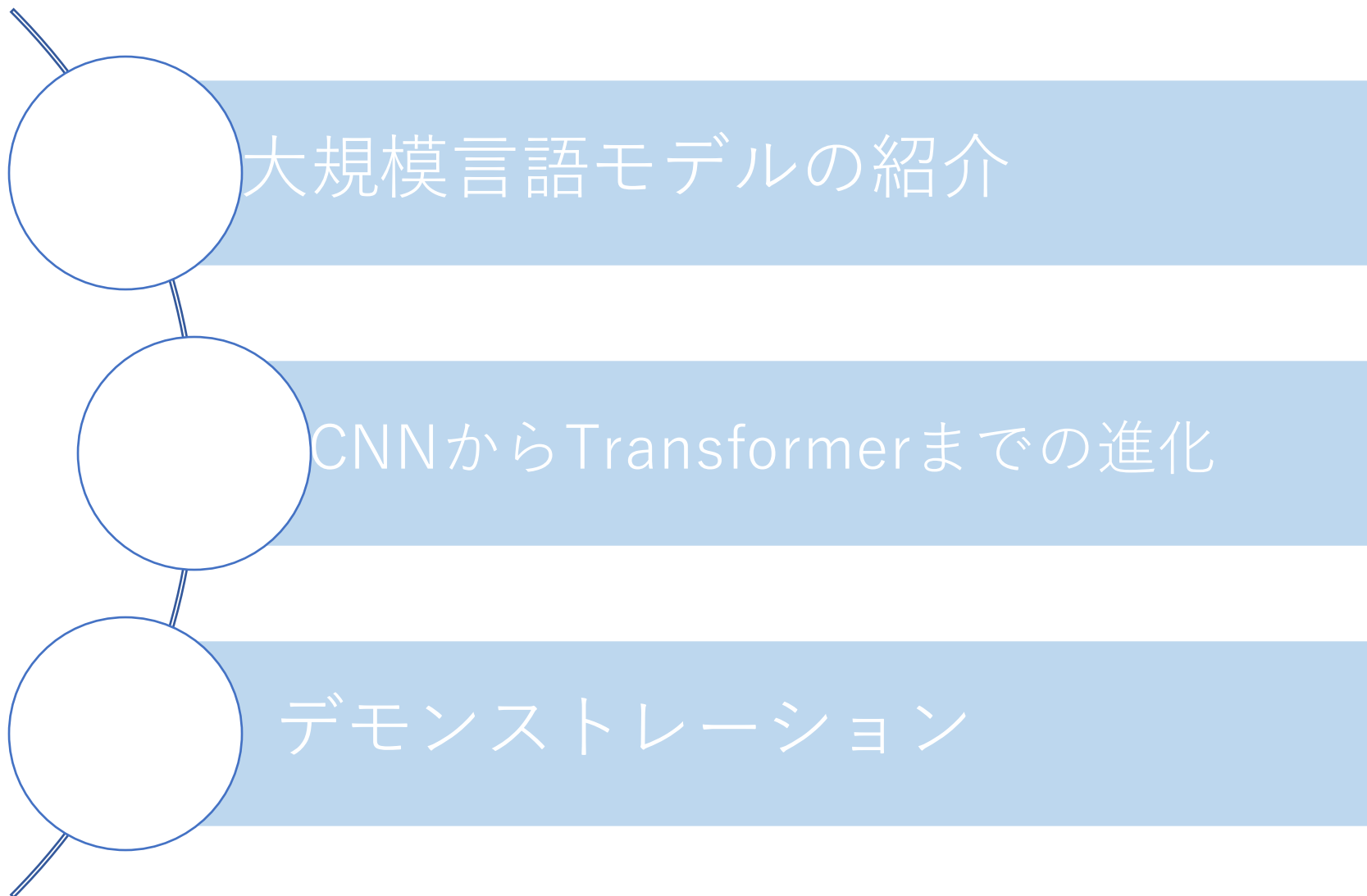


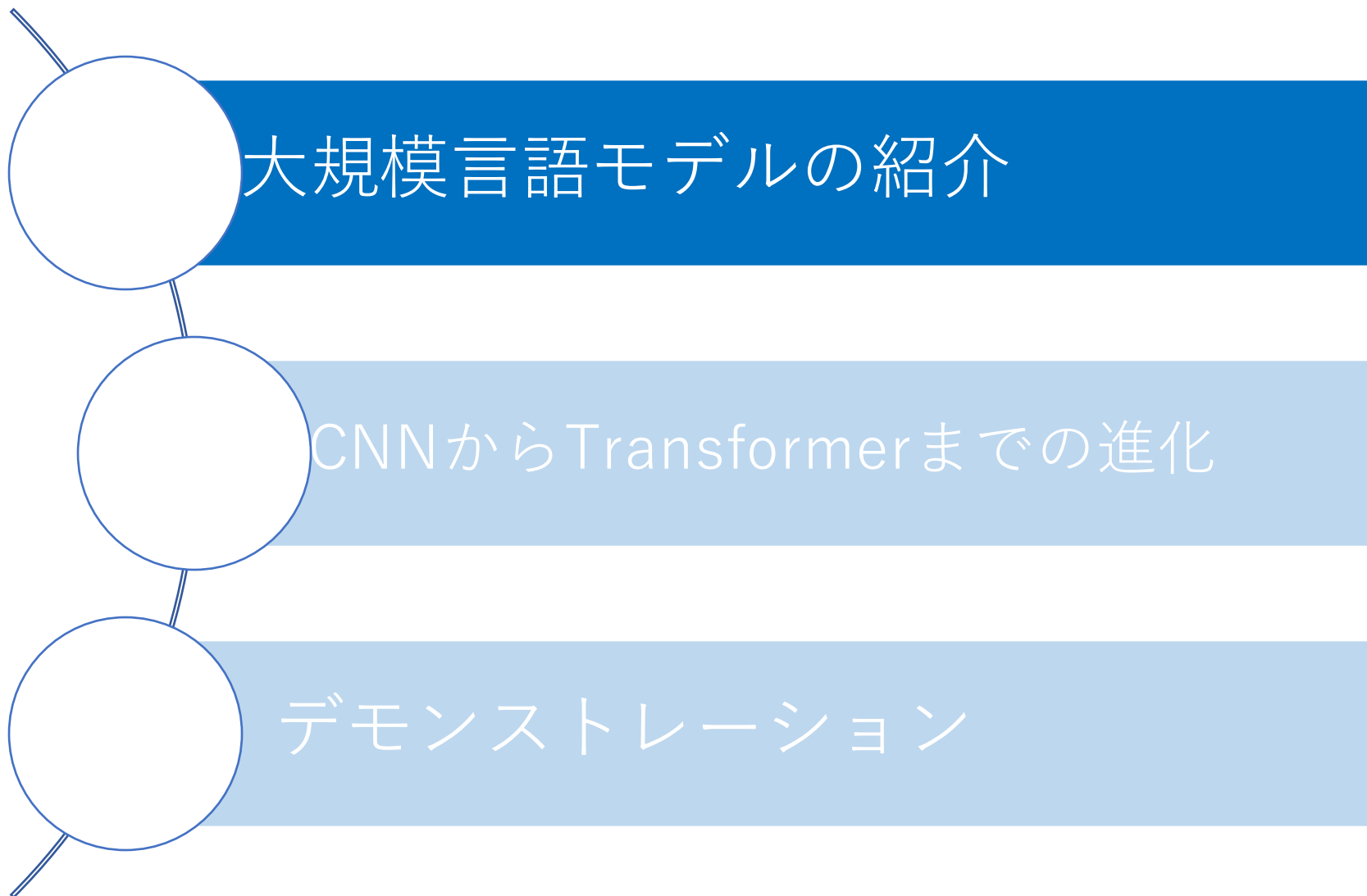
なぜTransformerが 王道のアーキテクチャになったか ～CNN、RNN、Transformerの回顧

丁 世堯

京都大学 情報学研究科 伊藤研

2025年3月28日



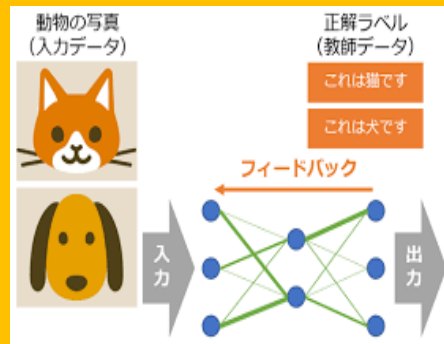


AI & 機械学習

人工知能

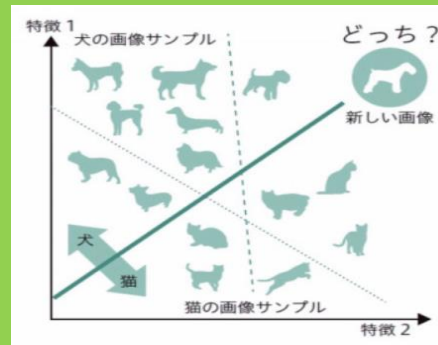
機械学習

教師あり学習



Learning method to learn a set of feature data and correct labels

教師なし学習



From feature data only
Learning Methods

強化学習



An "environment" is provided,
and the AI acquires data from the
environment.

