



Chatbot at 10:42



Section5の概要




講座の内容

Section 1. コースの概要とTwitter API

Section 2. RNNとSeq2Seq

Section 3. 自然言語処理の基礎

Section 4. モデルの訓練

 **Section 5. Attentionの導入**

Section 6. Twitterボットのデプロイ

今回の内容

1. Section5の概要
2. 過学習対策
3. 入力のパディング
4. データの前処理
5. Attentionの概要
6. Attentionの導入

教材の紹介

Pythonの基礎

Section5の教材:

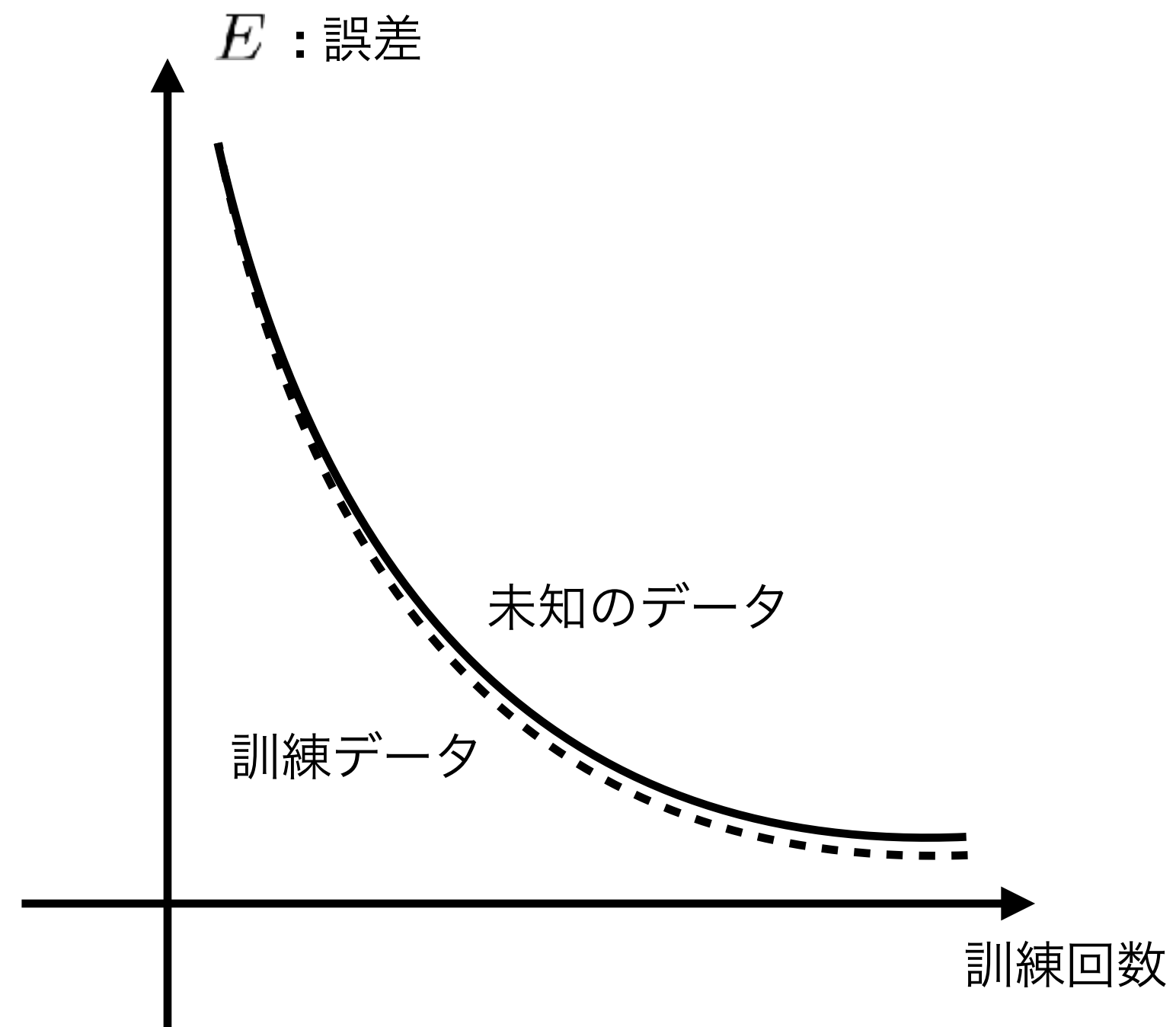
- 01_input_padding.ipynb
- 02_preprocessing.ipynb
- 03_attention.ipynb

