RoutesFormer: A sequence-based route choice Transformer for efficient path inference from sparse trajectories

Qiu, S., Qin, G., Wong, M., & Sun, J. (2024). RoutesFormer: A sequence-based route choice Transformer for efficient path inference from sparse trajectories. Transportation Research Part C: Emerging Technologies, 162, 104552.

2025/05/20 理論談話会#6

B4 水谷玲太

Abstract

1. 自然言語処理の深層学習モデルであるTransformerを応用し、**シーケンスベースの経路選択モデル** RoutesFormerを開発
2. 上海のタクシーデータセットで実験を行い、**既存のモデルを大きく上回る推定精度**を記録
3. モデル内のAttention機構の意義を**Ablation study**により考察し、経路選択行動のモデリングにおいて**過去の移動履歴を考慮する重要性**を提示

新規性・有用性・信頼性

- 新規性
  - 自然言語処理モデルであるTransformerの行動モデルへの応用
- 有用性
  - 疎な観測データの**経路選択**（Route Choice）と**経路補完**（Path Inference）を統一して扱うことが可能
  - マルコフ性仮定がなく、**過去の移動履歴の情報**を推論に用いることが可能
- 信頼性
  - 上海のタクシーデータを用いた実験で、既存の６つのモデルを全ての評価指標で上回る
  - アブレーション分析により、モデルの妥当性を検証した

目次

1. Introduction
2. Related work
    1. 既存の行動モデル
    2. Transformerの交通分野への応用例
3. Problem Description and Preliminaries
    1. 定義
    2. 自然言語処理の穴埋め問題とのアナロジー
4. Methodology
    1. Inputの下準備
        1. トークン化
        2. 埋め込み
        3. 位置エンコーディング
    2. エンコーダー・デコーダー
        1. Self-Attention
        2. Masked Self-Attention
        3. Cross-Attention
        4. Multi-Head Attention
        5. その他の構成
    3. RoutesFormerにおけるAttentionの解釈
    4. 補足事項
5. Case Study
    1. データ設定・比較対象
    2. 評価指標
    3. 推論精度の比較
    4. 計算時間の比較
    5. Ablation Study
        1. 精度の比較
        2. 精度の分布比較
        3. 推論結果の比較
6. Conclusion

1. Introduction

1.1 交通把握技術の進展とデータ統合の必要性

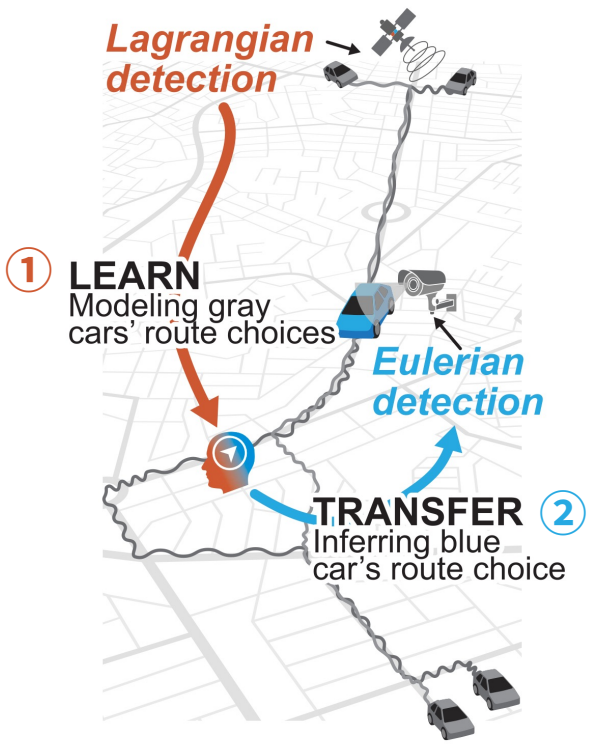
- センサー技術と機械学習の進展
  - ⇨交通システムの理解・管理・最適化はマクロ→ミクロレベルへ
- ⇔ミクロレベルの把握には**データの限界**：
  - **オイラー的観測**　ex)GPS
    →完全な軌跡を追えるが、サンプリング率が低い
  - **ラグランジュ的観測**　ex)路側センサー
    →全交通量を観測できるが、断片的な情報しか得られない
- ⇨ラグランジュ的データによる①**経路選択の学習** & オイラー的データの②**経路補完への転移**の枠組みが必要（次図）

1.1 交通把握技術の進展とデータ統合の必要性

	ラグランジュ (ex: GPS)	オイラー(ex:路側センサー)
網羅性	低い	高い
軌跡の質	**完全なデータ**	**断片的**

①経路選択（RC）の学習

②経路補完（PI）への転移



⇨密な完全軌跡データを推論可能🙌

⇔ **既存のモデルの限界**：

- マルコフ仮定…過去の移動履歴の影響無視
- 長い経路を複数の問題に分割 → 準最適に留まる

⇨

RoutesFormer：

- ①経路選択の学習と②経路補完の推論を**統一的に処理**
- 経路全体を一括で扱い、長期的な情報を学習

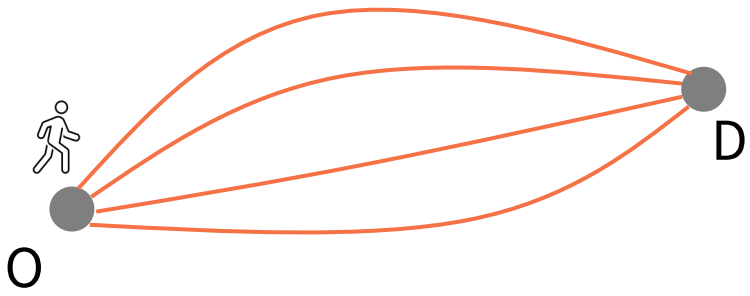
2. Related work

2.1 既存の行動モデル

①経路選択（RC）モデル

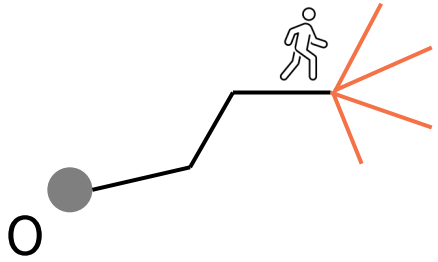
パスベース（パス列挙）

- PSL、DNN-PSL



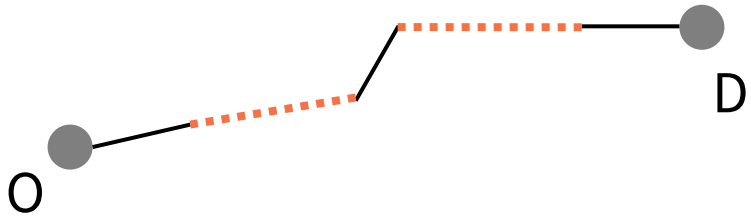
リンクベース（リンクを逐次選択）

- RL…マルコフ仮定



リンクからなる**系列全体を扱うシーケンスベース**のモデルへ（今回）

②経路補完（PI）モデル



- 最短経路での補完→現実の選好との乖離
- DNNを用いた例も出始めたが、不連続パス全体の考慮ができない／複数のパス候補を独立して扱う⇨相関考慮なし・メモリ容量増大

2.2 Transformerの交通分野への応用例

- Transformer：従来のRNNと異なり、attention機構のみで完結する自然言語処理モデルとして成功を収める
- これまでの適用例：
  - Sequence prediction → 交通流予測、軌跡予測
  - Sequence to sequence → マップマッチング
- BERTのような空所補充課題への応用はない
- TransformerのSequence to sequence（翻訳）タスクと、BERTのcloze（穴埋め）タスクの組み合わせが、今回取り組む長さ可変の穴埋めタスク

⇨モデル設計（３～４章）

3. Problem Description and Preliminaries

3.1 定義

$G = (N, E)$:ネットワーク

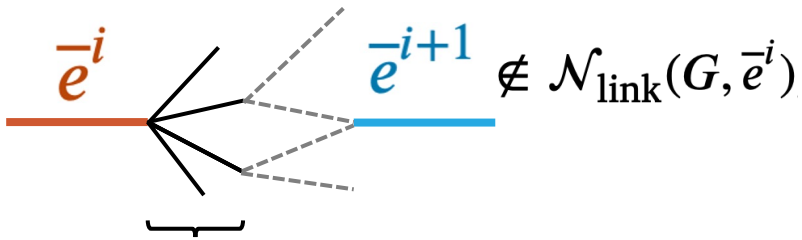
$\overline{\boldsymbol{R}} = [\bar{e}^1, \bar{e}^2, \ldots, \bar{e}^i, \ldots, \bar{e}^{|\overline{\boldsymbol{R}}|}]$:パス

$\mathcal{N}_{\text{link}}(G, \bar{e}^i)$:ネットワークGにおけるリンク$\bar{e}^i$の下流リンクの集合

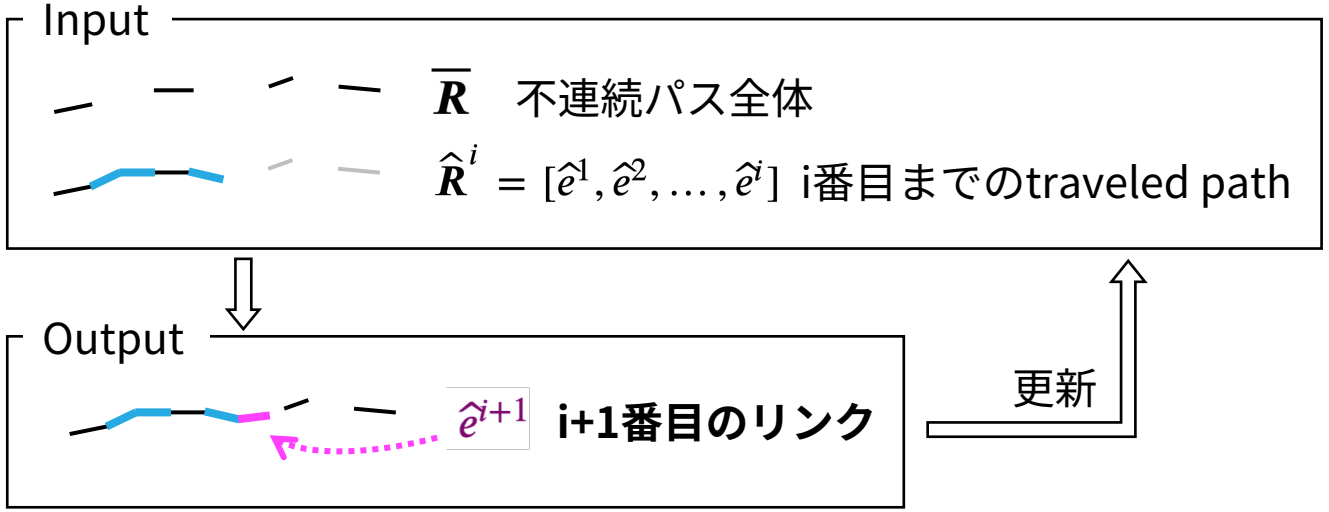
⇩

$\bar{e}^{i+1} \notin \mathcal{N}_{\text{link}}(G, \bar{e}^i)$ なる i がある時、

$\overline{\boldsymbol{R}} = [\bar{e}^1, \bar{e}^2, \ldots, \bar{e}^i, \ldots, \bar{e}^{|\overline{\boldsymbol{R}}|}]$ は**不連続パス**



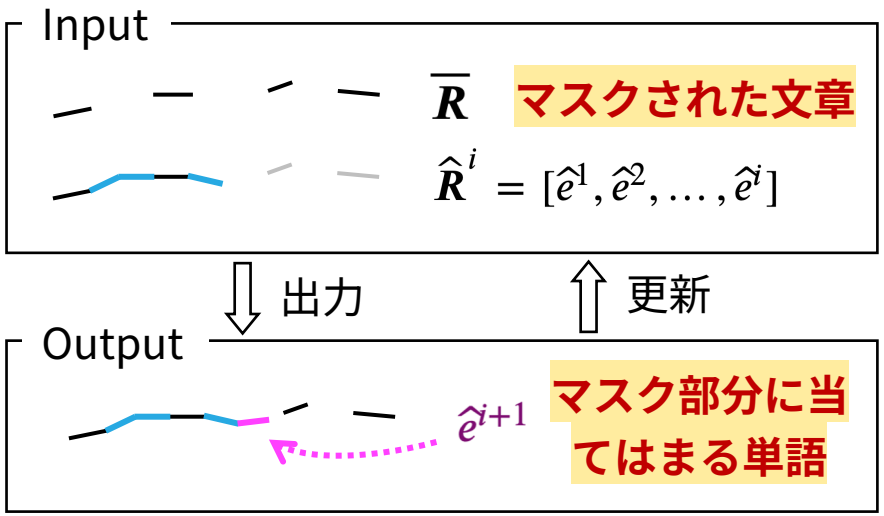
Sequence-based Route Choice(SRC) model

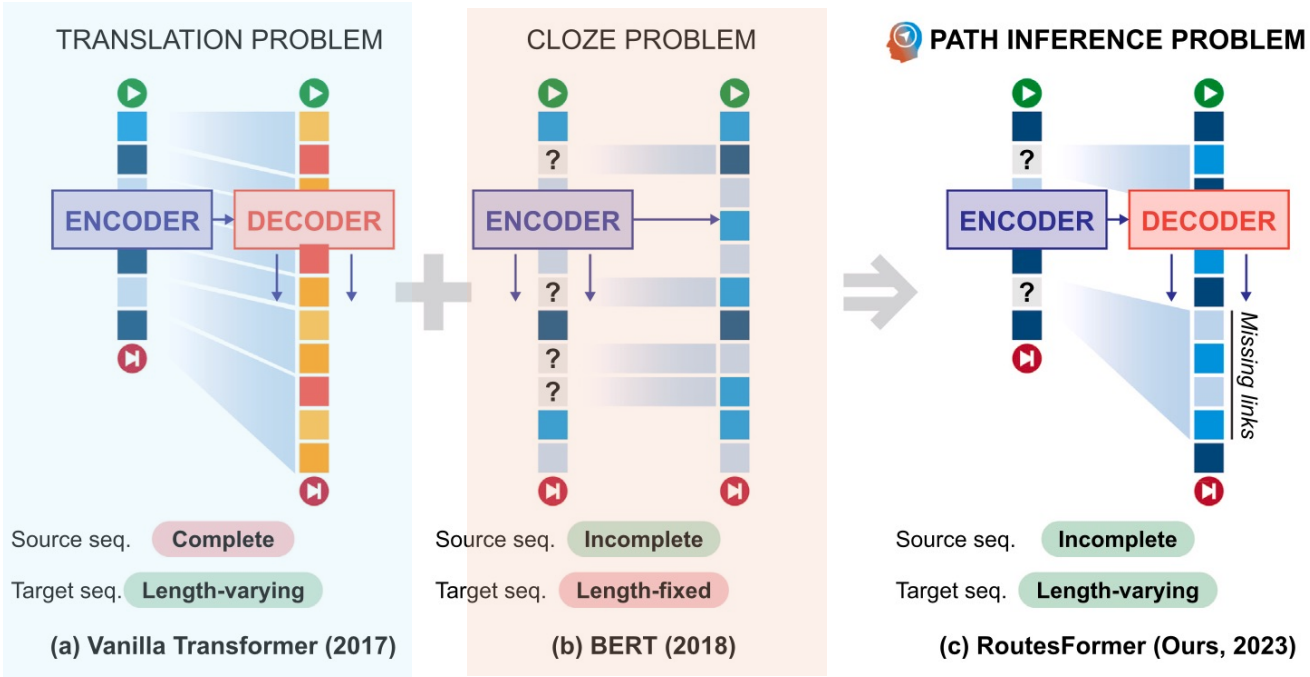


$$\min_{\hat{\boldsymbol{R}}} \quad \mathcal{L}(\boldsymbol{R}, \hat{\boldsymbol{R}})$$

$$\text{s.t.} \begin{cases} \hat{\boldsymbol{R}} = [\hat{e}^1, \hat{e}^2, \hat{e}^3, \ldots, \hat{e}^{|\hat{\boldsymbol{R}}|}] \\ \overline{\boldsymbol{R}} \text{ is the subsequence of } \hat{\boldsymbol{R}} \\ \hat{e}^{i+1} = \text{SRC}(\overline{\boldsymbol{R}}, \hat{\boldsymbol{R}}^i), \quad \forall i \in \{1, \ldots, |\hat{\boldsymbol{R}}| - 1\} \\ \hat{e}^{i+1} \in \mathcal{N}_{\text{link}}(G, \hat{e}^i), \quad \forall i \in \{1, \ldots, |\hat{\boldsymbol{R}}| - 1\}. \end{cases}$$

3.2 自然言語処理の穴埋め問題とのアナロジー





本モデル特有の課題＝
空欄に当てはまるリンクの数が不明

⇔自然言語処理の空欄補充では空欄に入る単語数が既知

→**出力の長さが可変**である必要がある

- Transformer（翻訳）の**Output長の可変性**
- BERT（**空欄補充**）

を組み合わせた、**不完全パス**から**完全パス**に
穴を埋めながら翻訳するモデル

4. Methodology

4.1 inputの下準備

$\overline{\boldsymbol{R}} = [\overline{e}^1, \overline{e}^2, \ldots, \overline{e}^i, \ldots, \overline{e}^{|\overline{\boldsymbol{R}}|}]$

$\hat{\boldsymbol{R}}^i = [\hat{e}^1, \hat{e}^2, \ldots, \hat{e}^i]$

トークン化(Tokenization)

$\overline{\boldsymbol{R}}_T$ $(\in \mathbb{N}^{l_{\max}})$

$\hat{\boldsymbol{R}}^i_{\mathrm{T}}$ $(\in \mathbb{N}^{i+1})$

埋め込み(Embedding)

$\overline{\boldsymbol{R}}_{\mathrm{E}}$ $(\in \mathbb{R}^{l_{\max} \times d_{\mathrm{e}}})$

$\hat{\boldsymbol{R}}^i_{\mathrm{E}}$ $(\in \mathbb{R}^{(i+1) \times d_{\mathrm{e}}})$

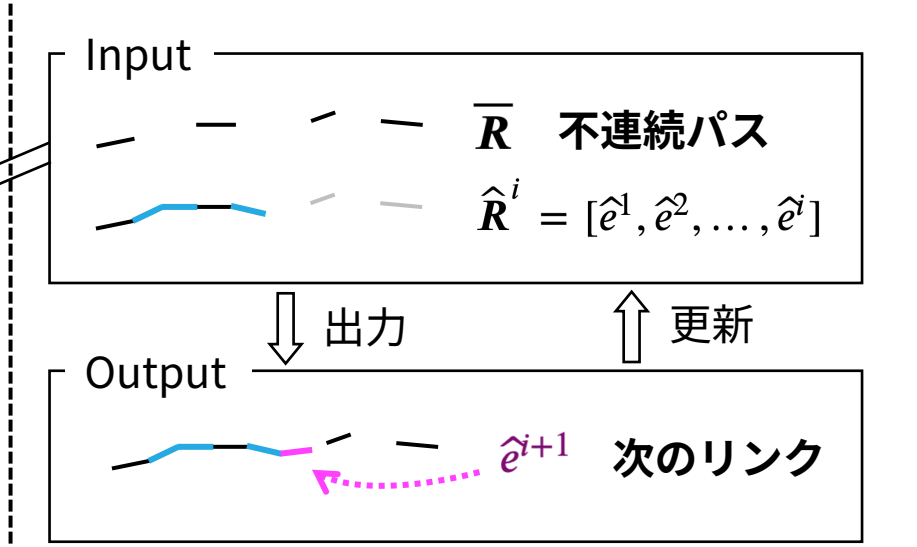
位置エンコーディング

$\overline{\boldsymbol{R}}_{\mathrm{E}}$

$\hat{\boldsymbol{R}}^i_{\mathrm{E}}$

ENCODER

DECODER



正解リンク$\boldsymbol{R}$との差を小さくするようパラメタ調整

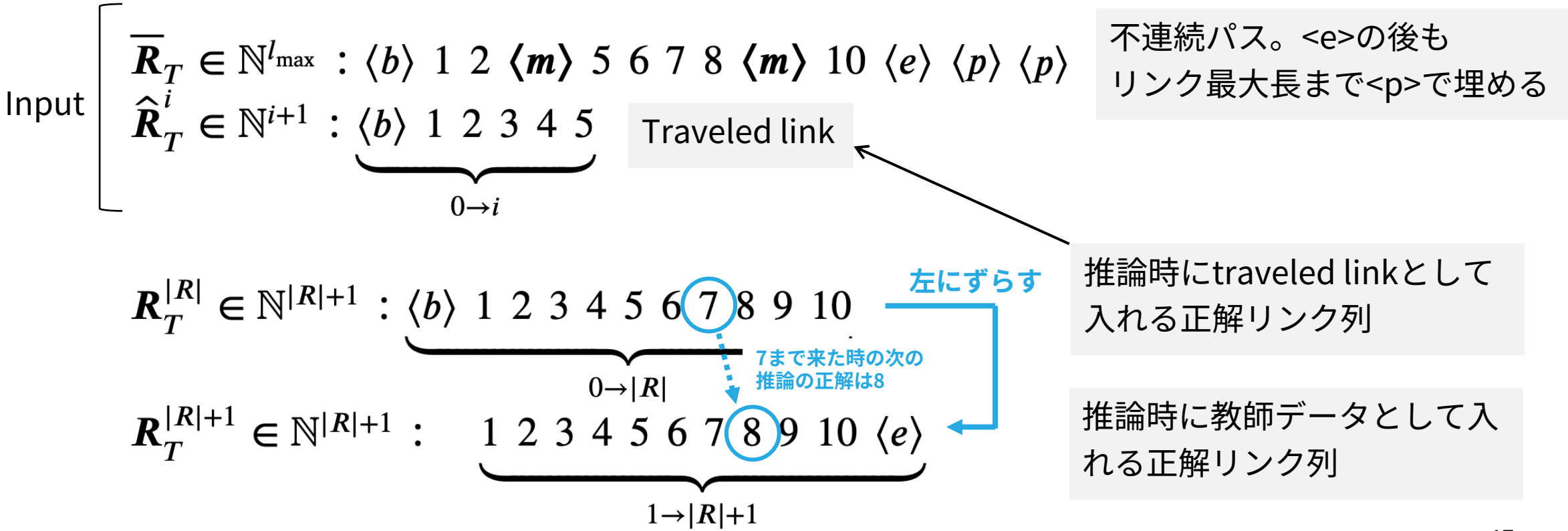
隣接しているリンクのうち確率が最大のものを選ぶ

$\boldsymbol{P}(\hat{\boldsymbol{R}}^{i+1})$ $(\in \mathbb{R}^{(i+1) \times |\mathcal{V}|})$