大規模言語モデル (LLM: Large language models)

MO

ChatGPTができることを教えてください



はい、私(ChatGPT)ができることは以下の通りです。



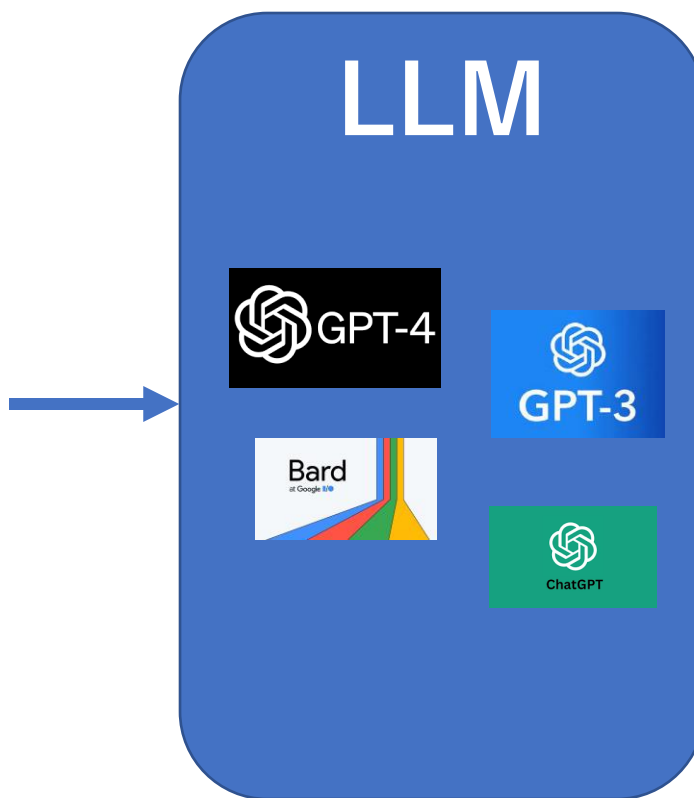
1. 質問に答える：私はあらゆるトピックについての質問に答えることができます。例えば、科学、歴史、文化、芸術、技術、スポーツ、料理など、幅広い分野に関する質問に対応できます。
2. 文章生成：私は文章を自動生成することができます。例えば、記事、小説、詩、レポート、ビジネス文書などを生成することができます。
3. 対話：私は自然な言語での対話を行うことができます。ユーザーが言葉を入力すると、それに応じた回答を生成することができます。
4. 翻訳：私は様々な言語を相互に翻訳することができます。
5. 推論：私は与えられた情報から論理的な推論を導き出すことができます。
6. 検索：私はインターネット上の情報を検索して、回答を提供することができます。
7. 感情認識：私は文章や発言の感情を認識することができます。
8. テキスト要約：私は与えられた長い文章を要約することができます。

以上が私(ChatGPT)ができる主なことです。ただし、私にはまだ限界があります。完璧な回答を提供できないこともあります。

↺ Regenerate response ↻

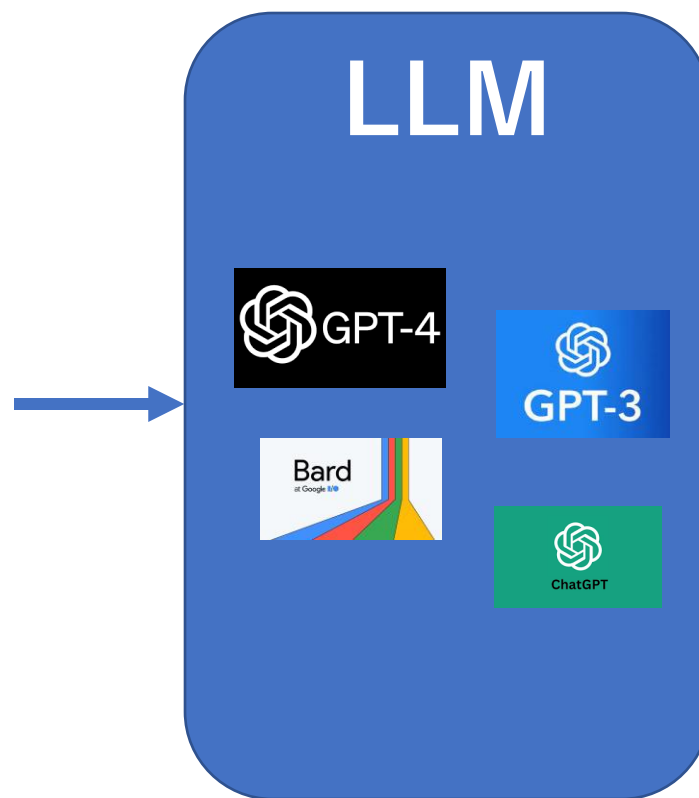
大規模言語モデル (LLM: Large language models)

今朝、スタバで



大規模言語モデル (LLM: Large language models)