Chatbot at 10:42

過学習対策

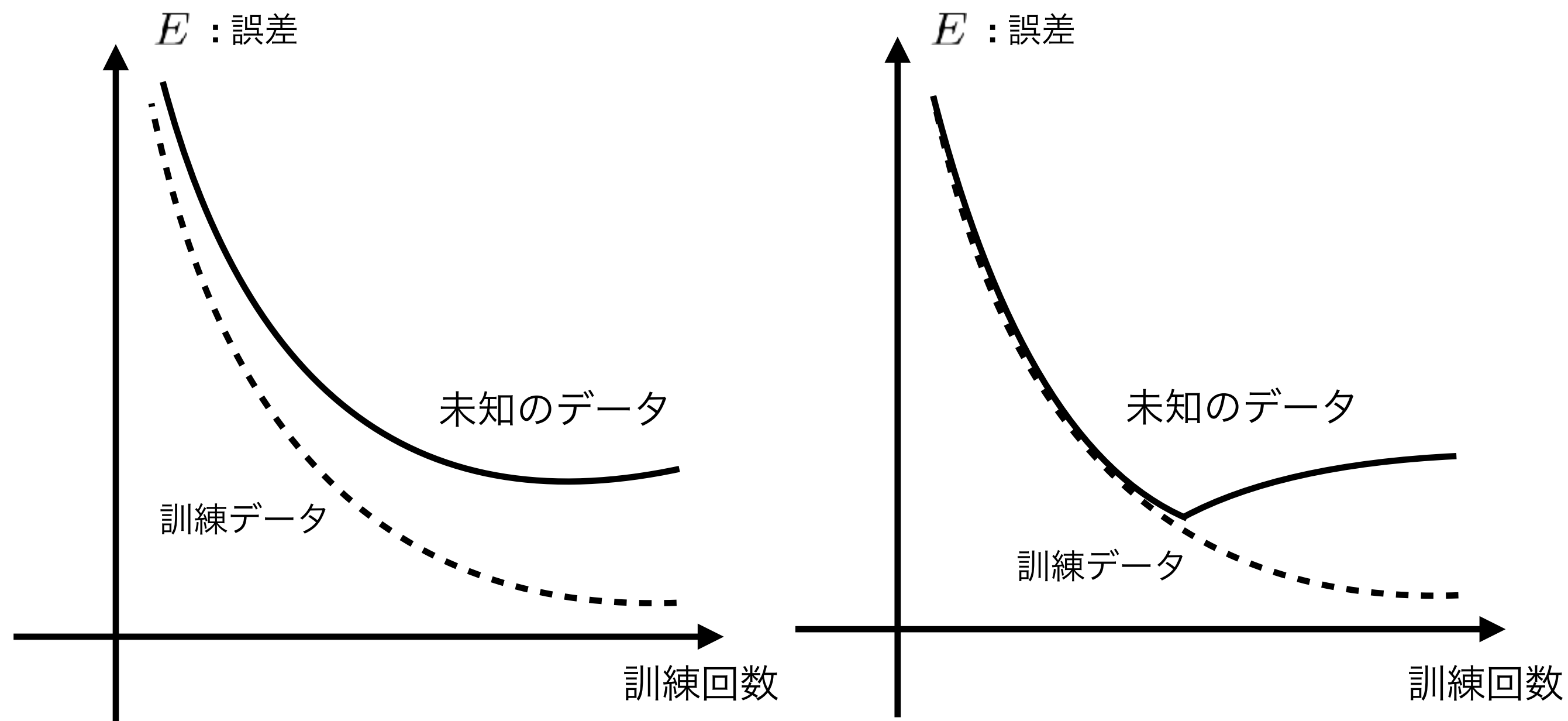


理想的な学習



