103. 深層学習の適用方法 (自然言語処理)

秋葉洋哉

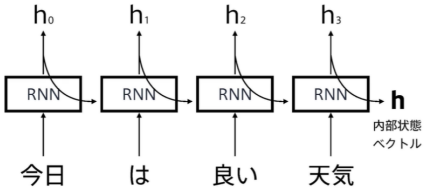
2024 年 7 月 15 日

1 Seq2Seq (Encoder-Decoder)

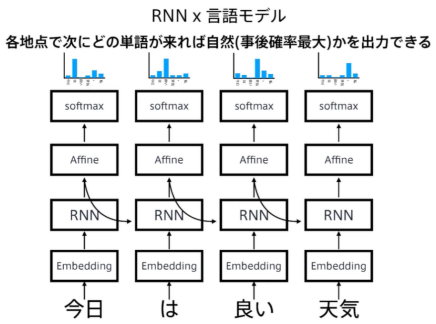
1.1 RNN と言語モデル

Seq2seq とは、Sequence to Sequence の略であり、系列 (Sequence) を入力として、系列を出力するものである。入力系列が Encode(=内部状態に変換) され、内部状態から Decode(=出力系列に変換) される。Seq2seq は、機械翻訳 (英語-日本語)、音声認識 (波形-テキスト)、質問応答システム (テキスト-テキスト) などに応用されている。

Seq2Seq は、RNN と言語モデルを組み合わせたモデルである。RNN は、再帰型ニューラルネットワーク (Recurrent Neural Network) の略であり、図 1a のような構造を持つ。最終的に内部状態ベクトル **h** を得ることが目的となる。



(a) RNN の構造: 最終的に内部状態ベクトル **h** を得る。



(b) RNN と言語モデルを組み合わせた構造

図 1: Seq2Seq の基本的な仕組み

言語モデルは、単語の並びに確率を与えるモデルであり、単語の並びに対してそれがどれだけ起こりうるか (尤度)、すなわち文章として自然かを確率で評価する。例えば、I am a student. という文章と、I student am a. という文章を比較すると、前者の方が自然であるため、言語モデルで出力される確率が高くなる。このような自然さを確率で評価するモデルを言語モデルと呼ぶ。RNN と言語モデルを組み合わせた時、図 1b のような構造を持つことになる。このモデルを Encoder、Decoder として、内部状態ベクトル h を媒介して組み合わせたものが Seq2Seq である。

1.2 Teacher Focing

Seq2Seq では、ひいては RNN では、予測を用いて次の予測を行うため、一つ間違えた時の影響が次々と伝播し、誤差が蓄積されてしまう。そこで、Decoder の入力に正解データを入力することで、学習を安定させることができる。この手法を Teacher Forcing と呼ぶ。本番時には、この手法は使えず、予測結果が次の入力となるため、誤差が蓄積されることに注意が必要となる。

2 Transformer

2.1 概要

Transformer は、Vaswani et al (2017) で提案されたモデルであり、Seq2Seq の問題点である長い系列を扱うことができる。Transformer のアーキテクチャは図 2 のように構成されており、RNN を用いず、代わりに Attention 機構というモジュールが用いられる。

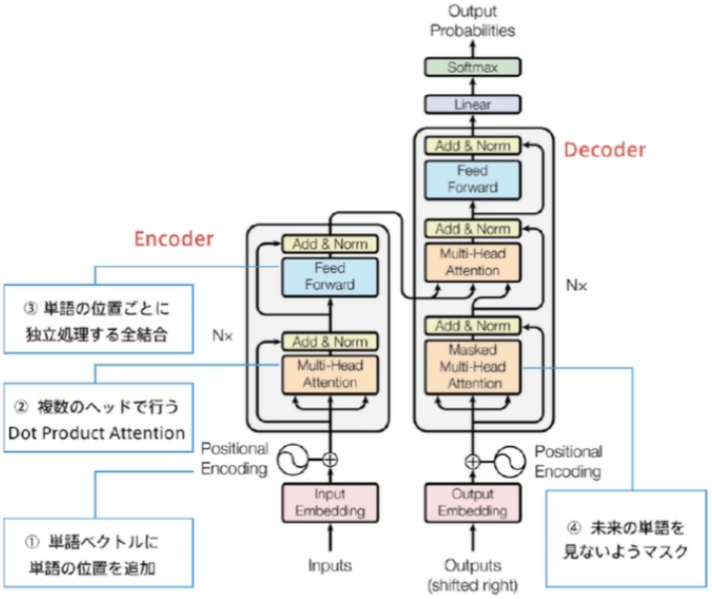


図 2: Transformer のアーキテクチャ

2.2 Attention

Transformer は、Attention という機構を導入することで、長い系列を扱うことができる。Attention は、入力系列の各要素に重みを付けることで、重要な要素に注目することができる。この機構を導入することで、長い系列を扱うことができるようになる。