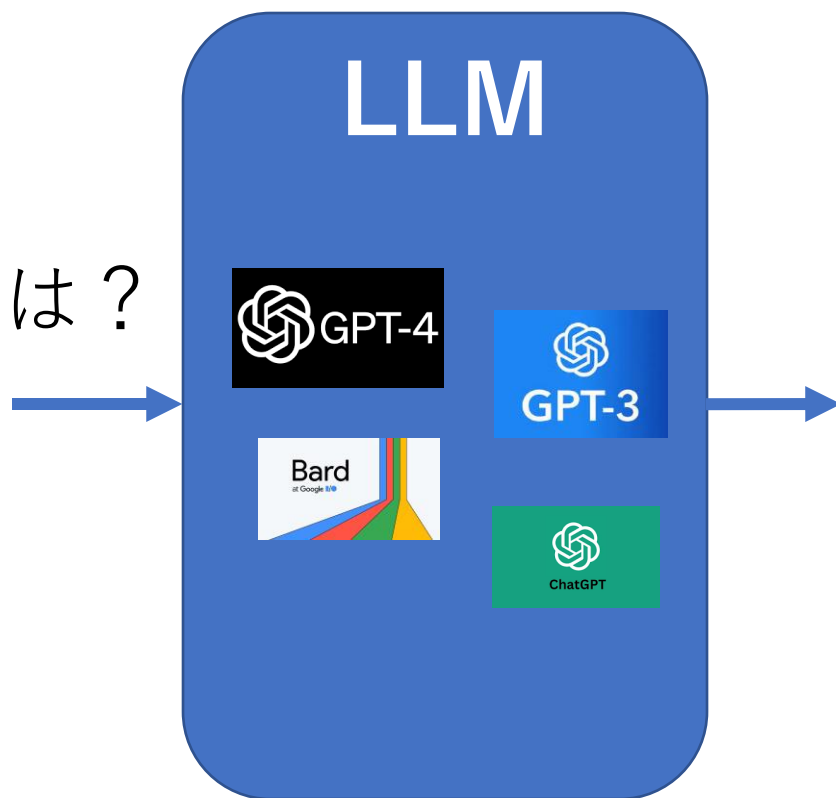
今朝、スタバで



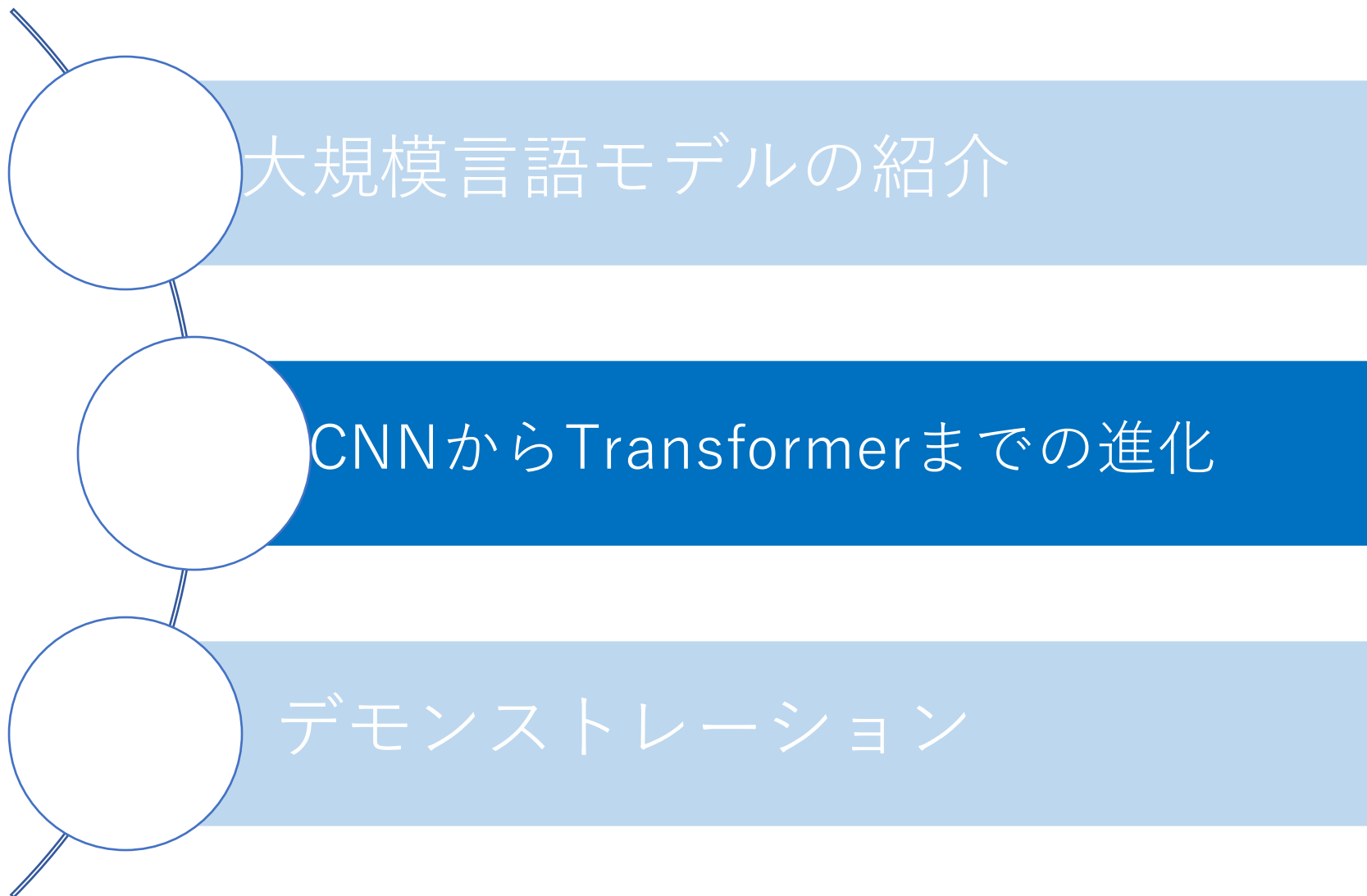
今朝、スタバで
コーヒーを飲みました

大規模言語モデル (LLM: Large language models)

岡崎出身の戦国武将は？



岡崎出身の戦国武将は？
徳川家康です。



なぜTransformerはそんなに賢くなるか？

以下の言葉を考えましょう

清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

長期依存性の問題

以下の言葉を考えましょう

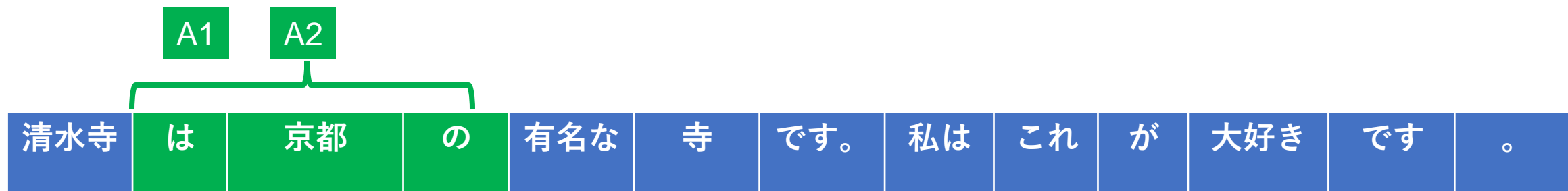
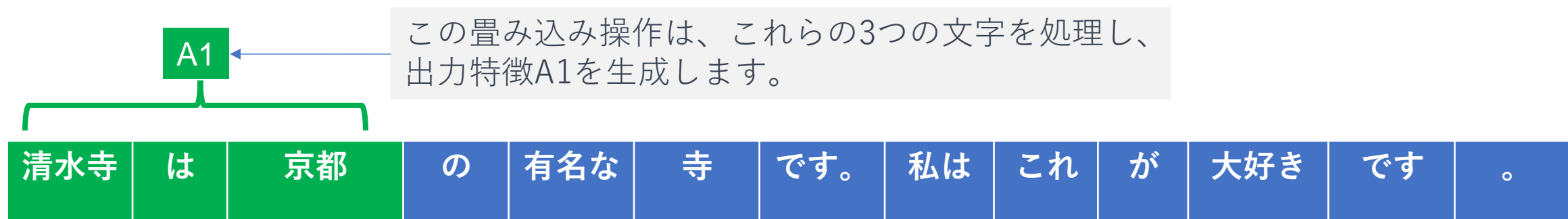
清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

- 「清水寺」と「これ」が指す対象は同じであり、この概念を理解することは、文全体を理解するための前提条件です。
- この問題は「長期依存性」の典型的な問題と見なすことができます。ここでの「長い」とは、「清水寺」と「これ」の2つの言葉の間に多くの他の言葉があるという事実を指しています。

解決法 1 : 畳み込みニューラルネットワーク

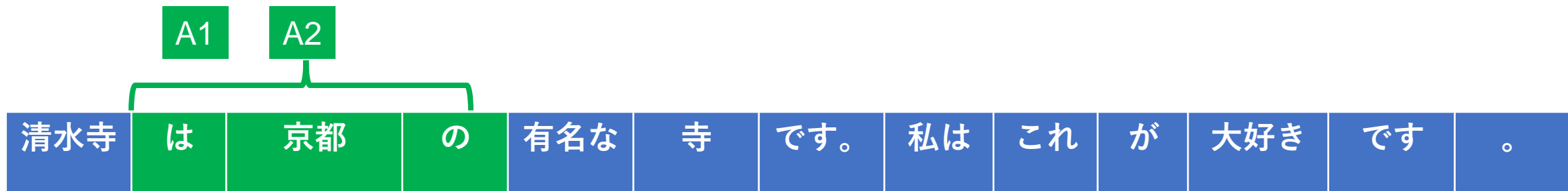
(CNN : convolutional neural network)

ウィンドウが左から右へと移動すると仮定すると、テキストの最も左の位置、つまり緑色のテキストエリアから始まります。



解決法 1 : 畳み込みニューラルネットワーク

(CNN : convolutional neural network)

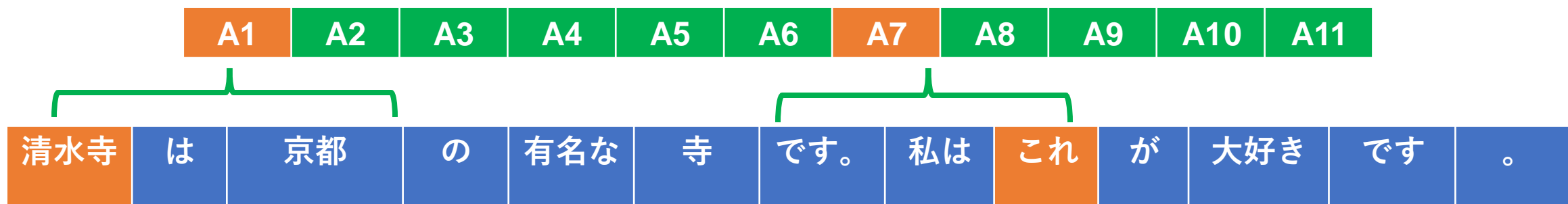


...