未知のデータにも高い性能を発揮

過学習の例

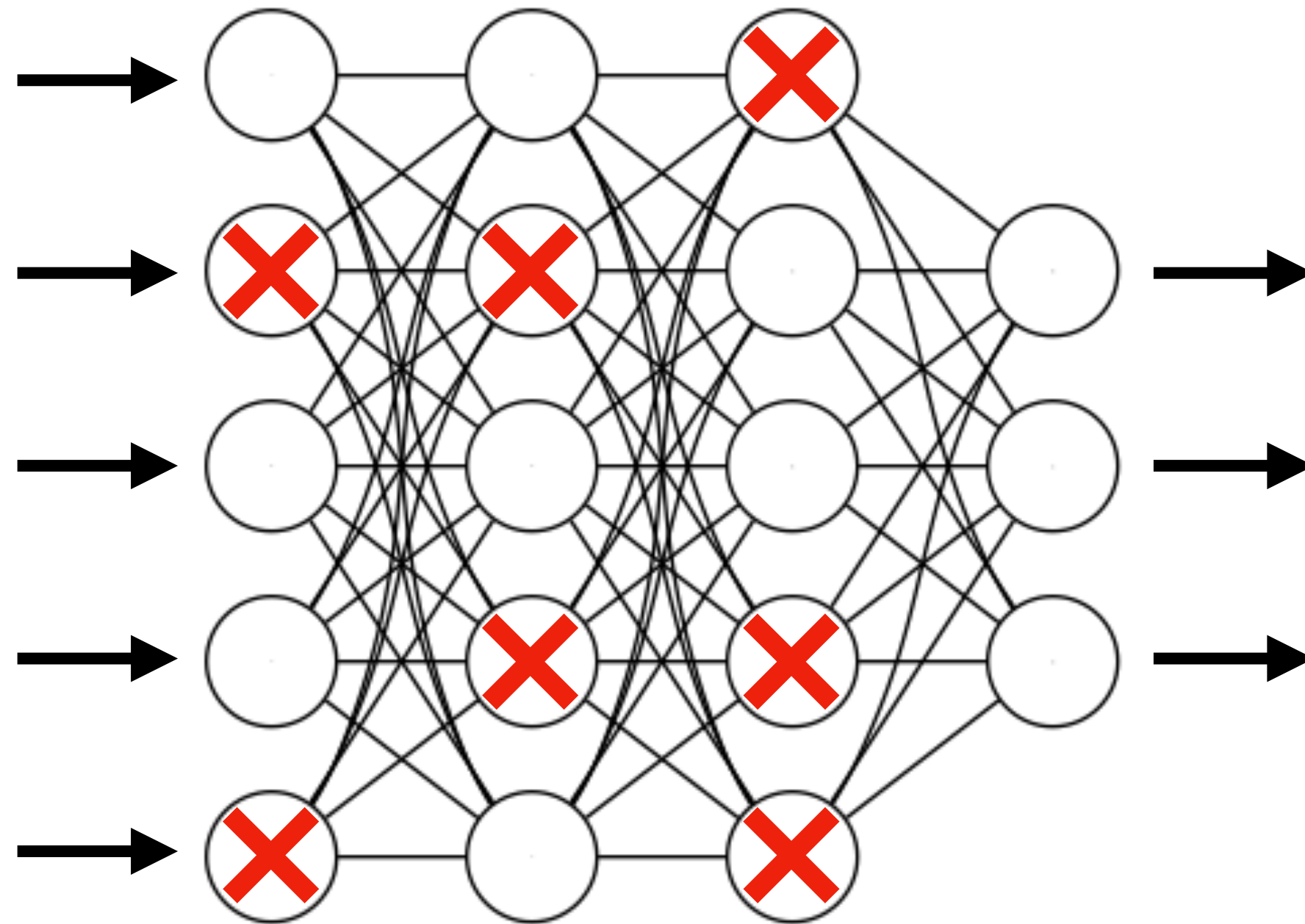


モデルが訓練データに過剰に適合する (過学習)