次の選択確率の分布

4.1.1 トークン化（Tokenization）

- Inputをトークンのベクトルにする。種類は下記：
  - リンクIDトークン（自然数）
  - スペシャルトークン<b><e><m><p>



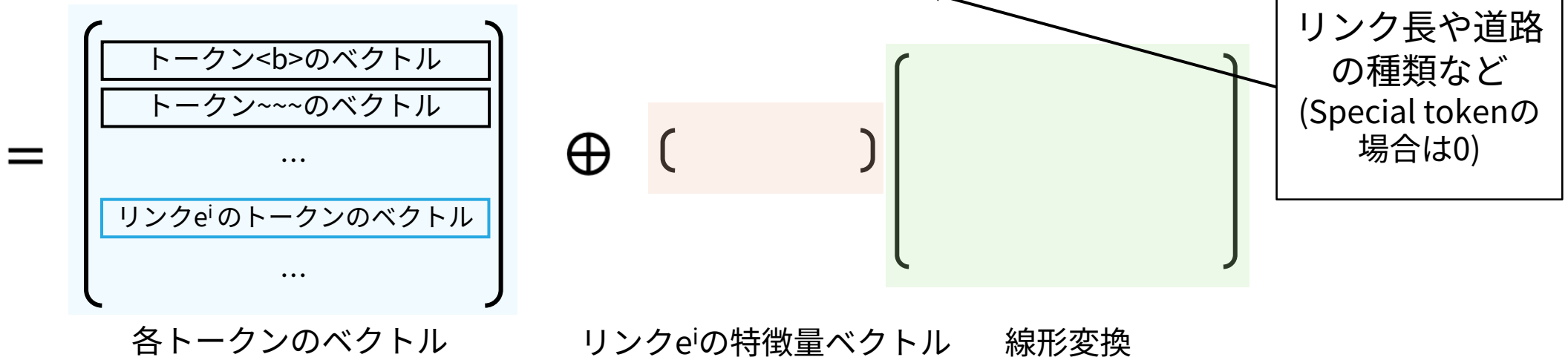
4.1.2 埋め込み（Embedding）

- トークン化したベクトルを、**リンク特徴量とともに**埋め込み次元ぶんのベクトルに
- 今回は埋め込み次元＝32（原文に128とあるが誤記？）

$$\mathrm{Embedding}(e^i \mid \mathcal{W}_e) = \boldsymbol{W}_{ie}[e^i, :] \oplus e_f^{iT} \boldsymbol{W}_{fe}$$

学習可能なパラメタ行列

リンク長や道路の種類など(Special tokenの場合は0)



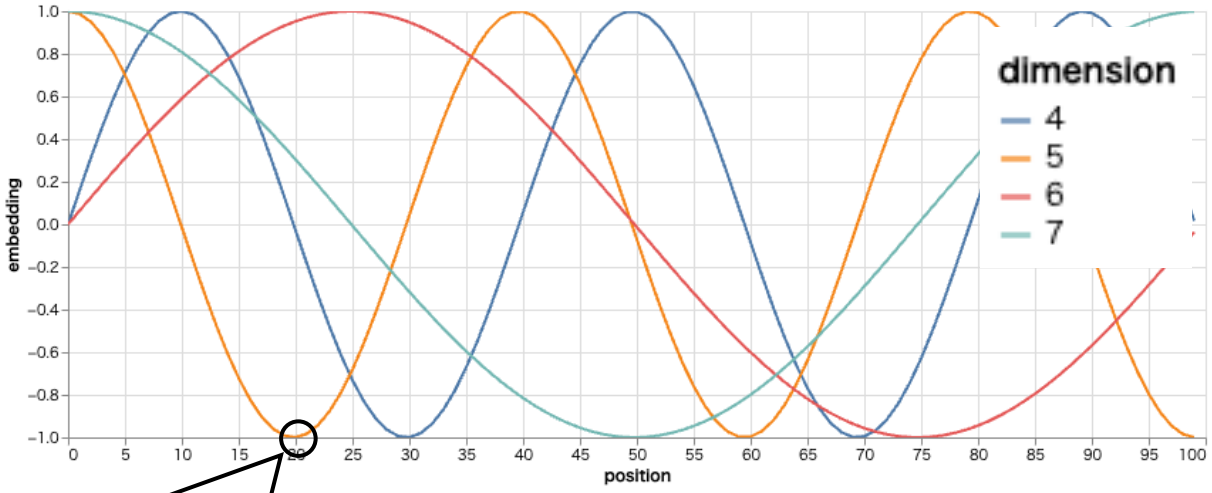
各トークンのベクトル　　リンクeⁱの特徴量ベクトル　　線形変換

＝（リンクeⁱのトークンのベクトル　線形変換されたリンクeⁱの特徴量ベクトル）

4.1.3 位置エンコーディング（Positional Encoding）

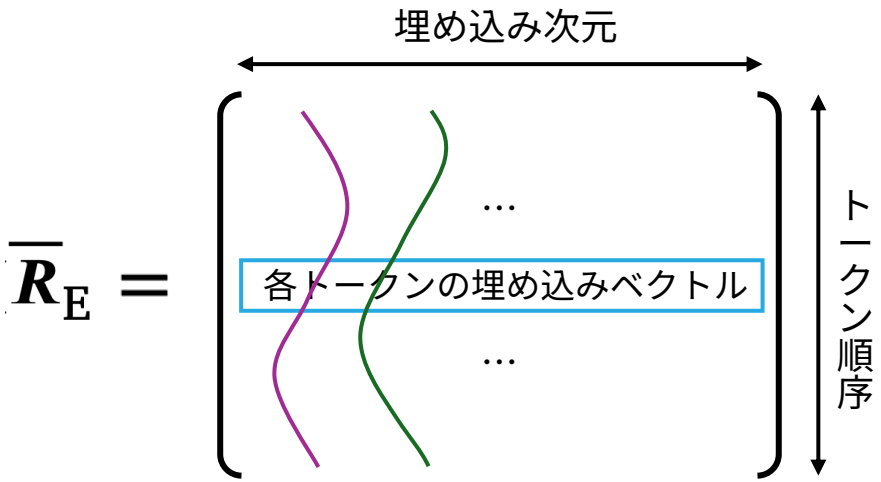
- この後のattention機構は系列の順序情報を考慮しない
- →順序をエンコーディングする

$$\mathrm{PE}_{(t,2i)} = \sin\left(t/10000^{2i/d_{\mathrm{model}}}\right)$$
$$\mathrm{PE}_{(t,2i+1)} = \cos\left(t/10000^{2i/d_{\mathrm{model}}}\right)$$
$$\left(0 \le i < \frac{d_{\mathrm{model}}}{2}\right)$$



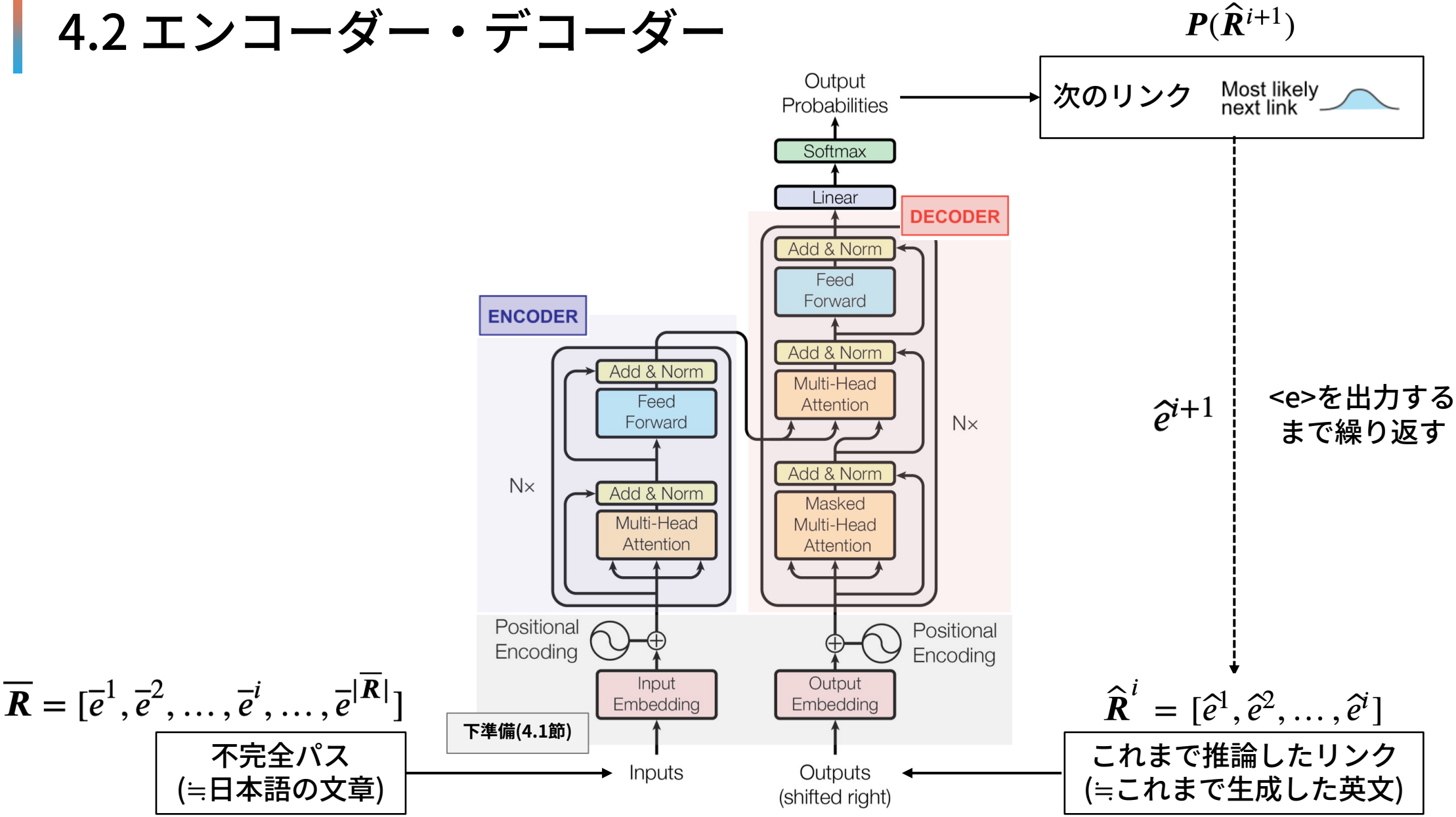
20番目のトークンの埋め込みベクトルの、５次元目に-1を足す

https://nlp.seas.harvard.edu/annotated-transformer/



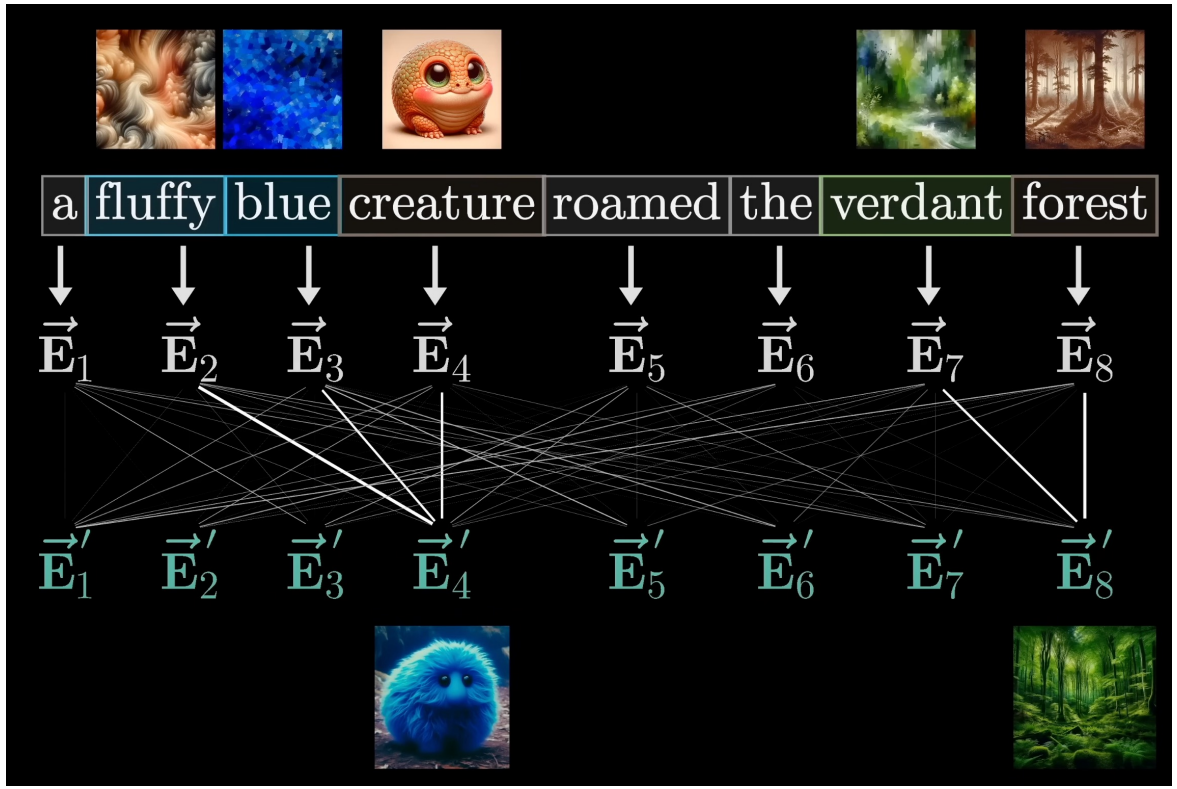
各次元ごとに色々な正弦波を足す

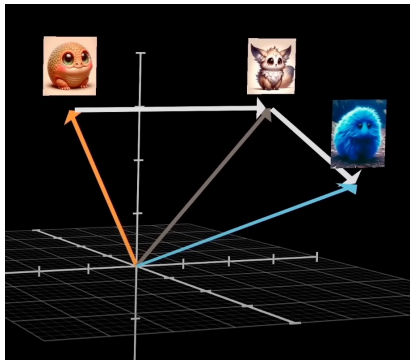
4.2 エンコーダー・デコーダー



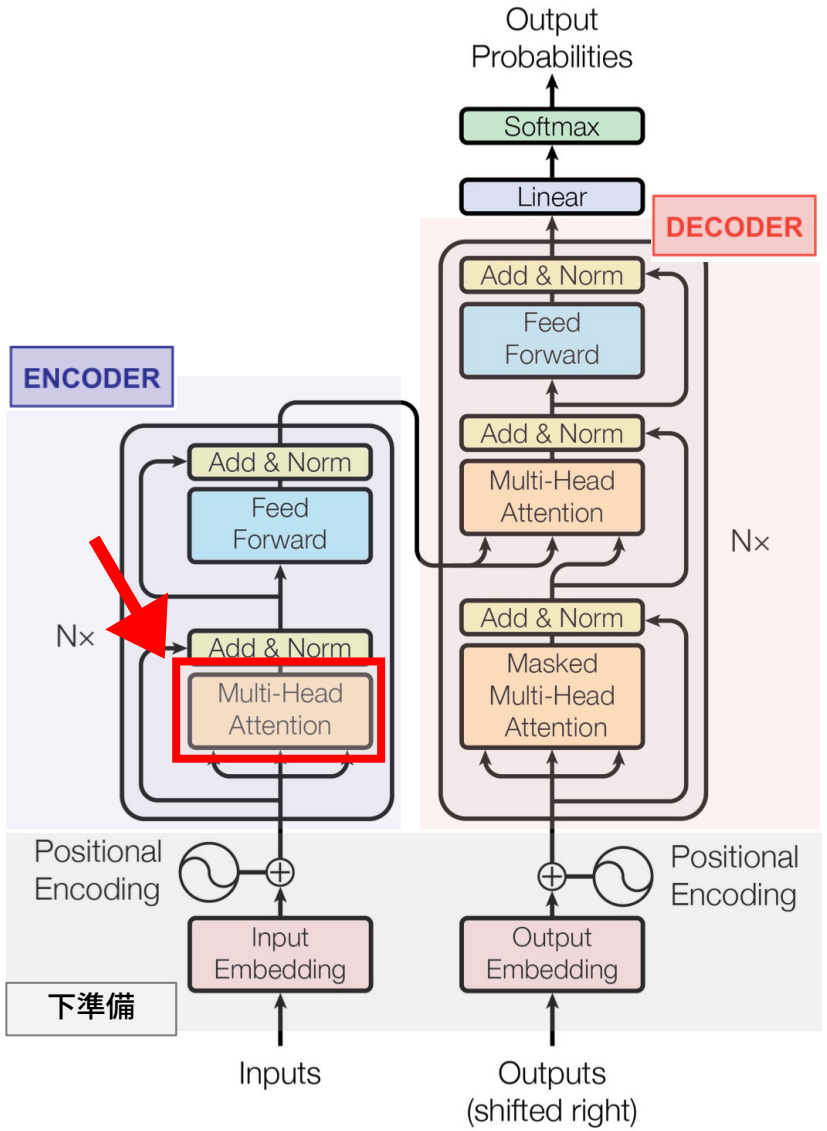
4.2.1 Self-Attention

- 文脈を加味していないinputの埋め込みベクトルを、
  文脈を考慮したベクトルに変換する





埋め込みベクトルの方向が意味を表すイメージ



以降添付している黒い背景の図：
https://www.youtube.com/watch?v=j3_VgCt18fA

4.2.1 Self-Attention

文脈を加味していないinputの埋め込みベクトルを、文脈を考慮したベクトルに変換する

①inputした埋め込みベクトル（今回はX＝Zで共通）の線形変換によりQ, K, Vを計算

②QとKの各要素の積（を次元の平方根で割ったもの）のsoftmaxを計算し、Aとする

③Vの重みつき和（＝Aの要素との重み付き和）を出力する

$$
\begin{aligned}
1\ & \boldsymbol{Q} \leftarrow \boldsymbol{X}\boldsymbol{W}_q + \mathbf{1}\boldsymbol{b}_q^T \quad [[\boldsymbol{Q}\mathrm{uery} \in \mathbb{R}^{l_x \times d_\mathrm{e}}]] \\
2\ & \boldsymbol{K} \leftarrow \boldsymbol{Z}\boldsymbol{W}_k + \mathbf{1}\boldsymbol{b}_k^T \quad [[\boldsymbol{K}\mathrm{ey} \in \mathbb{R}^{l_z \times d_\mathrm{e}}]] \\
3\ & \boldsymbol{V} \leftarrow \boldsymbol{Z}\boldsymbol{W}_v + \mathbf{1}\boldsymbol{b}_v^T \quad [[\boldsymbol{V}\mathrm{alue} \in \mathbb{R}^{l_z \times d_\mathrm{e}}]] \\
4\ & \boldsymbol{A} \leftarrow \mathrm{softmax}\left(\boldsymbol{M} + \frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_\mathrm{e}}}\right) \quad [[\boldsymbol{A}\mathrm{ttentionweight} \in \mathbb{R}^{l_x \times l_z}]] \\
5\ & \boldsymbol{Y} \leftarrow \boldsymbol{A} \cdot \boldsymbol{V}
\end{aligned}
$$