解決法 1 : 畳み込みニューラルネットワーク

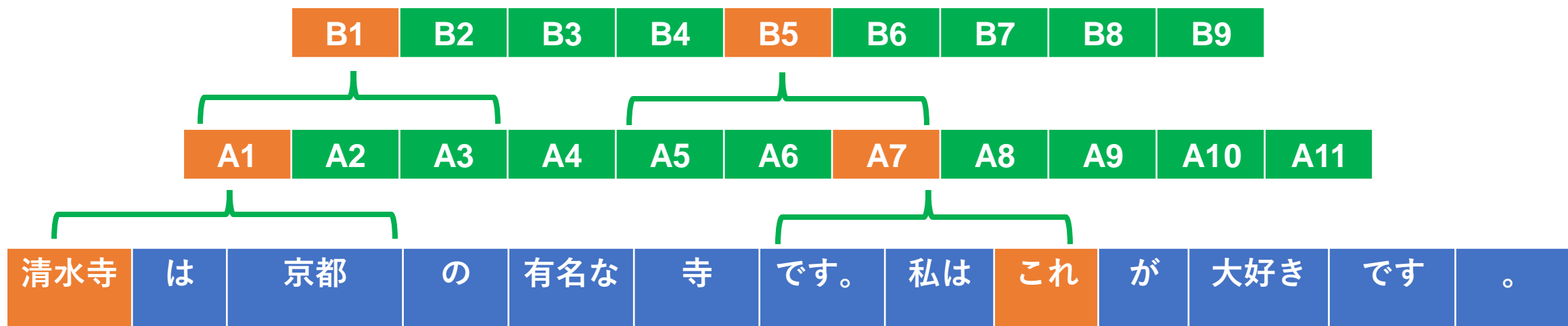
(CNN : convolutional neural network)



- 元のテキストでは、これら2つの単語は7単語離れています。
- 新しく計算された特徴量において、“清水寺”の2文字は特徴量A1の計算にのみ用いられています。
- 一方、「これ」は特徴量A7、A8、およびA9の計算に用いられており、A7がA1に最も近いものとなっています。

解決法 1 : 畳み込みニューラルネットワーク

(CNN : convolutional neural network)

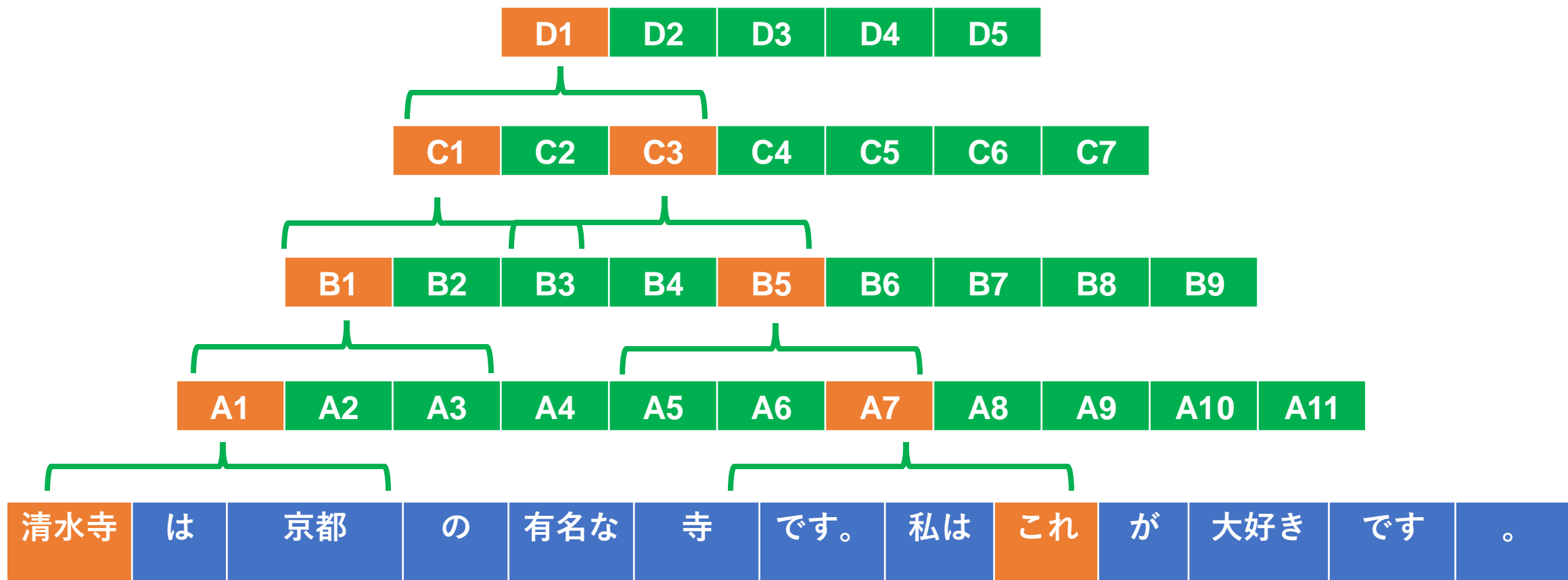


- 特徴量A1とA7の間の距離は、「清水寺」と「これ」の間の距離と比べて近くなりました。
- その後に、特徴量Aに別の畳み込みを積み重ねて特徴量Bを計算し、さらに畳み込みを繰り返すことができます。

解決法 1 : 畳み込みニューラルネットワーク

(CNN : convolutional neural network)

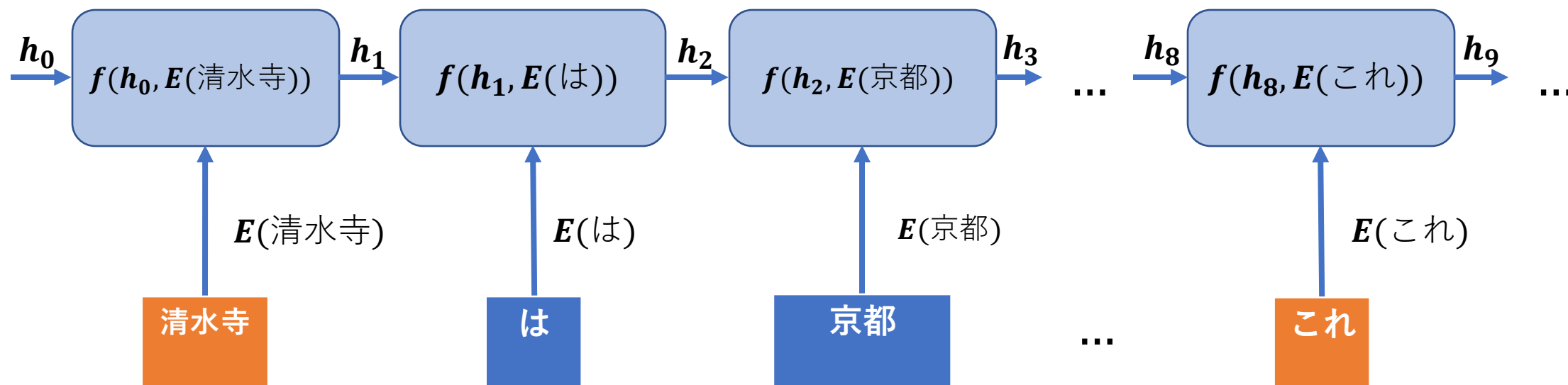
「清水寺」と「これ」に関する情報を直接取得できるように、畳み込み操作を再度計算します



解決法2：リカレントニューラルネットワーク

(RNN : Recurrent neural network)