過学習への対策

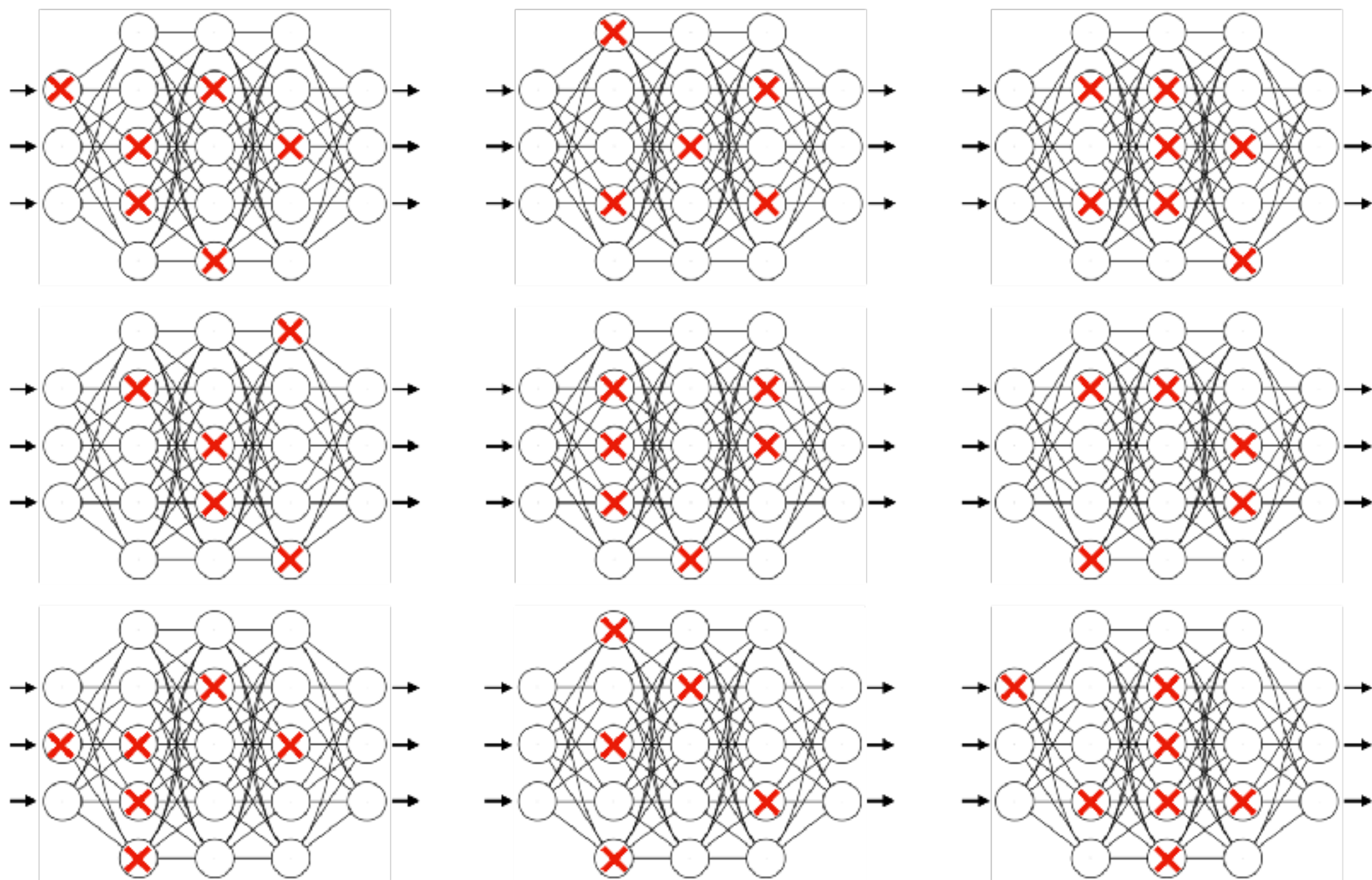
- **ドロップアウト**
 - ニューロンをランダムに無効にする
- **早期終了 (Early Stopping)**
 - 過学習の発生前に学習を終了する
- **データ拡張 (Data Augmentation)**
 - データを「水増し」する
- etc...