Attention は辞書オブジェクトである。辞書オブジェクトとは、キーと値のペアを持つオブジェクトであり、キーを用いて値を取得することができる。Attention は、Query、Key、Value の 3 つの要素を持つ。Query は、重要な要素を取得するためのキーであり、Key は、入力系列の各要素のキーであり、Value は、入力系列の各要素の値である。Query と Key の内積を取ることで、各要素の重要度を計算することができる。この重要度を用いて、Value の重み付き和を計算することで、Attention の出力を得ることができる。

Attention は、Source-Target-Attention と Self-Attention の 2 つの種類がある。Source-Target-Attention は、入力系列と出力系列の関係を表す。Self-Attention は、入力系列の各要素に重みを付けることができる。

2.3 Self-Attention の計算

Selt-Attention の計算は、以下のように行われる (図 3)。

1. Query(正解のラベル) と Key(入力のラベル) の内積を計算する。すると、各要素の重要度が計算される。
2. 効率的な逆伝播のために、内積をスケーリングする。$\frac{QK^T}{\sqrt{d_k}}V$
3. ソフトマックス関数を適用する。
4. 3. の出力と Value の内積を計算する。

これらの手順は、図 2 の②に対応している。図 2 の④は、Decoder において、

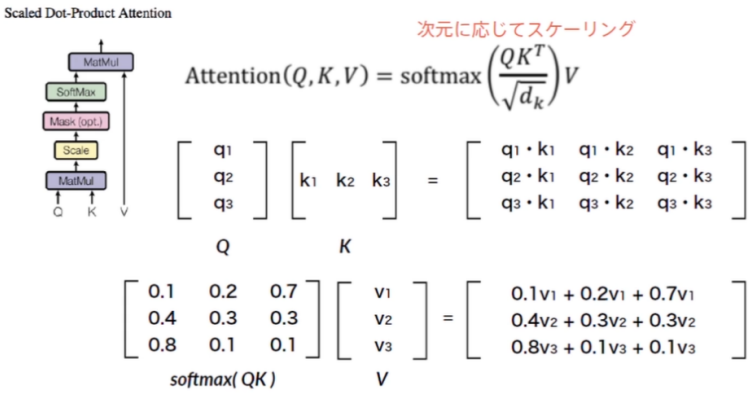


図 3: Self-Attention の計算

2.4 Add & Norm

Transformer では、Self-Attention の出力と入力を足し合わせることで、情報を保持する。その後、Layer Normalization を行うことで、情報を正規化する。この手法は、ResNet でも用いられている手法であり、勾配消失問題を回避するために用いられる。

2.5 Position Encoding

Transformer は、Attention 機構を用いるため、系列の順序情報を考慮することができない。そのため、Position Encoding という手法を用いて、系列の順序情報を考慮する。Position Encoding は、系列の各要素に対して、位置情報を付与する手法である。具体的には、各要素に対して、sin, cos 関数を用いて、単語の位置情報を付与 (エンコード) する。この手法は、図 2 の①に対応している。

2.6 Masking

Transformer の Decoder では、未来の情報を見ないようにするために、Masking という手法を用いる。この手法は、未来の情報を見ないようにするために、Attention の重みを 0 にする手法である。この手法は、図

2 の④に対応している。

2.7 Multi-Head Attention

Multi-Head Attention は、Self-Attention をアンサンブルで用いる手法である。Self-Attention を複数のヘッドに分割することで、複数の視点から情報を取得することができる。

3 BLEU

BLEU は、機械翻訳の評価指標の一つであり、Modified N-gram precision を用いて、翻訳の正確さを評価する指標である。BLEU は、N-gram を用いて、翻訳の正確さを評価する。

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{1}$$

ただし、BP(Brevity Penalty) は、短い文に ($c < r$) 対してペナルティを与えるための指標であり、w_n は、N-gram の重み、p_n は、Modified N-gram precision である。BP は以下で定義できる。

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{if } c \leq r \end{cases} \tag{2}$$