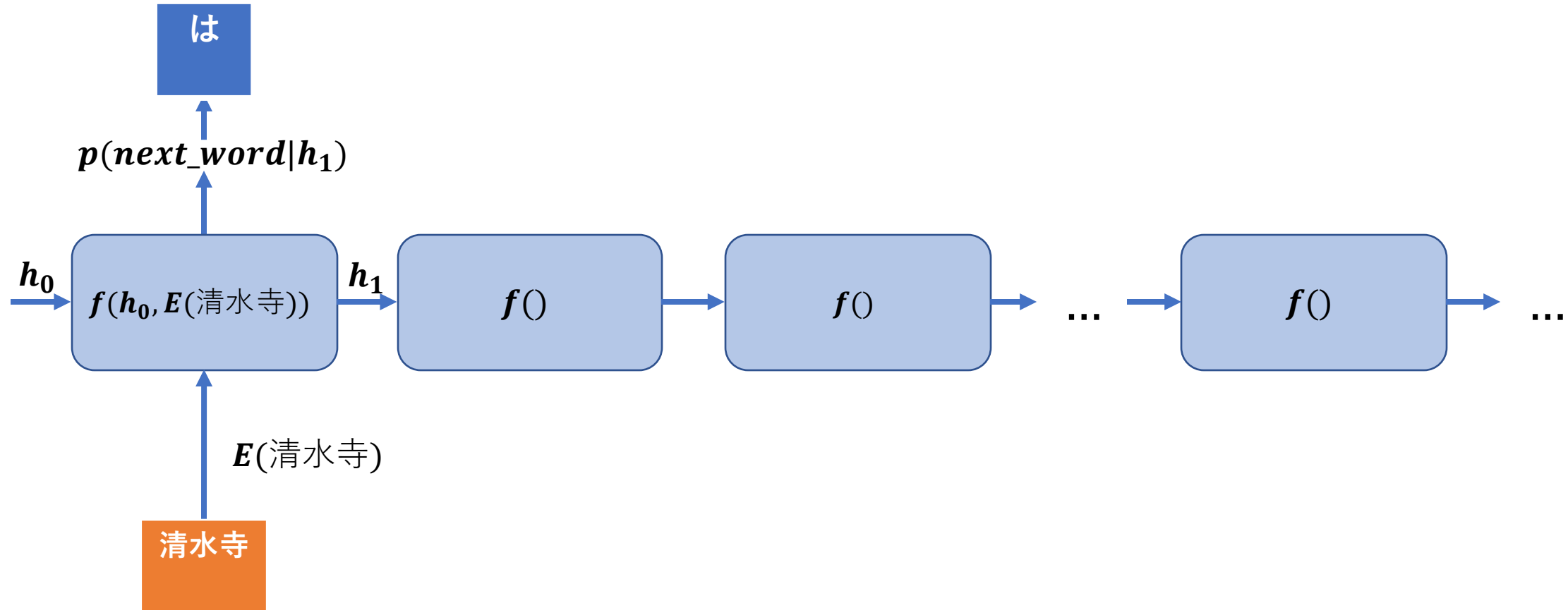
- RNNでは、ある単語の出力はその前の単語の出力に依存しており、任意の二つの単語の関係をRNNによって捉えることができます。
- どんなに単語が離れていても、それらの情報は最終的な出力に含まれています。



清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

解決法2：リカレントニューラルネットワーク

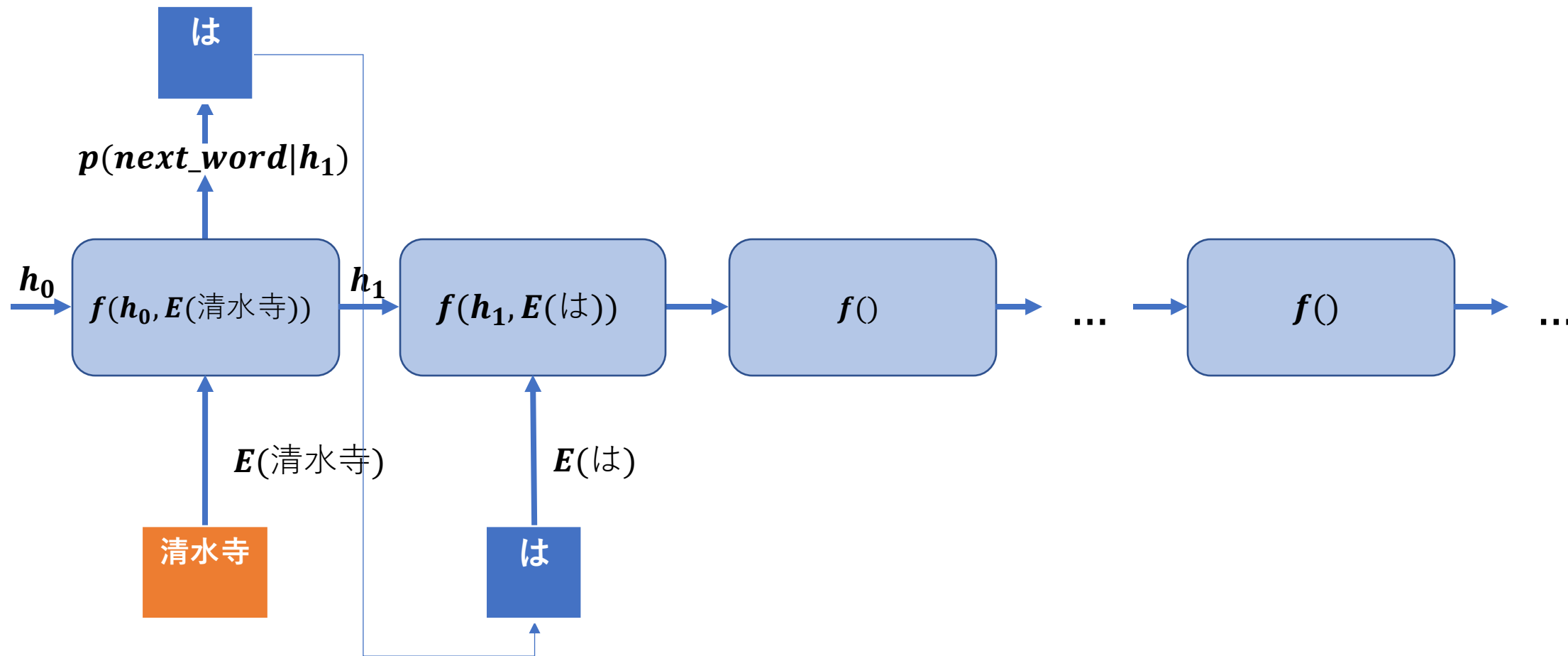
(RNN : Recurrent neural network)



清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

解決法2：リカレントニューラルネットワーク

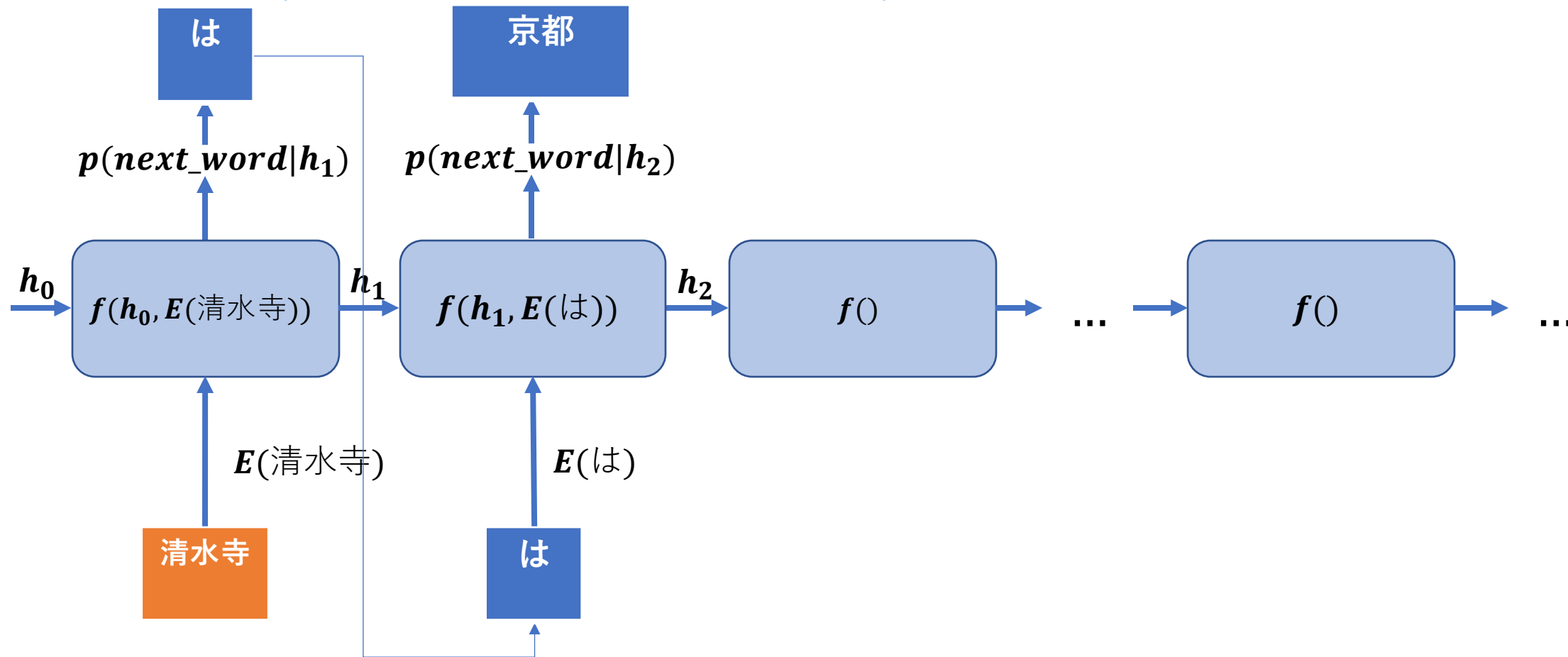
(RNN : Recurrent neural network)