ドロップアウト



ニューロンをランダムに無効にする

ドロップアウト



実質的に、毎回異なるネットワークで学習を行う

入力のパディング



入力のパディング

- 01_input_padding.ipynb

データの前処理



データの前処理

- 02_preprocessing.ipynb

Attentionの概要

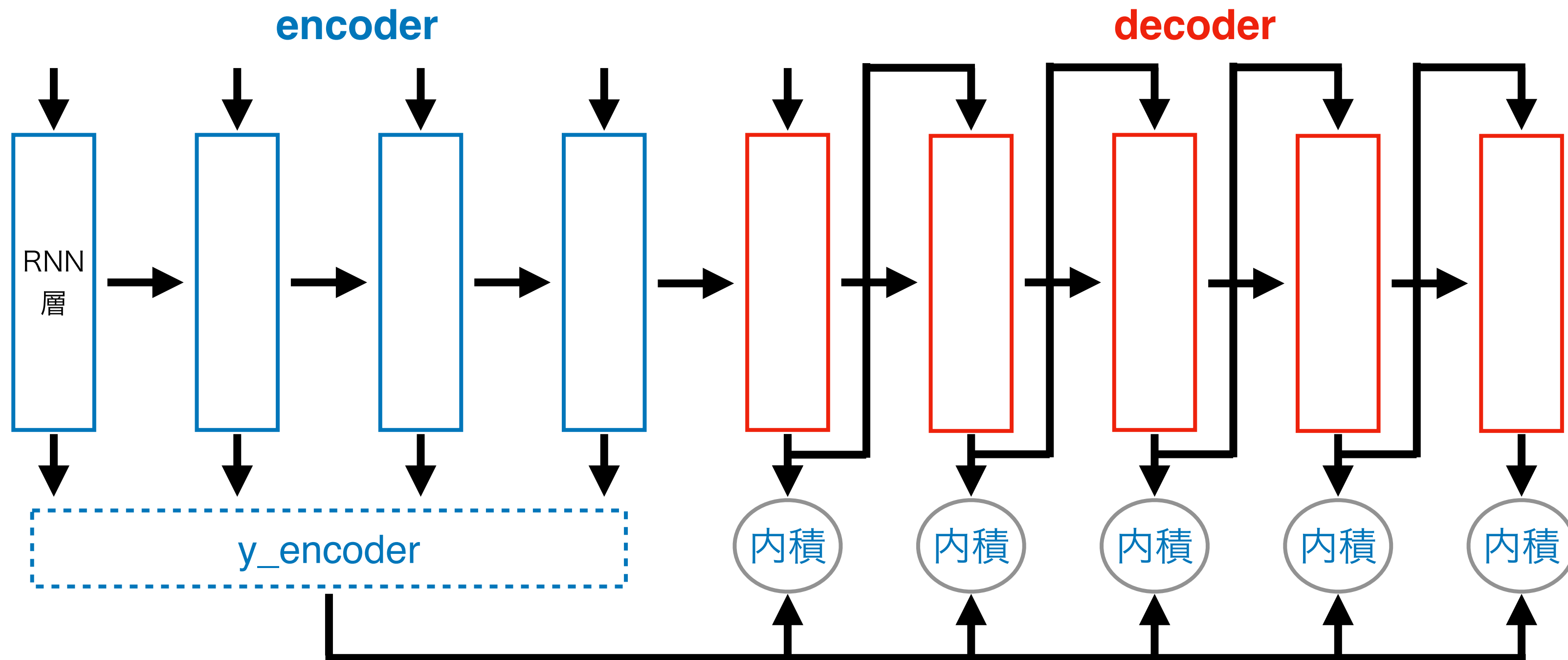