ただし、c は、翻訳文の長さ、r は、参照文の長さである。

p_n は、例を用いて説明する。以下の Reference と Candidate があるとする。

$$\text{Reference} = \text{the cat is on the mat} \tag{3}$$
$$\text{Candidate:1} = \text{the cate the on the mat} \tag{4}$$
$$\text{Candidate:2} = \text{the the the the the the the} \tag{5}$$
$$\text{Candidate:3} = \text{the cat are On The Mat} \tag{6}$$
$$\tag{7}$$

この時の Modified N-gram precision の値は、

$$\text{Reference} = \text{the cat is on the mat} : 5 \tag{8}$$
$$\text{Candidate:1} = \text{the cate the on the mat} : 2 \tag{9}$$
$$\text{Candidate:2} = \text{the the the the the the the} : 0 \tag{10}$$
$$\text{Candidate:3} = \text{the cat are On The Mat} : 0 \tag{11}$$
$$\tag{12}$$

から、$(2 + 0 + 0)/5 = 2/5$ と計算される。

BLEU で対数加重平均で評価するメリットとしては、n が大きくなると、指数的にスコアが低くなってしまうのを平坦化することで、n が小さいときのスコアは、妥当性を評価し、n が大きいときのスコアは、流暢性を評価することができる。

4 BERT

BERT は、Bidirectional Encoder Representations from Transformers の略であり、2018 年に Google によって提案されたモデルである。BERT は、Transformer をベースに開発されており、双方向の情報を考慮することができる。BERT は、Fine-Tuning アプローチの事前学習に工夫を加えることで、様々な自然言語処理のタスクに対応することができる。

BERT は、以下のような特徴を持つ。

- 双方向の情報を考慮することができる。
- 事前学習を行うことで、様々な自然言語処理のタスクに対応することができる。
- Attention 機構を用いることで、長い系列を扱うことができる。

5 GPT

5.1 概要

GPT は、Generative Pre-trained Transformer の略であり、OpenAI によって提案されたモデルである。GPT は、巨大な文章のデータセット (約 45TB) を用いて、事前学習を行い、パラメータ数は、1750 億個にも達する (GPT-3 の場合)。このおかげで、汎用的な特徴量を習得しており、手元の様々な新しいタスク (翻訳や質問応答など) に対して特化したデータセットの規模が小さくても、高精度で予測モデルを構築することができる。

GPT の構造は、Transformer をベースとしているが、BERT と異なり、双方向の情報を考慮することができない。だが、BERT では各タスクに対してファインチューニングが必要である一方で、GPT3 は、ファインチューニングを必要としないにも関わらず、幅広い言語タスクを高精度で実行することができる (図 4)。

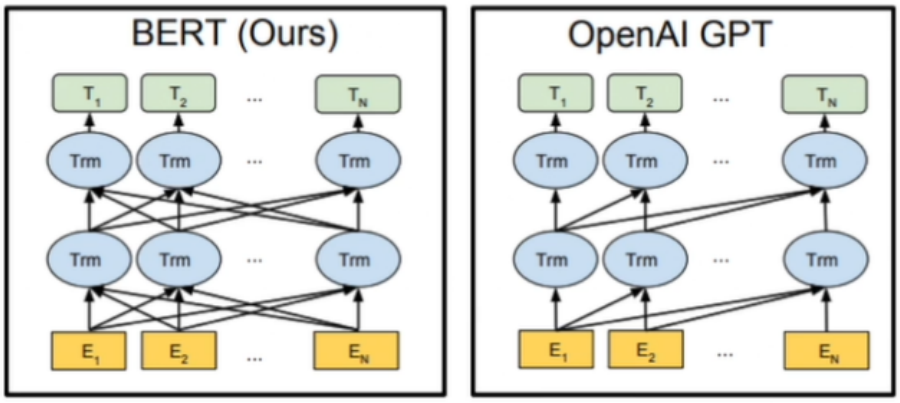


図 4: BERT と GPT の違い: BERT は文中のどの単語もマスクされる可能性があり、マスクした前の単語も後ろの単語にも注目する必要がある。一方で、GPT は常に次の単語を予測するため、双方向ではない。

5.2 GPT-3 の問題点

まず、膨大な数のパラメータを使用するため、学習に多大なコストがかかる。そして、モデルができたとしても、人間社会の慣習や常識を認識できないことで、不適切な回答をすることがある。特に、物理現象に関する推論を苦手としている。また、社会への課題も存在する。GPT-3 は、人間らしい文章を生成する能力を有するため、フェイクニュースなどの悪用のリスクがある。

