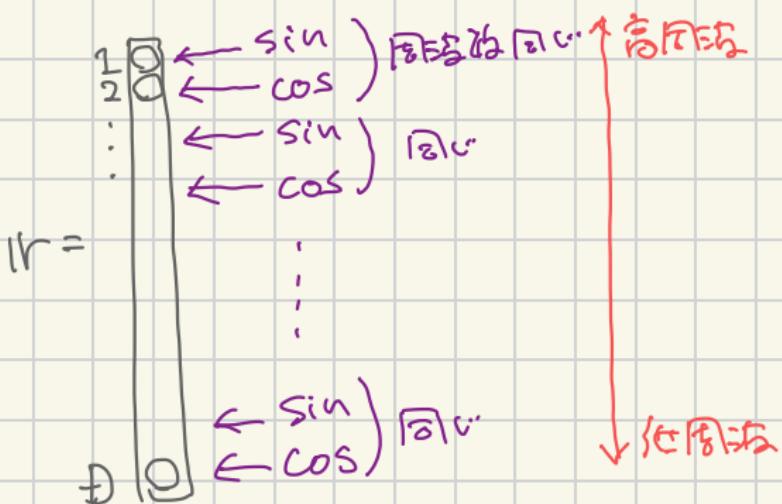
$$a_{nm} = \frac{\exp\left(\frac{\mathbf{q}_n^\top \mathbf{k}_m}{\sqrt{D_k}}\right)}{\sum_k \exp\left(\frac{\mathbf{q}_n^\top \mathbf{k}_k}{\sqrt{D_k}}\right)}.$$

位置の表現 $\mathbf{r}_n = [r_{n1}, r_{n2}, \dots, r_{nD}]^\top$ を生成して、入力トークンに足す

$$\tilde{\mathbf{x}}_n = \mathbf{x}_n + \mathbf{r}_n \quad \boxed{\quad} = \boxed{\quad} + \boxed{\quad} \quad \begin{array}{l} \text{← 位置表現} \\ \text{スカラビタクトセキ} \\ \text{ヤシ} \\ \text{ペアトリル 2. 選択} \\ \text{の冗長} \end{array}$$

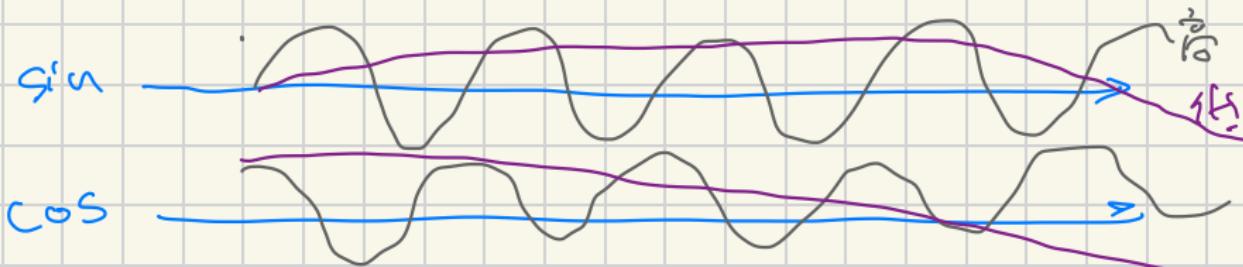
最大系列長より充分大きい数, L (例えば 10,000)

$$r_{ni} = \begin{cases} \sin\left(\frac{n}{L^{i-1/D}}\right), & \text{if } i \text{ is odd,} \\ \cos\left(\frac{n}{L^{(i-2)/D}}\right), & \text{if } i \text{ is even.} \end{cases}$$

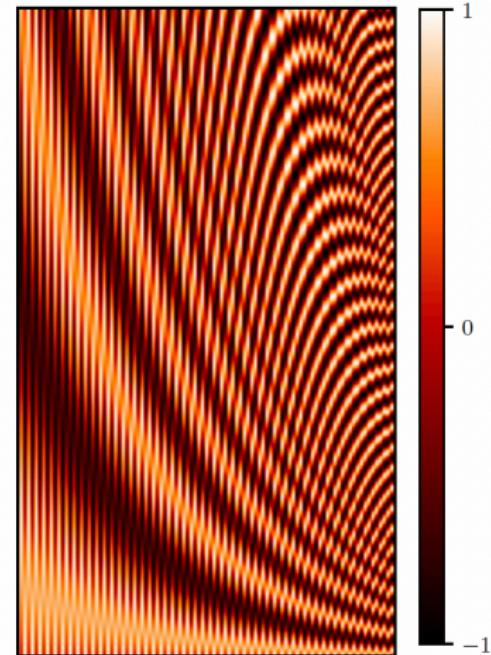
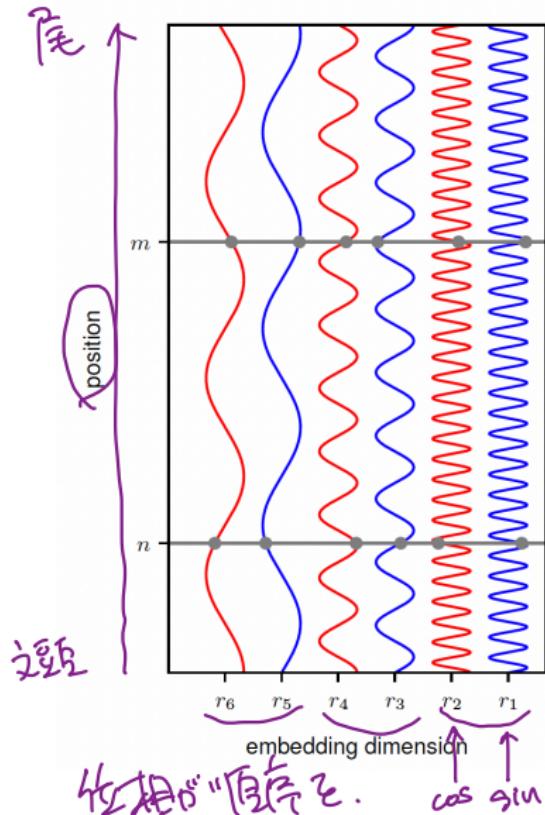