Attentionの概要

Attentionは、時系列データの特定の部分に
注意を向けるように学習させていく方法

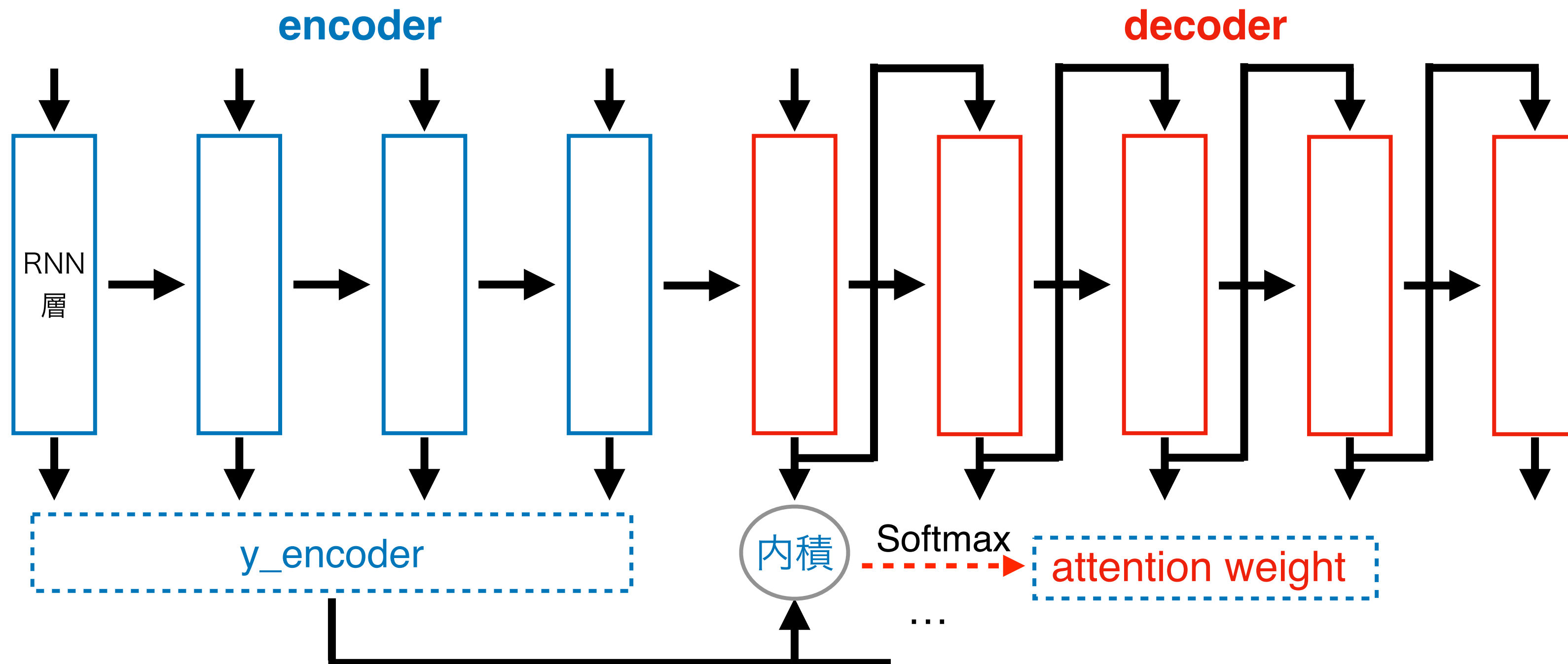
- I. Encoderの各時刻の出力をDecoderに渡す
- II. Decoder側の各時刻に、Encoderから渡された出力のうち、
最も注意すべきベクトルを選んで合流させる