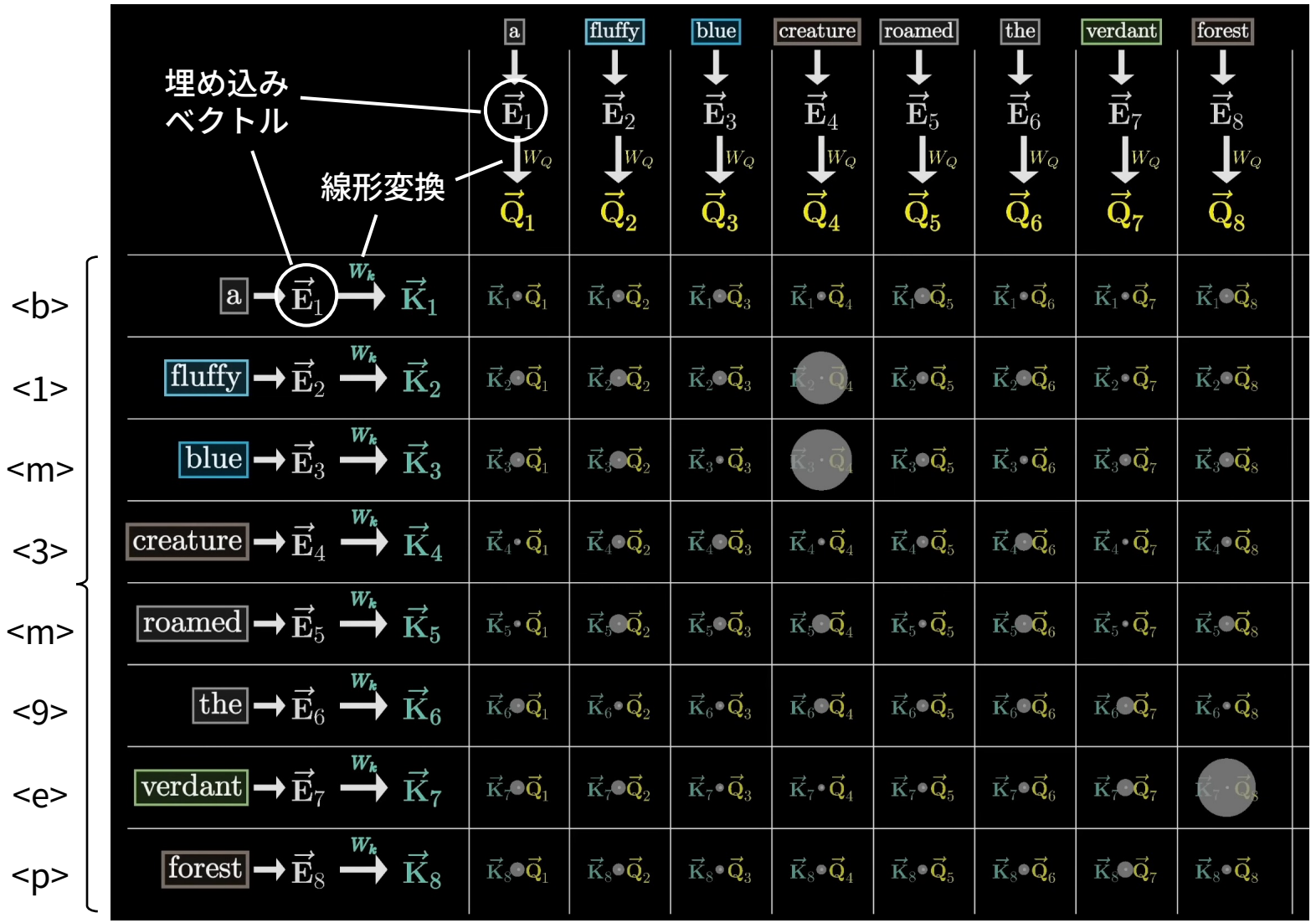
4.2.1 Self-Attention

①inputした埋め込みベクトルの線形変換によりQ, K, Vを計算

②QとKの各要素の積（を次元の平方根で割ったもの）

RoutesFormerでは下記のような不連続リンク列

<b> <1> <m> <3> <m> <9> <e> <p>



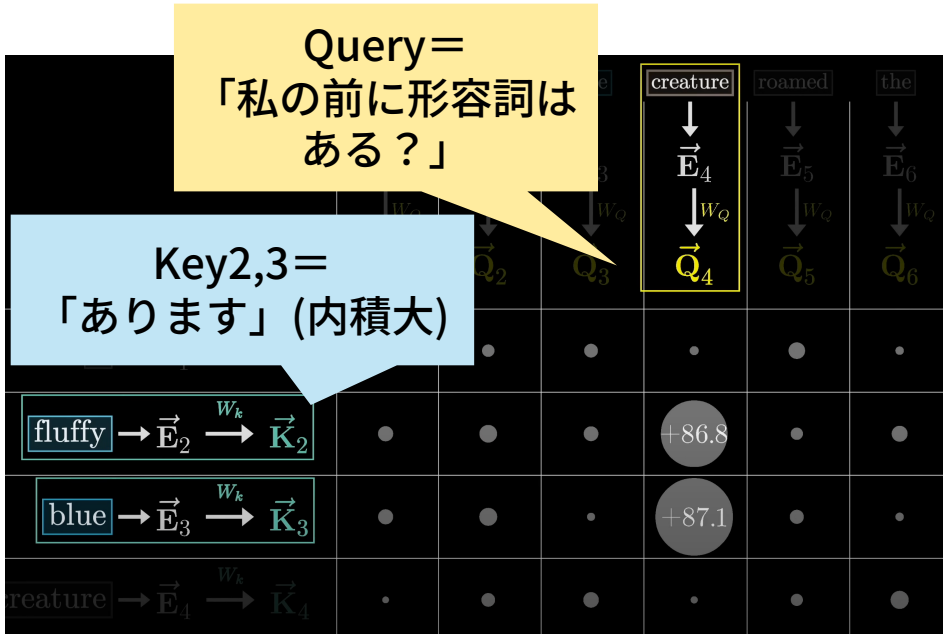
4.2.1 Self-Attention

RoutesFormerでは下記のような不連続リンク列

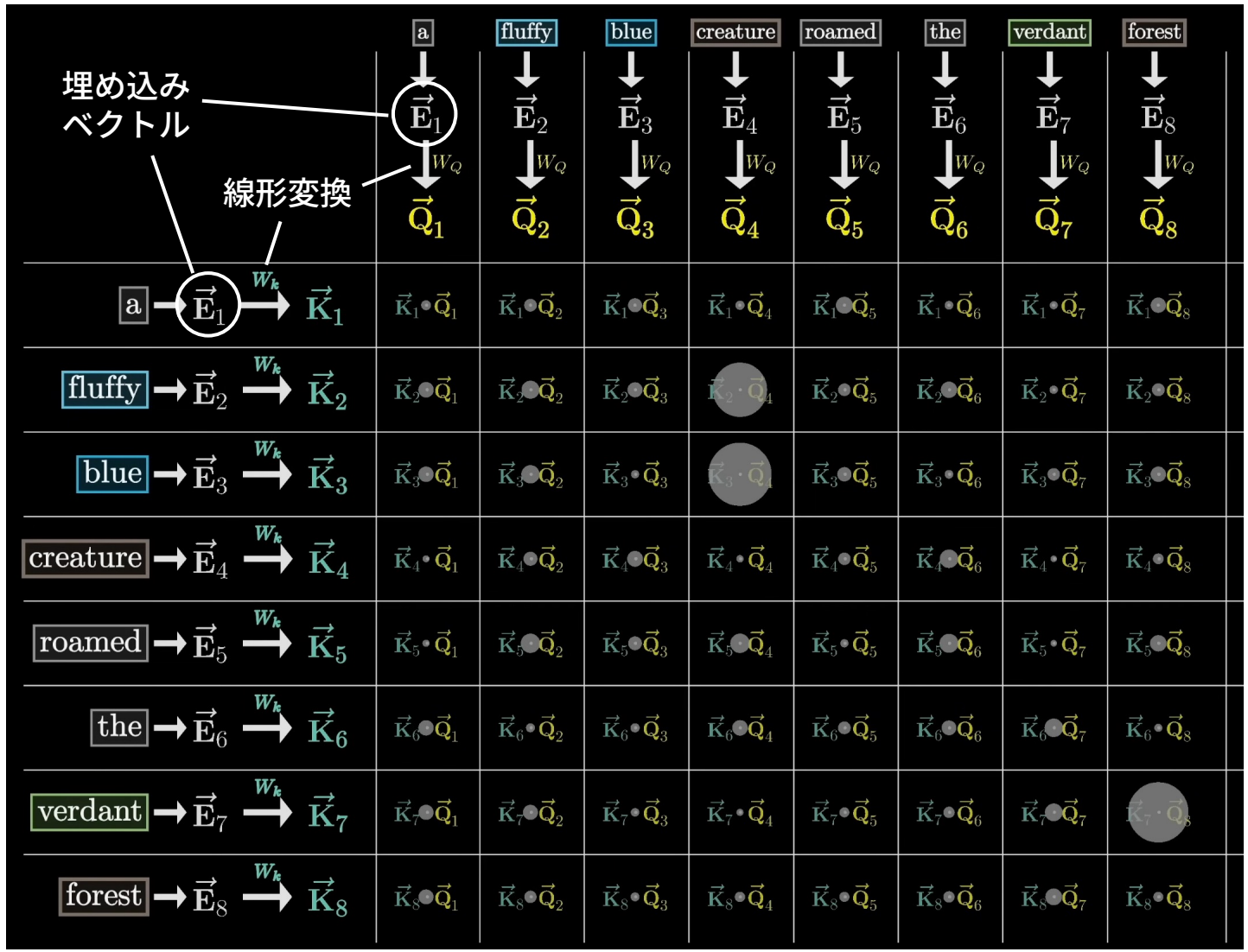
<b> <1> <m> <3> <m> <9> <e> <p>

①inputした埋め込みベクトルの線形変換によりQ, K, Vを計算

②QとKの各要素の積（を次元の平方根で割ったもの）



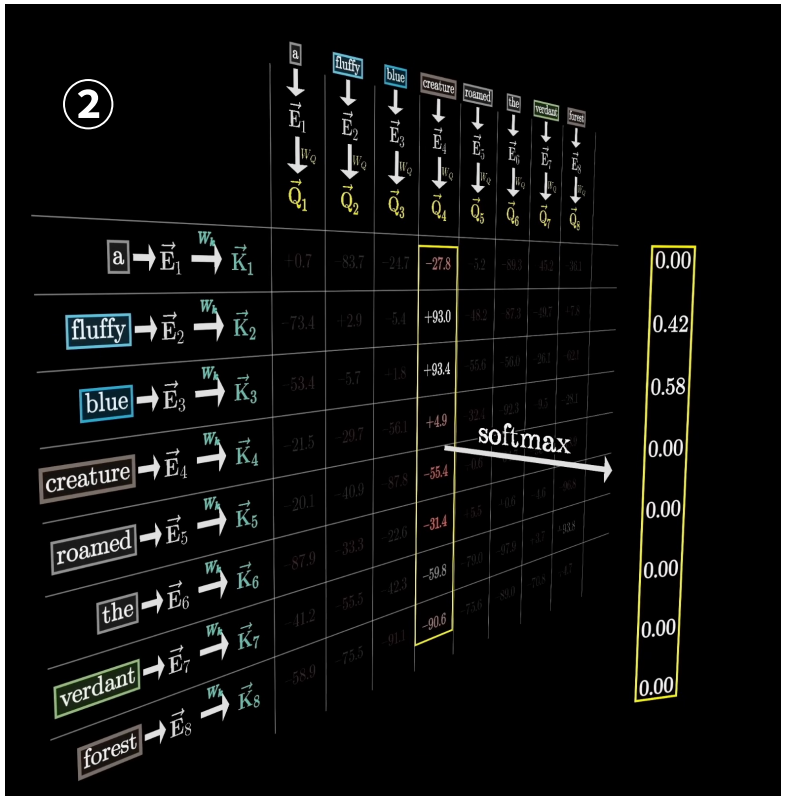
↑こうなるように変換行列W_Q,W_kの学習が進む



4.2.1 Self-Attention

①inputした埋め込みベクトルの線形変換によりQ, K, Vを計算

②QとKの各要素の積（を次元の平方根で割ったもの）のsoftmaxを計算し、Aとする



Softmax関数：

-∞～+∞の値の列を、0～1で和が1になるように変換する（→確率への変換）

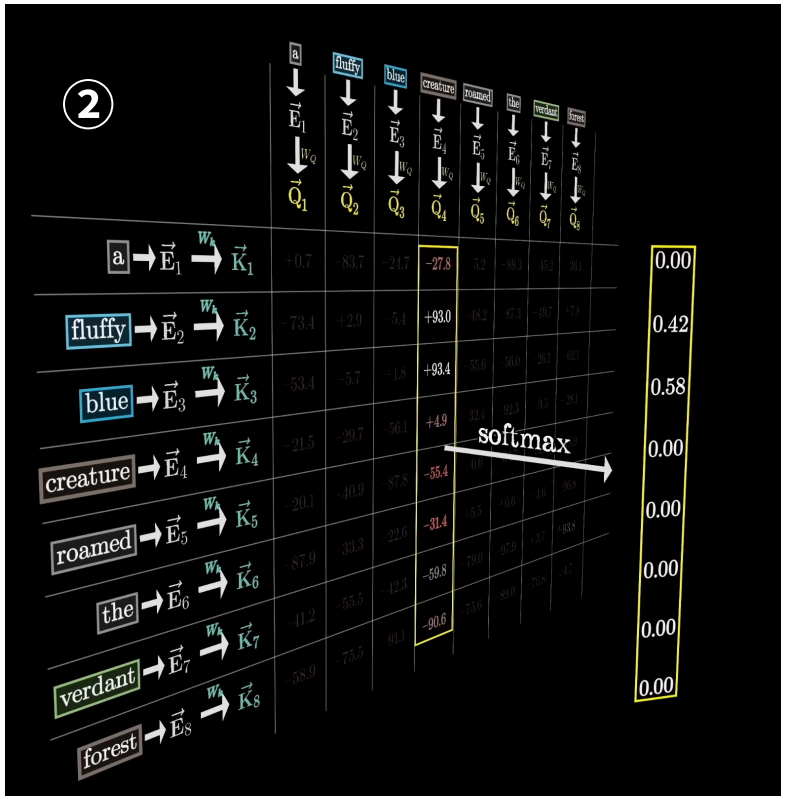
$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

4.2.1 Self-Attention

①inputした埋め込みベクトルの線形変換によりQ, K, Vを計算

②QとKの各要素の積（を次元の平方根で割ったもの）のsoftmaxを計算し、Aとする

③Vの重みつき和（＝Aの要素との重み付き和）を出力

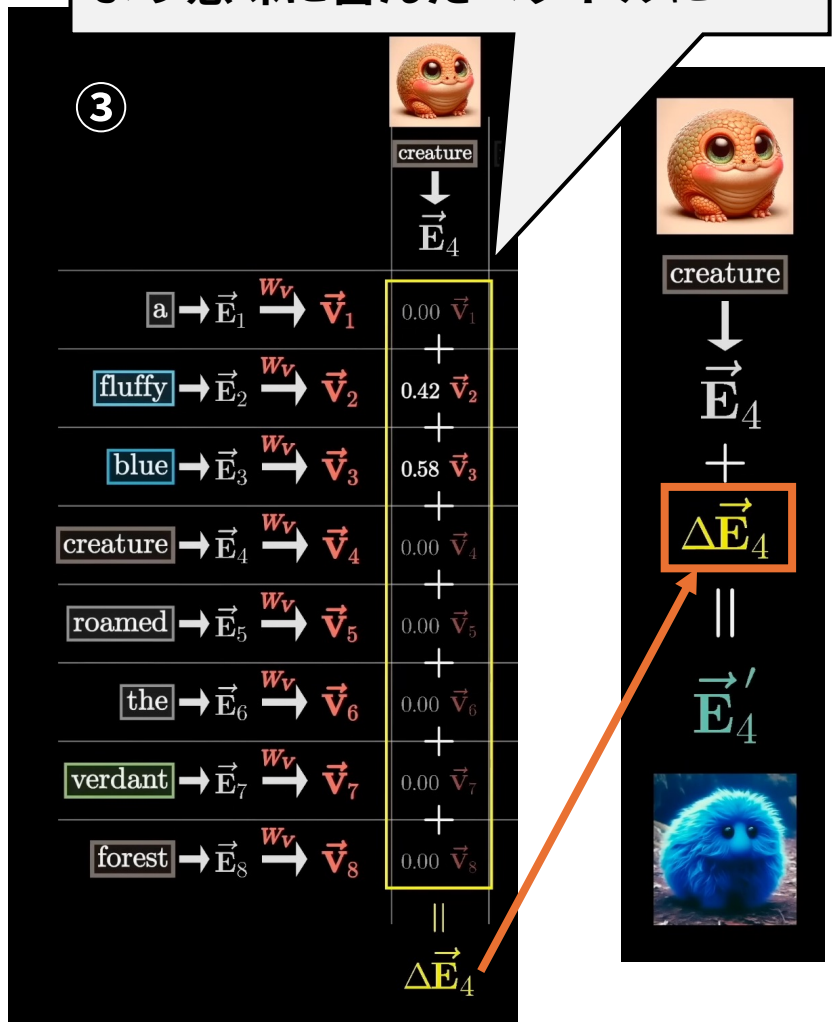


Softmax関数：

-∞～+∞の値の列を、0～1で和が1になるように変換する（→確率への変換）

$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

Creatureの埋め込みベクトルは、fluffyとblueのValueベクトルをAttentionの重みに応じて足し、より**意味に富んだベクトル**に



4.2.1 Self-Attention

①inputした埋め込みベクトルの線形変換によりQ, K, Vを計算

②QとKの各要素の積（を次元の平方根で割ったもの）のsoftmaxを計算し、Aとする

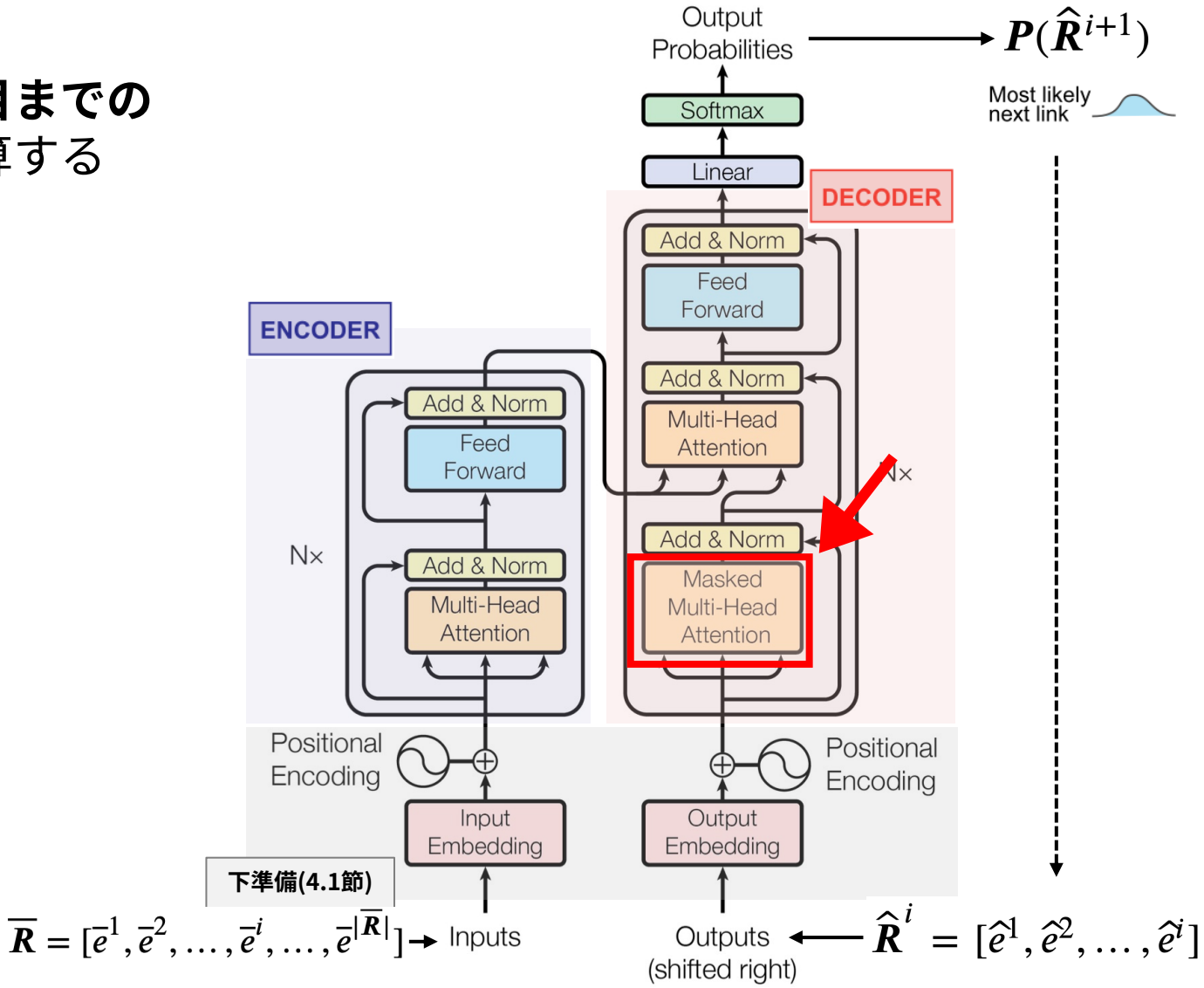
③Vの重みつき和（＝Aの要素との重み付き和）を出力

⇨出力結果は、各トークンのベクトルを、より文脈を考慮した方向へ向けるベクトル

$$
\begin{aligned}
&1\quad \boldsymbol{Q} \leftarrow \boldsymbol{X}\boldsymbol{W}_q + \mathbf{1}\boldsymbol{b}_q^T \quad [[\boldsymbol{Q}\text{uery} \in \mathbb{R}^{l_x \times d_e}]] \\
&2\quad \boldsymbol{K} \leftarrow \boldsymbol{Z}\boldsymbol{W}_k + \mathbf{1}\boldsymbol{b}_k^T \quad [[\boldsymbol{K}\text{ey} \in \mathbb{R}^{l_z \times d_e}]] \\
&3\quad \boldsymbol{V} \leftarrow \boldsymbol{Z}\boldsymbol{W}_v + \mathbf{1}\boldsymbol{b}_v^T \quad [[\boldsymbol{V}\text{alue} \in \mathbb{R}^{l_z \times d_e}]] \\
&4\quad \boldsymbol{A} \leftarrow \text{softmax}\left(\boldsymbol{M} + \frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_e}}\right) \quad [[\boldsymbol{A}\text{ttentionweight} \in \mathbb{R}^{l_x \times l_z}]] \\
&5\quad \boldsymbol{Y} \leftarrow \boldsymbol{A} \cdot \boldsymbol{V}
\end{aligned}
$$