I swam across the river

$n = 1 \quad 2 \quad 3 \quad 4 \quad 5 \dots$



位置の表現



embedding dimension

位置の表現

位置の表現 r の性質

- $\|r_n\| = D/2$
- 位置が一意に表現される
- 近い位置では似たベクトル、遠い位置では似ていないベクトル
- r_n と r_{n+k} の関係は n に依らない (k だけに依存する) 線形変換で表現可能 ($\phi = k/L^{j/D}$)

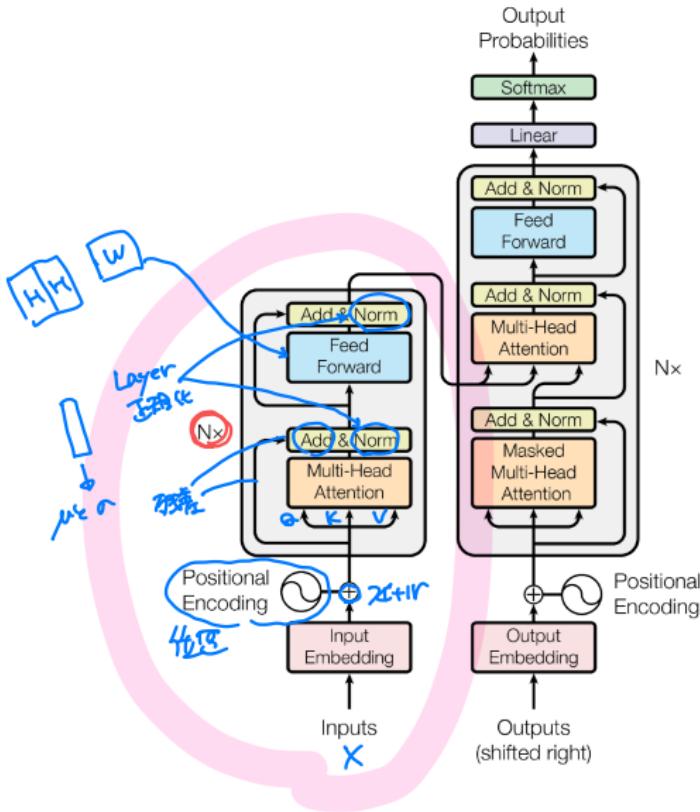


手書きをもと
仕組み

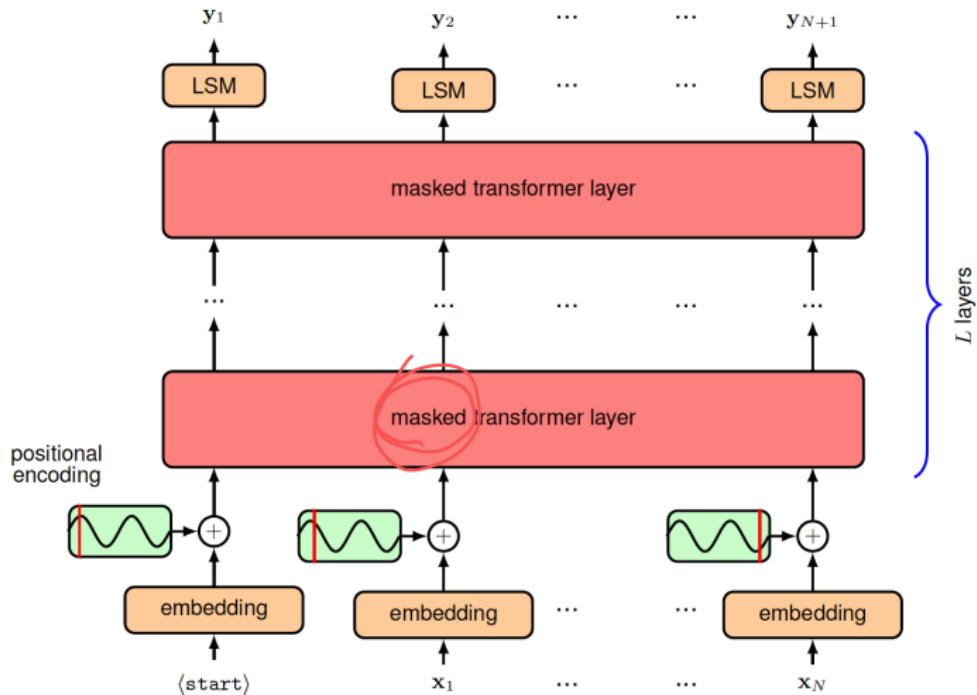
$$\sin \left(\frac{n+k}{L^{j/D}} \right) = \sin \left(\frac{n}{L^{j/D}} + \phi \right)$$
$$= \cos(\phi) \sin \left(\frac{n}{L^{j/D}} \right) + \sin(\phi) \cos \left(\frac{n}{L^{j/D}} \right)$$

- 系列内での相対位置が重要なときには、ニューラルネットワークが ($W^{(q)}, W^{(k)}$ で) 学習可能

トランسفォーマのアーキテクチャ



予告：GPT のアーキテクチャ



LSM: Linear SoftMax = 線形多クラス識別器