Attentionの手続き1



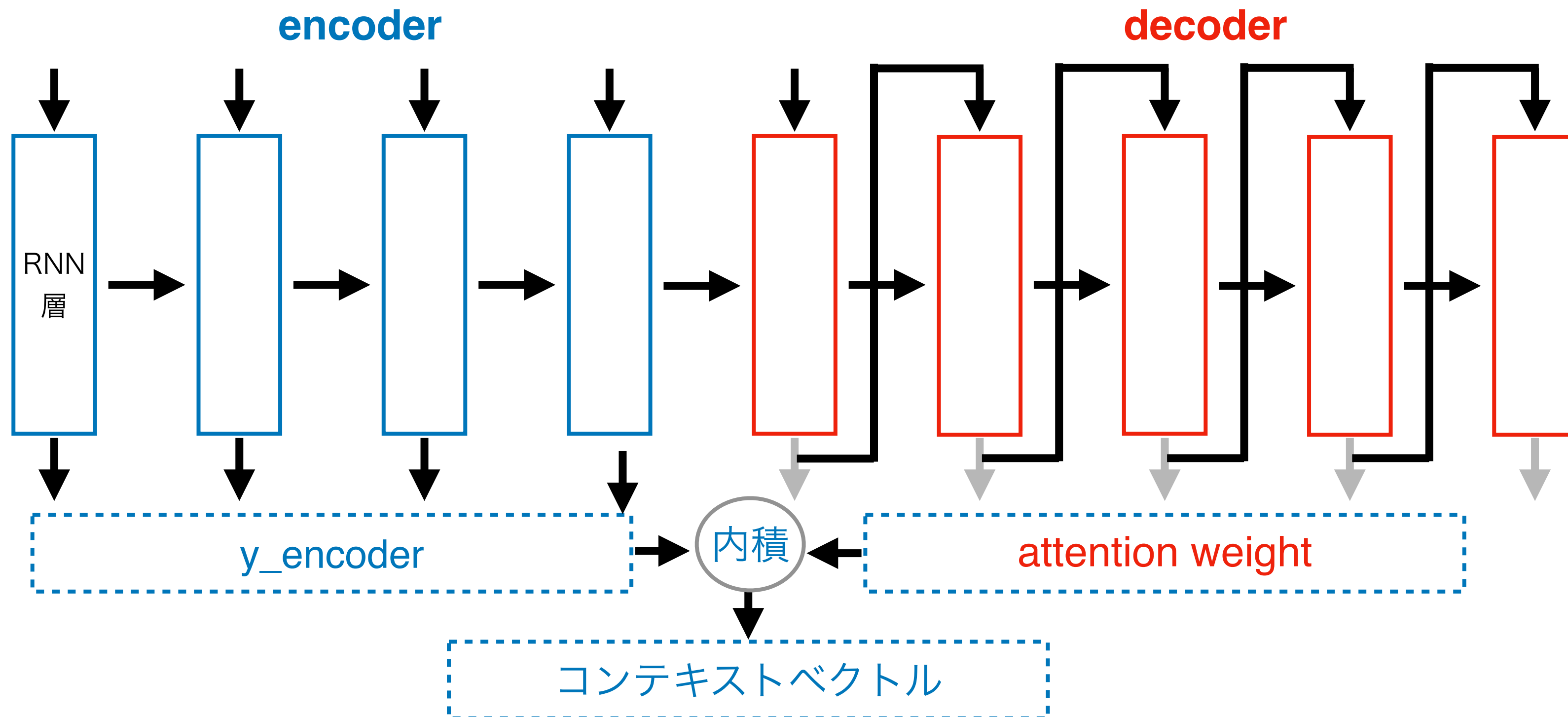
Encoderの全ての時刻の出力と、DecoderのRNNの各時刻における出力で内積をとる