4.2.2 Masked Self-Attention（計算上の処理）

- i+1番目の推論時、デコーダーでは**i番目までのtraveled linksのSelf-Attention**を計算する



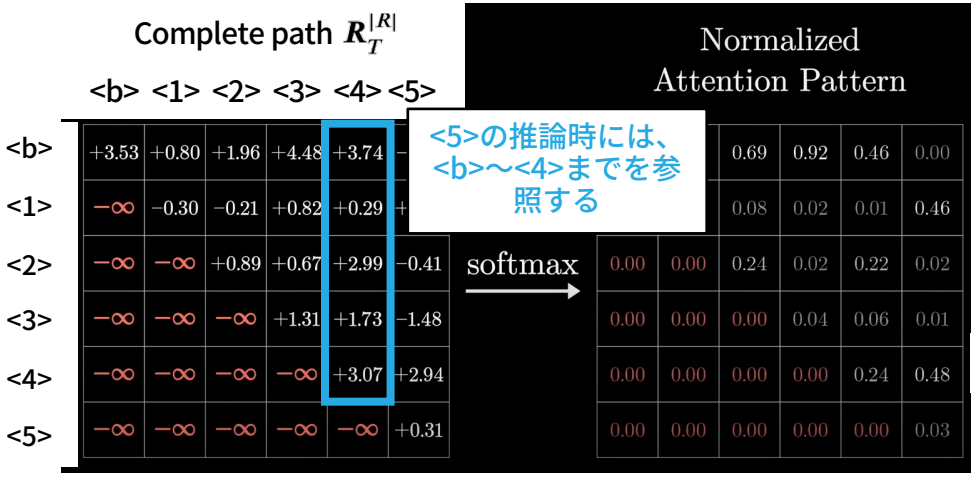
4.2.2 Masked Self-Attention（計算上の処理）

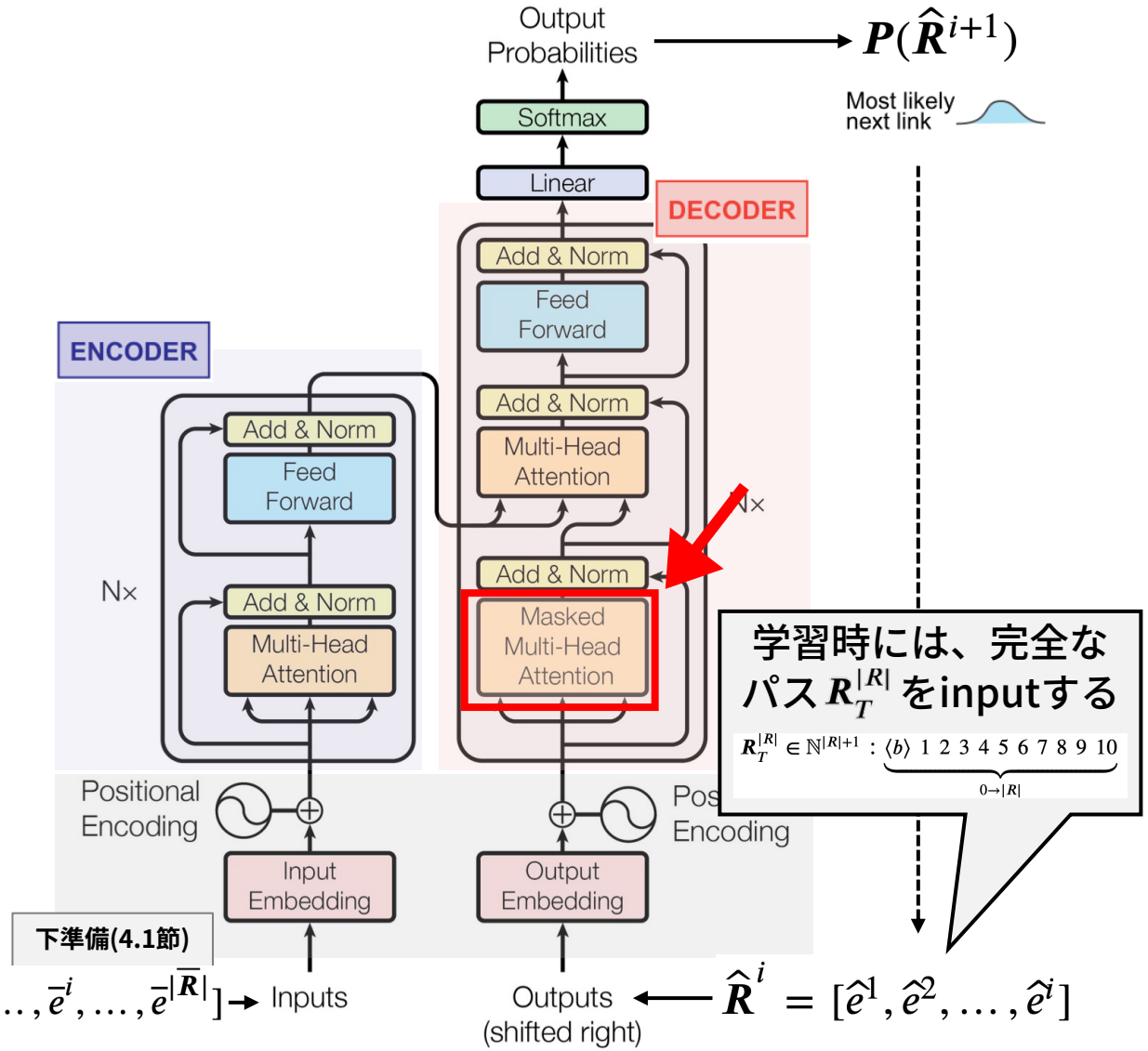
- i+1番目の推論時、デコーダーではi番目までのtraveled linksのSelf-Attentionを計算する

  ⇔学習時には、完全なパス $\boldsymbol{R}_T^{|R|}$ をinputするが、**学習時にi+1番目以降の正解を見て学習してほしくない**

- **⇨i+1番目以降のリンクへのattentionは-∞とする（マスクする）**

  →softmaxで0になる





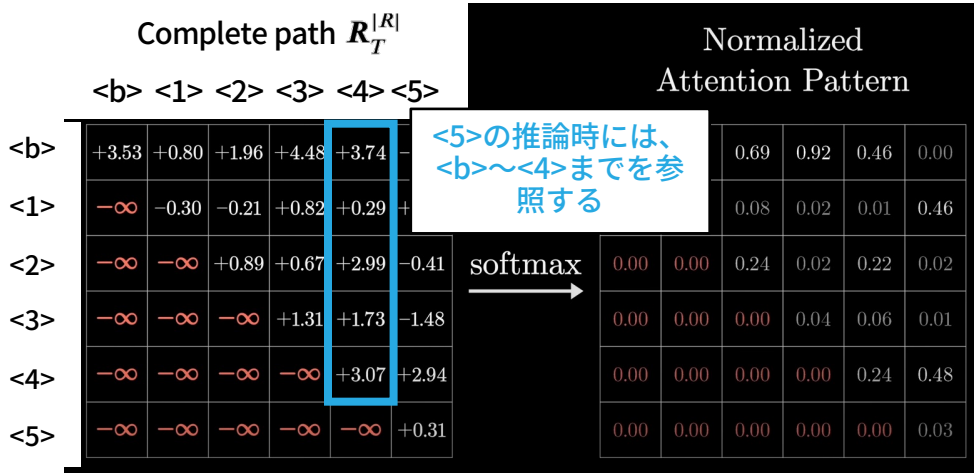
4.2.2 Masked Self-Attention（計算上の処理）

- i+1番目の推論時、デコーダーではi番目までのtraveled linksのSelf-Attentionを計算する

  ⇔学習時には、完全なパス $\boldsymbol{R}_T^{|R|}$ をinputするが、**学習時にi+1番目以降の正解を見て学習してほしくない**

- **⇨i+1番目以降のリンクへのattentionは-∞とする（マスクする）**

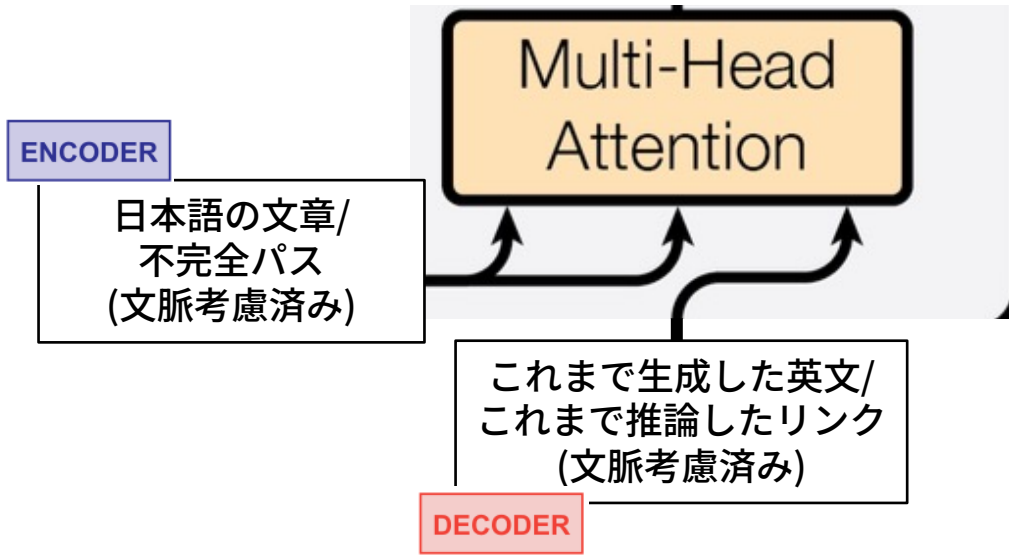
  →softmaxで0になる

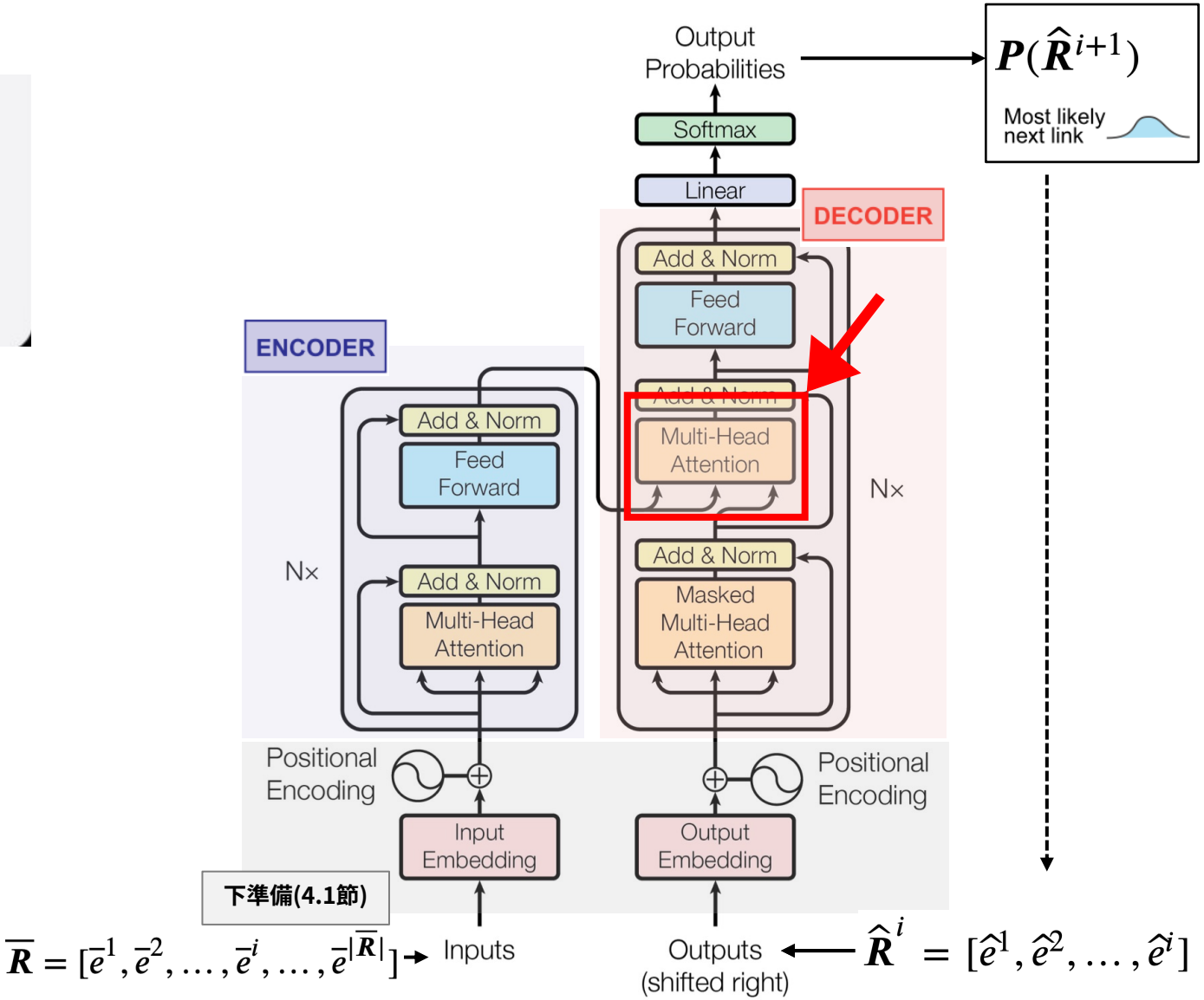


$$
\begin{aligned}
&\boldsymbol{Q} \leftarrow \boldsymbol{X}\boldsymbol{W}_q + \mathbf{1}\boldsymbol{b}_q^T \quad [[\boldsymbol{Q}\text{uery} \in \mathbb{R}^{l_x \times d_e}]] \\
&\boldsymbol{K} \leftarrow \boldsymbol{Z}\boldsymbol{W}_k + \mathbf{1}\boldsymbol{b}_k^T \quad [[\boldsymbol{K}\text{ey} \in \mathbb{R}^{l_z \times d_e}]] \\
&\boldsymbol{V} \leftarrow \boldsymbol{Z}\boldsymbol{W}_v + \mathbf{1}\boldsymbol{b}_v^T \quad [[\boldsymbol{V}\text{alue} \in \mathbb{R}^{l_z \times d_e}]] \\
&\boldsymbol{A} \leftarrow \mathrm{softmax}\left(\boldsymbol{M} + \frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_e}}\right) \quad [[\boldsymbol{A}\text{ttentionweight} \in \mathbb{R}^{l_x \times l_z}]] \\
&\boldsymbol{Y} \leftarrow \boldsymbol{A} \cdot \boldsymbol{V}
\end{aligned}
$$

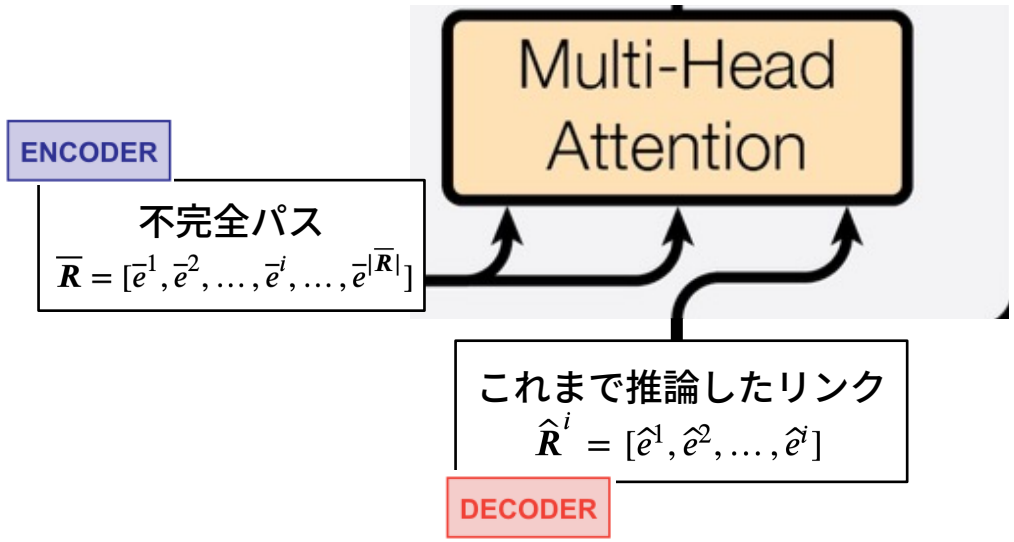
$$
\boldsymbol{M}_{\mathrm{U}}[i_x, i_z] = \begin{cases} 0, & i_z \le i_x, \\ -\infty, & \text{otherwise.} \end{cases}
$$

4.2.3 Cross-Attention





4.2.3 Cross-Attention



1 $\boldsymbol{Q} \leftarrow \boldsymbol{X}\boldsymbol{W}_q + \mathbf{1}\boldsymbol{b}_q^T$ $[[\boldsymbol{Q}\text{uery} \in \mathbb{R}^{l_x \times d_e}]]$

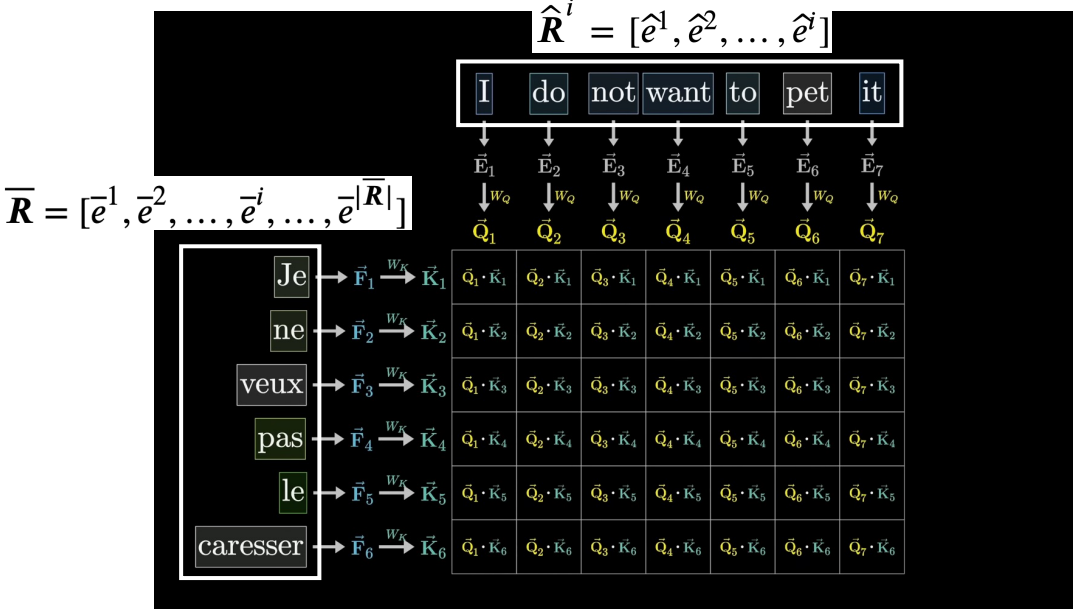
2 $\boldsymbol{K} \leftarrow \boldsymbol{Z}\boldsymbol{W}_k + \mathbf{1}\boldsymbol{b}_k^T$ $[[\boldsymbol{K}\text{ey} \in \mathbb{R}^{l_z \times d_e}]]$