清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

解決法2：リカレントニューラルネットワーク

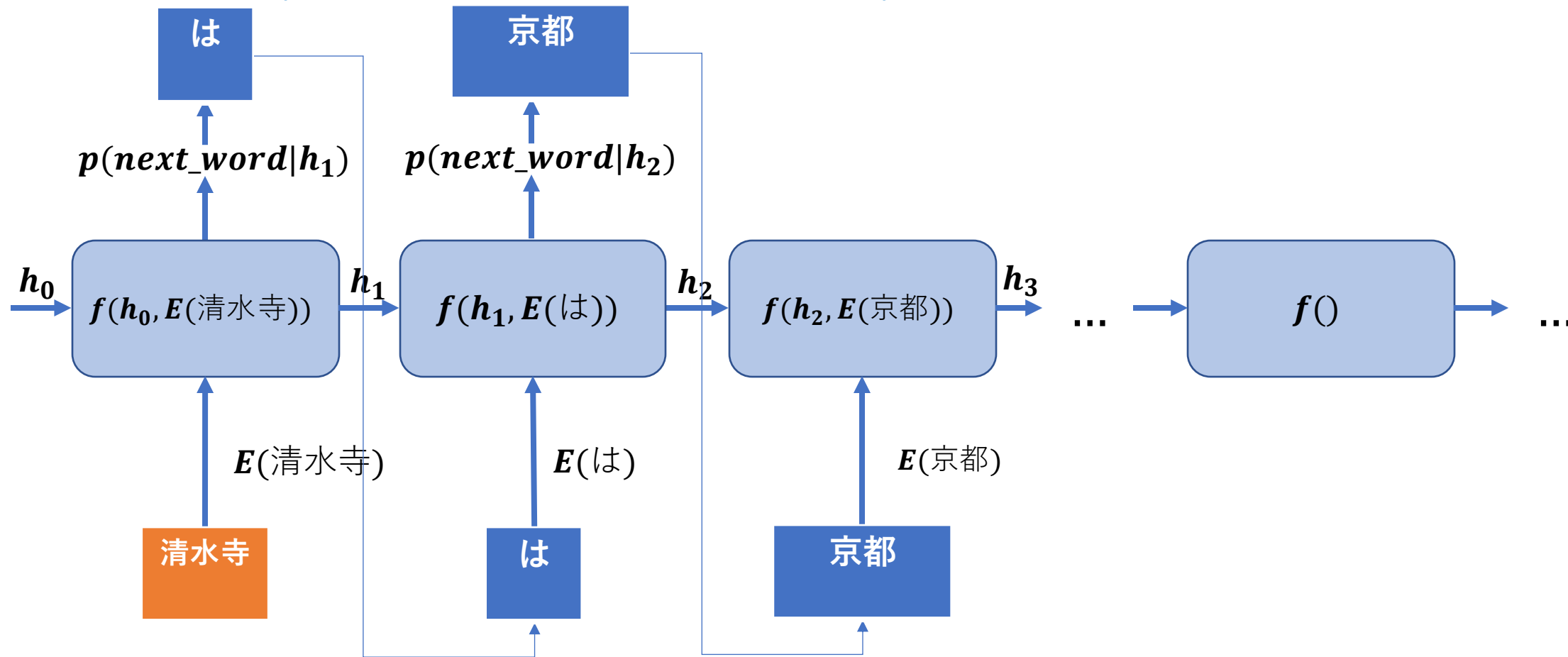
(RNN : Recurrent neural network)



清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

解決法2：リカレントニューラルネットワーク

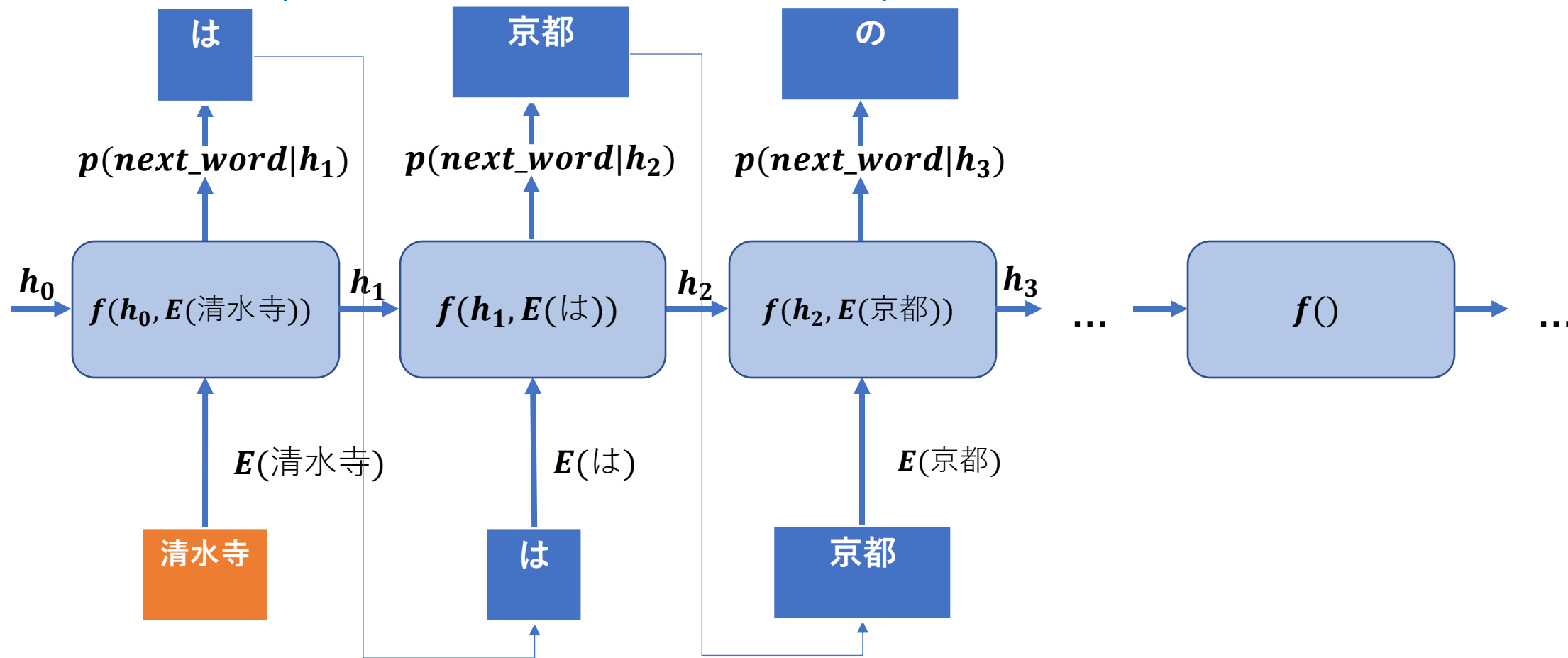
(RNN : Recurrent neural network)



清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	----	---	-----	----	---

解決法2：リカレントニューラルネットワーク

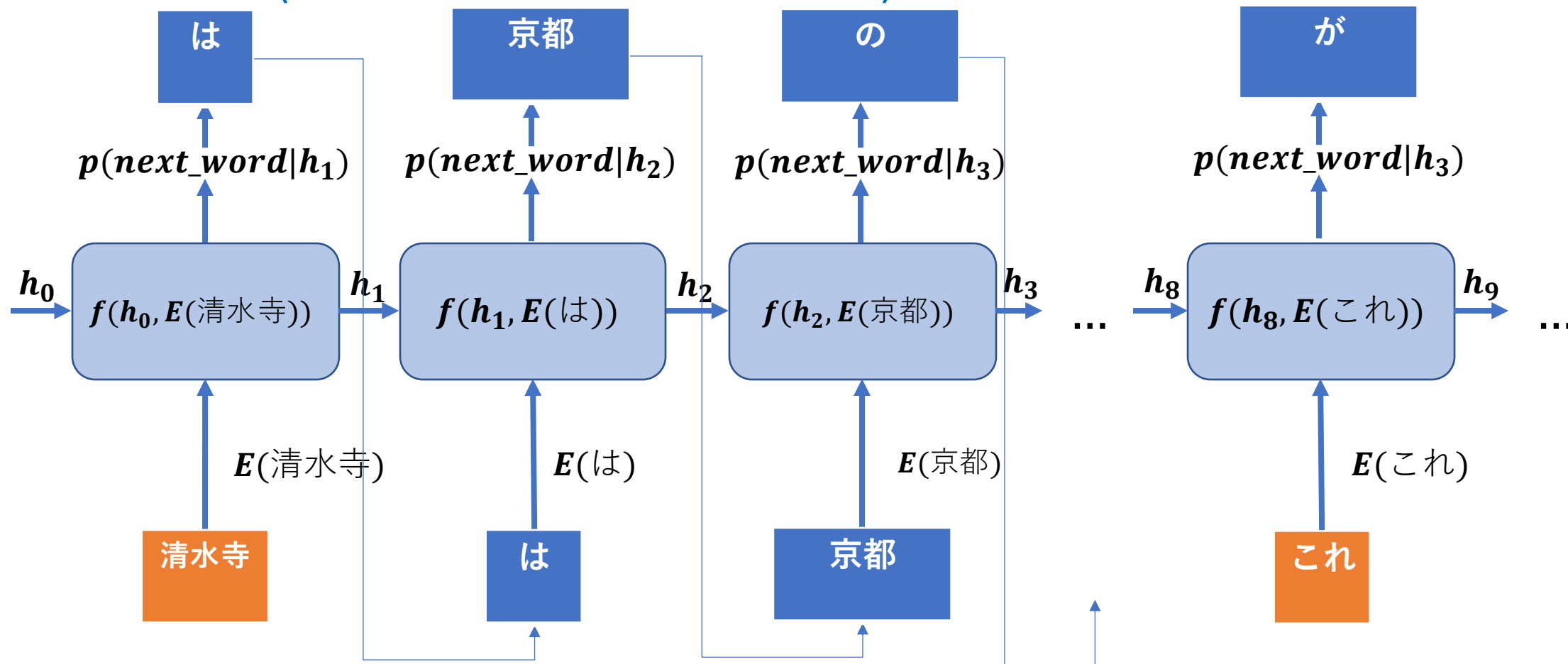
(RNN : Recurrent neural network)