Attentionの手続き2



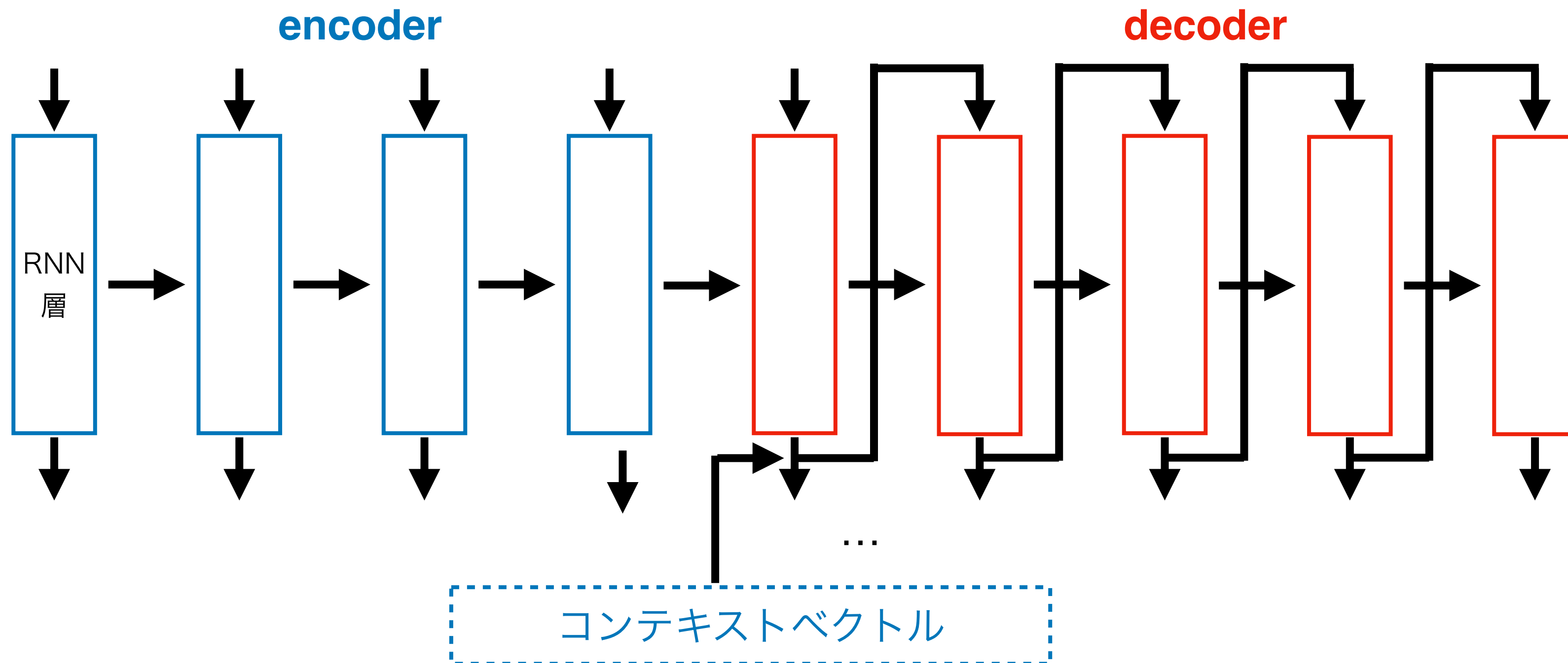
手続き1で計算した内積をSoftmax関数で確率表現にする (=attention weight)

Attentionの手続き3



Encoderの出力をattention weightで重み付けして足しあわせる (コンテキストベクトル)

Attentionの手続き4



DecoderのRNNの出力とコンテキストベクトルを合流させる

Attentionの導入



Attentionの導入

- 03_ attention.ipynb

次回の内容

Section 1. コースの概要とTwitter API

Section 2. RNNとSeq2Seq

Section 3. 自然言語処理の基礎

Section 4. モデルの訓練

Section 5. Attentionの導入



Section 6. Twitterボットのデプロイ