3 $\boldsymbol{V} \leftarrow \boldsymbol{Z}\boldsymbol{W}_v + \mathbf{1}\boldsymbol{b}_v^T$ $[[\boldsymbol{V}\text{alue} \in \mathbb{R}^{l_z \times d_e}]]$

4 $\boldsymbol{A} \leftarrow \text{softmax}\left(\boldsymbol{M} + \frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_e}}\right)$ $[[\boldsymbol{A}\text{ttentionweight} \in \mathbb{R}^{l_x \times l_z}]]$

5 $\boldsymbol{Y} \leftarrow \boldsymbol{A} \cdot \boldsymbol{V}$

- 推論時に、$\overline{R}$ のうちどのリンクに注目すべきか



4.2.4 Multi-Head Attention

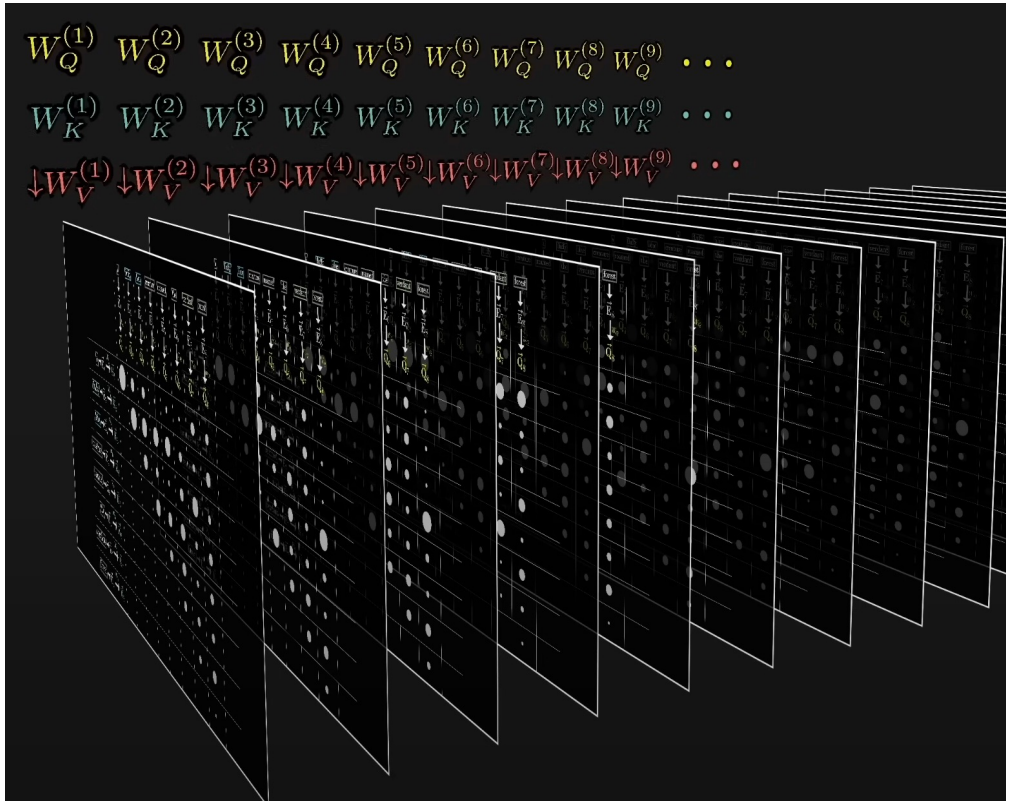
- 実際は何種類ものQ,K,Vの組み合わせを作り、**それぞれの組み合わせについてのAttentionとVの重み付き和を計算**し、それらを結合した行列を出力する

$$\boldsymbol{Y}^h = \mathrm{Attention}(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\mathcal{W}}^h_{qkv}, \boldsymbol{M}),$$

$$\mathrm{MHAttention}(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\mathcal{W}}, \boldsymbol{M}) = [\boldsymbol{Y}^1, \ldots, \boldsymbol{Y}^h, \ldots, \boldsymbol{Y}^H]\boldsymbol{W}_{\mathrm{o}} + \mathbf{1}\boldsymbol{b}_{\mathrm{o}}{}^T$$

⇨要素間の多様な表現（文法構造、意味構造…）を抽出

RoutesFormerではヘッド数８



4.2.5 その他の構成

Feed Forward

線形変換→ReLU→線形変換

$$\mathrm{FFN}(\boldsymbol{X}|\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}_2, \boldsymbol{b}_2) = \mathrm{ReLU}(\boldsymbol{X}\boldsymbol{W}_1 + \mathbf{1}\boldsymbol{b}_1{}^T)\boldsymbol{W}_2 + \mathbf{1}\boldsymbol{b}_2{}^T$$

（32次元（埋め込み次元）→ 中間層128次元 → 32次元）

Add & Norm

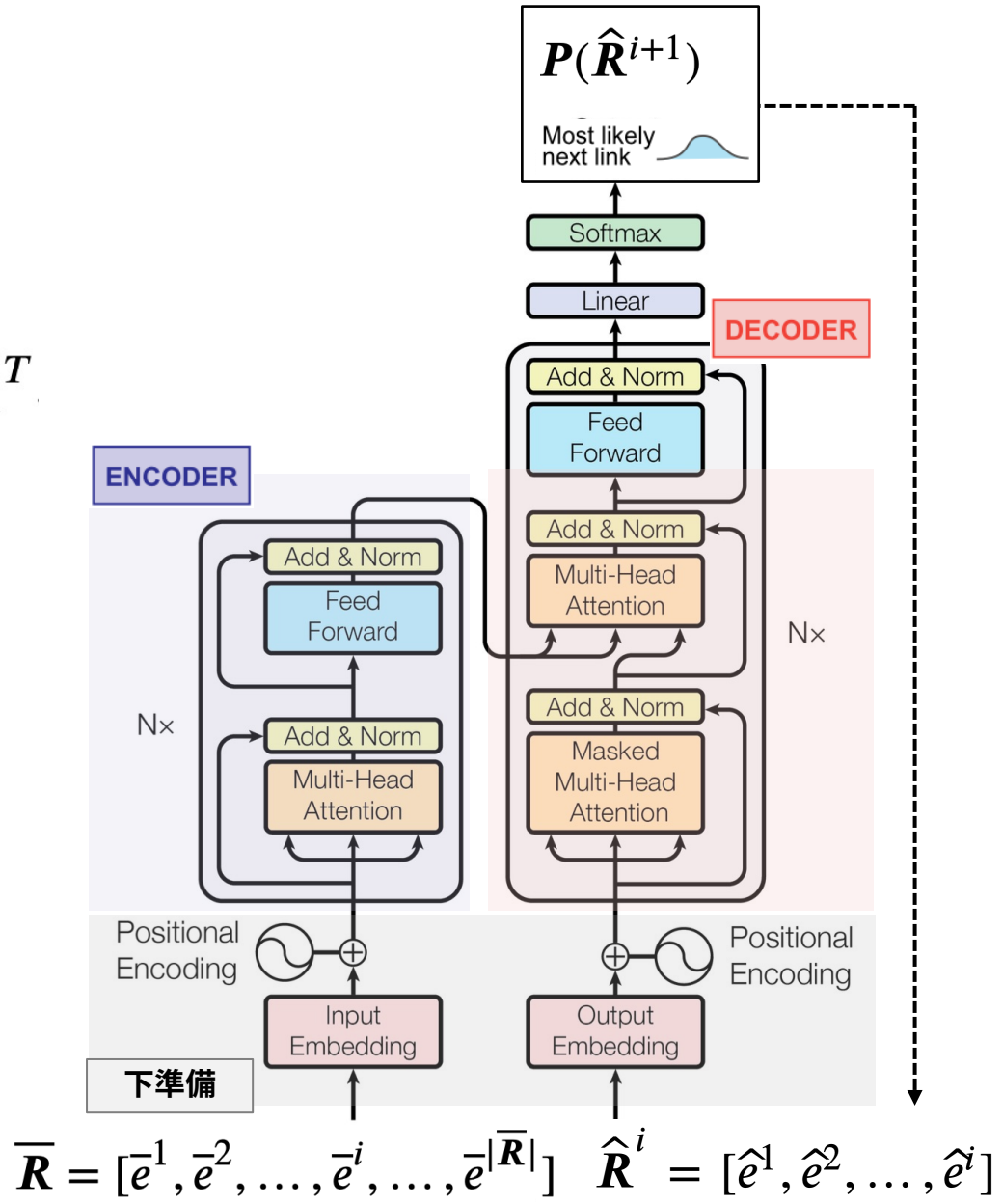
ネットワークを通す前の行列にネットワークの出力を足し（残差結合）、安定化のために正規化

$$\mathrm{LN}(\boldsymbol{X}^i|\boldsymbol{\gamma}, \boldsymbol{\beta}) = \boldsymbol{\gamma}\frac{\boldsymbol{X}^i - \mu}{\sqrt{\sigma^2 + \epsilon}} + \boldsymbol{\beta}$$

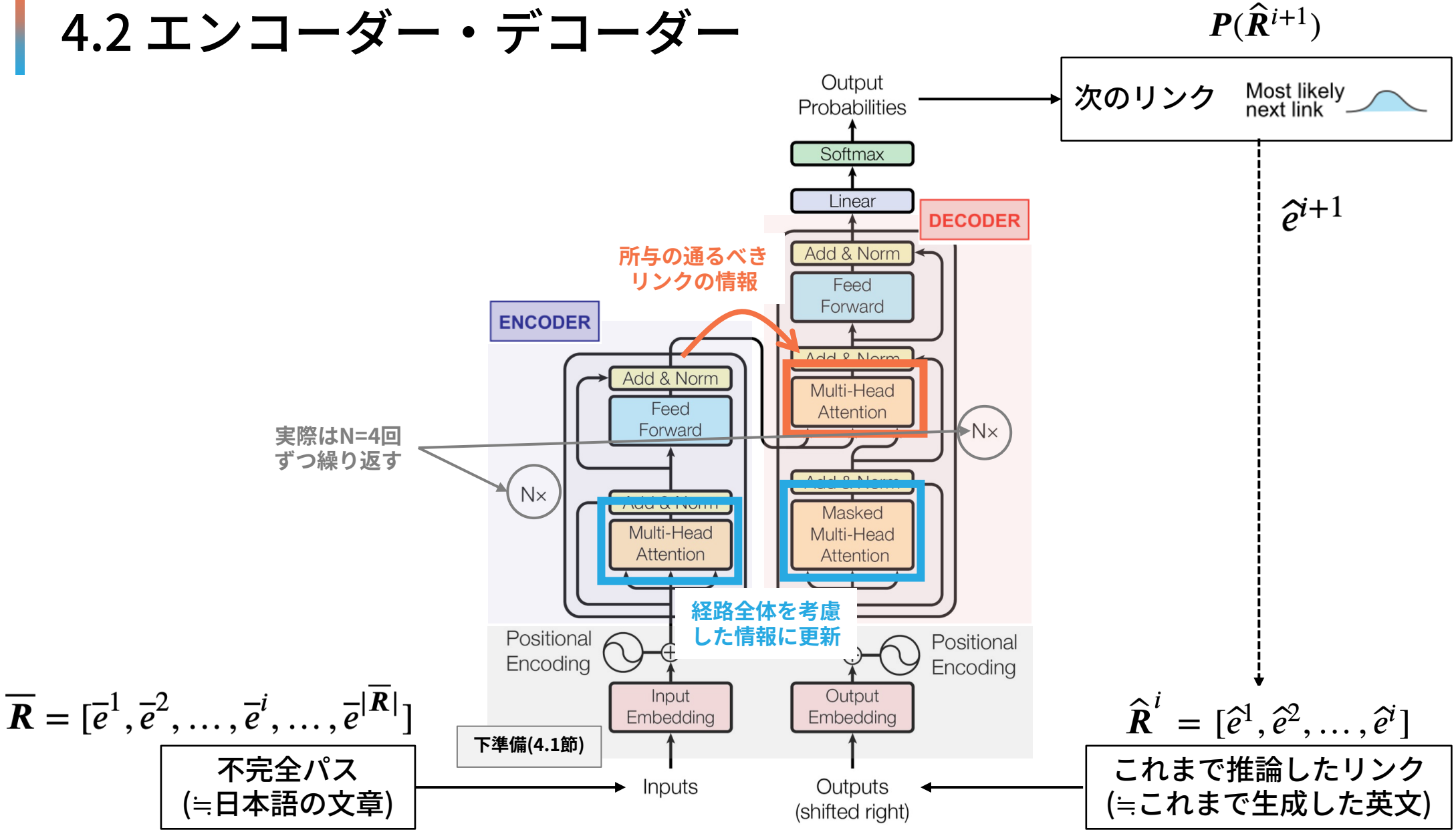
Softmax
Linear

出力を線形変換したのちに、softmax
→次の単語/リンクの選択確率として出力

$$\mathrm{Unembedding}(\boldsymbol{X}|\boldsymbol{W}_{\mathrm{ue}}) = \mathrm{softmax}(\boldsymbol{X}\boldsymbol{W}_{\mathrm{ue}})$$

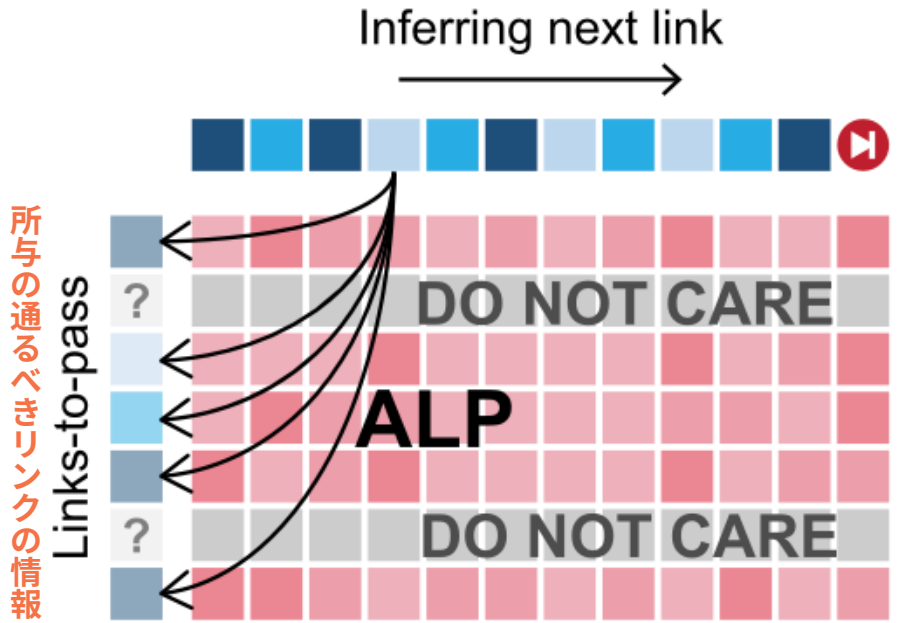


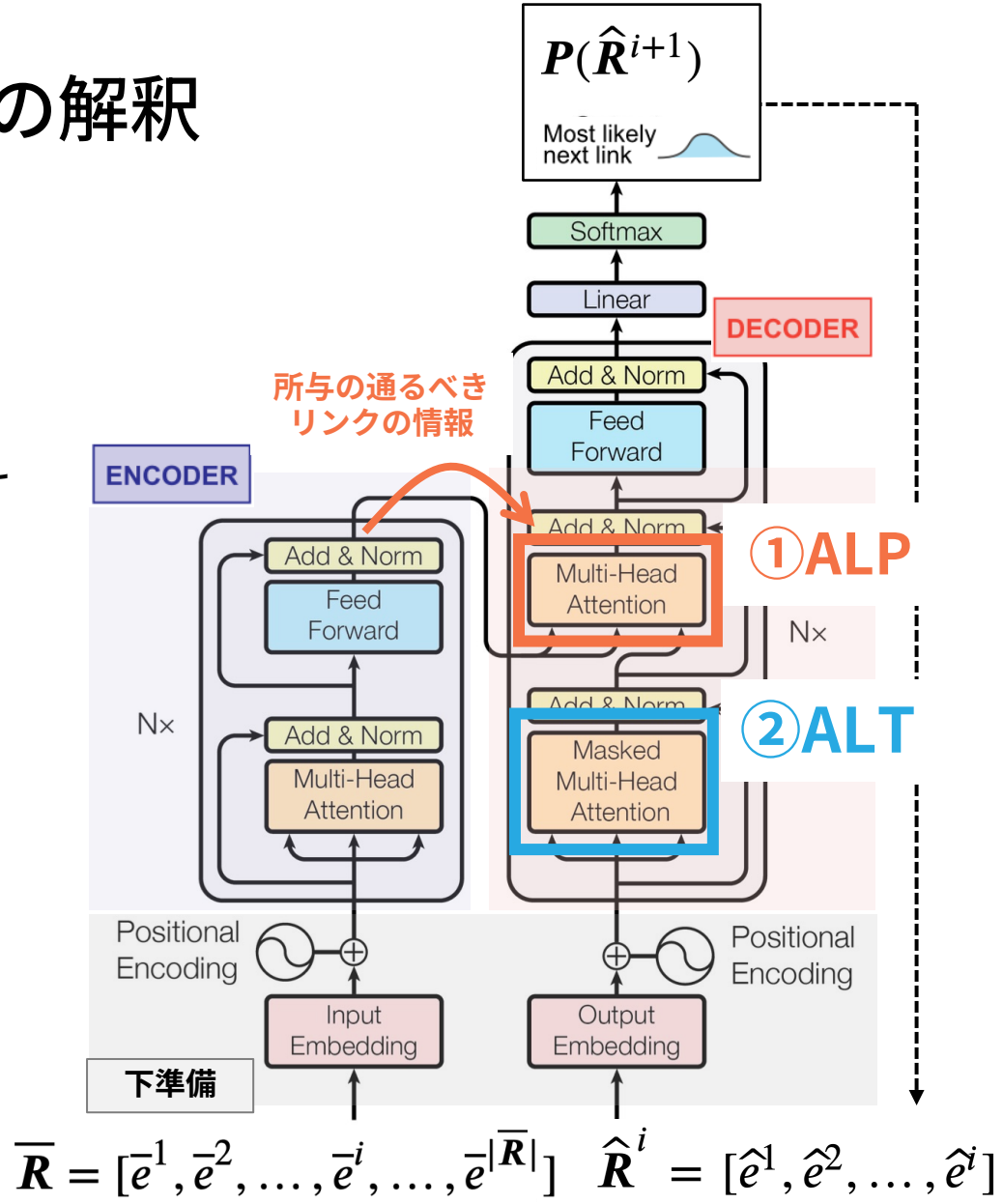
4.2 エンコーダー・デコーダー



4.3 RoutesFormerにおけるAttentionの解釈

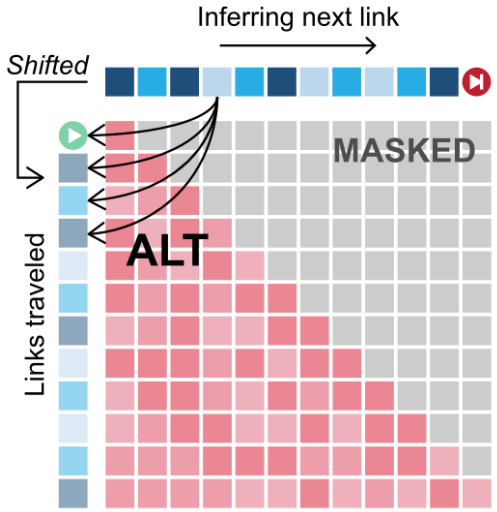
- **①Attention to links-to-pass（ALP）**
  - Decoder内のCross-Attention
  - i+1番目のリンクの推論時、スタートからゴールまでの**所与の通るべきリンク**に、どの程度注意を向けているか（＝そこからどの程度情報を得ているか）