清水寺 は 京都 の 有名な 寺 です。 私は これ が 大好き です 。

解決法2：リカレントニューラルネットワーク

(RNN : Recurrent neural network)



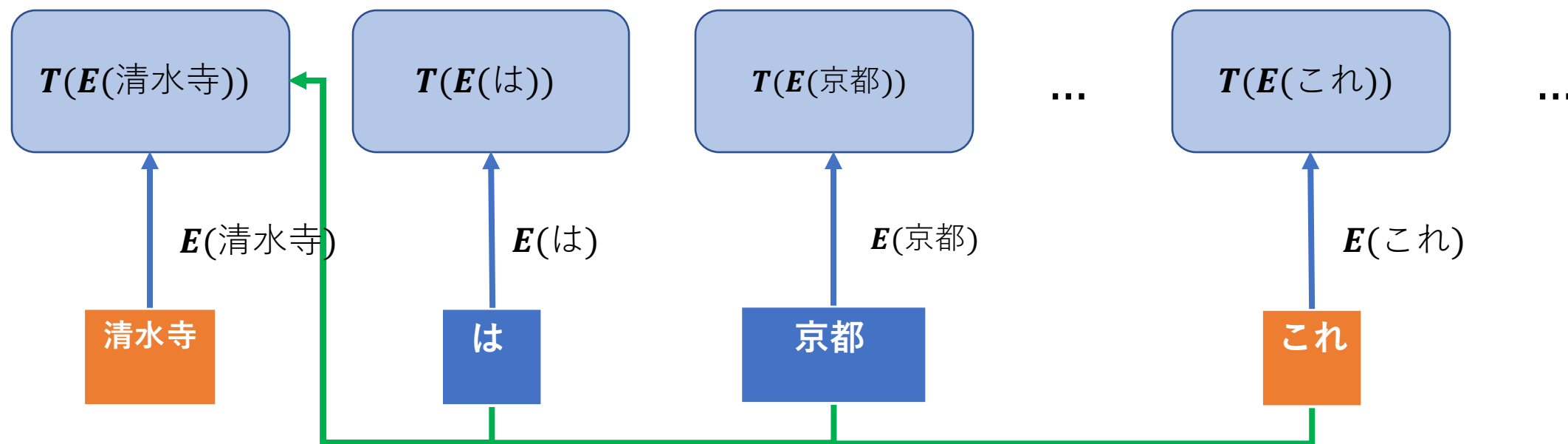
清水寺	は	京都	の	有名な	寺	です。	私は	これ	が	大好き	です	。
-----	---	----	---	-----	---	-----	----	-----------	---	-----	----	---

まとめ

- 実際には、RNNは訓練中に勾配が消失する、あるいは爆発してしまうという問題があり、RNNが長い依存関係を高精度に捉えるのは難しいです。
- いくつかの研究により、LSTMやGRUのような改良されたRNNであっても、長い依存関係を捉える能力は、通常20単語程度に限られていることが示されています。
- CNNでは一度に畳み込めるサイズが小さく、一度に長距離の依存関係を捉えることができません。これはCNNの構造に由来する問題です。
- RNNの長い依存関係の問題は、主に現在の最適化方法から生じています。RNNの構造自体の問題ではなく、現在の最適化方法ではこれを解決していません。

解決法 3 : Transformer

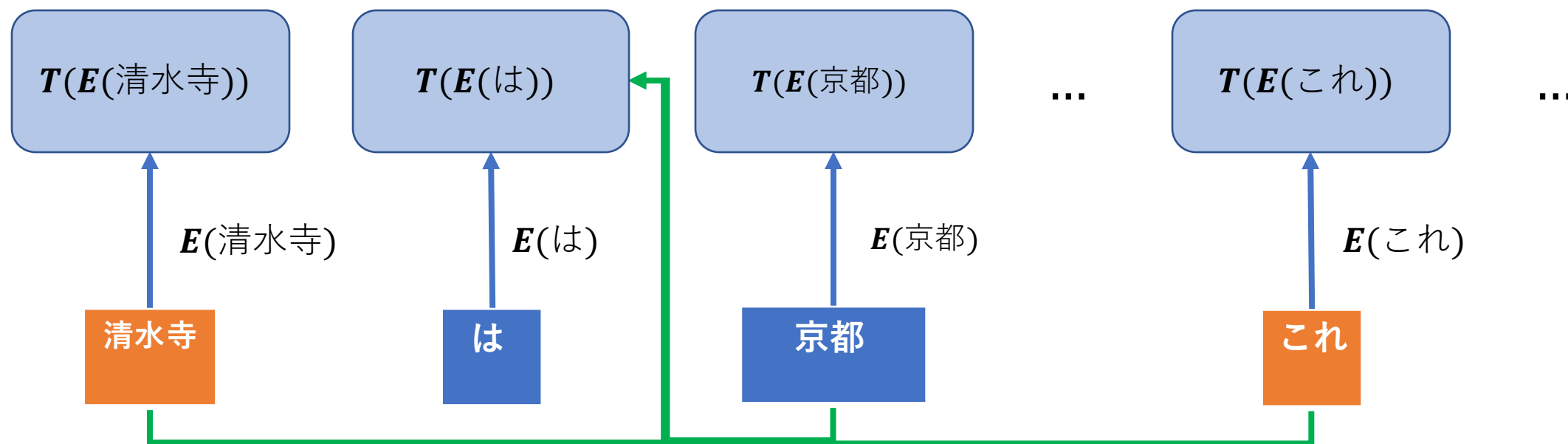
- Transformerは、構造または最適化に関する問題を解消しています。
- 任意の単語の新しい表現（特徴）を計算する際、Transformerは他のすべての単語からの情報を利用します。



単語「清水寺」の特徴 $T(\text{清水寺})$ は、すべての単語を用いた計算を通じて得られます。

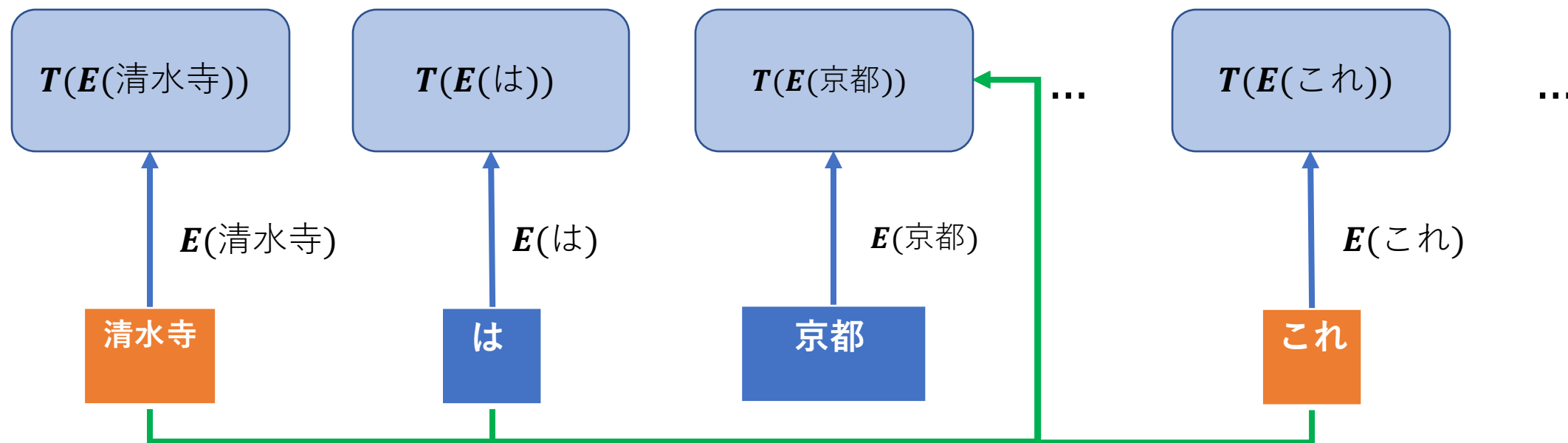
解決法 3 : Transformer

- これによりtransformerは長い依存関係の問題を解消することができます。すべての単語の注意度による重み付き和を取ることで、特定の単語の特徴を計算します。