5.3 GPT の事前学習

GPT の事前学習では、以下の式を最大化する。

$$L_1(u) = \sum_i \log P(u_i | u_{i-k}, \cdots, u_{i-1}; \Theta) \tag{13}$$

ただし、u は、入力系列、u_i は、入力系列の i 番目の要素、Θ は、モデルのパラメータである。k はコンテキストウィンドウのサイズであり、u_i を予測するために u_i の前にある k 個の要素を考慮することを意味する。

学習は、以下の数式で表される。

$$h_0 = UW_e + W_p \tag{14}$$

$$h_l = \text{TransformerBlock}(h_{l-1}) \quad \text{for } l = 1, \cdots, n \tag{15}$$

$$P(u) = \text{softmax}(h_n W_e^T) \tag{16}$$

ただし、U は、対処の単語を予測するために使う $u_{i-k}, \cdots, u_{i-1}$ で、W_e は、単語の埋め込み行列、W_p は、位置エンコーディング行列、h_l は、l 番目の Transformer ブロックの出力、n は、Transformer ブロックの数である。

GPT は Transformer の Decoder のみを用いている (図 5)。Transformer の Decoder と比較すると、Multi-Head Attention 部分が減っただけの構造になっていることがわかる (図 2)。

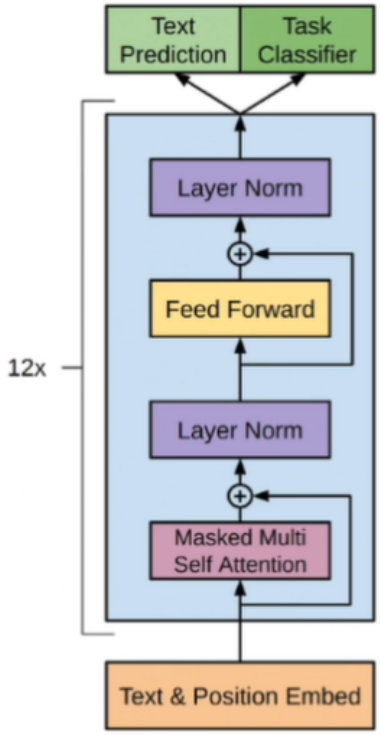


図 5: GPT の構造

5.4 GPT-1 のファインチューニング

GPT-2, GPT-3 で細かい変更があるものの、基本的な構造は GPT-1 と同じである。GPT-1 のファインチューニングは、始まりを表す記号、文と文を区切る記号、文の終わりを表す記号を用いる。図 6 は、上から順にテキスト分類タスク、文同士の関係予測タスク、文と文の類似度予測タスク、複数の文から一つを選ぶタスクを表している。

5.5 GPT-3 の推論方法

GPT-3 の推論は、以下のように分類できる。

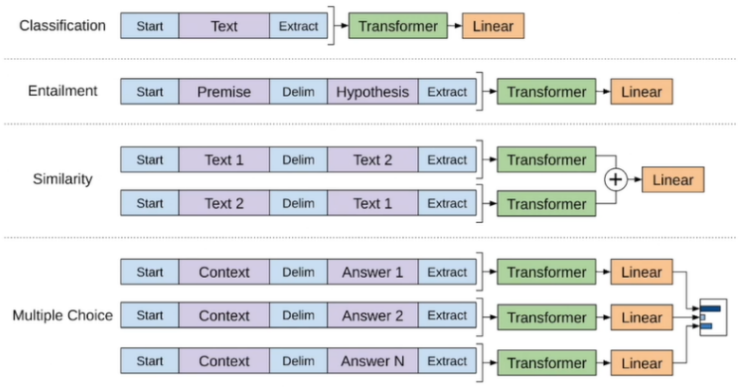


図 6: GPT-1 のファインチューニング

- zero-shot: 何のタスクかを指定した後、すぐに推論させる方法 [入力例: 英語からフランス語へ翻訳してください。cheese]
- one-shot: 一度だけラベル付きの例を教え、その後に推論させる方法 [入力例: 英語からフランス語へ翻訳してください。sea otter => loutre de mer, cheese =>]
- few-shot: 2 つ以上の例を教えた後、推論させる方法 [入力例: 英語からフランス語へ翻訳してください。sea otter => loutre de mer, plush girafe => girafe peluche, cheese =>]

■参考文献

1. 岡谷貴之/深層学習 改訂第 2 版 [機械学習プロフェッショナルシリーズ]/ 講談社サイエンティフィク/ 2022-01-17

6　実装演習キャプチャ

6.1　Seq2Seq