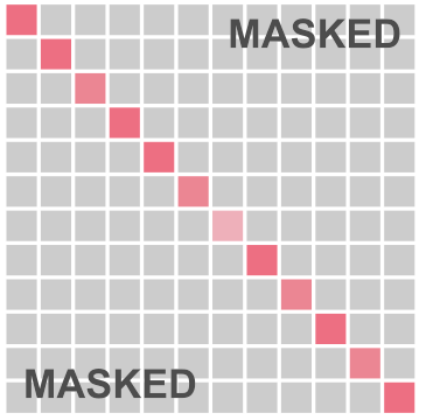


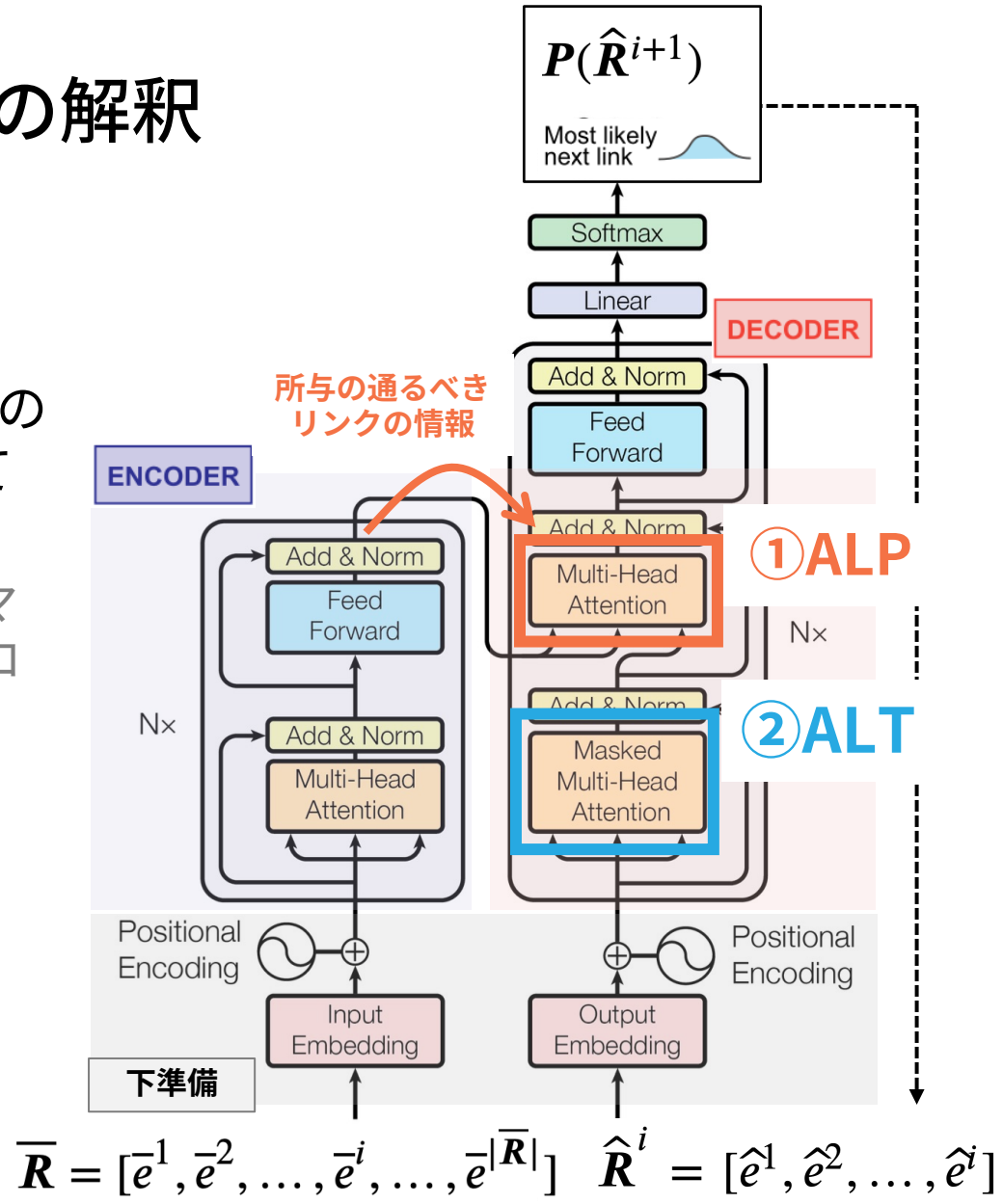


4.3 RoutesFormerにおけるAttentionの解釈

- **②Attention to links traveled（ALT）**
  - Decoder内のSelf-Attention
  - i+1番目のリンクの推論時、スタートからi番目までの**これまで辿ってきたリンク**にどの程度注意を向けているか（＝そこからどの程度情報を得ているか）
  - i+1番目のリンク推論時にi番目のリンク以外への注意をマスクすると、現在のリンクのみから情報を得る（≒マルコフ仮定）状況になる（右下図）→5.5節 Ablation Study







4.4 補足事項

- 次のリンクの選択確率を出した後、**隣接するリンクの中で**最も確率の高いものを出力する
- 学習時：クロスエントロピー誤差を用いてパラメタ調整
- 推論時：学習したパラメータで推論
  - **所与のリンクを通らない結果となる可能性**もある

    →その場合は所与リンクペア間の部分問題に分割して解く

if $\overline{R}$ *is not the subsequence of* $\hat{R}$ **then**

$$\hat{R} \leftarrow \mathrm{RFI}(G, [\bar{e}^1, \bar{e}^2], \hat{\theta}, \theta_{\mathrm{hyper}}) \oplus \cdots \oplus \mathrm{RFI}(G, [\bar{e}^{|\overline{R}|-1}, \bar{e}^{|\overline{R}|}], \hat{\theta}, \theta_{\mathrm{hyper}})$$

言語モデルをそのまま適応しているため、「所与のリンクを必ず通る」「隣接したリンクを選ぶ」という**経路選択特有の制約条件にモデルの中からアプローチすることはできていない**

5. Case study

5.1 データ設定・比較対象

- 2011年3月の、上海のタクシー8407台のGPSデータ
  - GPSのサンプリング間隔10秒
  - 人を乗せている区間のデータのみ用いる
- 80%を訓練用、20%をテスト用データとして用いる
- 疎な軌跡データの再現：(a)と(b)の混合
  - (a) 10%～90%の割合でデータをマスクする
  - (b) **GPSのサンプリング間隔**が**【60秒・180秒・300秒・420秒・OとDのみ】**である状況を再現してマスクする

比較対象のモデル

- Path size logit（PSL）
- DNN-PSL
- Recursive Logit（RL）
- Adversarial Inverse Reinforcement Learning（AIRL）
- Generative Adversarial Imitation Learning (GAIL) and Behavioral Cloning (BC)
- Dynamic LSTM (D-LSTM)

5.2 評価指標

- Total Link Length Accuracy（TLLA）
  【(推論パスのリンク長)/(正解パスのリンク長)】の平均

$$TLLA = \frac{1}{|P_{\text{test}}|} \sum_{n=1}^{|P_{\text{test}}|} \frac{\sum_{e \in \hat{R}_n \cap R_n} e.\text{length}}{\sum_{e \in R_n} e.\text{length}}$$

- **Edit Distance（ED）**
  【(add,delete,modifyの必要回数)/(リンク数)】の平均

$$ED = \frac{1}{|P_{\text{test}}|} \sum_{n=1}^{|P_{\text{test}}|} \min\left(\frac{\text{Edit}(\hat{R}_n, R_n)}{|R_n|}, 1\right)$$

- BiLingual Evaluation Understudy score（BLEU）
  - N-gramのマッチング度

$$\text{BLEU} = \frac{1}{|P_{\text{test}}|} \sum_{n=1}^{|P_{\text{test}}|} \text{Bleu}(\hat{R}_n, R_n)$$

- Jensen–Shannon Divergence
  - 確率分布の類似性

$$\text{JSD}(P, Q) = \frac{1}{2}\text{KLD}\left(P, \frac{P+Q}{2}\right) + \frac{1}{2}\text{KLD}\left(Q, \frac{P+Q}{2}\right)$$

$$\text{KLD}(P, Q) = \sum_n P_n \log \frac{P_n}{Q_n}$$

- Path completion rate (PCR)
  - 【所与リンクを繋げるのに成功したパス数/全パス数】

5.3 推論精度の比較

- RoutesFormerは**全てのタスク・全ての精度指標で旧来のモデルを凌駕**
  - PCR=所与のリンクを通れるか　は最下位
- 他のモデルは、タスクにより順位が入れ替わる

DNN-PSL > PSL
- DNNで非線形効用をうまく近似
- 間隔60sでは逆転：短いパスでの効用の違い

PSL > RL
- 実情に反した多くの経路の考慮

AIRL :
- サンプルの非効率性、いい関数を見つけづらい

Intervals	Models	Single-destination					Multi-destination				
		TLLA↑	ED↓	BLEU↑	JSD↓	PCR↑	TLLA↑	ED↓	BLEU↑	JSD↓	PCR↑
60 s	PSL (1999)	0.833#6	0.195#6	0.826#6	0.301#6	1.000#1	0.879#7	0.142#7	0.888#7	0.391#6	1.000#1
	DNN-PSL (2020)	0.812#7	0.232#7	0.811#7	0.332#7	1.000#1	0.882#6	0.136#6	0.889#6	0.448#7	1.000#1
	RL (2013)	0.901#5	0.100#5	0.898#5	0.215#5	1.000#1	0.967#2	0.047#3	0.963#2	0.109#2	1.000#1
	BC (2022)	0.921#4	0.080#4	0.924#4	0.213#4	0.851#7	0.956#4	0.062#5	0.945#4	0.146#5	0.792#7
	AIRL (2022)	0.976#2	0.029#2	0.972#2	0.081#2	0.995#4	0.960#3	0.046#2	0.958#3	0.121#3	0.988#4
	D-LSTM (2022)	0.961#3	0.054#3	0.950#3	0.138#3	0.981#5	0.944#5	0.050#4	0.941#5	0.133#4	0.985#5
	RoutesFormer	**0.982#1**	**0.028#1**	**0.975#1**	**0.074#1**	0.947#6	**0.975#1**	**0.037#1**	**0.969#1**	**0.101#1**	0.951#6
180 s	PSL (1999)	0.757#7	0.234#6	0.778#6	0.599#6	1.000#1	0.840#6	0.175#7	0.833#7	0.482#7	1.000#1
	DNN-PSL (2020)	0.820#5	0.171#4	0.839#4	0.413#4	1.000#1	0.849#5	0.171#6	0.838#6	0.440#6	1.000#1
	RL (2013)	0.626#6	0.389#7	0.577#7	0.760#7	1.000#1	0.821#7	0.157#4	0.839#5	0.336#3	1.000#1
	BC (2022)	0.824#4	0.184#5	0.830#5	0.506#5	0.877#7	0.858#4	0.165#5	0.850#3	0.369#5	0.843#7
	AIRL (2022)	0.910#3	0.090#1	0.911#2	0.248#2	0.993#4	0.863#3	0.138#2	0.871#2	0.318#2	0.990#4
	D-LSTM (2022)	0.911#2	0.123#3	0.885#3	0.252#3	0.986#5	0.887#2	0.146#3	0.843#4	0.355#4	0.988#5
	RoutesFormer	**0.922#1**	**0.090#1**	**0.916#1**	**0.218#1**	0.961#6	**0.906#1**	**0.105#1**	**0.906#1**	**0.246#1**	0.966#6
300 s	PSL (1999)	0.770#6	0.243#6	0.768#6	0.558#5	1.000#1	0.780#6	0.222#5	0.781#5	0.479#7	1.000#1
	DNN-PSL (2020)	0.835#4	0.173#4	0.831#4	0.339#4	1.000#1	0.791#5	0.217#4	0.787#4	0.468#5	1.000#1
	RL (2013)	0.548#7	0.440#7	0.520#7	0.876#7	1.000#1	0.765#7	0.225#6	0.771#7	0.439#4	1.000#1
	BC (2022)	0.802#5	0.222#5	0.795#5	0.598#6	0.888#7	0.799#4	0.239#7	0.780#6	0.477#6	0.867#7
	AIRL (2022)	0.877#2	0.133#2	0.868#2	0.318#3	0.990#4	0.811#3	0.209#3	0.802#3	0.420#3	0.993#4
	D-LSTM (2022)	0.872#3	0.141#3	0.868#2	0.290#2	0.982#5	0.831#2	0.181#2	0.833#2	0.397#2	0.989#5
	RoutesFormer	**0.902#1**	**0.113#1**	**0.895#1**	**0.258#1**	0.970#6	**0.860#1**	**0.158#1**	**0.857#1**	**0.325#1**	0.972#6
420 s	PSL (1999)	0.736#6	0.251#6	0.760#6	0.546#5	1.000#1	0.755#5	0.250#4	0.752#5	0.494#6	1.000#1
	DNN-PSL (2020)	0.798#4	0.180#4	0.823#4	0.305#3	1.000#1	0.760#4	0.244#3	0.758#4	0.481#5	1.000#1
	RL (2013)	0.501#7	0.460#7	0.497#7	0.887#7	1.000#1	0.743#7	0.261#6	0.736#7	0.480#4	1.000#1
	BC (2022)	0.777#5	0.244#5	0.772#5	0.619#6	0.889#7	0.747#6	0.280#7	0.739#6	0.521#7	0.869#7
	AIRL (2022)	0.851#3	0.163#3	0.838#3	0.335#4	0.995#4	0.765#3	0.250#4	0.762#3	0.463#3	0.992#4
	D-LSTM (2022)	0.862#2	0.129#2	0.860#2	0.286#2	0.989#5	0.794#2	0.223#2	0.795#2	0.407#2	0.986#5
	RoutesFormer	**0.890#1**	**0.125#1**	**0.884#1**	**0.268#1**	0.977#6	**0.819#1**	**0.190#1**	**0.828#1**	**0.360#1**	0.973#6
Only OD	PSL (1999)	0.790#5	0.230#5	0.789#5	0.616#5	1.000#1	0.755#4	0.251#4	0.753#4	0.496#4	1.000#1
	DNN-PSL (2020)	0.803#4	0.221#4	0.792#4	0.362#3	1.000#1	0.756#3	0.244#3	0.760#3	0.484#3	1.000#1
	RL (2013)	0.459#7	0.489#7	0.466#7	0.935#7	1.000#1	0.670#6	0.327#6	0.668#7	0.567#6	1.000#1
	BC (2022)	0.711#6	0.274#6	0.746#6	0.638#6	0.989#6	0.654#7	0.340#7	0.680#6	0.584#7	0.982#7
	AIRL (2022)	0.806#3	0.191#3	0.811#3	0.367#4	1.000#1	0.755#4	0.307#5	0.734#5	0.524#5	0.998#4
	D-LSTM (2022)	0.873#2	0.141#2	0.870#2	0.336#2	0.994#5	0.770#2	0.240#2	0.762#2	0.451#2	0.996#5
	RoutesFormer	**0.876#1**	**0.140#1**	**0.874#1**	**0.329#1**	0.989#6	**0.791#1**	**0.234#1**	**0.786#1**	**0.436#1**	0.993#6

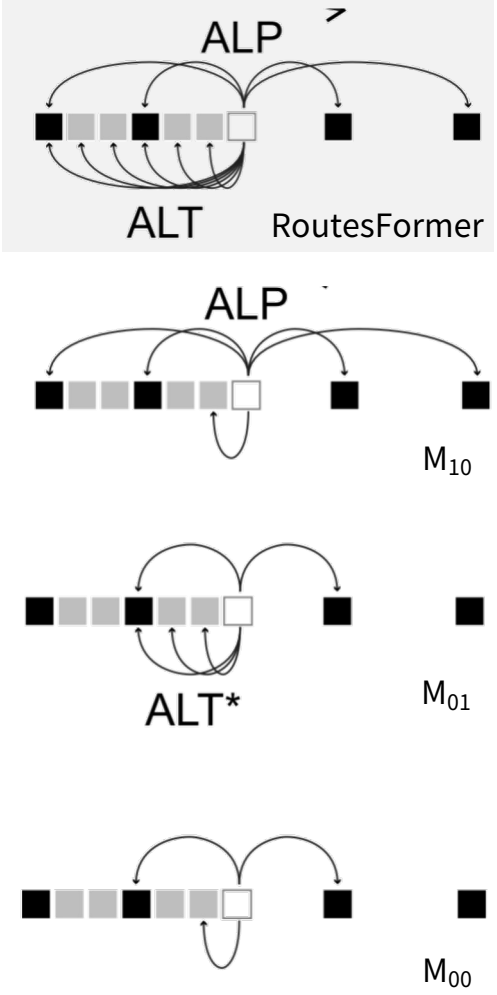
5.4 計算時間の比較

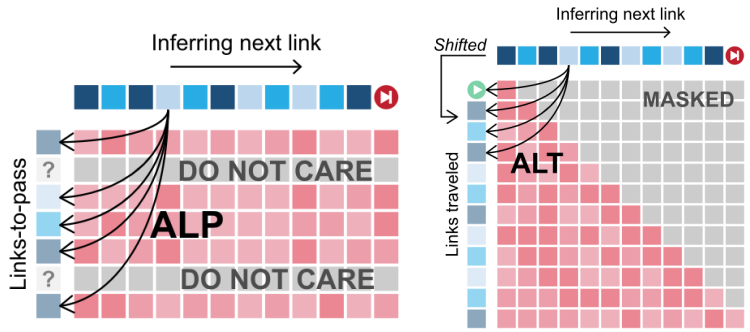
- 並列処理可能なRoutesFormerは、パラメタ数の多さの割に**短い学習時間**
  - ⇔**推論時間は長い**
- パスベースのPSL, DNN-PSLはメモリを使ってパス候補を事前に格納しているため、最短の学習・推論時間
- 古典的なRLはモデルサイズ・計算時間とも小さい
- AIRLは逆強化学習という特性上広範なサンプル収集が必要 → サイズの割に長い学習時間

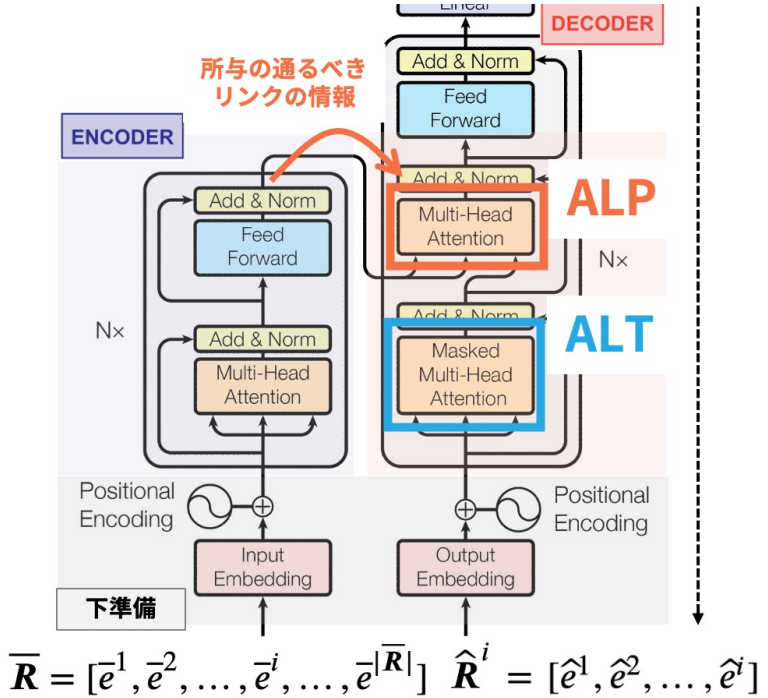
Models	Parameters	Training time(s) under different training size				Inference time(s)
		1k	2k	4k	8k	
PSL (1999)	13	30.5	52.4	81.1	170.3	0.00185
DNN-PSL (2020)	5249	41.7	86.3	157.7	289.4	0.00247
RL (2013)	56	417.2	1121.8	1910.3	2912.1	0.0072
BC (2022)	41 996	709.5	1346.2	2478.5	5021.3	0.0352
AIRL (2022)	41996(actor)	9935.6	13 948.1	15 822.9	19 204.2	0.0381
D-LSTM (2022)	803 629	945.7	2023.1	3981.6	8281.0	0.0498
RoutesFormer	2 260 653	2858.2	5943.0	10 531.1	19 140.5	0.436

5.5 Ablation study

- ALP, ALTができないようにした**切除モデルで検証し、Attention機構の意義を分析**する
  - M_{10}：ALPはできるが、ALTでは**現在リンクからしか情報を得られない**
  - M_{01}：ALPは**推論中の前後の所与リンクのみ**に向けられ、**ALTも最後に通った所与リンクまで**からしか情報を得られない
  - M_{00}：ALPは**推論中の前後の所与リンクのみ**に向けられ、 ALTでは**現在リンクからしか情報を得られない**







$\overline{\boldsymbol{R}} = [\overline{e}^1, \overline{e}^2, \ldots, \overline{e}^i, \ldots, \overline{e}^{|\overline{\boldsymbol{R}}|}]$ $\hat{\boldsymbol{R}}^i = [\hat{e}^1, \hat{e}^2, \ldots, \hat{e}^i]$