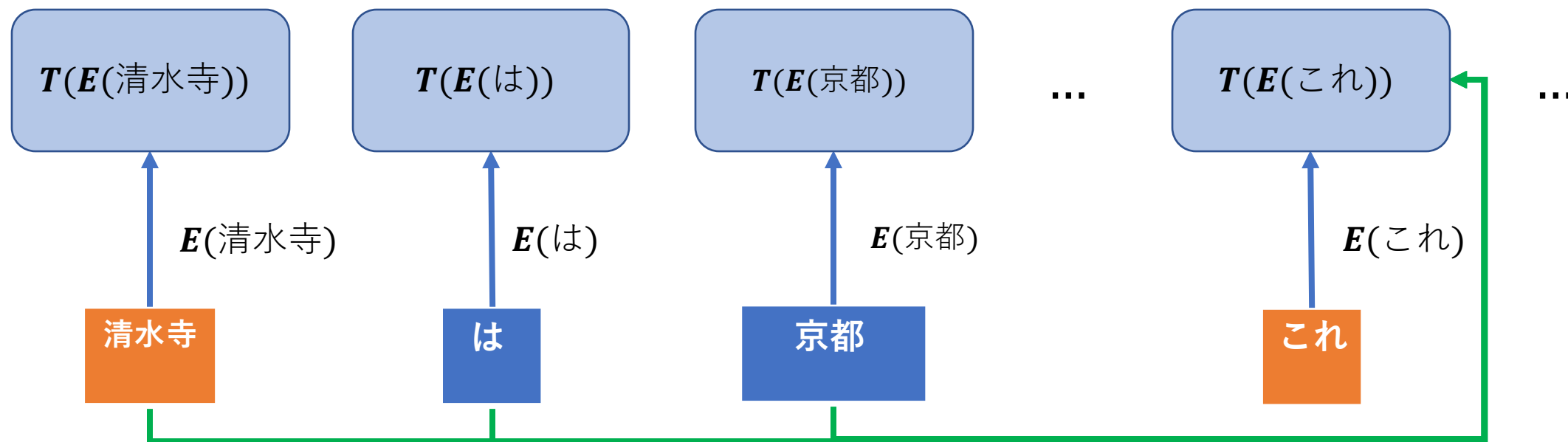
解決法 3 : Transformer

その結果、距離の概念がなく、長い依存関係の問題も生じません。



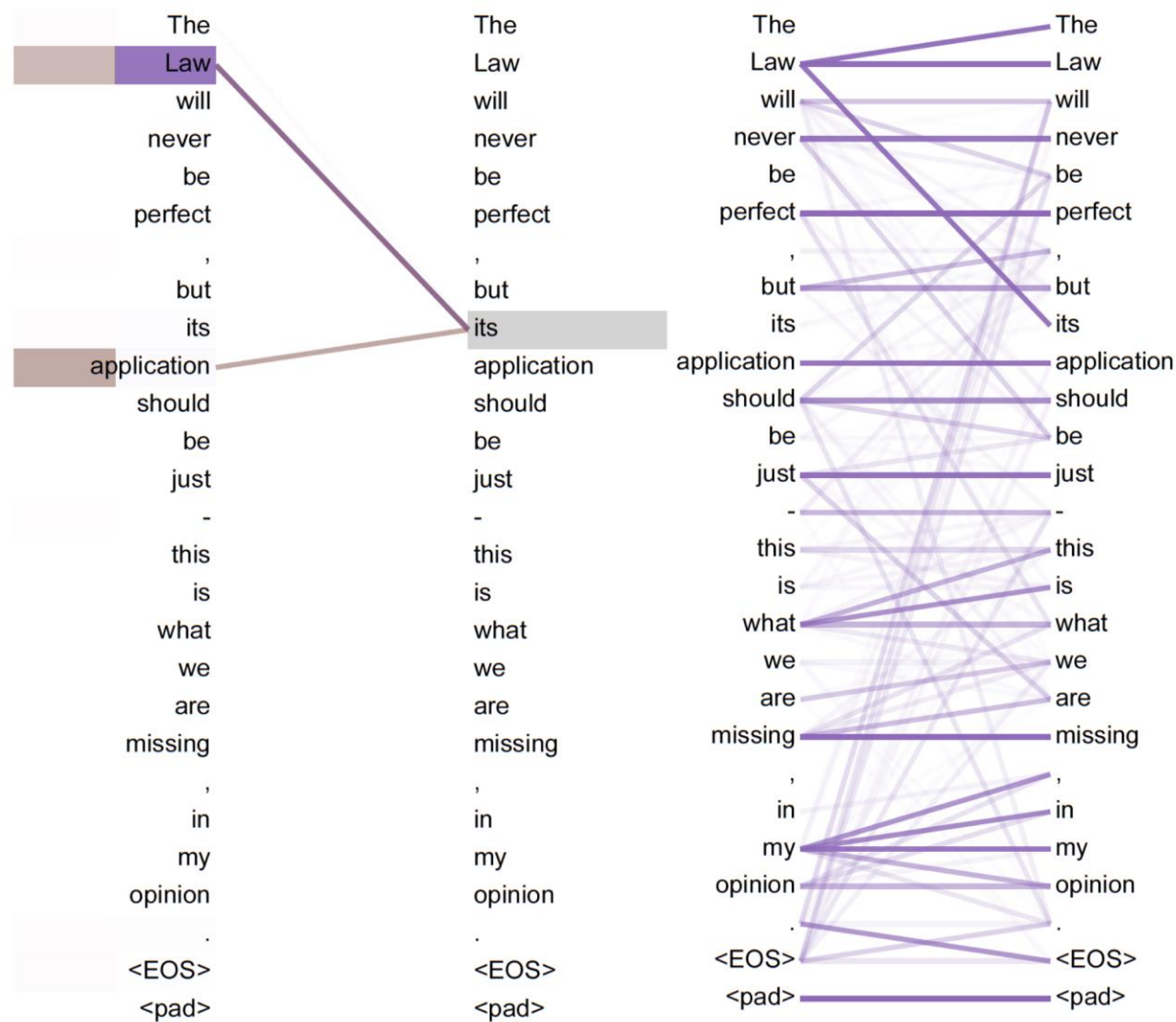
解決法 3 : Transformer

その結果、距離の概念がなく、長い依存関係の問題も生じません。



例：Transformerによる処理結果

Figure 4: Two attention heads, also in layer 5 of 6, apparently involved in anaphora resolution. Top: Full attentions for head 5. Bottom: Isolated attentions from just the word ‘its’ for attention heads 5 and 6. Note that the attentions are very sharp for this word.





2-1.CNN_for_long-term dependency

Colab:

https://colab.research.google.com/drive/1xk2BoBQFEZVwhBYt-69Rmq2WHIHO_K8S?usp=sharing

2-2.RNN_for_long-term dependency

Colab:

<https://colab.research.google.com/drive/1IMFGt5rB0OefrY1ZcpC7rBTT0BBrztOC?usp=sharing>

2-3.Transformer_for_long-term dependency

Colab:

<https://colab.research.google.com/drive/1f42TTIIJXkRnD85XFPgVXpQAdudLiF3r?usp=sharing>

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

TRANSFORMER EXPLAINER

Examples ▾ ta visualization empowers users to visualize

Generate

Temperature ⓘ 0.8

Sampling ⓘ ☒ Top-k ☐ Top-p k=5