5.5.1 精度の比較

#で書かれた上付き文字は全モデルとの比較における順位

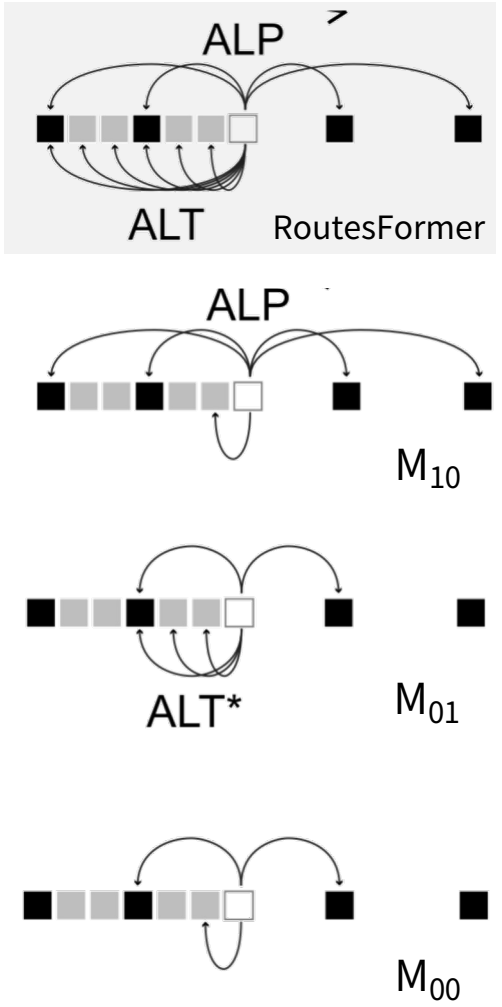
Sampling intervals /Sampled links	Models	Single-destination				Multi-destination			
		TLLA↑	ED↓	BLEU↑	JSD↓	TLLA↑	ED↓	BLEU↑	JSD↓
60 s	M_{00}	$0.946^{\#4}$	$0.078^{\#4}$	$0.931^{\#4}$	$0.195^{\#4}$	$0.934^{\#6}$	$0.081^{\#6}$	$0.927^{\#6}$	$0.172^{\#6}$
	M_{01}	$0.964^{\#3}$	$0.038^{\#3}$	$0.964^{\#3}$	$0.096^{\#3}$	$0.950^{\#4}$	$0.041^{\#2}$	$0.945^{\#4}$	$0.124^{\#4}$
	M_{10}	$0.968^{\#3}$	$0.066^{\#4}$	$0.942^{\#4}$	$0.180^{\#4}$	$0.955^{\#5}$	$0.067^{\#6}$	$0.949^{\#4}$	$0.153^{\#6}$
	$\boldsymbol{M_{11}}$	$\mathbf{0.982}^{\#1}$	$\mathbf{0.028}^{\#1}$	$\mathbf{0.975}^{\#1}$	$\mathbf{0.074}^{\#1}$	$\mathbf{0.975}^{\#1}$	$\mathbf{0.037}^{\#1}$	$\mathbf{0.969}^{\#1}$	$\mathbf{0.101}^{\#1}$
180 s	M_{00}	$0.889^{\#4}$	$0.161^{\#4}$	$0.842^{\#4}$	$0.369^{\#4}$	$0.868^{\#3}$	$0.157^{\#4}$	$0.861^{\#3}$	$0.383^{\#6}$
	M_{01}	$0.910^{\#3}$	$0.109^{\#3}$	$0.903^{\#3}$	$0.256^{\#4}$	$0.887^{\#2}$	$0.130^{\#2}$	$0.884^{\#2}$	$0.375^{\#5}$
	M_{10}	$0.894^{\#4}$	$0.159^{\#4}$	$0.858^{\#4}$	$0.351^{\#4}$	$0.872^{\#3}$	$0.145^{\#3}$	$0.878^{\#3}$	$0.314^{\#2}$
	$\boldsymbol{M_{11}}$	$\mathbf{0.922}^{\#1}$	$\mathbf{0.090}^{\#1}$	$\mathbf{0.916}^{\#1}$	$\mathbf{0.218}^{\#1}$	$\mathbf{0.906}^{\#1}$	$\mathbf{0.105}^{\#1}$	$\mathbf{0.906}^{\#1}$	$\mathbf{0.246}^{\#1}$
300 s	M_{00}	$0.862^{\#4}$	$0.203^{\#5}$	$0.811^{\#5}$	$0.421^{\#5}$	$0.826^{\#3}$	$0.241^{\#7}$	$0.816^{\#3}$	$0.411^{\#3}$
	M_{01}	$0.890^{\#2}$	$0.124^{\#2}$	$0.886^{\#2}$	$0.281^{\#2}$	$0.842^{\#2}$	$0.172^{\#2}$	$0.835^{\#2}$	$0.386^{\#2}$
	M_{10}	$0.871^{\#4}$	$0.189^{\#4}$	$0.833^{\#4}$	$0.412^{\#4}$	$0.813^{\#3}$	$0.208^{\#3}$	$0.821^{\#3}$	$0.403^{\#3}$
	$\boldsymbol{M_{11}}$	$\mathbf{0.902}^{\#1}$	$\mathbf{0.113}^{\#1}$	$\mathbf{0.895}^{\#1}$	$\mathbf{0.258}^{\#1}$	$\mathbf{0.860}^{\#1}$	$\mathbf{0.158}^{\#1}$	$\mathbf{0.857}^{\#1}$	$\mathbf{0.325}^{\#1}$
420 s	M_{00}	$0.843^{\#4}$	$0.226^{\#5}$	$0.796^{\#5}$	$0.465^{\#5}$	$0.742^{\#8}$	$0.264^{\#7}$	$0.751^{\#6}$	$0.487^{\#5}$
	M_{01}	$0.870^{\#2}$	$0.136^{\#3}$	$0.877^{\#2}$	$0.308^{\#4}$	$0.816^{\#2}$	$0.202^{\#2}$	$0.824^{\#2}$	$0.390^{\#2}$
	M_{10}	$0.859^{\#3}$	$0.204^{\#5}$	$0.820^{\#5}$	$0.425^{\#5}$	$0.811^{\#2}$	$0.245^{\#3}$	$0.788^{\#3}$	$0.443^{\#3}$
	$\boldsymbol{M_{11}}$	$\mathbf{0.890}^{\#1}$	$\mathbf{0.125}^{\#1}$	$\mathbf{0.884}^{\#1}$	$\mathbf{0.268}^{\#1}$	$\mathbf{0.819}^{\#1}$	$\mathbf{0.190}^{\#1}$	$\mathbf{0.828}^{\#1}$	$\mathbf{0.360}^{\#1}$
OD only	M_{00}	$0.836^{\#3}$	$0.253^{\#7}$	$0.781^{\#6}$	$0.531^{\#5}$	$0.753^{\#5}$	$0.298^{\#5}$	$0.740^{\#5}$	$0.526^{\#6}$
	M_{01}	$0.876^{\#1}$	$0.140^{\#1}$	$0.874^{\#1}$	$0.329^{\#1}$	$0.791^{\#1}$	$0.234^{\#1}$	$0.786^{\#1}$	$0.436^{\#1}$
	M_{10}	$0.836^{\#3}$	$0.253^{\#7}$	$0.781^{\#6}$	$0.531^{\#5}$	$0.753^{\#5}$	$0.298^{\#5}$	$0.740^{\#5}$	$0.526^{\#6}$
	$\boldsymbol{M_{11}}$	$\mathbf{0.876}^{\#1}$	$\mathbf{0.140}^{\#1}$	$\mathbf{0.874}^{\#1}$	$\mathbf{0.329}^{\#1}$	$\mathbf{0.791}^{\#1}$	$\mathbf{0.234}^{\#1}$	$\mathbf{0.786}^{\#1}$	$\mathbf{0.436}^{\#1}$

5.5.1 精度の比較

- M_{01}が比較的高い精度 ⇔ M_{10}、M_{00}の精度は低い
  - →**ALTの有無がより大きく影響**
  - 単一目的地のデータでは特に、過去の経路の影響大（⇔複数目的地では多様な経路選択）
  - 疎なサンプリング間隔ほど、ALTが顕著な影響（⇔密なサンプリング間隔では相対的にALPの影響が大きく）

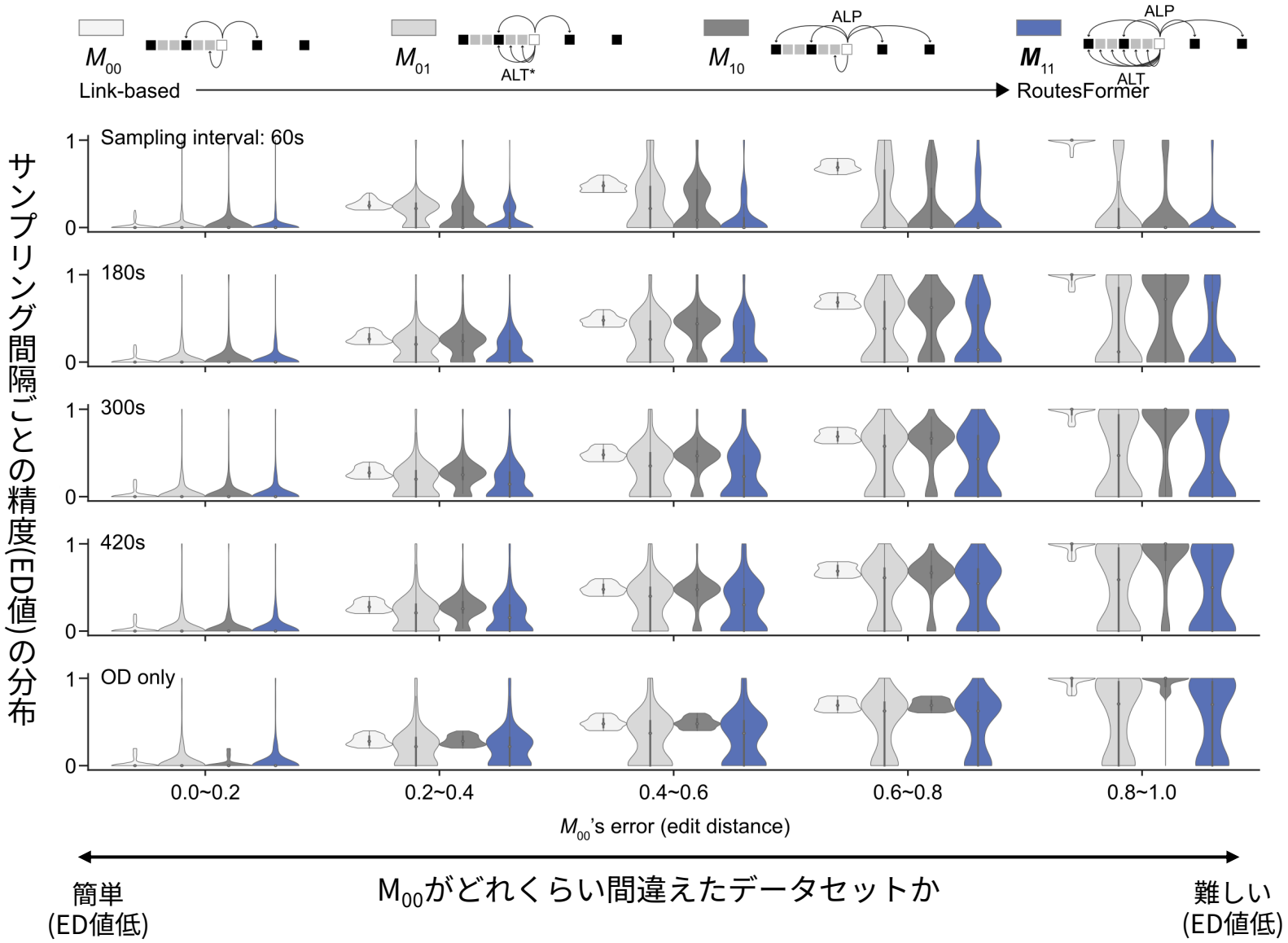
Edit Distanceの平均値（低いほど良い）

モデル	単一目的地データ	複数目的地データ
RoutesFormer	0.099	0.145
M_{10}（ALTなし）	**0.174**	**0.193**
M_{01}（ALPなし）	0.109	0.156
M_{00}（両方なし）	**0.184**	**0.208**



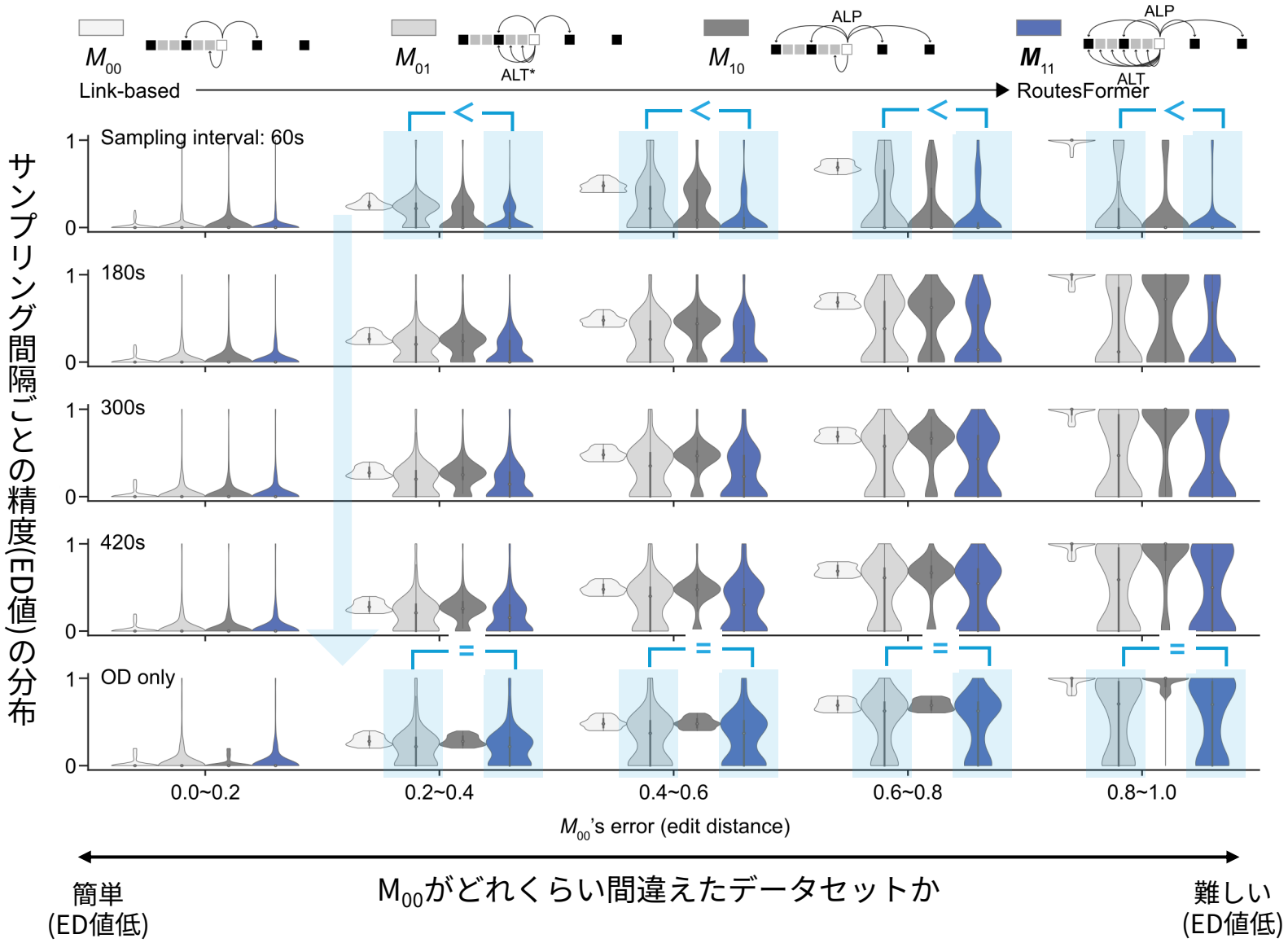
5.5.2 精度の分布比較

- Edit Distanceの分布を、M_{00}の精度に基づいて区分したデータセットグループごとに、各モデルについて可視化
- 分布の**上が細く下が太い**（＝低いED値が多い）ほど良い



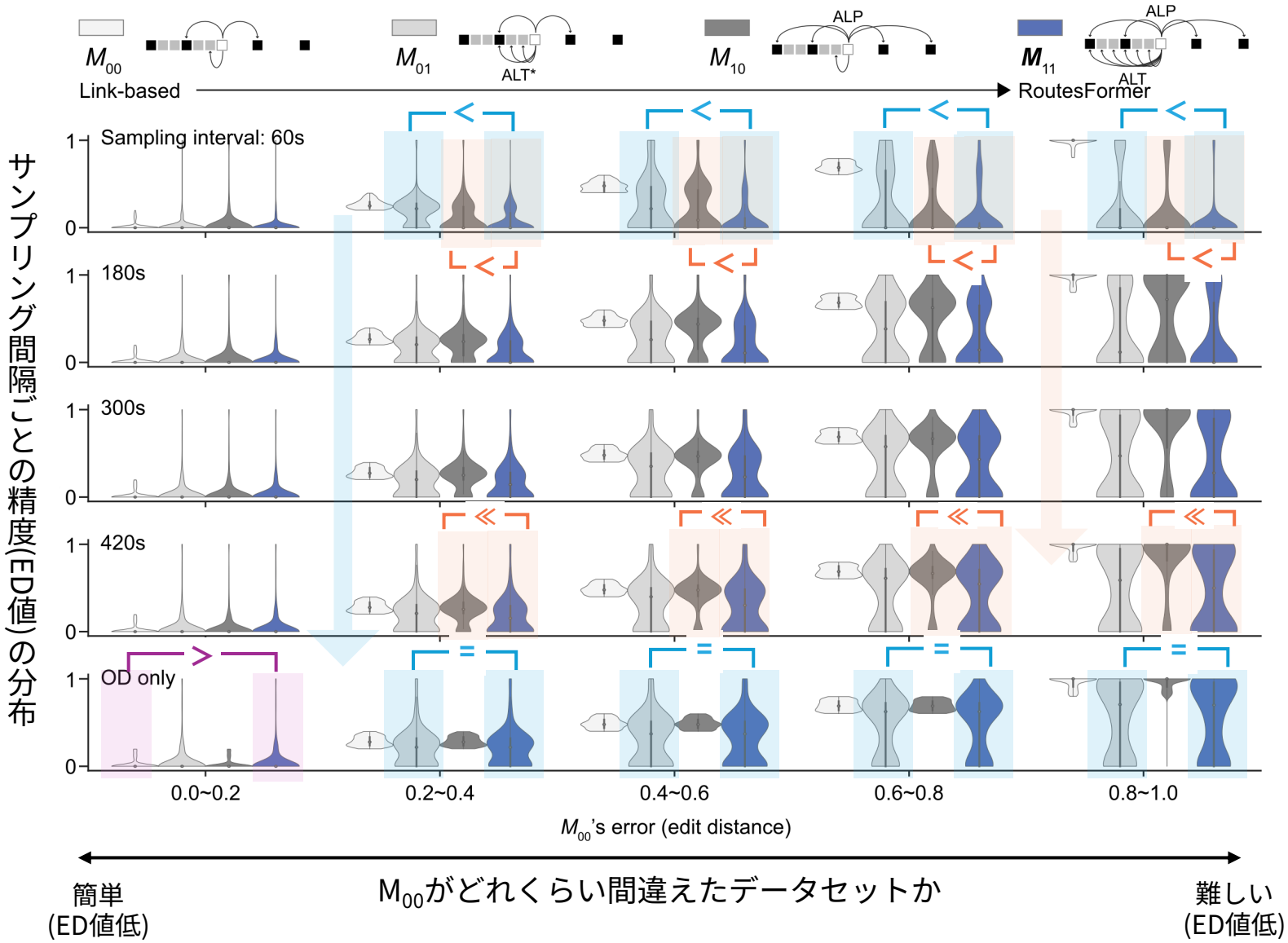
5.5.2 精度の分布比較

	M_{01} (ALPなし)
密なサンプリング間隔	悪い
疎なサンプリング間隔	**本家に近づく**



5.5.2 精度の分布比較

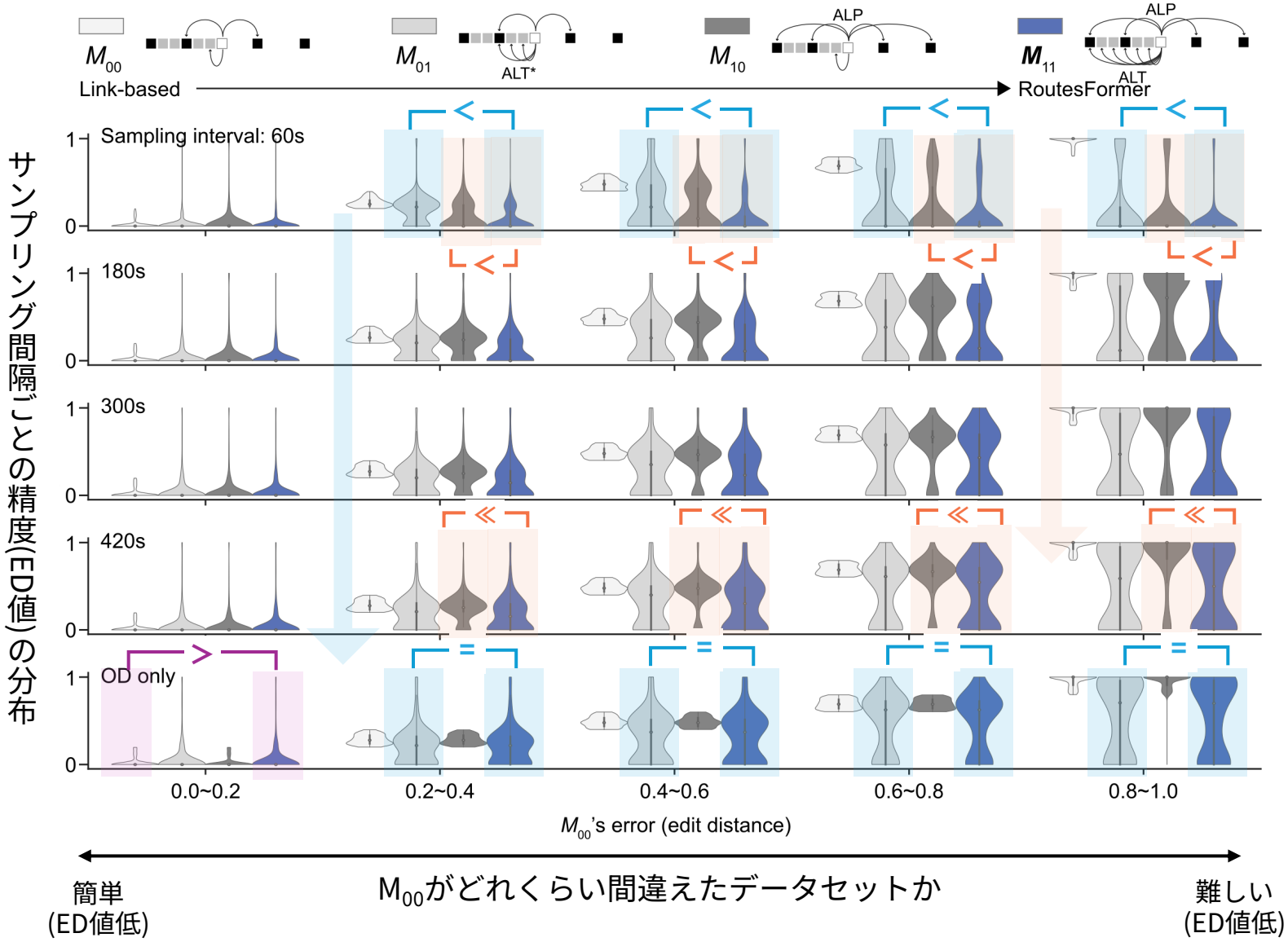
	M_{01} (ALPなし)	M_{10} (ALTなし)
密なサンプリング間隔	悪い	悪い
疎なサンプリング間隔	**本家に近づく**	**より悪い**



5.5.2 精度の分布比較

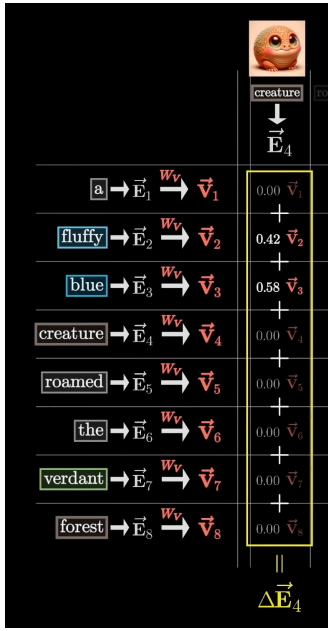
	M_{01} (ALPなし)	M_{10} (ALTなし)
密なサンプリング間隔	悪い	悪い
疎なサンプリング間隔	**本家に近づく**	**より悪い**

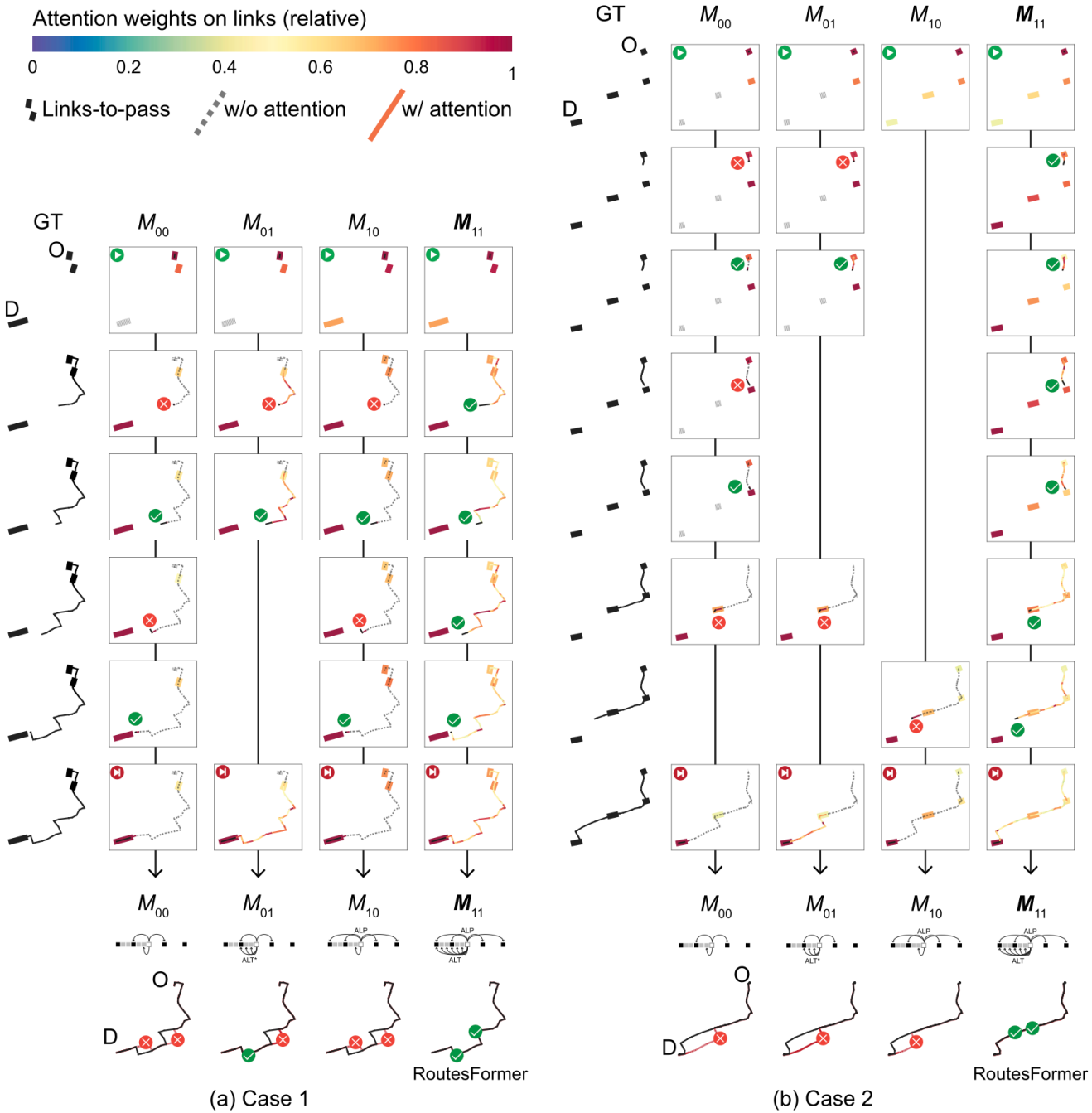
- 疎な間隔であるほど過去の情報を参照する余地が大きく、ALTの重要性が増す
- M_{00}の結果がいい場合も→嗜好の多様性



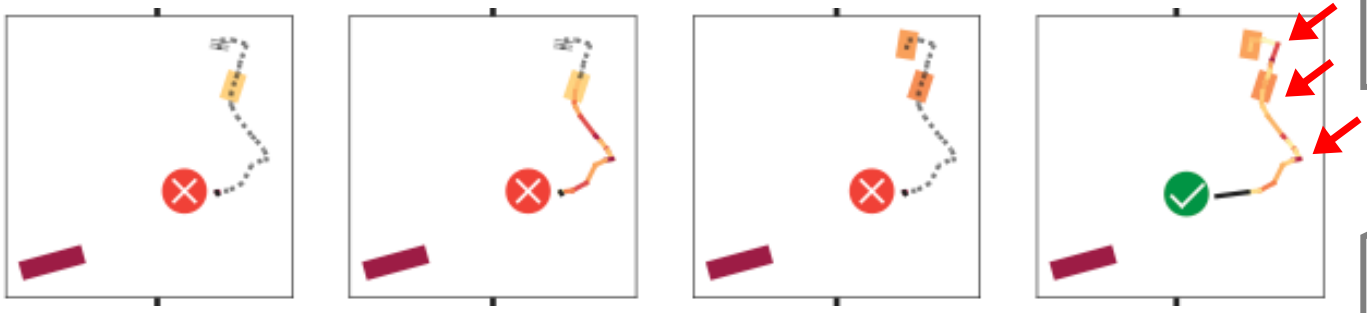
5.5.3 推論結果の比較

- あるリンクが推論されたとき、**どのリンクの情報に特に注目して決定されたか**は、**各リンクへのattentionの値の大小を見る**ことでわかる
- 各リンクへのattentionの値（相対値）と、次のリンクの推論結果をステップごとに可視化
- 本家モデル以外は、「推論を誤ったステップ」と「その後正しい道に戻ったステップ」を抽出している



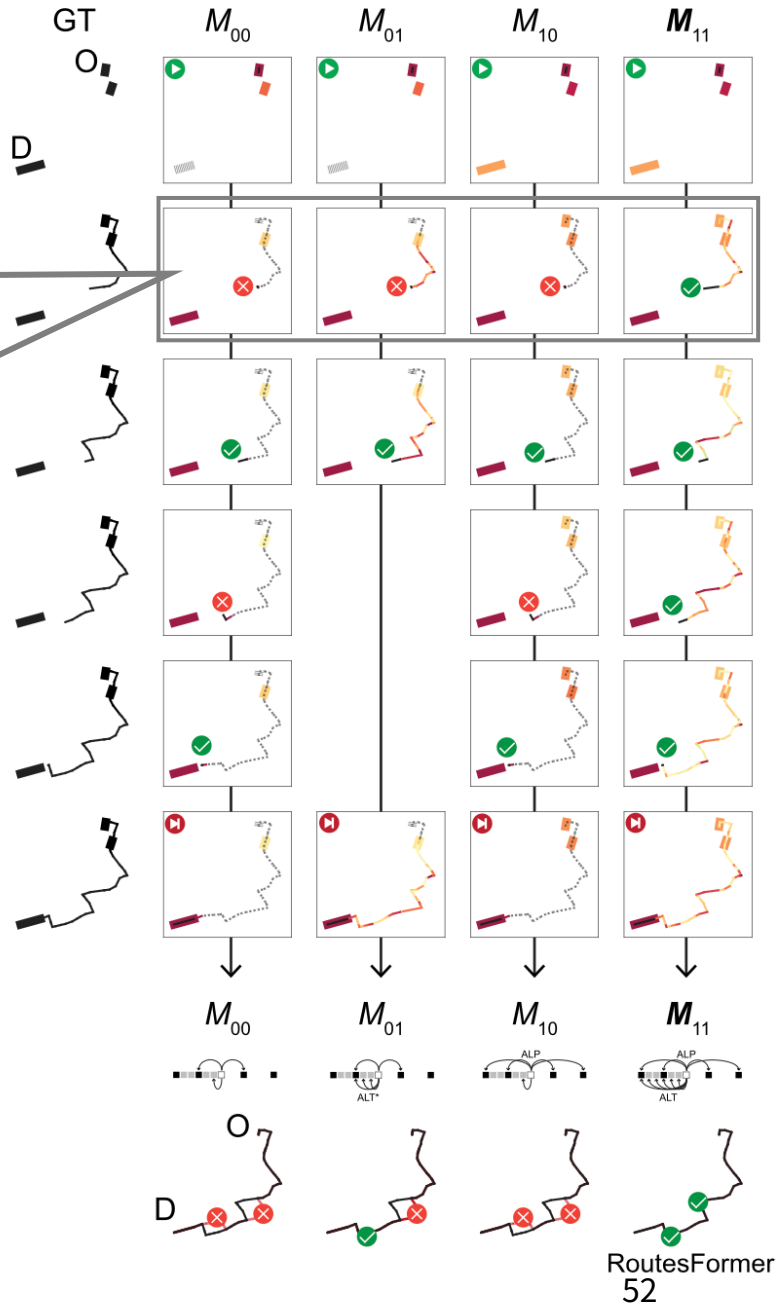


5.5.3 推論結果の比較

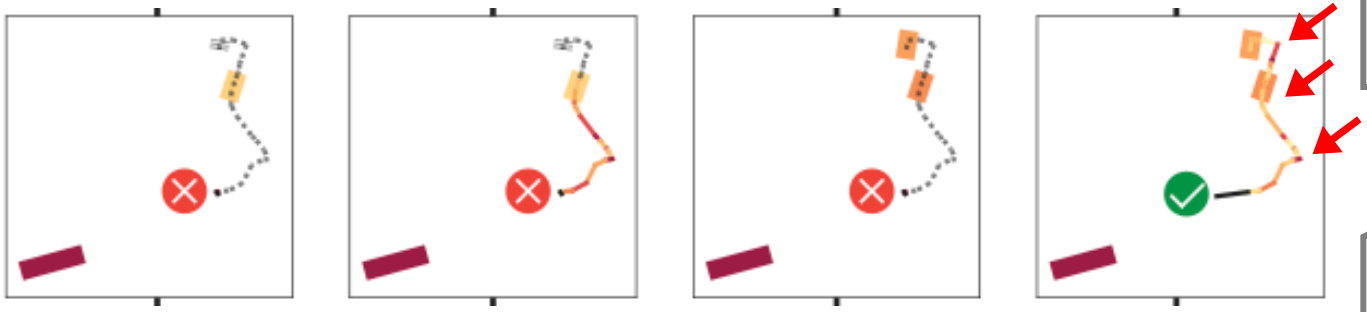


正解したRoutesFormerでは、現在リンクではなく、
- いくつかのより前に通ったリンク
- Oや、その次の通るべき所与(=太い)のリンク

に、高いattentionを示している

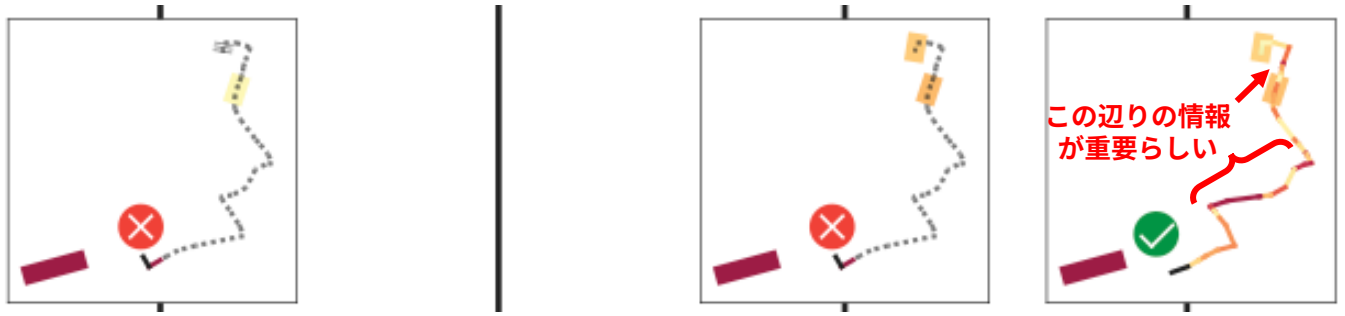


5.5.3 推論結果の比較

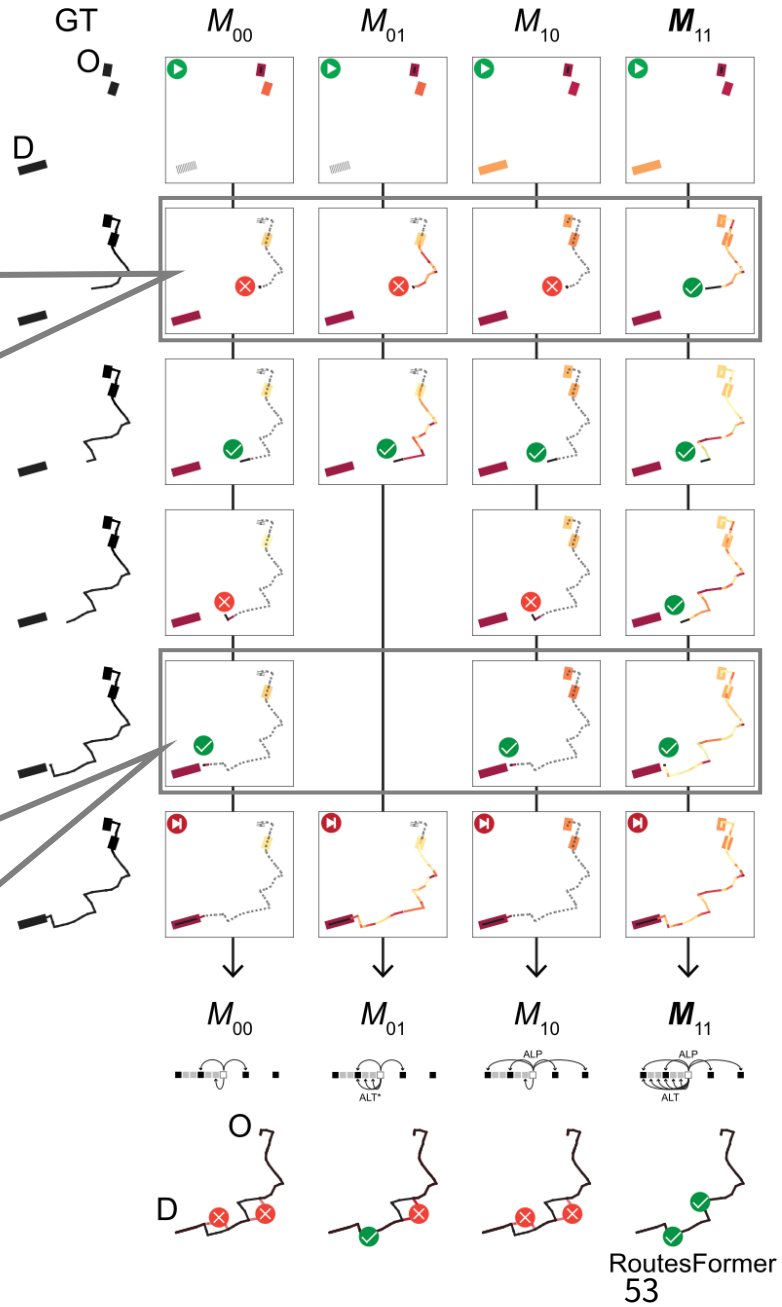


正解したRoutesFormerでは、現在リンクではなく、
- いくつかのより前に通ったリンク
- Oや、その次の通るべき所与(=太い)のリンク

に、高いattentionを示している



ALTのないモデルで、局所的な回り道が発生
（このケース以外にも多数観察される）



6. Conclusion

6.1 Conclusion

✓RoutesFormerは２つの面で既存の経路選択モデルの問題を解決した：

①**経路全体の情報を一括**して扱える

②**マルコフ性を仮定しない**

✓**細かい特徴をモデル化**でき、少ないデータ量でも**安定した学習**が可能

✓全タスクで**既存の経路選択モデルを上回る精度**を記録した

✓アブレーション分析により、**経路全体・過去の選択経路との関係を考慮する重要性を示す**とともに、現実世界の選択嗜好をよりよく理解する方法を提示した

6.2 発展可能性

今後のモデルの活用・改善：

- 既存の効用推定アーキテクチャよりも**言語モデルへのフィッティングに重き**を置いている
  - リンク特徴量は一般的なもののみで、ドメイン知識を入れていないので転移性が低い可能性
  - 全リンクを予測候補にした上で、隣接するものから選ぶという非合理性
- Attentionのメモリ消費と計算時間が大きい。①豊富で十分カバーされた訓練データがあり、②単純な仮定に収まらない多様な③連続的な選択の関係をみるときに向いている

さらなる研究：

✓数少ない軌跡データから広大なネットワークの行動を学習する場合は、強化学習との組み合わせが考えられる

✓ネットワーク間転移・都市間転移には、グラフ学習との組み合わせが有効

✓より精緻な推論のため、軌跡上の点の時空間情報の考慮が重要

……> **ViRT（倉澤さん修論）**

所感

- ４章まで辛かったが、最高精度で既存のモデルを全員倒しそのままAblation studyに突入する５章の勢いが良かった
  - 精度には感動したが、タクシーデータゆえの推定のしやすさはありそう
- Ablation Studyまで読んでようやくattention機構の役割や意義を含めて納得することができた
  - Attentionの可視化が本当面白い！
  - そこの推論にそんな手前のリンク使うなら、今までRecursive Logitとかで一体何を計算していたのか。。
- あまりにもtransformerそのままのモデル設計で驚いた
  - リンクの隣接制約・所与のパスの通過制約は、アウトプットで考慮しているだけだった
  - ほぼそのままの設計でも高精度を出すattention機構の万能さに感服
- リンク特徴量を埋め込んだことの精度への寄与の分析がなく残念
  - 機械学習なので使える特徴量は埋め込めるだけ埋め込んで損はない・・・？